

OPTIONS NODECK,LIST,XREF,NOREL,OBJ(P)

THE LIST OF OPTIONS USED DURING THIS ASSEMBLY IS-- NODECK,LIST,XREF,NOREL,OBJ

0000	2	#FMSTD	START	0
	3		PRINT	ON,NODATA
	4	*	@SYS	EXP-N
	215+		PRINT	ON
	216	*	@FXD	EXP-N
	621+		PRINT	ON
	622	*	@CAN	EXP-N
	725+		PRINT	ON
	726	*	@CY0	EXP-N
	799+		PRINT	ON
	800	*	@HLT	EXP-N
	855+		PRINT	ON
	856	*	@WKA	EXP-N
	926+		PRINT	ON
	927	*	@HDW	EXP-N
	1112+		PRINT	ON
	1113	*	@SPF	EXP-N
	1576+		PRINT	ON
	1577	*	@ERM	EXP-N
	2199+		PRINT	ON
	2200	*	@VMD	EXP-N
	2321+		PRINT	ON
	2322	*	\$V\$E	EXP-N
	2745+		PRINT	ON
	2746	*	@B@E	EXP-N
	3646+		PRINT	ON
	3647	*	\$I\$E	EXP-N
	3801+		PRINT	ON
	3802	*	\$I@E	EXP-Y,PREC-S
	3804+		PRINT	ON

ERR	LOC	OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE	3
				3806	+	*****				
				3807	+	STANDARD PRECISION DATA ELEMENT EQUATES				*
				3808	+	*****				
				3809	+	*				
			0007	3810	+	I@PRCS EQU 7			FLOATING POINT PRECISION	
			0006	3811	+	I@APRS EQU 6			APPARENT FLT PT PRECISION	
				3812	+	*				
			0005	3813	+	I@LPFS EQU 5			PACKED FLT PT ELEMENT LENGTH	
			0008	3814	+	I@LUFS EQU I@PRCS+1			UNPACKED FLT PT ELEMENT LENGTH	
				3815	+	*				
			0003	3816	+	I@PMRS EQU I@LPFS-2			PACKED MANTISSA RH BYTE DISP	
			0004	3817	+	I@PEXS EQU I@LPFS-1			PACKED FLT PT EXPONENT DISP	
			0007	3818	+	I@UMRS EQU I@LUFS-1			UNPACKED MANTISSA RH BYTE DISP	
			0007	3819	+	I@SGNS EQU I@UMRS			UNPACKED FLT PT SIGN BYTE DISP	
				3820	+	*				
			0087	3821	+	I@SPSW EQU @UCB			STANDARD PREC SWITCH SETTING	
			0000	3822	+	I@ASTS EQU X'00'			ARITHMETIC ELEMENT STATUS INDR	
				3823	+	*				
			F500	3824	+	I@ICBS EQU X'F500'			INTERNAL CONSTANT BASE VADDR	
			F531	3825	+	I@IVBS EQU X'F531'			INTERNAL VARIABLE BASE VADDR	
				3826	+	*				
				3827	+	*****				

\$I@EQ - S/3 BASIC INTERPRETER SHORT/LONG PREC. EQUATES

ERR	LOC	OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE	4
				3829+	*****					
				3830+	* LONG PRECISION DATA ELEMENT EQUATES					*
				3831+	*****					
				3832+	*					
		000F	3833+	I@PRCL	EQU	15				FLOATING POINT PRECISION
		000B	3834+	I@APRL	EQU	11				APPARENT FLT PT PRECISION
				3835+	*					
		0009	3836+	I@LPFL	EQU	9				PACKED FLT PT ELEMENT LENGTH
		0010	3837+	I@LUFL	EQU	I@PRCL+1				UNPACKED FLT PT ELEMENT LENGTH
				3838+	*					
		0007	3839+	I@PMRL	EQU	I@LPFL-2				PACKED MANTISSA RH BYTE DISP
		0008	3840+	I@PEXL	EQU	I@LPFL-1				PACKED FLT PT EXPONENT DISP
		000F	3841+	I@UMRL	EQU	I@LUFL-1				UNPACKED MANTISSA RH BYTE DISP
		000F	3842+	I@SGNL	EQU	I@UMRL				UNPACKED FLT PT SIGN BYTE DISP
				3843+	*					
		0080	3844+	I@LPSW	EQU	@NOP				LONG PRECISION SWITCH SETTING
		0020	3845+	I@ASTL	EQU	X'20'				ARITHMETIC ELEMENT STATUS INDR
				3846+	*					
		F000	3847+	I@ICBL	EQU	X'F000'				INTERNAL CONSTANT BASE VADDR
		F049	3848+	I@IVBL	EQU	X'F049'				INTERNAL VARIABLE BASE VADDR
				3849+	*					
				3850+	*****					

\$I@EQ - S/3 BASIC INTERPRETER SHORT/LONG PREC. EQUATES

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE	5
			3852+	*****					
			3853+	* -CURRENT- PRECISION DATA ELEMENT EQUATES (STANDARD) *					
			3854+	*****					
			3855+	*					
	0007	3856+	I@PREC	EQU	I@PRCS	FLOATING POINT PRECISION			
	0006	3857+	I@APRC	EQU	I@APRS	APPARENT FLT PT PRECISION			
			3858+	*					
	0005	3859+	I@LPFV	EQU	I@LPFS	PACKED FLT PT ELEMENT LENGTH			
	0008	3860+	I@LUFV	EQU	I@LUFs	UNPACKED FLT PT ELEMENT LENGTH			
	0007	3861+	I@LSFV	EQU	I@LUFV-1	SPECIAL UNPACKED FLT PT LENGTH			
			3862+	*					
	0003	3863+	I@PMNR	EQU	I@PMRS	PACKED MANTISSA RH BYTE DISP			
	0004	3864+	I@PEXP	EQU	I@PEXS	PACKED FLT PT EXPONENT DISP			
	0007	3865+	I@UMNR	EQU	I@UMRS	UNPACKED MANTISSA RH BYTE DISP			
	0007	3866+	I@SIGN	EQU	I@SGNS	UNPACKED FLT PT SIGN BYTE DISP			
			3867+	*					
	0087	3868+	I@PRSW	EQU	I@SPSW	CURRENT PREC SWITCH SETTING			
	0000	3869+	I@ASTA	EQU	I@ASTS	ARITHMETIC ELEMENT STATUS INDR			
			3870+	*					
	F500	3871+	I@ICBA	EQU	I@ICBS	INTERNAL CONSTANT BASE VADDR			
	F531	3872+	I@IVBA	EQU	I@IVBS	INTERNAL VARIABLE BASE VADDR			
			3873+	*					
			3874+	*****					

[illegible]

ERR	LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE	7
				3891	*****				
				3892	* ARITHMETIC FUNCTION REFERENCE EQUATES				*
				3893	*****				
				3894	+				
	0000		3895	I@1SE1	EQU 0			1ST	BYTE OF STACK ELEMENT 1
	0007		3896	I@RSE1	EQU I@1SE1+I@LUFV-1			LAST	BYTE OF STACK ELEMENT 1
				3897	+				
	0008		3898	I@1SE2	EQU I@1SE1+I@LUFV			1ST	BITE OF STACK ELEMENT 2
	000F		3899	I@RSE2	EQU I@1SE2+I@LUFV-1			LAST	BYTE OF STACK ELEMENT 2
				3900	+				
	0010		3901	I@1SE3	EQU I@1SE2+I@LUFV			1ST	BYTE OF STACK ELEMENT 3
	0017		3902	I@RSE3	EQU I@1SE3+I@LUFV-1			LAST	BYTE OF STACK ELEMENT 3
				3903	+				
	0000		3904	I@DEXP	EQU I@UEXP			FLOATING	VALUE EXPONENT DISP
	0001		3905	I@MANL	EQU I@UMN1			LEFTMOST	BYTE OF MANTISSA
	0007		3906	I@MANR	EQU I@UMNR			RIGHTMOST	BYTE OF MANTISSA
				3907	+				
				3908	*****				

\$I@EQ - S/3 BASIC INTERPRETER SHORT/LONG PREC. EQUATES

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER	15,	MOD	00	31/05/21	PAGE	8
					3910	+	*****							
					3911	+	* PSEUDO INSTRUCTION COMPONENT EQUATES							*
					3912	+	*****							
					3913	+	*							
				0000	3914	+	I@XOPC EQU 0					DISP	FOR	INSTRUCTION OPCODE
				0002	3915	+	I@XLNO EQU I@XOPC+2					DISP	FOR	STATEMENT NO. OPERAND
				0002	3916	+	I@XVAD EQU I@XOPC+2					DISP	FOR	VIRTUAL ADDR OPERAND
				0001	3917	+	I@XCNT EQU I@XOPC+1					DISP	FOR	INST EXECUTION COUNT
				0001	3918	+	I@XCOD EQU I@XOPC+1					DISP	FOR	INST EXECUTION CODE
				0003	3919	+	I@XBRC EQU I@XOPC+3					DISP	FOR	BRANCH CONDITION CODE
					3920	+	*							
					3921	+	*****							
					3922	+	* COMPARE CONDITION STATUS CODE EQUATES							*
					3923	+	*****							
					3924	+	*							
				0002	3925	+	I@CMLO EQU X'02'					COMPARE	CONDITION	- LOW
				0004	3926	+	I@CMEQ EQU X'04'					COMPARE	CONDITION	- EQUAL
				0008	3927	+	I@CMHI EQU X'08'					COMPARE	CONDITION	- HIGH
					3928	+	*							
					3929	+	*****							

\$I@EQ - S/3 BASIC INTERPRETER SHORT/LONG PREC. EQUATES

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER	15,	MOD	00	31/05/21	PAGE	9
					3931+	*****								
					3932+	* RUN-TIME STACK ELEMENT EQUATES								*
					3933+	*****								
					3934+	*								
		0001		3935+	I@SVAD	EQU	1						DISP	FOR STACKED VIRTUAL ADDR
		0000		3936+	I@SCOD	EQU	0						DISP	FOR STACKED EXECUTION CODE
		0001		3937+	I@SIDX	EQU	1						DISP	FOR STACKED INDEX VALUE
					3938+	*								
					3939+	*****								
					3940+	* VIRTUAL MEMORY CORE PAGE REGION EQUATES								*
					3941+	*****								
					3942+	*								
		0000		3943+	I@PRTE	EQU	0						DISP	FOR PAGE REF TABLE ENTRY
					3944+	*								
		0003		3945+	I@NXPT	EQU	3						NO.	OF EXPANDABLE PAGING TABLES
		0020		3946+	I@NXPG	EQU	32						NO.	OF EXPANSION PAGES (MAX)
		0060		3947+	I@LXPT	EQU	I@NXPT*I@NXPG						LENGTH	OF MAX TABLE EXPANSION
					3948+	*								
		000A		3949+	I@NCPG	EQU	10						NO.	OF 8K SYSTEM PAGES (MAX)
					3950+	*								
					3951+	*****								
					3952+	* MISCELLANEOUS EXECUTION EQUATES								*
					3953+	*****								
					3954+	*								
		0003		3955+	I@LPPZ	EQU	3						LENGTH	OR PACKED PRINT ZONE
		0012		3956+	I@LFPZ	EQU	18						LENGTH	OF FULL PRINT ZONE
					3957+	*								
		0000		3958+	I@NERR	EQU	0						NULL	EXECUTION ERROR CODE
					3959+	*								
					3960+	*****								
					3961+	*								
					3962+	* END OF EXECUTION SYSTEM EQUATES CODING								
					3963+	*								
0200					3964+	PRINT	ON							
					3965	ORG	\$\$\$FMS							

FSKLOG - LOGARITHM BASE E, 10, 2 - STANDARD PREC

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 10
		3967		*****			
		3968	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
		3969	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
		3970	*				*
		3971		*****			*
		3972	*	STATUS			*
		3973	*	VERSION 1 MODIFICATION 0			*
		3974	*				*
		3975	*	FUNCTION			*
		3976	*	* FKSLOG TAKES THE NATURAL LOGARITHM OF THE INPUT ARGUMENT, IN			*
		3977	*	STANDARD PRECISION.			*
		3978	*	* FKSLGT TAKES THE LOGARITHM BASE 10 OR THE INPUT ARGUMENT, IN			*
		3979	*	STANDARD PRECISION.			*
		3980	*	* FKSLTW TAKES THE LOGARITHM BASE 2 OR THE INPUT ARGUMENT, IN			*
		3981	*	STANDARD PRECISION.			*
		3982	*	* ALL THREE ENTRIES USE THE SAME TECHNIQUES, EXCEPT THAT BASES			*
		3983	*	10 AND 2 MULTIPLY THE LOG BASE E BY LOG(E) BASE 10 AND BASE 2			*
		3984	*	RESPECTIVELY.			*
		3985	*	* THE LOGARITHM OF THE INPUTTED VALUE IS CALCULATED BY MEANS OF			*
		3986	*	A RECURSIVE ALGORITHM ON THE SECOND PAGE OF THE MODULE.			*
		3987	*				*
		3988	*	ENTRY POINTS			*
		3989	*	* FKSLOG IS THE ENTRY TO COMPUTE THE NATURAL LOGARITHM OF THE			*
		3990	*	ARGUMENT. THE FORMAT OF THE CALLING SEQUENCE IS AS FOLLOW			*
		3991	*	B ISCALL			*
		3992	*	DC AL2(V\$FLOG)			*
		3993	*	* FKSLGT IS THE ENTRY TO COMPUTE THE LOGARITHM BASE 10. THE			*
		3994	*	FORMAT OF THE CALLING SEQUENCE IS AS FOLLOWS:			*
		3995	*	B ISCALL			*
		3996	*	DC AL2(V\$FLGT)			*
		3997	*	* FKSLTW IS THE ENTRY TO COMPUTE THE LOGARITHM BASE 2. THE			*
		3998	*	FORMAT OF THE CALLING SEQUENCE IS AS FOLLOWS:			*
		3999	*	B ISCALL			*
		4000	*	DC AL2(V\$FLTW)			*
		4001	*				*
		4002	*	INPUT			*
		4003	*	* THE INPUT IS AN UNPACKED FLOATING POINT NUMBER IN THE FIRST			*
		4004	*	INTERPRETER STACK ELEMENT. THE ADDRESS OR ITS FIRST BYTE IS			*
		4005	*	IN ISSTAK.			*
		4006	*	* INDEY REGISTER 1 (@BR) IS SET TO THE BEGINNING OF THE PAGE AT			*
		4007	*	ENTRY, FOR USE AS A BASE REGISTER.			*
		4008	*	* INDEX RESISTER 2 (@XR) IS MODIFIED TO REFERENCE THE INTERPRETER			*
		4009	*	STACK.			*
		4010	*				*
		4011	*	OUTPUT			*
		4012	*	* THE RESULT IS LEFT IN THE FIRST INTERPRETER STACK ELEMENT IN			*
		4013	*	UNPACKED FLOATING POINT FORMAT.			*
		4014	*	* IN THE EVENT OF AN ERROR, THE APPROPRIATE CODE IS PLACED IN THE			*
		4015	*	INTERPRETER ONE-BYTE ERROR LOCATION, I\$ERRC			*
		4016	*				*
		4017	*	EXTERNAL REFERENCES			*
		4018	*	INTERPRETER ONE-BYTE ERROR LOCATION, I\$ERRC			*
		4019	*				*
		4020	*	EXTERNAL REFERENCES			*
		4021	*	INTERPRETER STACK - FIRST TWO ELEMENTS PLUS 2 BYTES OF THIRD			*
		4022	*	I\$STAK - LOCATION OF THE ADDRESS OF THE INTERPRETER STACK			*

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 11
		4023	*	I\$FWRK - AN INTERPRETER WORK AREA OF WHICH 17 BYTES ARE USED	*
		4024	*	I\$ERRC - INTERPRETER ERROR CODE AREA	*
		4025	*	FZIMPY - MULTIPLICATION SUBROUTINE, INTERPRETER RESIDENT	*
		4026	*	CENYZD - CONVERT NORMALIZED EXPONENT TO ZONED DECIMAL	*
		4027	*	CDBNZD - CONVERT BINARY NUMBER TO ZONED DECIMAL	*
		4028	*	CCZDFP - CONVERT ZONED DECIMAL TO FLOATING POINT	*
		4029	*		*
		4030	*	*EXITS, NORMAL	*
		4031	*	* * EXIT IS BY BRANCHING TO I\$RTRN. THE ENTRY TO THE PAGING RETURN	*
		4032	*	ROUTINE.	*
		4033	*	* * THE RESULT IS PLACED IN THE FIRST STACK ELEMENT.	*
		4034	*		*
		4035	*	*EXITS, ERROR	*
		4036	*	* * AN ERROR CODE IS PLACED IN THE INTERPRETER ERROR AREA, I\$ERRC.	*
		4037	*	* * IF THERE IS AN ERROR, THE INPUT ARGUMENT IS NOT ALTERED, AND	*
		4038	*	IS RETURNED IN THE FIRST STACK ELEMENT.	*
		4039	*	* * EXIT IS BY BRANCHING TO I\$RTRN, THE ENTRY TO THE PAGING RETURN	*
		4040	*	ROUTINE.	*
		4041	*		*
		4042	*	*TABLES/WORK AREA	*
		4043	*	* * THE CONSTANTS RESIDE AT THE END OR THE EXECUTABLE CODE ON BOTH	*
		4044	*	PAGES.	*
		4045	*	* * THE CONSTANTS LABELED FKSCON AT THE END OF PAGE TWO ARE ALSO	*
		4046	*	REFERENCED BY FGSEXP, BY THE EQUATE FKSCNT.	*
		4047	*		*
		4048	*	*ATTRIBUTES	*
		4049	*	* NATURALLY RELOCATABLE, REUSABLE.	*
		4050	*		*
		4051	*	*CHARACTER CODE DEPENDENCY	*
		4052	*	* THE OPERATION OF THIS MODULE DEPENDS UPON A ZONED DECIMAL DIGIT	*
		4053	*	* BEING REPRESENTED WITH THE ZONE (FIRST FOUR BITS) BEING AN "F"	*
		4054	*	* FOR POSITIVE, AND A "D" TOR NEGATIVE. THE DECIMAL NUMBERS MUST	*
		4055	*	* BE CODED SO THAT THE LOW ORDER FOUR BITS, WHEN CONSIDERED AS A	*
		4056	*	* BINARY INTEGER, IDENTITY THE VALUE OR THE DIGIT.	*
		4057	*	* THESE PROPERTIES ARE USED FOLLOWING FKS020.	*
		4058	*		*
		4059	*	*NOTES	*
		4060	*	* ERROR PROCEDURES	*
		4061	*	* THE ERROR CODE IS SET. AND CONTROL RETURNED TO THE PAGING	*
		4062	*	ROUTINE.	*
		4063	*		*
		4064	*	* REGISTER USAGE	*
		4065	*	* INDEX REGISTER 1 (@BR) IS SET TO THE FIRST BYTE OF THE PAGE,	*
		4066	*	* AND IS USED AS THE BASE REGISTER.	*
		4067	*	* INDEX REGISTER 2 (@XR) IS LOADED TO CONTAIN THE ADDRESS OR THE	*
		4068	*	* FIRST BYTE OF THE INTERPRETER STACK, AND USED TO INDEX THE	*
		4069	*	* STACK.	*
		4070	*		*
		4071	*	* SAVED/RESTORED AREAS	*
		4072	*	* NONE.	*
		4073	*		*
		4074	*	* MODIFICATION CONSIDERATIONS	*
		4075	*	* THE EQUATED LABEL FKSCNT MUST BE SET TO REFERENCE THE STORED	*
		4076	*	* CONSTANTS, AS IT IS USED BY FGSEXP.	*
		4077	*		*
		4078	*	* REQUIRED MODULES	*

FSKLOG - LOGARITHM BASE E, 10, 2 - STANDARD PREC								
ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21 PAGE 12	
			4079 *		@SYSEQ - COMMON SYSTEM EQUATES		*	
			4080 *		@ERMEQ - ERROR MESSAGE EQUATES		*	
			4081 *		\$V\$EQU - FIXED VIRTUAL ADDRESSES		*	
			4082 *		\$B@EQU - COMPILER SYSTEM EQUATES		*	
			4083 *		\$I\$EQU - FIXED EQUATES		*	
			4084 *		\$I@SEQ - STANDARD PRECISION EXECUTION EQUATES		*	
			4085 *				*	
			4086 *	OTHER			*	
			4087 *	NONE			*	
			4088	*****				

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 13
		4090		*****	
		4091	*	FKSLOG - S/3.7 LOGARITHM BASE E ROUTINE - STANDARD PRECISION	*
		4092		*****	
		4093	*	FKSLFT - S/3.7 LOGARITHM BASE 10 ROUTINE - STANDARD PRECISION	*
		4094		*****	
		4095	*	FKSLTW - S/3.7 LOGARITHM BASE 2 ROUTINE - STANDARD PRECISION	*
		4096		*****	
		4097	*		
0200		4098		ORG *,B@LVP,0	
		0200 4099		USING *,@BR	BASE REG POINTS HERE AT ENTRY
		4101		*****	
		4102	*	EXECUTION ENTRY TO LOG BASE 10 ROUTINE FOLLOWS	
		4103	*		
		4104	*	COMPUTE LOG BASE E, THEN MULTIPLY RESULT BY LOG(E) BASE 10 TO GET	
		4105	*	LOG BASE 10	
		4106	*		
0200 5C 07 A3 AB		4107	FKSLGT MVC	FKSCNV(I@LUFV,@BR),FKSTEN(@BR) USE LOG(E) BASE 10 FOR COE	
0204 7C E3 30		4108	MVI	FKS025+@Q(@BR),@E778 SET ERROR MESSAGE FOR LOG 10	
0207 D0 87 12		4109	B	FKS010(@BR) USE LOG BASE E LOGIC	
		4111		*****	
		4112	*	EXECUTION ENTRY TO LOG BASE 2 ROUTINE FOLLOWS	
		4113	*		
		4114	*	COMPUTE LOG BASE E, THEN MULTIPLY RESULT BY LOG(E) BASE 2 TO GET	
		4115	*	LOG BASE 2	
		4116	*		
020B		4117	ORG	FKSLGT+11	
020B 5C 07 A3 B3		4118	FKSLTW MVC	FKSCNV(I@LUFV,@BR),FKSTWO(@BR) USE LOG(E) BASE 2 FOR COE	
020F 7C E4 30		4119	MVI	FKS025+@Q(@BR),@E779 SET ERROR MESSAGE FOR LOG 2	
0212 7C 80 8E		4120	FKS010 MVI	FKS600+@Q(@BR),@NOP SET BC INSTRUCTION TO NOP	
0215 D0 87 1F		4121	B	FKS020(@BR) USE LOG BASE E LOGIC	
		4123		*****	
		4124	*	EXECUTION ENTRY TO LOG BASE E ROUTINE FOLLOWS	
		4125	*		
0219		4126	ORG	FKSLGT+25	
0219 7C 87 8E		4127	FKSLOG MVI	FKS600+@Q(@BR),@UCB SET BC INSTR TO UNCOND BRANCH	
021C 7C E2 30		4128	MVI	FKS025+@Q(@BR),@E777 SET ERROR MESSAGE FOR LOG E	
021F 35 02 0D4E		4129	FKS020 L	I\$STAK,@XR LOAD STACK POINTER	
		4130	*		
		4131	*	CHECK FOR ZERO OR NEGATIVE ARGUMENT	
		4132	*		
0223 BD F0 01		4133	CLI	I@1SE1+I@MANL(@XR),@DZERO IS ARGUMENT ZERO ?	
0226 D0 81 2F		4134	BE	FKS025(@BR) YES, BRANCH TO SET ERROR FLAG	
0229 B8 F0 07		4135	TBN	I@RSE1(@XR),B@ZPOS IS THE ARGUMENT POSITIVE ?	
022C D0 10 36		4136	BT	FKS030(@BR) YES, BRANCH TO CONTINUE	
022F 3C 00 0CBC		4137	FKS025 MVI	I\$ERRC,*-* NEGATIVE OR ZERO AND ERROR ?	
0233 D0 87 98		4138	B	FKS700(@BR) RETURN TO CALLING PROGRAM	
		4140	*	PUT MANTISSA IN WORK AREA OF INTERPRETER SO THAT THE INDEX REGISTER	
		4141	*	IS FREE FOR USE AS A POINTER TO THE STORED CONSTANTS	
		4142	*		
0236 2C 06 0617 07		4143	FKS030 MVC	I\$FWRK+I@LUFV+I@LUFV(I@PREC),I@RSE1(@XR) SAVE MANTISSA	
		4144	*		
		4145	*	COMPUTE LOG(MANTISSA) IN THE SECOND PAGE. THE RESULT WILL BE LEFT	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 14
			4146	* IN I\$FWRK IN ZONED DECIMAL FORMAT (NOT FLOATING POINT) WITH INTEGER	
			4147	* AND DECIMAL PARTS AS FOLLOWS:	
			4148	* I@PREC BYTE FRACTIONAL PART WITH RIGHTMOST BYTE AT I\$FWRK+I@PREC	
			4149	* 1 BYTE INTEGER PART AT I\$FWRK	
			4150	* THE RESULT WILL BE A NEGATIVE NUMBER OR ZERO. MAXIMUM VALUE OF	
			4151	* MAGNITUDE WILL BE 2.30258510	
			4152	*	
023B C0 87 12B1			4153	B I\$CALL COMPUTE LOG(MANTISSA)	
023F 0300		0240	4154	DC AL(@VADDR)(V\$FLGT+256)	
			4155	*	
			4156	* NO ERROR CONDITIONS CAN ARISE IN THE SECOND PAGE, SO THE CONTENTS	
			4157	* OF I\$ERRC ARE NOT CHECKED	
			4158	*	
			4159	* CONVRRT THE EXPONENT OF THE ARGUMENT TO A ZONED DECIMAL NUMBER	
			4160	*	
0241 C0 87 12B1			4161	B I\$CALL CONVERT NORMALIZED EXPONENT TO	
0245 0470		0246	4162	DC AL(@VADDR)(V\$CNXZ) * ZONED DECIMAL	
			4163	*	
			4164	* CONVERT ROUTINE MAKES NO CHECKS FOR ERRORS, SO THE CONTENTS OF I\$ERRC	
			4165	* ARE NOT CHECKED ON RETURN	
			4166	*	
			4167	* EXPONENT IS NOW A SIGNED DECIMAL NUMBER BETWEEN -99 AND 99.	
			4168	* MULTIPLY IT BY LOG(10) BASE E BY USING THE SECOND AND FIRST DIGITS	
			4169	* RESPECTIVELY AS COUNTERS TO ADD LOG(10) AND 10*LOG(10) REPEATEDLY,	
			4170	* AND THEN SET THE SIGN OF THE RESULT TO THE SIGN OF THE EXPONENT.	
			4171	*	
0247 BC F0 11			4172	MVI I@1SE3+I@MANL(,@XR),@DZERO GET DECIMAL ZERO	
024A AC 08 10 11			4173	MVC I@1SE3(I@LUFV+FKSONE,@XR),I@1SE3+FKSONE(,@XR) ACCUMUL=0'S	
024E B9 0F 07			4174	FKS205 TBF I@RSE1(,@XR),X'0F' IS SECOND EXP DIGIT ZERO ?	
0251 D0 10 5F			4175	BT FKS210(,@BR) YES, EXAMINE FIRST EXP DIGIT	
0254 96 17 11 BB			4176	AZ I@1SE3+I@MANL(I@LUFV+FKSONE,@XR),FKSTNE(I@LUFV,@BR) +LOG1	
0258 9F 00 07 BC			4177	SLC I@RSE1(1,@XR),FKSDCR(,@BR) DECREMENT EXP DIGIT	
025C D0 87 4E			4178	B FKS205(,@BR) AND REPEAT TEST	
025F B9 0F 06			4179	FKS210 TBF I@RSE1-FKSONE(,@XR),X'0F' IS FIRST EXP DIGIT ZERO ?	
0262 D0 10 70			4180	BT FKS220(,@BR) YES, WE ARE DONE	
0265 96 17 10 BB			4181	AZ I@1SE3(I@LUFV+FKSONE,@XR),FKSTNE(I@LUFV,@BR) +10*LOG(10)	
			4182	*	
			4183	* ACTUALLY LOG(10) WAS ADDED SHIFTED 1 BYTE LEFT, THIS CAUSES NO	
			4184	* TROUBLE SINCE LOG(10) IS POSITIVE AS ARE THE CONTENTS OF ACCUMULATOR	
			4185	*	
0269 9F 00 06 BC			4186	SLC I@RSE1-FKSONE(1,@XR),FKSDCR(,@BR) DECREMENT EXP DIGIT	
026D D0 87 5F			4187	B FKS210(,@BR) AND REPEAT TEST	
0270 A8 00 11 07			4188	FKS220 MZZ I@1SE3+FKSONE(,@XR),I@RSE1(,@XR) RESULT HAS SAME SIGN EXP	
			4189	*	
			4190	* COMPUTE LOG BASE E OF (MANTISSA * 10**EXP) BY ADDING LOG(MANTISSA)	
			4191	* TO EXP*LOG(10). BOTH ARE ZONED DECIMAL NUMBERS WITH FRACTIONAL PARTS	
			4192	* OF LENGTH I@PREC BYTES	
			4193	*	
0274 86 27 11 060E			4194	AZ I@1SE3+FKSONE(I@LUFV+FKSADD,@XR),I\$FWRK+I@PREC(I@LUFV)	
			4195	* ADD LOG(MANTISSA)	
0279 2C 08 060F 10			4196	MVC I\$FWRK+I@LUFV(I@LUFV+FKSONE),I@1SE3(,@XR) SHIFT TRUNCATED	
027E 8C 08 0D 060F			4197	MVC I@RSE2-FKSSFT(I@LUFV+FKSONE,@XR),I\$FWRK+I@LUFV RESULT->STC	
0283 A8 00 0F 11			4198	MZZ I@RSE2(,@XR),I@1SE3+FKSONE(,@XR) INSERT CORRECT SIGN	
			4199	*	
			4200	* BYTES I@RSE1-2 TO I@RSE2-2 INCLUSIVE OF THE STACK, CONTAIN THE RESULT	
			4201	* BYTES I@RSE1 TO I@SE1-3 CONTAIN DECIMAL ZEROES BECAUSE THE EXPONENT	

#FMSTD - PROLOGUE - LOG BASE E, 10, 2 - STANDARD PREC

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 15
			4202	* TO ZONED DEC CONVERSION ROUTINE USED EARLIER LEFT THEM THERE. THE	
			4203	* DECIMAL POINT OF THE RESULT IS BETWEEN I@RSE1 AND I@RSE3 SO THAT THE	
			4204	* RESULT IS IN PROPER FORMAT TO BE CONVERTED INTO FLOATING POINT FORMAT	
			4205	*	
0287	C0 87 12B1		4206	B I\$CALL	CONVERT ZONED DEC TO FLT POINT
028B	04AD	028C	4207	DC AL(@VADDR)(V\$CZFP)	
			4208	*	
			4209	* THE CONVERT ROUTINE MAKES NO ERROR CHECKS, SO THE CONTENTS OF I\$ERRC	
			4210	* ARE NOT CHECKED ON RETURN.	
			4211	*	
028D	D0 00 98		4212	FKS600 BC FKS700(,@BR),*-*	RETURN IF WE WANT BASE E LOG
0290	9C 07 0F A3		4213	MVC I@RSE2(I@LUFV,@XR),FKSCNV(,@BR)	GET CORRECT MULTIPLIER
0294	C0 87 082A		4214	B I\$FMPY	CONVERT TO PROPER BASE
			4215	*	
			4216	* OVERFLOW OR ODERFLOW IMPOSSIBLE DUE TO NATURE OF LOS FUNCTION SO	
			4217	* I\$ERRC IS NOT CHECKED ON RETURN	
			4218	*	
0298	C0 87 12D3		4219	FKS700 B I\$RTRN	RETURN TO CALLING PROGRAM
			4220	*	
			4221	* NON-STORAGE CONSTANTS FOR THE LOG ROUTINES FOLLOW	
			4222	*	
		0001	4223	FKSONE EQU 1	LENGTH OF 1
		0002	4224	FKSADD EQU 2	INCREMENT FOR ADDITION
		0002	4225	FKSSFT EQU 2	LENGTH TO SHIFT RESULT TO STACK
			4226	*	
			4227	* WORK AREA	
			4228	*	
029C		02A3	4229	FKSCNV DS CL(I@LUFV)	BASE CONVERSION MULTIPLIER REG
			4230	*	
			4231	* CONSTANTS FOR SUBROUTINE	
			4232	*	
02A4	80	02A4	4233	DC AL1(B@NXZR)	LOG(E) BASE 10 IN FLOATING PT
02A5	F4F3F4F2F9F4F5	02AB	4234	FKSTEN DC DL(I@PREC)'4342945'	* FOR CONVERSION TO BASE E
02AC	81	02AC	4235	DC AL1(B@NXZR+1)	LOG(E) BASE 2 IN FLOATING POINT
02AD	F1F4F4F2F6F9F5	02B3	4236	FKSTWO DC DL(I@PREC)'1442695'	* FOR CONVERSION TO BASE E
02B4	F2F3F0F2F5F8F5F1	02BB	4237	FKSTNE DC DL(I@LUFV)'23025851'	LOS(10) BASE E
02BC	01	02BC	4238	FKSDCR DC AL1(@B1)	BINARY ONE FOR DECREMENTS

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 16
			4240	*****	
			4241	* SECOND PAGE OF FKSLOG	
			4242	*	
0300			4243	ORG *,B@LVP,0	
		0300	4244	USING *,@BR	BASE REG POINTS HERE AT ENTRY
		0300	4245	USING *,@XR	
			4246	*	
			4247	* EXECUTION ENTRY TO LOG BASE E ROUTINE FOR MANTISSA ARGUMENT	
			4248	*	
			4249	* CLEAR WORK AREAS AND INITIALIZE	
			4250	*	
0300	3C F0 0610		4251	FKS090 MVI I\$FWRK+I@LUFV+FKSONE,@DZERO GET DECIMAL ZERO	
0304	0C 08 060F 0610		4252	MVC I\$FWRK+I@LUFV(I@LUFV+FKSONE),I\$FWRK+I@LUFV+1 CLEAR TO 0'S	
030A	4C 06 7C 060F		4253	MVC FKSARG(I@PREC,@BR),I\$FWRK+I@LUFV CLEAR TO ZEROES	
030F	7C F1 75		4254	MVI FKSARG-I@PREC(,@BR),B@DEC1 INSERT LEADING 1	
			4255	*	
			4256	* THE ALGORITHM USES (1-MANTISSA) AS THE INITIAL ARGUMENT	
			4257	*	
0312	47 16 7C 0617		4258	SZ FKSARG(I@LUFV,@BR),I\$FWRK+I@LUFV+I@LUFV(I@PREC) 1-MANTISSA	
			4259	*	
			4260	* THE INSTRUCTION AT FKS120 MODIFIES THE SUBTRAHEND USED TO DECREMENT	
			4261	* THE ARGUMENT (INITIALLY 1-MANTISSA) BY MULTIPLYING IT BY (1+10**(-N))	
			4262	* WHERE N IS THE ITERATION NUMBER. INITIALLY THE SUBTRAHEND IS EQUAL	
			4263	* TO THE INPUT MANTISSA. IN ORDER TO DO THIS, THE INSTRUCTION AT	
			4264	* FKS120 IS MODIFIED AFTER EACH ITERATION. THE FOLLOWING INSTRUCTION	
			4265	* REINITIALIZES THE INSTRUCTION AT FKS120.	
			4266	*	
0317	5C 04 37 74		4267	MVC FKS120+@OP2(FKSINT,@BR),FKSINS+@OP2(,@BR) INITIAL. INSTR	
031B	7C 00 84		4268	MVI FKSITN(,@BR),@ZERO INITIALIZE ITERATION NTMBER	
			4269	*	
			4270	* THE INDEX REG IS SET TO THE BASE ADDRESS. IT IS USED AS A POINTER	
			4271	* TO THE STORED CONSTANTS USED BY THE ALGORITHM.	
			4272	*	
031E	74 01 24		4273	ST FKS095+@OP1(,@BR),@BR STORE CONTENTS OF BASE REG	
0321	C2 02 0000		4274	FKS095 LA *-*,@XR * INTO INDEY REG	
			4275	*	
			4276	* THE SUBTRAHEND IN THE FOLLOWING INSTRUCTION IS MODIFIED BY THE	
			4277	* INSTRUCTION AT FKS120	
			4278	*	
0325	47 07 7C 0617		4279	FKS100 SZ FKSARG(I@LUFV,@BR),I\$FWRK+I@LUFV+I@LUFV(I@LUFV) DECR, ARG	
032A	D0 82 3B		4280	BM FKS150(,@BR) DONE WITH THIS ITERATION	
			4281	*	
			4282	* THE FOLLOWING INSTRUCTION ACCUMULATES THE RESULT BY SUBTRACTING THE	
			4283	* POSITIVE STORED CONSTANT FOR THIS ITERATION. SINCE THE INPUT ARG IS	
			4284	* LESS THAN 1 THE ACCUMULATED RESULT IS ALWAYS NEGATIVE. THE CURRENT	
			4285	* VALUE OF THE STORED CONSTANT IS LOG(1+10**(-N)) BASE E, WHERE N IS	
			4286	* THE ITERATION NUMBER.	
			4287	*	
032D	27 17 060F 93		4288	SZ I\$FWRK+I@LUFV(I@LUFV+FKSONE),FKSCON(I@LUFV,@XR) ACC RESUL	
			4289	*	
			4290	* THE FOLLOWING INSTRUCTION MODIFIES THE SUBTRAHEND AT	
			4291	* I\$FWRK+I@LUFV+I@LUFV BY MULTIPLYING IT BY I+10**(-N).	
			4292	* SYMBOLICALLY, THIS INSTRUCTION IS	
			4293	* AZ I\$FWK+I@LUFV+I@LUFV(I@LUFV),I\$FWRK+I@LUFV+I@LUFV-N(I@LUFV-N)	
			4294	* WHERE N IS THE ITERATION NUMBER.	
			4295	*	

#FMSTD - PROLOGUE - LOG BASE E, 10, 2 - STANDARD PREC

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 17
0332 06 07 0617 0000		4296	FKS120 AZ	I\$FWRK+I@LUFV+I@LUFV(I@LUFV),*-(I@LUFV) MODIFY SUBTRAHEN	
0338 D0 87 25		4297	B	FKS100(,@BR) REPEAT INNER LOOP	
		4298	*		
		4299	*	END OF AN ITERATION	
		4300	*		
033B 5E 00 84 8A		4301	FKS150 ALC	FKSITN(1,@BR),FKSINC(,@BR) INCREMENT ITERATION NUMBER	
		4302	*		
		4303	*	I@LUFV ITERATIONS ARE REQUIRED TO GET I@PREC SIGNIFICANT DIGITS	
		4304	*		
033F 7D 07 84		4305	CLI	FKSITN(,@BR),I@PREC IS ITERATION FINISHED ?	
0342 D0 82 58		4306	BL	FKS175(,@BR) BRANCH IF NOT DONE YET	
		4307	*		
		4308	*	FINAL ITERATION HAS CONCLUDED	
		4309	*		
0345 17 80 060F 8B		4310	SZ	I\$FWRK+I@LUFV(I@LUFV+FKSONE),FKSRND(1,@BR) ROUND RESULT	
034A 08 00 060E 060F		4311	MZZ	I\$FWRK+I@PREC,I\$FWRK+I@LUFV MOVE SIGN ONE BYTE LEFT	
0350 35 02 0D4E		4312	L	I\$STAK,@XR RESTORE INDEX REGISTER	
		4313	*		
		4314	*	RETURN TO CALLING PROGRAM WITH I@CUFL BYTE ZONED DEC RESULT IN	
		4315	*	I\$WRK+I@PREC. DECIMAL POINT FOLLOWS LEADING BYTE OF RESULT.	
		4316	*		
0354 C0 87 12D3		4317	B	I\$RTRN	
		4318	*		
		4319	*	PREPARE FOR NEXT ITERATION	
		4320	*		
0358 46 07 7C 0617		4321	FKS175 AZ	FKSARG(I@LUFV,@BR),I\$FWRK+I@LUFV+I@LUFV(I@LUFV) RESTRE AR	
		4322	*	THE FOLLOWING INSTRUCTION MODIFIES INSTRUCTION AT FKS120 BY SIMULTA-	
		4323	*	NEOUSLY ADDING 1 TO LENGTH CODE OF FIRST OPERAND. SUBTRACTING 1 FROM	
		4324	*	LENGTH CODE OF SECOND OPERAND, AND SUBTRACTING 1 FROM ADDRESS OF	
		4325	*	SECOND OPERAND.	
		4326	*		
035D 5F 04 37 89		4327	SLC	FKS120+@OP2(FKSMOD,@BR),FKSMDY(,@BR) MODIFY INSTRUCTION	
0361 E2 02 08		4328	LA	I@LUFV(,@XR),@XR INCREMENT @VR FOR NEXT CONSTANT	
		4329	*		
		4330	*	SHIFT ARGUMENT 1 BYTE LEFT (MULTIPLY BY 10)	
		4331	*		
0364 5C 06 83 7C		4332	MVC	FKSSHT(I@PREC,@BR),FKSARG(,@BR) MOVE INTO SHIFT REG	
0368 5C 06 7B 83		4333	MVC	FKSARG-FKSONE(I@PREC,@BR),FKSSHT(,@BR) SHIFT LEFT 1 BYTE	
036C D0 87 25		4334	B	FKS100(,@BR) DO NEXT ITERATION	
		4335	*		
		4336	*	NON-STORAGE CONSTANTS FOR FFSLGM FOLLOW	
		4337	*		
		0005 4338	FKSINT EQU 5	LENGTH TO INIT INSTRUCTION.	
		0005 4339	FKSMOD EQU 5	LENGTH TO MODIFY INSTRUCTION	
		4340	*		
		4341	*	INITIAL INSTRUCTION FOR ALGORITHM STORED AS CONSTANT	
		4342	*		
036F 06 07 0617 0617		4343	FKSINS AZ	I\$FWRK+I@LUFV+I@LUFV(I@LUFV),I\$FWRK+I@LUFV+I@LUFV(I@LUFV)	
		4344	*		
		4345	*	WORK AREA FOR FFSLGM FOLLOWS	
		4346	*		
0375		037C 4347	FKSARG DS	CL(I@LUFV) ARGUMENT REGISTER	
037D		0383 4348	FKSSHT DS	CL(I@PREC) SHIFT REG	
0384		0384 4349	FKSITN DS	CL1 ITERATION NUMBER	
		4350	*		
		4351	*	CONSTANTS	

ERR	LOC	OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE	18
				4352	*					
	0385	F100000001	0389	4353	FKSMDY DC	XL(FKSMOD)'F100000001'			INSTRUCTION MODIFIER	
	038A	01	038A	4354	FKSINC DC	AL1(@B1)			BINARY	
	038B	F5	038B	4355	FKSRND DC	DL1'5'			FOR ROUNDING RESULT	
				4356	*					
				4357	* STORED CONSTANTS FOR USE OF LOG AND EXP SUBROUTINES: LOG(1+10**(-Q))					
				4358	*					
	038C	F6F9F3F1F4F7F1F8	0393	4359	FKSCON DC	DL(I@LUFV)'69314718'			Q=0	
	0394	F0F9F5F3F1F0F1F8	039B	4360		DC DL(I@LUFV)'09531018'			Q=1	
	039C	F0F0F9F9F5F0F3F3	03A3	4361		DC DL(I@LUFV)'00995033'			Q=2	
	03A4	F0F0F0F9F9F9F5F0	03AB	4362		DC DL(I@LUFV)'00099950'			Q=3	
	03AC	F0F0F0F1F0F0F0F0	03B3	4363		DC DL(I@LUFV)'00010000'			Q=4	
	03B4	F0F0F0F0F1F0F0F0	03BB	4364		DC DL(I@LUFV)'00001000'			Q=5	
	03BC	F0F0F0F0F0F1F0F0	03C3	4365		DC DL(I@LUFV)'00000100'			Q=6	
	03C4	F0F0F0F0F0F0F1F0	03CB	4366		DC DL(I@LUFV)'00000010'			Q=7	
	03CC	F0F0F0F0F0F0F0F1	03D3	4367		DC DL(I@LUFV)'00000001'			Q=8	
				4368	*					
			0093	4369	FKSCNT EQU	FKSCON-FKS090			DISP. TO STORED CONSTANTS	
				4370	*					
				4371	* END OF FKSLOG CODING					
				4372	*					

[illegible]

CENXZD - CONVERT A NORM EXPONENT TO A DEC NUM.

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 20
4379				*****			
4380	*			5703-XM1 COPYRIGHT IBM CORP. 1970			*
4381	*			REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
4382	*						*
4383				*****			*
4384	*			*STATUS			*
4385	*			VERSION 1 MODIFICATION 0			*
4386	*						*
4387	*			*FUNCTION			*
4388	*			CENXZD IS A VIRTUAL MEMORY SUBROUTINE TO CONVERT A NORMALIZED			*
4389	*			EXPONENT TO A DECIMAL NUMBER.			*
4390	*			THE EXPONENT IS FIRST CONVERTED TO A BINARY NUMBER AND THEN			*
4391	*			CDBNZD IS CALLED TO CONVERT THE BINARY TO ZONED DECIMAL. THE			*
4392	*			APPROPRIATE SIGN IS THEN ENTERED, AND CONTROL RETURNED TO THE			*
4393	*			CALLING ROUTINE.			*
4394	*						*
4395	*			*ENTRY POINTS			*
4396	*			* THE ENTRY IS CENYZD. THE FORMAT OF THE CALLING SEQUENCE IS			*
4397	*			AS FOLLOWS:			*
4398	*			B I\$CALL			*
4399	*			DC AL2(V\$CNXZ)			*
4400	*			* INDEX REGISTER 1 (@BR) POINTS TO THE PAGE BEGINNING, AND IS			*
4401	*			USED AS THE BASE REGISTER, ALTHOUGH CENXID IS DISPLACED FROM			*
4402	*			THE PAGE BEGINNING.			*
4403	*						*
4404	*			*INPUT			*
4405	*			THE INPUT IS A ONE-BYTE NORMALIZED EYPONENT IN THE FIRST BYTE			*
4406	*			OF THE FIRST ELEMENT OF THE INTERPRETER STACK. I\$STAK IS A			*
4407	*			CORE LACATION CONTAINING THE ADDRESS OF THE INTERPRETER STACK.			*
4408	*						*
4409	*			*OUTPUT			*
4410	*			THE OUTPUT IS A SIGNED DECIMAL NUMBER, RIGHT JUSTIFIED IN THE			*
4411	*			FIRST INTERPRETER STACK ELEMENT. IT IS NOT LESS THAN -99, NOR			*
4412	*			GREATER THAN +99.			*
4413	*						*
4414	*			*EXTERNAL REFERENCES			*
4415	*			INTERPRETER STACK - FIRST ELEMENT			*
4416	*			I\$STAK - CORE LOCATION OF THE ADDRESS OF TIE INTERPRETER STACK			*
4417	*			I\$CALL - ENTRY TO PAGING MODULE			*
4418	*			V\$CBNZ - SUBROUTINE TO CONVERT BINARY TO ZONED DECIMAL			*
4419	*			I\$RTRN - ENTRY TO PAGING MODULE RETURN			*
4420	*						*
4421	*			*EXITS, NORMAL			*
4422	*			* EXIT IS BY BRANCHING TO THE PAGING MODULE TO RETURN TO THE			*
4423	*			CALLING ROUTINE.			*
4424	*			* THE RESULT IS PLACED IN THE LAST TWO BYTES OF THE FIRST STACK			*
4425	*			ELEMENT, AND THE REMAINDER OF THE ELEMENT SET TO DECIMAL ZEROES.			*
4426	*						*
4427	*			*EXITS, ERROR			*
4428	*			NONE.			*
4429	*						*
4430	*			*TABLES/WORK AREAS			*
4431	*			CONSTANTS RESIDE AT THE END OF THE EYECUTABLE CODE, AND ARE			*
4432	*			REFERENCED BY A DISPLACEMENT RELATIVE TO THE VALUE IN @BR.			*
4433	*			THERE IS NO WORK AREA.			*
4434	*						*

CENXZD - CONVERT A NORM EXPONENT TO A DEC NUM.

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE	21
		4435	*	ATTRIBUTES				*
		4436	*	REUSABLE. NATURALLY RELOCATABLE.				*
		4437	*					*
		4438	*	CHARACTER CODE DEPENDENCY				*
		4439	*	THE OPERATION OF THIS MODULE DEPENDS UPON A ZONED DECIMAL DIGIT				*
		4440	*	BEING REPRESENTED WITH THE ZONE (FIRST FOUR BITS) BEING AN 'F'				*
		4441	*	FOR POSITIVE, AND A 'D' FOR NEGATIVE. THIS PROPERTY IS USED AT				*
		4442	*	LABEL CEN150.				*
		4443	*	THE LOW ORDER FOUR BITS OF DECIMAL NUMBERS, WHEN CONSIDERED AS A				*
		4444	*	BINARY INTEGER, IDENTIFY THE VALUE OF THE DIGIT.				*
		4445	*					*
		4446	*	NOTES				*
		4447	*	ERROR PROCEDURES				*
		4448	*	NONE.				*
		4449	*					*
		4450	*	REGISTER USAGE				*
		4451	*	* INDEX REGISTER 1 (@BR) POINTS TO THE FIRST BYTE OF THE PAGE,				*
		4452	*	AND IS USED AS THE BASE REGISTER.				*
		4453	*	* INDEX REGISTER 2 (@XR) IS USED TO REFERENCE THE INTERPRETER				*
		4454	*	STACK.				*
		4455	*					*
		4456	*	SAVED/RESTORED AREAS				*
		4457	*	NONE.				*
		4458	*					*
		4459	*	REQUIRED MODULES				*
		4460	*	QSYSEQ - COMMON SYSTEM EQUATES				*
		4461	*	\$V\$EQU - FIXED VIRTUAL ADDRESSES				*
		4462	*	\$B@EQU - COMPILER SYSTEM EQUATES				*
		4463	*	I\$EQU - FIXED EQUATES				*
		4464	*	\$I@SEQ/\$I@LEQ - STANDARD/LONG PRECISION EXECUTION EQUATES				*
		4465	*					*
		4466	*	OTHER				*
		4467	*	NONE.				*
		4468	*	*****				*

CENXZD - CONVERT A NORM EXPONENT TO A DEC NUM.

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 22

```

4470 *****
4471 * CENXZD - S/3.7 CONVERT NORMALIZED EXPONENT TO ZONED DECIMAL *
4472 *****
4473 *
4474 *CENAD1 VPAGE 112
0470 4475 ORG *,256,112 SET STARTING ADDRESS
0371 4476 CENAD1 EQU * START OR PROGRAM COOING
0400 4477 ORG *-255 RESET IAR TO PAGE
4478 ORG *,256,0 * BOUNDARY ADDRESS
0470 4479 USING *,@BR SET PAGE BASE ADDRESS
4480 ORG CENAD1 RESET STARTING ADDRESS
4481 *** END OF EXPANSION ***
4482 *
4483 * EXECUTION ENTRY TO CENXZD FOLLOWS
4484 *
0470 35 02 0D4E 4485 CENXZD L I$STAK,@XR LOAD STACK POINTER
0474 BD 80 00 4486 CLI I@1SE1+I@DEXP(,@XR),B@NXZR COMPARE EXP TO NORMAL. ZERO
0477 D0 84 9E 4487 BH CEN200(,@BR) BRANCN IF EXP IS POSITIVE
047A D0 82 87 4488 BL CEN100(,@BR) BRANCN TO NEG EXP ROUTINE
047D BC F0 07 4489 MVI I@RSE1(,@XR),@DZERO GET DECIMAL ZERO
0480 AC 06 06 07 4490 MVC I@RSE1-1(I@PREC,@XR),I@RSE1(,@XR) CLEAR A TO 0 AS EXP = 0
0484 D0 87 A8 4491 B CEN900(,@BR) RETURN TO CALLING PROGRAM
4492 *
4493 * FOR NEGATIVE EXPONENT (NORMALIZED VALUE LESS THAN X'80'), SUBTRACT
4494 * IT FROM X'80' AND THE RESULT IS A POSITIVE BINARY NUMDER, EQUAL TO
4495 * THE NON-NORMALIZED MAGNITUDE OR THE EXPONENT. CONVERT THIS TO ZONED
4496 * DECIMAL AND THEN MAKE TNE SIGN OF THE RESULT NEGATIVE.
4497 *
0487 BC 80 01 4498 CEN100 MVI I@1SE1+1(,@XR),B@NXZR SET BYTE TO X'80' (NORM. 0)
048A AF 00 01 00 4499 SLC I@1SE1+1(1,@XR),I@1SE1+I@DEXP(,@XR) SUBTRACT EXP FROM THI
048E AC 00 00 01 4500 MVC I@1SE1+I@DEXP(1,@XR),I@1SE1+1(,@XR) RESULT BACK.INTO
0492 C0 87 12B1 4501 B I$CALL CONVERT TO DECIMAL NUMBER
0496 0CB2 0497 4502 DC AL(@VADDR)(V$CBNZ)
0498 BB 20 07 4503 CEN150 SBF I@RSE1(,@XR),X'20' MAKE SIGN MINUS
049B D0 87 A8 4504 B CEN900(,@BR) RETURN
4505 *
4506 * FOR POSITIVE EXPONENT (NORMALIZED VALUE EQUAL TO OR GREATER THAN
4507 * X'80'), SUBTRACT X'80' FROM IT AND TNE RESULT IS A POSITIVE BINARY
4508 * NUMBER EQUAL TO THE NON-NORMALIZED VALUE OR THE EXPONENT.
4509 * CONVERT THIS TO ZONED DECIMAL AND RETURN.
4510 *
049E 9F 00 00 AC 4511 CEN200 SLC I@1SE1+I@DEXP(1,@XR),CENZRO(,@BR) SUBTRACT X'80' FROM EXP
04A2 C0 87 12B1 4512 B I$CALL CONVERT TO DEC NUMBER
04A6 0CB2 04A7 4513 DC AL(@VADDR)(V$CBNZ)
04A8 C0 87 12D3 4514 CEN900 B I$RTRN RETURN TO THE CALLING PROGRAM
4515 *
4516 * CONSTANTS FOR CENXZD FOLLOW
4517 *
04AC 80 04AC 4518 CENZRO DC AL1(B@NXZR) NORMALIZED ZERO
4519 *
4520 * END OF CENXZD CODING
4521 *

```

CENXZD - CONVERT A NORM EXPONENT TO A DEC NUM.

```
ERR LOC  OBJECT CODE          ADDR STMT SOURCE STATEMENT          VER 15, MOD 00  31/05/21  PAGE  23
```


CCZDFP - CONVERT DEC ZONED NUM TO FLOAT POINT NUM

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 24
4524				*****			
4525	*			5703-XM1 COPYRIGHT IBM CORP. 1970			*
4526	*			REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
4527	*						*
4528				*****			*
4529				*STATUS			*
4530	*			VERSION 1 MODIFICATION 0			*
4531	*						*
4532				*FUNCTION			*
4533	*	*		CCZDFP CONVERTS A ZONES DECIMAL NUMBER IN THE FIRST TWO			*
4534	*			INTERPRETER STACK ELEMENTS TO AN UNPACKED FLOATING POINT			*
4535	*			NUMBER IN THE FIRST STACK ELEMENT.			*
4536	*	*		THE MAGNITUDE OF THE NUMBER CANNOT BE GREATER THAN 10**(PREC+1)			*
4537	*			SINCE THE DECIMAL POINT OF THE INPUTTED NUMBER IS BETWEEN THE			*
4538	*			LAST BYTE OF THE FIRST STACK ELEMENT, AND THE FIRST BYTE OF			*
4539	*			THE SECOND STACK ELEMENT. THEREFORE. THE MAGNITUDE CAN ONLY			*
4540	*			BE LESS THAN 10**(-(PREC+2)) IF IT IS ZERO.			*
4541	*						*
4542				*ENTRY POINTS			*
4543	*	*		THE ENTRY DS CCIOFP. THE FORMAT OF THE CALLING SEQUENCE IS			*
4544	*			AS FOLLOWS:			*
4545	*		B	I\$CALL			*
4546	*		DC	AL2(V\$CZFP)			*
4547	*						*
4548				*INPUT			*
4549	*	*		THE INPUT IS A ZONED DECIMAL NUMBER IN THE FIRST TWO ELEMENTS			*
4550	*			OF THE INTERPRETER STACK. THE FIRST ELEMENT CONTAINS THE			*
4551	*			INTEGER PORTION OF THE NUMBER, AND THE SECOND THE TRACTIONAL.			*
4552	*	*		REGISTER 1 (@BR) CONTAINS THE ADDRESS OF THE FIRST BYTE OF THE			*
4553	*			PAGE. IT IS USED AS THE BASE REGISTER.			*
4554	*	*		I\$STAK CONTAINS THE ADDRESS OF THE FIRST BYTE OF THE STACK.			*
4555	*						*
4556				*OUTPUT			*
4557	*			THE UNPACKED FLOATING POINT NUMBER IS LEFT IN THE FIRST ELEMENT.			*
4558	*						*
4559				*EXTERNAL REFERENCES			*
4560	*			INTERPRETER STACK - FIRST ELEMENT			*
4561	*			I\$STAK - CORE LOCATION OF THE ADDRESS OF TIE INTERPRETER STACK			*
4562	*			I\$CALL - ENTRY TO PAGING MODULE			*
4563	*			I\$RTRN - ENTRY TO PAGING MODULE RETURN			*
4564	*			I\$FWRK - INTERPRETER WORK AREA, FIRST 8(16) BYTES			*
4565	*						*
4566				*EXITS, NORMAL			*
4567	*	*		EXIT IS BY BRANCHING TO THE PAGING MODULE TO RETURN TO THE			*
4568	*			CALLING ROUTINE.			*
4569	*	*		THE RESULT IS LEFT IN THE FIRST STACK ELEMENT.			*
4570	*						*
4571				*EXITS, ERROR			*
4572	*			NONE.			*
4573	*						*
4574				*TABLES/WORK AREAS			*
4575	*			CONSTANTS AND WORK AREAS RESIDE AT THE END OF THE EXECUTABLE CODE,			*
4576	*			AND ARE REFERENCED BY A DISPLACEMENT RELATIVE TO THE VALUE IN @BR.			*
4577	*						*
4578				*ATTRIBUTES			*
4579	*			REUSABLE, NATURALLY RELOCATABLE.			*

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 25
		4580	*				*
		4581	*	*CHARACTER CODE DEPENDENCY			*
		4582	*	THE OPERATION OF THIS MODULE DEPENDS UPON A ZONED DECIMAL DIGIT			*
		4583	*	BEING REPRESENTED WITH THE ZONE (FIRST FOUR BITS) BEING AN -F-			*
		4584	*	FOR POSITIVE, AND A -D- FOR NEGATIVE. THIS PROPERTY IS USED AT			*
		4585	*	THE ENTRY			*
		4586	*	THE LOW ORDER FOUR BITS OF DECIMAL NUMBERS, WHEN CONSIDERED AS A			*
		4587	*	BINARY INTEGER, IDENTIFY THE VALUE OF THE DIGIT.			*
		4588	*				*
		4589	*	*NOTES			*
		4590	*	ERROR PROCEDURES			*
		4591	*	NONE.			*
		4592	*				*
		4593	*	RESISTER USAGE			*
		4594	*	* REGISTER 1 (@BR) LSE0 AS TOIE SASE REGISTER.			*
		4595	*	* REGISTER 2 (@XR) IS USED TO REFERENCE THE INTERPRETER STACK.			*
		4596	*				*
		4597	*	SAVED/RESTORED AREAS			*
		4598	*	NONE.			*
		4599	*				*
		4600	*	REQUIRED MODULES			*
		4601	*	QSYSEQ - COMMON SYSTEM EQUATES			*
		4602	*	\$V\$EQU - FIXED VIRTUAL ADDRESSES			*
		4603	*	\$B@EQU - COMPILER SYSTEM EQUATES			*
		4604	*	I\$EQU - FIXED EQUATES			*
		4605	*	\$I@SEQ/\$I@LEQ - STANDARD/LONG PRECISION EXECUTION EQUATES			*
		4606	*				*
		4607	*	OTHER			*
		4608	*	NONE			*
		4609	*	*****			*

CCZDFP - CONVERT DEC ZONED NUM TO FLOAT POINT NUM

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 26
		4611		*****	
		4612	*	CCZDFP - S/3.7 CONVERT ZONED DECIMAL TO FLOATING POINT ROUTINE	*
		4613		*****	
		4614	*		
04AD		4615	*CCZAD1	VPAGE 0 SET PAGE ADDRESSABILITY	
		4616	ORG *	SET STARTING ADDRESS	
03AE		4617	CCZAD1 EQU *	START OF PROGRAM CODING	
0400		4618	ORG *-255	RESET IAR TO PAGE	
		4619	ORG *,256,0	* BOUNDARY ADDRESS	
		4620	USING *,@BR	SET PAGE BASE ADDRESS	
04AD		4621	ORG CCZAD1	RESET STARTING ADDRESS	
		4622	***	END OF EXPANSION ***	
		4623	*		
		4624	*	EXECUTION ENTRY TO CCZDFP FOLLOWS	
		4625	*		
04AD 35 02 0D4E		4626	CCZDFP L	I\$STAK,@XR LOAD STACK POINTER	
04B1 6C 00 F9 0F		4627	MVC	CCZSGN(1,@BR),I@RSE2(,@XR) SAVE SIGN OF INPUT WIER	
04B5 BA F0 0F		4628	SBN	I@RSE2(,@XR),B@ZPOS INSERT PLUS SIGN IN ITS PLACE	
04B8 7C 87 FA		4629	MVI	CCZEXP(,@BR),B@NXZR+I@PREC INITIALIZE EXPONENT TO PREC	
		4630	*		
		4631	*	CLEAR THIRD STACK ELEMENT TO ZEROES BECAUSE THE MANTISSA OF THE	
		4632	*	FLOATING POINT RESULT WILL INCLUDE THE LEADING BYTES OR IT IF THE	
		4633	*	DECIMAL POINT OR THE INPUT NUMBER IS MORE THAN ONE PLACE TO THE	
		4634	*	LEFT OF THE FIRST SIGNIFICANT DIGIT.	
		4635	*		
04BB BC F0 17		4636	MVI	I@RSE3(,@XR),@DZERO GET DECIMAL ZERO	
04BE AC 06 16 17		4637	MVC	I@RSE3-1(I@PREC,@XR),I@RSE3(,@XR) CLEAR THIRD STACK ELT	
		4638	*		
		4639	*	MOVING FROM LEFT TO RIGHT, TEST EACH DIGIT OF ZONED DECIMAL INPUT	
		4640	*	NUMBER UNTIL THE FIRST SIGNIFICANT DIGIT IS ENCOUNTERED.	
		4641	*		
04C2 B9 0F 01		4642	CCZ020 TBF	I@1SE1+I@MANL(,@XR),X'0F' IS THE DIGIT A ZERO	
04C5 D0 90 DF		4643	BF	CCZ100(,@BR) NO, EXIT FROM LOOP	
04C8 E2 02 01		4644	LA	CCZONE(,@XR),@XR YES, MOVE POINTER TO NEXT DIGIT	
04CB 5F 00 FA FB		4645	SLC	CCZEXP(1,@BR),CCZDC1(,@BR) DECREMENT EXPONENT BY 1	
		4646	*		
		4647	*	WHEN THE EXPONENT HAS BEEN DECREMENTED TO -(PREC+1), THE LOOP IS	
		4648	*	TERMINATED, BECAUSE THE ZONED DECIMAL INPUT WAS ALL ZEROES. TO	
		4649	*	OUTPUT A FLOATING POINT ZERO, THE EXPONENT IS SET TO -98 WITH THE	
		4650	*	MANTISSA LEFT AS ZEROES.	
		4651	*		
04CF 7D 78 FA		4652	CLI	CCZEXP(,@BR),B@NXZR-I@LUFV HAVE ALL DIGITS BEEN TESTED ?	
04D2 D0 84 C2		4653	BH	CCZ020(,@BR) YES, REPEAT TEST	
04D5 35 02 0D4E		4654	L	I\$STAK,@XR NO, RESTORE INDEX REG	
04D9 BC 1E 00		4655	MVI	I@1SE1+I@DEXP(,@XR),B@NXLO INSERT EXPONENT OF -98	
04DC D0 87 F5		4656	B	CCZ900(,@BR) RETURN TO CALLING PROGRAM	
		4657	*		
		4658	*	AT THIS POINT THE FIRST NON-ZERO DIGIT OF THE IONFD DECIMAL INPUT	
		4659	*	NUMBER IS AT I@1SE1+1. HENCE, THE MANTISSA OR THE RESULT STARTS HERE	
		4660	*	AND ENDS AT I@RSE1. SINCE THE INDEX REG HAS BEEN INCREMENTED, THIS	
		4661	*	MAY RUN INTO THE THIRD STACK ELEMENT. PLACE THE MANTISSA IN THE	
		4662	*	STACK VIA A TEMPORARY STORAGE LOCATION.	
		4663	*		
04DF 2C 06 060E 07		4664	CCZ100 MVC	I\$FWRK+I@PREC(I@PREC),I@RSE1(,@XR) GET MANTISSA	
04E4 35 02 0D4E		4665	L	I\$STAK,@XR RESTORE INDEX REG.	
04E8 8C 06 07 060E		4666	MVC	I@RSE1(I@PREC,@XR),I\$FWRK+I@PREC PUT MATISSA IN STACK	

FGSEXP - EXPONENTIAL FUNCTION, STANDARD PREC

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 28
4684				*****			
4685	*			5703-XM1 COPYRIGHT IBM CORP. 1970			*
4686	*			REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
4687	*						*
4688				*****			*
4689				*STATUS			*
4690	*			VERSION 1 MODIFICATION 0			*
4691	*						*
4692				*FUNCTION			*
4693	*			* FGSEXP CALCULATES THE VALUE OF THE EXPONENTIAL FUNCTION, WHERE			*
4694	*			THE INPUT ARGUMENT IS THE EXPONENT.			*
4695	*			* THE EXPONENTIAL FUNCTION IS CALCULATED BY A RECURSIVE ALGORITHM			*
4696	*			AFTER REDUCING THE ARGUMENT TO THE RANGE (0,LOG(10)).			*
4697	*						*
4698				*ENTRY POINTS			*
4699	*			THE ENTRY IS FGSEXP. THE FORMAT OF THE CALLING SEQUENCE IS AS			*
4700	*			FOLLOWS:			*
4701	*			B I\$CALL			*
4702	*			DC AL2(V\$FEXP)			*
4703	*						*
4704				*INPUT			*
4705	*			* THE INPUT IS AN UNPACKED FLOATING POINT NUMBER IN THE FIRST			*
4706	*			ELEMENT OF THE INTERPRETER STACK.			*
4707	*			* REGISTER 1 (@BR) POINTS TO THE BEGINNING OF THE PAGE.			*
4708	*			* I\$STAK POINTS TO THE INTERPRETER STACK.			*
4709	*						*
4710				*OUTPUT			*
4711	*			* THE RESULT IS LEFT IN THE FIRST STACK ELEMENT, IN UNPACKED			*
4712	*			FLOATING POINT FORMAT.			*
4713	*			* IN THE EVENT OF AN ERROR, AN ERROR CODE IS LEFT IN THE ONE-BYTE			*
4714	*			INTERPRETER ERROR AREA, I\$ERRC.			*
4715	*						*
4716				*EXTERNAL REFERENCES			*
4717	*			* INTERPRETER STACK - THE FIRST TWO ELEMENTS (16 BYTES)			*
4718	*			* I\$FWRK - INTERPRETER WORK AREA, FIRST 25 BYTES			*
4719	*			* I\$STAK - CORE LOCATION OF ADDRESS OF THE INTERPRETER STACK			*
4720	*			* I\$ERRC - INTERPRETER ONE-BYTE ERROR AREA			*
4721	*			* I\$CALL - ENTRY TO THE PAGING MODULE			*
4722	*			* I\$RTRN - ENTRY TO PAGING MODULE RETURN			*
4723	*			* CBFZD - CONVERT FLOATING POINT TO ZONED DECIMAL			*
4724	*			* FSKCNT - EQUATE TO CONSTANTS LOCATED IN FKSLOG			*
4725	*			* I\$LDXR - ENTRY, CONVERT VADDR, LOAD @XR			*
4726	*						*
4727				*EXITS, NORMAL			*
4728	*			* EXIT IS BY BRANCHING TO I\$RTRN, THE ENTRY TO THE PAGING RETURN			*
4729	*			ROUTINE.			*
4730	*			* THE RESULT IS PLACED IN THE FIRST STACK ELEMENT.			*
4731	*						*
4732				*EXITS, ERROR			*
4733	*			* AN ERROR CODE IS PLACED IN THE ERROR AREA, I\$ERRC.			*
4734	*			* EXIT IS BY BRANCHING TO I\$RTRN, THE ENTRY TO THE PAGING RETURN			*
4735	*			ROUTINE.			*
4736	*						*
4737				*TABLES/WORK AREAS			*
4738	*			* THE CONSTANTS FOR THE ITERATION ARE STORED IN THE SECOND PAGE			*
4739	*			OF FKSLOG, AND ARE REFERENCED BY LOADING THE ADDRESS OF THE			*

FGSEXP - EXPONENTIAL FUNCTION, STANDARD PREC

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 29
		4740	*	PAGE INTO @XR, AND USING THE EQUATED DISPLACEMENT, FKSCNT.			*
		4741	*	* THE REST OF THE CONSTANTS AND WORKAREAS ARE AT THE END OF THE			*
		4742	*	EXECUTABLE CODE ON EACH PAGE.			*
		4743	*				*
		4744	*	*ATTRIBUTES			*
		4745	*	REUSABLE, NATURALLY RELOCATABLE.			*
		4746	*				*
		4747	*	*CHARACTER CODE DEPENDENCY			*
		4748	*	THE OPERATION OF THIS MODULE DEPENDS UPON A ZONED DECIMAL DIGIT			*
		4749	*	BEING REPRESENTED WITH THE ZONE (FIRST FOUR BITS) BEING AN -F-			*
		4750	*	FOR POSITIVE, AND A -D- FOR NEGATIVE. THIS PROPERTY IS USED			*
		4751	*	AT THE ENTRY.			*
		4752	*	THE LOW ORDER FOUR BITS OF DECIMAL NUMBERS, WHEN CONSIDERED AS A			*
		4753	*	BINARY INTEGER, IDENTIFY THE VALUE OF THE DIGIT.			*
		4754	*				*
		4755	*	*NOTES			*
		4756	*	ERROR PROCEDURES			*
		4757	*	THE ERROR CODE IS SET AND CONTROL RETURNED TO THE PACING			*
		4758	*	ROUTINE.			*
		4759	*				*
		4760	*	REGISTER USAGE			*
		4761	*	* REGISTER 1 (@BR) IS USED AS THE BASE REGISTER.			*
		4762	*	* RESISTER 2 (@XR) IS USED TO REFERENCE THE STACK. IT IS ALSO			*
		4763	*	USED TO REFERENCE THE CONSTANTS IN FKSLOG.			*
		4764	*				*
		4765	*	SAVED/RESTORED AREAS			*
		4766	*	NONE.			*
		4767	*				*
		4768	*	MODIFICATION CONSIDERATIONS			*
		4769	*	NONE.			*
		4770	*				*
		4771	*	REQUIRED MODULES			*
		4772	*	@SYSEQ - COMMON SYSTEM EQUATES			*
		4773	*	@ERMEQ - ERROR MESSAGE EQUATES			*
		4774	*	\$V\$EQU - FIXED VIRTUAL ADDRESSES			*
		4775	*	\$B\$EQU - COMPILER SYSTEM EQUATES			*
		4776	*	\$I\$EQU - FIXED EQUATES			*
		4777	*	\$I@SEQ - STANDARD PRECISION EXECUTION EQUATES			*
		4778	*	FKSLOG - LOGARITHM FUNCTIONS STANDARD PRECISION			*
		4779	*				*
		4780	*	OTHER			*
		4781	*	NONE.			*
		4782	*				*
		4783	*	*****			*

FGSEXP - EXPONENTIAL FUNCTION, STANDARD PREC

ERR	LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 30
				4785	*****			
				4786	* FGSEXP - S/3.7 EXPONENTIAL FUNCTION ROUTINE - STANDARD PRECISION *			
				4787	*****			
				4788	*			
0500				4789	ORG *,B@LVP,0			
			0500	4790	USING *,@BR			
				4791	*			
				4792	* EXECUTION ENTRY TO FGSEXP FOLLOWS			
				4793	*			
				4794	* TEST IF THE ARG <= -1000. IF 50, SET THE ARGUMENT FOR AN ANSWER OR			
				4795	* ZERO. THIS WILL BE DONE ON THE SECOND PAGE.			
				4796	*			
0500	C0 87 12B1			4797	FGSEVP B I\$CALL CALL SECOND 1-3			
0504	0600		0505	4798	DC AL(@VADDR)(V\$FEXP+256) * PAGE OF FGSEXP 1-3			
				4799	*			
				4800	* CHECK MAGNITUDE OF THE EXPONENT OF FLOATING POINT ARGUMENT BEFORE			
				4801	* CONVERTING IT TO ZONED DECIMAL BECAUSE CONVERSION ROUTINE DOES NOT			
				4802	* CHECK THE EXPONENT SIZE			
				4803	*			
0506	BD 84 00			4804	CLI I@1SE1+I@DEXP(,@XR),B@NXZR+4 IS ARGUMENT >= 1000 ?			
0509	D0 82 13			4805	BL FGS010(,@BR) BRANCH IF NO			
050C	3C E1 0CBC			4806	FGS005 MVI I\$ERRC,@E776 SET ARGUMENT TOO LARGE CODE			
0510	D0 87 CB			4807	B FGS900(,@BR) RETURN TO CALLING PROGRAM			
				4808	*			
				4809	* CONVERT FLOATING POINT ARGUMENT TO ZONED DECIMAL NUMBER.			
				4810	* THE FRACTIONAL PART WILL BE I@LUFV BYTES LONG IN THE SECOND STACK -			
				4811	* ELEMENT. AND THE INTEGER PART WILL BE A MAXIMUM OF 3 BYTES LONG.			
				4812	* RIGHT JUSTIFIED, IN THE FIRST STACK ELEMENT.			
				4813	*			
0513	C0 87 12B1			4814	FGS010 B I\$CALL CONVERT			
0517	0C70		0518	4815	DC AL(@VADDR)(V\$CFPZ) * FLOATING POINT TO ZONED DEC			
				4816	*			
				4817	* THE CONVERSION ROUTINE MAKES NO ERROR CHECKS, SO I\$ERRC IS NOT			
				4818	* CHECKED ON RETURN			
				4819	*			
0519	B8 F0 0F			4820	TBN I@RSE2(,@XR),B@ZPOS IS ARGUMENT POSITIVE ?			
051C	D0 90 3B			4821	BF FGS110(,@BR) BRANCH IF NEGATIVE			
				4822	*			
				4823	* CHECK IF THE POSITIVE ARGUMENT IS GREATER THAN OR EQUAL TO 99*LOG(10)			
				4824	* BASE E. IF SO, EXP(ARG) WILL BE GREATER THAN OR EQUAL TO 10**99,			
				4825	* AND OVERFLOW WILL OCCUR.			
				4826	*			
051F	9D 09 0E DF			4827	CLC I@RSE2-FGSONE(I@PREC+FGSTHR,@XR),FGSNNN(,@BR) ARG TOO LRG			
0523	D0 02 0C			4828	BNL FGS005(,@BR) IF YES BRANCH TO SET ERROR FLAG			
				4829	*			
				4830	* REDUCE RANGE OF POSITIVE ARG TO (0,LOG(10))			
				4831	*			
0526	7C 00 FB			4832	MVI FGSXM1(,@BR),@ZERO SET N = 0			
0529	9D 0A 0F F4			4833	FGS100 CLC I@RSE2(I@LUFV+FGSTHR,@XR),FGSTEN(,@BR) IS ARG < LOG(10) ?			
052D	D0 82 5D			4834	BL FGS210(,@BR) IF YES, EVALUATE FUNCTION			
0530	97 28 0F F4			4835	SZ I@RSE2(I@LUFV+FGSTHR,@XR),FGSTEN(I@LUFV+FGSONE,@BR) -LOG 1			
0534	5E 00 FB CF			4836	ALC FGSXM1(1,@BR),FGSBN1(,@BR) INCREMENT N BY 1			
0538	D0 87 29			4837	B FGS100(,@BR) REPEAT TEST			
				4838	*			
				4839	* IF NEGATIVE ARGUMENT IS EQUAL TO OR LESS THAN -98*LOG(10) BASE,			
				4840	* SET RESULT TO ZERO AND RETURN TO CALLING PROGRAM			

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15,	MOD 00	31/05/21	PAGE	31
					4841	*						
					4842	*	IN CHECKING THE SIZE OF THE ARGUMENT, ONLY EXAMINE PART PRECEDING					
					4843	*	THE SIGN BYTE, SO THE NEGATIVE ARGUMENT CAN BE TREATED AS POSITIVE,					
					4844	*	WITH THE LAST BYTE MISSING					
					4845	*						
	053B	9D	09	0E	E9	4846	FGS110 CLC	I@RSE2-1(I@PREC+FGSTHR,@XR),FGSMNN(,@BR)	>	-98*LOG(10)		
	053F	D0	82	4C		4847	BL	FGS115(,@BR)	IF YES, CONTINUE			
	0542	BC	1E	00		4848	MVI	I@1SE1+I@DEXP(,@XR),B@NXLO	IF NO, SET EXP=-98, AS RESULT=			
	0545	A7	06	07	07	4849	SZ	I@RSE1(I@PREC,@XR),I@RSE1(I@PREC,@XR)	ZERO MANTISSA	1-4		
	0549	D0	87	CB		4850	B	FGS900(,@BR)	RETURN WITH ZERO RESULT			
					4851	*						
					4852	*	REDUCE RANGE OF NEGATIVE ARGUMENT TO (0,LOG(10))					
					4853	*						
	054C	7C	FF	FB		4854	FGS115 MVI	FGSXM1(,@BR),X'FF'	SET N = -1			
	054F	96	28	0F	F4	4855	FGS120 AZ	I@RSE2(I@LUFV+FGSTHR,@XR),FGSTEN(I@LUFV+FGSONE,@BR)	+LOG10			
	0553	D0	84	5D		4856	BP	FGS210(,@BR)	IF POSITIVE EVALUATE FUNCTION			
	0556	5F	00	FB	CF	4857	SLC	FGSXM1(1,@BR),FGSBN1(,@BR)	DECREMENT N BY 1			
	055A	D0	87	4F		4858	B	FGS120(,@BR)	REPEAT PROCESS			
					4859	*						
					4860	*	ARGUMENT IS NOW IN THE RANGE (0,LOG(10)) AND EXP(ARG) CAN BE					
					4861	*	EVALUATED BY THE ALGORITHM. THE WORK AREA IN THE INTERPRETER IS					
					4862	*	USED SO THAT THE INDEX REGISTER IS FREE FOR USE AS A BASE RESISTER					
					4863	*	FOR THE PAGE CONTAINING THE STORED CONSTANTS LOG(1+10**(-J))					
					4864	*						
	055D	2C	08	06	17 0F	4865	FGS210 MVC	I\$FWRK+I@LUFV+I@LUFV(I@LUFV+FGSONE),I@RSE2(,@XR)	ARG IN WA			
	0562	3C	F0	06	0E	4866	MVI	I\$FWRK+I@PREC,@DZERO	GET DECIMAL ZERO			
	0566	0C	06	06	0D 060E	4867	MVC	I\$FWRK+I@PREC-FGSONE(I@PREC),I\$FWRK+I@PREC	CLEAR TO ZEROS			
	056C	3C	F1	06	07	4868	MVI	I\$FWRK,B@DEC1	INSERT LEADING 1			
					4869	*						
					4870	*	FIRST ELEMENT OF WORK AREA CONTAINS 1.0000000 (LENGTH I@LUFV)					
					4871	*	SECOND ELEMENT OF WORK AREA CONTAINS THE ARGUMENT X.XXXXXXXXXX					
					4872	*	(LENGTH I@LUFV+1 WITH 1 DIGIT INTEGER PART).					
					4873	*						
					4874	*	SET INDEX REGISTER TO BASE ADDRESS OF PAGE CONTAINING STORED CON-					
					4875	*	STANTS, AND BRING THAT PAGE INTO CORE					
					4876	*						
	0570	C0	87	13	30	4877	B	I\$LDXR	SET INDEX RES			
	0574	03	00			4878	DC	AL(@VADDR)(V\$FLGT+256)	TO PAGE CONTAINING CONSTANTS			
					4879	*						
					4880	*	INITIALIZE INSTRUCTION AT FGS250. THIS INSTRUCTION MODIFIES THE					
					4881	*	NUMBER IN THE FIRST ELEMENT OF WORK AREA (INITIALLY 1.0000000) BY					
					4882	*	MULTIPLYING IT BY 1.10**(-J) WHERE J IS THE ITERATION NUMBER.					
					4883	*						
	0576	5C	04	8A	FA	4884	MVC	FGS250+@OP2(FGSINL,@BR),FGSINS+@OP2(,@BR)	INITIALIZE INST			
	057A	7C	00	FC		4885	MVI	FGSITN(,@BR),@ZERO	SET ITERATION NUMBER TO ZERO			
					4886	*						
					4887	*	*****					
					4888	*	ALGORITHM FOR EVALUATION OF EXPONENTIAL FUNCTION					*
					4889</							

FGSEXP - EXPONENTIAL FUNCTION, STANDARD PREC

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER	15,	MOD	00	31/05/21	PAGE	32
					4897	*	MULTIPLY NUMBER IN FIRST ELEMENT OF WORK AREA BY 1.10**(-J, WHERE							
					4898	*	J IS THE ITERATION NUMBER, THE SECOND OPERAND IS MODIFIED DURING							
					4899	*	EXECUTION. ITS CURRENT VALUE IS I\$FWRK+I@PREC-J(I@LUFL-J)							
					4900	*								
	0585	06	07	060E	0000	4901	FGS250 AZ I\$FWRK+I@PREC(I@LUFV),*-(I@LUFV)							
					4902	*								
					4903	*	SECOND DISPLACEMENT IS I\$FWRK+I@PREC-J(I@LUFV-J)							
					4904	*								
	058B	D0	87	7D		4905	B FGS220(,@BR) REPEAT TEST							
	058E	7D	07	FC		4906	FGS260 CLI FGSITN(,@BR),I@PREC WAS THIS FINAL ITERATION ?							
	0591	D0	81	A7		4907	BE FGS300(,@BR) BRANCH IF YES. IF NO, RESTORE							
	0594	26	17	0617	93	4908	AZ I\$FWRK+I@LUFV+I@LUFV(I@LUFV+FGSONE),FKSCNT(I@LUFV,@XR)							
	0599	5E	00	FC	CF	4909	ALC FGSITN(1,@BR),FGSBN1(,@BR) INCREMENT ITERATION NUMBER							
	059D	E2	02	08		4910	LA I@LUFV(,@XR),@XR INCREMENT @XR BY I\$LUFV							
					4911	*								
					4912	*	MODIFY INSTRUCTION AT FGS250 BY SIMULTANEOUSLY INCREMENTING LENGTH							
					4913	*	CODE OF THE FIRST OPERAND BY 1, DECREMENTING LENGTH CODE OF SECOND							
					4914	*	OPERAND BY 1, AND DECREMENTING THE ADDRESS OF SECOND OPERAND BY 1.							
					4915	*								
	05A0	5F	04	8A	D5	4916	SLC FGS250+@OP2(FGSINL,@BR),FGSMOD(,@BR) MODIFY INSTRUCTION							
	05A4	D0	87	7D		4917	B FGS220(,@BR) REPEAT ITERATION							
					4918	*								
					4919	*	SINCE THE ARGUMENT RANGE WAS REDUCED TO (0,LOG(10)), THE VALUE OF							
					4920	*	EXP(ARG) IS IN THE RANGE (1.0000000,9.9999999). THUS RESULT (WHICH							
					4921	*	IS IN FIRST ELEMENT OF THE WORK AREA) CAN BE USED AS THE MANTISSA OF							
					4922	*	THE EXP(ORIGINAL ARG) WITH A FLOATING POINT EXPONENT OF N+1 (FROM							
					4923	*	THE WAY IN WHICH THE RANGE WAS REDUCED).							
					4924	*								
	05A7	35	02	0D4E		4925	FGS300 L I\$STAK,@XR RESTORE STACK POINTER TO @XR							
	05AB	16	70	060E	D0	4926	AZ I\$FWRK+I@PREC(I@LUFV),FGSFVE(1,@BR) ROUND RESULT							
	05B0	3D	F0	0607		4927	CLI I\$FWRK,@DZERO HAS LEADING DIGIT BECOME ZERO ?							
	05B4	D0	01	BF		4928	BNE FGS305(,@BR) NO, NO OVERFLOW FROM ROUNDING							
					4929	*								
					4930	*	IF THE LEADING DIGIT IS NOW A ZERO, THE ONLY POSSIBLE EXPLANATION IS							
					4931	*	THAT 9.9999999X HAS BEEN CHANGED INTO 0.0000000X BY ROUNDING (LEADING							
					4932	*	1 WAS LOST DUE TO LENGTH CODE). THUS, SET EXP(ARG) TO 1.0000000 AND							
					4933	*	INCREMENT N (EYP) BY 1.							
					4934	*								
	05B7	3C	F1	0607		4935	MVI I\$FWRK,B@DEC1 MAKE VALUE 1.0000000							
	05BB	5E	00	FB	CF	4936	ALC FGSXM1(,@BR),FGSBN1(,@BR) INCREMENT N BY 1							
	05BF	8C	06	07	060D	4937	FGS305 MVC I@RSE1(I@PREC,@XR),I\$FWRK+I@PREC-1 MANTISSA OF RESULT							
	05C4	BC	81	00		4938	MVI I@1SE1+I@DEXP(,@XR),B@NXZR+1 SET EXP TO +1							
	05C7	9E	00	00	FB	4939	ALC I@1SE1+I@DEXP(1,@XR),FGSXM1(,@BR) ADD N TO UPO4ENT							
					4940	*								
					4941	*	RETURN							
					4942	*								
	05CB	C0	87	12D3		4943	FGS900 B I\$RTRN RETURN TO CALLING PROGRAM							
					4944	*								
					4945	*	NON-STORAGE CONSTANTS FOLLOW							
					4946	*								
	0001	4947	FGSONE	EQU	1		LENGTH OF ONE							
	0003	4948	FGSTHR	EQU	3		LENGTH OF THREE							
	0005	4949	FGSINL	EQU	5		LENGTH OF FIVE							
	000A	4950	FGSNNL	EQU	I@PREC+FGSTHR		LENGTH OF FLT PT 99*LOG(10)							
					4951	*								
					4952	*	STORED CONSTANTS							

FGSEXP - EXPONENTIAL FUNCTION, STANDARD PREC

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE	33
					4953	*					
	05CF	01		05CF	4954	FGSBN1 DC	AL1(@B1)			BINARY 1 FOR INCREMENTS	
	05D0	F5		05D0	4955	FGSFVE DC	DL1'5'			DEC 5 FOR ROUNDING	
	05D1	F100000001		05D5	4956	FGSMOD DC	XL(FGSINL)'F100000001'			INSTRUCTION MODIFIER	
	05D6	F2F2F7F9F5F5F9F2		05DF	4957	FGSNNN DC	DL(FGSNNL)'2279559242'			99*LOG(10) CHECK FOR OVERFLOW	
	05E0	F2F2F5F6F5F3F3F3		05E9	4959	FGSMNN DC	DL(FGSNNL)'2256533391'			99*LOG(10) CHECK FOR ZERO	
	05EA	F0F0F2F3F0F2F5F8		05F4	4961	FGSTEN DC	DL(FGSNNL+FGSONE)'00230258509'			LOG(10) BASE E	
					4962	*					
					4963	* INITIAL INSTRUCTION					
					4964	*					
	05F5	06 07 060E 060E			4965	FGSINS AZ	I\$FWRK+I@PREC(I@LUFV),I\$FWRK+I@PREC(I@LUFV)				
					4966	*					
					4967	* WORK AREA					
					4968	*					
	05FB			05FB	4969	FGSXM1 DS	CL1			N = (EXPONENT OF RESULT) - 1	
	05FC			05FC	4970	FGSITN DS	CL1			J = ITERATION NUMBER	
					4971	*					

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE	34
		4973		*****				
		4974	*	SECOND PAGE OF FGSEXP				
		4975	*					
		4976	*	THIS PAGE TESTS IF THE ARGUMENT IS LESS THAN -1000. IF SO, IT WILL				
		4977	*	BE SET IN THE RANGE OF -1000 < X < -900 IN ORDER TO GET AN ANSWER OF				
		4978	*	ZERO.				
		4979	*					
0600		4980	ORG	*,B@LVPG,0	SET PAGE BOUNDARY		1-3	
	0600	4981	USING	*,@BR	SET BASE REG		1-3	
0600 35 02 0D4E		4982	FGS001 L	I\$STAK,@XR	LOAD STACK POINTER		1-3	
0604 B8 F0 07		4983	TBN	I@RSE1(,@XR),B@ZPOS	IS ARG POSITIVE ?		1-3	
0607 D0 10 14		4984	BT	FGS004(,@BR)	IF SO, RETURN TO CONTINUE		1-3	
060A BD 84 00		4985	CLI	I@1SE1+I@DEXP(,@XR),B@NXZR+4	IS [ARG] >= 1000 ?		1-3	
060D D0 82 14		4986	BL	FGS004(,@BR)	IF NOT, RETURN		1-3	
0610 9C 01 01 19		4987	MVC	I@1SE1+I@MANL(2,@XR),FGSSFZ(,@BR)	SET ARG FOR 0 ANSWER		1-3	
0614 C0 87 12D3		4988	FGS004 B	I\$RTRN	RETURN TO FIRST PAGE		1-3	
		4989	*					
0618 83F9		0619 4990	FGSSFZ DC	XL2'83F9'	FAKE 9XX		1-3	
		4991	*					
0700		4992	ORG	*,B@LVPG,0	START FGSEXP DUMMY 3RD PAGE		1-3	
0800		4993	ORG	*+B@LVPG	SIMULATE FGSEXP DUMMY 3RD PAGE			

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 35
4995				*****			
4996	*			5703-XM1 COPYRIGHT IBM CORP. 1970			*
4997	*			REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
4998	*						*
4999				*****			*
5000				*STATUS			*
5001	*			VERSION 1 MODIFICATION 0			*
5002	*						*
5003				*FUNCTION			*
5004	*			* FBNPWR COMPUTES AN INPUTTED NUMBER RAISED TO AN INPUTTED POWER.			*
5005	*			* IF THE POWER IS A ONE-DIGIT INTEGER, THE RESULT IS CALCULATED			*
5006	*			BY REPEATED MULTIPLICATION OF THE ARGUMENT BY ITSELF, THE			*
5007	*			REQUIRED NUMBER OF TIMES.			*
5008	*			* IF THE POWER IS NOT A POSITIVE ONE-DIGIT INTEGER, A**B IS			*
5009	*			CALCULATED BY EXP(B*LOG(A)), WITH THE APPROPRIATE SIGN ADDED.			*
5010	*						*
5011				*ENTRY POINTS			*
5012	*			* THE ENTRY IS FBNPWR. THE FORMAT OF THE CALLING SEQUENCE IS AS			*
5013	*			FOLLOWS:			*
5014	*			B I\$CALL			*
5015	*			DC AL2(V\$FPWR)			*
5016	*						*
5017				*INPUT			*
5018	*			* THE INPUT IS TWO UNPACKED FLOATING POINT NUMBERS IN THE FIRST			*
5019	*			TWO STACK ELEMENTS. THE FIRST IS THE ARGUMENT, THE SECOND THE			*
5020	*			POWER.			*
5021	*			* REGISTER 1 (@BR) POINTS TO THE BEGINNING OR THE PAGE.			*
5022	*			* I\$STAK POINTS TO THE INTERPRETER STACK.			*
5023	*						*
5024				*OUTPUT			*
5025	*			* THE RESULT IS LEFT IN THE FIRST STACK ELEMENT, IN UNPACKED			*
5026	*			FLOATING POINT FORMAT.			*
5027	*			* IN THE EVENT OF AN ERROR, AN ERROR CODE IS LEFT IN THE ONE BYTE			*
5028	*			INTERPRETER ERROR AREA, I\$ERRC.			*
5029	*						*
5030				*EXTERNAL REFERENCES			*
5031	*			* INTERPRETER STACK - FIRST TWO ELEMENTS (16 (32) BYTES)			*
5032	*			* I\$FWRK - INTERPRETER WORK AREA, FIRST 8(16) BYTES			*
5033	*			* I\$STAK - CORE LOCATION OF ADDRESS OF THE INTERPRETER STACK			*
5034	*			* I\$ERRC - INTERPRETER ONE-BYTE ERROR AREA			*
5035	*			* I\$CALL - ENTRY TO THE PAGING MODULE			*
5036	*			* I\$RTRN - ENTRY TO THE PAGING MODULE RETURN			*
5037	*			* I\$FMPY - ENTRY TO THE MULTIPLY ROUTINE, FZIMPY			*
5038	*			* FJIMPY - MULTIPLY ROUTINE			*
5039	*			* FKSLOG/FKLLOG - LOGARITHM BASE E STANDARD/LONG PRECISION			*
5040	*			* FGSEXP/FGLEXP - EXPONENTIAL FUNCTION STANDARD/LONG PRECISION			*
5041	*						*
5042				*EXITS, NORMAL			*
5043	*			* EXIT IS BY BRANCHING TO I\$RTRN, THE ENTRY TO THE PAGING RETURN			*
5044	*			ROUTINE.			*
5045	*			* THE RESULT IS PLACED IN THE FIRST STACK ELEMENT.			*
5046	*						*
5047				*EXITS, ERROR			*
5048	*			* AN ERROR CODE IS PLACED IN THE ERROR AREA, I\$ERRC.			*
5049	*			* EXIT IS BY BRANCHING TO I\$RTRN			*
5050	*						*

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE	36
		5051	*	*TABLES/WORK AREAS				*
		5052	*	* THE CONSTANTS AND WORK AREAS RESIDE AT THE END OF THE EXECUTABLE				*
		5053	*	* CODE.				*
		5054	*					*
		5055	*	*ATTRIBUTES				*
		5056	*	* REUSABLE, NATURALLY RELOCATABLE.				*
		5057	*					*
		5058	*	*CHARACTER CODE DEPENDENCY				*
		5059	*	* THE OPERATION OF THIS MODULE DEPENDS UPON A ZEROD DECIMAL DIGIT				*
		5060	*	* BEING REPRESENTED WITH THE ZONE (FIRST FOUR BITS) BEING AN -F-				*
		5061	*	* FOR POSITIVE, AND A -D- FOR NEGATIVE. THIS PROPERTY IS USED				*
		5062	*	* FOLLOWING FNB300, AT FNB400, AND FOLLOWING FNB880.				*
		5063	*	* THE LOW ORDER FOUR BITS OF DECIMAL NUMBERS, WHEN CONSIDERED AS A				*
		5064	*	* BINARY INTEGER, IDENTIFY THE VALUE OF THE DIGIT.				*
		5065	*					*
		5066	*	*NOTES				*
		5067	*	* ERROR PROCEDURES				*
		5068	*	* THE ERROR CODE IS SET, AND CONTROL RETURNED TO THE CALLING				*
		5069	*	* ROUTINE.				*
		5070	*					*
		5071	*	* REGISTER USAGE				*
		5072	*	* REGISTER 1 (@BR) IS USED AS THE BASE REGISTER.				*
		5073	*	* REGISTER 2 (@XR) IS USED TO REFERENCE THE STACK.				*
		5074	*					*
		5075	*	* SAVED/RESTORED AREAS				*
		5076	*	* NONE.				*
		5077	*					*
		5078	*	* MODIFICATION CONSIDERATIONS				*
		5079	*	* NONE.				*
		5080	*					*
		5081	*	* REQUIRED MODULES				*
		5082	*	* @SYSEQ - COMMON SYSTEM EQUATES				*
		5083	*	* @ERMEQ - ERROR MESSAGE EQUATES				*
		5084	*	* \$V\$EQU - FIXED VIRTUAL ADDRESSES				*
		5085	*	* \$B\$EQU - COMPILER SYSTEM EQUATES				*
		5086	*	* \$I\$EQU - FIXED EQUATES				*
		5087	*	* \$I@SEQ/SPILE0 - STANDARD/LONG PRECISION EXECUTION EQUATES				*
		5088	*					*
		5089	*	* OTHER				*
		5090	*	* NONE.				*
		5091	*					*
		5092	*	*****				*

FBNPWR - POWER FUNCTION 3.7 BASIC INTERPRETER

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 37

```

5094 *****
5095 * FBNPWR - S/3.7 POWER FUNCTION ROUTINE - STANDARD AND LONG PRECISION *
5096 *****
5097 *
0800          5098      ORG      *,B@LVPG,0
                5099      USING *,@BR                      BASE REG POINTS HERE ON ENTRY
                5100 *
                5101 * EXECUTION ENTRY TO FBNPWR FOLLOWS
                5102 *
0800 35 02 0D4E 5103 FBNPWR L      I$STAK,@XR                      LOAD STACK POINTER
                5104 *
                5105 * CHECK FOR ZERO VALUES OF A AND/OR B IN A**B
                5106 * IF BOTH A AND B ARE ZERO, AN ERROR CONDITION HAS OCCURED SINCE 0**0
                5107 * IS INDETERMINATE. IF A IS ZERO AND B IS POSITIVE, LEAVE THE RESULT
                5108 * = 0. IF A IS ZERO AND B IS NEGATIVE, THE ERROR CONDITION DIVISION
                5109 * BY ZERO HAS OCCURED. IF B IS MA THEN THE RESULT IS 1, FOR ANY
                5110 * NON-ZERO A.
                5111 *
0804 BD F0 01   5112      CLI      I@1SE1+I@MANL(,@XR),@DZERO  IS A = 0 ?
0807 D0 01 1D   5113      BNE      FNB010(,@BR)                  BRANCH IF NOT
                5114 *
                5115 * A = 0
                5116 *
080A BD F0 09   5117      CLI      I@1SE2+I@MANL(,@XR),@DZERO  IS B = 0 ?
080D D0 81 59   5118      BE       FNB275(,@BR)                  YES BRANCH TO SET ERROR FLAG
                5119 *
                5120 * A = 0, B NOT = 0
                5121 *
0810 B9 20 0F   5122 FNB005 TBF      I@1SE2+I@MANR(,@XR),X'20' IS B NEGATIVE ?
0813 D0 90 DC   5123      BF       FNB900(,@BR)                  NO, LEAVE RESULT AS A = 0
0816 3C EC 0CBC 5124      MVI      I$ERRC,@@E790                YES, SET DIVIDE BY 0 ERROR FLAG
081A D0 87 DC   5125      B        FNB900(,@BR)                  RETURN TO CALLING PROGRAM
                5126 *
                5127 * A NOT = 0
                5128 *
081D BD F0 09   5129 FNB010 CLI      I@1SE2+I@MANL(,@XR),@DZERO  IS B = 0 ?
0820 D0 01 2E   5130      BNE      FNB030(,@BR)                  BRANCH, NEITHER A NOR B = 0
0823 A7 06 07 07 5131      SZ       I@RSE1(I@PREC,@XR),I@RSE1(I@PREC,@XR) MANTISSA RESULT = 0
0827 9C 01 01 EE 5132      MVC      I@1SE1+I@MANL(FNBMK1,@XR),FNBFP1(,@BR) RESULT = FLT PT 1
082B D0 87 DC   5133      B        FNB900(,@BR)                  RETURN TO CALLING PROGRAM
                5134 *
                5135 * BOTH A AND B ARE NON-ZERO
                5136 *
                5137 * DETERMINE IF B IS AN INTEGER BY CHECKING IF THE FLOATING POINT
                5138 * EXPONENT IS GREATER THAN OR EQUAL TO THE NUMBER OF DIGITS IN THE
                5139 * MANTISSA PRECEDING THE TRAILING ZEROES, IF ANY. THIS NUMBER IS
                5140 * EQUAL TO I@PREC - THE NUMBER OF TRAILING ZEROES.
                5141 *
082E 7C 00 E0   5142 FNB030 MVI      FNBCNT(,@BR),@ZERO          SET ZERO COUNT = 0
0831 B9 0F 0F   5143 FNB200 TBF      I@1SE2+I@MANR(,@XR),X'0F'    IS TRAILING DIGIT A ZERO ?
0834 D0 90 41   5144      BF       FNB250(,@BR)                  BRANCH WE HAVE REACHED NON-ZERO
0837 5E 00 E0 EC 5145      ALC      FNBCNT(1,@BR),FNBBN1(,@BR)    INCREMENT ZERO COUNT BY 1
083B 76 02 EB   5146      A        FNBMN1(,@BR),@XR              MOVE POINTER 1 BYTE LEFT
083E D0 87 31   5147      B        FNB200(,@BR)                  TEST NEXT DIGIT TO LEFT

0841 6C 00 E1 0F 5149 FNB250 MVC      FNBDGT(1,@BR),I@1SE2+I@MANR(,@XR) SAVE LAST NON - 0 DIGIT

```


ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 38

```

0845 35 02 0D4E      5150      L      I$STAK,@XR      RESTORE STACK POINTER
0849 6E 00 E0 08      5151      ALC      FNBCNT(1,@BR),I@1SE2(,@XR)  ADD EXP TO ZERO COUNT
084D 7D 87 E0         5152      CLI      FNBCNT(,@BR),I@PREC+B@NXZR  B IS AN INTEGER IF NOT LOW
0850 D0 02 60         5153      BNL      FNB300(,@BR)      BRANCH IF S IS AN INTEGER
                    5154 *
                    5155 *   IF A IS NEGATIVE THEN A**B IS IN GENERAL A COMPLEX VALUE, FOR NON-
                    5156 *   INTEGRAL B.  THIS IS AN ERROR CONDITION.
                    5157 *
0853 B8 F0 07         5158      TBN      I@1SE1+I@MANR(,@XR),B@ZPOS  IS A POSITIVE ?
0856 D0 10 A6         5159      BT       FNB500(,@BR)      YES, BRANCH TO EVALUATE A**B
0859 3C DF 0CBC       5160 FNB275 MVI      I$ERRC,@E774      EXPONENTIATION ERROR
085D D0 87 DC         5161      B       FNB900(,@BR)      RETURN TO CALLING PROGRAM
                    5162 *
                    5163 *   IF B IS A POSITIVE ONE DIGIT INTEGER, EVALUATE A**B BY MULTIPLICATION
                    5164 *   OR A BY ITSELF B TIMES.  OTHERWISE A**B = EXP(B*LOG(A)) FOR POSITIVE
                    5165 *   VALUES OF A.
                    5166 *
0860 BD 81 08         5167 FNB300 CLI      I@1SE2+I@DEXP(,@XR),B@NXZR+1  IS B ONE DIGIT INTEGER ?
0863 D0 01 8B         5168      BNE      FNB400(,@BR)      NO, BRANCH OVER DIRECT MULT
0866 B9 20 0F         5169      TBF      I@RSE2(,@XR),X'20'  IS B NEGATIVE 1 DIGIT INTEGER ?
0869 D0 10 8B         5170      BT       FNB400(,@BR)      YES, BRANCH OVER DIRECT MULT
                    5171 *
                    5172 *   B IS NOW A POSITIVE ONE DIGIT INTEGER
                    5173 *
086C 2C 07 060E 07    5174      MVC      I$FWRK+I@PREC(I@LUFV),I@RSE1(,@XR)  SAVE A IN INTERPRTR WA
0871 57 00 E1 EE      5175 FNB350 SZ      FNBDGT(1,@BR),FNBDCL(1,@BR)  REDUCE INTEGER B BY DECIMAL 1
0875 D0 81 DC         5176      BE       FNB900(,@BR)      IF ZERO, RETURN AS RTN IS DONE
0878 8C 07 0F 060E    5177      MVC      I@RSE2(I@LUFV,@XR),I$FWRK+I@PREC  NO, SET A
087D C0 87 082A       5178      B       I$FMPY      MULTIPLY ACCUMULATED RESULT * A
0881 3D 00 0CBC       5179      CLI      I$ERRC,I@NERR      DID OVER- OR UNDERFLOW OCCUR
0885 D0 01 DC         5180      BNE      FNB900(,@BR)      YES, RETURN TO CALLING PROGRAM
0888 D0 87 71         5181      B       FNB350(,@BR)      REPEAT
                    5182 *
                    5183 *   IF A IS NEGATIVE AND B IS AN INTEGER, THE MINUS SIGN FOR A IN
                    5184 *   A**B = EXP(B*LOG(A)) WHICH WILL CAUSE AN ERROR IN THE LOG ROUTINE
                    5185 *   IS AVOIDED BY USING A**B = (-1)**8 * EXP(B*LOG(-A)), WHERE NOW -A
                    5186 *   IS POSITIVE.  THEN THE SIGN OF THE RESULT IS POSITIVE IF B IS AN
                    5187 *   EVEN INTEGER AND NEGATIVE IF B IS AN ODD INTEGER.
                    5188 *
088B B9 20 07         5189 FNB400 TBF      I@1SE1+I@MANR(,@XR),X'20'  IS A NEGATIVE ?
088E D0 90 A6         5190      BF       FNB500(,@BR)      NO, SKIP SIGN MANIPULATION
0891 BA F0 07         5191      SBN      I@1SE1+I@MANR(,@XR),B@ZPOS  SET SIGN OF A TO POSITIVE
0894 79 01 E1         5192      TBF      FNBDGT(,@BR),X'01'  IS RIGHTMOST BIT OFF ?
0897 D0 10 A6         5193      BT       FNB500(,@BR)      YES, INTEGER IS EVEN
                    5194 *
                    5195 *   INTEGER B IS STILL EVEN IF LAST NON-ZERO DIGIT IS ODD PROVIDED THAT
                    5196 *   FLOATING POINT EXPONENT IS STRICTLY GREATER THAN THE NUMBER OR DIGITS
                    5197 *   IN NON-ZERO PART OF MANTISSA (PART PRECEDING TRAILING ZEROES).  THIS
                    5198 *   OCCURS IF EXP+ZERO COUNT (WHICH IS NOW IN FNBCNT) IS GREATER THAN
                    5199 *   I@REC + B@NXZR.
                    5200 *
089A 7D 87 E0         5201      CLI      FNBCNT(,@BR),I@PREC+B@NXZR  IS B AN EVEN INTEGER ?
089D D0 84 A6         5202      BH       FNB500(,@BR)      YES, BRANCH
08A0 7C 80 D7         5203      MVI      FNB880+@Q(,@BR),@NOP  SET CONDITIONAL BRANCH TO A NOP
                    5204 *
                    5205 *   THE BRANCH TO I$RTRN AT FNB880 WILL NOT TAKE PLACE SO THAT THE SIGN

```

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 39
					5206	*	OR THE RESULT IS MADE MINUS, IF B IS AN ODD INTEGER.	
					5207	*		
08A3	D0	87	A9		5208	B	FNB800(,@BR)	COMPUTE A..8
08A6	7C	87	D7		5209	FNB500 MVI	FNB880+@Q(,@BR),@UCB	MAKE FNB880 AN UNCOND BRANCH
08A9	6C	07	E9 0F		5210	FNB800 MVC	FNBSTR(I@LUFV,@BR),I@RSE2(,@XR)	SAVE B
					5211	*		
					5212	*	CALL THE LOG ROUTINE TO COMPUTE LOG(A) BASE E	
					5213	*		
08AD	C0	87	12B1		5214	B	I\$CALL	COMPUTE
08B1	0219			08B2	5215	DC	AL(@VADDR)(V\$FLOG)	* LOG(A) BASE E
					5216	*		
					5217	*	NO ERROR CAN OCCUR IN THE LOG ROUTINE SINCE A IS NEITHER ZERO NOR	
					5218	*	NEGATIVE, SO I\$ERRC IS NOT CHECKED	
					5219	*		
08B3	9C	07	0F E9		5220	MVC	I@RSE2(I@LUFV,@XR),FNBSTR(,@BR)	RESTORE B
					5221	*		
					5222	*	COMPUTE B*LOG(A)	
					5223	*		
08B7	C0	87	082A		5224	B	I\$FMPY	COMPUTE B*LOG(A)
08BB	3D	00	0CBC		5225	CLI	I\$ERRC,I@NERR	DID OVER- OR UNDERFLOW OCCUR ?
08BF	D0	01	DC		5226	BNE	FNB900(,@BR)	YES, RETURN TO CALLING PROGRAM
					5227	*		
					5228	*	COMPUTE A**B NOW, BY CALLING THE EYPONENTIAL ROUTINE TO COMPUTE	
					5229	*	EXP(B*LOG(A))	
					5230	*		
08C2	C0	87	12B1		5231	B	I\$CALL	COMPUTE
08C6	0500			08C7	5232	DC	AL(@VADDR)(V\$FEXP)	* EXP(B*LOG(A))
08C8	3D	00	0CBC		5233	CLI	I\$ERRC,I@NERR	DID OVERFLOW OCCUR
08CC	D0	81	D6		5234	BE	FNB880(,@BR)	NO, CONTINUE
08CF	3C	ED	0CBC		5235	MVI	I\$ERRC,@@E791	SET OVERFLOW FLAG
08D3	D0	87	DC		5236	B	FNB900(,@BR)	RETURN
					5237	*		
					5238	*	BRANCH IF NO SIGN CHANGE NEEDED	
					5239	*		
08D6	D0	00	DC		5240	FNB880 BC	FNB900(,@BR),*-*	RETURN IF B WAS NOT ODD
08D9	BB	20	07		5241	SBF	I@1SE1+I@MANR(,@XR),X'20'	MAKE SIGN OF RESULT MINUS
08DC	C0	87	12D3		5242	FNB900 B	I\$RTRN	RETURN TO CALLING PROGRAM
					5243	*		
					5244	*	NON-STORAGE CONSTANTS	
					5245	*		
				0002	5246	FNBK1 EQU	2	LENGTH TO MAKE ANSWER OF 1
					5247	*		
					5248	*	WORK AREA	
					5249	*		
08E0				08E0	5250	FNBCNT DS	CL1	ZERO COUNT
08E1				08E1	5251	FNBDGT DS	CL1	TEMPORARY STORAGE
08E2				08E9	5252	FNBSTR DS	CL(I@LUFV)	TEMPORARY STORAGE -- SAVE B
					5253	*		
					5254	*	CONSTANTS	
					5255	*		
08EA	FFFF			08EB	5256	FNBMN1 DC	XL(@REGL)'FFFF'	2 BYTE MINUS I FOR XR DECREMENT
08EC	01			08EC	5257	FNBBN1 DC	AL1(@B1)	BINARY 1
08ED	81			08ED	5258	DC	AL1(B@NXZR+1)	EXPONENT FOR FLOATING POINT
08EE	F1			08EE	5259	FNBFP1 DC	XL1'F1'	* DECIMAL 1
				08EE	5260	FNBDC1 EQU	*-1	DECIMAL 1
					5261	*		

5262 * END OF FNBPWR CODING
5263 *

FRBSQR - SQUARE ROOT FUNCTION

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 41
		5265		*****			
		5266	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
		5267	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
		5268	*				*
		5269		*****			*
		5270		*STATUS			*
		5271	*	VERSION 1 MODIFICATION 0			*
		5272	*				*
		5273		*FUNCTION			*
		5274	*	* FRBSQR FINDS THE SQUARE ROOT OF THE INPUT ARGUMENT, IN BOTH			*
		5275	*	STANDARD AND LONG PRECISION.			*
		5276	*	* FRBSQR CALCULATES BY MEANS OF AN ITERATIVE ALGORITHM.			*
		5277	*				*
		5278		*ENTRY POINTS			*
		5279	*	THE ENTRY IS FRBSQR. THE FORMAT OF THE CALLING SEQUENCE IS AS			*
		5280	*	FOLLOWS:			*
		5281	*	B I\$CALL			*
		5282	*	DC AL2(V\$FSQR)			*
		5283	*				*
		5284		*INPUT			*
		5285	*	* THE INPUT IS AN UNPACKED FLOATING POINT NUMBER IN THE FIRST			*
		5286	*	ELEMENT OF THE INTERPRETER STACK.			*
		5287	*	* REGISTER 1 (@BR) POINTS TO THE BEGINNING OF THE PAGE.			*
		5288	*				*
		5289		*OUTPUT			*
		5290	*	* THE RESULT IS LEFT IN THE FIRST STACK ELEMENT, IN UNPACKED			*
		5291	*	FLOATING POINT FORMAT.			*
		5292	*	* IN THE EVENT OF AN ERROR, AN ERROR CODE IS LEFT IN THE ONE-BYTE			*
		5293	*	INTERPRETER ERROR AREA, I\$ERRC.			*
		5294	*				*
		5295		*EXTERNAL REFERENCES			*
		5296	*	INTERPRETER STACK - FIRST 17 BYTES IN STANDARD PRECISION			*
		5297	*	- FIRST 32 BYTES IN LONG PRECISION			*
		5298	*	I\$STAK - CORE LOCATION OF ADDRESS OF THE INTERPRETER STACK			*
		5299	*	I\$ERRC - INTERPRETER ONE-BYTE ERROR AREA			*
		5300	*	I\$RTRN - ENTRY TO THE PAGING MODULE TO RETURN			*
		5301	*				*
		5302		*EXITS, NORMAL			*
		5303	*	* EXIT IS MADE BY BRANCHING TO I\$RTRN, THE ENTRY TO THE PAGING			*
		5304	*	RETURN ROUTINE.			*
		5305	*	* THE RESULT IS LEFT IN THE FIRST STACK ELEMENT.			*
		5306	*				*
		5307		*EXITS, ERROR			*
		5308	*	* AN ERROR CODE IS PLACED IN THE ERROR AREA, I\$ERRC.			*
		5309	*	* THE INPUTTED ARGUMENT IS RETURNED.			*
		5310	*	* EXIT IS MADE BY BRANCHING TO I\$RTRN.			*
		5311	*				*
		5312		*TABLES/WORK AREAS			*
		5313	*	THE CONSTANTS AND WORK AREAS RESIDE AT THE END OF THE EXECUTABLE			*
		5314	*				*
		5315		*ATTRIBUTES			*
		5316	*	REUSABLE, NATURALLY RELOCATABLE.			*
		5317	*				*
		5318		*CHARACTER CODE DEPENDENCY			*
		5319	*	THE LOW ORDER FOUR BITS OF DECIMAL NUMBERS, WHEN CONSIDERED AS A			*
		5320	*	BINARY INTEGER, IDENTIFY THE VALUE OF THE DIGIT.			*

FRBSQR - SQUARE ROOT FUNCTION

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE	42
		5321	*					*
		5322	*	*NOTES				*
		5323	*	ERROR PROCEDURES				*
		5324	*	THE ERROR CODE IS SET AND CONTROL RETURNED TO THE CALLING				*
		5325	*	ROUTINE.				*
		5326	*					*
		5327	*	REGISTER USAGE				*
		5328	*	* REGISTER 1 (@BR) IS USED AS THE BASE REGISTER.				*
		5329	*	* REGISTER 2 (@XR) IS USED TO INDEX THE INTERPRETER STACK.				*
		5330	*					*
		5331	*	SAVED/RESTORED AREAS				*
		5332	*	NONE.				*
		5333	*					*
		5334	*	MODIFICATION CONSIDERATIONS				*
		5335	*	THE ZONED DECIMAL INSTRUCTIONS HANDLE THE LARGEST FIELDS				*
		5336	*	POSSIBLE IN LONG PRECISION. THEREFORE NO GREATER ACCURACY				*
		5337	*	OR SIGNIFICANCE CAN BE HANDLED.				*
		5338	*					*
		5339	*	REQUIRED MODULES				*
		5340	*	@SYSEQ - COMMON SYSTEM EQUATES				*
		5341	*	@ERMEQ - ERROR MESSAGE EQUATES				*
		5342	*	\$V\$EQU - FIXED VIRTUAL ADDRESSES				*
		5343	*	\$B@EQU - COMPILER SYSTEM EQUATES				*
		5344	*	\$I\$EQU - FIXED EQUATES				*
		5345	*	\$I\$SEQ/\$I\$LEQ - STANDARD/LONG PRECISION EXECUTION EQUATES				*
		5346	*					*
		5347	*	OTHER				*
		5348	*	NONE.				*
		5349	*					*
		5350	*	*****				*

FRBSQR - SQUARE ROOT FUNCTION

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 43

```

5352 *****
5353 * FRBSQR - S/3.7 SQUARE ROOT FUNCTION ROUTINE *
5354 *****
0900 5355 *
5356 ORG *,B@LVP,0
0900 5357 USING *,@BR BASE REG POINTS HERE AT ENTRY
5358 *
5359 * EXECUTION ENTRY TO FRBSQR FOLLOWS
5360 *
0900 35 02 0D4E 5361 FRBSQR L I$STAK,@XR LOAD STACK POINTER
0904 B8 F0 07 5362 TBN I@1SE1+I@SIGN(,@XR),B@ZPOS IS ARGUMENT POSITIVE ?
0907 D0 10 11 5363 BT FRB005(,@BR) BRANCH IF YES 1-3
090A 3C E0 0CBC 5364 MVI I$ERRC,@E775 SQUARE ROOT OF NEG NUMBER ERROR
090E D0 87 97 5365 B FRB900(,@BR) RETURN TO CALLING PROGRAM
0911 BD F0 01 5366 FRB005 CLI I@1SE1+I@MANL(,@XR),@DZERO IS ARGUMENT = 0 1-3
0914 D0 81 97 5367 BE FRB900(,@BR) IF ZERO, RETURN 1-3
5368 *
5369 * IF FLOATING POINT EXPONENT OF INPUT AGS IS ODD. INCREMENT IT BY 1 AND
5370 * APPLY SQUARE ROOT ALGORITHM TO 10*MANTISSA.
5371 * IF FLOATING POINT EXPONENT OF INPUT ARG IS EVEN, DO NOT CHANGE IT AND
5372 * APPLY SQUARE ROOT ALGORITHM TO 100*MANTISSA.
5373 *
0917 B8 01 00 5374 FRB010 TBN I@1SE1+I@DEXP(,@XR),X'01' IS LAST BIT OF EXP ON ?
091A D0 10 27 5375 BT FRB020(,@BR) BRANCH AS EXP IS ODD
5376 *
5377 * EXP IS EVEN
5378 *
091D AC 06 0E 07 5379 MVC I@RSE2-FRBEVN(I@PREC,@XR),I@RSE1(,@XR) MOVE 100*MANTISSA
0921 BC F0 0F 5380 MVI I@RSE2(,@XR),@DZERO INSERT TRAILING ZERO
0924 D0 87 32 5381 B FRB030(,@BR) NOW DIVIDE EXP BY 2
5382 *
5383 * EXP IS ODD
5384 *
0927 9E 00 00 B4 5385 FRB020 ALC I@1SE1(1,@XR),FRBBN1(,@BR) INCREMENT EXP BY 1
092B AC 06 0F 07 5386 MVC I@RSE2(I@PREC,@XR),I@RSE1(,@XR) MOVE 10*MANTISSA IN
092F BC F0 08 5387 MVI I@1SE2(,@XR),@DZERO INSERT LEADING ZERO
5388 *
5389 * DIVIDE EXPONENT BY 2 TO GET FLOATING POINT EXPONENT FOR THE RESULT.
5390 *
0932 AC 00 01 00 5391 FRB030 MVC I@1SE1+FRBONE(1,@XR),I@1SE1+I@DEXP(,@XR) SHIFT EXP NXT BY
0936 BC 00 00 5392 MVI I@1SE1+I@DEXP(,@XR),@ZERO PUT 0 IN FRONT BYTE
0939 AE 01 01 01 5393 ALC I@1SE1+1(FRBEXP,@XR),I@1SE1+1(,@XR) ADD 2 BYTE EXP TO SELF
093D AE 01 01 01 5394 ALC I@1SE1+1(FRBEXP,@XR),I@1SE1+1(,@XR) ADD 2 BYTE EXP TO SELF
0941 AE 01 01 01 5395 ALC I@1SE1+1(FRBEXP,@XR),I@1SE1+1(,@XR) ADD 2 BYTE EXP TO SELF
0945 A8 01 00 00 5396 MZN I@1SE1+I@DEXP(,@XR),I@1SE1(,@XR) SHIFT DIVIDED
0949 A8 02 00 01 5397 MNZ I@1SE1+I@DEXP(,@XR),I@1SE1+1(,@XR) * EXP TO EXP BYTE
094D 9E 00 00 B6 5398 ALC I@1SE1+I@DEXP(1,@XR),FRBNRM(,@BR) RENORMALIZE EXPONENT
5399 *
5400 * CLEAR WORK AREAS AND INITIALIZE SUBTRAHEND TO 1000--- AND CONSTANTS
5401 * C1 AND C2 TO 20000--- AND 90000--- RESPECTIVELY. BOTH C1 AND
5402 * C2 WLL BE DIVIDED BY 10 (BY RIGHT SHIFT) EACH ITERATION. THE
5403 * SUBTRAHEND WILL BE MODIFIED BY C1 AND C2 DURING THE COURSE OF THE
5404 * ALGORITHM. THE RESULT IS BUILT UP DIRECTLY IN THE STACK, ONE DIGIT
5405 * PER ITERATION.
5406 *
0951 7C F0 B3 5407 MVI FRBFC2(,@BR),@DZERO GET DECIMAL ZERO

```

FRBSQR - SQUARE ROOT FUNCTION

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 44

```

0954 5C 17 B2 B3          5408      MVC      FRBFC2-1(I@LUFV+I@LUFV+I@LUFV,@BR),FRBFC2(,@BR)  CLEAR REG
0958 9C 06 07 B3          5409      MVC      I@RSE1(I@PREC,@XR),FRBFC2(,@BR)  CLEAR ACCUMULATOR TO 0'S
095C 9C 07 17 B3          5410      MVC      I@RSE3(I@LUFV,@XR),FRBFC2(,@BR)  CLEAR OVERFLOW REG TO 0'S
0960 7C F1 9D             5411      MVI      FRBSUB+1-I@LSFV(,@BR),B@DEC1 INITIALIZE TO 1000--- (SUBTR)
0963 7C F2 A6             5412      MVI      FRBFC1+1-I@LSFV(,@BR),B@DEC2 INITIALIZE TO 2000--- (C1)
0966 7C F9 AE             5413      MVI      FRBFC2+2-I@LSFV(,@BR),B@DEC9 INITIALIZE TO 9000--- (C2)
                                5414      *
                                5415      * START ALGORITHM FOR MANTISSA
                                5416      *
                                5417      * SUBTRACT THE SUBTRANEND FROM THE MANTISSA.  EACH TIME THE RESULT IS
                                5418      * NON-NEGATIVE, INCREMENT THE DIGIT IN THE ACCUMULATOR BYTE FOR THIS
                                5419      * ITERATION BY DECIMAL 1 (IT IS INITIALLY ZERO), AND MODIFY THE
                                5420      * SUBTRAHEND BY ADDING C1 TO IT.
                                5421      *
0969 97 27 0F A3          5422 FRB100 SZ      I@RSE2+FRBLNG(I@LUFV+2,@XR),FRBSUB(I@LSFV+1,@BR)
096D D0 82 7B             5423          BM      FRB150(,@BR)                THERE IS A DIGIT IN ACCUMULATOR
0970 56 26 A3 AC           5424          AZ      FRBSUB(I@LUFV+1,@BR),FRBFC1(I@LSFV,@BR)  MODIFY
0974 96 00 01 B5           5425          AZ      I@1SE1+FRBACC(1,@XR),FRBDC1(1,@BR) INCREMENT ACCUM BY 1
0978 D0 87 69             5426          B      FRB100(,@BR)                REPEAT...
                                5427      *
                                5428      * AFTER PREC ITERATIONS WE NAVE A PREC DIGIT MANTISSA IN THE STACK.
                                5429      * THE CORRECT EXPONENT IS ALREADY IN PLACE, SO RETURN TO CALLING PROG.
                                5430      *
097B 7D F9 B3             5431 FRB150 CLI      FRBFC2(,@BR),B@DEC9          HAVE I@PREC SHIFTS OCCUR
097E D0 81 93             5432 FRB400 BE      FRB850(,@BR)          YES, RETURN TO CALLING PROGRAM
                                5433      *
                                5434      * AT THE CONCLUSION OF AN ITERATION, AFTER RESTORING THE (MODIFIED)
                                5435      * INPUT MANTISSA TO ITS LAST NON-NEGATIVE VALUE, ADD C2 TO THE
                                5436      * SUBTRAHEND AND THEN DIVIDE BOTH C1 AND C2 BY 10 BY A SINGLE RIGHT
                                5437      * SHIFT OF 1.
                                5438      *
0981 96 27 0F A3          5439          AZ      I@RSE2+FRBLNG(I@LUFV+2,@XR),FRBSUB(I@LSFV+1,@BR) RESTORE
0985 57 25 A3 B3          5440          SZ      FRBSUB(I@LSFV+1,@BR),FRBFC2(I@LSFV-1,@BR)  MODIFY
0989 5C 0E B3 B2          5441          MVC      FRBFC2(I@LUFV+I@PREC,@BR),FRBFC2-1(,@BR)  SHIFT RIGHT 1
098D E2 02 01             5442          LA      1(,@XR),@XR                INCREMENT ACCUMULATOR POINTER
0990 D0 87 69             5443          B      FRB100(,@BR)                GET NEXT DIGIT OF RESULT
                                5444      *
                                5445      * RESTORE THE INDEX REGISTER AND RETURN
                                5446      *
0993 35 02 0D4E           5447 FRB850 L      I$STAK,@XR                RESTORE INDEX REG
0997 C0 87 12D3           5448 FRB900 B      I$RTRN                RETURN TO CALLER
                                5449      *
                                5450      * NON-STORAGE CONSTANTS
                                5451      *
                                0001 5452 FRBONE EQU      1                ONE
                                0001 5453 FRBEVN EQU      1                LENGTH OR 1
                                0001 5454 FRBACC EQU      1                LENGTH OR 1
                                0002 5455 FRBTWO EQU      2                LENGTH OR 2
                                0002 5456 FRBEXP EQU      2                LENGTH OF EX WORKING AREA
                                0000 5457 FRBLNG EQU      I@LSFV-I@PREC    1 FOR STD PREC, 0 FOR LONG PREC
                                5458      *
                                5459      * WORK AREA
                                5460      *
099B                      09A3 5461 FRBSUB DS      CL(I@LUFV+1)          SUBTRAHEND PLUS EXTRA BYTE
09A4                      09AC 5462 FRBFC1 DS      CL(I@LUFV+1)          CONSTANT C1 PLUS EXTRA BYTE
                                5463      *

```

FRBSQR - SQUARE ROOT FUNCTION

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE	45
					5464	*	THE EXTRA BYTE, WHICH IS CLEARED TO DEC ZERO BEFORE STARTING THE				
					5465	*	COMPUTATION, IS NECESSARY TO AVOID PROPAGATING LEADING NON-ZEROES				
					5466	*	DURING THE SHIFTING OF C1 AND C2 TO THE RIGHT.				
					5467	*					
	09AD			09B3	5468	FRBFC2 DS	CL(I@PREC)			CONSTANT C2	
					5469	*					
					5470	*	CONSTANTS				
					5471	*					
	09B4	01		09B4	5472	FRBBN1 DC	AL1(@B1)			BINARY	
	09B5	F1		09B5	5473	FRBDC1 DC	DL1'1'			DECIMAL 1	
	09B6	40		09B6	5474	FRBNRM DC	XL1'40'			FOR RENORMALIZING EXP	
					5475	*					
					5476	*	END OF FRBSQR CODING				
					5477	*					

FSSSIN - SINE/COSINE FUNCTION - STANDARD PREC

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 46
		5479		*****			
		5480	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
		5481	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
		5482	*				*
		5483	*	*****			*
		5484	*	*STATUS			*
		5485	*	VERSION 1 MODIFICATION 0			*
		5486	*				*
		5487	*	*FUNCTION			*
		5488	*	* THE FSSSIN MODULE CALCULATES THE SINE OR COSINE IN STANDARD			*
		5489	*	PRECISION OF THE INPUT ARGUMENT.			*
		5490	*	* THE INPUT ARGUMENT MUST BE GREATER THAN -100 AND LESS THAN 100.			*
		5491	*	* THE ANSWER IS CALCULATED BY MEANS OF A CHEBYSHEV RATIONAL			*
		5492	*	POLYNOMIAL ROUTINE.			*
		5493	*				*
		5494	*	*ENTRY POINTS			*
		5495	*	* THE ENTRY TO COMPUTE THE SINE IS FSSSIN. THE FORMAT OF THE			*
		5496	*	CALLING SEQUENCE IS AS FOLLOWS:			*
		5497	*	B I\$CALL			*
		5498	*	DC AL2(V\$FSIN)			*
		5499	*	* THE ENTRY TO COMPUTE THE COSINE IS FSSCOS. THE FORMAT OR THE			*
		5500	*	CALLING SEQUENCE IS AS FOLLOWS:			*
		5501	*	B I\$CALL			*
		5502	*	DC AL2(V\$FCOS)			*
		5503	*				*
		5504	*	*INPUT			*
		5505	*	* THE INPUT IS AN UNPACKED FLOATING POINT NUMBER IN THE FIRST			*
		5506	*	INTERPRETER STACK ELEMENT.			*
		5507	*	* REGISTER 1 (@BR) POINTS TO THE PAGE BEGINNING.			*
		5508	*				*
		5509	*	*OUTPUT			*
		5510	*	* THE RESULT IS PLACED IN THE FIRST STACK ELEMENT IN UNPACKED			*
		5511	*	FLOATING POINT FORMAT.			*
		5512	*	* IN THE EVENT OF AN ERROR, THE ERROR CODE IS PLACED IN THE			*
		5513	*	ONE-BYTE INTERPRETER ERROR AREA, I\$ERRC.			*
		5514	*				*
		5515	*	*EXTERNAL REFERENCES			*
		5516	*	INTERPRETER STACK - FIRST TWO ELEMENTS PLUS 1 BYTE (17 BYTES)			*
		5517	*	I\$STAK - CORE LOCATION OF ADDRESS OF THE INTERPRETER STACK			*
		5518	*	I\$ERRC - INTERPRETER ONE-BYTE ERROR AREA			*
		5519	*	I\$CALL - ENTRY TO THE PAGING ROUTINE			*
		5520	*	I\$RTRN - ENTRY TO THE PAGING ROUTINE RETURN			*
		5521	*	I\$FADD - ENTRY TO ADD ROUTINE			*
		5522	*	I\$FMPY - ENTRY TO MULTIPLY ROUTINE			*
		5523	*	CBFPZD - CONVERT FLOATING POINT TO ZONED DECIMAL			*
		5524	*	CCZDFP - CONVERT ZONED DECIMAL TO FLOATING POINT			*
		5525	*	FDIADD - ADD ROUTINE			*
		5526	*	FZIMPY - MULTIPLY ROUTINE			*
		5527	*				*
		5528	*	*EXITS, NORMAL			*
		5529	*	* EXIT IS BY BRANCHING TO I\$RTRN, THE ENTRY TO THE PAGING RETURN			*
		5530	*	ROUTINE.			*
		5531	*	* THE RESULT IS LEFT IN THE FIRST STACK ELEMENT.			*
		5532	*				*
		5533	*	*EXITS, ERROR			*
		5534	*	* AN ERROR CODE IS PLACED IN THE ERROR AREA, I\$ERRC.			*

FSSSIN - SINE/COSINE FUNCTION - STANDARD PREC

ERR LOC	OBJECT CODE	ADDR STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 47
		5535 *	* EXIT IS BY BRANCHING TO I\$RTRN.			*
		5536 *				*
		5537 *	*TABLES/WORK AREAS			*
		5538 *	THE CONSTANTS AND WORK AREAS RESIDE AT THE END OF THE EXECUTABLE			*
		5539 *	CODE.			*
		5540 *				*
		5541 *	*ATTRIBUTES			*
		5542 *	REUSABLE, NATURALLY RELOCATABLE.			*
		5543 *				*
		5544 *	*CHARACTER CODE DEPENDENCY			*
		5545 *	THE OPERATION OF THIS MODULE DEPENDS UPON A ZONED DECIMAL DIGIT			*
		5546 *	BEING REPRESENTED WITH THE ZONE (FIRST FOUR BITS) BEING AN 'F'			*
		5547 *	FOR POSITIVE, AND A 'D' FOR NEGATIVE. THIS PROPERTY IS USED			*
		5548 *	FOLLOWING FSSSIN.			*
		5549 *	THE LOW ORDER FOUR BITS OF DECIMAL NUMBERS, WHEN CONSIDERED AS A			*
		5550 *	BINARY INTEGER, IDENTIFY THE VALUE OF THE DIGIT.			*
		5551 *				*
		5552 *	*NOTES			*
		5553 *	ERROR PROCEDURES			*
		5554 *	THE ERROR CODE IS SET AND CONTROL RETURNED TO THE CALLING			*
		5555 *	PROGRAM.			*
		5556 *				*
		5557 *	REGISTER USAGE			*
		5558 *	* REGISTER 1 (@BR) IS USED AS THE BASE REGISTER.			*
		5559 *	* REGISTER 2 (@XR) IS USED TO REFERENCE THE STACK.			*
		5560 *				*
		5561 *	SAVED/RESTORED AREAS			*
		5562 *	NONE.			*
		5563 *				*
		5564 *	REQUIRED MODULES			*
		5565 *	@SYSEQ - COMMON SYSTEM EQUATES			*
		5566 *	@ERMEQ - ERROR MESSAGE EQUATES			*
		5567 *	\$V\$EQU - FIXED VIRTUAL ADDRESSES			*
		5568 *	\$B@EQU - COMPILER SYSTEM EQUATES			*
		5569 *	\$I\$EQU - FIXED EQUATES			*
		5570 *	\$I@SEQ - STANDARD PRECISION EXECUTION EQUATES			*
		5571 *				*
		5572 *	MODIFICATION CONSIDERATIONS			*
		5573 *	NONE.			*
		5574 *				*
		5575 *	OTHER			*
		5576 *	NONE.			*
		5577 *	*****			*

FSSSIN - SINE/COSINE FUNCTION - STANDARD PREC

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 48

```

5579 *****
5580 * FSSSIN - S/3.7 SINE AND COSINE FUNCTION ROUTINE *
5581 *****
5582 *
0A00 5583 ORG *,B@LVP,0
0A00 5584 USING FSSCOS,@BR BASE @BR POINTS HERE AT ENTRY
5585 *
5586 * EXECUTION ENTRY TO COSINE ROUTINE FOLLOWS.
5587 * THE ADDEND IN FSSOCT IS DECIMAL 2 FOR THE COSINE OF BOTH POSITIVE AND
5588 * NEGATIVE NUMBERS.
5589 *
0A00 35 02 0D4E 5590 FSSCOS L I$STAK,@XR LOAD STACK POINTER
0A04 7C F2 C0 5591 MVI FSSOCT(,@BR),B@DEC2 SET ADDEND TO DECIMAL 2
5592 *
5593 * IF ARGUMENT IS LESS THAN 1E(-PREC), THE ARG SHOULD BE SET TO ZERO.
5594 *
0A07 BD 79 00 5595 CLI I@1SE1+I@DEXP(,@XR),B@NXZR-I@PREC IS ARG < 1E(-PREC)
0A0A D0 84 14 5596 BH FSS050(,@BR) NO, BRANCH TO CONTINUE
0A0D A7 06 07 07 5597 SZ I@RSE1(I@PREC,@XR),I@RSE1(I@PREC,@XR) CLEAR TO ZEROES
0A11 BC 1E 00 5598 MVI I@1SE1+I@DEXP(,@XR),B@NXLO SET EXPONENT FOR ZERO
0A14 7C F2 3D 5599 FSS050 MVI FSS160+@Q(,@BR),@E796 ERROR IN COSINE ARG MESSAGE
0A17 D0 87 33 5600 B FSS100(,@BR) ENTER MAIN PROGRAM LOGIC
5601 *
5602 * EXECUTION ENTRY TO SINE ROUTINE FOLLOWS.
5603 * THE ADDEND IN FSSOCT IS DECIMAL ZERO FOR SINE OF POSITIVE NUMBER AND
5604 * DECIMAL 4 FOR SINE OF NEGATIVE NUMBER.
5605 *
0A1A 35 02 0D4E 5606 FSSSIN L I$STAK,@XR LOAD STACK POINTER
5607 *
5608 * IF THE ARGUMENT IS LESS THAN 1E(-PREC), THE SIN(ARG) = ARG.
5609 *
0A1E BD 79 00 5610 CLI I@1SE1+I@DEXP(,@XR),B@NXZR-I@PREC IS ARG < 1E(-PREC)
0A21 D0 82 B3 5611 BL FSS425(,@BR) IF YES, RETURN
0A24 7C F0 C0 5612 MVI FSSOCT(,@BR),B@DEC0 SET ADDEND TO DECIMAL ZERO
0A27 7C F1 3D 5613 MVI FSS160+@Q(,@BR),@E795 ERROR IN SINE ARG MESSAGE
0A2A B8 20 07 5614 TBN I@RSE1(,@XR),X'20' IS ARGUMENT NEGATIVE ?
0A2D D0 10 36 5615 BT FSS150(,@BR) IF NO, BRANCH
0A30 7C F4 C0 5616 MVI FSSOCT(,@BR),B@DEC4 SET ADDEND TO DECIMAL 4
0A33 BA F0 07 5617 FSS100 SBN I@RSE1(,@XR),B@ZPOS MAKE ARGUMENT POSITIVE
5618 *
5619 * IF THE ARGUMENT IS GREATER THAN ONE HUNDRED RADIANS, A LOSS OF MANY
5620 * SIGNIFICANT DIGITS WILL OCCUR DURING RANGE REDUCTION.
5621 *
0A36 BD 83 00 5622 FSS150 CLI I@1SE1+I@DEXP(,@XR),B@NXZR+3 IS ARG LESS THAN 100 ?
0A39 D0 82 43 5623 BL FSS200(,@BR) IF YES, REDUCE RANGE
0A3C 3C 00 0CBC 5624 FSS160 MVI I$ERRC,*-* SIN OR COS TRIG ARG TOO LARGE
0A40 D0 87 B3 5625 B FSS425(,@BR) RETURN TO CALLING PROGRAM
5626 *
5627 * MULTIPLY ARGUMENT BY 4/PI
5628 *
0A43 9C 07 0F C8 5629 FSS200 MVC I@RSE2(I@LUFV,@XR),FSSFP1(,@BR) SET 4/PI MULTIPLIER
0A47 C0 87 082A 5630 B I$FMPY MULTIPLY ARGUMENT BY 4/PI
5631 *
5632 * IF THE RESULT IS LESS THAN 1, SKIP THE RANGE REDUCTION.
5633 *
0A4B BD 80 00 5634 FSS205 CLI I@1SE1+I@DEXP(,@XR),B@NXZR COMPARE EXPONENT WITH ZERO

```

FSSSIN - SINE/COSINE FUNCTION - STANDARD PREC

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 49

```

0A4E D0 04 81          5635      BNH   FSS300(,@BR)          IF <= 0, SKIP RANGE REDUCTION
                        5636 *
                        5637 * BREAK ARGUMENT INTO INTEGER PART 1 AND FRACTIONAL PART F.
                        5638 *
0A51 C0 87 12B1        5639      B     I$CALL          CONVERT ARGUMENT
0A55 0C70              0A56 5640      DC     AL(@VADDR)(V$CFPZ)      * TO ZONED DECIMAL
                        5641 *
                        5642 * ADD ADDEND (DEFINED AT BEGINING OF PROGRAM) TO I AND COMPUTE I MOD 8
                        5643 * BY SUCCESSIVE COMPARISON WITH POWERS OF 8, STARTING WITH **2 = 64.
                        5644 * BUT FIRST CHECK TO SEE IF I IS ALREADY LESS THAN 1 (AS IT WILL BE IN
                        5645 * MOST CASES) TO AVOID THE RANGE REDUCTION.
                        5646 *
0A57 96 20 07 C0      5647      AZ     I@RSE1(FSSADD,@XR),FSSOCT(,@BR)  ADD ADDEND TO INTEGER PT
                        5648 *
                        5649 * REDUCE THE RANGE
                        5650 *
0A5B 9D 02 07 CB      5651 FSS225 CLC   I@RSE1(FSSINT,@XR),FSS064(,@BR) IS INTEGER I LESS THAN 64
0A5F D0 82 66          5652      BL     FSS230(,@BR)          BRANCH IF LESS THAN THIS POWER
0A62 97 02 07 CB      5653      SZ     I@RSE1(FSSINT,@XR),FSS064(FSSINT,@BR) SUBTRACT 8**2
0A66 9D 02 07 CE      5654 FSS230 CLC   I@RSE1(FSSINT,@XR),FSS008(,@BR) IS INTEGER I LESS THEN 8
0A6A D0 82 74          5655      BL     FSS260(,@BR)          YES, WE NOW HAVE I MOD 8
0A6D 97 02 07 CE      5656      SZ     I@RSE1(FSSINT,@XR),FSS008(FSSINT,@BR) SUBTRACT 8**1
0A71 D0 87 66          5657      B     FSS230(,@BR)          REPEAT WITH 8**1
0A74 6C 00 C0 07      5658 FSS260 MVC   FSSOCT(1,@BR),I@RSE1(,@XR)  STORE I MOD 8 AS NEW ADDED
                        5659 *
                        5660 * CONVERT F TO FLOATINS POINT NUMBER.  FIRST CLEAR INTEGER PART OF
                        5661 * ARGUMENT TO ZEROES.
                        5662 *
0A78 BC F0 07          5663      MVI   I@RSE1(,@XR),@DZERO      INTEGER PART TO ZERO
0A7B C0 87 12B1        5664      B     I$CALL          CONVERT F TO FLOATING POINT
0A7F 04AD              0A80 5665      DC     AL(@VADDR)(V$CZFP)
                        5666 *
                        5667 * DIVIDE F BY 2.  THIS IS DONE BY MULTIPLYING IT BY 0.5.
                        5668 *
0A81 9C 07 0F D6      5669 FSS300 MVC   I@RSE2(I@LUFV,@XR),FSSH1F(,@BR)  SET 0.5 MULTIPLIER
0A85 C0 87 082A        5670      B     I$FMPY          MULTIPLY F BY 0.5
                        5671 *
                        5672 * THE FINAL ARGUMENT TO BE USED BY THE CHEBYSHEV POLYNOMIAL ROUTINE
                        5673 * DEPENDS ON THE VALUE OF THE ADDEND (I MOD 8) AS FOLLOWS --
                        5674 *      *IF I MOD 8 IS 0, ARGUMENT = F
                        5675 *      *IF I MOD 8 IS 1, ARGUMENT = 0,5 + F
                        5676 *      *IF I MOD 8 IS 2, ARGUMENT = 1 - F
                        5677 *      *IF I MOD 8 IS 3, ARGUMENT = 0,5 - F
                        5678 *      *IF I MOD 8 IS 4, ARGUMENT = -F
                        5679 *      *IF I MOD 8 IS 5, ARGUMENT = -0,5 - F
                        5680 *      *IF I MOD 8 IS 6, ARGUMENT = -1 + F
                        5681 *      *IF I MOD 8 IG 7, ARGUMENT = -0,5 + F
                        5682 * THE FOLLOWING LOGIC SORTS OUT THESE CASES BY EXAMINING THE LOW-ORDER
                        5683 * 3 BITS OF THE ADDEND (I MOD 8),
                        5684 *
0A89 9C 07 0F D6      5685      MVC   I@RSE2(I@LUFV,@XR),FSSH1F(,@BR)  SET SECOND ELT TO 0.5
0A8D 78 01 C0          5686      TBN   FSSOCT(,@BR),X'01'      IS ADDEND 1, 3, 5 OR 7 ?
0A90 D0 10 B7          5687      BT     FSS450(,@BR)          BRANCH IF YES
0A93 78 02 C0          5688      TBN   FSSOCT(,@BR),X'02'      IS ADDEND 0 OR 4 ?
0A96 D0 90 A4          5689      BF     FSS380(,@BR)          BRANCH IF YES
0A99 9C 01 09 D8      5690      MVC   I@1SE2+FSSONE(FSSMOD,@XR),FSSMDY(,@BR)  SET B EQUAL TO 1

```

FSSSIN - SINE/COSINE FUNCTION - STANDARD PREC

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 50

0A9D	BB 20 07		5691	FSS360	SBF	I@RSE1(,@XR),X'20'	MAKE F MINUS
0AA0	C0 87 075D		5692	FSS370	B	I\$FADD	F = B + F
0AA4	78 04 C0		5693	FSS380	TBN	FSSOCT(,@BR),X'04'	IS ADDEND 4, 5, 6 OR 7 ?
0AA7	D0 90 AD		5694		BF	FSS400(,@BR)	NO, LEAVE F AS IS
0AAA	BB 20 07		5695		SBF	I@RSE1(,@XR),X'20'	MAKE F OR ARGUMENT MINUS
0AAD	C0 87 12B1		5696	FSS400	B	I\$CALL	COMPUTE CHEBYSHEV POLYNOMIAL
0AB1	0B00	0AB2	5697		DC	AL(@VADDR)(FSS900)	
0AB3	C0 87 12D3		5698	FSS425	B	I\$RTRN	RETURN TO CALLING PROGRAM
0AB7	78 02 C0		5699	FSS450	TBN	FSSOCT(,@BR),X'02'	IS ADDEND 3 OR 7 ?
0ABA	D0 10 9D		5700		BT	FSS360(,@BR)	YES, BRANCH
0ABD	D0 87 A0		5701		B	FSS370(,@BR)	NO, IT IS 1 OR 9 ?
			5702	*			
			5703	*		NON-STORAGE CONSTANTS FOR FSSSIN FOLLOW	
			5704	*			
		0001	5705	FSSONE	EQU	1	ONE
		0001	5706	FSSEQ8	EQU	1	
		0002	5707	FSSMOD	EQU	2	TO MODIFY B
		0003	5708	FSSADD	EQU	3	LENGTH OF ADDEND FOR OCTANT
		0003	5709	FSSINT	EQU	3	LENGTH OF INTEGER
			5710	*			
			5711	*		WORK AREA FOR FSSSIN FOLLOWS	
			5712	*			
0AC0		0AC0	5713	FSSOCT	DS	XL1	ADDEND FOR DETERMINING OCTANT
			5714	*			
			5715	*		CONSTANTS FOR FSSSIN FOLLOW	
			5716	*			
0AC1	81	0AC1	5717		DC	AL1(B@NXZR+1)	FLOATING
0AC2	F1F2F7F3F2F3F9	0AC8	5718	FSSFP1	DC	DL(I@PREC)'1273239'	* POINT 4/PI
0AC9	F0F6F4	0ACB	5719	FSS064	DC	DL(FSSINT)'64'	8**2
0ACC	F0F0F8	0ACE	5720	FSS008	DC	DL(FSSINT)'08'	8**1
0ACF	80	0ACF	5721		DC	AL1(B@NXZR)	FLOATING
0AD0	F5F0F0F0F0F0F0	0AD6	5722	FSSHLP	DC	DL(I@PREC)'5000000'	* POINT 1/2
0AD7	81	0AD7	5723		DC	AL1(B@NXZR+1)	EXPONENT AND FIRST BYTE OF
0AD8	F1	0AD8	5724	FSSMDY	DC	DL1'1'	* MANTISSA TO CONVERT 1/2 TO 1

FSSSIN - SINE/COSINE FUNCTION - STANDARD PREC

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 51
					5726	*****				
					5727	* SECOND PAGE OF FSSSIN				
					5728	*				
	0B00				5729	ORG	*,B@LVP,0			
				0B00	5730	USING	*,@BR			BASE REG POINTS AT THE PAGE
					5731	*				
					5732	* EXECUTION ENTRY TO CHEBYSHEV POLYNOMIAL ROUTINE FOLLOWS				
					5733	*				
	0B00	35	02	0D4E	5734	FSS900 L	I\$STAK,@XR			LOAD STACK POINTER
					5735	*				
					5736	* INITIALIZE THE DISPLACEMENT FSSCOF IN THE INSTRUCTION AT FSS920.				
					5737	* THIS DISPLACEMENT IS INCREMNTED THROUGH THE VHEBYSHEV COEFFICIENTS				
					5738	* DURING THE EXULTION OF THE SUBROUTINE.				
					5739	*				
	0B04	7C	04	5E	5740	MVI	FSSLOP(,@BR),X'4'			INITIALIZE LOOP COUNT
	0B07	6C	07	66 07	5741	MVC	FSSINP(I@LUFV,@BR),I@RSE1(,@XR)			SAVE INPUT ARGUMENT
	0B0B	AC	07	0F 07	5742	MVC	I@RSE2(I@LUFV,@XR),I@RSE1(,@XR)			ALSO MULTIPLY IT
	0B0F	C0	87	082A	5743	B	I\$FMPY			* AT ITSELF
	0B13	6C	07	55 07	5744	MVC	FSSSQD(I@LUFV,@BR),I@RSE1(,@XR)			AND STORE SQUARED VALUE
	0B17	5C	07	5D 70	5745	FSS905 MVC	FSSRST(I@LUFV,@BR),FSSCOF(,@BR)			INITIAL RESULT
	0B1B	5C	00	32 1A	5746	MVC	FSS920+@DD2(1,@BR),FSS905+@DD2(,@BR)			INITIAL INSTR FSS920
					5747	*				
					5748	* THE WORK AREA I\$FWRK CANNOT BE USED FOR THESE TEMPORARY RESULTS				
					5749	* BECAUSE THIS WORK AREA IS USED BY SUBROUTINE FDIADD				
					5750	*				
	0B1F	5E	00	32 67	5751	FSS910 ALC	FSS920+@DD2(1,@BR),FSSDCO(,@BR)			INCREMENT DISPLACEMENT
	0B23	9C	07	07 55	5752	MVC	I@RSE1(I@LUFV,@XR),FSSSQD(,@BR)			GET SQUARED ARGUMENT
	0B27	9C	07	0F 5D	5753	MVC	I@RSE2(I@LUFV,@XR),FSSRST(,@BR)			GET CURRENT RESULT AND
	0B2B	C0	87	082A	5754	B	I\$FMPY			MULTIPLY BY SQUARED ARGUMENT
					5755	*				
					5756	* INSTRUCTION FOLLOWING IS MODIFIED BY HAVING ITS SECOND DISPLACEMENT				
					5757	* INCREMENTED EACH ITERATION,				
					5758	*				
	0B2F	9C	07	0F 00	5759	FSS920 MVC	I@RSE2(I@LUFV,@XR),*-(,@BR)			GET NEXT COEFFICIENT
	0B33	C0	87	075D	5760	B	I\$FADD			ADD TO GET NEW RESULT
	0B37	6C	07	5D 07	5761	MVC	FSSRST(I@LUFV,@BR),I@RSE1(,@XR)			STORE CURRENT RESULT
	0B3B	5F	00	5E 68	5762	SLC	FSSLOP(,@BR),FSSMN1(,@BR)			DECREMENT COUNT BY 1
	0B3F	D0	84	1F	5763	BH	FSS910(,@BR)			NOT FINISHED, LOOP BACK
	0B42	9C	07	0F 66	5764	MVC	I@RSE2(I@LUFV,@XR),FSSINP(,@BR)			GET ORIGINAL ARGUMENT
	0B46	C0	87	082A	5765	B	I\$FMPY			DO FINAL MULTIPLICATION
	0B4A	C0	87	12D3	5766	B	I\$RTRN			RETURN TO CALLING PROGRAM
					5767	*				
					5768	* WORK AREA FOR FSSCHF FOLLOWS				
					5769	*				
	0B4E				0B55	5770 FSSSQD DS	XL(I@LUFV)			ROR STORING SQUARED ARGUMENT
	0B56				0B5D	5771 FSSRST DS	XL(I@LUFV)			FOR IMMEDIATE RESULT
	0B5E				0B5E	5772 FSSLOP DS	XL1			LOOP COUNT
	0B5F				0B66	5773 FSSINP DS	XL(I@LUFV)			FOR SAVING INPUT ARGUMENT
					5774	*				
					5775	* CONSTANTS FOR FSSCHF FOLLOW				
					5776	*				
	0B67	08			0B67	5777 FSSDCO DC	AL1(I@PREC+1)			INCR FOR DISPLACEMNT TO COEF
	0B68	01			0B68	5778 FSSMN1 DC	AL1(@B1)			DECREMENT FOR LOOP COUNT
					5779	*				
					5780	* CHEBYSHEV COEFFICIENTS				
					5781	*				

FSSSIN - SINE/COSINE FUNCTION - STANDARD PREC

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE	52
	0B69	7D		0B69	5782		DC AL1(B@NXZR-3)				
	0B6A	F1F5F1F4F8F4F2		0B70	5783	FSSCOF	DC DL(I@PREC)'1514842'				
	0B71	7E		0B71	5784		DC AL1(B@NXZR-2)				
	0B72	F4F6F7F3F7F6		0B77	5785		DC DL(I@PREC-1)'467376'				
	0B78	D6		0B78	5786		DC XL1'D6'				
	0B79	7F		0B79	5787		DC AL1(B@NXZR-1)				
	0B7A	F7F9F6F8F9F7F0		0B80	5788		DC DL(I@PREC)'7968970'				
	0B81	80		0B81	5789		DC AL1(B@NXZR)				
	0B82	F6F4F5F9F6F3		0B87	5790		DC DL(I@PREC-1)'645963'				
	0B88	D7		0B88	5791		DC XL1'D7'				
	0B89	81		0B89	5792		DC AL1(B@NXZR+1)				
	0B8A	F1F5F7F0F7F9F6		0B90	5793		DC DL(I@PREC)'1570796'				
					5794	*					
					5795	*	END OF FSSSIN CODING				
					5796	*					
	0C00				5797	ORG	*,B@LVPG,0			START FSSSIN DUMMY 3RD PAGE	

CBFPZD - PROLOGUE - FLOATING POINT TO ZONED DECIMAL

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 53
		5799		*****			
		5800	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
		5801	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
		5802	*				*
		5803	*	*****			*
		5804	*	*STATUS			*
		5805	*	VERSION 1 MODIFICATION 0			*
		5806	*				*
		5807	*	*FUNCTION			*
		5808	*	* CBFPZD CONVERTS AN UNPACKED FLOATING POINT NUMBER TO A ZONED			*
		5809	*	DECIMAL.			*
		5810	*	* THE INPUT ARGUMENT MUST BE LESS THAN 10**7(15) OR ELSE PARTS			*
		5811	*	OF CORE WILL BE DESTROYED.			*
		5812	*				*
		5813	*	*ENTRY POINTS			*
		5814	*	* THE ENTRY IS CBFPZD. THE FORMAT OR THE CALLING SEQUENCE IS AS			*
		5815	*	FOLLOWS:			*
		5816	*	B I\$CALL			*
		5817	*	DC AL2(V\$CFPZ)			*
		5818	*				*
		5819	*	*INPUT			*
		5820	*	* THE INPUT IS AN UNPACKED FLOATING POINT NUMBER IN THE FIRST			*
		5821	*	INTERPRETER STACK ELEMENT.			*
		5822	*	* REGISTER 1 (@BR) POINTS TO THE BEGINNING OR THE PAGE.			*
		5823	*	* I\$STAK CONTAINS THE ADDRESS OR THE INTERPRETER STACK.			*
		5824	*				*
		5825	*	*OUTPUT			*
		5826	*	THE RESULT IS LEFT IN THE FIRST TWO INTERPRETER STACK ELEMENTS.			*
		5827	*	THE DECIMAL POINT IS BETWEEN THE LAST BYTE OF THE FIRST ELEMENT			*
		5828	*	AND THE FIRST BYTE OF THE SECOND ELEMENT. THEREFORE THE FIRST			*
		5829	*	ELEMENT CONTAINS THE INTEGER PART, AND THE SECOND THE FRACTION.			*
		5830	*				*
		5831	*	*EXTERNAL REFERENCES			*
		5832	*	INTERPRETER STACK - FIRST TWO ELEMENTS, 16(32) BYTES			*
		5833	*	I\$FWRK - INTERPRETER WORK AREA, 9(17) BYTES			*
		5834	*	I\$STAK - CORE LOCATION OF ADDRESS OF THE INTERPRETER STACK			*
		5835	*	I\$RTRN - ENTRY TO PAGING RETURN ROUTINE			*
		5836	*				*
		5837	*	*EXITS, NORMAL			*
		5838	*	EXIT IS BY BRANCHING TO I\$RTRN, TO RETURN CONTROL TO THE CAL-			*
		5839	*	LING PROGRAM.			*
		5840	*				*
		5841	*	*EXITS, ERROR			*
		5842	*	NONE.			*
		5843	*				*
		5844	*	*TABLES/WORK AREAS			*
		5845	*	THE CONSTANTS AND WORK AREAS RESIDE AT THE END OF THE EXECUTABLE			*
		5846	*	CODE.			*
		5847	*				*
		5848	*	*ATTRIBUTES			*
		5849	*	REUSABLE, NATURALLY RELOCATABLE.			*
		5850	*				*
		5851	*	*CHARACTER CODE DEPENDENCY			*
		5852	*	THE LOW ORDER FOUR BITS OF DECIMAL NUMBERS, WHEN CONSIDERED AS A			*
		5853	*	BINARY INTEGER, IDENTIFY THE VALUE OF THE DIGIT.			*
		5854	*				*

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE	54
		5855	*	NOTES				*
		5856	*	ERROR PROCEDURES				*
		5857	*	THE ERROR CODE IS SET, AND CONTROL RETURNED TO THE CALLING				*
		5858	*	ROUTINE.				*
		5859	*					*
		5860	*	RESISTER USAGE				*
		5861	*	REGISTER 1 (@BR) IS USED AS THE BASE REGISTER.				*
		5862	*	REGISTER 2 (@XR) IS USED TO REFERENCE THE STACK.				*
		5863	*					*
		5864	*	SAVED/RESTORED AREAS				*
		5865	*	NONE.				*
		5866	*					*
		5867	*	MODIFICATION CONSIDERATIONS				*
		5868	*	CUPID MAY NOT USE ANY FUTHER STACK OR INTERPRETER WORK AREA				*
		5869	*	WITHOUT EXTENSIVE CHANGES IN THE ARITHMETIC ROUTINES.				*
		5870	*					*
		5871	*	REQUIRED MODULES				*
		5872	*	@SYSEC - COMMON SYSTEM EQUATES				*
		5873	*	@ERMEQ - ERROR MESSAGE EQUATES				*
		5874	*	\$V\$EQU - FIXED VIRTUAL ADDRESSES				*
		5875	*	\$B@EQU - COMPILER SYSTEM ECUATES				*
		5876	*	\$I\$EQU - FIXED EQUATES				*
		5877	*	\$I@SEQ/\$I@LEQ - STANDARD/LONG PRECISION EXECUTION EQUATES				*
		5878	*					*
		5879	*	OTHER				*
		5880	*	NONE.				*
		5881	*	*****				*

CBFPZD - PROLOGUE - FLOATING POINT TO ZONED DECIMAL

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER	15,	MOD	00	31/05/21	PAGE	55
					5883	*****								
					5884	*	CBFPZD - S/3.7 CONVERT FLOATING POINT TO ZONED DECIMAL ROUTINE							*
					5885	*****								
					5886	*								
					5887	*CREAM	VPAGE 112							
	0C70				5888	ORG	*,256,112						SET	STARTING ADDRESS
				0C70	5889	CBFAD1	EQU *						START	OF PROGRAM CODING
	0B71				5890	ORG	*-255						RESET	IAR TO PAGE
	0C00				5891	ORG	*,256,0						BOUNDARY	ADDRESS
				0C00	5892	USING	*,@BR						SET	PAGE BASE ADDRESS
	0C70				5893	ORG	CBFAD1						RESET	STARTING ADDRESS
					5894	***	END OF EXPANSION ***							
					5895	*								
					5896	*	EXECUTION ENTRY TO CBFPZD FOLLOWS							
					5897	*								
	0C70	35	02	0D4E	5898	CBFPZD	L I\$STAK,@XR						LOAD	STACK POINTER
	0C74	2C	07	060F 07	5899	MVC	I\$FWRK+I@LUFV(I@LUFV),I@RSE1(,@XR)						SAVE	INPUT NUMBER
	0C79	BC	F0	0F	5900	MVI	I@RSE2(,@XR),@DZERO						GET	DECIMAL ZERO
	0C7C	AC	0E	0E 0F	5901	MVC	I@RSE2-1(I@LUFV+I@PREC,@XR),I@RSE2(,@XR)						CLEAR	STACK WA
					5902	*								
					5903	*	IF THE EXPONENT OF THE FLOATING POINT NUMBER IS LESS THAN (PREC).							
					5904	*	OUTPUT ALL ZEROES. EXPONENTS LARGER THAN +(PREC+1) WILL CAUSE							
					5905	*	DESTRUCTION OF PARTS OF CORE BUT ARE NOT TESTED FOR, SO USERS OF THIS							
					5906	*	ROUTINE MUST BE CAREFUL NOT TO INPUT FLOATING POINT NUMBERS LARGER							
					5907	*	THAN 10**(PREC+1).							
					5908	*								
	0C80	3D	79	0608	5909	CLI	I\$FWRK+1,B@NXZR-I@PREC						TEST	SIZE OF EXPONENT
	0C84	D0	82	AC	5910	BL	CBF900(,@BR)						LESS	THAN PREC, OUTPUT ZEROES
					5911	*								
					5912	*	INCREMENT CONTENTS OF THE INDEX REG BY X'7F' (NORMALIZED EXPONENT -1)							
					5913	*	THEN DECREMENT CONTENTS OF INDEX REG BY THE NORMALIZED EXPONENT OF							
					5914	*	THE FLOATING POINT INPUT NUMBER. THE RESULT OF THIS IS TO MOVE THE							
					5915	*	EFFECTIVE STACK POINTER 1+EXP BYTES (TO THE LEFT IF 1+EXP IS POSITIVE							
					5916	*	OR TO THE RIGHT IF 1+EXP IS NEGATIVE).							
					5917	*								
	0C87	3C	00	0607	5918	MVI	I\$FWRK,@ZERO						PUT	ZERO IN FRONT OF EXPONENT
	0C8B	74	02	9A	5919	ST	CBF100+@OP1(,@BR),@XR						STORE	INDEX REG IN OPERAND
	0C8E	5E	01	9A B1	5920	ALC	CBF100+@OP1(CBFEXP,@BR),CBFSFT(,@BR)						ADD	X'7F' TO IT
	0C92	4F	01	9A 0608	5921	SLC	CBF100+@OP1(CBFEXP,@BR),I\$FWRK+1						SUBTRACT	EXP FROM IT
	0C97	C2	02	0000	5922	CBF100	LA *-*,@XR						DECREMENT	INDEX RED BY 1+EXP
					5923	*								
					5924	*	SHIFT MANTISSA OF INPUT FLOATING POINT NUMBER SO THAT THE DECIMAL							
					5925	*	POINT IS BETWEEN THE FIRST AND SECOND STACK ELEMENTS.							
					5926	*								
	0C9B	8C	06	0F 060F	5927	MVC	I@RSE2(I@PREC,@XR),I\$FWRK+I@LUFV						SHIFT	MANTISSA
	0CA0	BA	F0	0F	5928	SBN	I@RSE2(,@XR),B@ZPOS						MAKE	SIGN OF MANTISSA POSITIVE
	0CA3	35	02	0D4E	5929	L	I\$STAK,@XR						RESTORE	INDEX REG TO STACK PTR
	0CA7	88	00	0F 060F	5930	MZZ	I@RSE2(,@XR),I\$FWRK+I@LUFV						INSERT	CORRECT SIGN
					5931	*								
					5932	*	RETURN TO CALLING PROGRAM							
					5933	*								
	0CAC	C0	87	12D3	5934	CBF900	B I\$RTRN						RETURN	TO CALLER
					5935	*								
					5936	*	EQUATES AND CONSTANTS							
					5937	*								
					0002 5938	CBFEXP	EQU 2						LENGTH	OF EXP WORK AREA

[illegible]

CDBNZD - CONVERT BINARY NUMBER TO ZONED DECIMAL

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 57
5944				*****			
5945	*			5703-XM1 COPYRIGHT IBM CORP. 1970			*
5946	*			REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
5947	*						*
5948				*****			*
5949	*			*STATUS			*
5950	*			VERSION 1 MODIFICATION 0			*
5951	*						*
5952	*			*FUNCTION			*
5953	*			* CDBNZD CONVERTS A BINARY NUMBER TO A ZONED DECIMAL.			*
5954	*			* THE INPUT IS IN THE FIRST BYTE OF THE INTERPRETER STACK, AND			*
5955	*			THE ANSWER IS LEFT IN THE LAST THREE BYTES OF THE 1ST ELEMENT.			*
5956	*			THE MAXIMUM BINARY NUMBER IN ONE BYTE IS 255.			*
5957	*						*
5958	*			*ENTRY POINTS			*
5959	*			THE ENTRY IS CDBNZD. THE FORMAT OF THE CALLING SEQUENCE IS AS			*
5960	*			FOLLOWS:			*
5961	*			B I\$CALL			*
5962	*			DC AL2(V\$CBNZ)			*
5963	*						*
5964	*			*INPUT			*
5965	*			* THE INPUT IS A BINARY NUMBER IN THE FIRST BYTE OF THE INTER-			*
5966	*			PRETER STACK.			*
5967	*			* REGISTER 1 (@BR) POINTS TO THE BEGINNING OF THE PAGE.			*
5968	*			* I\$STAK CONTAINS THE ADDRESS OF THE FIRST BYTE OF THE INTER-			*
5969	*			PRETER STACK.			*
5970	*						*
5971	*			*OUTPUT			*
5972	*			THE OUTPUT IS A ZONED DECIMAL NUMBER, THE SAME VALUE AS THE			*
5973	*			INPUT, IN THE LAST THREE BYTES OF THE 1ST STACK ELEMENT.			*
5974	*			ITS MAXIMUM VALUE IS 255.			*
5975	*						*
5976	*			*EXTERNAL REFERENCES			*
5977	*			INTERPRETER STACK - FIRST ELEMENT, 8(16) BYTES			*
5978	*			I\$STAK - CORE LOCATION OF THE ADDRESS OF THE INTERPRETER STACK			*
5979	*			I\$RTRN - ENTRY TO THE PAGING MODULE TO RETURN			*
5980	*						*
5981	*			*EXITS, NORMAL			*
5982	*			* EXIT IS BY BRANCHING TO I\$RTRN TO RETURN CONTROL TO THE CALLING			*
5983	*			PROGRAM.			*
5984	*			* THE RESULT IS LEFT IN THE FIRST STACK ELEMENT.			*
5985	*						*
5986	*			*EXITS, ERROR			*
5987	*			NONE.			*
5988	*						*
5989	*			*TABLES/WORK AREAS			*
5990	*			THE CONSTANTS AND WORK AREAS RESIDE AT THE END OF THE EXECUTABLE			*
5991	*			CODE.			*
5992	*						*
5993	*			*ATTRIBUTES			*
5994	*			REUSABLE, NATURALLY RELOCATABLE.			*
5995	*						*
5996	*			*CHARACTER CODE DEPENDENCY			*
5997	*			THE LOW ORDER FOUR BITS OF DECIMAL NUMBERS, WHEN CONSIDERED AS A			*
5998	*			BINARY INTEGER, IDENTIFY THE VALUE OF THE DIGIT.			*
5999	*						*

CDBNZD - CONVERT BINARY NUMBER TO ZONED DECIMAL

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE	58
		6000	*	NOTES				*
		6001	*	ERROR PROCEDURES				*
		6002	*	NONE.				*
		6003	*					*
		6004	*	RESISTER USAGE				*
		6005	*	REGISTER 1 (@BR) IS USED AS THE BASE REGISTER.				*
		6006	*	REGISTER 2 (@XR) IS USED TO REFERENCE THE STACK.				*
		6007	*					*
		6008	*	SAVED/RESTORED AREAS				*
		6009	*	NONE.				*
		6010	*					*
		6011	*	MODIFICATION CONSIDERATIONS				*
		6012	*	NONE.				*
		6013	*					*
		6014	*	REQUIRED MODULES				*
		6015	*	@SYSEQ - COMMON SYSTEM EQUATES				*
		6016	*	@ERMEQ - ERROR MESSAGE EQUATES				*
		6017	*	V\$VEQU - FIXED VIRTUAL ADDRESSES				*
		6018	*	\$B@EQU - COMPILER SYSTEM EQUATES				*
		6019	*	\$I\$EQU - FIXED EQUATES				*
		6020	*	\$I@SEQ/\$I@LEQ - STANDARD/LONG PRECISION EXECUTION EQUATES				*
		6021	*					*
		6022	*	OTHER				*
		6023	*	NONE.				*
		6024	*	*****				*

CDBNZD - CONVERT BINARY NUMBER TO ZONED DECIMAL

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 59

```

6026 *****
6027 * CDBNZD - S/3.7 CONVERT BINARY NUMBER TO ZONED DECIMAL *
6028 *****
6029 *
6030 *CDBAD1 VPAGE 0 SET PAGE ADDRESSABILITY
0CB2 6031 ORG * SET STARTING ADDRESS
0BB3 0CB2 6032 CDBAD1 EQU * START OF PROGRAM CODING
0C00 6033 ORG *-255 RESET IAR TO PAGE
6034 ORG *,256,0 * BOUNDARY ADDRESS
0CB2 0C00 6035 USING *,@BR SET PAGE BASE ADDRESS
6036 ORG CDBAD1 RESET STARTING ADDRESS
6037 *** END OF EXPANSION ***
6038 *
6039 * EXECUTION ENTRY TO CDBNZD FOLLOWS
6040 *
0CB2 35 02 0D4E 6041 CDBNZD L I$STAK,@XR LOAD STACK POINTER
0CB6 94 60 07 DA 6042 ZAZ I@RSE1(I@PREC,@XR),CDBONE(1,@BR) ZERO AND SET ADDEND TO 1
6043 *
6044 * THE ADDEND WILL BE DOUBLED EACH TIME THE NEXT BIT (TO THE LEFT) OF
6045 * THE BINARY NUMBER IN THE BYTE I@1SE1(@XR) IS EXAMINED. THUS, THE
6046 * ADDEND WILL CONTAIN THE DECIMAL EQUIVALENT OF THE POWER OF TWO
6047 * CORRESPONDING TO THE BINARY NUMBER BIT BEING EXAMINED.
6048 *
0CBA 7C 01 BE 6049 MVI CDB010+@Q(@BR),@B1 INITIALIZE MASK TO 00000001
0CBD B8 00 00 6050 CDB010 TBN I@1SE1(@XR),*-* EXAMINE BIT
0CC0 D0 90 C7 6051 BF CDB100(@BR) IT IS ZERO SO DO NOT ADD ADDEND
0CC3 A6 02 04 07 6052 AZ I@1SE1+CDBACC(CDBADD,@XR),I@RSE1(CDBADD,@XR) ACCUM+ADDEND
6053 *
6054 * MAXIMUM SIZE OF ADDEND IS 128.
6055 * MAXIMUM SIZE OF ACCUMULATED RESULT IS 255.
6056 *
0CC7 A6 02 07 07 6057 CDB100 AZ I@RSE1(CDBADD,@XR),I@RSE1(CDBADD,@XR) DOUBLE THE ADDEND
0CCB 5E 00 BE BE 6058 ALC CDB010+@Q(1,@BR),CDB010+@Q(@BR) SHIFT MASK 1 BIT LEFT
0CCF D0 82 BD 6059 BL CDB010(@BR) IF NO OVERFLOW, DO NEXT BIT
6060 *
6061 * LEAVE ZONED DECIMAL RESULT RIGHT JUSTIFIED IN TOP OF STACK.
6062 *
0CD2 A4 52 07 04 6063 ZAZ I@RSE1(I@LUFV,@XR),I@1SE1+CDBACC(CDBADD,@XR) SHIFT RIGHT
0CD6 C0 87 12D3 6064 B I$RTRN RETURN TO CALLING PROGRAM
6065 *
6066 * CONSTANTS
6067 *
0004 6068 CDBACC EQU 4 ACCUMULATOR
0003 6069 CDBADD EQU 3 LENGTH OF ADDEND
6070 *
6071 * STORED CONSTANTS
6072 *
0CDA F1 0CDA 6073 CDBONE DC DL1'1' DECIMAL 1 TO INITIALIZE ADDEND
6074 *
6075 * END OF CDBNZD CODING
6076 *

```

FWSTAN - PROLOGUE - TANGENT/COTANGENT FUNCTION

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 60
		6078		*****			
		6079	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
		6080	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
		6081	*				*
		6082		*****			*
		6083	*	*STATUS			*
		6084	*	VERSION 1 MODIFICATION 0			*
		6085	*				*
		6086	*	*FUNCTION			*
		6087	*	* THE FNSTAN MODULE COMPUTES THE TANGENT OR COTANGENT IN STANDARD			*
		6088	*	PRECISION, OF THE INPUT ARGUMENT.			*
		6089	*	* THE INPUT ARGUMENT MUST BE IN THE RANGE (-100, 100), AND IS			*
		6090	*	REDUCED BY DIVIDING BY PI, AND REMOVING THE INTEGER PORTION OF			*
		6091	*	THAT RESULT.			*
		6092	*	* IF TIE REDUCED ARGUMENT IS LESS THAN .001, IT IS RETURNED AS			*
		6093	*	THE ANSWER. OTHERWISE A RECURSIVE ITERATION IS USED.			*
		6094	*				*
		6095	*	*ENTRY POINTS			*
		6096	*	* THE ENTRY TO COMPUTE THE TANGENT IS FASTAN. THE FORMAT OF THE			*
		6097	*	CALLING SEQUENCE IS AS FOLLOWS:			*
		6098	*	B I\$CALL			*
		6099	*	DC AL2(V\$FTAN)			*
		6100	*	* THE ENTRY TO COMPUTE THE COTANGENT IS FWSCOT. THE FORMAT OF			*
		6101	*	THE CALLING SEQUENCE IS AS FOLLOIS:			*
		6102	*	B I\$CALL			*
		6103	*	DC AL2(V\$FCOT)			*
		6104	*				*
		6105	*	*INPUT			*
		6106	*	* THE INPUT IS AN UNPACKED FLOATING POINT NUMBER IN TWE FIRST			*
		6107	*	STACK ELEMENT.			*
		6108	*	* REGISTER 1 (BR) POINTS TO THE PAGE BEGINNING.			*
		6109	*				*
		6110	*	*OUTPUT			*
		6111	*	* THE RESULT IS PLACED IN THE FIRST STACK ELEMENT IN UNPACKED			*
		6112	*	FLOATING POINT FORMAT.			*
		6113	*	* IN THE EVENT OF AN ERROR, THE ERROR CODE IS PLACED IN THE			*
		6114	*	ONE-BYTE INTERPRETER ERROR AREA, I\$ERRC.			*
		6115	*				*
		6116	*	*EXTERNAL REFERENCES			*
		6117	*	INTERPRETER STACK - FIRST TWO ELEMENTS, 16 BYTES			*
		6118	*	I\$FWRK - INTERPRETER WORK AREA, 17 BYTES			*
		6119	*	I\$STAK - CORE LOCATION OF ADDRESS OF THE INTERPRETER			*
		6120	*	I\$ERRC - INTERPRETER ONE-BYTE ERROR AREA			*
		6121	*	I\$CALL - ENTRY TO PAGING ROUTINE			*
		6122	*	I\$RTRN - ENTRY TO PAGING RETURN ROUTINE			*
		6123	*	I\$FSUB - ENTRY TO SUBTRACT ROUTINE			*
		6124	*	I\$FMPY - ENTRY TO MULTIPLY ROUTINE			*
		6125	*	I\$FDVD - ENTRY TO DIVIDE ROUTINE			*
		6126	*	CBFPZD - CONVERT FLOATING POINT TO ZONED DECIMAL			*
		6127	*	CCZDFP - CONVERT ZONED DECIMAL TO FLOATING POINT			*
		6128	*				*
		6129	*	*EXITS, NORMAL			*
		6130	*	* EXIT IS BY BRANCHING TO I\$RTRN, THE ENTRY TO THE PAGING RETURN			*
		6131	*	ROUTINE.			*
		6132	*	* THE RESULT IS LEFT IN TIE FIRST STACK ELEMENT.			*
		6133	*				*

FWSTAN - PROLOGUE - TANGENT/COTANGENT FUNCTION

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 61
		6134	*	EXITS, ERROR	*
		6135	*	* AN ERROR CODE IS PLACED IN THE ERROR AREA, I\$ERRC.	*
		6136	*	* EXIT IS BY BRANCHING TO I\$RTRN.	*
		6137	*		*
		6138	*	TABLES/WORK AREAS	*
		6139	*	THE CONSTANTS AND WORK AREAS RESIDE AT THE END OF THE EXECUTRBLE	*
		6140	*	CODE.	*
		6141	*		*
		6142	*	ATTRIBUTES	*
		6143	*	REUSABLE, NATURALLY RELOCATABLE.	*
		6144	*		*
		6145	*	CHARACTER CODE DEPENDENCY	*
		6146	*	THE OPERATION OF THIS MODULE DEPENDS UPON A ZONED DECIMAL DIGIT	*
		6147	*	BEING REPRESENTED WITH THE ZONE (FIRST FOUR BITS) REINS AN "F"	*
		6148	*	OR POSITIVE, AND A "D" FOR NEGATIVE. THIS PROPERTY IS USED	*
		6149	*	PRIOR TO AND FOLLOWING FWS300.	*
		6150	*	THE LOW ORDER FOUR BITS OR DECIMAL NUMBERSS, WHEN CONSIDERED AS A	*
		6151	*	BINARY INTEGER, IDENTIFY THE VALUE OR THE DIGIT.	*
		6152	*		*
		6153	*	NOTES	*
		6154	*	ERROR PROCEDURES	*
		6155	*	THE ERROR CODE IS SET AND CONTROL RETURNED TO THE CALLING	*
		6156	*	PROGRAM.	*
		6157	*		*
		6158	*	REGISTER USAGE	*
		6159	*	* REGISTER 1 (BR) IS USED AS THE BASE REGISTER.	*
		6160	*	* REGISTER 2 (XR) IS USED TO RERERENCE THE INTERPRETER STACK.	*
		6161	*		*
		6162	*	SAVED/RESTORED AREAS	*
		6163	*	NONE.	*
		6164	*		*
		6165	*	MODIFICATION CONSIDERATIONS	*
		6166	*	NONE.	*
		6167	*		*
		6168	*	REQUIRED MODULES	*
		6169	*	@SYSEQ - COMMON SYSTEM EQUATES	*
		6170	*	@ERMEQ - ERROR MESSAGE EQUATES	*
		6171	*	\$V\$EQU - FIXED VIRTUAL ADDRESSES	*
		6172	*	\$B@EQU - COMPILER SYSTEM EQUATES	*
		6173	*	\$I\$EQU - FIXED EQUATES	*
		6174	*	\$I@SEQ - STANDARD PRECISION EXECUTION EQUATES	*
		6175	*		*
		6176	*	OTHER	*
		6177	*	NONE.	*
		6178	*		*
		6179	*	*****	*

FWSTAN - PROLOGUE - TANGENT/COTANGENT FUNCTION

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 62
					6181	*****				
					6182	* FWSTAN - S/3.7 TANGENT/COTANGENT FUNCTION RTN - STANDARD PREC				*
					6183	*****				
					6184	*				
0D00					6185	ORG	*,B@LVP,0			
				0D00	6186	USING	*,@BR		SET PAGE BASE ADDRESS	
					6187	*				
					6188	* EXECUTION ENTRY TO FWSCOT FOLLOWS				
					6189	*				
0D00	35	02	0D4E		6190	FWSCOT	L I\$STAK,@XR		SET INDEX REG TO STACK	
0D04	BD	F0	01		6191		CLI I@1SE1+I@MANL(,@XR),@DZERO		IS THE INPUT ARG ZERO ?	
0D07	D0	01	10		6192		BNE FWS005(,@BR)		CONTINUE IF NOT ZERO	
0D0A	7C	E5	36		6193		MVI FWS009+@Q(,@BR),@@E780		ERROR, FUNCTION IS INFINITE	
0D0D	D0	87	35		6194		B FWS009(,@BR)		SET ERROR MESSAGE AND RETURN	
0D10	BD	83	00		6195	FWS005	CLI I@1SE1+I@DEXP(,@XR),B@NXZR+FWSLRG		IS ARG LESS THAN 100 ?	
0D13	7C	F0	36		6196		MVI FWS009+@Q(,@BR),@@E794		SET COS ERROR MESSAGE	
0D16	D0	02	35		6197		BNL FWS009(,@BR)		SET ERROR MESSAGE AND RETURN	
0D19	6C	07	27 07		6198		MVC FWSSAV(I@LUFV,@BR),I@RSE1(,@XR)		SAVE ARG X	1-5
0D1D	D0	87	62		6199		B FWS040(,@BR)		BRANCH AROUND TAN	1-5
0D20				0D27	6200	FWSSAV	DS XL(I@LUFV)		ARG SAVE AREA	1-5
					6201	*				
					6202	* EXECUTION ENTRY TO FWSTAN FOLLOWS				
					6203	*				
0D28	35	02	0D4E		6204	FWSTAN	L I\$STAK,@XR		SET INDEX TO STACK	
0D2C	7C	EF	36		6205		MVI FWS009+@Q(,@BR),@@E793		SET SIGN ERROR MESSAGE	
0D2F	BD	83	00		6206	FWS007	CLI I@1SE1+I@DEXP(,@XR),B@NXZR+FWSLRG		IS ARG LESS THAN 100 ?	
0D32	D0	82	3C		6207		BL FWS030(,@BR)		IF YES, CONTINUE	
0D35	3C	00	0CBC		6208	FWS009	MVI I\$ERRC,*-*		NO, SET ERROR: ARG TOO LARGE	
0D39	D0	87	80		6209		B FWS900(,@BR)		RETURN TO CALLING PROGRAM	
0D3C	6C	07	27 07		6210	FWS030	MVC FWSSAV(I@LUFV,@BR),I@RSE1(,@XR)		SAVE ARG X	1-5
0D40	C0	87	12B1		6211		B I\$CALL		CALL COS	1-5
0D44	0A00			0D45	6212		DC AL2(V\$FCOS)			1-5
0D46	2C	07	0617 07		6213		MVC I\$FWRK+I@LUFV+I@LUFV(I@LUFV),I@RSE1(,@XR)		SAVE COS(X)	1-5
0D4B	9C	07	07 27		6214		MVC I@RSE1(,@XR),FWSSAV(I@LUFV,@BR)		RESTORE ARG X	1-5
0D4F	C0	87	12B1		6215		B I\$CALL		CALL SIN	1-5
0D53	0A1A			0D54	6216		DC AL2(V\$FSIN)		NUM - SIN(X)	1-5
0D55	8C	07	0F 0617		6217		MVC I@RSE2(,@XR),I\$FWRK+I@LUFV+I@LUFV(I@LUFV)		DENUM=COS(X)	1-5
0D5A	C0	87	0919		6218		B I\$FDVD		TAN(X) = SIN(X)/COS(X)	1-5
0D5E	C0	87	12D3		6219		B I\$RTRN		RETURN	1-5
0D62	C0	87	12B1		6220	FWS040	B I\$CALL		CALL SIN	1-5
0D66	0A1A			0D67	6221		DC AL2(V\$FSIN)		NUM = SIN(X)	1-5
0D68	2C	07	0617 07		6222		MVC I\$FWRK+I@LUFV+I@LUFV(I@LUFV),I@RSE1(,@XR)		SAVE SIN(X)	1-5
0D6D	9C	07	07 27		6223		MVC I@RSE1(,@XR),FWSSAV(I@LUFV,@BR)		RESTORE ARG X	1-5
0D71	C0	87	12B1		6224		B I\$CALL		CALL COS	1-5
0D75	0A00			0D76	6225		DC AL2(V\$FCOS)		NUM = COS(X)	1-5
0D77	8C	07	0F 0617		6226		MVC I@RSE2(,@XR),I\$FWRK+I@LUFV+I@LUFV(I@LUFV)		DENOM=SIN(X)	1-5
0D7C	C0	87	0919		6227		B I\$FDVD		COT(X) = COS(X)/SIN(X)	1-5
0D80	C0	87	12D3		6228	FWS900	B I\$RTRN		RETURN	
					6229	*				
				0003	6230	FWSLRG	EQU 3		EXP FOR ARG OF 10*2	1-5
0D84				0DFB	6231	FWSPCH	DS CL(120)		PATCH AREA	1-5
					6232	*				
					6233	* END OF FWSTAN/FWSCOT CODING				
					6234	*				

[illegible]

FBSATN - PROLOGUE - ARCTANGENT FUNCTION - STD PREC

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 64
		6241		*****			
		6242	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
		6243	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
		6244	*				*
		6245		*****			*
		6246	*	*STATUS			*
		6247	*	VERSION 1 MODIFICATION 0			*
		6248	*				*
		6249	*	*FUNCTION			*
		6250	*	* FBSATN COMPUTES THE ARCTANGENT IN STANDARD PRECISION, OF THE			*
		6251	*	INPUT ARGUMENT.			*
		6252	*	* THE INPUT ARGUMENT IS REDUCED TO A VALUE LESS THAN 10, AND THEN			*
		6253	*	HANDLED BY A RECURSIVE ALGORITHM.			*
		6254	*				*
		6255	*	*ENTRY POINTS			*
		6256	*	* THE ENTRY IS FBSATN. THE FORMAT OF THE CALLING SEQUENCE IS AS			*
		6257	*	FOLLOWS:			*
		6258	*	B I\$CALL			*
		6259	*	DC AL2(V\$FATN)			*
		6260	*				*
		6261	*	*INPUT			*
		6262	*	* THE INPUT IS AN UNPACKED FLOATING POINT NUMBER IN TWE FIRST			*
		6263	*	STACK ELEMENT.			*
		6264	*	* REGISTER 1 (BR) POINTS TO THE PAGE BEGINNING.			*
		6265	*				*
		6266	*	*OUTPUT			*
		6267	*	* THE RESULT IS PLACED IN THE FIRST STACK ELEMENT IN UNPACKED			*
		6268	*	FLOATING POINT FORMAT.			*
		6269	*				*
		6270	*	*EXTERNAL REFERENCES			*
		6271	*	INTERPRETER STACK - FIRST TWO ELEMENTS, 16 BYTES			*
		6272	*	I\$FWRK - INTERPRETER WORK AREA, 17 BYTES			*
		6273	*	I\$STAK - CORE LOCATION OF ADDRESS OF THE INTERPRETER			*
		6274	*	I\$ERRC - INTERPRETER ONE-BYTE ERROR AREA			*
		6275	*	I\$CALL - ENTRY TO PAGING ROUTINE			*
		6276	*	I\$RTRN - ENTRY TO PAGING RETURN ROUTINE			*
		6277	*	I\$FSUB - ENTRY TO SUBTRACT ROUTINE			*
		6278	*	I\$FMPY - ENTRY TO MULTIPLY ROUTINE			*
		6279	*	I\$FDVD - ENTRY TO DIVIDE ROUTINE			*
		6280	*	CBFPZD - CONVERT FLOATING POINT TO ZONED DECIMAL			*
		6281	*	CCZDFP - CONVERT ZONED DECIMAL TO FLOATING POINT			*
		6282	*				*
		6283	*	*EXITS, NORMAL			*
		6284	*	* EXIT IS BY BRANCHING TO I\$RTRN, THE ENTRY TO THE PAGING RETURN			*
		6285	*	ROUTINE.			*
		6286	*	* THE RESULT IS LEFT IN TIE FIRST STACK ELEMENT.			*
		6287	*				*
		6288	*	*EXITS, ERROR			*
		6289	*	NONE.			*
		6290	*				*
		6291	*	*TABLES/WORK AREAS			*
		6292	*	THE CONSTANTS AND WORK AREAS RESIDE AT THE END OF THE EXECUTRBLE			*
		6293	*	CODE.			*
		6294	*				*
		6295	*	*ATTRIBUTES			*
		6296	*	REUSABLE, NATURALLY RELOCATABLE.			*

```
6297 *
6298 *CHARACTER CODE DEPENDENCY
6299 * THE OPERATION OF THIS MODULE DEPENDS UPON A ZONED DECIMAL DIGIT
6300 * BEING REPRESENTED WITH THE ZONE (FIRST FOUR BITS) REINS AN "F"
6301 * OR POSITIVE, AND A "D" FOR NESATIVE. THIS PROPERTY IS USED
6302 * AT THE ENTRY.
6303 * THE LOW ORDER FOUR BITS OR DECIMAL NUMBERSS, WHEN CONSIDERED AS A
6304 * BINARY INTEGER, IDENTIFY THE VALUE OR THE DIGIT.
6305 *
6306 *NOTES
6307 * ERROR PROCEDURES
6308 * NONE.
6309 *
6310 * REGISTER USAGE
6311 * * REGISTER 1 (BR) IS USED AS THE BASE REGISTER.
6312 * * REGISTER 2 (XR) IS USED TO RERERENCE THE STACK.
6313 *
6314 * SAVED/RESTORED AREAS
6315 * NONE.
6316 *
6317 * MODIFICATION CONSIDERATIONS
6318 * NONE.
6319 *
6320 * REQUIRED MODULES
6321 * @SYSEQ - COMMON SYSTEM EQUATES
6322 * $I$EQU - FIXED EQUATES
6323 * $V$EQU - FIXED VIRTUAL ADDRESSES
6324 * $B@EQU - COMPILER SYSTEM EQUATES
6325 * $I@SEQ - STANDARD PRECISION EXECUTION EQUATES
6326 *
6327 * OTHER
6328 * NONE.
6329 *
6330 *****
```

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15,	MOD 00	31/05/21	PAGE	66
					6332	*****	*****					
					6333	*	FIMATN - S/3.7 ARCTANGENT FUNCTION - STANDARD PRECISION					*
					6334	*****	*****					
	1100				6335	*						
				1100	6336	ORG	*,B@LVPG,0				PAGE BOUNDARY	
					6337	USING	*,@BR				BASE REG POINTS HERE AT ENTRY	
					6338	*						
					6339	*	EXECUTION ENTRY TO FBSATN					
					6340	*						
	1100	35	02	0D4E	6341	FBSATN L	I\$STAK,@XR				INDEX REG TO TOP OF STACK	
	1104	BD	7E	00	6342		CLI I@1SE1+I@DEXP(,@XR),B@NXZR-2				IS ARG TOO SMALL FOR ITERATN	
	1107	D0	82	7B	6343		BL FBS900(,@BR)				IF YES, RETURN ARG AS ANSWER	
	110A	68	00	7F 07	6344		MZZ FBSSGN(,@BR),I@1SE1+I@SIGN(,@XR)				SAVE THE SIGN OF THE ARG	
	110E	BA	20	07	6345		SBN I@1SE1+I@SIGN(,@XR),X'20'				MAKE ARG POSITIVE	
					6346	*						
					6347	*	IF THE ARGUMENT IS LESS THAN 10, SET THE BRANCH AT FBS800 TO AN					
					6348	*	UNCONDITIONAL BRANCH, IF 10 OR GREATER, SET TO A NOP.					
					6349	*						
	1111	BD	81	00	6350	FBS100 CLI	I@1SE1+I@DEXP(,@XR),B@NXZR+1				IS ARG LESS THAN 10 ?	
	1114	D0	84	1D	6351		BH FBS110(,@BR)				IF NO, BRANCH	
	1117	7C	87	67	6352		MVI FBS800+@Q(,@BR),@UCB				SET FBS800 TO AN UNCONDITIONAL	
	111A	D0	87	48	6353		B FBS190(,@BR)				BRANCH OVER RANGE REDUCTION	
					6354	*						
					6355	*	REDUCE RANGE TO (.818,1) BY DIVIDING (ARG-1)/(ARG+1)					
					6356	*						
	111D	7C	80	67	6357	FBS110 MVI	FBS800+@Q(,@BR),@NOP				SET FBS800 TO A NON-BRANCH	
	1120	9C	07	0F 90	6358		MVC I@RSE2(I@LUFV,@XR),FBSONE(,@BR)				GET ONE TO ADD TO ARG	
	1124	2C	07	0617 07	6359		MVC I\$FWRK+I@LUFV+I@LUFV(I@LUFV),I@RSE1(,@XR)				SAVE ARG	
	1129	C0	87	075D	6360		B I\$FADD				ADD ARG + 1	
	112D	2C	07	061F 07	6361		MVC I\$FWRK+I@LUFV+I@LUFV+I@LUFV,I@RSE1(I@LUFV,@XR)				SAVE ARG+1	
	1132	8C	07	07 0617	6362		MVC I@RSE1(I@LUFV,@XR),I\$FWRK+I@LUFV+I@LUFV				SET ORIGINAL ARG	
	1137	9C	07	0F 90	6363		MVC I@RSE2(I@LUFV,@XR),FBSONE(,@BR)				GET ONE TO SUBTRACT	
	113B	C0	87	0751	6364		B I\$FSUB				SUBTRACT ARG - 1	
	113F	8C	07	0F 061F	6365		MVC I@RSE2(I@LUFV,@XR),I\$FWRK+I@LUFV+I@LUFV+I@LUFV				GET ARG+1	
	1144	C0	87	0919	6366		B I\$FDVD				DIVIDE (ARG - 1)/(ARG + 1)	
					6367	*						
					6368	*	CONVERT NEW ARGUMENT TO ZONED DECIMAL					
					6369	*						
	1148	C0	87	12B1	6370	FBS190 B	I\$CALL					
	114C	0C70			114D 6371		DC AL(@VADDR)(V\$CFPZ)					
					6372	*						
					6373	*	PREPARE FOR THE ITERATION, NAMES OR STORAGE AREAS ARE CHOSEN TO					
					6374	*	MATCH THE FLOWCHART.					
					6375	*						
	114E	24	F4	061F 05	6376	FBS200 ZAZ	FBSWW(FBSLNW+FBSLNR),I@1SE1+FBSZER(FBSZER,@XR)				CLEAR WA	
	1153	3C	00	0607	6377		MVI I\$FWRK,@ZERO				SET ITERATION NUMBER J = 0	
	1157	3C	F1	0617	6378		MVI FBSWW-FBSLNW+3,B@DEC1					

FBSATN - PROLOGUE - ARCTANGENT FUNCTION - STD PREC

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 67

```

1166 D0 00 6D          6388 FBS800 BC    FBS810(,@BR),*-*          BRANCH IF ORIGINAL ARG < 10
1169 96 87 0F 88      6389          AZ    I@RSE2(I@LUFV+I@LUFV,@XR),FBSAT1(I@LUFV,@BR) ADD ATM(1)
116D 96 F0 0F 80      6390 FBS810 AZ    I@RSE2(I@LUFV+I@LUFV,@XR),FBSRND(1,@BR)  ROUND THE ANSWER
                        6391 *
                        6392 * CONVERT TO FLOATING POINT
                        6393 *
1171 C0 87 12B1        6394          B    I$CALL
1175 04AD              1176 6395          DC    AL(@VADDR)(V$CZFP)
1177 98 00 07 7F      6396          MZZ   I@RSE1(,@XR),FBSSGN(,@BR) INSERT SIGN
                        6397 *
                        6398 * RETURN TO CALLING PROGRAM
                        6399 *
117B C0 87 12D3        6400 FBS900 B    I$RTRN
                        6401 *
                        6402 * NON-STORAGE CONSTANTS
                        6403 *
                        0005 6404 FBSZER EQU    5                      LENGTH FOR ZEROING
                        0009 6405 FBSLNR EQU    I@LUFV+1              LENGTH OF WORK AREA R
                        000B 6406 FBSLNW EQU    I@LUFV+3              LENGTH OF WORK AREA W
                        0614 6407 FBSRRR EQU    I$FWRK+I@LUFV+I@LUFV-3  END OF WORK AREA R
                        061F 6408 FBSWWW EQU    I$FWRK+I@LUFV+I@LUFV+I@LUFV  END OR WORK AREA W
                        6409 *
                        6410 * STORED CONSTANTS AND WORK AREAS
                        6411 *
117F                  117F 6412 FBSSGN DS    XL1                      AREA TO SAVE SIGN ZONE
1180 F5              1180 6413 FBSRND DC    DL1'5'                  DECIMAL 5 FOR ROUNDING ANSWER
1181 F7F8F5F3F9F8F1F6 1188 6414 FBSAT1 DC    DL(I@LUFV)'78539816'    ATN(1)
1189 81              1189 6415          DC    AL1(B@NXZR+1)          EXP FOR
118A F1F0F0F0F0F0F0 1190 6416 FBSONE DC    DL(I@PREC)'1000000'      * + 1
                        6417 *
                        6418 *****
                        6419 * SECOND PAGE OF FBSATN
                        6420 *
1200                  1200 6421          ORG    *,B@LVP,0              PAGE BOUNDARY
                        6422          USING *,@BR                      SET BASE REG TO POINT HERE
                        6423 *
                        6424 * EXECUTION ENTRY TO SECOND PAGE OF FBSATN.  INITIALIZE NECESSARY
                        6425 * INSTRUCTIONS
                        6426 *
1200 5C 03 11 88      6427          MVC    FBS405+@DD2(@INST4,@BR),FBSINZ+@DD2(,@BR)  INIT WA Z INSTR
1204 7C 9F 28          6428          MVI    FBS425+@OPD2(,@BR),FBSADA INIT 2ND ADDRESS TO CONSTANTS
1207 D0 87 0E          6429          B      FBS405(,@BR)              FBS405 IS NOT MODIFIED AT ENTRY
                        6430 *
                        6431 * THE INSTRUCTION AT FBS405 WHICH SETS WORK AREA Z EQUAL TO 10**(2J) IS
                        6432 * MODIFIED BY SUBTRACTING TWO FROM THE LENGTH OF THE SECOND OPERAND.
                        6433 * WHICH NECESSITATES ADDING TWO TO THE LENGTH DIFFERENCE IN ORDER TO
                        6434 * MAINTAIN THE LENGTH OF Z, AND SHIFTING Y RIGHT TWO (SUBTRACTING TWO).
                        6435 *
120A 5F 03 11 8C      6436 FBS400 SLC    FBS405+@DD2(@INST4,@BR),FBSMDZ(,@BR)  MODIFY INSTR FOR /
120E 64 90 78 00      6437 FBS405 ZAZ    FBSZZZ(I@LUFV+2,@BR),*-(@VQ,@XR) GET Z = Y / (10**2J)
1212 87 0A 0F 061F    6438 FBS420 SZ    I@RSE2(FBSLNW,@XR),FBSWWW(FBSLNW)  Y = Y - W
1217 D0 82 3E          6439          BL     FBS600(,@BR)              IF NEGATIVE, BRANCH FOR NEXT J
                        6440 *
                        6441 * FOR J LESS THAN 4, THE FOUR CONSTANTS ATN(10**(-J)) ARE REFERENCED
                        6442 * DIRECTLY.  FOR J GREATER THAN 3 (OVER 4 ITERATIONS), THE LAST
                        6443 * CONSTANT IS SHIFTED ONE BYTE TO THE RIGHT (DEVIDED BY TEN).

```

FBSATN - PROLOGUE - ARCTANGENT FUNCTION - STD PREC

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER	15,	MOD	00	31/05/21	PAGE	68
					6444	*								
121A	3D	04	0607		6445	CLI	I\$FWRK,X'04'						MORE THAN 4 ITERATIONS	
121E	D0	81	31		6446	BE	FBS440(,@BR)						IF 5, BRANCH TO PREPARE FOR 5-7	
1221	D0	84	36		6447	BH	FBS450(,@BR)						OVER 5, CONTINUE WITH LAST 2	
1224	16	17	0614 00		6448	FBS425	AZ	FBSRRR(FBSLNR),*-(I@LUFV,@BR)					ADD R = R + ATN(10**(-J))	
1229	16	0A	061F 78		6449	FBS430	AZ	FBSWWW(FBSLNW),FBSZZZ(FBSLNW,@BR)					ADD W = W + Z	
122E	D0	87	0E		6450	B	FBS405(,@BR)						REPEAT UNTIL Y IS NEGATIVE	
					6451	*								
1231	16	0A	061F 78		6452	FBS440	AZ	FBSWWW(FBSLNW),FBSZZZ(FBSLNW,@BR)					ADD W = W + Z LAST TIME	
1236	16	80	0614 00		6453	FBS450	AZ	FBSRRR(FBSLNR),*-(@VQ,@BR)					ADD R = R + (ATN(10**(-J)))/1	
123B	D0	87	12		6454	B	FBS420(,@BR)							
					6455	*								
					6456	*	Y IS NEGATIVE							
					6457	*								
123E	3D	08	0607		6458	FBS600	CLI	I\$FWRK,I@LUFV					HAS THE ITERATION FINISHED ?	
1242	C0	81	12D3		6459	BE	I\$RTRN						IF YES, RETURN	
					6460	*								
					6461	*	THE INSTRUCTION AT FBS450 IS MODIFIED BY SUBTRACTING ONE FROM THE							
					6462	*	LENGTH OF THE SECOND OPERAND, WHICH NECESSITATES ADDING ONE TO THE							
					6463	*	LENGTH DIFFERENCE, AND SUBTRACTING 1 FROM THE SECOND OPERAND ADDRESS.							
					6464	*	THE INSTRUCTION AT FBS425 IS ALTERED BY INCREMENTING THE SECOND							
					6465	*	OPERAND ADDRESS TO THE NEXT VALUE OF ATN(10**(-J))							
					6466	*								
					6467	*	ALTER Y TO REPEAT PROCEDURE FOR NEW ITERATION							
					6468	*								
1246	5F	03	3A 96		6469	SLC	FBS450+@OPD2(@INST4,@BR),FBSMDS(,@BR)						MODIFY TO SHIFT	
124A	5E	00	28 8D		6470	ALC	FBS425+@OPD2(1,@BR),FBSLNT(,@BR)						INCREMENT TO NEXT ATN	
124E	86	0A	0F 061F		6471	AZ	I@RSE2(FBSLNW,@XR),FBSWWW(FBSLNW)						Y = Y + W	
1253	6C	0A	83 0F		6472	MVC	FBS10Y(FBSLNW,@BR),I@RSE2(,@XR)						MOVE Y TO TEMPORARY WA	
1257	9C	0B	0F 84		6473	MVC	I@RSE2(,@XR),FBS10Y+1(FBSLNW+1,@BR)						Y = 10 * Y	
125B	1E	00	0607 97		6474	ALC	I\$FWRK(1),FBSBN1(,@BR)						J = J + 1	
1260	3D	04	0607		6475	CLI	I\$FWRK,X'04'						ARE THERE FOUR ITERATIONS	
1264	D0	84	12		6476	BH	FBS420(,@BR)						OVER 4, CONTINUE WITHOUT Z	
1267	5C	03	3A 92		6477	MVC	FBS450+@OPD2(@INST4,@BR),FBSINS+@OPD2(,@BR)						INIT SHIFT	
126B	D0	87	0A		6478	B	FBS400(,@BR)						REPEAT	
					6479	*								
					6480	*	STORAGE AND WORK AREAS							
					6481	*								
126E				1278	6482	FBSZZZ	DS	XL(FBSLNW)					ADDEND TO W. WORK AREA Z	
1279				1283	6483	FBS10Y	DS	XL(FBSLNW)					STORAGE FOR 10 * Y	
1284	F0			1284	6484		DC	AL1(@DZERO)					TRAILING ZERO FOR 10 * Y	
					6485	*								
					6486	*	CONSTANTS AND INITIAL INSTRUCTIONS							
					6487	*								
1285	64	0A	78 0F		6488	FBSINZ	ZAZ	FBSZZZ(I@LUFV+3,@BR),I@RSE2(I@LUFV+3,@XR)					Z = Y/(10**(2J))	
1289	FFE20002			128C	6489	FBSMDZ	DC	XL(@INST4)'FFE20002'					MODIFICATION FOR FBS405	
128D	08			128D	6490	FBSLNT	DC	AL(1)(I@LUFV)					LENGTH OF ATN CONSTANT TO MODIFY	
128E	16	26	0614 B6		6491	FBSINS	AZ	FBSRRR(FBSLNR),FBSLST(I@PREC,@BR)					R=R+(ATN(10**(-J)))/10	
1293	F1000001			1296	6492	FBSMDS	DC	XL(@INST4)'F1000001'					MODIFY FBS450 TO SHIFT	
1297	01			1297	6493	FBSBN1	DC	AL1(@B1)					BINARY 1 TO INCREMENT J	
					6494	*								
					6495	*	ATN(10**(-J))							
					6496	*								
1298	F7F8F5F3F9F8F1F6			129F	6497	FBSATA	DC	DL(I@LUFV)'78539816'					J = 0	
12A0	F0F9F9F6F6F8F6F5			12A7	6498		DC	DL(I@LUFV)'09966865'					J = 1	
12A8	F0F0F9F9F9F9F6F7			12AF	6499		DC	DL(I@LUFV)'00999967'					J = 2	

FBSATN - PROLOGUE - ARCTANGENT FUNCTION - STD PREC

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE	69
	12B0	F0F0F1F0F0F0F0F0	12B7	6500		DC	DL(I@LUFV)'00100000'		J = 3		
				6501	*						
			12B6	6502	FBSLST	EQU	*-2		ATN(10**(-4))		
			009F	6503	FBSADA	EQU	FBSATA-FBSATN-256		INITIAL 2ND ADDRESS FOR FBS425		
				6504	*						
				6505	*	END OF FBSATN CODING					
				6506	*						
	1300			6507		ORG	*,B@LVPG,0		START FBSATN DUMMY 3RD PAGE		
	1400			6508		ORG	*+B@LVPG,0		SIMULATE FBSATN DUMMY 3RD PAGE		

FCSASN - ARCSINE/ARCCOSINE FUNCTIONS - STD PREC

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 70
		6510		*****			
		6511	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
		6512	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
		6513	*				*
		6514		*****			*
		6515	*	*STATUS			*
		6516	*	VERSION 1 MODIFICATION 0			*
		6517	*				*
		6518	*	*FUNCTION			*
		6519	*	* THE FCSASN MODULE COMPUTES THE ARCSINE OR ARCOSINE, IN STANDARD			*
		6520	*	PRECISION, OF THE INPUT ARGUMENT.			*
		6521	*	* THE RESULT IS COMPUTED BY DEFINITION THROUGH THE PYTHAGOREAN			*
		6522	*	THEOREM.			*
		6523	*				*
		6524	*	*ENTRY POINTS			*
		6525	*	* THE ENTRY TO COMPUTE THE ARCSINE IS FCSASN. THE FORMAT OF THE			*
		6526	*	CALLING SEQUENCE IS AS FOLLOWS:			*
		6527	*	B I\$CALL			*
		6528	*	DC AL2(VSFASN)			*
		6529	*	* THE ENTRY TO COMPUTE THE ARCCOSINE IS FCSACS. THE FORMAT OF THE			*
		6530	*	CALLING SEQUENCE IS AS FOLLOWS:			*
		6531	*	B I\$CALL			*
		6532	*	DC AL2(VSFACS)			*
		6533	*				*
		6534	*	*INPUT			*
		6535	*	* THE INPUT IS AN UNPACKED FLOATING POINT NUMBER IN THE FIRST			*
		6536	*	INTERPRETER STACK ELEMENT, IN STANDARD PRECISION.			*
		6537	*	* REGISTER 1 (@BR) POINTS TO THE PAGE BEGINNING.			*
		6538	*				*
		6539	*	*OUTPUT			*
		6540	*	* THE ANSWER IS PLACED IN THE FIRST STACK ELEMENT, IN UNPACKED			*
		6541	*	FLOATING POINT FORMAT.			*
		6542	*	* IN THE EVENT OF AN ERROR, AN ERROR CODE IS PLACED IN THE			*
		6543	*	ONE-BYTE INTERPRETER ERROR AREA, I\$ERRC.			*
		6544	*				*
		6545	*	*EXTERNAL REFERENCES			*
		6546	*	INTERPRETER STACK - FIRST TWO ELEMENTS, 16 BYTES			*
		6547	*	I\$STAK - CORE LOCATION OF ADDRESS OF THE INTERPRETER STACK			*
		6548	*	I\$ERRC - INTERPRETER ONE-BYTE ERROR LOCATION			*
		6549	*	I\$CALL - ENTRY TO THE PAGING ROUTINE			*
		6550	*	I\$RTRN - ENTRY TO THE PAGING RETURN ROUTINE			*
		6551	*	I\$FSUB - ENTRY TO THE SUBTRACT ROUTINE			*
		6552	*	I\$FMPY - ENTRY TO THE MULTIPLY ROUTINE			*
		6553	*	I\$FDVD - ENTRY TO THE DIVIDE ROUTINE			*
		6554	*	FRBSQR - SQUARE ROOT ROUTINE			*
		6555	*	FBSATN - ARCTANGENT ROUTINE			*
		6556	*				*
		6557	*	*EXITS, NORMAL			*
		6558	*	* EXIT IS BY BRANCHING TO I\$RTRN, THE ENTRY TO THE PAGING RETURN			*
		6559	*	ROUTINE.			*
		6560	*	* THE ANSWER IS LEFT IN THE FIRST STACK ELEMENT.			*
		6561	*				*
		6562	*	*EXITS, ERROR			*
		6563	*	* THE ERROR CODE IS PLACED IN I\$ERRC.			*
		6564	*	* RETURN IS BY BRANCHING TO I\$RTRN.			*
		6565	*				*

FCSASN - ARCSINE/ARCCOSINE FUNCTIONS - STD PREC

ERR LOC	OBJECT CODE	ADDR STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 71
		6566	*TABLES/WORK AREAS			*
		6567	* CONSTANTS AND WORK AREAS RESIDE AT THE END OF THE EXECUTABLE CODE.			*
		6568	*			*
		6569	*ATTRIBUTES			*
		6570	* REUSABLE, NATURALLY RELOCATABLE.			*
		6571	*			*
		6572	*CHARACTER CODE DEPENDENCY			*
		6573	* THE OPERATION OF THIS MODULE DEPENDS UPON A ZONED DECIMAL DIGIT			*
		6574	* BEING REPRESENTED WITH THE ZONE (FIRST FOUR BITS) BEING AN "F"			*
		6575	* FOR POSITIVE, AND A "D" FOR NEGATIVE. THIS PROPERTY IS USED			*
		6576	* AROUND FCS020, FCS050, FCS125, FCS250, AND FCS300.			*
		6577	* THE LOW ORDER FOUR BITS OF DECIMAL NUMBERS, WHEN CONSIDERED AS A			*
		6578	* BINARY INTEGER, IDENTIFY THE VALUE OF THE DIGIT.			*
		6579	*			*
		6580	*NOTES			*
		6581	* ERROR PROCEDURES			*
		6582	* THE ERROR CODE IS SET, AND CONTROL RETURNED.			*
		6583	*			*
		6584	* REGISTER USAGE			*
		6585	* * @BR IS USED AS THE SASE REGISTER.			*
		6586	* * @XR IS USED TO REFERENCE THE STACK.			*
		6587	*			*
		6588	* SAVED/RESTORED AREAS			*
		6589	* NONE			*
		6590	*			*
		6591	* MODIFICATION CONSIDERATIONS			*
		6592	* NONE			*
		6593	*			*
		6594	* REQUIRED MODULES			*
		6595	* @SYSEQ - COMMON SYSTEM EQUATES			*
		6596	* @ERMEQ - ERROR MESSAGE EQUATES			*
		6597	* \$V\$EQU - FIXED EQUATES			*
		6598	* \$B@EQU - COMPILER SYSYFM EQUATES			*
		6599	* \$I\$EQU - FIXED EQUATES			*
		6600	* \$I@SEQ - STANDARD PRECISION EXECUTION EQUATES			*
		6601	*			*
		6602	* OTHER			*
		6603	* NONE			*
		6604	*****			*

FCSASN - ARCSINE/ARCCOSINE FUNCTIONS - STD PREC

```

ERR LOC  OBJECT CODE      ADDR STMT SOURCE STATEMENT                                VER 15, MOD 00  31/05/21  PAGE  72

        6606 *****
        6607 * FCSASN - S/3.7 ARCSINE/ARCCOSINE FUNCTION ROUTINE - STANDARD PREC *
        6608 *****
        6609 *
1400      6610      ORG      *,B@LVPG,0                                PAGE BOUNDARY
        6611      USING *,@BR
        6612 *
        6613 * EXECUTION ENTRY TO FCSACS
        6614 *
1400 35 02 0D4E      6615 FCSACS L      I$STAK,@XR                      SET INDEX REG TO POINT TO STACK
1404 7C 00 E6      6616      MVI      FCSSWH(,@BR),@ZERO              SET SWITCH = 0 FOR ARCCOSINE
1407 7C E9 54      6617      MVI      FCS150+@Q(,@BR),@@E784        SET ERROR MESSAGE FOR ARCCOSINE
140A BD 79 00      6618      CLI      I@1SE1+I@DEXP(,@XR),B@NXZR-I@PREC  IS ARG < 10**-7 ?
140D D0 02 45      6619      BNL      FCS100(,@BR)                    IF NOT, CONTINUE WITH ROUTINE
        6620 *
        6621 * IF THE ARGUMENT IS LESS THAN 10**-7, THE ARCCOSINE(ARG) = PI/2.
        6622 *
1410 D0 87 3B      6623      B      FCS050(,@BR)                    BRANCH TO SET ANSWER TO PI/2
        6624 *
        6625 * EXECUTION ENTRY TO FCSASN
        6626 *
1413 35 02 0D4E      6627 FCSASN L      I$STAK,@XR                      SET INDEX REG TO POINT TO STACK
1417 7C 01 E6      6628      MVI      FCSSWH(,@BR),@B1              SET SWITCH = 1 FOR ARCSINE
141A 7C E8 54      6629      MVI      FCS150+@Q(,@BR),@@E783        SET ERROR MESSAGE FOR ARCSINE
        6630 *
        6631 * IF THE ARG < .001, THE ARCSINE(ARG) = ARG.
        6632 *
141D BD 7E 00      6633      CLI      I@1SE1+I@DEXP(,@XR),B@NXZR-2  IS ARG < .001 ?
1420 D0 82 C9      6634      BL      FCS900(,@BR)                    IF LESS, RETURN
        6635 *
        6636 * IF ARG = 1, THE ARCSINE(ARG) = PI/2 * SGN(ARG)
        6637 *
1423 68 00 D4 07      6638      MZZ      FCSP12(,@BR),I@1SE1+I@SIGN(,@XR)  SET PI/2 TO SIGN OF ARG
1427 BA F0 07      6639      SBN      I@1SE1+I@SIGN(,@XR),B@ZPOS      SET ARG POSITIVE
142A 9D 07 07 DC      6640      CLC      I@RSE1(I@LUFV,@XR),FCSONE(,@BR)  IS ARG = 1 ?
142E D0 81 3B      6641      BE      FCS050(,@BR)                    IF EQUAL, BRANCH
1431 58 00 DD D4      6642      MZZ      FCSSGN(,@BR),FCSP12(,@BR)  SAVE ARGUMENT SIGN
1435 7A F0 D4      6643      SBN      FCSP12(,@BR),B@ZPOS            KEEP PI/2 POSITIVE
1438 D0 87 4C      6644      B      FCS125(,@BR)                    CONTINUE
        6645 *
        6646 * ARG = 1, SO ARCSINE(ARG) = PI/2 * SGN(ARG)
        6647 *
143B 9C 07 07 D4      6648 FCS050 MVC      I@RSE1(I@LUFV,@XR),FCSP12(,@BR)  SET ANSWER TO PI/2
143F 7A F0 D4      6649      SBN      FCSP12(,@BR),B@ZPOS            KEEP PI/2 POSITIVE
1442 D0 87 C9      6650      B      FCS900(,@BR)                    RETURN
        6651 *
        6652 * IF ARG > 1 THE ARGUMENT IS NOT IN THE PROPER RANGE. THIS IS AN
        6653 * ERROR, AND CONTROL IS RETURNED WITH THE ERROR MESSAGE.
        6654 *
1445 68 00 DD 07      6655 FCS100 MZZ      FCSSGN(,@BR),I@1SE1+I@SIGN(,@XR)  SAVE ARGUMENT SIGN
1449 BA F0 07      6656      SBN      I@1SE1+I@SIGN(,@XR),B@ZPOS      SET ARGUMENT POSITIVE
144C 9D 07 07 DC      6657 FCS125 CLC      I@RSE1(I@LUFV,@XR),FCSONE(,@BR)  IS ARGUMENT GREATER 1 ?
1450 D0 04 5A      6658      BNH      FCS200(,@BR)                    IF NOT, BRANCH TO CONTINUE
        6659 *
        6660 * ARGUMENT > 1
        6661 *

```

FCSASN - ARCSINE/ARCCOSINE FUNCTIONS - STD PREC

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 73
1453	3C	00	0CBC		6662	FCS150	MVI I\$ERRC,*-*			SET ERROR CODE
1457	D0	87	C9		6663		B FCS900(,@BR)			RETURN
					6664	*				
					6665	*	BEGIN THE ROUTINE			
					6666	*				
145A	98	00	07 DD		6667	FCS200	MZZ I@1SE1+I@SIGN(,@XR),FCSSGN(,@BR) RESTORE SIGN			
145E	6C	07	E5 07		6668		MVC FCSARG(I@LUFV,@BR),I@RSE1(,@XR) SAVE THE ARG			
1462	AC	07	0F 07		6669		MVC I@RSE2(I@LUFV,@XR),I@RSE1(,@XR) PREPARE TO SQUARE THE ARG			
1466	C0	87	082A		6670		B I\$FMPY SQUARE THE ARGS			
146A	9C	07	0F DC		6671		MVC I@RSE2(I@LUFV,@XR),FCSONE(,@BR) PREPARE FOR 1 - ARS**2			
146E	C0	87	0751		6672		B I\$FSUB SUBTRACT (ARG*ARG) - 1			
1472	BA	F0	07		6673		SBN I@1SE1+I@SIGN(,@XR),B@ZPOS 1 - (ARG*ARG)			
1475	C0	87	12B1		6674		B I\$CALL TAKE THE SQUARE ROOT			
1479	0900			147A	6675		DC AL(@VADDR)(V\$FSQR) * OF (1 - ARG**2)			
					6676	*				
					6677	*	IF THE SWITCH IS ZERO, WE ARE LOOKING FOR THE ARCCOSINE, AND FOR A			
					6678	*	POSITIVE ARGUMENT WE DIVIDE (SQR(1-ARG**2))/ARG. FOR NEGATIVE VALUES			
					6679	*	WE USE THE IDENTITY: ACS(ARG) = PI/2 - ASN(ARG).			
					6680	*	IF THE SWITCH IS NOT ZERO, WE DIVIDE ARG/(SQR(1-ARG**2)).			
					6681	*				
147B	7D	00	E6		6682		CLI FCSSWH(,@BR),@ZERO IS SWITCH ZERO ?			
147E	D0	01	95		6683		BNE FCS300(,@BR) IF NOT, BRANCH			
1481	78	F0	E5		6684		TBN FCSARG(,@BR),B@ZPOS IS THE ARG POSITIVE ?			
1484	D0	90	95		6685		BF FCS300(,@BR) IF NOT, BRANCH			
					6686	*				
					6687	*	WE ARE LOOKING FOR THE ARCCOSINE OF A POSITIVE ARGUMENT			
					6688	*				
1487	7C	01	E6		6689		MVI FCSSWH(,@BR),@B1 SET SWITCH TO DENOTE ARG			
148A	9C	07	0F E5		6690		MVC I@RSE2(I@LUFV,@XR),FCSARG(,@BR) GET THE ARG			
148E	C0	87	0919		6691		B I\$FDVD DIVIDE (SQR(1-ARG**2))/ARG			
1492	D0	87	B1		6692		B FCS400(,@BR) BRANCH TO CONTINUE			
					6693	*				
					6694	*	IF THE SQR(1 - ARG**2) IS ZERO, THEN WE ARE LOOKING FOR THE			
					6695	*	ARCCOSINE OF -1. THIS IS EQUAL TO PI, SO WE SET THE STACK TO			
					6696	*	ARRIVE AT THIS ANSWER.			
					6697	*				
1495	BD	F0	01		6698	FCS300	CLI I@1SE1+I@MANL(,@XR),@DZERO IS SQR() ZERO ?			
1498	D0	01	A5		6699		BNE FCS350(,@BR) IF NOT CONTINUE			
149B	9C	07	07 D4		6700		MVC I@RSE1(I@LUFV,@XR),FCSPI2(,@BR) GET PI/2			
149F	BB	20	07		6701		SBF I@RSE1(,@XR),X'20' MAKE PI/2 MINUS			
14A2	D0	87	BD		6702		B FCS500(,@BR) BRANCH TO GET PI			
					6703	*				
					6704	*	COMPUTE THE ARCSINE			
					6705	*				
14A5	AC	07	0F 07		6706	FCS350	MVC I@RSE2(I@LUFV,@XR),I@RSE1(,@XR) MAKE SQR() THE DENOMINATR			
14A9	9C	07	07 E5		6707		MVC I@RSE1(I@LUFV,@XR),FCSARG(,@BR) GET ARS AS NUMERATOR			
14AD	C0	87	0919		6708		B I\$FDVD DIVIDE ARG/(SQR(1 - ARS**2))			
					6709	*				
					6710	*	TAKE THE ARCTANGENT OF THIS NEW ARGUMENT			
					6711	*				
14B1	C0	87	12B1		6712	FCS400	B I\$CALL			
14B5	1100			14B6	6713		DC AL(@VADDR)(V\$FATN)			
					6714	*				
					6715	*	IF THE SWITCH IS ZERO WE HAVE THE ARCCOSINE OF A NEGATIVE VALUE,			
					6716	*	WHICH IS COMPUTED BY THE IDENTITY: ACS(X) = PI/2 - ASN(X).			
					6717	*				

FCSASN - ARCSINE/ARCCOSINE FUNCTIONS - STD PREC

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 74
	14B7	7D	00	E6	6718	CLI	FCSSWH(,@BR),@ZERO			IS SWITCH ZERO
	14BA	D0	01	C9	6719	BNE	FCS900(,@BR)			IT NOT WE ARE DONE, RETURN
	14BD	AC	07	0F 07	6720	FCS500 MVC	I@RSE2(I@LUFV,@XR),I@RSE1(,@XR)			PREDARE TO SUBTRACT
	14C1	9C	07	07 D4	6721	MVC	I@RSE1(I@LUFV,@XR),FCSPI2(,@BR)			GET PI/2
	14C5	C0	87	0751	6722	B	I\$FSUB			ARCCOS = PI/2 - ASN(ARG)
					6723	*				
					6724	*	RETURN TO CALLING ROUTINE			
					6725	*				
	14C9	C0	87	12D3	6726	FCS900 B	I\$RTRN			
					6727	*				
					6728	*	CONSTANTS AND WORK AREAS			
					6729	*				
	14CD	81			14CD	6730	DC	AL1(B@NXZR+1)		EXPONENT
	14CE	F1F5F7F0F7F9F6			14D4	6731	FCSPI2 DC	DL(I@PREC)'1570796'		PI/2
	14D5	81			14D5	6732	DC	AL1(B@NXZR+1)		EXPONENT
	14D6	F1F0F0F0F0F0F0			14DC	6733	FCSONE DC	DL(I@PREC)'1000000'		+ 1
	14DD				14DD	6734	FCSSGN DS	XL1		SIGN OF ARGUNENT SAVE AREA
	14DE				14E5	6735	FCSARG DS	XL(I@LUFV)		AREA TO SAVE ARGUMENT
	14E6				14E6	6736	FCSSWH DS	XL1		SWITCH
					6737	*				
					6738	*	END OF CODING			
					6739	*				

FCSASN - ARCSINE/ARCCOSINE FUNCTIONS - STD PREC

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 75
		6741		*****			
		6742	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
		6743	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
		6744	*				*
		6745		*****			*
		6746	*	*STATUS -			*
		6747	*	VERSION 1 MODIFICATION 0			*
		6748	*				*
		6749	*	*FUNCTION -			*
		6750	*	* FIND THE HYPERBOLIC SINE OF THE INPUT ARGUMENT			*
		6751	*	* FIND THE HYPERBOLIC COSINE OF THE INPUT ARGUMENT			*
		6752	*	* FIND THE HYPERBOLIC TANGENT OF THE INPUT ARGUMENT			*
		6753	*				*
		6754	*	*ENTRY -			*
		6755	*	* FHSHPN HAS THREE ENTRY POINTS			*
		6756	*	* CALLING SEQUENCES ARE			*
		6757	*	* FHSHPN			*
		6758	*	B I\$CALL			*
		6759	*	DC AL2(VSFHPN)			*
		6760	*	* FHSPCS			*
		6761	*	B I\$CALL			*
		6762	*	DC AL2(VSFPCS)			*
		6763	*	* FHSPTN			*
		6764	*	B I\$CALL			*
		6765	*	DC AL2(VSFPTN)			*
		6766	*				*
		6767	*	*INPUT -			*
		6768	*	* I\$STAK - STACK POINTER			*
		6769	*	* REGISTER @BR POINTS TO THE FIRST BYTE IN THE PAGE			*
		6770	*	* STACK VALUE - FIRST VALUE IN THE STACK			*
		6771	*				*
		6772	*	*OUTPUT -			*
		6773	*	* THE RESULT IS PLACE IN THE FIRST STACK ELEMENT			*
		6774	*				*
		6775	*	*EXTERNAL REFERENCES -			*
		6776	*	I\$STAK - STACK POINTER			*
		6777	*	I\$RTRN - PAGING ROUTINE ENTRY TO RETURN			*
		6778	*	I\$CALL - PAGING ROUTINE ENTRY TO PASS EXECUTION CONTROL			*
		6779	*	I\$ERRC - INTERPRETER ERROR CODE PARAMETER			*
		6780	*	FGSEXP - ENTRY TO EXPONENTIATE			*
		6781	*	I\$FSUB - ENTRY TO SUBTRACT			*
		6782	*	I\$FADD - ENTRY TO ADD			*
		6783	*	I\$FMPY - ENTRY TO MULTIPLY			*
		6784	*	I\$FDVD - ENTRY TO DIVIDE			*
		6785	*				*
		6786	*	*EXITS, NORMAL -			*
		6787	*	FHSHPN HAS 1 NORMAL EXIT			*
		6788	*	I\$RTRN - PAGING ROUTINE ENTRY TO RETURN TO CALLING ROUTINE			*
		6789	*				*
		6790	*	*EXITS, ERROR -			*
		6791	*	I\$RTRN - WITH ERROR CODES			*
		6792	*	@@E789 - HSN FUNCTION--ARGUMENT > 225			*
		6793	*	@@E786 - HCS FUNCTION--ARGUMENT > 225			*
		6794	*				*
		6795	*	*TABLES/WORK AREAS -			*
		6796	*	THE CONSTANTS AND WORK AREAS RESIDE AT THE END OF EACH PAGE			*

6797	*	OF EXECUTABLE CODE AND ARE REFERENCED BY POR	*
6798	*		*
6799	*	*ATTRIBUTES -	*
6800	*	* REUSABLE	*
6801	*	* NATURALLY RELOCATEABLE	*
6802	*		*
6803	*	*CHARACTER CODE DEPENDENCY -	*
6804	*	N/A	*
6805	*		*
6806	*	*NOTES -	*
6807	*	ERROR PROCEDURES	*
6808	*	* ERROR CODE IS PLACED IN ISERRC (IF REQUIRED)	*
6809	*	* IMMEDIATE RETURN IS MADE	*
6810	*		*
6811	*	REGISTER USAGE	*
6812	*	* REGISTER @BR IS USED AS THE BASE REGISTER	*
6813	*	* RESISTER @XR IS USED AS THE STACK POINTER	*
6814	*	* AT EXIT @XR POINTS TO THE VALUE AT THE FIRST	*
6815	*	STACK POSITION	*
6816	*		*
6817	*	SAVED/RESTORED AREAS	*
6818	*	N/A	*
6819	*		*
6820	*	REQUIRED NODULES	*
6821	*	@SYSEQ - COMMON SYSTEM EQUATES	*
6822	*	\$V\$EQU - VIRTUAL ADDRESSES EQUATES	*
6823	*	\$B@EQU - COMPILER SYSTEM EQUATES	*
6824	*	\$I\$EQU - INTERPRETER FIXED EQUATES	*
6825	*	\$I@SEQ - PRECISION EXECUTION EQUATES	*
6826	*		*
6827	*	OTHER	*
6828	*	NONE	*
6829	*	*****	*

FHSHPN - HYPERBOLIC FUNCTIONS - STD PREC

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 77
		6831	*		
		6832	*****		
		6833	*		*
		6834	*****		
		6835	*		*
		6836	*	HYPERBOLIC FUNCTIONS, SINE, COSINE AND TANGENT (STANDARD PREC)	*
		6837	*		*
		6838	*****		
		6839	*		*
		6840	*****		
		6841	*		
		6842	*	ESTABLISH ADDRESSABILITY FOR VIRTUAL MEMORY PAGE	
		6843	*		
1500		6844	*FHSHPN VPAGE 0	SET PAGE ADDRESSABILITY	
		6845	ORG *,256,0	SET STARTING ADDRESS	
	1500	6846	FHSHPN EQU *	START OF PROGRAM CODING	
1401		6847	ORG *-255	RESET IAR TO PAGE	
1500		6848	ORG *,256,0	* BOUNDARY ADDRESS	
	1500	6849	USING *,@BR	SET PAGE BASE ADDRESS	
1500		6850	ORG FHSHPN	RESET STARTING ADDRESS	
		6851	*** END OF EXPANSION ***		
		6852	*		
		6853	*****		
		6854	*		*
		6855	*	HYPERBOLIC COSINE ROUTINE (STANDARD PRECISION)	*
		6856	*		*
		6857	*****		
		6858	*		
		6859	*	ENTER ROUTINE	
		6860	*		
	1500	6861	FHSHCS EQU *	ENTER HYPERBOLIC COSINE	
1500 35 02 0D4E		6862	L I\$STAK,@XR	CADDR STACK	
1504 7C EB 18		6863	MVI FHS017+@Q(,@BR),@E786	SET ERROR CODE TO USE	
1507 7C 87 3A		6864	MVI FHS030+@Q(,@BR),@UCB	SET BRANCH SWITCH	
		6865	*		
		6866	*	FIND THE EXPONTIAL OF STACK VALUE (X)	
		6867	*		
150A C0 87 12B1		6868	FHS010 B I\$CALL	OBTAIN E(X)	
150E 0500	150F	6869	DC AL(@VADDR)(V\$FEXP)	VADDR PARAM	
1510 3D 00 0CBC		6870	FHS015 CLI I\$ERRC,I@NERR	DID AN ERROR OCCUR	
1514 F2 81 07		6871	JE FHS020	NO, CONTINUE PROC	
1517 3C 00 0CBC		6872	FHS017 MVI I\$ERRC,*-*	SET ERROR CODE	
151B F2 87 33		6873	J FHS050	TO FIT	
		6874	*		
		6875	*	FIND TIE RECIPROCAL OF E(X)	
		6876	*		
151E 6C 07 F7 07		6877	FHS020 MVC FHSWK1(I@LUFV,@BR),I@RSE1(,@XR)	SAVE 1ST STACK ELEMENT	
1522 9C 07 07 EF		6878	MVC I@RSE1(I@LUFV,@XR),FHSONE(,@BR)	PLACE +1 IN STACK	
1526 9C 07 0F F7		6879	MVC I@RSE2(I@LUFV,@XR),FHSWK1(,@BR)	PLACE E(X) IN STACK	
152A C0 87 0919		6880	FHS025 B I\$FDVD	FIND RECIPROCAL OF E(V)	
152E 3D 00 0CBC		6881	CLI I\$ERRC,I@NERR	DID AN ERROR OCCUR ?	
1532 D0 01 17		6882	BNE FHS017(,@BR)	YES, SET ERROR CODE	
1535 9C 07 0F F7		6883	MVC I@RSE2(I@LUFV,@XR),FHSWK1(,@BR)	PLACE E(X) IN STACK	
1539 F2 00 03		6884	FHS030 JC FHS035,*-*	BR IF IN SINE RTN	
153C BB 20 07		6885	SBF I@1SE1+I@SIGN(,@XR),X'20'	SET SIGN NEG	
		6886	*		

FHSHFN - HYPERBOLIC FUNCTIONS - STD PREC

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 78
			6887	* ADD E(X) AND STACK VALUE			
			6888	*			
153F	C0 87 075D		6889	FHS035 B I\$FADD			ADD E(X) TO STACK VALUE
			6890	*			
			6891	* MULTIPLY STACK VALUE BY FIVE TENTHS			
			6892	*			
1543	96 50 0F E7		6893	FHS040 AZ I@RSE2(I@APRC,@XR),FHSD00(1,@BR)			SET THE 2ND STACK VALUE
1547	BC 80 08		6894	MVI I@1SE2(,@XR),B@NXZR			* TO A PLUS FIVE
154A	BC F5 09		6895	MVI I@1SE2+1(,@XR),B@DEC5			* TENTHS
154D	C0 87 082A		6896	FHS045 B I\$FMPY			MULTIPLY STACK VALLES
			6897	*			
			6898	* RETURN			
			6899	*			
1551	C0 87 12D3		6900	FHS050 B I\$RTRN			RETURN

FHSHFN - HYPERBOLIC FUNCTIONS - STD PREC

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE	79
					6902	*****					
					6903	*					*
					6904	* HYPERBOLIC SINE ROUTINE (STANDARD PRECISION)					*
					6905	*					*
					6906	*****					*
1557					6907	ORG	FHSHFN+X'0057'				START OF HYPERBOLIC SINE
					6908	*					
					6909	* ENTER ROUTINE					
					6910	*					
				1557	6911	FHSHSN EQU	*				ENTER HYPERBOLIC SINE
1557	35	02	0D4E		6912	L	I\$STAK,@XR				CADDR STACK
155B	7C	EA	18		6913	MVI	FHS017+@Q(,@BR),@@E785				SET ERROR CODE TO USE
155E	7C	80	3A		6914	MVI	FHS030+@Q(,@BR),@NOP				SET BRANCH S6
					6915	*					
					6916	* TEST IF STACK VALUE GREATER THAN .1					
					6917	*					
1561	BD	80	00		6918	FHS100 CLI	I@UEXP(,@XR),B@NXZR				VALUE LT .1
1564	D0	02	0A		6919	BNL	FHS010(,@BR)				NO, CONTINUE PROC
1567	BD	62	00		6920	CLI	I@1SE1+I@DEXP(,@XR),B@NXZR-30				IS VALUE > 1E-30 ?
156A	D0	04	51		6921	BNH	FHS050(,@BR)				IF NOT, ANSWER = ARGUMENT
					6922	*					
					6923	* BRANCH TO ALTERNATE ROUTINE TO SAVE SIGNIFICANT DIGITS					
					6924	*					
156D	C0	87	12B1		6925	FHS170 B	I\$CALL				FUNCTION D(X)
1571	1600			1572	6926	DC	AL(@VADDR)(FHS175)				VADDR PARAM
					6927	*					
					6928	* COMPLETE FUNCTION PROCESSING					
					6929	*					
1573	D0	87	43		6930	FHS230 B	FHS040(,@BR)				DIVIDE BY 2
					6931	*					
					6932	*****					*
					6933	*					*
					6934	* HYPERBOLIC TANGENT ROUTINE (STANDARD PRECISION)					*
					6935	*					*
					6936	*****					*
					6937	*					
1593					6938	ORG	FHSHFN+X'0093'				START OF HYPERBOLIC TANGENT
					6939	*					
					6940	* ENTER ROUTINE					
					6941	*					
				1593	6942	FHSHTN EQU	*				ENTER HYPERBOLIC TANGENT
1593	35	02	0D4E		6943	L	I\$STAK,@XR				CADDR STACK
					6944	*					
					6945	* TEST FOR A STACK VALUE GREATER THAN 100					
					6946	*					
1597	BD	83	00		6947	FHS300 CLI	I@UEXP(,@XR),B@NXZR+3				OVER 100 ?
159A	F2	82	0A		6948	JL	FHS305				NO, CONTINUE
159D	9C	06	06 EE		6949	MVC	I@RSE1-1(I@PREC,@XR),FHSONE-1(,@BR)				STACK 1 EXCEPT SIGN
15A1	BB	0F	07		6950	SBF	I@RSE1(,@XR),FHSMSK				SET NUMERIC ZONE BITS OFF
15A4	D0	87	51		6951	B	FHS050(,@BR)				EXIT RTN
					6952	*					
					6953	* DOUBLE STACK VALUE (X)					
					6954	*					
15A7	AC	07	0F 07		6955	FHS305 MVC	I@RSE2(I@LUFV,@XR),I@RSE1(,@XR)				SHIFT VALUE OVER
15AB	C0	87	075D		6956	B	I\$FADD				ADD STACK VALUES
					6957	*					

FHSHPN - HYPERBOLIC FUNCTIONS - STD PREC

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 80
					6958	*	TEST IF STACK VALUE GREATER THAN .1			
					6959	*				
	15AF	BD	80 00		6960	FHS310	CLI I@UEXP(, @XR), B@NXZR			VALUE LT .1 ?
	15B2	F2	82 28		6961		JL FHS370			YES, COMPUTE NEW ARG
					6962	*				
					6963	*	FIND THE EXPONENTIAL OF STACK VALUE (X)			
					6964	*				
	15B5	C0	87 12B1		6965	FHS320	B I\$CALL			OBTAIN E(X)
	15B9	0500		15BA	6966		DC AL(@VADDR) (V\$FEXP)			VADDR PARAM
					6967	*				
					6968	*	INCREMENT STACK VALUE BY +1			
					6969	*				
	15BB	9C	07 0F EF		6970	FHS330	MVC I@RSE2(I@LUFV, @XR), FHSONE(, @BR)			PLACE +1 IN STACK
	15BF	C0	87 075D		6971		B I\$FADD			ADD STACK VALUES
	15C3	6C	07 F7 07		6972		MVC FHSWK1(I@LUFV, @BR), I@RSE1(, @XR)			SAVE STACK VALUE
					6973	*				
					6974	*	DECREMENT STACK VALUE BY +2			
					6975	*				
	15C7	9C	07 0F EF		6976	FHS340	MVC I@RSE2(I@LUFV, @XR), FHSONE(, @BR)			MOVE A +1 TO STACK
	15CB	BC	F2 09		6977		MVI I@1SE2+I@UMN1(, @XR), B@DEC2			CHANGE VALUE TO +2
	15CE	C0	87 0751		6978		B I\$FSUB			SUBTRACT STACK VALUES
					6979	*				
					6980	*	DIVIDE THE NEW STACK VALUE BY THE SAVED VALUE			
					6981	*				
	15D2	9C	07 0F F7		6982	FHS350	MVC I@RSE2(I@LUFV, @XR), FHSWK1(, @BR)			MOVE SAVED VALUE TO STACK
	15D6	C0	87 0919		6983		B I\$FDVD			DIVIDE STACK VALUES
					6984	*				
					6985	*	RETURN			
					6986	*				
	15DA	D0	87 E3		6987	FHS360	B FHS400(, @BR)			GO RETURN
					6988	*				
					6989	*	BRANCH TO ALTERNATE ROUTINE TO SAVE SIGNIFICANT DIGITS			
					6990	*				
	15DD	C0	87 12B1		6991	FHS370	B I\$CALL			COMPUTE NEW ARG
	15E1	1627		15E2	6992		DC AL(@VADDR) (FHS380)			VADDR PARAM
					6993	*				
					6994	*	RETURN			
					6995	*				
	15E3	C0	87 12D3		6996	FHS400	B I\$RTRN			RETURN
					6997	*				
					6998	*****				
					6999	*				*
					7000	*	FHSHPN EQUATES, CONSTANTS AND WORK AREAS			*
					7001	*				*
					7002	*****				
					7003	*				
					7004	*	FHSHPN EQUATES			
					7005	*				
				000F	7006	FHSMSK	EQU X'0F'			TO SET NUMERIC ZONE BITS ORR
					7007	*				
					7008	*	FHSHPN CONSTANTS			
					7009	*				
	15E7	F0		15E7	7010	FHSD00	DC DL1'0'			DECIMAL 0 TO PROPAGATE
	15E8	81		15E8	7011		DC AL1(B@NXZR+@B1)			CONSTANT OF PLUS ONE FOR USE
	15E9	F1F0F0F0F0F0F0		15EF	7012	FHSONE	DC DL(I@PREC)'1000000'			* IN ARITHMETIC FUNCTIONS
					7013	*				

[illegible]

FHSHFN - HYPERBOLIC FUNCTIONS - STD PREC

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE	82
					7018	*					
					7019	*****					
					7020	*					*
					7021	*****					*
					7022	*					*
					7023	* SECOND VM INTERFACE HYPERBOLIC FUNCTION PAGE					*
					7024	*					*
					7025	*****					*
					7026	*					*
					7027	*****					*
					7028	*					*
					7029	* SET ADDRESSABILITY OF PAGE					*
					7030	*					*
					7031	*FHSPG2 VPFGE 0	SET PAGE ADDRESSABILITY				*
1600					7032	ORG *,256,0	SET STARTING ADDRESS				*
				1600	7033	FHSPG2 EQU *	START OF PROGRAM CODING				*
1501					7034	ORG *-255	RESET IAR TO PAGE				*
1600					7035	ORG *,256,0	* BOUNDARY ADDRESS				*
				1600	7036	USING *,@BR	SET PAGE BASE ADDRESS				*
1600					7037	ORG FHSPG2	RESET STARTING ADDRESS				*
					7038	*** END OF EXPANSION ***					*
					7039	*					*
					7040	*****					*
					7041	*					*
					7042	* HYPERBOLIC SINE ALTERNATE FUNCTION ROUTINE					*
					7043	*					*
					7044	*****					*
					7045	*					*
					7046	* FIND ALTERNATE ARGUMENT					*
					7047	*					*
1600	D0	87	48		7048	FHS175 B FHS800(,@BR)	TO FUNCTION RTN				*
					7049	*					*
					7050	* INCREMENT STACK VALUE BY +1					*
					7051	*					*
1603	6C	07	D8 07		7052	FHS180 MVC FHSWK2(I@LUFV,@BR),I@RSE1(,@XR)	SAVE STACK VALUE				*
1607	9C	07	0F B8		7053	MVC I@RSE2(I@LUFV,@XR),FHSI01(,@BR)	PLACE +1 IN STACK				*
160B	C0	87	075D		7054	FHS190 B I\$FADD	ADD STACK VALUES				*
					7055	*					*
					7056	* DIVIDE SAVED VALUE BY STACK VALUE					*
					7057	*					*
160F	AC	07	0F 07		7058	FHS200 MVC I@RSE2(I@LUFV,@XR),I@RSE1(,@XR)	SHIFT VALUE IN STACK				*
1613	9C	07	07 D8		7059	MVC I@RSE1(I@LUFV,@XR),FHSWK2(,@BR)	RESTORE SAVED VALUE				*
1617	C0	87	0919		7060	FHS210 B I\$FDVD	DIVIDE STACK VALUES				*
					7061	*					*
					7062	* ADD SAVED VALUE TO STACK VALUE					*
					7063	*					*
161B	9C	07	0F D8		7064	FHS220 MVC I@RSE2(I@LUFV,@XR),FHSWK2(,@BR)	SAVED VALUE TO STACK				*
161F	C0	87	075D		7065	B I\$FADD	ADD STACK VALUES				*
					7066	*					*
					7067	* RETURN FHS960(,@BR)					*
					7068	*					*
1623	C0	87	12D3		7069	B I\$RTRN	RETURN				*

FHSHEFN - HYPERBOLIC FUNCTIONS - STD PREC

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 83
		7071	*				
		7072	*	*****			
		7073	*				*
		7074	*	HYPERBOLIC TANGENT ALTERNATE FUNCTION ROUTINE			*
		7075	*				*
		7076	*	*****			
		7077	*				
		7078	*	FIND ALTERNATE ARGUMENT			
		7079	*				
1627	D0 87 48	7080	FHS380 B	FHS800(,@BR)		TO FUNCTION RTN	
		7081	*				
		7082	*	ADD +2 TO STACK VALUE			
		7083	*				
162A	6C 07 D8 07	7084	FHS385 MVC	FHSWK2(I@LUFV,@BR),I@RSE1(,@XR)		SAVE STACK VALUE	
162E	9C 07 0F B8	7085		MVC I@RSE2(I@LUFV,@XR),FHSI01(,@BR)		MOVE +1 TO STACK	
1632	BC F2 09	7086		MVI I@1SE2+I@UMN1(,@XR),B@DEC2		CHANGE TO +2	
1635	C0 87 075D	7087		B I\$FADD		ADD STACK VALUES	
		7088	*				
		7089	*	DIVIDE SAVED VALUE BY STACK VALUE			
		7090	*				
1639	AC 07 0F 07	7091	FHS390 MVC	I@RSE2(I@LUFV,@XR),I@RSE1(,@XR)		SHIFT VALUE OVER	
163D	9C 07 07 D8	7092		MVC I@RSE1(I@LUFV,@XR),FHSWK2(,@BR)		RESTORE SAVED VALUE	
1641	C0 87 0919	7093		B I\$FDVD		DIVIDE STACK VALUES	
		7094	*				
		7095	*	RETURN			
		7096	*				
1645	D0 87 AD	7097		B FHS960(,@BR)		RETURN	

FHSHPN - HYPERBOLIC FUNCTIONS - STD PREC

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER	MOD	DATE	PAGE	84
					7099	*						
					7100	*****						
					7101	*						*
					7102	*****						*
					7103	*						*
					7104	*	FHSHPN AUXILIARY FUNCTION					*
					7105	*						*
					7106	*****						*
					7107	*						*
					7108	*****						*
					7109	*						*
					7110	*	ENTER ROUTINE AND SAVE STACK VALUE					*
					7111	*						*
1648	74	08	AC		7112	FHS800	ST FHS890+@OP1(,@BR),@ARR SAVE RETURN ADDR					
					7113	*						
					7114	*	TEST (X) FOR A VALUE < 1E-30					
					7115	*						
164B	BD	62	00		7116	FHS805	CLI I@UEXP(,@XR),B@NXZR-30 IS VALUE < 1E-30 ?					
164E	F2	04	58		7117	JNH	FHS890 RETURN, ANSWER IS ARGUMENT					
1651	6C	07	D8 07		7118	MVC	FHSWK2(I@LUFV,@BR),I@RSE1(,@XR) SAVE STACK VALUE (X)					
					7119	*						
					7120	*	MULTIPLY STACK VALUE BY ITS SELF					
					7121	*						
1655	AC	07	0F 07		7122	FHS810	MVC I@RSE2(I@LUFV,@XR),I@RSE1(,@XR) SHIFT VALUE OVER					
1659	C0	87	082A		7123	B	I\$FMPY MULTIPLY STACK VALUES					
165D	6C	07	E0 07		7124	MVC	FHSWK3(I@LUFV,@BR),I@RSE1(,@XR) SAVE STACK VALUE (U)					
					7125	*						
					7126	*	MULTIPLY STACK VALUE BY CONSTANT B					
					7127	*						
1661	9C	07	0F C8		7128	FHS820	MVC I@RSE2(I@LUFV,@XR),FHSKNB(,@BR) STACK B IN 2ND POS					
1665	C0	87	082A		7129	B	I\$FMPY MULTIPLY STACK VALUES					
					7130	*						
					7131	*	ADD CONSTANT A TO STACK VALUE					
					7132	*						
1669	9C	07	0F C0		7133	FHS830	MVC I@RSE2(I@LUFV,@XR),FHSKNA(,@BR) STACK A IN 2ND POS					
166D	C0	87	075D		7134	B	I\$FADD ADD VALUES					
					7135	*						
					7136	*	MULTIPLY STACK VALUE BY SAVED VALUE (X)					
					7137	*						
1671	9C	07	0F D8		7138	FHS840	MVC I@RSE2(I@LUFV,@XR),FHSWK2(,@BR) STACK SAVED VALUE (X)					
1675	C0	87	082A		7139	B	I\$FMPY MULTIPLY VALUES					
1679	6C	07	D8 07		7140	MVC	FHSWK2(I@LUFV,@BR),I@RSE1(,@XR) SAVE STACK VALUE (P)					
					7141	*						
					7142	*	ADD SAVED VALUE (U) AND CONSTANT C					
					7143	*						
167D	9C	07	0F E0		7144	FHS850	MVC I@RSE2(I@LUFV,@XR),FHSWK3(,@BR) STACK U					
1681	9C	07	07 D0		7145	MVC	I@RSE1(I@LUFV,@XR),FHSKNC(,@BR) STACK C					
1685	C0	87	075D		7146	B	I\$FADD ADD VALUES					
					7147	*						
					7148	*	SUBTRACT SAVED VALUE P FROM STACK VALUE					
					7149	*						
1689	9C	07	0F D8		7150	GHS860	MVC I@RSE2(I@LUFV,@XR),FHSWK2(,@BR) STACK P					
168D	C0	87	0751		7151	B	I\$FSUB SUBTRACT VALUES					
1691	6C	07	E0 07		7152	MVC	FHSWK3(I@LUFV,@BR),I@RSE1(,@XR) SAVE Q					
					7153	*						
					7154	*	DOUBLE SAVED VALUE P					

FHSHFN - HYPERBOLIC FUNCTIONS - STD PREC

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE	85
					7155	*					
	1695	9C	07 07 D8		7156	FHS870 MVC	I@RSE1(I@LUFV,@XR),FHSWK2(,@BR) STACK P				
	1699	9C	07 0F D8		7157	MVC	I@RSE2(I@LUFV,@XR),FHSWK2(,@BR) STACK P				
	169D	C0	87 075D		7158	B	I\$FADD ADD VALUES				
					7159	*					
					7160	*	DIVIDE STACK VALUE BY SAVED VALUE P				
					7161	*					
	16A1	9C	07 0F E0		7162	FHS880 MVC	I@RSE2(I@LUFV,@XR),FHSWK3(,@BR) STACK Q				
	16A5	C0	87 0919		7163	B	I\$FDVD DIVIDE VALUES				
					7164	*					
					7165	*	RETURN				
					7166	*					
	16A9	C0	87 0000		7167	FHS890 B	*-* RETURN				
					7168	*					
					7169	*****					
					7170	*					*
					7171	*					*
					7172	*****					*
					7173	*					
	16AD	C0	87 12D3		7174	FHS960 B	I\$RTRN RETURN				
					7175	*					
					7176	*****					*
					7177	*					*
					7178	*	2ND FHSHFN PAGE EQUATES, CONSTANTS, AND WORK AREAS				*
					7179	*					*
					7180	*****					*
					7181	*					
					7182	*	2ND FHSHFN PAGE CONSTANTS				
					7183	*					
	16B1	81		16B1	7184	DC	AL1(B@NXZR+@B1) CONSTANT OF PLUS ONE FOR USE				
	16B2	F1F0F0F0F0F0F0		16B8	7185	FHSI01 DC	DL(I@PREC)'1000000' * IN ARITHMETIC FUNCTIONS				
	16B9	81		16B9	7186	DC	AL1(B@NXZR+@B1) CONSTANT A EXP				
	16BA	F5F0F0F0F0F9F0		16C0	7187	FHSKNA DC	DL(I@PREC)'5000090' CONSTANT A MANTISSA				
	16C1	7F		16C1	7188	DC	AL1(B@NXZR-@B1) CONSTANT B EXP				
	16C2	F8F3F3F2F5F8F6		16C8	7189	FHSKNB DC	DL(I@PREC)'8332586' CONSTANT B MANTISSA				
	16C9	82		16C9	7190	DC	AL1(B@NXZR+2) CONSTANT C EXP				
	16CA	F1F0F0F0F0F1F8		16D0	7191	FHSKNC DC	DL(I@PREC)'1000018' CONSTANT C MANTISSA				
					7192	*					
					7193	*	2ND FHSHFN PAGE WORK AREAS				
					7194	*					
	16D1			16D8	7195	FHSWK2 DS	CL(I@LUFV) WORK AREA 2				
	16D9			16E0	7196	FHSWK3 DS	CL(I@LUFV) WORK AREA 3				
					7197	*	END MODULE FHSHFN *****				
					7198	*					

FTBAD1 - SECANT & COSECANT FUNCTIONS

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 86
		7200		*****			
		7201	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
		7202	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
		7203	*				*
		7204		*****			*
		7205	*	*STATUS			*
		7206	*	VERSION 1 MODIFICATION 0			*
		7207	*				*
		7208	*	*FUNCTION -			*
		7209	*	* THE SECANT IS FOUND FOR THE STACK ELEMENT			*
		7210	*	* THE COSECANT IS FOUND FOR THE STACK ELEMENT			*
		7211	*				*
		7212	*	*ENTRY POINTS -			*
		7213	*	* FTSSEC HAS TWO ENTRY POINTS			*
		7214	*	* CALLING SEQUENCE ARE			*
		7215	*	* FTSSEC			*
		7216	*	B I\$CALL			*
		7217	*	DC AL2(V\$FSEC)			*
		7218	*	* FTSCSC			*
		7219	*	B I\$CALL			*
		7220	*	DC AL2(V\$FCSC)			*
		7221	*				*
		7222	*	*INPUT -			*
		7223	*	* I\$STAK - STACK POINTER			*
		7224	*	* REGISTER @BR POINTS TO THE FIRST BYTE IN THE PAGE			*
		7225	*	* STACK VALUE - 1ST VALUE IN THE STACK			*
		7226	*				*
		7227	*	*OUTPUT -			*
		7228	*	* THE RESULT IS PLACED IN THE 1ST STACK ELEMENT			*
		7229	*				*
		7230	*	*EXTERNAL REFERENCES -			*
		7231	*	I\$STAK - STACK POINTER			*
		7232	*	I\$RTRN - PAGING ROUTINE ENTRY TO RETURN			*
		7233	*	I\$CALL - PAGING ROUTINE ENTRY TO PASS EXECUTION CONTROL			*
		7234	*	I\$ERRC - INTERPRETER ERROR CODE PARAMETER			*
		7235	*	I\$FDVD - DIVIDE ENTRY POINT			*
		7236	*	FSSSIN - SINE ENTRY POINT			*
		7237	*	FSSCSN - COSINE ENTRY POINT			*
		7238	*				*
		7239	*	*EXITS, NORMAL -			*
		7240	*	* FTSSEC HAS 1 NORMAL EMIT			*
		7241	*	I\$RTRN - PAGING ROUTINE ENTRY TO RETURN TO CALLING ROUTINE			*
		7242	*				*
		7243	*	*EXITS, ERROR -			*
		7244	*	ISRTRN - HUH ERROR CODES			*
		7245	*	@@E797 - SEC FACTION ARGUMENT > 100			*
		7246	*	@@E781 - SEC FUNCTION RESULT GOES TO INFINITY			*
		7247	*	@@E798 - CSC FUNCTION ARGUMENT > 100			*
		7248	*	@@E782 - CSC FUNCTION RESULT GOES TO INFINITY			*
		7249	*				*
		7250	*	*TABLES/WORK AREAS -			*
		7251	*	* THE CONSTANTS AND WORK AREAS RESIDE AT THE END OF THE			*
		7252	*	EXECUTABLE CODE AND ARE REFERENCED BY @BR			*
		7253	*				*
		7254	*	*ATTRIBUTES -			*
		7255	*	* REUSABLE			*

FTBAD1 - SECANT & COSECANT FUNCTIONS									
ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE	87
			7256	*	* NATURALLY RELOCATABLE				*
			7257	*					*
			7258	*	CHARACTER CODE DEPENDENCE				*
			7259	*	NONE				*
			7260	*					*
			7261	*	NOTES -				*
			7262	*	ERROR PROCEDURES				*
			7263	*	* ERROR CODE IS PLACED IN I\$ERRC				*
			7264	*	* IMMEDIATE RETURN IS MADE				*
			7265	*					*
			7266	*	REGISTER USAGE				*
			7267	*	* REGISTER @BR IS USED AS THE BASE REGISTFR				*
			7268	*	* REGISTER @XR IS USED AS THE STACK POINTER				*
			7269	*	* AT EXIT @XR POINTS TO THE VALUE AT TIE FIRST				*
			7270	*	STACK POSITION				*
			7271	*					*
			7272	*	SAVED/RESTORED AREAS				*
			7273	*	N/A				*
			7274	*					*
			7275	*	REQUIRED MODULES				*
			7276	*	@SYSEQ - COMMON SYSTEM EQUATES				*
			7277	*	\$VSEQU - VIRTUAL ADDRESSES EQUATES				*
			7278	*	\$B@EQU - COMPILER SYSTEM EQUATES				*
			7279	*	\$I\$EQU - INTERPRETER FIXED EQUATES				*
			7280	*					*
			7281	*	OTHER				*
			7282	*	NONE				*
			7283	*	*****				*

FTBAD1 - SECANT & COSECANT FUNCTIONS									
ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE	87
			7256	*	* NATURALLY RELOCATABLE				*
			7257	*					*
			7258	*	CHARACTER CODE DEPENDENCE				*
			7259	*	NONE				*
			7260	*					*
			7261	*	NOTES -				*
			7262	*	ERROR PROCEDURES				*
			7263	*	* ERROR CODE IS PLACED IN I\$ERRC				*
			7264	*	* IMMEDIATE RETURN IS MADE				*
			7265	*					*
			7266	*	REGISTER USAGE				*
			7267	*	* REGISTER @BR IS USED AS THE BASE REGISTFR				*
			7268	*	* REGISTER @XR IS USED AS THE STACK POINTER				*
			7269	*	* AT EXIT @XR POINTS TO THE VALUE AT TIE FIRST				*
			7270	*	STACK POSITION				*
			7271	*					*
			7272	*	SAVED/RESTORED AREAS				*
			7273	*	N/A				*
			7274	*					*
			7275	*	REQUIRED MODULES				*
			7276	*	@SYSEQ - COMMON SYSTEM EQUATES				*
			7277	*	\$VSEQU - VIRTUAL ADDRESSES EQUATES				*
			7278	*	\$B@EQU - COMPILER SYSTEM EQUATES				*
			7279	*	\$I\$EQU - INTERPRETER FIXED EQUATES				*
			7280	*					*
			7281	*	OTHER				*
			7282	*	NONE				*
			7283	*	*****				*

FTBAD1 - SECANT & COSECANT FUNCTIONS									
ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE	87
			7256	*	* NATURALLY RELOCATABLE				*
			7257	*					*
			7258	*	CHARACTER CODE DEPENDENCE				*
			7259	*	NONE				*
			7260	*					*
			7261	*	NOTES -				*
			7262	*	ERROR PROCEDURES				*
			7263	*	* ERROR CODE IS PLACED IN I\$ERRC				*
			7264	*	* IMMEDIATE RETURN IS MADE				*
			7265	*					*
			7266	*	REGISTER USAGE				*
			7267	*	* REGISTER @BR IS USED AS THE BASE REGISTFR				*
			7268	*	* REGISTER @XR IS USED AS THE STACK POINTER				*
			7269	*	* AT EXIT @XR POINTS TO THE VALUE AT TIE FIRST				*
			7270	*	STACK POSITION				*
			7271	*					*
			7272	*	SAVED/RESTORED AREAS				*
			7273	*	N/A				*
			7274	*					*
			7275	*	REQUIRED MODULES				*
			7276	*	@SYSEQ - COMMON SYSTEM EQUATES				*
			7277	*	\$VSEQU - VIRTUAL ADDRESSES EQUATES				*
			7278	*	\$B@EQU - COMPILER SYSTEM EQUATES				*
			7279	*	\$I\$EQU - INTERPRETER FIXED EQUATES				*
			7280	*					*
			7281	*	OTHER				*
			7282	*	NONE				*
			7283	*	*****				*

FTBAD1 - SECANT & COSECANT FUNCTIONS

ERR	LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 88
				7285	*****	
				7286	*	*
				7287	*****	
				7288	*	*
				7289	* FTBSEC - SECANT AND COSECANT FUNCTIONS	*
				7290	*	*
				7291	*****	
				7292	*	*
				7293	*****	
				7294	*	
1700				7295	*FTBAD1 VPAGE 0	SET PAGE ADDRESSABILITY
				7296	ORG *,256,0	SET STARTING ADDRESS
	1700			7297	FTBAD1 EQU *	START OF PROGRAM CODING
1601				7298	ORG *-255	RESET IAR TO PAGE
1700				7299	ORG *,256,0	* BOUNDARY ADDRESS
	1700			7300	USING *,@BR	SET PAGE BASE ADDRESS
1700				7301	ORG FTBAD1	RESET STARTING ADDRESS
				7302	*** END OF EXPANSION ***	
				7303	*	
				7304	* ENTER FTBSEC - SECANT FUNCTION	
				7305	*	
			1700	7306	FTSSEC EQU *	SECANT FUNCTION ENTRY
				7307	*	
				7308	* OBTAIN THE COSINE OF THE STACKED DATA ELEMENT	
				7309	*	
1700 C0 87 12B1				7310	FTB100 B I\$CALL	LINK TO FIND COSINE OF STACKED
1704 0A00			1705	7311	DC AL(@VADDR)(V\$FCOS)	* VALUE
				7312	*	
				7313	* TEST FOR EXECUTION ERROR IN THE COSINE FUNCTION	
				7314	*	
1706 3D 00 0CBC				7315	FTB125 CLI I\$ERRC,I@NERR	DID A ERROR OCCUR
170A F2 81 07				7316	JE FTB150	NO, TEST FOR ZERO
170D 3C F3 0CBC				7317	MVI I\$ERRC,@@E797	SET ERROR CODE
1711 F2 87 48				7318	J FTB700	TO EXIT
				7319	*	
				7320	* TEST THE STACKED VALUE FOR A ZERO VALUE	
				7321	*	
1714 35 02 0D4E				7322	FTB150 L I\$STAK,@XR	CADDR STACKED VALUE
1718 BD F0 01				7323	FTB160 CLI I@UMN1(,@XR),B@DEC0	IS MANTISSA ZERO ?
171B F2 01 2C				7324	JNE FTB500	NO, FIND THE RECIPROCAL
171E 3C E6 0CBC				7325	FTB170 MVI I\$ERRC,@@E781	SET ERROR CODE
1722 F2 87 37				7326	J FTB700	TO EXIT
				7327	*	
				7328	* ENTER FTBSEC - COSECANT FUNCTION	
				7329	*	
			1725	7330	FTBCSC EQU *	COSECANT ENTRY POINT
				7331	*	
				7332	* OBTAIN SINE OF THE STACKED DATA ELEMENT	
				7333	*	
1725 C0 87 12B1				7334	FTB200 B I\$CALL	LINK TO FIND SINE OF STACKED
1729 0A1A			172A	7335	DC AL(@VADDR)(V\$FSIN)	* VALUE
				7336	*	
				7337	* TEST FOR EXECUTION ERROR IN THE SINE ROUTINE	
				7338	*	
172B 3D 00 0CBC				7339	FTB300 CLI I\$ERRC,I@NERR	DID A ERROR OCCUR ?
172F F2 81 07				7340	JE FTB400	NO, TFST FOR ZFRO

FTBAD1 - SECANT & COSECANT FUNCTIONS

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE	89
1732	3C	F4	0CBC	7341		MVI	I\$ERRC,@E798				SET ERROR
1736	F2	87	23	7342		J	FTB700				TO EXIT
				7343	*						
				7344	*	TEST	THE STACKED VALUE FOR A ZERO VALUE				
				7345	*						
1739	35	02	0D4E	7346	FTB400	L	I\$STAK,@XR				CADDR STACKED VALUE
173D	BD	F0	01	7347	FTB450	CLI	I@UMN1(,@XR),B@DEC0				IS MANTISSA ZERO ?
1740	F2	01	07	7348		JNE	FTB500				NO, FIND THE RECIPROCAL
1743	3C	E7	0CBC	7349	FTB475	MVI	I\$ERRC,@E782				SET OVERFLOW ERROR CODE
1747	F2	87	12	7350		J	FTB700				GO EXIT RTN
				7352	*****						
				7353	*						*
				7354	*	FIND	THE RECIPROCAL OF THE STACKED VALUE				*
				7355	*						*
				7356	*****						
				7357	*						
				7358	*	SHIFT	THE PRESENT STACKED VALUE TO THE SECOND STACK POSITION				
				7359	*						
174A	AC	07	0F 07	7360	FTB500	MVC	I@RSE2(,@XR),I@RSE1(I@LUFV,@XR)				TO 2ND STACK POSITION
				7361	*						
				7362	*	PLACE	THE FLOATING POINT VALUE OF ONE IN THE RIRST STACK POSITION				
				7363	*						
174E	94	50	07 60	7364	FTB550	ZAZ	I@UMNR(I@PREC-1,@XR),FTBCON(1,@BR)				ZERO MANTISSA
1752	BC	F1	01	7365		MVI	I@UMN1(,@XR),B@DEC1				SET 1ST MANTISSA BYTE TO ZERO
1755	BC	81	00	7366		MVI	I@UEXP(,@XR),B@NXZR+@B1				SET ELEMENT EXP TO +1
1758	C0	87	0919	7367	FTB600	B	I\$FDVD				FIND THE RECIPROCAL
				7368	*						
				7369	*	RETURN	TO THE CALLING ROUTINE				
				7370	*						
175C	C0	87	12D3	7371	FTB700	B	I\$RTRN				RETURN
				7373	*****						
				7374	*	FTBSEC	WORK AREA				*
				7375	*****						
				7376	*						
1760	F0			1760	7377	FTBCON	DC DL1'0'				DECIMAL 0 TO PROPAGATE
				7378	*						
				7379	*****						
				7380	*						
				7381	*		END OF ROUTINE FTBSEC	*****			

FABARS - ABSOLUTE VALUE FUNCTION

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 90
		7383		*****			
		7384	*	5703-XM1 COPYRIGHT IBM CORP, 1970			*
		7385	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
		7386	*				*
		7387		*****			*
		7388	*	*STATUS			*
		7389	*	VERSION 1 MODIFICATION 0			*
		7390	*				*
		7391	*	*FUNCTION -			*
		7392	*	SETS THE SIGN OF THE INPUT ARGUMENT TO PLUS AND RETURNS			*
		7393	*				*
		7394	*	*ENTRY POINTS -			*
		7395	*	* FABABS HAS ONLY 1 ENTRY - FABABS			*
		7396	*	* CALLING SEQUENCE IS			*
		7397	*	B I\$CALL			*
		7398	*	DC AL2(V\$FABS)			*
		7399	*				*
		7400	*	*INPUT -			*
		7401	*	* I\$STACK - 2 BYTES, CONTAINS THE CORE ADDRESS FIRST ELEMENT OF			*
		7402	*	THE STACK			*
		7403	*	* REGISTER @BR - CONTAINS CORE ADDRESS OF FIRST BYTE OF THE PAGE			*
		7404	*	* STACK VALUE - VALUE TO FIND THE ABSOLUTE			*
		7405	*				*
		7406	*	*OUTPUT -			*
		7407	*	* STACK VALUE - ABSOLUTE VALUE			*
		7408	*				*
		7409	*	*EXTERNAL REFERENCES -			*
		7410	*	I\$STAK - STACK POINTER			*
		7411	*	I\$RTRN - PAGING ROUTINE ENTRY TO RETURN			*
		7412	*				*
		7413	*	*EXITS, NORMAL -			*
		7414	*	* FABABS RETURNS TO THE PAGING ROUTINE TO RETURN TO THE CALLING			*
		7415	*	ROUTINE			*
		7416	*	I\$RTRN - AFTER FINDING ABSOLUTE VALUE			*
		7417	*				*
		7418	*	*EXITS, ERROR -			*
		7419	*	NONE			*
		7420	*				*
		7421	*	*TABLES/WORK AREAS -			*
		7422	*	NONE			*
		7423	*				*
		7424	*	*ATTRIBUTES -			*
		7425	*	* REUSABLE			*
		7426	*	* NATURALLY RELOCATABLE			*
		7427	*				*
		7428	*	*CHARACTER CODE DEPENDENCY -			*
		7429	*	N/A			*
		7430	*				*
		7431	*	*NOTES -			*
		7432	*	ERROR PROCEDURES			*
		7433	*	NONE			*
		7434	*				*
		7435	*	REGISTER			*
		7436	*	* REGISTER @BR IS USED AS THE BASE REGISTER			*
		7437	*	* REGISTER @XR IS USED TO INDEX THE STACK			*
		7438	*	* @XR IS LEFT POINTING TO THE 1ST STACK VALUE			*

FABARS - ABSOLUTE VALUE FUNCTION

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE	91	
					7439	*						*
					7440	*	SAVED/RESTORED AREAS					*
					7441	*	NONE					*
					7442	*						*
					7443	*	MODIFICATION CONSIDERATIONS					*
					7444	*	N/A					*
					7445	*						*
					7446	*	REQUIRED MODULES					*
					7447	*	@SYSEQ - COMMON SYSTEM EQUATES					*
					7448	*	\$B@EQU - COMPILER SYSTEM EQUATES					*
					7449	*	\$I\$EQU - INTERPRETER FIXED EQUATES					*
					7450	*	\$I@SEQ/\$I@LEQ - PRECISION EXECUTION EQUATES					*
					7451	*						*
					7452	*	OTHER					*
					7453	*	FASABS IS THE SECOND SHORTEST PROGRAM IN THE BASIC					*
					7454	*	SYSTEM					*
					7455	*	*****					*

FABARS - ABSOLUTE VALUE FUNCTION

ERR	LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 92
				7457	*****	
				7458	*	*
				7459	*****	
				7460	*	*
				7461	* FABABS - ABSOLUTE VALUE FUNCTION ROUTINE	*
				7462	*	*
				7463	*****	
				7464	*	*
				7465	*****	
				7466	*	
1761				7467	*FABAD1 VPAGE 0	SET PAGE ADDRESSABILITY
				7468	ORG *	SET STARTING ADDRESS
1662			1761	7469	FABAD1 EQU *	START OF PROGRAM CODING
1700				7470	ORG *-255	RESET IAR TO PAGE
				7471	ORG *,256,0	* BOUNDARY ADDRESS
			1700	7472	USING *,@BR	SET PAGE BASE ADDRESS
1761				7473	ORG FABAD1	RESET STARTING ADDRESS
				7474	*** END OF EXPANSION ***	
				7475	*	
				7476	* ENTER FABABS - ACCESS THE STACKED DATA ELEMENT	
				7477	*	
			1761	7478	FABABS EQU *	ENTER FABARS
1761 35 02 0D4E				7479	L I\$STAK,@XR	CADDR DATA ELEMENT
				7480	*	
				7481	* SET THE SIGN OF THE FLOATING POINT INPUT ARGUMENT POSITIVE	
				7482	*	
1765 BA F0 07				7483	FAB100 SBN I@SIGN(,@XR),B@ZPOS	SET SIGN POSITIVE
				7484	*	
				7485	* RETURN CONTROL TO THE CALLING ROUTINE	
				7486	*	
1768 C0 87 12D3				7487	B I\$RTRN	RETURN
				7488	*	
				7489	*****	
				7490	*	
				7491	*	END OF ROUTINE FABABS *****

FJBINT - INTEGER VALUE FUNCTION

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 93
		7493		*****	
		7494	*	5703-XM1 COPYRIGHT IBM CORP, 1970	*
		7495	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083	*
		7496	*		*
		7497		*****	
		7498	*	*STATUS	*
		7499	*	VERSION 1 MODIFICATION 0	*
		7500	*		*
		7501	*	*FUNCTION -	*
		7502	*	* OUTPUTS THE INTEGER PORTION OF THE INPUT ARGUMENT	*
		7503	*		*
		7504	*	*ENTRY POINTS	*
		7505	*	* FJBINT HAS ONLY 1 ENTRY POINT - FJBINT	*
		7506	*	* CALLING SEQUENCE IS	*
		7507	*	B I\$CALL	*
		7508	*	DC AL2(V\$FINT)	*
		7509	*		*
		7510	*	*INPUT -	*
		7511	*	* I\$STACK - 2 BYTES, CONTAINS THE CORE ADDRESS OF THE FIRST	*
		7512	*	ELEMENT IN THE STACK	*
		7513	*	* REGISTER @BR - CONTAINS CORE ADDRESS OF FIRST BYTE OF THE PAGE	*
		7514	*	* STACK VALUE - VALUE TO INTEGER PORTION	*
		7515	*		*
		7516	*	*OUTPUT -	*
		7517	*	* STACK VALUE - INTEGER VALUE	*
		7518	*		*
		7519	*	*EXTERNAL REFERENCES -	*
		7520	*	I\$STAK - STACK POINTER	*
		7521	*	I\$RTRN - PAGING ROUTINE ENTRY TO RETURN	*
		7522	*		*
		7523	*	*EXITS, NORMAL -	*
		7524	*	* FJBINT RETURNS TO THE PAGING ROUTINE TO RETURN TO THE CALLING	*
		7525	*	ROUTINE	*
		7526	*	I\$RTRN - AFTER FINDING INTEGER PORTION	*
		7527	*		*
		7528	*	*EXITS, ERROR -	*
		7529	*	NONE	*
		7530	*		*
		7531	*	*TABLES/WORK AREAS	*
		7532	*	NONE	*
		7533	*		*
		7534	*	*ATTRIBUTES	*
		7535	*	* REUSABLE	*
		7536	*	* NATURALLY RELOCATABLE	*
		7537	*		*
		7538	*	*CHARACTER CODE DEPENDENCY -	*
		7539	*	N/A	*
		7540	*		*
		7541	*	*NOTES	*
		7542	*	ERROR PROCEDURES	*
		7543	*	N/A	*
		7544	*		*
		7545	*	REGISTER USAGE	*
		7546	*	* REGISTER @BR IS USED AS THE BASE REGISTER	*
		7547	*	* REGISTER @XR IS USED TO INDEX THE STACK	*
		7548	*	* @XR IS LEFT POINTING TO THE 1ST STACK VALUE	*

FJBINT - INTEGER VALUE FUNCTION

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE	94
		7549	*					*
		7550	*	SAVED,RESTORED AREAS				*
		7551	*	NONE				*
		7552	*					*
		7553	*	MODIFICATION CONSIDERATIONS				*
		7554	*	NONE				*
		7555	*					*
		7556	*	REQUIRED MODULES				*
		7557	*	@SYSEQ - COMMON SYSTEM EQUATES				*
		7558	*	\$B@EQU - COMPILER SYSTEM EQUATES				*
		7559	*	\$I\$EQU - INTERPRETER FIXED EQUATES				*
		7560	*	\$I@SEQ/\$I@LEQ - PRECISION EXECUTION EQUATES				*
		7561	*					*
		7562	*	OTHER				*
		7563	*	NONE				*
		7564	*	*****				*

FJBINT - INTEGER VALUE FUNCTION

ERR	LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 95
			7566		*****	
			7567	*		*
			7568		*****	
			7569	*		*
			7570	*	FJBINT - INTEGER VALUE FUNCTION ROUTINE	*
			7571	*		*
			7572		*****	
			7573	*		*
			7574		*****	
			7575	*		
			7576	*	ENTER FJBINT	
			7577	*		
			176C 7578	FJBINT EQU *	FJBINT ENTRY PT	
			7579	*		
			7580	*	TEST IF THE STACKED ELEMENT HAS AN EXPONENT LESS THAN 1	
			7581	*		
176C	35	02 0D4E	7582	FJB100 L	I\$STAK,@XR CADDR OF STACKED ELEMENT	
1770	BD	81 00	7583	FJB150 CLI	I@UEXP(,@XR),B@NXZR+@B1 IS EXP LT 1 ?	
1773	F2	82 22	7584	JL	FJB600 YES, SET VALUE TO 0	
			7585	*		
			7586	*	IS ANY OF THE MANTISSA NON-INTEGERS	
			7587	*		
1776	BD	87 00	7588	FJB200 CLI	I@UEXP(,@XR),B@NXZR+I@PREC IS MANTISSA ALL INTEGER ?	
1779	F2	02 26	7589	JNL	FJB700 YES, EXIT RTN	
			7590	*		
			7591	*	ZERO THE NON-INTEGERS PORTION OF THE MANTISSA	
			7592	*		
177C	68	00 A6 07	7593	FJB300 MZZ	FJB\$GN(,@BR),I@SIGN(,@XR) SAVE SIGN	
1780	BC	F0 07	7594	MVI	I@UMNR(,@XR),B@DEC0 SET SIGN BYTE TO 0	
1783	7C	85 8E	7595	MVI	FJB400+@Q(,@BR),B@NXZR+I@PREC-2 INIT MOVE INST LNG	
1786	6F	00 8E 00	7596	SLC	FJB400+@Q(,@BR),I@UEXP(1,@XR) SUB EXP TO GET MOVE LNG	
178A	F2	82 04	7597	JL	FJB500 UNDERFLOW, SKIP MOVE	
178D	AC	00 06 07	7598	FJB400 MVC	I@UMNR-1(,@XR),I@UMNR(@VQ,@XR) SET NON-INTEGERS TO ZERO	
			7599	*		
			7600	*	RESTORE THE SIGN	
			7601	*		
1791	98	00 07 A6	7602	FJB500 MZZ	I@SIGN(,@XR),FJB\$GN(,@BR) RESTORE SIGN	
1795	F2	87 0A	7603	J	FJB700 EXIT RTN	
			7604	*		
			7605	*	SET STACKED MANTISSA TO ZERO AND RESET EXPONENT	
			7606	*		
1798	BC	F0 07	7607	FJB600 MVI	I@UMNR(,@XR),B@DEC0 SET THE STACKED VALUE MANTISSA	
179B	AC	05 06 07	7608	MVC	I@UMNR-1(,@XR),I@UMNR(I@PREC-1,@XR) * TO ZERO	
179F	BC	1E 00	7609	FJB650 MVI	I@UEXP(,@XR),B@NXLO SET VALUE EXP	
			7610	*		
			7611	*	RETURN TO THE CALLING ROUTINE	
			7612	*		
17A2	C0	87 12D3	7613	FJB700 B	I\$RTRN RETURN	
			7614	*		
			7615		*****	
			7616	*	FJBINT WORK AREA	*
			7617		*****	
			7618	*		
17A6			17A6 7619	FJB\$GN DS CL1	ELEMENT SIGN SAVE AREA	
			7620	*		
			7621		*****	

FJBINT - INTEGER VALUE FUNCTION

ERR LOC	OBJECT CODE	ADDR STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE	96
		7622 *					
		7623 *	END OF ROUTINE FJBINT *****				

FUBSGN - SIGN OF VALUE FUNCTION

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 97
		7625		*****	
		7626	*	5703-XM1 COPYRIGHT IBM CORP, 1970	*
		7627	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083	*
		7628	*		*
		7629		*****	
		7630	*	*STATUS	*
		7631	*	VERSION 1 MODIFICATION 0	*
		7632	*		*
		7633	*	*FUNCTION -	*
		7634	*	* FUBSGN REPLACES THE INPUT ARGUMENT WITH A +1, 0, -1 DEPENDING	*
		7635	*	ON WHETHER IT IS POSITIVE, ZERO OR NEGATIVE RESPECTIVELY.	*
		7636	*		*
		7637	*	*ENTRY POINTS -	*
		7638	*	* FUBSGN HAS 1 ENTRY POINT - FUBSGN	*
		7639	*	* CALLING SEQUENCE IS	*
		7640	*	B I\$CALL	*
		7641	*	DC AL2(V\$FSGN)	*
		7642	*		*
		7643	*	*INPUT -	*
		7644	*	* I\$STAK - STACK POINTER	*
		7645	*	* REGISTER @BR POINTS TO THE FIRST BYTE IN THE PAGE	*
		7646	*	* STACK VALLE - FIRST VALUE IN THE STACK	*
		7647	*		*
		7648	*	*OUTPUT -	*
		7649	*	THE RESULT IS PLACED IN THE FIRST STACK ELEMENT	*
		7650	*		*
		7651	*	*EXTERNAL REFERENCES -	*
		7652	*	I\$STAK - STACK POINTER	*
		7653	*	I\$RTRN - PAGING ROUTINE ENTRY TO RETURN	*
		7654	*		*
		7655	*	*EXITS, NORMAL -	*
		7656	*	* FUBSGN HAS ONLY 1 EXIT	*
		7657	*	I\$RTRN - PAGING ROUTINE ENTRY TO RETURN TO CALLING ROUTINE	*
		7658	*		*
		7659	*	*EXITS, ERROR -	*
		7660	*	N/A	*
		7661	*		*
		7662	*	*TABLES/WORK AREAS -	*
		7663	*	NONE	*
		7664	*		*
		7665	*	*ATTRIBUTES -	*
		7666	*	* REUSABLE	*
		7667	*	* NATURALLY RELOCATEABLE	*
		7668	*		*
		7669	*	*CHARACTER CODE DEPENDENCY -	*
		7670	*	N/A	*
		7671	*		*
		7672	*	*NOTES -	*
		7673	*	ERROR PROCEDURES	*
		7674	*	NONE	*
		7675	*		*
		7676	*	REGISTER USAGE	*
		7677	*	* REGISTER @BR IS USED AS THE BASE REGISTER	*
		7678	*	* REGISTER @XR IS USED TO INDEX THE STACK	*
		7679	*	* @XR IS LEFT POINTING TO THE 1ST STACK VALUE	*
		7680	*		*

FUBSGN - SIGN OF VALUE FUNCTION

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE	98
		7681	*	SAVED/RESTORED AREAS				*
		7682	*	NONE				*
		7683	*					*
		7684	*	REQUIRED NODULES				*
		7685	*	@SYSEQ - COMMON SYSTEM EQUATES				*
		7686	*	\$B@EQU - COMPILER SYSTEM EQUATES				*
		7687	*	\$I\$EQU - INTERPRETER FIXED EQUATES				*
		7688	*	\$I@SEQ/\$I@LEQ - PRECISION EXECUTION EQUATES				*
		7689	*					*
		7690	*	OTHER				*
		7691	*	NONE				*
		7692	*	*****				*

FUBSGN - SIGN OF VALUE FUNCTION

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 99
		7694		*****	
		7695	*		*
		7696		*****	
		7697	*		*
		7698	*	FUBSGN - SIGN OF VALUE FUNCTION ROUTINE	*
		7699	*		*
		7700		*****	
		7701	*		*
		7702		*****	
		7703	*		
		7704	*	ENTER FUBSGN	
		7705	*		
		17A7 7706	FUBSGN EQU *	FUBSGN ENTRY PT	
		7707	*		
		7708	*	TEST THE STACKED DATA ELEMENT FOR A NULL MANTISSA	
		7709	*		
17A7 35 02 0D4E		7710	FUB100 L	I\$STAK,@XR	CADDR STACKED VALUE
17AB BD F0 01		7711	FUB150 CLI	I@UMN1(,@XR),B@DEC0	IS MANTISSA ZERO ?
17AE F2 81 15		7712		JE FUB500	YES, GO EYIT STN
		7713	*		
		7714	*	SAVE MANTISSA SIGN	
		7715	*		
17B1 68 00 CA 07		7716	FUB200 MZZ	FUBSSA(,@BR),I@SIGN(,@XR)	SAVE SIGN
		7717	*		
		7718	*	SET STACKED VALUE TO PLUS ONE	
		7719	*		
17B5 BC F0 07		7720	FUB300 MVI	I@SIGN(,@XR),B@DEC0	SET THE STACKED ELEMENTS
17B8 AC 04 06 07		7721		MVC I@UMNR-1(,@XR),I@UMNR(I@PREC-2,@XR)	* MANTISSA TO ZERO
17BC BC F1 01		7722	FUB350 MVI	I@UMN1(,@XR),B@DEC1	SET 1ST MANTISSA BYTE TO 1
17BF BC 81 00		7723		MVI I@UEXP(,@XR),B@NXZR+@B1	SET ELEMENT EXP TO +1
		7724	*		
		7725	*	RESTORE THE STACKED ELEMENTS SIGN	
		7726	*		
17C2 98 00 07 CA		7727	FUB400 MZZ	I@SIGN(,@XR),FUBSSA(,@BR)	RESTORE SIGN
		7728	*		
		7729	*	RETURN TO THE CALLING ROUTINE	
		7730	*		
17C6 C0 87 12D3		7731	FUB500 B	I\$RTRN	RETURN
		7732	*		
		7733		*****	
		7734	*	FUBSGN WORK AREA	*
		7735		*****	
		7736	*		
17CA		17CA 7737	FUBSSA DS	CL1	ELEMENT SIGN DAVE AREA
		7738	*		
		7739		*****	
		7740	*		
		7741	*	END OF ROUTINE FUBSGN	*****

FPBRAD - RAD TO DEG CONVERSION ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 100
		7743		*****	*
		7744	*	5703-XM1 COPYRIGHT IBM CORP. 1970	*
		7745	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083	*
		7746	*		*
		7747		*****	*
		7748	*	*STATUS	*
		7749	*	VERSION 1 MODIFICATION 0	*
		7750	*		*
		7751	*	*FUNCTION -	*
		7752	*	* CONVERT RADIANS TO DEGREES	*
		7753	*	* CONVERT DEGREES TO RADIANS	*
		7754	*		*
		7755	*	*ENTRY POINTS -	*
		7756	*	* FPBRAD HAS TWO ENTRY POINTS	*
		7757	*	* CALLING SEQUENCES ARE	*
		7758	*	* FPBRAD	*
		7759	*	B I\$CALL	*
		7760	*	DC AL2(V\$FRAD)	*
		7761	*	* FPBDEG	*
		7762	*	B I\$CALL	*
		7763	*	DC AL2(V\$FDEG)	*
		7764	*		*
		7765	*	*INPUT -	*
		7766	*	* I\$STAK - STACK POINTER	*
		7767	*	* REGISTER @BR POINTS TO THE FIRST BYTE IN THE PAGE	*
		7768	*	* STACK VALUE - 1ST VALUE IN THE STACK	*
		7769	*		*
		7770	*	*OUTPUT	*
		7771	*	* THE RESULT IS PLACED IN THE 1ST STACK ELEMENT	*
		7772	*		*
		7773	*	*EXTERNAL REFERENCES -	*
		7774	*	I\$STAK - STACK POINTER	*
		7775	*	I\$RTRN - PAGING ROUTINE ENTRY TO ROUTINE	*
		7776	*	I\$FMPY - MULTIPLY ROUTINE ENTRY	*
		7777	*	I\$FDVD - DIVIDE ROUTINE ENTRY	*
		7778	*		*
		7779	*	*EXITS, NORMAL -	*
		7780	*	* FPBRAD HAS 1 NORMAL EXIT	*
		7781	*	I\$RTRN - PAGING ROUTINE ENTRY TO RETURN TO CALLING ROUTINE	*
		7782	*		*
		7783	*	*EXITS, ERROR -	*
		7784	*	N/A	*
		7785	*		*
		7786	*	*TABLES/WORK AREAS -	*
		7787	*	* THE CONSTANTS AND WORK AREAS RESIDE AT THE END OF THE	*
		7788	*	EXECUTABLE CODE AND ARE REFERENCED BY @BR	*
		7789	*		*
		7790	*	*ATTRIBUTES -	*
		7791	*	* REUSABLE	*
		7792	*	* NATURALLY RELOCATEABLE	*
		7793	*		*
		7794	*	*CHARACTER CODE DEPENDENCY -	*
		7795	*	N/A	*
		7796	*		*
		7797	*	*NOTES -	*
		7798	*	ERROR PROCEDURES	*

FPBRAD - RAD TO DEG CONVERSION ROUTINE										
ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 101
				7799	*		NONE			*
				7800	*					*
				7801	*		REGISTER USAGE			*
				7802	*		* REGISTER @BR IS USED AS THE BASE REGISTER			*
				7803	*		* REGISTER @XR IS USED AS THE STACK POINTER			*
				7804	*		* AT EXIT @XR POINTS TO THE VALUE AT THE FIRST			*
				7805	*		STACK POSITION			*
				7806	*					*
				7807	*		SAVED/RESTORED AREAS			*
				7808	*		NONE			*
				7809	*					*
				7810	*		REQUIRED MODULES			*
				7811	*		@SYSEQ - COMMON SYSTEM EQUATES			*
				7812	*		\$V\$EQU - VIRTUAL ADDRESSES EQUATES			*
				7813	*		\$B@EQU - COMPILER SYSTEM EQUATES			*
				7814	*		\$I\$EQU - INTERPRETER FIXED EQUATES			*
				7815	*		\$I@SEQ/\$I@LEQ - PRECISION EXECUTION EQUATES			*
				7816	*					*
				7817	*		OTHER			*
				7818	*		NONE			*
				7819	*					*
				7820	*		*****			*

FPBRAD - RAD TO DEG CONVERSION ROUTINE											
ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 101	
				7799	*		NONE			*	
				7800	*					*	
				7801	*		REGISTER USAGE			*	
				7802	*		* REGISTER @BR IS USED AS THE BASE REGISTER			*	
				7803	*		* REGISTER @XR IS USED AS THE STACK POINTER			*	
				7804	*		* AT EXIT @XR POINTS TO THE VALUE AT THE FIRST			*	
				7805	*		STACK POSITION			*	
				7806	*					*	
				7807	*		SAVED/RESTORED AREAS			*	
				7808	*		NONE			*	
				7809	*					*	
				7810	*		REQUIRED MODULES			*	
				7811	*		@SYSEQ - COMMON SYSTEM EQUATES			*	
				7812	*		\$V\$EQU - VIRTUAL ADDRESSES EQUATES			*	
				7813	*		\$B@EQU - COMPILER SYSTEM EQUATES			*	
				7814	*		\$I\$EQU - INTERPRETER FIXED EQUATES			*	
				7815	*		\$I@SEQ/\$I@LEQ - PRECISION EXECUTION EQUATES			*	
				7816	*					*	
				7817	*		OTHER			*	
				7818	*		NONE			*	
				7819	*					*	
				7820	*	*****					*

FPBRAD - RAD TO DEG CONVERSION ROUTINE											
ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 101	
				7799	*		NONE			*	
				7800	*					*	
				7801	*		REGISTER USAGE			*	
				7802	*		* REGISTER @BR IS USED AS THE BASE REGISTER			*	
				7803	*		* REGISTER @XR IS USED AS THE STACK POINTER			*	
				7804	*		* AT EXIT @XR POINTS TO THE VALUE AT THE FIRST			*	
				7805	*		STACK POSITION			*	
				7806	*					*	
				7807	*		SAVED/RESTORED AREAS			*	
				7808	*		NONE			*	
				7809	*					*	
				7810	*		REQUIRED MODULES			*	
				7811	*		@SYSEQ - COMMON SYSTEM EQUATES			*	
				7812	*		\$V\$EQU - VIRTUAL ADDRESSES EQUATES			*	
				7813	*		\$B@EQU - COMPILER SYSTEM EQUATES			*	
				7814	*		\$I\$EQU - INTERPRETER FIXED EQUATES			*	
				7815	*		\$I@SEQ/\$I@LEQ - PRECISION EXECUTION EQUATES			*	
				7816	*					*	
				7817	*		OTHER			*	
				7818	*		NONE			*	
				7819	*					*	
				7820	*	*****					*

FPBRAD - RAD TO DEG CONVERSION ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 102
		7822		*****	
		7823	*		*
		7824		*****	
		7825	*		*
		7826	*	FDBRAD - RAD TO DEG, DEG TO RAD CONVERSION ROUTINES	*
		7827	*		*
		7828		*****	
		7829	*		*
		7830		*****	
		7832	*		
		7833	*	ENTER FPBRAD - RAD TO DEG ENTRY	
		7834	*		
		17CB 7835	FPBRAD EQU *		RAD TO DEG ENTRY POINT
		7836	*		
		7837	*	MOVE THE CONSTANT OF PI DIVIDED BY 180 TO THE 2ND STACK ELEMENT	
		7838	*		
17CB 35 02 0D4E		7839	FPB100 L	I\$STAK,@XR	CADDR OF STACKED ELEMENT
17CF 9C 07 0F F1		7840	FPB200 MVC	I@RSE2(,@XR),FPBCON+I@PREC(I@LUFV,@BR)	STACK CONSTANT
		7841	*		
		7842	*	MULTIPLY THE STACKED VALUES	
		7843	*		
17D3 C0 87 082A		7844	FPB300 B	I\$FMPY	MULTIPLY ELEMENTS
17D7 F2 87 0C		7845	J	FPB600	GO EXIT ROUTINE
		7847	*		
		7848	*	ENTER FPBRAD - DEG TO RAD ENTRY	
		7849	*		
		17DA 7850	FPBDEG EQU *		DEG TO RAD ENTRY POINT
		7851	*		
		7852	*	MOVE THE CONSTANT OF PI DIVIDED BY 180 TO THE 2ND STACK ELEMENT	
		7853	*		
17DA 35 02 0D4E		7854	FPB400 L	I\$STAK,@XR	CADDA OF STACKED ELEMENT
17DE 9C 07 0F F1		7855	FPB450 MVC	I@RSE2(,@XR),FPBCON+I@PREC(I@LUFV,@BR)	STACK CONSTANT
		7856	*		
		7857	*	DIVIDE THE STACKED VALUES	
		7858	*		
17E2 C0 87 0919		7859	FPB500 B	I\$FDVD	DIVIDE ELEMENTS
		7860	*		
		7861	*	RETURN TO THE CALLING ROUTINE	
		7862	*		
17E6 C0 87 12D3		7863	FPB600 B	I\$RTRN	RETURN
		7864	*		
		7865		*****	
		7866	*	FPBRAD CONSTANTS	*
		7867		*****	
		7868	*		
		17EA 7869	FPBCON EQU *		CADDR OF CONSTANT
17EA 7F		17EA 7870	DC	AL1(B@NXZR-1)	CONSTANT EXPONENT
17EB F1F7F4F5F3F2F9F2	17F9	7871	DC	DL(I@PRCL)'174532925199433'	PI DIVIDED BY 180 CONSTANT
		7872	*		
		7873		*****	
		7874	*		
		7875	*	END OF ROUTINE FPSRAD	*****

FQSRND - RANDOM NUMBER GENERATOR FUNCTION

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 103
		7877		*****			
		7878	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
		7879	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
		7880	*				*
		7881		*****			*
		7882	*	*STATUS			*
		7883	*	VERSION 1 MODIFICATION 0			*
		7884	*				*
		7885	*	*FUNCTION -			*
		7886	*	* FQSRND PLACES A RANDOM NUMBER FROM THE UNIFORM DISTRIBUTION			*
		7887	*	OVER THE INTERVAL (0,1)			*
		7888	*				*
		7889	*	*ENTRY POINTS -			*
		7890	*	* FQSRND HAS 1 ENTRY POINT - FQSRND			*
		7891	*	* CALLING SEQUENCE IS			*
		7892	*	B I\$CALL			*
		7893	*	DC AL2(V\$FRND)			*
		7894	*	*INPUT -			*
		7895	*	* I\$STAK - STACK POINTER			*
		7896	*	* REGISTER @BR POINTS TO THE FIRST BYTE IN THE PAGE			*
		7897	*	* STACK VALUE - FIRST VALUE IN THE STACK (OPTIONAL)			*
		7898	*	* I\$PARM - WHETHER A INPUT ARGUMENT IS IN STACK			*
		7899	*	* I\$RNSW - 1 BYTE, INITIAL PROGRAM ENTRY SWITCH			*
		7900	*				*
		7901	*	*OUTPUT -			*
		7902	*	* THE NEXT RANDOM NUMBER OF THE SEQUENCE			*
		7903	*	* I\$RNSW - 1 BYTE, INITIAL PROGRAM ENTRY SWITCH			*
		7904	*				*
		7905	*	*EXTERNAL REFERENCES -			*
		7906	*	I\$STAK - STACK POINTER			*
		7907	*	I\$RTRN - PAGING ROUTINE ENTRY TO RETURN			*
		7908	*	I\$RNSW - INITIAL RANDOM ENTRY SWITCH			*
		7909	*	I\$MDFY - PAGING ROUTINE ENTRY TO SET MODIFY SWITCH			*
		7910	*	I\$PARM - INTERPRETER PARAMETER AREA			*
		7911	*	I\$FDVD - DIVIDE ROUTINE DIM,			*
		7912	*				*
		7913	*	*EXITS, NORMAL -			*
		7914	*	* FOUND HAS 1 NORMAL EXIT			*
		7915	*	I\$RTRN - PAGING ROUTINE ENTRY TO RETURN TO CALLING			*
		7916	*				*
		7917	*	*EXITS, ERROR -			*
		7918	*	N/A			*
		7919	*				*
		7920	*	*TABLES/WORK AREAS			*
		7921	*	* THE CONSTANTS AND WORK AREAS RESIDE AT THE END OP THE			*
		7922	*	EXECUTABLE CODE AND ARE REFERENCED BY @BR			*
		7923	*				*
		7924	*	*ATTRIBUTES -			*
		7925	*	* REUSABLE			*
		7926	*	* NATURALLY RELOCATEABLE			*
		7927	*				*
		7928	*	*CHARACTER CODE DEPENDENCY -			*
		7929	*	N/A			*
		7930	*				*
		7931	*	*NOTES			*
		7932	*	ERROR PROCEDURES			*

FQSRND - RANDOM NUMBER GENERATOR FUNCTION

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 104	
				7933	*		NONE				*
				7934	*						*
				7935	*		REGISTER USAGE				*
				7936	*		* REGISTER @BR IS USED AS THE BASE REGISTER				*
				7937	*		* REGISTER @XR IS USED AS THE STACK POINTER				*
				7938	*		* AT EXIT @XR POINTS TO THE VALUE AT THE				*
				7939	*		FIRST STACK POSITION				*
				7940	*						*
				7941	*		SAVED/RESTORED AREAS				*
				7942	*		NONE				*
				7943	*						*
				7944	*		REQUIRED MODULES				*
				7945	*		@SYSEQ - COMMON SYSTEM EQUATES				*
				7946	*		\$V\$EQU - VIRTUAL ADDRESSES EQUATES				*
				7947	*		\$B@EQU - COMPILER SYSTEM EQUATES				*
				7948	*		\$I\$EQU - INTERPRETER FIXED EQUATES				*
				7949	*		\$I@SEQ - PRECISION EXECUTION EQUATES				*
				7950	*						*
				7951	*		OTHER				*
				7952	*		NONE				*
				7953	*		*****				*

FQSRND - RANDOM NUMBER GENERATOR FUNCTION

ERR	LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 105
				7955	*****	
				7956	*	*
				7957	*****	
				7958	*	*
				7959	* FQSRND - RANDOM NUMBER GENERATOR (STANDARD PRECISION)	*
				7960	*	*
				7961	*****	
				7962	*	*
				7963	*****	
				7964	*	
				7965	* ENTER FQSRND	
				7966	*	
1800				7967	*FQSRND VPAGE 0	ENTER AND SET ADDRESSABILITY
				7968	ORG *,256,0	SET STARTING ADDRESS
	1800			7969	FQSRND EQU *	START OF PROGRAM CODING
1701				7970	ORG *-255	RESET IAR OF PAGE
1800				7971	ORG *,256,0	* BOUNDARY ADDRESS
	1800			7972	USING *,@BR	SET PAGE BASE ADDRESS
1800				7973	ORG FQSRND	RESET STARTING ADDRESS
				7974	*** END OF EXPANSION ***	
				7975	*	
				7976	* INITIALIZE RANDOM ROUTINE	
				7977	*	
1800 C0 87 1349				7978	FQS005 B I\$MDFY	SET PAGE MODIFY INDICTOR
1804 35 02 0D4E				7979	L I\$STAK,@XR	CADDR STACK
				7980	*	
				7981	* TEST FOR INITIAL PROGRAM ENTRY	
				7982	*	
1808 38 01 0D5C				7983	FQS010 TBN I\$RNSW,I\$RNMK	INITIAL ENTRY SW OFF ?
180C F2 10 08				7984	JT FQS030	NO, TEST FOR ARG IN STACK
				7985	*	
				7986	* INITIALIZE THE ALGORITHM VALUES	
				7987	*	
180F 5C 17 D1 AF				7988	FQS020 MVC FQSWKZ(,@BR),FQSSVZ(I@LUFV*3,@BR)	INIT X,Y,Z
1813 3A 01 0D5C				7989	SBN I\$RNSW,I\$RNMK	
				7990	*	
				7991	* TEST IF RND FUNCTION CALL USED AN ARGUMENT	
				7992	*	
1817 3D 00 0D57				7993	FQS030 CLI I\$PARM,FQSNUL	STACK CONTAIN AN ARG ?
181B F2 81 0E				7994	JE FQS100	NO, BEGIN ITERATION
				7995	*	
				7996	* SET INITIAL VALUE OF U TO THE ARGUMENT VALUE	
				7997	*	
181E BA F0 07				7998	FQS040 SBN I@SIGN(,@XR),B@ZPOS	SET SIGN POSITIVE
1821 64 16 B9 07				7999	ZAZ FQSWKU(I@PREC+1,@BR),I@UMNR(I@PREC,@XR)	SET U TO ARG
				8000	*	
				8001	* INITIALIZE THE ALGORITHM VALUES	
				8002	*	
1825 5C 17 D1 AF				8003	FQS050 MVC FQSWKZ(,@BR),FQSSVZ(I@LUFV*3,@BR)	INIT X,Y,Z
1829 F2 87 08				8004	J FQS110	SKIP U ASSIGNMENT
				8005	*	
				8006	* U EQUALS VALUES X PLUS Y	
				8007	*	
182C 5C 07 B9 C1				8008	FQS100 MVC FQSWKU(,@BR),FQSWKX(I@LUFV,@BR)	SET U TO Y
1830 56 07 B9 C9				8009	AZ FQSWKU(I@LUFV,@BR),FQSWKY(I@LUFV,@BR)	ADD Y
				8010	*	

FQSRND - RANDOM NUMBER GENERATOR FUNCTION

ERR	LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 106
				8011	* TEST IF GREATER THAN THE PRIME			
				8012	*			
1834	5D	07 B9 97		8013	FQS110 CLC FQSWKU(,@BR),FQSSVP(I@LUFV,@BR) GT PRIME ?			
1838	F2	04 07		8014	JNH FQS130 NO. PROC LOOP			
				8015	*			
				8016	* SUBTRACT PRIME FROM U TO KEEP VALUE IN RANGE			
				8017	*			
183B	57	16 B9 97		8018	FQS120 SZ FQSWKU(I@LUFV,@BR),FQSSVP(I@LUFV-1,@BR) DEC U BY P			
183F	D0	87 34		8019	B FQS110(,@BR) CHECK VALLE AGAIN			
				8020	*			
				8021	* SET X EQUAL TO Y			
				8022	*			
1842	5C	07 C1 C9		8023	FQS130 MVC FQSWKX(,@BR),FQSWKY(I@LUFV,@BR) SHIFT Y			
				8024	*			
				8025	* SET Y EQUAL TO Z			
				8026	*			
1846	5C	07 C9 D1		8027	FQS140 MVC FQSWKY(,@BR),FQSWKZ(I@LUFV,@BR) SHIFT Z			
				8028	*			
				8029	* SET Z EQUAL TO U			
				8030	*			
184A	5C	07 D1 B9		8031	FQS150 MVC FQSWKZ(,@BR),FQSWKU(I@LUFV,@BR) SHIFT U			
				8032	*			
				8033	* TEST FOR END OF LOOP			
				8034	*			
184E	5E	00 B0 8F		8035	FQS160 ALC FQSCNT(,@BR),FQSI64(1,@BR) INCR LOOP COUNT AND RECYCLE			
1852	D0	20 2C		8036	BNOL FQS100(,@BR) * LOOP UNTIL COUNT = 0			
				8037	*			
				8038	* MOVE SUBSEQUENCE TO STACK			
				8039	*			
1855	9C	06 07 B9		8040	FQS170 MVC I@RSE1(,@XR),FQSWKU(I@PREC,@BR) SHIFT SUBSEQUENCE			
1859	BC	80 00		8041	MVI I@UEXP(,@XR),B@NXZR SET NORMALIZED EXP			
				8042	*			
				8043	* INITIALIZE NORMALIZING ROUTINE			
				8044	*			
185C	7C	07 B1		8045	FQS180 MVI FQSCT2(,@BR),I@PREC SET LOOP COUNT			
185F	BC	F0 08		8046	MVI I@RSE1+1(,@XR),B@DEC0 INIT NORMALIZING RTN			
				8047	*			
				8048	* NORMALIZE THE MANTISSA			
				8049	*			
1862	BD	F0 01		8050	FQS190 CLI I@UMN1(,@XR),B@DEC0 IS 1ST DIGIT ZERO ?			
1865	F2	01 16		8051	JNE FQS230 NO, EXIT RTN			
				8052	*			
				8053	* SHIFT MANTISSA DOWN 1 DIGIT			
				8054	*			
1868	6C	06 B9 08		8055	FQS200 MVC FQSWKU(,@BR),I@UMNR+1(I@PREC,@XR) SHIFT THE STACK			
186C	9C	06 07 B9		8056	MVC I@UMNR(I@PREC,@XR),FQSWKU(,@BR) * MANTISSA DOWN			
1870	9F	00 00 8E		8057	SLC I@UEXP(,@XR),FQSI01(1,@BR) DECR EXP			
				8058	*			
				8059	* TEST IF MANTISSA WAS ZERO			
				8060	*			
1874	5F	00 B1 8E		8061	FQS210 SLC FQSCT2(,@BR),FQSI01(1,@BR) TEST IF MANTISSA			
1878	D0	84 62		8062	BH FQS190(,@BR) * IS ZERO ?			
				8063	*			
				8064	* SET EXPONENT FOR ZERO			
				8065	*			
187B	BC	1E 00		8066	FQS220 MVI I@UEXP(,@XR),B@NXLO SET STACK EXP TO MAX LOW			

FQSRND - RANDOM NUMBER GENERATOR FUNCTION

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER	15,	MOD	00	31/05/21	PAGE	107
					8067	*								
					8068	*	DIVIDE U BY THE PRIME							
					8069	*								
187E	9C	07	0F 97		8070	FQS230	MVC I@RSE2(,@XR),FQSSVP(I@LUFV,@BR) STACK PRIME							
1882	BC	80	08		8071		MVI I@1SE2+I@UEXP(,@XR),B@NXZR SET EXP FOR PRIME							
1885	C0	87	0919		8072	FQS240	B I\$FDVD TO DIVIDE ROUTINE							
					8073	*								
					8074	*	EXIT FQSRND AND RETURN TO CALLING RTN							
					8075	*								
1889	C0	87	12D3		8076	FQS250	B I\$RTRN RETURN							
					8078	*								
					8079	*****								
					8080	*							*	
					8081	*	FQSRND EQUATES, CONSTANTS AND WORK AREAS						*	
					8082	*							*	
					8083	*****							*	
					8084	*								
					8085	*	FQSRND - EQUATES							
					8086	*								
				0000	8087	FQSNUL	EQU 0 TEST FOR NULL ARG							
					8088	*								
					8089	*	FQSRND - CONSTANTS							
					8090	*								
188D	0001			188E	8091	FQSI01	DC IL2'01' INTEGER OF 1							
188F	40			188F	8092	FQSI64	DC IL1'64' INTEGER OF 64							
1890	F0F6F6F8F4F6F7F3			1897	8093	FQSSVP	DC CL(I@LUFV)'06684673' STANDARD PREC PRIME NO.							
1898	F0F3F9F2F6F9F9F1			189F	8094	FQSSVX	DC CL(I@LUFV)'03926991' INITIAL VALUE X							
18A0	F0F1F4F4F2F6F9F5			18A7	8095	FQSSVY	DC CL(I@LUFV)'01442695' INITIAL VALUE Y							
18A8	F0F8F4F1F4F7F0F9			18AF	8096	FQSSVZ	DC CL(I@LUFV)'08414709' INITIAL VALUE Z							
					8097	*								
					8098	*	FQSRND - WORK AREAS							
					8099	*								
18B0				18B0	8100	FQSCNT	DS CL1 LOOP COUNTER							
18B0					8101		ORG *-1 * INITIALLY SET FOR THE							
18B0	00			18B0	8102		DC XL1'00' * VALUE ZERO							
18B1				18B1	8103	FQSCT2	DS CL1 MANTISSA LNG COUNTER							
18B2				18B9	8104	FQSWKU	DS CL(I@LUFV) VALUE OF U WORK AREA							
18BA				18C1	8105	FQSWKX	DS CL(I@LUFV) VALUE OF X WORK AREA							
18C2				18C9	8106	FQSWKY	DS CL(I@LUFV) VALUE OF Y WORK AREA							
18CA				18D1	8107	FQSWKZ	DS CL(I@LUFV) VALUE OF Z WORK AREA							
					8108	*								
					8109	*	END OF ROUTINE FQSRND *****							

IDDVST - S/3 BASIC INTERPRETER VECTOR STACKING ROUTINES

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 108
8111				*****			
8112	*			5703-XM1 COPYRIGHT IBM CORP. 1970			*
8113	*			REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
8114	*						*
8115				*****			*
8116	*			*STATUS			*
8117	*			VERSION 1 MODIFICATION 0			*
8118	*						*
8119	*			*FUNCTION			*
8120	*			* IDDVST CONTAINS THE RUN-TIME ROUTINES WHICH INTERPRET AND			*
8121	*			EXECUTE THE FOLLOWING PSEUDO MACHINE INSTRUCTIONS -			*
8122	*			* 'SD0' - STACK DOPE VECTOR, NO REDIMENSIONING			*
8123	*			* 'SD1' - STACK DOPE VECTOR, REDIMENSION 1			*
8124	*			* 'SD2' - STACK DOPE VECTOR, REDIMENSION 2			*
8125	*			* THE FOLLOWING DESCRIPTIONS GIVE FUNCTIONAL SPECIFICATIONS FOR			*
8126	*			THE ROUTINES WHICH ARE USED TO EXECUTE THE PSEUDO INSTRUCTIONS			*
8127	*			LISTED ABOVE. THESE INSTRUCTIONS EACH INVOLVE ARRAY DOPE			*
8128	*			VECTOR (DESCRIPTOR) OPERATIONS IN THE RUN-TIME STACK.			*
8129	*			* 'SD0' - STACK DOPE VECTOR, NO REDIMEN (FORMAT - OP VADR)			*
8130	*			THE ARITHMETIC ARRAY DESCRIPTOR AT VIRTUAL ADDRESS VADR IS			*
8131	*			PLACED AT THE TOP OF THE STACK.			*
8132	*			* 'SD1' - STACK DOPE VECTOR, REDIMEN 1 (FORMAT - OP VADR)			*
8133	*			THE FLOATING POINT VALUE AT THE TOP OF THE STACK IS CON-			*
8134	*			VERTED TO AN ARRAY INDEX WHICH IS USED TO MODIFY THE			*
8135	*			SINGLE DIMENSION IN THE ARITHMETIC ARRAY DESCRIPTOR AT			*
8136	*			VADR. THE INDEX VALUE IS DELETED FROM THE STACK AND THE			*
8137	*			REDIMENSIONED ARRAY DESCRIPTOR IS PLACED AT THE TOP OF THE			*
8138	*			STACK. THE DESCRIPTOR IS ALSO REDIMENSIONED IN VIRTUAL			*
8139	*			MEMORY STORAGE.			*
8140	*			* 'SD2' - STACK DOPE VECTOR, REDIMEN 2 (FORMAT - OP VADR)			*
8141	*			THE FLOATING POINT VALUE SECOND IN THE STACK IS CONVERTED			*
8142	*			TO AN ARRAY INDEX WHICH IS USED TO MODIFY THE ROW DIMEN-			*
8143	*			SION, AND THE VALUE AT THE TOP OF THE STACK IS CONVERTED			*
8144	*			TO AN ARRAY INDEX WHICH IS USED TO MODIFY THE COLUMN DIMEN-			*
8145	*			SION, IN THE ARITHMETIC ARRAY DESCRIPTOR AT VADR. BOTH			*
8146	*			INDEX VALUES ARE DELETED FROM THE STACK AND THE REDIMEN-			*
8147	*			SIONED ARRAY DESCRIPTOR IS PLACED AT THE TOP OF THE STACK.			*
8148	*			THE DESCRIPTOR IS ALSO REDIMENSIONED IN VIRTUAL MEMORY			*
8149	*			STORAGE.			*
8150	*						*
8151	*			*ENTRY POINTS			*
8152	*			THIS ROUTINE HAS A SINGLE ENTRY POINT - IDDVST - WHOSE FUNCTIONS			*
8153	*			ARE DEFINED ABOVE. CALLING SEQUENCE IS			*
8154	*			B IPGCAL			*
8155	*			DC AL2(V\$ISDN)			*
8156	*			WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL ADDRESS			*
8157	*			OF ENTRY POINT IDDVST. EXECUTION IS SUBJECT TO INPUT CONDITIONS			*
8158	*			DESCRIBED BELOW.			*
8159	*						*
8160	*			*INPUT			*
8161	*			* I\$STAK - 2 RYTES, FOR THE RUN-TIME STACK POINTER. THIS IS TO			*
8162	*			CONTAIN THE CORE ADDRESS OF THE FIRST AVAILABLE STACK LOCATION.			*
8163	*			FOR INSTRUCTION 'SD1' OR 'SD2', THIS WOULD BE THE STACK LOCA-			*
8164	*			TION FOLLOWING THE LAST STACKED FLOATING POINT VALUE.			*
8165	*			* I\$XIAR - 2 BYTES, FOR THE PMC ADDRESS REGISTER. THIS IS TO			*
8166	*			CONTAIN THE CORE ADDRESS OF THE OPCODE FIELD IN THE PSEUDO			*

IDDVST - S/3 BASIC INTERPRETER VECTOR STACKING ROUTINES

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 109
		8167	*	INSTRUCTION BEING EXECUTED.	*
		8168	*	* RUN-TIME STACK - THIS CONTAINS AN UNPACKED FLOATING POINT VALUE	*
		8169	*	AT THE TOP STACK POSITION (FOR 'SD1'), OR TWO UNPACKED FLOATING	*
		8170	*	POINT VALUES - ONE AT THE TOP AND ONE AT THE SECOND STACK POST-	*
		8171	*	TIONS (FOR 'SD2').	*
		8172	*	* VIRTUAL MEMORY - THIS CONTAINS THE ARRAY DESCRIPTOR REFERENCED	*
		8173	*	BY THE OPERAND IN THE CURRENT PSEUDO INSTRUCTION.	*
		8174	*		*
		8175	*	*OUTPUT	*
		8176	*	* I\$STAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER. THIS CON-	*
		8177	*	TAINS THE CORE ADDRESS OF THE STACK LOCATION IMMEDIATELY	*
		8178	*	FOLLOWING THE STACKED ARITHMETIC ARRAY DOPE VECTOR.	*
		8179	*	* I\$ERRC - 1 BYTE, FOR THE ERROR CONDITION CODE. THIS CONTAINS	*
		8180	*	A NULL CODE (I@NERR) WHEN NO ERROR CONDITION EXISTS, OR AN	*
		8181	*	ERROR CODE SPECIFYING THE PARTICULAR ERROR CONDITION DISCOVERED.	*
		8182	*	* RUN-TIME STACK - ANY INPUT REDIMENSIONING VALUES ARE DELETED,	*
		8183	*	AND THE STACK CONTAINS THE ORIGINAL OR MODIFIED ARFHMETIC	*
		8184	*	ARRAY DOPE VECTOR AT THE TOP STACK POSITION.	*
		8185	*	* VIRTUAL MEMORY - THIS CONTAINS THE MODIFIED ARRAY WE VECTOR	*
		8186	*	WHEN REDIMENSIONING HAS BEEN SPECIFIED.	*
		8187	*		*
		8188	*	*EXTERNAL REFERENCES	*
		8189	*	* I\$STCK - ENTRY POINT FOR INTERPRETER ELEMENT STACKING ROUTINE.	*
		8190	*	* I\$USTK - ENTRY POINT FOR INTERPRETER ELEMENT UNSTACKING RTN	*
		8191	*	* I\$CFBS - ENTRY POINT FOR FLT. PT. TO BINARY SUBSCRIPT CONV. RTN	*
		8192	*	* I\$ADST - ENTRY POINT FOR INTERPRETER STACK POINTER INCREMENTER.	*
		8193	*	* I\$RTRN - ENTRY POINT FOR PAGING MODULE V.M. RETURN CONTROL RTN.	*
		8194	*	* I\$BASE - CORE ADDRESS FOR INTERP BASE ADDRESSABILITY.	*
		8195	*	* I\$STAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER.	*
		8196	*	* I\$XIAR - 2 BYTES, FOR THE PSEUDO INSTRUCTION ADDRESS REGISTER.	*
		8197	*	* I\$STKI - 1 BYTE, FOR THE STACK INCREMENT PARAMETER TO INTADS.	*
		8198	*	* I\$VADR - 2 BYTES, FOR PAGING MODULE VIRTUAL ADDRESS PARAMETER.	*
		8199	*	* I\$SLNG - 1 BYTE, FOR ELEMENT STACKING LENGTH PARAM TO ISTACK.	*
		8200	*	* I\$ULNG - 1 BYTE, FOR ELEMENT UNSTACKING LENGTH ?ARP TO IUSTAK.	*
		8201	*	* I\$ERRC - 1 BYTE, FOR THE INTERPRETER EXECUTION ERROR CODE.	*
		8202	*	* I\$CLIF - 1 BYTE, FOR LENGTH OF AN UNPACKED FLOATING POINT VALUE	*
		8203	*		*
		8204	*	*EXITS, NORMAL	*
		8205	*	CONTROL IS ALWAYS PASSED TO THE PAGING ROUTINE AT ENTRY POINT	*
		8206	*	I\$RTRN (IPGRTN) FOR A RETURN TO THE INTERPRETER CALLING ROUTINE.	*
		8207	*		*
		8208	*	*EXITS, ERROR	*
		8209	*	CONTROL IS PASSED TO THE PAGING ROUTINE AT ENTRY POINT V\$RTRN-	*
		8210	*	(I\$RTRN) WITH PARAMETER I\$ERRC CONTAINING THE APPROPRIATE ERROR	*
		8211	*	MESSAGE CODE.	*
		8212	*		*
		8213	*	*TABLES/WORK AREAS	*
		8214	*	IDDVST PMC EXECUTION BRANCH ADDRESS TABLE - 6 BYTES, FOR DOPE	*
		8215	*	VECTOR STACKING PMC OPCODE TRANSLATION TO AN IDDVST ENTRY POINT	*
		8216	*	ADDRESS. THIS TABLE CONSISTS OF THREE 2-BYTE ENTRIES CONTAINING	*
		8217	*	THE FOLLOWING INFORMATION	*
		8218	*	* BYTE 0 - DUMMY SPACER.	*
		8219	*	* BYTE 1 - PAGE DISPLACEMENT WITHIN IDDVST FOR THE INTERNAL	*
		8220	*	ENTRY POINT ASSOCIATED WITH AN 'SD0', 'SD1' OR 'SD2' PSEUDO	*
		8221	*	INSTRUCTION.	*
		8222	*		*

IDDVST - S/3 BASIC INTERPRETER VECTOR STACKING ROUTINES

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 110
		8223	*	*ATTRIBUTES			*
		8224	*	* * REUSABLE			*
		8225	*	* * NATURALLY RELOCATABLE			*
		8226	*				*
		8227	*	*CHARACTER CODE DEPENDENCY			*
		8228	*	* THE OPERATION OF THIS MODULE DOES NOT DEPEND UPON A PARTICULAR			*
		8229	*	* INTERNAL REPRESENTATION OF THE EXTERNAL CHARACTER SET.			*
		8230	*				*
		8231	*	*NOTES			*
		8232	*	* ERROR PRCCEDURES			*
		8233	*	* * ERROR 1 - AN ERROR HAS OCCURRED DURING CONVERSION OF A			*
		8234	*	* FLOATING POINT VALUE TO A BINARY DIMENSION USING CAFPBS.			*
		8235	*	* * ERROR 2 - A CONVERTED BINARY DIMENSION (FOR A 1-DIMENSIONAL			*
		8236	*	* ARRAY) OR THE PRODUCT OF BOTH CONVERTED BINARY DIMENSIONS			*
		8237	*	* (FOR A 2-DIMENSIONAL ARRAY) EXCEEDS THE MAXIMUM SIZE ALLOCA-			*
		8238	*	* TED FOR THE REFERENCED ARRAY.			*
		8239	*	* * IN EACH OF THESE CASES, AN ERROR CODE FOR THE MESSAGE			*
		8240	*	* 'DIMENSIONED OUTSIDE MAXIMUM ARRAY SIZE LIMIT' IS ESTAB-			*
		8241	*	* LISHED IN INTERPRETER PARAMETER I\$ERRC AND CONTROL IS PASSED			*
		8242	*	* IMMEDIATELY TO PAGING MODULE ENTRY POINT I\$RTRN (IPGRTN).			*
		8243	*				*
		8244	*	* REGISTER USAGE			*
		8245	*	* * REGISTER @BR IS TO CONTAIN THE CORE PAGE BASE ADDRESS			*
		8246	*	* ESTABLISHED THROUGH PAGING MODULE CONTROL FOR THE PAGE WHICH			*
		8247	*	* INCLUDES IDDVST.			*
		8248	*	* * REGISTER @XR IS NOT SAVED. IT IS USED IN IDDVST FOR GENERAL			*
		8249	*	* PURPOSE INDEXING OPERATIONS.			*
		8250	*				*
		8251	*	* SAVED/RESTORED AREAS			*
		8252	*	* NONE			*
		8253	*				*
		8254	*	* MODIFICATION CONSIDERATIONS			*
		8255	*	* DOPE VECTOR STACKING PMC EXECUTION IS BASED UPON, THE SEQUENCE			*
		8256	*	* AND LENGTH OF THE ENTRIES IN THE IDDVST EXECUTION BRANCH			*
		8257	*	* ADDRESS TABLE. TABLE ENTRIES ARE SELECTED USING THE NUMERIC			*
		8258	*	* REPRESENTATION OR OPCODE 'SD0' AS A BASE DISPLACEMENT, AND AN			*
		8259	*	* CHANGES TO THE RELATIONSHIP BETWEEN THE CONSTANTS FOR ALL			*
		8260	*	* OPCODES EXECUTED BY THIS ROUTINE MUST TAKE FULL CONSIDERATION			*
		8261	*	* OF THIS TABLE USAGE AND ORGANIZATION.			*
		8262	*				*
		8263	*	* REQUIRED MODULES			*
		8264	*	* * @SYSEQ - COMMON SYSTEM EQUATES.			*
		8265	*	* * @ERMEQ - SYSTEM ERROR MESSAGE CODE EQUATES.			*
		8266	*	* * \$B@EQU - COMPILER PARAMETER AN CONSTANT EQUATES.			*
		8267	*	* * \$I\$EQU - INTERPRETER FIXED LOCATION ADDRESS EQUATES.			*
		8268	*	* * \$I@SEQ - INTERPRETER PARAMETER EQUATES (FOR STD. PREC. ONLY).			*
		8269	*	* * \$I@LEQ - INTERPRETER PARAMETER EQUATES (FOR LONG PREC. ONLY).			*
		8270	*				*
		8271	*	* OTHER			*
		8272	*	* NONE			*
		8273	*	*****			*

IDDVST - S/3 BASIC INTERPRETER VECTOR STACKING ROUTINES

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 111
					8275	*****				
					8276	* START OF PMC EXECUTION MODULE IDDVST				*
					8277	*****				
					8278	*				
					8279	* ESTABLISH ADDRESSABILITY FOR DOPE VECTOR STACKING ROUTINES				
					8280	*				
1900					8281	ORG	*,B@LVPG,0			BEGIN AT PAGE BOUNDARY
				1900	8282	USING	*,@BR			DEFINE DOPE VCTR RTNS BASE ADDR
					8283	*				
					8284	* ENTER IDDVST - ACCESS THE PMC INSTRUCTION				
					8285	*				
				1900	8286	IDDVST EQU	*			IDDVST ENTRY POINT
1900	35	02	0D4C		8287	L	I\$XIAR,@XR			LOAD INSTRUCTION CORE ADDRESS
					8288	*				
					8289	* ESTABLISH BRANCH ADDRESS FROM OPCODE DISPLACEMENT TABLE				
					8290	*				
1904	6C	00	0E 00		8291	IDD010 MVC	IDD020+@DD2(,@BR),I@XOPC(B@LCOP,@XR)			MOVE OPCODE TO DISP
1908	D2	02	7D		8292	LA	IDDBAT-B@CSD0+1(,@BR),@XR			LOAD BRANCH TABLE BASE ADDR
190B	6C	00	1E 00		8293	IDD020 MVC	IDD040+@D1(,@BR),*-(1,@XR)			MOVE TABLE ENTRY TO BR INST
					8294	*				
					8295	* INITIALIZE FOR GENERAL DOPE VECTOR STACKING EXECUTION				
					8296	*				
190F	35	02	0D4C		8297	IDD030 L	I\$XIAR,@XR			LOAD INSTRUCTION CORE ADDRESS
1913	2C	01	144A 02		8298	MVC	I\$VADR,I@XVAD(B@LCVA,@XR)			SET PAGING PARAM FOR D/V VADDR
1918	3C	07	0BA2		8299	MVI	I\$SLNG,B@LADV-1			SET STACKING RTN FOR D/V LENGTH
					8300	*				
					8301	* BRANCH TO EXECUTION ROUTINE SPECIFIED BY THE INSTRUCTION OPCODE				
					8302	*				
191C	D0	87	00		8303	IDD040 B	*-(,@BR)			GO EXECUTE CURRENT PSEUDO INST
					8304	*				
					8305	*****				

IDDVST - S/3 BASIC INTERPRETER VECTOR STACKING ROUTINES

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 112
				8307		*****				
				8308	*	IDDS0	- STACK ARRAY DOPE VECTOR (NO REDIMENSIONING)			*
				8309		*****				
				191F	8310	IDDS0 EQU	*			BEGIN IDDS0 EXECUTION
				8311	*					
				8312	*	STACK THE ARITHMETIC ARRAY DOPE VECTOR				
				8313	*					
191F	35	02	0D4E	8314	IDD050 L	I\$STAK,@XR				LOAD THE STACK POINTER
1923	C0	87	0B50	8315	B	I\$STCK				LINK TO STACK THE DOPE VECTOR
1927	F2	87	4F	8316	J	IDD180				BRANCH TO COMPLETE D/V STACKING
				8318		*****				
				8319	*	IDDS1	- STACK ARRAY DOPE VECTOR (REDIMENSION AS VECTOR ARRAY)			*
				8320		*****				
				8321	*					
				192A	8322	IDDS1 EQU	*			BEGIN IDDS1 EXECUTION
				8323	*					
				8324	*	STACK CONTAINS A SINGLE REDIMENSIONING VALUE - SET ROUTINE TO				
				8325	*	PROCESS SINGLE DIMENSION ONLY				
				8326	*					
192A	7C	87	43	8327	IDD060 MVI	IDD100+@Q(,@BR),@UCB				SET SINGLE DIMENSION SWITCH
192D	F2	87	03	8328	J	IDD080				BRANCH TO CONTINUE PROCESSING
				8330		*****				
				8331	*	IDDS2	- STACK ARRAY DOPE VECTOR (REDIMENSION AS MATRIX ARRAY)			*
				8332		*****				
				8333	*					
				1930	8334	IDDS2 EQU	*			BEGIN IDDS2 EXECUTION
				8335	*					
				8336	*	STACK CONTAINS TWO REDIMENSIONING VALUES - SET ROUTINE TO PROCESS				
				8337	*	BOTH DIMENSIONS				
				8338	*					
1930	7C	80	43	8339	IDD070 MVI	IDD100+@Q(,@BR),@NOP				SET DOUBLE DIMENSION SWITCH
				8340	*					
				8341		*****				

IDDVST - S/3 BASIC INTERPRETER VECTOR STACKING ROUTINES

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 113
		8343		*****	
		8344		* DOPE VECTOR REDIMENSION PROCESSING ROUTINE *	
		8345		*****	
		8346		*	
		8347		* SAVE THE DOPE VECTOR VIRTUAL ADDRESS FOR LATER PROCESSING	
		8348		*	
1933	6C 01 B3 02	8349	IDD080 MVC	IDDSAV(,@BR),I@XVAD(B@LCVA,@XR) MOVE D/V VADDR TO SAVE	
		8350		*	
		8351		* ESTABLISH BINARY DIMENSIONS FOR POSSIBLE VECTOR ARRAY	
		8352		*	
1937	5F 01 B5 B5	8353	IDD090 SLC	IDDWK1(,@BR),IDDWK1(B@LDMN,@BR) SET DIMENSION-1 EQUAL 0	
		8354		*	
193B	D0 87 89	8355		B IDD200(,@BR) LINK TO CONVERT DIMENSION-2	
193E	6C 01 B7 01	8356		MVC IDDWK2(,@BR),I@SIDX(B@LDMN,@XR) SAVE BINARY DIMENSION-2	
		8357		*	
		8358		* TEST FOR SINGLE OR DOUBLE STACKED DIMENSION PROCESSING	
		8359		*	
1942	F2 00 07	8360	IDD100 JC	IDD120,*-* BRANCH IF SINGLE STACKED DIMEN	
		8361		*	
		8362		* ESTABLISH BINARY 'ROW' DIMENSION FOR THE MATRIX ARRAY	
		8363		*	
1945	D0 87 89	8364	IDD110 B	IDD200(,@BR) LINK TO CONVERT DIMENSION-1	
1948	6C 01 B5 01	8365		MVC IDDWK1(,@BR),I@SIDX(B@LDMN,@XR) SAVE BINARY DIMENSION-1	
		8366		*	
		8367		* STACK THE ARITHMETIC ARRAY DOPE VECTOR - REPLACE DOPE VECTOR	
		8368		* DIMENSIONS WITH NEW BINARY DIMENSIONS	
		8369		*	
194C	C0 87 0B50	8370	IDD120 B	I\$STCK LINK TO STACK THE DOPE VECTOR	
1950	9C 03 03 B7	8371		MVC B@ACD2(,@XR),IDDWK2(2*B@LDMN,@BR) MOVE NEW DIMENS TO D/V	
		8372		*	
		8373		* COMPUTE THE NEW ARRAY SIZE (DIM1*DIM2) IN WORK AREA 2	
		8374		*	
1954	F2 87 07	8375	IDD130 J	IDD150 BRANCH TO INITIALIZE DECR DIM1	
		8376		*	
1957	6E 01 B7 03	8377	IDD140 ALC	IDDWK2(,@BR),B@ACD2(B@LDMN,@XR) ADD DIM2 TO WORK AREA	
195B	F2 A0 44	8378		JOL IDD240 IF OVERFLOW, GO SET ERROR COND	
195E	5F 01 B5 B1	8379	IDD150 SLC	IDDWK1(,@BR),IDDBN1(B@LDMN,@BR) DECREMENT DIMENSION-1	
1962	D0 84 57	8380		BH IDD140(,@BR) REPEAT MPY LOOP UNTIL DIM1 = 0	

IDDVST - S/3 BASIC INTERPRETER VECTOR STACKING ROUTINES

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 114
		8382	*				
		8383	*	TEST OR THE NEW SIZE EXCEEDING THAT ALLOWED FOR THE ARRAY AT			
		8384	*	COMPILE-TIME (AN ERROR CONDITION)			
		8385	*				
1965	6D 01 B7 05	8386	IDD160	CLC IDDWK2(,@BR),B@AMAX(B@LDMN,@XR) IF NEWSIZE EXCEED\$ MAX			
1969	F2 84 36	8387		JH IDD240 * GO TERMINATE ON SIZE ERROR			
		8388	*				
		8389	*	RESTORE THE MODIFIED DOPE VECTOR TO VIRTUAL MEMORY			
		8390	*				
196C	1C 01 144A B3	8391	IDD170	MVC I\$VADR,IDDSAV(@VADDR,@BR) SET PAGING PARAM FOR D/V VADDR			
1971	3C 07 0C3A	8392		MVI I\$ULNG,B@LADV-1 SET UNSTACK RTN FOR D/V LENGTH			
1975	C0 87 0BB0	8393		B I\$USTK LINK TO UNSTACK THE DOPE VECTOR			
		8394	*				
		8395	*	COMPLETE THE PSEUDO INSTRUCTION EXECUTION			
		8396	*				
1979	3C 08 0D4F	8397	IDD180	MVI I\$STKI,B@LADV SET STACK POINTER INCR FOR D/V			
197D	C2 01 0C60	8398		LA I\$BASE,@BR LOAD INTERPRETER BASE ADDRESS			
1981	C0 87 0C9D	8399		B I\$ADST LINK TO INCR THE STACK POINTER			
		8400	*				
1985	C0 87 12D3	8401		B I\$RTRN RETURN TO TE INTERPRETER			
		8402	*				
		8403	*	*****			

IDDVST - S/3 BASIC INTERPRETER VECTOR STACKING ROUTINES						
ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21 PAGE 115
		8405		*****		*
		8406		* STACKED DIMENSION FLOATING POINT TO BINARY CONVERSION ROUTINE		*
		8407		*		*
		8408		* THIS ROLTIME OPERATES ON A FLOATING POINT DIMENSION LOCATED IN THE		*
		8409		* RUN-TIME STACK. THE EQUIVALENT BINARY DIMENSION IS LEFT IN THE		*
		8410		* STACK IN PLACE OF THE FLOATING POINT VALUE.		*
		8411		*		*
		8412		* INPUT -		*
		8413		* I\$STAK - CONTAINS THE CORE ADDRESS OF THE STACK LOCATION FOLLOW-		*
		8414		* ING THE DIMENSION FLOATING POINT VALE.		*
		8415		* OUTPUT -		*
		8416		* I\$STAK - CONTAINS THE CORE ADDRESS OF THE LEFT BYTE OF THE		*
		8417		* STACKED 2-BYTE BINARY DIMENSION.		*
		8418		*****		*
		8419		*		*
		1989	8420	IDD200 EQU *	CONVERSION ROUTINE ENTRY POINT	
			8421	*		
1989	74 08 A1		8422	ST IDD230+@OP1(,@BR),@ARR	SET RETURN BRANCH ADDRESS	
			8423	*		
			8424	* CONVERT THE FLOATING POINT DIMENSION TO BINARY		
			8425	*		
198C	0F 00 0D4E 0D44		8426	IDD210 SLC I\$STAK,I\$CL1F(@CADDR-1)	DECREMENT THE STACK	
1992	35 02 0D4E		8427	L I\$STAK,@XR	LOAD THE STACK POINTER	
1996	C0 87 0AE3		8428	B I\$CFBS	LINK TO CONVERT THE DIMENSION	
			8429	*		
			8430	* TEST FOR SUCCESSFUL CONVERSION - RETURN IF NO ERROR DETECTED		
			8431	*		
199A	3D 00 0CBC		8432	IDD220 CLI I\$ERRC,I@NERR	IF NO CONVERSION ERROR	
199E	C0 81 0000		8433	IDD230 BE *-*	* RETURN TO CALLING ROUTINE	
			8434	*		
			8435	* ERROR EXIT - SET TO DISPLAY 'INVALID MATRIX DECLARATION' ERROR MSG		
			8436	*		
19A2	3C D2 0CBC		8437	IDD240 MVI I\$ERRC,@@E761	SET THE ERROR MESSAGE CODE	
19A6	C0 87 12D3		8438	B I\$RTRN	RETURN TO THE INTERPRETER	
			8439	*		
			8440	*****		*

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 115
					8405		*****			*
					8406	*	STACKED DIMENSION FLOATING POINT TO BINARY CONVERSION ROUTINE			*
					8407	*				*
					8408	*	THIS ROLTIME OPERATES ON A FLOATING POINT DIMENSION LOCATED IN THE			*
					8409	*	RUN-TIME STACK. THE EQUIVALENT BINARY DIMENSION IS LEFT IN THE			*
					8410	*	STACK IN PLACE OF THE FLOATING POINT VALUE.			*
					8411	*				*
					8412	*	INPUT -			*
					8413	*	I\$STAK - CONTAINS THE CORE ADDRESS OF THE STACK LOCATION FOLLOW-			*
					8414	*	ING THE DIMENSION FLOATING POINT VALE.			*
					8415	*	OUTPUT -			*
					8416	*	I\$STAK - CONTAINS THE CORE ADDRESS OF THE LEFT BYTE OF THE			*
					8417	*	STACKED 2-BYTE BINARY DIMENSION.			*
					8418		*****			*
					8419	*				*
				1989	8420	IDD200 EQU	*	CONVERSION ROUTINE ENTRY POINT		
					8421	*				
1989	74	08	A1		8422	ST	IDD230+@OP1(,@BR),@ARR	SET RETURN BRANCH ADDRESS		
					8423	*				
					8424	*	CONVERT THE FLOATING POINT DIMENSION TO BINARY			
					8425	*				
198C	0F	00	0D4E	0D44	8426	IDD210 SLC	I\$STAK,I\$CL1F(@CADDR-1)	DECREMENT THE STACK		
1992	35	02	0D4E		8427	L	I\$STAK,@XR	LOAD THE STACK POINTER		
1996	C0	87	0AE3		8428	B	I\$CFBS	LINK TO CONVERT THE DIMENSION		
					8429	*				
					8430	*	TEST FOR SUCCESSFUL CONVERSION - RETURN IF NO ERROR DETECTED			
					8431	*				
199A	3D	00	0CBC		8432	IDD220 CLI	I\$ERRC,I@NERR	IF NO CONVERSION ERROR		
199E	C0	81	0000		8433	IDD230 BE	*-*	* RETURN TO CALLING ROUTINE		
					8434	*				
					8435	*	ERROR EXIT - SET TO DISPLAY 'INVALID MATRIX DECLARATION' ERROR MSG			
					8436	*				
19A2	3C	D2	0CBC		8437	IDD240 MVI	I\$ERRC,@@E761	SET THE ERROR MESSAGE CODE		
19A6	C0	87	12D3		8438	B	I\$RTRN	RETURN TO THE INTERPRETER		
					8439	*				
					8440		*****			*

IDDVST - S/3 BASIC INTERPRETER VECTOR STACKING ROUTINES

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 116

```
      8442 *****
      8443 * DOPE VECTOR STACKING PSEUDO OPCODE EXECUTION BRANCH ADDRESS TABLE *
      8444 *****
      8445 *
19AA 001F 19AA 8446 IDDBAT EQU * D/V STACKING BRANCH TABLE ADDR
19AC 002A 19AB 8447 DC AL(@CADDR)(IDDS0-IDDVST) SD0 (X'2E') STACK D/V (NO REDM)
19AE 0030 19AD 8448 DC AL(@CADDR)(IDDS1-IDDVST) SD1 (X'30') STACK D/V (REDIM 1)
      19AF 8449 DC AL(@CADDR)(IDDS2-IDDVST) SD2 (X'32') STACK D/V (REDIM 2)

      8451 *****
      8452 * DOPE VECTOR STACKING ROUTINE CONSTANTS *
      8453 *****
      8454 *
19B0 0001 19B1 8455 IDDBN1 DC IL2'1' BINARY INTEGER +1

      8457 *****
      8458 * DOPE VECTOR STACKING ROUTINE WORK AREAS *
      8459 *****
      8460 *
19B2 19B3 8461 IDDSAV DS CL(@VADDR) VIRTUAL ADDRESS SAVE AREA
19B4 19B5 8462 IDDWK1 DS CL(B@LDMN) DIMENSION-1 WORK AREA
19B6 19B7 8463 IDDWK2 DS CL(B@LDMN) DIMENSION-2 WORK AREA
      8464 *
      8465 *****
      8466 *
      8467 * END OF ARRAY DOPE VECTOR STACKING PMC ROUTINES CODING *****
```

#FMSTD - S/3 BASIC INTERPRETER INPUT/OUTPUT PMC ROUTINES

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 117
		8469		*****			
		8470	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
		8471	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
		8472	*				*
		8473		*****			*
		8474	*	STATUS			*
		8475	*	VERSION 1 MODIFICATION 0			*
		8476	*				*
		8477	*	FUNCTION			*
		8478	*	* IDFILE CONTAINS THE RUN-TIME ROUTINES WHICH EXECUTE OR INTER-			*
		8479	*	FACE TO EXECUTE THE FOLLOWING PSEUDO MACHINE INSTRUCTIONS -			*
		8480	*	* 'SET' - INPUT DATA ELEMENT			*
		8481	*	* 'PUT' - OUTPUT DATA ELEMENT			*
		8482	*	* 'INI' - INITIATE KEYBOARD INPUT			*
		8483	*	* 'ADF' - ACTIVATE EXTERNAL DATA FILE			*
		8484	*	* 'RSR' - RESTORE INTERNAL 'DATA' FILE POINTER			*
		8485	*	* 'RST' - RESET EXTERNAL DATA FILE POINTER			*
		8486	*	* 'CLS' - CLOSE EXTERNAL DATA FILE			*
		8487	*	* 'PRS' - PRINT AND SPACE CARRIER			*
		8488	*	* 'PRU' - PRINT USING IMAGE			*
		8489	*	* THE FOLLOWING DESCRIPTIONS GIVE FUNCTIONAL SPECIFICATIONS FOR			*
		8490	*	THE ROUTINES WHICH ARE USED TO EXECUTE THE I/O PSEUDO INSTRUCTIONS LISTED ABOVE.			*
		8491	*				*
		8492	*	* 'GET' - INPUT DATA ELEMENT (FORMAT - OP VADDR)			*
		8493	*	THE NEXT DATA ELEMENT ENTERED FROM A FILE, UNDER CONTROL			*
		8494	*	OF THE INPUT ROUTINE WHOSE EXECUTION ENTRY ADDRESS IS VADDR			*
		8495	*	IS STORED IN VIRTUAL MEMORY AT THE VIRTUAL ADDRESS WHICH			*
		8496	*	IS STORED AT THE TOP OF THE STACK. THE REFERENCED ADDRESS			*
		8497	*	IS DELETED FROM THE STACK. 'GET' IS USED AS THE PRIMARY			*
		8498	*	PSEUDO INSTRUCTION FOR THE EXECUTION OF 'GET', 'READ', AND			*
		8499	*	'INPUT' PROGRAM STATEMENTS.			*
		8500	*	* 'PUT' - OUTPUT DATA ELEMENT (FORMAT - OP XX)			*
		8501	*	THE DATA ELEMENT AT THE TOP OF THE STACK, WHOSE TYPE IS			*
		8502	*	DEFINED BY PARAMETER XX, IS STORED AT THE NEXT AVAILABLE			*
		8503	*	LOCATION IN THE CURRENTLY ACTIVE EXTERNAL DATA FILE. THE			*
		8504	*	DATA ELEMENT IS DELETED FROM THE TOP OF THE STACK.			*
		8505	*	* 'INI' - INITIATE KEYBOARD INPUT (FORMAT - OP NN)			*
		8506	*	THE EXECUTION CONTROL CODES CONTAINED IN STACK POSITIONS			*
		8507	*	(NN) THROUGH (1) ARE SUPPLIED AS PARAMETERS TO THE KEY-			*
		8508	*	BOARD INPUT ROUTINE FOR DATA TYPE VERIFICATION. EACH OF			*
		8509	*	THE REFERENCED CODES ARE DELETED FROM THE STACK.			*
		8510	*	* 'ADF' - ACTIVATE EXTERNAL DATA FILE (FORMAT - OP XX')			*
		8511	*	THE DISPLACEMENT OF A FILE SPECIFICATION RECORD WITHIN			*
		8512	*	EXECUTION FILE DIRECTORY 2 IS DEFINED BY PARAMETER XX.			*
		8513	*	THE REFERENCED FILE IS TESTED FOR VALIDITY AND PREPARED			*
		8514	*	FOR INPUT OR OUTPUT.			*
		8515	*	* 'RSR' - RESTORE INTERNAL 'DATA' FILE POINTER (FORMAT - OP)			*
		8516	*	THE 'DATA' FILE POINTER IS RESTORED TO REFERENCE THE FIRST			*
		8517	*	ELEMENT IN THE FILE.			*
		8518	*	* 'RST' - RESET EXTERNAL DATA FILE POINTER (FORMAT - OP)			*
		8519	*	THE CURRENTLY ACTIVE DATA FILE POINTER IS RESET TO REFER-			*
		8520	*	ENCE THE FIRST DATA LOCATION IN THAT FILE.			*
		8521	*	* 'CLS' - CLOSE EXTERNAL DATA FILE (FORMAT - OP)			*
		8522	*	THE CURRENTLY ACTIVE DATA FILE IS CLOSED AND THE ASSOCI-			*
		8523	*	ATED POINTER IS RESET TO REFERENCE THE FIRST DATA LOCATION			*
		8524	*	IN THAT FILE.			*

#FMSTD - S/3 BASIC INTERPRETER INPUT/OUTPUT PMC ROUTINES

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 118
		8525	*	* 'PRS' - PRINT AND SPACE CARRIER (FORMAT - OP XX)			*
		8526	*	THE DATA ELEMENT AT THE TOP OF THE STACK IS OUTPUT ON THE			*
		8527	*	SYSTEM PRINT DEVICE, OR THE SYSTEM PRINT DEVICE CARRIER IS			*
		8528	*	POSITIONED, UNDER CONTROL OF PARAMETER XX. WHEN XX SPECI-			*
		8529	*	FIES DATA ELEMENT OUTPUT, THAT ELEMENT IS DELETED FROM THE			*
		8530	*	TOP OF THE STACK.			*
		8531	*	* 'PRU' - PRINT USING IMAGE (FORMAT - OP XX)			*
		8532	*	THE DATA ELEMENT AT THE TOP OF THE STACK IS OUTPUT ACCOR-			*
		8533	*	DING TO THE CURRENT IMAGE, OR THE CURRENT IMAGE IS OUTPUT,			*
		8534	*	ON THE SYSTEM PRINT DEVICE UNDER CONTROL OF PARAMETER XX.			*
		8535	*	WHEN XX SPECIFIES DATA ELEMENT OUTPUT, THAT ELEMENT IS			*
		8536	*	DELETED FROM THE TOP OF THE STACK.			*
		8537	*				*
		8538	*	*ENTRY POINTS			*
		8539	*	THIS ROUTINE HAS A SINGLE ENTRY POINT - IDFILE - WHOSE FUNCTIONS			*
		8540	*	ARE DEFINED ABOVE. CALLING SEQUENCE IS			*
		8541	*	B IPGCAL			*
		8542	*	DC AL2(V\$IFIO)			*
		8543	*	WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL ADDRESS			*
		8544	*	OF ENTRY POINT IDFILE. EXECUTION IS SUBJECT TO INPUT CONDITIONS			*
		8545	*	DESCRIBED BELOW.			*
		8546	*				*
		8547	*	*INPUT			*
		8548	*	* I\$XIAR - 2 BYTES, FOR THE PSEUDO INSTRUCTION ADDRESS REGISTER.			*
		8549	*	THIS IS TO CONTAIN THE CORE ADDRESS OF THE OPCODE FIELD IN THE			*
		8550	*	PSEUDO INSTRUCTION BEING EXECUTED.			*
		8551	*	* I\$STAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER. THIS IS TO			*
		8552	*	CONTAIN THE CORE ADDRESS OF THE FIRST AVAILABLE STACK LOCATION,			*
		8553	*	WHICH WOULD BE THE LOCATION IMMEDIATELY FOLLOWING ANY DATA IMEM			*
		8554	*	STACKED AS INPUT TO IDFILE.			*
		8555	*	* I\$DMSW - 1 BYTE, FOR THE ELEMENT MATCHING SWITCH. THIS SWITCH			*
		8556	*	IS REQUIRED ONLY FOR THE 'GET' INSTRUCTION, AND IS SET TO CODE			*
		8557	*	@NOP TO ENABLE DATA MATCHING DURING UNSTACKING OPERATIONS.			*
		8558	*	* I\$DATA - 2 BYTES, FOR THE PROGRAM INTERNAL 'DATA' FILE POINTER.			*
		8559	*	THIS IS TO CONTAIN THE VIRTUAL ADDRESS OF THE PMC RERERENCING			*
		8560	*	THE NEXT AVAILABLE 'DATA' FILE ELEMENT, AND IS REQUIRED ONLY			*
		8561	*	FOR THE 'GET' INSTRUCTION USED IN CONJUNCTION WITH A 'READ'			*
		8562	*	PROGRAM STATEMENT, AND FOR THE 'RSR' INSTRUCTION.			*
		8563	*	* I\$DAT1 - 2 BYTES, FOR THE PROGRAM INTERNAL 'DATA' FILE BASE			*
		8564	*	POINTER. THIS IS TO CONTAIN THE VIRTUAL ADDRESS OR THE FIRST			*
		8565	*	'DCA' PSEUDO NSTRUCTION GENERATED IN VIRTAL MEMORY, AND IS			*
		8566	*	REQUIRED ONLY FOR THE 'RSR' INSTRUCTION.			*
		8567	*	* PMC OPERAND - SEE 'FUNCTION'. FOR 'ADF', THE EXECUTION CODE			*
		8568	*	PARAMETER CONTAINS, IN ADDITION TO THE FILE DIRECTORY DISPLACE-			*
		8569	*	MENT, A BIT SETTING (BIT 7 = '1') WHEN THE 'ADF' INSTRUCTION IS			*
		8570	*	ASSOCIATED WITH AN OUTPUT ('PUT' OR 'MAT PUT') STATEMENT.			*
		8571	*	* RUN-TIME STACK - SEE 'FUNCTION'.			*
		8572	*	* INPUT DATA - SEE INPUT SPECIFICATIONS FOR VIRTUAL MEMORY RESI			*
		8573	*	DENT ROUTINES SFGETR, FZREAD, AND FZXINP.			*
		8574	*				*
		8575	*	*OUTPUT			*
		8576	*	* I\$STAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER. THIS CON-			*
		8577	*	TAINS THE CORE ADDRESS OF THE FIRST AVAILABLE STACK LOCATION			*
		8578	*	AFTER ALL STACKED INPUT DATA ITEMS HAVE BEEN DELETED.			*
		8579	*	* I\$ERRC - 1 BYTE, FOR THE ERROR CONDITION CODE. THIS CONTAINS			*
		8580	*	A NULL CODE (I@NERR) WHEN NO ERROR CONDITION EXISTS, OR AN			*

#FMSTD - S/3 BASIC INTERPRETER INPUT/OUTPUT PMC ROUTINES

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 119
		8581	*	ERROR CODE SPECIFYING THE PARTICULAR ERROR CONDITION DISCOVERED.	*
		8582	*	* REGISTER @XR - THIS CONTAINS THE CORE ADDRESS OF INTERPRETER	*
		8583	*	ENTRY POINT INTAD1, INTAD2 OR INTAD3, DEPENDING ON THE LENGTH	*
		8584	*	OF THE CURRENT PSEUDO INSTRUCTION.	*
		8585	*	* I\$DATA - 2 BYTES, FOR THE PROGRAM INTERNAL 'DATA' FILE POINTER.	*
		8586	*	THIS IS SET EQUAL TO THE VIRTUAL ADDRESS IN I\$DAT1 WHEN THE	*
		8587	*	'RSR' INSTRUCTION IS EXECUTED.	*
		8588	*	* VIRTUAL MEMORY - FOR THE 'GET' INSTRUCTION ONLY, THIS IS UP-	*
		8589	*	DATED TO CONTAIN THE DATA ELEMENT INPUT FOR A 'GET' 'READ' OR	*
		8590	*	'INPUT' PROGRAM STATEMENT LIST REFERENCE.	*
		8591	*	* EXTERNAL DATA FILE - FOR THE 'PUT' INSTRUCTION ONLY. REFER TO	*
		8592	*	SPECIFICATIONS FOR VIRTUAL MEMORY RESIDENT ROUTINE SFPUTR.	*
		8593	*	* 'INPUT' DATA BUFFER - FOR THE 'INI' INSTRUCTION ONLY. REFER TO	*
		8594	*	SPECIFICATIONS FOR VIRTUAL MEMORY RESIDENT ROUTINE FZXINP.	*
		8595	*	* EXECUTION FILE DIRECTORY 2 - FOR INSTRUCTIONS 'ADF', 'RST', AND	*
		8596	*	'CLS' ONLY. REFER TO SPECIFICATIONS FOR VIRTUAL MEMORY REST-	*
		8597	*	DENT ROUTINES SFADFR AND SFRSET.	*
		8598	*	* PRINTED OUTPUT OR IMAGE BUFFER UPDATE - FOR THE 'PRS' AND 'PRU'	*
		8599	*	INSTRUCTIONS ONLY. REFER TO SPECIFICATIONS FOR VIRTUAL MEMORY	*
		8600	*	RESIDENT ROUTINES FZSPRT AND FZUPRT.	*
		8601	*		*
		8602	*	*REFERENCES	*
		8603	*	* I\$USTK - ENTRY POINT FOR INTERPRETER ELEMENT UNSTACKING ROUTINE.	*
		8604	*	* I\$CUF - ENTRY POINT FOR FLOATING POINT VALUE PACKING ROUTINE.	*
		8605	*	* I\$CALL - ENTRY POINT FOR PAGING MODULE V.M. PROGRAM CALL RTN.	*
		8606	*	* I\$RTRN - ENTRY POINT FOR PAGING MODULE V.M. RETURN CONTROL RTN.	*
		8607	*	* I\$XAD1 - ENTRY POINT FOR INTERPRETER 1-BYTE PMC INCREMENT RTN.	*
		8608	*	* I\$XAD2 - ENTRY POINT FOR INTERPRETER 2-BYTE PMC INCREMENT RTN.	*
		8609	*	* I\$XAD3 - ENTRY POINT FOR INTERPRETER 3-BYTE PMC INCREMENT RTN.	*
		8610	*	* V\$XSPT - VIRTUAL ENTRY ADDRESS FOR SFPUTR, FILE OUTPUT RTN.	*
		8611	*	* V\$XKAF - VIRTUAL ENTRY ADDRESS FOR SFADFR, FILE ACTIVATE RTN.	*
		8612	*	* V\$XKRS - VIRTUAL ENTRY ADDRESS FOR SFRSET, FILE PT. RESET RTN.	*
		8613	*	* V\$XKCL - VIRTUAL ENTRY ADDRESS FOR SFRCLS, FILE CLOSE ROUTINE.	*
		8614	*	* V\$XKIN - VIRTUAL ENTRY ADDRESS FOR FZXINI, 'INPUT' INITIALIZER.	*
		8615	*	* V\$XSPR - VIRTUAL ENTRY ADDRESS FOR FZSPRT, 'PRINT' EXEC. RTN.	*
		8616	*	* V\$XSPU - VIRTUAL ENTRY ADDRESS FOR FZLPQT, 'PRINT USING' RTN.	*
		8617	*	* I\$STAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER.	*
		8618	*	* I\$XIAR - 2 BYTES, FOR THE PSEUDO INSTRUCTION ADDRESS REGISTER.	*
		8619	*	* I\$VADR - 2 BYTES, FOR PAGING MODULE VIRTUAL ADDRESS PARAMETER.	*
		8620	*	* I\$PARM - 2 BYTES, FOR THE INTERPRETER COMMUNICATION PARAMETER.	*
		8621	*	* I\$LLC - 1 BYTE, FOR STACKING ROUTINE ELEMENT LENGTH PARAMETER.	*
		8622	*	* I\$ULNG - 1 BYTE, FOR UNSTACKING ROUTINE ELEMENT LENGTH PARAM.	*
		8623	*	* I\$DATA - 2 BYTES, FOR THE INTERNAL 'DATA' FILE POINTER.	*
		8624	*	* I\$DAT1 - 2 BYTES, FOR THE INTERNAL 'DATA' FILE BASE MONTER.	*
		8625	*	* I\$ERRC - 1 BYTE, FOR THE INTERPRETER EXECUTION ERROR CODE.	*
		8626	*	* I\$CLVA - 1 BYTE, FOR THE LENGTH OF A VIRTUAL ADDRESS.	*
		8627	*	* I\$CL1F - 1 BYTE, FOR LENGTH OF AN UNPACKED FLOATING POINT VALUE.	*
		8628	*	* I\$CL1C - 1 BYTE, FOR THE LENGTH OF A CHARACTER ELEMENT.	*
		8629	*		*
		8630	*	*EXITS, NORMAL	*
		8631	*	CONTROL IS ALWAYS PASSED TO THE PAGING ROUTINE AT ENTRY POINT	*
		8632	*	I\$RTRN (IPGRTN) FOR A RETURN TO THE INTERPRETER CALLING ROUTINE.	*
		8633	*		*
		8634	*	*EXITS, ERROR	*
		8635	*	CONTROL IS PASSED TO THE PAGING ROUTINE AT ENTRY POINT I\$RTRN	*
		8636	*	(IPGRTN) WITH PARAMETER I\$ERRC CONTAINING THE APPROPRIATE ERROR	*

#FMSTD - S/3 BASIC INTERPRETER INPUT/OUTPUT PMC ROUTINES

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 120
		8637	*	MESSAGE CODE.			*
		8638	*				*
		8639	*	TABLES /WORK AREAS			*
		8640	*	IDFILE PMC EXECUTION BRANCH ADDRESS TABLE - I8 BYTES, FOR			*
		8641	*	INPUT/OUTPUT PMC OPCODE TRANSLATION TO AN IDFILE ENTRY POINT			*
		8642	*	ADDRESS. THIS TABLE CONSISTS OF NINE 2-BYTE ENTRIES CONTAINING			*
		8643	*	THE FOLLOWING INFORMATION -			*
		8644	*	* BYTE 0 - DUMMY SPACER.			*
		8645	*	* BYTE 1 - PAGE DISPLACEMENT WITHIN IDFILE FOR THE INTERNAL			*
		8646	*	ENTRY POINT ASSOCIATED WITH A 'GET', 'PUT', 'INI', 'ADF',			*
		8647	*	'RSR', 'RST', 'CLS', 'PRS', OR 'PRU' PSEUDO INSTRUCTION.			*
		8648	*				*
		8649	*	ATTRIBUTES			*
		8650	*	REUSABLE AND NATURALLY RELOCATABLE			*
		8651	*				*
		8652	*	CHARACTER CODE DEPENDENCY			*
		8653	*	THE OPERATION OF THIS MODULE DOES NOT DEPEND UPON A PARTICULAR			*
		8654	*	INTERNAL REPRESENTATION OF THE EXTERNAL CHARACTER SET.			*
		8655	*				*
		8656	*	NOTES			*
		8657	*	ERROR PROCEDURES			*
		8658	*	* ERROR 1 - AN ERROR HAS OCCURRED DURING EXECUTION OF ONE OF			*
		8659	*	THE INPUT/OUTPUT ROUTINES (E.G, SFGETR), AND AN APPROPRIATE			*
		8660	*	ERROR CODE HAS BEEN LEFT IN PARAMETER I\$ERRC.			*
		8661	*	* ERROR 2 - EXECUTION OF AN 'ADF' INSTRUCTION IS ATTEMPTED			*
		8662	*	WHERE THE OPERAND DOES NOT DEFINE A FILE. AN ERROR CODE FOR			*
		8663	*	THE MESSAGE 'REQUIRED FILE NOT ALLOCATED' IS ESTABLISHED IN			*
		8664	*	PARAMETER I\$ERRC.			*
		8665	*	* IN EACH OF THESE CASES, CONTROL IS EVENTUALLY PASSED TO			*
		8666	*	PAGING MODULE ENTRY POINT I\$PTRN (IPGRTN).			*
		8667	*				*
		8668	*	REGISTER USAGE			*
		8669	*	* REGISTER @BR IS TO CONTAIN THE CORE PAGE BASE ADDRESS			*
		8670	*	ESTABLISHED THROUGH PAGING MODULE CONTROL FOR THE PAGE WHICH			*
		8671	*	INCLUDES IDFILE.			*
		8672	*	* REGISTER @XR IS NOT SAVED. IT IS USED IN IDFILE FOR GENERAL			*
		8673	*	PURPOSE INDEXING OPERATIONS, AND CONTAINS THE CORE ADDRESS			*
		8674	*	OF ONE OF THE PMC INCREMENTING ROUTINES AT IDFILE EXIT.			*
		8675	*				*
		8676	*	SAVED/RESTORED AREAS			*
		8677	*	NONE			*
		8678	*				*
		8679	*	MODIFICATION CONSIDERATIONS			*
		8680	*	INPUT/OUTPUT PMC EXECUTION IS BASED UPON THE SEQUENCE AND			*
		8681	*	LENGTH OF THE ENTRIES IN THE IDFILE EXECUTION BRANCH ADDRESS			*
		8682	*	TABLE. TABLE ENTRIES ARE SELECTED USING THE NUMERIC REPRESENTATION OF OPCODE 'GET' AS A BASE DISPLACEMENT, AND ANY CHANGES			*
		8683	*	TO THE RELATIONSHIP BETWEEN THE CONSTANTS FOR ALL OPCODES			*
		8684	*	EXECUTED BY THIS ROUTINE MUST TAKE FULL CONSIDERATION OF THIS			*
		8685	*	TABLE USAGE AND ORGANIZATION.			*
		8686	*				*
		8687	*				*
		8688	*	REQUIRED MODULES			*
		8689	*	* @SYSEQ - COMMON SYSTEM EQUATES.			*
		8690	*	* @ERMEQ - SYSTEM ERROR MESSAGE CODE EQUATES.			*
		8691	*	* @VMDEQ - VIRTUAL MEMORY FIXED DIRECTORY EQUATES.			*
		8692	*	* \$V\$EQU - VIRTUAL MEMORY FIXED ADDRESS EQUATES.			*

#FMSTD - S/3 BASIC INTERPRETER INPUT/OUTPUT PMC ROUTINES

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 121
		8693	*		* \$B@EQU - COMPILER PARAMETER AND CONSTANT EQUATES.			*
		8694	*		* \$I\$EQU - INTERPRETER FIXED LOCATION ADDRESS EQUATES.			*
		8695	*		* \$I@SEQ - INTERPRETER PARAMETER EQUATES (FOR STD. PREC. ONLY).			*
		8696	*		* \$I@LEQ - INTERPRETER PARAMETER EQUATES (FOR LONG PREC. ONLY).			*
		8697	*					*
		8698	*	OTHER				*
		8699	*	NONE.				*
		8700		*****				*

#FMSTD - S/3 BASIC INTERPRETER INPUT/OUTPUT PMC ROUTINES

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 122
					8702	*****				
					8703	* START OF PMC EXECUTION MODULE IDFILE				*
					8704	*****				
					8705	*				
					8706	* ESTABLISH ADDRESSABILITY FOR FILE I/O ROUTINES				
					8707	*				
1A00					8708	ORG	*,B@LVPG,0			BEGIN AT PAGE BOUNDARY
				1A00	8709	USING	*,@BR			DEFINE FILE I/O RTN BASE ADOR
					8710	*				
					8711	* ENTER IDFILE - ACCESS THE PMC INSTRUCTION				
					8712	*				
				1A00	8713	IDFILE EQU	*			IDFILE ENTRY POINT
1A00	35	02	0D4C		8714	L	I\$XIAR,@XR			LOAD INSTRUCTION CORE ADDRESS
					8715	*				
					8716	* ESTABLISH BRANCH ADDRESS FROM OPCODE DISPLACEMENT TABLE				
					8717	*				
1A04	6C	00	0E 00		8718	IDF010 MVC	IDF020+@DD2(,@BR),I@XOPC(B@LCOP,@XR)			MOVE OPCODE TO DISP
1A08	D2	02	8F		8719	LA	IDFBAT-B@CGET+1(,@BR),@XR			LOAD BRANCH TABLE BASE ADDR
1A0B	6C	00	1A 00		8720	IDF020 MVC	IDF040+@D1(,@BR),*-(1,@XR)			MOVE TABLE ENTRY TO BR INST
					8721	*				
					8722	* INITIALIZE FOR GENERAL FILE I/O PSEUDO INSTRUCTION EXECUTION				
					8723	*				
1A0F	35	02	0D4C		8724	IDF030 L	I\$XIAR,@XR			LOAD INSTRUCTION CORE ADDRESS
1A13	2C	00	0D57 01		8725	MVC	I\$PARM,I@XCOD(1,@XR)			MOVE POSSIBLE OPERAND TO PARAM
					8726	*				
					8727	* BRANCH TO EXECUTION ROUTINE SPECIFIED BY THE INSTRUCTION CODE				
					8728	*				
1A18	D0	87	00		8729	IDF040 B	*-(,@BR)			GO EXECUTE CURRENT PSEUDO INST
					8730	*				
					8731	*****				

#FMSTD - S/3 BASIC INTERPRETER INPUT/OUTPUT PMC ROUTINES

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 123
		8733		*****	
		8734	*		*
		8735	*	RUN-TIME STACK POINTER DECREMENTING SUBROUTINE -	*
		8736	*	* DECREASES STACK POINTER FOR UNPACKED FLOATING POINT OR	*
		8737	*	CHARACTER ELEMENT WHICHEVER WAS LAST STACKED.	*
		8738	*	* PACKS THE UNPACKED FLOATING POINT ELEMENT, WHEN REQUIRED,	*
		8739	*	WHEN THE ELEMENT IS ARITHMETIC.	*
		8740	*		*
		8741	*	INPUT -	*
		8742	*	* I\$STAK - 2 BYTES, FOR RUN-TIME STACK POINTER - CONTAINS CORE	*
		8743	*	ADDRESS OF BYTE FOLLOWING THE ELEMENT IN THE STACK	*
		8744	*	* I\$SLLC - 1 BYTE, FOR LENGTH CODE (LENGTH - 1) OF LAST STACKED	*
		8745	*	ELEMENT	*
		8746	*	* IDF065+@Q - 1 BYTE, FOR PACKING SWITCH - @UCB CAUSES PACKING	*
		8747	*	TO OCCUR WHEN ELEMENT IS ARITHMETIC, @NOP DISABLES PACKING	*
		8748	*	OPERATION	*
		8749	*		*
		8750	*	OUTPUT -	*
		8751	*	* I\$STAK - CONTAINS CORE ADDRESS OF LEFTMOST BYTE OF THE STACKED	*
		8752	*	DATA ELEMENT	*
		8753		*****	
		8754	*		
1A1B 74 08 3F		8755	IDF050 ST	IDF075+@OP1(,@BR),@ARR STORE RETURN BRANCH ADDRES	
		8756	*		
		8757	*	TEST FOR TYPE OF DATA ELEMENT LAST STACKED	
		8758	*		
1A1E 3D 12 0BA1		8759	IDF055 CLI	I\$SLLC,I@LCRV-1 IF CHARACTER ELEMENT STACKED	
1A22 F2 81 11		8760	JE	IDF070 * GO DECREMENT FOR CHARACTER	
		8761	*		
		8762	*	FLOATING POINT ELEMENT - DECREMENT POINTED AND PACK IF SPECIFIED	
		8763	*		
1A25 0F 00 0D4E 0D44		8764	IDF060 SLC	I\$STAK,I\$CL1F(1) DECK POINTER FOR UNPACKED	
		8765	*	* FLOATING POINT ELEMENT	
1A2B 35 02 0D4E		8766	L	I\$STAK,@XR LOAD STACK POINTER TO PACK	
1A2F C0 00 0A85		8767	IDF065 BC	I\$CUPF,*-* LINK TC PACK FLT POINT ELEMENT	
1A30		8768	ORG	IDF065+@Q * WHEN PACK ENABLED - INITLZ	
1A30 80	1A30	8769	DC	AL1(@NOP) * THE BRANCH INSTRUCTION TO	
1A33		8770	ORG	IDF065+@INST4 * DISABLE THE PACK OPTION	
		8771	*		
1A33 F2 87 06		8772	J	IDF075 GO RETURN TO CALLER	
		8773	*		
		8774	*	CHARACTER ELEMENT - DECREMENT THE STACK POINTER	
		8775	*		
1A36 0F 00 0D4E 0D46		8776	IDF070 SLC	I\$STAK,I\$CL1C(1) DECR POINTER FOR CHAR LEMENT	
		8777	*		
		8778	*	RETURN CONTROL TO CALLING ROUTINE	
		8779	*		
1A3C C0 87 0000		8780	IDF075 B	*-* RETURN TO CALLING ROUTINE	
		8781	*		
		8782		*****	

#FMSTD - S/3 BASIC INTERPRETER INPUT/OUTPUT PMC ROUTINES

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 124
					8784	*****	*****	
					8785	* IDFGET - PERFORM GET FROM INPUT FILE (GET, INPUT, READ STMENTS)	*	
					8786	*****	*****	
					8787	*		
				1A40	8788	IDFGET EQU *	BEGIN IDFGET EXECUTION	
					8789	*		
					8790	* STACK A DATA ELEMENT FOR 'GET', 'INPUT', OR 'READ' OPERATION		
					8791	*		
	1A40	6C	01 49 02		8792	IDF100 MVC IDF110(, @BR), I@XVAD(@VADDR, @XR)	SET INPUT ROUTINE VADDR	
					8793	*	* ENTRY FROM INST OPER	
	1A44	C0	87 12B1		8794	B I\$CALL	LINK TO STACK ELEMENT FROM FILE	
	1A48			1A49	8795	IDF110 DS CL(@VADDR)	INPUT ROUTINE VADDR ENTRY ARE	
					8796	*		
					8797	* ESTABLISH VIRTUAL ADDRESS DESTINATION FOR STACKED ELEMENT		
					8798	*		
	1A4A	0F	00 0D4E 0D49		8799	IDF120 SLC I\$STAK, I\$CLVA(1)	DECR STACK POINTER TO VADDR	
	1A50	35	02 0D4E		8800	L I\$STAK, @XR	LOAD THE STACK POINTER	
	1A54	2C	01 144A 01		8801	MVC I\$VADR, I@SVAD(@VADDR, @XR)	SET VADDR PAGING PARAMETER	
					8802	*		
					8803	* TEST FOR A STACKED CHARACTER ELEMENT		
					8804	*		
	1A59	E2	02 02		8805	IDF130 LA @VADDR(, @XR), @XR	INCR STACK POINTER REGISTER	
	1A5C	B8	40 00		8806	TBN I@STAT(, @XR), B@DTYP	IF NOT STACKED CHAR ELEMENT	
	1A5F	F2	90 04		8807	JF IDF140	* GO UNSTACK FLT PT ELEMENT	
	1A62	3C	12 0C3A		8808	MVI I\$ULNG, I@LCRV-1	* ELSE SET TO UNSTACK CHAR ELEM	
					8809	*		
					8810	* TEST FOR INPUT ERROR - UNSTACK ELEMENT IF NO ERROR CONDITION		
					8811	*		
	1A66	3D	00 0CBC		8812	IDF140 CLI I\$ERRC, I@NERR	IF NO FILE INPUT ERROR SET	
	1A6A	C0	81 0BB0		8813	BE I\$USTK	* LINK TO UNSTACK THE ELEMENT	
					8814	*		
					8815	* SET PSEUDO INSTRUCTION INCREMENT AND BRANCH TO RETURN		
					8816	*		
	1A6E	C2	02 0C7B		8817	IDF150 LA I\$XAD3, @XR	LOAD 3-BYTE PMC INCR RTN ADDR	
	1A72	D0	87 DC		8818	B IDF990(, @BR)	GO RETURN TO INTERPRETER	
					8819	*		
					8820	*****	*****	

#FMSTD - S/3 BASIC INTERPRETER INPUT/OUTPUT PMC ROUTINES

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 125
		8822		*****			
		8823	*	IDFPUT - PERFORM PUT TO OUTPUT FILE (PUT STATEMENT)			*
		8824		*****			
		8825	*				
		1A75	8826	IDFPUT EQU *		BEGIN IDFPUT EXECUTION	
		8827	*				
		8828	*	RUN-TIME STACK CONTAINS ELEMENT TO BE OUTPUT - ADJUST STACK POINTER			
		8829	*	TO LEFTMOST BYTE OF THE ELEMENT, PACKING ELEMENT IF ARITHMETIC			
		8830	*				
1A75	7C 87 30	8831	IDF200	MVI IDF065+@Q(,@BR),@UCB		ENABLE FLT PT ELEMENT PACKING	
1A78	D0 87 1B	8832		B IDF050(,@BR)		LINK TO DECK POINTER FOR PUT	
1A7B	7C 80 30	8833		MVI IDF065+@Q(,@BR),@NOP		DISABLE FLT PT ELEMENT PACKING	
		8834	*				
		8835	*	OUTPUT HE DATA ELEMENT FOR THE 'PUT' OPERATION			
		8836	*				
1A7E	C0 87 12B1	8837	IDP210	B I\$CALL		LINK TO OUTPUT ELEMENT TO FILE	
1A82	1D00	1A83	8838	DC AL(@VADDR)(V\$XSPT)		OUTPUT ROUTINE VADDR ENTRY PT	
		8839	*				
		8840	*	BRANCH TO SET THE PSEUDO INSTRUCTION INCREMENT			
		8841	*				
1A84	D0 87 B9	8842	IDF220	B IDF620(,@BR)		GO LOAD PMC INCREMENT RTN ADDR	
		8843	*				
		8844		*****			

#FMSTD - S/3 BASIC INTERPRETER INPUT/OUTPUT PMC ROUTINES

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 126
		8846		*****	
		8847	*	IDFINI - PERFORM INITIATE INPUT OPERATION (INPUT STATEMENT)	*
		8848		*****	
		8849	*		
		1A87 8850	IDFINI EQU *	BEGIN IDFINI EXECUTION	
		8851	*		
		8852	*	RUN-TIME STACK CONTAINS A SERIES OF INPUT VERIFICATION CODES -	
		8853	*	ADJUST STACK POINTER TO THE LEFTMOST BYTE OF THIS SERIES	
		8854	*		
1A87 2F 00 0D4E 01		8855	IDF300 SLC	I\$STAK,I@XCNT(1,@XR)	DECR STACK PT USING PMC COUNT
		8856	*		
		8857	*	VERIFY CURRENT INPUT USING STACKED CODE SERIES	
		8858	*		
1A8C C0 87 12B1		8859	IDF310 B	I\$CALL	LINK TO VERIFY INPUT ELEMENTS
1A90 2B00	1A91	8860	DC	AL(@VADDR)(V\$XKIN)	VERIFY ROUTINE VADDR ENTRY PT
		8861	*		
		8862	*	BRANCH TO SET THE PSEUDO INSTRUCTION INCREMENT	
		8863	*		
1A92 D0 87 B9		8864	IDF320 B	IDF620(,@BR)	GO LOAD PMC INCREMENT RTN ADDR
		8865	*		
		8866		*****	

#FMSTD - S/3 BASIC INTERPRETER INPUT/OUTPUT PMC ROUTINES

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 127
			8868	*****			
			8869	* IDFADF - PERFORM DATA FILE ACTIVATION (EXTERNAL FILE STATEMENTS)			*
			8870	*****			
			8871	*			
		1A95	8872	IDFADF EQU *			BEGIN IDFADF EXECUTION
			8873	*			
			8874	* FILE ALLOCATED - ACTIVATE THE FILE FOR INPUT, OUTPUT OR CONTROL			
			8875	*			
1A95 C0 87 12B1			8876	IDF420 B I\$CALL			LINK TO ACTIVATE THE FILE
1A99 1C00		1A9A	8877	DC AL(@VADDR)(V\$XKAF)			FILE ACTIVATION RTN VADDR ENTRY
			8878	*			
			8879	* BRANCH TO SET THE PSEUDO INSTRUCTION INCREMENT			
			8880	*			
1A9B D0 87 B9			8881	IDF430 B IDF620(,@BR)			GO LOAD PMC INCREMENT RTN ADDR
			8882	*			
			8883	*****			

#FMSTD - S/3 BASIC INTERPRETER INPUT/OUTPUT PMC ROUTINES

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 128
			8885	*****	
			8886	* IDFPRS - PERFORM SEMI-FORMATTED PRINT OPERATION (PRINT STATEMENT) *	
			8887	*****	
			8888	*	
		1A9E	8889	IDFPRS EQU * BEGIN IDFPRS EXECUTION	
			8890	*	
			8891	* ADJUST STACK POINTER WHEN DATA ELEMENT TO BE OUTPUT	
			8892	*	
1A9E	BD 05 01		8893	IDF500 CLI I@XCOD(, @XR), B@PRSL IF CONTROL CODE SPECIFIES DATA	
1AA1	D0 82 1B		8894	BL IDF050(, @BR) * LINK TO DECR STACK POINTER	
			8895	*	
			8896	* OUTPUT DATA ELEMENT OR PERFORM CARRIER CONTROL OPERATION	
			8897	*	
1AA4	C0 87 12B1		8898	IDF510 B I\$CALL LINK TO PERFORM PRINTER OUTPUT	
1AA8	3400	1AA9	8899	DC AL(@VADDR) (V\$XSPR) PRINT ROUTINE VADDR ENTRY POINT	
			8900	*	
			8901	* BRANCH TO SET THE PSEUDO INSTRUCTION INCREMENT	
			8902	*	
1AAA	D0 87 B9		8903	IDF520 B IDF620(, @BR) GO LOAD PMC INCREMENT RTN ADDR	
			8904	*	
			8905	*****	

#FMSTD - S/3 BASIC INTERPRETER INPUT/OUTPUT PMC ROUTINES

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 129
					8907	*****				
					8908	* IDFPRU - PERFORM IMAGE-FORMATTED PRINT OPERATION (PRINT USING STMT) *				
					8909	*****				
					8910	*				
				1AAD	8911	IDFPRU EQU *	BEGIN IDFPRU EXECUTION			
					8912	*				
					8913	* ADJUST STACK POINTER WHEN DATA ELEMENT TO BE OUTPUT				
					8914	*				
		1AAD	B9 0C 01		8915	IDF600 TBF I@XCOD(, @XR), IDFSMK	IF CONTROL CODE SPECIFIES DATA			
		1AB0	D0 90 1B		8916	BF IDF050(, @BR)	* LINK TO DECR STACK POINTER			
					8917	*				
					8918	* OUTPUT DATA ELEMENT, CONTROL CARRIER, OR ESTABLISH OUTPUT IMAGE				
					8919	*				
		1AB3	C0 87 12B1		8920	IDF610 B I\$CALL	LINK TO PERFORM PRINT OPERATION			
		1AB7	3800	1AB8	8921	DC AL(@VADDR) (V\$XSPU)	PRINT-USING RTN VADDR ENTRY PT			
					8922	*				
					8923	* SET PSEUDO INSTRUCTION INCREMENT AND BRANCH TO RETURN				
					8924	*				
		1AB9	C2 02 0C82		8925	IDF620 LA I\$XAD2, @XR	LOAD 2-BYTE PMC INCR RTN ADDR			
		1ABD	D0 87 DC		8926	B IDF990(, @BR)	GO RETURN TO INTERPRETER			
					8927	*				
					8928	* PRU PSEUDO INSTRUCTION CONTROL CODE MASK - ASSUMES CONTROL CODES				
					8929	* 1 TO 3 SPECIFY NO DATA IN STACK, CODES 4 TO 15 SPECIFY STACKED DATA				
					8930	*				
				000C	8931	IDFSMK EQU X'0C'	PRU INST CONTROL CODE MASK			
					8932	*				
					8933	*****				

#FMSTD - S/3 BASIC INTERPRETER INPUT/OUTPUT PMC ROUTINES

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 130
					8935	*****				
					8936	* IDFRSR - PERFORM 'RESTORE' INTERNAL DATA FILE POINTER				*
					8937	*****				
					8938	*				
				1AC0	8939	IDFRSR EQU	* BEGIN IDFOR EXECUTION			
					8940	*				
					8941	* RESTORE DATA POINTER TO REFERENCE 1ST INTERNAL FILE ELEMENT				
					8942	*				
		1AC0	0C 01 0D53 0D55		8943	IDF700 MVC I\$DATA,I\$DAT1(@VADDR)	SET DATA POINTER TO 1ST ELEMENT			
					8944	*				
					8945	* BRANCH TO SET THE PSEUDO INSTRUCTION INCREMENT				
					8946	*				
				1AC6	D0 87 D8	IDF710 B IDF910(,@BR)	GO LOAD PMC INCREMENT RTN ADDR			
					8948	*				
					8949	*****				

#FMSTD - S/3 BASIC INTERPRETER INPUT/OUTPUT PMC ROUTINES

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 131
			8951	*****			
			8952	* IDFRST - PERFORM 'RESET' EXTERNAL DATA FILE POINTER			*
			8953	*****			
			8954	*			
		1AC9	8955	IDFRST EQU *			BEGIN IDFOR EXECUTION
			8956	*			
			8957	* RESET DATA POINTER TO REFERENCE 1ST EXTERNAL FILE ELEMENT			
			8958	*			
1AC9 C0 87 12B1			8959	IDF800 B I\$CALL			LINK TO RESET FILE POINTER
1ACD 240D		1ACE	8960	DC AL(@VADDR)(V\$XKRS)			FILE RESET RTN VADDR ENTRY PT
			8961	*			
			8962	* BRANCH TO SET THE PSEUDO INSTRUCTION INCREMENT			
			8963	*			
1ACF D0 87 D8			8964	IBR810 B IDF910(,@BR)			GO LOAD PMC INCREMENT RTN ADDR
			8965	*			
			8966	*****			

#FMSTD - S/3 BASIC INTERPRETER INPUT/OUTPUT PMC ROUTINES

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 132
		8968		*****	
		8969	*	IDFCLS - PERFORM EXTERNAL DATA FILE 'CLOSE' OPERATION	*
		8970		*****	
		8971	*		
		1AD2	8972	IDFCLS EQU *	BEGIN IDFCLS EXECUTION
		8973	*		
		8974	*	CLOSE THE CURRENTLY ACTIVE EXTERNAL DATA FILE	
		8975	*		
1AD2 C0 87 12B1		8976	IDF900 B	I\$CALL	LINK TO CLOSE EXTERNAL FILE
1AD6 240A		1AD7	8977	DC AL(@VADDR)(V\$XKCL)	FILE CLOSE RTN VADDR ENTRY PT
		8978	*		
		8979	*	SET THE PSEUDO INSTRUCTION INCREMENT	
		8980	*		
1AD8 C2 02 0C89		8981	IDF910 LA	I\$XAD1,@XR	LOAD 1-BYTE PMC INCR RTN ADDR
		8982	*		
		8983	*	RETURN CONTROL TO THE CORE-RESIDENT INTERPRETER	
		8984	*		
1ADC C0 87 12D3		8985	IDF990 B	I\$RTRN	RETURN TO THE INTERPRETER
		8986	*		
		8987		*****	

#FMSTD - S/3 BASIC INTERPRETER INPUT/OUTPUT PMC ROUTINES

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 133
				8989		*****				
				8990	*	FILE INPUT/OUTPUT PSEUDO OPCODE EXECUTION BRANCH ADDRESS TABLE	*			
				8991		*****				
				8992	*					
				1AE0	8993	IDFBAT EQU	* FILE I/O BRANCH TABLE ADDRESS			
1AE0	0040			1AE1	8994	DC	AL(@CADDR)(IDFGET-IDFILE) GET (X'52') INPUT DATA ITEM			
1AE2	0075			1AE3	8995	DC	AL(@CADDR)(IDFPUT-IDFILE) PUT (X'54') OUTPUT DATA ITEM			
1AE4	0087			1AE5	8996	DC	AL(@CADDR)(IDFINI-IDFILE) INI (X'56') INITIATE INPUT			
1AE6	0095			1AE7	8997	DC	AL(@CADDR)(IDFADF-IDFILE) ADF (X'98') ACTIVATE FILE			
1AE8	00C0			1AE9	8998	DC	AL(@CADDR)(IDFRSR-IDFILE) RSR (X'5A') RESTORE DATA PT			
1AEA	00C9			1AEB	8999	DC	AL(@CADDR)(IDFRST-IDFILE) RST (X'5C') RESET FILE PT			
1AEC	00D2			1AED	9000	DC	AL(@CADDR)(IDFCLS-IDFILE) CLS (X'5E') CLOSE FILE			
1AEE	009E			1AEF	9001	DC	AL(@CADDR)(IDFPRS-IDFILE) DRS (X'60') PRINT & SPACE			
1AF0	00AD			1AF1	9002	DC	AL(@CADDR)(IDFPRU-IDFILE) PRU (X'62') PRINT USING			
				9003	*					
				9004		*****				
				9005	*					
				9006	*	END OF FILE INPUT/OUTPUT PMC ROUTINES CODING	*****			

SFADFR - ACTIVATE DATA FILE - PART 1

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 134
		9008		*****			
		9009	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
		9010	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
		9011	*				*
		9012		*****			*
		9013	*	STATUS			*
		9014	*	VERSION 1 MODIFICATION 0			*
		9015	*				*
		9016	*	FUNCTION			*
		9017	*	* IDIFNC IS THE RUN-TIME ROUTINE WHICH INTERPRETES AND EXECUTES			*
		9018	*	PSEUDO MACHINE INSTRUCTION 'FCI' (FUNCTION CALL, INDIRECT).			*
		9019	*	* THE FOLLOWING DESCRIPTION GIVES FUNCTIONAL SPECIFICATIONS FOR			*
		9020	*	THE 'FCI' INSTRUCTION EXECUTION ROUTINE.			*
		9021	*	* 'FCI' - FUNCTION CALL, INDIRECT (FORMAT - OP VADR)			*
		9022	*	THE FLOATING POINT VALUE AT THE TOP OF THE STACK IS USED			*
		9023	*	AS THE ARGUMENT FOR THE USER FUNCTION WHOSE LINKING			*
		9024	*	ADDRESS IS DEFINED AT VIRTUAL ADDRESS VADR. THE VALUE AT			*
		9025	*	THE TOP OF THE STACK IS DELETED AND CONTROL IS TRANSFERRED			*
		9026	*	TO THE PSEUDO INSTRUCTION WHICH BEGINS THE USER FUNCTION			*
		9027	*	EXECUTION. LINKAGE IS ESTABLISHED SUCH THAT THE FUNCTION			*
		9028	*	EXECUTION SEQUENCE RETURNS CONTROL TO THE PSEUDO INSTRUC-			*
		9029	*	TION WHICH FOLLOWS THE 'FCI'. PRIOR TO USER FUNCTION			*
		9030	*	EXECUTION, THE FUNCTION ACTIVITY TABLE IS SEARCHED FOR AN			*
		9031	*	ENTRY WHICH MATCHES VADR. WHEN NO MATCH OCCURS, VADR IS			*
		9032	*	ADDED TO THE TABLE. WHEN A MATCH DOES OCCUR, OR WHEN THE			*
		9033	*	TABLE SIZE IS EXCEEDED, A TERMINAL ERROR CONDITION IS			*
		9034	*	INDICATED.			*
		9035	*				*
		9036	*	ENTRY POINTS			*
		9037	*	THIS ROUTINE HAS A SINGLE ENTRY POINT - IDIFNC - WHOSE FUNCTION			*
		9038	*	IS DEFINED ABOVE. CALLING SEQUENCE IS			*
		9039	*	B IPSCAL			*
		9040	*	DC AL2(V\$IFCI)			*
		9041	*	WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL ADDRESS			*
		9042	*	OF ENTRY POINT IDIFNC. EXECUTION IS SUBJECT TO INPUT CONDITIONS			*
		9043	*	DESCRIBED BELOW.			*
		9044	*				*
		9045	*	INPUT			*
		9046	*	* I\$TAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER. THIS IS TO			*
		9047	*	CONTAIN THE CORE ADDRESS OF THE STACK LOCATION IMMEDIATELY			*
		9048	*	FOLLOWING THE FLOATING POINT ELEMENT AT THE TOP OF THE STACK.			*
		9049	*	* I\$XIAR - 2 BYTES, FOR THE PMC ADDRESS REGISTFR. THIS IS TO			*
		9050	*	CONTAIN THE CORE ADDRESS OF THE OPCODE FIELD IN THE PSEUDO			*
		9051	*	INSTRUCTION BEING EXECUTED.			*
		9052	*	* I\$FATP - 2 BYTES, FOR THE USER FUNCTION ACTIVITY TABLE POINTER.			*
		9053	*	THIS IS TO CONTAIN THE CORE ADDRESS OF THE LEFT BYTE OR THE			*
		9054	*	LAST FUNCTION ADDRESS ENTRY PLACED IN THE TABLE.			*
		9055	*	* RUN-TIME STACK - THIS CONTAINS AN UNPACKED FLOATING POINT VALUE			*
		9056	*	IN THE TOP STACK POSITION. THIS VALUE IS TO BE USED AS THE			*
		9057	*	EXECUTION ARGUMENT FOR THE USER FUNCTION ASSOCIATED WITH THE			*
		9058	*	CURRENT 'FCI' INSTRUCTION.			*
		9059	*	* USER FUNCTION ACTIVITY TABLE - THIS CONTAINS VIRTUAL ADDRESS			*
		9060	*	ENTRIES DEFINING THE CURRENTLY ACTIVE USER FUNCTIONS.			*
		9061	*				*
		9062	*	OUTPUT			*
		9063	*	* I\$STAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER. THIS CON-			*

SFADFR - ACTIVATE DATA FILE - PART 1

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 135
		9064	*	TAINS THE CORE ADDRESS OF THE FIRST AVAILABLE STACK LOCATION	*		
		9065	*	AFTER DELETION OF THE USER FUNCTION ARGUMENT.	*		
		9066	*	* I\$FATP - 2 BYTES, FOR THE USER FUNCTION ACTIVITY TABLE POINTER.	*		
		9067	*	WHEN NO ERROR CONDITIONS OCCUR, THIS CONTAINS THE CORE ADDRESS	*		
		9068	*	OF THE LEFT BYTE OF THE CURRENT 'FCI' OPERAND ENTRY TO THE	*		
		9069	*	TABLE.	*		
		9070	*	* I\$WRK1 - 2 BYTES, FOR INTERPRETER COMMON WORKEARA 1. THIS	*		
		9071	*	CONTAINS THE VIRTUAL ADDRESS OF THE FIRST PSEUDO INSTRUCTION IN	*		
		9072	*	THE USER FUNCTION EXPRESSION ASSOCIATED WITH THE CURRENT 'FCI'	*		
		9073	*	INSTRUCTION.	*		
		9074	*	* I\$ERRC - 2 BYTES, FOR THE ERROR CONDITION CODE. THIS CONTAINS	*		
		9075	*	A NULL CODE (I@NERR) WHEN NO ERROR CONDITION EXISTS, OR AN	*		
		9076	*	ERROR CODE SPECIFYING THE PARTICULAR ERROR CONDITION DISCOVERED.	*		
		9077	*	* USER FUNCTION ACTIVITY TABLE - THIS CONTAINS THE CURRENT 'FCI'	*		
		9078	*	VIRTUAL ADDRESS OPERAND AS THE LATEST TABLE ENTRY.	*		
		9079	*	* USER FUNCTION EXPRESSION SAVE AREA - FOR A SINGLE PACKED FLOAT-	*		
		9080	*	ING POINT VALUE. THE FUNCTION ARGUMENT ORIGINALLY IN THE STACK	*		
		9081	*	IS STORED HERE DURING FUNCTION EXECUTION. THIS 'DWA' DEFINED	*		
		9082	*	SAVE AREA IS LOCATED IN VIRTUAL MEMORY JUST BEFORE THE FIRST	*		
		9083	*	EXECUTION PSEUDO INSTRUCTION FOR THE CURRENT FUNCTION.	*		
		9084	*	* USER FUNCTION EXPRESSION RETURN BRANCH - THIS 'BRA' PSEUDO	*		
		9085	*	INSTRUCTION IS LOCATED IN VIRTUAL MEMORY JUST BEFORE THE	*		
		9086	*	EXPRESSION SAVE AREA ABOVE. THE OPERAND FIELD OF THIS INSTRUC-	*		
		9087	*	TION IS SET TO CONTAIN THE VIRTUAL ADDRESS OF THE PSEUDO	*		
		9088	*	INSTRUCTION IMMEDIATELY FOLLOWING THE CURRENT 'FCI' INSTRUCTION	*		
		9089	*	AND PROVIDES RETURN LINKAGE FROM FUNCTION EXPRESSION EXECUTION.	*		
		9090	*		*		
		9091	*	*EXTERNAL REFERENCES	*		
		9092	*	* I\$STCK - ENTRY POINT FOR INTERPRETER ELEMENT STACKING ROUTINE.	*		
		9093	*	* I\$USTK - ENTRY POINT FOR INTERPRETER ELEMENT UNSTACKING ROUTINE.	*		
		9094	*	* I\$CUPF - ENTRY POINT FOR FLOATING POINT VALUE PACKING ROUTINE.	*		
		9095	*	* I\$RTRN - ENTRY POINT FOR PAGING MODULE V.M. RETURN CONTROL RTN.	*		
		9096	*	* I\$STAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER.	*		
		9097	*	* I\$XIAR - 2 BYTES, FOR THE PSEUDO INSTRUCTION ADDRESS REGISTER.	*		
		9098	*	* I\$XPAG - 1 BYTE, FOR THE CURRENT EXECUTION PAGE NUMBER.	*		
		9099	*	* I\$VADR - 2 BYTES, FOR PAGING MUDULE VIRTUAL ADDRESS PARAMETER.	*		
		9100	*	* I\$CADR - 2 BYTES, FOR PAGING MODULE CORE ADDRESS OUTPUT PARAM.	*		
		9101	*	* I\$WRK1 - 2 BYTES, FOR INTERPRETER COMMON WORK AREA 1.	*		
		9102	*	* I\$FATP - 2 BYTES, FOR THE USER FUNCTION ACTIVITY TABLE ENTRY PT.	*		
		9103	*	* I\$FATE - CORE ADDRESS OF USER FUNCTION ACTIVITY TABLE LAST BYTE.	*		
		9104	*	* I\$SLNG - 1 BYTE, FOR ELEMENT STACKING LENGTH PARAM TO ISTACK	*		
		9105	*	* I\$ERRC - 1 BYTE, FOR THE INTERPRETER EXECUTION ERROR CODE.	*		
		9106	*	* I\$CLVA - 1 BYTE, FOR THE LENGTH OF A VIRTUAL ADDRESS.	*		
		9107	*	* I\$CLIF - 1 BYTE, FOR LENGTH OF AN UNPACKED FLOATING POINT VALUE.	*		
		9108	*		*		
		9109	*	*EXITS, NORMAL	*		
		9110	*	CONTROL IS ALWAYS PASSED TO THE PAGING ROUTINE AT ENTRY POINT	*		
		9111	*	I\$RTRN (IPGRTN) FOR A RETURN TO THE INTERPRETER CALLING ROUTINE,	*		
		9112	*		*		
		9113	*	*EXITS, ERROR	*		
		9114	*	CONTROL IS PASSED TO THE PAGING ROUTINE AT ENTRY POINT I\$RTRN	*		
		9115	*	(IPGRTN) WITH PARAMETER I\$ERRC CONTAINING THE APPROPRIATE ERROR	*		
		9116	*	MESSAGE CODE.	*		
		9117	*		*		
		9118	*	*TABLES/WORK AREAS	*		
		9119	*	* AN IN-LINE 'DWA' DEFINED SAVE AREA IS UTILIZED TO STORE THE	*		

SFADFR - ACTIVATE DATA FILE - PART 1

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 136
		9120	*	FUNCTION ARGUMENT (SEE OUTPUT). THIS AREA IS LARGE ENOUGH TO			*
		9121	*	CONTAIN A SINGLE PACKED FLOATING POINT VALUE, AND IMMEDIATELY			*
		9122	*	PRECEDES THE FIRST PSEUDO INSTRUCTION OF THE CURRENT FUNCTION			*
		9123	*	EXECUTION SEQUENCE IN VIRTUAL MEMORY.			*
		9124	*	* USER FUNCTION ACTIVITY TABLE - 22 BYTES, FOR A 'PUSH-DOWN'			*
		9125	*	STACK USED TO CONTROL RECURSIVE USER-DEFINED FUNCTION EXECUTION.			*
		9126	*	THIS TABLE, ESTABLISHED IN EXECUTIVE MODULE INTERP, CONSISTS OF			*
		9127	*	ELEVEN 2-BYTE ENTRY LOCATIONS, THE FIRST OF WHICH IS SET EQUAL			*
		9128	*	TO X'0000' TO GUARD THE BOTTOM OF THE TABLE. EACH TABLE ENTRY			*
		9129	*	LOCATION IS USED TO SAVE THE VIRTUAL ADDRESS OF A USER FUNCTION			*
		9130	*	WHICH IS IN THE PROCESS OF BEING EXECUTED.			*
		9131	*				*
		9132	*	*ATTRIBUTES			*
		9133	*	* REUSABLE			*
		9134	*	* NATURALLY RELOCATABLE			*
		9135	*				*
		9136	*	*CHARACTER CODE DEPENDENCY			*
		9137	*	THE OPERATION OF THIS MODULE DOES NOT DEPEND UPON A PARTICULAR			*
		9138	*	INTERNAL REPRESENTATION OF THE EXTERNAL CHARACTER SET.			*
		9139	*				*
		9140	*	*NOTES			*
		9141	*	ERROR PROCEDURES			*
		9142	*	* ERROR 1 - THE USER FUNCTION ACTIVITY TABLE IS SEARCHED AND A			*
		9143	*	VIRTUAL ADDRESS ENTRY IS FOUND WHICH MATCHES THE CURRENT			*
		9144	*	'FCI' OPERAND. AN ERROR CODE FOR THE MESSAGE 'RECURSIVE			*
		9145	*	FUNCTION REFERENCE' IS ESTABLISHED IN PARAMETER I\$ERRC.			*
		9146	*	* ERROR 2 - ADDITION OF AN ENTRY TO THE USER FUNCTION ACTIVITY			*
		9147	*	TABLE FOR THE CURRENT 'FCI' CAUSES TAKE OVERFLOW. AN ERROR			*
		9148	*	CODE FOR THE MESSAGE 'TOO MANY ACTIVE FUNCTIONS' IS ESTAB-			*
		9149	*	LISHED IN PAPAMETER I\$ERRC.			*
		9150	*	* ERROR 3 - THE VIRTUAL MEMORY LOCATION REFERENCED BY THE			*
		9151	*	CURRENT 'FCI' OPERAND DOES NOT CONTAIN A VALID PMC VIRTUAL			*
		9152	*	ADDRESS. AN ERROR CODE FOR THE MESSAGE 'UNDEFINED USER			*
		9153	*	FUNCTION REFERENCED' IS ESTABLISHED IN PARAMETER I\$ERRC.			*
		9154	*	* WHEN ANY OF THESE CONDITIONS OCCUR, CONTROL IS PASSED			*
		9155	*	IMMEDIATELY TO PAGING MODULE ENTRY POINT I\$RTRN (IPGRTN).			*
		9156	*				*
		9157	*	REGISTER USAGE			*
		9158	*	* REGISTER @BR IS TO CONTAIN THE CORE PAGE BASE ADDRESS			*
		9159	*	ESTABLISHED THROUGH PAGING MODULE CONTROL FOR THE PAGE WHICH			*
		9160	*	INCLUDES IDIFNC.			*
		9161	*	* REGISTER @XR IS NOT SAVED. IT IS USED IN IDIFNC FOR GENERAL			*
		9162	*	PURPOSE INDEXING OPERATIONS.			*
		9163	*				*
		9164	*	SAVED/ESTORED AREAS			*
		9165	*	NONE			*
		9166	*				*
		9167	*	MODIFICATION CONSIDERATIONS			*
		9168	*	NONE			*
		9169	*				*
		9170	*	REQUIRED MODULES			*
		9171	*	* @SYSEQ - COMMON SYSTEM EQUATES.			*
		9172	*	* @ERMEQ - SYSTEM ERROR MESSAGE CODE EQUATES.			*
		9173	*	* \$B@EQU - COMPILER PARAMETER AND CONSTANT EQUATES.			*
		9174	*	* \$I\$EQU - INTEPRETER FIXED LOCATION ADDRESS EQUATES.			*
		9175	*	* \$I@SEQ - INTERPRETER PARAMETER EQUATES (FOR STD, PREC. ONLY)			*

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 137
		9176	*		* \$I@LEQ - INTERPRETER PARAMETER EQUATES (FOR LONG PREC. ONLY)	*		
		9177	*			*		
		9178	*	OTHER		*		
		9179	*	NONE		*		
		9180	*****					

SFADFR - ACTIVATE DATA FILE - PART 1

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 138
			9182	*****	
			9183	* START OF PMC EXECUTION MODULE IDIFNC	*
			9184	*****	
			9185	*	
			9186	* ESTABLISH ADDRESSABILIY FOR THE 'FCI' EXECUTION ROUTINE	
			9187	*	
1B00			9188	ORG *,B@LVPg,0	BEGIN AT PAGE BOUNDARY
		1B00	9189	USING *,@BR	DEFINE 'FCI' ROUTINE BASE ADDR
			9190	*	
			9191	* ENTER IDIFNC - STORE THE FUNCTION CALL INSTRUCTION VADDR OPERAND	
			9192	* (I.E. THE VIRTUAL ADDRESS WHICH REFERENCES THE ACTUAL VIRTUAL	
			9193	* ADDRESS OF THE USER FUNCTION PMC) FOR LATER PAGING OPERATION.	
			9194	*	
		1B00	9195	IDIFNC EQU *	IDIFNC ENTRY POINT
1B00 35 02 0D4C			9196	L I\$XIAR,@XR	LOAD 'FCI' PSEUDO INST CADDR
1B04 2C 01 144A 02			9197	MVC I\$VADR,I@XVAD(@VADDR,@XR)	SET VADDR PAGING PARAMETER
			9198	*	
			9199	* SEARCH THE USER FUNCTION ACTIVITY TABLE FOR AN IDENTICAL VADDR -	
			9200	* A MATCHING TABLE ENTRY SPECIFIES THIS FUNCTION IS ALREADY ACTIVE	
			9201	* (AN ERROR CONDITION).	
			9202	*	
1B09 0E 01 0DE8 0D49			9203	IDI010 ALC I\$FATP,I\$CLVA(@CADDR)	INCR FUNC ACTIVITY TBL POINTER
1B0F 35 02 0DE8			9204	L I\$FATP,@XR	LOAD FUNC ACTIVITY TBL POINTER
1B13 76 02 A3			9205	IDI020 A IDIBM2(,@BR),@XR	DECR TBL IDX REG TO PRIOR ENTRY
1B16 BD 56 00			9206	CLI IDIFVA-1(,@XR),@VENTA	DUMMY ENTRY (BOTTOM OF TBL) ?
1B19 F2 82 10			9207	JL IDI040	* GO TEST FOR TABLE OVERFLOW
1B1C 2D 01 144A 01			9208	CLC I\$VADR,IDIFVA(@VADDR,@XR)	IF ENTRY NOT EQUAL 'FCI' OPERND
1B21 D0 01 13			9209	BNE IDI020(,@BR)	* GO REPEAT LOOP FOR NEXT ENTRY
			9210	*	
			9211	* ACTIVE FUNCTION - SET 'RECURSIVE FUNCTION REFERENCE' ERROR MESSAGE	
			9212	*	
1B24 3C C5 0CBC			9213	IDI030 MVI I\$ERRC,@@E728	SET ERROR MESSAGE CODE
1B28 C0 87 12D3			9214	B I\$RTRN	RETURN TO THE INTERPRETER
			9215	*	
			9216	* TEST FOR FILLED FUNCTION ACTIVITY TABLE	
			9217	*	
1B2C 1D 01 0DE8 A8			9218	IDI040 CLC I\$FATP,IDIFTE(@CADDR,@BR)	IF POINTER NOT PAST END OF TBL
1B31 F2 04 08			9219	JNH IDI060	* GO PLACE NEW ENTRY IN TABLE
			9220	*	
			9221	* TABLE OVERFLOW - SET 'TOO MANY ACTIVE FUNCTIONS' ERROR MESSAGE	
			9222	*	
1B34 3C C8 0CBC			9223	IDI050 MVI I\$ERRC,@@E732	SET ERROR MESSAGE CODE
1B38 C0 87 12D3			9224	B I\$RTRN	RETURN TO THE INTERPRETER
			9225	*	
			9226	* INSERT 'FCI' OPERAND AT OFF TO THE FUNCTION ACTIVITY TABLE	
			9227	*	
1B3C 35 02 0DE8			9228	IDI060 L I\$FATP,@XR	LOAD FUNC ACTIVITY TABLE PT
1B40 8C 01 01 144A			9229	MVC IDIFVA(,@XR),I\$VADR(@VADDR)	ENTER FUNC VADDR AT TABLE TOP
			9230	*	
			9231	* STACK THE ACTUAL FUNCTION EXECUTION VADDR (PAGING PARAMETER HAS	
			9232	* ALREADY BEEN SET FOR THIS OPERATION) - WHEN COMPLETED, STACK WILL	
			9233	* CONTAIN THE FUNCTION EXECUTION ARGUMENT VALUE FOLLOWED BY THE	
			9234	* EXECUTION VIRTUAL ADDRESS.	
			9235	*	
1B45 35 02 0D4E			9236	IDI070 L I\$STAK,@XR	LOAD THE STACK POINTER
1B49 3C 01 0BA2			9237	MVI I\$SLNG,@VADDR-1	SET STACK RTN LENGTH PARAMETER

SFADFR - ACTIVATE DATA FILE - PART 1

ERR LOC		OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 139
1B4D	C0	87	0B50	9238	B	I\$STCK			LINK TO STACK FUNC EXEC VADDR
				9239	*				
				9240	*	TEST FOR AN UNDEFINED USER FUNCTION (AN ERROR CONDITION).			
				9241	*				
1B51	BD	56	00	9242	IDI080	CLI I@SVAD-1(,@XR),@VENTA			IF FUNC EXEC VADDR IS DEFINED
1B54	F2	02	08	9243		JNL IDI100			* GO PROCESS FUNC EXEC ARGUMENT
				9244	*				
				9245	*	UNDEFINED VADDR - SET 'UNDEFINED USER FUNCTION REFERENCE' ERROR MSG			
				9246	*				
				9247	IDI090	MVI I\$ERRC,@@E701			SET ERROR MESSAGE CODE
1B5B	C0	87	12D3	9248	B	I\$RTRN			RETURN TO THE INTERPRETER

SFADFR - ACTIVATE DATA FILE - PART 1

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 140
					9250	*				
					9251	*	MOVE THE STACKED FUNCTION EXECUTION ARGUMENT VALUE TO IN-LINE			
					9252	*	PMC WORK AREA ASSOCIATED WITH THE REFERENCED USER FUNCTION.			
					9253	*				
1B5F	9E	00	01	A4	9254	IDI100	ALC I@SVAD(,@XR),IDILBI(@VADDR-1,@BR) INCR FUNC EXEC VADDR TO			
1B63	6C	01	AA	01	9255		MVC IDIVAD(,@BR),I@SVAD(@VADDR,@XR) * REF WRK AREA, THEN SAVE			
1B67	2C	01	144A	01	9256		MVC I\$VADR,I@SVAD(@VADDR,@XR) SET VADDR PAGING PARM FOR WK/A			
1B6C	0F	00	0D4E	0D44	9257		SLC I\$STAK,I\$CL1F(@CADDR-1) DECR STACK PT TO REF FUNC ARG			
1B72	35	02	0D4E		9258		L I\$STAK,@XR LOAD THE STACK POINTER			
1B76	C0	87	0A85		9259		B I\$CUPF LINK TO PACK THE FUNC ARGUMENT			
1B7A	C0	87	0BB0		9260		B I\$USTK LINK TO UNSTACK THE FUNC ARG			
					9261	*				
					9262	*	THE UNSTACKING OPERATION HAS ACCESSED THE FUNCTION EXECUTION PMC -			
					9263	*	SET THE FUNCTION EXECUTION RETURN BRANCH INSTRUCTION TO REFERENCE			
					9264	*	THE PSEUDO INSTRUCTION IMMEDIATELY FOLLOWING THE CURRENT 'FCI' INST			
					9265	*				
1B7E	1F	00	144C	A4	9266	IDI110	SLC I\$CADR,IDILBI(@CADDR-1,@BR) GET CADDR OF RETURN BR INST			
1B83	35	02	144C		9267		L I\$CADR,@XR LOAD RETURN BRANCH INST ADDR			
1B87	8C	00	01	0C61	9268		MVC I@XVAD-1(,@XR),I\$XPAG(1) SET RETURN PAGE = CURRENT PG			
1B8C	8C	00	02	0D4C	9269		MVC I@XVAD(,@XR),I\$XIAR(@VADDR-1) SET RETURN PAGE DISP = DISP			
1B91	9E	00	02	A5	9270		ALC I@XVAD(,@XR),IDILFI(@VADDR-1,@BR) * OF INST AFTER 'FCI'			
					9271	*				
					9272	*	ESTABLISH EXECUTION VIRTUAL ADDRESS FOR 1ST INSTRUCTION IN THE USER			
					9273	*	FUNCTION EXECUTION PMC - RETURN TO INTERPRETER			
					9274	*				
1B95	5E	00	AA	A6	9275	IDI130	ALC IDIVAD(,@BR),IDILPV(@VADDR-1,@BR) INCR FUNC EXEC VADDR			
1B99	1C	01	0D59	AA	9276		MVC I\$WRK1,IDIVAD(@VADDR,@BR) SET NEW EXECUTION VIR/U111			
1B9E	C0	87	12D3		9277		B I\$RTRN RETURN TO THE INTERPRETER			
					9278	*				
					9279		*****			

SFADFR - ACTIVATE DATA FILE - PART 1

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 141
		9281		*****	
		9282		* 'FCI' EXECUTION ROUTINE CONSTANTS	*
		9283		*****	
		9284		*	
1BA2	FFFE	1BA3	9285	IDIBM2 DC IL(@REGL) '-2'	VIRTUAL ADDR LENGTH DECREMENT
		9286		*	
1BA4	05	1BA4	9287	IDILBI DC AL1(B@LBRA+B@LDWA)	LENGTH OF 'BRA' & 'DWA' PMC'S
1BA5	03	1BA5	9288	IDILFI DC AL1(B@LFCI)	LENGTH OF 'FCI' PSEUDO INST
1BA6	05	1BA6	9289	IDILPV DC AL1(I@LPFV)	LENGTH OF PACKED FLT PT VALUE
		9290		*	
1BA7	0DE6	1BA8	9291	IDIFTE DC AL(@CADDR)(I\$FATE)	FUNC ACTIVITY TBL ENDING ADDR
		9292		*	
		9293		*****	
		9294		* 'FCI' EXECUTION ROUTINE WORK AREAS	*
		9295		*****	
		9296		*	
1BA9		1BAA	9297	IDIVAD DS CL(@VADDR)	FUNC EXECUTION VADDR WORK AREA
		9298		*	
		9299		*****	
		9300		* 'FCI' EXECUTION ROUTINE EQUATES	*
		9301		*****	
		9302		*	
		0001	9303	IDIFVA EQU 1	DISP FOR FUNC TBL VADDR ENTRY
		9304		*	
		9305		*****	
		9306		*	
		9307		* END OF FUNCTION CALL INDIRECT EXECUTION PMC ROUTINE CODING	*****

SFADFR - ACTIVATE DATA FILE - PART 1

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 142
	1C00				9309	ORG	*,B@LVPG,0			PLACE MODULE AT PAGE BOUNDARY
				1C00	9310	SFADFR	EQU *			ENTRY POINT
				1C00	9311		USING SFADFR,@BR			ESTABLISH BASE ADDRESSING
					9312	****				****
					9313	*	ACCESS FILE DIRECTORY 2 AND LOCK IT IN CORE			*
					9314	****				****
	1C00	5C	01	F8 EE	9315	MVC	SFAWK1(@VADDR,@BR),SFAVD1(@BR)			SET UP TO GET DIR 1
	1C04	1C	01	144A F0	9316	MVC	I\$VADR,SFAVD2(@VADDR,@BR)			PASS VADDR OF DIRECTORY 2
	1C09	C0	87	1354	9317	B	I\$LOCK			GET D2 AND LOCK IT IN CORE
	1C0D	0C	01	0D59 144C	9318	MVC	I\$WRK1(@CADDR),I\$CADR			SAVE CADDR OF DIRECTORY 2
					9319	****				****
					9320	*	POINT AT FILENAME IN RUN-TIME STACK			*
					9321	****				****
	1C13	7C	00	F6	9322	MVI	SFACTR(@BR),@ZERO			INITIALIZE COUNTER
	1C16	0F	00	0D4E 0D46	9323	SLC	I\$STAK,I\$CL1C			POINT AT LEFT BYTE OF FILENAME
	1C1C	1E	00	0D4E F1	9324	ALC	I\$STAK,SFA001(@BR)			
	1C21	35	02	0D4E	9325	SFA010 L	I\$STAK,@XR			
	1C25	BD	40	00	9326	CLI	@ZERO(@XR),@BLANK			IS THIS CHARACTER A BLANK ?
	1C28	F2	81	0C	9327	JE	SFA020			YES-GO CHECK FILENAME LENGTH
	1C2B	5E	00	F6 F1	9328	ALC	SFACTR(@BR),SFA001(@BR)			NO-ADD 1 TO FILENAME LENGTH
	1C2F	1E	00	0D4E F1	9329	ALC	I\$STAK,SFA001(@BR)			INCREMENT STACK ADDRESS
	1C34	D0	87	21	9330	B	SFA010(@BR)			CONTINUE LOOP
					9331	****				****
					9332	*	CHECK STACKED FILENAME LENGTH			*
					9333	****				****
	1C37	7D	08	F6	9334	SFA020 CLI	SFACTR(@BR),@\$L1DF			IS FILENAME LENGTH < 8 ?
	1C3A	F2	04	07	9335	JNH	SFA030			YES-CHECK FOR FILENAME PRESEN/
	1C3D	3C	B6	0CBC	9336	MVI	I\$ERRC,@E710			NO-FILE NOT ALLOCATED
	1C41	F2	87	9F	9337	J	SFA120			GO RETURN TO INTRP
	1C44	7D	00	F6	9338	SFA030 CLI	SFACTR(@BR),@ZERO			IS FILENAME LENGTH > 0 ?
	1C47	F2	84	07	9339	JH	SFA040			YES-SEARCH DIRECTORY 1
	1C4A	3C	B6	0CBC	9340	MVI	I\$ERRC,@E710			NO-FILE NOT ALLOCATED
	1C4E	F2	87	92	9341	J	SFA120			GO RETURN TO INTRP
					9342	****				****
					9343	*	REMOVE FILENAME FROM RUN-TIME STACK			*
					9344	****				****
	1C51	1F	00	0D4E F6	9345	SFA040 SLC	I\$STAK,SFACTR(@BR)			POINT AT LEFT BYTE OF FILENAME
	1C56	1E	00	0D4E F2	9346	ALC	I\$STAK,SFA007(@BR)			POINT AT FILENAME FIELD
	1C5B	4C	01	86 0D4E	9347	MVC	SFA070+@OP1(2,@BR),I\$STAK			SET UP D1 COMPARISON INSTR
	1C60	1F	00	0D4E F3	9348	SLC	I\$STAK,SFA008(@BR)			REMOVE FILENAME FROM STACK
					9349	****				****
					9350	*	ACCESS PAGE 1 OF DIRECTORY 1 (PAGE 2 IF NECESSARY)			*
					9351	****				****
	1C65	1C	01	144A F8	9352	SFA050 MVC	I\$VADR,SFAWK1(@VADDR,@BR)			ACCESS FILE DIRECTORY 1 AND
	1C6A	C0	87	1358	9353	B	I\$CVAD			* LOAD IT INTO CORE 1-3
	1C6E	35	02	144C	9354	L	I\$CADR,@XR			POINT @XR AT FILE DIRECTORY 1
	1C72	6C	00	F7 1F	9355	MVC	SFAWK1-1(@B1,@BR),@\$D1SW(@XR)			SET UP TO READ PAGE 2
	1C76	BD	00	08	9356	SFA060 CLI	@\$D1BF(@XR),@ZERO			IS D1 FILENAME NAME NULL ?
	1C79	F2	01	07	9357	JNE	SFA070			NO-COMPARE NAMES
	1C7C	3C	B6	0CBC	9358	SFA065 MVI	I\$ERRC,@E710			ERROR-FILENAME NOT FOUND
	1C80	F2	87	57	9359	J	SFA115			GO-RETURN TO INTRP
	1C83	2D	07	0000 08	9360	SFA070 CLC	*-*(@\$L1DF),@\$D1BF(@XR)			COMPARE FILENAMES
	1C88	F2	81	29	9361	JE	SFA100			EQUAL-ACCESS D2 ENTRY
	1C8B	5E	00	F4 F5	9362	ALC	SFAD2D(@BR),SFA032(@BR)			INCREMENT D2 DISPLACEMENT
	1C8F	7D	B0	F4	9363	SFA075 CLI	SFAD2D(@BR),SFA0B0			D2 DISPLACEMENT < X'B0' ?
	1C92	F2	84	06	9364	JH	SFA080			NO-TEST FOR 2 D1 PAGES PRESENT

SFADFR - ACTIVATE DATA FILE - PART 1

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 143
	1C95	E2	02	20	9365	LA	@\$L1E(,@XR),@XR			YES-INCR D1 DISP & CONTINUE LOOP
	1C98	D0	87	76	9366	B	SFA060(,@BR)			GO COMPARE FILENAMES
	1C9B	38	10	03E0	9367	SFA080 TBN	\$DBGUF,\$IOPGS			IS 2 SECTOR SW ON ?
	1C9F	F2	10	03	9368	JT	SFA090			YES-GO ACCESS PAGE 2
	1CA2	D0	87	7C	9369	B	SFA065(,@BR)			NO-GO RETURN ERR TO INTRP
					9370	****				****
					9371	*	RESET LOOP TO ACCESS PAGE 2 OF D1 AND CONTINUE			*
					9372	****				****
	1CA5	7C	FF	90	9373	SFA090 MVI	SFA075+@Q(,@BR),@PWAIT			SET UP LOOP TO CONTINUE
	1CA8	1C	01	144A EE	9374	MVC	I\$VADR(@VADDR),SFAVD1(,@BR)			UNLOCK PAGE 1 OF DIRECTORY
	1CAD	C0	87	1350	9375	B	I\$UNLK			* 1 IF ENTRY IS IN PAGE 2
	1CB1	D0	87	65	9376	B	SFA050(,@BR)			GO ACCESS PAGE 2 OF D1
					9377	****				****
					9378	*	NAME FILE DIRECTORY 2 AND ACTIVATE FILE			*
					9379	****				****
	1CB4	74	02	CF	9380	SFA100 ST	SFA110+@OP1(,@BR),@XR			SAVE D1 POINTER FOR LATER
	1CB7	35	02	0D59	9381	L	I\$WRK1,@XR			POINT AT DIRECTORY 2
	1CBB	9C	00	01 F4	9382	MVC	@\$D2CF(,@XR),SFAD2D(,@BR)			SET CURR FILE DISP IN D2
	1CBF	B6	02	01	9383	A	@\$D2CF(,@XR),@XR			INCR @XR TO CURR ENTRY IN D2 ?
	1CC2	BD	00	00	9384	CLI	@\$D2DC(,@XR),@ZERO			IS THIS FILE ACTIVE YET ?
	1CC5	F2	01	12	9385	JNE	SFA115			YES-GO RETURN TO INTRP
					9386	****				****
					9387	*	FIRST REFERENCE TO CURRENT C!LE			*
					9388	****				****
	1CC8	34	02	0D59	9389	ST	I\$WRK1,@XR			SAVE CURR D2 POINTER
	1CCC	C2	02	0000	9390	SFA110 LA	*-*,@XR			RESTORE D1 POINTER
	1CD0	34	02	0D5B	9391	ST	I\$WRK2,@XR			SAVE D1 POINTER FOR PAGE 2
					9392	****				****
					9393	*	ACCESS PAGE 2 TO CONTINUE FILE ACTIVATION			*
					9394	*	PROCEDURE AND RETURN TO CALLER WHEN FINISHED.			*
					9395	****				****
	1CD4	C0	87	12B1	9396	B	I\$CALL			EXECUTE PART 2 OF SFADFR
	1CD8	5000			1CD9 9397	DC	AL(@VADDR)(V\$SFA2)			VADDR OF PART 2
	1CDA	1C	01	144A F8	9398	SFA115 MVC	I\$VADR,SFAWK1(@VADDR,@BR)			UNLOCK FILE DIRECTORY 1 BEFORE
	1CDF	C0	87	1350	9399	B	I\$UNLK			* RETURNING TO CALLER
	1CE3	7C	B0	90	9400	SFA120 MVI	SFA075+@Q(,@BR),SFA0B0			MAKE SFADFR REUSABLE
	1CE6	7C	40	37	9401	MVI	SFA020(,@BR),@\$D2E1			
	1CE9	C0	87	12D3	9402	B	I\$RTRN			RETURN TO CALLER
					9403	****				****
					9404	*	EQUATES, CONSTANTS AND WORKAREA FOR PART 1			*
					9405	****				****
				00B0	9406	SFA0B0 EQU	X'B0'			DISP TO 8TH ENTRY
	1CED	0000			1CEE 9407	SFAVD1 DC	AL(@VADDR)(V\$SFD1)			DIRECTORY 1 VM ADDR
	1CEF	0100			1CF0 9408	SFAVD2 DC	AL(@VADDR)(V\$SFD2)			DIRECTORY 2 VM ADDR
	1CF1	01			1CF1 9409	SFA001 DC	XL1'01'			CONSTANT 1 FOR COUNTING
	1CF2	07			1CF2 9410	SFA007 DC	XL1'07'			INCREMENT ADDR OF STACK FILENAME
	1CF3	08			1CF3 9411	SFA008 DC	XL1'08'			CONSTANT TO REMVE NAME FRM STACK
	1CF4	40			1CF4 9412	SFAD2D DC	AL1(@\$D2E1)			DISP TO 1ST D2 ENTRY
	1CF5	10			1CF5 9413	SFA032 DC	AL1(@\$L2E)			INCREMENT D2 DISP VALUE
	1CF6				1CF6 9414	SFACTR DS	CL1			USED TO DECREMENT STACK POINTER
	1CF7				1CF8 9415	SFAWK1 DS	AL(@VADDR)			SFADFR WORKAREA

SFPUTR - PROLOGUE - VM PUT ROUTINE

ERR	LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 144
			9417		*****			
			9418	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
			9419	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
			9420	*				*
			9421		*****			*
			9422	*	STATUS			*
			9423	*	VERSION 1 MODIFICATION 0			*
			9424	*				*
			9425	*	FUNCTION			*
			9426	*	* SFPUTR OUTPUTS A SINGLE ARITHMETIC OR CHARACTER ELEMENT TO A			*
			9427	*	SEQUENTIAL DATA FILE. THIS DATA FILE MAY BE TO DISK, CARD,			*
			9428	*	PRINTER, OR CRT.			*
			9429	*	* FOR OUTPUT TO THE PRINTER OR CRT, THE CONSTANT IS CONVERTED TO			*
			9430	*	EXTERNAL NOTATION AND OUTPUTTED IMMEDIATELY.			*
			9431	*	* FOR A CARD OUTPUT FILE, THE CONSTANT IS CONVERTED AND PLACED IN			*
			9432	*	A BUFFER. WHEN THE BUFFER IS FILLED, A CARD I/O ROUTINE IS			*
			9433	*	CALLED TO PUNCH THE CONTENTS OF THE BUFFER.			*
			9434	*	* FOR A DISK FILE, THE CONSTANT IS PLACED IN A BUFFER.			*
			9435	*	WHEN THE BUFFER IS FULL, IT IS TRANSFERRED TO THE AT FILE IN			*
			9436	*	THE FILE LIBRARY.			*
			9437	*				*
			9438	*	ENTRY POINTS			*
			9439	*	* THE ENTRY IS SFPUTR. THE FORMAT OF THE CALLING SEQUENCE IS AS			*
			9440	*	FOLLOWS:			*
			9441	*	B MALL			*
			9442	*	DC AL2(V\$XSPT)			*
			9443	*				*
			9444	*	INPUT			*
			9445	*	* THE DISPLACEMENT TO THE SPECIFIED ENTRY IS IN THE D2 HEADER.			*
			9446	*	* THE ADDRESS OF THE LOCATION OF THE CONSTANT IS AT I\$STAK.			*
			9447	*				*
			9448	*	OUTPUT			*
			9449	*	* THE CONSTANTS ARE OUTPUTTED TO THE SPECIFIED DATA FILE.			*
			9450	*				*
			9451	*	EXTERNAL REFERENCES			*
			9452	*	I\$LOCK - PAGING ENTRY TO LOCK PAGE IN CORE			*
			9453	*	I\$CALL - PAGING ENTRY TO PASS CONTROL TO A SUBROUTINE			*
			9454	*	I\$MDFY - PAGING ENTRY TO SET WRITE BACK INDICATOR			*
			9455	*	I\$UNLK - PAGING ENTRY TO UNLOCK CORE PAGE			*
			9456	*	I\$RTRN - RETURN IN CALLING ROUTINE			*
			9457	*	I\$ERRC - ERROR CODE PASSING AREA			*
			9458	*	I\$STAK - ADDRESS OF 2-BYTE FIELD CONTAINING THE ADDRFS OF THE			*
			9459	*	RUN-TIME STACK			*
			9460	*	DKDISK - DISK IOCR			*
			9461	*	SFLOAD - TRANSFER VM RESIDENT BUFFER TO SAVED FILE ROUTINE			*
			9462	*	FZXPRS - PRINTER OR CRT OUTPUT ROUTINE			*
			9463	*	DFCOUT - CARD OUTPUT ROUTINE			*
			9464	*				*
			9465	*	EXITS, NORMAL			*
			9466	*	THE CONSTANTS ARE OUTPUTTED AND CONTROL RETURNED VIA I\$RTRN			*
			9467	*				*
			9468	*	EXITS, ERROR			*
			9469	*	IN THE EVENT OF AN ERROR, AN ERROR CODE IS PLACED IN I\$ERRC, AND			*
			9470	*	CONTROL RETURNED VIA I\$RTRN.			*
			9471	*				*
			9472	*	TABLES/WORK AREAS			*

SFPUTR - PROLOGUE - VM PUT ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 145
		9473	*	D2 - DIRECTORY 2 - CONTAINING CURRENT FILE USAGE INFORMATION	*		
		9474	*	FOR THE PROGRAM FILE.	*		
		9475	*		*		
		9476	*	ATTRIBLTES	*		
		9477	*	REUSABLE, RELOCATABLE	*		
		9478	*		*		
		9479	*	CHARACTER CODE DEPENDENCY	*		
		9480	*	THE OPERATION OF THIS MODULE DEPENDS UPON AN INTERNAL REPRESENTA-	*		
		9481	*	TION OF THE EXTERNAL CHARACTER SET WHICH IS EQUIVALENT TO THE ONE	*		
		9482	*	USED AT ASSEMBLY TIME.	*		
		9483	*		*		
		9484	*	NOTES	*		
		9485	*	ERROR PROCEDURES	*		
		9486	*	AN ERROR CODE IS PLACED IN I\$ERRC AND CONTROL RETURNED TO THE	*		
		9487	*	USER.	*		
		9488	*		*		
		9489	*	REGISTER USAGE	*		
		9490	*	REGISTER 1 (@BR) IS USED AS THE BASE REGISTER.	*		
		9491	*	REGISTER 2 (@XR) IS USED TO REFERENCE THE CURRENT FILE ENTRY	*		
		9492	*	AND D2, THE CURRENT BURFER LOCATION, AND THE RUN-TIME STACK.	*		
		9493	*		*		
		9494	*	SAVED/RESTORED AREAS	*		
		9495	*	NONE.	*		
		9496	*		*		
		9497	*	MODIFICATION CONSIDERATIONS	*		
		9498	*	NONE.	*		
		9499	*		*		
		9500	*	REQUIRED MODULES	*		
		9501	*	@SYSEQ - SYSTEM EQUATES	*		
		9502	*	@FXDEQ - FIXED ADDRESSES FOR SYSTEM NUCLEUS	*		
		9503	*	@CANEQ - SYSTEM LOCATION EQUATES	*		
		9504	*	@WKAEQ - SYSTEM WORKAREA DADDR EQUATES	*		
		9505	*	@CYOEQ - CYLINDER ZERO EQUATES	*		
		9506	*	@VMDEQ - VM DIRECTORY EQUATES	*		
		9507	*	@ERMEQ - ERROR MESSAGE EQUATES	*		
		9508	*	@SPFEQ - SYSTEM PROGRAM FILE EQUATES	*		
		9509	*	\$V\$EQU - FIXED VIRTUAL ADDRESSES	*		
		9510	*	\$B@EQU - BASIC COMPILER EQUATES	*		
		9511	*	\$I@EQU - BASIC INTERPRETER EQUATES	*		
		9512	*	\$I@SEQ/\$I@LEQ - STANDARD/LONG PRESISION EXECUTION EQUATES	*		
		9513	*		*		
		9514	*	OTHER	*		
		9515	*	NONE.	*		
		9516	*	*****	*		

SFPUTR - PROLOGUE - VM PUT ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 146

1D00				9518	ORG	* ,B@LVPG,0	PLACE MODULE AT PAGE BOUNDARY
			1D00	9519	SFPBS1 EQU	*	ESTABLISH BASE
			1D00	9520	USING	SFPBS1,@BR	* REGISTER USAGE
				9521	*		
			1D00	9522	SFPUTR EQU	*	ENTRY POINT
1D00	1C	01	144A	F2	9523	MVC I\$VADR,SFPVD2(@VADDR,@BR)	ACCESS VM DIRECTORY 2
1D05	C0	87	1354		9524	B I\$LOCK	LOCK IT IN CORE
1D09	C0	87	1349		9525	B I\$MDFY	SET INDR FOR WRITE BACK
1D0D	35	02	144C		9526	L I\$CADR,@XR	POINT @XR TO THE DIRECTORY
1D11	B6	02	01		9527	A @\$D2CF(,@XR),@XR	INCR TO CURRENT ENTRY
1D14	B8	04	01		9528	TBN @\$D2IO(,@XR),@\$M2CO	CURRENT USAGE = OUTPUT ?
1D17	F2	10	16		9529	JT SFP100	YES, GO CHECK DEVICE CODE
1D1A	B8	08	01		9530	TBN @\$D2IO(,@XR),@\$M2CI	CURRENT USAGE = INPUT ?
1D1D	F2	10	06		9531	JT SFP050	YES, GO TO FILE USAGE ERROR
1D20	B8	40	01		9532	TBN @\$D2IO(,@XR),@\$M2FO	FILE USAGE = OUTPUT ?
1D23	F2	10	07		9533	JT SFP075	YES, GO SET CURRENT USAGE
1D26	3C	B7	0CBC		9534	SFP050 MVI I\$ERRC,@@E712	SET INVALID FILE USAGE ERR CODE
1D2A	F2	87	51		9535	J SFP150	GO TO RETURN
1D2D	BA	04	01		9536	SFP075 SBN @\$D2IO(,@XR),@\$M2CO	SET CURRENT USAGE TO OUTPUT
				9537	*		
				9538	*	FILE MAY CURRENTLY BE USED FOR OUTPUT - CHECK DEVICE	
				9539	*		
1D30	4C	01	EB	044B	9540	SFP100 MVC SFPDEV(@CADDR,@BR),\$PRDEV	SAVE SYS OUTPUT INDR
1D35	B8	10	00		9541	TBN @\$D2DC(,@XR),@\$MBPT	PRINTER FILE ?
1D38	F2	90	08		9542	JF SFP120	NO, GO CHECK FOR CRT
1D3B	1C	01	044B	EE	9543	MVC \$PRDEV,SFPMPT(@CADDR,@BR)	SET SYS OUTPUT INDR FOR PRINTER
1D40	F2	87	11		9544	J SFP130	
1D43	B8	08	00		9545	SFP120 TBN @\$D2DC(,@XR),@\$MBCR	CRT FILE ?
1D46	F2	90	42		9546	JF SFP175	NO, GO SET UP BUFFER PAGE
1D49	1C	01	044B	F0	9547	MVC \$PRDEV,SFPCRT(@CADDR,@BR)	SET SYS OUTPUT INDR
1D4E	0E	00	044A	043B	9548	ALC \$PRDEV-1,\$EXFTR(1)	* FOR CRT
				9549	*		
				9550	*	OUTPUT TO PRINTER OR CRT COMMON ROUTINE	
				9551	*		
1D54	B8	02	01		9552	SFP130 TBN @\$D2IO(,@XR),@\$M2EF	END OF FILE ?
1D57	F2	90	07		9553	JF SFP133	NO, BYPASS SET FOR CARRIAGE RTN
1D5A	3C	07	0D57		9554	MVI I\$PARM,B@PRRC	SET PARM TO CARRIAGE RETURN
1D5E	F2	87	12		9555	J SFP140	GO TO OUTPUT CALL
1D61	35	02	0D4E		9556	SFP133 L I\$STAK,@XR	POINT @XR TO CONSTANT IN STACK
1D65	B8	40	00		9557	TBN I@STAT(,@XR),B@DTYP	CHAR. CONSTANT IN STACK ?
1D68	F2	10	04		9558	JT SFP135	YES, BYPASS CONVERSION
1D6B	C0	87	0A27		9559	B I\$CPUF	CONVERT TO OUTPUT FORMAT
1D6F	3C	02	0D57		9560	SFP135 MVI I\$PARM,B@PRPL	SET FOR PRINT SPACE LONG ZONE
1D73	C0	87	12B1		9561	SFP140 B I\$CALL	GO OUTPUT
1D77	3400			1D78	9562	DC AL2(V\$XSPR)	* CONSTANT
1D79	1C	01	044B	EB	9563	MVC \$PRDEV,SFPDEV(@CADDR,@BR)	RESTORE SYS OUTPUT INDR
				9564	*		
				9565	*	COMMON RETURN TO CALLING ROUTINE	
				9566	*		
1D7E	1C	01	144A	F2	9567	SFP150 MVC I\$VADR,SFPVD2(@VADDR,@BR)	SET UP ACCESS TO VM DIRECTORY 2
1D83	C0	87	1350		9568	B I\$UNLK	UNLOCK IT
1D87	C0	87	12D3		9569	B I\$RTRN	RETURN TO CALLING ROUTINE
				9570	*		
				9571	*	CARD OR DISK FILE - IF DISK FILE, CALL SFPUT2	
				9572	*		
1D8B	B8	20	00		9573	SFP175 TBN @\$D2DC(,@XR),@\$MBCD	CARD FILE ?

SFPUTR - PROLOGUE - VM PUT ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 147

1D8E	F2 10 09		9574	JT	SFP200	YES, GO CHECK FOR END OR FILE
1D91	C0 87 12B1		9575	B	I\$CALL	DISK FILE. GO TO PAGE 2 OF
1D95	1E00	1D96	9576	DC	AL(@VADDR)(V\$XSPT+B@BLSZ)	* SFPUTR
1D97	D0 87 7E		9577	B	SFP150(, @BR)	GO UNLOCK D2 AND RETURN
			9578	*		
			9579	*	CARD FILE	
			9580	*		
1D9A	74 02 E0		9581	SFP200 ST	SFPCXI(, @BR), @XR	SAVE D2 ENTRY POINTER
1D9D	B8 02 01		9582	TBN	@\$D2IO(, @XR), @\$M2EF	END OF FILE INDR ON ?
1DA0	F2 10 3A		9583	JT	SFP320	GO GET TO CARD FILE PART 2
			9584	*		
			9585	*	DATA ITEM IN STACK MUST BE CONVERTED TO CARD OUTPUT	
			9586	*	FORMAT AND ITS LENGTH DETERMINED.	
			9587	*		
1DA3	35 02 0D4E		9588	SFP220 L	I\$STAK, @XR	POINT @XR AT DATA ITEM
1DA7	B8 40 00		9589	TBN	I@STAT(, @XR), B@DTYP	CHARACTER CONSTANT ?
1DAA	F2 90 30		9590	JF	SFP320	NO, GO CONVERT TO UNPACKED FLT
			9591	*		
			9592	*	CHARACTER CONSTANT TO BE FORMATED	
			9593	*		
1DAD	BC 7D 00		9594	MVI	I@STAT(, @XR), B@SQUO	MOVE A QUOTE MARK TO LEFT END
1DB0	7C 13 EB		9595	MVI	SFPDIC(, @BR), I@LCRF+1	INITLZ CHAR ELEMENT LENGTH + 1
1DB3	E2 02 01		9596	SFP230 LA	@B1(, @XR), @XR	INCR @XR BY 1
1DB6	5F 00 EB EC		9597	SLC	SFPDIC(, @BR), SFPONE(1, @BR)	DECR COUNTER BY ONE
1DBA	F2 81 13		9598	JZ	SFP250	JUMP OUT OF LOOP IF ZERO
1DBD	BD 7D 00		9599	CLI	@ZERO(, @XR), B@SQUO	IS REF. CHAR. A QUOTE MARK ?
1DC0	D0 01 B3		9600	BNE	SFP230(, @BR)	NO, GO INCR TO NEXT CHARACTER
1DC3	AC 10 12 11		9601	MVC	I@LCRF(, @XR), I@LCRF-1(I@LCRF-1, @XR)	MOVE CONT TO RIGHT
1DC7	E2 02 01		9602	LA	@B1(, @XR), @XR	INCR @XR BY ONE
1DCA	BC 7D 00		9603	MVI	@ZERO(, @XR), B@SQUO	MOVE IN A MATCHING QUOTE
1DCD	D0 87 B3		9604	B	SFP230(, @BR)	GO INCR TO NEXT CHARACTER
1DD0	BC 7D 00		9605	SFP250 MVI	@ZERO(, @XR), B@SQUO	MOVE IN RIGHT CLOSING QUOTE
			9606	*		
			9607	*	ENTIRE CHARACTER ELEMENT HAS BEEN FORMATTED	
			9608	*		
1DD3	34 02 0D59		9609	ST	I\$WRK1, @XR	SAVE CHAR-CON RIGHT END ADDR
1DD7	0F 01 0D59 0D4E		9610	SLC	I\$WRK1, I\$STAK(@CADDR)	DECR BY START CADDR - LENGTH-1
			9611	*		
			9612	*	ACCESS PART 2 OR PUT TO CARD FILE ROUTINE	
			9613	*		
1DDD	C2 02 0000		9614	SFP320 LA	*-*, @XR	RESTORE D2 ENTRY ROUTINE
		1DE0	9615	SFPCXI EQU	SFP320+@OP1	* FROM POINTER SAVE AREA
			9616	*		
1DE1	C0 87 12B1		9617	B	I\$CALL	GO TO CARD FILE ENTRY IN
1DE5	2000	1DE6	9618	DC	AL(@VADDR)(V\$XSPT+3*B@BLSZ)	* SFPUTR - PART 4
			9619	*		
			9620	*	CARD OUTPUT COMPLETED - GO TO SFPUTR GENERAL RETURN	
			9621	*		
1DE7	D0 87 7E		9622	B	SFP150(, @BR)	GO TO UNLOCK D2 RCD
			9623	*		
			9624	*	PART 1 - WORKAREAS, EQUATES AND CONSTANTS	
			9625	*		
1DEA		1DEB	9626	SFPDEV DS	CL(@CADDR)	SAVE SYS OUTPUT DEVICE STATUS
			9627	*		
		1DEB	9628	SFPDIC EQU	SFPDEV	CHAR-CON ELEMENT COUNTER
			9629	*		

SFPUTR - PROLOGUE - VM PUT ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 148
	1DEC	01		1DEC	9630	SFPONE	DC XL1'01'			
	1DED	0707		1DEE	9631	SFPMPT	DC AL2(\$\$PRNT)			INDR FOR OUTPUT TO PRINTER
	1DEF	2004		1DF0	9632	SFPCRT	DC AL2(\$\$PLYN)			INDR FOR OUTPUT TO CRT
	1DF1	0100		1DF2	9633	SFPVD2	DC AL2(V\$SFD2)			VADDR OF VM DIRECTORY 2

SFPUTR - PROLOGUE - VM PUT ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 149
1E00				9635		ORG	*,B@LVPG,0	PLACE MODULE AT PAGE BOUNDARY
				1E00 9636	SFPBS2	EQU	*	ESTABLISH
				1E00 9637		USING	SFPBS2,@BR	* BASE RESISTER
				9638	*			
				9639	*		THE FILE SPECIFIED IS A DISK FILE.	
				9640	*		@XR POINTS TO THE CURRENT D2 ENTRY.	
				9641	*		THE CONSTANT TO BE PUT IN THE FILE IS IN THE	
				9642	*		STACK IN PACKED FORMAT, IF NUMERIC.	
				9643	*			
1E00	AD	00	04 03	9644		CLC	@\$D2CS(@\$L2VB,@XR),@\$D2BS(@XR)	IS BUFFER FULL ?
1E04	F2	01	06	9645		JNE	SFP350	NO, BYPASS BUFFER WRITE
1E07	D0	87	E0	9646		B	SFP580(@BR)	GO WRITE BUFFER
1E0A	F2	87	0E	9647		J	SFP370	BYPASS BFR ACCESS
1E0D	2C	01	144A 05	9649	SFP350	MVC	I\$VADR,@\$D2CP(@\$L2CP,@XR)	COMBINE CURR POINTER & VM BASE
1E12	2E	00	1449 02	9650		ALC	I\$VADR-1,@\$D2VB(@\$L2VB,@XR)	* PAGE TO SET CURR BFR PAGE
1E17	C0	87	1349	9651		B	I\$MDFY	ACCESS & SET CURR BFR MODIFIED
1E1B	7C	00	F4	9652	SFP370	MVI	SFPWRK(@BR),@ZERO	INITLZ SPACE LEFT COUNTER
1E1E	6F	00	F4 05	9653		SLC	SFPWRK(@BR),@\$D2CB(@\$L2CB,@XR)	CALC SPACE LEFT IN BFR
1E22	6C	00	F6 01	9654		MVC	SFPSIO(@BR),@\$D2IO(@\$L2IO,@XR)	SAVE FILE PREC
1E26	74	02	7C	9655		ST	SFPXR1(@BR),@XR	SAVE POINTER TO D2 ENTRY
1E29	B8	02	01	9656		TBN	@\$D2IO(@XR),@\$M2EF	END OF FILE INDR ON ?
1E2C	F2	90	10	9657		JF	SFP385	NO, GO SET POINTER TO STACK
1E2F	35	02	144C	9658		L	I\$CADR,@XR	SET POINTER TO CURR BFR LOCO.
1E33	BC	1C	00	9659		MVI	@ZERO(@XR),@EOF	MOVE IN AN EOF CODE
1E36	75	02	7C	9660		L	SFPXR1(@BR),@XR	RELOAD D2 ENTRY POINTER
1E39	D0	87	E0	9661		B	SFP580(@BR)	GO WRITE BUFFER
1E3C	D0	87	A9	9662		B	SFP490(@BR)	GO TO RETURN
1E3F	35	02	0D4E	9664	SFP385	L	I\$STAK,@XR	SET POINTER TO STACK
1E43	B8	40	00	9665		TBN	I@STAT(@XR),B@DTYP	CHARACTER CONSTANT ?
1E46	F2	10	2A	9666		JT	SFP430	YES, SO SET LENGTH
1E49	78	20	F6	9667		TBN	SFPSIO(@BR),@\$M2FP	IS FILE PREC = LONG ?
1E4C	F2	10	10	9668		JT	SFP400	YES, GO CHECK RUN PREC.
1E4F	7C	05	F2	9669		MVI	SFPCNL(@BR),I@LPFS	SET CON LNG FOR SHORT PREC.
				9670	*		IF RUN PREC = SHORT, NEXT INSTR IS A JUMP TO RELOAD POINTER	
				9671	*		TO D2 ENTRY. IF RUN PREC = LONG, INSTR IS A NOP	
1E52	F2	87	21	9672		JC	SFP450,I@PRSW	JUMP IF RUN PREC = SHORT
1E55	AC	00	04 08	9673		MVC	I@PEXS(@XR),I@PEXL(@B1,@XR)	SET EXPONENT TO SHORT PREC.
1E59	BB	20	00	9674		SBF	I@STAT(@XR),B@PREC	SET CON INDR TO SHORT PREC.
1E5C	F2	87	17	9675		J	SFP450	GO RELOAD D2 ENTRY POINTER
1E5F	7C	09	F2	9676	SFP400	MVI	SFPCNL(@BR),I@LPFL	SET CON LNG FOR LONG PREC.
1E62	F2	80	11	9677		JC	SFP450,@UCB-I@PRSW+@NOP	IF RUN = LONG, GO RELOAD D2 PTR
1E65	BA	20	00	9678	SFP410	SBN	I@STAT(@XR),B@PREC	SET CON INDR TO LONG PREC
1E68	AC	00	08 04	9679		MVC	I@PEXL(@XR),I@PEXS(@B1,@XR)	SET EXPONENT FOR LONG PREC.
1E6C	AF	03	07 07	9680		SLC	I@PEXL-1(@XR),I@PEXL-1(SFPDLS,@XR)	SET EXTRA DIGITS TO 0
1E70	F2	87	03	9681		J	SFP450	GO RELOAD D2 ENTRY POINTER
1E73	7C	13	F2	9683	SFP430	MVI	SFPCNL(@BR),I@LCRV	SET CON LNG FOR CHAR CONSTANT
1E76	74	02	CA	9684	SFP450	ST	SFPSTK(@BR),@XR	INITLZ CON-MOVE STACK CADDR
1E79	C2	02	0000	9685	SFP460	LA	*-*,@XR	RESTORE CURR D2 ENTRY POINTER
				1E7C 9686	SFPXR1	EQU	SFP460+@OP1	CADDR OF CURRENT D2 ENTRY
1E7D	7D	00	F4	9687		CLI	SFPWRK(@BR),@ZERO	IS SPACE LEFT AT MAXIMUM ?
1E80	F2	81	2A	9688		JE	SFP500	YES, BYPASS REST OF TESTING
1E83	5C	00	F8 F4	9689		MVC	SFPprt(@BR),SFPWRK(1,@BR)	MOVE SPACE LEFT TO MOVE LENGTH
1E87	5F	00	F4 F2	9690		SLC	SFPWRK(@B1,@BR),SFPCNL(@BR)	DECR SPACE LEFT BY CON-LNG

SFPUTR - PROLOGUE - VM PUT ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 150
	1E8B	F2	84 1F		9691	JH	SFP500	GO MOVE CONSTANT IF SPACE
	1E8E	6C	01 F6 0B		9692	MVC	SFPWK2(@\$L2FS,@BR),@\$D2FS(@XR)	MOVE FILE SIZE TO WORKFLD
	1E92	6C	01 F4 09		9693	MVC	SFPWRK(@BR),@\$D2DD(@\$L2DD,@XR)	DECR BY
	1E96	5F	01 F6 F4		9694	SLC	SFPWK2(@BR),SFPWRK(@\$L2FS,@BR)	* SECTORS USED
	1E9A	6F	00 F6 04		9695	SLC	SFPWK2(@BR),@\$D2CS(@\$L2CS,@XR)	CALC SECTORS LEFT IN
	1E9E	5F	01 F6 FC		9696	SLC	SFPWK2(@\$L2FS,@BR),SFPC01(@BR)	* FILE
	1EA2	F2	84 0C		9697	JH	SFP510	MORE, GO SET UP PART-CON MOVE
	1EA5	3C	BA 0CBC		9698	SFP480 MVI	I\$ERRC,@E715	SET EOF ERROR CODE
	1EA9	C0	87 12D3		9699	SFP490 B	I\$RTRN	RETURN
	1EAD	5C	00 F8 F2		9701	SFP500 MVC	SFPprt(@BR),SFPCNL(1,@BR)	SET LENGTH OF CON-MOVE
	1EB1	4C	01 C8 144C		9702	SFP510 MVC	SFPBFR(@CADDR,@BR),I\$CADR	INITLZ BUFFER CADDR FOR MOVE
	1EB6	7C	FF C6		9703	MVI	SFPMVL(@BR),SFPMs1	CALC LENGTH
	1EB9	5E	00 C6 F8		9704	ALC	SFPMVL(@B1,@BR),SFPprt(@BR)	* OF MOVE
	1EBD	5E	00 CA C6		9705	ALC	SFPSTK(@B1,@BR),SFPMVL(@BR)	INCR STACK BUFFER CADDR'S
	1EC1	5E	00 C8 C6		9706	ALC	SFPBFR(@B1,@BR),SFPMVL(@BR)	* BY LENGTH OF MOVE
	1EC5	0C	00 0000 0000		9707	SFP550 MVC	*-*(@VQ),*-*	MOVE CONSTANT FROM STACK TO BFR
				1EC8	9708	SFPBFR EQU	SFP550+@OP1	CADDQ OF BUFFER LOCATION
				1EC6	9709	SFPMVL EQU	SFP550+@Q	LENGTH OF MOVE
				1ECA	9710	SFPSTK EQU	SFP550+@OP2	CADDR OF STACK LOCATION
	1ECB	9E	01 05 F8		9711	ALC	@\$D2CP(@\$L2CP,@XR),SFPprt(@BR)	INCR D2 ENTRY BFR POINTER
	1ECF	5F	00 F2 F8		9712	SLC	SFPCNL(@B1,@BR),SFPprt(@BR)	DECR CON-LNG BY LNG MOVED
	1ED3	D0	81 A9		9713	BZ	SFP490(@BR)	IF ALL MOVED, GO TO RETURN
	1ED6	5E	00 CA FC		9714	SFP560 ALC	SFPSTK(@B1,@BR),SFPC01(@BR)	INCR STACK CADDR FOR NXT MVE
	1EDA	D0	87 E0		9715	B	SFP580(@BR)	
	1EDD	D0	87 AD		9716	B	SFP500(@BR)	
					9717	*		
					9718	*	VM BUFFER MUST BE WRITTEN TO THE SAVED FILE	
					9719	*		
	1EE0	74	08 F1		9720	SFP580 ST	SFP590+@OP1(@BR),@ARR	SAVE RETURN
	1EE3	C0	87 12B1		9721	B	I\$CALL	EXECUTE PART 3
	1EE7	1F00		1EE8	9722	DC	AL(@VADDR)(V\$XSPT+2*B@BLSZ)	* OF SFPUTR
	1EE9	1C	01 144C FA		9723	MVC	I\$CADR,SFPSCA(@CADDR,@BR)	
	1EEE	C0	87 0000		9724	SFP590 B	*-*	RETURN TO CALLING LOCATION
					9725	*		
					9726	*	PART 2 - WORK AREAS AND CONSTANTS	
					9727	*		
	1EF2			1EF2	9728	SFPCNL DS	CL1	SAVE AREA FOR CONSTANT LENGTH
	1EF2				9729	ORG	SFPCNL	* INITIALIZE TO
	1EF2	00		1EF2	9730	DC	XL1'0'	* ZERO
	1EF3	00		1EF3	9731	DC	XL1'00'	INITLZ TO '0' TO USE WRK AS 2
	1EF4			1EF4	9732	SFPWRK DS	CL1	WORKAREA TO CALC SPACE LEFT
	1EF5			1EF6	9733	SFPWK2 DS	CL2	WORKAREA TO CHECK FOR EOF
	1EF7			1EF8	9734	SFPprt DS	CL2	WORKAREA TO FOR MOVE LENGTH
	1EF7				9735	ORG	SFPprt-1	* INITLZ TO ZERO, 1ST BYTE WILL
	1EF7	0000		1EF8	9736	DC	XL2'0'	* ALWAYS BE 0.
	1EF9			1EFA	9737	SFPSCA DS	CL2	SAVE AREA FOR VM BUFFER CADDR
	1EFB	0001		1EFC	9738	SFPC01 DC	IL2'1'	ONE CONSTANT
					9739	*		
					9740	*	PART 2 - EQUATES	
					9741	*		
				1EF6	9742	SFPsIO EQU	SFPWK2	SAVE AREA FOR FILE PRECISION
				0004	9743	SFPDLS EQU	I@LPFL-I@LPFS	LNG LONG PREC, EXTRA SIGNIFICNCE
				00FF	9744	SFPMS1 EQU	X'FF'	MINUS 1

SFPUTR - PROLOGUE - VM PUT ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 151
	1F00				9746	ORG	*,B@LVP,0	PLACE MODULE_AT PAGE BOUNDARY
				1F00	9747	SFPBS3 EQU	*	* FOR THIS PAGE
				1F00	9748	USING	SFPBS3,@BR	ESTABLISH BASE REGISTER USAGE
	1F00	1C 01	144A 8B		9749	MVC	I\$VADR,SFPRT2(@CADDR,@BR)	REFERENCE PAGE 2 OF
	1F05	C0 87	1358		9750	B	I\$CVAD	* THE SFPUTR ROUTINE
	1F09	4C 00	76 144B		9751	MVC	SFP680+@OP1-1(1,@BR),I\$CADDR-1	SET UP RTRN CADDR FOR PG CA
	1F0E	AD 00	04 03		9752	CLC	@\$D2CS(@\$L2VB,@XR),@\$D2BS(@XR)	IS VM BUFFER FULL ?
	1F12	F2 81	09		9753	JE	SFP610	YES, GO SET UP FOR SFLOAD
	1F15	B8 02	01		9754	TBN	@\$D2IO(@XR),@\$M2EF	EOF INDR SET ?
	1F18	F2 10	03		9755	JT	SFP610	YES, GO SET UP FOR SFLOAD
					9756	*		
					9757	*		
					9758	*		CURRENT BUFFER PAGE IN CORE IS FILLED, BUT VM BUFFER
					9759	*		NOT FULL. GET NEXT PAGE OF BUFFER INTO CORE.
	1F1B	F2 87	48		9760	J	SFP675	
					9761	*		
					9762	*		VM BUFFER MUST BE PUT TO SAVED FILE. SET UP & EXEC SFLOAD
					9763	*		
	1F1E	4C 00	2C 144B		9764	SFP610 MVC	SFP625+@OP2-1(1,@BR),I\$CADDR-1	SET UP CADDR FOR CON-LNG
	1F23	4C 00	32 144B		9765	MVC	SFP630+@OP2-1(1,@BR),I\$CADDR-1	SET UP CADDR FOF CON-CADDR
	1F28	0C 00	0D59 1EF2		9766	SFP625 MVC	I\$WRK1,SFPCNL(1)	MOVE CON-LNG FOR SFLOAD
	1F2E	0C 01	0D5B 1ECA		9767	SFP630 MVC	I\$WRK2,SFPSTK(@CADDR)	MOVE CADDR OF CON FOR SFLOAD
	1F34	3C 02	0D58		9768	MVI	I\$WRK1-1,@DPUT	SET OUTPUT INDR FOR SFLOAD
	1F38	74 01	43		9769	ST	SFP635+SFP5(@BR),@BR	SET UP CADDR OF DPL TO
	1F3B	7C 7E	43		9770	MVI	SFP635+SFP5(@BR),SFPD1D	* WRITE OUT INTERPRETER
					9771	*SFP635 DISK	@ZERO	WRITE IT OUT
	1F3E	C0 87	0025		9772	SFP635 B	\$DISKN	PERFORM PHYSICAL DISK OP
	1F42	0000		1F43	9773	DC	AL2(@ZERO)	DPL ADDRESS
					9774	***	END OF EXPANSION ***	
	1F44	74 01	55		9776	ST	SFP640+SFP5(@BR),@BR	SET UP CADDR OF DPL TO
	1F47	7C 84	55		9777	MVI	SFP640+SFP5(@BR),SFPD2D	* READ & EXECUTE SFLOAD
	1F4A	74 01	5B		9778	ST	SFP650+@OP1(@BR),@BR	SAVE BASE REGISTER
	1F4D	74 02	5F		9779	ST	SFP655+@OP1(@BR),@XR	SAVE POINTER TO D2 ENTRY
					9780	*SFP640 BLOAD	@ZERO	GO EXECUTE SFLOAD
	1F50	C0 87	0522		9781	SFP640 B	\$BLOAD	LOAD AND EXECUTE WORK AREA PGM
	1F54	0000		1F55	9782	DC	AL2(@ZERO)	DPL ADDRESS
					9783	***	END OF EXPANSION ***	
					9785	*		
					9786	*		RETURN FROM SFLOAD
					9787	*		
	1F56	0444		1F57	9788	DC	AL(@CADDR)(\$DPLSV-5)	CADDR OF INTERPRETER READ DPL
	1F58	C2 01	0000		9789	SFP650 LA	*-*,@BR	RESTORE BASE
	1F5C	C2 02	0000		9790	SFP655 LA	*-*,@XR	RESTORE D2 POINTER
					9791	*	DISK \$WAITF	WAIT FOR INTERPRETER
	1F60	C0 87	0025		9792	B	\$DISKN	PERFORM PHYSICAL DISP OP
	1F64	057F		1F65	9793	DC	AL2(\$WAITF)	DPL ADDRESS
					9794	***	END OF EXPANSION ***	
	1F66	2C 01	144A 05		9796	SFP675 MVC	I\$VADR,\$D2CP(@\$L2CP,@XR)	SET UP VADDR OF NEXT PAGE
	1F6B	2E 00	1449 02		9797	ALC	I\$VADR-1,\$D2VB(@\$L2VB,@XR)	* OF VM BUFFER
	1F70	C0 87	1349		9798	B	I\$MDFY	ACCESS & SET FOR MODIFICATION
					9799	*		
					9800	*		THE NEXT INST HAS BEEN MODIFIED TO UP THE CADDR OF THE
					9801	*		BUFFER INTO A SAVE AREA IN PART 2 OF SFPUTR

SFPUTR - PROLOGUE - VM PUT ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 152
					9802	*				
	1F74	0C	01	1EFA	144C	9803	SFP680 MVC SFPSCA,I\$CADR(@CADDR)			
	1F7A	C0	87	12D3		9804	B I\$RTRN			RETURN TO SFPUTR - PART 2
					9805	*				
					9806	*	PART 3 - DPL'S			
					9807	*				
					9808	*SFPD1 DPL	FUNC=@DPUT,DADDR=#@VSFI,CNT=#@VSL,CADDR=\$\$INS			
				1F7E	9809	SFPD1 EQU	*			DISK PARAMETER LIST
	1F7E	02		1F7E	9810		DC AL1(@DPUT)			REQUESTED FUNCTION
	1F7F	09A1		1F80	9811		DC AL2(#@VSFI)			DISK ADDRESS
	1F81	0F		1F81	9812		DC AL1(#@VSL)			SECTOR COUNT
	1F82	0600		1F83	9813		DC AL2(\$\$INS)			BUFFER ADDRESS
					9814	***	END OF EXPANSION ***			
				007E	9816	SFPD1D EQU	SFPD1-SFPBS3			DISP TO INTRP WRITE DPL
					9817	*				
					9818	*SFPD2 DPL	FUNC=@DGET,DADDR=#@SFLO,CNT=#@SFL,CADDR=\$\$SFL			
				1F84	9819	SFPD2 EQU	*			DISK PARAMRMI RST
	1F84	01		1F84	9820		DC AL1(@DGET)			REQUESTED FUNCTION
	1F85	0499		1F86	9821		DC AL2(#@SFLO)			DISK ADDRESS
	1F87	05		1F87	9822		DC AL1(#@SFL)			SECTOR COUNT
	1F88	0F00		1F89	9823		DC AL2(\$\$SFL)			BUFFER ADDRESS
					9824	***	END OF EXPANSION ***			
				0084	9826	SFPD2D EQU	SFPD2-SFPBS3			DISP TO SFLOAD READ DPL
					9827	*				
					9828	*	PART 3 - CONSTANTS, SAVE AREAS & EQUATES			
					9829	*				
	1F8A	1E00		1F8B	9830	SFPRT2 DC	AL(@VADDR)(V\$XSPT+B@LVPG)			VADDR OF SFPUTR PART 2
					9831	*				
				0005	9832	SFP5 EQU	5			DISP TO DPL-ADDR
				0F00	9833	SFPSAO EQU	\$KLD1+X'0900'			CORE LOAD ADDR OF #SFLOA

SFPUTR - PROLOGUE - VM PUT ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 153

2000				9835	ORG	*,B@LVPG,0	PLACE MODULE AT PAGE BOUNDARY
			2000	9836	SFPBS4 EQU	*	* REGISTER USAGE
			2000	9837	USING	SFPBS4,@BR	ESTABLISH BASE
				9838	*		
				9839	*	ENTRY TO PART 2 OR PT TO CARD FILE ROUTINE	
				9840	*		
2000	74	02	AD	9841	ST	SFP830+@OP1(,@BR),@XR	SAVE D2 POINTER
2003	6C	01	C0 05	9842	MVC	SFPVCA(,@BR),@\$D2CP(@\$L2CP,@XR)	SAVE CURRENT BFR POINTER
2007	6C	00	94 02	9843	MVC	SFPCBP(,@BR),@\$D2VB(@\$L2VB,@XR)	SAVE BFR BASE PAGE NUM.
				9844	*		
200B	B8	02	01	9845	TBN	@\$D2IO(,@XR),@\$M2EF	END OF FILE INDR ON ?
200E	F2	10	7F	9846	JT	SFP800	YES, GO CLEAR & FLUSH BUFFER
				9847	*		
2011	35	02	0D4E	9848	L	I\$STAK,@XR	POINT @XR AT STACKED CONSTANT
2015	74	02	E1	9849	ST	SFPSTC(,@BR),@XR	SAVE STACK POINTER
2018	4C	00	DE 0D59	9850	MVC	SFPCFL(@\$L2CB,@BR),I\$WRK1	SET CHAR-CON LENGTH-1
201D	BD	7D	00	9851	CLI	I@STAT(,@XR),B@SQUO	IS IT A CHAR-CON ?
2020	F2	81	55	9852	JE	SFP785	YES, GO CHECK FOR FIT
				9853	*		
				9854	*	DATA ITEM IS NUMERIC - CONVERT IT TO OUTPUT FORMAT	
				9855	*		
2023	C0	87	0A27	9856	B	I\$CPUF	UNPACK THE NUMBER
2027	AC	06	08 07	9857	MVC	I@PREC+1(,@XR),I@PREC(I@PREC,@XR)	SHIFT MANTISSA RIGHT 1
202B	BC	4B	01	9858	MVI	I@STAT+1(,@XR),B@DPNT	INSERT DECIMAL POINT
				9859	*		
				9860	*	CONVERT EXPONENT TO OUTPUT FORMAT	
				9861	*		
202E	9C	03	0C FA	9862	MVC	I@PREC+5(,@XR),SFPEXI(SFPLEX,@BR)	MOVE IN EXPONENT IMAGE
2032	9F	00	00 FB	9863	SLC	I@STAT(,@XR),SFPEZR(1,@BR)	DETERMINE EXP. MAGNITUDE
2036	F2	81	2D	9864	JZ	SFP750	BYPASS CONVERSION TO DEC IF 0
2039	F2	84	0E	9865	JH	SFP720	BYPASS RE-COMPLEMENT IF POSITIVE
				9866	*		
				9867	*	NEGATIVE EXPONENT - MODIFY SIGN AND RECOMPUTE BINARY EXPONENT	
				9868	*		
203C	7C	00	FD	9869	MVI	SFPNGE(,@BR),@ZERO	DETERMINE BINARY MAGNITUDE OF
203F	6F	00	FD 00	9870	SLC	SFPNGE(1,@BR),I@STAT(,@XR)	* NEGATIVE EXPONENT
2043	9C	00	00 FD	9871	MVC	I@STAT(,@XR),SFPNGE(1,@BR)	PUT ABS VALUE IN STACK EXP POS
2047	BC	60	0A	9872	MVI	I@PREC+3(,@XR),B@MINS	MAKE EXPONENT SIGN NEGATIVE
				9873	*		
				9874	*	CONVERT BINARY EXPONENT MAGNITUDE TO ZONED DECIMAL	
				9875	*		
204A	54	10	FD F6	9876	SFP720 ZAZ	SFPDAC(SFPLXM,@BR),SFPZD1(1,@BR)	SET ACCUMULATOR - 1
204E	7C	01	52	9877	MVI	SFP725+@Q(,@BR),@B1	SET BINARY MASK FOR 2**6 BIT
2051	B8	01	00	9878	SFP725 TBN	I@STAT(,@XR),@VQ	TEST BINARY EXP MAGNITUDE BIT
2054	F2	90	04	9879	JF	SFP730	BYPASS DEC. EXP. INCR IF ZERO
2057	96	01	0C FD	9880	AZ	I@PREC+5(SFPLXM,@XR),SFPDAC(SFPLXM,@BR)	INCR DECIMAL EXP
205B	5E	00	52 52	9881	SFP730 ALC	SFP725+@Q(,@BR),SFP725+@Q(1,@BR)	SHIFT BINARY MASK LEFT
205F	56	01	FD FD	9882	AZ	SFPDAC(SFPLXM,@BR),SFPDAC(SFPLXM,@BR)	DOUBLE DEC ACCUM
2063	D0	08	51	9883	BNOZ	SFP725(,@BR)	REPEAT LOOP UNTIL ACCUM > 64
				9884	*		
				9885	*	DETERMINE AND SET SIGN OF MANTISSA	
				9886	*		
2066	BC	40	00	9887	SFP750 MVI	I@STAT(,@XR),B@BLNK	INITLZ SIGN TO POSITIVE
2069	B8	F0	08	9888	TBN	I@PREC+1(,@XR),B@ZPOS	IS MANTISSA POSITIVE ?
206C	F2	10	06	9889	JT	SFP760	YES, BYPASS NEGATIVE HANDLING
206F	BC	60	00	9890	MVI	I@STAT(,@XR),B@MINS	SET SIGN TO NEGATIVE

SFPUTR - PROLOGUE - VM PUT ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 154
	2072	BA	F0 08		9891	SBN	I@PREC+1(,@XR),B@ZPOS			CLEAR MANTISSA SIGN INDR.
					9892	*				
					9893	*	SET LENGTH OF NUMERIC VALUE IN STACK			
					9894	*				
	2075	7C	0C DE		9895	SFP760 MVI	SFPCFL(,@BR),I@PREC+SFPENC			SET EXTERNAL LENGTH OF NUM-CON
					9896	*				
					9897	*	DETERMNE IF THIS DATA ITEM WILL FIT ON CURRENT CARD			
					9898	*				
	2078	5C	00 FD C0		9899	SFP785 MVC	SFPCPW(,@BR),SFPCPT(1,@BR)			MOVE CURR PT TO WORK AREA
	207C	5E	00 FD DE		9900	ALC	SFPCPW(@\$L2CB,@BR),SFPCFL(,@BR)			CALC NEXT BFR SPACE
	2080	38	80 03DD		9901	TBN	\$CONFIG,\$BIGCD			IS IBM 129 CONFIGURED ? 1-4
	2084	F2	90 03		9902	JF	SFP790			JUMP IF NOT 1-4
	2087	7C	4E 8B		9903	MVI	SFP790+1(,@BR),@BCRD-2			CHANGE LENGTH TO BIG CARD 1-4
	208A	7D	00 FD		9904	SFP790 CLI	SFPCPW(,@BR),*-*			WILL ADJ LNG FIT ON CARD ? 1-4
	208B				9905	ORG	SFP790+@Q			POINT TO INITLZ LENGTH 1-4
	208B	5E		208B	9906	DC	AL1(@CARDL-2)			INITIALIZE FOR 96 COL CARD 1-4
	208D				9907	ORG	SFP790+@INST3			RESTORE INSTRUCTION ADDR REG 1-4
	208D	F2	04 27		9908	JNH	SFP850			YES, GO PUT IT IN BUFFER
					9909	*				
					9910	*	CARD SUFFER CAN NOT CONTAIN THE ENTIRE NEXT CONSTANT OR			
					9911	*	END OF FILE HAS OCCURRED.			
					9912	*				
					9913	*	CLEAR THE UNUSED PORTION OF THE BUFFER			
					9914	*				
	2090	C0	87 1330		9915	SFP800 B	I\$LDXR			POINT @XR TO 1ST BYTE
	2094			2095	9916	SFPCBV DS	CL(@VADDR)			VADDR OF CARD BUFFER PAGE
				2094	9917	SFPCBP EQU	SFPCBV-1			PAGE ADDRESS
	2095				9918	ORG	SFPCBV			INITLZ THE DISPLACEMENT TO
	2095	00		2095	9919	DC	XL(@\$L2CB)'00'			* THE FIRST BYTE OF THE BUFFER
	2096	76	02 C0		9920	A	SFPCPT(,@BR),@XR			INCR @XR TO FIRST UNUSED BYTE
	2099	BC	40 60		9921	MVI	@CARDL(,@XR),B@BLNK			BLANK OUT THE UNUSED PORTION
	209C	AC	5F 5F 60		9922	MVC	@CARDL-1(,@XR),@CARDL(@CARDL,@XR)			* OF THE CARD BUFFER
					9923	*				
					9924	*	CALL THE I/O ROUTINE TO PUNCH THE CARD			
					9925	*				
	20A0	35	02 144C		9926	L	I\$CADR,@XR			POINT @XR TO BUFFER
	20A4	C0	87 12B1		9927	B	I\$CALL			EXECUTE THE CARD
	20A8	2A96		20A9	9928	DC	AL(@VADDR)(V\$SCDO)			* PUT ROUTINE
					9929	*				
					9930	*	POINT @XR BACK TO D2 ENTRY AND CHECK FOR END OF FILE			
					9931	*				
	20AA	C2	02 0000		9932	SFP830 LA	*-*,@XR			POINT @XR TO D2 ENTRY
	20AE	B8	02 01		9933	TBN	@\$D2IO(,@XR),@\$M2EF			END OF FILE INDR ON ?
	20B1	D0	10 F1		9934	BT	SFP950(,@BR)			YES, GO TO RETURN
	20B4	7C	00 C0		9935	MVI	SFPCPT(,@BR),@ZERO			MOVE A ZERO TO CURR BYTE PT
					9936	*				
					9937	*	PUT DATA ITEM IN CARD BUFFER			
					9938	*				
	20B7	5E	00 BF 94		9939	SFP850 ALC	SFPVCA-1(,@BR),SFPCBP(@\$L2VB,@BR)			INCR BFR DISP BY VM BASE
	20BB	C0	87 1330		9940	B	I\$LDXR			POINT @XR AT
	20BF			20C0	9941	SFPVCA DS	CL(@VADDR)			* 1ST UNUSED BUFFER POSITION
				20C0	9942	SFPCPT EQU	SFPVCA			DISP. TO 1ST UNUSED SPACE
	20C1	C0	87 1349		9943	B	I\$MDFY			SET BUFFER MODIFIED 1-4
	20C5	7D	00 C0		9944	CLI	SFPCPT(,@BR),@ZERO			1ST DATA ITEM ON CARD
	20C8	F2	81 0A		9945	JE	SFP865			YES, BYPASS SEPERATOR PLACEMENT
					9946	*				

SFPUTR - PROLOGUE - VM PUT ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 155
					9947	*	PLACE A SEPERATOR BEFORE THE CURRENT DATA ITEM	
					9948	*		
	20CB	BC	6B 00		9949		MVI @ZERO(,@XR),B@CMMA PUT SEPORATOR IN CARD BUFFER	
	20CE	E2	02 01		9950		LA @B1(,@XR),@XR INCR @XR TO NEXT UNUSED BYTE	
	20D1	5E	00 C0 F5		9951		ALC SFPCPT(,@BR),SFPX01(1,@BR) INCR CARD BUFR PT BY 1	
					9952	*		
					9953	*	PLACE THE DATA ITEM IN THE CARD BUFFER	
					9954	*		
	20D5	5E	00 E1 DE		9955	SFP865	ALC SFPSTC(1,@BR),SFPCFL(,@BR) * OF DATA ITEM	
	20D9	5C	00 DF DE		9956		MVC SFPDCA(,@BR),SFPCFL(1,@BR) SET BUFFER DISP TO CON LEFTEND	
	20DD	8C	00 00 0000		9957	SFP875	MVC 0(@VQ,@XR),*-* MOVE DATA ITEM TO BUFFER	
				20DF	9958	SFPDCA	EQU SFP875+@D1 DISP FROM @XR TO LH BFR ADDR	
				20DE	9959	SFPCFL	EQU SFP875+@Q EXTERNAL FORMAT LENGTH MINUS 1	
				20E1	9960	SFPSTC	EQU SFP875+@DOP2 LEFT HAND STACK CADDR OF ITEM	
					9961	*		
					9962	*	UPDATE D2 CURRENT ENTRY BUFFER POINTER AND RETURN	
					9963	*		
	20E2	75	02 AD		9964		L SFP830+@OP1(,@BR),@XR RESTORE D22 ENTRY POINTER	
	20E5	9C	00 05 C0		9965		MVC @\$D2CB(@\$L2CB,@XR),SFPCPT(,@BR) SET IN STARTING DISP.	
	20E9	9E	00 05 DE		9966		ALC @\$D2CB(@\$L2CB,@XR),SFPCFL(,@BR) INCR TO ENDING DISP.	
	20ED	9E	00 05 F5		9967		ALC @\$D2CB(@\$L2CB,@XR),SFPX01(,@BR) INCR TO NEXT UNUSED SPACE	
	20F1	C0	87 12D3		9968	SFP950	B I\$RTRN RETURN	
					9969	*		
					9970	*	PART 4 - EQUATES, WORKAREAS AND CONSTANTS	
					9971	*		
				0004	9972	SFPLEX	EQU 4 LENGTH OF EXPONENT IMAGE	
				0002	9973	SFPLXM	EQU 2 LENGTH OF ZONED DEC. EXPONENT	
				0005	9974	SFPENC	EQU SFPLEX+2-1 LENGTH-1 OF EXTRA NON-PREC CHAR	
					9975	*		
	20F5	01		20F5	9976	SFPX01	DC XL1'1' BINARY 1	
	20F6	F1		20F6	9977	SFPZD1	DC DL1'1' ZONED DECIMAL 1	
	20F7	C54EF0F0		20FA	9978	SFPEXI	DC CL(SFPLEX)'E+00' EXPONENT IMAGE, INITLZ ZERO	
	20FB	80		20FB	9979	SFPEZR	DC AL1(B@NXZR) ZERO NORMALIZED EXPONENT	
					9980	*		
	20FC			20FD	9981	SFPDAC	DS CL(@CADDR) DECIMAL ACCUMULATOR WORK AREA	
				20FD	9982	SFPNGE	EQU SFPDAC NEGATIVE EXPONENT WORK AREA	
				20FD	9983	SFPCPW	EQU SFPNGE BUFFER OVERFLOW CALC WORK AREA	
					9984	*		
					9985	*	END OF SFPUTR ROUTINE	
					9986	*		

SFGETR - PROLOGUE - VM GET ROUTINE

ERR	LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 157
			9989		*****			
			9990	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
			9991	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
			9992	*				*
			9993		*****			*
			9994	*	STATUS			*
			9995	*	VERSION 1 MODIFICATION 0			*
			9996	*				*
			9997	*	FUNCTION			*
			9998	*	* SFGETR IS CALLED TO INPUT THE NEXT SEQUENTIAL DATA ELEMENT FROM			*
			9999	*	AN EXTERNAL DATA FILE. THIS DATA FILE MAY BE ON DISK OR CARD.			*
				*	* THE NEXT SEQUENTIAL DATA ELEMENT, ARITHMETIC OR CHARACTER, IS			*
			1	*	ACCESSED AND PLACED IN THE RUN-TIME STACK AREA.			*
			2	*	* IF INPUT IS FROM THE CARD READER, THE DATA ELEMENT MUST BE CON-			*
			3	*	VERTED TO INTERNAL NOTATION BEFORE IT IS PASSED			*
			4	*	* WHEN ALL DATA ELEMENTS IN THE BUFFERS ALLOCATED TO THE FILE ARE			*
			5	*	DEPLETED, A CALL IS MADE TO THE APPROPRRATE ROUTINE TO REFILL			*
			6	*	THE BUFFERS IN VIRTUAL MEMORY.			*
			7	*				*
			8	*	ENTRY POINTS			*
			9	*	* THE ENTRY IS SFGETR. THE FORMAT OF THE CALLING SEQUENCE IS AS			*
			10	*	FOLLOWS:			*
			11	*	B I\$CALL			*
			12	*	DC AL2(V\$XSGT)			*
			13	*				*
			14	*	INPUT:			*
			15	*	* THE DISPLACEMENT TO THE CURRENT FILE ENTRY IS IN THE D2 HEADER.			*
			16	*	* THE ADDRESS OR THE LOCATION TO PLACE THE NEXT CONSTANT FROM THE			*
			17	*	FILE IS IN I\$STAK.			*
			18	*				*
			19	*	OUTPUT			*
			20	*	* THE CONSTANTS ARE MOVED INTO THE STACK REFERENCED IN THE ADDR			*
			21	*	AT I\$STAK			*
			22	*				*
			23	*	EXTERNAL REFERENCES			*
			24	*	I\$MDFY - PAGING ENTRY TO SET BASE PAGE WRITE BACH INDICATOR			*
			25	*	I\$LOCK - PAGING ENTRY TO LOCK PAGE IN CORE			*
			26	*	I\$CVAD - PAGING ENTRY TO ACCESS PAGE			*
			27	*	I\$CALL - PAGING ENTRY TO CALL ANOTHER ROUTINE			*
			28	*	I\$UNLK - PAGING ENTRY TO UNLOCK PAGE			*
			29	*	I\$RTRN - PAGING ENTRY TO RETURN TO USER			*
			30	*	DKDISK - DISK IOCR			*
			31	*	SFLOAD - BUFFER TRANSFER ROUTINE			*
			32	*	DFRDIN - CARD READ ROUTINE			*
			33	*	I\$STAK - LOCATION OF ADDRESS OF STACK TO PLACE CONSTANT			*
			34	*	I\$ERRC - ERROR CODE SAVE AREA			*
			35	*				*
			36	*	EXITS, NORMAL			*
			37	*	THE CONSTANTS ARE PLACED IN THE RUN-TIME STACK, AND CONTROL			*
			38	*	RETURNED VIA I\$RTRN.			*
			39	*				*
			40	*	EXITS, ERROR			*
			41	*	IN THE EVENT OF AN ERROR, AN ERROR CODE IS PLACED IN \$\$ERRC, AND			*
			42	*	CONTROL RETURNED TO THE USER VIA I\$RTRN.			*
			43	*				*
			44	*	TABLES/WORK AREAS			*

SFGETR - PROLOGUE - VM GET ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 158
		45	*	D2 - VM DIRECTORY 2 - CURRENT USAGE I/O INFORMATION			*
		46	*				*
		47	*	*ATTRIBUTES			*
		48	*	REUSABLE, RELOCATABLE			*
		49	*				*
		50	*	*CHARACTER CODE DEPENDENCY			*
		51	*	THE OPERATION OF THIS MODULE DEPENDS UPON AN INTERNAL REPRESENTA-			*
		52	*	TION OF THE EXTERNAL CHARACTER SET WHICH IS EQUIVALENT TO THE ONE			*
		53	*	USED AT ASSEMBLY TIME.			*
		54	*				*
		55	*	*NOTES			*
		56	*	ERROR PROCEDURES			*
		57	*	AN ERROR CODE IS PLACED IN I\$ERRC, AND CONTROL RETURNED TO THE			*
		58	*	USER			*
		59	*				*
		60	*	RESISTER USAGE			*
		61	*	REGISTER 1 (@BR) IS USED AS THE BASE REGISTER			*
		62	*	REGISTER 2 (@XR) IS USED TO REFERENCE THE CURRENT FILE ENTRY			*
		63	*	AND D2, AND TO REFERENCE THE RUN-TIME STACK.			*
		64	*				*
		65	*	SAVED/RESTORED AREAS			*
		66	*	NONE.			*
		67	*				*
		68	*	MODIFICATION CONSIDERATIONS			*
		69	*	NONE.			*
		70	*				*
		71	*	REQUIRED NODULES			*
		72	*	@SYSEQ - SYSTEM EQUATES			*
		73	*	@FXDEQ - FIXED ADDRESSES FOR SYSTEM NUCLEUS			*
		74	*	@CANEQ - SYSTEM LOCATION EQUATES			*
		75	*	@WKAEQ - SYSTEM WORKAREA DADDR EQUATES			*
		76	*	@CYOEQ - CYLINDER ZERO EQUATES			*
		77	*	@VMDEQ - VM DIRECTORY EQUATES			*
		78	*	@ERMEQ - ERROR MESSAGE EQUATES			*
		79	*	@SPFEQ - SYSTEM PROGRAM FILE EQUATES			*
		80	*	\$V\$EQU - FIXED VIRTUAL ADDRESSES			*
		81	*	\$B@EQU - BASIC COMPILER EQUATES			*
		82	*	\$I@EQU - BASIC INTERPRETER EQUATES			*
		83	*	\$I@SEQ/\$I@LEQ - STANDARD/LONG PRESISION EXECUTION EQUATES			*
		84	*				*
		85	*	OTHER			*
		86	*	NONE			*
		87	*				*
		88	*	*****			*

SFGETR - PROLOGUE - VM GET ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 159

2100			90	ORG	*,B@LVPG,0	PLACE AT A PAGE BOUNDARY
		2100	91	SFGBS1 EQU	*	ESTABLISH BASE REGISTER USE
		2100	92	USING	SFGBS1,@BR	
		2100	93	SFGETR EQU	*	ENTRY POINT
2100 1C 01 144A FA			94	MVC	I\$VADR,SFGVD2(@VADDR,@BR)	ACCESS VM DIRECTORY 2 (D2) AND
2105 C0 87 1349			95	B	I\$MDFY	* SET FOR WRITE BACK
2109 C0 87 1354			96	B	I\$LOCK	LOCK IT IN CORE ALSO
210D 35 02 144C			97	L	I\$CADR,@XR	POINT XR TO CURRENT FILE ENTRY
2111 B6 02 01			98	A	@\$D2CF(,@XR),@XR	*
2114 B8 08 01			99	TBN	@\$D2IO(,@XR),@\$M2CI	CURRENT USAGE = INPUT ?
2117 F2 10 16			100	JT	SFG200	YES, GO CHECK DEVICE CODE
211A B8 04 01			101	TBN	@\$D2IO(,@XR),@\$M2CO	CURRENT USAGE = OUTPUT ?
211D F2 10 06			102	JT	SFG120	YES, GO SET USAGE ERR
2120 B8 80 01			103	TBN	@\$D2IO(,@XR),@\$M2FI	FILE USAGE - INPUT ?
2123 F2 10 07			104	JT	SFG150	YES, GO SET CURRENT USAGE
2126 3C B7 0CBC			105	SFG120 MVI	I\$ERRC,@E712	SET INVALID FILE USAGE ERR CODE
212A F2 87 BF			106	J	SFG295	RETURN
212D BA 08 01			107	SFG150 SBN	@\$D2IO(,@XR),@\$M2CI	SET CURRENT USAGE TO INPUT
			108	*		
			109	*	FILE MAY CURRENTLY BE USED FOR INPUT. CHECK INPUT DEVICE.	
			110	*		
2130 B8 20 00			111	SFG200 TBN	@\$D2DC(,@XR),@\$MBCD	CARD FILE ?
2133 F2 90 B0			112	JF	SFG290	NO, GO ACCESS DISK ROUTINE
			113	*		
			114	*	CARD FILE - ACCESS BUFFER AND LOCK IT IN CORE	
			115	*		
2136 BD FF 05			116	CLI	@\$D2CB(,@XR),SFGNFM	FIRST ACCESS OF THIS FILE ?
2139 F2 81 06			117	JE	SFG205	NO, BYPASS INITIALIZATION
213C 7C 80 5B			118	MVI	SFG210+@Q(,@BR),@NOP	FORCE AN INITIAL CARD RFAD
213F BC FF 05			119	MVI	@\$D2CB(,@XR),SFGNFM	SET OFF FIRST ACCESS INDICATOR
2142 6C 00 4A 02			120	SFG205 MVC	SFGBVA-1(,@BR),@\$D2VB(@\$L2VB,@XR)	SET UP VADDR OF BUFFER
2146 C0 87 1330			121	B	I\$LDXR	ACCESS BFR & POINT @XR TO IT
214A		214B	122	SFGBVA DS	CL(@VADDR)	VADDR OF BUFFER
214B			123	ORG	SFGBVA	* INITIALIZE BYTE DISPLACEMENT
214B 00		214B	124	DC	XL1'0'	* TO ZERO
214C 74 02 FC			125	ST	SFGCBA(,@BR),@XR	SAVE BUFFER CORE ADDRESS
214F C0 87 1349			126	B	I\$MDFY	SET FOR WRITE BACK 1-3
2153 C0 87 1354			127	B	I\$LOCK	* IN CORE
2157 BD 60 FF			128	CLI	SFGCBP(,@XR),@CARDL	MORE VALUES ON CARD ?
215A F2 01 70			129	SFG210 JC	SFG280,@BNE	YES, BYPASS CARD READ-FORCE 1ST
215D 7C 01 5B			130	MVI	SFG210+@Q(,@BR),@BNE	RESET FORCE 1ST READ SWITCH
			131	*		
			132	*	A CARD READ IS REQUIRED	
			133	*		
2160 C0 87 12B1			134	SFG215 B	I\$CALL	GO READ A
2164 2A00		2165	135	DC	AL(@VADDR)(V\$SCDI)	* CARD
			136	*		
2166 BC 1E 60			137	MVI	@CARDL(,@XR),@EOS	PUT AN EOS FOLLOWING CARD
2169 7C 40 74			138	MVI	SFG225+@D1(,@BR),SFGICR	INITLX FOR BLANKS VALID FIRST
216C 7C 87 C5			139	MVI	SFG270+@Q(,@BR),@UCB	INITLX FOR A BLANK CARD
			140	*		
			141	*	FORCE A COMMA DELIMITER BETWEEN DATA ITEMS	
			142	*		
216F BD 40 00			143	SFG220 CLI	@ZERO(,@XR),@BLANK	THIS CHARACTER A BLANK ?
2172 F2 81 40			144	SFG225 JE	SFG255	YES, GO INCR IF TRANSPARENT NOW
			145	*		

SFGETR - PROLOGUE - VM GET ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 160
2175	BD	6B	00	146	SFG227	CLI	@ZERO(,@XR),@COMMA			THIS CHARACTER A COMMA ?
2178	F2	81	40	147		JE	SFG260			YES, GO SET TRANSPARENT BLANK SW
				148	*					
217B	BD	1E	00	149		CLI	@ZERO(,@XR),@EOS			THIS CHARACTER A EOS ?
217E	F2	81	40	150		JE	SFG265			YES, GO RELOAD BUFFER POINTER
				151	*					
2181	7C	80	C5	152		MVI	SFG270+@Q(,@BR),@NOP			SET NON-BLANK CARD SWITCH
				153	*					
				154	*		CHECK FOR AND HANDLE CHARACTER CONSTANTS			
				155	*					
2184	BD	7D	00	156		CLI	@ZERO(,@XR),B@SQUO			THIS CHARACTER A QUOTE ?
2187	F2	01	15	157		JNE	SFG235			NO, BYPASS CHAR CON HANDLING
				158	*					
218A	E2	02	01	159	SFG230	LA	@B1(,@XR),@XR			INCR BFR POINTER TO NEXT CHAR
				160	*					
218D	BD	1E	00	161		CLI	@ZERO(,@XR),@EOS			THIS CHARACTER THE OEOS ?
2190	F2	81	4C	162		JE	SFG285			YES, GO SET ERROR CODE
				163	*					
2193	BD	7D	00	164		CLI	@ZERO(,@XR),B@SQUO			THIS CHARACTER A QUOTE ?
2196	D0	01	8A	165		BNE	SFG230(,@BR)			NO, GO INCR TO NEXT
				166	*					
2199	BD	7D	01	167		CLI	@B1(,@XR),B@SQUO			NEXT CHARACTER A QUOTE ?
219C	F2	81	16	168		JE	SFG255			YES, GO INCR TWICE
				169	*					
				170	*		CHECK FOR COMMA DELIMITER REQUIRED			
				171	*					
219F	7D	3D	74	172	SFG235	CLI	SFG225+@D1(,@BR),SFGBLK			IS A BLANK(S) THE ONLY DELIMETER
21A2	7C	3A	74	173		MVI	SFG225+@D1(,@BR),SFGRST			SET SW TO: NEXT BLANK A DELIMITR
21A5	F2	01	0D	174		JNE	SFG255			NOT ONLY DELIM, GO INCR TO NEXT
				175	*					
21A8	3C	6B	0000	176	SFG240	MVI	*-*,@COMMA			REPLACE BLANK DELIM WITH COMMA
21AC	F2	87	06	177		J	SFG255			GO INCR TO NEXT CHARACTER
				178	*					
21AF	7C	3D	74	179	SFG245	MVI	SFG225+@D1(,@BR),SFGBLK			SET SW TO: BLANK ONLY DELIMETER
				180	*					
21B2	74	02	AB	181	SFG250	ST	SFG240+@OP1(,@BR),@XR			SAVE BLANK DELIMETER LOCATION
				182	*					
				183	*		INCREMENT BUFFER POINTER TO NEXT CHARACTER			
				184	*					
21B5	E2	02	01	185	SFG255	LA	@B1(,@XR),@XR			INCR POINTER
21B8	D0	87	6F	186		B	SFG220(,@BR)			GO CHECK NEXT CHARACTER
				187	*					
				188	*		COMMA RECOGNIZED AS THE REAL DELIMITER			
				189	*					
21BB	7C	40	74	190	SFG260	MVI	SFG225+@D1(,@BR),SFGICR			SET SW TO: BLANKS TRANSPARENT
21BE	D0	87	B5	191		B	SFG255(,@BR)			GO INCR TO NEXT CHARACTER
				192	*					
				193	*		ENTIRE CARD HAS BEEN CORRECTLY DELIMITED			
				194	*					
21C1	75	02	FC	195	SFG265	L	SFGCBA(,@BR),@XR			SET POINTER TO BUFFER START
				196	*					
21C4	D0	87	60	197	SFG270	BC	SFG215(,@BR),@UCB			GO READ NEXT CARD IF THIS BLANK
				198	*					
				199	*		GO TO PAGE 3 OF SFGETR TO SYNTAX CHECK THE DATA ITEMS			
				200	*					
21C7	C0	87	12B1	201		B	I\$CALL			GO TO PAGE 3 OF STGETR

SFGETR - PROLOGUE - VM GET ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 161
	21CB	23B7		21CC	202	DC	AL(@VADDR)(V\$XSGT+SFG920-SFGBS1) * FOR SYNTAX CHECK			
					203	*				
					204	*	GO TO PAGE 3 OF SFGETR TO CONVERT AND STACK THE NEXT DATA ITEM			
					205	*				
	21CD	C0 87 12B1			206	SFG280 B	I\$CALL GO TO PAGE 3 OF SFGETR TO			
	21D1	23C6		21D2	207	DC	AL(@VADDR)(V\$XSGT+SFG930-SFGBS1) * CONVERT & STACK ITEM			
					208	*				
	21D3	1C 01 144A 4B			209	SFG282 MVC	I\$VADR,SFGBVA(@VADDR,@BR) UNLOCK PAGE 1-3			
	21D8	C0 87 1350			210	B	I\$UNLK * BUFFER PAGE 1-3			
	21DC	F2 87 0D			211	J	SFG295 GO TO GENERAL SFGETR EXIT			
					212	*				
	21DF	3C BD 0CBC			213	SFG285 MVI	I\$ERRC,@E718 SET ERROR CODE			
	21E3	D0 87 D3			214	B	SFG282(,@BR) GO UNLOCK BUFFER 1-3			
					215	*				
					216	*	DISK FILE - GO TO NEXT PAGE OF SFGETR			
					217	*				
	21E6	C0 87 12B1			218	SFG290 B	I\$CALL EXEC PART 2 OF SFGETR - DISK			
	21EA	2200		21EB	219	DC	AL(@VADDR)(V\$XSGT+B@BLSZ) * FILE DATA ACCESS			
					220	*				
					221	*	GENERAL EXIT FROM SFGETR			
					222	*				
	21EC	1C 01 144A FA			223	SFG295 MVC	I\$VADR,SFGVD2(@VADDR,@BR) UNLOCK			
	21F1	C0 87 1350			224	B	I\$UNLK * DIRECTORY 2			
	21F5	C0 87 12D3			225	B	I\$RTRN RETURN TO CALLER			
					226	*				
					227	*	CONSTANTS, WORKAREAS & EQUATES			
					228	*				
	21F9	0100		21FA	229	SFGVD2 DC	AL(@VADDR)(V\$SFD2) VADDR OF VM DIRECTORY 2			
					230	*				
	21FB			21FC	231	SFGCBA DS	CL(@CADDR) SAVE FLD FOR CORE BFR ADDR			
					232	*				
				00FF	233	SFGNFM EQU	X'FF' NOT FIRST CARD FILE ACCESS MASK			
				00FF	234	SFGCBP EQU	X'FF' DISP. TO CARD BUFFER POINTER			
				0040	235	SFGICR EQU	SFG255-SFG227 DISP. TO BLANK TRANSPARENT			
				003D	236	SFGBLK EQU	SFG250-SFG227 DISP. TO BLANK DEAMITER			
				003A	237	SFGRST EQU	SFG245-SFG227 DISP. TO BLANK RESET			
					238	*				
					239	*	END OF SFGETR - PART 1			
					240	*				

SFGETR - PROLOGUE - VM GET ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 162

```

2200          242      ORG      *,B@LVPG,0          PLACE AT A PAGE BOUNDARY
          2200 243 SFGBS2 EQU      *          ESTABLISH BASE
          2200 244          USING SFGBS2,@BR      * REGISTER USAGE
          245 *-----
          246 *          UPON ENTRY TO PART 2:
          247 *
          248 *          1. A DISK BEEN SET FOR INPUT IS TO BE ACCESSED.
          249 *          2. D2 HAS BEEN LOCKED IN CORE WITH THE MODIFY
          250 *          INDR SET ON.
          251 *          3. @XR POINTS TO THE CURRENT D2 ENTRY.
          252 *-----
          253 *
          254 *          TEST IF CURRENT SEGMENT EMPTY
          255 *
2200 9D 01 0D E2 256      CLC      @$D2LC(@$L2LC,@XR),SFGZRO(@BR)  THIS SEGMENT = ZERO ?
2204 F2 01 06    257      JNE      SFG450          YES, BYPASS NEXT SEGMENT ACCESS
          258 *
          259 *          CALL PART 3 OF SFGETR TO ACCESS NEXT SEGMENT
          260 *
2207 C0 87 12B1 261      B          I$CALL          EXECUTE PART 3 OF SFGETR TO
220B 2300        220C 262      DC      AL(@VADDR)(V$XSGT+2*B@BLSZ)  * ACCESS NEXT SEGMENT
          263 *
          264 *          ACCESS CURRENT BUFFER PAGE AND CHECK FOR EOS
          265 *
220D 7C 05 E8    266 SFG450 MVI      SFGCNL(,@BR),I@LPFS          SET DATA ITEM LNG TO SHORT PREC
2210 7C 80 B0    267      MVI      SFG575+@Q(,@BR),@NOP          SET PREC ADJ. SWITCH FOR SHORT
2213 B8 20 01    268      TBN      @$D2IO(,@XR),@$M2FP          LONG PRECISION ?
2216 F2 90 07    269      JF       SFG470          NO, BYPASS ADD TO LONG PRECISION
2219 7C 87 B0    270      MVI      SFG575+@Q(,@BR),@UCB          SET PREC ADJ. SWITCH FOR LONG
221C 5E 00 E8 E3 271      ALC      SFGCNL(,@BR),SFGDLS(1,@BR)  INCR DATA ITEM LNG TO LONG PRC
2220 6C 01 E6 0D 272 SFG470 MVC      SFGSDF(,@BR),@$D2LC(@$L2LC,@XR)  SAVE SDF COUNT
2224 74 02 76    273      ST       SFGD2P(,@BR),@XR          SAVE D2 ENTRY POINTER
2227 6C 01 34 05 274      MVC      SFGVCB(,@BR),@$D2CP(@$L2CP,@XR)  SET UP VADDR OF CURRENT
222B 6E 00 33 02 275      ALC      SFGVCB-1(,@BR),@$D2VB(@$L2VB,@XR)  * DATA ITEM IN VM BUFFER
222F C0 87 1330 276      B          I$LDXR          ACCESS AND POINT
2233          2234 277 SFGVCB DS      CL(@VADDR)          * @XR TO IT
2235 BD 1C 00    278      CLI      @ZERO(,@XR),@EOF          END OF FILE ?
2238 F2 81 95    279      JE       SFG690          YES, GO SET ERROR CODE
          280 *
          281 *          DETERMINE LENGTH OF DATA ITEM & PLACE IT IN STACK
          282 *
223B 4C 01 70 0D4E 283      MVC      SFGMTA(@CADDR,@BR),I$STAK  INITIALIZE MOVE TO ADDRESS
2240 B8 40 00    284      TBN      @ZERO(,@XR),B@DTYP          CHARACTER CONSTANT ?
2243 F2 90 03    285      JF       SFG500          NO, NUM LNG ALREADY SET. BYPASS
2246 7C 13 E8    286      MVI      SFGCNL(,@BR),I@LCRV          SET DATA ITEM LENGTH FOR CHAR.
          287 *
2249 5C 01 EA E8 288 SFG500 MVC      SFGPCL(,@BR),SFGCNL(2,@BR)  INIT FOR FULL DATA ITEM MOVE
224D 5D 01 E6 E8 289      CLC      SFGSDF(,@BR),SFGCNL(2,@BR)  ALL OF DATA ITEM IN BUFFER
2251 F2 02 04    290      JNL      SFG520          YES, GO SET UP MOVE
2254 5C 01 EA E6 291      MVC      SFGPCL(,@BR),SFGSDF(2,@BR)  NO, RESET MOVE LNG FOR PARTIAL
          292 *
2258 7C FF EC    293 SFG520 MVI      SFGMLQ(,@BR),SFGMS1          SET MOVE LENGTH FOR PART OF
225B 5E 00 EC EA 294      ALC      SFGMLQ(,@BR),SFGPCL(1,@BR)  * DATA ITEM IN CURRENT BUFFER
225F 5C 00 6E EC 295      MVC      SFG550+@Q(,@BR),SFGMLQ(1,@BR)  SET IN MOVE INSTRUCTION
          296 *
2263 76 02 EC    297      A          SFGMLQ(,@BR),@XR          INCR @XR TO END OR BFR DATA

```

SFGETR - PROLOGUE - VM GET ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 163

2266	74 02 72		298 *			
			299	ST	SFGMFA(, @BR), @XR	SET CADDR IN MOVE FROM CADDR
			300 *			
2269	5E 00 70 EC		301	ALC	SFGMTA(, @BR), SFGMLQ(1, @BR)	INCR MOVE TO CADDR
			302 *			
226D	0C 00 0000 0000		303	SFG550 MVC	*-*(@VQ), *-*	MOVE DATA FROM BUFFER TO STACK
		2270	304	SFGMTA EQU	SFG550+@OP1	* MOVE TO STACK ADDRESS
		2272	305	SFGMFA EQU	SFG550+@OP2	* MOVE FROM BUFFER ADDRESS
			306 *			
			307 *		UPDATE D2 ENTRY POINTERS & CHECK IF ALL OF	
			308 *		DATA ITEM MOVED	
			309 *			
2273	C2 02 0000		310	SFG555 LA	*-*, @XR	POINT @XR BACK TO D2 ENTRY
		2276	311	SFGD2P EQU	SFG555+@OP1	* D2 ENTRY CADDR_SAVE AREA
2277	9E 01 05 EA		312	ALC	@\$D2CP(@\$L2CP, @XR), SFGPCL(, @BR)	INCR BFR PT BY MOVE LNG
227B	9F 01 0D EA		313	SLC	@\$D2LC(@\$L2LC, @XR), SFGPCL(, @BR)	INCR SDF COUNT BY MOV LNG
227F	5F 00 E8 EA		314	SLC	SFGCNL(, @BR), SFGPCL(1, @BR)	DECR REQ'D BY ACTUAL LENGTH
2283	F2 81 1F		315	JZ	SFG570	BYPASS BFR REFILL IF DIFRNCE = 0
			316 *			
			317 *		ONLY PART OF THE DATA ITEM WAS IN THE CURRENT	
			318 *		SEGMENT, ACCESS NEXT SEGMENT.	
			319 *		POINT @XR TO NEW SEGMENT.	
			320 *		REDO MOVE PROCESSING FOR SECOND PART OF DATA ITEM	
			321 *		MOVE.	
			322 *			
2286	C0 87 12B1		323	B	I\$CALL	EXECUTE PART 3 OF SFGETR TO
228A	2300	228B	324	DC	AL(@VADDR)(V\$XSGT+2*B@BLSZ)	* ACCESS NEXT SEGMENT
			325 *			
228C	6C 01 E6 0D		326	MVC	SFGSDF(, @BR), @\$D2LC(@\$L2LC, @XR)	SET NEW SEG CT IN SAVEFLD
2290	6C 01 9D 05		327	MVC	SFGVNB(, @BR), @\$D2CP(@\$L2CP, @XR)	SET UP VADDR OF NEW
2294	6E 00 9C 02		328	ALC	SFGVNB-1(, @BR), @\$D2VB(@\$L2VB, @XR)	* SEGMENT
2298	C0 87 1330		329	B	I\$LDXR	ACCESS & POINT @XR AT IT
229C		229D	330	SFGVNB DS	CL(@VADDR)	VADDR OF NEW SEGMENT
229E	5E 00 70 E4		331	ALC	SFGMTA(, @BR), SFGONE(1, @BR)	INCR MOVE TO ADDR ROR NEXT MOV
22A2	D0 87 49		332	B	SFG500(, @BR)	SO MOVE REST OF DATA ITEM
			333 *			
			334 *		ENTIRE DATA ITEM MOVED TO STACK. SET CORRECT	
			335 *		PRECISION IF NUMERIC.	
			336 *			
22A5	35 02 0D4E		337	SFG570 L	I\$STAK, @XR	POINT @XR TO STACKED DATA ITEM
22A9	B8 40 00		338	TBN	I@STAT(, @XR), B@DTYP	CHARACTER ITEM ?
22AC	F2 10 25		339	JT	SFG695	YES, GO TO RETURN
			340 *			
22AF	F2 80 11		341	SFG575 JC	SFG585, @NOP	JUMP IF FILE PREC = LONG
22B2	F2 87 1F		342	JC	SFG695, I@PRSW	JUMP TO EXIT IF RUN PREC = SHORT
			343 *			
			344 *		CHANGE PRECISION FROM SHORT TO LONG	
			345 *			
22B5	BA 20 00		346	SBN	I@STAT(, @XR), B@PREC	SET PREC = LONG
22B8	AC 00 08 04		347	MVC	I@PEXL(, @XR), I@PEXS(@B1, @XR)	MOVE EXP TO LONG POSITION
22BC	AF 03 07 07		348	SLC	I@PEXL-1(, @XR), I@PEXL-1(SFGELS, @XR)	SET EXTRA DIGITS = 0
22C0	F2 87 11		349	J	SFG695	EXIT
			350 *			
22C3	F2 80 0E		351	SFG585 JC	SFG695, @UCB-I@PRSW+@NOP	JUMP TO EXIT IF RUN PREC = LONG
			352 *			
			353 *		CHANGE PRECISION FROM LONG TO S4ORT	

SFGETR - PROLOGUE - VM GET ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 164
					354	*				
	22C6	BB	20 00		355	SBF	I@STAT(,@XR),B@PREC		SET PREC = SHORT	
	22C9	AC	00 04 08		356	MVC	I@PEXS(,@XR),I@PEXL(1,@XR)		MOVE EXP TO SHORT POSITION	
	22CD	F2	87 04		357	J	SFG695		JUMP TO EXIT	
					358	*				
					359	*	SET END OF FILE ERROR CODE			
					360	*				
	22D0	3C	B9 0CBC		361	SFG690	MVI I\$ERRC,@@E714		SET EOF CODE	
					362	*				
					363	*	RETURN TO PART 2 OF SFGETR			
					364	*				
	22D4	1C	01 144A 9D		365	SFG695	MVC I\$VADR,SFGVNB(@VADDR,@BR)		MOVE BUFFER PAGE	1-5
	22D9	C0	87 1350		366	B	I\$UNLK		UNLOCK PAGE	1-5
	22DD	C0	87 12D3		367	B	I\$RTRN		EXIT	
					368	*				
					369	*	PART 2 - CONSTANTS, WORKAREAS & EQUATES			
					370	*				
	22E1	0000		22E2	371	SFGZRO	DC XL(@\$L2CP)'0'		ZERO	
	22E3	04		22E3	372	SFGDLS	DC AL1(I@LPFL-I@LPFS)		DIFFERENCE IN PRECISION LENGTHS	
	22E4	01		22E4	373	SFGONE	DC XL1'1'		ONE	
					374	*				
				00FF	375	SFGMS1	EQU X'FF'		MINUS 1	
				0004	376	SFGELS	EQU I@LPFL-I@LPFS		LNG LONG PREC EXTRA SIGNIFICNCE	
					377	*				
	22E5			22E6	378	SFGSDF	DS CL(@\$L2LC)		SDF COUNT WORKAREA	
	22E7			22E8	379	SFGCNL	DS CL(@CADDR)		ACTUAL LENGTH OF DATA ITEM	
	22E7				380	ORG	SFGCNL-1		* INITIALIZE TO	
	22E7	0000		22E8	381	DC	XL(@CADDR)'0'		* ZERO	
	22E9			22EA	382	SFGPCL	DS CL(@CADDR)		BUFFER LNG OF DATA ITEM	
	22EB			22EC	383	SFGMLQ	DS CL(@CADDR)		PHYS. MOVE LNG & DISPLACEMENT	
	22EB				384	ORG	SFGMLQ-1		* INITIALIZE TO	
	22EB	0000		22EC	385	DC	XL(@CADDR)'0'		* ZERO	
					386	*				
					387	*	END OF SFGETR - PART 2			
					388	*				

SFGETR - PROLOGUE - VM GET ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 165

```

2300          2300 390      ORG    *,B@LVPG,0      PLACE AT A PAGE BOUNDARY
                2300 391 SFGBS3 EQU    *          ESTABLISH BASE
                2300 392      USING SFGBS3,@BR    * REGISTER USAGE
                393 *-----
                394 *          UPON ENTRY TO PART 3:
                395 *
                396 *          1. D2 HAS BEEN LOCKED IN CORE WITH THE
                397 *             MODIFY INDICATOR SET ON.
                398 *          2. @XR POINTS TO THE CURRENT D2 ENTRY
                399 *          3. THE CURRENT D2 ENTRY VM BUFFER POINTERS
                400 *             MUST BE SET TO THE FIRST DATA ITEM IN THE
                401 *             NEXT(FIRST) SEGMENT OR BUFFER.
                402 *-----
                403 *
                404 *          CHECK FOR MORE SEGMENTS IN CURRENT BUFFER
                405 *
2300 BD 00 05    406 SFG750 CLI    @$D2CB(,@XR),@ZERO    ANY SPACE LEFT IN CURR BFR ?
2303 F2 01 52    407          JNE    SFG830              YES, GO ACCESS BUFFER
2306 BD 00 04    408          CLI    @$D2CS(,@XR),@ZERO    INITIAL FILL-UP CALL ?
2309 F2 81 07    409          JE     SFG780              YES, GO TO GET SFLOAD
230C AD 00 03 04 410 SFG760 CLC    @$D2BS(,@XR),@$D2CS(@$L2CS,@XR) MORE VM BUFFERS ?
2310 F2 84 32    411          JH     SFG810              YES, GO CHECK DATA FILE TYPE
                412 *
                413 *          VM BUFFERS MUST BE REFILLED.  WRITE OUT INTERPRETER
                414 *          AND ACCESS & EXECUTE SFLOAD.
                415 *
2313 74 01 1E    416 SFG780 ST     SFGWPL(,@BR),@BR      SET UP DPL TO WRITE OUT
2316 7C E3 1E    417          MVI    SFGWPL(,@BR),SFGDWL    * INTERPRETER
                418 *SFG785 DISK    @ZERO          GO WRITE IT OUT
2319 C0 87 0025  419 SFG785 B     $DISKN          PERFORM PHYSICAL DISK OP
231D 0000        420          DC     AL2(@ZERO)          DPL ADDRESS
                421 *** END OF EXPANSION ***
                231E 423 SFGWPL EQU    SFG785+5          ADDRESS OF WRITE DPL
                424          ST     SFGRPL(,@BR),@BR      SET UP DPL TO READ IN
                425          MVI    SFGRPL(,@BR),SFGDRL    * #SFLOA
                426          MVI    I$WRK1-1,@DGET        SET INPUT INDR FOR #SFLOA
2322 7C E9 34    427          ST     SFGSBR(,@BR),@BR      SAVE BASE REGISTER
2325 3C 01 0D58  428          ST     SFGSXR(,@BR),@XR      SAVE D2 POINTER
2329 74 01 3A    429 *SFG790 BLOAD @ZERO          GO EXECUTE #SFLOA
232C 74 02 3E    430 SFG790 B     $BLOAD          LOAD AND EXECUTE WORK AREA PGM
                431          DC     AL2(@ZERO)          DPL ADDRESS
232F C0 87 0522  432 *** END OF EXPANSION ***
                2334 434 SFGRPL EQU    SFG790+5          ADDRESS OF READ DPL
                435 *
                436 *          RETURN FROM $SFLOA
                437 *
2335 0444        438          DC     AL(@CADDR)($DPLSV-5)  CADDR OF DPL TO RELOAD INTERP
2337 C2 01 0000  439 SFG795 LA     *-*,@BR          RESTORE BASE REGISTER
                233A 440 SFGSBR EQU    SFG795+@OP1        CADDR OF @BR SAVE AREA
233B C2 02 0000  441 SFG800 LA     *-*,@XR          RESTORE D2 POINTER
                233E 442 SFGSXR EQU    SFG800+@OP1        CADDR OF D2 POINTER SAVE AREA
                443 *          DISK    $WAITF          WAIT FOR INTERPRETER
233F C0 87 0025  444          B     $DISKN          PERFORM PHYSICAL DISK OP
2343 057F        445          DC     AL2($WAITF)          DPL ADDRESS

```

SFGETR - PROLOGUE - VM GET ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 166
				446	***	END OF EXPANSION	***			
				448	*					
				449	*	DETERMINE DATA FILE TYPE				
				450	*					
2345	B8	10	01	451	SFG810	TBN	@\$D2IO(, @XR), @\$M2FT		PROG-GENERATED DATA FILE ?	
2348	F2	90	0A	452		JF	SFG825		NO, GO HANDLE KEYBOARD DATA FILE	
234B	BC	00	05	453		MVI	@\$D2CB(, @XR), @ZERO		SET BYTE POINTER TO ZERO	
234E	9C	01	0D	454		MVC	@\$D2LC(@\$L2LC, @XR), SFGSSZ(, @BR)		SET SEG COUNT TO MAX.	
2352	F2	87	5E	455		J	SFG900		GO TO EXIT	
2355	BC	01	05	457	SFG825	MVI	@\$D2CB(, @XR), @B1		SET BYTE POINTER TO 1ST SDF	
				458	*					
				459	*	KEYBOARD DATA FILE - CHECK FIRST/NEXT SDF				
				460	*					
2358	74	02	3E	461	SFG830	ST	SFGSXR(, @BR), @XR		SAVE D2 ENTRY POINTER	
235B	6C	01	68	462		MVC	SFGCBV(, @BR), @\$D2CP(@VADDR, @XR)		SET UP VADDR OF	
235F	6E	00	67	463		ALC	SFGCBV-1(, @BR), @\$D2VB(@\$L2VB, @XR)		* CURRENT BUFFER	
2363	C0	87	1330	464		B	I\$LDXR		ACCESS & PT @XR TO NEXT SDF	
2367				465	SFGCBV	DS	CL(@VADDR)		VADDR OF CURRENT BUFFER	
				466	*					
2369	BD	80	00	467		CLI	@SDF0(, @XR), @SNULL		IS THE NEXT SEGMENT NULL ?	
236C	F2	01	0A	468		JNE	SFG840		NO, GO CHECK SEGMENT TYPE	
				469	*					
236F	75	02	3E	470		L	SFGSXR(, @BR), @XR		RESTORE D2 ENTRY POINTER	
2372	9E	00	04	471		ALC	@\$D2CS(@\$L2CS, @XR), SFGPAF(, @BR)		NULL, INCR PT TO NEXT PAGE	
2376	D0	87	0C	472		B	SFG760(, @BR)		GO ACCESS NEXT BUFFER	
				473	*					
				474	*	TEST NEXT SEGMENT TYPE AND USAGE STATUS				
				475	*					
2379	6C	06	FA	476	SFG840	MVC	SFGSHD(SFGHDL, @BR), SFGDEH(, @XR)		MOVE SEG HDR TO SAVE AREA	
237D	75	02	3E	477		L	SFGSXR(, @BR), @XR		RESTORE D2 ENTRY POINTER	
2380	78	02	F6	478		TBN	SFGLEH+SDF2(, @BR), @SLAST		PRIMARY SEGMENT ?	
2383	F2	90	0E	479		JF	SFG860		YES, GO CHECK IF DISABLED	
				480	*					
				481	*	SECONDARY SEGMENT				
				482	*					
2386	F2	80	14	483	SFG850	JC	SFG870, @NOP		JUMP IF LINE DISABLED	
2389	9E	00	05	484		ALC	@\$D2CB(@\$L2CB, @XR), SFGSSL(, @BR)		INCR CURR PT BY HDR LNG	
238D	5F	00	F5	485		SLC	SFGLEH+SDF1(1, @BR), SFGSSL(, @BR)		DECR SEG CT BY HDR LNG	
2391	F2	87	1B	486		J	SFG890		GO SET ADJ SEG CT IN D2 ENTRY	
				487	*					
				488	*	PRIMARY SEGMENT				
				489	*					
2394	78	80	FA	490	SFG860	TBN	SFGLEH+@STYPE(, @BR), B@SDMK		STATEMENT DISABLED ?	
2397	F2	90	0A	491		JF	SFG880		NO, BYPASS BYPASS OF SEG	
239A	7C	87	87	492		MVI	SFG850+@Q(, @BR), @UCB		SET SWITCH FOR 2NDARY SEGMENTS	
239D	9E	01	05	493	SFG870	ALC	@\$D2CP(@\$L2CP, @XR), SFGLEH+SDF1(, @BR)		INCR CURR PT BY LNG	
23A1	D0	87	00	494		B	SFG750(, @BR)		GO ACCESS THE NEXT SEGMENT	
				495	*					
23A4	7C	80	87	496	SFG880	MVI	SFG850+@Q(, @BR), @NOP		RESET DISABLED SWITCH	
23A7	9E	00	05	497		ALC	@\$D2CB(@\$L2CB, @XR), SFGPSL(, @BR)		INCR CURR PT BY HDR LNG	
23AB	5F	00	F5	498		SLC	SFGLEH+SDF1(1, @BR), SFGPSL(, @BR)		DECR SEG CT BY HDR LNG	
23AF	9C	01	0D	499	SFG890	MVC	@\$D2LC(@\$L2LC, @XR), SFGLEH+SDF1(, @BR)		SET SEG CT IN ENTRY	
				500	*					
				501	*	ALL DONE - GO AWAY				

SFGETR - PROLOGUE - VM GET ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 167

```

      23B3 C0 87 12D3          502 *
      503 SFG900 B      I$RTRN          RETURN TO PART 2 OF SFGETR
      504 *
      505 *          SYNTAX CHECK THE ENTIRE CARD
      506 *
      23B7 74 02 FA          507 SFG920 ST      SFGBS2(,@BR),@XR          SAVE THE BUFFER POINTER
      508 *
      23BA C0 87 12B1          509          B      I$CALL          GO SYNTAY CWECK FOR
      23BE 2E00          23BF 510          DC      AL(@VADDR)(V$CDSY)          * VALID DATA ITEMS
      511 *
      23C0 75 02 FA          512          L      SFGBS2(,@BR),@XR          RESTORE THE BUFFER POINTER
      23C3 F2 87 16          513          J      SFG940          JUMP TO CONFIGURE 2ND PASS      1-3
      514 *
      515 *          CONVERT AND STACK THE NEXT DATA ITEM UNLESS AN ERROR HAS OCCURED
      516 *
      23C6 F2 80 07          517 SFG930 JC      SFG935,@NOP          JOMP FIRST PASS      1-3
      23C9 74 02 FA          518          ST      SFGBS2(,@BR),@XR          SAVE XR(BUFFER CADDR)      1-3
      23CC 9C 00 FE F9          519          MVC      SFGXRD(@CADDR-1,@XR),SFGBS2-1(,@BR) SET TRUE BUFR CADR 1-3
      520 *
      23D0 C0 87 12B1          521 SFG935 B      I$CALL          SO CONVERT AND STACK NEXT
      23D4 3100          23D5 522          DC      AL(@VADDR)(V$CDCV)          * DATA ITEM
      523 *
      23D6 7C 80 C7          524          MVI      SFG930+@Q(,@BR),@NOP          FORCE SAVE BUFFER CADDR      1-3
      23D9 F2 87 03          525          J      SFG945          JUMP TO RETURN TO CALLER      1-3
      526 *
      23DC 7C 87 C7          527 SFG940 MVI      SFG930+@Q(,@BR),@UCB          FORCE NO SAVE OR BUFR ADDR      1-3
      528 *
      23DF C0 87 12D3          529 SFG945 B      I$RTRN          RETURN TO CALLER      1-3
      530 *
      531 *          PART 3 - DISK PARAMETER LISTS.
      532 *
      533 *          WRITE OUT INTERPRETER
      534 *
      535 *SFGPLW DPL      FUNC=@DPUT,DADDR=#@VSFI,CNT=#@@VSL,CADDR=##$INS
      23E3 02          23E3 536 SFGPLW EQU      *          DISK PARAMETER LIST
      23E4 09A1          23E5 537          DC      AL1(@DPUT)          REQUESTED FUNCTION
      23E6 0F          23E6 538          DC      AL2(#@VSFI)          DISK ADDRESS
      23E7 0600          23E8 539          DC      AL1(#@@VSL)          SECTOR COUNT
      540          DC      AL2(##$INS)          BUFFER ADDRESS
      541 *** END OF EXPANSION ***
      00E3          543 SFGDWL EQU      SFGPLW-SFGBS3          DISP. TO WRITE DPL
      544 *
      545 *          READ IN SFLOAD
      546 *
      547 *SFGPLR DPL      FUNC=@DGET,DADDR=#@SFLO,CNT=#@@SFL,CADDR=##$$SFL
      23E9 01          23E9 548 SFGPLR EQU      *          DISK PARAMETER LIST
      23EA 0499          23EB 549          DC      AL1(@DGET)          REQUESTED FUNCTION
      23EC 05          23EC 550          DC      AL2(#@SFLO)          DISK ADDRESS
      23ED 0F00          23EE 551          DC      AL1(#@@SFL)          SECTOR COUNT
      552          DC      AL2(##$$SFL)          BUFFER ADDRESS
      553 *** END OF EXPANSION ***
      00E9          555 SFGDRL EQU      SFGPLR-SFGBS3          DISP TO READ DPL
      556 *
      557 *          PART 3 - CONSTANTS, RORKAREAS AND EQUATES.

```

SFGETR - PROLOGUE - VM GET ROUTINE

ERR LOC		OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21		PAGE 168
				558	*				
23EF	0100		23F0	559	SFGSSZ DC	AL(@\$L2LC)(B@BLSZ)		MAX	SEGMENT COUNT
23F1	01		23F1	560	SFGPAF DC	XL(@\$L2CS)'1'			
23F2	04		23F2	561	SFGSSL DC	XL(@\$L2CB)'4'		LENGTH OF	2NDARY SEG. HDR.
23F3	07		23F3	562	SFGPSL DC	XL(@\$L2CB)'7'		LENGTH OF	PRIMARY SEG. HDR.
				563	*				
			0F00	564	SFGSA0 EQU	\$\$KLD1+X'0900'		CORE LOAD	ADOR OF #SFLOAD
			0007	565	SFGHDL EQU	@STEXT		SEGMENT	HEADER LENGTH
			0006	566	SFGDEH EQU	SFGHDL-1		DISP TO	RIGHT END OF SEG. HDR.
				567	*				
			23F4	568	SFGLEH EQU	*		LEFT END	OF HEADER SAVE AREA
23F4			23FA	569	SFGSHD DS	CL(SFGHDL)		SEGMENT	HEADER SAVE AREA
			23FA	570	SFGSB2 EQU	SFGSHD		SAVE AREA	FOR CARD SBR ADDRESS
			00FE	571	SFGXRD EQU	X'FE'		BUFFER	CADDR DISP INTO BFR 1-3
				572	*				
				573	*	END OF SFGETR PART 3			
				574	*				

SFGETR - PROLOGUE - VM GET ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 169

2400			576	ORG	*,B@LVPG,0	PLACE MODULE AT PAGE BOUNDARY
			2400 577	USING	SFRBS1,@BR	ESTABLISH BASE REGISTER
			2400 578	SFRBS1 EQU	*	IT ON FIRST BYTE OF PAGE
			579 *			
			580 *			TERMINATION ENTRY TO CLOSE ALL DATA FILES
			581 *			
			2400 582	SFRCAL EQU	*	
2400 7C 80 73			583	MVI	SFR900+@Q(,@BR),@NOP	SET SW FOR CLOSE ALL LOOP
2403 7C 80 17			584	MVI	SFR110+@Q(,@BR),@NOP	SET SW FOR CLOSE ALL INITIALIZED
2406 3A 1E 03E4			585	SBN	\$LPRP3,@KENAB	SET MATRIX PRINTER MODE 1-3
			586 *			
			587 *			ENTRY TO CLOSE A SPECIFIED DATA FILE
			588 *			
			240A 589	SFRCLS EQU	*	
240A 7C 87 50			590	MVI	SFR140+@Q(,@BR),@UCB	SET SWITCH FOR CLOSE
			591 *			
			592 *			ENTRY TO RESET A SPECIFIED FILE
			593 *			
			240D 594	SFRSET EQU	*	
			595 *			
			596 *			ACCESS DIRECTORY 2 & REFERENCE SPECIFIED FILE
			597 *			
240D C0 87 1330			598	SFR100 B	I\$LDXR	GET VM DIRECTORY 2 AND
2411 0100		2412	599	SFRVD2 DC	AL(@VADDR)(V\$SFD2)	* POINT @XR TO IT
2413 74 02 84			600	ST	SFRIXR(,@BR),@XR	SAVE POINTER TO D2 RECORD
			601 *			
			602 *			SET TO FIRST ENTRY IF CLOSE ALL
			603 *			
2416 F2 87 03			604	SFR110 JC	SFR115,@UCB	JUMP IF NOT CLOSE ALL
2419 BC 40 01			605	MVI	@\$D2CF(,@XR),@\$D2E1	SET DISPLACEMENT TO 1ST ENTRY
			606 *			
241C B6 02 01			607	SFR115 A	@\$D2CF(,@XR),@XR	INCR @XR TO SPECIFIED FILE
			608 *			
			609 *			DETERMINE IF THE FILE IS INPUT OR OUTPUT
			610 *			
241F BD 00 00			611	SFR130 CLI	@\$D2DC(,@XR),@ZERO	THIS FILE ACTIVE ?
2422 F2 81 4D			612	JE	SFR900	NO, GO CHECK IF CLOSE ALL
2425 B8 08 01			613	TBN	@\$D2IO(,@XR),@\$M2CI	CURRENT USAGE - INPUT ?
2428 F2 10 24			614	JT	SFR140	YES, BYPASS CLOSE CALL TO SFPUT
242B B8 04 01			615	TBN	@\$D2IO(,@XR),@\$M2CO	CURRENT USAGE - OUTPUT ?
242E F2 90 41			616	JF	SFR900	NO, NOT ACTIVE. GO CHK CLOSE ALL
2431 BA 02 01			617	SBN	@\$D2IO(,@XR),@\$M2EF	SET END OF FILE INDR
2434 74 02 4B			618	ST	SFR135+@OP1(,@BR),@XR	SAVE D2 ENTRY POINTER
2437 35 02 0D4E			619	L	I\$STAK,@XR	MOVE AN END OF FILE CODE
243B BC 1C 00			620	MVI	I@STAT(,@XR),@EOF	* TO THE STACK
243E C0 87 1354			621	B	I\$LOCK	LOCK D2 IN CORE
2442 C0 87 12B1			622	B	I\$CALL	CALL SFPUTR TO EMPTY THE
2446 1D00		2447	623	DC	AL(@VADDR)(V\$XSPT)	* FILE BUFFER(S)
2448 C2 02 0000			624	SFR135 LA	*-*,@XR	RESTORE D2 ENTRY POINTER
244C BB 02 01			625	SBF	@\$D2IO(,@XR),@\$M2EF	SET THE END OF FILE INDR OFF
			626 *			
			627 *			CHECK IF RESET OR CLOSE
			628 *			
244F F2 80 0F			629	SFR140 JC	SFR300,@NOP	JUMP IF CLOSE

SFGETR - PROLOGUE - VM GET ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 170
				631	*					
				632	*		RESET REQUIRED			
				633	*					
2452	AF	01	05 05	634	SFR200	SLC	@\$D2CP(,@XR),@\$D2CP(@\$L2CP,@XR)		CLEAR CURRENT POINTER	
2456	AF	01	09 09	635		SLC	@\$D2DD(@\$L2DD,@XR),@\$D2DD(,@XR)		CLEAR DISK DISP	
245A	AF	01	0D 0D	636		SLC	@\$D2LC(@\$L2LC,@XR),@\$D2LC(,@XR)		CLEAR SPF COUNT	
245E	F2	87	11	637		J	SFR900		GO CHECK IF CLOSE ALL	
				638	*					
				639	*		CLOSE REQUIRED			
				640	*					
2461	BB	0C	01	641	SFR300	SBF	@\$D2IO(,@XR),@\$M2CI+@\$M2CO		SET CURRENT USAGE OFF	
2464	B8	40	00	642		TBN	@\$D2DC(,@XR),@\$MBSD		SCRATCH DISK FILE ?	
2467	D0	10	52	643		BT	SFR200(,@BR)		YES, GO CLEAR CURRENT POINTER'S	
246A	AF	01	01 01	644		SLC	@\$D2IO(,@XR),@\$D2IO(@\$L2DC+@\$L2IO,@XR)		CLEAR ENTRY EXCEPT	
246E	AF	0B	0F 0F	645		SLC	@\$D2EE(,@XR),@\$D2EE(@\$L2E-@\$D2CS,@XR)		* FOR VM BFR BSE&SIZ	
				646	*					
				647	*		SPECIFIED FILE HAS SEEN CLOSED OR RESET AS REQUIRED.			
				648	*		IF CLOSE ALL CONTINUE TILL ALL 8 ENTRIES CLOSED			
				649	*					
2472	F2	87	17	650	SFR900	JC	SFR995,@UCB		JUMP TO RETURN IF NOT CLOSE ALL	
2475	5F	00	AB AA	651		SLC	SFRNOE(,@BR),SFRONE(1,@BR)		DECR NO. OF ENTRIES COUNTER	
2479	F2	81	10	652		JZ	SFR995		JUMP TO RETURN IF ZERO.	
247C	1C	01	144A 12	653		MVC	I\$VADR,SFRVD2(@VADDR,@BR)		RESTORE VADDR OF D2 TO PG.RTN.	
2481	C2	02	0000	654	SFR950	LA	*-*,@XR		RESTORE POINTER TO D2 RECORD	
				2484 655	SFR1XR	EQU	SFR950+@OP1		SAVE AREA FOR D2 RCD POINTER	
2485	9E	00	01 AC	656		ALC	@\$D2CF(,@XR),SFRX10(1,@BR)		INCR FILE PT TO NEXT ENTRY	
2489	D0	87	1C	657		B	SFR115(,@BR)		GO INCR @XR TO NEXT ENTRY & CHK	
				658	*					
				659	*		FUNCTION COMPLETE - RESTORE ROUTINE & EXIT			
				660	*					
248C	7C	80	50	661	SFR995	MVI	SFR140+@Q(,@BR),@NOP		SET RTN FOR RESET FUNCTION	
248F	3B	1E	03E4	662		SBF	\$LPRP3,@KENAB		RESET MATRIX PRINT MODE	1-3
2493	1C	01	144A 12	663		MVC	I\$VADR,SFRVD2(@VADDR,@BR)		SPECIFY DIRECTORY 2	
2498	C0	87	1349	664		B	I\$MDFY		SET PAGE TO MODIFIED	
249C	7C	87	17	665		MVI	SFR110+@Q(,@BR),@UCB		RESET JUMP CONDITION	1-5
249F	7C	87	73	666		MVI	SFR900+@Q(,@BR),@UCB		RESET JUMP CONDITION	1-5
24A2	C0	87	1350	667		B	I\$UNLK		UNLOCK PAGE	
24A6	C0	87	12D3	668		B	I\$RTRN		RETURN TO CALLER	
				669	*					
				670	*		CONSTANTS, WORKAREAS & EQUATES			
				671	*					
24AA	01			24AA 672	SFRONE	DC	XL1'1'		SIMPLY ONE	
				673	*					
24AB				24AB 674	SFRNOE	DS	CL1		NUMBER OF D2 ENTRIES COUNTER	
24AB				675		ORG	SFRNOE		* INITIALIZE TO MAXIMUM	
24AB	0C			24AB 676		DC	AL1(@\$MBEN)		* NUMBER OF D2 ENTRIES	
24AC	10			24AC 677	SFRX10	DC	XL1'10'		D2 ENTRY LENGTH	

SFGETR - PROLOGUE - VM GET ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 171
					679	*****	*****			
					680	*				*
					681	*	LINE PRINTER CLOSE OUT ROUTINE			*
					682	*				*
					683	*****	*****			
				2400	684	USING	SFRBS1,@BR	SET BASE REGISTER USAGE	1-4	
24AD	F1	E2	00		685	APL	@PBUSY	LOOP PRINTER BUSY	1-4	
24B0	3D	00	03E3		686	CLI	\$BUFPT,@ZERO	IS LINE PRINTER BUFFER EMPTY	1-4	
24B4	F2	81	13		687	JE	SFR997	JUMP IF BUFFER EMPTY	1-4	
24B7	74	02	C6		688	SFRLPR	ST SFR996+@OP1(,@BR),@XR	SAVE XR	1-4	
24BA	D2	02	E5		689	LA	SF1000(,@BR),@XR	XR = CADDR PPL	1-4	
24BD	C0	87	12B1		690	B	I\$CALL	BRANCH TO CALL ROUTINE	1-4	
24C1	2800			24C2	691	DC	AL(@VADDR)(V\$SPRT)	MATRIX PRINTER PAGE	1-4	
24C3	C2	02	0000		692	SFR996	LA *-*,@XR	RESTORE XR	1-4	
24C7	F2	87	07		693	J	SFR998	GO RESTORE TRUE POSITION	1-4	
24CA	38	01	03E4		694	SFR997	TBN \$LPRP3,@INDEX	IS DUMMY PRT POSITION IN USE	1-4	
24CE	F2	90	06		695	JF	SFR999	JUMP IF NOT	1-4	
24D1	0C	00	03C2 03E5		696	SFR998	MVC \$PRPOS(1),\$LPROS	RESTORE TRUE PRINT POSITION	1-4	
24D7	3C	00	03E4		697	SFR999	MVI \$LPRP3,@ZERO	RESET LINE PRINTER FLAGS	1-4	
24DB	F1	E2	00		698	APL	@PBUSY	LOOP IF PRINTER BUSY	1-4	
24DE	D1	E0	B7		699	TIO	SFRLPR(,@BR),@PERR	BRANCH IF PRINTER UNIT CHECK	1-4	
24E1	C0	87	12D3		700	B	I\$RTRN	RETURN TO CALLER	1-4	
					701	*			1-4	
24E5	80			24E5	702	SF1000	DC XL1'80'	PPL - FORCE CARRAGE RETURN	1-4	
					703	*				
					704	*****	END SFRSET	*****		

DFKEYN - VIRTUAL MEMORY KEYBOARD ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 172
		706		*****	
		707	*	5703-XM1 COPYRIGHT IBM CORP. 1970	*
		708	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083	*
		709	*		*
		710		*****	
		711	*	*STATUS	*
		712	*	VERSION 1 MODIFICATION 0	*
		713	*		*
		714	*	*FUNCTION	*
		715	*	DFKEYN IS DIVIDED INTO TWO SECTIONS PERFORMING TWO GENERAL	*
		716	*	FUNCTIONS:	*
		717	*	* CALL SECTION	*
		718	*	THE CALL SECTION ENABLES AND UNLOCKS THE KEYBOARD IN	*
		719	*	PREPARATION FOR LINE INPUT. IT THEN SETS THE INTERRUPT	*
		720	*	ADDRESS WHICH IS ENTERED ON THE KEYBOARD INTERRUPT LEVEL WHEN	*
		721	*	A KEY IS DEPRESSED.	*
		722	*	* INTERRUPT SECTION	*
		723	*	THE INTERRUPT SECTION SAVES THE SYSTEM STATUS (BR, XR & PSR)	*
		724	*	AND HANDLES THE INPUT FROM THE KEYBOARD. UPON COMPLETION OF	*
		725	*	THE INPUT LINE, \$KYBSY IS SET TO ZERO INDICATING THAT THE	*
		726	*	LINE IS COMPLETE. THEN THE KEYBOARD IS LOCKED.	*
		727	*	THE INPUT FROM THE KEYBOARD IS CLASSIFIED AND HANDLED AS	*
		728	*	FOLLOWS:	*
		729	*	* DATA KEYS -- THE CHARACTER REPRESENTATION IS PLACED IN	*
		730	*	THE INPUT LINE BUFFER AND PRINTED ON THE	*
		731	*	SYSTEM PRINTER.	*
		732	*	* CMD KEYS -- IF THE CRT IS AVAILABLE, DSPLYN IS CALLED	*
		733	*	TO SET THE FUNCTION FOR KEYS 12-16.	*
		734	*	AN INDICATOR IS PLACED IN THE INPUT LINE	*
		735	*	BUFFER (SPECIFIED LOCATION) FOR COMMAND	*
		736	*	KEYS 1-11.	*
		737	*	* FUNCTION KEYS -- AS FOLLOWS	*
		738	*	TAB - IF THE CURRENT POSITION IN THE LINE BUFFER IS	*
		739	*	POINTING WITHIN AN EXISTING LINE, THE OLD	*
		740	*	CHARACTER IS PRINTED. IF NOT, A BLANK IS PRINTED	*
		741	*	THIS POSITIONS THE CARRIER ONE SPACE TO THE RIGHT	*
		742	*	IF THE KEY IS HELD DOWN, THE TYPOMATIC FEATURE IS	*
		743	*	ACTIVATED, REPEATING THE ABOVE FUNCTION UNTIL	*
		744	*	KEY IS RELEASED.	*
		745	*	BACKSPACE - IF THE SYSTEM PRINTER IS THE MATRIX PRINTR	*
		746	*	AND IF THIS WAS THE FIRST BACKSPACE FOR THE	*
		747	*	CURRENT LINE, THE CARRIAGE IS INDEXED AND	*
		748	*	BACKSPACED ONE POSITION. OTHERWISE, THE INDEX	*
		749	*	FEATURE IS NOT EXECUTED. IF THE KEY IS HELD DOWN	*
		750	*	THE TYPOMATIC FEATURE IS ACTIVATED AND THE ABOVE	*
		751	*	FUNCTION IS REPEATED UNTIL THE KEY IS RELEASED.	*
		752	*	RETURN - THE CARRIAGE IS RETURNED ON THE SYSTEM PRINTR	*
		753	*	AND \$KYBSY SET TO ZERO INDICATING THE LINE IS	*
		754	*	COMPLETE. THE KEYBOARD IS THEN LOCKED.	*
		755	*	ERASE - THE CARRIAGE IS RETURNED ON THE SYSTEM PRINTER	*
		756	*	ALLOWING THE LINE TO BE RE-ENTERED.	*
		757	*	INQUIRY REQUEST - THE CURRENT OPERATION IS ABORTED.	*
		758	*	THIS KEY IS NEVER LOCKED.	*
		759	*	NOTE: THE ENTER(+) AND PROGRAM START KEYS ARE IGNORED	*
		760	*		*
		761	*	*ENTRY POINTS	*

DFKEYN - VIRTUAL MEMORY KEYBOARD ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 173
		762	*	ENTRY TO DFKEYN IS VIA THE VIRTUAL MEMORY PAGING ROUTINE.	*
		763	*	THE CALLING SEQUENCE IS:	*
		764	*	B I\$CALL	*
		765	*	DC AL(@CADDR)(V\$SKEY)	*
		766	*	WHERE V\$SKEY IS THE VIRTUAL ADDR OF THE VIRTUAL MEMORY KEYBOARD	*
		767	*	INPUT IOCR. THE CALL TO DFKEYN INCLUDES THE PASSING IN @XR OF	*
		768	*	THE ADDRESS OF THE INPUT DATA BUFFER.	*
		769	*		*
		770	*	*INPUT	*
		771	*	INPUT TO THE VIRTUAL MEMORY IOCR IS THE ADDRESS IN @XR OF THE	*
		772	*	INPUT LINE BUFFER AND THE INPUT DATA.	*
		773	*		*
		774	*	*OUTPUT	*
		775	*	THE OUTPUT FROM THIS ROUTINE IS AN EBCDIC CHARACTER TO THE SYSTEM	*
		776	*	PRINTER AND THE LINE BUFFER.	*
		777	*		*
		778	*	*EXTERNAL REFERENCES	*
		779	*	\$BRSAB - COMMON SAVE AREA FOR BASE REGISTER	*
		780	*	\$CIENT - NUCLEUS ENTRY FOR INTERRUPTS	*
		781	*	\$PRDEV - INDICATOR FOR CURRENT I/O DEVICE	*
		782	*	\$KEYCD - THUNCATED LINE INDICATOR (\$TRUNK)	*
		783	*	\$IOIND - HARD I/O ERROR INDICATOR (\$HRDER) SYSTEM STATUS	*
		784	*	\$INDR2 - ERROR PENDING INDICATOR (\$ERPND)	*
		785	*	\$HIST1 - ERROR HISTORY LOG	*
		786	*	\$PLYN - ENTRY TO CRT IOCS	*
		787	*	I\$CALL - VIRTUAL MEMORY PAGING ROUTINE	*
		788	*	* INDICATORS FOR VM ROUTINE	*
		789	*	I\$LDBR	*
		790	*	I\$LOCK	*
		791	*	I\$LDXR	*
		792	*	I\$VADR	*
		793	*	I\$UNLK	*
		794	*	I\$RTRN	*
		795	*	V\$SKEY - VIRTUAL ADDRESS OF DFKEYN	*
		796	*	V\$SPRT - VIRTUAL ADDRESS OF DFPRNT	*
		797	*		*
		798	*	*EXITS, NORMAL	*
		799	*	EXIT IS TO THE CALLING PROGRAM VIA A BRANCH TO THE VIRTUAL MEMORY	*
		800	*	PAGING ROUTINE.	*
		801	*	B I\$RTRN	*
		802	*		*
		803	*	*EXITS, ERROR	*
		804	*	A DATA PARITY ERROR WILL BE RETRIED ONCE. THE SUCCESSIVE PARITY	*
		805	*	ERRORS WILL CAUSE A SYSTEM GENERATED HARD HALT.	*
		806	*		*
		807	*	*TABLES/WORKAREAS	*
		808	*	DEPTBL - KEYBOARD TABLE CONTAINING THE EBCDIC CHARACTER CODES	*
		809	*	ARRANGED SUCH THAT AN INDEX VALUE IS SENSED FROM THE	*
		810	*	KEYBOARD AND USED AS A DISPLACEMENT INTO THE TABLE TO	*
		811	*	FETCH THE PROPER EBCDIC VALUE. THE TABLE IS INITIALIZED	*
		812	*	TO KEYBOARD TYPE KB1, BUT MAY BE CHANGED TO REFLECT	*
		813	*	THE CONFIGURATION RECORD.	*
		814	*		*
		815	*	*ATTRIBUTES	*
		816	*	NATURALLY RELOCATABLE	*
		817	*		*

DFKEYN - VIRTUAL MEMORY KEYBOARD ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 174
		818	*	*CHARACTER CODE DEPENDENCY	*
		819	*	* THE OPERATION OF THIS MODULE DEPENDS UPON AN INTERNAL	*
		820	*	* REPRESENTATION OF THE EXTERNAL CHARACTER SET WHICH IS EQUIVALENT	*
		821	*	* TO THE ONE USED AT ASSEMBLY TIME, THE CODING HAS BEEN ARRANGED	*
		822	*	* SC THAT REDEFINITION OF CHARACTER CONSTANTS, BY REASSEMBLY, WILL	*
		823	*	* RESULT IN A CORRECT MODULE FOR THE NEW DEFINITIONS.	*
		824	*		*
		825	*	*NOTES	*
		826	*	* ERROR PRJCEDURES	*
		827	*	* UPON DETECTION OF A DATA REGISTER PARITY ERROR, THE SYSTEM WILL	*
		828	*	* HALT INDICATING TO THE USER THAT A PARITY ERROR SAS OCCURRED.	*
		829	*	* TO CONTINUE, OR RETRY THE CHARACTER, THE START SWITCH MUST BE	*
		830	*	* PRESSED. THE ERROR IS LOGGED IN THE COUNT LOG ON DISK.	*
		831	*	* IF ANOTHER IS DETECTED, THE HISTORY LOG IS UPDATED AND A HARD	*
		832	*	* HALT EXECUTED.	*
		833	*		*
		834	*	* RESISTER USAGE	*
		835	*	* THE @XR IS USED FOR PASSING THE ADDRESS OF THE INPUT DATA	*
		836	*	* BUFFER.	*
		837	*	* THE @XR IS ALSO USED AS A BASE REGISTER FOR PAGE 3.	*
		838	*	* THE @BR IS USED AS A BASE REGISTER FOR PAGE 2.	*
		839	*	* BOTH PliAR AND IliAR ARE USED FOR BRANCHING BETWEEN PROGRAM	*
		840	*	* AND INTERRUPT LEVEL.	*
		841	*	* THE RESISTERS ARE SAVED AND RESTORED.	*
		842	*		*
		843	*	* SAVED/RESTORED AREAS	*
		844	*	* NONE	*
		845	*		*
		846	*	* MODIFICATION CONSIDERATIONS	*
		847	*	* NONE	*
		848	*		*
		849	*	* REQUIRED MODULES	*
		850	*	* FFPRNT - VIRTUAL MEMORY MATRIX PRINTER IOCR	*
		851	*	* @SYSE0 - GENERAL SYSTEM ELATES	*
		852	*	* @HDWEQ - HARDWARE VALUE EQUATES	*
		853	*	* @FXDEQ - NUCLEUS LOCATION EQUATES	*
		854	*	* @CANEQ - COMMON CORE LOCATION EQUATES	*
		855	*	* @CY0EQ - CYLINDER ZERO EQUATES	*
		856	*	* @HLTEQ - HALT CODE EQUATES	*
		857	*		*
		858	*	* OTHER	*
		859	*	* NONE	*
		860	*		*
		861	*	*****	*

DFKEYN - VIRTUAL MEMORY KEYBOARD ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 175
					863	*****	*****	
					864	* PAGE 1	*	
					865	*****	*****	
	2500				866	ORG	*,256,0	
				2500	867	USING	DFKEYN,@BR	INITIAL BASE REG FOR PAGE 1
				2700	868	USING	DFKBS3,@XR	BASE VALUE FOR PAGE 3
					869	*		
				2500	870	DFKEYN EQU	*	ENTRY TO ENABLE INPUT
	2500	34	01	03C5	871	ST	\$BRSAV,@BR	SAVE PAGE 1 ADDRESS
	2504	74	01	68	872	ST	DFK100+@OP1(,@BR),@BR	SET DFK100 TO
	2507	7C	65	68	873	MVI	DFK100+@OP1(,@BR),DFKDIO	* BRANCH TO ITSELF
	250A	C0	87	1329	874	B	I\$LDBR	LOAD PAGE 2 USING BR
	250E	2600			875	DC	AL2(V\$SKEY+DFKBS2-DFKEYN)	VADDR FOR PAGE 2
	2510	C0	87	1354	876	B	I\$LOCK	LOCK PAGE 2
				2600	877	USING	DFKBS2,@BR	BASE VALUE FOR PAGE 2
	2514	7C	20	BC	878	MVI	DFKP10-1(,@BR),X'20'	INITIALIZE DSPLYN ADDR
	2517	4E	00	BC 043B	879	ALC	DFKP10-1(1,@BR),\$EXFTR	CALCULATE DSPLYN ENTRY ADDRESS
	251C	74	02	28	880	ST	DFKLMG(,@BR),@XR	SAVE INPUT LINE ADDRESS
	251F	74	02	26	881	ST	DFKSTN(,@BR),@XR	SET STARTING DATA ADDRESS
	2522	74	02	2A	882	ST	DFKRMG(,@BR),@XR	SET STARTING ADDR IN RIGHT ADDR
	2525	4C	00	1F 03C0	883	MVC	DFKNPS(1,@BR),\$RMRGN	RIGHT JUSTIFY RIGHT MRGN VALUE
	252A	4F	00	1F 03C1	884	SLC	DFKNPS(1,@BR),\$LMRGN	CALCULATE PRINTER WIDTH
	252F	5E	01	2A 1F	885	ALC	DFKRMG(@CADDR,@BR),DFKNPS(,@BR)	CALC RIGHT MARGIN ADDR
	2533	D2	02	03	886	LA	DFKNTR-DFKBS2(,@BR),@XR	PUT INTERRUPT ADDR IN XR
	2536	74	02	15	887	ST	DFKIAR(,@BR),@XR	SAVE INTERRUPT ADDR FOR LOAD
	2539	7C	00	1F	888	MVI	DFKNPS(,@BR),@ZERO	SET NO LINE POSITION CHANGE
	253C	D2	02	53	889	LA	DFKENT-DFKBS2(,@BR),@XR	LOAD MAINLINE ENTRY ADDR
	253F	74	02	32	890	ST	DFKROS(,@BR),@XR	SAVE MAINLINE ADDR FOR PliAR
	2542	35	02	03C5	891	L	\$BRSAV,@XR	POINT XR TO PAGE 1
	2546	E2	02	65	892	LA	DFK100-DFKEYN(,@XR),@XR	XR = HALT ADDRESS
	2549	74	02	30	893	ST	DFKRET(,@BR),@XR	SAVE MAINLINE RETURN ADDRESS
	254C	E2	02	5B	894	LA	DFKTBL-DFK100(,@XR),@XR	XR = DATA TABLE ADDRESS
	254F	74	02	17	895	ST	DFKBLE(,@BR),@XR	SAVE DATA TABLE ADDRESS
	2552	C0	87	1330	896	B	I\$LDXR	READ IN PAGE 3 USING XR
	2556	2700			897	DC	AL2(V\$SKEY+DFKBS3-DFKEYN)	VADDR FOR PAGE 3
	2558	C0	87	1354	898	B	I\$LOCK	LOCK PAGE 3
	255C	74	02	36	899	ST	DFKXRS(,@BR),@XR	SAVE PAGE 3 ADDRESS
	255F	75	C0	15	900	L	DFKIAR(,@BR),@IliAR	LOAD INTERRUPT ADDRESS
	2562	F3	10	1E	901	SIO	@KENAB,@KEYBD	ENABLE, UNLOCK KEYBOARD
				0065	902	DFKDIO EQU	*-DFKEYN	DISPLACEMENT TO DFK100
	2565	C0	87	0000	903	DFK100 B	*-*	WAIT FOR LINE
	2569	1C	01	144A 3C	904	DFK120 MVC	I\$VADR(@VADDR),DFKPG3(,@BR)	SET PAGE 3 VADDR
	256E	C0	87	1350	905	B	I\$UNLK	UNLOCK PAGE 3
	2572	1C	01	144A 3A	906	MVC	I\$VADR(@VADDR),DFKPG2(,@BR)	SET PAGE 2 VADDR
	2577	C0	87	1350	907	B	I\$UNLK	UNLOCK PAGE 2
	257B	75	02	28	908	L	DFKLMG(,@BR),@XR	RESTORE XR TO DATA ADDRESS
	257E	C0	87	12D3	909	DFK140 B	I\$RTRN	RETURN TO CALLING PGM
					910	*****	*****	

DFKEYN - VIRTUAL MEMORY KEYBOARD ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 176

25C0			912	ORG	DFKEYN+256-64	PLACE DATA TABLE TO END OF PAGE
		25C0	913	DFKTBL EQU	*	FIRST BYTE OF DATA TABLE
25C0	F0	25C0	914	DC	CL1'0'	0
25C1	F1	25C1	915	DC	CL1'1'	1
25C2	F2	25C2	916	DC	CL1'2'	2
25C3	F3	25C3	917	DC	CL1'3'	3
25C4	F4	25C4	918	DC	CL1'4'	4
25C5	F5	25C5	919	DC	CL1'5'	5
25C6	F6	25C6	920	DC	CL1'6'	6
25C7	F7	25C7	921	DC	CL1'7'	7
25C8	F8	25C8	922	DC	CL1'8'	8
25C9	F9	25C9	923	DC	CL1'9'	9
25CA	C1	25CA	924	DC	CL1'A'	A
25CB	C2	25CB	925	DC	CL1'B'	B
25CC	C3	25CC	926	DC	CL1'C'	C
25CD	C4	25CD	927	DC	CL1'D'	D
25CE	C5	25CE	928	DC	CL1'E'	E
25CF	C6	25CF	929	DC	CL1'F'	F
25D0	5D	25D0	930	DC	XL1'5D')
25D1	5A	25D1	931	DC	AL1(@UPARW)	UP ARROW
25D2	7C	25D2	932	DC	XL1'7C'	@
25D3	7B	25D3	933	DC	XL1'7B'	#
25D4	5B	25D4	934	DC	XL1'5B'	\$
25D5	6C	25D5	935	DC	XL1'6C'	%
25D6	4A	25D6	936	DC	XL1'4A'	CENTS SIGN
25D7	50	25D7	937	DC	XL1'50'	&
25D8	7D	25D8	938	DC	XL1'7D'	'
25D9	4D	25D9	939	DC	XL1'4D'	(
25DA	C7	25DA	940	DC	CL1'G'	G
25DB	C8	25DB	941	DC	CL1'H'	H
25DC	C9	25DC	942	DC	CL1'I'	I
25DD	D1	25DD	943	DC	CL1'J'	J
25DE	D2	25DE	944	DC	CL1'K'	K
25DF	D3	25DF	945	DC	CL1'L'	L
25E0	D4	25E0	946	DC	CL1'M'	M
25E1	D5	25E1	947	DC	CL1'N'	N
25E2	D6	25E2	948	DC	CL1'O'	O
25E3	D7	25E3	949	DC	CL1'P'	P
25E4	D8	25E4	950	DC	CL1'Q'	Q
25E5	D9	25E5	951	DC	CL1'R'	R
25E6	E2	25E6	952	DC	CL1'S'	S
25E7	E3	25E7	953	DC	CL1'T'	T
25E8	E4	25E8	954	DC	CL1'U'	U
25E9	E5	25E9	955	DC	CL1'V'	V
25EA	E6	25EA	956	DC	CL1'W'	W
25EB	E7	25EB	957	DC	CL1'X'	X
25EC	E8	25EC	958	DC	CL1'Y'	Y
25ED	E9	25ED	959	DC	CL1'Z'	Z
25EE	60	25EE	960	DC	XL1'60'	-
25EF	7E	25EF	961	DC	XL1'7E'	= (EQUAL SIGN)
25F0	4E	25F0	962	DC	CL1'+'	+ (PLUS)
25F1	4B	25F1	963	DC	CL1'.'	PERIOD
25F2	5E	25F2	964	DC	CL1';'	; (SEMICOLON)
25F3	5C	25F3	965	DC	CL1'*'	*
25F4	6B	25F4	966	DC	CL1','	COMMA
25F5	4B	25F5	967	DC	CL1'.'	PERIOD

DFKEYN - VIRTUAL MEMORY KEYBOARD ROUTINE

ERR LOC		OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 177
	25F6	61	25F6	968	DC	XL1'61'			/
	25F7	6F	25F7	969	DC	XL1'6F'			?
	25F8	4F	25F8	970	DC	XL1'4F'			LOGICAL 'OR'
	25F9	40	25F9	971	DFKLKA	DC CL1' '			BLANK
	25FA	7A	25FA	972	DC	XL1'7A'			COLON
	25FB	7F	25FB	973	DC	XL1'7F'			NOT EQUAL
	25FC	4C	25FC	974	DC	XL1'4C'			LESS NAN
	25FD	6E	25FD	975	DC	XL1'6E'			> (GREATER THAN)
	25FE	6D	25FE	976	DC	XL1'6D'			UNDER SCORE
	25FF	5F	25FF	977	DC	XL1'5F'			LOGICAL 'NOT'
				978	*****				
			0039	979	DFKLNK EQU	DFKLKA-DFKTBL			DISP OF BLANK IN TABLE
				980	*****				

DFKEYN - VIRTUAL MEMORY KEYBOARD ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 178
					982	*****				
					983	* PAGE 2				*
					984	*				*
					985	* ONCE THE KEYBOARD HAS BEEN UNLOCKED, ALL KEYBOARD INTERRUPTS				*
					986	* WILL ENTER AT DFKNTR. THE INTERRUPT WILL BE SERVICED AND THE				*
					987	* LEVEL EXITED.				*
					988	*****				
2600					989	ORG DFKEYN+256	PLACE PAGE 2			
				2600	990	DFKBS2 EQU *	PAGE 2 BASE ADDRESS			
2600	F3	10	19		991	DFK160 SIO DFKEYL,@KEYBD	EXIT LEVEL, LOCK KEYBOARD			
					992	*				
				2603	993	DFKNTR EQU *	INTERRUPT ENTRY UDR			
2603	75	20	32		994	L DFKROS(,@BR),@PLIAR	LOAD PLIAR WITH PROCESSOR ENTRY			
2606	70	10	1D		995	SNS DFKNSK(,@BR),@KEYBD	SENSE KEYBOARD DATA			
2609	5D	01	1D 34		996	CLC DFKNSK(@REGL,@BR),DFKIRK(,@BR) IS IT INQUIRY REQUEST ?				
260D	D0	01	00		997	BNE DFK160(,@BR)	GO EXIT LEVEL IF NOT			
2610	C0	87	0483		998	B \$CIENT	GO CHECK MASK STATUS			
					999	*****				

DFKEYN - VIRTUAL MEMORY KEYBOARD ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 179

```
1001 *****
1002 *   CONSTANTS AND WORK AREAS FOR KEYBOARD IOCR   *
1003 *****
1004 *
2614 0000      2615 1005 DFKIAR DC      AL2(*-*)          INTERRUPT ENTRY ADDR
2616 0000      2617 1006 DFKBLE DC      AL2(*-*)          ADDR OF DATA TABLE
2618 0483      2619 1007 DFKIET DC      AL2($CIENT)        ADDR OF CI ENTRY
261A 10        261A 1008          DC      AL1(@KEYBD)       SIO Q BYTE
261B 1E        261B 1009          DC      AL1(@KENAB)       SID R BYTE - ENABLE KEYBOARD
261C          261C 1010 DFKATA DS      CL1                DATA BYTE
261D          261D 1011 DFKNSK DS      CL1                SENSE BYTE
261E 0000      261F 1012 DFKNPS DC      XL2'000'          LINE POSITION CHANGE
2620 0001      2621 1013 DFKC01 DC      XL2'0001'         CONSTANT 1
2622 00        2622 1014          DC      XL1'00'          INDEX PPL CNT BYTE
                2621 1015 DFKIST EQU    DFKC01            OBR ENTRY
                2623 1016 DFKPPL EQU    *                PRINT PPL
2623 40        2623 1017          DC      XL1'40'          PRINT COMMAND
2624          2624 1018 DFKCNT DS      CL1                PRINT COUNT
2625 0000      2626 1019          DC      AL2(*-*)          INITIAL PRINT DOSTION
                2626 1020 DFKSTN EQU    DFKPPL+@PDATA      ADDR OF CURRENT POS IN LINE BUF
2627 0000      2628 1021 DFKLMG DC      AL2(*-*)          ADDR OF LEFT POS OF LINE BUFFER
2629 0000      262A 1022 DFKRMG DC      AL2(*-*)          ADDR OF RIGHT MARGIN IN LINE
262B          262C 1023 DFKIME DS      CL2                100 MS LOOP CNTR
262D 15B3      262E 1024 DFKMCT DC      IL2'5555'         INITIAL CNT FOR 100 MS
262F          2630 1025 DFKRET DS      CL2                INTERRUPT RETURN ADDR
2631 0000      2632 1026 DFKROS DC      AL2(*-*)          MAINLINE ENTRY ADDRESS
2633 11        2633 1027          DC      AL1(DFKRKY)       I R KEY CODE
2634 10        2634 1028 DFKIRK DC      AL1(@KFUNK)        FUNCTION KEY CODE
2635          2636 1029 DFKXRS DS      CL(@CADDR)          PAGE 3 ADDR SAVE AREA
2637 0004      2638 1030 DFKXDP DC      AL2(DFK120-DFK100) INCREMENT TO JUMP HPL
2639 2600      263A 1031 DFKPG2 DC      AL2(V$SKEY+DFKBS2-DFKEYN) VADDR FOR PAGE 2
263B 2700      263C 1032 DFKPG3 DC      AL2(V$SKEY+DFKBS3-DFKEYN) VADDR FOR PAGE 3
1033 *****
```

DFKEYN - VIRTUAL MEMORY KEYBOARD ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 180
				1035		*****				
				1036	*	EQUATES USED FOR KEYBOARD IOCR				*
				1037		*****				
				1038	*					
	0001			1039	DFK001	EQU	1			ONE
	0005			1040	DFKTAB	EQU	X'05'			TAB KEY
	0016			1041	DFKBSP	EQU	X'16'			BACKSPACE KEY
	0015			1042	DFKRTN	EQU	X'15'			RETURN KEY
	0003			1043	DFKERS	EQU	X'03'			ERASE KEY
	0040			1044	DFKSPC	EQU	X'40'			SPACE BAR
	0011			1045	DFKRKY	EQU	X'11'			IQUIRY REQUEST KEY
	0002			1046	DFKEMS	EQU	X'02'			ENTER MINUS KEY
	0010			1047	DFKACK	EQU	X'10'			BACK SPACE CTRL
	0011			1048	DFKKIX	EQU	X'11'			BACKSPACE &INDX CTRL
	001D			1049	DFKEUD	EQU	X'1D'			EXIT, UNLOCK, DISABLE CTRL
	0018			1050	DFKLOK	EQU	X'18'			LOCK KEYBOARD CTRL
	0012			1051	DFKENB	EQU	X'12'			ENABLE INTERRUPTS CTRL
	001C			1052	DFKULK	EQU	X'1C'			UNLOCK KEYBOARD CTRL
	0019			1053	DFKEXL	EQU	X'19'			EXIT LEVEL, LOCK KEYBOARD CTRL
	0040			1054	DFKDTK	EQU	X'40'			DATA KEY FUNCTION BIT
				1055		*****				

DFKEYN - VIRTUAL MEMORY KEYBOARD ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 181

```
1057 *****
263D 75 C0 19      1058 DFK180 L    DFKIET(,@BR),@I1IAR    RESTORE INTERRUPT ADDR TO NUC
2640 F3 10 18      1059      SIO    DFKLOK,@KEYBD        LOCK KEYBOARD
2643 5E 01 30 38    1060      ALC    DFKRET(@CADDR,@BR),DFKXDP(,@BR)  DON'T DO HALT
2647 F2 87 03      1061      J      DFKNAB            DON'T UNLOCK KEYBOARD
264A      1062 DFKXIT EQU    *                ENTRY TO EXIT DEPRES
264A F3 10 1C      1063      SIO    DFKULK,@KEYBD        UNLOCK KEYBOARD
264D      1064 DFKNAB EQU    *                ENTRY TO ENABLE
264D F3 10 12      1065      SIO    DFKENB,@KEYBD        ENABLE INTERRUPTS
2650 75 20 30      1066      L      DFKRET(,@BR),@P1IAR    RETURN TO INTERRUPTED PROGRAM
2653      1067 *
2653      1068 DFKENT EQU    *                ENTRY TO PROCESS INTERRUPT DATA
2653 D0 FF 96      1069      BC     DFKDLP(,@BR),X'FF'        UPDATE LINE POSITION
2656 78 80 1D      1070      TBN    DFKNSK(,@BR),@PRITY    TEST FOR PARITY ERROR
2659 E0 10 BB      1071      BT     DFKROR(,@XR)        JUMP IF PARITY ERROR
265C BC 87 BC      1072      MVI    DFK520+@Q(,@XR),@UCB    SET PARITY INDR OFF
265F 78 10 1D      1073      TBN    DFKNSK(,@BR),@KFUNK    FUNCTION KEY ?
2662 E0 10 00      1074      BT     DFK350(,@XR)        JUMP IF YES
2665 78 40 1D      1075      TBN    DFKNSK(,@BR),DFKDTK    DATA KEY ?
2668 D0 90 4A      1076      BF     DFKXIT(,@BR)        NO -- GO EXIT
266B D0 87 DD      1077      B      DFKTST(,@BR)        GO CHK CMND KEY ONLY, RI MRGN
266E BC 80 51      1078      MVI    DFK380+@Q-DFKBS3(,@XR),@NOP    SET BACKSPACE INDEX OFF
2671 5C 00 7C 1C    1079 DFK200 MVC    DFK220+@OPD2(1,@BR),DFKATA(,@BR)  SET DATA TBL DISP
2675 75 02 17      1080      L      DFKBLE(,@BR),@XR        *** LOAD XR WITH TABLE ADDR
2678 2C 00 0000 00  1081 DFK220 MVC    *-*(1),*-(,@XR)        MOVE DATA CHAR TO LINE BUFFER
267D D0 87 83      1082      B      DFKRT1(,@BR)        PRINT AND UPDATE POSITION
2680 D0 87 4A      1083      B      DFKXIT(,@BR)        GO EXIT
1084 *****
```

DFKEYN - VIRTUAL MEMORY KEYBOARD ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 182

```
1086 *****
1087 * THIS ROUTINE UPDATES ALL LINE BUFFER ADDRESSES IN DFKEYN BY THE *
1088 * VALUE PLACED IN 'DFKNPS'. IT CHECKS FOR MARGIN REQUIREMENTS. IF *
1089 * THE RIGHT MARGIN IS HIT, A CARRIAGE RETURN AND EOS ARE GENERATED. *
1090 * IF LEFT MARGIN IS HIT, NOTHING IS UPDATED. *
1091 * TWO ENTRY POINTS ARE PROVIDED: *
1092 * B DFKRT1(,@BR) PRINTS 1 CHAR AND UPDATES POSITION *
1093 * B DFKDLP(,@BR) UPDATES POSITION AND TEST RT MARGIN *
1094 *****
2683 1095 DFKRT1 EQU *
2683 7C 01 1F 1096 MVI DFKNPS(,@BR),DFK001 SET CHARACTER COUNT TO 1
2686 74 08 AB 1097 ST DFK260+@OP1(,@BR),@ARR SAVE RETURN ADDRESS
2689 5C 00 24 1F 1098 MVC DFKCNT(1,@BR),DFKNPS(,@BR) SET PRINT COUNT
268D D2 02 23 1099 LA DFKPPL(,@BR),@XR XR = PPL ADDRESS
2690 D0 87 AC 1100 B DFKPRT(,@BR) GO PRINT CHARACTER ON SYS PRINT
2693 F2 87 03 1101 J DFK240 GO UPDATE POSITION
1102 *
2696 1103 DFKDLP EQU * ENTRY TO UPDATE POSITION
2696 74 08 AB 1104 ST DFK260+@OP1(,@BR),@ARR SAVE RETURN ADDRESS
2699 5E 01 26 1F 1105 DFK240 ALC DFKPPL+@PDATA(@CADDR,@BR),DFKNPS(,@BR) UPDATE DATA ADDR
269D 5C 01 7B 26 1106 MVC DFK220+@OP1(@CADDR,@BR),DFKSTN(,@BR) UPDATE POS ADDR
26A1 9C 01 81 26 1107 MVC DFK480-DFKBS3+@OP1(@CADDR,@XR),DFKSTN(,@BR)
26A5 7C 00 1F 1108 MVI DFKNPS(,@BR),@ZERO ZERO LINE POSITION INCREMENT
26A8 C0 87 0000 1109 DFK260 B *-* RETURN
1110 *****
```

DFKEYN - VIRTUAL MEMORY KEYBOARD ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 183

```
1112 *****
1113 * THIS ROUTINE DETERMINES WHICH DEVICE(S) IS TO BE USED FOR OUTPUT. *
1114 * IT THEN CALL THE CORRECT IOCS. INPUT IS THE ADDRESS OF THE PPL *
1115 * STORED IN XR. UPON EXIT XR IS RESTORED TO PAGE 3 BASE ADDRESS. *
1116 *****
26AC 1117 DFKPRT EQU * ENTRY TO INTERFACE
26AC 74 08 DC 1118 ST DFK320+@OP1(,@BR),@ARR SAVE RETURN ADDRESS
26AF 74 02 BF 1119 ST DFKP20(,@BR),@XR SET PPL ADDRESS FOR DSPLYN
26B2 1D 00 044A BC 1120 CLC $PRDEV-1(1),DFKP10-1(,@BR) TEST FOR CRT USE
26B7 F2 01 0E 1121 JNE DFK280 SKIP CRT IF NOT IN USE
26BA C0 87 2004 1122 B $$PLYN GO TO CRT IOCS
26BD 1123 DFKP10 EQU *-1 ADDR OF DSPLYN ENTRY
26BE 0000 26BF 1124 DFKP20 DC AL2(*-*) PPL ADDRESS
26C0 1D 01 044B BD 1125 CLC $PRDEV(@CADDR),DFKP10(,@BR) IS PRINTER USED TOO ?
26C5 F2 81 0E 1126 JE DFK300 SKIP PRINTER OP IF NOT
26C8 3A 1E 03E4 1127 DFK280 SBN $LPRP3,@KENAB FORCE MATRIX PRINT MODE 1-3
26CC C0 87 12B1 1128 B I$CALL GO TO DFPRNT
26D0 2800 26D1 1129 DC AL2(V$SPRT) VADDR OF DFPRNT
26D2 3B 1E 03E4 1130 SBF $LPRP3,@KENAB RESET MATRIX PTR. FLAGS 1-3
26D6 75 02 36 1131 DFK300 L DFKXRS(,@BR),@XR RESTORE PAGE 3 ADDRESS
26D9 C0 87 0000 1132 DFK320 B *-* RETURN TO CALLING ROUTINE
1133 *****
```

DFKEYN - VIRTUAL MEMORY KEYBOARD ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 184
					1135	*****	*****			
				26DD	1136	DFKTST EQU *	ENTRY TO TEST RIGHT MARGIN			
26DD	74	08	ED		1137	ST DFK340+@OP1(,@BR),@ARR	SAVE RETURN ADDRESS			
26E0	5D	01	26 2A		1138	CLC DFKPPL+@PDATA(@CADDR,@BR),DFKRMG(,@BR)	AT RIGHT MARGIN ?			
26E4	E0	02	72		1139	BNL DFK440(,@XR)	DO CARRIER RETURN IF YES			
26E7	F3	10	1C		1140	SIO DFKULK,@KEYBD	UNLOCK KEYBOARD			
26EA	C0	87	0000		1141	DFK340 B *-*	RETURN TO CALLING ROUTINE			
					1142	*****	*****			

DFKEYN - VIRTUAL MEMORY KEYBOARD ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 185
				1144		*****		
				1145	*	PAGE 3		*
				1146	*			*
				1147	*	THIS ROUTINE CHOOSES THE DESIRED ROUTINE PER REQUEST.		*
				1148		*****		
2700				1149	ORG	DFKBS2+256	PLACE PAGE 3	
				2700 1150	DFKBS3 EQU	*	BASE ADDRESS FOR PAGE 3	
				2700 1151	DFK350 EQU	*	ENTRY FOR FNCT KEY PROCESSING	
2700	7D	11	1C	1152	CLI	DFKNSK-1(,@BR),DFKRKY	INQUIRY REQUEST ?	
2703	D0	81	3D	1153	BE	DFK180(,@BR)	GO EXIT	
2706	7D	16	1C	1154	CLI	DFKNSK-1(,@BR),DFKBSP	BACKSPACE KEY ?	
2709	F2	81	41	1155	JE	DFKSPB	JUMP YES	
270C	7D	15	1C	1156	CLI	DFKNSK-1(,@BR),DFKRTN	RETURN KEY ?	
270F	F2	81	66	1157	JE	DFK460	JUMP YES	
2712	7D	03	1C	1158	CLI	DFKNSK-1(,@BR),DFKERS	ERASE KEY ?	
2715	F2	81	71	1159	JE	DFKERA	JUMP YES	
2718	D0	87	DD	1160	B	DFKTST(,@BR)	CHECK FOR RIGHT MARGIN	
271B	7D	40	1C	1161	CLI	DFKNSK-1(,@BR),DFKSPC	SPACE BAR ?	
271E	F2	81	7C	1162	JE	DFKSPA	JUMP YES	
2721	7D	02	1C	1163	CLI	DFKNSK-1(,@BR),DFKEMS	ENTER MINUS KEY ?	
2724	F2	81	8B	1164	JE	DFK500	DO FORMS INDEV IF YES	
2727	7D	05	1C	1165	CLI	DFKNSK-1(,@BR),DFKTAB	TAB KEY ?	
272A	D0	01	4A	1166	BNE	DFKXIT(,@BR)	EXIT IF NO	
				1167	*	CONTINUE		

DFKEYN - VIRTUAL MEMORY KEYBOARD ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 186
				1169		*****		
				1170	*		ENTRY FOR TAB OPERATIONS	
272D	BC	80	51	1171		MVI DFK380+@Q(,@XR),@NOP	SET BACK SPACE INDR OFF	
2730	D0	87	83	1172		B DFKRT1(,@BR)	GO PRINT ONE CHARACTER	
				1173	*		CONTINUE TO TEST TYPO.	
				1174		*****		
				2733	1175	DFKATC EQU *	ENTRY TO TEST TYPAMATIC	
2733	79	02	1D	1176		TBF DFKNSK(,@BR),@TYPAM	TYPAMATIC MODE ?	
2736	D0	10	4A	1177		BT DFKXIT(,@BR)	EXIT IF NO	
2739	F3	10	18	1178		SIO DFKLOK,@KEYBD	RESET BAIL FOR TYPO	
273C	5C	01	2C 2E	1179		MVC DFKIME(2,@BR),DFKMCT(,@BR)	INITIALIZE TIMING LOOP	
2740	5F	01	2C 21	1180	DFK360	SLC DFKIME(2,@BR),DFKC01(,@BR)	DECREMENT COUNTER	
2744	E0	84	40	1181		BH DFK360(,@XR)	LOOP FOR 100 MS	
2747	70	10	1D	1182		SNS DFKNSK(,@BR),@KEYBD	SENSE DATA	
274A	E0	87	00	1183		B DFK350(,@XR)	RETURN FOR CONTINUED TYPO	
				1184		*****		

DFKEYN - MATRIX PRINTER ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 187

```
1186 *****
274D 1187 DFKSPB EQU * ENTRY TO HANDLE BACKSPACE
1188 MVI DFKPL1+@PCTRL(,@XR),DFKACK SET BACKSPACE CTRL
2750 F2 80 06 1189 DFK380 JC DFK400,@NOP JUMP IF NOT FIRST BACKSPACE
2753 BC 11 70 1190 MVI DFKPL1+@PCTRL(,@XR),DFKKIX SET BACKSPACE ANC INDE
2756 BC 87 51 1191 MVI DFK380+@Q(,@XR),@UCB SET INDEX INDR OFF
2759 5D 01 26 28 1192 DFK400 CLC DFKSTN(@CADDR,@BR),DFKLMG(,@BR) TEST LEFT MARGIN
275D F2 81 0D 1193 JE DFK420 JUMP TO NOT BACKSPACE
2760 E2 02 70 1194 LA DFKPL1(,@XR),@XR XR = PPL ADDRESS
2763 D0 87 AC 1195 B DFKPRT(,@BR) GO DO BACKSPACE
2766 5F 01 26 21 1196 SLC DFKSTN(@CADDR,@BR),DFKC01(,@BR) SET NEW POSITION
276A D0 87 96 1197 B DFKDLP(,@BR) GO UPDATE LINE POSITION
276D E0 87 33 1198 DFK420 B DFKATC(,@XR) GO TEST TYPAMATIC
1199 *****
2770 1200 DFKPL1 EQU * 1ST BYTE OF BACKSPACE PPL
2770 1201 DS CL1 CONTROL BYTE
2771 00 2771 1202 DC XL1'00' COUNT BYTE
1203 *****
```


DFKEYN - MATRIX PRINTER ROUTINE

```
ERR LOC  OBJECT CODE      ADDR STMT SOURCE STATEMENT      VER 15, MOD 00  31/05/21  PAGE 188

          1205 *****
2772 78 02 1D          1206 DFK440 TBN   DFKNSK( ,@BR) ,@TYPAM      TYPO BIT ON
2775 E0 10 33          1207          BT    DFKATC( ,@XR)          YES, GO SENSE AGAIN
2778 E2 02 A3          1208 DFK460 LA    DFKPL2( ,@XR) ,@XR        XR = PPL ADDRESS
277B D0 87 AC          1209          B     DFKPRT( ,@BR)          RETURN CARRIAGE
277E 3C 1E 0000        1210 DFK480 MVI   *-*,@EOS              MOVE EOS TO CURRENT LOCATION
2782 5C 01 26 28        1211          MVC   DFKSTN-DFKBS2(@CADDR,@BR),DFKLMG( ,@BR)  SET NEW POSITION
2786 D0 87 3D          1212          B     DFK180( ,@BR)          GO EXIT LEVEL - LOCK KEYBOARD
          1213 *****
          2789 1214 DFKERA EQU    *              ENTRY FOR ERASE DEY
2789 B4 02 A8          1215          ST    DFKPL3+@PDATA( ,@XR) ,@XR  SET PAGE ADDR IN PPL
278C AE 01 A8 B1        1216          ALC   DFKPL3+@PDATA(@CADDR,@XR),DFKMSD( ,@XR)  CALC DATA ADDR
2790 E2 02 A5          1217          LA    DFKPL3( ,@XR) ,@XR        XR = PPL ADDRESS
2793 D0 87 AC          1218          B     DFKPRT( ,@BR)          PRINT ERASED MESSAGE & RETURN
2796 5C 01 26 28        1219          MVC   DFKSTN-DFKBS2(@CADDR,@BR),DFKLMG( ,@BR)  SET NEW POSITION
279A D0 87 4A          1220          B     DFKXIT( ,@BR)          GO EXIT LEVEL
          1221 *****
          279D 1222 DFKSPA EQU    *              ENTRY FOR SPACE BAR KEY
279D 7C 39 1C          1223          MVI   DFKATA-DFKBS2( ,@BR),DFKLNK  MOVE IN DISP OF BLANK
27A0 D0 87 71          1224          B     DFK200( ,@BR)          BRANCH TO HANDLE DATA KEYS
          1225 *****
          27A3 1226 DFKPL2 EQU    *              ADDR OF RETURN PPL
27A3 8080          27A4 1227          DC    XL2'8080'          RETURN CARRIAGE PPL
          27A5 1228 DFKPL3 EQU    *              FIRST BYTE 'ERASE' PPL
27A5 C0          27A5 1229          DC    XL1'C0'            PRINT & RETURN CTRL
27A6 07          27A6 1230          DC    AL1(DFKSGL)          COUNT BYTE
27A7 0000          27A8 1231          DC    AL2(*-*)          ADDR OF MESSAGE 'ERASE'
          27A9 1232 DFKSG1 EQU    *              START OF MESSAGE
27A9 40C5D9C1E2C5C4    27AF 1233          DC    CL7' ERASED'          MESSAGE
          0007 1234 DFKSGL EQU    *-DFKSG1          LENGTH OF MESSAGE
27B0 00A9          27B1 1235 DFKMSD DC    AL2(DFKSG1-DFKBS3)      DISP TO ERASE MESSAGE
          1236 *****
27B2 D2 02 21          1237 DFK500 LA    DFKC01( ,@BR) ,@XR        POINT XR TO INDEX PPL
27B5 D0 87 AC          1238          B     DFKPRT( ,@BR)          INDEX A LINE
27B8 D0 87 4A          1239          B     DFKXIT( ,@BR)          GO EXIT
          1240 *****
```

DFKEYN - ERP SECTION

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 189
					1242		*****			
				27BB	1243	DFKROR EQU	*			ENTRY TO ERP
27BB	F2	87	07		1244	DFK520 JC	DFK540,@UCB			JUMP IF 1ST ERROR
27BE	3A	20	03D2		1245	SBN	\$IOIND,\$HRDER			SET HARD ERROR INDR
27C2	E0	87	7E		1246	B	DFK480(,@XR)			GO EXIT - HARD ERROR
					1247	*				
27C5	1C	07	0435 21		1248	DFK540 MVC	\$HIST1(#HISLN),DFKIST(,@BR)			SET UP HISTORY ENTRY
27CA	BC	80	BC		1249	MVI	DFK520+@Q(,@XR),@NOP			SET PARITY INDR
27CD	F0	00	00		1250	HPL	*-*,*-*			WAIT ON FIRST ERROR
27CE					1251	ORG	*-2			PLACE ERROR CODE
27CE	2040			27CF	1252	DC	AL2(@HKBER)			WAIT CODE
27D0	3A	04	03D5		1253	SBN	\$INDR2,\$ERPND			SET ERROR PENDING INDR
27D4	D0	87	4A		1254	B	DFKXIT(,@BR)			GO RETRY CHARACTER
					1255		*****			

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 190

```

1257 *****
1258 * THIS IOCR IS USED FOR ALL MATRIX PRINTER FUNCTIONS. *
1259 * IT IS ALSO USED BY DLFPRT 'LINE PRINTER ROUTINE' FOR IOCR OPERATION *
1260 * AVAILABLE FUNCTIONS INCLUDE... *
1261 * PRINT ONLY *
1262 * PRINT AND RETURN CARRIAGE *
1263 * RETURN CARRIAGE ONLY *
1264 * BACKSPACE *
1265 * INDEX AND BACKSPACE *
1266 * CHANGES TO DFPRNT MAY DIRECTLY AFFECT IT'S INTERFACE WITH DLFPRT *
1267 *****
2800 1268 ORG *,256,0
2800 1269 USING DFPA SE,@BR SET BASE REG
2800 1270 DFP100 EQU * ENTRY TO PRINTER IOCR
2800 1C 01 144A FD 1271 MVC I$VADR,DFPPCH(@CADDR,@BR) VM PATCH PAGE ENTRY ADDR 1-5
2805 C0 87 1358 1272 DFP100 B I$CVAD LOAD PATCH PAGE 1-5
2809 4C 01 11 144C 1273 MVC DFP101+@OP1(@CADDR,@BR),I$CADR MOVE CADDR TO BRANCH 1-5
280E C0 87 0000 1274 DFP101 B *-* BRANCH TO PATCH PAGE 1-5
1275 * 1-5
2812 1C 01 144A FD 1276 DFP102 MVC I$VADR,DFPPCH(@CADDR,@BR) VM PATCH PAGE ENTRY ADDR 1-5
2817 3C 39 144A 1277 MVI I$VADR,DFPX39 ADD DISP X'39' 1-5
281B D0 87 05 1278 B DFP100(,@BR) BRANCH TO LOAD PAGE 1-5
281E 4D00 281F 1279 DFP105 DC AL(@VADDR)(V$LPRT) LINE PRINTER PAGE
2820 E0 87 00 1280 B 0(,@XR) BRANCH TO LINE PRINTER ROUTINE
2823 1281 DFP115 EQU * MATRIX PRINTER ROUTINE
2823 F1 E2 00 1282 APL @PBUSY WAIT FOR PRINTER NOT BUSY 1-4
2826 78 40 F5 1283 TBN DFPIST+@PCTRL(,@BR),@PRINT DOE THIS OP PRINT
2829 F2 10 11 1284 JT DFP120 JUMP IF YES
282C 7C 00 F6 1285 MVI DFPIST+@PRCNT(,@BR),@ZERO SET PPL CNTR BYTE TO ZERO
282F 78 10 DE 1286 TBN DFPPCF+@PCTRL(,@BR),@TBLEF TAB LEFT OPERATION ?
2832 F2 90 3D 1287 JF DFP180 GO DO OP IF NOT
2835 1F 00 03C2 E7 1288 SLC $PRPOS(1),DFP001(,@BR) SET NEW CURRENT POSITION
283A F2 87 55 1289 J DFP240 GO DO OP
1290 *
1291 * PRINTING IS REQUIRED - SET UP PRINT PCF
1292 *
283D 71 E4 F8 1293 DFP120 LIO DFPIST+@PDATA(,@BR),@PDAR LOAD DATA LSR WITH DATA ADDR
2840 4E 00 F6 03C2 1294 ALC DFPIST+@PRCNT(1,@BR),$PRPOS ADD CURRENT POSITION
2845 4F 00 F6 03C0 1295 SLC DFPIST+@PRCNT(1,@BR),$RMGRN SUBTRACT RIGHT MARGIN VALUE
284A F2 84 06 1296 JH DFP140 JUMP IF RIGHT MARGIN HIT
284D 7C 00 F6 1297 MVI DFPIST+@PRCNT(,@BR),@ZERO SET COUNT BYTE TO ZERO
2850 F2 87 0F 1298 J DFP160 GO SET NEW PRINT POSITION
2853 5F 00 DF F6 1299 DFP140 SLC DFPPCF+@PRCNT(1,@BR),DFPIST+@PRCNT(,@BR) SET CNT TO HIT
1300 * * MARGIN
2857 7A 80 DE 1301 SBN DFPPCF+@PCTRL(,@BR),@RETRN SET CARRIAGE TO RETURN
285A 5C 00 E5 DF 1302 MVC DFPOK(1,@BR),DFPPCF+@PRCNT(,@BR) RIGHT JUSTIFY CNT
285E 5E 01 F8 E5 1303 ALC DFPIST+@PDATA(@CADDR,@BR),DFPOK(,@BR) ADD CNT TO DATA
1304 * * ADDRESS IN LIST
2862 1E 00 03C2 DF 1305 DFP160 ALC $PRPOS(1),DFPPCF+@PRCNT(,@BR) UPDATE HEAD POSITION
2867 5F 00 DF E7 1306 SLC DFPPCF+@PRCNT(1,@BR),DFP001(,@BR) SET PCF CNT = CNT-1...
1307 * * THIS IS HARDWARE REQUIREMENT
286B F2 02 04 1308 JNL DFP180 JUMP IF SOMETHING TO PRINT
286E 5C 01 DF E9 1309 MVC DFPPCF+@PRCNT(2,@BR),DFPETN(,@BR) SET CARRIER RTRN ONLY
2872 78 80 DE 1310 DFP180 TBN DFPPCF+@PCTRL(,@BR),@RETRN OP FOR CARRIAGE RETURN
2875 F2 90 1A 1311 JF DFP240 JUMP IF NO
2878 4C 00 E1 03C2 1312 DFP200 MVC DFPPCF+@RTCNT(1,@BR),$PRPOS SET CURRENT POS IN

```

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 191
				1313	*		* CARRIAGE RETURN CNT			
287D	4F	00	E1 03C1	1314	SLC	DFPPCF+@RTCNT(1,@BR), \$LMRGN	SUBTRACT LEFT MARGIN VALUE			
2882	F2	84	03	1315	JH	DFP220	JUMP IF NO			
2885	7C	01	DE	1316	MVI	DFPPCF+@PCTRL(,@BR),@INDEX	SET OP TO INDEY ONLY			
2888	0C	00	03C2 03C1	1317	DFP220 MVC	\$PRPOS(1), \$LMRGN	SET CURRENT POS TO LEFT MARGIN			
288E	5F	00	E1 E7	1318	SLC	DFPPCF+@RTCNT(1,@BR), DFP001(,@BR)	SET HARDWARE COUNT			
2892	74	01	DD	1319	DFP240 ST	DFPAPC(,@BR),@BR	SET PAGE ADDR IN PCF ADDR BYTE			
2895	5E	01	DD EB	1320	ALC	DFPAPC(@CADDR,@BR), DFPCFD(,@BR)	ADD DISP TO GET TRUE ADDR			
				2899 1321	DFP250 EQU	*	LINE PRINTER I/O ENTRY	1-4		
2899	71	E6	DD	1322	LIO	DFPAPC(,@BR),@PCAR	LOAD CONTROL LSR WITH NORMAL PCF			
289C	F3	E0	00	1323	DFP260 SIO	@PSIOR,@PSIOQ	START THE PRINT OPERATION			
289F	E0	00	B3	1324	DFP270 BC	RETURN-DLFPRT(,@XR), *-*	RETURN TO LINE PRINTER RTN.	1-4		
28A0				1325	ORG	DFP270+@Q	* INITIALIZE	1-4		
28A0	80			28A0 1326	DC	AL1(@NOP)	* TO NOT BRANCH	1-4		
28A2				1327	ORG	DFP270+@INST3	* TO LINE PRINTER RTN.	1-4		
28A2	F2	80	07	1328	DFP280 JC	DFP320,@NOP	JUMP TO ERP IF ERP IN PROCESS			
				1329	*					
				1330	*****					

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 192
					1332	*****	*****			
					1333	*	THIS ROUTINE WAITS FOR THE OPERATION TO COMPLETE AND CHECKS			*
					1334	*	FOR ERRORS. FORMS CHECKS WILL CAUSE A SOFT HALT.			*
					1335	*	UNIT CHECKS WILL CAUSE ENTRY TO THE ERP.			*
					1336	*****	*****			
				28A5	1337	DFPRCK EQU	*			ENTRY TO CHECK FOR ERRORS
28A5	5C	01	ED EE		1338	MVC	DFPRCT(DFPRCL,@BR),DFPERC(@BR)			INITILIZE RETRY COUNTER
28A9	7C	87	A3		1339	MVI	DFP280+@Q-DFPASE(@BR),@UCB			SET ERP IN PROCESS INDR
28AC	F1	E2	00		1340	DFP320 APL	@PBUSY			WAIT FOR NOT BUSY
28AF	7C	00	9E		1341	MVI	DFP260+2(@BR),@ZERO			SET MATRIX PRINT
28B2	D1	E1	CD		1342	DFP340 TIO	DFP360(@BR),@PFORM			TEST FOR END OF FORMS
28B5	71	E2	E3		1343	LIO	DFPOFF(@BR),@PLITE			TURN END OF FORMS LAMP OFF
28B8	D1	E0	00		1344	DFP335 TIO	*-*(@BR),@PERR			BRANCH TO ERP IF UNIT CHECK 1-4
28B8					1345	ORG	DFP335			* INITIALIZE DFP335 1-4
28B8	E1	E0	CD		1346	TIO	DLFRPE-DLFPRT(@XR),@PERR			* TO BRANCH TO 1-4
28B8					1347	ORG	DFP335			* DFP335 1-4
28B8	D1	E0	D3		1348	TIO	DFPRPE(@BR),@PERR			* ENTRY TO LOAD ERP SECTION 1-4
				28BA	1349	DFP333 EQU	*-1			LAST BYTE OF TIO INST. 1-4
28BB	E0	00	00		1350	DFP330 BC	*-*(@XR),*-*			BRANCH TO LINE PRINTER RTN. 1-4
28BC					1351	ORG	DFP330+@Q			* INITIALIZE 1-4
28BC	80			28BC	1352	DC	AL1(@NOP)			* TO NOT BRANCH 1-4
28BD					1353	ORG	DFP330+@D1			* INITIALIZE FOR 1-4
28BD	25			28BD	1354	DC	AL1(DLF100-DLFPRT)			* RETURN TO DLFPRT ENTRY 1-4
28BE					1355	ORG	DFP330+@INST3			* TO LINE PRINTER ROUTINE 1-4
28BE	1C	01	144A FD		1356	MVC	I\$VADR,DFPPCH(@VADDR,@BR)			VM PATCH PAGE 1-5
28C3	3C	00	144A		1357	MVI	I\$VADR,@ZERO			SET DISP = 0 1-5
28C7	D0	87	05		1358	B	DFP100(@BR)			BRANCH TO LOAD PAGE 1-5
28CA	D0	87	12		1359	DFP300 B	DFP102(@BR)			BRANCH TO LOAD PATCH PAGE 1-5
					1360	*				
					1361	*****	*****			
					1362	*				
28CD	71	E2	E7		1363	DFP360 LIO	DFPITE(@BR),@PLITE			TURN ON FORMS INDR LAMP
28D0	D0	87	B2		1364	B	DFP340(@BR)			GO TEST FORMS AGAIN
					1365	*				
				28D3	1366	DFPRPE EQU	*			ENTRY TO LOAD ERP SECTUIN
28D3	C0	87	1330		1367	B	I\$LDXR			LOAD ERP PAGE USING XR
28D7	2900			28D8	1368	DC	AL2(V\$SPRT+DFPNDX-DFPRNT)			PRINTER ERROR IOCR VADDR
28D9	E0	87	00		1369	B	0(@XR)			EXECUTE ERP
					1370	*****	*****			

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 193

```

1372 *****
1373 * CONSTANT AND EQUATE AREA FOR DPRINT *
1374 *****
2800 1375 DFPASE EQU DFPRNT BASE VALUE FOR CALL SECTION
0002 1376 DFPRCL EQU 2 NUMBER OF RETRY COUNTERS
28DC 28DD 1377 DFPAPC DS CL(@CADDR) ADDRESS OF NRML PCF
28DE 28DE 1378 DFPPCF EQU * LEFT BYTE OF PCF
28DE 28DF 1379 DS CL2 CTRL AND CNT BYTES
28E0 11 28E0 1380 DC XL1'11' RETURN CARRIAGE INDEX CMND
28E1 28E2 1381 DS CL2 COUNT & INDEX
28E2 1382 DFPPCO EQU *-1 LAST BYTE OF CCF
28E3 00 28E3 1383 DFPOFF DC XL1'00' TURN OFF INDR LAMP CTRL
28E4 0000 28E5 1384 DFPORK DC XL2'0000' WORK AREA
28E6 0001 28E7 1385 DFP001 DC XL2'0001' CONSTANT OF ONE
28E8 8080 28E9 1386 DFPETN DC 2AL1(@RETRN) CARRIER RETURN CTRL
28EA 00DE 28EB 1387 DFPCFD DC AL2(DFPPCF-DFPASE) DISPLACEMENT OF PCF IN PAGE
1388 *
28EC 28ED 1389 DFPRCT DS CL(DFPRCL) ERROR COUNT
28EE 03 28EE 1390 DFPERC DC XL1'03' RETRY COUNT
28EF 00F9 28F0 1391 DFPYCD DC AL2(DFPSYC-DFPASE) DISPLACEMENT OF SYNC PCF IN PAGE
28F1 00000000 28F4 1392 DFPDSV DC XL4'00' SAVE AREA FOR CNT AND DATA ADDR
28F5 28F5 1393 DFPIST EQU *
28F5 28F8 1394 DS CL4 PRINT PARAMETER LIST (PPL)
28F5 1395 ORG DFPIST RESET INSTR CNTR
28F5 00000000 28F8 1396 DC XL4'00' SET INITIAL LIST TO ZERO
28F9 0520 28F9 1397 DFPSYC EQU * LEFT BYTE OF SYNC CHECK PCF
28FB 28FA 1398 DC XL2'0520' RETURN AND INDEX, TAB RIGHT
28FC 5309 28FB 1399 DS CL1
28FD 1400 DFPPCH DC AL2(V$PCH2+DFP100-DFPASE+@DOP2) PATCH PAGE 2 1-5
0039 1401 DFPX39 EQU X'39' DISP = X'39' 1-5
28E7 1402 DFPITE EQU DFP001 FORMS INDR LIGHT CTRL
0001 1403 DFPYCT EQU 1 DISPLACEMENT CYNCK CNTR
1404 *
1405 * THE FOLLOWING EQUATES ARE FOR THE LINE PRINTER MODULE (DLFPRT)
1406 *
28F5 1407 DLFIST EQU DFPIST
28E5 1408 DLFORK EQU DFPORK
28F4 1409 DLFDSV EQU DFPDSV
28E7 1410 DLF001 EQU DFP001
28DE 1411 DLFPCF EQU DFPPCF

```


DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 194

```

1413 *****
1414 * THIS ROUTINE DETERMINES THE ERROR AND BRANCHES TO THE PROPER ERP *
1415 *****
2900 1416 ORG *,256,0
2800 1417 USING DFPASE,@BR SET BASE REGS
2900 1418 USING DFPNDX,@XR
2900 1419 DFPNDX EQU * ENTRY TO ERP SECTION
2900 38 01 03E4 1420 TBN $LPRP3,@INDEX TEST DUMMY PRINT POS. USED 1-3
2904 F2 90 0A 1421 JF DFP378 JUMP NO
2907 0C 00 03C2 03E5 1422 MVC $PRPOS(1),$LPROS RESTORE CORRECT POSITION
290D 3B 01 03E4 1423 SBF $LPRP3,@INDEX RESET DUMMY POS. FLAG 1-3
2911 1424 DFP378 EQU * ENTRY SENSE ERROR
2911 B0 E2 D9 1425 SNS DFPRSN(,@XR),@PSNSQ SENSE ERROR BYTES
2914 38 04 03D5 1426 TBN $INDR2,$ERPND HAS LOG ENTRY BEEN SET UP
2918 F2 10 0C 1427 JT DFP380 JUMP IF YES
291B 2C 07 0435 DD 1428 MVC $HIST1(#HISLN),DFPOGE(,@XR) MOVE LOG TO NUCLEUS
2920 3A 04 03D5 1429 SBN $INDR2,$ERPND SET ENTRY PENDING INDR
2924 F0 00 00 1430 HPL *-*,*-* SOFT HALT ON INITIAL ERROR
2925 1431 ORG *-2 PLACE HALT CODE
2925 0070 2926 1432 DC AL2(@HPRER) DISPLAY CODE '123'
2927 1E 00 0434 E7 1433 DFP380 ALC $HISTE+@HSTPE(1),DFP001(,@BR) ADD ONE TO RETRY COUNTER
292C B8 20 D9 1434 TBN DFPRSN(,@XR),@PMGCK MARGIN CHECK
292F F2 10 07 1435 JT DFPMCK JUMP IF YES
2932 1436 DFPSCK EQU * ENTRY FOR SYNC CHK.
1437 *
1438 * LINE PRINTER MODE ONLY
1439 *
2932 38 40 03E4 1440 TBN $LPRP3,@PRINT LINE PRINTER ERROR 1-3
2936 F2 90 0F 1441 JF DFPSC2 JUMP IF NOT PRINT OP
2939 1442 DFPMCK EQU * ENTRY FOR MARGIN CHECK
2939 5F 00 ED E7 1443 SLC DFPRCT-DFPGCT(1,@BR),DFP001(,@BR) DECREMENT RETRY CNT
293D F2 81 72 1444 JZ DFP400 JUMP IF NO MORE RETRIES
2940 4C 00 FB 03C1 1445 MVC DFPSYC+@SYCNT(1,@BR),$LMRGN SET CNT TO HARD LEFT MARGIN
2945 F2 87 0B 1446 J DFP420 GO DO FIRST PART OF SYNC CHK
2948 1447 DFPSC2 EQU *
2948 5F 00 EC E7 1448 SLC DFPRCT-DFPYCT(1,@BR),DFP001(,@BR) DECREMENT CYNC CNT
294C F2 81 63 1449 JZ DFP400 JUMP IF NO MORE TRYs
294F 5C 02 F8 F4 1450 MVC DFPIST+@PDATA(@CADDR+1,@BR),DFPDSV(,@BR) RESTORE ORIGINAL
1451 * * COUNT AND DATA ADDR
2953 B4 01 D5 1452 DFP420 ST DFPASY(,@XR),@BR SET PAGE ADDR IN PCF ADDR
2956 9E 01 D5 F0 1453 ALC DFPASY(@CADDR,@XR),DFPYCD(,@BR) CALC PCF ADDR
295A B1 E6 D5 1454 LIO DFPASY(,@XR),@PCAR LOAD CONTROL LSR WITH SYNC SCF
295D 7A 80 F9 1455 SBN DFPSYC+@PCTRL(,@BR),@RETRN SET CHAIN BIT ON
2960 1C 00 03C2 FB 1456 MVC $PRPOS(1),DFPSYC+@SYCNT(,@BR) SET UP NEW HEAD POSITION
2965 5F 00 FB E7 1457 SLC DFPSYC+@SYCNT(1,@BR),DFP001(,@BR) SUBTRACT 1
2969 F2 02 03 1458 JNL DFP440 JUMP IF NOT NEG
296C 7B 80 F9 1459 SBF DFPSYC+@PCTRL(,@BR),@RETRN SET CHAIN BIT OFF
296F 38 40 03E4 1460 DFP440 TBN $LPRP3,@PRINT CHECK IF ENTRY FROM LINE PTR 1-3
2973 F2 90 39 1461 JF DLF450 JUMP NOT
2976 3A 01 03E4 1462 SBN $LPRP3,@INDEX SET DUMMY PRINT POS. FLAG 1-3
297A 0C 00 03E5 03C2 1463 MVC $LPROS(1),$PRPOS SET LINE PRINTER PRINT POSITION
2980 6C 00 BD D3 1464 MVC DFP330+@D1(1,@BR),DFPEXT(,@XR) SET DLRPRT ERROR ENTRY 1-4
2984 2C 01 144A D2 1465 MVC I$VADR,DFPLBU(2,@XR) GET LINE PRINTER BUFFER ADDR 1-4
2989 C0 87 1354 1466 B I$LOCK GET LINE PRINTER BUFFER 1-4
298D 4C 01 E5 144C 1467 MVC DLFORK(2,@BR),I$CADR SAVE BUFFER CADDR ADDR 1-4
2992 C0 87 1330 1468 B I$LDXR

```


DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 195

```

2996 4D00          2997 1469      DC      AL(@VADDR)(V$LPRT)      LINE PRINTER PAGE
2998 B9 04 D9      1470      TBF      DFPRSN(, @XR),DFPVCK      TEST VERTICLE CYCLE CHECK      1-4
299B F2 90 11      1471      JF       DLF450      IF VERTICAL CYCLE CHK      1-4
299E 9C 01 DE E5      1472      MVC      BUFRWK-DLFPRT(2, @XR),DLFORK(, @BR)  GET BUFFER ADDR      1-4
29A2 8C 01 DF 03EA      1473      MVC      DLFBPT-DLFPRT(2, @XR), $LPRIO  RESTORE BUF PTR & PDAR      1-4
29A7 2C 00 03E3 DF      1474      MVC      $BUFPT(1),DLFBPT-DLFPRT(, @XR)  RESTORE BUFFER POINTER      1-4
29AC BC 80 89      1475      MVI      DLF350-DLFPRT+@Q(, @XR), @NOP    FORCE ERROR CHECK
29AF D0 87 9C      29AF 1476 DLF450 EQU      *
29AF 1477      B      DFP260(, @BR)      GO TO MATRIX PRINTER

1479 *****
1480 * MATRIX PRINTER HARD FAILURE ROUTINE *
1481 *****
29B2 3A 21 03D2      1482 DFP400 SBN      $IOIND, $MPDWN+$HRDER      SET MAT4IX PRINTER DOWN INDR
29B6 3C 00 0434      1483      MVI      $HISTE+@HSTPE, @ZERO      SET HARD ERROR INDR
29BA 38 40 03E4      1484      TBN      $LPRP3, @PRINT      ENTRY FROM LINE PTR.      1-3
29BE F2 90 0D      1485      JF       DFP480      JUMP IF NOT
29C1 C0 87 1330      1486      B      I$LDXR      LOAD PAGE
29C5 4D00          29C6 1487      DC      AL2(V$LPRT)      LINE PRINTER PAGE
29C7 3C 00 03E3      1488      MVI      $BUFPT, @ZERO      RESET LINE PTR. BUFFER PTR.      1-3
29CB E0 87 B3      1489      B      RETURN-DLFPRT(, @XR)      GO TO LINE PRINTER PAGE
29CE D0 87 CA      29CE 1490 DFP480 EQU      *
29CE 1491      B      DFP300(, @BR)      RETURN TO MATRIX PRINTER
29D1 4F00          29D2 1493 DFPLBU DC      AL2(V$LPRB)      LINE PRINTER BUFFER VADDR      1-4
29D3 88          29D3 1494 DFPEXT DC      AL1(DLF350-DLFPRT)      DISPLACEMENT TO DLFPRT ERROR      1-4
29D4          29D5 1495 DFPASY DS      CL(@CADDR)      ADDR OF ERP PCF
29D6 E0          29D6 1496      DC      AL1(@PSIOQ)      HISTORY LOG SIO Q BYTE
29D7 00          29D7 1497 DFPIOR DC      AL1(@PSIOR)      HISTORY LOG SIO R BYTE
29D8          29D9 1498 DFPRSN DS      CL2      ERROR SENSE BYTES
29DA 00000001      29DD 1499 DFPERR DC      XL4'00000001'      ERROR INFO
29DD 1500 DFPOGE EQU      *-1      LAST BYTE OF HISTORY LOG
0000 1501 DFPGCT EQU      0      DISPLACEMENT MARGIN CK CNT
0004 1502 DFPVCK EQU      X'04'      PRINTER VERTICAL CYCLE CK.      1-4
1503 *****
1504 ##### X'29FF' IMG_0188 #####
1505 *      N O T      S C A N N E D      O R      O B J      C H E C K E D      !      !
1506 *      96 COLUMN CARD READER / PUNCHER
1507 ##### X'2AFF' #####
2AFD          1508      ORG      X'2AFD'
1509 *****
1510 * 5703-XM1 COPYRIGHT IBM CORP. 1970 *
1511 *      REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083 *
1512 * *
1513 *****
1514 *STATUS *
1515 *      VERSION 1 MODIFICATION 0 *
1516 * *
1517 *FUNCTION *
1518 * * FZXINP EXECUTION CAUSES KEYBOARD DATA ENTRY TO BE ENABLED *
1519 *      DURING PROGRAM OPERATION. ENTERED DATA ARE SYNTAX CHECKED WITH *
1520 *      RESPECT TO FORM AND TYPE, AND VALID ELEMENTS ARE CONVERTED TO *
1521 *      INTERNAL FORMAT AND PLACED IN THE RUN-TIME STACK ON AN INDI- *
1522 *      VIDUAL BASIS. *
1523 * * THIS ROUTINE PERFORMS THE PRIMARY FUNCTION OF SUPPORTING THE *
1524 *      EXECUTION OF BASIC PROGRAM 'INPUT' STATEMENTS. ON A SECONDARY *

```

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 196

		1525 *	LEVEL, THE MESSAGE PRINTING, SYNTAX CHECKING, AND DATA CONVER-	*
		1526 *	SION FACILITIES REQUIRED FOR 'INPUT' ARE ORGANIZED FOR ALTER-	*
		1527 *	NATE USE DURING THE EXECUTION OF 'MAT INPUT' AND 'GET' (CARD)	*
		1528 *	BASIC STATEMENTS.	*
		1529 *	* TWO ENTRY POINTS ARE PROVIDED FOR 'INPUT' MODE OPERATIONS. THE	*
		1530 *	FIRST (FZXIP1) OPERATES IN CONJUNCTION WITH STACKED DATA TYPE	*
		1531 *	CODES AND A COUNT PARAMETER IN I\$PARM TO ALLOW KEYBOARD DATA	*
		1532 *	INPUT AND DATA LINE VALIDITY CHECKING. THE SECOND ENTRY POINT	*
		1533 *	(FZXIP2) OPERATES ON THE VALIDITY-CHECKED DATA LINE TO CONVERT	*
		1534 *	AND STACK SEQUENTIALLY OCCURRING DATA ELEMENTS.	*
		1535 *	* SIX ALTERNATE ENTRY POINTS ARE PROVIDED FOR USE WITH 'MAT INPUT'	*
		1536 *	AND 'GET' (CARD) OPERATIONS.	*
		1537 *	* ENTRY POINTS FZXPQ1, FZXPQ2, AND FZXPEM ARE USED TO PRINT	*
		1538 *	QUESTION MARK(S) OR ERROR MESSAGES ON THE SYSTEM PRINT	*
		1539 *	DEVICE.	*
		1540 *	* ENTRY POINT FZXGCS IS USED TO SYNTAX CHECK AN ENTIRE 'GET'	*
		1541 *	(CARD) INPUT LINE (INTO WHICH COMMA DELIMITERS HAVE BEEN	*
		1542 *	INSERTED WERE NOT ORIGINALLY EXISTENT).	*
		1543 *	* ENTRY POINT FZXMIS IS USED TO VALIDITY CHECK A PARTIAL OR	*
		1544 *	ENTIRE ARRAY ROW.	*
		1545 *	* ENTRY POINT FZXCNV IS USED TO CONVERT AND STACK INDIVIDUAL	*
		1546 *	INPUT LINE ELEMENTS AFTER THE LINE HAS BEEN SYNTAX OR	*
		1547 *	VALIDITY CHECKED.	*
		1548 *		*
		1549 *	ENTRY POINTS	*
		1550 *	* ENTRY FZXIP1 - FOR ENABLING 'INPUT' KEYBOARD DATA ENTRY AND	*
		1551 *	SYNTAX CHECKING THE RESULTING DARA LINE. CALLING SEQUENCE IS	*
		1552 *	B I\$CALL	*
		1553 *	DC AL2(V\$XKIN)	*
		1554 *	WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL	*
		1555 *	ADDRESS OF ENTRY POINT FZXIP1.	*
		1556 *	* ENTRY FZXIP2 - FOR CONVERTING AND STACKING A SINGLE DATA ELE-	*
		1557 *	MENT FROM THE 'INPUT' DATA LINE. CALLING SEQUENCE IS	*
		1558 *	B I\$CALL	*
		1559 *	DC AL2(V\$XSIN)	*
		1560 *	WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL	*
		1561 *	ADDRESS OF ENTRY POINT FZXIP2.	*
		1562 *	* ENTRY FZXPQ1 - FOR PRINTING A SINGLE QUESTION MARK (?) ON THE	*
		1563 *	SYSTEM PRINT DEVICE, CALLING SEQUENCE IS	*
		1564 *	B I\$CALL	*
		1565 *	DC AL2(FZXPQ1)	*
		1566 *	WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL	*
		1567 *	ADDRESS OF ENTRY POINT FZXPQ1.	*
		1568 *	* ENTRY FZXPQ1 - FOR PRINTING A DOUBLE QUESTION MARK (??) ON THE	*
		1569 *	SYSTEM PRINT DEVICE, CALLING SEQUENCE IS	*
		1570 *	B I\$CALL	*
		1571 *	DC AL2(FZXPQ2)	*
		1572 *	WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL	*
		1573 *	ADDRESS OF ENTRY POINT FZXPQ2.	*
		1574 *	* ENTRY FZXPEM - FOR PRINTING A DATA INPUT ERROR MESSAGE ON THE	*
		1575 *	SYSTEM PRINT DEVICE. CALLING SEQUENCE IS	*
		1576 *	B I\$CALL	*
		1577 *	DC AL2(FZXPEM)	*
		1578 *	WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL	*
		1579 *	ADDRESS OF ENTRY POINT FZXPEM.	*
		1580 *	* ENTRY FZXGCS - FOR SYNTAX CHECKING A DATA LINE RESULTING FROM	*

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 197
1581	*		A	'GET' USING A CARD INPUT FILE. CALLING SEQUENCE IS	*
1582	*		B	I\$CALL	*
1583	*		DC	AL2(V\$CDSY)	*
1584	*			WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL	*
1585	*			ADDRESS OF ENTRY POINT FZXGCS.	*
1586	*	*		ENTRY FZXMIS - FOR SYNTAX CHECKING A DATA LINE RESULTING FROM	*
1587	*		A	'MAT INPUT' ARRAY ROW INPUT. CALLING SEQUENCE IS	*
1588	*		B	I\$CALL	*
1589	*		DC	AL2(FZXMIS)	*
1590	*			WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL	*
1591	*			ADDRESS OF ENTRY POINT FZXMIS.	*
1592	*	*		ENTRY FZXCNV - FOR CONVERTING AND STACKING A SINGLE DATA ELE-	*
1593	*			MENT FROM A 'GET' (CARD) OR 'MAT INPUT' DATA LINE.	*
1594	*			CALLING SEQUENCE IS	*
1595	*		B	I\$CALL	*
1596	*		DC	AL2(V\$CDCV)	*
1597	*			WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL	*
1598	*			ADDRESS OF ENTRY POINT FZXCNV.	*
1599	*	*		ENTRY POINT EXECUTION FOR THESE OPERATIONS IS SUBJECT TO INPUT	*
1600	*			CONDITIONS DESCRIBED BELOW.	*
1601	*				*
1602	*			*INPUT	*
1603	*	*		I\$STAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER.	*
1604	*		*	FOR ENTRY FZXIP1, THIS CONTAINS THE CORE ADDRESS OF THE	*
1605	*			LEFTMOST BYTE OF THE STACKED NATA TYPE SPECIFICATION CODES.	*
1606	*		*	FOR ENTRY FZXIP2, THIS CONTAINS THE CORE ADDRESS OF THE	*
1607	*			FIRST AVAILABLE STACK LOCATION.	*
1608	*		*	FOR ENTRY FZXMIS, THIS CONTAINS THE CORE ADDRESS OF THE	*
1609	*			SINGLE STACKED DATA TYPE SPECIFICATION CODE.	*
1610	*		*	FOR ENTRY FZXCNV, THIS CONTAINS THE CORE ADDRESS OF THE	*
1611	*			STACK LOCATION INTO WHICH THE CONVERTED DATA ELEMENT IS	*
1612	*			TO BE STACKED.	*
1613	*	*		I\$PARM - 2 BYTES, FOR THE INTERPRETER COMMUNICATIONS PARAMETER.	*
1614	*		*	FOR ENTRY FZXIP1, THE RIGHT BYTE IN I\$PARM CONTAINS A	*
1615	*			COUNT OF THE STACKED DATA TYPE SPECIFICATION CODES.	*
1616	*		*	FOR ENTRY FZXMIS, THE RIGHT BYTE IN I\$PARM CONTAINS A	*
1617	*			VALUE OF '1' FOR THE SINGLE STACKED DATA TYPE SPEC CODE.	*
1618	*	*		I\$PUB1 - 2 BYTES, FOR THE DATA BUFFER CORE ADDRESS. FOR ENTRY	*
1619	*			FZXIP2 ONLY, THIS CONTAINS THE CORE ADDRESS OF THE 'INPUT' DATA	*
1620	*			BUFFER LEFTMOST BYTE.	*
1621	*	*		I\$ERRC - 1 BYTE, FOR THE INTERPRETER ERROR CONDITION CODE, FOR	*
1622	*			ENTRY FZXPEM ONLY, THIS CONTAINS AN ERROR CODE IN THE DECIMAL	*
1623	*			NUMBER RANGE X'F0' THROUGH X'F4', INDICATING OUTPUT OF AN ERROR	*
1624	*			MESSAGE IN THE RANGE 800 THROUGH 804 (SYSTEM ERROR MESSAGE	*
1625	*			NUMBERS) RESPECTIVELY.	*
1626	*	*		REGISTER @XR - FOR ENTRIES FZXGCS, FZXMIS AND FZXCNV, THIS	*
1627	*			CONTAINS THE CORE ADDRESS OF THE DATA BUFFER LEFTMOST BYTE.	*
1628	*	*		RUN-TIME STACK - FOR ENTRIES FZXIP1 AND FZXMIS, THIS CONTAINS	*
1629	*			DATA TYPE SPECIFICATION CODES BEGINNING AT THE CORE ADDRESS	*
1630	*			REFERENCED BY I\$STAK.	*
1631	*	*		KEYBOARD DATA ENTRY - FOR ENTRY FZXIP1 ONLY, NUMERIC, SIGNED	*
1632	*			INTERNAL (-&PI, ETC.), AND CHARACTER CONSTANTS ENCLOSED IN	*
1633	*			QUOTES ARE ENTERED FROM THE SYSTEM CONSOLE KEYBOARD IN COMPLI-	*
1634	*			ANCE WITH THE CURRENT 'INPUT' STATEMENT ASSIGNMENT LIST.	*
1635	*	*		INPUT DATA BUFFER - 256 BYTES, FOR DATA ELEMENT STORAGE DURING	*
1636	*			SYNTAX CHECKING AND CONVERSION.	*

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 198
1637	*		*	FOR ENTRIES FZXGCS AND FZXMIS, THIS BUFFER CONTAINS THE	*
1638	*			DATA INPUT LINE (AS ENTERED) BEGINNING AT THE LEFTMOST	*
1639	*			BUFFER BYTE.	*
1640	*		*	FOR ENTRIES FZXIP2 AND FZXCNV, THIS BUFFER CONTAINS, IN	*
1641	*			ADDITION TO THE ORIGINAL DATA, A 2-BYTE FIELD IN THE LAST	*
1642	*			TWO BUFFER BYTES. THIS FIELD CONTAINS THE CORE ADDRESS OF	*
1643	*			THE CHARACTER PRECEDING THE FIRST CHARACTER OF THE NEXT	*
1644	*			DATA ELEMENT TO BE CONVERTED AND STACKED.	*
1645	*				*
1646	*		OUTPUT		*
1647	*		*	INPUT DATA BUFFER - 256 BYTES, FOR DATA ELEMENT STORAGE DURING	*
1648	*			SYNTAX CHECKING AND CONVERSION.	*
1649	*		*	AFTER ENTRY FZXIP1, THIS BUFFER IS LOCKED INTO CORE, DATA	*
1650	*			ARE LOADED INTO THE BUFFER FROM THE SYSTEM KEYBOARD, AND	*
1651	*			THESE DATA ARE COMPLETELY SYNTAX/TYPE CHECKED (EXCEPT WHEN	*
1652	*			CONSOLE INTERRUPT IS INVOKED).	*
1653	*		*	AFTER ENTRY FZXIP2, THIS BUFFER IS UNLOCKED FROM CORE	*
1654	*			FOLLOWING CONVERSION OF THE FINAL DATA ELEMENT (DEFINED BY	*
1655	*			AN EOS CHARACTER DELIMITER).	*
1656	*		*	AFTER ENTRY FZXMIS, THE DATA CONTAINED IN THIS BUFFER ARE	*
1657	*			COMPLETELY SYNTAX/TYPE CHECKED (UP TO THE POINT WOCRE AN	*
1658	*			ERROR CONDITION, IF ONE EXISTS. IS ENCOUNTERED).	*
1659	*		*	AFTER ENTRY FZXGCS, THE DATA CONTAINED IN THIS BUFFER ARE	*
1660	*			COMPLETELY SYNTAX CHECKED (UP TO THE POINT WHERE AN ERROR	*
1661	*			CONDITION, IF ONE EXISTS. IS ENCOUNTERED).	*
1662	*		*	DATA BUFFER POINTER - 2 BYTES, FOR THE CORE ADDRESS OF THE	*
1663	*			CURRENTLY REFERENCED BUFFER CHARACTER. THIS POINTER IS CON-	*
1664	*			TAINED IN THE TWO RIGHTMOST BYTES OF THE INPUT DATA BUFFER.	*
1665	*		*	AFTER ENTRIES FZXIP1, FIXMIS, AND FIXGCS, THIS IS SET TO	*
1666	*			CONTAIN THE CORE ADDRESS OF THE BYTE PRECEDING THE FIRST	*
1667	*			BUFFER BYTE,	*
1668	*		*	AFTER ENTRIES FZXIP2 AND FZXCNV, THIS IS SET TO CONTAIN	*
1669	*			THE CORE ADDRESS OF THE DELIMITER FOLLOWING THE CONVERTED	*
1670	*			DATA ELEMENT.	*
1671	*		*	I\$PUB1 - 2 BYTES, FOR THE INPUT DATA BUFFER CORE ADDRESS.	*
1672	*			AFTER ENTRY FZXIP1 ONLY, THIS IS SET TO CONTAIN THE CORE	*
1673	*			ADDRESS OF THE INPUT DATA BUFFER LEFTMOST BYTE.	*
1674	*		*	REGISTER @XR - AFTER ENTRIES FZXGCS, FZXMIS AND FZXCNV, THIS	*
1675	*			CONTAINS THE CORE ADDRESS OF THE LAST REFERENCED INPUT DATA	*
1676	*			BUFFER CHARACTER,	*
1677	*		*	I\$PARM - 2 BYTES, FOR THE INTERPRETER COMMUNICATIONS PARAMETER,	*
1678	*			AFTER ENTRIES FZXMIS AND FZXGCS, THE LEFT BYTE IN I\$PARM IS SET	*
1679	*			TO CONTAIN A COUNT OF THE DATA ELEMENTS SYNTAX CHECKED IN THE	*
1680	*			BUFFER.	*
1681	*		*	I\$ERRC - 1 BYTE, FOR THE INTERPRETER ERROR CONDITION CODE.	*
1682	*		*	AFTER ENTRY FZXPEN, THIS IS RESET TO CONTAIN NULL ERROR	*
1683	*			CODE I@NERR.	*
1684	*		*	AFTER ENTRY FZXMIS, THIS IS SET TO CONTAIN ERROR CODE	*
1685	*			X'F0', X'F2', OR X'F4' (CORRESPONDING TO ERROR MESSAGES	*
1686	*			800, 802, OR 804 RESPECTIVELY) WHEN A SYNTAX OR DATA TYPE	*
1687	*			ERROR IS ENCOUNTERED.	*
1688	*		*	AFTER ENTRY FZXGCS, THIS IS SET TO CONTAIN ERROR CODE	*
1689	*			X'FF' WHEN A SYNTAX ERROR IS ENCOUNTERED.	*
1690	*		*	RUN-TIME STACK - AFTER ENTRIES FZXIP2 AND FZXCNV, THIS CONTAINS	*
1691	*			THE CONVERTED INPUT DATA ELEMENT BEGINNING AT THE CORE ADDRESS	*
1692	*			SPECIFIED IN I\$STAK. ARITHMETIC ELEMENTS ARE STACKED IN PACKED	*

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 199
1693	*			FLOATING POINT FORMAT OF CURRENT PRECISION. CHARACTER ELEMENTS	*
1694	*			ARE STACKED IN 19-BYTE FORMAT.	*
1695	*		*	PRINTER/CRT MESSAGES -	*
1696	*		*	FOR ENTRY FZXIP1, FZXPQ1 IS EXECUTED EACH TIME THAT KEY-	*
1697	*			BOARD DATA INPUT IS REQUESTED.	*
1698	*		*	FOR ENTRY FZXIP1, AN ERROR MESSAGE (800, 801, 802, OR 803)	*
1699	*			IS PRINTED WHENEVER A SYNTAX/TYPE ERROR IS ENCOUNTERED,	*
1700	*			AND KEYBOARD DATA INPUT RE-ENTRY IS REQUESTED.	*
1701	*		*	FOR ENTRY FZXPQ1, THE PRINT DEVICE CARRIER IS RETURNED,	*
1702	*			A SINGLE QUESTION MARK IS PRINTED, AND THE CARRIER IS	*
1703	*			RETURNED AGAIN TO START A NEW LINE.	*
1704	*		*	FOR ENTRY FZXPQ2, THE PRINT DEVICE CARRIER IS RETURNED,	*
1705	*			A DOUBLE QUESTION MARK IS PRINTED, AND THE CARRIER IS	*
1706	*			RETURNED AGAIN TO START A NEW LINE.	*
1707	*		*	FOR ENTRY FZXPPEM, THE PRINT DEVICE CARRIER IS RETURNED,	*
1708	*			A MESSAGE OF THE FORM	*
1709	*			ERROR NNN AT LINE LLLL MESSAGE...	*
1710	*			IS PRINTED, AND THE CARRIER IS RETURNED TO START A NEW	*
1711	*			LINE. MESSAGES ARE -	*
1712	*		*	800 - INVALID INPUT DATA - NUMERIC CONSTANT	*
1713	*		*	801 - INVALID INPUT DATA - CHARACTER DATA	*
1714	*		*	802 - TOO MANY INPUT DATA ELEMENTS	*
1715	*		*	803 - NOT ENOUGH DATA ELEMENTS ENTERED	*
1716	*		*	804 - NOT ENOUGH ARRAY ROW ELEMENTS ENTERED	*
1717	*		*	I\$STAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER. FOR ENTRY	*
1718	*			FZXIP1 ONLY, THIS POINTER IS DECREMENTED BY THE LENGTH OF A	*
1719	*			VIRTUAL ADDRESS (2 BYTES) WHENEVER CONSOLE INTERRUPT IS	*
1720	*			INVOKED DURING KEYBOARD INPUT.	*
1721	*		*	I\$WRK1 - 2 BYTES, FOR INTERPRETER COMMON WORK AREA 1. AFTER	*
1722	*			ENTRY FZXIP1 ONLY, THIS AREA IS SET TO CONTAIN THE VIRTUAL	*
1723	*			ADDRESS OF THE CURRENT 'INPUT' STATEMENT 'STH' PSEUDO INSTRU-	*
1724	*			CTION WHENEVER CONSOLE INTERRUPT IS INVOKED DURING KEYBOARD I/P.	*
1725	*		*	ISRESW - 1 BYTE, FOR THE RECURSIVE STATEMENT ERROR SWITCH.	*
1726	*			THIS SWITCH IS SET TO CODE @NOP (DISABLES THE ERROR CONDITION)	*
1727	*			WHENEVER CONSOLE INTERRUPT IS INVOKED DURING KEYBOARD INPUT.	*
1728	*				*
1729	*		*	EXTERNAL REFERENCES	*
1730	*		*	V\$SKEY - VIRTUAL ENTRY ADDRESS FOR DFKEYN, V.M. KEYBOARD IOCS.	*
1731	*		*	V\$SPRT - VIRTUAL ENTRY ADDRESS FOR DFPRNT, V.M. MATRIX PRT IOCS.	*
1732	*		*	DSPLIN - ENTRY POINT FOR THE SYSTEM CRT IOCS (LABEL DSPLIN IS	*
1733	*			REFERENCED INDIRECTLY USING I\$CSXA TO BUILD A CORE ADDRESS).	*
1734	*		*	I\$CALL - ENTRY POINT FOR PAGING MODULE V.M. PROGRAM CALL RTN.	*
1735	*		*	I\$RTRN - ENTRY POINT FOR PAGING MODULE V.M. RETURN CONTROL RTN.	*
1736	*		*	I\$LOCK - ENTRY POINT FOR PAGING MODULE V.M. PAGE LOCKING RTN.	*
1737	*		*	I\$UNLK - ENTRY POINT FOR PAGING MODULE V.M. PAGE UNLOCKING RTN.	*
1738	*		*	I\$STCK - ENTRY POINT FOR INTERPRETER ELEMENT STACKING ROUTINE.	*
1739	*		*	I\$CUPF - ENTRY POINT FOR FLOATING POINT VALUE PACKING ROUTINE.	*
1740	*		*	I\$BSET - ENTRY POINT FOR ICBAN EXECUTION CONTROL BRANCH RTN.	*
1741	*		*	I\$STAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER.	*
1742	*		*	I\$PARM - 2 BYTES, FOR THE INTERPRETER COMMUNICATION PARAMETER.	*
1743	*		*	I\$WRK1 - 2 BYTES, FOR INTERPRETER COMMON WORK AREA 1.	*
1744	*		*	I\$ERRC - 1 BYTE, FOR THE INTERPRETER EXECUTION ERROR CODE.	*
1745	*		*	I\$STHA - 2 BYTES, FOR VIRTUAL ADDRESS OF CURRENT STMT 'STH'.	*
1746	*		*	I\$RESW - 1 BYTE, FOR INTERPRETER STATEMENT RECURSION ERROR SW.	*
1747	*		*	I\$PUB1 - 2 BYTES, FOR THE DATA BUFFER SAVED CORE ADDRESS.	*
1748	*		*	I\$VADR - 2 BYTES, FOR PAGING MODULE VIRTUAL ADDRESS PARAMETER.	*

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 200
		1749	*	* I\$CADR - 2 BYTES, FOR PAGING MODULE CORE ADDRESS OUTPUT PARAM.	*
		1750	*	* \$INLNO - 2 BYTES, FOR THE SYSTEM EXECUTION LINE NUMBER AREA.	*
		1751	*	* \$PRDEV - 2 BYTES, FOR THE SYSTEM PRINT DEVICE INDICATOR.	*
		1752	*	* \$EXFTR - 1 BYTE, FOR THE SYSTEM CORE EXTENSION FACTOR.	*
		1753	*	* \$CISUS - 1 BYTE FOR THE SYSTEM CONSOLE INTERRUPT INDICATOR.	*
		1754	*	* V\$BFR1 - VIRTUAL ADDRESS FOR GENERAL V.M. BUFFER 1.	*
		1755	*	* I\$BASE - CORE ADDRESS FOR INTERP BASE ADDRESSABILITY.	*
		1756	*	* I\$CSXA - CORE ADDRESS OF 1ST BYTE IN CORE EXTENSION PAST 8K.	*
		1757	*		*
		1758	*	*EXITS, NORMAL	*
		1759	*	CONTROL IS NORMALLY PASSED TO THE PAGING ROUTINE AT ENTRY POINT	*
		1760	*	I\$RTRN (IPGRTN) FOR A RETURN TO THE CALLING PROGRAM. THE SINGLE	*
		1761	*	EXCEPTION OCCURS WHEN A CONSOLE INTERRUPT IS INVOKED WHILE DATA	*
		1762	*	ARE BEING ENTERED THROUGH THE KEYBOARD DURING FZXIP1 EXECUTION.	*
		1763	*	IN THIS EVENT, CONTROL IS PASSED TO THE CORE-RESIDENT INTERPRETER	*
		1764	*	AT ENTRY POINT I\$BSET (ICBSET) TO FORCE RE-EXECUTION OF THE LAST	*
		1765	*	EXECUTED 'STH' PSEUDO INSTRUCTION (I.E, THE STATEMENT HEADER	*
		1766	*	ASSOCIATED WITH THE CURRENT 'INPUT' STATEMENT).	*
		1767	*		*
		1768	*	*EXITS, ERROR	*
		1769	*	* FOR ENTRY FZXMIS, CONTROL IS PASSED TO THE PAGING ROUTINE AT	*
		1770	*	ENTRY POINT I\$RTRN (IPGRTN) WHENEVER A SYNTAX OR DATA TYPE	*
		1771	*	ERROR IS ENCOUNTERED. ERROR PARAMETER I\$ERRC IS SET TO ONE OF	*
		1772	*	THE CODES X'F0', X'F2', OR X'F4' WHEN THIS SITUATION OCCURS	*
		1773	*	(SEE OUTPUT ABOVE).	*
		1774	*	* FOR ENTRY FZXSCS, CONTROL IS PASSED TO THE PAGING ROUTINE AT	*
		1775	*	ENTRY POINT I\$RTRN (IPGRTN) WHENEVER A SYNTAX ERROR IS	*
		1776	*	ENCOUNTERED. ERROR PARAMETER I\$ERRC IS SET TO CODE X'FF'.	*
		1777	*	* FOR ALL OTHER ENTRY POINTS, ENCOUNTERED ERROR CONDITIONS CAUSE	*
		1778	*	SPECIFIC ERROR-CORRECTION ACTIONS TO BE EXECUTED INTERNALLY.	*
		1779	*		*
		1780	*	*TABLES/WORK AREAS	*
		1781	*	* INPUT DATA BUFFER - SINGLE VIRTUAL PAGE USED TO CONTAIN	*
		1782	*	BOARD ENTERED DATA CONSTANTS AND INTERNAL CONSTANT SYMBOLS.	*
		1783	*	BYTES 254, 255 IN THIS PAGE ARE USED TO CONTAIN THE DATA BUFFER	*
		1784	*	CHARACTER POINTER.	*
		1785	*	* ERROR MESSAGE PAGE - THE 3RD VIRTUAL PAGE OR FZXINP CODING CON-	*
		1786	*	TAINS FIVE ERROR MESSAGE PARAMETER LISTS AND MESSAGE TEXTS FOR	*
		1787	*	SYSTEM ERROR MESSAGES 800, 801, 802, 803, AND 804. THIS PAGE	*
		1788	*	IS CREATED BY THE MTEXT MESSAGE GENERATOR, AND INCLUDES PATCH_	*
		1789	*	AREA SUFFICIENT FOR REASONABLE MESSAGE TEXT EXPANSION.	*
		1790	*	* INTERNAL CONSTANT SYNTAX CHECKING TABLE - THIS CONTAINS A LIST	*
		1791	*	OF INTERNAL CONSTANT SYMBOLS FOR VALIDITY CHECKING.	*
		1792	*	* NUMBER OF TABLE ENTRIES - 3	*
		1793	*	* TABLE ENTRY LENGTH - 4 BYTES	*
		1794	*	* ENTRY FORMAT -	*
		1795	*	* FILLED FROM LEFT TO RIGHT WITH THE LETTER PORTION	*
		1796	*	OF INTERNAL CONSTANT SYMBOLS (REVERSE SPELLING).	*
		1797	*	* PADDED ON RIGHT WITH BINARY ZEROS.	*
		1798	*	* INTERNAL CONSTANT CONVERSION TABLE - THIS CONTAINS A LIST OF	*
		1799	*	INTERNAL CONSTANT IDENTIFIERS AND THE VIRTUAL ADDRESS ASSOCI-	*
		1800	*	ATED WITH EACH (SIGNED) INTERNAL CONSTANT.	*
		1801	*	* NUMBER OF TABLE ENTRIES - 6	*
		1802	*	* TABLE ENTRY LENGTH - 4 BYTES	*
		1803	*	* ENTRY FORMAT -	*
		1804	*	* BYTE 0 - CONTAINS A SIGN CHARACTER (+ OR -)	*

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 201
		1805	*	* BYTE 1 - CONTAINS SYMBOL 1ST LETTER CHARACTER	*		
		1806	*	* BYTES 2, 3 - CONTAIN INTERNAL CONSTANT VIRTUAL ADDR	*		
		1807	*	ASSOCIATED WITH THE SIGNED CONSTANT SYMBOL.	*		
		1808	*	* ARITHMETIC CONSTANT CONVERSION BUCKET - 16 BYTES, FOR ACCUMU-	*		
		1809	*	LATING CHARACTERS USED TO GENERATE AN INTERNAL FORM ARITHMETIC	*		
		1810	*	FLOATING POINT VALUE.	*		
		1811	*	* CHARACTER CONSTANT CONVERSION BUCKET - 19 BYTES, FOR ACCUMU-	*		
		1812	*	LATING CHARACTERS USED TO GENERATE AN INTERNAL FORM CHARACTER	*		
		1813	*	ELEMENT.	*		
		1814	*		*		
		1815	*	*ATTRIBUTES	*		
		1816	*	REUSABLE, NATURALLY RELOCATABLE	*		
		1817	*		*		
		1818	*	*CHARACTER CODE DEPENDENCY	*		
		1819	*	THE OPERATION OF THIS MODULE DEPENDS UPON THE FOLLOWING PROPER-	*		
		1820	*	TIES OF THE INTERNAL REPRESENTATION OF THE EXTERNAL CHAR SET,	*		
		1821	*	* MOST CODING HAS BEEN ARRANGED SO THAT REDEFINITION OF CHAR-	*		
		1822	*	ACTER CONSTANTS, BY REASSEMBLY, WILL RESULT IN A CORRECT	*		
		1823	*	MODULE FOR THE NEW DEFINITION.	*		
		1824	*	* NUMERIC CHARACTERS 0 THROUGH 9 ARE PRESUMED TO BE CODED IN	*		
		1825	*	INCREASING COLLATING SEQUENCE, AND THE RANGE OF CHARACTER	*		
		1826	*	CONSTANTS FOR THIS SERIES IS EXPECTED TO COLLATE HIGHER THAN	*		
		1827	*	THAT FOR ANY OTHER CHARACTER IN THE EXTERNAL CHARACTER SET.	*		
		1828	*	* ALPHABETIC LETTERS A THROUGH Z ARE PRESUMED TO EF CODED IN	*		
		1829	*	INCREASING COLLATING SEQUENCE, AND THE RANGE OF CKARACTER	*		
		1830	*	CONSTANTS FOR THIS SERIES IS EXPECTED TO COLLATE HIGHER THAN	*		
		1831	*	THAT FOR ANY OTHER NON-NUMERIC CHARACTER IN THE EXTERNAL	*		
		1832	*	CHARACTER SET.	*		
		1833	*	* DECIMAL NUMBERS MUST BE CODED SO TWAT THE LOW ORDER FOUR	*		
		1834	*	BITS, WHEN CONSIDERED AS A BINARY INTEGER, IDENTIFY THE	*		
		1835	*	VALUE OF THE DIGIT.	*		
		1836	*	THE SPECIFIC INSTRUCTIONS (INSTRUCTION SEQUENCES) WHICH REQUIRE	*		
		1837	*	MODIFICATION IF THESE PROPERTIES OF THE CHARACTER SET ARE CHANGED	*		
		1838	*	MAY BE IDENTIFIED EY -	*		
		1839	*	* THE 2 INSTRUCTIONS BEGINNING AT LABEL FZX575.	*		
		1840	*	* THE 8 INSTRUCTIONS BEGINNING AT LABEL FZX750.	*		
		1841	*	* THE 2 INSTRUCTIONS BEGINNING AT LABEL FZX845.	*		
		1842	*	* THE 2 INSTRUCTIONS BEGINNING AT LABEL FZX904.	*		
		1843	*	* THE 2 INSTRUCTIONS BEGINNING AT LABEL FZX940.	*		
		1844	*	* THE SINGLE INSTRUCTION IDENTIFIED BY LABEL F7X996.	*		
		1845	*		*		
		1846	*	*NOTES	*		
		1847	*	ERROR PROCEDURES	*		
		1848	*	* ERROR 1 - AN ARITHMETIC DATA ELEMENT IS EXPECTED DURING	*		
		1849	*	'INPUT' SYNTAX CHECKING, AND THE PROCESSED ELEMENT IS EITHER	*		
		1850	*	NON-ARITHMETIC OR OTHERWISE INVALID. THE ERROR MESSAGE	*		
		1851	*	'INVALID INPUT DATA - NUMERIC CONSTANT' IS DISPLAYED.	*		
		1852	*	* ERROR 2 - A CHARACTER DATA ELEMENT IS EXPECTED DURING 'INPUT'	*		
		1853	*	SYNTAX CHECKING, AND THE PROCESSED ELEMENT IS EITHER NON-	*		
		1854	*	CHARACTER OR OTHERWISE INVALID. THE ERROR MESSAGE 'INVALID	*		
		1855	*	INPUT DATA - CHARACTER DATA' IS DISPLAYED.	*		
		1856	*	* ERROR 3 - MORE THAN THE EXPECTED NUMBER OF DATA ELEMENTS ARE	*		
		1857	*	ENCOUNTERED DURING 'INPUT' SYNTAX' CHECKING. THE ERROR	*		
		1858	*	MESSAGE 'TOO MANY INPUT DATA ELEMENTS' IS DISPLAYED.	*		
		1859	*	* ERROR 4 - LESS THAN THE EXPECTED NUMBER OF DATA ELEMENTS ARE	*		
		1860	*	ENCOUNTERED DURING 'INPUT' SYNTAX CHECKING. THE ERROR	*		

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 202
		1861	*	MESSAGE 'NOT ENOUGH DATA ELEMENTS ENTERED' IS DISPLAYED.	*
		1862	*	* IN EACH OF THE ABOVE CASES, THE DISPLAY IS PERFORMED ON THE	*
		1863	*	CURRENT SYSTEM PRINT DEVICE(S), AND THE ERROR MESSAGE IS	*
		1864	*	FOLLOWED (ON A SEPARATE LINE) WITH A SINGLE QUESTION MARK	*
		1865	*	REQUESTING CORRECT RE-ENTRY OF THE 'INPUT' DATA LINE.	*
		1866	*	* ERROR CONDITIONS ENCOUNTERED DURING 'MAT INPUT' AND 'GET'	*
		1867	*	(CARD) DATA SYNTAX CHECKING ARE NOT CONSIDERED TO BE ASSOCI-	*
		1868	*	ATED WITH FZXINP, BUT EACH IS REGARDED (AS A CODE IN I\$ERRC)	*
		1869	*	TO BE OUTPUT TO THE 'MAT INPUT' OR 'GET' (CARD) PROCESSING	*
		1870	*	ROUTINES.	*
		1871	*	* KEYBOARD DATA INPUT CAN BE ABORTED AT ANY TIME, DURING DATA	*
		1872	*	ENTRY, THROUGH CONSOLE INTERRUPT. WHEN THIS OCCURS, THE	*
		1873	*	'STH' PSEUDO INSTRUCTION FOR THE CURRENT 'INPUT' STATEMENT	*
		1874	*	IS RE-EXECUTED AND THE PROGRAM IS HALTED IN 'PAUSE' MODE.	*
		1875	*	* FZXINP OTHERWISE UTILIZES INPUT IOCS ROUTINE DFKEYN (FOR THE	*
		1876	*	KEYBOARD) AND OUTPUT IOCS ROUTINES DFPRNT (MATRIX PRINTER)	*
		1877	*	AND DSPLYN (CRT), AND IS STASKCT TO THE ERP'S INHERENT IS	*
		1878	*	THESE CONTROL PROGRAMS.	*
		1879	*		*
		1880	*	REGISTER USAGE	*
		1881	*	* REGISTER @BR IS TO CONTAIN THE CORE PAGE BASE ADDRESS	*
		1882	*	ESTABLISHED THROUGH PAGING MODULE CONTROL FOR THE PAGE WHICH	*
		1883	*	INCLUDES FZXINP, AND IS RESTORED TWROUGH NE PAGING MODULE.	*
		1884	*	* REGISTER @XR IS NOT SAVED. IT IS USED IN FZXINP FOR GENERAL	*
		1885	*	PURPOSE INDEXING OPERATIONS, AND IN CERTAIN CASES IS ALSO	*
		1886	*	USED AS AN INPUT OR OUTPUT PARAMETER FOR THIS ROUTINE.	*
		1887	*		*
		1888	*	SAVED/RESTORED AREAS	*
		1889	*	NONE	*
		1890	*		*
		1891	*	MODIFICATION CONSIDERATIONS	*
		1892	*	NONE	*
		1893	*		*
		1894	*	REQUIRED MODULES	*
		1895	*	* @SYSEQ - COMMON SYSTEM EQUATES.	*
		1896	*	* @FXDEQ - SYSTEM NUCLE4S ADDRESSES AND INDICATOR EQUATES.	*
		1897	*	* \$V\$EQU - VIRTUAL MEMORY FIXED ADDRESS EQUATES.	*
		1898	*	* \$B@EQU - COMPILER PARAMETER AND CONSTANT EQUATES.	*
		1899	*	* \$I\$EQU - INTERPRETER FI\ED LOCATION ADDRESS ELATES.	*
		1900	*	* \$I@SEQ - INTERPRETER PARAMETER EQUATES (FOR STD. PREC, ONLY)	*
		1901	*	* \$I@LEQ - INTERPRETER PARAMETER EQUATES (FOR LONG PREC, ONLY)	*
		1902	*	* MTEXT - SYSTEM ERROR MESSAGE GENERATOR.	*
		1903	*		*
		1904	*	OTHER	*
		1905	*	NONE	*
		1906	*	*****	*

[illegible]

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 203
					1908	*****			
					1909	* START OF INPUT STATEMENT EXECUTION MODULE			*
					1910	*****			
					1911	*			
					1912	* ESTABLISH ADDRESSABILITY FOR INPUT ROUTINE 1ST VM PAGE			
					1913	*			
					1914	*FZXPI8 VPAGE 0			SET 1ST PAGE ADDRESSABILITY
	2B00				1915	ORG *,256,0			SET STARTING ADDRESS
				2B00	1916	FZXPIB EQU *			START OF PROGRAM CODING
	2A01				1917	ORG *-255			RESET IAR TO PAGE
	2B00				1918	ORG *,256,0			* BOUNDARY ADDRESS
				2B00	1919	USING *,@BR			SET PAGE RASE ADDRESS
	2B00				1920	ORG FZXPIB			RESET STARTING ADDRESS
					1921	*** END OF EXPANSION ***			
					1922	*			
					1923	*****			

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 204
					1925		*****	
					1926	*	INPUT ENTRY 1 - PERMIT DATA INPUT VIA KEYBOARD AND CHECK SYNTAX	*
					1927		*****	
					1928	*		
				2B00	1929	FZXIP1 EQU *	FZXINP INPUT/CHECKING ENTRY PT	
					1930	*		
					1931	*	LOCK THE KEYBOARD INPUT BUFFER INTO CORE VIRTUAL MEMORY	
					1932	*		
	2B00	1C	01	144A	91	1933	MVC I\$VADR,FZXBVA(@VADDR,@BR)	SET PAGING VADDR PARAM FOR BFR
	2B05	C0	87	1354		1934	B I\$LOCK	LINK TO CORELOAD AND LOCK BUFFER
	2B09	0C	01	0DC8	144C	1935	MVC FZXBCA,I\$CADDR(@CADDR)	SAVE BUFFER CADDR IN FIXED AREA
	2B0F	C0	87	0E24		1936	B I\$I700	CHECK LINE PRINTER BUFFER
					1937	*		
					1938	*	INITIALIZE THE INPUT BUFFER TO BLANKS	
					1939	*		
	2B13	35	02	0DC8		1940	L FZXBCA,@XR	LOAD THE INPUT BUFFER CADDR
	2B17	BC	40	FF		1941	MVI B@LVPG-1(,@XR),B@BLNK	INITIALIZE THE KEYBOARD INPUT
	2B1A	AC	FE	FE	FF	1942	MVC B@LVPG-2(,@XR),B@LVPG-1(B@LVPG-1,@XR)	* BUFFER TO BLANKS
					1943	*		
					1944	*	PRINT '?' TO REQUEST DATA INPUT VIA SYSTEM KEYBOARD	
					1945	*		
	2B1E	C0	87	12B1		1946	FZX010 B I\$CALL	LINK TO RETURN CARRIER AND
	2B22	2C00			2B23	1947	DC AL(@VADDR)(FZX050)	* PRINT SINGLE QUESTION MARK
					1948	*		
	2B24	3B	1E	03E4		1949	SBF \$LPRP3,@KENAB	RESET MATRIX PRINT IND. 1-3
					1950	*		
					1951	*	EXECUTE THE KEYBOARD INPUT ROUTINE	
					1952	*		
	2B28	35	02	0DC8		1953	L FZXBCA,@XR	LOAD THE INPUT BUFFER CADDR
	2B2C	C0	87	12B1		1954	B I\$CALL	LINK TO EXECUTE KEYBOARD INPUT
	2B30	2500			2B31	1955	DC AL(@VADDR)(V\$SKEY)	VADDR OF KEYBOARD INPUT IOCR
					1956	*		
					1957	*	TEST FOR CONSOLE INTERRUPT DURING KEYBOARD INPUT - RE-EXECUTE	
					1958	*	CURRENT STATEMENT 'STH' INSTRUCTION WHEN INTERRUPT HAS BEEN INVOKED	
					1959	*		
	2B32	3D	80	0496		1960	CLI \$CISUS,@NOP	IF NO PENDING CONSOLE INTERRUPT
	2B36	F2	01	1E		1961	JNE FZX020	* BRANCH TO SYNTAX CHECK INPUT
					1962	*		
	2B39	1F	00	0D4E	8F	1963	SLC I\$STAK,FZXLVA(@CADDR-1,@BR)	DECR STACK PT PAST BR VADDR
	2B3E	0C	01	0D59	0D51	1964	MVC I\$WRK1,I\$STHA(@VADDR)	SET BRANCH VADDR FOR LAST 'STH'
	2B44	3C	80	0CE9		1965	MVI I\$RESW,@NOP	DISABLE STMT NO. RECURSION ERR
	2B48	1C	01	130E	56	1966	MVC I\$MOD4+@OP1,FZXPSA(@CADDR,@BR)	RE-INITLZ PAGE STCK PT 1-4
	2B4D	C2	01	0C60		1967	LA I\$BASE,@BR	LOAD THE INTERPRETER BASE CADDR
	2B51	C0	87	119D		1968	B I\$BSET	EXIT TO RE-EXECUTE 'STH' PMC
	2B55	15CB			2B56	1969	FZXPSA DC AL(@CADDR)(I\$PSTK+1)	PASS LINK STACK PT INITLZN
					1970	*		
					1971	*	SYNTAX CHECK THE ENTIRE INPUT LINE - CHECK INPUT LINE DATA ELEMENTS	
					1972	*	FOR CORRESPONDENCE WITH PROGRAM INPUT STATEMENT VARIABLE DATA LIST	
					1973	*		
	2B57	C0	87	12B1		1974	FZX020 B I\$CALL	LINK TO SYNTAX CHECK INPUT LINE
	2B5B	2E20			2B5C	1975	DC AL(@VADDR)(FZX270)	VADDR OF 'INPUT' SYNTAX CHECKER
					1976	*		
					1977	*	TEST FOR A DISCOVERED SYNTAX OR DATA TYPE ERROR - ON ERROR CONDITION	
					1978	*	PRINT APPROPRIATE MESSAGE AND REPEAT KEYBOARD DATA INPUT ROUTINE	
					1979	*		
	2B5D	3D	00	0CBC		1980	CLI I\$ERRC,I@NERR	IF NO DATA SYNTAX/TYPE ERROR

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC		OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 205
2B61	C0	81 12D3		1981	BE	I\$RTRN			* EXIT TO THE CALLING PROGRAM
				1982	*				
2B65	C0	87 12B1		1983	B	I\$CALL			LINK TO RETURN CARRIER AND
2B69	2C18		2B6A	1984	DC	AL(@VADDR)(FZX080)			* PRINT SPECIFIED ERROR MESSAGE
				1985	*				
2B6B	D0	87 1E		1986	B	FZX010(,@BR)			GO REPEAT DATA LINE INPUT
				1987	*				* AND SYNTAX/TYPE ERROR CHECK
				1988	*				
				1989	*****				

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 206
					1991	*****				
					1992	* INPUT ENTRY 2 - CONVERT AND STACK SINGLE INPUT LINE DATA ELEMENT	*			
					1993	*****				
					1994	*				
				2B6E	1995	FZXIP2 EQU	*		FZXINP CONVERSION ENTRY POINT	
					1996	*				
					1997	* CONVERT NEXT AVAILABLE INPUT LINE ELEMENT AND MOVE TO RUN-TIME STACK				
					1998	*				
	2B6E	35	02	0DC8	1999	L	FZXBCA,@XR		LOAD THE INPUT BUFFER CADDR	
					2000	*				
	2B72	C0	87	12B1	2001	B	I\$CALL		LINK TO CONVERT AND STACK NEXT	
	2B76	3100			2002	DC	AL(@VADDR)(FZX860)		* DATA ELEMENT IN INPUT BUFFER	
					2003	*				
					2004	* TEST FOR FINAL ELEMENT IN INPUT BUFFER - RELEASE BUFFER PAGE				
					2005	* FROM CORE WHEN END OF INPUT LINE IS ENCOUNTERED				
					2006	*				
	2B78	BD	1E	00	2007	CLI	B@CHAR(,@XR),B@EOST		IF ELEMENT DELIMITER NOT EOS	
	2B7B	F2	01	0D	2008	JNE	FZX030		* GO EXIT LEAVING BUFFER LOCKED	
					2009	*				
	2B7E	1C	01	144A 91	2010	MVC	I\$VADR,FZXBVA(@VADDR,@BR)		SET PAGING VADDR PARAM FOR BFR	
	2B83	C0	87	1350	2011	B	I\$UNLK		LINK TO UNLOCK BUFFER FROM CORE	
					2012	*				
	2B87	3B	1E	03E4	2013	SBF	\$LPRP3,@KENAB		RESET MATRIX PRINT IND.	1-3
	2B8B	C0	87	12D3	2014	FZX030 B	I\$RTRN		EXIT TO THE CALLING PROGRAM	
					2016	*****				
					2017	* INPUT EXECUTION ROUTINE CONSTANTS (1ST VM PAGE)				
					2018	*****				
					2019	*				
	2B8F	02			2020	FZXLVA DC	AL(@CADDR-1)(@VADDR)		LENGTH OF A VIRTUAL ADDRESS	
	2B90	5400			2021	FZXBVA DC	AL(@VADDR)(V\$BFR1)		KEYBOARD INPUT BUFFER VADDR	
					2022	*				
					2023	*****				

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 207
		2025		*****	
		2026	*	VIRTUAL MEMORY INPUT EXECUTION ROUTINE 2ND VM PAGE -	*
		2027	*	PERFORMS FOLLOWING ACTIVITIES DEPENDING ON ENTRY POINT	*
		2028	*		*
		2029	*	SINGLE QUESTION MARK ENTRY -	*
		2030	*	* RETURNS CARRIER ON SYSTEM PRINT DEVICE	*
		2031	*	* PRINTS '?' AND RETURNS THE CARRIER	*
		2032	*		*
		2033	*	DOUBLE QUESTION MARK ENTRY -	*
		2034	*	* RETURNS CARRIER ON SYSTEM PRINT DEVICE	*
		2035	*	* PRINTS '??' AND RETURNS THE CARRIER	*
		2036	*		*
		2037	*	ERROR MESSAGE ENTRY -	*
		2038	*	* RETURNS CARRIER ON SYSTEM PRINT DEVICE	*
		2039	*	* PRINTS ERROR MESSAGE INDICATED BY INTEGER VALUE IN I\$ERRC	*
		2040	*	(I\$ERRC = DECIMAL 0,... 4 TO SPECIFY MESSAGE 800,... 804)	*
		2041	*	* RETURNS PRINT DEVICE CARRIER AFTER ERROR MESSAGE OUTPUT	*
		2042	*	* RESTORES ERROR CODE I\$ERRC TO NULL STATUS	*
		2043		*****	
		2044	*		
		2045	*	ESTABLISH ADDRESSABILITY FOR INPUT ROUTINE 2ND VM PAGE	
		2046	*		
2C00		2047	*FZXP2B	VPAGE 0 SET 2ND PAGE ADDRESSASILIFI	
		2048	ORG	*,256,0 SET STARTING ADDRESS	
2B01	2C00	2049	FZXP2B	EQU * START OF PROGRAM CODING	
2C00		2050	ORG	*-255 RESET IAR TO PAGE	
		2051	ORG	*,256,0 * BOUNDARY ADDRESS	
	2C00	2052	USING	*,@BR SET PAGE BASE ADDRESS	
2C00		2053	ORG	FZXP2B RESET STARTING ADDRESS	
		2054	***	END OF EXPANSION ***	
		2056	2D00	USING FZXP3B,@XR SET ERROR MESSAGE ADDRESSIBILITY	
		2057	*		
		2058		*****	

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 208
		2060		*****	
		2061		* ENTRY FOR SINGLE QUESTION MARK OUTPUT TO PRINTER	*
		2062		*****	
		2063		*	
		2C00	2064	FZXPQ1 EQU *	PRINT (?) ENTRY POINT
		2065		*	
2C00 D2 02 E2		2066	FZX050 LA	FZXQM1(, @BR), @XR	LOAD SINGLE QUESTION MARK CADDR
2C03 F2 87 03		2067	J	FZX070	BRANCH TO PRINT QUESTION MARK
		2069		*****	
		2070		* ENTRY FOR DOUBLE QUESTION MARK OUTPUT TO PRINTER	*
		2071		*****	
		2072		*	
		2C06	2073	FZXPQ2 EQU *	PRINT (??) ENTRY POINT
		2074		*	
2C06 D2 02 E1		2075	FZX060 LA	FZXQM2(, @BR), @XR	LOAD DOUBLE QUESTION MARK CADDR
		2076		*	
2C09 74 02 F2		2077	FZX070 ST	FZXPRP+@PDATA(, @BR), @XR	STORE OUTPUT FIELD CADDR IN PPL
		2078		*	
		2079		* RETURN PRINTER CARRIER AND PRINT SPECIFIED QUESTION MARK(S)	
		2080		*	
2C0C D0 87 A1		2081	B	FZX150(, @BR)	LINK TO RETURN PRINTER CARRIER
		2082		*	
2C0F D2 02 EF		2083	LA	FZXPRP(, @BR), @XR	LOAD PRINT & RETURN PPL CADDR
2C12 D0 87 A8		2084	B	FZX170(, @BR)	LINK TO PRINT QUESTION MARK(S)
		2085		*	
2C15 F2 87 85		2086	J	FZX140	BRANCH TO EXIT PRINT ROUTINE
		2087		*	
		2088		*****	

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 209
					2090		*****	
					2091		* ENTRY FOR ERROR MESSAGE OUTPUT TO PRINTER *	
					2092		*****	
					2093		*	
				2C18	2094	FZXPEM EQU *	PRINT MESSAGE ENTRY POINT	
					2095		*	
					2096		* CONVERT BINARY LINE NUMBER TO ZONED DECIMAL FOR OUTPUT	
					2097		*	
	2C18	4C	01 E5 03CF		2098	FZX080 MVC	FZXBLN(, @BR), \$INLNO(B@LSNO) MOVE BINARY LINE NO. TO WORK	
	2C1D	54	30 ED DB		2099	ZAZ	FZXDLN(B@LDSN, @BR), FZX2D0(1, @BR) INITLZ DECIMAL LINE NO.	
	2C21	54	30 E9 DC		2100	ZAZ	FZXDAC(B@LDSN, @BR), FZX2D1(1, @BR) SET DECML ACCUM EQUAL 1	
	2C25	7C	01 29		2101	MVI	FZX090+@Q(, @BR), @B1 SET BINARY MASK FOR 2**0 BIT	
	2C28	78	00 E5		2102	FZX090 TBN	FZXBLN(, @BR), *- * TEST BINARY LINE NUMBER BIT	
	2C2B	F2	90 04		2103	JF	FZX100 * AND BRANCH IF BIT IS ZERO	
	2C2E	56	03 ED E9		2104	AZ	FZXDLN(B@LDSN, @BR), FZXDAC(B@LDSN, @BR) INCR DECIMAL NUMBER	
	2C32	5E	00 29 29		2105	FZX100 ALC	FZX090+@Q(, @BR), FZX090+@Q(1, @BR) SHIFT BINARY MASK LEFT	
	2C36	F2	20 07		2106	JNOL	FZX110 BRANCH UNLESS MASK EXCEEDS 2**7	
	2C39	5C	00 E5 E4		2107	MVC	FZXBLN(, @BR), FZXBLN-1(1, @BR) MOVE HIGH ORDER BYTE TO LOW	
	2C3D	7C	01 29		2108	MVI	FZX090+@Q(, @BR), @B1 SET BINARY MASK FOR 2**8 BIT	
	2C40	56	03 E9 E9		2109	FZX110 AZ	FZXDAC(B@LDSN, @BR), FZXDAC(B@LDSN, @BR) DOUBLE DECML ACCUM	
	2C44	D0	08 28		2110	BNOZ	FZX090(, @BR) REPEAT LOOP UNTIL ACCLM > 8192	
					2111		*	
					2112		* RETURN PRINTER CARRIER FOR ERROR MESSAGE OUTPUT	
					2113		*	
	2C47	D0	87 A1		2114	B	FZX150(, @BR) LINK TO RETURN PRINTER CARRIER	
					2115		*	
					2116		* ACCESS THE INPUT ROUTINE ERROR MESSAGES	
					2117		*	
	2C4A	1C	01 144A E0		2118	MVC	I\$VADR, FZXEVA(@VADDR, @BR) SET PAGING VADDR PARAMETER	
	2C4F	C0	87 1354		2119	B	I\$LOCK LINK TO READ AND LOCK MESSAGES	
	2C53	35	02 144C		2120	L	I\$CADR, @XR LOAD MESSAGE PAGE BASE CADDR	
	2C57	74	02 69		2121	ST	FZX120+@OP1(, @BR), @XR SAVE MESSAGE PAGE BASE CADDR	
					2122		*	
					2123		* PRINT ERROR NUMBER SEGMENT OF THE ERROR MESSAGE	
					2124		*	
	2C5A	E2	02 00		2125	LA	@M250(, @XR), @XR LOAD ERROR NO. SEGMENT PPL CADDR	
	2C5D	D0	87 A4		2126	B	FZX160(, @BR) LINK TO PRINT ERROR NO. SEGMENT	
					2127		*	
	2C60	D2	02 F7		2128	LA	FZXPNP(, @BR), @XR LOAD ERROR NO. DIGIT PPL CADDR	
	2C63	D0	87 A8		2129	B	FZX170(, @BR) LINK TO PRINT DIGIT IN I@ERRC	
					2130		*	
					2131		* PRINT LINE NUMBER SEGMENT OF THE ERROR MESSAGE	
					2132		*	
	2C66	C2	02 0000		2133	FZX120 LA	*-*, @XR LOAD MESSAGE PAGE BASE CADDR	
	2C6A	E2	02 04		2134	LA	@M251(, @XR), @XR LOAD LINE NO. SEGMENT CADDR	
	2C6D	D0	87 A4		2135	B	FZX160(, @BR) LINK TO PRINT LINE NO. SEGMENT	
					2136		*	
	2C70	D2	02 EA		2137	LA	FZXELN(, @BR), @XR LOAD DECIMAL LINE NO. CADDR	
	2C73	74	02 F6		2138	ST	FZXPS+@PDATA(, @BR), @XR STORE LINE NO. CORE ADDR IN PPL	
	2C76	D2	02 F3		2139	LA	FZXPS(, @BR), @XR LOAD PRINT & STOP PPL CORE ADDR	
	2C79	D0	87 A8		2140	B	FZX170(, @BR) LINK TO PRINT ERROR LINE NUMBER	
					2141		*	
					2142		* PRINT ERROR MESSAGE SPECIFIED BY DIGIT IN ERROR CODE I\$ERRC -	
					2143		* RETURN CARRIER TOLLOWING ERROR MESSASE OUTPUT	
					2144		*	
	2C7C	75	02 69		2145	L	FZX120+@OP1(, @BR), @XR LOAD MESSAGE PAGE BASE ADDR	

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC		OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 210
2C7F	E2	02 04	2146		LA	@@M256-@PPLNG(,@XR),@XR	INITLZ ERROR MESSAGE PPL POINTER
2C82	E2	02 04	2147	FZX130	LA	@PPLNG(,@XR),@XR	INCREMENT ERROR MESSAGE PPL PT
2C85	17	00 0CBC DC	2148		SZ	I\$ERRC(1),FZX2D1(1,@BR)	DECK ERROR MESSAGE DIGIT CODE
2C8A	D0	02 82	2149		BNM	FZX130(,@BR)	REPEAT LOOP UNTIL CGG_JS <
2C8D	D0	87 A4	2150		B	FZX160(,@BR)	LINK TO PRINT ERROR MESSAGE
			2151	*			
2C90	3C	00 0CBC	2152		MVI	I\$ERRC,I@NERR	RESET ERROR CODE TO_ LL STATUS
			2153	*			
			2154	*	RELEASE THE ERROR MESSAGE PAGE FROM CORE VIRTUAL MEMORY		
			2155	*			
2C94	1C	01 144A E0	2156		MVC	I\$VADR,FZXEVA(@VADDR,@BR)	SET PAGING VADDR PARAMETER
2C99	C0	87 1350	2157		B	I\$UNLK	LINK TO UNLOCK MESSAGE PAGE
			2158	*			
			2159	*	ERROR MESSAGE ROUTINE EXIT - RETURN TO CALLING ROUTINE		
			2160	*			
2C9D	C0	87 12D3	2161	FZX140	B	I\$RTRN	RETURN TO CALLING ROUTINE
			2162	*			
			2163	*****			

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 211

```

2165 *****
2166 * PRINTER/CRT MESSAGE OUTPUT ROUTINE *
2167 * * PRINTS SPECIFIED MESSAGE AND/OR CONTROLS CARRIER FOR MATRIX *
2168 * PRINTER AND/OR CRT AS DEFINED AT SYSTEM DEVICE PARAMETER *
2169 * *
2170 * INPUT - *
2171 * * REGISTER @XR - CONTAINS CORE ADDRESS OF PRINT PARAMETER LIST *
2172 * *
2173 * OUTPUT - *
2174 * * PRINTED LINE AND/OR CARRIER CONTROL AS SPECIFIED IN PARAM LIST *
2175 *****
2176 *
2177 * SPECIAL ENTRY 1 - SET MESSAGE OUTPUT ROUTINE TO PERFORM CARR RETURN
2178 *
2CA1 D2 02 FB 2179 FZX150 LA FZXCRP(,@BR),@XR LOAD CARRIER RETURN PPL CADDR
2180 *
2181 * SPECIAL ENTRY 2 - MODIFY PRINT FIELD CADDR IN PPL FOR RELOCATION
2182 *
2CA4 9C 00 02 68 2183 FZX160 MVC @PDATA-1(,@XR),FZXECA(1,@BR) ADJUST ERROR MESSAGE CADDR
2184 * * IN PPL FOR CURR CORE LOCATION
2185 *
2186 * NORMAL PRINT ROUTINE ENTRY - SAVE THE RETURN ADDRESS
2187 *
2CA8 74 08 DA 2188 FZX170 ST FZX210+@OP1(,@BR),@ARR SAVE RETURN BRANCH ADDRESS
2189 *
2190 * DETERMINE POSSIBLE CORE ENTRY ADDRESS FOR THE CRT IOCR
2191 *
2192 MVC FZX180+@OP1(,@BR),FZXPPA(@CADDR,@BR) SET UP BASE CADDR
2CA1 5C 01 C2 DE 2193 ALC FZX180+@OP1-1(,@BR),$EXFTR(1) * AND ADD EXTENSION FACTOR
2CAF 4E 00 C1 043B 2194 *
2195 * TEST FOR TYPE OF PRINT DEVICE ACTIVE ON SYSTEM
2196 *
2CB4 1D 00 044A C1 2197 CLC $PRDEV-1,FZX180+@OP1-1(,@BR) TEST PRINT DEVICE PARAMETER
2CB9 F2 82 11 2198 JL FZX200 * AND BRANCH IF PRINTER ONLY
2199 *
2200 * CRT (AND POSSIBLY PRINTER) ACTIVE - OUTPUT AREA AREA ON THE CRT
2201 *
2CBC 74 02 C4 2202 ST FZX190(,@BR),@XR STORE PRINT PARAM LIST CADDR
2203 *
2CBF C0 87 0000 2204 FZX180 B *-* LINK TO EXECUTE THE CRT IOCR
2CC3 2CC4 2205 FZX190 DS CL(@CADDR) PRINT PARAMETER LIST CADDR
2206 *
2207 * TEST FOR MATRIX PRINTER ACTIVE ON THE SYSTEM
2208 *
2CC5 1D 01 044B C2 2209 CLC $PRDEV,FZX180+@OP1(@CADDR,@BR) TEST PRINT DEVICE PARAM
2CCA F2 02 0A 2210 JNL FZX210 * AND BRANCH IF CRT ONLY
2211 *
2212 * MATRIX PRINTER ACTIVE - OUTPUT PRINT AREA ON THE MATRIX PRINTER
2213 *
2CCD 3A 1E 03E4 2214 FZX200 SBN $LPRP3,@KENAB SET MATRIX PRINT IND. 1-3
2CD1 C0 87 12B1 2215 B ISCALL GO TO MATRIX PRINTER ROUTINE
2CD5 2800 2CD6 2216 DC AL(@VADDR)(V$SPRT) MATRIX PRINTER IOCR VADDR
2217 *
2218 * RETURN CONTROL TO THE MESSAGE OUTPUT ROUTINE
2219 *
2CD7 C0 87 0000 2220 FZX210 B *-* RETURN TO CALLING ROUTINE

```

[illegible]

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 213
		2224		*****	
		2225		* INPUT EXECUTION ROUTINE CONSTANTS (2ND VM PAGE)	*
		2226		*****	
		2227		*	
2CDB F0		2CDB 2228	FZX2D0 DC	DL1 '0'	DECIMAL INTEGER 0
2CDC F1		2CDC 2229	FZX2D1 DC	DL1 '1'	DECIMAL INTEGER +1
		2230		*	
2CDD 2004		2CDE 2231	FZXPDA DC	AL (@CADDR) (I\$CSXA+4)	CRT CORE ENTRY BASE ADDRESS
		2232		*	
2CDF 2D00		2CE0 2233	FZXEVA DC	AL (@VADDR) (FZXP3B)	ERROR MESSAGE PAGE BASE VADDR
		2C68 2234	FZXECA EQU	FZX120+@OP1-1	ERROR MESSAGE PAGE BASE CADDR
		2235		*	
2CE1 6F		2CE1 2236	FZXQM2 EQU	*	DOUBLE QUESTION MARK FIELD CADDR
		2CE1 2237	DC	CL1 '?'	QUESTION MARK FIELD CONSTANT
		2CE2 2238	FZXQM1 EQU	*	SINGLE QUESTION MARK FIELD CADDR
2CE2 6F40		2CE3 2239	DC	CL2 '?'	QUESTION MARK FIELD CONSTANT
		0002 2240	FZXQML EQU	2	LENGTH OF QUESTION MARK FIELD
		2241		*	
		2242		*****	

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 214
					2244	*****	*****			
					2245	* INPUT EXECUTION ROUTINE WORK AREAS (2ND VM PAGE)	*			
					2246	*****	*****			
					2247	*				
2CE4				2CE5	2248	FZXBLN DS	CL(B@LSNO)			BINARY LINE NUMBER WORK AREA
2CE6				2CE9	2249	FZXDAC DS	CL(B@LDSN)			DECIMAL CONVERSION ACCUMULATOR
					2250	*				
				2CEA	2251	FZXELN EQU	*			LINE NUMBER OUTPUT FIELD CADDR
2CEA				2CED	2252	FZXDLN DS	CL(B@LDSN)			LINE NUMBER OUTPUT WORK AREA
2CEE	40			2CEE	2253		DC CL1 ' '			OUTPUT FIELD BLANK CHARACTER
					2254	*				
					2255	*FZXPRP PPL	FUNC=@PRETR,CNT=FZXQML			PRINT '? (?)' AND RETURN PPL
				2CEF	2256	FZXPRP EQU	*			PPL ADDRESS
2CEF	C0			2CEF	2257		DC AL1(@PRETR)			FUNCTION REQUESTED
2CF0	02			2CF0	2258		DC AL1(FZXQML)			PRINT COUNT
2CF1	0000			2CF2	2259		DC AL2(*-*)			DATA ADDRESS
					2260	*** END OF EXPANSION ***				
					2262	*FZXPPSP PPL	FUNC=@PRINT,CNT=B@LDSN+1			PRINT LINE NO. AND STOP PPL
				2CF3	2263	FZXPPSP EQU	*			PPL ADDRESS
2CF3	40			2CF3	2264		DC AL1(@PRINT)			FUNCTION REQUESTED
2CF4	05			2CF4	2265		DC AL1(B@LDSN+1)			PRINT COUNT
2CF5	0000			2CF6	2266		DC AL2(*-*)			DATA ADDRESS
					2267	*** END OF EXPANSION ***				
					2269	*FZXPNP PPL	FUNC=@PRINT,CNT=1,CADDR=I\$ERRC			PRINT ERROR DIGIT PPL
				2CF7	2270	FZXPNP EQU	*			PPL ADDRESS
2CF7	40			2CF7	2271		DC AL1(@PRINT)			FUNCTION REQUESTED
2CF8	01			2CF8	2272		DC AL1(1)			PRINT COUNT
2CF9	0CBC			2CFA	2273		DC AL2(I\$ERRC)			DATA ADDRESS
					2274	*** END OF EXPANSION ***				
					2276	*FZXCRP PPL	FUNC=@RETRN,CNT=@RTRNC			CARRIER RETURN PARA4ETER LIST
				2CFB	2277	FZXCRP EQU	*			PPL ADDRESS
2CFB	80			2CFB	2278		DC AL1(@RETRN)			FUNCTION REQUESTED
2CFC	80			2CFC	2279		DC AL1(@RTRNC)			PRINT COUNT
2CFD	0000			2CFE	2280		DC AL2(*-*)			DATA ADDRESS
					2281	*** END OF EXPANSION ***				
					2282	*				
					2283	*****	*****			

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 215
					2285		*****	
					2286		* VIRTUAL MEMORY INPUT EXECUTION ROUTINE (3RD VM PAGE)	*
					2287		* CONTAINS ERROR MESSAGE TEXT AND PRINT PARAMETER LISTS	*
					2288		*****	
					2289		*	
					2290		* ESTABLISH ADDRESSABILITY FOR INPUT ROUTINE 3RD VM PACE	
					2291		*	
					2292	*FZXP3B	VPAGE 0 SET 3RD PAGE ADDRESSABILITY	
	2D00				2293	ORG	*,256,0 SET STARTING ADDRESS	
				2D00	2294	FZXP3B	EQU * START OF PROGRAM CODING	
	2C01				2295	ORG	*-255 RESET IAR TO PAGE	
	2D00				2296	ORG	*,256,0 * BOUNDARY ADDRESS	
				2D00	2297	USING	*,@BR SET PAGE BASE ADDRESS	
	2D00				2298	ORG	FZXP3B RESET STARTING ADDRESS	
					2299	***	END OF EXPANSION ***	
					2301		*****	
					2302		* INPUT EXECUTION ERROR MESSAGE PARAMETERS	*
					2303		*****	
					2304		*	
					2305	*	MTEXT @@M250-@PRINT,	
					2306	*	@@M251-@PRINT,	
					2307	*	@@M256-@PRETR,	
					2308	*	@@M257-@PRETR,	
					2309	*	@@M258-@PRETR,	
					2310	*	@@M259-@PRETR,	
					2311	*	@@M260-@PRETR,	
					2312	*	PATCH-040	
					2313		*****	
					2314	*	PPL'S AND TEXT FOR MESSAGE	*
					2315		*****	
	2D00 40			2D00	2316	@M250 DC	AL1 (@PRINT) PRINT CONTROL FUNCTION	
	2D01 08			2D01	2317	DC	IL1 '08' LENGTH OF MESSAGE	
	2D02 2D1C			2D03	2318	DC	AL (@CADDR) (@@T250) ADDR OF MESSAGE	
					2319	*		
	2D04 40			2D04	2320	@M251 DC	AL1 (@PRINT) PRINT CONTROL FUNCTION	
	2D05 09			2D05	2321	DC	IL1 '09' LENGTH OF MESSAGE	
	2D06 2D24			2D07	2322	DC	AL (@CADDR) (@@T251) ADDR OF MESSAGE	
					2323	*		
	2D08 C0			2D08	2324	@M256 DC	AL1 (@PRETR) PRINT CONTROL FUNCTION	
	2D09 25			2D09	2325	DC	IL1 '37' LENGTH OF MESSAGE	
	2D0A 2D2D			2D0B	2326	DC	AL (@CADDR) (@@T256) ADDR OF MESSAGE	
					2327	*		
	2D0C C0			2D0C	2328	@M257 DC	AL1 (@PRETR) PRINT CONTROL FUNCTION	
	2D0D 23			2D0D	2329	DC	IL1 '35' LENGTH OF MESSAGE	
	2D0E 2D52			2D0F	2330	DC	AL (@CADDR) (@@T257) ADDR OF MESSAGE	
					2331	*		
	2D10 C0			2D10	2332	@M258 DC	AL1 (@PRETR) PRINT CONTROL FUNCTION	
	2D11 17			2D11	2333	DC	IL1 '23' LENGTH OF MESSAGE	
	2D12 2D75			2D13	2334	DC	AL (@CADDR) (@@T258) ADDR OF MESSAGE	
					2335	*		
	2D14 C0			2D14	2336	@M259 DC	AL1 (@PRETR) PRINT CONTROL FUNCTION	
	2D15 20			2D15	2337	DC	IL1 '32' LENGTH OF MESSAGE	
	2D16 2D8C			2D17	2338	DC	AL (@CADDR) (@@T259) ADDR OF MESSAGE	
					2339	*		
	2D18 C0			2D18	2340	@M260 DC	AL1 (@PRETR) PRINT CONTROL FUNCTION	

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC		OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00		31/05/21	PAGE 216
2D19	25		2D19	2341		DC	IL1'37'		LENGTH OF MESSAGE	
2D1A	2DAC		2D1B	2342		DC	AL(@CADDR)(@@T260)		ADDR OF MESSAGE	
				2343	*					
			2D1C	2344	@@T250	EQU	*		LEFT BYTE OF MESSAGE	
2D1C	C5D9D9D6D940F8F0		2D23	2345		DC	CL008'ERROR 80'			
			2D24	2346	@@T251	EQU	*		LEFT BYTE OF MESSAGE	
2D24	40C1E340D3C9D5C5		2D2C	2347		DC	CL009' AT LINE '			
			2D2D	2348	@@T256	EQU	*		LEFT BYTE OF MESSAGE	
2D2D	C9D5E5C1D3C9C440		2D51	2349		DC	CL037'INVALID INPUT DATA - NUMERIC CONSTANT'			
			2D52	2350	@@T257	EQU	*		LEFT BYTE OF MESSAGE	
2D52	C9D5E5C1D3C9C440		2D74	2351		DC	CL035'INVALID INPUT DATA - CHARACTER DATA'			
			2D75	2352	@@T258	EQU	*		LEFT BYTE OF MESSAGE	
2D75	E3D6D640D4C1D5E8		2D8B	2353		DC	CL023'TOO MANY INPUT ELEMENTS'			
			2D8C	2354	@@T259	EQU	*		LEFT BYTE OF MESSAGE	
2D8C	D5D6E340C5D5D6E4		2DAB	2355		DC	CL032'NOT ENOUGH DATA ELEMENTS ENTERED'			
			2DAC	2356	@@T260	EQU	*		LEFT BYTE OF MESSAGE	
2DAC	D5D6E340C5D5D6E4		2DD0	2357		DC	CL037'NOT ENOUGH ARRAY ROW ELEMENTS ENTERED'			
				2358	*					
				2359	* PATCH AREA FOR MESSAGES					
				2360	*					
2DD1			2DF8	2361	\$\$\$001	DS	CL040		MSG EXPANSION PATCH AREA	

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 217
		2363		*****	
		2364	*	VIRTUAL MEMORY INPUT EXECUTION ROUTINE (4TH VM PAGE)	*
		2365	*	SYNTAX CHECKING ROUTINE FOR THE FOLLOWING DATA INPUT MODES -	*
		2366	*	* INPUT - CHECKS SYNTAX AND SEQUENCE OF KEYBOARD INPUT LINE DATA	*
		2367	*	* MAT INPUT - CHECKS SYNTAX AND TYPE OF ARRAY ROW INPUT LINE DATA	*
		2368	*	* GET(CARD) - CHECKS SYNTAX OF DATA READ FROM A CARD DATA FILE	*
		2369	*		*
		2370	*	INPUT -	*
		2371	*	* INPUT, MAT INPUT -	*
		2372	*	* I\$STAK - REFERENCES LEFTMOST BYTE OF STACKED DATA TYPE	*
		2373	*	SPECIFICATION CODES	*
		2374	*	* I\$PARM - CONTAINS NUMBER OF STACKED SPECIFICATION CODES	*
		2375	*	* INPUT, MAT INPUT, GET(CARD) -	*
		2376	*	* REGISTER @XR - REFERENCES LEFTMOST BYTE OF INPUT LINE BFR	*
		2377	*		*
		2378	*	OUTPUT -	*
		2379	*	* I\$PARM-1 - CONTAINS INDEX NO OF LAST PROCESSED DATA ELEMENT	*
		2380	*	* I\$ERRC - CONTAINS CODE I@NERR IF NO SYNTAX ERROR ENCOUNTERED	*
		2381	*	- CONTAINS DECIMAL DIGIT 0,1,... 4 (CORRESPONDING TO ERROR	*
		2382	*	MESSAGE 800,801,... 804) FOR INPUT, MAT INPUT ERROR	*
		2383	*	- CONTAINS CODE X'FF' FOR GET(CARD) SYNTAX ERROR	*
		2384	*	* REGISTER @XR - REFERENCES LAST SCANNED INPUT BUFFER CHARACTER	*
		2385	*	* INPUT BUFFER - LAST 2 BYTES IN BUFFER PAGE INITIALIZED TO	*
		2386	*	(CADDR-1) WHERE CADDR IS ADDRESS OF INPUT BUFFER LEFT BYTE	*
		2387	*	*****	
		2388	*		
		2389	*	ESTABLISH ADDRESSABILITY FOR INPUT ROUTINE 4TH VM PAGE	
		2390	*		
2E00		2391	*FZXP4B	VPAGE 0 SET 3RD PAGE ADDRESSABILITY	
		2392	ORG	*,256,0 SET STARTING ADDRESS	
2D01	2E00	2393	FZXP4B	EQU * START OF PROGRAM CODING	
2E00		2394	ORG	*-255 RESET IAR TO PAGE	
		2395	ORG	*,256,0 * BOUNDARY ADDRESS	
	2E00	2396	USING	*,@BR SET PAGE BASE ADDRESS	
2E00		2397	ORG	FZXP4B RESET STARTING ADDRESS	
		2398	***	END OF EXPANSION ***	
		2399	*		
		2400	*	*****	

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 218
		2402		*****	
		2403		* ENTRY FOR GET(CARD) DATA SYNTAX CHECKING	*
		2404		*****	
		2405		*	
	2E00	2406	FZXGCS EQU	*	GET(CARD) SYNTAX CHECKER ENTRY
		2407		*	
		2408		* INITIALIZE SYNTAX CHECKER FOR GET(CARD) MODE	
		2409		*	
2E00	3C 80 0D58	2410	FZX250 MVI	FZXETS,@NOP	DISABLE ELEMENT TYPE CHECKING
2E04	3C FF 0CBC	2411		MVI ISERRC,FZXSEC	SET CONTINGENCY ERROR CODE
2E08	7C 87 6E	2412		MVI FZX320+@Q(,@BR),@UCB	DISALLOW BUFFER TERMINAL COMMA
2E0B	7C 87 94	2413		MVI FZX340+@Q(,@BR),@UCB	DISABLE ELEMENT COUNT CHECKING
2E0E	7C 87 B3	2414		MVI FZX390+@Q(,@BR),@UCB	DISABLE ELEMENT TYPE PROCESSING
2E11	7C 80 9B	2415		MVI FZX370+@Q(,@BR),@NOP	FORCE GET(CARD) ERROR EXIT 1-3
2E14	F2 87 19	2416		J FZX290	BRANCH TO CONTINUE INITIAL1ZING
		2417		*	
		2418		*****	

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC		OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 219	
				2420	*****		
				2421	* ENTRY FOR MAT INPUT DATA SYNTAX, TYPE, COUNT CHECKING	*	
				2422	*****		
				2423	*		
			2E17	2424	FZXMIS EQU *	MAT INPUT SYNTAX CHECKER ENTRY	
				2425	*		
				2426	* INITIALIZE SYNTAX CHECKER FOR MAT INPUT MODE		
				2427	*		
2E17	7C	80 6E		2428	FZX260 MVI FZX320+@Q(,@BR),@NOP	ALLOW BUFFER TERMINAL COMMA	
2E1A	7C	F4 97		2429	MVI FZX350+@Q(,@BR),FZXER4	SET ERROR 804 FOR TOO FEW DATA	
2E1D	F2	87 06		2430	J FZX280	SKIP TO CONTINUE INITIALIZATION	
				2431	*		
				2432	*****		

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 220
				2434			*****	
				2435	*		ENTRY FOR INPUT MODE DATA SYNTAY, TYPE, COUNT CHECKING	*
				2436			*****	
				2437	*			
				2438	*		INITIALIZE SYNTAX CHECKER FOR INPUT MODE	
				2439	*			
2E20	7C	87	6E	2440	FZX270	MVI	FZX320+@Q(, @BR), @UCB	DISALLOW BUFFER TERMINAL COMMA
2E23	7C	F3	97	2441		MVI	FZX350+@Q(, @BR), FZXER3	SET ERROR 803 FOR TOO FEW DATA
				2442	*			
2E26	3C	01	0D58	2443	FZX280	MVI	FZXETS, @BNE	ENABLE ELEMENT TYPE CHECKING
2E2A	7C	81	94	2444		MVI	FZX340+@Q(, @BR), @BE	ENABLE ELEMENT COUNT CHECKING
2E2D	7C	80	B3	2445		MVI	FZX390+@Q(, @BR), @NOP	ENABLE ELEMENT TTPE PROCESSING
				2446	*			
2E30	4C	01	C0 0D4E	2447	FZX290	MVC	FZXSTP(, @BR), I\$STAK(@CADDR)	INITLZ DATA SPEC CODE POINTER
2E35	4C	00	F8 0D57	2448		MVC	FZXSTC(, @BR), I\$PARM(B@LCNN)	MOVE SPEC CODE COUNT TO 4RK
2E3A	7C	00	F9	2449		MVI	FZXDT(, @BR), @ZERO	SET DATA TYPE COUNTER EQUAL ZERO
2E3D	3C	00	0D59	2450		MVI	FZXSER, I@NERR	SET ELEMENT ERROR CODE TO NULL
2E41	3C	00	0D56	2451		MVI	FZXCNT, @ZERO	SET BUFFER ELEMENT COUNT = ZERO
				2452	*			
				2453	*		ESTABLISH DATA STARTING ADDRESS IN LAST 2 BYTES OF BUFFER	
				2454	*		REGISTER @XR CONTAINS CORE ADDRESS OR LEFTMOST BYTE IN BUFFER	
				2455	*			
2E45	B4	02	FF	2456		ST	FZXBPT(, @XR), @XR	STORE BUFFER LEFT BYTE CADDR
2E48	9F	01	FF F7	2457		SLC	FZXBPT(, @XR), FZX4B1(@CADDR, @BR)	DECR BUFFER POWER TO
2E4C	B5	02	FF	2458		L	FZXBPT(, @XR), @XR	* BYTE PRECEDING BFR - LOAD PT
				2459	*			
				2460	*		IF ACCESS 1ST BYTE OF NEXT ELEMENT IN BUFFER - SET ELEMENT PARAMETERS	
				2461	*			
2E4F	D0	87	AF	2462	FZX300	B	FZX380(, @BR)	LINK TO SET EXPECTED DATA TYPE
2E52	D0	87	E6	2463		B	FZX450(, @BR)	LINK TO GET NEXT DATA CHARACTER
2E55	1E	00	0D56 F7	2464	FZX310	ALC	FZXCNT, FZX4B1(1, @BR)	INCREMENT COUNT FOR CURR ELEMENT
				2465	*			
				2466	*		SYNTAX/TYPE CHECK CURRENTLY REFERENCED ELEMENT	
				2467	*			
2E5A	C0	87	12B1	2468		B	I\$CALL	LINK TO PERFORM ELEMENT SYNTAX
2E5E	2F00			2E5F 2469		DC	AL(@VADDR)(FZX500)	* AND TYPE CHECKING
				2470	*			
2E60	3D	00	0D59	2471		CLI	FZXSER, I@NERR	IF SYNTAX/TYPE ERROR LOAD FOUND
2E64	F2	01	33	2472		JNE	FZX370	* GO RETURN TO CALLING ROUTINE
				2473	*			
				2474	*		DELIMITER PROCESSING - CHECK FOR COMMA FOLLOWING VALID ELEMENT	
				2475	*			
2E67	BD	6B	00	2476		CLI	B@CHAR(, @XR), B@CMA	IF ELEMENT DELIMITER NOT A COMMA
2E6A	F2	01	19	2477		JNE	FZX330	* GO CHECK FOR END-OF-DATA CHAR
				2478	*			FOR INPUT AND GET(CARD) MODES.
2E6D	D0	00	4F	2479	FZX320	BC	FZX300(, @BR), *-*	* GO PROCESS ELEMENT AFTER COMMA
				2480	*			
				2481	*		MAT INPUT COMMA DELIMETER PROCESSING - CHECK FOR TERMINAL COMMA	
				2482	*			
2E70	D0	87	AF	2483		B	FZX380(, @BR)	LINK TO SET EXPECTED DATA TYPE
2E73	D0	87	E6	2484		B	FZX450(, @BR)	LINK TO GET NEXT DATA CHARACTER
2E76	BD	1E	00	2485		CLI	B@CHAR(, @XR), B@EOST	IF CHAR FOLLOWING COMMA NOT EOS
2E79	D0	01	55	2486		BNE	FZX310(, @BR)	* GO PROCESS ELEMENT AFTER COMMA
				2487	*			CURRENT DATA LINE ENDS W/ COMMA-
2E7C	3D	F0	0CBC	2488		CLI	I\$ERRC, FZXER0	* IF MORE ELEMENTS EXPECTED FOR
2E80	F2	81	21	2489		JE	FZX360	* CURR ARRAY ROW, GO RETURN FOR

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 221
2E83	F2 87 14	2490	*					
		2491	J	FZX370	* NEXT LINE, ELSE EXIT ON ERROR			
		2492	*					
		2493	*	DELIMITER PROCESSING - CHECK FOR EOS FOLLOWING LAST DATA ELEMENT				
		2494	*					
2E86	BD 1E 00	2495	FZX330	CLI	B@CHAR(,@XR),B@EOST	IF ELEMENT DELIMITER NOT AN EOS		
2E89	F2 01 0E	2496		JNE	FZX370	* GO EXIT ON DATA SYNTAX ERROR		
		2497	*					
		2498	*	EOS DELIMITER ENCOUNTERED - CHECK FOR VALID ELEMENT COUNT				
		2499	*					
2E8C	D0 87 AF	2500		B	FZX380(,@BR)	LINK TO SET EXPECTED DATA TYPE		
2E8F	3D F2 0CBC	2501		CLI	I\$ERRC,FZXER2	IF NO MORE ELEMENTS EXPECTED		
2E93	F2 00 0E	2502	FZX340	JC	FZX360,*-*	* GO EXIT ON VALID DATA INPUT -		
		2503	*			* BRANCH UNCONDITIONALLY FOR GET		
		2504	*					
2E96	3C 00 0CBC	2505	FZX350	MVI	I\$ERRC,*-*	PREMATURE EOS - SET ERROR CODE		
		2506	*			* SPECIFYING MORE DATA NEEDED		
		2507	*			* AND EXIT ON DATA CNT ERROR 1-3		
2E9A	F2 87 0B	2508	FZX370	JC	FZX375,@UCB	JUMP IF NOT GET(CARD) ENTRY 1-3		
2E9D	3C BD 0CBC	2509		MVI	I\$ERRC,@@E718	SET ERROR CODE 1-3		
2EA1	F2 87 04	2510		J	FZX375	JUMP TO RETURN 1-3		
		2511	*					
		2512	*	SYNTAX CHECKER EXIT - SUPPRESS ERROR CODE FOR VALID DATA LINE				
		2513	*					
2EA4	3C 00 0CBC	2514	FZX360	MVI	I\$ERRC,I@NERR	SET ERROR CODE TO NULL STATUS		
		2515	*			* WHEN NO SYNTAX/TYPE/COUNT ERR		
2EA8	7C 87 9B	2516	FZX375	MVI	FZX370+@Q(,@BR),@UCB	FORCE NO GET(CARD) ENTRY 1-3		
2EAB	C0 87 12D3	2517		B	I\$RTRN	RETURN TO CALLING ROUTINE 1-3		
		2518	*					
		2519	*****					

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 222

```

2521 *****
2522 * INPUT ELEMENT TYPE SPECIFICATION ROUTINE *
2523 * * OPERATES ON STACKED ELEMENT SPECIFICATION CODES TO DEFINE DATA *
2524 * ELEMENT TYPE (ARITHMETIC OR CHARACTER) OR END OF DATA LIST *
2525 * *
2526 * INPUT - *
2527 * * RUN-TIME STACK - CONTAINS DATA SPECIFICATION CODES WITH LEFT *
2528 * BYTE OF STACKED CODES REFERENCED BY POINTER I$STAK. *
2529 * * FZXSTC - STACKED CODE COUNTER, INITIALLY SET TO VALUE IN *
2530 * I$PARM AND USED TO DETERMINE END-OF-DATA CONDITION. *
2531 * *
2532 * OUTPUT - *
2533 * * I$ERRC - SET TO ONE OF 3 CODES (EXCEPT WHEN ROUTINE IS DISABLED) *
2534 * * CODE X'F0' - INDICATES NEXT ELEMENT MUST BE ARITHMETIC *
2535 * * CODE X'F1' - INDICATES NEXT ELEMENT MUST BE CHARACTER *
2536 * * CODE X'F2' - INDICATES NO MORE ELEMENTS TO BE ACCEPTED *
2537 *****
2538 *
2539 * ENTRY - SAVE RETURN ADDRESS AND CHECK ENABLED STATUS
2540 *
2EAF 74 08 E5 2541 FZX380 ST FZX440+@OP1(,@BR),@ARR SET RETURN BRANCH ADDRESS
2EB2 F2 00 2D 2542 FZX390 JC FZX440,*-* GO RETURN IF SPEC RTN DISABLED
2543 *
2544 * CHECK CURRENT SPECIFICATION CODE FOR INDICATION OR ADDITIONAL
2545 * REQUIRED DATA ELEMENT(S) OBEYING SPECIFIED DATA TYPE - SPEC CODE
2546 * FORMAT IS THAT OF A 1-BYTE COUNT WITH BIT 0 INDICATING THE TYPE
2547 * OF ELEMENT BEING SPECIFIED: 0 - ARITHMETIC,
2548 * 1 - CHARACTER.
2549 *
2EB5 5F 00 F9 F7 2550 FZX400 SLC FZXDTC(,@BR),FZX4B1(B@LCXX,@BR) DECR CODE ELEMENT COUNT
2EB9 F2 02 22 2551 JNM FZX430 GO SET TYPE INDICATOR FOR CURR
2552 * * SPEC IF COUNT NOT LESS THAN 0
2553 *
2554 * CURRENT SPECIFICATION CODE DEPLETED - GET NEXT CODE TO WORK STACK AND
2555 * RESET TYPE INDICATOR ACCORDINGLY
2556 *
2EBC 4C 00 F9 0000 2557 FZX410 MVC FZXDTC(,@BR),*-(B@LCXX) MOVE STACKED CODE TO WORK AREA
2EC1 5E 00 C0 F7 2558 ALC FZXSTP(,@BR),FZX4B1(@CADDR-1,@BR) INCREMENT STACK POINTER
2559 *
2EC5 7C F0 DF 2560 MVI FZX430+@Q(,@BR),FZXER0 SET TYPE INDR ASSUMING ARITH
2EC8 78 80 F9 2561 TBN FZXDTC(,@BR),FZXDTM TEST SPEC CODE TYPE BIT AND
2ECB F2 90 06 2562 JF FZX420 * BRANCH IF ARITHMETIC
2ECE 7C F1 DF 2563 MVI FZX430+@Q(,@BR),FZXER1 SET TYPE INDR FOR CHAR SPEC
2ED1 7B 80 F9 2564 SBF FZXDTC(,@BR),FZXDTM SUPPRESS TYPE BIT IN SPEC CODE
2565 *
2ED4 5F 00 F8 F7 2566 FZX420 SLC FZXSTC(,@BR),FZX4B1(B@LCNN,@BR) DECR THE SPEC CODE COUNT
2ED8 D0 02 B5 2567 BNM FZX400(,@BR) GO PROCESS NEW SPEC IF VALID
2568 * IF NO MORE SPECIFICATION CODES
2EDB 7C F2 DF 2569 MVI FZX430+@Q(,@BR),FZXER2 * SET TYPE INDR FOR END-OF-DATA
2570 *
2571 * TYPE SPECIFICATION ROUTINE EXIT - SET TYPE SPEC AND RETURN
2572 *
2EDE 3C 00 0CBC 2573 FZX430 MVI I$ERRC,*-* SET DATA TYPE SPECIFICATION
2574 *
2EE2 C0 87 0000 2575 FZX440 B *-* RETURN TO CALLING ROUTINE
2576 *

```


[illegible]

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 224
				2579		*****				
				2580		* DATA CHARACTER 'SET' ROUTINE -				*
				2581		* * ADVANCES DATA BUFFER POINTER (REG @XR) TO NEXT NON-BLANK CHAR				*
				2582		*****				
				2583		*				
2EE6	74	08	F5	2584	FZX450	ST	FZX470+@OP1(,@BR),@ARR	SET RETURN BRANCH ADDRESS		
				2585		*				
2EE9	E2	02	01	2586	FZX460	LA	@B1(,@XR),@XR	INCR DATA CHARACTER POINTER		
2EEC	BD	40	00	2587		CLI	B@CHAR(,@XR),B@BLNK	TEST FOR A BLANK CHARACTER		
2EEF	D0	81	E9	2588		BE	FZX460(,@BR)	REPEAT LOOP UNTIL NON-BLANK		
				2589		*				
2EF2	C0	87	0000	2590	FZX470	B	*-*	RETURN TO CALLING ROUTINE		
				2591		*				
				2592		*****				

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 225
		2594		*****	
		2595		* INPUT EXECUTION ROUTINE CONSTANTS (4TH VM PAGE)	*
		2596		*****	
		2597		*	
2EF6 0001		2EF7 2598	FZX4B1 DC	IL2'1'	BINARY INTEGER +1
		2600		*****	
		2601		* INPUT EXECUTION ROUTINE WORK AREAS (4TH VM PAGE)	*
		2602		*****	
		2603		*	
2EF8		2EC0 2604	FZXSTP EQU	FZX410+@DOP2	TEMPORARY STACK POINTER
		2EF8 2605	FZXSTC DS	CL(B@LCNN)	STACKED SPEC CODE COUNTER
		2606		*	
2EF9		2EF9 2607	FZXDTC DS	CL(B@LCXX)	DATA TYPE CODE WORK AREA
		0080 2608	FZXDTM EQU	X'80'	SPEC CODE DATA TYPE MASK
		2609		*	
		0D58 2610	FZXETS EQU	I\$WRK1-1	ELEMENT TYPE CHECKING SWITCH
		2611		*	* CHECK - @BNE, NOCHECK - @NOP
		0D59 2612	FZXSER EQU	I\$WRK1	SYNTAX CHECKER ERROR CODE BYTE
		00FF 2613	FZXSEC EQU	X'FF'	SYNTAX CHECKER SPECIAL ERR CODE
		2614		*	
		0D56 2615	FZXCNT EQU	I\$PARM-1	PROCESSED BUFFER ELEMENT COUNT
		2616		*	
		2617		*****	

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 226
					2619		*****			
					2620	*	VIRTUAL MEMORY INPUT EXECUTION ROUTINE (5TH VM PAGE) -			*
					2621	*	ELEMENT SYNTAX CHECKING ROUTINE FOR FOLLOWING DATA TYPES -			*
					2622	*	* ARITHMETIC CONSTANTS (I, F, E FORMATS)			*
					2623	*	* BASIC INTERNAL CONSTANTS			*
					2624	*	* CHARACTER CONSTANTS			*
					2625	*				*
					2626	*	INPUT -			*
					2627	*	* I\$ERRC - SET TO X'F0' FOR EXPECTED ARITHMETIC CONSTANT			*
					2628	*	- SET TO X'F1' FOR EXPECTED CHARACTER CONSTANT			*
					2629	*	* FZXETS - SET TO @BNE WHEN CONSTANT TYPE IS TO BE CHECKED			*
					2630	*	- SET TO @NOP WHEN CONSTANT TYPE NOT TO BE CHECKED			*
					2631	*	* FZXSER - INITIALIZED TO CODE I@NERR (X'00')			*
					2632	*	* REGISTER @XR - REFERENCES 1ST CHARACTER OF CONSTANT			*
					2633	*				*
					2634	*	OUTPUT -			*
					2635	*	* FZXSER - SET TO CODE X'FF' WHEN SYNTAX/TYPE ERROR OCCURS			*
					2636	*	* REGISTER @XR - REFERENCES DELIMITER IMMEDIATELY FOLLOWING			*
					2637	*	CONSTANT WHEN NO SYNTAX/TYPE ERROR HAS BEEN ENCOUNTERED			*
					2638		*****			*
					2639	*				*
					2640	*	ESTABLISH ADDRESSABILITY FOR INPUT ROUTINE (5TH VM PAGE)			*
					2641	*				*
					2642		*FZXP5B VPAGE 0 SET 5TH PAGE ADDRESSABILITY			*
2F00					2643		ORG *,256,0 SET STARTING ADDRESS			*
				2F00	2644	FZXP5B EQU *	START OF PROGRAM CODING			*
2E01					2645		ORG *-255 RESET IAR TO PAGE			*
2F00					2646		ORG *,256,0 * BOUNDARY ADDRESS			*
				2F00	2647		USING *,@BR SET PAGE BASE ADDRESS			*
2F00					2648		ORG FZXP5B RESET STARTING ADDRESS			*
					2649		*** END OF EXPANSION ***			*
					2650	*				*
					2651	*	ELEMENT SYNTAX CHECKER ENTRY - INITIALIZE DATA TYPE CHECKING			*
					2652	*				*
2F00 4C 00 15 0D58					2653	FZX500 MVC	FZX510+@Q(,@BR),FZXETS(1) SET ARITHMETIC TYPE CHECKING			*
2F05 4C 00 3A 0D58					2654	MVC	FZX540+@Q(,@BR),FZXETS(1) SET CHARACTER TYPE CHECKING			*
					2655	*				*
					2656	*	TEST FOR CHARACTER CONSTANT PROCESSING			*
					2657	*				*
2F0A BD 7D 00					2658	CLI	B@CHAR(,@XR),B@SQUO IF 1ST CHARACTER IS SINGLE QUOTE			*
2F0D F2 81 25					2659	JE	FZX530 * GO PERFORM CHAR CONSTANT CHECK			*
					2660	*				*
					2661	*	ARITHMETIC CONSTANT ASSUMED - PERFORM ARITHMETIC TYPE CHECK			*
					2662	*				*
2F10 3D F0 0CBC					2663	CLI	I\$ERRC,FZXER0 IF ARITH CONSTANT NOT EXPECTED			*
2F14 F2 00 79					2664	FZX510 JC	FZX590,*-* * GO EXIT ON DATA TYPE ERROR -			*
					2665	*	* TEST IS DISABLED FOR GET(CARD)			*
					2666	*				*
					2667	*	ARITHMETIC CONSTANT ALLOWABLE - TEST FOR NUMERIC OR INTERNAL			*
					2668	*				*
2F17 BD 4E 00					2669	CLI	B@CHAR(,@XR),B@PLUS IF 1ST CHARACTER IS PLUS SIGN			*
2F1A F2 81 03					2670	JE	FZX520 * BRANCH TO GET NEXT CHARACTER			*
2F1D BD 60 00					2671	CLI	B@CHAR(,@XR),B@MINS IF 1ST CHARACTER IS MINUS SIGN			*
2F20 D0 81 98					2672	FZX520 BE	FZX610(,@BR) * LINK TO GET NEXT DATA CHAR			*
2F23 BD 50 00					2673	CLI	B@CHAR(,@XR),B@ICON IF INTERNAL CONSTANT IDENTIFIER			*
2F26 F2 81 34					2674	JE	FZX560 * FOUND, GO TEST REST OF SYMBOL			*

[illegible]

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 228
		2678		*****	
		2679		* NUMERIC CONSTANT SYNTAX CHECKING	*
		2680		*****	
		2681		*	
		2682		* CALL SYNTAX CHECKER FOR ASSUMED NUMERIC CONSTANT	
		2683		*	
2F29	76 02 A9	2684	A	FZX5M1(,@BR),@XR	DECREMENT THE DATA POINTER
		2685	*		
2F2C	C0 87 12B1	2686	B	I\$CALL	LINK TO SYNTAX CHECK ASSUMED
2F30	3000	2F31 2687	DC	AL(@VADDR)(FZX650)	* NUMERIC ARITHMETIC CONSTANT
		2688	*		
2F32	F2 87 5F	2689	J	FZX600	GO RETURN TO CALLING ROUTINE
		2690	*		
		2691		*****	

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 229
				2693		*****				
				2694	*	CHARACTER CONSTANT SYNTAX CHECKING				*
				2695		*****				
				2696	*					
				2697	*	PERFORM CHARACTER TYPE CHECK				
				2698	*					
2F35	3D	F1	0CBC	2699	FZX530	CLI	I\$ERRC,FZXER1			IF CHAR CONSTANT NOT EXPECTED
2F39	F2	00	54	2700	FZX540	JC	FZX590,*-*			* GO EXIT ON DATA TYPE ERROR -
				2701	*					* TEST IS DISABLED FOR GET(CARD)
				2702	*					
				2703	*	CHARACTER CONSTANT ALLOWABLE - CHECK SYNTAX VALIDITY				
				2704	*					
2F3C	E2	02	01	2705	FZX550	LA	@B1(,@XR),@XR			INCREMENT THE DATA POINTER
2F3F	BD	1E	00	2706		CLI	B@CHAR(,@XR),B@EOST			IF PREMATURE DATA TERMINATION
2F42	F2	81	4B	2707		JE	FZX590			* GO SET SYNTAX ERROR AND RETURN
2F45	BD	7D	00	2708		CLI	B@CHAR(,@XR),B@SQUO			IF CHARACTER IS NOT A QUOTE
2F48	D0	01	3C	2709		BNE	FZX550(,@BR)			* BRANCH TO CONTINUE SCAN LOOP
2F4B	E2	02	01	2710		LA	@B1(,@XR),@XR			INCREMENT THE DATA POINTER
2F4E	BD	7D	00	2711		CLI	B@CHAR(,@XR),B@SQUO			IF QUOTE PAIR IS INDICATED
2F51	D0	81	3C	2712		BE	FZX550(,@BR)			* BRANCH TO CONTINUE SCAN LOOP
				2713	*					* ELSE LAST QUOTE ENDS CONSTANT
2F54	BD	40	00	2714		CLI	B@CHAR(,@XR),B@BLNK			IF BLANK CHAR AFTER CONSTANT
2F57	D0	81	98	2715		BE	FZX610(,@BR)			* LINK TO GET ELEMENT DELIMITER
2F5A	F2	87	37	2716		J	FZX600			GO RETURN TO CALLING ROUTINE
				2717	*					
				2718		*****				

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 230
				2720		*****				
				2721	*	INTERNAL CONSTANT SYNTAX CHECKING				*
				2722		*****				
				2723	*					
				2724	*	MOVE INTERNAL, CONSTANT SYMBOL TO WORK AREA				
				2725	*					
2F5D	5F	03	B9 B9	2726	FZX560	SLC	FZXICW(,@BR),FZXICW(FZXICL,@BR) CLEAR AREA TO BINARY 0'S			
				2727	*					
2F61	D0	87	98	2728	FZX570	B	FZX610(,@BR) LINK TO GET NEXT DATA CHARACTER			
2F64	BD	C1	00	2729		CLI	B@CHAR(,@XR),B@LETA IF CHAR NOT A LETTER OR DIGIT			
2F67	F2	82	11	2730		JL	FZX580 * GO TEST FOR SYMBOL VALIDITY			
2F6A	5C	02	B9 B8	2731		MVC	FZXICW(,@BR),FZXICW-1(FZXICL-1,@BR) SHIFT SYMBOL RIGHT			
2F6E	6C	00	B6 00	2732		MVC	FZXICC(,@BR),B@CHAR(1,@XR) MOVE NEW CHAR INTO WORK AREA			
2F72	7D	00	B9	2733		CLI	FZXICR(,@BR),@ZERO IF SYMBOL WORK AREA NOT FILLED			
2F75	D0	81	61	2734		BE	FZX570(,@BR) * LOOP TO GET NEXT SYMBOL CHAR			
2F78	D0	87	98	2735		B	FZX610(,@BR) * ELSE LINK TO GET DELIMITER			
				2736	*					
				2737	*	CHECK INTERNAL CONSTANT SYMBOL FOR VALID SYNTAX				
				2738	*					
2F7B	5D	03	B9 AD	2739	FZX580	CLC	FZXICW(,@BR),FZXIEX(FZXICL,@BR) IF SYMBOL IS &E			
2F7F	F2	81	12	2740		JE	FZX600 * GO RETURN TO CALLER			
2F82	5D	03	B9 B1	2741		CLC	FZXICW(,@BR),FZXIPI(FZXICL,@BR) IF SYMBOL IS &PI			
2F86	F2	81	0B	2742		JE	FZX600 * GO RETURN TO CALLER			
2F89	5D	03	B9 B5	2743		CLC	FZXICW(,@BR),FZXIS2(FZXICL,@BR) IF SYMBOL IS &SQR2			
2F8D	F2	81	04	2744		JE	FZX600 * GO RETURN TO CALLER			
				2745	*					
				2746	*	EXIT - SET ERROR CONDITION CODE IF INVALID SYNTAX OR TYPE ENCOUNTERED				
				2747	*					
2F90	3C	FF	0D59	2748	FZX590	MVI	FZXSER,FZXSEC SET ERROR CODE FOR INVALIDITY			
				2749	*					
2F94	C0	87	12D3	2750	FZX600	B	I\$RTRN RETURN TO CALLING ROUTINE			
				2751	*					
				2752		*****				

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 231
				2754			*****			
				2755			* DATA CHARACTER 'GET' ROUTINE -			*
				2756			* * ADVANCES DATA BUFFER POINTER (REG @XR TO NEXT NON-BLANK CHAR)			*
				2757			*****			
				2758			*			
2F98	74	08	A7	2759	FZX610	ST	FZX630+@OP1(,@BR),@ARR	SET RETURN BRANCH ADDRESS		
				2760			*			
2F9B	E2	02	01	2761	FZX620	LA	@B1(,@XR),@XR	INCR DATA CHARACTER POINTER		
2F9E	BD	40	00	2762		CLI	B@CHAR(,@XR),B@BLNK	TEST FOR A BLANK CHARACTER		
2FA1	D0	81	9B	2763		BE	FZX620(,@BR)	REPEAT LOOP UNTIL NON-BLANK		
				2764			*			
2FA4	C0	87	0000	2765	FZX630	B	*-*	RETURN TO CALLING ROUTINE		
				2766			*			
				2767			*****			

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 232
					2769	*****	*****			
					2770	* INPUT EXECUTION ROUTINE CONSTANTS (5TH VM PAGE)	*			
					2771	*****	*****			
					2772	*				
2FA8	FFFF			2FA9	2773	FZX5M1 DC	IL2'-1'			BINARY INTEGER -1
					2774	*				
				0004	2775	FZXICL EQU	4			LENGTH OF INT CON TEST SYMBOL
					2776	*				
2FAA	C5			2FAA	2777		DC CL1'E'			TEST SYMBOL FOR &E
2FAB	000000			2FAD	2778	FZXIEX DC	XL(FZXICL-1)'00'			SYMBOL CONSTANT FILLER
					2779	*				
2FAE	C9D7			2FAF	2780		DC CL2'IP'			TEST SYMBOL FOR &PI
2FB0	0000			2FB1	2781	FZXIPI DC	XL(FZXICL-2)'00'			SYMBOL CONSTANT FILLER
					2782	*				
2FB2	F2D9D8E2			2FB5	2783	FZXIS2 DC	CL4'2RQS'			TEST SYMBOL FOR &SQR2
					2785	*****	*****			
					2786	* INPUT EXECUTION ROUTINE WORK AREAS (5TH VM PAGE)	*			
					2787	*****	*****			
					2788	*				
				2FB6	2789	FZXICC EQU	*			CURR INTERNAL CON SYMBOL CHAR
2FB6				2FB9	2790	FZXICW DS	CL(FZXICL)			INTERNAL SYMBOL WORK AREA
				2FB9	2791	FZXICR EQU	FZXICW			SYMBOL WORK AREA RIGHT BYTE ADDR
					2792	*				
					2793	*****	*****			

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 233
		2795		*****	
		2796	*	VIRTUAL MEMORY INPUT EXECUTION ROUTINE (6TH VM PAGE)	*
		2797	*	ELEMENT SYNTAX CHECKING ROUTINE FOR NUMERIC ARITHMETIC CONSTANTS	*
		2798	*		*
		2799	*	INPUT -	*
		2800	*	* FZXSER - INITIALIZED TO CODE I@NERR (X'00')	*
		2801	*	* REGISTER @XR - REFERENCES CHARACTER PRECEDING 1ST NON-SIGN CHAR	*
		2802	*		*
		2803	*	OUTPUT -	*
		2804	*	* FZXSER - SET TO CODE X'FF' WHEN SYNTAX ERROR IS FOUND	*
		2805	*	* REGISTER @XR - REFERENCES DELIMITER IMMEDIATELY FOLLOWING	*
		2806	*	CONSTANT WHEN NO SYNTAX ERROR HAS BEEN ENCOUNTERED	*
		2807		*****	
		2808	*		
		2809	*	ESTABLISH ADDRESSABILITY FOR INPUT ROUTINE (6TH VM PAGE)	
		2810	*		
3000		2811	*FZXP6B	VPAGE 0 SET 6TH PAGE ADDRESSABILITY	
		2812	ORG	*,256,0 SET STARTING ADDRESS	
	3000	2813	FZXP6B EQU	* START OF PROGRAM CODING	
2F01		2814	ORG	*-255 RESET IAR TO PAGE	
3000		2815	ORG	*,256,0 * BOUNDARY ADDRESS	
	3000	2816	USING	*,@BR SET PAGE BASE ADDRESS	
3000		2817	ORG	FZXP6B RESET STARTING ADDRESS	
		2818	***	END OF EXPANSION ***	
		2819	*		
		2820	*	ENTRY - INITIALIZE FOR NUMERIC CONSTANT SYNTAX CHECKING	
		2821	*		
3000 5C 01 EA E4		2822	FZX650 MVC	FZXXCT(,@BR),FZXXZR(FZXXCL,@BR) SET 'ZERO' EXPONENT COUNT	
3004 7C 80 30		2823	MVI	FZX690+@Q(,@BR),@NOP SET CONSTANT FRACTION SWITCH ON	
3007 7C 87 49		2824	MVI	FZX720+@Q(,@BR),@UCB SET CONSTANT DIGIT SWITCH OFF	
300A 7C 87 B0		2825	MVI	FZX800+@Q(,@BR),@UCB SET MANTISSA ZERO SWITCH ON	
		2826	*		
		2827		*****	

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 234
				2829			*****	
				2830			* NUMERIC CONSTANT MANTISSA SYNTAX CHECKING	*
				2831			*****	
				2832			*	
				2833			* INCREMENT PAST INSIGNIFICANT LEADING ZEROS	
				2834			*	
300D	D0	87	C8	2835	FZX660	B	FZX830(,@BR)	LINK TO GET NEXT DATA CHARACTER
3010	D0	81	0D	2836		BE	FZX660(,@BR)	LOOP IF CHARACTER IS DECML ZERO
3013	F2	82	10	2837		JL	FZX680	BRANCH IF CHARACTER NOT DECIMAL
				2838			*	
				2839			* PROCESS INTEGER DIGITS - INCREMENT EXPONENT COUNT FOR EACH DIGIT	
				2840			*	
3016	7C	87	30	2841	FZX670	MVI	FZX690+@Q(,@BR),@UCB	SET CONSTANT FRACTION SWITCH OFF
3019	7C	80	B0	2842		MVI	FZX800+@Q(,@BR),@NOP	SET MANTISSA ZERO SWITCH OFF
				2843			*	
301C	5E	01	EA E2	2844	FZX675	ALC	FZXXCT(,@BR),FZX6B1(FZXXCL,@BR)	INCREMENT EXPONENT COUNT
3020	D0	87	C8	2845		B	FZX830(,@BR)	LINK TO GET NEXT DATA CHARACTER
3023	D0	02	1C	2846		BNL	FZX675(,@BR)	LOOP IF CHARACTER IS A DECIMAL
				2847			*	
				2848			* TEST FOR A DECIMAL POINT OR MANTISSA DELIMITER	
				2849			*	
3026	BD	4B	00	2850	FZX680	CLI	B@CHAR(,@XR),B@DPNT	IF CHARACTER NOT A DECIMAL POINT
3029	F2	01	1C	2851		JNE	FZX720	* GO CHECK FOR AT LEAST 1 DIGIT
302C	F2	87	07	2852		J	FZX700	* ELSE SKIP TO TEST FRACTIONALS
				2853			*	
				2854			* PROCESS LEADING FRACTIONAL ZEROS - DECREMENT EXPONENT COUNT FOR	
				2855			* EACH ZERO WHEN CONSTANT CONTAINS NO SIGNIFICANT INTEGER DIGITS	
				2856			*	
302F	F2	00	04	2857	FZX690	JC	FZX700,*-*	BRANCH IF ANY INTEGER COMPONENT
3032	5F	01	EA E2	2858		SLC	FZXXCT(,@BR),FZX6B1(FZXXCL,@BR)	DECREMENT EXPONENT COUNT
3036	D0	87	C8	2859	FZX700	B	FZX830(,@BR)	LINK TO SET NEXT DATA CHARACTER
3039	D0	81	2F	2860		BE	FZX690(,@BR)	LOOP IF CHARACTER IS DECML ZERO
303C	F2	82	09	2861		JL	FZX720	BRANCH IF CHARACTER NOT DECIMAL
				2862			*	
				2863			* INCREMENT PAST TRAILING FRACTIONAL DIGITS	
				2864			*	
303F	7C	80	B0	2865		MVI	FZX800+@Q(,@BR),@NOP	SET MANTISSA ZERO SWITCH OFF
				2866			*	
3042	D0	87	C8	2867	FZX710	B	FZX830(,@BR)	LINK TO GET NEXT DATA CHARACTER
3045	D0	02	42	2868		BNL	FZX710(,@BR)	LOOP IF CHARACTER IS A DECIMAL
				2869			*	
				2870			* TEST FOR NO MANTISSA DIGITS (AN ERROR CONDITION)	
				2871			*	
3048	F2	00	75	2872	FZX720	JC	FZX810,*-*	BRANCH IF NO 401\$\$A DIGITS
				2873			*	
				2874			*****	

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 235
				2876			*****	
				2877			* NUMERIC CONSTANT EXPONENT SYNTAX CHECKING	*
				2878			*****	
				2879			*	
				2880			* TEST FOR E-FORMAT SPECIFICATION	
				2881			*	
304B	BD	C5	00	2882		CLI	B@CHAR(,@XR),B@EXPC	IF NO E-FORMAT SPECIFICATION
304E	F2	01	5E	2883		JNE	FZX800	* BRANCH TO TEST VALUE MAGNITUDE
				2884			*	
				2885			* E-FORMAT - ESTABLISH THE EXPONENT SIGN	
				2886			*	
3051	7C	80	A2	2887		MVI	FZX780+@Q(,@BR),@NOP	SET POSITIVE EXPONENT STATUS
3054	D0	87	C8	2888		B	FZX830(,@BR)	LINK TO GET NEXT DATA CHARACTER
3057	BD	4E	00	2889		CLI	B@CHAR(,@XR),B@PLUS	IF EXPONENT PLUS SIGN SPECIFIED
305A	F2	81	09	2890		JE	FZX730	* SKIP TO GET THE NEXT CHARACTER
305D	BD	60	00	2891		CLI	B@CHAR(,@XR),B@MINS	IF EXP MINUS SIGN NOT SPECIFIED
3060	F2	01	06	2892		JNE	FZX740	* SO PROCESS EXPONENT DIGITS
3063	7C	87	A2	2893		MVI	FZX780+@Q(,@BR),@UCB	* ELSE SET NEGATIVE EXP STATUS
3066	D0	87	C8	2894	FZX730	B	FZX830(,@BR)	LINK TO GET CHARACTER AFTER SIGN
				2895			*	
				2896			* TEST FOR NO EXPONENT DIGITS (AN ERROR CONDITION)	
				2897			*	
3069	BD	F0	00	2898	FZX740	CLI	B@CHAR(,@XR),B@DEC0	IF NO EXPONENT DIGIT FOUND
306C	F2	82	51	2899		JL	FZX810	* BRANCH TO SET SYNTAX ERROR
				2900			*	
				2901			* ESTABLISH EXPONENT DIGIT(S) FOR CONVERSION TO BINARY	
				2902			*	
306F	64	10	EC 00	2903		ZAZ	FZX6DX(FZDXL,@BR),B@CHAR(1,@XR)	SAVE 1ST EXPONENT DIGIT
3073	D0	87	C8	2904		B	FZX830(,@BR)	LINK TO GET NEXT DATA CHARACTER
3076	F2	82	0B	2905		JL	FZX750	BRANCH IF CHAR NOT A DECIMAL
3079	5C	00	EB EC	2906		MVC	FZX6DX-1(,@BR),FZX6DX(1,@BR)	SHIFT 1ST EXPONENT DIGIT AND
307D	6C	00	EC 00	2907		MVC	FZX6DX(,@BR),B@CHAR(1,@XR)	* STORE NEW DIGIT IN UNITS POS
3081	D0	87	C8	2908		B	FZX830(,@BR)	LINK TO GET CHAR AFTER EXPONENT
				2909			*	
				2910			* CONVERT EXPONENT DECIMAL DIGITS TO BINARY	
				2911			*	
3084	7B	F0	EC	2912	FZX750	SBF	FZX6DX(,@BR),B@ZPOS	SUPPRESS UNITS DIGIT ZONE BITS
3087	7B	F0	EB	2913		SBF	FZX6DX-1(,@BR),B@ZPOS	SUPPRESS TENS DIGIT ZONE BITS
308A	5E	00	EB EB	2914		ALC	FZX6DX-1(,@BR),FZX6DX-1(1,@BR)	DOUBLE TENS DIGIT FOR 2X
308E	5E	00	EC EB	2915		ALC	FZX6DX(,@BR),FZX6DX-1(1,@BR)	ADD 2 X TM TO 1.NITS PCS
3092	5E	00	EB EB	2916		ALC	FZX6DX-1(,@BR),FZX6DX-1(1,@BR)	DOUBLE TENS DIGIT FOR 4X
3096	5E	00	EB EB	2917		ALC	FZX6DX-1(,@BR),FZX6DX-1(1,@BR)	DOUBLE TENS DIGIT FOR 8X
309A	5E	00	EC EB	2918		ALC	FZX6DX(,@BR),FZX6DX-1(1,@BR)	ADD I 1 ?T\S 101441T\$ P05
309E	7C	00	EB	2919		MVI	FZX6DX-1(,@BR),@ZERO	ZERO TO FORM 2-BYTE BINARY NO.
				2920			*	
				2921			* MODIFY EXPONENT COUNT WITH SPECIFIED EXPONENT	
				2922			*	
30A1	F2	00	07	2923	FZX780	JC	FZX790,*-*	BRANCH IF SPEC EXP IS NEGATIVE
30A4	5E	01	EA EC	2924		ALC	FZXXCT(,@BR),FZX6BX(FZXXCL,@BR)	ADD SPECIFIED EXPONENT
30A8	F2	87	04	2925		J	FZX800	* TO COUNT AND GO TEST RANGE
30AB	5F	01	EA EC	2926	FZX790	SLC	FZXXCT(,@BR),FZX6BX(FZXXCL,@BR)	SUBTRACT SPEC EXPONENT
				2927			*	
				2928			* TEST FOR ZERO MANTISSA - BYPASS CONSTANT RANGE ERROR CHECK WHEN	
				2929			* MANTISSA IS ZERO, SINCE EXPONENT WILL ALWAYS BE SET TO E-98	
				2930			*	
30AF	F2	00	12	2931	FZX800	JC	FZX820,*-*	BRANCH IF MANTISSA IS ZERO

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 236
		2932	*		
		2933	*	TEST FOR CONSTANT RANGE ERROR - COMBINED EXPONENT MUST FALL WITHIN	
		2934	*	RANGE (128-98) TO (128+99) ... 'ZERO' EXPONENT IS 128	
		2935	*		
30B2	5D 01 EA E6	2936	CLC	FZXXCT(,@BR),FZXXHI(FZXXCL,@BR) IF EXPONENT EXCEEDS HIGH	
30B6	F2 84 07	2937	JH	FZX810 * BOUNDARY, GO EXIT ON ERROR	
30B9	5D 01 EA E8	2938	CLC	FZXXCT(,@BR),FZXXLO(FZXXCL,@BR) IF EXPONENT NOT LESS THAN	
30BD	F2 02 04	2939	JNL	FZX820 * LOW BOUNDARY, RETURN TO CALLER	
		2940	*		
		2941	*	EXIT - SET ERROR CODE IF INVALID SYNTAX ENCOUNTERED	
		2942	*		
30C0	3C FF 0D59	2943	FZX810 MVI	FZXSER,FZXSEC SET ERROR CODE FOR INVALIDITY	
		2944	*		
30C4	C0 87 12D3	2945	FZX820 B	I\$RTRN RETURN TO CALLING ROUTINE	
		2946	*		
		2947	*****		

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 237
		2949		*****	
		2950		* DATA CHARACTER 'GET' ROUTINE -	*
		2951		* * ADVANCES DATA BUFFER POINTER (REG @XR) TO NEXT NON-BLANK CHAR	*
		2952		* * SETS PSR DEPENDING ON CHARACTER TYPE -	*
		2953		* * LOW - NON-DECIMAL CHARACTER	*
		2954		* * EQUAL - DECIMAL ZERO CHARACTER	*
		2955		* * HIGH - DECIMAL DIGIT OTHER THAN ZERO	*
		2956		* * SETS CONSTANT DIGIT SWITCH ON WHEN DIGIT ENCOUNTERED	*
		2957		*****	
		2958		*	
30C8	74 08 E0	2959	FZX830 ST	FZX850+@OP1(,@BR),@ARR SET RETURN BRANCH ADDRESS	
		2960		*	
30CB	E2 02 01	2961	FZX840 LA	@B1(,@XR),@XR INCR DATA CHARACTER POINTER	
30CE	BD 40 00	2962	CLI	B@CHAR(,@XR),B@BLNK TEST FOR A BLANK CHARACTER	
30D1	D0 81 CB	2963	BE	FZX840(,@BR) REPEAT LOOP UNTIL NON-BLANK	
		2964	*		
30D4	BD F0 00	2965	CLI	B@CHAR(,@XR),B@DEC0 IF CHARACTER NOT A DECIMAL DIGIT	
30D7	F2 82 03	2966	JL	FZX850 * GO RETURN TO CALLING ROUTINE	
30DA	7C 80 49	2967	MVI	FZX720+@Q(,@BR),@NOP * ELSE SET CONSTANT DIGIT SW ON	
		2968	*		
30DD	C0 87 0000	2969	FZX850 B	*-* RETURN TO CALLMS MANE	
		2970		*	
		2971		*****	

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC		OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 238
				2973	*****	
				2974	* INPUT EXECUTION ROUTINE CONSTANTS (6TH VM PAGE)	*
				2975	*****	
				2976	*	
30E1	0001		30E2	2977	FZX6B1 DC IL2'1' BINARY INTEGER +1	
				2978	*	
			0002	2979	FZXXCL EQU 2 LENGTH OF EXPONENT TEST COUNT	
30E3	0180		30E4	2980	FZXXZR DC AL(FZXXCL)(256+B@NXZR) TEST NORM EXPONENT - ZERO	
30E5	01E3		30E6	2981	FZXXHI DC AL(FZXXCL)(256+B@NXHI) TEST NORM EXPONENT - MAXIMUM	
30E7	011E		30E8	2982	FZXALO DC AL(FZXXCL)(256+B@NXLO) TEST NORM EXPONENT - MINIMUM	
				2984	*****	
				2985	* INPUT EXECUTION ROUTINE WOTK AREAS (6TH VM PAGE)	*
				2986	*****	
				2987	*	
30E9			30EA	2988	FZXACT DS CL(FZXXCL) EXPONENT TEST COUNT AREA	
				2989	*	
			0002	2990	FZDXL EQU 2 LENGTH OF MAX DECIMAL EXPONENT	
30EB			30EC	2991	FZX6DX DS CL(FZDXL) DECIMAL EXPONENT WORK AREA	
			30EC	2992	FZX6BX EQU FZX6DX BINARY SPECIFIED EXPONENT	
				2993	*	
				2994	*****	

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 239
		2996		*****	
		2997	*	VIRTUAL MEMORY INPUT EXECUTION ROUTINE (7TH VM PAGE) -	*
		2998	*	ELEMENT CONVERSION ROUTINE FOR FOLLOWING DATA TYPES -	*
		2999	*	* ARITHMETIC CONSTANTS (I, F, E FORMATS)	*
		3000	*	* BASIC INTERNAL CONSTANTS	*
		3001	*	* CHARACTER CONSTANTS	*
		3002	*		*
		3003	*	INPUT -	*
		3004	*	* REGISTER @XR - REFERENCES LEFTMOST BYTE OF FULL PAGE BUFFER	*
		3005	*	* DATA POINTER LOCATED IN LAST 2 BYTES OF BUFFER - POINTER	*
		3006	*	REFERENCES CHARACTER PRECEDING 1ST CHARACTER OF CONSTANT.	*
		3007	*		*
		3008	*	OUTPUT -	*
		3009	*	* CONVERTED ELEMENT STACKED BEGINNING AT I\$STAK - STACKED DATA	*
		3010	*	FORMAT IS PACKED FLOATING POINT (CURRENT PRECISION) OR	*
		3011	*	19-BYTE CHARACTER ELEMENT.	*
		3012	*	* REGISTER @XR - REFERENCES DELIMITER FOLLOWING CONVERTED CON.	*
		3013	*	* DATA POINTER IN BUFFER UPDATED TO REFERENCE THE CON DELIMITER	*
		3014		*****	
		3015	*		
		3016	*	ESTABLISH ADDRESSABILITY FOR INPUT ROUTINE 7TH VM PAGE	
		3017	*		
3100		3018	*FZXP7B	VPAGE 0 SET 7TH PAGE ADDRESSABILITY	
		3019	ORG	*,256,0 SET STARTING ADDRESS	
3001	3100	3020	FZXP7B	EQU * START OF PROGRAM CODING	
3100		3021	ORG	*-255 RESET IAR TO PAGE	
		3022	ORG	*,256,0 * BOUNDARY ADDRESS	
	3100	3023	USING	*,@BR SET PAGE BASE ADDRESS	
3100		3024	ORG	FZXP7B RESET STARTING ADDRESS	
		3025	***	END OF EXPANSION ***	
		3026	*		
		3027		*****	

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 240
				3029		*****				
				3030	*	ELEMENT CONVERSION ENTRY - ACCESS 1ST CHARACTER OF CONSTANT				*
				3031		*****				
				3032	*					
				3100	3033	FZXCNV EQU *	INPUT DATA CONVERSION ENTRT PT			
				3034	*					
3100	74	02	B7	3035	FZX860	ST FZX906+@OP1(,@BR),@XR	SAVE OUFF1R_LEFT BYTE CORE CDR			
3103	B5	02	FF	3036		L FZXBPT(,@XR),@XR	LOAD THE BUFFER DATA POINTER			
3106	D0	87	BF	3037		B FZX910(,@BR)	LINK TO GET NUT DATA CHARACTER			
				3038	*					
				3039	*	TEST FOR CHARACTER CONSTANT PROCESSING				
				3040	*					
3109	BD	7D	00	3041		CLI B@CHAR(,@XR),B@SQUO	IF 1ST CHARACTER IS SINGLE COTE			
310C	F2	81	2B	3042		JE FZX873	* GO PERFORM CHAR CON CONVERSION			
				3043	*					
				3044	*	TEST FOR NUMERIC OR INTERNAL ARITHMETIC CONSTANT - PERFORM SIGN				
				3045	*	PROCESSING FOR THE POSSIBLE INTERNAL CONSTANT SYMBOL				
				3046	*					
310F	74	02	30	3047		ST FZX870+@OP1(,@BR),@XR	SAVE CON 1ST CHIM IN CASE OF NUM			
3112	7C	4E	CF	3048		MVI FZXICB-1(,@BR),B@PLUS	SET POSITIVE INTERNAL CON INDR			
3115	BD	4E	00	3049		CLI B@CHAR(,@XR),B@PLUS	IF 1ST CHARACTER IS PLUS SIGN			
3118	F2	81	09	3050		JE FZX863	* BRANCH TO GET NEXT CHARACTER			
311B	BD	60	00	3051		CLI B@CHAR(,@XR),B@MINS	IF 1ST CHARACTER NOT MINUS			
311E	F2	01	06	3052		JNE FZX866	* BRANCH TO TEST CONSTANT TYPE			
3121	7C	60	CF	3053		MVI FZXICB-1(,@BR),B@MINS	* ELSE SET NEG INTERNAL CON INDR			
				3054	*					
3124	D0	87	BF	3055	FZX863	B FZX910(,@BR)	LINK TO GET NEXT DATA CHARACTER			
3127	BD	50	00	3056	FZX866	CLI B@CHAR(,@XR),B@ICON	IF INTERNAL CONSTANT IDENTIFIER			
312A	F2	81	56	3057		JE FZX893	* FOUND, GO TEST REST OF SYMBOL			
				3058	*					
				3059		*****				

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 241
					3061	*****				
					3062	* NUMERIC ARITHMETIC CONSTANT CONVERSION				*
					3063	*****				
					3064	*				
					3065	* CALL FLOATING POINT CONVERSION/STACKINS ROUTINE				
					3066	*				
	312D	C2	01	0000	3067	FZX870 LA	*-*,@BR			LOAD CADDR OF CONSTANT 1ST CHAR
					3068	*				
	3131	C0	87	12B1	3069	B	I\$CALL			LINK TO CONVERT AND STACK THE
	3135	3200			3070	DC	AL(@VADDR)(FZX920)			* CONSTANT AS PACKED FLOATING PT
				3136	3071	*				
	3137	F2	87	7A	3072	J	FZX906			GO EXIT CONVERSION ROUTINE
					3073	*				
					3074	*****				

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 242
				3076		*****		
				3077		* CHARACTER CONSTANT CONVERSION AND STACKING		*
				3078		*****		
				3079		*		
				3080		* INITIALIZE THE CONSTANT BUCKET FOR CHARACTER PROCESSING.		
				3081		*		
313A	7C	40	CF	3082	FZX873	MVI	FZXSTS(, @BR), B@DTYP	SET BUCKET STATUS FOR CHAR CON
313D	5C	11	E1 E2	3083		MVC	FZXCRR(, @BR), FZXBLK(I@LCRF, @BR)	BLANK THE CHAR DATA FIELD
3141	7C	D0	62	3084		MVI	FZX883+@D1(, @BR), FZXCRI-FZXP7B	INITLZ CHAR DATA FLD DISP
				3085		*		
				3086		* ACCESS NEXT CHARACTER - TEST FOR PAIRED QUOTES OR END OF CONSTANT		
				3087		*		
3144	E2	02	01	3088	FZX876	LA	@B1(, @XR), @XR	INCREMENT THE DATA POINTER
3147	BD	7D	00	3089		CLI	B@CHAR(, @XR), B@SQUO	IF CNACharacter IS NOT A QUOTE
314A	F2	01	09	3090		JNE	FZX880	* BRANCH TO PROCESS CHARACTER
314D	E2	02	01	3091		LA	@B1(, @XR), @XR	INCREMENT THE DATA POINTER
3150	BD	7D	00	3092		CLI	B@CHAR(, @XR), B@SQUO	IF QUOTE PAIR NOT SPECIFIED GO
3153	F2	01	15	3093		JNE	FZX886	* EXECUTE END OF CONSTANT RTN
				3094		*		
				3095		* PROCESS THE DATA CHARACTER - STORE IN CONSTANT BUCKET WHEN BUCKET		
				3096		* IS NOT FILLED, OTHERWISE DISREGARD THE CHARACTER.		
				3097		*		
3156	7D	E1	62	3098	FZX880	CLI	FZX883+@D1(, @BR), FZXCRR-FZXP7B	IF BUCKET ALREADY FILLED
3159	D0	84	44	3099		BH	FZX876(, @BR)	IT BYPASS CHAR AND GO PROC NEXT
				3100		*		
315C	5E	00	CF C4	3101		ALC	FZXSTS(, @BR), FZX7B1(1, @BR)	INCREMENT STATUS BYTE COUNTER
3160	6C	00	00 00	3102	FZX883	MVC	*-(, @BR), B@CHAR(1, @XR)	MOVE CURRENT CHARACTER TO BUCKET
3164	5E	00	62 C4	3103		ALC	FZX883+@D1(, @BR), FZX7B1(1, @BR)	INCREMENT BUCKET POINTER
3168	D0	87	44	3104		B	FZX876(, @BR)	BRANCH TO PROCESS NEXT CHARACTER
				3105		*		
				3106		* END OR CONSTANT - ACCESS DELIMITER FOLLOWING CONSTANT		
				3107		*		
316B	BD	40	00	3108	FZX886	CLI	B@CHAR(, @XR), B@BLNK	IF BLANK AFTER CONSTANT
316E	D0	81	BF	3109		BE	FZX910(, @BR)	* LINK TO GET ELEMENT DELIMETER
				3110		*		
				3111		* MOVE CONVERTED CONSTANT TO THE RUN-TIME STACK		
				3112		*		
3171	74	02	7F	3113		ST	FZX890+@OP1(, @BR), @XR	SAVE THE DELIMETER CORE ADDRESS
3174	35	02	0D4E	3114		L	I\$STAK, @XR	LOAD THE RUN-TIME STACK POINTER
3178	9C	12	12 E1	3115		MVC	I@LCRV-1(, @XR), FZXCRR(I@LCRV, @BR)	STACK THE CHAR CONSTANT
317C	C2	02	0000	3116	FZX890	LA	*-*, @XR	RELOAD THE DELIMETER CORE ADDR
3180	F2	87	31	3117		J	FZX906	GO EXIT THE CONVERSION ROUTINE
				3118		*		
				3119		*****		

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 243
				3121		*****				
				3122	*	INTERNAL ARITHMETIC CONSTANT STACKING				
				3123		*****				
				3124	*					
				3125	*	INTERNAL CONSTANT BUCKET CONSISTS OF 2 BYTES, THE 1ST OF WHICH				
				3126	*	CONTAINS THE CONSTANT SIGN CHARACTER - MOVE 1ST LETTER OF CONSTANT				
				3127	*	SYMBOL TO THE 2ND BUCKET BYTE				
				3128	*					
3183	D0	87	BF	3129	FZX893	B	FZX910(,@BR)		LINK TO GET 1ST SYMBOL LETTER	
3186	6C	00	D0 00	3130		MVC	FZXICB(,@BR),B@CHAR(1,@XR)		MOVE 1ST LETTER TO THE BUCKET	
318A	74	02	AA	3131		ST	FZX900+@OP1(,@BR),@XR		SAVE CURRENT CHARACTER CORE ADDR	
				3132	*					
				3133	*	SEARCH THE INTERNAL CONSTANT SYMBOL TABLE FOR MATCH WITH BUCKET				
				3134	*					
318D	D2	02	DF	3135		LA	FZXICT-FZXITL(,@BR),@XR		LOAD SYMBOL TABLE BASE ADDRESS	
3190	E2	02	04	3136	FZX896	LA	FZXITL(,@XR),@XR		INCR TABLE POINTER TO NEXT ENTRY	
3193	6D	01	D0 01	3137		CLC	FZXICB(,@BR),FZXICN(2,@XR)		COMPARE BUCKET WITH TBL ENTRY	
3197	D0	01	90	3138		BNE	FZX896(,@BR)		CONTINUE SEARCH IF NO ID MATCH	
				3139	*					
				3140	*	MATCHNINA ENTRY FOUND - STACK THE INTERNAL CONSTANT LOCATED IN				
				3141	*	VIRTUAL MEMORY AT THE ADDRESS SPECIFIED IN THE SYMBOL TABLE				
				3142	*					
319A	2C	01	144A 03	3143		MVC	I\$VADR,FZXICA(@VADDR,@XR)		SET PASING VIRTUAL ADDR PARAM	
319F	35	02	0D4E	3144		L	I\$STAK,@XR		LOAD THE RUN-TIME STACK POINTER	
31A3	C0	87	0B50	3145		B	I\$STCK		LINK TO STACK THE INTERNAL CON.	
				3146	*					
				3147	*	ACCESS DELIMITER FOLLOWING INTERNAL CONSTANT SYMBOL				
				3148	*					
31A7	C2	02	0000	3149	FZX900	LA	*-*,@XR		LOAD THE DATA BUFFER POINTER	
31AB	D0	87	BF	3150	FZX903	B	FZX910(,@BR)		LINK TO SET NEXT DATA CHARACTER	
31AE	BD	C1	00	3151		CLI	B@CHAR(,@XR),B@LETA		IF CHAR IS A LETTER OR DIGIT	
31B1	D0	02	AB	3152		BNL	FZX903(,@BR)		* LOOP TO SET NEXT CHARACTER	
				3153	*					
				3154	*	EXIT - SAVE DATA POINTER AND RETURN TO CALLING ROUTINE				
				3155	*					
31B4	C2	01	0000	3156	FZX906	LA	*-*,@BR		LOAD DATA BUFFER BASE CORE ADDR	
31B8	74	02	FF	3157		ST	FZXBPT(,@BR),@XR		STORE DATA POINTER IN BUFFER	
				3158	*					
31BB	C0	87	12D3	3159		B	I\$RTRN		RETURN TO CALLING ROUTINE	
				3160	*					
				3161		*****				

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 244
				3163		*****				
				3164	*	DATA CHARACTER 'GET' ROUTINE -				*
				3165	*	* ADVANCES DATA BUFFER POINTER (REG @XR) TO NEXT NON-BLANK CHAR				*
				3166		*****				
				3167	*					
	31BF	74	08	CE	3168	FZX910 ST	FZX916+@OP1(,@BR),@ARR	SET RETURN BRANCH ADDRESS		
				3169	*					
	31C2	E2	02	01	3170	FZX913 LA	@B1(,@XR),@XR	INCR DATA CHARACTER POINTER		
	31C5	BD	40	00	3171	CLI	B@CHAR(,@XR),B@BLNK	TEST FOR A BLANK CHARACTER		
	31C8	D0	81	C2	3172	BE	FZX913(,@BR)	REPEAT LOOP UNTIL NON-BLANK		
				3173	*					
	31CB	C0	87	0000	3174	FZX916 B	*-*	RETURN TO CALLING ROUTINE		
				3175	*					
				3176		*****				

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 245
			3178	*****	
			3179	* INPUT EXECUTION ROUTINE CONSTANTS (7TH VM PAGE)	*
			3180	*****	
			3181	*	
		31C4	3182	FZX7B1 EQU FZX913+@D1	BINARY INTEGER +1
			3183	*	
			3184	*****	
			3185	* INPUT EXECUTION ROUTINE WORK AREAS (7TH VM PAGE)	*
			3186	*****	
			3187	*	
		31CF	3188	FZXBKT EQU *	CONVERSION BUCKET BASE ADDRESS
31CF		31E1	3189	DS CL(I@LCRV)	CONVERSION BUCKET AREA
31E2 40		31E2	3190	FZXBLK DC AL1(B@BLNK)	BLANK INITIALIZATION CHAR
			3191	*	
		31CF	3192	FZXSTS EQU FZXBKT+I@STAT	BUCKET STATUS BYTE ADDRESS
		31D0	3193	FZXCR1 EQU FZXBKT+I@STAT+1	BUCKET 1ST CHAR FIELD BYTE ADDR
		31E1	3194	FZXCRR EQU FZXBKT+I@LCRF	BUCKET LAST CHAR FIELD BYTE ADDR
			3195	*	
		31D0	3196	FZXICB EQU FZXBKT+1	INTERNAL CONSTANT BUCKET ADDR
			3197	*	
			3198	*****	

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 246
				3200		*****				
				3201		* INTERNAL CONSTANT SYMBOL TABLE				*
				3202		*****				
				3203		*				
				0004	3204	FZXITL EQU 4	LENGTH OF A SYMBOL TABLE ENTRY			
				0001	3205	FZXICN EQU 1	DISP FOR TABLE ENTRY CON SYMBOL			
				0003	3206	FZXICA EQU 3	DISP FOR TABLE ENTRY CON VADDR			
				3207		*				
				31E3	3208	FZXICT EQU *	SYMBOL TABLE STARTING ADDRESS			
				3209		*				
31E3	4E			31E3	3210	DC AL1(B@PLUS)	SIGN IDENTIFIER FOR +&SQR2			
31E4	E2			31E4	3211	DC AL1(B@CIS2)	LETTER IDENTIFIER FOR &SQR2			
31E5	F500			31E6	3212	DC AL(@VADDR)(I@ICBA+0*I@LPFV)	VIRTUAL ADDR OF +&SQR2			
				3213		*				
31E7	4E			31E7	3214	DC AL1(B@PLUS)	SIGN IDENTIFIER FOR +&PI			
31E8	D7			31E8	3215	DC AL1(B@CIPI)	LETTER IDENTIFIER FOR &PI			
31E9	F505			31EA	3216	DC AL(@VADDR)(I@ICBA+1*I@LPFV)	VIRTUAL ADDR OF +&PI			
				3217		*				
31EB	4E			31EB	3218	DC AL1(B@PLUS)	SIGN IDENTIFIER FOR +&E			
31EC	C5			31EC	3219	DC AL1(B@CIEX)	LETTER IDENTIFIER FOR &E			
31ED	F50A			31EE	3220	DC AL(@VADDR)(I@ICBA+2*I@LPFV)	VIRTUAL ADDR OF +&E			
				3221		*				
31EF	60			31EF	3222	DC AL1(B@MINS)	SIGN IDENTIFIER FOR -&SQR2			
31F0	E2			31F0	3223	DC AL1(B@CIS2)	LETTER IDENTIFIER FOR &SQR2			
31F1	F50F			31F2	3224	DC AL(@VADDR)(I@ICBA+3*I@LPFV)	VIRTUAL ADDR OF -&SQR2			
				3225		*				
31F3	60			31F3	3226	DC AL1(B@MINS)	SIGN IDENTIFIER FOR -&PI			
31F4	D7			31F4	3227	DC AL1(B@CIPI)	LETTER IDEV/F/ER FOR &PI			
31F5	F514			31F6	3228	DC AL(@VADDR)(I@ICBA+4*I@LPFV)	VIRTUAL ADDR OF -&PI			
				3229		*				
31F7	60			31F7	3230	DC AL1(B@MINS)	SIGN IDENTIFIER FOR -&E			
31F8	C5			31F8	3231	DC AL1(B@CIEX)	LETTER IDENTIFIER FOR &E			
31F9	F519			31FA	3232	DC AL(@VADDR)(I@ICBA+5*I@LPFV)	VIRTUAL ADDR OF -&E			
				3233		*				
				3234		*****				

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 247
				3236	*****	
				3237	* VIRTUAL MEMORY INPUT EXECUTION ROUTINE (8TH VM PAGE)	*
				3238	* ELEMENT CONVERSION ROUTINE FOR NUMERIC ARITHMETIC CONSTANTS	*
				3239	*	*
				3240	* INPUT -	*
				3241	* * REGISTER @XR - REFECENCES 1ST CHARACTER OF CONSTANT	*
				3242	*	*
				3243	* OUTPUT -	*
				3244	* * CONVERTED VALUE STACKED BEGINNING AT I\$STAK - STACKED DATA	*
				3245	* * FORMAT IS PACKED FLOATING POINT (CURRENT PRECISION)	*
				3246	* * REGISTER @XR - REFERENCES DELIMITER FOLLOWING CONVERTED CON.	*
				3247	*****	
				3248	*	
				3249	* ESTABLISH ADDRESSABILITY FOR INPUT ROUTINE (8TH VM PAGE)	
				3250	*	
3200				3251	*FZXP8B VPAGE 0 SET 8TH PAGE ADDRESSABILITY	
				3252	ORG *,256,0 SET STARTING ADDRESS	
			3200	3253	FZXP8B EQU * START OF PROGRAM CODING	
3101				3254	ORG *-255 RESET IAR TO PAGE	
3200				3255	ORG *,256,0 * BOUNDARY ADDRESS	
			3200	3256	USING *,@BR SET PAGE BASE ADDRESS	
3200				3257	ORG FZXP8B RESET STARTING ADDRESS	
				3258	*** END OF EXPANSION ***	
				3259	*	
				3260	* ENTRY - INITIALIZE FOR NUMERIC CONSTANR CONVERSION	
				3261	*	
3200	7C	80	EA	3262	FZX920 MVI FZXEXP(,@BR),B@NXZR SET ZERO FLT PT NORMALIZED EXP	
3203	54	60	F1 E7	3263	ZAZ FZXMNR(I@PREC,@BR),FZX8D0(1,@BR) ZERO THE VALUE MANTISSA	
3207	7C	EB	41	3264	MVI FZX950+@D1(,@BR),FZXMN1-FZXP8B INITLZ MANTISSA CHAR D1SP	
320A	D0	87	BE	3265	B FZX990(,@BR) LINK TO ESTABLISH MANTISSA SIGN	
				3266	* * AND ACCESS 1ST NON-SIGN CHAR	
				3267	*	
				3268	*****	

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 248
				3270			*****	
				3271			* NUMERIC CONSTANT MANTISSA CONVERSION	*
				3272			*****	
				3273			*	
				3274			* INCREMENT PAST INSIGNIFICANT LEADING ZEROS	
				3275			*	
320D	D0	81	D3	3276	FZX923	BE	FZX992(,@BR)	LINK TO GET NEXT DATA CHARACTER
3210	D0	81	0D	3277		BE	FZX923(,@BR)	* UNTIL NON-ZERO CHAR IS FOUND
				3278			*	
				3279			* ASSUME AN INTEGER COMPONENT TEST FOR FRACTIONAL COMPONENT ONLY	
				3280			*	
3213	7C	80	33	3281	FZX926	MVI	FZX943+@Q(,@BR),@NOP	SET FRACTION COMPONENT SW OFF
3216	BD	4B	00	3282		CLI	B@CHAR(,@XR),B@DPNT	IF CHARACTER NOT A DECIMAL POINT
3219	F2	01	10	3283		JNE	FZX940	* GO TEST FOR INTEGER DIGITS
				3284			*	
				3285			* FRACTIONAL COMPONENT ONLY - ADJUST EXPONENT FOR ZEROS AFTER POINT	
				3286			*	
321C	F2	87	04	3287		J	FZX933	SKIP TO SET CEAR AFTER POINT
321F	5F	00	EA D8	3288	FZX930	SLC	FZXEXP(,@BR),FZX8B1(1,@BR)	DECREMENT NORMALIZED E*PONENT
3223	D0	87	D3	3289	FZX933	B	FZX992(,@BR)	LINK TO GET NEXT DATA CN&RACTER
3226	D0	81	1F	3290		BE	FZX930(,@BR)	LOOP IF CHAR IS A DECIMAL ZERO
				3291			*	
				3292			* TEST FOR FRACTIONAL DIGITS WHICH DO NOT AFFECT EXPONENT	
				3293			*	
3229	7C	87	33	3294	FZX936	MVI	FZX943+@Q(,@BR),@UCB	SET FRACTION COMPONENT SW ON
322C	BD	F0	00	3295	FZX940	CLI	B@CHAR(,@XR),B@DEC0	IF NO MORE FRACTIONAL DIGITS
322F	F2	82	27	3296		JL	FZX956	* GC WRAP-UP MANTISSA PROCESSING
				3297			*	
				3298			* PROCESS MANTISSA DIGITS - STORE DIGITS IN SUCCESSIVE MANTISSA	
				3299			* POSITIONS UNTIL MANTISSA BUCKET IS FILLED - ADJUST NORMALATED	
				3300			* EXPONENT FOR SIGNIFICANT INTEGER DIGITS ONLY.	
				3301			*	
3232	F2	00	04	3302	FZX943	JC	FZX946,*-*	BRANCH IF DIGIT IS FRACTIONAL
3235	5E	00	EA D8	3303		ALC	FZXEXP(,@BR),FZX8B1(1,@BR)	INCREMENT NORMALIZED EXPONENT
				3304			*	
3239	7D	F1	41	3305	FZX946	CLI	FZX950+@D1(,@BR),FZXMNR-FZXP8B	IF BUCKET ALREADY FILLED
323C	F2	84	08	3306		JH	FZX953	* BYPASS DIGIT AND GO PROC NEXT
323F	6C	00	00 00	3307	FZX950	MVC	*-*(,@BR),B@CHAR(1,@XR)	* ELSE MOVE DIGIT TO MANTISSA
3243	5E	00	41 D8	3308		ALC	FZX950+@D1(,@BR),FZX8B1(1,@BR)	AND BUMP BUCKET POINTER
				3309			*	
3247	D0	87	D3	3310	FZX953	B	FZX992(,@BR)	LINK TO GET NEXT DATA CHARACTER
324A	D0	02	32	3311		BNL	FZX943(,@BR)	LOOP IF CHAR IS A DECIMAL DIGIT
				3312			*	
				3313			* TEST FOR A DECIMAL POINT FOLLOWING SIGNIFICANT DIGITS	
				3314			*	
324D	BD	4B	00	3315		CLI	B@CHAR(,@XR),B@DPNT	IF CHARACTER NOT A DECIMAL POINT
3250	F2	01	06	3316		JNE	FZX956	* GO WRAP-UP MANTISSA PROCESSING
3253	D0	87	D3	3317		B	FZX992(,@BR)	LINK TO GET NEXT DATA CHARACTER
3256	D0	87	29	3318		B	FZX936(,@BR)	BRANCH TO PROCESS FRACTIONALS
				3319			*	
				3320			* TEST MANTISSA SIGN - MODIFY SIGN ZONE FOR NEGATIVE VALUE	
				3321			*	
3259	7D	80	8E	3322	FZX956	CLI	FZX973+@Q(,@BR),@NOP	IF MANTISSA SIGN IS POSITIVE
325C	F2	81	03	3323		JE	FZX960	* GO BEGIN EXPONENT PROCESSING
325F	7B	20	F1	3324		SBF	FZXSGN(,@BR),B@ZPOS-B@ZNEG	* ELSE SET NEG MANTISSA ZONE
				3325			*	

[illegible]

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 250
				3328			*****	
				3329	*		NUMERIC CONSTANT EXPONENT CONVERSION	*
				3330			*****	
				3331	*			
				3332	*		TEST FOR E-FORMAT SPECIFICATION	
				3333	*			
3262	BD	C5	00	3334	FZX960	CLI	B@CHAR(,@XR),B@EXPC	IF NO E-FORMAT SPECIFICATION
3265	F2	01	33	3335		JNE	FZX980	* SKIP PAST EXPONENT PROCESSING
				3336	*			
3268	D0	87	D3	3337		B	FZX992(,@BR)	LINK TO GET NEXT DATA CHARACTER
326B	D0	87	BE	3338		B	FZX990(,@BR)	LINK TO ESTABLISH SPEC EXPONENT
				3339	*			* SIGN AND GET 1ST EXP DIGIT
				3340	*			
				3341	*		ESTABLISH EXPONENT DIGIT(S) FOR CONVERSION TO BINARY	
				3342	*			
326E	7C	00	E9	3343		MVI	FZX8DX(,@BR),@ZERO	SET EXP AREA RH BYTE = BINARY 0
3271	5C	00	E8 E9	3344	FZX963	MVC	FZX8DX-1(,@BR),FZX8DX(1,@BR)	SHIFT UP ABEA CONTENTS LEFT
3275	68	03	E9 00	3345		MNN	FZX8DX(,@BR),B@CHAR(,@XR)	MOVE EXP DIGIT NUMERIC TO AREA
3279	D0	87	D3	3346		B	FZX992(,@BR)	* RH BYTE, LINK TO GET NEXT CHAR
327C	D0	02	71	3347		BNL	FZX963(,@BR)	LOOP IF CHAR IS EXPONENT DIGIT
327F	F2	87	04	3348		J	FZX970	* ELSE GO CONV EXPONENT TO BIN.
				3349	*			
				3350	*		CONVERT EXPONENT TO A BINARY NUMBER	
				3351	*			
3282	5E	00	E9 E6	3352	FZX966	ALC	FZX8DX(,@BR),FZXB10(1,@BR)	ADD BINARY 10 TO EXP UNITS
3286	5F	00	E8 D8	3353	FZX970	SLC	FZX8DX-1(,@BR),FZX8B1(1,@BR)	DECR EXPONENT TENS DINT
328A	D0	02	82	3354		BNM	FZX966(,@BR)	LOOP UNTIL TENS DIGIT < ZERO
				3355	*			
				3356	*		MODIFY NORMALIZED EXPONENT WITH SPECIFIED EXPONENT	
				3357	*			
328D	F2	00	07	3358	FZX973	JC	FZX976,*-*	BRANCH IF SPEC EXP IS NEGATIVE
3290	5E	00	EA E9	3359		ALC	FZXEXP(,@BR),FZX8BX(1,@BR)	ADD SPEC EXPONENT MAGNITUDE
3294	F2	87	04	3360		J	FZX980	* TO NORM EXP, GO TEST MANTISSA
3297	5F	00	EA E9	3361	FZX976	SLC	FZXEXP(,@BR),FZX8BX(1,@BR)	SUBTRACT SPEC EYP MAGNITUDE
				3362	*			* FROM NORMALIZED EXPONENT
				3363	*			
				3364	*		TEST FOR SIGNIFICANT UNITS IN MANTISSA	
				3365	*			
329B	7D	F0	EB	3366	FZX980	CLI	FZXMN1(,@BR),B@DEC0	IF LEADING MANTISSA DIGIT NOT
329E	F2	01	06	3367		JNE	FZX984	* ZERO, GO STACK THE ELEMENT
32A1	7C	1E	EA	3368		MVI	FZXEXP(,@BR),B@NXLO	* ELSE SET MINIMUM EXPONENT
32A4	7A	F0	F1	3369		SBN	FZXSGN(,@BR),B@ZPOS	* AND FORCE MANTISSA POSITIVE
				3370	*			
				3371	*		MOVE CONVERTED FLOATING POINT VALUE TO RUN-TIME STACK	
				3372	*			
32A7	74	02	B9	3373	FZX984	ST	FZX986+@OP1(,@BR),@XR	SAVE ELEMENT DELIMITER CADDR
32AA	35	02	0D4E	3374		L	I\$STAK,@XR	LOAD THE RUN-TIME STACK POINTER
32AE	9C	07	07 F1	3375		MVC	I@LUFV-1(,@XR),FZXMNR(I@LUFV,@BR)	MOVE VALUE TO STACK
32B2	C0	87	0A85	3376		B	I\$CUPF	LINK TO PACK THE STACKED VALUE
				3377	*			
				3378	*		EXIT - RESTORE BUFFER POINTER AND RETURN TO CALLING ROUTINE	
				3379	*			
32B6	C2	02	0000	3380	FZX986	LA	*-*,@XR	RELOAD ELEMENT DELIMITER CADDR
				3381	*			
32BA	C0	87	12D3	3382		B	I\$RTRN	RETURN TO CALLING ROUTINE
				3383	*			

[illegible]

DFPRNT - MATRIX PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 252
				3386		*****		
				3387	*	ARITHMETIC	CONSTANT SIGN PROCESSING ROUTINE	*
				3388	*	* SETS	SIGN SWITCH (FZX973+@Q) DEPENDING ON CHARACTER REFFRENCED	*
				3389	*	BY REGISTER	@XR -	*
				3390	*	* SWITCH	SET TO CODE @NOP WHEN CHARACTER NOT A SIGN	*
				3391	*	* SWITCH	SET TO CODE @NOP WHEN CHARACTER IS PLUS SIGN	*
				3392	*	* SWITCH	SET TO CODE @UCB WHEN CHARACTER IS MINUS SIGN	*
				3393	*	* ADVANCES	DATA BUFFER POINTER (REG @XR) TO NEXT NON-BLANK	*
				3394	*	CHARACTER	WHEN REFERENCED CHARACTER IS A SIGN	*
				3395	*	* SETS	PSR DEPENDING ON TYPE OF FINALLY REFERENCED CHARACTER -	*
				3396	*	* LOW	- NON-DECIMAL CHARACTER	*
				3397	*	* EQUAL	- DECIMAL ZERO CHARACTER	*
				3398	*	* HIGH	- DECIMAL DIGIT OTHER THAN ZERO	*
				3399		*****		
				3400	*			
32BE	74	08	E5	3401	FZX990	ST	FZX998+@OP1(,@BR),@ARR	SET RETURN BRANCH ADDRESS.
				3402	*			
32C1	7C	80	8E	3403		MVI	FZX973+@Q(,@BR),@NOP	SET SIGN SWITCH FOR POSITIVE
32C4	BD	4E	00	3404		CLI	B@CHAR(,@XR),B@PLUS	IF CHARACTER IS A PLUS SIGN
32C7	F2	81	0C	3405		JE	FZX994	* SKIP TO GET THE NENT CHARACTER
32CA	BD	60	00	3406		CLI	B@CHAR(,@XR),B@MINS	IF CHARACTER NOT A MINUS SIGN
32CD	F2	01	0F	3407		JNE	FZX996	* SKIP TO SET PSR AND RETURN
32D0	7C	87	8E	3408		MVI	FZX973+@Q(,@BR),@UCB	SET SIGN SWITCH FOR NEGATIVE
				3409	*			
				3410		*****		

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 253
		3412		*****	
		3413	*	DATA CHARACTER 'GET' ROUTINE -	*
		3414	*	* ADVANCES DATA BUFFER POINTER (REG @XR) TO NEXT NON-BLANK CHAR	*
		3415	*	* SETS PSR DEPENDING ON TYPE OF FINALLY REFERENCED CHARACTER	*
		3416	*	* LOW - NON-DECIMAL CHARACTER	*
		3417	*	* EQUAL - DECIMAL ZERO CHARACTER	*
		3418	*	* HIGH - DECIMAL DIGIT OTHER THAN ZERO	*
		3419		*****	
		3420	*		
32D3	74 08 E5	3421	FZX992 ST	FZX998+@OP1(,@BR),@ARR	SET RETURN BRANCH ADDRESS
		3422	*		
32D6	E2 02 01	3423	FZX994 LA	@B1(,@XR),@XR	INCR DATA CHARACTER POINTER
32D9	BD 40 00	3424	CLI	B@CHAR(,@XR),B@BLNK	TEST FOR A BLANK CHARACTER
32DC	D0 81 D6	3425	BE	FZX994(,@BR)	REPEAT LOOP UNTIL NON-BLANK
		3426	*		
32DF	BD F0 00	3427	FZX996 CLI	B@CHAR(,@XR),B@DEC0	SET PSR FOR NEW CHARACTER
32E2	C0 87 0000	3428	FZX998 B	*-*	RETURN TO CALLING ROUTINE
		3429	*		
		3430		*****	

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 254
		3432		*****	
		3433		* INPUT EXECUTION ROUTINE CONSTANTS (8TH VM PAGE)	*
		3434		*****	
		3435		*	
32E6 0A		32D8 3436	FZX8B1 EQU	FZX994+@D1	BINARY INTEGER +1
		32E6 3437	FZXB10 DC	IL1 '10'	BINARY INTEGER +10
32E7 F0		32E7 3438	FZX8D0 DC	DL1 '0'	DECIMAL INTEGER 0
		3439		*	
		3440		*****	
		3441		* INPUT EXECUTION ROUTINE WORK AREAS (8TH VM PAGE)	*
		3442		*****	
		3443		*	
32E8		32E9 3444	FZX8DX DS	CL(FZXDXL)	DECIMAL EXPONENT WORK AREA
		32E9 3445	FZX8BX EQU	FZX8DX	BINARY SPECIFIED EXPONENT CADDR
		3446		*	
		32EA 3447	FZX8BK EQU	*	CONVERSION BUCKET BASE ADDRESS
32EA		32F1 3448	DS	CL(I@LUFV)	CONVERSION BUCKET AREA
		3449		*	
		32EA 3450	FZXEXP EQU	FZX8BK+I@UEXP	BUCKET EXPONENT BYTE ADDRESS
		32EB 3451	FZXMN1 EQU	FZX8BK+I@UMN1	BUCKET IST MANTISSA BYTE ADDR
		32F1 3452	FZXMNR EQU	FZX8BK+I@UMNR	BUCKET RIGHT MANTISSA BYTE ADDR
		32F1 3453	FZXSGN EQU	FZX8BK+I@SIGN	BUCKET MANTISSA SIGN BYTE ADDR
		3454		*	
		3455		*****	

DFPRNT - MATRIX PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 255
			3457	*****	
			3458	* INPUT EXECUTION ROUTINE COMMON EQUATES	*
			3459	*****	
			3460	*	
		0DC8	3461	FZXBCA EQU I\$PUB1	INPUT BUFFER CADDR SAVE AREA
			3462	*	
		00FF	3463	FZXBPT EQU 255	DISP FOR DATA BUFFER POINTER
			3464	*	
		00F0	3465	FZXER0 EQU C'0'	CODE SPECIFYING ERROR 800
		00F1	3466	FZXER1 EQU C'1'	CODE SPECIFYING ERROR 801
		00F2	3467	FZXER2 EQU C'2'	CODE SPECIFYING ERROR 802
		00F3	3468	FZXER3 EQU C'3'	CODE SPECIFYING ERROR 803
		00F4	3469	FZXER4 EQU C'4'	CODE SPECIFYING ERROR 804
			3470	*	
			3471	*****	
			3472	*	
			3473	*** END OF VIRTUAL MEMORY INPUT EXECUTION ROUTINE CODING ***	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 256
		3475		*****	*
		3476	*	5703-XM1 COPYRIGHT IBM CORP. 1970	*
		3477	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083	*
		3478	*		*
		3479		*****	*
		3480	*	*STATUS	*
		3481	*	VERSION 1 MODIFICATION 0	*
		3482	*		*
		3483	*	*FUNCTION	*
		3484	*	* FZREAD EXECUTION CAUSES THE CURRENTLY REFERENCED ELEMENT IN THE	*
		3485	*	PROGRAM 'DATA' FILE TO BE MOVED TO THE RUN-TIME STACK. THE	*
		3486	*	'DATA' FILE POINTER IS ADVANCED TO REFREVE THE NEXT 'DATA'	*
		3487	*	FILE ELEMENT.	*
		3488	*	* THIS ROUTINE OPERATES ON THE FOLLOWING PSEUDO INSTRUCTIONS TO	*
		3489	*	ACCESS AND STACK THE CURRENTY REFERENCED PROGRAM 'DATA' FILE	*
		3490	*	ELEMENT.	*
		3491	*	* 'DCA' - DEFINE CONSTANT ADDRESS (FORMAT - OP VADR)	*
		3492	*	THE DATA ELEMENT AT VIRTUAL ADDRESS VADR IS DEFINED AS AN	*
		3493	*	ELEMENT IN THE 'DATA' FILE. THE POSITION OF THE ELEMENT	*
		3494	*	IN THE FILE IS DIRECTLY RELATED TO THE POSITION OF THE	*
		3495	*	'DCA' INSTRUCTION WITH RESPECT TO OTHER 'DCA' INSTRUCTIONS	*
		3496	*	IN THE PROGRAM.	*
		3497	*	* 'DDL' - DEFINE 'DATA' LINKAGE (FORMAT - OP VADR)	*
		3498	*	'DDL' ALWAYS FOLLOWS A STRING OF 'DCA' INSTRUCTIONS.	*
		3499	*	THE 'DCA' INSTRUCTION BEGINNING AT VADR IS THE NEXT	*
		3500	*	SEQUENTIAL 'DCA' IN THE PROGRAM. WHEN VADR = X'0000',	*
		3501	*	'DDL' MARKS THE END OF THE 'DATA' FILE.	*
		3502	*	* 'EOP' - END OF PMC PAGE (FORMAT - OP)	*
		3503	*	EACH PSEUDO MACHINE CODE VIRTUAL PAGE IS TERMINATED WITH	*
		3504	*	AT LEAST ONE 'EOP' INSTRUCTION. 'EOP' EXECUTION RESULTS	*
		3505	*	IN CONTROL BEING PASSED TO THE FIRST PSUEDO INSTRUCTION	*
		3506	*	WHICH APPEARS IN THE NEXT SEQUENTIAL VIRTUAL PAGE.	*
		3507	*	* 'DATA' FILE POINTER I\$DATA CONTAINS EITHER THE VIRTUAL ADDRESS	*
		3508	*	OF A 'DCA' INSTRUCTION OR THAT OF A 'DDL' OR 'EOP' FOLLOWING A	*
		3509	*	STRING OF 'DCA' INSTRUCTIONS. IN THE LATTER CASE, THE CURRENT	*
		3510	*	'DCA' INSTRUCTION IS THAT INDICATED BY THE 'DDL' OR 'EOP'.	*
		3511	*	THE ELEMENT REFERENCED BY THE OPERAND OF THE CURRENT 'DCA'	*
		3512	*	INSTRUCTION IS STACKED, AND I\$DATA IS INCREMENTED TO REFERENCE	*
		3513	*	THE NEXT 'DCA' INSTRUCTION.	*
		3514	*		*
		3515	*	*ENTRY POINTS	*
		3516	*	THIS ROUTINE HAS A SINGLE ENTRY POINT - FZREAD - WHOSE FUNCTION	*
		3517	*	IS DEFINED ABOVE. CALLING SEQUENCE IS	*
		3518	*	B I\$CALL	*
		3519	*	DC AL2(V\$XS?O)	*
		3520	*	WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL ADDRESS	*
		3521	*	OF ENTRY POINT FXREAD. EXECUTION IS SUBJECT TO INPUT CONDITIONS	*
		3522	*	DESCRIBED BELOW.	*
		3523	*		*
		3524	*	*INPUT	*
		3525	*	* I\$STAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER. THIS IS TO	*
		3526	*	CONTAIN THE CORE ADDRESS OF THE FIRST AVAILABLE STACK LOCATION.	*
		3527	*	* I\$DATA - 2 BYTES, FOR THE 'DATA' FILE POINTER. THIS IS TO	*
		3528	*	CONTAIN THE VIRTUAL ADDRESS OF THE CURRENT 'DCA' INSTRUCTION	*
		3529	*	OR THAT OF A 'DDL' OR 'EOP' INDICATING THE 'DCA' INSTRUCTION.	*
		3530	*	* PMC 'DATA' FILE - 'DATA' FILE ELEMENT-REFERENCING PSEUDO	*

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 257

```

3531 *      INSTRUCTIONS GENERATED FOR EACH 'DATA' FILE STATEMENT AND      *
3532 *      ACCESSED USING FILE POINTER I$DATA.                             *
3533 *                                                                      *
3534 *OUTPUT                                                                    *
3535 *      * I$STAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER, WHEN NO      *
3536 *      ERROR OCCURS, THIS CONTAINS THE CORE ADDRESS OF THE LEFTMOST    *
3537 *      BYTE OF THE 'DATA' FILE ELEMENT STACKED DURING FZREAD EXECUTION.*
3538 *      * I$DATA - 2 BYTES, FOR THE 'DATA' FILE POINTER.  WHEN NO ERROR  *
3539 *      OCCURS, THIS CONTAINS THE VIRTUAL ADDRESS OF THE NEXT SEQUEN-    *
3540 *      TIAL 'DCA' INSTRUCTION OR THAT OF A 'DDL' OR 'EOP' INDICATING    *
3541 *      THE NEXT 'DCA' INSTRUCTION.                                       *
3542 *      * I$ERRC - 1 BYTE, FOR THE ERROR CONDITION CODE.  THIS CONTAINS *
3543 *      A NULL CODE (I@NERR) WHEN NO ERROR CONDITION EXISTS OR AN        *
3544 *      ERROR CODE SPECIFYING THE PARTICULAR ERROR CONDITION DISCOVERED.*
3545 *      * RUN-TIME STACK - WHEN NO ERROR CONDITION OCCURS, THIS CONTAINS *
3546 *      THE CURRENT 'DATA' FILE ELEMENT AT THE TOP STACK POSITION.        *
3547 *                                                                      *
3548 *EXTERNAL REFERENCES                                                        *
3549 *      * I$STCK - ENTRY POINT FOR INTERPRETER ELEMENT STACKING ROUTINE.  *
3550 *      * ISLDYR - ENTRY POINT FOR PAGING MODLLE CONVERT AND LOAD @XR RTN.*
3551 *      * I$QTRN - ENTRY POINT FOR PAGING MODLLE V.M. TETURN CONTROL RTN.*
3552 *      * I$STAK - 2 BYTES, FOR THE STACK POINTER.                       *
3553 *      * I$DATA - 2 BYTES, FOR THE PROGRAM 'DATA' FILE POINTER.         *
3554 *      * I$VADR - 2 BYTES, FOR PAGING MODULE VIRTUAL ADDRESS PARAMETER.  *
3555 *      * I$SLN5 - 1 BYTE, FOR ELEMENT STACKING LENGTH PARAM TO ISTACK.  *
3556 *      * I$ERRC - 1 BYTE, FOR THE INTERPRETER EXECUTION ERROR CODE.     *
3557 *                                                                      *
3558 *EXITS, NORMAL                                                            *
3559 *      CONTROL IS ALWAYS PASSED TO THE PAGING ROUTINE AT ENTRY POINT    *
3560 *      I$RTRN (IPGRTN) FOR A RETURN TO CALLING PROGRAM.                 *
3561 *                                                                      *
3562 *EXITS, ERROR                                                            *
3563 *      CONTROL IS PASSED TO THE PAGING ROUTINE AT ENTRY POINT I$RTRN    *
3564 *      (IPFRTN) WITH THE PARAMETER I$ERRC CONTAINING THE APPROPRIATE    *
3565 *      ERROR MESSAGE CODE.                                               *
3566 *                                                                      *
3567 *TABLES/WORK AREAS                                                        *
3568 *      FZREAD PMC EXECUTION BRANCH ADDRESS TABLE - 6 BYTES, FOR 'DATA' *
3569 *      FILE DEFINITION PMC OPCODE TRANSLATION TO AN FZREAD ENTRY POINT  *
3570 *      ADDRESS.  THIS TABLE CONSITS OF THREE 2 BYTE ENTRIES CONTAINING *
3571 *      THE FOLLOWING INFORMATION -                                       *
3572 *      * BYTE 0 - DUMMY SPACER.                                          *
3573 *      * BYTE 1 - PAGE DISPLACEMENT WITHIN FZREAD FOR THE INTERNAL      *
3574 *      ENTRY POINT ASSOCIATED WITH A 'DCA', 'DDL' OR 'EOP' PSEUDO      *
3575 *      INSTRUCTION.                                                     *
3576 *                                                                      *
3577 *ATTRIBUTES                                                                *
3578 *      * REUSABLE                                                         *
3579 *      * NATURALLY RELOCATBLE                                           *
3580 *                                                                      *
3581 *CHARACTER CODE DEPENDENCY                                                *
3582 *      THE OPERATION OF THIS MODULE DOES NOT DEPEND UPON A PARTICULAR  *
3583 *      INTERNAL REPRESENTATION OF THE EXTERNAL CHARACTER SET.          *
3584 *                                                                      *
3585 *NOTES                                                                    *
3586 *      ERROR PROCEDURES                                                 *

```


ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 258
		3587	*	* ERROR 1 - FILE POINTER CONTAINS AN INVALID 'DATA'			*
		3588	*	FILE VIRTUAL ADDRESS. AN ERROR CODE FOR THE MESSAGE			*
		3589	*	'NO DATA STATEMENT SPECIFIED IS ESTABLISHED IN INTERPRETER			*
		3590	*	PARAMETER I\$ERRC.			*
		3591	*	* ERROR 2 - A 'DDL' INSTRUCTION WITH OPERAND X'0000' IS EN-			*
		3592	*	COUNTERED WHILE ATTEMPTING TO ACCESS THE NEXT 'DCA' INSTRU-			*
		3593	*	TION. AN ERROR CODE FOR THE MESSAGE 'INSUFFICIENT DATA FOR			*
		3594	*	READ' IS ESTABLISHED IN INTERPRETER PARAMETER I\$ERRC.			*
		3595	*	* IN EACH OF THESE CASES, CONTROL IS PASSED IMMEDIATELY TO			*
		3596	*	PAGING MODULE ENTRY POINT I\$RTRN (IPGRTN).			*
		3597	*				*
		3598	*	REGISTER USAGE			*
		3599	*	* REGISTER @BR IS TO CONTAIN THE CORE PAGE BASE ADDRESS			*
		3600	*	ESTABLISHED THROUGH PAGING MODULE CONTROL FOR THE PAGE WHICH			*
		3601	*	INCLUDES FZREAD, AND IS RESTORED THROUGH THE PAGING MODULE.			*
		3602	*	* RESISTER @XR IS NOT SAVED. IT IS USED IN FZREAD FOR GENERAL			*
		3603	*	PURPOSE INDEXING OPERATIONS.			*
		3604	*				*
		3605	*	SAVED/RESTORED AREAS			*
		3606	*	NONE			*
		3607	*				*
		3608	*	MODIFICATION CONSIDERATIONS			*
		3609	*	'DATA' FILE ELEMENT REFERENCING PMC OPERATION IS BASED UPON			*
		3610	*	THE SEQUENCE AND LENGTH OF THE ENTRIES IN THE FZREAD PSEUDO			*
		3611	*	INSTRUCTION BRANCH ADDRESS TABLE. TABLE ENTRIES ARE SELECTED			*
		3612	*	USING THE NUMERIC REPRESENTATION OF OPCODE 'EOP' AS A BASE			*
		3613	*	DISPLACEMENT, AND ANY CHANGES TO THE RELATIONSHIP BETWEEN THE			*
		3614	*	CONSTANTS FOR ALL OPCODES OPERATED ON BY THIS ROUTINE MUST			*
		3615	*	TAKE FULL CONSIDERATIONS OF THIS TABLE USAGE AND ORGANIZATION.			*
		3616	*				*
		3617	*	REQUIRED MODULES			*
		3618	*	* @SYSEQ - COMMON SYSTEM EQUATES.			*
		3619	*	* @ERMEQ - SYSTEM ERROR MESSAGE CODE EQUATES			*
		3620	*	* \$B@EQU - COMPILER PARAMETER AND CONSTANT EQUATES.			*
		3621	*	* \$I\$EQU - INTERPRETER FIXED LOCATION ADDRESS EQATES.			*
		3622	*	* \$I@SEQ - INTERPRETER PARAMETER EQUATES (FOR STD. PREC, ONLY)			*
		3623	*	* \$I@LEQ - INTERPRETER PARAMETER EQUATES (FOR LONG PREC, ONLY)			*
		3624	*				*
		3625	*	OTHER			*
		3626	*	NONE			*
		3627	*	*****			*

FZREAD - S/3 BASIC INTERPRETER STATEMENT EXEC RTN

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 259
					3629	*****	*****	
					3630	* START OF READ STATEMENT EXECUTION MODULE	*	
					3631	*****	*****	
					3632	*		
					3633	* ESTABLISH ADDRESSABILITY FOR THE READ STATEMENT ROUTINE		
					3634	*		
3300					3635	ORG *,B@LVPG,0	BEGIN AT PAGE BOUNDARY	
				3300	3636	USING *,@BR	DEFINE READ RTN BASE ADDRESS	
					3637	*		
					3638	* ENTER FZREAD - TEST FOR A DATA STATEMENT SPECIFICATION.		
					3639	*		
				3300	3640	FZREAD EQU *	FZREAD ENTRY POINT	
					3641	*		
3300	3D	56	0D52		3642	CLI I\$DATA-1,@VENTA	IF DATA POINTER IS DEFINED	
3304	F2	02	08		3643	JNL FZR020	* GO CONTINUE 'READ' EXECUTION	
					3644	*		
					3645	* NO DATA STATEMENT - SET 'NO DATA STATEMENT SPECIFIED' ERROR MESSAGE		
					3646	*		
3307	3C	BE	0CBC		3647	FZR010 MVI I\$ERRC,@E720	SET INTERPRETER ERROR CODE	
330B	C0	87	12D3		3648	B I\$RTRN	RETURN TO TERMINATE EXECUTION	
					3649	*		
					3650	* LOAD THE DATA PMC PAGE INTO CORE VIRTUAL MEMORY - THIS PAGE CONTAINS		
					3651	* (IN GENERAL) A SERIES OF 'DCA' INSTRUCTIONS WHICH DEFINE THE VADDR'S		
					3652	* OF THE CONSTANTS WHICH COMPRISE THE PROGRAM DATA FILE.		
					3653	*		
330F	4C	01	19 0D53		3654	FZR020 MVC FZR030(,@BR),I\$DATA(@VADDR)	SET PAGING PARAMETER TO LOAD	
					3655	*	* CURRENT DATA FILE OP CODE	
3314	C0	87	1330		3656	B I\$LDXR	LINK TO LOAD CURR DATA FILE PMC	
3318				3319	3657	FZR030 DS CL(@VADDR)	VADDR OF CURR DATA FILE OP CODE	
					3658	*		
					3659	* ESTABLISH BRANCH ADDRESS FROM OP CODE DISPLACEMENT TABLE		
					3660	*		
331A	74	02	2B		3661	FZR040 ST FZR060+@OP1(,@BR),@XR	SAVE THE DATA FILE OP CODE CADDR	
331D	6C	00	27 00		3662	MVC FZR050+@DD2(,@BR),I@XOPC(B@LCOP,@XR)	MOVE OP CODE TO DISP	
3321	D2	02	06		3663	LA FZRBAT-B@CEOP+1(,@BR),@XR	LOAD BRANCH TABLE BASE ADDR	
3324	6C	00	2E 00		3664	FZR050 MVC FZR070+@D1(,@BR),*-(1,@XR)	MOVE TABLE ENTRY TO BR INST	
3328	C2	02	0000		3665	FZR060 LA *-*,@XR	RESTORE DATA FILE OP CODE CADDR	
					3666	*		
					3667	* BRANCH TO EXECUTION ROUTINE SPECIFIED BY THE DATA FILE OP CODE		
					3668	*		
332C	D0	87	00		3669	FZR070 B *-*(,@BR)	GO EXECUTE CURR DATA FILE PMC	
					3670	*		
					3671	*****	*****	

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 260
					3673	*****	*****			
					3674	*	FZRDCA - DEFINE VIRTUAL ADDRESS OR CURRENT DATA FILE ELEMENT			*
					3675	*****	*****			
					3676	*				
				332F	3677	FZRDCA EQU	*		BEGIN FZRDCA EXECUTION	
					3678	*				
					3679	*	STACK THE DATA ELEMENT SPECIFIED BY THE 'DCA' VIRTUAL ADDRESS OPERAND			
					3680	*				
	332F	2C	01	144A	02	3681	FZR080 MVC I\$VADR,I@XVAD(B@LCVA,@XR)		SET PAGING PARAM FOR DATA VADDR	
	3334	3C	12	0BA2		3682	MVI I\$SLNG,I@LCRV-1		SET STACKING ROUTINE TO STACK	
					3683	*			* MAXIMUM LENGTH DATA ELEMENT	
	3338	35	02	0D4E		3684	L I\$STAK,@XR		LOAD THE STACK POINTER	
	333C	C0	87	0B50		3685	B I\$STCK		LINK TO STACK THE DATA ELEMENT	
					3686	*				
					3687	*	ADVANCE DATA FILE POINTER TO REFERENCE NEXT DATA FILE PMC			
					3688	*				
	3340	1E	00	0D53	6C	3689	FZR090 ALC I\$DATA,FZRLDA(@VADDR-1,@BR)		INCREMENT DATA FILE POINTER	
	3345	C0	87	12D3		3690	B I\$RTRN		RETURN TO THE INTERPRETER	
					3691	*				
					3692	*****	*****			

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 261
		3694		*****	
		3695	*	FZRDDL - DEFINE LINKAGE ADDRESS FOR NEXT DATA FILE PSEUDO INSTR.	*
		3696		*****	
		3697	*		
		3349	3698	FZRDDL EQU * BEGIN FZRDDL EXECUTION	
			3699	*	
			3700	* TEST FOR END OF THE PROGRAM DATA FILE	
			3701	*	
3349	BD 56 01		3702	FZR100 CLI I@XVAD-1(,@XR),@VENTA IF 'DDL' OPERAND IS VALID VADDR	
334C	F2 02 08		3703	JNL FZR120 * GO PERFORM LINKAGE OPERATION	
			3704	*	
			3705	* END OF DATA FILE - SET 'INSUFFICIENT DATA FOR READ' ERROR MESSAGE	
			3706	*	
334F	3C BF 0CBC		3707	FZR110 MVI I\$ERRC,@E721 SET INTERPRETER ERROR CODE	
3353	C0 87 12D3		3708	B I\$RTRN RETURN TO TERMINATE EXECUTION	
			3709	*	
			3710	* DATA FILE CONTINUED - LINK TO NEXT DATA FILE PMC SEQUENCE	
			3711	*	
3357	2C 01 0D53 02		3712	FZR120 MVC I\$DATA,I@XVAD(B@LCVA,@XR) SET DATA FILE PT - LINKAGE ADDR	
335C	D0 87 0F		3713	B FZR020(,@BR) GO PROCESS NEXT DATA FILE PMC	
			3714	*	
		3715		*****	

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 262
					3717	*****	*****			
					3718	*	FZREOP - CONTINLE DATA FILE PMC ON NEXT VIRTUAL PAGE			*
					3719	*****	*****			
					3720	*				
				335F	3721	FZREOP EQU	*			BEGIN FZREOP EXECUTION
					3722	*				
					3723	*	ADVANCE DATA FILE POINTER TO REFERENCE 1ST PSUEDO INSTRUCTION ON			
					3724	*	NEXT SEQUENTIAL VIRTUAL PAGE.			
					3725	*				
335F	1E	00	0D52	6B	3726	FZR130	ALC I\$DATA-1,FZRBNI(1,@BR)			INCREMENT POINTER PAGE NUMBER
3364	3C	00	0D53		3727		MVI I\$DATA,@ZERO			SET POINTER PAGE DISP TO ZERO
3368	D0	87	0F		3728		B FZR020(,@BR)			GO PROCESS NEXT DATA FILE PMC
					3729	*				
					3730	*****	*****			

FZREAD - S/3 BASIC INTERPRETER STATEMENT EXEC RTN

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 263
		3732		*****	
		3733		* READ STATEMENT EXECUTION ROUTINE CONSTANTS *	
		3734		*****	
		3735		*	
336B 01		336B 3736	FZRBN1 DC	IL1'1' BINARY INTEGER +1	
336C 03		336C 3737	FZRLDA DC	AL1(B@LDCA) LENGTH OF 'DCA' PSEUDO INST	
		3738		*	
		3739		*****	
		3740		* READ STMT RTN PSEUDO OPCODE EXECUTION BRANCH ADDRESS TABLE	
		3741		*****	
		3742		*	
		336D 3743	FZRBAT EQU *	BRACH TABLE STARTING ADDRESS	
		3744		*	
336D 005F		336E 3745	DC	AL(@CADDR)(FZREOP-FZREAD) EOP (X'68') END OF PMC PAGE	
336F 002F		3370 3746	DC	AL(@CADDR)(FZRDCA-FZREAD) DCA (X'6A') DEFINE CON VADDR	
3371 0049		3372 3747	DC	AL(@CADDR)(FZRDDL-FZREAD) DDL (X'6C') DEFINE DATA LINK	
		3748		*	
		3749		*****	
		3750		*	
		3751		*** END OF READ STATEMENT EXECUTION ROUTINE CODING ***	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 264
		3753		*****	
		3754	*	5703-XM1 COPYRIGHT IBM CORP. 1970	*
		3755	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083	*
		3756	*		*
		3757		*****	
		3758	*	*STATUS	*
		3759	*	VERSION 1 MODIFICATION 0	*
		3760	*		*
		3761	*	*FUNCTION -	*
		3762	*	* FZSPRT EXECUTION CAUSES DATA OUTPUT AND/OR CARRIER/CURSOR	*
		3763	*	POSITIONING ON THE SYSTEM PRINT DEVICE UNDER CONTROL OF CODES	*
		3764	*	DEVELOPED FROM THE FORMAT SPECIFIED IN A BASIC PROGRAM 'PRINT'	*
		3765	*	STATEMENT.	*
		3766	*	* THE FOLLOWING ACTIONS ARE PERFORMED, DEPENDING ON THE CODE	*
		3767	*	STORED IN INTERPRETER PARAMETER I\$PARM -	*
		3768	*	* CODE X'01' - PRINT AND NO SPACE.	*
		3769	*	THE DATA ELEMENT AT THE TOP OF THE RUN?TIME STACK IS CON-	*
		3770	*	VERTED TO OUTPUT FORMAT AND PRINTED. IF THE ELEMENT IS	*
		3771	*	ARITHMETIC, THE CARRIER/CURSOR IS RETURNED TO THE START OF	*
		3772	*	THE NEXT LINE (BEFORE PRINTING) WHEN THE CURRENT LINE CAN-	*
		3773	*	NOT CONTAIN THE FORMATTED VALUE. THE CARRIER/CURSOR IS	*
		3774	*	LEFT POSITIONED AT THE END OF THE PRINTED VALUE.	*
		3775	*	* CODE X'02' - PRINT AND SPACE FULL ZONE.	*
		3776	*	THE DATA ELEMENT AT THE TOP OF THE RUN-TIME STACK IS CON-	*
		3777	*	VERTED TO OUTPUT FORMAT AND PRINTED. IF THE ELEMENT IS	*
		3778	*	ARITHMETIC, THE CARRIER/CURSOR IS RETURNED TO THE START OF	*
		3779	*	THE NEXT LINE (BEFORE PRINTING) WHEN THE CURRENT LINE CAN-	*
		3780	*	NOT CONTAIN THE FORMATTED VALUE. IF THE ELEMENT IS A	*
		3781	*	CHARACTER REFERENCE, THE CARRIER/CURSOR IS RETURNED TO THE	*
		3782	*	START OF THE NEXT LINE (BEFORE PRINTING) WHEN THE CURRENT	*
		3783	*	LINE DOES NOT CONTAIN A FULL PRINT ZONE (18 SPACES). AT	*
		3784	*	THE END OF PRINTING, THE CARRIER/CURSOR IS SPACED TO THE	*
		3785	*	END OF THE FULL PRINT ZONE.	*
		3786	*	* CODE X'03' - PRINT AND SPACE PACKED ZONE.	*
		3787	*	THE DATA ELEMENT AT THE TOP OF THE RUN-TIME STACK IS CON-	*
		3788	*	VERTED TO OUTPUT FORMAT AND PRINTED. IF THE ELEMENT IS	*
		3789	*	ARITHMETIC, THE CARRIER/CURSOR IS RETURNED TO THE START OF	*
		3790	*	THE NEXT LINE (BEFORE PRINTING) WHEN THE CURRENT LINE CAN	*
		3791	*	NOT CONTAIN THE FORMATTED VALUE. AFTER AN ARITHMETIC ELE-	*
		3792	*	MENT IS PRINTED, THE CARRIER/CURSOR IS SPACED TO THE END	*
		3793	*	OF THE PACKED PRINT ZONE DEFINED IN FUNCTIONAL SPECIF1-	*
		3794	*	CATIONS. AFTER A CHARACTER ELEMENT IS PRINTED, THE	*
		3795	*	CARRIER/CURSOR IS LEFT POSITIONED AT THE END OF THE	*
		3796	*	PRINTED ELEMENT.	*
		3797	*	* CODE X'04' - PRINT AND RETURN CARRIER/CURSOR.	*
		3798	*	THE DATA ELEMENT AT THE TOP OF THE RUN-TIME STACK IS CON-	*
		3799	*	VERTED TO OUTPUT FORMAT AND PRINTED. IF THE ELEMENT IS	*
		3800	*	ARITHMETIC, THE CARRIER/CURSOR IS RETURNED TO THE START OF	*
		3801	*	THE NEXT LINE (BEFORE PRINTING) WHEN THE CURRENT LINE CAN-	*
		3802	*	NOT CONTAIN THE FORMATTED VALUE. AFTER THE ELEMENT IS	*
		3803	*	PRINTED, THE CARRIER/CURSOR IS RETURNED TO THE START OF	*
		3804	*	THE NEXT LINE.	*
		3805	*	* CODE X'05' - SPACE FULL ZONE.	*
		3806	*	THE CARRIER/CURSOR IS SPACED 18 CHARACTERS. IF NO MORE	*
		3807	*	THAN 18 CHARACTERS REMAIN IN THE CURRENT LINE, THE	*
		3808	*	CARRIER/CURSOR IS RETURNED TO THE START OF THE NEXT LINE.	*


```

3809 *      * CODE X'06' - SPACE PACKED ZONE. *
3810 *      THE CARRIER/CURSOR IS SPACED 3 CHARACTERS, IF NO MORE *
3811 *      THAN 3 CHARACTERS REMAIN IN THE CURRENT LINE, THE *
3812 *      CARRIER/CURSOR IS RETURNED TO THE START OF THE NEXT LINE. *
3813 *      * CODE X'07' - RETURN CARRIER/CURSOR, *
3814 *      THE CARRIER/CURSOR IS RETURNED. TO THE START OF THE NEXT *
3815 *      LINE. *
3816 *      * CODE X'08' - RETURN CARRIER/CURSOR ON CONDITION. *
3817 *      WHEN THE CURRENT LINE DOES NOT CONTAIN MORE THAN 18 CHAR- *
3818 *      ACTERS, THE CARRIER/CURSOR IS RETURNED TO THE START OF THE *
3819 *      NEXT LINE. *
3820 *      * WHEN REQUIRED, ELEMENT CONVERSION AND OUTPUT ARE PERFORMED IN *
3821 *      THE RUN-TIME STACK, SO TWAT THE STACKED ELEMENT IS NOT RECOVER- *
3822 *      ABLE. AFTER PRINTING, ARITHMETIC ELEMENT OUTPUT FORMAT DEPENDS *
3823 *      ON THE MAGNITUDE AND FRACTIONAL CHARACTERISTICS OF THE VALUE. *
3824 *      CHARACTER REFERENCE FORMATTING INVOLVES TRUNCATION OF TRAILING *
3825 *      BLANKS. CHARACTER CONSTANTS (LITERALS) ARE PRINTED AS SPECI- *
3826 *      FIED IN THE 'PRINT' STATEMENT. *
3827 *      * EITHER THE MATRIX PRINTER OR THE CRT (OR BOTH) MAY BE USED FOR *
3828 *      OUTPUT, DEPENDING ON THE CURRENT DEFINITION OF THE SYSTEM PRINT *
3829 *      DEVICE. CRT OUTPUT IS BASED ON A FIXED DISPLAY WIDTH OF 64 *
3830 *      CHARACTERS, WHILE PRINTER LINE WIDTH IS BASED ON THAT ASSIGNED *
3831 *      THROUGH THE 'WIDTH' SYSTEM COMMAND. *
3832 * *
3833 *ENTRY POINTS *
3834 *      THIS ROUTINE HAS A SINGLE ENTRY POINT - FZSPRT - WHOSE FUNCTION *
3835 *      IS DEFINED ABOVE. CALLING SEQUENCE IS - *
3836 *      B      I$CALL *
3837 *      DC      AL2(V$XSPR) *
3838 *      WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL ADDRESS *
3839 *      OF ENTRY POINT FZSPRT. EXECUTION IS SUBJECT TO INPUT CONDITIONS *
3840 *      DESCRIBED BELOW. *
3841 * *
3842 *INPUT *
3843 *      * #ISPARM - 2 BYTES, FOR THE PRINT CONTROL PARAMETER. THIS CON- *
3844 *      TAINS A CONTROL CODE, AS INDICATED UNDER 'FUNCTION', IN THE *
3845 *      RIGHTMOST BYTE. *
3846 *      * I$STAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER. FOR THOSE *
3847 *      CONTROL CODES SPECIFYING A DATA ELEMENT (SEE 'FUNCTION') THIS *
3848 *      CONTAINS, THE CORE ADDR OF THE FIRST AVAILABLE STACK LOCATION. *
3849 *      * RUN-TIME STACK - THIS CONTAINS AN UNPACKED FLOATING POINT VALUE *
3850 *      OR CHARACTER ELEMENT IN THE TOP STACK POSITION FOR CONTROL *
3851 *      CODES SPECIFYING DATA OUTPUT (SEE 'FUNCTION'). *
3852 *      * I$SLLC - 1 BYTE, FOR THE LENGTH CODE DEFINING THE LAST STACKED *
3853 *      DATA ELEMENT. WHEN DATA OUTPUT IS SPECIFIED, THIS IS USED TO *
3854 *      DETERMINE THE TYPE OF DATA ITEM (ARITHMETIC OR CHARACTER) CON- *
3855 *      TAINED IN THE TOP STACK POSITION. *
3856 *      * $PRPOS - 1 BYTE, FOR THE MATRIX PRINTER CARRIER POSITION *
3857 *      INDICATORS. THIS CONTAINS THE CARRIER POSITION, RELATIVE TO *
3858 *      THE HARDWARE LEFT MARGIN AS 0, OF THE MATRIX PRINTER CARRIER. *
3859 *      * $RMRGN - 1 BYTE, FOR THE MATRIX PRINTER SOFTWARE RIGHT MARGIN *
3860 *      INDICATOR. *
3861 *      * $CRPOS - 1 BYTE, FOR THE CRT CURSOR POSITION INDICATOR. THIS *
3862 *      CONTAINS THE CURSOR POSITION, RELATIVE TO THE LEFT CRT MARGIN *
3863 *      AS 0, OF THE CRT CURSOR. *
3864 *      * $PRDEV - 2 BYTES, FOR THE SYSTEM PRINT DEVICE INDICATOR. *

```

```

3865 * * $EXFTR - 1 BYTE, FOR THE SYSTEM CORE EXTENSION FACTOR. *
3866 * * *
3867 *OUTPUT *
3868 * * PRINTED OUTPUT AND/OR CARRIER/CURSOR CONTROL - AS SPECIFIED BY *
3869 * * THE CODE IN I$PARM, THE TYPE OF DATA ELEMENT IN THE STACK, AND *
3870 * * THE CURRENTLY DEFINED SYSTEM PRINT DEVICE(S). *
3871 * * I$PARM - 2 BYTES, FOR THE PRINT CONTROL PARAMETER, THIS INPUT *
3872 * * CONTROL CODE IS DESTROYED DURING EXECUTION. *
3873 * * RUN-TIME STACK - WHEN A DATA ELEMENT HAS BEEN PRINTED, THE *
3874 * * STACKED ELEMENT HAS BEEN CONVERTED IN PLACE TO OUTPUT FORMAT. *
3875 * * $PRPOS - 1 BYTE, FOR THE MATRIX PRINTER CARRIER POSITION *
3876 * * INDICATOR. THIS HAS BEEN MODIFIED TO INDICATE THE CURRENT *
3877 * * CARRIER POSITION AFTER PRINTED OUTPUT WHEN THE MATRIX PRINTER *
3878 * * IS A SYSTEM PRINT DEVICE. *
3879 * * $CRPOS - 1 BYTE, FOR THE CRT CURSOR POSITION INDICATOR. THIS *
3880 * * HAS BEEN MODIFIED TO INDICATE CURRENT CURSOR POSITION AFTER *
3881 * * DISPLAYED OUTPUT WHEN THE CRT IS A SYSTEM PRINT DEVICE. *
3882 * * *
3883 *EXTERNAL REFERENCES *
3884 * * VSSPRT - VIRTUAL ENTRY ADDRESS FOR DFPRNT, V.M. MATRIX PRT IOCS. *
3885 * * DSPLYN - ENTRY POINT FOR THE SYSTEM CRT IOCS (LABEL DSPLYN IS *
3886 * * REFERENCED INDIRECTLY USING I$CSXA TO BUILD A CODE ADDRESS). *
3887 * * I$CALL - ENTRY POINT FOR PAGING MODULE V.M. PROGRAM CALL RTN. *
3888 * * I$RTRN - ENTRY POINT FOR PAGING MODULE V.M. RETURN CONTROL RTN. *
3889 * * I$CSXA - CORE ADDRESS OF 1ST BYTE IN CORE EXTENSION PAST 8K. *
3890 * * I$PARM - 2 BYTES, FOR THE INTERPRETER COMMUNICATIONS PARAMETER. *
3891 * * I$STAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER. *
3892 * * I$SLLC - 1 BYTE, FOR LENGTH CODE (L-1) OF LAST STACKED ELEMENT. *
3893 * * I$WRK1 - 2 BYTES, FOR INTERPRETER COMMON WORK AREA 1. *
3894 * * I$WRK2 - 2 BYTES, FOR INTERPRETER COMMON WORK AREA 2. *
3895 * * $PRPOS - 1 BYTE, FOR MATRIX PRINTER CARRIER POSITION INDICATOR. *
3896 * * $RMRGN - 1 BYTE, FOR POSITION OF SOFTWARE RIGHT PRINTER MARGIN. *
3897 * * $CRPOS - 1 BYTE, FOR CRT CURSOR POSITION INDICATOR. *
3898 * * $PRDEV - 2 BYTES, FOR THE SYSTEM PRINT DEVICE INDICATOR. *
3899 * * $EXFTR - 1 BYTE, FOR THE SYSTEM CORE EXTENSION FACTOR. *
3900 * * *
3901 *EXITS, NORMAL *
3902 * * CONTROL IS ALWAYS PASSED TO THE PAGING ROUTINE AT ENTRY POINT *
3903 * * I$RTRN (IPGRTN) FOR A RETURN TO THE CALLING PROGRAM. *
3904 * * *
3905 *EXITS, ERROR *
3906 * * N/A *
3907 * * *
3908 *TABLES/WORKAREAS *
3909 * * FZSPRT BRANCH DISPLACEMENT TABLE - USED TO DIRECT OUTPUT OPERA- *
3910 * * TIONS FOR SPECIFIC ELEMENT TYPE - CONTROL CODE COMBINATIONS. *
3911 * * * NUMBER OF TABLE ENTRIES - 16 *
3912 * * * TABLE ENTRY LENGTH - 1 BYTE *
3913 * * * ENTRY FORMAT - SINGLE BYTE DISPLACEMENT WITHIN AN FZSPRT *
3914 * * * VIRTUAL PAGE FOR THE INTERNAL ENTRY POINT ASSOCIATED WITH *
3915 * * * EACH ELEMENT-CONTROL COMBINATION. *
3916 * * * RUN-TIME STACK - THE FIRST 20 AVAILABLE STACK LOCATIONS *
3917 * * * (INCLUDING LOCATIONS CONTAINING AN ELEMENT TO BE CONVERTED) ARE *
3918 * * * USED AS THE 'PRINT' OUTPUT BUFFER. *
3919 * * *
3920 *ATTRIBLIES *

```

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 267
		3921	*	* REUSABLE			*
		3922	*	* NATURALLY RELOCATABLE			*
		3923	*				*
		3924	*	*CHARACTER CODE DEPENDENCY			*
		3925	*	OPERATION OR THIS MODULE DEPENDS UPON THE FOLLOWING PROPER-			*
		3926	*	TIES QF THE INTERNAL REPRESENTATION OF THE EXTERNAL CHARACTER SET.			*
		3927	*	* MOST CODING HAS BEEN ARRANGED SO THAT REDEFINITION OF CHAR-			*
		3928	*	ACTER CONSTANTS, BY REASSEMBLY, WILL RESULT IN A CORRECT			*
		3929	*	MODULE FOR THE NEW DEFINITION.			*
		3930	*	* NUMERIC CHARACTERS 0 THROUGH 9 ARE PRESUMED TO BE CODED SUCH			*
		3931	*	THAT THE HIGH ORDER FOUR BITS CONTAIN A SIGN ZONE WITH X'F'			*
		3932	*	DEFINING A POSITIVE DIGIT.			*
		3933	*	THE SPECIFIC INSTRUCTIONS (INSTRUCTION SEQUENCES) WHICH REQUIRE			*
		3934	*	MODIFICATION IF THESE PROPERTIES OF THE CHARACTER SET ARE CHANGED			*
		3935	*	MAY OF IDENTIFIED BY -			*
		3936	*	* THE 4 INSTRUCTIONS BEGINNING AT LABEL FZS035.			*
		3937	*	* THE SINGLE INSTRUCTION IDENTIFIED BY LABEL FZS410.			*
		3938	*	* THE SINGLE INSTRUCTION IDENTIFIED BY LABEL FZS435.			*
		3939	*				*
		3940	*	*NOTES			*
		3941	*	ERROR PROCEDURES			*
		3942	*	FZSPRT UTILIZES OUTPUT IOCS ROUTINES DFPRNT (MATRIX PRINTER)			*
		3943	*	AND DSPLYN (CRT), AND IS SUBJECT TO THE ERP'S INHERENT IN			*
		3944	*	THESE PROGRAMS. FZSPRT OTHERWISE CONTAINS NO ERROR CONDITION			*
		3945	*	TESTS.			*
		3946	*				*
		3947	*	REGISTER USAGE			*
		3948	*	* REGISTER @BR IS TO CONTAIN THE CORE PAGE BASE ADDRESS			*
		3949	*	ESTABLISHED THROUGH PAGING MODULE CONTROL FOR THE PAGE WHICH			*
		3950	*	INCLUDES FZSPRT, AND IS RESTORED THROUGH THE PAGING MODULE.			*
		3951	*	* REGISTER @XR IS NOT SAVED, IT IS USED IN FZSPRT FOR GENERAL			*
		3952	*	PURPOSE INDEXING OPERATIONS.			*
		3953	*				*
		3954	*	SAVED/RESTORED AREAS			*
		3955	*	NONE			*
		3956	*				*
		3957	*	MODIFICATION CONSIDERATIONS			*
		3958	*	NONE			*
		3959	*				*
		3960	*	REQUIRED MODULES			*
		3961	*	* @SYSEQ - COMMON SYSTEM EQUATES.			*
		3962	*	* @FXDEQ - SYSTEM NUCLEUS ADDRESSES AND INDICATOR EQUATES.			*
		3963	*	* \$V\$EQU - VIRTUAL MEMORY FIXED ADDRESS EQUATES.			*
		3964	*	* \$B@EQU - COMPILER PARAMETER AND CONSTANT EQUATES.			*
		3965	*	* \$I@EQU - INTERPRETER FIXED LOCATION ADDRESS EQUATES.			*
		3966	*	* \$I@SEQ - INTERPRETER PARAMETER EQUATES (FOR STD. PREC. ONLY).			*
		3967	*	* \$I@LEQ - INTERPRETER PARAMETER EQUATES (FOR LONG PREC. ONLY).			*
		3968	*				*
		3969	*	OTHER			*
		3970	*	NONE			*
		3971	*	*****			*

FZSPRT - S/3 BASIC INTERPRETER PRINT STATEMENT EXEC RTN

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 268
		3973		*****	
		3974	*	START OF PRINT STATEMENT EXECUTION MODULE	*
		3975		*****	
		3976	*		
		3977	*	ESTABLISH ADDRESSABILITY FOR PRINT ROUTINE 1ST VM PAGE	
		3978	*		
3400		3979	*FZSP1B	VPAGE 0	SET 1ST PAGE ADDRESSABILITY
		3980		ORG *,256,0	SET STARTING ADDRESS
	3400	3981	FZSP1B	EQU *	START OF PROGRAM CODING
3301		3982		ORG *-255	RESET IAR TO PAGE
3400		3983		ORG *,256,0	* BOUNDARY ADDRESS
	3400	3984		USING *,@BR	SET PAGE BASE ADDRESS
3400		3985		ORG FZSP1B	RESET STARTING ADDRESS
		3986	***	END OF EXPANSION ***	
		3987	*		
		3988	*	ENTER FZSPRT - ACCESS THE STACKED DATA ELEMENT	
		3989	*		
3400 35 02 0D4E		3990	FZSPRT	EQU *	FZSPRT ENTRY POINT
		3991		L I\$STAK,@XR	LOAD THE STACK POINTER
		3992	*		
		3993	*	INITIALIZE AND TEST FOR CARRIER CONTROL (ONLY) PARAMETER	
		3994	*		
3404 7C 00 C7		3995	FZS010	MVI FZSCNT(,@BR),@ZERO	CLEAR DATA CHARACTER COUNTER
		3996	*		
3407 3D 05 0D57		3997		CLI I\$PARM,B@PRSL	IF CARRIER CONTROL ONLY,
340B D0 02 A4		3998		BNL FZS180(,@BR)	* GO PERFORM THE OPERATION
		3999	*		
		4000	*	TEST FOR CHARACTER ELEMENT PROCESSING	
		4001	*		
340E 3D 12 0BA1		4002	FZS020	CLI I\$SLLC,I@LCRV-1	IF STACK CONTAINS CHAR ELEMENT
3412 D0 81 73		4003		BE FZS130(,@BR)	* GO ESTABLISH CHARACTER OUTPUT
		4004	*		
		4005		*****	

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 269
				4007		*****		
				4008		* ARITHMETIC ELEMENT CONVERSION TO OUTPUT FORMAT		*
				4009		*****		
				4010		*		
				4011		* PROCESS THE SIGN OF THE STACKED ARITHMETIC VALUE		
				4012		*		
3415	7C	40	6E	4013	FZS030	MVI	FZS120+@Q(,@BR),B@BLNK	SET SIGN CHARACTER TO BLANK
3418	B8	F0	07	4014	FZS035	TBN	I@SIGN(,@XR),B@ZPOS	IF STACKED VALUE IS POSITIVE
341B	F2	10	06	4015		JT	FZS040	* SKIP PAST MINUS PROCESSING
341E	7C	60	6E	4016		MVI	FZS120+@Q(,@BR),B@MINS	SET SIGN CHARACTER TO MINUS
3421	BA	F0	07	4017		SBN	I@SIGN(,@XR),B@ZPOS	MAKE STACKED VALUE POSITIVE
3424	7C	01	C7	4018	FZS040	MVI	FZSCNT(,@BR),@B1	SET CHARACTER COUNT FOR SIGN
				4019		*		
				4020		* TEST FOR A ZERO VALUE (CATEGORIZED AS AN INTEGER) - A ZERO VALUE IS		
				4021		* LEFT IN THE STACK IN THE FORM 'S0', WHERE 'S' IS THE SIGN POSITION		
				4022		*		
3427	BD	F0	01	4023	FZS050	CLI	I@MANL(,@XR),B@DEC0	IF MOST SIGNIFICANT DIGIT NOT
342A	F2	01	07	4024		JNE	FZS060	* ZERO, GO ESTABLISH FORMAT
342D	5E	00	C7 DF	4025		ALC	FZSCNT(,@BR),FZSBN1(1,@BR)	INCR CHAR COUNT FOR ZERO DIGIT
3431	F2	87	39	4026		J	FZS120	* AND GO SET FOR ARITH OUTPUT
				4027		*		
				4028		* VALUE NOT ZERO - TEST MAGNITUDE FOR OUTPUT IN E- OR F-FORMAT		
				4029		*		
3434	BD	81	00	4030	FZS060	CLI	I@DEXP(,@XR),B@NXZR+1	IF VALUE LESS THAN 1E+0, OR
3437	F2	82	28	4031		JL	FZS110	* GREATER THAN OR EQUAL TO
343A	BD	86	00	4032		CLI	I@DEXP(,@XR),B@NXZR+I@APRC	* 1E+6 (1E+11 FOR LONG PREC),
343D	F2	84	22	4033		JH	FZS110	* GO CONVERT TO E OR F FORMAT
				4034		*		
				4035		* POSSIBLE I-FORMAT - TEST FOR A FRACTIONAL COMPONENT		
				4036		*		
3440	6C	00	56 00	4037	FZS070	MVC	FZS090+@Q(,@BR),I@DEXP(1,@XR)	ESTABLISH THE NUMBER OF
3444	5F	00	56 E0	4038		SLC	FZS090+@Q(,@BR),FZSNXZ(1,@BR)	* INTEGER DIGIT POSITIONS
3448	7C	07	4D	4039		MVI	FZS080+@D1(,@BR),I@PREC	SET DISP FOR MANTISSA RH BYTE
				4040		*		
				4041		*		
344B	BD	F0	00	4042	FZS080	CLI	*-*(,@XR),B@DEC0	IF FRACTIONAL DIGIT, GO CONVERT
344E	F2	01	11	4043		JNE	FZS110	* THE VALUE FOR E- OR F-FORMAT
3451	5F	00	4D DF	4044		SLC	FZS080+@D1(,@BR),FZSBN1(1,@BR)	DECR THE MANTISSA POINTER
3455	7D	00	4D	4045	FZS090	CLI	FZS080+@D1(,@BR),*-*	IF MORE FRACTIONAL POSITIONS
3458	D0	84	4B	4046		BH	FZS080(,@BR)	* REMAIN, GO REPEAT LOOP
				4047		*		
				4048		* NO FRACTIONAL COMPONENT - VALUE IS LEFT IN THE STACK IN THE FORM		
				4049		* 'S123' (I-FORMAT) WHERE 'S' IS THE SIGN POSITION		
				4050		*		
345B	5E	00	C7 4D	4051	FZS100	ALC	FZSCNT(,@BR),FZS080+@D1(1,@BR)	INCR CHAR COUNT FOR DIGITS
345F	F2	87	0B	4052		J	FZS120	* AND GO SET FOR ARITH OUTPUT
				4053		*		
				4054		* VALUE CANNOT BE HANDLED USING I-FORMAT - ROUND AND CONVERT VALUE,		
				4055		* LEAVING IN STACK IN THE FORM 'S123.45' (F-FORMAT) OR 'S1.239E+9'		
				4056		* (E-FORMAT) WHERE 'S' IS THE SIGN POSITION.		
				4057		*		
3462	C0	87	12B1	4058	FZS110	B	I\$CALL	LINK TO ROUND AND CONVERT THE
3466	3500			3467 4059		DC	AL(@VADDR)(FZS300)	* VALUE TO E- OR F-FORMAT
				4060		*		
3468	4E	00	C7 0D56	4061		ALC	FZSCNT(,@BR),I\$PARM-1(1)	INCR CHAR COUNT FROM CONVERSION
				4062		*		

[illegible]

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 271
				4071		*****				
				4072	*	CHARACTER ELEMENT CONVERSION TO OUTPUT FORMAT				*
				4073		*****				
				4074	*					
				4075	*	DETERMINE THE TYPE OF CHARACTER ELEMENT IN THE STACK				
				4076	*					
3473	B8	20	00	4077	FZS130	TBN	I@STAT(,@XR),B@CTYP		IF ELEMENT IS A STRING SEGMENT	
3476	F2	10	1C	4078		JT	FZS160		* GO ESTABLISH SEGMENT PARAMS	
				4079	*					
				4080	*	ELEMENT IS FROM A CHARACTER REFERENCE - LEAVE ELEMENT IN STACK IN				
				4081	*	THE FORM 'REFERENCE' (NO TRAILING BLANKS)				
				4082	*					
3479	1E	00	0D57 E1	4083	FZS140	ALC	I\$PARM,FZSCAJ(1,@BR)		ADJUST OUTPUT CONTROL PARAMETER	
				4084	*				* FOR CHARACTER REFERENCE	
347E	7C	13	8A	4085		MVI	FZS155+@D1(,@BR),I@LCRF+1		SET DISP FOR BYTE AFTER ELEMENT	
3481	5F	00	8A DF	4086	FZS150	SLC	FZS155+@D1(,@BR),FZSBN1(1,@BR)		DECR THE ELEMENT POINTER	
3485	F2	81	06	4087		JE	FZS158		BRANCH IF ALL CHARS ARE BLANKS	
3488	BD	40	00	4088	FZS155	CLI	*-(,@XR),B@BLNK		TEST ELEMENT CHAR FOR BLANK	
348B	D0	81	81	4089		BE	FZS150(,@BR)		* AND REPEAT LOOP UNTIL RIGHT-	
				4090	*				* MOST NON-BLANK CHAR IS FOUND	
348E	5C	00	C7 8A	4091	FZS158	MVC	FZSCNT(,@BR),FZS155+@D1(1,@BR)		SET CHAR COUNT FOR NUMBER	
				4092	*				* OF SIGNIFICANT ELEMENT CHARS	
3492	F2	87	0C	4093		J	FZS170		GO SET FOR CHARACTER OUTPUT	
				4094	*					
				4095	*	ELEMENT IS A CHARACTER STRING SEGMENT - LEAVE ELEMENT IN STACK IN				
				4096	*	THE FORM 'SEGMENT' (TRAILING BLANKS ALLOWED)				
				4097	*					
3495	1E	00	0D57 E2	4098	FZS160	ALC	I\$PARM,FZSSAJ(1,@BR)		ADJUST OUTPUT CONTROL PARAMETER	
				4099	*				* FOR CHARACTER STRING SEGMENT	
349A	BB	E0	00	4100		SBF	I@STAT(,@XR),X'FF'-B@CCNT		SET CHAR COUNT EQUAL TO COUNT	
349D	6C	00	C7 00	4101		MVC	FZSCNT(,@BR),I@STAT(1,@XR)		* FIELD IN ELEMENT STATUS BYTE.	
				4102	*					
				4103	*	ADJUST OUTPUT AREA POINTER FOR THE CHARACTER ELEMENT				
				4104	*					
34A1	E2	02	01	4105	FZS170	LA	@B1(,@XR),@XR		INCR POINTER PAST STATUS BYTE	
				4106	*					
				4107		*****				

FZSPRT - S/3 BASIC INTERPRETER PRINT STATEMENT EXEC RTN

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 272
				4109		*****				
				4110		* OUTPUT OPERATION INTERFACE ROUTINE				*
				4111		*****				
				4112		*				
				4113		* PAD THE CONVERTED DATA FIELD WITH BLANKS TO A FULL PRINT ZONE				
				4114		*				
34A4	7C	11	B2	4115	FZS180	MVI	FZS190+@Q(,@BR),I@LFPZ-1	SET LENGTH OF FIELD TO BE		
34A7	5F	00	B2 C7	4116		SLC	FZS190+@Q(,@BR),FZSCNT(1,@BR)	* PADDED - BYPASS PADDING		
34AB	F2	82	07	4117		JL	FZS200	* OPERATION IF LENGTH - 0		
34AE	BC	40	12	4118		MVI	I@LFPZ(,@XR),B@BLNK	PROPAGATE BLANKS TO FILL		
34B1	AC	00	11 12	4119	FZS190	MVC	I@LFPZ-1(,@XR),I@LFPZ(@VQ,@XR)	* THE FIELD TO FULL ZONE		
				4120		*				
				4121		* CONVERT THE OUTPUT PARAMETER TO AN ENTRY POINT DISPLACEMENT				
				4122		*				
34B5	34	02	0D59	4123	FZS200	ST	I\$WRK1,@XR	SAVE THE PRINT FIELD POINTER		
34B9	D2	02	E4	4124		LA	FZSCAT-1(,@BR),@XR	LOAD CONTROL ADDRESS TABLE BASF		
34BC	4C	00	C5 0D57	4125		MVC	FZS210+@OPD2(,@BR),I\$PARM(1)	SET THE TABLE DISPLACEMENT		
34C1	2C	00	0D57 00	4126	FZS210	MVC	I\$PARM,*-(1,@XR)	MOVE ENTRY PT DISP TO PARAMETER		
				4127		*				
				4128		* ESTABLISH THE DATA FIELD CHARACTER COUNT PARAMETER				
				4129		*				
34C6	3C	00	0D56	4130	FZS230	MVI	I\$PARM-1,*-	MOVE DATA FIELD COUNT TO PARAM		
				4131		*				
				4132		* ESTABLISH POSSIBLE CORE ENTRY ADDRESS FOR THE CRT IOCR				
				4133		*				
34CA	1C	01	0D5B E4	4134		MVC	I\$WRK2,FZSPDA(@CADDR,@BR)	SET BASE CRT ENTRY CORE ADDRESS		
34CF	0E	00	0D5A 043B	4135		ALC	I\$WRK2-1,\$EXFTR(1)	ADJUST CADDR FOR CORE EYTENSION		
				4136		*				
				4137		* OUTPUT THE DATA FIELD AS SPECIFIED BY CONTROL PARAMETER				
				4138		*				
34D5	C0	87	12B1	4139	FZS240	B	I\$CALL	LINK TO OUTPUT THE DATA FIELD		
34D9	3600			34DA 4140		DC	AL(@VADDR)(FZS600)	OUTPUT RTN VIRTUAL ADDRESS		
				4141		*				
				4142		* RETURN CONTROL TO THE INTERPRETER CALLING ROUTINE				
				4143		*				
34DB	C0	87	12D3	4144	FZS260	B	I\$RTRN	RETURN TO INTERPRETER		
				4145		*				
				4146		*****				

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 273
				4148			*****			
				4149	*		PRINT EXECUTION ROUTINE CONSTANTS (1ST VM PAGE)			*
				4150			*****			
				4151	*					
34DF	01			34DF	4152	FZSBN1 DC	IL1 '1' BINARY INTEGER+1			
				4153	*					
34E0	80			34E0	4154	FZSNXZ DC	AL1(B@NXZR) ZERO NORMALIZED EXPONENT			
34E1	08			34E1	4155	FZSCAJ DC	AL1(B@PRRL) CTL PARAM ADJUST - CHAR REF			
34E2	0C			34E2	4156	FZSSAJ DC	AL1(B@PRPR+B@PRRL) CTL PARAM ADJUST - CHAR STRING			
				4157	*					
34E3	2004			34E4	4158	FZSPDA DC	AL(@CADDR)(I\$CSXA+@INST4) CRT IOCR CORE ENTRY ADDR BASE			
				4159	*					
				4160			*****			

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 274
				4162		*****				
				4163	*	OUTPUT CONTROL PARAMETER FUNCTION ADDRESS TABLE				*
				4164		*****				
				4165	*					
				4166	*	DISPLACEMENT ENTRIES IN THE FOLLOWING TABLE REFERENCE THE MATRIX				
				4167	*	PRINTER OUTPUT ROUTINE (3RD VM PAGE), BUT ARE USED ALSO IN CON-				
				4168	*	JUNCTION WITH THE CRT OUTPUT ROUTINE (4TH VM PAGE). THUS, 4TH PAGE				
				4169	*	DISPLACEMENTS MUST BE KEPT IDENTICAL WITH 3RD PAGE DISPLACEMENTS				
				4170	*	WHICH ARE REFERENCED IN THE TABLE (E.G, FOR CODE 9, FZS860-FZS810				
				4171	*	MUST BE KEPT IDENTICAL TO FZS660-FZS610).				
				4172	*					
				34E5	4173	FZSCAT EQU *	CONTROL ADDR TABLE ADDRESS			
				4174	*					
34E5	00			34E5	4175	DC	AL1(FZS610-FZS610)	CODE 1 - PRT ARITH, NO SPACE		
34E6	18			34E6	4176	DC	AL1(FZS620-FZS610)	CODE 2 - PRT ARITH, SPACE FULL		
34E7	1E			34E7	4177	DC	AL1(FZS630-FZS610)	CODE 3 - PRT ARITH, SPACE PACK		
34E8	4D			34E8	4178	DC	AL1(FZS650-FZS610)	CODE 4 - PRT ARITH, RTRN CARR		
				4179	*					
34E9	59			34E9	4180	DC	AL1(FZS660-FZS610)	CODE 5 - SPACE FULL		
34EA	5F			34EA	4181	DC	AL1(FZS670-FZS610)	CODE 6 - SPACE PACKED		
34EB	73			34EB	4182	DC	AL1(FZS680-FZS610)	CODE 7 - RETURN CARRIER		
34EC	79			34EC	4183	DC	AL1(FZS690-FZS610)	CODE 8 - RETURN CARR ON COND		
				4184	*					
34ED	00			34ED	4185	DC	AL1(FZS610-FZS610)	CODE 9 - PRI CHAR, NO SPACE		
34EE	82			34EE	4186	DC	AL1(FZS695-FZS610)	CODE 10 - PRT CHAR, SPACE FULL		
34EF	00			34EF	4187	DC	AL1(FZS610-FZS610)	CODE 11 - PRT CHAR, SPACE PACK		
34F0	4D			34F0	4188	DC	AL1(FZS650-FZS610)	CODE 12 - PRT CHAR, RTRN CARR		
				4189	*					
34F1	00			34F1	4190	DC	AL1(FZS610-FZS610)	CODE 13 - PRT STRING, NO SPACE		
34F2	88			34F2	4191	DC	AL1(FZS700-FZS610)	CODE 14 - PRT STRING, SPACE LNG		
34F3	00			34F3	4192	DC	AL1(FZS610-FZS610)	CODE 15 - PRT STRING, SPACE PKD		
34F4	4D			34F4	4193	DC	AL1(FZS650-FZS610)	CODE 16 - PRT STRING, RTRN CARR		
				4194	*					
				4195		*****				
				4196	*	PRINT EXECUTION ROUTINE EQUATES (1ST VM PAGE)				*
				4197		*****				
				4198	*					
				0000	4199	FZSPAL EQU 0	DISP FOR OUTPUT AREA LEFT BYTE			
				4200	*					
				34C7	4201	FZSCNT EQU FZS230+@Q	DATA CHARACTER COUNTER			
				4202	*					
				4203		*****				

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 275
					4205	*****	*****			
					4206	*	VIRTUAL MEMORY PRINT E-EXECUTION ROUTINE 2ND VM PAGE -			*
					4207	*	* ROUNDS THE ARITHMETIC VALUE IN THE RUN-TIME STACK			*
					4208	*	* CONVERTS ARITHMETIC VALUE TO E- OR F-FORMAT FOR OUTPUT			*
					4209	*				*
					4210	*	INPUT -			*
					4211	*	* RUN-TIME STACK - CONTAINS ARITHMETIC VALUE TO BE CONVERTED			*
					4212	*	* REGISTER @XR - CONTAINS CORE ADDRESS OF VALUE EXPONENT BYTE			*
					4213	*				*
					4214	*	OUTPUT -			*
					4215	*	* RUN-TIME STACK - CONTAINS CONVERTED ARITHMETIC VALUE			*
					4216	*	* REGISTER @XR - CONTAINS CORE ADDRESS OF VALUE SIGN POSITION			*
					4217	*	* I\$PARM-1 - 1 BYTE, CONTAINS VALUE CHAR COUNT (NOT INCL SIGN)			*
					4218	*****	*****			*
					4219	*				*
					4220	*	ESTABLISH ADDRESSABILITY FOR PRINT ROUTINE 2ND VM PAGE			
					4221	*				
					4222	*FZSP2B	VPAGE 0 SET 2ND PAGE ADDRESSABILITY			
3500					4223	ORG	*,256,0 SET STARTING ADDRESS			
				3500	4224	FZSP2B EQU	* START OF PROGRAM CODING			
3401					4225	ORG	*-255 RESET IAR TO PAGE			
3500					4226	ORG	*,256,0 * BOUNDARY ADDRESS			
				3500	4227	USING	*,@BR SET PAGE BASE ADDRESS			
3500					4228	ORG	FZSP2B RESET STARTING ADDRESS			
					4229	***	END OF EXPANSION ***			
					4230	*				
					4231	*	CONVERSION ENTRY - ROUND THE ARITHMETIC VALUE FOR E- OR F-FORMAT			
					4232	*				
				3500	4233	FZS300 EQU	* CONVERSION ROUTINE ENTRY POINT			
3500 96 60 07 CC					4234	AZ	I@APRC+1(I@APRC+1,@XR),FZSDC5(1,@BR) ROUND THE VALUE UP			
3504 F2 08 07					4235	JNOZ	FZS310 IF NO OVFLOW SKIP TO CONTINUE,			
3507 BC F1 01					4236	MVI	I@MANL(,@XR),B@DEC1 * ELSE SET MOST SIGNIFICANT			
350A 9E 00 00 CA					4237	ALC	I@DEXP(,@XR),FZS2B1(1,@BR) * DIGIT = 1 AND INCR EXPONENT			
					4238	*				
					4239	*	TEST MAGNITUDE OF VALUE FOR OUTPUT IN E- OR F-FORMAT			
					4240	*				
350E BD 80 00					4241	FZS310 CLI	I@DEXP(,@XR),B@NXZR IF VALUE LESS THAN 1E-1, OR			
3511 D0 82 4D					4242	BL	FZS400(,@BR) * GREATER THAN OR EQUAL TO			
3514 BD 86 00					4243	CLI	I@DEXP(,@XR),B@NXZR+I@APRC * 1E+6 (1E+11 FOR LONG PREC),			
3517 D0 84 4D					4244	BH	FZS400(,@BR) * GO CONVERT VALUE TO E-FORMAT			
					4246	*****	*****			
					4247	*	F-FORMAT OUTPUT CONVERSION ROUTINE			*
					4248	*****	*****			
					4249	*				
					4250	*	SHIFT FRACTIONAL-COMPONENT RIGHT TO INSERT DECIMAL POINT			
					4251	*				
351A 7C 85 25					4252	FZS320 MVI	FZS330+@Q(,@BR),B@NXZR+I@APRC-1 ESTABLISH LENGTH CODE FOR			
351D 6F 00 25 00					4253	SLC	FZS330+@Q(,@BR),I@DEXP(1,@XR) * FRACTIONAL COMPONENT			
3521 F2 82 04					4254	JL	FZS340 BRANCH IF NO FRACTION			
3524 AC 00 07 06					4255	FZS330 MVC	I@APRC+1(,@XR),I@APRC(@VQ,@XR) SHIFT FRACTION RIGHT BY 1			
					4256	*				
					4257	*	ESTABLISH F-FORMAT DECIMAL POINT - VALUE IS LEFT IN STACK IN FORM			
					4258	*	'S.123456', S123.456', OR 'S123456.' WHERE 'S' IS THE SIGN POSITION			
					4259	*				
3528 6C 00 36 00					4260	FZS340 MVC	FZS350+@D1(,@BR),I@DEXP(1,@XR) CALCULATE DISPLACEMENT			

FZSPRT - S/3 BASIC INTERPRETER PRINT STATEMENT EXEC RTN						
ERR LOC	OBJECT CODE	ADDR STMT	SOURCE STATEMENT	VER 15,	MOD 00	31/05/21 PAGE 276
352C	5E 00 36 CA	4261	ALC FZS350+@D1(,@BR),FZS2B1(1,@BR)	*	FOR THE DECIMAL POINT	
3530	5F 00 36 CD	4262	SLC FZS350+@D1(,@BR),FZS2XZ(1,@BR)	*	IN F-FORMAT FIELD	
3534	BC 4B 00	4263	FZS350 MVI *-*(,@XR),B@DPNT		INSERT THE DECIMAL POINT	
		4264	*			
		4265	* TRUNCATE INSIGNIFICANT ZEROS FROM THE ROUNDED VALUE			
		4266	*			
3537	7C 08 40	4267	FZS360 MVI FZS380+@D1(,@BR),I@APRC+2	SET DISP FOR BYTE AFTER VALUE		
353A	5F 00 40 CA	4268	FZS370 SLC FZS380+@D1(,@BR),FZS2B1(1,@BR)	DECR VALUE CHAR POINTER		
353E	BD F0 00	4269	FZS380 CLI *-*(,@XR),B@DEC0	TEST VALUE CHARACTER FOR ZERO		
3541	D0 81 3A	4270	BE FZS370(,@BR)	*	AND REPEAT UNTIL NON-ZERO	
		4271	*			
		4272	* SET COUNT PARAMETER AND RETURN TO CALLING PAGE			
		4273	*			
3544	1C 00 0D56 40	4274	FZS390 MVC I\$PARM-1,FZS380+@D1(1,@BR)	MOVE DATA CHARACTER COUNT		
		4275	*	*	TO THE OUTPUT PARAMETER	
3549	C0 87 12D3	4276	B I\$RTRN	RETURN TO CALLING PAGE		
		4277	*			
		4278	*****			

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 277
				4280		*****				
				4281	*	E-FORMAT OUTPUT CONVERSION ROUTINE				*
				4282		*****				
				4283	*					
				4284	*	SHIFT MANTISSA (EXCEPT MOST SIGNIFICANT DIGIT) RIGHT TO INSERT				
				4285	*	DECIMAL POINT - ESTABLISH E-FORMAT DECIMAL POINT, LEAVING VALUE				
				4286	*	IN STACK IN FORM 'S1.23496' WHERE 'S' IS THE SIGN POSITION				
				4287	*					
354D	AC	04	07	06	4288	FZS400	MVC I@APRC+1(,@XR),I@APRC(I@APRC-1,@XR) SHIFT MANTISSA RIGHT			
3551	BC	4B	02		4289		MVI FZSPAL+2(,@XR),B@DPNT INSERT E-FORMAT DECIMAL POINT			
3554	9F	00	00	CA	4290		SLC I@DEXP(,@XR),FZS2B1(1,@BR) ADJUST EXPONENT TO COMPENSATE			
				4291	*					
				4292	*	TRUNCATE INSIGNIFICANT ZEROS FROM ROUNDED VALUE - KEEP AT LEAST ONE				
				4293	*	DIGIT TO RIGHT OF DECIMAL POINT				
				4294	*					
3558	BB	F0	03		4295	FZS410	SBF FZSPAL+3(,@XR),B@ZPOS FLAG DIGIT AFTER DECIMAL POINT			
355B	7C	08	64		4296		MVI FZS430+@D1(,@BR),I@APRC+2 SET DISP FOR BYTE AFTER VALUE			
355E	5F	00	64	CA	4297	FZS420	SLC FZS430+@D1(,@BR),FZS2B1(1,@BR) DECR VALUE CHAR POINTER			
3562	BD	F0	00		4298	FZS430	CLI *-*(,@XR),B@DEC0 TEST VALUE CHARACTER FOR ZERO			
3565	D0	81	5E		4299		BE FZS420(,@BR) * AND REPEAT UNTIL NON-ZERO			
3568	BA	F0	03		4300	FZS435	SBN FZSPAL+3(,@XR),B@ZPOS RESTORE DIGIT AFTER DEC POINT			
				4301	*					
				4302	*	SET COUNT PARAMETER FOR FORMATTED MANTISSA PLUS 4 BYTE EXPONENT				
				4303	*					
356B	3C	04	0D56		4304	FZS440	MVI I\$PARM-1,FZSLXB SET DATA CHAR CNT FOR EXPONENT			
356F	1E	00	0D56	64	4305		ALC I\$PARM-1,FZS430+@D1(1,@BR) INCR DATA CHAR COUNT FOR VALUE			
				4306	*					
				4307	*	INITIALIZE OUTPUT FORM OF EXPONENT - TEST FOR EXPONENT SIGN				
				4308	*					
3574	5C	03	D6	D1	4309	FZS450	MVC FZSXWK(,@BR),FZSEXB(FZSLXB,@BR) MOVE EXPONENT IMAGE TO			
				4310	*		* EXPONENT WORK AREA			
3578	6C	00	D2	00	4311		MVC FZS2BX(,@BR),I@DEXP(1,@XR) DETERMINE BINARY MAGNITUDE			
357C	5F	00	D2	CD	4312		SLC FZS2BX(,@BR),FZS2XZ(1,@BR) * ASSUMING POSITIVE EXPONENT			
3580	F2	81	29		4313		JE FZS480 BRANCH IF EXPONENT IS ZERO			
3583	F2	84	0A		4314		JH FZS470 BRANCH IF EXPONENT IF POSITIVE			
				4315	*					
				4316	*	NEGATIVE EXPONENT - MODIFY SIGN AND RECOMPUTE BINARY EXPONENT				
				4317	*					
3586	7C	60	D4		4318	FZS460	MVI FZSXWK-FZSLXM(,@BR),B@MINS MAKE EXPONENT SIGN NEGATIVE			
3589	7C	80	D2		4319		MVI FZS2BX(,@BR),B@NXZR DETERMINE BINARY MAGNITUDE			
358C	6F	00	D2	00	4320		SLC FZS2BX(,@BR),I@DEXP(1,@XR) * FOR NEGATIVE EXPONENT			
				4321	*					
				4322	*	CONVERT BINARY EXPONENT MAGNITUDE TO ZONED DECIMAL				
				4323	*					
3590	54	10	D8	CB	4324	FZS470	ZAZ FZSDAC(FZSLXM,@BR),FZSDC1(1,@BR) SET DEC ACCUMULATOR = 1			
3594	7C	01	98		4325		MVI FZS472+@Q(,@BR),@B1 SET BINARY MASK FOR 2**0 BIT			
3597	78	00	D2		4326	FZS472	TBN FZS2BX(,@BR),*- TEST BINARY EXP MAGNITUDE BIT			
359A	F2	90	04		4327		JF FZS474 * AND BRANCH IF BIT IS ZERO			
359D	56	01	D6	D8	4328		AZ FZSXWK(FZSLXM,@BR),FZSDAC(FZSLXM,@BR) INCR DECIMAL EXP			
35A1	5E	00	98	98	4329	FZS474	ALC FZS472+@Q(,@BR),FZS472+@Q(1,@BR) SHIFT BINARY MASK LEFT			
35A5	56	01	D8	D8	4330		AZ FZSDAC(FZSLXM,@BR),FZSDAC(FZSLXM,@BR) DOUBLE DEC ACCUM			
35A9	D0	08	97		4331		BNOZ FZS472(,@BR) REPEAT LOOP UNTIL ACCUM > 644			
				4332	*					
				4333	*	TEST FOR AND DELETE ANY INSIGNIFICANT ZERO IN THE DECIMAL EXPONENT				
				4334	*					
35AC	7D	F0	D5		4335	FZS480	CLI FZSXWK-1(,@BR),B@DEC0 TEST FOR EXPONENT LEFTMOST ZERO			

FZSPRT - S/3 BASIC INTERPRETER PRINT STATEMENT EXEC RTN									
ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15,	MOD 00 31/05/21 PAGE 278
	35AF	F2	01 09		4336	JNE	FZS490	BRANCH IF NO INSIGNIFICANT ZERO	
	35B2	5C	00 D5 D6		4337	MVC	FZSXWK-1(,@BR),FZSXWK(1,@BR)	SHIFT SIGNIFICANT DIGIT	
	35B6	1F	00 0D56 CA		4338	SLC	I\$PARM-1,FZS2B1(1,@BR)	DECREMENT DATA CHARACTER COUNT	
					4339	*			
					4340	*	MOVE OUTPUT FORM OF EXPONENT TO THE DATA PRINT FIELD		
					4341	*			
	35BB	7C	04 C4		4342	FZS490 MVI	FZS500+@D1(,@BR),FZSLXB	SET DIP TO ESTABLISH	
	35BE	5E	00 C4 64		4343	ALC	FZS500+@D1(,@BR),FZS430+@D1(1,@BR)	* EXPONENT POSITION	
	35C2	9C	03 00 D6		4344	FZS500 MVC	*-*(,@XR),FZSXWK(FZSLXB,@BR)	MOVE EXPONENT TO PRINT FIELD	
					4345	*			
					4346	*	RETURN CONTROL TO THE CALLING PAGE		
					4347	*			
	35C6	C0	87 12D3		4348	FZS510 B	I\$RTRN	RETURN TO CALLING PAGE	
					4349	*			
					4350	*****			

[illegible]

FZSPRT - S/3 BASIC INTERPRETER PRINT STATEMENT EXEC RTN									
ERR	LOC	OBJECT CODE		ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 278	
	35AF	F2	01 09		4336	JNE	FZS490	BRANCH IF NO INSIGNIFICANT ZERO	
	35B2	5C	00 D5 D6		4337	MVC	FZSXWK-1(,@BR),FZSXWK(1,@BR)	SHIFT SIGNIFICANT DIGIT	
	35B6	1F	00 0D56 CA		4338	SLC	I\$PARM-1,FZS2B1(1,@BR)	DECREMENT DATA CHARACTER COUNT	
					4339	*			
					4340	*	MOVE OUTPUT FORM OF EXPONENT TO THE DATA PRINT FIELD		
					4341	*			
	35BB	7C	04 C4		4342	FZS490 MVI	FZS500+@D1(,@BR),FZSLXB	SET DIP TO ESTABLISH	
	35BE	5E	00 C4 64		4343	ALC	FZS500+@D1(,@BR),FZS430+@D1(1,@BR)	* EXPONENT POSITION	
	35C2	9C	03 00 D6		4344	FZS500 MVC	*-*(,@XR),FZSXWK(FZSLXB,@BR)	MOVE EXPONENT TO PRINT FIELD	
					4345	*			
					4346	*	RETURN CONTROL TO THE CALLING PAGE		
					4347	*			
	35C6	C0	87 12D3		4348	FZS510 B	I\$RTRN	RETURN TO CALLING PAGE	
					4349	*			
					4350	*****			

FZSPRT - S/3 BASIC INTERPRETER PRINT STATEMENT EXEC RTN									
ERR	LOC	OBJECT CODE		ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 278	
	35AF	F2	01 09		4336	JNE	FZS490	BRANCH IF NO INSIGNIFICANT ZERO	
	35B2	5C	00 D5 D6		4337	MVC	FZSXWK-1(,@BR),FZSXWK(1,@BR)	SHIFT SIGNIFICANT DIGIT	
	35B6	1F	00 0D56 CA		4338	SLC	I\$PARM-1,FZS2B1(1,@BR)	DECREMENT DATA CHARACTER COUNT	
					4339	*			
					4340	*	MOVE OUTPUT FORM OF EXPONENT TO THE DATA PRINT FIELD		
					4341	*			
	35BB	7C	04 C4		4342	FZS490 MVI	FZS500+@D1(,@BR),FZSLXB	SET DIP TO ESTABLISH	
	35BE	5E	00 C4 64		4343	ALC	FZS500+@D1(,@BR),FZS430+@D1(1,@BR)	* EXPONENT POSITION	
	35C2	9C	03 00 D6		4344	FZS500 MVC	*-*(,@XR),FZSXWK(FZSLXB,@BR)	MOVE EXPONENT TO PRINT FIELD	
					4345	*			
					4346	*	RETURN CONTROL TO THE CALLING PAGE		
					4347	*			
	35C6	C0	87 12D3		4348	FZS510 B	I\$RTRN	RETURN TO CALLING PAGE	
					4349	*			
					4350	*****			

FZSPRT - S/3 BASIC INTERPRETER PRINT STATEMENT EXEC RTN						
ERR LOC	OBJECT CODE	ADDR STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 278		
35AF	F2 01 09	4336	JNE FZS490	BRANCH IF NO INSIGNIFICANT ZERO		
35B2	5C 00 D5 D6	4337	MVC FZSXWK-1(,@BR),FZSXWK(1,@BR)	SHIFT SIGNIFICANT DIGIT		
35B6	1F 00 0D56 CA	4338	SLC I\$PARM-1,FZS2B1(1,@BR)	DECREMENT DATA CHARACTER COUNT		
		4339	*			
		4340	* MOVE OUTPUT FORM OF EXPONENT TO THE DATA PRINT FIELD			
		4341	*			
35BB	7C 04 C4	4342	FZS490 MVI FZS500+@D1(,@BR),FZSLXB	SET DIP TO ESTABLISH		
35BE	5E 00 C4 64	4343	ALC FZS500+@D1(,@BR),FZS430+@D1(1,@BR)	* EXPONENT POSITION		
35C2	9C 03 00 D6	4344	FZS500 MVC *-*(,@XR),FZSXWK(FZSLXB,@BR)	MOVE EXPONENT TO PRINT FIELD		
		4345	*			
		4346	* RETURN CONTROL TO THE CALLING PAGE			
		4347	*			
35C6	C0 87 12D3	4348	FZS510 B I\$RTRN	RETURN TO CALLING PAGE		
		4349	*			
		4350	*****			

FZSPRT - S/3 BASIC INTERPRETER PRINT STATEMENT EXEC RTN						
ERR LOC	OBJECT CODE	ADDR STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 278		
35AF	F2 01 09	4336	JNE FZS490	BRANCH IF NO INSIGNIFICANT ZERO		
35B2	5C 00 D5 D6	4337	MVC FZSXWK-1(,@BR),FZSXWK(1,@BR)	SHIFT SIGNIFICANT DIGIT		
35B6	1F 00 0D56 CA	4338	SLC I\$PARM-1,FZS2B1(1,@BR)	DECREMENT DATA CHARACTER COUNT		
		4339	*			
		4340	* MOVE OUTPUT FORM OF EXPONENT TO THE DATA PRINT FIELD			
		4341	*			
35BB	7C 04 C4	4342	FZS490 MVI FZS500+@D1(,@BR),FZSLXB	SET DIP TO ESTABLISH		
35BE	5E 00 C4 64	4343	ALC FZS500+@D1(,@BR),FZS430+@D1(1,@BR)	* EXPONENT POSITION		
35C2	9C 03 00 D6	4344	FZS500 MVC *-*(,@XR),FZSXWK(FZSLXB,@BR)	MOVE EXPONENT TO PRINT FIELD		
		4345	*			
		4346	* RETURN CONTROL TO THE CALLING PAGE			
		4347	*			
35C6	C0 87 12D3	4348	FZS510 B I\$RTRN	RETURN TO CALLING PAGE		
		4349	*			
		4350	*****			

FZSPRT - S/3 BASIC INTERPRETER PRINT STATEMENT EXEC RTN									
ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15,	MOD 00 31/05/21 PAGE 278
	35AF	F2	01 09		4336	JNE	FZS490	BRANCH IF NO INSIGNIFICANT ZERO	
	35B2	5C	00 D5 D6		4337	MVC	FZSXWK-1(,@BR),FZSXWK(1,@BR)	SHIFT SIGNIFICANT DIGIT	
	35B6	1F	00 0D56 CA		4338	SLC	I\$PARM-1,FZS2B1(1,@BR)	DECREMENT DATA CHARACTER COUNT	
					4339	*			
					4340	*	MOVE OUTPUT FORM OF EXPONENT TO THE DATA PRINT FIELD		
					4341	*			
	35BB	7C	04 C4		4342	FZS490 MVI	FZS500+@D1(,@BR),FZSLXB	SET DIP TO ESTABLISH	
	35BE	5E	00 C4 64		4343	ALC	FZS500+@D1(,@BR),FZS430+@D1(1,@BR)	* EXPONENT POSITION	
	35C2	9C	03 00 D6		4344	FZS500 MVC	*-*(,@XR),FZSXWK(FZSLXB,@BR)	MOVE EXPONENT TO PRINT FIELD	
					4345	*			
					4346	*	RETURN CONTROL TO THE CALLING PAGE		
					4347	*			
	35C6	C0	87 12D3		4348	FZS510 B	I\$RTRN	RETURN TO CALLING PAGE	
					4349	*			
					4350	*****			

FZSPRT - S/3 BASIC INTERPRETER PRINT STATEMENT EXEC RTN									
ERR	LOC	OBJECT CODE		ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 278	
	35AF	F2	01 09		4336	JNE	FZS490	BRANCH IF NO INSIGNIFICANT ZERO	
	35B2	5C	00 D5 D6		4337	MVC	FZSXWK-1(,@BR),FZSXWK(1,@BR)	SHIFT SIGNIFICANT DIGIT	
	35B6	1F	00 0D56 CA		4338	SLC	I\$PARM-1,FZS2B1(1,@BR)	DECREMENT DATA CHARACTER COUNT	
					4339	*			
					4340	*	MOVE OUTPUT FORM OF EXPONENT TO THE DATA PRINT FIELD		
					4341	*			
	35BB	7C	04 C4		4342	FZS490 MVI	FZS500+@D1(,@BR),FZSLXB	SET DIP TO ESTABLISH	
	35BE	5E	00 C4 64		4343	ALC	FZS500+@D1(,@BR),FZS430+@D1(1,@BR)	* EXPONENT POSITION	
	35C2	9C	03 00 D6		4344	FZS500 MVC	*-*(,@XR),FZSXWK(FZSLXB,@BR)	MOVE EXPONENT TO PRINT FIELD	
					4345	*			
					4346	*	RETURN CONTROL TO THE CALLING PAGE		
					4347	*			
	35C6	C0	87 12D3		4348	FZS510 B	I\$RTRN	RETURN TO CALLING PAGE	
					4349	*			
					4350	*****			

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 279
			4352	*****				
			4353	* PRINT EXECUTION ROUTINE CONSTANTS (2ND VM PAGE)				*
			4354	*****				
			4355	*				
35CA	01	35CA	4356	FZS2B1 DC	IL1'1'	BINARY	INTEGER	+1
35CB	F1	35CB	4357	FZSDC1 DC	DL1'1'	DECIMAL	INTEGER	+1
35CC	F5	35CC	4358	FZSDC5 DC	DL1'5'	DECIMAL	INTEGER	+5
			4359	*				
35CD	80	35CD	4360	FZS2XZ DC	AL1(B@NXZR)	ZERO	NORMALIZED	EXPONENT
			4361	*				
		0004	4362	FZSLXB EQU	4	LENGTH OF EXPONENT IMAGE		
35CE	C54EF0F0	35D1	4363	FZSEXB DC	CL(FZSLXB)'E+00'	EXPONENT IMAGE FOR OUTPUT		
			4364	*				
			4365	*****				
			4366	* PRINT EXECUTION ROUTINE WORK AREAS (2ND VM PAGE)				*
			4367	*****				
			4368	*				
35D2		35D2	4369	FZS2BX DS	CL1	BINARY	EXPONENT	MAGNITUDE
35D3		35D6	4370	FZSXWK DS	CL(FZSLXB)	EXPONENT	CONSTRUCT	AREA
			4371	*				
		0002	4372	FZSLXM EQU	2	LENGTH OF DECMAL EXP MAGNITUDE		
35D7		35D8	4373	FZSDAC DS	CL(FZSLXM)	B TO D	DECIMAL	ACCUMULATOR
			4374	*				
			4375	*****				

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 280
		4377		*****	
		4378	*	VIRTUAL MEMORY PRINT EXECUTION ROUTINE (3RD VM PAGE)	*
		4379	*	* OUTPUTS FORMATTED DATA ELEMENT TO MATRIX PRINTER	*
		4380	*	* CONTROLS PRINTER CARRIER DEPENDING ON SPECIFIED CONTROL CODE	*
		4381	*	INPUT -	*
		4382	*	* RUN-TIME STACK - CONTAINS FORMATTED ELEMENT, IF PRESENT	*
		4383	*	* I\$PARM - 1 BYTE, CONTAINS CONTROL CODE BRANCH DISPLACEMENT	*
		4384	*	* I\$PARM-1 - 1 BYTE, CONTAINS FORMATTED ELEMENT CHARACTER COUNT	*
		4385	*	* I\$WRK1 - 2 BYTES, CONTAINS CORE ADDR OF PRINT AREA LEFT BYTE	*
		4386	*	* I\$WRK2 - 2 BYTES, CONTAINS VALUE FOR \$PRDEV 'CRT ONLY' COND	*
		4387	*	* I\$SLLC - 1 BYTE, CONTAINS OUTPUT ELEMENT LENGTH CODE (LNG - 1)	*
		4388	*		*
		4389	*	OUTPUT -	*
		4390	*	* PRINTED ELEMENT AND/OR CARRIER CONTROL ON MATRIX PRINTER	*
		4391		*****	
		4392	*		
		4393	*	ESTABLISH ADDRESSABILITY FOR PRINT ROUTINE (3RD VM PAGE)	
		4394	*		
3600		4395	*FZSP3B	VPAGE 0 SET 3RD PAGE ADDRESSABILITY	
		4396	ORG	*,256,0 SET STARTING ADDRESS	
	3600	4397	FZSP3B EQU	* START OF PROGRAM CODING	
3501		4398	ORG	*-255 RESET IAR TO PAGE	
3600		4399	ORG	*,256,0 * BOUNDARY ADDRESS	
	3600	4400	USING	*,@BR SET PAGE BASE ADDRESS	
3600		4401	ORG	FZSP3B RESET STARTING ADDRESS	
		4402	***	END OF EXPANSION ***	
		4403	*		
		4404	*	PAGE ENTRY - TEST FOR MATRIX PRINTER ACTIVE ON SYSTEM	
		4405	*		
3600 0D 01 044B 0D5B		4406	FZS600 CLC	\$PRDEV,I\$WRK2(@CADDR) IF PRINTER NOT A SYSTEM PRINT ?	
3606 F2 02 BF		4407	JNL	FZS740 * DEVICE, GO OUTPUT TO THE CRT	
		4408	*		
		4409	*	INITIALIZE FOR OUTPUT TO THE MATRIX PRINTER	
		4410	*		
3609 4C 00 6A 03C0		4411	MVC	FZS3RM(,@BR),\$RMRGN(1) SET MP RIGHT MARGIN PARAMETER	
		4412	*		
		4413	*	INITIALIZE THE ELEMENT PRINT PARAMETER LIST	
		4414	*		
360E 7C 40 F2		4415	MVI	FZS3PF(,@BR),@PRINT SET FUNCTION FOR PRINT ONLY	
3611 4C 00 F3 0D56		4416	MVC	FZS3PC(,@BR),I\$PARM-1(1) SET COUNT = ELEMENT CHAR COUNT	
3616 4C 01 F5 0D59		4417	MVC	FZS3PA(,@BR),I\$WRK1(@CADDR) SET PRINT AREA CORE ADDRESS	
		4418	*		
		4419	*	TEST FOR AN ARITHMETIC ELEMENT - RETURN CARRIER IF ARITHMETIC	
		4420	*	ELEMENT LENGTH EXCEEDS OUTPUT LINE MARGIN	
		4421	*		
361B 5C 00 DB F3		4422	MVC	FZS3CC(,@BR),FZS3PC(1,@BR) SET PARAM = ELEMENT CHAR CNT	
361F 3D 12 0BA1		4423	CLI	I\$SLLC,I@LCRV-1 IF CURR ELEMENT IS ARITHMETIC ?	
3623 D0 01 D2		4424	BNE	FZS760(,@BR) * LINK TO RETURN CARR ON COND	
		4425	*		
		4426	*	BRANCH TO APPROPRIATE ROUTINE DEPENDING ON CONTROL CODE	
		4427	*		
3626 4C 00 2D 0D57		4428	MVC	FZS605+@D1(,@BR),I\$PARM(1) MOVE CONTROL DISP TO JUMP INST	
362B F2 87 00		4429	FZS605 J	*-* GO EXECUTE CONTROL CODE ROUTINE	
		4430	*		
		4431		*****	

FZSPRT - S/3 BASIC INTERPRETER PRINT STATEMENT EXEC RTN

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 281
		4433		*****			
		4434		* OUTPUT ROUTINE FOR PRINT CONTROL CODES 1, 9, 11, 13, 15			*
		4435		*****			
		4436		*			
		4437		* PRINT THE FORMATTED ELEMENT ONLY (WHEN SIGNIFICANT)			
		4438		*			
362E	7D 00 F3	4439	FZS610	CLI FZS3PC(,@BR),@ZERO	IF ELEMENT CHAR COUNT NOT ZERO		
3631	F2 81 9A	4440		JE FZS750	EXIT ROUTINE W/O PRINTING	1-5	
3634	1C 01 144A FB	4441		MVC I\$VADR,FZSPCH(@VADDR,@BR)	VM PATCH PAGE ENTRY ADDR	1-5	
3639	C0 87 1358	4442		B I\$CVAD	LOAD PATCH PAGE	1-5	
363D	4C 01 45 144C	4443		MVC FZS615+@OP1(@CADDR,@BR),I\$CADR	MOVE CADDR TO BRANCH	1-5	
3642	C0 87 0000	4444	FZS615	B *-*	BRANCH TO PATCH PAGE	1-5	
		4445		*			
		4446		*****			
		4447		* OUTPUT ROUTINE FOR PRINT CONTROL CODE 2			*
		4448		*****			
		4449		*			
		4450		* ESTABLISH FULL PRINT ZONE OUTPUT FORMAT (ARITHMETIC ELEMENT)			
		4451		*			
3646	7C 12 DB	4452	FZS620	MVI FZS3CC(,@BR),I@LFPZ	SET PARAM - FULL PRINT ZONE		
3649	F2 87 18	4453		J FZS636	BRANCH TO TEST LINE CAPACITY		
		4454		*			
		4455		*****			

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 282
		4457		*****	
		4458		* OUTPUT ROUTINE FOR PRINT CONTROL CODE 3	*
		4459		*****	
		4460		*	
		4461		* ESTABLISH PACKED PRINT ZONE OUTPUT FORMAT (ARITHMETIC ELEMENT) -	
		4462		* THIS ZONE WILL BE 6, 9, 12, 15, OR 18 CHARACTERS LONG DEPENDING ON	
		4463		* THE LENGTH OF THE ARITHMETIC ELEMENT TO BE PRINTED.	
		4464		*	
364C	7C 04 DB	4465	FZS630 MVI	FZS3CC(,@BR),2*I@LPPZ-2	SET LENGTH ACCUM TO MINIMUM
		4466		*	* ELEMENT LENGTH LIMIT (4)
364F	5D 00 F3 DB	4467	FZS632 CLC	FZS3PC(,@BR),FZS3CC(1,@BR)	IF ELEMENT LENGTH WITHIN LIMIT
3653	F2 04 0A	4468		JNH FZS634	* BRANCH TO EXIT THIS LOOP
3656	5E 00 DB F1	4469		ALC FZS3CC(,@BR),FZS3PZ(1,@BR)	ADD PACKED ZONE INCR TO ACCUM
365A	7D 10 DB	4470	FZS633 CLI	FZS3CC(,@BR),I@LFPZ-2	IF LENGTH ACCUM NOT MAXIMUM
365D	D0 82 4F	4471		BL FZS632(,@BR)	* GO REPEAT ELEMENT LENGTH TEST
		4472		*	
3660	5E 00 DB F0	4473	FZS634 ALC	FZS3CC(,@BR),FZS3B2(1,@BR)	ADJUST ACCUM TO MAKE PACKED
		4474		*	* PRINT ZONE FIELD LENGTH
		4475		*	
		4476		* TEST LINE CAPACITY TO CONTAIN CURRENT PRINT ZONE FIELD - WHEN RIGHT	
		4477		* MARGIN IS EXCEEDED, LINE HAS CAPACITY FOR THE DATA ELEMENT BUT NOT	
		4478		* FOR THE ENTIRE PRINT ZONE ... IN THIS CASE, PRINT ELEMENT ONLY AND	
		4479		* RETURN THE CARRIER	
		4480		*	
3664	4E 00 DB 03C2	4481	FZS636 ALC	FZS3CC(,@BR),\$PRPOS(1)	ADD PRINT ZONE LNG TO CURRENT
3669	7D 00 DB	4482	FZS638 CLI	FZS3CC(,@BR),*-*	* CARRIER POSITION - BRANCH
366C	F2 84 12	4483		JH FZS655	* IF RIGHT MARGIN IS EXCEEDED
		4484		*	
		4485		* LINE HAS CAPACITY FOR ENTIRE PRINT ZONE - PRINT ELEMENT AND SPACE	
		4486		* TO THE SPECIFIED ZONE POSITION	
		4487		*	
366F	4F 00 DB 03C2	4488	FZS640 SLC	FZS3CC(,@BR),\$PRPOS(1)	RESTORE CURRENT PRINT ZONE LNG
3674	5C 00 F3 DB	4489		MVC FZS3PC(,@BR),FZS3CC(1,@BR)	SET COUNT - CAR PRT ZONE LNG
3678	F2 87 3E	4490		J FZS710	GO PRINT ELEMENT AND SPACE CARR
		4491		*	
		4492		*****	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 283
		4494		*****	
		4495		* OUTPUT ROUTINE FOR PRINT CONTROL CODES 4, 12, 16	*
		4496		*****	
		4497		*	
		4498		* TEST ELEMENT SIGNIFICANCE - RETURN CARRIER ONLY WHEN NOT SIGNIFICANT	
		4499		*	
367B	7D 00 F3	4500	FZS650	CLI FZS3PC(,@BR),@ZERO	ELEMENT CHAR COUNT IS ZERO ?
367E	F2 81 20	4501		JE FZS680	* GO RETURN THE CARRIER ONLY
		4502		*	
		4503		* ELEMENT IS SIGNIFICANT - PRINT ELEMENT AND RETURN CARRIER	
		4504		*	
3681	7C C0 F2	4505	FZS655	MVI FZS3PF(,@BR),@PRETR	SET PRINT & CARR RETURN FUNC
3684	F2 87 32	4506		J FZS710	GO PRINT ELEMENT AND RTRN CARR
		4508		*****	
		4509		* OUTPUT ROUTINE FOR PRINT CONTROL CODE 5	*
		4510		*****	
		4511		*	
		4512		* ESTABLISH FULL PRINT ZONE SPACING ONLY	
		4513		*	
3687	7C 12 F3	4514	FZS660	MVI FZS3PC(,@BR),I@LFPZ	SET COUNT FOR FULL PRINT ZONE
368A	F2 87 03	4515		J FZS675	BRANCH TO EXECUTE SPACING
		4517		*****	
		4518		* OUTPUT ROUTINE FOR PRINT CONTROL CODE 6	*
		4519		*****	
		4520		*	
		4521		* ESTABLISH PACKED PRINT ZONE INCREMENT SPACING ONLY	
		4522		*	
368D	7C 03 F3	4523	FZS670	MVI FZS3PC(,@BR),I@LPPZ	SET COUNT FOR PACKED ZONE INCR
		4524		*	
		4525		* PRINT CURRENT ZONE SPACE, OR RETURN CARRIER IF END OF LINE IS HIT	
		4526		*	
3690	5C 00 DB F3	4527	FZS675	MVC FZS3CC(,@BR),FZS3PC(1,@BR)	SET PARAM FOR CURRENT ZONE LNG
3694	D0 87 D2	4528		B FZS760(,@BR)	LINK TO RETURN CARRIER ON COND
3697	5D 00 DB 6A	4529		CLC FZS3CC(,@BR),FZS3RM(1,@BR)	IF CARRIER WAS NOT RETURNED
369B	F2 04 1B	4530		JNH FZS710	* GO PRINT CURRENT ZONE SPACE,
369E	F2 87 2D	4531		J FZS750	* ELSE EXIT RTN W/0 PRINTING
		4532		*	
		4533		*****	

FZSPRT - S/3 BASIC INTERPRETER PRINT STATEMENT EXEC RTN

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 284
		4535		*****	
		4536		* OUTPUT ROUTINE FOR PRINT CONTROL CODE 7	*
		4537		*****	
		4538		*	
		4539		* ESTABLISH CARRIER RETURN ONLY	
		4540		*	
36A1 D2 02 F6		4541	FZS680 LA	FZS3CR(,@BR),@XR	LOAD CARRIER RETURN PPL CADDR
36A4 F2 87 15		4542	J	FZS720	GO EXECUTE CARRIER RETURN
		4544		*****	
		4545		* OUTPUT ROUTINE FOR PRINT CONTROL CODE 8	*
		4546		*****	
		4547		*	
		4548		* RETURN CARRIER IF FULL PRINT ZONE EXCEEDS LINE CAPACITY	
		4549		*	
36A7 7C 12 DB		4550	FZS690 MVI	FZS3CC(,@BR),I@LFPZ	SET PARAM FOR PRINT ZONE
36AA D0 87 D2		4551	B	FZS760(,@BR)	LINK TO RETURN CARRIER ON COND
36AD F2 87 0F		4552	J	FZS730	GO TEST FOR CRT ACTIVE ON SYSTEM
		4554		*****	
		4555		* OUTPUT ROUTINE FOR PRINT CONTROL CODE 10	*
		4556		*****	
		4557		*	
		4558		* RETURN CARRIER IF FULL PRINT ZONE EXCEEDS LINE CAPACITY	
		4559		*	
36B0 7C 12 DB		4560	FZS695 MVI	FZS3CC(,@BR),I@LFPZ	SET PARAM FOR FULL PRINT ZONE
36B3 D0 87 D2		4561	B	FZS760(,@BR)	LINK TO RETURN CARRIER ON COND
		4562		*	
		4563		*****	

FZSPRT - S/3 BASIC INTERPRETER PRINT STATEMENT EXEC RTN

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 285
		4565		*****	
		4566		* OUTPUT ROUTINE FOR PRINT CONTROL CONTROL CODE 14	
		4567		*****	
		4568		*	
		4569		* ESTABLISH FULL PRINT ZONE OUTPUT FORMAT (CHARACTER ELEMENT)	
		4570		*	
36B6	7C 12 F3	4571	FZS700	MVI FZS3PC(, @BR), I@LFPZ SET COUNT FOR ZONE	
		4572		*	
		4573		* EXECUTE ELEMENT OUTPUT TO THE MATRIX PRINTER	
		4574		*	
36B9	D2 02 F2	4575	FZS710	LA FZS3PL(, @BR), @XR LOAD DATA OLTOLT CORE ADOR	
36BC	D0 87 E3	4576	FZS720	B FZS780(, @BR) LINK TO EXECUTE PRINTER OUTPUT	
		4577		*	
		4578		* TEST FOR THE CRT ACTIVE AS A SISTEM PRINT DEVICE	
		4579		*	
36BF	0D 00 044A 0D5A	4580	FZS730	CLC \$PRDEV-1, I\$WRK2-1(1) IF CRT IS NOT A SYSTEM PRINT	
36C5	F2 82 06	4581		JL FZS750 * DEVICE, GO EXIT THIS ROUTINE	
		4582		*	
		4583		* CRT ACTIVE - SET UP AND OUTPUT TO CRT USINS CRT LINE WIDTH	
		4584		*	
36C8	C0 87 12B1	4585	FZS740	B I\$CALL LINK TO EXECUTE PRINT ON CRT	
36CC	3700	36CD 4586		DC AL(@VADDR)(FZS800) PRINT CRT RTN VIRTUAL ADDRESS	
		4587		*	
		4588		* RETURN TO PTINT ROUTINE 1ST VM PAGE	
		4589		*	
36CE	C0 87 12D3	4590	FZS750	B I\$RTRN RETURN TO 1ST PRINT RTN PAGE	
		4591		*	
		4592		*****	

FZSPRT - S/3 BASIC INTERPRETER PRINT STATEMENT EXEC RTN

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 286
		4594		*****	
		4595	*	PRINTER CARRIER RETURN ROUTINE -	*
		4596	*	* RETURNS PRINTER CARRIER WHEN SPECIFIED LENGTH PARAMETER	*
		4597	*	(FZS3CC) EXCEEDS THE CURRENT PRINT LINE CAPACITY.	*
		4598		*****	
		4599	*		
36D2	74 08 EF	4600	FZS760 ST	FZS790+@OP1(,@BR),@ARR	STORE RETURN BRANCH ADDRESS
		4601	*		
		4602	*	TEST LINE CAPACITY TO CONTAIN CURRENT PRINT REGION LENGTH	
		4603	*		
36D5	4E 00 DB 03C2	4604		ALC FZS3CC(,@BR),\$PRPOS(1)	ADD PRINT REGION LENGTH TO CURR
36DA	7D 00 6A	4605	FZS770 CLI	FZS3RM(,@BR),*-*	* CARRIER POSITION - BRANCH IF
36DD	F2 02 0C	4606		JNL FZS790	* RIGHT MARGIN NOT EXCEEDED
		4607	*		
		4608	*	RIGHT MARGIN EXCEEDED - RETURN MATRIX PRINTER CARRIER	
		4609	*		
36E0	D2 02 F6	4610		LA FZS3CR(,@BR),@XR	LOAD CARRIER RETURN PPL CADDR
		4612		*****	
		4613	*	PRINTER OUTPUT INTERFACE -	*
		4614	*	* EXECUTES MATRIX PRINTER OUTPUT AS SPECIFIED IN PRINT PARAM-	*
		4615	*	ETER LIST REFERENCED BY REGISTER @XR.	*
		4616		*****	
36E3	74 08 EF	4617	FZS780 ST	FZS790+@OP1(,@BR),@ARR	STORE RETURN BRANCH ADDRESS
36E6	C0 87 12B1	4618		B I\$CALL	LINK TO EXECUTE PRINTER IOCR
36EA	2800	36EB 4619		DC AL(@VADDR)(V\$SPRT)	MATRIX PRINTER IOCR VADDR
		4620	*		
		4621	*	RETURN TO CALLING ROUTINE	
		4622	*		
36EC	C0 87 0000	4623	FZS790 B	*-*	RETURN BRANCH
		4624		*****	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 287
		4626		*****	
		4627	*	PRINT EXECUTION ROUTINE CONSTANTS (3RD VM PAGE)	*
		4628		*****	
		4629	*		
36F0 02		36F0 4630	FZS3B2 DC	IL1'2'	BINARY INTEGER +2
		4631	*		
36F1 03		36F1 4632	FZS3PZ DC	AL1(I@LPPZ)	LENGTH OF PACKED ZONE INCR
		4634		*****	
		4635	*	PRINT EXECUTION ROUTINE WORK AREAS (3RD VM PAGE)	*
		4636		*****	
		366A 4637	FZS3RM EQU	FZS638+@Q	MATRIX PRINTER RIGHT MARGIN
		36DB 4638	FZS3CC EQU	FZS770+@Q	PRINT AREA CHARACTER COUNT
		4639	*		
		4640	*FZS3PL PPL		
		36F2 4641	FZS3PL EQU	*	PPL ADDRESS
36F2 00		36F2 4642	DC	AL1(*-*)	FUNCTION REQUESTED
36F3 00		36F3 4643	DC	AL1(*-*)	PRINT COUNT
36F4 0000		36F5 4644	DC	AL2(*-*)	DATA ADDRESS
		4645	***	END OF EXPANSION ***	
		4646	*		
		36F2 4647	FZS3PF EQU	FZS3PL+@PCTRL	PRINT FUNCTION PARAMETER
		36F3 4648	FZS3PC EQU	FZS3PL+@PRCNT	PRINT AREA COUNT PARAMETER
		36F5 4649	FZS3PA EQU	FZS3PL+@PDATA	PRINT AREA COUNT PARAMETER
		4650	*		
		4651	*FZS3CR PPL	FUNC-@RETRN,CNT-@RTRNC	
		36F6 4652	FZS3CR EQU	*	PPL ADDRESS
36F6 80		36F6 4653	DC	AL1(@RETRN)	FUNCTION REQUESTED
36F7 80		36F7 4654	DC	AL1(@RTRNC)	PRINT COUNT
36F8 0000		36F9 4655	DC	AL2(*-*)	DATA ADDRESS
		4656	***	END OF EXPANSION ***	
		4657	*		
36FA 5359		36FB 4658	FZSPCH DC	AL2(V\$PCH2+FZS633-@Q-FZSP3B)	PATCH PAGE ENTRY ADDR 1-3
		4659		*****	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 288
		4661		*****	
		4662	*	VIRTUAL MEMORY PRINT EXECUTION ROUTINE 4TH VM PAGE	*
		4663	*	* OUTPUTS FORMATTED DATA ELEMENT TO CRT DISPLAY UNIT	*
		4664	*	* CONTROLS CRT CURSOR DEPENDING ON SPECIFIED CONTROL CODE	*
		4665	*		*
		4666	*	INPUT -	*
		4667	*	* RUN-TIME STACK - CONTAINS FORMATTED ELEMENT, IF PRESENT	*
		4668	*	* I\$PARM - 1 BYTE, CONTAINS CONTROL CODE BRANCH DISPLACEMENT	*
		4669	*	* I\$PARM-1 - 1 BYTE, CONTAINS FORMATTED ELEMENT CHARACTER COUNT	*
		4670	*	* I\$WRK1 - 2 BYTES, CONTAINS CORE ADDR OF PRINT AREA LEFT BYTE	*
		4671	*	* I@WRK2 - 2 BYTES, CONTAINS VALUE FOR \$PRDEV 'CRT ONLY' COND	*
		4672	*	* ISSLLC - 1 BYTE, CONTAINS OUTPUT ELEMENT LENGTH CODE (LNG - 1)	*
		4673	*		*
		4674	*	OUTPUT -	*
		4675	*	* DISPLAYED ELEMENT AND/OR CURSOR CONTROL ON CRT DISPLAY UNIT	*
		4676		*****	
		4677	*		
		4678	*	ESTABLISH ADDRESSABILITY FOR PRINT ROUTINE (4TH VM PAGE)	
		4679	*		
		4680		*FZSP4B VPAGE 0 SET 4TH PAGE ADDRESSABILITY	
3700		4681		ORG *,256,0 SET STARTING ADDRESS	
		4682	FZSP4B EQU *	START OF PROGRAM CODING	
3601		4683		ORG *-255 RESET IAR TO PAGE	
3700		4684		ORG *,256,0 * BOUNDARY ADDRESS	
		4685		USING *,@BR SET PAGE BASE ADDRESS	
3700		4686		ORG FZSP4B RESET STARTING ADDRESS	
		4687		*** END OF EXPANSION ***	
		4688	*		
		4689	*	PAGE ENTRY - ESTABLISH CRT IOCR EXECUTION CORE ADDRESS	
		4690	*		
3700 4C 01 D7 0D5B		4691	FZS800 MVC	FZS982+@OP1(,@BR),I\$WRK2(@CADDR) SET CRT EXECUTION CADDR	
		4692	*		
		4693	*	INITIALIZE FOR OUTPUT TO THE CRT DISPLAY UNIT	
		4694	*		
3705 7C 40 64		4695		MVI FZS4RM(,@BR),@DLNLG SET CRT RIGHT MARGIN PARAMETER	
		4696	*		
		4697	*	INITIALIZE THE ELEMENT PRINT PARAMETER LIST	
		4698	*		
3708 7C 40 E0		4699		MVI FZS4PF(,@BR),@PRINT SET FUNCTION FOR PRINT ONLY	
370B 4C 00 E1 0D56		4700		MVC FZS4PC(,@BR),I\$PARM-1(1) SET COUNT - ELEMENT CHAR COUNT	
3710 4C 01 E3 0D59		4701		MVC FZS4PA(,@BR),I\$WRK1(@CADDR) SET PRINT AREA CODE ADDRESS	
		4702	*		
		4703	*	TEST FOR AN ARITHMETIC ELEMENT - RETURN CURSOR IF ARITHMETIC	
		4704	*	ELEMENT LENGTH EXCEEDS OUTPUT LINE MARGIN	
		4705	*		
3715 5C 00 C6 E1		4706		MVC FZS4CC(,@BR),FZS4PC(1,@BR) SET PARAM = ELEMENT CHAR CNT	
3719 3D 12 0BA1		4707		CLI I\$SLLC,I@LCRV-1 IF CURR ELEMENT IS ARITHMETIC	
371D D0 01 BD		4708		BNE FZS960(,@BR) * LINK TO RTRN CURSOR ON COND	
		4709	*		
		4710	*	BRANCH TO APPROPRIATE ROUTINE DEPENDING ON CONTROL CODE	
		4711	*		
3720 4C 00 27 0D57		4712		MVC FZS805+@D1(,@BR),I\$PARM(1) MOVE CONTROL DISP TO JUMP INST	
3725 F2 87 00		4713	FZS805 J	*-* GO EXEC CONTROL CODE ROUTINE	
		4714	*		
		4715		*****	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 289
		4717		*****			
		4718		* OUTPUT ROUTINE FOR PRINT CONTROL CODES 1, 9, 11, 13, 15			*
		4719		*****			
		4720		*			
		4721		* DISPLAY THE FORMATTED ELEMENT ONLY (WHEN SIGNIFICANT)			
		4722		*			
3728	7D 00 E1	4723	FZS810	CLI FZS4PC(, @BR), @ZERO	IF ELEMENT CHAR COUNT NOT ZERO		
372B	F2 01 85	4724		JNE FZS910	* GO DISPLAY ELEMENT ONLY,		
372E	F2 87 88	4725		J FZS950	* ELSE EXIT RTN W/O DISPLAYING		
		4726		*			
3731	000000000000000000	373F 4727		DC XL15'00'	PATCH SPACE	1-5	
		4729		*****			
		4730		* OUTPUT ROUTINE FOR PRINT CONTROL CODE 2			*
		4731		*****			
		4732		*			
		4733		* ESTABLISH FULL PRINT ZONE OUTPUT FORMAT (ARITHMETIC ELEMENT)			
		4734		*			
3740	7C 12 C6	4735	FZS820	MVI FZS4CC(, @BR), I@LFPZ	SET PARAM = FULL PRINT ZONE		
3743	F2 87 18	4736		J FZS836	BRANCH TO TEST LINE CAPACITY		
		4737		*			
		4738		*****			

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 290
		4740		*****	
		4741	*	OUTPUT ROUTINE FOR PRINT CONTROL CODE 3	*
		4742		*****	
		4743	*		
		4744	*	ESTABLISH PACKED PRINT ZONE OUTPUT FORMAT (ARITHMETIC ELEMENT) -	
		4745	*	THIS ZONE WILL BE 6, 9, 12, 15, OR 18 CHARACTERS LONG DEPENDING ON	
		4746	*	THE LENGTH OF THE ARITHMETIC ELEMENT TO BE PRINTED	
		4747	*		
3746	7C 04 C6	4748	FZS830 MVI	FZS4CC(,@BR),2*I@LPPZ-2	SET LENGTH ACCUN TO MINIMUM
		4749	*		* ELEMENT LENGTH LIMIT (4)
3749	5D 00 E1 C6	4750	FZS832 CLC	FZS4PC(,@BR),FZS4CC(1,@BR)	IF ELEMENT LENGTH WITHIN LIMIT
374D	F2 04 0A	4751	JNH	FZS834	* BRANCH TO EXIT THIS LOOP
3750	5E 00 C6 DF	4752	ALC	FZS4CC(,@BR),FZS4PZ(1,@BR)	ADD PACKED ZONE INCR TO ACCUM
3754	7D 10 C6	4753	CLI	FZS4CC(,@BR),I@LFPZ-2	IF LENGTH ACCUM NOT MAXIMUM
3757	D0 82 49	4754	BL	FZS832(,@BR)	* GO REPEAT ELEMENT LENGTH TEST
		4755	*		
375A	5E 00 C6 DE	4756	FZS834 ALC	FZS4CC(,@BR),FZS4B2(1,@BR)	ADJUST ACCUM TO MAKE PACKED
		4757	*		* PRINT ZONE FIELD LENGTH
		4758	*		
		4759	*	TEST LINE CAPACITY TO CONTAIN CURRENT POINT ZONE FIELD - WHEN RIGHT	
		4760	*	MARGIN IS EXCEEDED, LINE HAS CAPACITY FOR TED DATA ELEMENT BUT NOT	
		4761	*	FOR THE ENTIRE PRINT ZONE ... IN THIS CASE, DISPLAY ELMEMENMT ONLY	
		4762	*	AND RETURN THE CURSOR.	
		4763	*		
375E	4E 00 C6 03E2	4764	FZS836 ALC	FZS4CC(,@BR),\$CRPOS(1)	ADD PRINT ZONE LNG TO CURRENT
3763	7D 00 C6	4765	FZS838 CLI	FZS4CC(,@BR),*-*	* CURSOR POSITION - BRANCH
3766	F2 84 12	4766	JH	FZS855	* IF RIGHT MARGIN IS EXCEEDED
		4767	*		
		4768	*	LINE HAS CAPACITY FOR ENTIRE PRINT ZONE - DISPLAY ELEMENT AND SPACE	
		4769	*	TO THE SPECIFIED ZONE POSITION	
		4770	*		
3769	4F 00 C6 03E2	4771	FZS840 SLC	FZS4CC(,@BR),\$CRPOS(1)	RESTORE CURRENT PRINT ZONE LNG
376E	5C 00 E1 C6	4772	MVC	FZS4PC(,@BR),FZS4CC(1,@BR)	SET COUNT = CURR PRT ZONE LNG
3772	F2 87 3E	4773	J	FZS910	GO DISPLAY ELEM & SPACE CURSOR
		4774	*		
		4775		*****	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 291
		4777		*****	
		4778		* OUTPUT ROUTINE FOR PRINT CONTROL CODES 4, 12, 16	*
		4779		*****	
		4780		*	
		4781		* TEST ELEMENT SIGNIFICANCE - RETURN CURSOR NO WHEN NOT SIGNIFICANT	
		4782		*	
3775	7D 00 E1	4783	FZS850	CLI FZS4PC(, @BR), @ZERO	IF ELEMENT CHAR COUNT IS ZERO
3778	F2 81 20	4784		JE FZS880	* GO RETURN THE CURSOR ONLY
		4785		*	
		4786		* ELEMENT IS SIGNIFICANT - DISPLAY ELEMENT AND RETURN CURSOR	
		4787		*	
377B	7C C0 E0	4788	FZS855	MVI FZS4PF(, @BR), @PRETR	SET PRINT & CARR RETURN FUNC
377E	F2 87 32	4789		J FZS910	GO DISPLAY ELEM AND RTRN CURSOR
		4791		*****	
		4792		* OUTPUT ROUTINE FOR PRINT CONTROL CODE 5	*
		4793		*****	
		4794		*	
		4795		* ESTABLISH FULL PRINT ZONE SPACING ONLY	
		4796		*	
3781	7C 12 E1	4797	FZS860	MVI FZS4PC(, @BR), I@LFPZ	SET CO:AT R04 FLU *QM ZONE
3784	F2 87 03	4798		J FZS875	BRANCH TO EXEC?TE SPACINS
		4800		*****	
		4801		* OUTPUT ROUTINE FOR PRINT COHT4OL CODE 6	*
		4802		*****	
		4803		*	
		4804		* ESTABLISH PACKED PRINT ZONE INCREMENT SPACING ONLY	
		4805		*	
3787	7C 03 E1	4806	FZS870	MVI FZS4PC(, @BR), I@LPPZ	SET COUNT FOR PACKED ZONE INCR
		4807		*	
		4808		* DISPLAY CURRENT ZONE, OR RETURN CURSOR IF END OF LINE IS HIT	
		4809		*	
378A	5C 00 C6 E1	4810	FZS875	MVC FZS4CC(, @BR), FZS4PC(1, @BR)	SET PARAM FOR CURRENT ZONE LNG
378E	D0 87 BD	4811		B FZS960(, @BR)	LINK TO RETURN CURSOR ON COND
3791	5D 00 C6 64	4812		CLC FZS4CC(, @BR), FZS4RM(1, @BR)	IF CURSOS WAS NOT RETURNED
3795	F2 04 1B	4813		JNH FZS910	* GO DISPLAY CURR ZONE SPACE
3798	F2 87 1E	4814		J FZS950	* ELSE EXIT RTN W/O DISPLAYING
		4815		*	
		4816		*****	

FZSPRT - S/3 BASIC INTERPRETER PRINT STATEMENT EXEC RTN

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 292
		4818		*****			
		4819		* OUTPUT ROUTINE FOR PRINT CONTROL CODE 7			*
		4820		*****			
		4821		*			
		4822		* ESTABLISH CURSOR RETURN ONLY			
		4823		*			
379B	D2 02 E4	4824	FZS880	LA FZS4CR(,@BR),@XR		LOAD CURSOR RETURN PPL CADDR	
379E	F2 87 15	4825		J FZS920		GO EXECUTE CURSOR RETURN	
		4827		*****			
		4828		* OUTPUT ROUTINE FOR PRINT CONTROL CODE 8			*
		4829		*****			
		4830		*			
		4831		* RETURN CURSOR IF FULL PRINT ZONE EXCEEDS LINE CAPACITY			
		4832		*			
37A1	7C 12 C6	4833	FZS890	MVI FZS4CC(,@BR),I@LFPZ		SET PARAM FOR FULL PRINT ZONE	
37A4	D0 87 BD	4834		B FZS960(,@BR)		LINK TO RETURN CLRSR ON COND	
37A7	F2 87 0F	4835		J FZS950		GO EXIT DISPLAY ROUTINE	
		4837		*****			
		4838		* OUTPUT ROUTINE FOR PRINT CONTROL CODE 10			*
		4839		*****			
		4840		*			
		4841		* RETURN CURSOR IF FULL PRINT ZONE EXCEEDS LINE CAPACITV			
		4842		*			
37AA	7C 12 C6	4843	FZS895	MVI FZS4CC(,@BR),I@LFPZ		SET PARAM FOR FULL PRINT ZONE	
37AD	D0 87 BD	4844		B FZS960(,@BR)		LINK TO RETURN CURSOS ON COND	
		4845		*			
		4846		*****			

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 293
		4848		*****	
		4849		* OUTPUT ROUTINE FOR PRINT CONTROL CODE 14	*
		4850		*****	
		4851		*	
		4852		* ESTABLISH FULL PRINT ZONE OUTPUT FORMAT (CHARACTER ELEMENT)	
		4853		*	
37B0	7C 12 E1	4854	FZS900	MVI FZS4PC(,@BR),I@LFPZ	SET COUNT FOR FULL PRINT ZONE
		4855		*	
		4856		* EXECUTE ELEMENT OUTPUT TO THE CRT DISPLAY UNIT	
		4857		*	
37B3	D2 02 E0	4858	FZS910	LA FZS4PL(,@BR),@XR	LOAD DATA OUTPUT PPL CORE ADDR
		4859		*	
37B6	D0 87 CE	4860	FZS920	B FZS980(,@BR)	LINK TO EXECUTE CRT OUTPUT
		4861		*	
		4862		* RETURN TO PRINT ROUTINE 3RD VM PAGE	
		4863		*	
37B9	C0 87 12D3	4864	FZS950	B I\$RTRN	RETURN TO 3RD PRINT RTN PAGE
		4865		*	
		4866		*****	

FZSPRT - S/3 BASIC INTERPRETER PRINT STATEMENT EXEC RTN

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 294
		4868		*****	
		4869	*	DISPLAY UNIT CURSOR RETURN ROUTINE -	*
		4870	*	* RETURNS CURSOR WHEN SPECIFIED LENGTH PARAMETER (FZS4CC)	*
		4871	*	EXCEEDS THE CURRENT CRT DISPLAY LINE CAPACITY.	*
		4872		*****	
		4873	*		
37BD	74 08 DD	4874	FZS960 ST	FZS990+@OP1(,@BR),@ARR	STORE RETURN BRANCH ADDRESS
		4875	*		
		4876	*	TEST LINE CAPACITY TO CONTAIN CURRENT DISPLAY REGION LENGTH	
		4877	*		
37C0	4E 00 C6 03E2	4878		ALC FZS4CC(,@BR),\$CRPOS(1)	ADD PRINT REGION LENGTH TO CURR
37C5	7D 00 64	4879	FZS970 CLI	FZS4RM(,@BR),*-*	* CURSOR POSITION - BRANCH IF
37C8	F2 02 0F	4880		JNL FZS990	* RIGHT MARGIN NOT EXCEEDED
		4881	*		
		4882	*	RIGHT MARGIN EXCEEDED - RETURN DISPLAY UNIT CURSOR	
		4883	*		
37CB	D2 02 E4	4884		LA FZS4CR(,@BR),@XR	LOAD CURSOR RETURN PPL CADDR
		4885	*		
		4886		*****	
		4887	*	DISPLAY UNIT OUTPUT INTERFACE -	*
		4888	*	* EXECUTES CRT DISPLAY OUTPUT AS SPECIFIED IN PRINT PARAMETER	*
		4889	*	* LIST REFERENCED BY REGISTER @XR.	*
		4890		*****	
		4891	*		
37CE	74 08 DD	4892	FZS980 ST	FZS990+@OP1(,@BR),@ARR	STORE RETURN BRANCH ADDRESS
		4893	*		
37D1	74 02 D9	4894		ST FZS984(,@BR),@XR	STORE PPL CORE ADDRESS
37D4	C0 87 0000	4895	FZS982 B	*-*	LINK TO EXECUTE CRT IOCR
37D8		4896	FZS984 DS	CL(@CADDR)	CRT IOCS PARAMETER LIST CADDR
		4897	*		
		4898	*	RETURN TO CALLING ROUTINE	
		4899	*		
37DA	C0 87 0000	4900	FZS990 B	*-*	RETURN BRANCH
		4901	*		
		4902		*****	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 295
		4904		*****	
		4905		* PRINT EXECUTION ROUTINE CONSTANTS (4TH VM PAGE)	*
		4906		*****	
		4907		*	
37DE 02		37DE 4908	FZS4B2 DC	IL1'2'	BINARY INTEGER +2
		4909		*	
37DF 03		37DF 4910	FZS4PZ DC	AL1(I@LPPZ)	LENGTH OF PACKED ZONE INCR
		4911		*	
		4912		*****	
		4913		* PRINT EXECUTION ROUTINE WORK AREAS (4TH VM PAGE)	*
		4914		*****	
		4915		*	
		3764 4916	FZS4RM EQU	FZS838+@Q	CRT DISPLAY RIGHT MARGIN
		37C6 4917	FZS4CC EQU	FZS970+@Q	PRINT AREA CHARACTER COUNT
		4918		*	
		4919		*FZS4PL PPL	
		37E0 4920	FZS4PL EQU	*	PPL ADDRESS
37E0 00		37E0 4921		DC AL1(*-*)	FUNCTION REQUESTED
37E1 00		37E1 4922		DC AL1(*-*)	PRINT COUNT
37E2 0000		37E3 4923		DC AL2(*-*)	DATA ADDRESS
		4924		*** END OF EXPANSION ***	
		37E0 4926	FZS4PF EQU	FZS4PL+@PCTRL	PRINT FUNCTION PARAMETER
		37E1 4927	FZS4PC EQU	FZS4PL+@PRCNT	PRINT AREA COUNT PARAMETER
		37E3 4928	FZS4PA EQU	FZS4PL+@PDATA	PRINT AKEA CADDR PARAMETER
		4929		*	
		4930		*FZS4CR DPL FUNC=@REYRN,CNT=@RTRNC	
		37E4 4931	FZS4CR EQU	*	PPL ADDRESS
37E4 80		37E4 4932		DC AL1(@RETRN)	FUNCTION REQUESTED
37E5 80		37E5 4933		DC AL1(@RTRNC)	PRINT COUNT
37E6 0000		37E7 4934		DC AL2(*-*)	DATA ADDRESS
		4935		*** END OF EXPANSION ***	
		4936		*	
		4937		*****	
		4938		*	
		4939		*** END OF PRINT EXECUTION ROUTINE CODING ***	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 296
		4941		*****			
		4942	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
		4943	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
		4944	*				*
		4945		*****			*
		4946	*	*STATUS			*
		4947	*	VERSION 1 MODIFICATION 0			*
		4948	*				*
		4949	*	*FUNCTION			*
		4950	*	* FZLPRT EXECUTION CAUSES A PRINT IMAGE TO BE ESTABLISHED IN			*
		4951	*	VIRTUAL MEMORY BUFFERS AND DATA ELEMENTS TO BE OUTPUT ON THE			*
		4952	*	SYSTEM PRINT DEVICE UNDER FORMAT CONTROL OF THE IMAGE CONVER-			*
		4953	*	SION SPECIFICATIONS.			*
		4954	*	* THE FOLLOWING ACTIONS ARE PERFORMED DEPENDING ON THE CODE			*
		4955	*	STORED IN INTERPRETER PARAMETER I\$PARM -			*
		4956	*	* CODE X'00' - RELEASE IMAGE.			*
		4957	*	VIRTUAL MEMORY PAGES CONTAINING THE CURRENTLY ESTABLISHED			*
		4958	*	IMAGE ARE UNLOCKED FOR REPLACEMENT DURING NORMAL PAGING			*
		4959	*	OPERATIONS.			*
		4960	*	* CODE X'01' - NULL IMAGE SPECIFICATION.			*
		4961	*	THIS CODE CAUSES A NULL IMAGE INDICATOR TO BE SET FOR			*
		4962	*	FUTURE 'PRINT USING' OPERATIONS. NO IMAGE BUFFERS ARE			*
		4963	*	ESTABLISHED.			*
		4964	*	* CODE X'02' - NULL PRINT LIST SPECIFICATION.			*
		4965	*	THIS CODE CAUSES THE CURRENTLY ESTABLISHED IMAGE TO BE			*
		4966	*	PRINTED, UP TO THE CHARACTER PRECEDING THE FIRST CONVER-			*
		4967	*	SION SPECIFICATION OR END OF IMAGE, AND THE CARRIER/CURSOR			*
		4968	*	RETURNED TO THE START OF THE NEXT LINE. A NULL IMAGE			*
		4969	*	RESULTS IN A SIMPLE CARRIER/CURSOR RETURN.			*
		4970	*	* CODE X'03' - NULL CHARACTER CONSTANT.			*
		4971	*	THIS CODE CAUSES THE NEXT AVAILABLE CONVERSION SPECIFICA-			*
		4972	*	TION IN THE IMAGE WORK BUFFER TO BE FILLED WITH BLANKS.			*
		4973	*	* CODE X'04' - FIRST IMAGE SEGMENT.			*
		4974	*	THIS CODE CAUSES THE CHARACTER CONSTANT SEGMENT AT THE TOP			*
		4975	*	OF THE RUN-TIME STACK TO BE ESTABLISHED AS THE FIRST IMAGE			*
		4976	*	SEGMENT IN THE IMAGE SAVE BUFFER.			*
		4977	*	* CODE X'05' - SECONDARY IMAGE SEGMENT.			*
		4978	*	THIS CODE CAUSES THE CHARACTER CONSTANT SEGMENT AT THE TOP			*
		4979	*	OF THE RUN-TIME STACK TO BE ADDED TO THE EXISTING IMAGE			*
		4980	*	SEGMENTS IN THE IMAGE SAVE BUFFER.			*
		4981	*	* CODE X'06' - PRIMARY DATA ELEMENT.			*
		4982	*	A PRIMARY DATA ELEMENT IS DEFINED AS A FLOATING POINT			*
		4983	*	VALUE, A CHARACTER ELEMENT, OR THE FIRST SEGMENT OF A			*
		4984	*	MULTI-SEGMENT CHARACTER CONSTANT LITERAL. THIS CODE			*
		4985	*	CAUSES THE PRIMARY DATA ELEMENT AT THE TOP OF THE RUN-TIME			*
		4986	*	STACK TO CONVERTED AND PLACED IN THE IMAGE WORK BUFFER			*
		4987	*	ACCORDING TO THE NEXT AVAILABLE CONVERSION SPECIFICATION.			*
		4988	*	* CODE X'07' - SECONDARY DATA ELEMENT.			*
		4989	*	A SECONDARY DATA ELEMENT IS DEFINED AS ANY SEGMENT (EXCEPT			*
		4990	*	THE FIRST) OF A MULTI-SEGMENT CHARACTER CONSTANT LITERAL.			*
		4991	*	THIS CODE CAUSES THE SECONDARY DATA ELEMENT AT THE TOP OF			*
		4992	*	THE RUN-TIME STCK TO BE CONVERTED AND PLACED IN THE IMAGE			*
		4993	*	WORK BUFFER ACCORDING TO THE CURRENTLY REFERENCED CONVER-			*
		4994	*	SION SPECIFICATION (I.E, ADDED TO THE CURRENT CONTENTS OF			*
		4995	*	THE CONVERSION SPECIFICATION).			*
		4996	*				*

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 297
		4997	*	* OPERATIONS INVOLVING THE 'NEXT AVAILABLE' CONVERSION SPECIFI-			*
		4998	*	CATION IMPLY THE FOLLOWING ACTIONS -			*
		4999	*	* WHEN AN IMAGE IS TO BE PRINTED, THE CARRIER/CURSOR IS			*
		5000	*	RETURNED TO THE START OF THE NEXT LINE (BEFORE PRINTING			*
		5001	*	OCCURS) WHEN NOT ALREADY POSITIONED AT THE START OF THE			*
		5002	*	CURRENT LINE.			*
		5003	*	* WHEN NO UNFILLED CONVERSION SPECIFICATION REMAINS IN THE			*
		5004	*	IMAGE WORK BUFFER, THE FILLED IMAGE IS PRINTED AND THE			*
		5005	*	CARRIER/CURSOR RETURNED TO THE START OF THE NEXT LINE.			*
		5006	*	* FOLLOWING THE PRECEDING ACTION, ALL CONVERSION SPECIFICA-			*
		5007	*	TIONS IN THE IMAGE BECOME AVAILABLE, WITH THE 'NEXT AVAIL-			*
		5008	*	ABLE' SPECIFICATION BEING THE FIRST CONTAINED IN THE IMAGE.			*
		5009	*				*
		5010	*	* IN CONJUNCTION WITH THE ABOVE CODES, THESE INDICATORS MAY BE			*
		5011	*	SET IN I\$PARM AS CODE MODIFIERS -			*
		5012	*	* MASK X'10' - TERMINATE PRINT USING.			*
		5013	*	THIS INDICATOR CAUSES THE IMAGE TO BE PRINTED, UP TO THE			*
		5014	*	CHARACTER PRECEDING THE NEXT CONVERSION SPECIFICATION OR			*
		5015	*	END OF IMAGE, FOLLOWING THE ACTIVITY SPECIFIED BY THE CON-			*
		5016	*	TOOL CODE ITSELF. THE CARRIER/CURSOR IS RETURNED TO THE			*
		5017	*	START OF THE NEXT LINE, AND ALL IMAGE BUFFERS ARE RELEASED			*
		5018	*	(UNLOCKED) FROM CORE VIRTUAL MEMORY.			*
		5019	*	* MASK X'20' - MATRIX END OF ROW.			*
		5020	*	THIS INDICATOR CAUSES THE IMAGE TO BE PRINTED, UP TO THE			*
		5021	*	CHARACTER PRECEDING THE NEXT CONVERSION SPECIFICATION OR			*
		5022	*	END OF IMAGE, FOLLOWING THE ACTIVITY SPECIFIED BY THE CON-			*
		5023	*	TROL CODE ITSELF. THE CARRIER/CURSOR IS RETURNED TO THE			*
		5024	*	START OF THE NEXT LINE, BUT ALL IMAGE BUFFERS REMAIN			*
		5025	*	LOCKED IN CORE VIRTUAL MEMORY. ALL CONVERSION SPECIFICA-			*
		5026	*	TIONS IN THE IMAGE BECOME AVAILABLE, WITH THE 'NEXT AVAIL-			*
		5027	*	ABLE' SPECIFICATION BEING THE FIRST CONTAINED IN THE IMAGE.			*
		5028	*	* EITHER THE MATRIX PRINTER OF THE CRT (OR BOTH) MAY BE USED FOR			*
		5029	*	OUTPUT, DEPENDING ON THE CURRENT DEFINITION OF THE SYSTEM			*
		5030	*	PRINT DEVICE.			*
		5031	*				*
		5032	*	*ENTRY POINTS			*
		5033	*	THIS ROUTINE HAS A SINGLE ENTRY POINT - FZUPRT - WHOSE FUNCTION			*
		5034	*	IS DEFINED ABOVE. CALLING SEQUENCE IS			*
		5035	*	B I\$CALL			*
		5036	*	DC AL2(VSXSPU)			*
		5037	*	WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL ADDRESS			*
		5038	*	OR ENTRY POINT FZUPRT. EXECUTION IS SUBJECT TO INPUT CONDITIONS			*
		5039	*	DESCRIBED BELOW.			*
		5040	*				*
		5041	*	*INPUT			*
		5042	*	* I\$PARM - 2 BYTES, FOR THE PRINT CONTROL PARAMETER. THIS CON-			*
		5043	*	TAINS A CONTROL CODE, AS SPECIFIED UNDER 'FUNCTION', IN THE			*
		5044	*	RIGHTMOST BYTE.			*
		5045	*	* I\$STAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER. FOR THOSE			*
		5046	*	CONTROL CODES WHICH SPECIFY A STACKED DATA ELEMENT, THIS CON-			*
		5047	*	TAINS THE CORE ADDRESS OF THE LEFTMOST BYTE OR THE STACKED			*
		5048	*	ELEMENT (SEE 'FUNCTION').			*
		5049	*	* RUN-TIME STACK - THIS CONTAINS AN IMAGE SEGMENT, AN UNPACKED			*
		5050	*	FLOATING POINT VALUE, OR A CHARACTER ELEMENT IN THE TOP STACK			*
		5051	*	POSITION FOR APPROPRIATE CONTROL CODES (SEE 'FUNCTION').			*
		5052	*	* I\$SLLC - 1 BYTE, FOR THE LENGTH CODE DEFINING THE LAST STACKED			*

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 298
		5053	*	DATA ELEMENT. WHEN DATA OUTPUT IS SPECIFIED, THIS IS USED TO	*		
		5054	*	DETERMINE THE TYPE OF DATA ITEM (ARITHMETIC OR CHARACTER) CON-	*		
		5055	*	TAINED IN THE TOP STACK POSITION.	*		
		5056	*	* \$PRPOS - 1 BYTE, FOR THE MATRIX PRINTER CARRIER POSITION	*		
		5057	*	INDICATOR. THIS CONTAINS THE CARRIER POSITION, RELATIVE TO THE	*		
		5058	*	HARDWARE LEFT MARGIN AS 0, OF THE MATRIX PRINTER CARRIER.	*		
		5059	*	* \$LMRGN - 1 BYTE, FOR THE MATRIX PRINTER SOFTWARE LEFT MARGIN	*		
		5060	*	INDICATOR.	*		
		5061	*	* \$CRPOS - 1 BYTE, FOR THE CRT CURSOR POSITION INDICATOR. THIS	*		
		5062	*	CONTAINS THE CURSOR POSITION, RELATIVE TO THE LEFT CRT MARGIN	*		
		5063	*	AS 0, OF THE CRT CURSOR.	*		
		5064	*	* \$PRDEV - 2 BYTES, FOR THE SYSTEM PRINT DEVICE INDICATOR.	*		
		5065	*	* \$EXFTR - 1 BYTE, FOR THE SYSTEM CORE EXTENSION FACTOR.	*		
		5066	*		*		
		5067	*	*OUTPUT	*		
		5068	*	* PRINTED OUTPUT - FILLED IMAGES ARE PRINTED, AS SPECIFIED BY THE	*		
		5069	*	CODES DEFINED UNDER 'FUNCTION', ON THE CURRENTLY SPECIFIED	*		
		5070	*	SYSTEM PRINT DEVICE(S).	*		
		5071	*	* IMAGE SAVE BUFFER - THIS PAGE (V\$BFR1) IS LOCKED INTO CORE	*		
		5072	*	VIRTUAL MEMORY AND FILLED WITH IMAGE SEGMENT CONSTRUCTS UNDER	*		
		5073	*	CONTROL OR 'PRU' CODES X'04' AND X'05'.	*		
		5074	*	* IMAGE WORK BUFFER - THIS PAGE (V\$BFR2) IS LOCKED INTO CORE AND	*		
		5075	*	LOADED WITH THE CONTENTS OF THE IMAGE SAVE BUFFER WHENEVER THE	*		
		5076	*	FIRST CONVERSION SPECIFICATION IN THE IMAGE IS TO BE FILLED.	*		
		5077	*	CONVERSION SPECIFICATIONS ARE FILLED SEQUENTIALLY FROM LEFT TO	*		
		5078	*	RIGHT AS REQUIRED (SEE 'FUNCTION').	*		
		5079	*	* \$PRPOS - 1 BYTE, FOR THE MATRIX PRINTER CARRIER POSITION	*		
		5080	*	INDICATOR. THIS IS SET EQUAL TO \$LMRGN WHENEVER AN IMAGE IS	*		
		5081	*	PRINTED AND THE CARRIER RETURNED.	*		
		5082	*	* \$CRPOS - 1 BYTE, FOR THE CRT CURSOR POSITION INDICATOR.	*		
		5083	*	THIS IS SET TO ZERO WHENEVER AN IMAGE IS PRINTED AND CURSOR	*		
		5084	*	RETURNED.	*		
		5085	*		*		
		5086	*	*EXTERNAL REFERENCES	*		
		5087	*	* V\$SPRT - VIRTUAL ENTRY ADDR FOR DFPRNT, V.M. MATRIX PRT IOCS.	*		
		5088	*	* DSPLYN - ENTRY POINT FOR THE SYSTEM CRT IOCS (LABEL DSDLYN IS	*		
		5089	*	REFERENCED INDIRECTLY USING I\$CSXA TO BUILD A CORE ADDRESS.	*		
		5090	*	* I\$CALL - ENTRY POINT FOR PAGING MODULE V.M. PROGRAM CALL RTN.	*		
		5091	*	* I\$RTRN - ENTRY POINT FOR PAGING MODULE V.M. RETURN CONTROL RTN.	*		
		5092	*	* I\$LOCK - ENTRY POINT FOR PAGING MODULE V.M. PAGE LOCKING RTN.	*		
		5093	*	* I\$UNLK - ENTRY POINT FOR PAGING MODULE V.M. PAGE UNLOCKING RTN.	*		
		5094	*	* I\$CSXA - CORE ADDRESS OF 1ST BYTE IN CORE EXTENSION PAST 8K.	*		
		5095	*	* V\$BFR1 - VIRTUAL ADDRESS OF GENERAL V.M. BUFFER 1.	*		
		5096	*	* V\$BFR2 - VIRTUAL ADDRESS OF GENERAL V.M. BUFFER 2.	*		
		5097	*	* I\$PARM - 2 BYTES, FOR THE INTERPRETER COMMUNICATIONS PARAMETER.	*		
		5098	*	* I\$STAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER.	*		
		5099	*	* I\$SLLC - 1 BYTE, FOR LENGTH CODE (L-1) OF LAST STACKED ELEMENT.	*		
		5100	*	* I\$VADR - 2 BYTES, FOR PAGING MODULE VIRTUAL ADDRESS PARAMETER.	*		
		5101	*	* I\$CADR - 2 BYTES, FOR PAGING MODULE CORE ADDRESS OUTPUT PARAM.	*		
		5102	*	* I\$ERRC - 1 BYTE, FOR THE INTERPRETER EXECUTION ERROR CODE.	*		
		5103	*	* \$PRPOS - 1 BYTE, FOR MATRIX PRINTER CARRIER POSITION INDICATOR.	*		
		5104	*	* \$LMRGN - 1 BYTE, FOR POSITION OF SOFTWARE LEFT PRINTER MARGIN.	*		
		5105	*	* \$CRPOS - 1 BYTE, FOR CRT CURSOR POSITION INDICATOR.	*		
		5106	*	* \$PRDEV - 2 BYTES, FOR THE SYSTEM PRINT DEVICE INDICATOR.	*		
		5107	*	* \$EXFTR - 1 BYTE, FOR THE SYSTEM CORE EXTENSION FACTOR.	*		
		5108	*	* I\$INDR - 1 BYTE, FOR CORE-RESIDENT FZUPRT STATUS INDICATORS -	*		

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 299
		5109	*	* I\$NISW (X'80') - NULL IMAGE INDICATOR,	*
		5110	*	* I\$BISW (X'40') - BUFFER 1 (SAVE) INDICATOR,	*
		5111	*	* I\$B2SW (X'20') - BUFFER 2 (WORK) INDICATOR,	*
		5112	*	* ISCSSW (X'10') - CONVERSION SPECIFICATION INDICATOR,	*
		5113	*	* I\$STSW (X'08') - CHARACTER LITERAL (STRING) INDICATOR,	*
		5114	*	* I\$ECSW (X'04') - E-FORMAT CONVERSION INDICATOR,	*
		5115	*	* I\$NDSW (X'02') - END OF IMAGE INDICATOR,	*
		5116	*	* I\$SNSW (X'01') - CONVERSION SPECIFICATION SIGN INDICATOR.	*
		5117	*		*
		5118	*	* I\$PUB1 - 2 BYTES, FOR NE BUFFER 1 CORE ADDRESS.	*
		5119	*	* I\$FUB2 - 2 BYTES, FOR THE BUFFER 2 CORE ADDRESS.	*
		5120	*	* I\$IMLN - 1 BYTE, FOR THE IMAGE LENGTH COUNTER.	*
		5121	*	* I\$IMPT - 2 BYTES, FOR THE IMAGE SCAN POINTER.	*
		5122	*	* I\$IMC1 - 2 BYTES, FOR CORE ADDRESS OF CURR CONV SPEC 1ST BYTE.	*
		5123	*	* I\$SDPT - 2 BYTES, FOR CORE ADDRESS OF CURR CONV SPEC DECIMAL PT.	*
		5124	*	* I\$CSCT - 3 BYTES, FOR CONY SPEC COMPONENT COUNTERS, INCLUDING -	*
		5125	*	* I\$SSCT - 1 BYTE, FOR CONV SPEC CHARACTER COUNT.	*
		5126	*	* I\$SDCT - 1 BYTE, FOR CONV SPEC DIGIT CHARACTER COUNT.	*
		5127	*	* I\$SICT - 1 BYTE, FOR CONY SPEC INTEGER CHARACTER COUNT.	*
		5128	*	* I\$SFCT - 1 BYTE, FOR CONVERSION SPECIFICATION FRACTION CHAR CT.	*
		5129	*	* ISADJX - 1 BYTE, FOR FLOATING POINT EXPONENT SAVE AREA.	*
		5130	*		*
		5131	*	*EXITS, NORMAL	*
		5132	*	CONTROL IS ALWAYS PASSED TO THE PAGING ROUTINE AT ENTRY POINT	*
		5133	*	I\$RTRN (IPGRTN) FOR A RETURN TO THE CALLING PROGRAM.	*
		5134	*		*
		5135	*	*EXITS, ERROR	*
		5136	*	CONTROL IS PASSED TO THE PAGING ROUTINE AT ENTRY POINT I\$RTRN	*
		5137	*	(IPGRTN) WITH PARAMETER I\$ERRC CONTAINING THE APPROPRIATE ERROR	*
		5138	*	MESSAGE CODE.	*
		5139	*		*
		5140	*	*TABLES/WORK AREAS	*
		5141	*	* FZUPRT BRANCH DISPLACEMENT TABLE - USED TO DIRECT OUTPUT	*
		5142	*	OPERATIONS FOR SPECIFIC CONTROL CODES IN I\$WARM -	*
		5143	*	* NUMBER OF TABLE ENTRIES - 8	*
		5144	*	* TABLE ENTRY LENGTH - 1 BYTE	*
		5145	*	* ENTRY FORMAT - SINGLE BYTE DISPLACEMENT WITHIN 1ST FZUPRT	*
		5146	*	VIRTUAL PAGE FOR INTERNAL ENTRY POINT ASSOCIATED WITH EACH	*
		5147	*	CONTROL CODE.	*
		5148	*	* IMAGE SAVE BUFFER - SINGLE VIRTUAL PAGE (V\$BFR1) USED TO CON-	*
		5149	*	TAIN THE IMAGE AS SPECIFIED IN THE BASIC PROGRAM.	*
		5150	*	* IMAGE WORK BUFFER - SINGLE VIRTUAL PAGE (V\$BFR2) USED TO CON-	*
		5151	*	TAIN THE IMAGE MODIFIED WITH CONVERTED DATA ELEMENTS. THIS	*
		5152	*	ALSO ACTS AS THE PRINT OUTPUT BUFFER.	*
		5153	*	* INTERPRETER WORK AREAS - CORE RESIDENT AREAS USED AS SAVE AND	*
		5154	*	COMMUNICATION PARAMETERS BETWEEN FZUPRT PAGES. THESE ARE LOCA-	*
		5155	*	TED IN MODULE INTERP, AND INCLUDE ALL AREAS DEFINED UNDER	*
		5156	*	'EXTERNAL SPECIFICATIONS' BEGINNING WITH LABEL I\$INDR.	*
		5157	*		*
		5158	*	*ATTRIBUTES	*
		5159	*	* REUSABLE	*
		5160	*	* NATURALLY RELOCATABLE	*
		5161	*		*
		5162	*	*CHARACTER CODE DEPENDENCY	*
		5163	*	THE OPERATION OF THIS MODULE DEPENDS UPON THE FOLLOWING PROPER-	*
		5164	*	TIES OF THE INTERNAL REPRESENTATION OF THE EXTERNAL CHAR. SET.	*

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 300
		5165	*	* HOST CODING HAS BEEN ARRANGED SO THAT REDEFINITION OF CHAR-			*
		5166	*	ACTER CONSTANTS, BY REASSEMBLY, WILL RESULT IN A CURRENT			*
		5167	*	MODULE FOR THE NEW DEFINITION.			*
		5168	*	* NUMERIC CHARACTERS 0 THROUGH 9 ARE PRESUMED TO BE CODED			*
		5169	*	SUCH THAT THE HIGH ORDER FOUR BITS CONTAIN A SIGN ZONE WITH			*
		5170	*	X'F' DEFINING A POSITIVE DIGIT.			*
		5171	*	THE SPECIFIC INSTRUCTIONS (INSTRUCTION SEQUENCES) WHICH REQUIRE			*
		5172	*	MODIFICATION IF THESE PROPERTIES OF THE CHARACTER SET ARE CHANGED			*
		5173	*	MAY BE IDENTIFIED BY -			*
		5174	*	* THE 2 INSTRUCTIONS BEGINNING AT LABEL FZU735.			*
		5175	*				*
		5176	*	*NOTES			*
		5177	*	ERROR PROCEDURES			*
		5178	*	* ERROR 1 - A DATA ELEMENT IS TO BE PRINTED, AND THE REFER-			*
		5179	*	ENCED IMAGE DOES NOT CONTAIN AT LEAST ONE CONVERSION SPECI-			*
		5180	*	FICATION. AN ERROR CODE FOR THE MESSAGE 'NO CONVERSION			*
		5181	*	SPECIFICATION IN IMAGE' IS ESTABLISHED IN PARAMETER I\$ERRC,			*
		5182	*	AND CONTROL IS RETURNED TO THE CALLING PROGRAM.			*
		5183	*	* FZUPRT OTHERWISE UTILIZES OUTPUT IOCS ROUTINES DFPRNT			*
		5184	*	(MATRIX PRINTER) AND DSPLYN (CRT), AND IS SUBJECT TO THE			*
		5185	*	ERP'S INHERENT IN THESE PROGRAMS.			*
		5186	*				*
		5187	*	REGISTER USAGE			*
		5188	*	* RESISTER @BR IS TO CONTAIN THE CORE PAGE BASE ADDRESS			*
		5189	*	ESTABLISHED THROUGH PAGING MODULE CONTROL FOR THE PAGE WHICH			*
		5190	*	INCLUDES FZUPRT, AND IS RESTORED THROUGH THE PAGING MODULE.			*
		5191	*	* REGISTFR @XR IS NOT SAVED. IT IS USED IN FZUPRT FOR GENERAL			*
		5192	*	PURPOSE INDEXING OPERATIONS.			*
		5193	*				*
		5194	*	SAVED/RESTORED AREAS			*
		5195	*	NONE			*
		5196	*				*
		5197	*	MODIFICATION CONSIDERATIONS			*
		5198	*	NONE			*
		5199	*				*
		5200	*	REQUIRED MODULES			*
		5201	*	* @SYSEQ - COMMON SYSTEM EQUATES.			*
		5202	*	* @FXDEQ - FIXED ADDRESSES FOR SYSTEM NUCLEUS.			*
		5203	*	* @ERMEQ - SYSTEM ERROR MESSAGE EQUATES.			*
		5204	*	* \$V\$EQU - VIRTUAL VIRTUAL FIXED ADDRESSES.			*
		5205	*	* \$B@EQU - BASIC COMPILER EQUATES.			*
		5206	*	* \$I\$EQU - INTERPRETER FIXED LOCATION ADDRESS EQUATES.			*
		5207	*	* \$I@SEQ - INTERPRETER PARAMETER EQUATES (FOR STD. PREC. ONLY)			*
		5208	*	* \$I@LEQ - INTERPRETER DARAMETER EQUATES (FOR LONG PREC. ONLY)			*
		5209	*				*
		5210	*	OTHER			*
		5211	*	NONE			*
		5212	*	*****			*

FZUPRT - S/3 BASIC INTERPRETER PRINT USING EXEC RTN

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 301
		5214		*****	
		5215	*	START OF PRINT USING EXECUTION MODULE	*
		5216		*****	
		5217	*		
		5218	*	ESTABLISH ADDRESSABILITY FOR INTERFACE 1ST VM PAGE	
		5219	*		
3800		5220		ORG *,B@LVP,0	BEGIN AT PAGE BOUNDARY
		3800 5221		USING *,@BR	SET BASE ADDRESS AT PAGE BOUND
		5222	*		
		5223	*	ENTER FZUPRT - BRANCH TO ROUTINE DEFINED BY CONTROL CODE CONTAINED	
		5224	*	IN COMMON PARAMETER ISPARM (OPERAND OF THE 'PRU' PSEUDO INSTRUCTION)	
		5225	*		
		3800 5226	FZUPRT EQU	*	FZUPRT ENTRY POINT
3800 D2 02 F7		5227	LA	FZUCAT(,@BR),@XR	LOAD BRANCH ADDRESS TABLE BASE
3803 4C 00 0E 0D57		5228	MVC	FZU010+@DD2(,@BR),I\$PARM(1)	STORE CONTROL CODE IN MOVE
3808 7B 30 0E		5229	SBF	FZU010+@DD2(,@BR),B@PUTM+B@PURE	* INST AND SUPPRESS INDRS
380B 6C 00 11 00		5230	FZU010 MVC	FZU020+@D1(,@BR),*-(1,@XR)	MOVE TBL ENTRY TO BRANCH INST
380F D0 87 00		5231	FZU020 B	*-(,@BR)	BRANCH TO EXECUTE SPECIFIED RTN
		5232	*		
		5233		*****	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 302
		5235		*****	
		5236	*	CONTROL CODE 1 - NULL IMAGE STATEMENT PROCESSING	*
		5237		*****	
		5238	*		
		5239	*	SET THE NULL IMAGE INDICATOR AND RETURN TO INTERPRETER	
		5240	*		
3812	3C 80 0DC5	5241	FZU030 MVI	I\$INDR,I\$NISW	SET NULL IMAGE INDICATOR ON
		5242	*		* AND CLEAR REMAINING INDRS
3816	D0 87 EF	5243	B	FZU190(,@BR)	GO RETURN TO INTERPRETER
		5245		*****	
		5246	*	CONTROL CODE 4 - INITIALIZE FOR ASSEMBLY OF IMAGE STRING SEGMENTS	*
		5247		*****	
		5248	*		
		5249	*	ESTABLISH THE IMAGE SAVE BUFFER IN CORE VIRTUAL MEMORY	
		5250	*		
3819	3C 40 0DC5	5251	FZU040 MVI	I\$INDR,I\$B1SW	SET BUFFER 1 INDICATOR ON
		5252	*		* AND CLEAR REMAINING INDRS
381D	1C 01 144A F4	5253	MVC	I\$VADR,FZUAB1(@VADDR,@BR)	SET PAGING RTN VADDR PARAMETER
3822	C0 87 1354	5254	B	I\$LOCK	LINK TO GET AND LOCK BFR-1 PAGE
		5255	*		
3826	0C 01 0DC8 144C	5256	MVC	I\$PUB1,I\$CADR(@CADDR)	SAVE BUFFER 1 CORE ADDRESS
382C	3C 00 0DC6	5257	MVI	I\$IMLN,@ZERO	SET IMAGE LENGTH EQUAL ZERO
		5258	*		
		5259		*****	

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15,	MOD 00	31/05/21	PAGE 303
					5261	*****	*****				
					5262	*	CONTROL CODE 5 - ESTABLISH IMAGE STRING SEGMENT IN SAVE BUFFER				*
					5263	*****	*****				
					5264	*					
					5265	*	ACCESS THE IMAGE STRING SEGMENT IN RUN-TIME STACK				
					5266	*					
3830	35	02	0D4E		5267	FZU050 L	I\$STAK,@XR			LOAD THE STACK PIONTER	
3834	BB	E0	00		5268		SBF I@STAT(,@XR),X'FF'-B@CCNT			MAKE \$STATUS BYTE A CHAR COUNT	
					5269	*					
					5270	*	SET INSTRUCTIONS TO MOVE CURRENT SEGMENT TO BUFFER				
					5271	*					
3837	7C	11	4F		5272	FZU055 MVI	FZU065+@D1(,@BR),I@LCRF-1			ESTABLISK DISP TO RIGHT BYTE	
383A	4E	00	4F 0DC6		5273		ALC FZU065+@D1(,@BR),I\$IMLN(1)			* OF SEGMENT AREA IN BUFFER	
383F	2E	00	0DC6 00		5274		ALC I\$IMLN,I@STAT(1,@XR)			INCR IMAGE LNG BY SEGMENT LNG	
3844	4C	00	53 0DC6		5275		MVC FZU070+@D1(,@BR),I\$IMLN(1)			SET DISP TO BYTE AFTER SEGMENT	
					5276	*					
					5277	*	MOVE CORRENT SEGMENT TO BUFFER AND FOLLOW WITH 'EOS' CHARACTER.				
					5278	*					
3849	35	01	0DC8		5279	FZU060 L	I\$PUB1,@BR			LOAD BUFFER 1 CORE ADDRESS	
384D	6C	11	00 12		5280	FZU065 MVC	*-*(,@BR),I@LCRV-1(I@LCRF,@XR)			MOVE SEGMENT TO BUFFER	
3851	7C	1E	00		5281	FZU070 MVI	*-*(,@BR),@EOS			INSERT 'EOS' AFTER SEGMENT	
					5282	*					
3854	F2	87	98		5283		J FZU190			GO RETURN TO INTERPRETER	
					5284	*					
					5285	*****	*****				

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 304
			5287	*****	
			5288	* CONTROL CODE 2 - PROCESS A NULL PRINT USING LIST	*
			5289	*****	
			5290	*	
			5291	* PROCESS NULL LIST AND TERMINATE PRINT USING STATEMENT EXECUTION	
			5292	*	
3857	C0 87 12B1		5293	FZU080 B I\$CALL	LINK TO PROCESS NULL DATA LIST
385B	3900	385C	5294	DC AL(@VADDR)(FZU200)	NULL LIST ROUTINE VIRTUAL ADDR
			5295	*	
385D	D0 87 DF		5296	B FZU180(,@BR)	GO UNLOCK BUFFER 1 AND EXIT
			5297	*	
			5298	*****	

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 305
				5300			*****			
				5301			* CONTROL CODES 3,6 - PROCESS A PRIMARY DATA ELEMENT OR NULL CONSTANT *			
				5302			*****			
				5303			*			
				5304			* TEST FOR PREVIOUS ESTABLISHMENT OF THE IMAGE WORK BUFFER			
				5305			*			
3860	38	20	0DC5	5306	FZU090	TBN	I\$INDR,I\$B2SW			IF WORK BUFFER IS ESTABLISHED
3864	F2	10	47	5307		JT	FZU140			* GO PROCESS THE DATA ELEMFNT
				5308			*			
				5309			* WORK BUFFER NOT ESTABLISHED - CHECK FOR A NULL IMAGE WITH FINITE			
				5310			* PRINT USING LIST (ERROR 'NO CONVERSION SPECIFICATIONS IN IMAGE')			
				5311			*			
3867	38	80	0DC5	5312	FZU100	TBN	I\$INDR,I\$NISW			IF CURRENT IMAGE IS NOT NULL
386B	F2	90	07	5313		JF	FZU110			* SKIP TO ESTABLISH WORK BUFFER
386E	3C	C1	0CBC	5314		MVI	I\$ERRC,@E724			* ELSE SET ERROR MESSAGE CODE
3872	D0	87	EF	5315		B	FZU190(,@BR)			* AND RETURN TO INTERPRETER
				5316			*			
				5317			* IMAGE NOTR NULL - ESTABLISH THE IMAGE WORK BUFFER IN CORE VM			
				5318			*			
3875	3A	20	0DC5	5319	FZU110	SBN	I\$INDR,I\$B2SW			SET BUFFER 2 INDICATOR ON
				5320			*			
3879	1C	01	144A F6	5321		MVC	I\$VADR,FZUAB2(@VADDR,@BR)			SET PAGING RTN VADDR PARAMETER
387E	C0	87	1354	5322		B	I\$LOCK			LINK TO GET AND LOCK BFR-2 PAGE
				5323			*			
3882	0C	01	0DCA 144C	5324		MVC	I\$PUB2,I\$CADR(@CADDR)			SAVE BUFFER 2 CORE ADDRESS
				5325			*			
				5326			* MOVE SAVED IMAGE IN BUFFER 1 TO BUFFER 2 FOR CONVERSION SPECIFICATION			
				5327			* PROCESSING AND PRINTER OUTPUT.			
				5328			*			
3888	3B	02	0DC5	5329	FZU120	SBF	I\$INDR,I\$NDSW			SET END-OF-IMAGE INDICATOR OFF
388C	0C	01	0DCC 0DCA	5330		MVC	I\$IMPT,I\$PUB2(@CADDR)			SET IMAGE PT TO 1ST BUFFER BYTE
				5331			*			
3892	4C	00	A9 0DC6	5332		MVC	FZU130+@DD2(,@BR),I\$IMLN(1)			SET INST TO MOVE BUFFER 1 TO
3897	5C	01	A8 A9	5333		MVC	FZU130+@DD2-1(,@BR),FZU130+@DD2(FZULQD,@BR)			* BUFFER 2
389B	74	01	AD	5334		ST	FZU135+@OP1(,@BR),@BR			SAVE CURRENT BASE REGISTER
389E	35	01	0DC8	5335		L	I\$PUB1,@BR			LOAD BUFFER 1 CORE ADDRESS
38A2	35	02	0DCA	5336		L	I\$PUB2,@XR			LOAD BUFFER 2 CORE ADDRESS
38A6	9C	00	00 00	5337	FZU130	MVC	*-(,@XR),*-(@Q,@BR)			MOVE BUFFER 1 TO BUFFER 2
38AA	C2	01	0000	5338	FZU135	LA	*-*,@BR			RESTORE THE BASE REGISTER
				5339			*			
				5340			* ATTEMPT TO PROCESS THE DATA ELEMENT ACCORDING TO NEXT CONVERSION SPEC			
				5341			*			
38AE	C0	87	12B1	5342	FZU140	B	I\$CALL			LINK TO PROCESS DATA ELEMENT
38B2	390D			5343		DC	AL(@VADDR)(FZU210)			PRIMARY DATA RTN VIRTUAL ADDR
				5344			*			
				5345			* TEST FOR PROCESSED ELEMENT - WHEN END-OF-IMAGE WAS BEEN ENCOUNTERED.			
				5346			* FILLED IMAGE WAS BEEN PRINTED AND CURRENT ELEMENT HAS NOT YET BEEN			
				5347			* PLACED IN A CONVERSION SPECIFICATION.			
				5348			*			
38B4	38	02	0DC5	5349	FZU145	TBN	I\$INDR,I\$NDSW			IF END-OF-IMAGE INDICATOR ON
38B8	D0	10	88	5350		BT	FZU120(,@BR)			* GO RESTORE IMAGE AND PROCESS
				5351			*			* CURRENT DATA ELEMENT
38BB	D0	87	C4	5352		B	FZU160(,@BR)			GO RETURN TO INTERPRETER
				5353			*			
				5354			*****			

[illegible]

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 307
				5367			*****			
				5368	*		EXIT ROUTINE - CHECK BUFFER RETENTION AND RETURN TO INTERPRETER			*
				5369			*****			
				5370	*					
				5371	*		TEST FOR TERMINATION OF THE PRINT USING DATA LIST			
				5372	*					
38C4	38	10	0D57	5373	FZU160	TBN	I\$PARM,B@PUTM			IF DATA LIST TERMINATION SW OFF
38C8	39	20	0D57	5374		TBF	I\$PARM,B@PURE			* OR END-MATRIX-ROW INDR IS ON,
38CC	F2	90	20	5375		JF	FZU190			* BRANCH FOR RETURN TO INTERP
				5376	*					
				5377	*		DATA LIST TERMINATED - RELEASE BUFFER 2 FROM CORE VIRTUAL MEMORY			
				5378	*					
38CF	38	20	0DC5	5379	FZU170	TBN	I\$INDR,I\$B2SW			IF BUFFER 2 NOT LOCKED IN CORE
38D3	F2	90	09	5380		JF	FZU180			* SKIP TO RELEASE BUFFER 1
				5381	*					
38D6	1C	01	144A F6	5382		MVC	I\$VADR,FZUAB2(@VADDR,@BR)			SET PAGING RTN VADDR PARAMETER
38DB	C0	87	1350	5383		B	I\$UNLK			LINK TO RELEASE BUFFER 2
				5384	*					
				5385	*		DATA LIST TERMINATED - RELEASE BUFFER 1 FROM CORE VIRTUAL MEMORY			
				5386	*					
38DF	38	40	0DC5	5387	FZU180	TBN	I\$INDR,I\$B1SW			IF BUFFER 1 NOT LOCKED IN CORE
38E3	F2	90	09	5388		JF	FZU190			* SKIP TO RETURN TO INTERPRETER
				5389	*					
38E6	1C	01	144A F4	5390		MVC	I\$VADR,FZUAB1(@VADDR,@BR)			SET PAGING RTN VADDR PARAMETER
38EB	C0	87	1350	5391		B	I\$UNLK			LINK TO RELEASE BUFFER 1
				5392	*					
				5393	*		RETURN TO INTERPRETER CALLING ROUTINE			
				5394	*					
38EF	C0	87	12D3	5395	FZU190	B	I\$RTRN			RETURN TO THE INTERPRETER
				5396	*					
				5397			*****			

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 308
					5399		*****			
					5400	*	PRINT USING INTERFACE CONSTANTS (1ST VM PAGE)			*
					5401		*****			
					5402	*				
38F3	5400			38F4	5403	FZUAB1 DC	AL(@VADDR)(V\$BFR1) IMAGE BUFFER 1 VIRTUAL ADDRESS			
38F5	5500			38F6	5404	FZUAB2 DC	AL(@VADDR)(V\$BFR2) IMAGE BUFFER 2 VIRTUAL ADDRESS			
					5406		*****			
					5407	*	PRINT USING CONTROL PARAMETER FUNCTION ADDRESS TABLE			*
					5408		*****			
					5409	*				
				38F7	5410	FZUCAT EQU	* CONTROL ADDR TABLE ADDRESS			
38F7	CF			38F7	5411	DC	AL1(FZU170-FZUPRT) CODE 0 - RELEASE IMAGE			
38F8	12			38F8	5412	DC	AL1(FZU030-FZUPRT) CODE 1 - NULL IMAGE			
38F9	57			38F9	5413	DC	AL1(FZU080-FZUPRT) CODE 2 - NULL PRINT LIST			
38FA	60			38FA	5414	DC	AL1(FZU090-FZUPRT) CODE 3 - NULL CHAR CONSTANT			
					5415	*				
38FB	19			38FB	5416	DC	AL1(FZU040-FZUPRT) CODE 4 - 1ST IMAGE SEGMENT			
38FC	30			38FC	5417	DC	AL1(FZU050-FZUPRT) CODE 5 - SECONDARY IMASE SEG			
38FD	60			38FD	5418	DC	AL1(FZU090-FZUPRT) CODE 6 - PRIMARY DATA ELEMENT			
38FE	BE			38FE	5419	DC	AL1(FZU150-FZUPRT) CODE 7 - SECNDARF DATA ELEMENT			
					5420	*				
					5421		*****			

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 309
		5423		*****			
		5424	*	VIRTUAL MEMORY PRINT USING INTERFACE ROUTINE 2ND VM GAGE -			*
		5425	*	PERFORMS FOLLOWING ACTIVITIES DEPENDING ON ENTRY POINT -			*
		5426	*				*
		5427	*	CONTROL CODE 2 (NULL PRINT LIST) ENTRY -			*
		5428	*	* RETURNS CARRIER IF NOT INITIALLY AT LEFT MARGIN			*
		5429	*	* IF IMAGE IS NULL -			*
		5430	*	* RETURNS CARRIER ONLY			*
		5431	*	* IF IMAGE IS NOT NULL -			*
		5432	*	* PRINTS IMAGE TO 1ST CONVERSION SPECIFICATION OR 'EOS'			*
		5433	*	* RETURNS CARRIER AFTER PRINTING IMAGE			*
		5434	*				*
		5435	*	CONTROL CODES 3,6 (PRIMARY DATA ELEMENT) ENTRY -			*
		5436	*	* SCANS IMAGE TO NEXT AVAILABLE CONVERSION SPECIFICATION OR 'EOS'			*
		5437	*	* IF CONVERSION SPECIFICATION ENCOUNTERED -			*
		5438	*	* CONVERTS DATA ELEMENT AND INSERTS IN CONLERSION SPEC			*
		5439	*	* IF ELEMENT IS FINAL LIST ELEMENT -			*
		5440	*	* RETURNS CARRIER IF NOT AT LEFT MARGIN			*
		5441	*	* PRINTS IMAGE TO NEXT CONVERSION SPEC OR 'EOS'			*
		5442	*	* RETURNS CARRIER AFTER PRINTING NAGE			*
		5443	*	* IF 'EOS' IS ENCOUNTERED -			*
		5444	*	* RETURNS CARRIER IF NOT AT LEFT MARGIN			*
		5445	*	* PRINTS ENTIRE IMAGE UP TO THE 'EOS'			*
		5446	*	* RETURNS CARRIER AFTER PRINTING IMAGE			*
		5447	*				*
		5448	*	CONTROL CODE 7 (SECONDARY CHARACTER CONSTANT SEGMENT) ENTRY -			*
		5449	*	* CONVERTS SECONDARY ELEMENT AND ADDS TO CURRENT CONVERSION SPEC			*
		5450	*	* IF SEGMENT IS FINAL LIST ELEMENT -			*
		5451	*	* RETURNS CARRIER IF NOT AT LEFT MARGIN			*
		5452	*	* PAINTS IMAGE TC NEY? CONVERSION SPECIFICATION OR 'EOS'			*
		5453	*	* RETURNS CARRIER AFTER PRINTING IMAGE			*
		5454		*****			
		5455	*				
		5456	*	ESTABLISH ADDRESSABILIFY FOR INTERFACE 2ND VM PAGE			
		5457	*				
3900		5458		ORG *,B@LVPG,0			BEGIN AT PAGE BOUNDARY
	3900	5459		USING *,@BR			SET BASE ADDRESS AT PAGE BOUND
		5460	*				
		5461		*****			

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 310
		5463		*****	
		5464	*	ENTRY FOR CONTROL CODE 2 - NULL PRINT USING LIST	*
		5465		*****	
		5466	*		
		5467	*	UNFILLED IMAGE IS NORMALLY IN IMAGE BUFFER 1 - SET SCAN AND PRINT	
		5468	*	PARAMETERS TO PRINT IMAGE TO 1ST CONVERSION SPECIFICATION OR 'EOS'	
		5469	*		
3900	4C 01 F8 0DC8	5470	FZU200 MVC	FZUPAD(,@BR),I\$PUB1(@CADDR) SET PRINT RTN PARAM FOR BFR-1	
3905	1C 01 0DCC F8	5471	MVC	I\$IMPT,FZUPAD(@CADDR,@BR) SET IMAGE PT TO 1ST BFR-1 BYTE	
		5472	*		
390A	D0 87 46	5473	B	FZU280(,@BR) GO PRINT THE BUFFER 1 IMAGE OR	
		5474	*	* RETURN CARRIER FOR NULL IMAGE	
		5476		*****	
		5477	*	ENTRY FOR CONTROL CODES 3,6 - PRIMARY ELEMENT OR NULL CHAR CONSTANT	*
		5478		*****	
		5479	*		
		5480	*	IN GENERAL, PARTIALLY FILLED IMAGE IS IN BUFFER 2 - SCAN TO THE	
		5481	*	NEXT CONVERSION SPECIFICATION OR END-OF-IMAGE CHARACTER	
		5482	*		
390D	D0 87 76	5483	FZU210 B	FZU330(,@BR) LINK TO SCAN IMAGE TO NEXT C/S	
3910	BD 1E 00	5484	CLI	B@CHAR(,@XR),@EOS IF END-OF-IMAGE NOT ENCOUNTERED	
3913	F2 01 15	5485	JNE	FZU240 * GO CONVERT ELEMENT TO IMAGE	
		5486	*		
		5487	*	END-OF-IMAGE ENCOUNTERED - TEST FOR PRESENCE OF AT LEAST ONE	
		5488	*	CONVERSION SPECIFICATION IN IMAGE - AN ERROR CONDITION (NO CONVER-	
		5489	*	SION SPECIFICATION IN IMAGE) OCCURS WHEN THIS IS NOT TRUE.	
		5490	*		
3916	38 10 0DC5	5491	FZU220 TBN	I\$INDR,I\$CSSW IF CONV SPEC SWITCH IS ON	
391A	F2 10 07	5492	JT	FZU230 * SKIP TO INDICATE END OF IMAGE	
391D	3C C1 0CBC	5493	MVI	I\$ERRC,@E724 * ELSE SET ERROR MESSAGE CODE	
3921	D0 87 72	5494	B	FZU320(,@BR) * AND RETURN TO CALLING PAGE	
		5495	*		
		5496	*	NO IMAGE ERROR - SET END-OF-IMAGE INDICATOR AND BRANCH TO PRINT IMAGE	
		5497	*		
3924	3A 02 0DC5	5498	FZU230 SBN	I\$INDR,I\$NDSW SET END-OF-IMAGE INDICATOR ON	
3928	D0 87 41	5499	B	FZU270(,@BR) GO PRINT THE BUFFER 2 IMAGE	
		5500	*		
		5501	*	END-OF-IMAGE NOT ENCOUNTERED - CONVERT ELEMENT AND INSERT IN IMAGE	
		5502	*		
392B	C0 87 12B1	5503	FZU240 B	I\$CALL LINK TO CONV AND INSERT ELEMENT	
392F	3A00	3930 5504	DC	AL(@VADDR)(FZU500) PRIMARY DATA RTN VIRTUAL ADDR	
		5505	*		
3931	D0 87 3A	5506	B	FZU260(,@BR) GO TEST FOR END OF PRINT LIST	
		5507	*		
		5508		*****	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 311
		5510		*****	
		5511	*	ENTRY FOR CONTROL CODE 7 - SECONDARY CHARACTER CONSTANT SEGMENT	*
		5512		*****	
		5513	*		
		5514	*	IN GENERAL, CURRENT CONVERSION SPECIFICATION IS PARTIALLY FILLED -	
		5515	*	ADD SECONDARY CHARACTER CONSTANT SEGMENT TO THIS SPECIFICATION.	
		5516	*		
3934	C0 87 12B1	5517	FZU250 B	I\$CALL	LINK TO INSERT SEGMENT IN IMAGE
3938	3AAA	3939 5518	DC	AL(@VADDR)(FZU660)	SECONDARY DATA RTN VIRTUAL ADDR
		5519	*		
		5520	*	TEST FOR TERMINATION OF THE PRINT USING DATA LIST	
		5521	*		
393A	38 10 0D57	5522	FZU260 TBN	I\$PARM,B@PUTM	IF CONTROL CODE TERMINATION SW
393E	F2 90 31	5523	JF	FZU320	* IS OFF, RETURN TO CALLING PGM
		5524	*		
		5525	*	SET PRINT ROUTINE PARAMETER TO OUTPUT BUFFER 2 MARGIN	
		5526	*		
3941	4C 01 F8 0DCA	5527	FZU270 MVC	FZUPAD(,@BR),I\$PUB2(@CADDR)	SET PRINT RTN PARAM FOR BFR-2
		5528	*		
		5529	*	RETURN PRINTER CARRIER IF NOT ALREADY AT LEFT MARGIN	
		5530	*		
3946	D2 02 F9	5531	FZU280 LA	FZURPL(,@BR),@XR	LOAD CARR RETURN PPL CORE ADDR
3949	0D 00 03C2 03C1	5532	CLC	\$PRPOS,\$LMRGN(1)	IF CARRIER NOT AT LEFT MARGIN
394F	D0 01 C4	5533	BNE	FZU400(,@BR)	* LINK TO RETURN THE CARRIER
3952	3D 00 03E2	5534	CLI	\$CRPOS,@ZERO	IF CURSOR NOT AT LEFT MARGIN
3956	D0 01 C4	5535	BNE	FZU400(,@BR)	* LINK TO RETURN THE CURSOR
		5536	*		
		5537	*	TEST FOR A VALID NULL IMAGE - IN THIS EVENT, A SIMPLE CARRIER RETURN	
		5538	*	IS TO BE PERFORMED.	
		5539	*		
3959	38 80 0DC5	5540	FZU290 TBN	I\$INDR,I\$NISW	IF NULL IMAGE INDICATOR IS ON
395D	F2 10 0F	5541	JT	FZU310	* SKIP TO PERFORM CHAR RETURN
		5542	*		
		5543	*	IMAGE NOT NULL - ESTABLISH PARAMETERS TO PRINT THE BUFFER IMAGE	
		5544	*		
3960	38 02 0DC5	5545	FZU300 TBN	I\$INDR,I\$NDSW	IF END-OF-IMAGE INDICATOR OFF
3964	D0 90 76	5546	BF	FZU330(,@BR)	* LINK TO SCAN IMAGE TO NEXT
		5547	*		* CONVERSION SPECIFICATION
3967	4C 00 F6 0DCC	5548	MVC	FZUPCT(,@BR),I\$IMPT(1)	SET PRINT RTN COUNT PARAMETER
396C	D2 02 F5	5549	LA	FZUPPL(,@BR),@XR	LOAD PRINT A CR PPL CORE ADDR
		5550	*		
		5551	*	PRINT THE IMAGE (NORMAL CASE) OR RETURN CARRIER ONLY (NULL IMAGE)	
		5552	*		
396F	D0 87 C4	5553	FZU310 B	FZU400(,@BR)	LINK TO PRINT & RETURN CARRIER
		5554	*		* OR RETURN CARRIER ONLY
		5555	*		
		5556	*	EXIT 2ND VM PAGE - RETURN TO CALLING PAGE	
		5557	*		
3972	C0 87 12D3	5558	FZU320 B	I\$RTRN	RETURN TO CALLING PAGE
		5559	*		
		5560		*****	

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 312
				5562		*****				
				5563	*	GENERAL IMAGE SCAN ROUTINE -				*
				5564	*	* DETERMINES CORE ADDRESS OF NEXT AVAILABLE CONVERSION				*
				5565	*	SPECIFICATION OR END-OF-IMAGE CHARACTER				*
				5566	*					*
				5567	*	INPUT -				*
				5568	*	* F\$1MPT - 2 BYTES1 CONTAINS CORE ADDRESS OF BEGINNING SCAN				*
				5569	*	LOCATION WITHIN THE IMAGE				*
				5570	*					*
				5571	*	OUTPUT -				*
				5572	*	* I\$IMPT - 2 BYTES, CONTAINS CORE ADDRESS OF LEFT BYTE OF NEXT				*
				5573	*	ENCOUNTERED CONVERSION SPECIFICATION OR 'EOS' CHARACTER				*
				5574		*****				
				5575	*					
				5576	*	SUBROUTINE ENTRY - SAVE RETURN ADDRESS AND SET REGISTER FOR SCAN				
				5577	*					
3976	74	08	C3	5578	FZU330	ST	FZU390+@OP1(,@BR),@ARR			STORE RETURN BRANCH ADDRESS
3979	35	02	0DCC	5579		L	I\$IMPT,@XR			LOAD BEGINNING SCAN CORE ADDR
				5580	*					
				5581	*	TEST CURRENT CHARACTER OR SERIES OF CHARACTERS FOR START OF A				
				5582	*	CONVERSION SPECIFICATION OR 'EOS' CHARACTER				
				5583	*					
397D	BD	1E	00	5584	FZU340	CLI	B@CHAR(,@XR),@EOS			IF CURRENT CHAR IS END-OF-IMAGE
3980	F2	81	39	5585		JE	FZU380			* GO TERMINATE IMAGE SCAN
3983	BD	7B	00	5586		CLI	B@CHAR(,@XR),B@DIGS			IF CURRENT CHAR IS DIGIT SPEC
3986	F2	81	33	5587		JE	FZU380			* GO TERMINATE IMAGE SCAN
3989	BD	4E	00	5588		CLI	B@CHAR(,@XR),B@PLUS			IF CURRENT CHAR IS A PLUS SIGN
398C	F2	81	06	5589		JE	FZU350			* GO TEST NEXT !WU CHARACTER
398F	BD	60	00	5590		CLI	B@CHAR(,@XR),B@MINS			IF CURRENT CHAR NOT MINUS SIGN
3992	F2	01	15	5591		JNE	FZU360			* GO TEST FOR A DECIMAL POINT
				5592	*					
3995	BD	7B	01	5593	FZU350	CLI	B@CHAR+1(,@XR),B@DIGS			IF 2ND CHARACTER IS DIGIT SPEC
3998	F2	81	21	5594		JE	FZU380			IT GO TERMINATE IMAGE SCAN
399B	BD	4B	01	5595		CLI	B@CHAR+1(,@XR),B@DPNT			IF 2ND CHAR NOT DECIMAL POINT
399E	F2	01	15	5596		JNE	FZU370			* GO INCREMENT IMAGE POINTER
39A1	BD	7B	02	5597		CLI	B@CHAR+2(,@XR),B@DIGS			IF 3RD CHARACTER IS DIGIT SPEC
39A4	F2	81	15	5598		JE	FZU380			* GO TERMINATE IMAGE SCAN
39A7	F2	87	0C	5599		J	FZU370			* ELSE GO INCR IMAGE POINTER
39AA	BD	4B	00	5601	FZU360	CLI	B@CHAR(,@XR),B@DPNT			IF CURRENT CHAR NOT DECIMAL PT
39AD	F2	01	06	5602		JNE	FZU370			* GO INCREMENT IMAGE POINTER
39B0	BD	7B	01	5603		CLI	B@CHAR+1(,@XR),B@DIGS			IF 2ND CHARACTER IS DIGIT SPEC
39B3	F2	81	06	5604		JE	FZU380			* GO TERMINATE IMAGE SCAN
				5605	*					
				5606	*	CURRENT IMAGE CHARACTER DOES NOT START A CONVERSION SPECIFICATION -				
				5607	*	INCREMENT THE IMAGE POINTER AND REPEAT THE CHARACTER TESTS				
				5608	*					
39B6	E2	02	01	5609	FZU370	LA	@B1(,@XR),@XR			INCREMENT THE IMAGE POINTER
39B9	D0	87	7D	5610		B	FZU340(,@BR)			GO REPEAT CHARACTER TESTS
				5611	*					
				5612	*	CURRENT IMAGE CHARACTER IS START OF A CONVERSION SPECIFICATION				
				5613	*	OR 'EOS' - SAVE IMAGE POINTER AND RETURN TO CALLER				
				5614	*					
39BC	34	02	0DCC	5615	FZU380	ST	I\$IMPT,@XR			STORE CORE ADDR OF C/S OR 'EOS'
39C0	C0	87	0000	5616	FZU390	B	*-*			RETURN TO CALLING ROUTINE
				5617	*					

[illegible]

FZUPRT - S/3 BASIC INTERPRETER PRINT USING EXEC RTN

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 314
		5620		*****	
		5621	*	PRINTER/CRT IMAGE OUTPUT ROUTINE	*
		5622	*	* PRINTS SPECIFIED IMAGE AND/OR CONTROLS CARRIER FOR MATRIX	*
		5623	*	PRINTER AND/OR CRT AS DEFINED BY SYSTEM DEVICE PARAMETER	*
		5624	*		*
		5625	*	INPUT -	*
		5626	*	* REGISTER @XR - CONTAINS CORE ADDRESS OF PRINT PARAMETER LIST	*
		5627	*		*
		5628	*	OUTPUT_	*
		5629	*	* PRINTED LINE AND/OR CARRIER CONTROL AS SPECIFIED IN PARAM LIST	*
		5630		*****	
		5631	*		
		5632	*	SUBROUTINE ENTRY - SAVE THE RETURN ADDRESS	
		5633	*		
39C4	74 08 F2	5634	FZU400 ST	FZU480+@OP1(,@BR),@ARR STORE RETURN BRANCH ADDRESS	
		5635	*		
		5636	*	DETERMINE POSSIBLE CORE ENTRY ADDRESS FOR THE CRT IOCR	
		5637	*		
39C7	5C 01 DE F4	5638	FZU410 MVC	FZU440+@OP1(,@BR),FZUPDA(@CADDR,@BR) SET UP BASE CADDR	
39CB	4E 00 DD 043B	5639	ALC	FZU440+@OP1-1(,@BR),\$EXFTR(1) * AND ADD EXTENSION FACTOR	
		5640	*		
		5641	*	TEST FOR TYPE OF PRINT DEVICE ACTIVE ON SYSTEM	
		5642	*		
39D0	1D 00 044A DD	5643	FZU420 CLC	\$PRDEV-1,FZU440+@OP1-1(1,@BR) TEST PRINT DEVICE PARAMETER	
39D5	F2 82 11	5644	JL	FZU470 * AND BRANCH IF PRINTER ONLY	
		5645	*		
		5646	*	CRT (AND POSSIBLE PRINTER) ACTIVE - OUTPUT PRINT AREA ON THE CRT	
		5647	*		
39D8	74 02 E0	5648	FZU430 ST	FZU450(,@BR),@XR STORE PRINT PARAM LIST CADDR	
		5649	*		
39DB	C0 87 0000	5650	FZU440 B	*-* LINK TO EXECUTE THE CRT IOCR	
39DF		5651	FZU450 DS	CL(@CADDR) PRINT PARAMETER LIST CADDR	
		5652	*		
		5653	*	TEST FOR MATRIX PRINTER ACTIVE ON THE SYSTEM	
		5654	*		
39E1	1D 01 044B DE	5655	FZU460 CLC	\$PRDEV,FZU440+@OP1(@CADDR,@BR) TEST PRINT DEVICE PARAM	
39E6	F2 02 06	5656	JNL	FZU480 * AND BRANCH IF CRT ONLY	
		5657	*		
		5658	*	MATRIX PRINTER ACTIVE - OUTPUT PRINT AREA ON THE MATRIX PRINTER	
		5659	*		
39E9	C0 87 12B1	5660	FZU470 B	I\$CALL LINK TO EXECUTE MAT PRINT IOCR	
39ED	2800	5661	DC	AL(@VADDR)(V\$SPRT) MATRIX-PRINTER IOCR VADDR	
		5662	*		
		5663	*	RETURN CONTROL TO THE CALLING PROGRAM	
		5664	*		
39EF	C0 87 0000	5665	FZU480 B	*-* RETURN TO CALLING ROUTINE	
		5666	*		
		5667		*****	

FZUPRT - S/3 BASIC INTERPRETER PRINT USING EXEC RTN

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 315
		5669		*****	
		5670		* PRINT USING INTERFACE CONSTANTS (2ND VM PAGE)	*
		5671		*****	
		5672		*	
39F3	2004	39F4	5673	FZUPDA DC AL(@CADDR)(I\$CSXA+4) CRT CORE ENTRY BASE ADDRESS	
		5674		*	
		5675		*****	
		5676		* PRINT USING INTERFACE WORK AREAS (2ND VM PAGE)	*
		5677		*****	
		5678		*	
		5679		*FZUPPL PPL FUNC=@PRETR PRINT & CARR RETURN PARAM LIST	
		39F5	5680	FZUPPL EQU * PPL ADDRESS	
39F5	C0	39F5	5681	DC AL1(@PRETR) FUNCTION REQUESTED	
39F6	00	39F6	5682	DC AL1(*-*) PRINT COUNT	
39F7	0000	39F8	5683	DC AL2(*-*) DATA ADDRESS	
		5684		*** END OF EXPANSION ***	
		39F6	5686	FZUPCT EQU FZUPPL+@PRCNT PRINT AREA COUNT PARAMETER	
		39F8	5687	FZUPAD EQU FZUPPL+@PDATA PRINT AREA CADDR PARAMETER	
		5688		*	
		39F9	5689	FZURPL EQU * CARRAGE RETURN PPL	
39F9	80	39F9	5690	DC AL1(@RETRN) CARRAGE RETURN FUNCTION	
39FA	80	39FA	5691	DC AL1(@RETRN) CARRAGE RETURN FUNCTION	
		5692		*	
		5693		*****	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 316
		5695		*****			
		5696	*	VIRTUAL MEMORY PRINT USING INTERFACE ROUTINE 3RD VM PAGE -			*
		5697	*	PERFORMS FOLLOWING ACTIVITIES DEPENDING ON ENTRY POINT -			*
		5698	*				*
		5699	*	CONTROL CODES 3,6 (PRIMARY DATA ELEMENT) ENTRY -			*
		5700	*	* SCANS CURRENT CONVERSION SPECIFICATION FOR CHARACTERISTICS			*
		5701	*	* IF DATA ELEMENT IS ARITHMETIC -			*
		5702	*	* CONVERTS ELEMENT TO OUTPUT FORM AS SPECIFIED			*
		5703	*	* INSERTS ELEMENT INTO IMAGE, PADDING UNUSED CHARACTERS			*
		5704	*	* IF DATA ELEMENT IS CHARACTER AND NOT NULL -			*
		5705	*	* CLEARS CONVERSION SPECIFICATION TO BLANKS			*
		5706	*	* INSERTS ELEMENT INTO SPECIFICATION AT LEFTMOST BYTE			*
		5707	*	* IF DATA ELEMENT IS CHARACTER AND NULL -			*
		5708	*	* CLEARS CONVERSION SPECIFICATION TO BLANKS			*
		5709	*				*
		5710	*	CONTROL CODE 7 (SECONDARY CHARACTER CONSTANT SEGMENT) ENTRY -			*
		5711	*	* IF CURRENT CONVERSION SPECIFICATION NOT FILLED -			*
		5712	*	* INSERTS CHARACTER SEGMENT FOLLOWING PRIOR INSERTIONS			*
		5713	*	* IF CURRENT CONVERSION SPECIFICATION IS FILLED -			*
		5714	*	* BYPASSES UTILIZATION OF THE CHARACTER SEGMENT			*
		5715		*****			
		5716	*				
		5717	*	ESTABLISH ADDRESSABILITY FOR INTERFACE 3RD VM PAGE			
		5718	*				
3A00		5719		ORG *,B@LVPG,0			BEGIN AT PAGE BOUNDARY
	3A00	5720		USING *,@BR			SET BASE ADDRESS AT PAGE BOUND
		5721	*				
		5722		*****			

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 317
				5724		*****				
				5725	*	ENTRY FOR CONTROL CODES 3,6 - PRIMARY ELEMENT OR NULL CHAR CONSTANT *				
				5726		*****				
				5727	*					
				5728	*	INDEX REGISTER @XR REFERENCES THE 1ST BYTE OF CURRENT CONVERSION				
				5729	*	SPECIFICATION - INITIALIZE THE CONVERSION SPECIFICATION SCAN ROUTINE				
				5730	*					
3A00	34	02	0DCE	5731	FZU500	ST	I\$IMC1,@XR	SAVE CONV SPEC BEGINNING CADDR		
3A04	3B	05	0DC5	5732		SBF	I\$INDR,I\$ECSW+I\$SNSW	SET E-FORMAT AND SIGN INDRS OFF		
3A08	5F	03	FD FD	5733		SLC	FZUCNT(,@BR),FZUCNT(FZUNCT+1,@BR)	ZERO THE FRACTION INCR		
				5734	*			* AND SPEC, DIGIT & FRAC CNTS		
3A0C	7C	40	2D	5735		MVI	FZU550+@Q(,@BR),B@BLNK	SET FILL CPIARACTER TO BLANK		
				5736	*					
				5737	*	TEST FOR A LEADING CONVERSION SPECIFICATION DECIMAL POINT				
				5738	*					
3A0F	BD	4B	00	5739	FZU510	CLI	B@CHAR(,@XR),B@DPNT	IF LEADING CIS CHAR NOT DECIMAL		
3A12	F2	01	11	5740		JNE	FZU540	* PT, GO TEST FOR A DIGIT SPEC		
				5741	*					
				5742	*	PROCESS CONVERSION SPECIFICATION DECIMAL POINT				
				5743	*					
3A15	34	02	0DD0	5744	FZU520	ST	I\$SDPT,@XR	SAVE CADDR OF C'S DECIMAL POINT		
3A19	7C	01	FA	5745		MVI	FZUFIN(,@BR),@B1	SET FRACTION INCREMENT EQUAL I		
3A1C	7C	F0	2D	5746		MVI	FZU550+@Q(,@BR),B@DEC0	SET FILL CHAR TO DECIMAL ZERO		
				5747	*					
3A1F	5E	00	FB F8	5748	FZU530	ALC	FZUSCT(,@BR),FZUSIN(1,@BR)	INCREMENT THE SPEC CHAR COUNT		
3A23	F2	87	0D	5749		J	FZU560	* AND RESUME DIGIT SPEC LOOP		
				5750	*					
				5751	*	LEADING C/S CHARACTER NOT A DECIMAL POINT - TEST FOR A DIGIT SPEC				
				5752	*					
3A26	BD	7B	00	5753	FZU540	CLI	B@CHAR(,@XR),B@DIGS	IF C/S CHAR IS NOT A DIGIT SPEC		
3A29	D0	01	1F	5754		BNE	FZU530(,@BR)	* ASSUME SIGN (4,-) AND BRANCH		
				5755	*					
				5756	*	DIGIT SPECIFICATION SCAN LOOP - REPLACE EACH DIGIT SPEC W/ FILL CHAR				
				5757	*					
3A2C	BC	00	00	5758	FZU550	MVI	B@CHAR(,@XR),*-*	REPLACE DIGIT SPEC W/ FILL CHAR		
3A2F	5E	02	FD FA	5759		ALC	FZUCNT(,@BR),FZUCIN(FZUNCT,@BR)	INCREMENT SPECIFICATION,		
				5760	*			* DIGIT, AND FRACTION COUNTERS		
3A33	E2	02	01	5761	FZU560	LA	@B1(,@XR),@XR	INCREMENT THE CONV SPEC POINTER		
3A36	BD	7B	00	5762		CLI	B@CHAR(,@XR),B@DIGS	IF US CHARACTER IS DIGIT SPEC		
3A39	D0	81	2C	5763		BE	FZU550(,@BR)	* REPEAT DIGIT SPEC PROCESSING		
				5764	*					
				5765	*	CONVERSION SPECIFICATION CHARACTER NOT A DIGIT SPECIFICATION -				
				5766	*	TEST FOR AN EMBEDDED DECIMAL POINT				
				5767	*					
3A3C	BD	4B	00	5768	FZU570	CLI	B@CHAR(,@XR),B@DPNT	IF C/S CWAR NOT A DECIMAL POINT		
3A3F	F2	01	09	5769		JNE	FZU590	* GO TERMINATE OS SCAN LOOP		
				5770	*					
3A42	7D	01	FA	5771	FZU580	CLI	FZUFIN(,@BR),@B1	IF DECIMAL POINT WAS ALREADY		
3A45	F2	81	1F	5772		JE	FZU600	ENCOUNTERED, GO EXIT SCAN		
3A48	D0	87	15	5773		B	FZU520(,@BR)	* ELSE PROCESS DIP ARO CONTINUE		
				5774	*					
				5775	*	SCAN LOOP TERMINATED - TEST FOR AN ENCOUNTERED DECIMAL POINT.				
				5776	*					
3A4B	7D	01	FA	5777	FZU590	CLI	FZUFIN(,@BR),@B1	IF EXPLICIT DECIMAL POINT WAS		
3A4E	F2	81	04	5778		JE	FZU595	* ENCOUNTERED, GO TEST E-FORMAT		
3A51	34	02	0DD0	5779		ST	I\$SDPT,@XR	* ELSE SAVE TINE CORE ADDR OF		

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 318
				5780	*		* IMPLIED DECIMAL POINT			
				5781	*	TEST FOR AND PROCESS POSSIBLE E-FORMAT SDECIFICATION				
				5782	*					
3A55	9D	03	03	F7	5783	FZU595	CLC FZULXB-1(,@XR),FZUECS(FZULXB,@BR) IF E-FORMAT SPEC NOT			
3A59	F2	01	0B		5784	JNE	FZU600 * PRESENT, GO EXIT THE SCAN			
				5785	*					
3A5C	3A	04	0DC5	5786		SBN	I\$INDR,I\$EC SW SET E-FORMAT INDICATOR ON			
3A60	5E	00	FB	F3	5787	ALC	FZUSCT(,@BR),FZULNX(1,@BR) INCREMENT SPEC CHAR COUNT AND			
3A64	E2	02	04		5788	LA	FZULXB(,@XR),@XR * POINTER FOR E-FORMAT SPEC			
				5789	*					
				5790	*	EXIT CONVERSION SPECIFICATION SCAN - SAVE POINTER AND ALL COUNTERS				
				5791	*					
3A67	1C	02	0D5A	FD	5792	FZU600	MVC I\$CSCT,FZUCNT(FZUNCT,@BR) STORE COUNTERS IN FIXED AREA			
3A6C	34	02	0DCC		5793	ST	I\$IMPT,@XR STORE CADDR OF BYTE IMMEDIATELY			
				5794	*		* FOLLOWING CONVERSION SPEC			
				5795	*					
				5796	*	*****				

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 319
		5798		*****	
		5799	*	PRIMARY DATA ELEMENT PROCESSING FOR CHARACTERIZED CONVERSION SPEC	*
		5800		*****	
		5801	*		
		5802	*	DETERMINE TYPE OF DATA ELEMENT TO RE PROCESSED	
		5803	*		
3A70	3A 18 0DC5	5804	FZU610	SBN I\$INDR,I\$CSSW+I\$STSW	SET CONVERSION SPECIFICATION
		5805	*		* AND CHAR STRING INDICATORS ON
3A74	38 03 0D57	5806	TBN	I\$PARM,B@PUNS	IF DATA ELEMENT IS NULL STRING
3A78	F2 10 10	5807	JT	FZU630	* GO SET COW SPEC FOR CHARS
3A7B	3D 12 0BA1	5808	CLI	I\$SLLC,I@LCRV-1	IF CHAR ELEMENT LAST STACKED
3A7F	F2 81 09	5809	JE	FZU630	* GO SET CONV SPEC FOR CHARS
		5810	*		
		5811	*	ARITHMETIC DATA ELEMENT - CONVERT AND INSERT ELEMENT IN IMAGE	
		5812	*	ACCORDING TO THE CONVERSION SPECIFICATION	
		5813	*		
3A82	C0 87 12B1	5814	FZU620	B I\$CALL	LINK TO CONVERT ARITH ELEMENT
3A86	3B00	5815	DC	AL(@VADDR)(FZU730)	ARITHMETIC CONVERSION RTN VADDR
		5816	*		
3A88	D0 87 EB	5817	B	FZU720(,@BR)	GO RETURN TO CALLING PAGE
		5818	*		
		5819	*	CHARACTER ELEMENT OR NULL CHARACTER CONSTANT - CLEAR TNT SCANNED	
		5820	*	CONVERSION SPECIFICATION TO BLANK CHARACTERS	
		5821	*		
3A8B	76 02 F1	5822	FZU630	A FZU3M2(,@BR),@XR	DER NG TO 2ND TO LAST C/S CHAR
3A8E	BC 40 01	5823	MVI	FZUCSP+1(,@XR),B@BLNK	SET RIGHTMOST CIS CHAR - BLANK
3A91	5C 00 9D FB	5824	MVC	FZU640+@Q(,@BR),FZUSCT(1,@BR)	SET PROPAGATION INST LENGTH
3A95	5F 00 9D EF	5825	SLC	FZU640+@Q(,@BR),FZU3B2(1,@BR)	* CODE FOR CONV SPEC LENGTH
3A99	F2 82 04	5826	JL	FZU650	BRANCH IF CONV SPEC LENGTH . I
3A9C	AC 00 00 01	5827	FZU640	MVC FZUCSP(,@XR),FZUCSP+1(@VQ,@XR)	CLEAR CONV SPEC TO BLANKS
		5828	*		
		5829	*	TEST FOR A NULL CHARACTER CONSTANT - EXIT IF PRESENT	
		5830	*		
3AA0	38 03 0D57	5831	FZU650	TBN I\$PARM,B@PUNS	IF DATA ELEMENT NOT NULL STRING
3AA4	F2 90 19	5832	JF	FZU680	* 10 INSERT CHAR ELEMENT IN CIS
3AA7	D0 87 EB	5833	B	FZU720(,@BR)	* ELSE GO RETURN TO CALLING PG
		5834	*		
		5835		*****	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 320
		5837		*****	
		5838	*	ENTRY FOR CONTROL CODE 7 - SECONDARY CHARACTER CONSTANT SEGMENT	*
		5839		*****	
		5840	*		
		5841	*	TEST FOR A FZULED CONVERSION SPECIFICATIS	
		5842	*		
3AAA 38 08 0DC5		5843	FZU660 TBN	I\$INDR,I\$STSW	IF CHAR STRING INDICATOR OFF
3AAE D0 90 EB		5844	BF	FZU720(,@BR)	* GO RETURN TO CALLING PACE
		5845	*		
		5846	*	CONVERSION SPECIFICATION NOT FILLED - ADJUST IMAGE PARAMETERS TO	
		5847	*	PERMIT ADDITION OF SECONDARY SEGMENT TO CURRENT CONVERSION SPEC	
		5848	*		
3AB1 1E 00 0DCE F2		5849	FZU670 ALC	I\$IMC1,FZULCF(1,@BR)	ADJUST C/S 1ST BYTE POINTER AND
3AB6 1F 00 0D58 F2		5850	SLC	I\$SSCT,FZULCF(1,@BR)	* CHAR COUNT FOR CHAR ELEM LNG
3ABB 4C 00 FB 0D58		5851	MVC	FZUSCT(,@BR),I\$SSCT(1)	MOVE ADJ CHAR CNT TO WORK AREA
		5852	*		
		5853	*	ESTABLISH LENGTH OF CHARACTER FIELD TO BE MOVED TO CONVERSION SPEC -	
		5854	*	LENGTH DETERMINED FROM SPECIFICATION CAPACITY OR CHARACTER ELEMENT	
		5855	*	LENGTH, WHICHEVER IS SHORTER.	
		5856	*		
3AC0 7D 12 FB		5857	FZU680 CLI	FZUSCT(,@BR),I@LCRF	IF CORR C'S CAPACITY SREATRR,
		5858	*		* THAN CHARACTER ELEMENT LENGTH
3AC3 F2 84 07		5859	JH	FZU690	* SKIP TO LIMIT WORK AREA CHAR
		5860	*		* COUNT TO CHAR ELEMENT LENGIS,
3AC6 3B 08 0DC5		5861	SBF	I\$INDR,I\$STSW	* ELSE SET CHAR STRING MDR OF
		5862	*		* FOR NO REMAINING C/5 CAPACITY
3ACA F2 87 03		5863	J	FZU700	* AND SKIP TO MOVE CHARS T3 C/S
		5864	*		
3ACD 7C 12 FB		5865	FZU690 MVI	FZUSCT(,@BR),I@LCRF	SET WORK AREA CHAR COUNT EOUAL
		5866	*		* LENGTH OR A CHARACTER ELEMENT
		5867	*		
		5868	*	MOVE STACKED CHARACTER ELEMENT (WHOLE OR TRUNCATED) TO CURRENTLY	
		5869	*	IF REFERENCED LOCATION IN THE CONVERSION SPECIFICATION.	
		5870	*		
3AD0 5C 00 EA FB		5871	FZU700 MVC	FZU710+@DD2(,@BR),FZUSCT(1,@BR)	SET DATA MOVE 1ST FOR
3AD4 5F 00 EA E6		5872	SLC	FZU710+@DD2(,@BR),FZU3B1(1,@BR)	* CURRENT CAPACITY OF
3AD8 5C 01 E9 EA		5873	MVC	FZU710+@DD2-1(,@BR),FZU710+@DD2(FZULQD,@BR)	* CONV SPEC
		5874	*		
3ADC 35 02 0DCE		5875	L	I\$IMC1,@XR	LOAD CADDR 1ST AVAIL C/S BYTE
3AE0 35 01 0D4E		5876	L	I\$STAK,@BR	LOAD THE STACK POINTER AND
3AE4 D2 01 01		5877	FZU705 LA	@B1(,@BR),@BR	* INCR TO 1ST CHAR ELEMENT BYTE
		5878	*		
3AE7 9C 00 00 00		5879	FZU710 MVC	*-*(,@XR),*-(@VQ,@BR)	MOVE CHAR ELEMENT TO CONV SPEV
		5880	*		
		5881	*	EXIT 3RD VM PAGE - RETURN TO CALLING PAGE	
		5882	*		
3AEB C0 87 12D3		5883	FZU720 B	I\$RTRN	RETURN TO CALLING PAGE.
		5884	*		
		5885		*****	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 321
		5887		*****	
		5888		* PRINT USING INTERFACE CONSTANTS (3RD VM PAGE)	*
		5889		*****	
		5890		*	
3AEF 02		3AE6 5891	FZU3B1 EQU	FZU705+@D1	BINARY INTEGER +1
3AF0 FFFE		3AEF 5892	FZU3B2 DC	IL1'2'	BINARY INTEGER +2
		3AF1 5893	FZU3M2 DC	IL(@REGL)'-2'	BINARY INSFGER -2
		5894		*	
3AF2 12		3AF2 5895	FZULCF DC	AL1(I@LCRF)	LENGTH OF CSAR ELEM DATA FIELD
		5896		*	
		0004 5897	FZULXB EQU	4	LENGTH OF EXTERNAL EXPONENT
3AF3 04		3AF3 5898	FZULNX DC	AL1(FZULXB)	LENGTH OF EXTERNAL EXPONENT
3AF4 5A5A5A5A		3AF7 5899	FZUECS DC	CL(FZULXB)'!!!!'	E-FORMAT CONVERSION SPEC
		5901		*****	
		5902		* PRINT USING INTERFACE WORK AREAS (3RD VM PAGE)	*
		5903		*****	
		5904		*	
		0003 5905	FZUNCT EQU	3	NO. OF CONV SPEC SCAN COUNTERS
		5906		*	
3AF8 01		3AF8 5907	FZUSIN DC	IL1'1'	CONV SPEC NARACTER INCREMENT
3AF9 01		3AF9 5908	FZUDIN DC	IL1'1'	CONV SPEC DIGIT INCREMENT
3AFA		3AFA 5909	FZUFIN DS	CL1	CONV SPEC FRACTION INCREMENT
		3AFA 5910	FZUCIN EQU	FZUFIN	CONV SPECIFICATION INCREMENTS
		5911		*	
3AFB		3AFB 5912	FZUSCT DS	CL1	CONV SPEC CNARACTER COUNTER
3AFC		3AFC 5913	FZUDCT DS	CL1	CONV SPEC DIGIT COUNTER
3AFD		3AFD 5914	FZUFCT DS	CL1	CONV SPEC FRACTION COUNTER
		3AFD 5915	FZUCNT EQU	FZUFCT	CONV SPECIFICATION COUNTERS
		5916		*	
		5917		*****	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 322
		5919		*****			
		5920	*	VIRTUAL MEMORY PRINT USING INTERFACE ROUTINE 4TH VM PAGE -			*
		5921	*	ARITHMETIC ELEMENT CONVERSION TO SPECIFICATION (PART I) -			*
		5922	*				*
		5923	*	PERFORMS FOLLOWING ACTIVITIES DEPENDING ON CONVERSION SPEC -			*
		5924	*	* PROCESSES SIGN OF THE ARITHMETIC ELEMENT			*
		5925	*	* ROUNDS THE ARITHMETIC ELEMENT (SIMPLE 5 ADDITION)			*
		5926	*	* NORMALIZES ARITHMETIC ELEMENT FOR E-FORMAT SPECIFICATION			*
		5927	*	* SETS CONVERSION SPECIFICATION TO (*) IF ELEMENT TOO LARGE			*
		5928	*	* CAUSES VALID ELEMENT TO BE CONVERTED FOR OUTPUT AS SPECIFIED			*
		5929		*****			
		5930	*				
		5931	*	ESTABLISH ADDRESSABILITY FOR INTERFACE 4TH VM PAGE			
		5932	*				
3B00		5933		ORG *,B@LVPG,0			BEGIN AT PAGE BOUNOARY
	3B00	5934		USING *,@BR			SET BASE ADDRESS AT PAGE BOUND
		5935	*				
		5936		*****			

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 323
				5938		*****				
				5939	*	ARITHMETIC ELEMENT SIGN PROCESSING ROUTINE				*
				5940		*****				
				5941	*					
				5942	*	TEST FOR SIGN OF THE STACKED ARITHMETIC ELEMENT				
				5943	*					
3B00	35	02	0D4E	5944	FZU730	L	I\$STAK,@XR		LOAD THE STACK POINTER	
3B04	B8	F0	07	5945	FZU735	TBN	I@SIGN(,@XR),B@ZPOS		TEST FOR POSITIVE STACKED ELEM	
3B07	BA	F0	07	5946		SBN	I@SIGN(,@XR),B@ZPOS		MAKE STACKED ELEMENT POSITIVE	
3B0A	35	02	0DCE	5947		L	I\$IMC1,@XR		LOAD CONV SPEC 1ST BYTE CADOR	
3B0E	F2	90	0C	5948		JF	FZU750		BRANCH IF ELEMENT WAS NEGATIVE	
				5949	*					
				5950	*	POSITIVE DATA ELEMENT - MODIFY	CONVERSION SIGN SPEC ACCORDINGLY			
				5951	*					
3B11	BD	60	00	5952	FZU740	CLI	B@CHAR(,@XR),B@MINS		IF SIGN SPEC IS NOT MINUS (-)	
3B14	F2	01	2D	5953		JNE	FZU780		* GO EXIT SIGN PROCESSING RTN,	
3B17	BC	40	00	5954		MVI	B@CHAR(,@XR),B@BLNK		* ELSE MAKE SIGN SPEC A BLANK	
3B1A	F2	87	27	5955		J	FZU780		* AND EXIT SIGN PROCESSING RTN	
				5956	*					
				5957	*	NEGATIVE DATA ELEMENT - TEST FOR AN IMPLICIT-SIGN SPECIFICATION				
				5958	*					
3B1D	BD	40	00	5959	FZU750	CLI	B@CHAR(,@XR),B@BLNK		IF 1ST C/S CHAR NOT DIGIT SPEC	
3B20	F2	01	0F	5960		JNE	FZU760		* SO TEST FOR PLUS SIGN SPEC	
3B23	3A	01	0DC5	5961		SBN	I\$INDR,I\$SNSW		SET IMPLIED SIGN INDICATOR ON	
3B27	1F	00	0D59 FB	5962		SLC	I\$SDCT,FZU4B1(1,@BR)		DECR DIGIT COUNT FOR SIGN SPACE	
3B2C	F2	01	15	5963		JNZ	FZU780		IF AT LEAST 1 DIGIT SPACE LEFT	
				5964	*				* GO EXIT SIGN PROCESSING RTN,	
3B2F	D0	87	DD	5965		B	FZU860(,@BR)		* ELSE GO EXEC C/S OVFLOW RTN.	
				5966	*					
				5967	*	MODIFY CONVERSION SIGN SPECIFICATION FOR NEGATIVE DATA ELEMENT				
				5968	*					
3B32	BD	4E	00	5969	FZU760	CLI	B@CHAR(,@XR),B@PLUS		IF SIGN SPEC IS NOT PLUS (4)	
3B35	F2	01	06	5970		JNE	FZU770		* GO TEST FOR LEADING DECML PT	
3B38	BC	60	00	5971		MVI	B@CHAR(,@XR),B@MINS		* ELSE MAKE SIGN SPEC A MINUS	
3B3B	F2	87	06	5972		J	FZU780		* AND EXIT SIGN PROCESSING RTN	
				5973	*					
				5974	*	DECIMAL POINT TEST - LEADING DECIMAL POINT COMBINED WITH A NEGATIVE				
				5975	*	DATA ELEMENT CAUSES CONVERSION SPECIFICATION OVERFLOW				
				5976	*					
3B3E	BD	4B	00	5977	FZU770	CLI	B@CHAR(,@XR),B@DPNT		IF LEADING C/S CHAR IS DECIMAL	
3B41	D0	81	DD	5978		BE	FZU860(,@BR)		* PT, GO EXEC C/S OVERFLOW RRN	
				5979	*					
				5980	*	EXIT SIGN PROCESSING ROUTINE - COMPUTE REMAINING INTEGER DIGITS IN				
				5981	*	THE CONVERSION SPECIFICATION.				
				5982	*					
3B44	0C	00	0D5B 0D59	5983	FZU780	MVC	I\$SICT,I\$SDCT(1)		SET INTEGER COUNT = DIGIT COUNT	
3B4A	0F	00	0D5B 0D5A	5984		SLC	I\$SICT,I\$SFCT(1)		* AND SUBTRACT FRACTION COUNT	
				5985	*					
				5986	*	TEST FOR A ZERO STACKED VALUE - ZERO VALUE REQUIRES NO ROUNDING OR				
				5987	*	E-FORMAT MODIFICATION FOR OUTPUT - E-FORMAT EXPONENT IS ALWAYS -99				
				5988	*					
3B50	35	02	0D4E	5989	FZU785	L	I\$STAK,@XR		LOAD THE STACK POINTER	
3B54	3C	1D	0D56	5990		MVI	I\$ADJX,B@NXZR-99		SET OUTPUT EYP ASSUMING A ZERO	
3B58	BD	F0	01	5991		CLI	I@MANL(,@XR),B@DEC0		IF STACKED VALUE ACTUALLY IS	
3B5B	D0	81	D4	5992		BE	FZU850(,@BR)		* ZERO, GO CONVERT VALUE TO C/S	
				5993	*					

[illegible]

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 325
		5996		*****	
		5997		* ARITHMETIC ELEMENT MANTISSA ROUNDING ROUTINE *	
		5998		*****	
		5999		*	
		6000		* INITIALIZE ROUNDING INSTRUCTION AND TEST FOR E-FORMAT CONVERSION	
		6001		*	
3B5E	4C 00 8D 0D59	6002	FZU790 MVC	FZU825+@D1(,@BR),I\$SDCT(1) SET MANTISSA DISP EQUAL TO	
		6003	*	* CONV SPEC DIGIT COUNT	
3B63	38 04 0DC5	6004	TBN	I\$INDR,I\$ECSW IF E-FORMAT CONVERSION INDR ON	
3B67	F2 10 13	6005	JT	FZU810 * BRANCH TO CONTINUE ROUNDING	
		6006	*		
		6007	*	RE-INITIALIZE ROUNDING INSTRUCTION FOR I- OR F-FORMAT SPECIFICATION -	
		6008	*	FOLLOWING TERM 'EXPONENT' REFERS TO NORMALIZED SIGNED EXPONENT,	
		6009	*	RATHER THAN TO NORMALIZED EXCESS 2**7 EXPONENT	
		6010	*		
3B6A	6C 00 8D 00	6011	FZU800 MVC	FZU825+@D1(,@BR),I@DEXP(1,@XR) SET MANTISSA DISP EQUAL	
3B6E	4E 00 8D 0D5A	6012	ALC	FZU825+@D1(,@BR),I\$SFCT(1) * EXPONENT + COS FRAC CNT	
3B73	F2 A0 27	6013	JOL	FZU830 IF FRACTION CNT IS LARGE	
3B76	5F 00 8D B8	6014	SLC	FZU825+@D1(,@BR),FZU4XZ(1,@BR) * OR SUM IS LESS THAN ZERO	
3B7A	F2 82 20	6015	JL	FZU830 * EXIT ROUNDING ROUTINE	
		6016	*		
3B7D	7D 06 8D	6017	FZU810 CLI	FZU825+@D1(,@BR),I@PREC-1 IF NO MANTISSA ROUNDING IS	
3B80	F2 84 1A	6018	JH	FZU830 * REQUIRED, SO EXIT THIS RTN	
		6019	*		
		6020	*	ROUNDING REQUIRED - ADD 5 TO MANTISSA DIGIT FOLLOWING LEAST	
		6021	*	SIGNIFICANT DIGIT TO BE INCLUDED IN THE CONVERSION SPECIFICATION	
		6022	*		
3B83	58 01 8C 8D	6023	FZU820 MZN	FZU825+@Q(,@BR),FZU825+@D1(,@BR) SET ROUNDING LENGTH CODE	
3B87	5E 00 8D FB	6024	ALC	FZU825+@D1(,@BR),FZU4B1(1,@BR) INCR MANTISSA DISP TO REF	
		6025	*	* DIGIT AFTER LEAST SIGNIFICANT	
3B8B	96 00 00 FF	6026	FZU825 AZ	*-*(@VQ,@XR),FZU4D5(1,@BR) ADD 5 TO ROUND MANTISSA UP	
3B8F	F2 08 0B	6027	JNOZ	FZU830 EXIT IF NO MANTISSA OVERFLOW	
		6028	*		
		6029	*	MANTISSA OVERFLOW - ADJUST FLOATING POINT EXPONENT AND MANTISSA	
		6030	*		
3B92	AC 05 07 06	6031	FZU828 MVC	I@MANR(,@XR),I@MANR-1(I@PREC-1,@XR) SHIFT MANTISSA RIGHT	
3B96	BC F1 01	6032	MVI	I@MANL(,@XR),B@DEC1 SET LEADING MANTISSA DIGIT = 1	
3B99	9E 00 00 FB	6033	ALC	I@DEXP(,@XR),FZU4B1(1,@BR) INCREMENT THE VALUE EXPONENT	
		6034	*		
		6035		*****	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 326
		6037		*****	
		6038	*	EXPONENT ADJUSTMENT POW E-FORMAT CONVERSION SPECIFICATION	*
		6039		*****	
		6040	*		
		6041	*	TEST FOR AN E-FORMAT CONVERSION SPECIFICATION	
		6042	*		
3B9D 38 04 0DC5		6043	FZU830 TBN	I\$INDR,I\$ECSW	IF E-FORMAT INDICATOR IS OFF
3BA1 F2 90 1E		6044	JF	FZU845	* EXIT EXPONENT ADJUSTMENT RTN
		6045	*		
		6046	*	E-FORMAT SPECIFICATION - VALUE EXPONENT IS TO BE MODIFIED TO FORCE	
		6047	*	LEFT JUSTIFICATION OF THE MANTISSA IN THE CONVERSION SPECIFICATION -	
		6048	*	COMPUTE THE EXPONENT ADJUSTED OUTPUT VALUE CAUSED BY THIS 'SHIFT'	
		6049	*		
3BA4 8F 00 00 0D5B		6050	FZU835 SLC	I@DEXP(,@XR),I\$SICT(1)	CORRECT VALUE EXPONENT FOR
3BA9 D0 82 DD		6051	BL	FZU860(,@BR)	* CONV SPECIFICATION 'SHIFT'
3BAC BD 1D 00		6052	CLI	I@DEXP(,@XR),B@NXZR-99	* IF CORRECTED EXPONENT < -99,
3BAF D0 82 DD		6053	BL	FZU860(,@BR)	* GO EXECUTE CONV SPEC OVFLOW
3BB2 2C 00 0D56 00		6054	MVC	I\$ADJX,I@DEXP(1,@XR)	* ELSE SAVE ADJ EXP FOR OUTPUT
		6055	*		
		6056	*	INSERT VALUE EXPONENT WHICH WILL FORCE LEFT JUSTIFICATION	
		6057	*		
3BB7 BC 80 00		6058	FZU840 MVI	I@DEXP(,@XR),B@NXZR	SET VALUE EXPONENT EQUAL TO
3BBA 8E 00 00 0D5B		6059	ALC	I@DEXP(,@XR),I\$SICT(1)	* CONVERSION SPEC INTEGER CNT
3BBF D0 A0 DD		6060	BOL	FZU860(,@BR)	IF INTEGER COUNT GREATER THAN
		6061	*		* (2**7 - 1), GO EXEC C/S OVFLOW
		6062		*****	

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15,	MOD 00	31/05/21	PAGE 327
					6064	*****	*****				
					6065	*	ARITHMETIC ELEMENT CONVERSION TO SPECIFICATION				*
					6066	*****	*****				
					6067	*					
					6068	*	TEST FOR LOSS OF SIGNIFICANCE (CONVERSION SPECIFICATION OVERFLOW)				
					6069	*					
	3BC2	6C	00	CE	00	6070	FZU845 MVC FZU846+@Q(,@BR),I@DEXP(1,@XR) IF CURRENT STACKED VALUE IS				
	3BC6	5F	00	CE	B8	6071	SLC FZU846+@Q(,@BR),FZU4XZ(1,@BR) * LESS THAN 1, SKIP TO CONV				
	3BCA	F2	04	07	6072	JNH	FZU850 * VALUE TO SPECIFICATION				
					6073	*					
	3BCD	3D	00	0D5B	6074	FZU846 CLI	I\$SICT,*-* IF VALUE INTEGERS EXCEED C/S				
	3BD1	D0	82	DD	6075	BL	FZU860(,@BR) * INT CNT, GO EXEC C/S OVFL0				
					6076	*					
					6077	*	CONVERT THE VALUE ACCORDING TO SPECIFICATION				
					6078	*					
	3BD4	C0	87	12B1	6079	FZU850 B	I\$CALL LINK TO CONVERT VALUE TO SPEC				
	3BD8	3C00			6080	DC	AL(@VADDR)(FZU880) ARITHMETIC CONVERSION RTN VADDR				
					6081	*					
	3BDA	D0	87	F7	6082	B	FZU870(,@BR) GO RETURN TO CALLING PAGE				
					6084	*****	*****				
					6085	*	CONVERSION SPECIFICATION OVERFLOW ROUTINE				*
					6086	*****	*****				
					6087	*					
	3BDD	35	02	0DCC	6088	FZU860 L	I\$IMPT,@XR LOAD CONVERSION SPEC POINTER				
	3BE1	76	02	FE	6089	A	FZU4M2(,@BR),@XR DECR XR TO 2ND TO LAST C/S CHAR				
	3BE4	BC	5C	01	6090	MVI	FZUCSP+1(,@XR),B@FOFL SET RIGHTMOST C/S CHAR = (*)				
	3BE7	4C	00	F4	0D58	6091	MVC FZU865+@Q(,@BR),I\$SSCT(1) SET PROPAGATION INST LEN6TH				
	3BEC	5F	00	F4	FC	6092	SLC FZU865+@Q(,@BR),FZU4B2(1,@BR) * CODE FOR CONV SPEC LENGTH				
	3BF0	F2	82	04	6093	JL	FZU870 BRANCH IF CONV SPEC LENGTH = 1				
	3BF3	AC	00	00	01	6094	FZU865 MVC FZUCSP(,@XR),FZUCSP+1(@VQ,@XR) SET CONV SPEC TO ALL (*)				
					6095	*					
					6096	*	EXIT 4TH VM PAGE - RETURN TO CALLING PAGE				
					6097	*					
	3BF7	C0	87	12D3	6098	FZU870 B	I\$RTRN RETURN TO CALLING PAGE				
					6099	*					
					6100	*****	*****				

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 328
				6102		*****				
				6103		* PRINT USING INTERFACE CONSTANTS (4TH VM PAGE)				*
				6104		*****				
				6105		*				
3BFB	01			3BFB	6106	FZU4B1 DC	IL1 '1'			BINARY INTEGER +1
3BFC	02			3BFC	6107	FZU4B2 DC	IL1 '2'			BINARY INTEGER +2
3BFD	FFFE			3BFE	6108	FZU4M2 DC	IL(@REGL) '-2'			BINARY INTEGER -2
3BFF	F5			3BFF	6109	FZU4D5 DC	DL1 '5'			DECIMAL INTEGER +5
				6110		*				
				3BB8	6111	FZU4XZ EQU	FZU840+@Q			ZERO NORMALIZED EXPONENT
				6112		*				
				6113		*****				

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 329
		6115		*****			
		6116	*	VIRTUAL MEMORY PRINT USING INTERFACE ROUTINE 5TH VM PAGE -			*
		6117	*	ARITHMETIC ELEMENT CONVERSION TO SPECIFICATION (PART 2)			*
		6118	*				*
		6119	*	CONVERTS VALID ELEMENT TO OUTPUT FORMAT AS SPECIFIE: -			*
		6120	*	* CONVERTS FLOATING POINT EXPONENT FOR E-FORMAT SPECIFICATION			*
		6121	*	* INSERTS INTEGER AND FRACTIONAL COMPONENTS OF ELEMENT INTO SPEC			*
		6122		*****			
		6123	*				
		6124	*	ESTABLISH ADDRESSABILITY FOR INTERFACE 5TH VM PAGE			
		6125	*				
3C00		6126		ORG *,B@LVPG,0			BEGIN AT PAGE BOUNDARY
	3C00	6127		USING *,@BR			SET BASE ADDRESS AT PAGE BOUND
		6128	*				
		6129	*	TEST FOR E-FORMAT CONVERSION SPECIFICATION			
		6130	*				
3C00 38 04 0DC5		6131	FZU880 TBN	I\$INDR,I\$ECSW			IF E-FORMAT INDICATOR IS OFF
3C04 D0 90 48		6132	BF	FZU900(,@BR)			* BRANCH AROUND EXPONENT CONV
		6133	*				
		6134		*****			

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 330
				6136		*****		
				6137	*	FLOATING POINT (ADJUSTED) EXPONENT CONVERSION TO OUTPUT FORMAT	*	
				6138		*****		
				6139	*			
				6140	*	INITIALIZE OUTPUT FORM OF EXPONENT IN CONV SPEC - TEST FOR EXP SIGN		
				6141	*			
3C07	35	02	0DCC	6142	FZU885	L	I\$IMPT,@XR	LOAD IMAGE POINTER - DECR TO
3C0B	76	02	F3	6143		A	FZU5M5(,@BR),@XR	* BYTE PRECEDING EXPONENT SPEC
				6144	*			
3C0E	9C	03	04 F9	6145		MVC	FZUIXP(,@XR),FZUEXB(FZULXB,@BR)	MOVE EXPONENT IMAGE TO
				6146	*			* CONV SPEC EXPONENT AREA
3C12	4C	00	CD 0D56	6147		MVC	FZU5XM(,@BR),I\$ADJX(1)	DETERMINE BINARY EXP MAGNITUDE
3C17	5F	00	CD 8B	6148		SLC	FZU5XM(,@BR),FZU5XZ(1,@BR)	* ASSUMING POSITIVE EXPONENT
3C1B	F2	81	2A	6149		JZ	FZU900	BRANCH IF EXPONENT IS ZERO
3C1E	F2	84	0B	6150		JH	FZU890	BRANCH IF EXPONENT IS POSITIVE
				6151	*			
				6152	*	NEGATIVE EXPONENT - MODIFY SIGN AND RECOMPUTE BINARY MAGNITUDE		
				6153	*			
3C21	BC	60	02	6154	FZU888	MVI	FZUIXP-FZULXM(,@XR),B@MINS	MAKE EXPONENT SIGN NEGATIVE
3C24	7C	80	CD	6155		MVI	FZU5XM(,@BR),B@NXZR	DETERMINE NEW BINARY MAGNITUDE
3C27	4F	00	CD 0D56	6156		SLC	FZU5XM(,@BR),I\$ADJX(1)	* FOR THE NEGATIVE EXPONENT
				6157	*			
				6158	*	CONVERT BINARY EXPONENT MAGNITUDE TO ZONED DECIMAL		
				6159	*			
3C2C	54	10	FB F4	6160	FZU890	ZAZ	FZUDAC(FZULXM,@BR),FZU5D1(1,@BR)	SET DEC ACCUMULATOR = 1
3C30	7C	01	34	6161		MVI	FZU892+@Q(,@BR),@B1	SET BINARY MASK FOR 2**0 BIT
3C33	78	00	CD	6162	FZU892	TBN	FZU5XM(,@BR),*-*	TEST BINARY END MAGNITUDE BIT
3C36	F2	90	04	6163		JF	FZU894	* AND BRANCH IF BIT IS ZERO
3C39	96	01	04 FB	6164		AZ	FZUIXP(FZULXM,@XR),FZUDAC(FZULXM,@BR)	INCR DECIMAL EXP
3C3D	5E	00	34 34	6165	FZU894	ALC	FZU892+@Q(,@BR),FZU892+@Q(1,@BR)	SHIFT BINARY MASK LEFT
3C41	56	01	FB FB	6166		AZ	FZUDAC(FZULXM,@BR),FZUDAC(FZULXM,@BR)	DOUBLE DECML ACCUM
3C45	D0	08	33	6167		BNOZ	FZU892(,@BR)	REPEAT LOOP UNTIL ACCUM > 64
				6168	*			
				6169		*****		

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15,	MOD 00	31/05/21	PAGE 331
					6171		*****				
					6172	*	CONVERT ELEMENT FRACTIONAL COMPONENT TO SPECIFICATION				*
					6173		*****				
					6174	*					
					6175	*	ACCESS STACKED DATA ELEMENT AND SAVE CURRENT BASE REGISTER				
					6176	*					
	3C48	35	02	0D4E	6177	FZU900	L I\$STAK,@XR				LOAD THE STACK POINTER
	3C4C	74	01	77	6178		ST FZU914+@OP1(,@BR),@BR				STORE BASE REG FOR CONV LOOP
					6179	*					
					6180	*	INITIALIZE THE STACK-TO-IMAGE MOVE INSTRUCTION - WHEN EXPONENT IS				
					6181	*	ZERO OR POSITIVE, CONVERSION SPEC DISPLACEMENT RELATIVE TO DECIMAL				
					6182	*	POINT IS SET TO ZERO AND ELEMENT DISPLACEMENT RELATIVE TO STACKED				
					6183	*	VALLE EXPONENT IS SET TO EXPONENT MAGNITUDE - WHEN EXPONENT IS				
					6184	*	NEGATIVE, THE REVERSE SITUATION IS ESTABLISHED.				
					6185	*					
	3C4F	5F	01	73 73	6186	FZU902	SLC FZU912+@DD2(,@BR),FZU912+@DD2(FZULDD,@BR)				SET DISPS TO 0
	3C53	6C	00	CD 00	6187		MVC FZU962+@DD2(,@BR),I@DEXP(1,@XR)				CALC EXPONENT MAGNITUDE
	3C57	5F	00	CD 8B	6188		SLC FZU962+@DD2(,@BR),FZU5XZ(1,@BR)				* ASSUMING POSITIVE EXP
	3C5B	F2	82	07	6189		JL FZU904				BRANCH IF EXPONENT IS NEGATIVE
					6190	*					
	3C5E	5E	00	73 CD	6191		ALC FZU912+@DD2(,@BR),FZU962+@DD2(1,@BR)				ADD EXP MAG TO DATA
	3C62	F2	87	13	6192		J FZU916				* DISP AND GO ENTER MOVE LOOP
					6193	*					
	3C65	5F	00	72 CD	6194	FZU904	SLC FZU912+@D1(,@BR),FZU962+@DD2(1,@BR)				ADD EXP MAG TO C/S
	3C69	F2	87	0C	6195		J FZU916				* DISP AND GO ENTER MOVE LOOP
					6196	*					
					6197	*	MOVE SINGLE FRACTIONAL DIGIT FROM STACKED ELEMENT TO CONVERSION SPEC				
					6198	*					
	3C6C	35	01	0DD0	6199	FZU910	L I\$SDPT,@BR				LOAD C/S DECIMAL POINT CADDR
	3C70	6C	00	00 00	6200	FZU912	MVC *-*(,@BR),*-(1,@XR)				MOVE FRAC DIGIT FR/ DATA TO C/S
	3C74	C2	01	0000	6201	FZU914	LA *-*,@BR				RESTORE PAGE BASE REGISTER
					6202	*					
					6203	*	INCREMENT DATA ELEMENT AND CONV SPEC DISPLACEMENTS - TEST FOR				
					6204	*	BOUNDARY CONDITION WITH RESPECT TO EITHER MANTISSA LENGTH OR				
					6205	*	CONVERSION SPECIFICATION FRACTION LENGTH				
					6206	*					
	3C78	5E	01	73 EF	6207	FZU916	ALC FZU912+@DD2(,@BR),FZUB11(FZULDD,@BR)				INCR BOTH INST DISP
	3C7C	7D	07	73	6208		CLI FZU912+@DD2(,@BR),I@PREC				IF MANTISSA DISP EXCEEDS PREC
	3C7F	F2	84	08	6209		JH FZU920				* GO PROCESS INTEGER COMPONENT
	3C82	4D	00	72 0D5A	6210		CLC FZU912+@D1(,@BR),I\$SFCT(1)				IF C/S DISP IS WITHIN FRACTION
	3C87	D0	04	6C	6211		BNH FZU910(,@BR)				* COUNT, BRANCH TO MOVE A DIGIT
					6212	*					
					6213		*****				

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 332
				6215		*****		
				6216	*	CONVERT ELEMENT INTEGER COMPONENT TO SPECIFICATION		*
				6217		*****		
				6218	*			
				6219	*	TEST FOR INTEGER COMPONENT IN THE DATA ELEMENT		
				6220	*			
3C8A	BD	80	00	6221	FZU920	CLI	I@DEXP(, @XR), B@NXZR	IF VALUE IS NOT LESS THAN 1
3C8D	F2	84	06	6222		JH	FZU925	* SKIP TO CONTINUE CONVERSION
3C90	7C	01	CD	6223		MVI	FZU962+@DD2(, @BR), @B1	SET INTEGER MOVE INST FOR ONE
3C93	BC	F0	01	6224		MVI	I@MANL(, @XR), B@DEC0	* DIGIT, AND MAKE DIGIT A ZERO
				6225	*			
				6226	*	INSERT MINUS SIGN BEFORE ELEMENT MANTISSA FOR POSSIBLE PROCESSING		
				6227	*	OF NEGATIVE ELEMENT WITH IMPLIED CONVERSION SIGN SPECIFICATION.		
				6228	*			
3C96	BC	60	00	6229	FZU925	MVI	I@DEXP(, @XR), B@MINS	OVERLAY FLY PT EXP WITH MINUS
				6230	*			
				6231	*	ACCESS POSSIBLE INTEGER PORTION OF THE CONVERSION SPECIFICATION		
				6232	*			
3C99	35	02	0DD0	6233	FZU930	L	I\$SDPT, @XR	LOAD C/S DECIMAL POINT CADDR
3C9D	76	02	F1	6234		A	FZU5M2(, @BR), @XR	DECR TO BYTE PRECEDING UNIT POS
				6235	*			
				6236	*	TEST FOR CONVERSION SPECIFICATION INTEGER POSITIONS		
				6237	*			
3CA0	3D	00	0D5B	6238	FZU940	CLI	I\$SICT, @ZERO	IF C/S INTEGER COUNT NOT ZERO
3CA4	F2	01	0D	6239		JNE	FZU950	* SKIP TO CONTINUE CONVERSION
				6240	*			
3CA7	38	01	0DC5	6241		TBN	I\$INDR, I\$SNSW	IF IMPLIED MINUS INDICATOR OFF
3CAB	D0	90	EA	6242		BF	FZU990(, @BR)	* SO RETURN TO CALLING PAGE
3CAE	BC	60	01	6243		MVI	B@CHAR+1(, @XR), B@MINS	* ELSE SET MINUS BEFORE DEC PT
3CB1	D0	87	EA	6244		B	FZU990(, @BR)	* AND RETURN TO CALLING PAGE
				6245	*			
				6246	*	ESTABLISH INTEGER BYTE LENGTH FOR CONVERSION SPECIFICATION		
				6247	*			
3CB4	5C	00	CB CD	6248	FZU950	MVC	FZU962+@Q(, @BR), FZU962+@DD2(1, @BR)	SET INST LENGTH CODE
				6249	*			* FOR DATA INTEGER LENGTH 4 1
3CB8	38	01	0DC5	6250		TBN	I\$INDR, I\$SNSW	IF IMPLIED MINUS INDICATOR ON
3CBC	F2	10	04	6251		JT	FZU960	* GO MOVE INTEGER & SIGN TO C/S
3CBF	5F	00	CB EE	6252		SLC	FZU962+@Q(, @BR), FZU5B1(1, @BR)	* ELSE DECR LNG FOR NO SIGN
				6253	*			
				6254	*	MOVE DATA ELEMENT INTEGER TO CONVERSION SPECIFICATION - INTEGERS		
				6255	*	GREATER THAN OR EQUAL TO 10**I@PREC IN MAGNITUDE LEAVE TRAILING		
				6256	*	GARBAGE IN THE CONVERSION SPECIFICATION.		
				6257	*			
3CC3	74	01	D1	6258	FZU960	ST	FZU964+@OP1(, @BR), @BR	SAVE CURRENT BASE REGISTER
3CC6	35	01	0D4E	6259		L	I\$STAK, @BR	LOAD THE STACK POINTER
3CCA	9C	00	01 00	6260	FZU962	MVC	FZUCSP+1(, @XR), *-*(@VQ, @BR)	MOVE DATA INTEGER TO C/S
3CCE	C2	01	0000	6261	FZU964	LA	*-*, @BR	RESTORE PAGE BASE REGISTER
				6262	*			
				6263	*	ADD REQUIRED TRAILING ZEROS TO CONVERSION SPECIFICATION INTEGER		
				6264	*			
3CD2	7D	08	CD	6265	FZU970	CLI	FZU962+@DD2(, @BR), I@PREC+1	IF DATA ELEMENT IS LESS THAN
3CD5	F2	82	12	6266		JL	FZU990	* 10**I@PREC, RETURN TO THE
				6267	*			* CALLING PAGE
3CD8	BC	F0	01	6268		MVI	FZUCSP+1(, @XR), B@DEC0	INSERT ZERO DIGIT BEFORE DEC PT
3CDB	F2	81	0C	6269		JE	FZU990	* AND SKIP IF PADDING COMPLETE
3CDE	5C	00	E7 CD	6270		MVC	FZU972+@Q(, @BR), FZU962+@DD2(1, @BR)	* ELSE SET PADDING

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 333
	3CE2	5F	00	E7	F5	6271	SLC FZU972+@Q(,@BR),FZUPP2(1,@BR) * INST LENGTH CODE AND			
	3CE6	AC	00	00	01	6272	FZU972 MVC FZUCSP(,@XR),FZUCSP+1(@VQ,@XR) * PROPAGATE DECML 7ERO			
						6273	*			
						6274	* END 5TH VM PAGE - RETURN TO CALLING PAGE			
						6275	*			
	3CEA	C0	87	12	D3	6276	FZU990 B I\$RTRN RETURN TO CALLING PAGE			
						6277	*			
						6278	*****			

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 334
		6280		*****	
		6281		* PRINT USING INTERFACE CONSTANTS (5TH VM PAGE)	*
		6282		*****	
		6283		*	
3CEE 01		3CEE 6284	FZU5B1 DC	IL1'1'	BINARY INTEGER *1
3CEF 01		3CEF 6285	FZUB11 DC	IL1'1'	DOUBLE BINARY INTEGER 1,1
3CF0 FFFE		3CF1 6286	FZU5M2 DC	IL(@REGL)'-2'	BINARY INTEGER -2
3CF2 FFFB		3CF3 6287	FZU5M5 DC	IL(@REGL)'-5'	BINARY INTEGER -5
3CF4 F1		3CF4 6288	FZU5D1 DC	DL1'1'	DECIMAL INTEGER +1
		6289		*	
		3C8B 6290	FZU5XZ EQU	FZU920+@Q	ZERO NORMALIZED EXPONENT
3CF5 09		3CF5 6291	FZUPP2 DC	AL1(I@PREC+2)	CURRENT FLT PT PRECISION + 2
		6292		*	
3CF6 C54EF0F0		3CF9 6293	FZUEXB DC	CL(FZULXB)'E+00'	EXPONENT IMAGE FOR OUTPUT
		6295		*****	
		6296		* PRINT USING INTERFACE WORK AREAS (5TH VM PAGE)	*
		6297		*****	
		6298		*	
		3CCD 6299	FZU5XM EQU	FZU962+@DD2	BINARY EXPONENT MAGNITUDE
		6300		*	
		0002 6301	FZULXM EQU	2	LENGTH OF DECML EXP MAGNITUDE
3CFA		3CFB 6302	FZUDAC DS	CL(FZULXM)	B TO D DECIMAL ACCUMULATOR
		6304		*****	
		6305		* PRINT USING INTERFACE EQUATES REFERENCING CONSTANTS	*
		6306		*****	
		6307		*	
		0002 6308	FZULQD EQU	2	LENGTH OF INST 1-CODE & DISP
		0002 6309	FZULDD EQU	2	LENGTH OF 2 INST DISPLACEMENTS
		6310		*	
		0000 6311	FZUCSP EQU	0	CONVERSION SPECIFICATION DISP
		0004 6312	FZUIXP EQU	4	DISP FOR EXPONENT SPEC IN IMAGE
		6313		*	
		6314		*****	
		6315		*	
		6316		*** END OF VIRTUAL MEMORY PRINT USING INTERFACE CODING *****	

ERR LOC	OBJECT CODE	ADDR STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 335
6318			*****			*
6319	*		5703-XM1 COPYRIGHT IBM CORP. 1970			*
6320	*		REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
6321	*					*
6322			*****			*
6323	*		*STATUS			*
6324	*		VERSION 1 MODIFICATION 0			*
6325	*					*
6326	*		*FUNCTION			*
6327	*	*	FZDMIP CONTAINS THE RUN-TIME ROUTINE WHICH EXECUTES MATRIX			*
6328	*		OPERATIONS FOR AN ARRAY REFERENCED IN A BASIC PROGRAM 'MAT			*
6329	*		INPUT' STATEMENT.			*
6330	*	*	THIS ROUTINE PERFORMS 'INPUT' OPERATIONS FOR EACH ELEMENT OF			*
6331	*		THE MATRIX. REFERENCED BY THE ARITHMETIC ARRAY DOPE VECTOR AT			*
6332	*		THE TOP OF THE RUN-TIME STACK. ELEMENTS ARE ENTERED ON A ROW-			*
6333	*		BY-ROW BASIS. EACH DATA LINE CONSISTING OF A PARTIAL OR ENTIRE			*
6334	*		ARRAY ROW. PARTIAL ARRAY ROWS ARE TERMINATED WITH A COMMA PRE-			*
6335	*		CEDING THE KEYBOARD CARRIAGE RETURN, AND THE END OF THE ROW IS			*
6336	*		SIGNIFIED WITH A CARRIAGE RETURN WITHOUT PRECEDING COMMA.			*
6337	*	*	A SINGLE QUESTION MARK IS PRINTED TO REQUEST ENTRY OF THE FIRST			*
6338	*		ARRAY ROW. THEREAFTER, TWO QUESTION MARKS ARE PRINTED TO RE-			*
6339	*		QUEST DATA LINE ENTRY UNTIL THE ARRAY IS COMPLETELY ASSIGNED.			*
6340	*		INPUT ERRORS IN ANY SINGLE LINE CAUSE A REQUEST (??) FOR THE			*
6341	*		RE-ENTRY OF THE ENTIRE ROW ASSOCIATED WITH THAT LINE (AFTER AN			*
6342	*		APPROPRIATE ERROR MESSAGE HAS BEEN PRINTED). INPUT IS AUTO-			*
6343	*		MATICALLY TERMINATED WHEN EACH ARRAY ELEMENT HAS BEEN ASSIGNED			*
6344	*		A VALUE.			*
6345	*	*	INQUIRY REQUEST MAY BE INVOKED WHENEVER THE KEYBOARD HAS BEEN			*
6346	*		ENABLED FOR INPUT. THIS RESULTS IN RE-EXECUTION OF THE 'STH'			*
6347	*		PSEUDO INSTRUCTION ASSOCIATED WITH THE CURRENT 'MAT INPUT'			*
6348	*		STATEMENT.			*
6349	*					*
6350	*		*ENTRY POINTS			*
6351	*		THIS ROUTINE HAS A SINGLE ENTRY POINT - FZDMIP - WHOSE FUNCTION			*
6352	*		IS DEFINED ABOVE. CALLING SEQUENCE IS			*
6353	*		B IPGCAL			*
6354	*		DC AL2(V\$XMIN)			*
6355	*		WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL ADDRESS			*
6356	*		OF ENTRY POINT FZDMIP. EXECUTION IS SUBJECT TO INPUT CONDITIONS			*
6357	*		DESCRIBED BELOW.			*
6358	*					*
6359	*		*INPUT			*
6360	*	*	I\$STAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER, THIS CON-			*
6361	*		TAINS THE CORE ADDRESS OR THE LEFTMOST BYTE OR THE ARITHMETIC			*
6362	*		ARRAY DOPE VECTOR AT THE TOP OF THE STACK.			*
6363	*	*	RUN-TIME STACK - THIS CONTAINS THE ARITHMETIC MAT DOPE VECTOR,			*
6364	*		ASSOCIATED WITH THE MATRIX TO BE AFFECTED, IN THE TOP STACK			*
6365	*		POSITION.			*
6366	*	*	KEYBOARD DATA ENTRY - NUMERIC AND (SIGNED) INTERNAL (&PI, ETC.)			*
6367	*		CONSTANTS ARE ENTERED FROM THE SYSTEM CONSOLE KEYBOARD IN COM-			*
6368	*		PLIANCE WITH THE CURRENT 'MAT INPUT' ARRAY DIMENSIONS(S).			*
6369	*					*
6370	*		*OUTPUT			*
6371	*	*	VIRTUAL MEMORY - ELEMENTS IN THE SPECIFIED ARITHMETIC ARRAY ARE			*
6372	*		ASSIGNRD VALUES, ON A ROW-BY-ROW BASIS, IN THE SEQUENCE ENTERED			*
6373	*		AT THE KEYBOARD.			*

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 336

```

6374 * * PRINTER/CRT MESSAGES - *
6375 * * TO REQUEST INITIAL INPUT FOR THE 1ST ARRAY ROW, THE PRINT *
6376 * DEVICE CARRIER/CURSOR IS RETURNED, A SINGLE QUESTION MARK *
6377 * IS PRINTED, AND THE CARRIER/CURSOR IS RETURNED AGAIN. *
6378 * * TO REQUEST INPUT FOR ANY OTHER CONDITION, THE PRINT DEVICE *
6379 * CARRIER/CURSOR IS RETURNED, DOUBLE QUESTION MARK IS *
6380 * PRINTED, AND THE CARRIER/CURSOR IS RETURNED AGAIN. *
6381 * * IN RESPONSE TO A DATA ERROR CONDITION, THE CARRIER/CURSOR *
6382 * IS RETURNED, AN ERROR MESSAGE OF THE FORM *
6383 * ERROR NNN AT LINE LLLL MESSAGE... *
6384 * IS PRINTED, AND THE CARRIER/CURSOR IS RETURNED AGAIN. *
6385 * * POSSIBLE MESSAGES ARE - *
6386 * * 800 - INVALID INPUT DATA - NUMERIC CONSTANT. *
6387 * * 802 - TOO MANY INPUT DATA ELEMENTS. *
6388 * * 804 - NOT ENOUGH ARRAY ROW ELEMENTS ENTERED. *
6389 * * I$WRK1 - 2 BYTES, FOR INTERPRETER COMMON WORK AREA 1. THIS *
6390 * AREA IS SET TO CONTAIN THE VIRTUAL ADDRESS OF THE CURRENT 'MAT *
6391 * INPUT' STATEMENT 'STH' PSEUDO INSTRUCTION WHENEVER CONSOLE *
6392 * INTERRUPT IS INVOKED DURING KEYBOARD INPUT. *
6393 * * I$RESW - 1 BYTE, FOR THE RECURSIVE STATEMENT ERROR SWITCH. *
6394 * THIS SWITCH IS SET TO CODE @NOP (DISABLES THE ERROR CONDITION) *
6395 * WHENEVER CONSOLE INTERRUPT IS INVOKED DURING KEYBOARD INPUT. *
6396 * *
6397 *EXTERNAL REFERENCES *
6398 * * V$SKEY - VIRTUAL ENTRY ADDRESS FOR DFKEYN, V.M. KEYBOARD IOCS. *
6399 * * I$CALL - ENTRY POINT FOR PAGING MODULE V.M. PROGRAM CALL RTN. *
6400 * * I$RTRN - ENTRY POINT FOR PAGING MODULE V.M. RETURN CONTROL RTN. *
6401 * * I$LOCK - ENTRY POINT FOR PAGING MODULE V.M. PAGE LOCKING RTN. *
6402 * * I$UNLK - ENTRY POINT FOR PAGING MODULE V.M. PAGE UNLOCKING RTN. *
6403 * * I$USTK - ENTRY POINT FOR INTERPRETER ELEMENT UNSTACKING RTN. *
6404 * * I$BSET - ENTRY POINT FOR ICBRAN EXECUTION CONTROL BRANCH RTN. *
6405 * * FZYPQ1 - 'INPUT' ROUTINE ENTRY, PRINTS A SINGLE QUESTION MARK. *
6406 * * FZXPQ2 - 'INPUT' ROUTINE ENTRY, PRINTS A DOUBLE QUESTION RAJIK. *
6407 * * FZXPEM - 'INPUT' ROUTINE ENTRY, PRINTS ERROR MSG, USING I$ERRC. *
6408 * * FZXMIS - 'INPUT' ROUTINE ENTRY, VALIDITY CHECKS MAT INPUT LINE. *
6409 * * FZXCNV - 'INPUT' ROUTINE ENTRY, CONVERTS AND STACKS AN ELEMENT. *
6410 * * I$BASE - CORE ADDRESS FOR INTERP BASE ADDRESSABILITY. *
6411 * * I$STAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER. *
6412 * * I$PARM - 2 BYTES, FOR THE INTERPRETER COMMUNICATION PARAMETER. *
6413 * * I$WRKI - 2 BYTES, FOR INTERPRETER COMMON WORK AREA 1. *
6414 * * I$ERRC - 1 BYTE, FOR THE INTERPRETER EXECUTION ERROR CODE. *
6415 * * I$STHA - 2 BYTES, FOR VIRTUAL ADDRESS OF CURR STMT 'STH' INST. *
6416 * * I$RESW - 1 BYTE, FOR INTERPRETER STATEMENT RECURSION ERROR SW. *
6417 * * I$VADR - 2 BYTES, FOR PAGING MODULE VIRTUAL ADDRESS PARAMETER. *
6418 * * F$CADR - 2 BYTES, FOR PAGING MODULE CORE ADDRESS OUTPUT PARAM. *
6419 * * $CISUS - 1 BYTE, FOR THE SYSTEM CONSOLE INTERRUPT INDICATOR. *
6420 * * V$BFR1 - VIRTUAL ADDRESS OF GENERAL V.M. BUFFER 1. *
6421 * *
6422 *EXITS, NORMAL *
6423 * CONTROL IS NORMALLY PASSED TO THE PAGING ROUTINE AT ENTRY POINT *
6424 * I$RTRN (IPGRTN) FOR A RETURN TO THE CALLING PROGRAM. THE SINGLE *
6425 * EXCEPTION OCCURS WHEN A CONSOLE INTERRUPT IS INVOKED 4HILE KEY- *
6426 * BOARD DATA ENTRY IS ENABLED. IN THIS EVENT, CONTROL IS PASSED TO *
6427 * THE CORE-RESIDENT INTERPRETER AT ENTRY POINT I$BSET (ICBSET) TO *
6428 * FORCE RE-EXECUTION OF THE LAST EXECUTED 'STH' PSEUDO INSTRUCTION *
6429 * (I.E, THE STATEMENT HEADER ASSOCIATED WITH THE CURRENT 'MAT *

```

ERR LOC	OBJECT CODE	ADDR STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 337
		6430 *	INPUT' STATEMENT).			*
		6431 *				*
		6432 *	EXITS, ERROR			*
		6433 *	N/A			*
		6434 *				*
		6435 *	TABLES/WORK AREAS			*
		6436 *	DOPE VECTOR WORK AREA - 8 BYTES, FOR ARRAY ELEMENT VIRTUAL			*
		6437 *	ADDRESS CALCLATIONS INVOLVING THE ORIGINALLY STACKED ARITH-			*
		6438 *	METIC ARRAY DOPE VECTOR.			*
		6439 *				*
		6440 *	ATTRIBUTES			*
		6441 *	* REUSABLE			*
		6442 *	* NATURALLY RELOCATABLE			*
		6443 *				*
		6444 *	CHARACTER CODE DEPENDENCY			*
		6445 *	THE OPERATION OF THIS MODULE DEPENDS UPON AN INTERNAL REPRES-			*
		6446 *	TATION OF THE EXTERNAL CHARACTER SET WHICH IS EQUIVALENT TO THE			*
		6447 *	ONE USED AT ASSEMBLY TIME. THE CODING HAS BEEN ARRANGED SO THAT			*
		6448 *	REDEFINITION OF CHARACTER CONSTANTS, BY REASSEMBLY, WILL RESULT			*
		6449 *	IN A CORRECT MODULE FOR THE NEW DEFINITIONS.			*
		6450 *				*
		6451 *	NOTES			*
		6452 *	ERROR PROCEDURES			*
		6453 *	* ERROR 1 - AN ARITHMETIC DATA ELEMENT IS EXPECTED DURING THE			*
		6454 *	SYNTAX CHECKING STEP, AND THE PROCESSED ELEMENT IS EITHER			*
		6455 *	NON-ARITHMETIC OR OTHERWISE INVALID. THE ERROR MESSAGE			*
		6456 *	'INVALID INPUT DATA - NUMERIC CONSTANT' IS PRINTED.			*
		6457 *	* ERROR 2 - MORE ELEMENTS ARE ENCOUNTERED IN THE INPUT DATA			*
		6458 *	LINE THAN CAN BE ACCOMODATED IN A SINGLE ARRAY ROW. THE			*
		6459 *	ERROR MESSAGE 'TOO MANY INPUT DATA ELEMENTS' IS PRINTED.			*
		6460 *	* ERROR 3 - ROW-END IS SIGNIFIED BY A CARRIAGE RETURN AFTER			*
		6461 *	DATA INPUT, AND THE CURRENT ARRAY ROW HAS NOT BEEN FULLY			*
		6462 *	ASSIGNED. THE ERROR MESSAGE 'NOT ENOUGH ARRAY ROW ELEMENTS'			*
		6463 *	IS PRINTED.			*
		6464 *	* IN EACH OF THE ABOVE CASES, PRINTING IS PERFORMED ON THE			*
		6465 *	CURRENT SYSTEM PRINT DEVICE(S), AND THE ERROR MESSAGE IS			*
		6466 *	FOLLOWED (ON A SEPARATE LINE) WITH A DOUBLE QUESTION MARK			*
		6467 *	REQUESTING CORRECT RE-ENTRY OF THE ENTIRE CURRENT ARRAY ROW.			*
		6468 *	* KEYBOARD DATA INPUT CAN BE ABORTED AT ANY TIME, DURING DATA			*
		6469 *	ENTRY, THROUGH CONSOLE INTERRUPT. WHEN THIS OCCURS, THE			*
		6470 *	'STH' PSEUDO INSTRUCTION FOR THE CURRENT 'MAT INPUT' STATE-			*
		6471 *	MENT IS RE-EXECUTED AND PROGRAM EXECUTION HALTS IN THE			*
		6472 *	'PAUSE' STATE.			*
		6473 *	* FZDMIP OTHERWISE UTILIZES INPUT IOCS ROUTINE DFKEYN (KEY-			*
		6474 *	BOARD) AND OUTPUT IOCS ROUTINES DFPRNT (MATRIX PRINTER) AND			*
		6475 *	DSPLYN (CRT), AND IS SUBJECT TO THE ERP'S INHEREN IN THESE			*
		6476 *	PROGRAMS.			*
		6477 *				*
		6478 *	REGISTER USAGE			*
		6479 *	* REGISTER @BR IS TO CONTAIN THE CORE PAGE SASE ADDRESS			*
		6480 *	ESTABLISHED THROUGH PAGING MODULE CONTROL FOR THE PAGE WHICH			*
		6481 *	INCLUDES FZDMIP, AND IS RESTORED THROUGH THE PAGING MODULE.			*
		6482 *	* REGISTER @XRIS NOT SAVED. IT IS USED IN FZDMIP FOR GENERAL			*
		6483 *	PURPOSE INDEXING OPERATIONS.			*
		6484 *				*
		6485 *	SAVED/RESTORE AREAS			*

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 338
		6486	*	N/A			*
		6487	*				*
		6488	*	MODIFICATION CONSIDERATIONS			*
		6489	*	N/A			*
		6490	*				*
		6491	*	REQUIRED MODULES			*
		6492	*	@SYSEQ - COMMON SYSTEM EQUATES.			*
		6493	*	@FXDEQ - SYSTEM NUCLEUS ADDRESSES AND INDICATOR EQUATES.			*
		6494	*	\$V\$EQU - VIRTUAL MEMORY FIXED ADDRESS EQUATES.			*
		6495	*	\$B\$EQU - COMPILER PARAMETER AND CONSTANT EQUATES.			*
		6496	*	\$I\$EQU - INTERPRETER FIXED LOCATION ADDRESS EQUATES.			*
		6497	*	\$I@SEQ - INTERPRETER PARAMETER EQUATES (FOR STD. PREC. ONLY).			*
		6498	*	\$I@LEQ - INTERPRETER PARAMETER EQUATES (FOR LONG PREC. ONLY).			*
		6499	*	FZXINP - INPUT' STATEMENT EXECUTION ROUTINE.			*
		6500	*				*
		6501	*	OTHER			*
		6502	*	N/A			*
		6503	*	*****			*

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 339
		6505		*****	
		6506		* START OF MAT INPUT STATEMENT EXECUTION MODULE	*
		6507		*****	
		6508		*	
		6509		* ESTABLISH ADDRESSABILITY FOR MAT INPUT EXECUTION ROUTINE	
		6510		*	
3D00		6511		*FZDPGB VPAGE 0	SET MAT INPUT ADDRESSABILITY
		6512		ORG *,256,0	SET STARTING ADDRESS
	3D00	6513		FZDPGB EQU *	START OF PROGRAM CODING
3C01		6514		ORG *-255	RESET IAR TO PAGE
3D00		6515		ORG *,256,0	* BOUNDARY ADDRESS
	3D00	6516		USING *,@BR	SET PAGE BASE ADDRESS
3D00		6517		ORG FZDPGB	RESET STARTING ADDRESS
		6518		*** END OF EXPANSION ***	
		6519		*	
	3D00	6520		FZDMIP EQU *	MAT INPUT ENTRY POINT
		6521		*	
		6522		* ARRAY DOPE VECTOR FOR THE NATRIY TO BE AFFECTED IS CONTAINED IN	
		6523		* THE RUN-TIME STACK - SAVE THIS DOPE VECTOR TO FREE STACK FOR DATA	
		6524		* TRANSFER OPERATIONS, AND SAVE ARRAY STARTING VIRTUAL ADDRESS FOR	
		6525		* POSSIBLE ERROR RECOVERY	
		6526		*	
3D00 C0 87 0E24		6527		B I\$I700	GO TO CHECK LINE PRINTER BUFFER
3D04 35 02 0D4E		6528		L I\$STAK,@XR	LOAD THE STACK POINTER
3D08 6C 07 EE 07		6529		MVC FZDDVR(,@BR),B@LADV-1(B@LADV,@XR)	SAVE THE DOPE VECTOR
3D0C 5C 01 F0 EE		6530		MVC FZDVAS(,@BR),FZDVAD(@VADDR,@BR)	SAVE INITIAL AWAY VADDQ
		6531		*	
		6532		* LOCK THE KEYBOARD INPUT BUFFER INTO CORE VIRTUAL MEMORY	
		6533		*	
3D10 1C 01 144A E4		6534		MVC I\$VADR,FZDBVA(@VADDR,@BR)	SET PAGING VADDR PARAM FOR BFR
3D15 C0 87 1354		6535		B I\$LOCK	LINK TO CORELCAD AND LOCK SUFFER
3D19 4C 01 55 144C		6536		MVC FZD050+@OP1(,@BR),I\$CADR(@CADDR)	SAVE BUFFER CORE ADDR
		6537		*	
		6538		* PRINT '?' TO REQUEST ARRAY 1ST ROW INPUT VIA SYSTEM KEYBOARD	
		6539		*	
3D1E C0 87 12B1		6540		B I\$CALL	LINK TO RETURN CARRIER AND
3D22 2C00	3D23	6541		DC AL(@VADDR)(FZXPQ1)	PRINT SINGLE QUESTION MARK
		6542		*	
3D24 7C 80 57		6543		MVI FZD052+@Q(,@BR),@NOP	ENABLE BUFFER INITIALIZATION
3D27 F2 87 0A		6544		J FZD020	BRANCH TO PROCESS THE 1ST ROW

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 340
					6546	*****	*****			
					6547	*	ARRAY ELEMENT INPUT AND ERROR CHECKING			*
					6548	*****	*****			
					6549	*				
					6550	*	SAVE ROW STARTING VIRTUAL ADDRESS FOR POSSIBLE ERROR RECOVERY			
					6551	*				
	3D2A	5C	01	F0	EE	6552	FZD005 MVC FZDVAS(,@BR),FZDVAD(@VADDR,@BR) SAVE IOW STARTING VADDR			
					6553	*				
					6554	*	PRINT '?? ' TO REQUEST SECONDARY ARRAY ROW INPUT VIA KEYBOARD			
					6555	*				
	3D2E	C0	87	12B1		6556	FZD010 B I\$CALL LINK TO RETURN CARRIER AND			
	3D32	2C06			3D33	6557	DC AL(@VADDR)(FZXPQ2) * PRINT DOUBLE QUESTION MARK			
					6558	*				
					6559	*	INITIALIZE THE MATRIX COLUMN COUNT FOR INPUT OPERATIONS WITHIN			
					6560	*	A SINGLE ROW OF THE ARRAY			
					6561	*				
	3D34	5C	01	EC	EA	6562	FZD020 MVC FZDDM2(,@BR),FZDD2S(B@LDMN,@BR) SET DM2 FOR COLUMN COUNT			
	3D38	5C	01	EE	F0	6563	MVC FZDVAD(,@BR),FZDVAS(@VADDR,@BR) SET ROW STARTING VADDR			
					6564	*				
					6565	*	ESTABLISH DATA TYPE CONTROL CODE IN THE RUN-TIME STACK TO SIMULATE			
					6566	*	SCALAR INPUT STATEMENT OPERATION - CONTROL CODE IS NUMBER OF ROW			
					6567	*	ELEMENTS STILL TO BE INPUT, OR MAXIMUM CONTROL CODE VALUE, WHICHEVER			
					6568	*	IS SMALLER - I\$PARM IS SET EQUAL 1 TO SPECIFY THE SINGLE CONTROL CODE			
					6569	*				
	3D3C	35	02	0D4E		6570	FZD030 L I\$STAK,@XR LOAD THE STACK POINTER			
	3D40	9C	00	00	EC	6571	MVC I@SCOD(,@XR),FZDDM2(B@LDMN-1,@BR) STACK DM2 RIGHT BYTE			
	3D44	5D	01	EC	E6	6572	CLC FZDDM2(,@BR),FZDDL(M@LDMN,@BR) IF DM2 NOT MORE THAN CODE			
	3D48	F2	04	03		6573	JNH FZD040 * MAY, SKIP TO SET COUNT PARAM			
	3D4B	BC	7F	00		6574	MVI I@SCOD(,@XR),FZDDTM-1 * ELSE STACK MAXIMUM CODE VALUE			
	3D4E	3C	01	0D57		6575	FZD040 MVI I\$PARM,@B1 * THEN SET CODE COUNT PARAMETER			
					6576	*				
					6577	*	INITIALIZE BUFFER AND EXECUTE KEYBOARD INPUT ROUTINE			
					6578	*				
	3D52	C2	02	0000		6579	FZD050 LA *-*,@XR LOAD THE INPUT BUFFER CADDR			
	3D56	F2	00	07		6580	FZD052 JC FZD055, *-* BRANCH IF BFR INITLZN DISABLED			
	3D59	BC	40	FF		6581	MVI B@LVPG-1(,@XR),B@BLNK INITIALIZE THE KEYBOARD INPUT			
	3D5C	AC	FE	FE	FF	6582	MVC B@LVPG-2(,@XR),B@LVPG-1(B@LVPG-1,@XR) * BUFFER TO BLANKS			
	3D60	7C	80	57		6583	FZD055 MVI FZD052+@Q(,@BR),@NOP ENABLE BUFFER INITIALIZATION			
					6584	*				
	3D63	C0	87	12B1		6585	B I\$CALL LINK TO EXECUTE KEYBOARD INPUT			
	3D67	2500			3D68	6586	DC AL(@VADDR)(V\$SKEY) VADDR OF KEYBOARD INPUT IOCR			
					6587	*				
					6588	*	TEST FOR CONSOLE INTERRUPT DURING KEYBOARD INPUT - RE-EXECUTE THE			
					6589	*	CURRENT STATEMENT 'STH' INSTRUCTION WHEN INTERRUPT HAS BEEN INVOKED			
					6590	*				
	3D69	3D	80	0496		6591	CLI \$CISUS,@NOP IF NO PENDING CONSOLE INTERRUPT			
	3D6D	F2	01	12		6592	JNE FZD060 * BRANCH TO SYNTAX CHECK INPUT			
					6593	*				
	3D70	0C	01	0D59	0D51	6594	MVC I\$WRK1,I\$STHA(@VADDR) SET BRANCH VADDR FOR LAST 'STH'			
	3D76	3C	80	0CE9		6595	MVI I\$RESW,@NOP DISABLE STMT NO. RECURSION ERR			
	3D7A	C2	01	0C60		6596	LA I\$BASE,@BR LOAD THE INTERPRETER BASE CADDR			
	3D7E	C0	87	119D		6597	B I\$BSET EXIT TO RE-EXECUTE 'STH' PMC			
					6598	*				
					6599	*	SYNTAX CHECK THE ENTIRE INPUT LINE - CHECK INPUT LINE DATA ELEMENTS			
					6600	*	FOR ARITHMETIC DATA TYPE. INPUT LINE MAY CONTAIN ENTIRE ROW OF ARRAY			
					6601	*	OR PART OF A ROW - PARTIAL ROW INPUT DATA LINE IS TERMINATE WITH A			

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 341
			6602	*	COMMA, FINAL DATA ELEMENT IN ROW TERMINATED WITH 'EOS' (CARR RETURN).	
			6603	*		
3D82	C0 87 12B1		6604	FZD060 B	I\$CALL LINK TO SYNTAX CHECK INPUT LINE	
3D86	2E17	3D87	6605	DC	AL(@VADDR)(FZXMIS) 'MAT INPUT' SYNTAX CHECKER VADUR	
			6606	*		
			6607	*	TEST FOR DISCOVERED SYNTAX/TYPE/COUNT ERROR - ON ERROR CONDITION	
			6608	*	PRINT APPROPRIATE MESSAGE AND REPEAT DATA INPUT ROUTINE FOR THE ROW	
			6609	*		
3D88	3D 00 0CBC		6610	CLI	I\$ERRC,I@NERR IF NO SYNTAX/TYPE/COUNT ERROR	
3D8C	F2 81 0C		6611	JE	FZD070 * GO STORE INP LINE DATA IN VHF	
			6612	*		
3D8F	C0 87 12B1		6613	B	I\$CALL LINK TO RETURN CARRIER AND	
3D93	2C18	3D94	6614	DC	AL(@VADDR)(FZXPEM) * PRINT SPECIFIED ERROR MESSAGE	
			6615	*		
3D95	7C 87 57		6616	MVI	FZD052+@Q(,@BR),@UCB DISABLE BUFFER INITIALIZATION	
3D98	D0 87 2E		6617	B	FZD010(,@BR) SO REPEAT INPUT OF ENTIRE ROW	
			6618	*	* AND SYNTAY/TYPE/COUNT CHECK	
			6619	*		
			6620	*****		

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 342
		6622			*****	
		6623			* CONVERT AND STORE INPUT LINE ELEMENTS IN ARRAY ROW	*
		6624			*****	
		6625			*	
		6626			* ADVANCE MATRIX ELEMENT POINTER TO NEXT ARRAY ELEMENT	
		6627			*	
3D9B 5E 01 EE E2		6628	FZD070	ALC	FZDVAD(, @BR), FZDPFL(@VADDR, @BR) INCR ARRAY ELEMENT VADDR	
		6629			*	
		6630			* CONVERT AND STACK AN INPUT LINE DATA ELEMENT	
		6631			*	
3D9F 75 02 55		6632		L	FZD050+@OP1(, @BR), @XR LOAD THE INPUT BUFFER CADDR	
		6633			*	
3DA2 C0 87 12B1		6634		B	I\$CALL LINK TO CONVERT AND STACK NEXT	
3DA6 3100	3DA7	6635		DC	AL(@VADDR) (FZXCNV) * DATA ELEMENT IN INPUT BUFFER	
		6636			*	
		6637			* UNSTACK THE ELEMENT TO THE DESTINATION MATRIX	
		6638			*	
3DA8 35 02 0D4E		6639		L	I\$STAK, @XR LOAD THE STACK POINTER	
3DAC 1C 01 144A EE		6640		MVC	I\$VADR, FZDVAD(@VADDR, @BR) SET VADDR PAGING PARAMETER	
3DB1 C0 87 0BB0		6641		B	I\$USTK LINK TO UNSTACK THE ELEMENT	
		6642			*	
		6643			* TEST FOR END OF THE CURRENT MATRIX ROW	
		6644			*	
3DB5 5F 01 EC E0		6645		SLC	FZDDM2(, @BR), FZDBN1(B@LDMN, @BR) DECR MATRIX COLUMN COUNT	
3DB9 F2 04 0B		6646		JNP	FZD080 BRANCH IF END-OF-ROW INDICATED	
		6647			*	
		6648			* NOT END-OF-ROW - TEST FOR END OC INPUT LINE DATA	
		6649			*	
3DBC 1F 00 0D56 E0		6650		SLC	I\$PARM-1, FZDBN1(1, @BR) DECR INPUT LINE ELEMENT COUNT	
3DC1 D0 84 9B		6651		BP	FZD070(, @BR) IF MORE ELEMENTS, GO ADD TO ROW	
3DC4 D0 87 3C		6652		B	FZD030(, @BR) * ELSE GO INPUT MORE ROW DATA	
		6653			*	
		6654			* END OF CURRENT ROW TEST FOR FINAL ROW OF MATRIX.	
		6655			*	
3DC7 5F 01 E8 E0		6656	FZD080	SLC	FZDDM1(, @BR), FZDBN1(B@LDMN, @BR) DECR MATRIX ROW COUNT	
3DCB D0 84 2A		6657		BP	FZD005(, @BR) IF NOT 0, GO PROCESS NEW ROW	
		6658			*	
		6659			* EXIT - RELEASE BUFFER PAGE FROM CORE AND RETURN TO CALLER	
		6660			*	
3DCE 1C 01 144A E4		6661		MVC	I\$VADR, FZDBVA(@VADDR, @BR) SET PAGING VADDR PARAM FOR FR	
3DD3 C0 87 1350		6662		B	I\$UNLK LINK TO UNLOCK BUFFER FROM CORE	
		6663			*	
3DD7 3B 1E 03E4		6664		SBF	\$LPRP3, @KENAB RESET MATRIX PRINT IND. 1-3	
3DDB C0 87 12D3		6665		B	I\$RTRN RETURN TO CALLING ROUTINE	
		6666			*	
		6667			*****	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 343
		6669		*****	
		6670		* MAT INPUT EXECUTION ROUTINE CONSTANTS	*
		6671		*****	
		6672		*	
3DDF 0001		3DE0 6673	FZDBN1 DC	IL2'1'	BINARY INTEGER +1
		6674		*	
3DE1 0005		3DE2 6675	FZDPFL DC	AL(@VADDR)(I@LPFV)	LENGTH OF PACKED FLT PT VALUE
3DE3 5400		3DE4 6676	FZDBVA DC	AL(@VADDR)(V\$BFR1)	KEYBOARD INPUT BUFFER VADDR
		6677		*	
3DE5 007F		3DE6 6678	FZDDL M DC	AL(B@LDMN)(FZDDTM-1)	MAX NO. ELEMENTS IN INPUT LINE
		6679		*	
		6680		*****	
		6681		* MAT INPUT EXECUTION ROUTINE WORK AREAS	*
		6682		*****	
		6683		*	
3DE7		3DE7 6684	FZDDVA EQU	*	MATRIX DOPE VECTOR SAVE ADDR
		3DEE 6685	FZDDVR DS	CL(B@LADV)	MATRIX DOPE VECTOR SAVE AREA
		6686		*	
		3DE8 6687	FZDDM1 EQU	FZDDVA+B@ACD1	DOPE VECTOR ROW DIMENSION
		3DEA 6688	FZDD2S EQU	FZDDVA+B@ACD2	DOPE VECTOR COLUMN DIMENSION
		3DEC 6689	FZDDM2 EQU	FZDDVA+B@AMAX	COLUMN DIMENSION COUNT AREA
		3DEE 6690	FZDVAD EQU	FZDDVA+B@ABAS	MATRIX ELEMENT VIRTUAL ADDRESS
		6691		*	
3DEF		3DF0 6692	FZDVAS DS	CL(@VADDR)	ELEMENT VIRTUAL ADDR SAVE AREA
		6693		*	
		6694		*****	
		6695		* MAT INPUT EXECUTION ROUTINE EQUATES REFERENCING CONSTANTS	*
		6696		*****	
		6697		*	
		0080 6698	FZDDTM EQU	X'80'	CONTROL CODE DATA TYPE MASK
		6699		*	* BIT OFF = ARITH, ON = CHAR
		6700		*	
		6701		*****	
		6702		*	
		6703		*** END OF MATRIX INPUT EXECUTION ROUTINE CODING	*****

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 344
		6705		*****			
		6706	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
		6707	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
		6708	*				*
		6709		*****			*
		6710	*	STATUS			*
		6711	*	VERSION 1 MODIFICATION 0			*
		6712	*				*
		6713	*	FUNCTION			*
		6714	*	* FZAMIO CONTAINS THE RUN-TIME ROUTINES WHICH EXECUTE MATRIX			*
		6715	*	OPERATIONS FOR AN ARRAY REFERENCED IN A BASIC PROGRAM			*
		6716	*	'MAT READ', 'MAT GET', OR 'MAT PUT' STATEMENT.			*
		6717	*	* THIS ROUTINE PERFORMS 'READ', 'GET', OR 'PUT' OPERATIONS FOR			*
		6718	*	EACH ELEMENT OF THE MATRIX REFERENCED BY THE ARITHMETIC ARRAY			*
		6719	*	DOPE VECTOR AT THE TOP OF THE RUN-TIME STACK.			*
		6720	*	* 'READ' - SUCCESSIVE ELEMENTS FROM THE PROGRAM 'DATA' FILE			*
		6721	*	ARE ASSIGNED, BEGINNING AT THE 'DATA' FILE ELEMENT			*
		6722	*	CURRENTLY REFERENCED BY I\$DATA, TO ELEMENTS IN THE SPECI-			*
		6723	*	FIED MATRIX ON A ROW-BY-ROW BASIS. I\$DATA IS LEFT REFER-			*
		6724	*	ENCING THE FIRST UNUSED 'DATA' ELEMENT.			*
		6725	*	* 'GET' - SUCCESSIVE ELEMENTS FROM THE CURRENTLY ACTIVE			*
		6726	*	EXTERNAL INPUT FILE ARE ASSIGNED, BEGINNING AT THE ELEMENT			*
		6727	*	CURRENTLY REFERENCED BY THE FILE POINTER, TO ELEMENTS IN			*
		6728	*	THE SPECIFIED MATRIX ON A ROW-BY-ROW BASIS. THE FILE			*
		6729	*	POINTER IS LEFT REFERENCING THE FIRST UNUSED FILE ELEMENT.			*
		6730	*	* 'PUT' - ELEMENTS FROM THE SPECIFIED MATRIX ARE ASSIGNED,			*
		6731	*	ON A ROW-BY-ROW BASIS, TO SUCCESSIVE ELEMENT POSITIONS IN			*
		6732	*	THE CURRENTLY ACTIVE EXTERNAL OUTPUT FILE BEGINNING AT THE			*
		6733	*	ELEMENT POSITION CURRENTLY REFERENCED BY THE FILE POINTER.			*
		6734	*	THE FILE POINTER IS LEFT REFERENCING THE FIRST UNUSED FILE			*
		6735	*	ELEMENT POSITION.			*
		6736	*				*
		6737	*	ENTRY POINTS			*
		6738	*	* ENTRY FZAMRD - FOR PERFORMING THE 'MAT READ' FUNCTION.			*
		6739	*	CALLING SEQUENCE IS			*
		6740	*	B IPGCAL			*
		6741	*	DC AL2(V\$XMRD)			*
		6742	*	WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL			*
		6743	*	ADDRESS OF ENTRY POINT FZAMRD.			*
		6744	*	* ENTRY FZAMGT - FOR PERFORMING THE 'MAT GET' FUNCTION.			*
		6745	*	CALLING SEQUENCE IS			*
		6746	*	B IPGCAL			*
		6747	*	DC AL2(V\$XMGT)			*
		6748	*	WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL			*
		6749	*	ADDRESS OF ENTRY POINT FZAMGT.			*
		6750	*	* ENTRY FZAMPT - FOR PERFORMING THE 'MAT PUT' FUNCTION,			*
		6751	*	CALLING SEQUENCE IS			*
		6752	*	B IPGCAL			*
		6753	*	DC AL2(V\$XMPT)			*
		6754	*	WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL			*
		6755	*	ADDRESS OF ENTRY POINT FZAMPT.			*
		6756	*	* ENTRY POINT EXECUTION FOR THESE OPERATIONS IS SUBJECT TO INPUT			*
		6757	*	CONDITIONS DESCRIBED BELOW.			*
		6758	*				*
		6759	*	INPUT			*
		6760	*	* I\$STAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER, THIS CON-			*

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 345
		6761	*	TAINS THE CORE ADDRESS OR THE LEFTMOST BYTE OF THE ARITHMETIC	*		
		6762	*	ARRAY DOPE VECTOR AT THE TOP OF THE STACK.	*		
		6763	*	* RUN-TIME STACK - THIS CONTAINS THE ARITHMETIC_ ARROY DOPE	*		
		6764	*	VECTOR (ASSOCIATED WITH THE MATRIX TO BE AFFECTED) IN THE TOP	*		
		6765	*	STACK POSITION.	*		
		6766	*	* FOR ENTRY FZAMRD - ELEMENTS FROM THE PROGRAM 'DATA' FILE FOR	*		
		6767	*	THE ARITHMETIC ARRAY SPECIFIED BY THE STACKED DOPE VECTOR.	*		
		6768	*	* FOR ENTRY FZAMGT - ELEMENTS FROM THE CURRENTLY ACTIVE EXTERNAL	*		
		6769	*	INPUT FILE FOR THE ARITHMETIC ARRAY SPECIFIED BY THE STACKED	*		
		6770	*	DOPE VECTOR.	*		
		6771	*	* FOR ENTRY FZAMPT - ELEMENTS IN VIRTUAL MEMORY FROM THE ARITH-	*		
		6772	*	METIC ARRAY SPECIFIED BY THE STACKED DOPE VECTOR FOR THE	*		
		6773	*	CURRENTLY ACTIVE EXTERNAL OUTPUT FILE.	*		
		6774	*		*		
		6775	*	*OUTPUT	*		
		6776	*	* I\$ERRC - 1 BYTE, FOR THE ERROR CONDITION CODE. THIS CONTAINS	*		
		6777	*	A NULL CODE (I@NERR) WHEN NO ERROR CONDITION EXISTS, OR AN	*		
		6778	*	ERROR CODE SPECIFYING THE PARTICULAR ERROR CONDITION DISCOVERED	*		
		6779	*	* AFTER ENTRY FZAMRD - ELEMENTS IN VIRTUAL MEMORY FOR THE ARITH-	*		
		6780	*	METIC ARRAY SPECIFIED BY THE STACKED DOPE VECTOR.	*		
		6781	*	* AFTER ENTRY FZAMGT - ELEMENTS IN VIRTUAL MEMORY FOR THE ARITH-	*		
		6782	*	METIC ARRAY SPECIFIED BY THE STACKED DOPE VECTOR.	*		
		6783	*	* AFTER ENTRY FZAMPT - ELEMENTS IN THE CURRENTLY ACTIVE EXTERNAL	*		
		6784	*	OUTPUT FILE FROM THE ARITHMETIC ARRAY SPECIFIED BY THE STACKED	*		
		6785	*	DOPE VECTOR.	*		
		6786	*		*		
		6787	*	*EXTERNAL REFERENCES	*		
		6788	*	* V\$XSRD - VIRTUAL ENTRY ADDRESS FOR FIREAD, 'READ' EXECUTION RTN	*		
		6789	*	* V\$XSGT - VIRTUAL ENTRY ADDRESS FOR SFGETR, 'GET' EXECUTION RTN.	*		
		6790	*	* V\$XSPT - VIRTUAL ENTRY ADDRESS FOR SFPUTR, 'PUT' EXECUT:ON RTN.	*		
		6791	*	* I\$CALL - ENTRY POINT FOR PAGING MODULE V.M. PROGRAM CALL RTN.	*		
		6792	*	* I\$RTRN - ENTRY POINT FOR PAGING MODULE V.M. RETURN CONTROL RTN.	*		
		6793	*	* I\$STCK - ENTRY POINT FOR INTERPRETER ELEMENT STACKING ROUTINE.	*		
		6794	*	* I\$USTK - ENTRY POINT FOR INTERPRETER ELEMENT UNSTACKING ROUTINE	*		
		6795	*	* I\$STAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER.	*		
		6796	*	* I\$VADR - 2 BYTES, FOR PAGING MODULE VIRTUAL ADDRESS PARAMETER.	*		
		6797	*	* I\$DMSW - 1 BYTE, FOR THE DATA MATCHING SWITCH (SEE IUSTAK).	*		
		6798	*	* I\$ERRC - 1 BYTE, FOR THE INTERPREER EXECUTION ERROR CODE.	*		
		6799	*		*		
		6800	*	*EXITS, NORMAL	*		
		6801	*	CONTROL IS ALWAYS PASSED TO THE PAGING ROUTINE AT ENTRY POINT	*		
		6802	*	I\$RTRN (IPGRTN) FOR A RETURN TO THE CALLING PROGRAM.	*		
		6803	*		*		
		6804	*	*EXITS, ERROR	*		
		6805	*	CONTROL IS PASSED TO THE PAGING ROUTINE AT ENTRY POINT I\$RTRN	*		
		6806	*	(IPGRTN) WITH PARAMETER I\$ERRC CONTAINING THE APPROPRIATE ERROR	*		
		6807	*	MESSAGE CODE.	*		
		6808	*		*		
		6809	*	*TABLES/WOK AREAS	*		
		6810	*	DOPE VECTOR WORK AREA - 8 BYTES, FOR ARRAY ELEMENT VIRTUAL	*		
		6811	*	ADDRESS CALCULATIONS INVOLVING THE ORIGINALLY STACKED ARITH-	*		
		6812	*	METIC DOPE VECTOR.	*		
		6813	*		*		
		6814	*	*ATTRIBUTES	*		
		6815	*	REUSABLE	*		
		6816	*	NATURALLY RELOCATAILE	*		

```
6817 *
6818 *CHARACTER CODE DEPENDENCY
6819 * THE OPERATION OF THIS MODULE DOES NOT DEPEND UPON A PARTICULAR
6820 * INTERNAL REPRESENTATION OF THE EXTERNAL CHARACTER SET.
6821 *
6822 *NOTES
6823 * ERROR PROCEDURES
6824 * ZAMIO UTILIZES I/O ROUTINES FZREAD, SFGETR, AND SFPUTR FOR
6825 * ACTUAL INPUT/OUTPUT OPERATIONS. WHEN AN ERROR OCCURS DURING
6826 * EXECUTION OF ONE OF THESE ROUTINES, AN APPROPRIATE ERROR CODE
6827 * IS LEFT IN INTERPRETER PARAMETER ISERRC. THIS PARAMETER IS
6828 * TESTED WHEN CONTROL IS RETURNED TO FZAMIO AFTER EACH ELEMENT
6829 * OPERATION AND, IF AN ERROR HAS BEEN DISCOVERED, CONTROL IS
6830 * RETURNED IMMEDIATELY TO THE CALLING PROGRAM.
6831 *
6832 * REGISTER USAGE
6833 * REGISTER @BR IS TO CONTAIN THE CORE PAGE BASE ADDRESS
6834 * ESTABLISHED THROUGH PAGING MODULE CONTROL FOR THE PAGE WHICH
6835 * INCLUDES FZAMIO, AND IS RESTORED THROUGH THE PAGING MODULE.
6836 * REGISTER @XR IS NOT SAVED. IT IS USED IN FZAMIO FOR GENERAL
6837 * PURPOSE INDEXING OPERATIONS.
6838 *
6839 * SAVED/RESTORED AREAS
6840 * N/A
6841 *
6842 * MODIFICATION CONSIDERATIONS
6843 * N/A
6844 *
6845 * REQUIRED MODULES
6846 * @SYSEQ - COMMON SYSTEM EQUATES.
6847 * $V$EOU - VIRTUAL MEMORY FIXED ADDRESS EQUATES.
6848 * $B$EQU - COMPILER PARAMETER AND CONSTANT EQUATES.
6849 * $I$EQU - INTERPRETER FIXED LOCATION ADDRESS EQUATES.
6850 * $I@SEQ - INTERPRETER PARAMETER EQUATES (FOR STD. REC. ONLY).
6851 * $I@LEQ - INTERPRETER PARAMETER EQUATES (FOR LONG PREC. ONLY).
6852 *
6853 * OTHER
6854 * N/A
6855 *****
```

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 347
		6857		*****	
		6858	*	START OF MATRIX FILE INPUT/OUTPUT EXECUTION MODULE	*
		6859		*****	
3E00		6860	*		
		6861		ORG *,B@LVP,0	BEGIN AT PAGE BOUNDARY
	3E00	6862		USING *,@BR	DEFINE BASE ADDR AT PAGE BOUND
		6863	*		
	3E00	6864	FZAMIO EQU *		START OF MATRIX I/O RTNS CODING
		6865	*		
		6866		*****	
		6867	*	ENTRY FZAMRD - 'MAT READ' EXECUTION ROUTINE	*
		6868		*****	
		6869	*		
	3E00	6870	FZAMRD EQU *		FZAMRD ENTRY POINT
		6871	*		
3E00 7C 3C 25		6872		MVI FZA050+@D1(,@BR),FZA200-FZAMIO	SET MATRIX LOOP FOR 'READ'
3E03 F2 87 09		6873		J FZA010	SKIP TO PERFORM 'MAT READ' FUNC
		6874	*		
		6875		*****	
		6876	*	ENTRY FZAMGT - 'MAT GET' EXECUTION ROUTINE	*
		6877		*****	
		6878	*		
	3E06	6879	FZAMGT EQU *		FZAMGT ENTRY POINT
		6880	*		
3E06 7C 59 25		6881		MVI FZA050+@D1(,@BR),FZA300-FZAMIO	SET MATRIX LOOP FOR 'GET'
3E09 F2 87 03		6882		J FZA010	SKIP TO PERFORM 'MAT SET' FUNC
		6883	*		
		6884		*****	
		6885	*	ENTRY FZAMPT - 'MAT PUT' EXECUTION ROUTINE	*
		6886		*****	
		6887	*		
	3E0C	6888	FZAMPT EQU *		FZAMPT ENTRY POINT
		6889	*		
3E0C 7C 72 25		6890		MVI FZA050+@D1(,@BR),FZA400-FZAMIO	SET MATRIX LOOP FOR 'PUT'
		6891	*		
		6892		*****	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 348
		6894		*****	
		6895	*	ENABLE DATA TYPE MATCHING FOR 'READ' AND 'GET' OPFRATIONS	*
		6896		*****	
		6897	*		
3E0F	3C 80 0BC1	6898	FZA010 MVI	I\$DMSW,@NOP	ENABLE DATA TYPE MATCHING
		6899	*		
		6900	*	ARRAY DOPE VECTOR FOR THE MATRIX TO BE AFFECTED IS CONTAINED IN	
		6901	*	THE RUN-TIME STACK - SAVE THIS DOPE VECTOR TO FREE STACK FOR DATA	
		6902	*	TRANSFER OPERATIONS.	
		6903	*		
3E13	35 02 0D4E	6904	FZA020 L	I\$STAK,@XR	LOAD THE STACK POINTER
3E17	6C 07 9A 07	6905	MVC	FZADVR(,@BR),B@LADV-1(B@LADV,@XR)	SAVE THE DOPE VECTOR
		6906	*		
		6907	*	INITIALIZE THE ?ATRIX COLUMN COUNT FOR INPUT/OUTPUT OPERATIONS	
		6908	*	WITHIN A SINGLE ROW	
		6909	*		
3E1B	5C 01 98 96	6910	FZA030 MVC	FZADM2(,@BR),FZAD2S(B@LDMN,@BR)	SET DM2 FOR COLUMN COUNT
		6911	*		
		6912	*	ADVANCE MATRIX ELEMENT POINTER TO NEXT ARRAY ELEMENT	
		6913	*		
3E1F	5E 01 9A 92	6914	FZA040 ALC	FZAVAD(,@BR),FZAPFL(@VADDR,@BR)	INCR VADDR TO NEXT ELEM
		6915	*		
		6916	*	EXECUTE ELEMENT INPUT/OUTPUT - 'READ', 'GET', OR 'PUT'	
		6917	*		
3E23	D0 87 00	6918	FZA050 B	*-*(,@BR)	LINK TO EXECUTE ELEMENT
		6919	*		
		6920	*	TEST FOR END OF THE CURRENT MATRIX ROW	
		6921	*		
3E26	5F 01 98 90	6922	FZA060 SLC	FZADM2(,@BR),FZABN1(B@LDMN,@BR)	DECR MATRIX COLUMN COUNT
3E2A	D0 84 1F	6923	BH	FZA040(,@BR)	IF NOT 0, GO PROCESS NEXT ELEM
		6924	*		
		6925	*	END OF CURRENT ROW - TEST FOR FINAL ROW OF MATRIX	
		6926	*		
3E2D	5F 01 98 90	6927	FZA070 SLC	FZADM2(,@BR),FZABN1(B@LDMN,@BR)	DECR MATRIX ROW COUNT
3E31	D0 84 1B	6928	BH	FZA030(,@BR)	IF NOT 0, GO PROCESS NEW ROW
		6929	*		
		6930	*	EXIT - DISABLE DATA TYPE MATCHING AND RETURN TO INTERPRETER	
		6931	*		
3E34	3C 87 0BC1	6932	FZA080 MVI	I\$DMSW,@UCB	DISABLE DATA TYPE MATCHING
		6933	*		
3E38	C0 87 12D3	6934	B	I\$RTRN	RETURN TO INTERPRETER
		6935	*		
		6936		*****	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 349
		6938		*****	
		6939	*	'READ' FUNCTION MATRIX ELEMENT PROCESSING ROUTINE	*
		6940		*****	
		6941	*		
		6942	*	STACK A PROGRAM DATA FILE ELEMENT FOR 'READ' OPERATION	
		6943	*		
3E3C C0 87 12B1		6944	FZA200 B	I\$CALL	LINK TO STACK ELEMENT FROM DATA
3E40 3300		3E41 6945	DC	AL(@VADDR)(V\$XSRD)	'READ' ROUTINE VADDR ENTRY PT
		6946	*		
		6947	*	TEST FOR A 'READ' OPERATION ERROR CONDITION	
		6948	*		
3E42 3D 00 0CBC		6949	FZA210 CLI	I\$ERRC,I@NERR	IF 'READ' ERROR OCCURRED ?
3E46 D0 01 34		6950	BNE	FZA080(,@BR)	* SO EXIT THE MATRIX I/O RTN
		6951	*		
		6952	*	UNSTACK THE ELEMENT TO THE DESTINATION MATRIX	
		6953	*		
3E49 35 02 0D4E		6954	FZA220 L	I\$STAK,@XR	LOAD THE STACK POINTER
3E4D 1C 01 144A 9A		6955	MVC	I\$VADR,FZAVAD(@VADDR,@BR)	SET VADDR PAGING PARAMETER
3E52 C0 87 0BB0		6956	B	I\$USTK	LINK TO UNSTACK THE ELEMENT
		6957	*		
3E56 D0 87 26		6958	B	FZA060(,@BR)	RETURN TO MATRIX LOOP
		6959	*		
		6960		*****	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 350
		6962		*****	
		6963	*	'GET' FUNCTION MATRIX ELEMENT PROCESSING ROUTINE	*
		6964		*****	
		6965	*		
		6966	*	STACK A USER INPUT FILE ELEMENT FOR 'GET' OPERATION	
		6967	*		
3E59 C0 87 12B1		6968	FZA300 B	I\$CALL	LINK TO STACK ELEMENT FROM FILE
3E5D 2100		3E5E 6969	DC	AL(@VADDR)(V\$XSGT)	'GET' ROUTINE VADDR ENTRY PT
		6970	*		
		6971	*	TEST FOR A 'GET' OPERATION ERROR CONDITION	
		6972	*		
3E5F 3D 00 0CBC		6973	FZA310 CLI	I\$ERRC,I@NERR	IF 'GET' ERROR OCCURRED ?
3E63 D0 01 34		6974	BNE	FZA080(,@BR)	* GO EXIT THE MATRIX I/O RTN
		6975	*		
		6976	*	UNSTACK THE ELEMENT TO THE DESTINATION MATRIX	
		6977	*		
3E66 35 02 0D4E		6978	FZA320 L	I\$STAK,@XR	LOAD THE STACK POINTER
3E6A 1C 01 144A 9A		6979	MVC	I\$VADR,FZAVAD(@VADDR,@BR)	SET VADDR PAGING PARAMETER
		6980	*	B	I\$USTK
3E6F D0 87 26		6981	B	FZA060(,@BR)	LINK TO UNSTACK THE ELEMENT
		6982	*		RETURN TO MATRIX LOOP
		6983		*****	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 351
		6985		*****	
		6986	*	'PUT' FUNCTION MATRIX ELEMENT PROCESSING ROUTINE	*
		6987		*****	
		6988	*		
		6989	*	STACK A MATRIX ELEMENT FOR 'PUT' OPERATION	
		6990	*		
3E72	35 02 0D4E	6991	FZA400 L	I\$STAK,@XR	LOAD THE STACK POINTER
3E76	1C 01 144A 9A	6992		MVC I\$VADR,FZAVAD(@VADDR,@BR)	SET VADDR PAGING PARAMETER
3E7B	C0 87 0B50	6993		B I\$STCK	LINK TO STACK MATRIX ELEMENT
		6994	*		
		6995	*	UNSTACK THE ELEMENT TO THE USER OUTPUT FILE	
		6996	*		
3E7F	C0 87 12B1	6997	FZA410 B	I\$CALL	LINK TO 'PUT' ELEMENT TO FILE
3E83	1D00	3E84 6998		DC AL(@VADDR)(V\$XSPT)	'PUT' ROUTINE VSADDR ENAN PT
		6999	*		
		7000	*	TEST FOR A 'PUT' OPERATION ERROR CONDITION.	
		7001	*		
3E85	3D 00 0CBC	7002	FZA420 CLI	I\$ERRC,I@NERR	IF 'PUT' ERROR OCCURED ?
3E89	D0 01 34	7003		BNE FZA080(,@BR)	* GO EXIT THE MATRIX I/O RTN
		7004	*		
3E8C	D0 87 26	7005		B FZA060(,@BR)	GETURN TO !ARTIX LOW
		7006	*		
		7007		*****	

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 352
		7009		*****	
		7010		* MATRIX FILE INPUT/OUTPUT ROUTINE CONSTANTS	*
		7011		*****	
		7012		*	
3E8F 0001		3E90 7013	FZABN1 DC	IL(B@LDMN)'1'	BINARY INTEGER +1
3E91 0005		3E92 7014	FZAPFL DC	AL(@VADDR)(I@LPFV)	LENGTH OF PACKED FLT PT VALUE
		7015		*	
		7016		*****	
		7017		* MATRIX FIFE INPUT/OUTPUT ROUTINR WORK AREAS	*
		7018		*****	
		7019		*	
		3E93 7020	FZADVA EQU	*	MATRIX DOPE VECTOR SAVE ADDR
3E93		3E9A 7021	FZADVR DS	CL(B@LADV)	MATRIX DOPE VECTOR SAVE AREA
		7022		*	
		3E94 7023	FZA041 EQU	FZADVA+B@ACD1	DOPE VECTOR ROW DIMENSION
		3E96 7024	FZAD2S EQU	FZADVA+B@ACD2	DOPE VECTOR COLUMN DIMENSION
		3E98 7025	FZADM2 EQU	FZADVA+B@AMAX	COLUMN DIMENSION COUNT AREA
		3E9A 7026	FZAVAD EQU	FZADVA+B@ABAS	MATRIX ELEMENT VIRTUAL ADDR
		7027		*	
		7028		*****	
		7029		*	
		7030		*** END OF MATRIX FILE INPUT/OUTPUT ROUTINE IOCS	*****

ERR LOC	OBJECT CODE	ADDR STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 353
		7032	*****			*
		7033	* 5703-XM1 COPYRIGHT IBM CORP. 1970			*
		7034	* REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
		7035	*			*
		7036	*****			*
		7037	*STATUS			*
		7038	* VERSION 1 MODIFICATION 0			*
		7039	*			*
		7040	*FUNCTION			*
		7041	* * FZCMR CONTAINS THE RUN-TIME ROUTINES WHICH EXECUTE MATRIX			*
		7042	* OPERATIONS FOR AN ARRAY REFERENCED IN A BASIC PROGRAM			*
		7043	* 'MAT PRINT' OR 'MAT PRINT USING' STATEMENT.			*
		7044	* * THIS ROUTINE PERFORMS 'PRINT' (FULL ZONE FORMAT). 'PRINT'			*
		7045	* (PACKED ZONE FORMAT), OR 'PRINT USING' OPERATIONS FOR EACH			*
		7046	* ELEMENT OF THE MATRIX REFERENCED BY THE ARITHMETIC DOPE VECTOR			*
		7047	* AT THE TOP OF THE RUN-TIME STACK.			*
		7048	* * 'PRINT' (FULL ZONE FORMAT) - SUCCESSIVE ELEMENTS FROM THE			*
		7049	* REFERENCED MATRIX ARE PRINTED ON A ROW-BYROW BASIS, ON			*
		7050	* THE SYSTEM PRINT DEVICE. EACH ELEMENT IS PRINTED AS			*
		7051	* DEFINED FOR FULL ZONE OUTPUT IN FUNCTIONAL SPECIFICATIONS.			*
		7052	* * 'PRINT' (PACKED ZONE FORMAT) - SUCCESSIVE ELEMENTS FROM			*
		7053	* THE REFERENCED MATRIX ARE PRINTED, ON A ROW-BY-ROW BASIS,			*
		7054	* ON THE SYSTEM PRINT DEVICE. EACH ELEMENT IS PRINTED AS			*
		7055	* DEFINED FOR PACKED ZONE OUTPUT IN THE FUNCTIONAL SPECS.			*
		7056	* * 'PRINT USING' - SUCCESSIVE ELEMENTS FROM THE REFERENCED			*
		7057	* MATRIX ARE PRINTED ON A ROW-BY-ROW BASIS, ON THE SYSTEM			*
		7058	* PRINT DEVICE. EACH ELEMENT UTILIZES THE NEXT AVAILABLE			*
		7059	* CONVERSION SPECIFICATION IN THE CURRENTLY ACTIVE IMAGE.			*
		7060	* * THE PRINT DEVICE CARRIER/CURSOR IS POSITIONED, PRIOR TO OUTPUT			*
		7061	* BETWEEN THE FIRST MATRIX ROW AND THE PREVIOUS PRINTED LINE.			*
		7062	* OF THE FIRST ARRAY ELEMENT, SUCH THAT TWO BLANK LINES EXIST			*
		7063	* EACH MATRIX ROW IS SEPARATED FROM THE PRIOR ROW WIRH A BLANK			*
		7064	* LINE, AND THE CARRIER/CURSOR IS RETURNED FOLLOWING OUTPUT OF			*
		7065	* THE FINAL MATRIX ROW.			*
		7066	*			*
		7067	*ENTRY POINTS			*
		7068	* * ENTRY FZCMPS - FOR PERFORMING THE 'MAT PRINT' (PACKED ZONE)			*
		7069	* FUNCTION. CALLING SEQUENCE IS			*
		7070	* B IPGCAL			*
		7071	* DC AL2(V\$XMPS)			*
		7072	* WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL			*
		7073	* ADDRESS OF ENTRY POINT FZCMPS.			*
		7074	* * ENTRY FZCMPL - FOR PERFORMING THE 'MAT PRINT' (LONG ZONE)			*
		7075	* FUNCTION. CALLING SEQUENCE IS			*
		7076	* B IPGCAL			*
		7077	* DC AL2(V\$XMPS)			*
		7078	* WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL			*
		7079	* ADDRESS OF ENTRY POINT FZCMPS.			*
		7080	* * ENTRY FZCMPU - FOR PERFORMING THE 'MAT PRINT USING' FUNCTION.			*
		7081	* CALLING SEQUENCE IS			*
		7082	* B IPGCAL			*
		7083	* DC AL2(V\$XMPU)			*
		7084	* WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL			*
		7085	* ADDRESS OF ENTRY POINT FZSMPU.			*
		7086	* * ENTRY POINT EXECUTION FOR THESE OPERATIONS IS SUBJECT TO INPUT			*
		7087	* CONDITIONS DESCRIBED BELOW.			*

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 354
		7088	*				*
		7089	*INPUT				*
		7090	*	* I\$STAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER. THIS CON-			*
		7091	*	CONTAINS CORE ADDRESS OR THE LEFTMOST BYTE OF THE ARITHMETIC			*
		7092	*	ARRAY DOPE FACTOR AT THE TOP OF THE STACK.			*
		7093	*	* RUN-TIME STACK - THIS CONTAINS THE ARITHMETIC ARRAY DOPE VECTOR			*
		7094	*	(ASSOCIATED WITH THE MATRIX TO BE PRINTED) IN THE TOP STACK			*
		7095	*	POSITION.			*
		7096	*	* VIRTUAL MEMORY - THIS CONTAINS ELEMENTS FOR THE ARITHMETIC			*
		7097	*	ARRAY SPECIFIED BY THE STACKED DOPE VECTOR.			*
		7098	*	* \$PRPOS - 1 BYTE, FOR THE MATRIX PRINTER CARRIER POSITION			*
		7099	*	INDICATOR. THIS CONTAINS THE CARRIER POSITION, RELATIVE TO THE			*
		7100	*	HARDWARE LEFT MARGIN AS 0, OF THE MATRIX PRINTER CARRIER.			*
		7101	*	* \$LMRGN - 1 BYTE, FOR THE MATRIX PRINTER SOFTWARE LEFT MARGIN			*
		7102	*	INDICATOR.			*
		7103	*	* \$CRPOS - 1 BYTE, FOR THE CRT CURSSOR POSITION INDICATOR. THIS			*
		7104	*	CONTAINS THE CURSOR POSITION, RELATIVE TO THE LEFT CRT MARGIN			*
		7105	*	AS 0, OF THE CRT CURSOR.			*
		7106	*				*
		7107	*OUTPUT				*
		7108	*	* I\$ERRC - 1 BYTE, FOR THE ERROR CONDITION CODE. THIS CONTAINS			*
		7109	*	A NULL CODE (I@NERR) WHEN NO ERROR CONDITION EXISTS, OR AN			*
		7110	*	ERROR CODE SPECIFYING THE PARTICULAR ERROR COND DISCOVERED.			*
		7111	*	* \$PRPOS - 1 BYTE, FOR THE MATRIX PRINTER CARRIER POSITION			*
		7112	*	INDICATOR. WHEN PRINTING IS COMPLETED, THE CARRIER IS RETURNED			*
		7113	*	TO THE SOFTWARE LEFT MARGIN, CAUSING \$PRPOS TO BE SET EQUAL TO			*
		7114	*	\$LMRGN.			*
		7115	*	* \$CRPOS - 1 BYTE, FOR THE CRT CURSOR POSITION INDICATOR. WHEN			*
		7116	*	PRINTING IS COMPLETED. THE CURSOR IS RETURNED TO THE CRT LEFT			*
		7117	*	MARGIN, CAUSING \$CRPOS TO BE SET EQUAL ZERO.			*
		7118	*	* PRINTED OUTPUT - AS SPECIFIED UNDER 'FUNCTION'. PRINTING			*
		7119	*	OCCURS ON THE SYSTEM MATRIX PRINTER AND/OR CRT, DEPENDING ON			*
		7120	*	THE CURRENT STATUS OF PRINT DEVICE PARAMETER \$PRDEV.			*
		7121	*				*
		7122	*EXTERNAL REFERENCES				*
		7123	*	* V\$XSPR - VIRTUAL ENTRY ADDRESS FOR FZSPRT. 'PRINT' EXEC RTN.			*
		7124	*	* V\$XSPU - VIRTUAL ENTRY ADDRESS FOR FZUPRT. 'PRINT USING' RTN.			*
		7125	*	* I\$CALL - ENTRY POINT FOR PAGING MODULE V.M. PROGRAM CALL RTN.			*
		7126	*	* I\$RTRN - ENTRY POINT FOR PAGING MODULE V.M. RETURN CONTROL RTN.			*
		7127	*	* I\$STCK - ENTRY POINT FOR INTERPRETER ELEMENT STACKING ROUTINE.			*
		7128	*	* I\$CPUF - ENTRY POINT FOR FLOATING POINT VALUE UNPACKING ROUTINE.			*
		7129	*	* I\$STAK - 2 BYTES, FOR THE RUN-TIME STACK POINTER.			*
		7130	*	* I\$VADR - 2 BYTES, FOR PAGING MODULE VIRTUAL ADDRESS PARAMETER.			*
		7131	*	* I\$PARM - 2 BYTES, FOR THE INTERPRETER COMMUNICATIONS PARAMETER.			*
		7132	*	* I\$INDR - 1 BYTE, FOR THE 'PRINT USING' STATUS INDICATOR BYTE.			*
		7133	*	* I\$PROS - 1 BYTE, FOR MATRIX PRINTER CARRIER POSITION INDICATOR.			*
		7134	*	* \$LMTGN - 1 BYTE, FOR POSITION OF MATRIX PRINTER SOFT LEFT MRGN.			*
		7135	*	* \$CRPOS - 1 BYTE, FOR THE CRT CURSOR POSITION INDICATOR.			*
		7136	*	* I\$ERRC - 1 BYTE, FOR THE INTERPRETER EXECUTION ERROR CODE.			*
		7137	*				*
		7138	*EXIT, NORMAL				*
		7139	*	CONTROL IS ALWAYS PASSED TO THE PAGING ROUTINE AT ENTRY POINT			*
		7140	*	I\$RTRN (IPGRTN) FOR A RETURN TO THE CALLING PROGRAM.			*
		7141	*				*
		7142	*EXIT, ERROR				*
		7143	*	CONTROL IS PASSED TO THE PAGING ROUTINE AT ENTRY POINT I\$RTRN			*

ERR LOC	OBJECT CODE	ADDR STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 355
		7144 *	(IPGRTN) WITH PARAMETER I\$ERRC CONTAINING THE APPROPRIATE ERROR	*		
		7145 *	MESSAGE CODE.	*		
		7146 *		*		
		7147 *	*TABLES/WORK AREAS	*		
		7148 *	DOPE VECTOR WORK AREA - 8 BYTES, FOR ARRAY ELEMENT VIRTUAL	*		
		7149 *	ADDRESS CALCULATIONS INVOLVING THE ORIGINALLY STACKED ARITH-	*		
		7150 *	METIC DOPE VECTOR.	*		
		7151 *		*		
		7152 *	*ATTRIBUTES	*		
		7153 *	* REUSABLE	*		
		7154 *	* NATURALLY RELOCATABLE	*		
		7155 *		*		
		7156 *	*CHARACTER CODE DEPENDENCY	*		
		7157 *	THE OPERATION OF THIS MODULE DOES NOT DEPEND UPON A PARTICULAR	*		
		7158 *	INTERNAL REPRESENTATION OF THE EXTERNAL CHARACTER SET.	*		
		7159 *		*		
		7160 *	*NOTES	*		
		7161 *	ERROR PROCEDURES	*		
		7162 *	FZEMPR UTILIZES PRINT EXECUTION ROUTINES FZSPRT AND FZUPRT FOR	*		
		7163 *	ACTUAL PRINT OPERATIONS. WHEN AN ERROR OCCURS DURING EXECU-	*		
		7164 *	TION OF ONE OF THESE ROUTINES, AN APPROPRIATE ERROR CODE IS	*		
		7165 *	LEFT IN INTERPRETER PARAMETER I\$ERRC. THIS PARAMETER IS	*		
		7166 *	TESTED WHEN CONTROL IS RETURNED TO FZCMR AFTER EACH ELEMENT	*		
		7167 *	OPERATION AND, IF AN ERROR HAS BEEN DISCOVERED, CONTROL IS	*		
		7168 *	RETURNED IMMEDIATELY TO THE CALLING PROGRAM.	*		
		7169 *		*		
		7170 *	REGISTER USAGE	*		
		7171 *	* REGISTER @BR IS TO CONTAIN THE CORE PAGE BASE ADDRESS	*		
		7172 *	ESTABLISHED THROUGH PAGING MODULE CONTROL FOR THE PAGE WHICH	*		
		7173 *	INCLUDES FZCMR, AND IS RESTORED THROUGH THE PAGING MODULE.	*		
		7174 *	* REGISTER @XR IS NOT SAVED. IT IS USED IN FZCMR FOR GENERAL	*		
		7175 *	PURPOSE INDEXING OPERATIONS.	*		
		7176 *		*		
		7177 *	SAVED/RESTORED AREAS	*		
		7178 *	N/A	*		
		7179 *		*		
		7180 *	MODIFICATION CONSIDERATIONS	*		
		7181 *	N/A	*		
		7182 *		*		
		7183 *	REDUIRED MODULES	*		
		7184 *	* @SYSEQ - COMMON SYSTEM EQUATES.	*		
		7185 *	* @FXDEQ - SYSTEM NUCLEUS ADDRESSES AND INDICATOR EQUATES.	*		
		7186 *	* \$V\$EQU - VIRTUAL MEMORY FIXED ADDRESS EQUATES.	*		
		7187 *	* \$B\$EQU - COMPILER PARAMETER AND CONSTANT EQUATES.	*		
		7188 *	* \$I\$EQU - INTERPRETER FIXED LOCATION ADDRESS EQUATES.	*		
		7189 *	* \$I@SEQ - INTERPRETER PARAMETER EQUATES (FOR STD. PREC. ONLY)	*		
		7190 *	* \$I@LEQ - INTERPRETER PARAMETER EQUATES (FOR LONG PREC. ONLY)	*		
		7191 *		*		
		7192 *	OTHER	*		
		7193 *	N/A	*		
		7194 *	*****	*		

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 356
		7196		*****	
		7197	*	START OF MATRIX PRINT EXECUTION MODULE	*
		7198		*****	
3F00		7199	*		
		7200		ORG *,B@LVPG,0	BEGIN AT PAGE BOUNDARY
	3F00	7201		USING *,@BR	DEFINE BASE ADDR AT PAGE BOUND
		7202	*		
	3F00	7203	FITMPQ EQU *		START OF 'MAT PRINT' RTN CODING
		7204		*****	
		7205	*	ENTRY FZCMPS - 'MAT PRINT' (SHORT ZONE) EXECUTION ROUTINE	*
		7206		*****	
		7207	*		
		3F00 7208	FZCMPS EQU *		FZCMPS ENTRY POINT
		7209	*		
3F00 7C 03 71		7210		MVI FZC080+@Q(,@BR),B@PRPS	SET SHORT PRINT ZONE OUTPUT
3F03 F2 87 03		7211		J FZC005	SKIP TO COMPLETE INITIALIZATION
		7212	*		
		7213		*****	
		7214	*	ENTRY FZCMPL - 'MAT PRINT' (LONG ZONE) EXECUTION ROUTINE	*
		7215		*****	
		7216	*		
		3F06 7217	FZCMPL EQU *		FZCMPL ENTRY POINT
		7218	*		
3F06 7C 02 71		7219		MVI FZC080+@Q(,@BR),B@PRPL	SET LONG PRINT ZONE OUTPUT
		7220	*		
3F09 7C 04 7C		7221	FZC005 MVI	FZC085+@Q(,@BR),B@PRPR	SET END-OF-ROW PRINT PARAMETER
3F0C 5C 01 84 A3		7222		MVC FZC095(,@BR),FZCAPR(@VADDR,@BR)	SET 'PRINT' RTN VADDR
3F10 F2 87 0A		7223		J FZC010	BRANCH TO PERFORM 'MAT PRINT'
		7224	*		
		7225		*****	
		7226	*	ENTRY FZCMPL 'MAT PRINT USING' EXECUTION ROUTINE	*
		7227		*****	
		7228	*		
		3F13 7229	FZCMPU EQU *		FZCMPU ENTRY POINT
		7230	*		
3F13 7C 06 71		7231		MVI FZC080+@Q(,@BR),B@PUD1	SET ARITH ELEMENT OUTPUT PARAM
3F16 7C 36 7C		7232		MVI FZC085+@Q(,@BR),B@PUD1+B@PUTM+B@PURE	SET END-OF-ROW PARAM
3F19 5C 01 84 A5		7233		MVC FZC095(,@BR),FZCAPU(@VADDR,@BR)	SET 'PRINT USING' VADDR
		7234	*		
		7235		*****	

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 357
					7237	*****				
					7238	*	ARRAY DOPE VECTOR FOR MATRIX TO BE PRINTED IS CONTAINED IN THE			*
					7239	*	RUN-TIME STACK - SAVE THIS DOPE VECTOR TO FREE STACK FOR ELEMENT			*
					7240	*	PRINT OPERATIONS.			*
					7241	*****				
					7242	*				
3F1D	35	02	0D4E		7243	FZC010	L I\$STAK,@XR			LOAD THE STACK POINTER
3F21	6C	07	AD 08		7244		MVC FZCDVR(,@BR),B@LADV(B@LADV,@XR)			SAVE THE DOPE VECTOR
					7245	*				
					7246	*	SPACE A LINE FOLLOWING LAST PRINTED LINE			
					7247	*				
3F25	0D	00	03C2 03C1		7248	FZC020	CLC \$PRPOS,\$LMRGN(1)			IF CARRIER NOT AT LEFT MARGIN
3F2B	F2	01	07		7249		JNE FZC025			* SKIP TO RETURN THE CARRIER
3F2E	3D	00	03E2		7250		CLI \$CRPOS,@ZERO			IF CURSOR IS AT LEFT MARGIN
3F32	F2	81	0A		7251		JE FZC030			* SKIP INITIAL CURSOR RETURN
					7252	*				
3F35	3C	07	0D57		7253	FZC025	MVI I\$PARM,B@PRRC			SET PRINT RTN FOR CARR RETURN
3F39	C0	87	12B1		7254		B I\$CALL			LINK TO RETURN THE CARRIER
3F3D	3400			3F3E	7255		DC AL(@VADDR)(V\$XSPR)			PRINT ROUTINE VADDR ENTRY PT
					7256	*				
3F3F	3C	07	0D57		7257	FZC030	MVI I\$PARM,B@PRRC			SET PRINT RTN FOR CARR RETURN
3F43	C0	87	12B1		7258		B I\$CALL			LINK TO SPACE DOWN ONE LINE
3F47	3400			3F48	7259		DC AL(@VADDR)(V\$XSPR)			PRINT ROUTINE VADDR ENTRY PT
					7260	*				
					7261	* INITIALIZE TO PROCESS A NEW MATRIX ROW - SPACE A LINE AND SET MATRIX				
					7262	* COLUMN COUNT FOR PRINT OPERATIONS FROM A SINGLE ROW				
					7263	*				
3F49	3C	07	0D57		7264	FZC040	MVI I\$PARM,B@PRRC			SET PRINT RTN FOR CARR RETURN
					7265	*				
3F4D	C0	87	12B1		7266		B I\$CALL			LINK TO SPACE DOWN ONE LINE
3F51	3400			3F52	7267		DC AL(@VADDR)(V\$XSPR)			PRINT ROUTINE VADDR ENTRY PT
					7268	*				
3F53	5C	01	AB A9		7269		MVC FZCDM2(,@BR),FZCD2S(B@LDMN,@BR)			NSSET DM2 COLUMN COURT
3F57	3B	20	0DC5		7270		SBF I\$INDR,I\$B2SW			RESTART IMAGE FOR 'PRINT USING'
					7271	*				
					7272	* ADVANCE MATRIX ELEMENT POINTRQ TO %EAT ARRAY ELEMENT				
					7273	*				
3F5B	5E	01	AD A1		7274	FZC060	ALC FZCVAD(,@BR),FZCPFL(@VADDR,@BR)			INCR VADDR TO NEXT ELEM
					7275	*				
					7276	* STACK AND UNPACK NE CURRENT MATRIX ELEMENT				
					7277	*				
3F5F	35	02	0D4E		7278	FZC070	L I\$STAK,@XR			LOAD THE STACK DOWER
3F63	1C	01	144A AD		7279		MVC I\$VADR,FZCVAD(@VADDR,@BR)			SET VADDR PAGING PARAMETER
3F68	C0	87	0B50		7280		B I\$STCK			LINK TO STACK MATRIX ELEMENT
3F6C	C0	87	0A27		7281		B I\$CPUF			LINK TO UNPACK MATRIX ELEMENT
					7282	*				
					7283	* PRINT THE MATRIX ELEMENT USING SPECIFIED ZONE FORMAT OR ACCORDING				
					7284	* NEXT AVAILABLE IMAGE CONVERSION SPECIFICATION - WHEN THIS IS FINAL				
					7285	* ELEMENT IN CURRENT ROW, PRINTOUT IS FOLLOWED WITH A CARRIER RETURN				
					7286	*				
3F70	3C	00	0D57		7287	FZC080	MVI I\$PARM,*-*			SET PRINT OUTPUT PARAMETER
					7288	* SPECIFIED BY THE ENTRY POINT				
3F74	5D	01	AB 9F		7289		CLC FZCDM2(,@BR),FZCBN1(B@LDMN,@BR)			IF NOT FINAL ELEM IN ROW
3F78	F2	01	04		7290		JNE FZC090			* SKIP TO PRINT W/O CARR RETURN
3F7B	3C	00	0D57		7291	FZC085	MVI I\$PARM,*-*			* ELSE SET PRINT AND CARR RETURN
					7292	*				

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 358
3F7F	C0 87 12B1		7293	FZC090 B	I\$CALL			LINK TO PRINT THE ELEMENT
			7294	*				
3F83		3F84	7295	FZC095 DS	CL(@VADDR)			PRINT ROUTINE VADDR ENTRY POINT
			7296	*				
3F85	3D 00 0CBC		7297		CLI I\$ERRC,I@NERR			IF A PRINT ERROR OCCURRED ?
3F89	F2 01 0E		7298		JNE FZC120			* GO EXIT TO THE INTERPRETER
			7299	*				
			7300	*	TEST FOR END OF THE CURRENT MATRIX ROW			
			7301	*				
3F8C	5F 01 AB 9F		7302	FZC100 SLC	FZCDM2(,@BR),FZCBN1(B@LDMN,@BR)			DECR MATRIX COLUMN COUNT
3F90	D0 84 5B		7303		BH FZC060(,@BR)			IF NOT 0, GO PROCESS NEXT ELEM
			7304	*				
			7305	*	END OF CURRENT ROW - TEST FOR FINAL ROW OF MATRIX			
			7306	*				
3F93	5F 01 A7 9F		7307	FZC110 SLC	FZCDM1(,@BR),FZCBN1(B@LDMN,@BR)			DECR MATRIX ROW COUNT
3F97	D0 84 49		7308		BH FZC040(,@BR)			IF NOT 0, GO PROCESS NEW ROW
			7309	*				
			7310	*	EXIT - RETURN TO INTERPRETER MATRIX FUNCTION CALL ROUTINE			
			7311	*				
3F9A	C0 87 12D3		7312	FZC120 B	I\$RTRN			RETURN TO INTERPRETER
			7313	*				
			7314	*	*****			

ERR	LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 359
			7316		*****	
			7317		* MATRIX PRINT ROUTINE CONSTANTS *	
			7318		*****	
			7319		*	
3F9E	0001		3F9F	7320	FZCBN1 DC IL(B@LDMN)'1' BINARY INTEGER +1	
3FA0	0005		3FA1	7321	FZCPFL DC AL(@VADDR)(I@LPFV) LENGTH OF PACKED FLT PT VALUE	
			7322		*	
3FA2	3400		3FA3	7323	FZCAPR DC AL(@VADDR)(V\$XSPR) PRINT ROUTINE VIRTUAL ADDRESS	
3FA4	3800		3FA5	7324	FZCAPU DC AL(@VADDR)(V\$XSPU) PRINT USING RTN VIRTUAL ADDR	
			7325		*	
			7326		*****	
			7327		* MATRIX PRINT ROUTINE WORK AREAS *	
			7328		*****	
			7329		*	
			3FA6	7330	FZCDVA EQU * MATRIX DOPE VECTOR SAVE AREA	
3FA6			3FAD	7331	FZCDVR DS CL(B@LADV) MATRIX DOPE VECTOR SAVE AREA	
			7332		*	
			3FA7	7333	FZCDM1 EQU FZCDVA+B@ACD1 DOPE VECTOR ROW DIMENSION	
			3FA9	7334	FZCD2S EQU FZCDVA+B@ACD2 DOPE VECTOR COLUMN DIMENSION	
			3FAB	7335	FZCDM2 EQU FZCDVA+B@AMAX COLUMN DIMENSION COUNT AREA	
			3FAD	7336	FZCVAD EQU FZCDVA+B@ABAS MATRIX ELEMENT VIRTUAL ADDR	
			7337		*	
			7338		*****	
			7339		*	
			7340		*** END OF MATRIX PRINT ROUTINE CODING *****	

FEBMAD - MATRIX ADD & SUBTRACT - PROLOGUE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 360
		7342		*****	
		7343	*	5703-XM1 COPYRIGHT IBM CORP. 1970	*
		7344	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083	*
		7345	*		*
		7346		*****	
		7347	*	*STATUS	*
		7348	*	VERSION 1 MODIFICATION 0	*
		7349	*		*
		7350	*	*FUNCTION	*
		7351	*	FEBMAD DOES ERROR CHECKING AND PERFORMS CALCULATION FOR MAT A=B+C	*
		7352	*	AND MAT A=B-C	*
		7353	*		*
		7354	*	*ENTRY POINTS	*
		7355	*	FEBMAD--MAT A=B+C	*
		7356	*	B IPGCAL	*
		7357	*	DC AL2(FEBMAD)	*
		7358	*	FEBMSB--MAT A=B-C	*
		7359	*	B IPGCAL	*
		7360	*	DC AL2(FEBMSB)	*
		7361	*	*INPUT	*
		7362	*	DOPE VECTORS FOR A, B, AND C. IN THAT ORDER, ARE IN THE RUN TIME	*
		7363	*	STACK WITH I\$STAK POINTING TO THE LEFT BYTE OF THE DOPE VECTOR	*
		7364	*	FOR MATRIX A	*
		7365	*		*
		7366	*	*OUTPUT	*
		7367	*	N/A	*
		7368	*		*
		7369	*	*EXTERNAL REFERENCES	*
		7370	*	I\$STAK	*
		7371	*	I\$VADR	*
		7372	*	I\$STCK	*
		7373	*	I\$CPUF	*
		7374	*	I\$ERRC	*
		7375	*	I\$CUPF	*
		7376	*	I\$USTK	*
		7377	*	I\$RTRN	*
		7378	*	I\$FSUB	*
		7379	*	I\$FADD	*
		7380	*		*
		7381	*	*EXITS, NORMAL	*
		7382	*	I\$RTRN WITH I\$ERRC SET TO I@NERR	*
		7383	*		*
		7384	*	*EXITS, ERROR	*
		7385	*	I\$RTRN WITH I\$ERRC SET TO THE APPROPRIATE ERROR CODE	*
		7386	*		*
		7387	*	*TABLES/WORKAREAS	*
		7388	*	N/A	*
		7389	*		*
		7390	*	*ATTRIBUTES	*
		7391	*	* NATURALLY RELOCATABLE	*
		7392	*	* REUSABLE	*
		7393	*		*
		7394	*	*CHARACTER CODE DEPENDENCY	*
		7395	*	N/A	*
		7396	*		*
		7397	*	*NOTES	*

FEBMAD - MATRIX ADD & SUBTRACT - PROLOGUE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 361
		7398	*	ERROR PROCEDURES			*
		7399	*	@@E762 - OPERANDS NOT EQUAL			*
		7400	*	@@E755 - ASSIGNED MATRIX NOT EQUAL			*
		7401	*	* UNDERFLOW AND OVERFLOW SET BY ARITHMETIC ROUTINES			*
		7402	*				*
		7403	*	REGISTER USAGE			*
		7404	*	* REGISTER 1 (@BR) IS SET BY THE PAGING MODULE TO THE FIRST			*
		7405	*	BYTE OF THE PAGE. IT IS USED FOR BASE ADDRESSING ONLY.			*
		7406	*	* REGISTER 2 (@XR) IS LOADED FROM I\$STAK AND USED AS A POINTER			*
		7407	*	TO THE RUN TIME STACK.			*
		7408	*				*
		7409	*	REQUIRED MODULES			*
		7410	*	* @SYSEQ			*
		7411	*	* \$I\$EQU			*
		7412	*	* \$B@EQU			*
		7413	*	* \$I@EQU			*
		7414	*	*****			*

FEBMAD/FEBMSB - VM MAT ADD & SUB ROUTINES

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 362
					7416	*				
					7417	*	ESTABLISH ADDRESSABILITY			
					7418	*				
4000					7419		ORG *,B@LVPG,0			
				4000	7420		USING *,@BR			
					7421	*				
					7422	*	ENTRY POINT FOR MAT C = A - B			
					7423	*				
				4000	7424	FEBMSB EQU	* MAT SUBTRACT ENTRY POINT			
4000	5C	01	5E	99	7425	MVC	FEB400+@OP1(@CADDR,@BR),FEBSAD(,@BR) SET SUBTRACT SWITCH			
4004	F2	87	04		7426	J	FEB020 GO TO ADD			
					7427	*				
					7428	*	ENTRY POINT FOR MAT C = A + B			
					7429	*				
				4007	7430	FEBMAD EQU	* MAT ADD ENTRY POINT			
4007	5C	01	5E	9B	7431	MVC	FEB400+@OP1(@CADDR,@BR),FEBAAD(,@BR) SET ADD SWITCH			
400B	35	02	0D	4E	7432	FEB020 L	I\$STAK,@XR FETCH STACK POINTER			
					7433	*				
					7434	*	CHECK CONFORMABILITY OF MATRICES			
					7435	*				
400F	7C	D3	85		7436	MVI	FEB450+@Q(,@BR),@@E762 OPERANDS NOT EQUAL ERROR CODE			
4012	AD	03	0B	13	7437	CLC	B@LADV+B@ACD2(B@LDMN*2,@XR),B@LADV*2+B@ACD2(,@XR) BRANCH			
4016	F2	01	6B		7438	JNE	FEB450 * IF EXPRIS!ON MATS NOT EQUAL			
4019	7C	CC	85		7439	MVI	FEB450+@Q(,@BR),@@E755 ASS MAT NOT EQUAL ERROR CODE			
401C	AD	03	03	0B	7440	CLC	B@ACD2(B@LDMN*2,@XR),B@LADV+B@ACD2(,@XR) BRANCH IF			
4020	F2	01	61		7441	JNE	FEB450 ASSIGNMENT MAT NOT EQUAL			
					7442	*				
					7443	*	SAVE VIRTUAL ADDRESSES AND DIMENSIONS			
					7444	*				
4023	6C	01	91	07	7445	FEB100 MVC	FEBVAC(@VADDR,@BR),B@ABAS(,@XR) FETCH VA OF A			
4027	6C	01	8D	0F	7446	MVC	FEBVAA(@VADDR,@BR),B@LADV+B@ABAS(,@XR) FETCH VA OF B			
402B	6C	01	8F	17	7447	MVC	FEBVAB(@VADDR,@BR),B@LADV*2+B@ABAS(,@XR) FETCH VA OF C			
					7448	*				
402F	6C	03	97	03	7449	MVC	FEBDM3(B@LDMN*2,@BR),B@ACD2(,@XR) FETCH DIMENSIONS			
					7450	*				
					7451	*	INITIALIZE LARGE LOOP TO ROW COUNT			
					7452	*				
4033	5C	01	93	95	7453	FEB200 MVC	FEBDM1(B@LDMN,@BR),FEBDM2(,@BR) INITIALIZE ROW COUNT			
					7454	*				
					7455	*	LOOP TO HANDLE A ROW			
					7456	*				
4037	5E	05	91	A1	7457	FEB300 ALC	FEBVAC(@VADDR*3,@BR),FEBIVA(,@BR) INCR VA OF ALL MATRICES			
					7458	*				
					7459	*	STACK AND UNPACK MAT A ELEMENT			
					7460	*				
403B	1C	01	14	4A 8D	7461	MVC	I\$VADR(@VADDR),FEBVAA(,@BR) SET VA OF A FOR STACK			
4040	C0	87	0B	50	7462	B	I\$STCK CALL STACK ROUTINE			
4044	C0	87	0A	27	7463	B	I\$CPUF CALL UNPACK ROUTINE			
					7464	*				
					7465	*	STACK AND UNPACK MAT B ELEMENT			
					7466	*				
4048	E2	02	08		7467	LA	I@LUFV(,@XR),@XR INCR POINTER TO STACK ELEM TWO			
404B	1C	01	14	4A 8F	7468	MVC	I\$VADR(@VADDR),FEBVAB(,@BR) SET VA OF B FOR STACK			
4050	C0	87	0B	50	7469	B	I\$STCK CALL STACKING ROUTINE			
4054	C0	87	0A	27	7470	B	I\$CPUF CALL UNPACK ROUTINE			
					7471	*				

FEBMAD/FEBMSB - VM MAT ADD & SUB ROUTINES

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 363

			7472 *		PERFORM ARITHMETIC OPERATION AND TEST FOR ERROR
			7473 *		
4058	76	02	A3	7474	A FEBDEC(, @BR), @XR DECR POINTER
405B	C0	87	0000	7475	FEB400 B *- * CALL ADD OR SUBTRACT ROUTINE
405F	3D	00	0CBC	7476	CLI I\$ERRC, I@NERR TAKE ERROR EXIT IF OVERFLOW
4063	F2	01	22	7477	JNE FEB500 * OR UNDER FLOW HAS OCCURED
			7478 *		
			7479 *		PACK AND UNSTACK MAT C ELEMENT
			7480 *		
4066	C0	87	0A85	7481	B I\$CUPF CALL PACK ROUTINE
406A	1C	01	144A 91	7482	MVC I\$VADR(@VADDR), FEBVAC(, @BR) SET VA OF C FOR UNSTACK
406F	C0	87	0BB0	7483	B I\$USTK CALL UNSTACK ROUTINE
			7484 *		
			7485 *		RETURN TO SMALL LOOP IF ROW NOT FINISHED
			7486 *		
4073	5F	01	93 A5	7487	SLC FEBDM1(B@LDMN, @BR), FEBBN1(, @BR) DECR ROW COUNT BY ONE
4077	D0	84	37	7488	BP FEB300(, @BR) RETURN TO LOOP IF MORE ELEMENTS
			7489 *		
			7490 *		RETURN TO LARGE LOOP IF COLUMNS NOT FINISHED
			7491 *		
407A	5F	01	97 A5	7492	SLC FEBDM3(B@LDMN, @BR), FEBBN1(, @BR) DECR COLUMN COUNT BY ONE
407E	D0	84	33	7493	BP FEB200(, @BR) GOTO BIG LOOP IF MORE COLUMNS
			7494 *		
			7495 *		RETURN
			7496 *		
4081	F2	87	04	7497	J FEB500 EXIT
4084	3C	00	0CBC	7499	FEB450 MVI I\$ERRC, *- * SET ERROR CORR
4088	C0	87	12D3	7500	FEB500 B I\$RTRN RETURN

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 364
			7502	*		
			7503	*	ELEMENT ADDR AND DIMENSION COUNT WORK AREA	
			7504	*		
408C		408D	7505	FEBVAA DS	AL(@VADDR)	VA OF CURRENT MAT A ELEMENT
408E		408F	7506	FEBVAB DS	AL(@VADDR)	VA OF CURRENT MAT B ELEMENT
4090		4091	7507	FEBVAC DS	AL(@VADDR)	VA OF CURRENT MAT C ELEMENT
			7508	*		
4092		4093	7509	FEBDM1 DS	IL(B@LDMN)	NUMBER OF ROWS LEFT
4094		4095	7510	FEBDM2 DS	IL(B@LDMN)	TOTAL ROW COUNT
4096		4097	7511	FEBDM3 DS	IL(B@LDMN)	NUMBER OF COLUMNS LEFT
			7512	*		
			7513	*	CONSTANTS	
			7514	*		
4098 0751		4099	7515	FEB SAD DC	AL(@CADDR)(I\$FSUB)	ADDR OF FLOATING SUBTRACT RTN
409A 075D		409B	7516	FEB AAD DC	AL(@CADDR)(I\$FADD)	ADDR OF FLOATING ADD ROUTINE
			7517	*		
409C 000500050005		40A1	7518	FEBIVA DC	3AL(@VADDR)(I@LPFV)	CONSTANT TO INCR THREE VA
			7519	*		
40A2 FFF8		40A3	7520	FEBDEC DC	AL(@CADDR)(-I@LUFV)	DECR REGISTER BY ELEMENT LENGTH
			7521	*		
40A4 0001		40A5	7522	FEBBN1 DC	IL(B@LDMN)'1'	DECR ROW AND COLUMN COUNT
			7523	*		
			7524	*****		
			7525	*		
			7526	***	END OF FEBMAD/FEBMSB	

FMBMPY - MATRIX MULTIPLICATION

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 365
		7528		*****			
		7529	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
		7530	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
		7531	*				*
		7532		*****			*
		7533	*	STATUS			*
		7534	*	VERSION 1 MODIFICATION 0			*
		7535	*				*
		7536	*	FUNCTION			*
		7537	*	PERFORMS ERROR CHECKING AND CALCULATES MAT A=B*C			*
		7538	*				*
		7539	*	ENTRY POINTS			*
		7540	*	FMBMPY			*
		7541	*				*
		7542	*	INPUT			*
		7543	*	THE DOPE VECTORS FOR MATRICES A, B AND C IN THE TOP OF THE STACK			*
		7544	*				*
		7545	*	OUTPUT			*
		7546	*	N/A			*
		7547	*				*
		7548	*	EXTERNAL REFERENCES			*
		7549	*	I\$CALL			*
		7550	*	I\$STAK			*
		7551	*	I\$ERRC			*
		7552	*	I\$RTRN			*
		7553	*	I\$VADR			*
		7554	*	I\$STCK			*
		7555	*	I\$CPUF			*
		7556	*	I\$FMPY			*
		7557	*	I\$FADD			*
		7558	*	I\$CUPF			*
		7559	*	I\$USTK			*
		7560	*				*
		7561	*	EXITS, NORMAL			*
		7562	*	I\$RTRN WITH I\$ERRC SET TO I\$NERR			*
		7563	*				*
		7564	*	EXITS, ERROR			*
		7565	*	I\$RTRN WITH I\$ERRC SET TO ONE OF THE FOLLOWING:			*
		7566	*	@@E792			*
		7567	*	@@E753			*
		7568	*	@@E759			*
		7569	*	@@E758			*
		7570	*	@@E755			*
		7571	*	OVERFLOW OR UNDERFLOW SET BY ARITHMETIC ROUTINES			*
		7572	*				*
		7573	*	TABLES/WORK AREAS			*
		7574	*	N/A			*
		7575	*				*
		7576	*	ATTRIBUTES			*
		7577	*	* NATURALLY RELOCATABLE			*
		7578	*	* REUSABLE			*
		7579	*				*
		7580	*	CHARACTER CODE DEPENDENCY			*
		7581	*	N/A			*
		7582	*				*
		7583	*	NOTES			*

[illegible]

FMBMPY - MATRIX MULTIPLICATION

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 367
					7587	*				
					7588	*	ESTABLISH ADDRESSABILITY			
					7589	*				
4100					7590		ORG *,B@LVP,0			PAGE BOUNDARY
				4100	7591		USING *,@BR			BASE VALUE
					7592	*				
					7593	*	ENTRY POINT--CALL SYNTAX CHECKER			
					7594	*				
				4100	7595	FMBMPY EQU	*			MODULE ENTRY POINT
4100	C0	87	12B1		7596	B	I\$CALL			CALL SYNTAX
4104	4200			4105	7597	DC	AL(@VADDR)(FMBMPY+B@LVP)	*		CHECKER
4106	3D	00	0CBC		7598	CLI	I\$ERRC,I@NERR			BRANCH IF ERROR
410A	F2	01	A5		7599	JNE	FMB800	*		RETURN
					7600	*				
					7601	*	INITIALIZATION			
					7602	*				
410D	6C	17	CD 17		7603	MVC	FMBADC(B@LADV*3,@BR),B@LADV+B@LADV+B@ABAS(,@XR)			FETCH DV
4111	5C	01	B7 CD		7604	MVC	FMBABC(@VADDR,@BR),FMBADC(,@BR)			SET MAT C START ADDR
4115	5F	01	BB BB		7605	SLC	FMBCLG(B@LDMN,@BR),FMBCLG(,@BR)			INIT ROW LENGTH
4119	5E	01	BB D1		7606	FMB100 ALC	FMBCLG(B@LDMN,@BR),FMBLEL(,@BR)			INCR ROW LENGTH BY ECEM
411D	5F	01	C9 CF		7607	SLC	FMBL2C(B@LDMN,@BR),FMBONE(,@BR)			DECR ROW COUNT
4121	D0	84	19		7608	BP	FMB100(,@BR)	*		BRANCH IF MORE
4124	F2	87	08		7609	J	FMB550			START
					7610	*				
					7611	*	START LOOP 1			
					7612	*				
4127	5C	01	CD B7		7613	FMB500 MVC	FMBADC(@VADDR,@BR),FMBABC(,@BR)			RESTORE MAT C 1ST ELEM AD
412B	5C	01	C5 C3		7614	MVC	FMBADB(@VADDR,@BR),FMBACB(,@BR)			RESET MAT B CURRENT ELEM
412F	5C	01	C9 B9		7615	FMB550 MVC	FMBL2C(B@LDMN,@BR),FMBL2K(,@BR)			RESET LOOP 2 CONTROL
					7616	*				
					7617	*	START LOOP 2			
					7618	*				
4133	5C	01	C7 C1		7619	FMB600 MVC	FMBLIC(B@LDMN,@BR),FMBL1K(,@BR)			RESET LOOP 1 CONTROL
4137	5C	01	C3 C5		7620	MVC	FMBACB(@VADDR,@BR),FMBADB(,@BR)			RESET MAT B CDR
413B	5E	01	CD D1		7621	ALC	FMBADC(@VADDR,@BR),FMBLEL(,@BR)			INCR MAT C ADDR
413F	5C	01	CB CD		7622	MVC	FMBACC(@VADDR,@BR),FMBADC(,@BR)			SET CURRENT ELEM ADDR
4143	94	60	07 D2		7623	ZAZ	I@UMNR(I@PREC,@XR),FMBDZR(1,@BR)			INIT STACK POS 1
4147	BC	1E	00		7624	MVI	I@UEXP(,@XR),B@NXLO	*		TO ZERO
					7625	*				
					7626	*	START LOOP 3			
					7627	*				
414A	E2	02	08		7628	FMB700 LA	I@LUFV(,@XR),@XR			POINT TO SECOND
414D	34	02	0D4E		7629	ST	I\$STAK,@XR	*		STACK POSITION
4151	5E	01	C3 D1		7630	ALC	FMBACB(@VADDR,@BR),FMBLEL(,@BR)			INCR MAT B ELEMENT ADDR
4155	1C	01	144A C3		7631	MVC	I\$VADR,FMBACB(@VADDR,@BR)			SET PAGING PARAMETER
415A	C0	87	0B50		7632	B	I\$STCK			STACK ELEMENT
415E	C0	87	0A27		7633	B	I\$CPUF			UNPACK ELEMENT
4162	1C	01	144A CB		7634	MVC	I\$VADR,FMBACC(@VADDR,@BR)			MAT C ELEMENT ADDR
4167	E2	02	08		7635	LA	I@LUFV(,@XR),@XR			INCR POINTER TO THIRD POS
416A	C0	87	0B50		7636	B	I\$STCK			STACK ELEMENT
416E	C0	87	0A27		7637	B	I\$CPUF			UNPACK ELEMENT
4172	5E	01	CB BB		7638	ALC	FMBACC(@VADDR,@BR),FMBCLG(,@BR)			INCR MAT C BY 1 ROW
4176	C0	87	082A		7639	B	I\$FMPY			MULTIPLY STACK POS 2 AND 3
417A	76	02	D4		7640	A	FMBDEC(,@BR),@XR			DECR STACK POINTER TO DOS 1
417D	34	02	0D4E		7641	ST	I\$STAK,@XR			SET IT
4181	C0	87	075D		7642	B	I\$FADD			ADD PRODUCT TO TOTAL

FMBMPY - MATRIX MULTIPLICATION

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 368
4185	3D	00	0CBC		7643	CLI	I\$ERRC,I@NERR			BRANCH IF
4189	F2	01	26		7644	JNE	FMB800			* ERROR RETURN
					7645	*				
					7646	*	LOOP 1 END			
					7647	*				
418C	5F	01	C7 CF		7648	SLC	FMBLIC(B@LDMN,@BR),FMBONE(,@BR)			DECR LOOP CONTROL
4190	D0	84	4A		7649	BP	FMB700(,@BR)			BRANCH UNLESS FINISHED
4193	5E	01	BD D1		7650	ALC	FMBADA(@VADDR,@BR),FMBLEL(,@BR)			INCR MAT A ELEMENT ADDR
4197	1C	01	144A BD		7651	MVC	I\$VADR,FMBADA(@VADDR,@BR)			SET PAGING PARAMETER
419C	C0	87	0A85		7652	B	I\$CUPF			PACK ELEMENT
41A0	C0	87	0BB0		7653	B	I\$USTK			UNSTACK ELEMENT
					7654	*				
					7655	*	LOOP 2 END			
					7656	*				
41A4	5F	01	C9 CF		7657	SLC	FMBL2C(B@LDMN,@BR),FMBONE(,@BR)			DECR LOOP 2 CONTROL
41A8	D0	84	33		7658	BP	FMB600(,@BR)			RETURN IF MORE
					7659	*				
					7660	*	LOOP 3 END			
					7661	*				
41AB	5F	01	BF CF		7662	SLC	FMBL3C(B@LDMN,@BR),FMBONE(,@BR)			DECR LOOP 3 CONTROL
41AF	D0	84	27		7663	BP	FMB500(,@BR)			BRANCH UNLESS FINISHED
					7664	*				
					7665	*	RETURN			
					7666	*				
41B2	C0	87	12D3		7667	FMB800 B	I\$RTRN			EXIT

FMBMPY - MATRIX MULTIPLICATION

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 369
					7669	*				
					7670	*	DEMINSION AND ADDRESS SAVE AREA			
					7671	*				
41B6				41B7	7672	FMBABC DS	CL(B@LDMN)			OTH ELEMENT MAT C
41B8				41B9	7673	FMBL2K DS	CL(B@LDMN)			COL COUNT MAT A
41BA				41BB	7674	FMBCLG DS	CL(B@LDMN)			COL LENGTH MAT C
41BC				41BD	7675	FMBADA DS	CL(@VADDR)			ADDR CURRENT MAT A ELEMENT
41BE				41BF	7676	FMBL3C DS	CL(B@LDMN)			LOOP 3 CONTROL
41C0				41C1	7677	FMBL1K DS	CL(B@LDMN)			COL COUNT MAT B
41C2				41C3	7678	FMBACB DS	CL(B@LDMN)			ADDR CURRENT MAT B ELEMENT
41C4				41C5	7679	FMBADB DS	CL(@VADDR)			MAT B BASE ADDR
41C6				41C7	7680	FMBLIC DS	CL(B@LDMN)			LOOP 1 CONTROL
41C8				41C9	7681	FMBL2C DS	CL(B@LDMN)			LOOP 2 CONTROL
41CA				41CB	7682	FMBACC DS	CL(B@LDMN)			ADDR CURRENT MAT C ELEMENT
41CC				41CD	7683	FMBADC DS	CL(@VADDR)			MAT C BASE ADDR
					7684	*				
					7685	*	CONSTANTS			
					7686	*				
41CE 0001				41CF	7687	FMBONE DC	IL(B@LDMN)'1'			DECK LOOP CONTROLS
41D0 0005				41D1	7688	FMBLEL DC	AL(@VADDR)(I@LPFV)			LENGTH OF PACKED ELEMENT
41D2 F0				41D2	7689	FMBDZR DC	DL1'0'			INIT TOTAL
41D3 FFF8				41D4	7690	FMBDEC DC	AL(@REGL)(-I@LUFV)			DECR REGISTER

FMBMPY - MATRIX MULTIPLICATION

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 370
				7692		*****				
				7693	*		SECOND PAGE FMBMPY			*
				7694		*****				
4200				7695		ORG	*,B@LVP,0			PAGE BOUNDARY
				4200 7696		USING	*,@BR			BASE VALUE
				7697	*					
				7698	*		ERROR CHECKING			
				7699	*					
4200	35	02	0D4E	7700		L	I\$STAK,@XR			FETCH DOPE VECTOR ADDR
4204	7C	C9	4F	7701		MVI	FMB900+@Q(,@BR),@@E752			ASG MAT 1 DIM ERROR CODE
4207	9D	01	01 57	7702		CLC	B@ACD1(B@LDMN,@XR),FMBZER(,@BR)			BRANCH IF MAT A
420B	F2	81	40	7703		JE	FMB900			* IS ONE DIMENSIONAL
420E	7C	CA	4F	7704		MVI	FMB900+@Q(,@BR),@@E753			MAT OPERAND 1 DIM ERROR CODE
4211	9D	01	09 57	7705		CLC	B@LADV+B@ACD1(B@LDMN,@XR),FMBZER(,@BR)			BRANCH IF
4215	F2	81	36	7706		JE	FMB900			* * EITHER OPERAND
4218	9D	01	11 57	7707		CLC	B@LADV+B@LADV+B@ACD1(B@LDMN,@XR),FMBZER(,@BR)			* MATRIX
421C	F2	81	2F	7708		JE	FMB900			* IS ONE DIMENSION
421F	7C	D0	4F	7709		MVI	FMB900+@Q(,@BR),@@E759			IN PLACE ERROR CODE
4222	AD	01	07 0F	7710		CLC	B@ABAS(@VADDR,@XR),B@LADV+B@ABAS(,@XR)			BRANCH IF EITHER
4226	F2	81	25	7711		JE	FMB900			* OPERAND MATRIX
4229	AD	01	07 17	7712		CLC	B@ABAS(@VADDR,@XR),B@LADV+B@LADV+B@ABAS(,@XR)			* IS THE
422D	F2	81	1E	7713		JE	FMB900			* SAME AS MAT A
4230	7C	CF	4F	7714		MVI	FMB900+@Q(,@BR),@@E758			EXPRESION NOT CONFORMABLE
4233	AD	01	0B 11	7715		CLC	B@LADV+B@ACD2(B@LDMN,@XR),B@LADV+B@LADV+B@ACD1(,@XR)			BR
4237	F2	01	14	7716		JNE	FMB900			* IF NOT CONFORMABLE
423A	7C	CC	4F	7717		MVI	FMB900+@Q(,@BR),@@E755			ASG MAT NOT EQUAL ERROR CODE
423D	AD	01	01 09	7718		CLC	B@ACD1(B@LDMN,@XR),B@LADV+B@ACD1(,@XR)			BRANCH IF MAT A
4241	F2	01	0A	7719		JNE	FMB900			* ROW NOT EQ MAT B
4244	AD	01	03 13	7720		CLC	B@ACD2(B@LDMN,@XR),B@LADV+B@LADV+B@ACD2(,@XR)			BRANCH IF
4248	F2	01	03	7721		JNE	FMB900			* MAT A COL NOT EQ MAT B
424B	F2	87	04	7722		J	FMB950			EXIT
424E	3C	00	0CBC	7723	FMB900	MVI	I\$ERRC,*-*			SET ERROR CODE
4252	C0	87	12D3	7724	FMB950	B	I\$RTRN			RETURN
4256	0000			4257 7725	FMBZER	DC	XL(B@LDMN)'0'			ZERO
				7726		*****				
				7727	*					
				7728	***	END OF FMBMPY				

FYBSMM - SCALAR/MATRIX MULTIPLY

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 371
		7730		*****			*
		7731	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
		7732	*	REFER TO INSTRUCTIONS ON COPYRIGHT NOTICE, 120-2083			*
		7733	*				*
		7734		*****			*
		7735	*	STATUS			*
		7736	*	VERSION 1 MODIFICATION 0			*
		7737	*				*
		7738	*	FUNCTION			*
		7739	*	PERFORMS ERROR CHECKING AND CALCULATES MAT A=XB			*
		7740	*				*
		7741	*	ENTRY POINTS			*
		7742	*	FYBSMM			*
		7743	*				*
		7744	*	INPUT			*
		7745	*	THE DOPE VECTORS FOR A AND B ARE IN THE TOP OF THE STACK SEPERA-			*
		7746	*	TED BY THE UNPACKED FLOATING POINT SCALAR VALUE.			*
		7747	*				*
		7748	*	OUTPUT			*
		7749	*	N/A			*
		7750	*				*
		7751	*	EXTERNAL REFERENCFS			*
		7752	*	I\$STAK			*
		7753	*	I\$ERRC			*
		7754	*	I\$AADR			*
		7755	*	I\$STCK			*
		7756	*	I\$CPLF			*
		7757	*	I\$FMPY			*
		7758	*	I\$CUPF			*
		7759	*	I\$USTK			*
		7760	*	I\$RTRN			*
		7761	*				*
		7762	*	EXITS, NORMAL			*
		7763	*	I\$RTRN WITH I\$ERRC SET TO I@NERR			*
		7764	*				*
		7765	*	EXITS, ERROR			*
		7766	*	I\$RTRN WITH I@ERRC SET TO THE FOLLOWING			*
		7767	*	@@E759			*
		7768	*	OVERFLOW AND UNDER FLOW SET BY MULTIPLY ROUTINE			*
		7769	*				*
		7770	*	TABLES/WORKAREAS			*
		7771	*	N/A			*
		7772	*				*
		7773	*	ATTRIBUTES			*
		7774	*	* NATURALLY RELOCATABLE			*
		7775	*	* REUSABLE			*
		7776	*				*
		7777	*	CHARACTER CODE DEPENDENCY			*
		7778	*	N/A			*
		7779	*				*
		7780	*	NOTES			*
		7781	*	N/A			*
		7782		*****			*

FYBSMM - SCALAR/MATRIX MULTIPLY

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 372

```

4264          7784      ORG      *,B@LVPG,100          LEAVE ROOM FOR FMBMPY
              7785      *
              7786      *          MATRIX/SCALAR MULTIPLY ENTRY POINT
              7787      *
4264 35 02 0D4E      4264 7788 FYBSMM EQU      *          ENTRY POINT
              7789      L      I$STAK,@XR          INIT STACK POINTER
4268 AD 03 03 13      7790      CLC      B@ACD2(B@LDMN*2,@XR),B@LADV+I@LUFV+B@ACD2(,@XR)  BRANCH
426C F2 81 07      7791      JE      FYB100          * IF MATRICES CONFORMABLE
426F 3C CC 0CBC      7792      MVI      I$ERRC,@E755      NOT CONFORMABLE ERROR CODE
4273 F2 87 47      7793      J      FYB400          RETURN
              7794      *
              7795      *          SAVE STACK INFORMATION
              7796      *
4276 6C 0F D0 0F      7797 FYB100 MVC      FYBSCL(I@LUFV+B@LADV,@BR),B@LADV+I@UMNR(,@XR)
427A 6C 01 C6 17      7798      MVC      FYBAMB(@VADDR,@BR),B@LADV+I@LUFV+B@ABAS(,@XR)
              7799      *
              7800      *          ROW LOOP
              7801      *
427E 5C 01 D2 C4      7802 FYB200 MVC      FYBCOL(B@LDMN,@BR),FYBSAV(,@BR)  INIT COLUMN COUNT
              7803      *
              7804      *          COLUMN LOOP
              7805      *
4282 5E 03 C8 D6      7806 FYB300 ALC      FYBAMA(@VADDR*2,@BR),FYBINC(,@BR)  INCR VIRTUAL ADDRESSES
              7807      *
              7808      *          STACK AND UNPACK MATRIX B ELEMENT
              7809      *
4286 1C 01 144A C6      7810      MVC      I$VADR,FYBAMB(@VADDR,@BR) SET MATRIX B PARM FOR STACK
428B C0 87 0B50      7811      B      I$STCK          STACK MATRIX 9 ELEMENT
428F C0 87 0A27      7812      B      I$CPUF          UNPACK ELEMENT
              7813      *
              7814      *          MULTIPLY MATRIX B ELEMENT BY SCALAR ARGUMENT
              7815      *
4293 9C 07 0F D0      7816      MVC      I@RSE2(I@LUFV,@XR),FYBSCL(,@BR)  MOVE SCALAR VALUE TO STK
4297 C0 87 082A      7817      B      I$FMPY          MULTIPLY B ELEMENT BY SCALAR
429B 3D 00 0CBC      7818      CLI      I$ERRC,I@NERR      BRANCH IF ERROR
429F F2 01 1B      7819      JNE      FYB400          * RETURN
              7820      *
              7821      *          PACK AND UNSTACK NEW MATRIX A ELEMENT
              7822      *
42A2 C0 87 0A85      7823      B      I$CUPF          RACK NEW VALUE
42A6 1C 01 144A C8      7824      MVC      I$VADR,FYBAMA(@VADDR,@BR) SET MATRIX A ELEMENT ADDR
42AB C0 87 0BB0      7825      B      I$USTK          UNSTACK MATRIX A ELEMENT
              7826      *
              7827      *          END OF COLUMN LOOP
              7828      *
42AF 5F 01 D2 D8      7829      SLC      FYBCOL(B@LDMN,@BR),FYBDEC(,@BR)  DECR COLUMN COUNT
42B3 D0 84 82      7830      BP      FYB300(,@BR)          BRANCH IF COLUMN NOT FINISHED
              7831      *
              7832      *          END OF ROW LOOP
              7833      *
42B6 5F 01 C2 D8      7834      SLC      FYBROW(B@LDMN,@BR),FYBDEC(,@BR)  DECR ROW COUNT
42BA D0 84 7E      7835      BP      FYB200(,@BR)          BRANCH IF ROWS NOT FINISHED
              7836      *
              7837      *          RETURN TO PAGING MODULE EXIT
              7838      *
42BD C0 87 12D3      7839 FYB400 B      I$RTRN          RETURN

```

FYBSMM - SCALAR/MATRIX MULTIPLY

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 373
					7841	*				
					7842	*	INPUT PARAMETER SAVE AREA			
					7843	*				
42C1				42C2	7844	FYBROW DS	IL(B@LDMN)			CURRENT ELEMENT ROW
42C3				42C4	7845	FYBSAV DS	IL(B@LDMN)			NUMBER OF COLUMNS
42C5				42C6	7846	FYBAMB DS	AL(@VADDR)			CURRENT ELEMENT 8 VIRTUAL ADDR
42C7				42C8	7847	FYBAMA DS	AL(@VADDR)			CURRENT ELEMENT A VIRTUAL ADDR
42C9				42D0	7848	FYBSCL DS	IL(I@LUFV)			SCALAR VALUE
42D1				42D2	7849	FYBCOL DS	IL(B@LDMN)			CURRENT ELEMENT COLUMN
					7850	*				
					7851	*	CONSTANTS			
					7852	*				
42D3	00050005			42D6	7853	FYBINC DC	2AL(@VADDR)(I@LPFV)			INCR 2 VIRTUAL ADDR 81 ELEM LGT
42D7	0001			42D8	7854	FYBDEC DC	IL(B@LDMN)'1'			DECK ROW AND COLUMN COUNT
					7855	*****				
					7856	*				
					7857	***	END OF FYBSMM			

FZBSET - CON/ZER/IDN MATRIX FUNCTIONS

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 374
		7859		*****			
		7860	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
		7861	*	REFER TO INSTRUCTIONS ON COPYRIGHT NOTICE 120-2083			*
		7862	*				*
		7863		*****			
		7864	*	STATUS			*
		7865	*	VERSION 1 MODIFICATION 0			*
		7866	*				*
		7867	*	FUNCTION			*
		7868	*	PERFORMS ERROR CHECKING AND CREATES A MATRIX OF ALL ZEROS, ONES,			*
		7869	*	OR AN IDENTITY MATRIX.			*
		7870	*				*
		7871	*	ENTRY POINTS			*
		7872	*	* FZBION - CREATE IDENTITY MATRIX			*
		7873	*	* FZBCON - CREATE MATRIX OF ALL ONES			*
		7874	*	* FZBZER - CREATE MATRIX OF ALL ZEROS			*
		7875	*				*
		7876	*	INPUT			*
		7877	*	THE DOPE VECTOR FOR THE MATRIX IN THE TOP OF THE STACK			*
		7878	*				*
		7879	*	OUTPUT			*
		7880	*	N/A			*
		7881	*				*
		7882	*	EXTERNAL REFERENCES			*
		7883	*	I\$STAK			*
		7884	*	I\$ERRC			*
		7885	*	I\$VADR			*
		7886	*	I\$USTK			*
		7887	*	I\$RTRN			*
		7888	*				*
		7889	*	EXITS, NORMAL			*
		7890	*	I\$RTRN WITH I\$ERRC SET TO I@NERR			*
		7891	*				*
		7892	*	EXITS, ERROR			*
		7893	*	I\$RTRN WITH I\$ERRC SET TO ONE OF THE FOLLOWING			*
		7894	*	@@E752			*
		7895	*	@@E766			*
		7896	*				*
		7897	*	TABLES/WORKAREAS			*
		7898	*	N/A			*
		7899	*				*
		7900	*	ATTRIBUTES			*
		7901	*	* NATURALLY RELOCATABLE			*
		7902	*	* REUSABLE			*
		7903	*				*
		7904	*	CHARACTER CODE DEPENDENCY			*
		7905	*	N/A			*
		7906	*				*
		7907	*	NOTES			*
		7908	*	N/A			*
		7909		*****			

FZBSET - CON/ZER/IDN MATRIX FUNCTIONS

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 375
					7911	*		
					7912	*	ESTABLISH ADDRESSABILITY	
					7913	*		
4300					7914	ORG	*,B@LVPG,0	PAGE BOUNDARY
				4300	7915	USING	*,@BR	BASE VALUE
					7916	*		
					7917	*	FZBIDN ENTTY POINT	
					7918	*		
				4300	7919	FZBIDN EQU	*	ENTRY POINT
4300	5C	09	84	76	7920	MVC	FZBDAG(I@LPFV*2,@BR),FZBC01(,@BR)	SET DIAG AND NON-DIAG
4304	35	02	0D4E		7921	L	I\$STAK,@XR	* CONSTANTS AND LOAD STK PTR
4308	9D	01	01	75	7922	CLC	B@ACD1(B@LDMN,@XR),FZBZRO(,@BR)	BRANCH IF MATRIX
430C	F2	01	07		7923	JNE	FZB100	* NOT ONE DIM
430F	3C	C9	0CBC		7924	MVI	I\$ERRC,@E752	SET ONE DIM ERROR CODE
4313	F2	87	53		7925	J	FZB700	RETURN
4316	AD	01	03	01	7926	FZB100 CLC	B@ACD2(B@LDMN,@XR),B@ACD1(,@XR)	BRANCH IF MATRIX
431A	F2	81	1A		7927	JE	FZB300	* IS SQUARE
431D	3C	D7	0CBC		7928	MVI	I\$ERRC,@E766	NON-SQUARE ERROR CODE
4321	F2	87	45		7929	J	FZB700	RETURN
					7930	*		
					7931	*	FZBCON - ENTRY POINT	
					7932	*		
				4324	7933	FZBCON EQU	*	ENTRY POINT
4324	5C	04	84	76	7934	MVC	FZBDAG(I@LPFV,@BR),FZBC01(,@BR)	SET DIAG CONSTANT TO 1
4328	F2	87	04		7935	J	FZB200	PROCESS
					7936	*		
					7937	*	FZBZER - ENTRY POINT	
					7938	*		
				432B	7939	FZBZER EQU	*	
432B	5C	04	84	71	7940	MVC	FZBDAG(I@LPFV,@BR),FZBC00(,@BR)	SET DIAG CONSTANT TO 0
432F	5C	04	7F	84	7941	FZB200 MVC	FZBELE(I@LPFV,@BR),FZBDAG(,@BR)	NON-DIAG EQUALS DIAG
4333	35	02	0D4E		7942	L	I\$STAK,@XR	FETCH STACK POINTER
4337	6C	07	8C	07	7943	FZB300 MVC	FZBADR(B@LADV,@BR),B@ABAS(,@XR)	SAVE ROW AND COL COUNT
					7944	*		
					7945	*	ROW LOOP	
					7946	*		
433B	5C	01	8A	88	7947	FZB400 MVC	FZBCOL(B@LDMN,@BR),FZBSAV(,@BR)	INIT COL COUNT
					7948	*		
					7949	*	COLUMN LOOP	
					7950	*		
433F	9C	04	04	7F	7951	FZB500 MVC	I@LPFV-1(I@LPFV,@XR),FZBELE(,@BR)	NON DIAS ELEM TO STACK
4343	5D	01	8A	86	7952	CLC	FZBCOL(B@LDMN,@BR),FZBROW(,@BR)	BRANCH IF ROW AND
4347	F2	01	04		7953	JNE	FZB600	COLUMN NOT SAME
					7954	*		
					7955	*	DIAGONAL ELEMENT	
					7956	*		
434A	9C	04	04	84	7957	MVC	I@LPFV-1(I@LPFV,@XR),FZBDAG(,@BR)	DIAG ELEM TO STACK
					7958	*		
					7959	*	UNSTACK	
					7960	*		
434E	5E	01	8C	78	7961	FZB600 ALC	FZBADR(,@BR),FZBLNG(@VADDR,@BR)	INCR VIRTUAL ADDR BY ELEM
4352	1C	01	144A	8C	7962	MVC	I\$VADR,FZBADR(@VADDR,@BR)	SET STACK PAGE PARAMETER
4357	C0	87	0BB0		7963	B	I\$USTK	UNSTACK
					7964	*		
					7965	*	END OF COLUMN LOOP	
					7966	*		

FZBSET - CON/ZER/IDN MATRIX FUNCTIONS

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 376
435B	5F	01	8A	7A	7967	SLC	FZBCOL(B@LDMN,@BR),FZBBN1(,@BR)	DECR COLUMN COUNT		
435F	D0	84	3F		7968	BP	FZB500(,@BR)	BRANCH IF ROW NOT FINISHED		
					7969	*				
					7970	*	END OF ROW LOOP			
					7971	*				
4362	5F	01	86	7A	7972	SLC	FZBROW(B@LDMN,@BR),FZBBN1(,@BR)	DECR ROW COUNT		
4366	D0	84	3B		7973	BP	FZB400(,@BR)	BRANCH IF MORE ROWS		
					7974	*				
					7975	*	EXIT TO PAGING MODULE			
					7976	*				
4369	C0	87	12D3		7977	FZB700 B	I\$RTRN	RETURN		
					7978	*				
					7979	*	PACKED FLOATING POINT CONSTANTS			
					7980	*				
436D	00			436D	7981	DC	AL1(I@ASTA)	PACKED FLOATING		
436E	000000			4370	7982	DC	XL(I@LPFV-2)'0'	* POINT CONSTANT		
4371	1E			4371	7983	FZBC00 DC	AL1(B@NXLO)	* OF ZERO		
4372	01			4372	7984	DC	AL1(I@ASTA+1)	PACKED FLOATING		
4373	000000			4375	7985	FZBZRO DC	XL(I@LPFV-2)'0'	* POINT CONSTANT		
4376	81			4376	7986	FZBC01 DC	AL1(B@NXZR+1)	* OF ONE		
4377	0005			4378	7987	FZBLNG DC	AL(@VADDR)(I@LPFV)	INCR VIRTUAL ADDR		
4379	0001			437A	7988	FZBBN1 DC	IL(B@LDMN)'1'	DECR COLUMN AND ROW COUNT		
					7989	*				
					7990	*	ELEMENT AND PARAMETER SAVE AREAS			
					7991	*				
437B				437F	7992	FZBELE DS	XL(I@LPFV)	NON-DIAGONAL ELEMENT		
4380				4384	7993	FZBDAG DS	XL(I@LPFV)	DIAGONAL ELEMENT		
4385				4386	7994	FZBROW DS	IL(B@LDMN)	CURRENT ROW COUNT		
4387				4388	7995	FZBSAV DS	IL(B@LDMN)	TOTAL COLUMN COUNT		
4389				438A	7996	FZBCOL DS	IL(B@LDMN)	CURRENT COLUMN COUNT		
438B				438C	7997	FZBADR DS	AL(@VADDR)	VA OF CURRENT ELEMENT		
					7998	*****				
					7999	* END OF FZBSET				

FLBMAS - MATRIX ASSIGNMENT

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 377
		8001		*****			
		8002	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
		8003	*	REFER TO INSTRUCTIONS ON COPYRIGHT NOTICE, 120-2083			*
		8004	*				*
		8005		*****			
		8006	*	STATUS			*
		8007	*	VERSION 1 MODIFICATION 0			*
		8008	*				*
		8009	*	FUNCTION			*
		8010	*	PERFORMS ERROR CHECKING AND CALCULATION FOR MAT A=B			*
		8011	*				*
		8012	*	ENTRY POINTS			*
		8013	*	FLBMAS			*
		8014	*				*
		8015	*	INPUT			*
		8016	*	DOPE VECTORS FOR A AND B IN TOP OF STACK			*
		8017	*				*
		8018	*	OUTPUT			*
		8019	*	N/A			*
		8020	*				*
		8021	*	EXTERNAL REFERENCES			*
		8022	*	I\$STAK			*
		8023	*	I\$ERRC			*
		8024	*	I\$VADR			*
		8025	*	I\$STCK			*
		8026	*	I\$USTK			*
		8027	*	I\$RTRN			*
		8028	*				*
		8029	*	EXITS, NORMAL			*
		8030	*	I\$RTRN-- I\$ERRC SET TO I@ENUL			*
		8031	*				*
		8032	*	EXITS, ERROR			*
		8033	*	I\$RTRN-- I\$ERRC SET TO @@E755			*
		8034	*				*
		8035	*	TABLES/WORK AREAS			*
		8036	*	N/A			*
		8037	*				*
		8038	*	ATTRIBUTES			*
		8039	*	* NATURALLY RELOCATABLE			*
		8040	*	* REUSABLE			*
		8041	*				*
		8042	*	CHARACTER CODE DEPENDENCY			*
		8043	*	N/A			*
		8044	*				*
		8045	*	NOTES			*
		8046	*	N/A			*
		8047		*****			

FLBMAS - MATRIX ASSIGNMENT

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 378
					8049	*				
					8050	*	ESTABLISH ADDRESSABILITY			
					8051	*				
					8052	*FLBAD1	VPAGE 160			
43A0					8053		ORG *,256,160			SET STARTING ADDRESS
				43A0	8054	FLBAD1	EQU *			START OF PROGRAM CODING
42A1					8055		ORG *-255			RESET IAR TO PAGE
4300					8056		ORG *,256,0			* BOUNDARY ADDRESS
				4300	8057		USING *,@BR			SET PAGE BASE ADDRESS
43A0					8058		ORG FLBAD1			RESET STARTING ADDRESS
					8059	***	END OF EXPANSION ***			
					8060	*				
					8061	*	MAT A=B ENTRY POINT			
					8062	*				
				43A0	8063	FLBMAS	EQU *			MODULE ENTRY POINT
43A0	35	02	0D4E		8064		L I\$STAK,@XR			FETCH STACK POINTER
					8065	*				
					8066	*	ERROR CHECKING			
					8067	*				
43A4	AD	03	03 0B		8068		CLC B@ACD2(B@LDMN*2,@XR),B@LADV+B@ACD2(,@XR)			BRANCH IF
43A8	F2	81	08		8069		JE FLB050			* DIMENSIONS IDENTICAL
43AB	3C	CC	0CBC		8070		MVI I\$ERRC,@E755			SET INCOMATIBLE ERROR CODE
43AF	C0	87	43E3		8071		B FLB800			EXIT
					8072	*				
					8073	*	INITIALIZATION			
					8074	*				
43B3	6C	07	EE 0F		8075	FLB050	MVC FLBADB(B@LADV,@BR),B@LADV+B@ABAS(,@XR)			FETCH MAT B DOPE
43B7	6C	01	F0 07		8076		MVC FLBADA(@VADDR,@BR),B@ABAS(,@XR)			* VECTOR AND MAT A ADDR
					8077	*				
					8078	*	ROW LOOP			
					8079	*				
43BB	5C	01	EC EA		8080	FLB100	MVC FLBCOL(B@LDMN,@BR),FLBSAV(,@BR)			INIT COLUMN COUNT
					8081	*				
					8082	*	COLUMN LOOP			
					8083	*				
43BF	5E	03	F0 F4		8084	FLB200	ALC FLBADA(@VADDR*2,@BR),FLBLEL(,@BR)			INDR ELEMENT ADDRESSES
43C3	1C	01	144A EE		8085		MVC I\$VADR,FLBADB(@VADDR,@BR)			MAT B ELEMENT ADDR
43C8	C0	87	0B50		8086		B I\$STCK			FETCH MAT B ELEMENT
43CC	1C	01	144A F0		8087		MVC I\$VADR,FLBADA(@VADDR,@BR)			MAT A ELEMENT ADDR
43D1	C0	87	0BB0		8088		B I\$USTK			PLACE MAT A ELEMENT
					8089	*				
					8090	*	END OF COLUMN LOOP			
					8091	*				
43D5	5F	01	EC F6		8092		SLC FLBCOL(B@LDMN,@BR),FLBONE(,@BR)			DECR COLUMN COUNTER
43D9	D0	84	BF		8093		BP FLB200(,@BR)			BRANCH IF MORE
					8094	*				
					8095	*	END OF ROW LOOP			
					8096	*				
43DC	5F	01	E8 F6		8097		SLC FLBROW(B@LDMN,@BR),FLBONE(,@BR)			DECR ROW COUNTER
43E0	D0	84	BB		8098		BP FLB100(,@BR)			BRANCH IF MORE
					8099	*				
					8100	*	RETURN			
					8101	*				
43E3	C0	87	12D3		8102	FLB800	B I\$RTRN			RETURN

FLBMAS - MATRIX ASSIGNMENT

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 379
			8104	*		
			8105	*	DIM COUNTERS AND ADDR SAVE AREA	
			8106	*		
43E7		43E8	8107	FLBROW DS	CL(B@LDMN)	ROW COUNTER
43E9		43EA	8108	FLBSAV DS	CL(B@LDMN)	COLUMN INITIALIZER
43EB		43EC	8109	FLBCOL DS	CL(B@LDMN)	COLUMN COUNTER
43ED		43EE	8110	FLBADB DS	CL(@VADDR)	CURRENT MAT B ELEMENT ADDR
43EF		43F0	8111	FLBADA DS	CL(@VADDR)	CURRENT MAT A ELEMENT ADOR
			8112	*		
			8113	*	CONSTANTS	
			8114	*		
43F1 00050005		43F4	8115	FLBLEL DC	2AL(@VADDR)(I@LPFV)	INCR ELEMENT ADDRESSES
43F5 0001		43F6	8116	FLBONE DC	IL(B@LDMN)'1'	DECR COUNTERS
			8117	*****		
			8118	*		
			8119	***	END OF FLBMAS	

FXBTRN - MATRIX TRANSPOSITION

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 380
		8121		*****	
		8122	*	5703-XM1 COPYRIGHT IBM CORP. 1970	*
		8123	*	REFER TO INSTRUCTIONS ON COPYRIGHT NOTICE, 120-2083	*
		8124	*		*
		8125	*	*****	*
		8126	*	STATUS	*
		8127	*	VERSION 1 MODIFICATION 0	*
		8128	*		*
		8129	*	FUNCTION	*
		8130	*	PERFORMS ERROR CHECKING AND CALCULATES MAT A = TRAN B	*
		8131	*		*
		8132	*	ENTRY POINTS	*
		8133	*	FXBTRN	*
		8134	*		*
		8135	*	INPUT	*
		8136	*	THE DOPE VECTORS FOR A AND 8 IN THE TOP OF THE STACK	*
		8137	*		*
		8138	*	OUTPUT	*
		8139	*	N/A	*
		8140	*		*
		8141	*	EXTERNAL REFERENCES	*
		8142	*	I\$STAK	*
		8143	*	I\$VADR	*
		8144	*	I\$STCK	*
		8145	*	I\$USTK	*
		8146	*	I\$ERRC	*
		8147	*	I\$RIRN	*
		8148	*		*
		8149	*	EXITS, NORMAL	*
		8150	*	I\$RTRN WITH MEW SET TO I\$ERRN	*
		8151	*		*
		8152	*	EXITS, ERROR	*
		8153	*	I\$RTRN WITH I\$ERRC SET TO ONE OF THE FOLLOWING:	*
		8154	*	@@E752	*
		8155	*	@@E754	*
		8156	*	@@E756	*
		8157	*	@@E767	*
		8158	*		*
		8159	*	TABLES/WORK AREAS	*
		8160	*	N/A	*
		8161	*		*
		8162	*	ATTRIBUTES	*
		8163	*	* NATURALLY RELOCATABLE	*
		8164	*	* REUSABLE	*
		8165	*		*
		8166	*	CHARACTER CODE DEPENDENCY	*
		8167	*	N/A	*
		8168	*		*
		8169	*	NOTES	*
		8170	*	N/A	*
		8171	*	*****	*

FXBTRN - MATRIX TRANSPOSITION

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 381
					8173	*				
					8174	*	ESSTABLISH ADDRESSABILITY			
					8175	*				
4400					8176		ORG *,B@LVP		PAGE BOUNDARY	
				4400	8177		USING *,@BR		BASE VALUE	
					8178	*				
					8179	*	MAT A - TRN(B) ENTRY POINT			
					8180	*				
				4400	8181	FXBTRN	EQU *		MODULE ENTRY POINT	
4400	35	02	0D4E		8182		L I\$STAK,@XR		FETCH STACK POINTER	
					8183	*				
					8184	*	ERROR CHECKING			
					8185	*				
4404	7C	C9	7F		8186		MVI FXB800+@Q(,@BR),@@E752		ASS MAT NOT 2 DIM ERROR CODE	
4407	9D	01	01 97		8187		CLC B@ACD1(B@LDMN,@XR),FXBZER(,@BR)		BRANCH IF MAT A	
440B	F2	81	70		8188		JE FXB800		* NOT 2 DIM	
440E	7C	CB	7F		8189		MVI FXB800+@Q(,@BR),@@E754		ARG MAT NOT 2 DIM ERROR CODE	
4411	9D	01	09 97		8190		CLC B@LADV+B@ACD1(B@LDMN,@XR),FXBZER(,@BR)		BRANCH IF MAT	
4415	F2	81	66		8191		JE FXB800		* NOT 2 DIM	
4418	7C	CD	7F		8192		MVI FXB800+@Q(,@BR),@@E756		NOT CONFORMABLE ERROR CODE	
441B	AD	01	03 09		8193		CLC B@ACD2(B@LDMN,@XR),B@LADV+B@ACD1(,@XR)		BRANCH IF	
441F	F2	01	5C		8194		JNE FXB800		* MATRICES	
4422	AD	01	01 0B		8195		CLC B@ACD1(B@LDMN,@XR),B@LADV+B@ACD2(,@XR)		* NOT	
4426	F2	01	55		8196		JNE FXB800		* CONFORMABLE	
4429	7C	D8	7F		8197		MVI FXB800+@Q(,@BR),@@E767		IN PLACE ERROR CODE	
442C	AD	01	07 0F		8198		CLC B@ABAS(@VADDR,@XR),B@LADV+B@ABAS(,@XR)		BRANCH IR MAT	
4430	F2	81	4B		8199		JE FXB800		* A AND B ARE SAME	
					8200	*				
					8201	*	INITIALIZATION			
					8202	*				
4433	6C	0F	95 0F		8203		MVC FXBADB(B@LADV*2,@BR),B@LADV+B@ABAS(,@XR)		FETCH DOPE VCTR	
4437	5C	01	8B 97		8204		MVC FXBRDP(@VADDR,@BR),FXBZER(,@BR)		INIT ROW DISP	
443B	5E	01	8B 9A		8205	FXB100	ALC FXBRDP(@VADDR,@BR),FXBLEL(,@BR)		CALCULATE	
443F	5F	01	89 98		8206		SLC FXBCIN(B@LDMN,@BR),FXBONE(,@BR)		* ROW	
4443	D0	84	3B		8207		BP FXB100(,@BR)		* * DISPLACEMENT	
					8208	*				
					8209	*	ROW LOOP			
					8210	*				
4446	5E	01	8D 9A		8211	FXB200	ALC FXBADA(@VADDR,@BR),FXBLEL(,@BR)		INIT MAT B ADDR TO	
444A	5C	01	87 8D		8212		MVC FXBADC(@VADDR,@BR),FXBADA(,@BR)		* ROW ONE	
444E	5C	01	93 91		8213		MVC FXBCOL(B@LDMN,@BR),FXBSAV(,@BR)		INIT COLUMN COUNT	
					8214	*				
					8215	*	COLUMN LOOP			
					8216	*				
4452	5E	01	95 9A		8217	FXB300	ALC FXBADB(@VADDR,@BR),FXBLEL(,@BR)		UPDATE MAT A ELEM ADDR	
4456	1C	01	144A 95		8218		MVC I\$VADR,FXBADB(@VADDR,@BR)		SET PAGING PPM	
445B	C0	87	0B50		8219		B I\$STCK		FETCH MAT A ELEMENT	
445F	1C	01	144A 87		8220		MVC I\$VADR,FXBADC(@VADDR,@BR)		SET MAT B ELEM ADDR	
4464	C0	87	0BB0		8221		B I\$USTK		PLACE MAT B ELEMENT	
4468	5E	01	87 8B		8222		ALC FXBADC(@VADDR,@BR),FXBRDP(,@BR)		INCR MAT B ELEM ADDR	
					8223	*				
					8224	*	END OF COLUMN LOOP			
					8225	*				
446C	5F	01	93 98		8226		SLC FXBCOL(B@LDMN,@BR),FXBONE(,@BR)		DECR COLUMN COUNTER	
4470	D0	84	52		8227		BP FXB300(,@BR)		BRANCH IF MORE	
					8228	*				

FXBTRN - MATRIX TRANSPOSITION

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 382
					8229	*	END OF ROW LOOP			
					8230	*				
4473	1F	01	448F	98	8231	SLC	FXBROW(B@LDMN),FXBONE(,@BR) DECR ROW COUNT			
4478	D0	84	46		8232	BP	FXB200(,@BR) BRANCH IF MORE			
447B	F2	87	04		8233	J	FXB900 EXIT			
447E	3C	00	0CBC		8234	FXB800 MVI	I\$ERRC,*-* SET ERROR CODE			
4482	C0	87	12D3		8235	FXB900 B	I\$RTRN RETURN			
					8236	*				
					8237	*	DIMENSION AND ADDR SAVE AREA			
					8238	*				
4486				4487	8239	FXBADC DS	CL(@VADDR) CURRENT MAT A ELEMENT ADDR			
4488				4489	8240	FXBCIN DS	CL(B@LDMN) MAT A COLUMN COUNTER			
448A				448B	8241	FXBRDP DS	CL(@VADDR) ROW DISP FOR MAT A			
448C				448D	8242	FXBADA DS	CL(@VADDR) ROW ADDR OF MAT A			
448E				448F	8243	FXBROW DS	CL(B@LDMN) ROW COUNTER			
4490				4491	8244	FXBSAV DS	CL(B@LDMN) COLUMN COUNT INITIALIZER			
4492				4493	8245	FXBCOL DS	CL(B@LDMN) COLUMN COUNTER			
4494				4495	8246	FXBADB DS	CL(@VADDR) CURRENT MAT B ELEMENT ADDR			
					8247	*				
					8248	*	CONSTANTS			
					8249	*				
4496	0000			4497	8250	FXBZER DC	XL(B@LDMN)'0' ROW COUNT OF 1 DIM MATRIX			
4498	01			4498	8251	FXBONE DC	XL1'1' DECR COUNTERS			
4499	0005			449A	8252	FXBLEL DC	AL(@VADDR)(I@LPFV) INCR VIRTUAL ADDRESSES			
					8253	*****				
					8254	*				
					8255	*** END OF FXBTRN				

FVBINV - MATRIX INVERSION/DETERMINANT ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 383
		8257		*****			
		8258	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
		8259	*	REFER TO INSTRUCTIONS ON COPYRIGHT NOTICE, 120-2083			*
		8260	*				*
		8261		*****			*
		8262	*	STATUS			*
		8263	*	VERSION 1 MODIFICATION 0			*
		8264	*				*
		8265	*	FUNCTION			*
		8266	*	* FVBINV IS THE VIRTUAL MEMORY INTERFACE ROUTINE FOR THE MATRIX			*
		8267	*	INVERSE/DETERMINANT.			*
		8268	*	* THIS MODULE CHECKS FOR POSSIBLE DIMENSIONING ERRORS AND			*
		8269	*	MATRICES OF EXCESSIVE SIZE.			*
		8270	*	* FVBINV ASSURES THAT ALL PAGES ARE 'PUSHED' TO DISK MEMORY			*
		8271	*	BEFORE THE MATRIX INVERSE EXECUTION.			*
		8272	*	* AT THE CONCLUSION OF THE INVERSE EXECUTION, CONTROL IS RETURNED			*
		8273	*	TO FVBINV.			*
		8274	*	* ALL UNLOCKED PAGES ARE THEN 'PULLED' TO REAL CORE AND CONTROL			*
		8275	*	IS RETURNED TO THE INTREPRETER.			*
		8276	*				*
		8277	*	ENTRY POINTS			*
		8278	*	FVBINV HAS TWO(2) ENTRY POINTS; DEPENDING ON THE FUNCTION			*
		8279	*	REQUIRED.			*
		8280	*	FOR THE INVERSE FUNCTION - FVBINV LOCATION X'00'			*
		8281	*	FOR THE DET FUNCTION - FVBDET LOCATION X'40'			*
		8282	*				*
		8283	*	INPUT			*
		8284	*	I\$STAK CONTAINS A POINTER TO THE RUN TIME STACK			*
		8285	*	WHERE THE DOPE VECTOR(S) FOR THE ARRAY(5) ARE STORED.			*
		8286	*				*
		8287	*	OUTPUT			*
		8288	*	* ALL OUTPUT IS CREATED BY FISINV.			*
		8289	*				*
		8290	*	EXTERNAL REFERENCES			*
		8291	*	I\$CERR - ERROR INDICATOR			*
		8292	*	I\$STAK - STACK POINTER			*
		8293	*	I\$RTRN - INTERPRETER RETURN ADDRESS			*
		8294	*	I\$CALL - CALLING ROUTINE			*
		8295	*	\$BLOAD - ENTRY TO BLAST LOADER			*
		8296	*	\$BLNOE - SWITCH IN \$BLOAD			*
		8297	*	\$LDRTN - DESIRED BRANCH TO ADDRESS IN \$BLOAD			*
		8298	*	\$BLRTN - ENTRY POINT TO \$BLOAD			*
		8299	*	I\$WRK2 - INTERPRETER WORK AREA(2 BYTES)			*
		8300	*				*
		8301	*	EXITS, NORMAL			*
		8302	*	NORMAL RETURN TO THE INTERPRETER			*
		8303	*	B I\$RTRN			*
		8304	*				*
		8305	*	EXITS, ERROR			*
		8306	*	AN APPROPRIATE ERROR CODE IS SET AT I\$CERR AND CONTROL			*
		8307	*	IS RETURNED TO I\$RTRN VIA:			*
		8308	*	B I\$RTRN			*
		8309	*				*
		8310	*	TABLES/WORKAREAS			*
		8311	*	N/A			*
		8312	*				*

FVBINV - MATRIX INVERSION/DETERMINANT ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 384	
		8313	*	*ATTRIBUTES				*
		8314	*	* * NATURALLY RELOCATABLE AND REUSABLE				*
		8315	*					*
		8316	*	*CHARACTER CODE DEPENDENCY				*
		8317	*	* N/A				*
		8318	*					*
		8319	*	*NOTES				*
		8320	*	* ERROR PROCEDURES				*
		8321	*	* N/A				*
		8322	*					*
		8323	*	* REGISTER USAGE				*
		8324	*	* XR1 - BASE REGISTER				*
		8325	*	* XR2 - WORK REGISTER				*
		8326	*					*
		8327	*	* SAVED/RESTORED AREAS				*
		8328	*	* NONE				*
		8329	*					*
		8330	*	* MODIFICATION CONSIDERATIONS				*
		8331	*	* N/A				*
		8332	*					*
		8333	*	* REQUIRED MODULES				*
		8334	*	* @SYSEQ - SYSTEM EQUATES				*
		8335	*	* @FXDEQ - FINED ADDRESS EQUATES				*
		8336	*	* \$B@EQU - COMPILER EQUATES				*
		8337	*	* \$ILEQU - STANDARD OR LONG PRECISION EQUATES				*
		8338	*	* \$I\$EQU - INTERPRETER EQUATES				*
		8339	*					*
		8340	*	* OTHER				*
		8341	*	* N/A				*
		8342	*	*****				*

FVBINV - MATRIX INVERSION/DETERMINANT ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 385

```

      8344 *FVBINV VPAGE 0          SET BASE AND INFORM ASSEMBLER
4500      8345      ORG      *,256,0      SET STARTING ADDRESS
      4500 8346 FVBINV EQU      *          START OF PROGRAM CODING
4401      8347      ORG      *-255        RESET IAR TO PAGE
4500      8348      ORG      *,256,0      * BOUNDARY ADDRESS
      4500 8349      USING *,@BR          SET PAGE BASE ADDRESS
4500      8350      ORG      FVBINV        RESET STARTING ADDRESS
      8351 *** END OF EXPANSION ***

4500 7C 1C 98      8353      MVI      FVB015+@Q(,@BR),FVBLGM      SET VALUE FOR INVERSE
4503 35 02 0D4E    8354      L        I$STAK,@XR                STACK POINTER
4507 7C 00 E2      8355      MVI      FVBADD(,@BR),@ZERO          SET TO INVERSE ENTRY
450A 3C CB 0CBC    8356      MVI      I$ERRC,@E754                SET ERROR CODE
450E 9D 01 09 E4   8357      CLC      B@ACD1+B@LADV(B@LDMN,@XR),FVBZRO(,@BR) 2-DIMENSIONAL ?
4512 F2 81 BC      8358      JE       FVB030                      NO- EXIT
4515 3C D7 0CBC    8359      MVI      I$ERRC,@E766
4519 AD 01 09 0B   8360      CLC      B@ACD1+B@LADV(B@LDMN,@XR),B@ACD2+B@LADV(,@XR)
      8361 *                                DIMENSIONS EQUAL ?
451D F2 01 B1      8362      JNE      FVB030
4520 3C D6 0CBC    8363      MVI      I$ERRC,@E765
4524 AD 01 0F 07   8364      CLC      B@ABAS+B@LADV(@VADDR,@XR),B@ABAS(,@XR) INVERSE IN PLACE ?
4528 F2 81 A6      8365      JE       FVB030                      YES- ERROR EXIT
452B 3C CE 0CBC    8366      MVI      I$ERRC,@E757
452F AD 03 0B 03   8367      CLC      B@ACD2+B@LADV(2*B@LDMN,@XR),B@ACD2(,@XR) MAT IDENTICAL ?
4533 F2 01 9B      8368      JNE      FVB030                      NO- ERROR
4536 D0 87 43      8369      B        FVBDET(,@BR)                CONTINUE
4540      8370      ORG      FVBINV+X'40'      DET. ENTRY POINT
4540 7C 17 E2      8371      MVI      FVBADD(,@BR),FVBDIS          DET. ENTRY POINT
4543 7C 12 D8      8372 FVBDET MVI      FVBBDPL+@DCNT(,@BR),FVB08K ADJUST AVAILABE CORE
4546 3D 00 043B    8373      CLI      $EXFTR,@ZERO                CORE AVAILABE ?
454A F2 81 05      8374      JE       FVB010                      NO- ONLY 8K
454D 4E 00 D8 043B 8375      ALC      FVBBDPL+@DCNT(,@BR),$EXFTR(1) ADJUST AVAILABLE CORE
      8376 * COMPUTE ACTUAL ADDRESS
4552 5C 01 D7 E8   8377 FVB010 MVC      FVBBDPL+@DSAD(@DADDR,@BR),FVBIDS(,@BR) SAVE AREA ADDR 1-3
4556 4E 01 D7 0587 8378      ALC      FVBBDPL+@DSAD(@DADDR,@BR),$BSADR ADD BASE CDR 1-3
455B D2 02 D5      8379      LA       FVBBDPL(,@BR),@XR          COMPUTE ACTUAL ADDRESS
455E 74 02 89      8380      ST       FVBBDP(,@BR),@XR          STORE ADDRESS IN PARAMETER LIST
4561 35 02 0D4E    8381      L        I$STAK,@XR                STACK POINTER
4565 3C CB 0CBC    8382      MVI      I$ERRC,@E754                ASSIGNED MAT NOT 2-DIMENSIONAL
4569 9D 01 01 E4   8383      CLC      B@ACD1(B@LDMN,@XR),FVBZRO(,@BR) 2-DIMENSIONAL ?
456D F2 81 61      8384      JE       FVB030                      NO-
      8385 * DETERMINANT ERROR MESSAGES
4570 3C D7 0CBC    8386      MVI      I$ERRC,@E766                SET NON-SQUARE ERROR
4574 AD 01 01 03   8387      CLC      B@ACD1(B@LDMN,@XR),B@ACD2(,@XR) NON-SQUARE ?
4578 F2 01 56      8388      JNE      FVB030                      ERROR, EXIT
457B F1 E2 00      8389      APL      @PBUSY                        LOOP IF PRINTER BUSY 1-4
457E C0 87 12B1    8390      B        I$CALL                      PUSH DATA TO DISK
4582 4C00          4583 8391      DC      AL(@VADDR)(V$VMPS)      ADDRSS OF PUSH ROUTINE
      8392 * SAVE ALL OF CORE
4584 C0 87 0025    8393      B        $DISKN                      SAVE CORE VIA DISK ROUTINE
4588          4589 8394 FVBBDP@ DS      CL(@CADDR)              ADDRESS OF DPL
458A C0 87 0025    8395      B        $DISKN                      WAIT FOR WRITE TO COMPETE
458E 057F          458F 8396      DC      AL(@CADDR)($WAITF)      WAIT FUNCTION CODE
4590 F2 87 07      8397      JC       FVB020,I@PRSW                @NOP=LONG -- @UCB=STD.
4593 5C 01 DD E6   8398      MVC      FVBBDP2+@DSAD(@DADDR,@BR),FVBLNG(,@BR) SET LONG PRECISION
4597 7C 1C A3      8399 FVB015 MVI      FVB021+@Q(,@BR),FVBLGM      SET MAXIMUM DIMENSION

```

FVBINV - MATRIX INVERSION/DETERMINANT ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 386

459A	35	02	0D4E		8400	FVB020	L	I\$STAK,@XR	FETCH PROPER ROUTINE
459E	3C	D5	0CBC		8401		MVI	I\$ERRC,@E764	SET ERROR CODE
45A2	BD	23	03		8402	FVB021	CLI	B@ACD2(,@XR),FVBSTM	TOO LARGE (ASSUME STD) ?
45A5	F2	84	29		8403		JH	FVB030	TOO LARGE BBB
45A8	4E	01	DD 0587		8404		ALC	FVBDP2+@DSAD(@CADDR,@BR), \$BSADR	ADJUST LOAD ADDRESS
45AD	3C	00	0CBC		8405		MVI	I\$ERRC,I@NERR	SET ERROR CODE
45B1	D2	02	CB		8406		LA	FVB025(,@BR),@XR	SET RETURN ADDRESS
45B4	34	02	0D5B		8407		ST	I\$WRK2,@XR	SAVE IN COMMUNICATION AREA
45B8	C2	02	0522		8408		LA	\$BLOAD,@XR	SET \$BLOAD BASE
				0522	8409		USING	\$BLOAD,@XR	INFORM ASSEMBLER
45BC	BC	80	47		8410		MVI	\$BLNOE(,@XR),@NOP	NO ADDRESS INCREMENT
45BF	9C	01	4F E2		8411		MVC	\$LDRTN(@CADDR,@XR),FVBADD(,@BR)	
45C3	9C	05	5C E0		8412		MVC	\$BLDPL+@DBFR2(,@XR),FVBDP2+@DBFR2(@DPLNG,@BR)	MOVE DPL
45C7	C0	87	0550		8413		B	\$BLRTN	BLOAD ENTRY POINT
45CB	C0	87	12B1		8414	FVB025	B	I\$CALL	CALL ROUTINE
45CF	4C06			45D0	8415		DC	AL(@VADDR)(V\$VMPL)	VIRTUAL ADDR OF PULL ROUTINE
45D1	C0	87	12D3		8416	FVB030	B	I\$RTRN	RETURN

FVBINV - MATRIX INVERSION/DETERMINANT ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 387

		8418	*FVBDPL DPL	FUNC=@DPUT,DADDR-FVBSAV,CNT-FVBSAV,CADDR-#\$\$FIS	
		45D5 8419	FVBDPL EQU *	DISK PARAMETER LIST	
45D5	02	45D5 8420	DC	AL1(@DPUT)	REQUESTED FUNCTION
45D6	2A00	45D7 8421	DC	AL2(FVBSAV)	DISK ADDRESS
45D8	12	45D8 8422	DC	AL1(FVB08K)	SECTOR COUNT
45D9	0E00	45DA 8423	DC	AL2(#\$\$FIS)	BUFFER ADDRESS
		8424	*** END OF EXPANSION ***		
		8426	*FVBDP2 DPL	FUNC=@DGET,DADDR-#\$FIST,CNT-#\$@FIS,CADDR-#\$\$FIS	
		45DB 8427	FVBDP2 EQU *	DISK PARAMETER LIST	
45DB	01	45DB 8428	DC	AL1(@DGET)	REQUESTED FUNCTION
45DC	1700	45DD 8429	DC	AL2(#\$FIST)	DISK ADDRESS
45DE	09	45DE 8430	DC	AL1(#\$@FIS)	SECTOR COUNT
45DF	0E00	45E0 8431	DC	AL2(#\$\$FIS)	BUFFER ADDRESS
		8432	*** END OF EXPANSION ***		
45E1		45E2 8434	FVBADD DS	CL(@CADDR)	CORE ADDRESS OF INVERT
45E1		8435	ORG	FVBADD-1	RESET LOCATION COUNTER
45E1	0E17	45E2 8436	DC	XL(@CADDR)'0E17'	INITIALIZE TO E DET ENTRY
45E3	0000	45E4 8437	FVBZRO DC	IL(@CADDR)'0'	ZERO
45E5	1724	45E6 8438	FVBLNG DC	AL(@CADDR)(#\$FILN)	DISK ADDRESS FOR #FVLNG
45E7	212C	45E8 8439	FVBIDS DC	AL(@DADDR)(#\$#INV)	FIRST AVAILABLE SPACE 1-3
		8440	*	* BEYOND SYSTEM END	1-3
		0012 8441	FVB08K EQU	18	NUMBER OF SECTORS TO SAVE-8K
		0017 8442	FVBDIS EQU	X'17'	DISPLACEMNT TO DET D ENTRY
		001C 8443	FVBLGM EQU	28	MAXIMUM INVERTABLE MATRIX-LONG
		0023 8444	FVBSTM EQU	35	MAXIMUM STANDARD
		2A00 8445	FVBSAV EQU	X'2A00'	SAVE CORE ADDRESS

FZLINT - S/3 BASIC INTERPRETER LINE NR TRACE ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 388
8447				*****			
8448	*			5703-XM1 COPYRIGHT IBM CORP. 1970			*
8449	*			REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
8450	*						*
8451				*****			*
8452	*			*STATUS			*
8453	*			VERSION 1 MODIFICATION 0			*
8454	*						*
8455	*			*FUNCTION			*
8456	*			* FZLINT EXECUTION CAUSES THE CURRENT EXECUTION LINE NUMBER TO			*
8457	*			BE OUTPUT ON THE SYSTEM PRINT DEVICE(S).			*
8458	*			* THIS ROUTINE CONVERTS THE BINARY LINE NUMBER IN SYSTEM PARA-			*
8459	*			METER \$INLNO TO A 4-DIGIT DECIMAL INTEGER, AND OUTPUTS THIS			*
8460	*			VALUE ON THE SYSTEM PRINT DEVICE USING THE FOLLOWING RULES -			*
8461	*			* IF THE PRINT DEVICE CARRIER/CURSOR IS POSITIONED SUCH THAT			*
8462	*			MORE THAN SEVEN CHARACTERS CAN BE OUTPUT ON THE CURRENT			*
8463	*			PRINT LINE, THE LINE NUMBER IS PRINTED AS ' *NNNN ' AND			*
8464	*			THE CARRIER/CURSOR IS LEFT AT THE PRINT POSITION WHICH			*
8465	*			FOLLOWS THE OUTPUT FIELD.			*
8466	*			* IF THE PRINT DEVICE CARRIER/CURSOR IS POSITIONED SUCH THAT			*
8467	*			SIX OR SEVEN CHARACTERS (EXACTLY) CAN BE OUTPUT ON THE			*
8468	*			CURRENT PRINT LINE, THE LINE NUMBER IS PRINTED AS ' *NNNN '			*
8469	*			AND THE CARRIER/CURSOR IS RETURNED TO THE START OF THE			*
8470	*			NEXT PRINT LINE.			*
8471	*			* IF THE PRINT DEVICE CARRIER CURSOR IS POSITIONED SUCH THAT			*
8472	*			LESS THAN SIX CHARACTERS CAN BE OUTPUT ON THE CURRENT			*
8473	*			PRINT LINE, THE CARRIER/CURSOR IS RETURNED TO THE START OF			*
8474	*			NEXT PRINT LINE AND THE LINE NUMBER IS PRINTED AS			*
8475	*			' *NNNN ' AS SPECIFIED IN THE FIRST CASE ABOVE.			*
8476	*			* EITHER THE MATRIX PRINTER OR THE CRT (OR BOTH) MAY BE USED FOR			*
8477	*			OUTPUT, DEPENDING ON THE CURRENT DEFINITION OF THE SYSTEM PRINT			*
8478	*			DEVICE. CRT OUTPUT IS BASED ON A FIXED DISPLAY WIDTH OR 64			*
8479	*			CHARACTERS, WHILE PRINTER LINE WIDTH IS BASED ON THAT ASSIGNED			*
8480	*			THROUGH THE 'WIDTH' SYSTEM COMMAND.			*
8481	*						*
8482	*			*ENTRY POINTS			*
8483	*			THIS ROUTINE HAS A SINGLE ENTRY POINT - FZLINT - WHOSE FUNCTION			*
8484	*			IS DEFINED ABOVE. CALLING SEQUENCE IS:			*
8485	*			B IPGCAL			*
8486	*			DC AL2(V\$DTLN)			*
8487	*			WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL ADDRESS			*
8488	*			OR ENTRY POINT FZLINT. EXECUTION IS SUBJECT TO INPUT CONDITIONS			*
8489	*			DESCRIBED BELOW.			*
8490	*						*
8491	*			*INPUT			*
8492	*			* \$INLNO - 2 BYTES, FOR THE EXECUTION LINE NUMBERS. THIS CONTAINS			*
8493	*			THE BINARY LINE NUMBER ASSOCIATED WITH THE BASIC PROGRAM STATE-			*
8494	*			MENT BEING EXECUTED.			*
8495	*			* \$PRPOS - 1 BYTE, FOR THE MATRIX PRINTER CARRIER POSITION INDI-			*
8496	*			CATOR. THIS CONTAINS THE CARRIER POSITION, RELEATIVE TO THE			*
8497	*			HARDWARE LEFT MARGIN AS 0, OF THE MATRIX PRINTER CARRIER.			*
8498	*			* \$RMRGN - 1 BYTE, FOR THE SOFTWARE RIGHT MATGIN INDICATOR.			*
8499	*			* \$CRPOS - 1 BYTE, FOR THE CRT CURSOS POSITION INDICATOR. THIS			*
8500	*			CONTAINS THE CURSOS POSITION, RELATIVE TO THE LEFT CRT MARGIN			*
8501	*			AS 0, OF THE CRT CURSOR.			*
8502	*			* \$PRDEV - 2 BYTES, FOR THE SYSTEM PRINT DEVICE INDICATOR.			*

FZLINT - S/3 BASIC INTERPRETER LINE NR TRACE ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 389
		8503	*	* \$EXFTR - 1 BYTE, FOR THE SYSTEM CORE EXTENSION FACTOR.	*
		8504	*		*
		8505	*	*OUTPUT	*
		8506	*	* PRINTED OUTPUT AND CARRIER/CURSOR CONTROL - AS SPECIFIED UNDER	*
		8507	*	'FUNCTION' ABOVE.	*
		8508	*	* \$PRPOS - 1 BYTE, FOR THE MATRIX PRINTER CARRIER POSITION INDI-	*
		8509	*	CATOR. THIS IS MODIFIED TO INDICATE THE CURRENT CARRIER POST-	*
		8510	*	TION AFTER PRINTED OUTPUT.	*
		8511	*	* \$CRPOS - 1 BYTE, FOR THE CRT CURSOR POSITION INDICATOR. THIS	*
		8512	*	IS MODIFIED TO INDICATE THE CURRENT CURSOR POSITION AFTER DIS-	*
		8513	*	PLAYED OUTPUT.	*
		8514	*		*
		8515	*	*EXTERNAL REFERENCES	*
		8516	*	* V\$SPRT - VIRTUAL ENTRY ADDRESS FOR DFPRNT, V.M. MATRIX PRT IOCS	*
		8517	*	* DSPLYN - ENTRY POINT FOR THE SYSTEM CRT ROCS (LABEL DSPLYN IS	*
		8518	*	REFERENCED INDIRECTLY USING I\$CSXA TO BUILD A CORE ADDRESS).	*
		8519	*	* I\$CALL - ENTRY POINT FOR PAGING MODULE V.M. PROGRAM CALL RTN.	*
		8520	*	* I\$RTRN - ENTRY POINT FOR PAGING MODULE V.M. RETURN CONTROL RTN.	*
		8521	*	* I\$CSXA - CORE ADDRESS OF 1ST BYTE IN CORE EXTENSION PAST 8K.	*
		8522	*	* \$INLNO - 2 BYTES, FOR THE BINARY EXECUTION LINE NUMBER.	*
		8523	*	* \$PRPOS - 1 BYTE FOR MARTRIX PRINTER CARRIER POSITION INDICATOR.	*
		8524	*	* \$RMGRN - 1 BYTE, FOR POSITION OR SOFTWARE RIGHT PRINTER MARGIN.	*
		8525	*	* \$CRPOS - 1 BYTE, FOR THE CRT CURSOR POSITION INDICATOR.	*
		8526	*	* \$PRDEV - 2 BYTES, FOR THE SYSTEM PRINT DEVICE INDICATOR.	*
		8527	*	* \$EXFTR - 1 BYTE, FOR THE SYSTEM CORE EXTENSION FACTOR.	*
		8528	*		*
		8529	*	*EXITS, NORMAL	*
		8530	*	CONTROL IS ALWAYS PASSED TO THE PAGING ROUTINE AT ENTRY POINT	*
		8531	*	I\$RTRN (IPGRIN) FOR A RETURN TO THE CALLING PROGRAM.	*
		8532	*		*
		8533	*	*EXITS, ERROR	*
		8534	*	NONE	*
		8535	*		*
		8536	*	*TABLESNORK AREAS	*
		8537	*	* LINE NUMBER PRINT OUTPUT AREA - 7 BYTES, TO ACCOMADATE	*
		8538	*	OUTPUT FORMAT SPECIFIED UNDER FUNCTION.	*
		8539	*	* NUMBER CONVERSION WORK AREAS - 7 BYTES, FOR BINARY TO DECIMAL	*
		8540	*	LINE NUMBER CONVERSION.	*
		8541	*	* PRINT PARAMETER LISTS - TWO 4-BYTE PARAMETER LISTS FOR LINE	*
		8542	*	NUMBER OUTPUT AND CARRIER/CURSOR RETURN CONTROL.	*
		8543	*		*
		8544	*	*ATTRIBUTES	*
		8545	*	* REUSABLE	*
		8546	*	* NATURALLY RELOCATABLE	*
		8547	*		*
		8548	*	*CHARACTER-CODE DEPENDENCY	*
		8549	*	THE OPERATION OF THIS MODULE DEPENDS UPON AN INTERNAL REPRESENTA-	*
		8550	*	TION OF THE EXTERNAL CHARACTER SET WHICH IS EQUIVALENT TO THE ONE	*
		8551	*	USED AT ASSEMBLY TIME. THE CODING HAS BEEN ARRANGED SO THAT	*
		8552	*	REDEFINITION OF CHARACTER CONSTANTS, BY REASSEMBLY, WILL RESULT	*
		8553	*	IN A CORRECT MODULE FOR THE NEW DEFINITIONS.	*
		8554	*	.	*
		8555	*	*NOTES	*
		8556	*	ERROR PROCEDURES	*
		8557	*	FZLINT UTILIZES OUTPUT IOCS ROUTINES DFPRTN (MATRIX PRINTER)	*
		8558	*	AND DSPLYN (CRT), AND IS SUBJECT TO THE ERP'S INHERENT IN	*

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 390
		8559	*	THESE PROGRAMS. FZLINT OTHERWISE CONTAINS NO ERROR TESTS.			*
		8560	*				*
		8561	*	REGISTER USAGE			*
		8562	*	* REGISTER @BR IS TO CONTAIN THE CORE PAGE BASE ADDRESS			*
		8563	*	ESTABLISHED THROUGH PAGING MODULE CONTROL FOR THE PAGE WHICH			*
		8564	*	INCLUDES FZLINT, AND IS RESTORED THROUGH THE PAGING MODULE.			*
		8565	*	* REGISTER @XR IS NOT SAVED. IT IS USED IN FZLINT FOR GENERAL			*
		8566	*	PURPOSE INDEXING OPERATIONS.			*
		8567	*				*
		8568	*	SAVED/RESTORED AREAS			*
		8569	*	NONE			*
		8570	*				*
		8571	*	MODIFICATION CONSIDERATIONS			*
		8572	*	NONE			*
		8573	*				*
		8574	*	REQUIRED MODULES			*
		8575	*	* @SYSEQ - COMMON SYSTEM EQUATES.			*
		8576	*	* @FXDEQ - SYSTEM NUCLEUS ADDRESSES AND INDICATOR EQUATES.			*
		8577	*	* \$V\$EQU - VIRTUAL MEMORY FIXED ADDRESS EQUATES.			*
		8578	*	* \$B@EQU - COMPILER PARAMETER AND CONSTANT EQUATES.			*
		8579	*	* \$I\$EQU - INTERPRETER FIXED LOCATION ADDRESS EQUATES.			*
		8580	*				*
		8581	*	OTHER			*
		8582	*	NONE			*
		8583	*	*****			*

FZLINT - S/3 BASIC INTERPRETER LINE NR TRACE ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 391
		8585		*****	
		8586	*	START OF LINE NUMBER TRACE EXECUTION MODULE	*
		8587		*****	
		8588	*		
		8589	*	ESTABLISH VIRTUAL PAGE ADDRESSABILITY	
		8590	*		
4600		8591	*FZLP1B	VPAGE 0	SET PAGE ADDRESSABILITY
		8592		ORG *,256,0	SET STARTING ADDRESS
	4600	8593	FZLP1B	EQU *	START OF PROGRAM CODING
4501		8594		ORG *-255	RESET IAR TO PAGE
4600		8595		ORG *,256,0	BOUNDARV ADDRESS
	4600	8596		USING *,@BR	SET PAGE BASE ADDRESS
4600		8597		ORG FZLP1B	RESET START/NS ADDRESS
		8598	***	END OF EXPANSION ***	
		8599	*		
		8600		*****	

FZLINT - S/3 BASIC INTERPRETER LINE NR TRACE ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 392
					8602	*****	*****	
					8603	* LINE NUMBER TRACE ROUTINE ENTRY POINT	*	
					8604	*****	*****	
					8605	*		
					8606	* ENTER FZLINT - STORE CURRENT EXECUTION LINE NO. FOR CONVERSION.		
					8607	*		
4600	4C	01	BC	03CF	4600	8608	FZLINT EQU * FZLINT ENTRY POINT	
					8609	MVC	FZLBLN(, @BR), \$INLNO(B@LSNO) MOVE LINE NO. TO WORK AREA	
					8610	*		
					8611	* CONVERT 2-BYTE BINARY LINE NO. TO A ZONED DECIMAL VALUE		
					8612	*		
4605	54	30	C0	B7	8613	FZL010 ZAZ	FZLDAC(B@LDSN, @BR), FZLDC1(1, @BR) SET DEC ACCUMULATOR = 1	
4609	54	30	C6	BF	8614	ZAZ	FZLDLN(B@LDSN, @BR), FZLDAC-1(1, @BR) SET DECML LINE NO. = 0	
460D	7C	01	11		8615	FZL020 MVI	FZL030+@Q(, @BR), @B1 SET BINARY MASK FOR 2**0 BIT	
4610	78	00	BC		8616	FZL030 TBN	FZLBLN(, @BR), *- * TEST BINARY LINE NUMBER BIT	
4613	F2	90	04		8617	JF	FZL040 * AND BRANCH IF BIT IS ZERO	
4616	56	03	C6	C0	8618	AZ	FZLDLN(B@LDSN, @BR), FZLDAC(B@LDSN, @BR) INCR DECML LINE NR.	
461A	56	03	C0	C0	8619	FZL040 AZ	FZLDAC(B@LDSN, @BR), FZLDAC(B@LDSN, @BR) DOUBLE DECAL ACCUM	
461E	F2	88	0E		8620	JOZ	FZL050 EXIT LOOP IF ACCUMULATOR > 8192	
4621	5E	00	11	11	8621	ALC	FZL030+@Q(, @BR), FZL030+@Q(1, @BR) SHIFT BINARY MASK LEFT	
4625	D0	20	10		8622	BNOL	FZL030(, @BR) BRANCH IF MASK STILL HAS A BIT	
4628	5C	00	BC	BB	8623	MVC	FZLBLN(, @BR), FZLBLN-1(1, @BR) MOVE LEFT BIN BYTE TO RIGHT	
462C	D0	87	0D		8624	B	FZL020(, @BR) REPEAT LOOP FOR LEFT BIN BYTE	
					8625	*		
					8626	* ESTABLISH PRINT OUTPUT AREA CORE ADDRESS		
					8627	*		
462F	D2	02	C1		8628	FZL050 LA	FZLPAR(, @BR), @XR STORE PRINT AREA CORE ADDRESS	
4632	74	02	CB		8629	ST	FZLPAD(, @BR), @XR * IN PRINT PARAMETER LIST	
					8630	*		
					8631	* DETERMINE POSSIBLE CORE ENTRY ADDRESS FOR CRT IOCR - THIS ADDRESS		
					8632	* IS USED TO TEST WHICH PRINT DEVICES ARE ACTIVE ON THE SYSTEM.		
					8633	*		
4635	5C	01	B0	B9	8634	MVC	FZL230+@OP1(, @BR), FZLPDA(@CADDR, @BR) SET UP CRT BASE ADDR	
4639	4E	00	AF	043B	8635	ALC	FZL230+@OP1-1(, @BR), \$EXFTR(1) * AND ADD EXTENSION FACTOR	
					8636	*		
					8637	* TEST FOR TYPE OF PRINT DEVICE(S) ACTIVE ON SYSTEM		
					8638	*		
463E	1D	01	044B	B0	8639	CLC	\$PRDEV, FZL230+@OP1(@CADDR, @BR) TEST PRINT DEVICE PARAM	
4643	F2	02	16		8640	JNL	FZL054 * AND BRANCH IF ONLY CRT ACTIVE	
					8641	*		
					8642	* PRINTER (AND POSSIBLY CRT) ACTIVE - TEST FOR CURRENT MATRIX PRINTER		
					8643	* LINE CAPACITY		
					8644	*		
4646	7C	02	95		8645	FZL052 MVI	FZL110+@Q(, @BR), @BNL SET CRT TEST AFTER M.P. OUTPUT	
4649	7C	80	9F		8646	MVI	FZL210+@Q(, @BR), @NOP SET OUTPUT ROUTINE FOR PRINTER	
					8647	*		
464C	7C	06	BA		8648	MVI	FZLPAL(, @BR), B@LDSN+2 SET MINIMUM PRINT FIELD LENGTH	
464F	4E	00	BA	03C2	8649	ALC	FZLPAL(, @BR), \$PRPOS(1) ADD CURRENT CARRIER POSITION	
4654	4D	00	BA	03C0	8650	CLC	FZLPAL(, @BR), \$RMGRN(1) * TO THE PRINT FIELD LENGTH	
4659	F2	87	11		8651	J	FZL058 GO PERFORM RIGHT MARGIN BRANCH	
					8652	*		
					8653	* CRT ACTIVE - TEST FOR CURRENT OR DISPLAY LINE CAPACITY		
					8654	*		
465C	7C	80	95		8655	FZL054 MVI	FZL110+@Q(, @BR), @NOP DISABLE CRT TEST AFTER CRT O/P	
465F	7C	87	9F		8656	MVI	FZL210+@Q(, @BR), @UCB SET OUTPUT ROUTINE FOR CRT UNIT	
					8657	*		

FZLINT - S/3 BASIC INTERPRETER LINE NR TRACE ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 393
4662	7C	06	BA	8658		MVI	FZLPAL(,@BR),B@LDSN+2			SET MINIMUM PRINT FIELD LENGTH
4665	4E	00	BA 03E2	8659		ALC	FZLPAL(,@BR),\$CRPOS(1)			ADD CURRENT CRT CURSOR POSITION
466A	7D	40	BA	8660		CLI	FZLPAL(,@BR),@DLNLG			* TO THE DISPLAY FIELD LENGTH
				8661	*					
466D	F2	84	0D	8662	FZL058	JH	FZL070			BRANCH IF RIGHT MARGIN EXCEEDED
4670	F2	82	10	8663		JL	FZL080			BRANCH IF RIGHT MARGIN NOT HIT
				8664	*					
				8665	*		RIGHT MARGIN NIT - SET LINE NUMBER OUTPUT FORMAT ' *NNNN' FOLLOWED			
				8666	*		WITH A CARRIER RETURN			
				8667	*					
4673	7C	C0	C8	8668	FZL060	MVI	FZLPFN(,@BR),@PRETR			SET PRINT & RETURN PPL FUNC
4676	7C	06	C9	8669		MVI	FZLPCT(,@BR),B@LDSN+2			SET MINIMUM PRINT FIELD LENGTH
4679	C0	87	4689	8670		B	FZL090			BRANCH TO PRINT THE LINE NO.
				8671	*					
				8672	*		RIGHT MARGIN EXCEEDED - RETURN CARRIER BEFORE PRINTING LINE NO,			
				8673	*					
467D	D2	02	CC	8674	FZL070	LA	FZLRPL(,@BR),@XR			LOAD THE CARR RETURN PPL CADDR
4680	D0	87	9B	8675		B	FZL200(,@BR)			LINK TO EXECUTE CARR RETURN
				8676	*					
				8677	*		SET NORMAL LINE NUMBER OUTPUT FORMAT ' *NNNN '			
				8678	*					
4683	7C	40	C8	8679	FZL080	MVI	FZLPFN(,@BR),@PRINT			SET PRINT PPL FUNCTION
4686	7C	07	C9	8680		MVI	FZLPCT(,@BR),B@LDSN+3			SET NORMAL PRINT FIELD LENGTH
				8681	*					
				8682	*		PRINT THE DECIMAL LINE NUMBER			
				8683	*					
4689	D2	02	C8	8684	FZL090	LA	FZLPPL(,@BR),@XR			LOAD THE PRINT PPL CADDR
468C	D0	87	9B	8685		B	FZL200(,@BR)			LINK TO PRINT THE LINE NUMBER
				8686	*					
				8687	*		LINE NUMBER PRINTED - TEST ROR OUTPUT TO THE CRT DISPLAY UNIT			
				8688	*					
468F	1D	00	044A AF	8689	FZL100	CLC	\$PRDEV-1,FZL230+@OP1-1(1,@BR)			TEST PRINT DEVICE PARAMETER
				8690	*					GO OUTPUT TO CRT WHEN BRANCH IS
4694	D0	00	5C	8691	FZL110	BC	FZL054(,@BR),*-*			* ENABLED AND PARAMETER = CRT
				8692	*					
				8693	*		LINE NUMBER OUTPUT COMPLETED - RETURN TO THE INTERPRETER			
				8694	*					
4697	C0	87	12D3	8695	FZL120	B	I\$RTRN			RETURN TO INTERPRETER
				8696	*					
				8697	*		*****			

FZLINT - S/3 BASIC INTERPRETER LINE NR TRACE ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 394
		8699		*****	
		8700	*	SUBROUTINE - PRINTS SPECIFIED FIELD AND CONTROLS CARRIER FOR	*
		8701	*	M4TRIX PRINTER AND/OR CRT AS DEFINED BY SYSTEM DEVICE PARAMETER	*
		8702		*****	
		8703	*		
469B	74 08 B6	8704	FZL200 ST	FZL250+@OP1(,@BR),@ARR	STORE RETURN BRANCH ADDRESS
		8705	*		
469E	F2 00 09	8706	FZL210 JC	FZL220,*-*	BRANCH IF ROUTINE SET FOR CRT
		8707	*		
		8708	*	ROUTINE SET FOR MATRIX PRINTER - PERFORM PRINTER OUTPUT	
		8709	*		
46A1	C0 87 12B1	8710	B	I\$CALL	LINK TO EXECUTE PRINTER IOCR
46A5	2800	46A6 8711	DC	AL(@VADDR)(V\$SPRT)	MATRIX PRINTER IOCR VADDR
		8712	*		
46A7	F2 87 09	8713	J	FZL250	GO EXECUTE RETURN BRANCH
		8714	*		
		8715	*	ROUTINE SET FOR CRT DISPLAY UNIT - PERFORM CRT OUTPUT	
		8716	*		
46AA	74 02 B2	8717	FZL220 ST	FZL240(,@BR),@XR	STORE PRINT PARAM LIST CADDR
		8718	*		
46AD	C0 87 0000	8719	FZL230 B	*-*	LINK TO EXECUTE THE CRT !OCR
46B1		46B2 8720	FZL240 DS	CL(@CADDR)	PRINT PARAMETER LIST CADDR
		8721	*		
		8722	*	RETURN CONTROL TO THE CALLING PROGRAM	
		8723	*		
46B3	C0 87 0000	8724	FZL250 B	*-*	RETURN BRANCH
		8725	*		
		8726		*****	

FZLINT - S/3 BASIC INTERPRETER LINE NR TRACE ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 395
		8728		*****		
		8729		* LINE NUMBER TRACE ROUTINE CONSTANTS		*
		8730		*****		
		8731		*		
46B7	F1	46B7	8732	FZLDC1 DC	DL1'1'	DECIMAL INTEGER +1
		8733		*		
46B8	2004	46B9	8734	FZLPDA DC	AL(@CADDR)(I\$CSXA+4)	CRT CORE ENTRY ADDRESS BASE
		8736		*****		
		8737		* LINE NUMBER TRACE ROUTINE WORK AREAS		*
		8738		*****		
		8739		*		
46BA		46BA	8740	FZLPAL DS	CL1	PRINT AREA LENGTH WORK AREA
46BB		46BC	8741	FZLBLN DS	CL(B@LSNO)	BINARY LINE NUMBER WORK AREA
46BD		46C0	8742	FZLDAC DS	CL(B@LDSN)	DECIMAL CONVERSION ACCUMULATOR
		8743		*		
		46C1	8744	FZLPAR EQU	*	PRINT AREA CORE ADDRESS
46C1	405C	46C2	8745	DC	CL2' *'	DECIMAL LINE NO. PREFIX
46C3		46C6	8746	FZLDLN DS	CL(B@LDSN)	DECIMAL LINE NO. PRINT AREA
46C7	40	46C7	8747	DC	CL1' '	DECIMAL LINE NO. POSTFIX
		8748		*		
		8749		*FZLPPL PPL		
		46C8	8750	FZLPPL EQU	*	PPL ADDRESS
46C8	00	46C8	8751	DC	AL1(*-*)	FUNCTION REQUESTED
46C9	00	46C9	8752	DC	AL1(*-*)	PRINT COUNT
46CA	0000	46CB	8753	DC	AL2(*-*)	DATA ADDRESS
		8754		*** END OF EXPANSION ***		
		46C8	8756	FZLPFN EQU	FZLPPL+@PCTRL	PRINT FUNCTION PARAMETER
		46C9	8757	FZLPCT EQU	FZLPPL+@PRCNT	PRINT AREA COUNT PARAMETER
		46CB	8758	FZLPAD EQU	FZLPPL+@PDATA	PRINT AREA CADDR PARAMETER
		8759		*		
		8760		*FZLPL PPL	FUNC=@RETRN,CNT=@RTRNC	
		46CC	8761	FZLRPL EQU	*	PPL ADDRESS
46CC	80	46CC	8762	DC	AL1(@RETRN)	FUNCTION REQUESTED
46CD	80	46CD	8763	DC	AL1(@RTRNC)	PRINT COUNT
46CE	0000	46CF	8764	DC	AL2(*-*)	DATA ADDRESS
		8765		*** END OF EXPANSION ***		
		8766		*		
		8767		*****		
		8768		*		
		8769		*	END OF LINE NUMBER TRACE ROUTINE CODING	*****

FZVART - VARIABLE TRACE ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 396
		8771		*****			
		8772	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
		8773	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
		8774	*				*
		8775		*****			*
		8776	*	*STATUS -			*
		8777	*	VERSION 1 MODIFICATION 0			*
		8778	*				*
		8779	*	*FUNCTION -			*
		8780	*	* FZVART IS DESIGNED TO CONVERT THE VIRTUAL ADDRESS INPUT			*
		8781	*	PARAMETER INTO A VARIABLE NAME AND SUBSCRIPTS (IF APPLICABLE)			*
		8782	*	* THE VARIABLE NAME AND ITS ASSOCIATED VALUE ARE DISPLAYED ON			*
		8783	*	THE SYSTEM PRINTER			*
		8784	*				*
		8785	*	*ENTRY POINTS -			*
		8786	*	* FZVART HAS ONLY ONE ENTRY POINT - FZVART			*
		8787	*	* CALLING SEQUENCE IS			*
		8788	*	B IPGCAL			*
		8789	*	DC AL2(V\$DTR)			*
		8790	*	WHERE V\$DTR IS THE VIRTUAL ADDRESS OF ENTRY POINT FZVART.			*
		8791	*				*
		8792	*	*INPUT -			*
		8793	*	* I\$PARM - 2 BYTES, CONTAINS THE VIRTUAL ADDRESS OF THE			*
		8794	*	VARIABLE TO BE DISPLAYED			*
		8795	*	* SYMBOL AND ARRAY TABLE			*
		8796	*	* LETTER VARIABLE TABLE (LVT) - 58 BYTES, 29 2-BYTE ENTRIES			*
		8797	*	* LETTER DIGIT TABLE (LDT) - 580 BYTES, 290 2-BYTE ENTRIES			*
		8798	*	* CHARACTER VARIABLE TBL (CVT) - 58 BYTES, 29 2-BYTE ENTRIES			*
		8799	*	* ARITHMETIC ARRAY TABLE (NAT) - 58 BYTES, 29 2-BYTE ENTRIES			*
		8800	*	* CHARACTER ARRAY TABLE (CAT) - 58 BYTES, 29 2-BYTE ENTRIES			*
		8801	*	* FUNCTION AND ARRAY TABLE (FAT)			*
		8802	*	* ARITHMETIC ARRAY DOPE VECTOR IMAGES - 29 8-BYTE ENTRIES			*
		8803	*	* CHARACTER ARRAY DOPE VECTOR IMAGES - 29 4-BYTE ENTRIES			*
		8804	*				*
		8805	*	*OUTPUT -			*
		8806	*	* THE VARIABLE NAME AND ITS ASSOCIATED VALUE WILL BE DISPLAYED			*
		8807	*	ON THE SYSTEM OUTPUT DEVICE			*
		8808	*				*
		8809	*	*EXTERNAL REFERENCES -			*
		8810	*	\$DISKN - SYSTEM DISK IOCR			*
		8811	*	I\$PARM - VIRTUAL ADDRESS OF ELEMENT TO BE TRACED			*
		8812	*	IPGCAL - PAGING RTN ENTRY - LOADS VM PAGE PASSES CONTROL			*
		8813	*	IPGLXR - PAGING RTN ENTRY - LOADS VM PAGE SETS @XR			*
		8814	*	IPGRTN - PAGING RTN ENTRY - RETURNS TO BYTE AFTER CALL			*
		8815	*	I\$FWRK - 50 BYTES, FOR RUN-TIME OPERATIONS			*
		8816	*	I\$WRK2 - 2 BYTES, INTERPRETER COMMON WORK AREA 2			*
		8817	*	\$PRPOS - 1 BYTE, MATRIX PRINTER CARRIER POSITION INDICATOR			*
		8818	*	\$CRPOS - 1 BYTE, CRT CURSOR POSITION INDICATOR			*
		8819	*	\$RMRGN - 1 BYTE, POSITION OF SOFTWARE RIGHT PRINTER MARGIN			*
		8820	*	\$PRDEV - 2 BYTES, SYSTEM PRINT DEVICE INDICATORS			*
		8821	*	\$EXFTR - 1 BYTE, CORE EXTENSION FACTOR			*
		8822	*				*
		8823	*	*EXITS, NORMAL -			*
		8824	*	* FZVART HAS 1 NORMAL EXIT TO THE FIRST INSTRUCTION FOLLOWING THE			*
		8825	*	CALLING SEQUENCE, REGISTER @XR IS RESTORED AND THE RETURN			*
		8826	*	ADDRESS IS LOCATED IN THE ADDRESS RECALL REGISTER (@ARR)			*

FZVART - VARIABLE TRACE ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 397

```

8827 *
8828 *EXITS, ERROR -
8829 * N/A
8830 *
8831 *TABLES/WORK AREAS -
8832 * * SYMBOL AND ARRAY TABLES (SEE INPUT)
8833 * * FUNCTION AND ARRAY TABLE (SEE INPUT)
8834 * * ALPHABETIC SYMBOL TABLE - 29 BYTES, CONTAINS ALL BASIC
8835 * ALPHABETIC CHARACTERS
8836 * * I$FWRK - FORMAT BUFFER
8837 *
8838 *ATTRIBUTES -
8839 * * REUSABLE
8840 * * NATURALLY RELOCATEABLE
8841 *
8842 *CHARACTER CODE DEPENDENCY
8843 * THE OPERATION OF THIS MODULE DEPENDS UPON THE FOLLOWING
8844 * PROPERTIES OF THE INTERNAL REPRESENTATION OF THE EXTERNAL
8845 * CHARACTER SET
8846 * * MOST CODING HAS BEEN ARRANGED SO THAT REDEFINITION OF
8847 * CHARACTER CONSTANTS, BY REASSEMBLY, WILL RESULT IN A CORRECT
8848 * MODULE FOR THE NEW DEFINITION
8849 * * ALPHABETIC LETTERS A THROUGH Z ARE PRESUMED TO BE CODED IN
8850 * INCREASING COLLATING SEQUENCE, AND THE RANGE OF CHARACTER
8851 * CONSTANTS FOR THIS SERIES IS EXPECTED TO EXCLUDE ALL NUMERIC
8852 * CHARACTER CONSTANTS
8853 * * NUMERIC CHARACTERS 0 - 9 ARE PRESUMED TO BE CODED IN
8854 * INCREASING COLLATING SEQUENCE
8855 * * EXTENDED ALPHABETIC LETTERS ($, #, @) ARE PRESUMEMED TO BE
8856 * IN INCREASING COLLATING SEQUENCE, AND ARE ALL EXPECTED TO
8857 * COLLATE LOWER THAN LETTER (A)
8858 * * DECIMAL NUMBERS MUST BE CODED SO THAT THE LOW ORDER FOUR
8859 * BITS, WHEN CONSIDERED AS A BINARY INTEGER, IDENTIFY THE
8860 * VALUE OF THE DIGIT
8861 * THE SPECIFIC INSTRUCTIONS (INSTRUCTION SEQUENCES) WHICH REQUIRE
8862 * MODIFICATION IF THESE PROPERTIES OF THE CHARACTER SET ARE CHANGE
8863 * MAY BE IDENTIFIED BY -
8864 * * THE TABLE IDENTIFIED BY LABEL FZVATB
8865 *
8866 *NOTES
8867 * ERROR PROCEDURES
8868 * N/A
8869 *
8870 * REGISTER USAGE
8871 * * REGISTER @BR IS USED AS BASE DURING PROGRAM EXECUTION
8872 * * REGISTER @XR IS SAVED ON ENTRY AND RESTORED ON EXIT
8873 *
8874 * SAVED/RESTORED AREAS
8875 * * THE CORE I/O FUNCTIONS AT CORE ADDRESSES 0700 TO OCFF ARE
8876 * SAVED AT ENTRY FOR USE AS THE SYMBOL TABLE AREA. THIS AREA
8877 * IS RESTORED BEFORE EXIT
8878 *
8879 * MODIFICATION CONSIDERATIONS
8880 * N/A
8881 *
8882 * REQUIRED MODULES

```


FZVART - VARIABLE TRACE ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 398
		8883	*		@SYSEQ - COMMON SYSTEM EQUATES			*
		8884	*		@FXDEQ - SYSTEM NUCLEUS ADDRESSES AND INDICATORS			*
		8885	*		\$V\$EQ - FIXED VIRTUAL ADDRESSES			*
		8886	*		\$B@EQ - COMPILER SYSTEM EQUATES			*
		8887	*		\$I\$EQ - INTERPRETER FIXED EQUATES			*
		8888	*		\$I@SEQ/\$I@LEQ - STANDARD/LONG PRECISION EXECUTION EQUATES			*
		8889	*					*
		8890	*	OTHER				*
		8891	*	NONE				*
		8892	*	*****				

FZVART - VARIABLE TRACE ROUTINE

ERR	LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 399
				8894	*****	
				8895	*	*
				8896	*****	
				8897	*	*
				8898	* FZVART - VARIABLE TRACE ROUTINE	*
				8899	*	*
				8900	*****	
				8901	*	*
				8902	*****	
				8903	*	
				8904	* ESTABLISH ADDRESSABILITY FOR VM PAGE	
				8905	*	
4700				8906	*FZVPG1 VPAGE 0	SET PAGE ADDRESSABILITY
				8907	ORG *,256,0	SET STARTING ADDRESS
			4700	8908	FZVPG1 EQU *	START OF PROGRAM CODING
4601				8909	ORG *-255	RESET IAR TO PAGE
4700				8910	ORG *,256,0	* BOUNDARY ADDRESS
			4700	8911	USING *,@BR	SET PAGE BASE ADDRESS
4700				8912	ORG FZVPG1	RESET STARTING ADDRESS
				8913	*** END OF EXPANSION ***	
				8915	*	
				8916	* ENTER FZVART AND PERFORM REGISTER OPERATIONS	
				8917	*	
4700 74 02 A6			4700	8918	FZVART EQU *	FZVART ENTRY PT
				8919	ST FZV860+@OP1(,@BR),@XR	SAVE PT
				8920	*	
				8921	* SAVE CORE PAGES THE SYMBOL TABLES WILL OVERLAY	
				8922	*	
4703 D2 02 E5				8923	FZV010 LA FZVSAV(,@BR),@XR	LOAD CADDR DPL
4706 74 02 11				8924	ST FZV015(,@BR),@XR	SET CALL INST WITH DPL CADDR
4709 7C 02 E5				8925	MVI FZVSAV+@DCTRL(,@BR),@DPUT	SET DPL TO WRITE
470C C0 87 0025				8926	B \$DISKN	SAVE CORE PGS
4710			4711	8927	FZV015 DS CL(@CADDR)	CADDR DPL
				8928	*	
				8929	* LOAD CORE WITH THE SYMBOL AND ARRAY TABLES	
				8930	*	
4712 D2 02 EB				8931	FZV020 LA FZVSYM(,@BR),@XR	LOAD CADDR DPL
4715 74 02 1D				8932	ST FZV025(,@BR),@XR	SET CALL INST WITH DPL CADDR
4718 C0 87 0025				8933	B \$DISKN	GET THE SYM TBLS
471C			471D	8934	FZV025 DS CL(@CADDR)	CADDR DPL
471E C0 87 0025				8935	B \$DISKN	WAIT FOR COMPLETION
4722 057F			4723	8936	DC AL(@CADDR)(\$WAITF)	WAIT PARAM
4724 C0 87 12B1				8937	B I\$CALL	CALL RTN TO LOAD D/V TABLES
4728 48A3			4729	8938	DC AL(@VADDR)(FZV155)	VADDR PARAM
				8939	*	
				8940	* CLEAR THE FORMAT BUFFER TO BLANKS	
				8941	*	
472A 3C 40 0628				8942	FZV030 MVI I\$FWRK+33,B@BLNK	CLEAR THE FORMAT BUFFER TO
472E 0C 20 0627 0628				8943	MVC I\$FWRK+32,I\$FWRK+33(FZVL33)	* BLANKS
				8944	*	
				8945	* DETERMINE ELEMENT SYMBOL FROM THE VADDR	
				8946	*	
4734 C0 87 12B1				8947	FZV095 B I\$CALL	GO TO NEXT PAGE
4738 4800			4739	8948	DC AL(@VADDR)(FZVPG2)	VADDB PARAM
				8949	*	

FZVART - VARIABLE TRACE ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 400
					8950	*	STACK THE DATA ELEMENT AT THE VADDR IN I\$PARM	
					8951	*		
473A	C0	87	12B1		8952	FZV775 B	I\$CALL GO TO INTERNAL STACK RTN	
473E	4B00			473F	8953	DC	AL(@VADDR)(FZVPG4) VADDR PARAM	
					8954	*		
					8955	*	SHIFT THE DATA BUCKET TO THE PRINT BUFFER	
					8956	*		
				4B00	8957	USING	FZVPG4,@XR BASE FOR 4TH VM PAGE	
4740	35	02	0D5B		8958	L	I\$WRK2,@XR LOAD INDEX	
4744	3D	40	0609		8959	FZV780 CLI	I\$FWRK+2,B@BLNK A SCALAR REF ?	
4748	F2	81	08		8960	JE	FZV785 YES, SHIFT SCALAR VALUE	
474B	2C	15	0628 EB		8961	MVC	I\$FWRK+33,FZVB21(FZVL22,@XR) SHIFT ARRAY VALUE	
4750	F2	87	05		8962	J	FZV790 SKIP TO FIND BFR LNG	
4753	2C	15	0621 EB		8963	FZV785 MVC	I\$FWRK+26,FZVB21(FZVL22,@XR) SHIFT SCALAR VALUE	
					8964	*		
					8965	*	DETERMINE PRINT LENGTH OF THE PRINT BUFFER	
					8966	*		
4758	C2	02	0606		8967	FZV790 LA	I\$FWRK-1,@XR CADDR PRINT BFR	
475C	7C	22	61		8968	MVI	FZV795+@D1(,@BR),FZVL34 SET MAX BFR LNG	
475F	BD	40	00		8969	FZV795 CLI	*-(,@XR),B@BLNK IS BYTE A BLANK ?	
4762	F2	01	07		8970	JNE	FZV798 NO, SET PPL LNG	
4765	5F	00	61 DA		8971	SLC	FZV795+@D1(,@BR),FZVI01(1,@BR) DECR LNG DISP	
4769	D0	87	5F		8972	B	FZV795(,@BR) REPEAT LOOP	
476C	5C	00	DE 61		8973	FZV798 MVC	FZVPPL+1(,@BR),FZV795+@D1(1,@BR) SET PPL LNG COUNT	
					8975	*****		
					8976	*		
					8977	*	PRINT THE OUTPUT BUFFER	
					8978	*		
					8979	*****		
					8980	*		
					8981	*	TEST FOR CARRIAGE AT LEFT MARGIN	
					8982	*		
4770	0D	00	03C2 03C1		8983	FZV800 CLC	\$PRPOS,\$LMRGN(1) AT LEFT MARGIN ?	
4776	F2	01	09		8984	JNE	FZV810 NO RETURN CARRIAGE	
4779	0D	00	03E2 0000		8985	CLC	\$CRPOS,@ZERO AT LEFT MARGIN ?	
477F	F2	81	06		8986	JE	FZV820 YES, PRINT BFR	
					8987	*		
					8988	*	RETURN THE CARRIAGE	
					8989	*		
4782	D2	02	E1		8990	FZV810 LA	FZVRPL(,@BR),@XR CADDR PPL	
4785	D0	87	AB		8991	B	FZV900(,@BR) RETURN THE CARRIAGE	
					8992	*		
					8993	*	PRINT THE OUTPUT BUFFER	
					8994	*		
4788	D2	02	DD		8995	FZV820 LA	FZVPPL(,@BR),@XR CADDR PPL	
478B	D0	87	AB		8996	FZV830 B	FZV900(,@BR) PRINT BFR	
					8997	*		
					8998	*	RESTORE THE SAVED CORE AREA	
					8999	*		
478E	D2	02	E5		9000	FZV840 LA	FZVSAV(,@BR),@XR CADDR DPL	
4791	74	02	9C		9001	ST	FZV855(,@BR),@XR SET CALL INST WITH DPL CADDR	
4794	7C	01	E5		9002	MVI	FZVSAV+@DCTRL(,@BR),@DGET SET DPL TO READ	
4797	C0	87	0025		9003	FZV850 B	\$DISKN SET THE SAVED CORE PGS	
479B				479C	9004	FZV855 DS	CL(@CADDR) CADDR DPL	
479D	C0	87	0025		9005	B	\$DISKN WAIT FOR COMPLETION	

FZVART - VARIABLE TRACE ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 401

47A1	057F	47A2	9006	DC	AL(@CADDR)(\$WAITF)	WAIT PARAM
			9007	*		
			9008	*	RETURN TO CALLING PROGRAM	
			9009	*		
47A3	C2 02 0000		9010	FZV860 LA	*-*,@XR	RESTORE PT
47A7	C0 87 12D3		9011	B	I\$RTRN	RETURN
			9013	*****		
			9014	*		*
			9015	*	PRINTER & CRT IMAGE OUTPUT ROUTINE	*
			9016	*	* PRINTS SPECIFIED IMAGE AND/OR CONTROLS CARRIER FOR MATRIX	*
			9017	*	* PRINTER AND/OR CRT AS DEFINED BY SYSTEM DEVICE PARAMETER	*
			9018	*		*
			9019	*	INPUT -	*
			9020	*	* REGISTER @XR - CONTAINS CORE ADDRESS OF PRINT PARAMETER LIST	*
			9021	*		*
			9022	*	OUTPUT -	*
			9023	*	* PRINTED LINE AND/OR CARRIER CONTROL AS SPECIFIED IN PARAM LIE	*
			9024	*		*
			9025	*****		
			9026	*		
			9027	*	SUBROUTINE ENTRY - SAVE THE RETURN ADDRESS	
			9028	*		
47AB	74 08 D9		9029	FZV900 ST	FZV980+@OP1(,@BR),@ARR	STORE RETURN ADDR
			9030	*		
			9031	*	DETERMINE POSSIBLE CORE ENTRY ADDRESS FOR CRT IOCR	
			9032	*		
47AE	5C 01 C5 DC		9033	FZV910 MVC	FZV940+@OP1(,@BR),FZVPDA(@CADDR,@BR)	SET UP BASE CADDR
47B2	4E 00 C4 043B		9034	ALC	FZV940+@OP1-1(,@BR),\$EXFTR(1)	* AND ADD EXT FACTOR
			9035	*		
			9036	*	TEST FOR TYPE OF PRINT DEVICE ACTIVE ON SYSTEM	
			9037	*		
47B7	1D 00 044A C4		9038	FZV920 CLC	\$PRDEV-1,FZV940+@OP1-1(1,@BR)	TEST PRINT DEVICE PARAM
47BC	F2 82 11		9039	JL	FZV970	* AND BR IF PRINTER ONLY
			9040	*		
			9041	*	CRT (AND POSSIBLE PRINTER) ACTIVE - OUTPUT PRINT AREA ON THE CRT	
			9042	*		
47BF	74 02 C7		9043	FZV930 ST	FZV950(,@BR),@XR	STORE PPL CADDR
			9044	*		
47C2	C0 87 0000		9045	FZV940 B	*-*	LINK TO EXECUTE THE CRT IOCR
47C6		47C7	9046	FZV950 DS	CL(@CADDR)	PPL CADDR
			9047	*		
			9048	*	TEST FOR MATRIX PRINTER ACTIVE ON THE SYSTEM	
			9049	*		
47C8	1D 01 044B C5		9050	FZV960 CLC	\$PRDEV,FZV940+@OP1(@CADDR,@BR)	TEST PRINT DEVICE PARAM
47CD	F2 02 06		9051	JNL	FZV980	
			9052	*		
			9053	*	MATRIX PRINTER ACTIVE - OUTPUT PRINT AREA ON THE MATRIX PRINTER	
			9054	*		
47D0	C0 87 12B1		9055	FZV970 B	I\$CALL	LINK TO EXECUTE MAT PRINT IOCR
47D4	2800	47D5	9056	DC	AL(@VADDR)(V\$SPRT)	MATRIX PRINTER IOCR VADDR
			9057	*		
			9058	*	RETURN CONTROL TO YNZ CALLING PROGRAM	
			9059	*		
47D6	C0 87 0000		9060	FZV980 B	*-*	RETURN TO CALLINE KYR4
			9061	*		

FZVART - VARIABLE TRACE ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 402

```

          9062 *****
          9064 *****
          9065 *
          9066 * VARIABLE TRACE EQUATES CONTANTS AND WORK AREAS (1ST VM PAGE)
          9067 *
          9068 *****
          9069 *
          9070 * 1ST INTERFACE PAGE EQUATES
          9071 *
          0000 9072 FZVH00 EQU 0 INIT PAGE DISP TO ZERO
          0006 9073 FZVSCN EQU 6 SECTOR COUNT OF OVERLAY
          0008 9074 FZVL08 EQU 8 MAX LNG POSSIBLE OF SUBSC
          0020 9075 FZVL32 EQU 32 BFR LNG TO BLANK
          0021 9076 FZVL33 EQU 33 LNG OF PRINT BFR
          0022 9077 FZVL34 EQU 34 MAX LNG OF THE PRINT BFR
          0041 9078 FZVTBD EQU X'41' SECTOR DISP OF SYM TBLS
          0700 9079 FZVCAD EQU X'0700' BEGINNING CADDR OF OVERLAY
          0959 9080 FZVSDA EQU X'0959' DADDR OF SAVE AREA
          9081 *
          9082 * 1ST INTERFACE PAGE CONSTANTS
          9083 *
          47DA 01 47DA 9084 FZVI01 DC XL1'01' INTEGER OF 1
          47DB 2004 47DC 9085 FZVPDA DC AL(@CADDR)(I$CSXA+4) CRT CORE ENTRY BASE ADDR
          9086 *
          9087 * PRINT PARAMETER LISTS
          9088 *
          9089 *FZVPPL PPL FUNC=@PRETR,CADDR=I$FWRK PRINT PPL
          47DD C0 47DD 9090 FZVPPL EQU * PPL ADDRESS
          47DE 00 47DD 9091 DC AL1(@PRETR) FUNCTION REQUESTED
          47DF 0607 47DE 9092 DC AL1(*-*) PRINT COUNT
          47E0 9093 DC AL2(I$FWRK) DATA ADDRESS
          9094 *** END OF EXPANSION ***

          9096 *FZVRPL PPL FUNC=@RETRN,CNT=@RTRNC CARR RETURN PPL
          47E1 9097 FZVRPL EQU * PPL ADDRESS
          47E1 80 47E1 9098 DC AL1(@RETRN) FUNCTION REQUESTED
          47E2 80 47E2 9099 DC AL1(@RTRNC) PRINT COUNT
          47E3 0000 47E4 9100 DC AL2(*-*) DATA ADDRESS
          9101 *** END OF EXPANSION ***
          9102 *
          9103 * DISK PARAMETER LISTS
          9104 *
          9105 *FZVSAV DPL DADDR=FZVSDA,CNT=FZVSCN,CADDR=FZVCAD
          47E5 9106 FZVSAV EQU * DISK PARAMETER LIST
          47E5 00 47E5 9107 DC AL1(*-*) REQUESTED FUNCTION
          47E6 0959 47E7 9108 DC AL2(FZVSDA) DISK ADDRESS
          47E8 06 47E8 9109 DC AL1(FZVSCN) SECTOR COUNT
          47E9 0700 47EA 9110 DC AL2(FZVCAD) BUFFER ADDRESS
          9111 *** END OF EXPANSION ***

          9113 *FZVSYM DPL FUNC+@DGET,CYL=@DCBCY,SCTR=FZVTBD,CNT=FZVSCN,CADDR=FZVCAD
          47EB 9114 FZVSYM EQU * DISK PARAMETER LIST
          47EB 01 47EB 9115 DC AL1(@DGET) REQUESTED FUNCTION
          47EC 09 47EC 9116 DC AL1(@DCBCY) CYLINDER ADDRESS
          47ED 41 47ED 9117 DC AL1(FZVTBD) HEAD/SECTOR/DRIVE/DISK SPEC

```

FZVART - VARIABLE TRACE ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 403

47EE	06	47EE	9118	DC	AL1(FZVSCN)	SECTOR COUNT
47EF	0700	47F0	9119	DC	AL2(FZVCAD)	BUFFER ADDRESS
			9120	***	END OF EXPANSION	***

FZVART - VARIABLE TRACE ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 404
					9122	*****				
					9123	*				*
					9124	*****				
					9125	*				*
					9126	* SECOND VM INTERFACE PAGE - SCALAR VARIABLE TABLE SEARCH				*
					9127	*				*
					9128	*****				
					9129	*				*
					9130	*****				
					9131	*				
					9132	* ESTABLISH ADDRESSABILITY FOR INTERFACE 2ND VM PAGE				
					9133	*				
4800					9134	*FZVPG2 VPAGE 0	SET PAGE ADDRESSABILITY			
					9135	ORG *,256,0	SET STARTING ADDRESS			
				4800	9136	FZVPG2 EQU *	START OF PROGRAM COOING			
4701					9137	ORG *-255	RESET IAR TO PAGE			
4800					9138	ORG *,256,0	* BOUNDARY ADDRESS			
				4800	9139	USING *,@BR	SET PAGE BASE ADDRESS			
4800					9140	ORG FZVPG2	RESET STARTING ADDRESS			
					9141	*** END OF EXPANSION ***				
					9143	*****				
					9144	*				*
					9145	* LETTER VARIABLE TABLE SCAN				*
					9146	*				*
					9147	*****				
					9148	*				
					9149	* TEST FOR A VIRTUAL ADDRESS IN THE LETTER VARIABLE TABLE				
					9150	*				
4800	7C	39	13		9151	FZV100 MVI FZV110+@D1(,@BR),FZVEND	INIT PT DISP			
4803	7C	1C	95		9152	MVI FZV145+@D1(,@BR),FZVH28	INIT CNT			
4806	7C	40	C4		9153	MVI FZVSSA(,@BR),B@BLNK	SET SYM FOR LETTER VAR			
4809	3C	7E	060A		9154	MVI I\$FWRK+3,B@EQL	SET SCALAR EQUAL SIGN			
480D	C2	02	070C		9155	FZV105 LA FZVLVT,@XR	LETTER VAR TBL CADDR			
4811	E2	02	00		9156	FZV110 LA *-*(,@XR),@XR	SELECT ENTRY			
4814	2D	01	0D57 00		9157	CLC I\$PARM,0(@VADDR,@XR)	VADDRS EQUAL ?			
4819	F2	81	74		9158	JE FZV140	YES, GET SYMBOL			
					9159	*				
					9160	* UPDATE POINTERS				
					9161	*				
481C	5F	00	95 BD		9162	FZV115 SLC FZV145+@D1(,@BR),FZVH01(1,@BR)	DECR TBL CNT			
4820	5F	00	13 BF		9163	SLC FZV110+@D1(,@BR),FZVH02(1,@BR)	DECR PT DISP UNTIL DISP			
4824	D0	84	0D		9164	BH FZV105(,@BR)	* IS 0			
					9166	*****				
					9167	*				*
					9168	* CHARACTER VARIABLE TABLE SCAN				*
					9169	*				*
					9170	*****				
					9171	*				
					9172	* TEST FOR A VIRTUAL ADDRESS IN THE CHARACTER VARIABLE TABLE				
					9173	*				
4827	7C	39	36		9174	FZV120 MVI FZV130+@D1(,@BR),FZVEND	INIT PT DISP			
482A	7C	1C	95		9175	MVI FZV145+@D1(,@BR),FZVH28	INIT TBL CNT			
482D	7C	5B	C4		9176	MVI FZVSSA(,@BR),B@CVAR	SET SM FOR CHAR VAR			
4830	C2	02	098A		9177	FZV125 LA FZVCVT,@XR	CHAR VAR TBL CADDR			

FZVART - VARIABLE TRACE ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 405

4834	E2	02	00		9178	FZV130	LA	*-*(,@XR),@XR	SELECT ENTRY
4837	2D	01	0D57	00	9179		CLC	I\$PARM,0(@VADDR,@XR)	VADDR MATCH ?
483C	F2	81	51		9180		JE	FZV140	YES, GET SYMBOL
					9181	*			
					9182	*	UPDATE	POINTERS	
					9183	*			
483F	5F	00	95	BD	9184	FZV135	SLC	FZV145+@D1(@BR),FZVH01(1,@BR)	DECR TBL CNT
4843	5F	00	36	BF	9185		SLC	FZV130+@D1(@BR),FZVH02(1,@BR)	DECR PT DISP UNTIL DISP
4847	D0	84	30		9186		BH	FZV125(@BR)	* IS 0
					9188	*****			
					9189	*			*
					9190	*	LETTER-DIGIT	VARIABLE TABLE SCAN	*
					9191	*			*
					9192	*****			
					9193	*			
					9194	*	TEST	LETTER-DIGIT TABLE FOR VADDR MATCH	
					9195	*			
484A	5C	01	57	C1	9196	FZV200	MVC	FZV210+@OP1(@CADDR,@BR),FZVLDT(@BR)	SET ENTRY CADDR
484E	7C	1C	95		9197		MVI	FZV145+@D1(@BR),FZVH28	INIT TBL CNT
4851	7C	09	C2		9198	FZV205	MVI	FZVDGT(@BR),FZVH09	
4854	C2	02	0000		9199	FZV210	LA	*-*,@XR	
4858	2D	01	0D57	00	9200		CLC	I\$PARM,0(@VADDR,@XR)	VADDR MATCH ?
485D	F2	81	29		9201		JE	FZV215	YES, SET DIGIT FOR SYMBOL
					9202	*			
					9203	*	UPDATE	POINTERS	
					9204	*			
4860	5F	01	57	BF	9205	FZV220	SLC	FZV210+@OP1(@CADDR,@BR),FZVH02(@BR)	DECR ENTRY PT
4864	5F	00	C2	BD	9206		SLC	FZVDGT(1,@BR),FZVH01(@BR)	DECR DIGIT UNTIL DIGIT IS
4868	D0	02	54		9207		BNM	FZV210(@BR)	* LESS THAN 0
486B	5F	00	95	BD	9208		SLC	FZV145+@D1(@BR),FZVH01(1,@BR)	DECR ALPHA TBL PT UNTIL
486F	D0	02	51		9209		BNM	FZV205(@BR)	* LESS THAN 1 1-3
					9210	*			
					9211	*	NOT A SCALAR VARIABLE, TEST FOR A ARRAY ELEMENT VARIABLE		
					9212	*			
4872	C0	87	12B1		9213	FZV230	B	I\$CALL	GO TO NEXT PG FOR ARRAY TEST
4876	4900			4877	9214		DC	AL(@VADDR)(FZVPG3)	VADDR PARAM
					9215	*			
					9216	*	INITIALIZE	SCALAR ROUTINE FOR ARRAY PARAMETERS	
					9217	*			
4878	4C	00	95	0D5B	9218	FZV240	MVC	FZV145+@D1(1,@BR),I\$WRK2	SET TBL COUNT
487D	4C	00	C4	0D5A	9219		MVC	FZVSSA(1,@BR),I\$WRK2-1	SET SYM TYPE
4882	3C	7E	0611		9220		MVI	I\$FWRK+10,B@EQL	SET ARRAY = SIGN
4886	F2	87	07		9221		J	FZV140	FIND SYMBOL
					9222	*			
					9223	*	SET	DIGIT IN SYMBOL WORK IMAGE	
					9224	*			
4889	7A	F0	C2		9225	FZV215	SBN	FZVDGT(@BR),B@ZPOS	SET DIGIT TO DEC
488C	5C	00	C4	C2	9226		MVC	FZVSSA(1,@BR),FZVDGT(@BR)	SHIFT DIGIT

FZVART - VARIABLE TRACE ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 406

```

          9228 *****
          9229 *
          9230 * DETERMINE THE SYMBOL, THAT CORRESPONDS TO THE VADDR
          9231 *
          9232 *****
          9233 *
4890 D2 02 C5      9234 FZV140 LA      FZVATB(,@BR),@XR      CADDR ALPHA TBL
4893 E2 02 00      9235 FZV145 LA      *-*(,@XR),@XR      INCR TO SYMBOL
4896 6C 00 C3 00   9236          MVC      FZVSSA-1(1,@BR),0(,@XR)  SAVE LETTER
          9237 *
          9238 * MOVE SYMBOL WORK AREA TO THE PRINT BUFFER
          9239 *
489A 1C 01 0608 C4 9240 FZV150 MVC      I$FWRK+1,FZVSSA(FZVSLN,@BR)  PUT SYM IN FORMAT BFR
489F C0 87 12D3    9241          B      I$RTRN      RETURN TO 1ST PG TO PRINT BFR

          9243 *****
          9244 *
          9245 * OVERLAY THE FUNCTION AND ARRAY TABLE WITH VM ONE
          9246 *
          9247 *****
48A3 C0 87 1330    9248 FZV155 B      I$LDXR      GET 1ST PG OF FAT
48A7 FEFF          48A8 9249          DC      AL(@VADDR)(FZV2LS)  VADDR PARAM
48A9 2C 94 0B07 00 9250          MVC      FZV1PG(FZV1LN),0(,@XR)  SHIFT TO TBL AREA
48AE C0 87 1330    9251          B      I$LDXR      GET 2ND PG OF FAT
48B2 FFFF          48B3 9252          DC      AL(@VADDR)(FZVLST)  VADDR PARAM
48B4 2C FF 0C07 00 9253          MVC      FZV2PG(B@LVP),0(,@XR)  SHIFT TO TBL AREA
48B9 C0 87 12D3    9254          B      I$RTRN      RETURN

          9256 *****
          9257 *
          9258 * VARIABLE TRACE EQUATES, CONSTANTS AND WORK AREAS (2ND VM PAGE)
          9259 *
          9260 *****
          9261 *
          9262 * 2ND INTERFACE PAGE EQUATES
          9263 *
          0002 9264 FZVSLN EQU      2
          0009 9265 FZVH09 EQU      9      INIT VALUE FOR DIGIT WORK AREA
          001C 9266 FZVH28 EQU      28     LNG OF ALPHA TBL -1
          0039 9267 FZVEND EQU      X'39'  DISP TO LAST ENTRY IN TBLS
          FEFF 9268 FZV2LS EQU      X'FEFF' RH BYTE IN 1ST PG FAT
          FFFF 9269 FZVLST EQU      X'FFFF' RH BYTE IN 2ND PG FAT
          0B07 9270 FZV1PG EQU      X'0B07' CADDR OF RH BYTE OF 1ST PG FAT
          0C07 9271 FZV2PG EQU      X'0C07' CADDR OF RH BYTE OF 2ND PG FAT
          0095 9272 FZV1LN EQU      X'95'  LNG OF 1ST SEGMENT OF FAT
          9273 *
          9274 * 2ND INTERFACE PAGE CONSTANTS
          9275 *
48BD 01          48BD 9276 FZVH01 DC      IL1'01'      INTEGER 1
48BE 0002        48BF 9277 FZVH02 DC      IL2'02'      INTEGER OF TWO
48C0 0989        48C1 9278 FZVLDT DC      AL(@CADDR)(FZVCVT-1) CADDR LAST BYTE LDT
          9279 *
          9280 * 2ND INTERFACE PAGE WORK AREAS
          9281 *
48C2          48C2 9282 FZVDGT DS      CL1      DIGIT WORK AREA
48C3          48C4 9283 FZVSSA DS      CL(FZVSLN)  SYMBOL SAVE AREA

```

FZVART - VARIABLE TRACE ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 407

9284 *
9285 * FZVART ALPHABETIC TABLE
9286 *

48C5 5B7B7CC1C2C3C4C5 48C5 9287 FZVATB EQU * CADDR OF ALPHA TBL
48E1 9288 DC CL29 '\$#@ABCDEFGHIJKLMNOPQRSTUVWXYZ' ALPHA TBL

FZVART - VARIABLE TRACE ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 408
					9290	*****				
					9291	*				*
					9292	*****				
					9293	*				*
					9294	*	THIRD VM INTERFACE PAGE - ARRAY TABLE SEARCH			*
					9295	*				*
					9296	*****				
					9297	*				*
					9298	*****				
					9299	*				
					9300	*	ESTABLISH ADDRESSABILITY FOR INTERFACE 3RD VM PAGE			
					9301	*				
4900					9302	*FZVPG3	VPAGE 0 SET PAGE ADDRESSABILITY			
					9303	ORG	*,256,0 SET STARTING ADDRESS			
	4900				9304	FZVPG3 EQU	* START OF PROGRAM CODING			
4801					9305	ORG	*-255 RESET IAR TO PAGE			
4900					9306	ORG	*,256,0 * BOUNDARY ADDRESS			
	4900				9307	USING	*,@BR SET PAGE BASE ADDRESS			
4900					9308	ORG	FZVPG3 RESET STARTING ADDRESS			
					9309	***	END OF EXPANSION ***			
					9311	*****				
					9312	*				*
					9313	*	ARITHMETIC ARRAY TABLE SCAN			*
					9314	*				*
					9315	*****				
					9316	*				
					9317	*	INITIALIZE TABLE SCAN			
					9318	*				
4900	7C	39	0F		9319	FZV300 MVI	FZV310+@D1(,@BR),FZVEND INIT PT DISP			
4903	7C	1C	EB		9320	MVI	FZVHLD(,@BR),FZVH28 INIT TBL CNT			
4906	7C	4D	EA		9321	MVI	FZVTYP(,@BR),B@LPAR			
					9322	*				
					9323	*	TEST NUMERIC ARRAY TABLE FOR A VIRTUAL ADDRESS			
					9324	*				
4909	C2	02	09C4		9325	FZV305 LA	FZVNAT,@XR NUMERIC ARRAY TBL CADDR			
490D	B5	02	00		9326	FZV310 L	*-(,@XR),@XR SELECT ENTRY			
4910	76	02	DF		9327	A	FZVI00(,@BR),@XR CONTAIN A VADDR ?			
4913	F2	01	2C		9328	JNE	FZV400 YES, CHECK FOR VADDR MATCH			
					9329	*				
					9330	*	UPDATE POINTERS			
					9331	*				
4916	5F	00	EB E1		9332	FZV320 SLC	FZVHLD(1,@BR),FZVIN1(,@BR) DECR TBL CNT			
491A	5F	00	0F E3		9333	SLC	FZV310+@D1(,@BR),FZVI02(1,@BR) DECR PT DISP UNTIL			
491E	D0	84	09		9334	BH	FZV305(,@BR) * DISP IS 0			
					9335	*				
					9336	*****				
					9337	*				*
					9338	*	CHARACTER ARRAY TABLE SCAN			*
					9339	*				*
					9340	*****				
					9341	*				
					9342	*	INITIALIZE TABLE SCAN			
					9343	*				
4921	7C	39	30		9344	FZV330 MVI	FZV340+@D1(,@BR),FZVEND INIT PT DISP			
4924	7C	1C	EB		9345	MVI	FZVHLD(,@BR),FZVH28 INIT TBL COUNT			

FZVART - VARIABLE TRACE ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 409
	4927	7C	5B	EA	9346	MVI	FZVTYP(, @BR), B@CVAR	SET VAR TYPE
					9347	*		
					9348	*	TEST CHARACTER ARRAY TABLE FOR A VIRTUAL ADDRESS	
					9349	*		
	492A	C2	02	09FE	9350	FZV335 LA	FZVCAT, @XR	CHAR ARRAY TBL CADDR
	492E	B5	02	00	9351	FZV340 L	*-(, @XR), @XR	SELECT ENTRY
	4931	76	02	DF	9352	A	FZVI00(, @BR), @XR	CONTAIN A VADDR ?
	4934	F2	01	50	9353	JNE	FZV470	YES, CHECK FOR VADDR MATCH
					9354	*		
					9355	*	UPDATE POINTERS	
					9356	*		
	4937	5F	00	EB E1	9357	FZV350 SLC	FZVHLD(1, @BR), FZVIN1(, @BR)	DECR TBL CNT
	493B	5F	00	30 E3	9358	SLC	FZV340+@D1(, @BR), FZVI02(1, @BR)	DECR PT DISP UNTIL
	493F	D0	87	2A	9359	B	FZV335(, @BR)	* DISP IS 0
					9361	*****		
					9362	*		*
					9363	*	ARITHMETIC DOPE VECTOR SCAN	*
					9364	*		*
					9365	*****		
					9366	*		
					9367	*	CONVERU VADDR TO CADDR	
					9368	*		
	4942	76	02	E9	9369	FZV400 A	FZVAAC(, @BR), @XR	CONVERT VADDR
	4945	5C	01	ED E1	9370	MVC	FZVWS1(, @BR), FZVIN1(B@LBIN, @BR)	SET SUBSC TO 1
	4949	9D	01	01 DF	9371	CLC	B@ACD1(, @XR), FZVI00(B@LBIN, @BR)	IS D/V A VECTOR ?
	494D	F2	01	04	9372	JNE	FZV415	NO, LEAVE SUBSC AS 1
					9373	*		
					9374	*	INITIALIZE FOR SUBSCRIPT SEARCH	
					9375	*		
	4950	5F	01	ED ED	9376	FZV410 SLC	FZVWS1(, @BR), FZVWS1(FZVL02, @BR)	CLEAR WORK AREA
	4954	6C	01	F3 07	9377	FZV415 MVC	FZVBAS(, @BR), B@ABAS(@VADDR, @XR)	SET BASE ACCUM
	4958	5C	01	EF E1	9378	FZV420 MVC	FZVWS2(, @BR), FZVIN1(B@LBIN, @BR)	SET 2ND SUBSC
					9379	*		
					9380	*	TEST FOR A VADDR MATCH	
					9381	*		
	495C	5E	01	F3 E5	9382	FZV430 ALC	FZVBAS(B@LBIN, @BR), FZVALN(, @BR)	INCR BASE ACCUM
	4960	1D	01	0D57 F3	9383	CLC	I\$PARM, FZVBAS(@CADDR, @BR)	VADDR MATCH ?
	4965	D0	82	16	9384	BL	FZV320(, @BR)	LOW, CHECK NEXT TBL ENTRY
	4968	F2	81	47	9385	JE	FZV520	YES, PROCESS SUBSC
					9386	*		
					9387	*	INCREMENT THE BASE ACCUMULATOR	
					9388	*		
					9389	*		
					9390	*	TEST IF DIMENSION TWO EXECEDED	
					9391	*		
	496B	9D	01	03 EF	9392	FZV440 CLC	B@ACD2(B@LBIN, @XR), FZVWS2(, @BR)	ARE DIMENSIONS
	496F	F2	81	07	9393	JE	FZV460	YES, INCR SUBSC 1
	4972	5E	01	EF E1	9394	FZV450 ALC	FZVWS2(B@LBIN, @BR), FZVIN1(, @BR)	INCR SUBSC 2
	4976	D0	87	5C	9395	B	FZV430(, @BR)	GO TEST FOR VADDR MATCH
					9396	*		
					9397	*	INCREMENT SUBSCRIPT ONE AND TEST FOR END OF ARRAY	
					9398	*		
	4979	5E	01	ED E1	9399	FZV460 ALC	FZVWS1(, @BR), FZVIN1(B@LBIN, @BR)	INCR SUBSC 1
	497D	6D	01	ED 01	9400	CLC	FZVWS1(B@LBIN, @BR), B@ACD1(, @XR)	SUBSC GT DIM 1
	4981	D0	84	16	9401	BH	FZV320(, @BR)	YES, TEST NEXT TBL ENTRY

FZVART - VARIABLE TRACE ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 410

```

4984 D0 87 58          9402      B      FZV420(,@BR)          NO, RECYCLE LOOP

          9404 *****
          9405 *
          9406 * CHARACTER DOPE VECTOR SCAN
          9407 *
          9408 *****
          9409 *
          9410 *   CONVERT VADDR TO CADDR
          9411 *
4987 76 02 E9          9412 FZV470 A      FZVAAC(,@BR),@XR      CONVERT VADDR
          9413 *
          9414 * INITIALIZE FOR SUBSCRIPT SEARCH
          9415 *
498A 6C 01 F3 03          9416 FZV480 MVC      FZVBAS(,@BR),B@CBAS(@VADDR,@XR)  SET BASE ACCUM
498E 5F 03 EF EF          9417          SLC      FZVWS2(,@BR),FZVWS2(FZVL04,@BR)  CLEAR WORK AREAS
4992 7C 01 EF          9418          MVI      FZVWS2(,@BR),FZVL01      SET SUBSCRIPT
          9419 *
          9420 * TEST FOR A VADDR MATCH
          9421 *
4995 5E 01 F3 E7          9422 FZV490 ALC      FZVBAS(B@LBIN,@BR),FZVCLN(,@BR)  INCR BASE ACCUM
4999 1D 01 0D57 F3          9423          CLC      I$PARM,FZVBAS(@VADDR,@BR)  VADDR MATCH ?
499E D0 82 37          9424          BL      FZV350(,@BR)
49A1 F2 81 0E          9425          JE      FZV520
          9426 *
          9427 * INCREMENT THE BASE ACCUMULATOR
          9428 *
          9429 *
          9430 * INCREMENT SUBSCRIPT AND TEST FOR END OF ARRAY
          9431 *
49A4 5E 01 EF E1          9432 FZV510 ALC      FZVWS2(B@LBIN,@BR),FZVIN1(,@BR)  INCR SUBSC      1-5
49A8 6D 01 EF 01          9433          CLC      FZVWS2(,@BR),B@CDMN(B@LBIN,@XR)  SUBSC GT DIM ?      1-5
49AC D0 84 37          9434          BH      FZV350(,@BR)          YES, TEST NEXT TBL ENTRY      1-5
49AF D0 87 95          9435          B      FZV490(,@BR)
          9436 *
          9437 * BRANCH TO SUBSCRIPT PACKING ROUTINE
          9438 *
49B2 34 01 0D5B          9439 FZV520 ST      I$WRK2,@BR          SAVE BASE CADDR
49B6 C0 87 12B1          9440          B      I$CALL          TO PACK SUBSC
49BA 4A00          49BB 9441          DC      AL(@VADDR)(FZVPG5)  VADDR PARAM
          9442 *
          9443 * SHIFT THE SUBSCRIPT IN AN ARITHMETIC ARRAY
          9444 *
49BC 7D 5B EA          9445 FZV525 CLI      FZVTYP(,@BR),B@CVAR      A CHAR ARRAY ?
49BF F2 81 0E          9446          JE      FZV530          YES, SKIP TO SET PARAMS
          4A00 9447          USING FZVPG5,@XR          BASE OR VM PG 5
          9448          L      I$WRK2,@XR          LOAD INDEX
          9449          MVC      FZVP11(FZVL12,@XR),I$FWRK+13  SHIFT THE SUBSCRIPT
49C2 35 02 0D5B          9450          MVC      I$FWRK+12,FZVP11(FZVL12,@XR)  DOWN ONE SPACE
49C6 8C 0B D5 0614          9451 *
49CB 2C 0B 0613 D5          9452 * SET SUBSCRIPT PARAMETERS
          9453 *
49D0 1C 00 0D5B EB          9454 FZV530 MVC      I$WRK2,FZVHLD(1,@BR)      SET TBL CNT
49D5 1C 00 0D5A EA          9455          MVC      I$WRK2-1,FZVTYP(1,@BR)      SET SYM TYPE
          9456 *
          9457 * RETURN

```

FZVART - VARIABLE TRACE ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 411
		9458	*		
49DA	C0 87 12D3	9459	FZV540 B	I\$RTRN	RETURN
		9461	*****		
		9462	*		*
		9463	* VARIABLE TRACE EQUATES, CONSTANTS AND WORK AREAS (3RD VM PAGE)		*
		9464	*		*
		9465	*****		
		9466	*		
		9467	* 3RD INTERFACE PAGE EQUATES		
		9468	*		
		0001 9469	FZVL01 EQU 1		INITIAL SUBSC VALUE
		0002 9470	FZVL02 EQU 2		LNG OF SUBSC TO CLEAR
		9471	*		
		9472	* 3RD INTERFACE PAGE CONSTANTS		
		9473	*		
49DE 0000		49DF 9474	FZVI00 DC	XL2'00'	INTEGER OF 0
49E0 0001		49E1 9475	FZVIN1 DC	XL2'01'	INTEGER OF 1
49E2 0002		49E3 9476	FZVI02 DC	XL2'02'	INTEGER OF 2
49E4 0005		49E5 9477	FZVALN DC	AL2(I@LPFV)	ARITS FIELD LNG
49E6 0013		49E7 9478	FZVCLN DC	AL2(I@LCRV)	CHAR FIELD LNG
49E8 0C08		49E9 9479	FZVAAC DC	AL(@CADDR)(FZVCAD+B@DL16+1)	ARRAY DOPE VECTOR VIRTUAL TO
		9480	*		* CADDR CONVERSION CONSTANT
		9481	*		
		9482	* 3RD INTERFACE PAGE WORK AREAS		
		9483	*		
49EA		49EA 9484	FZVTYP DS	CL1	VAR TYPE
49EB		49EB 9485	FZVHLD DS	CL1	TEL COUNT DISP
49EC		49ED 9486	FZVWS1 DS	CL2	SUBSC WORK 1
49EE		49EF 9487	FZVWS2 DS	CL2	SUBSC WORK 2
49F0		49F3 9488	FZVBAS DS	CL4	BASE WORK AREA

FZVART - VARIABLE TRACE ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 412
					9490	*****				
					9491	*				*
					9492	*****				
					9493	*				*
					9494	* FIFTH VM INTERFACE PAGE - SUBSCRIPT PACKING				*
					9495	*				*
					9496	*****				
					9497	*				*
					9498	*****				
					9499	*				
					9500	* ESTABLISH ADDRESSABILITY FOR INTERFACE 5TH VM PAGE				
					9501	*				
4A00					9502	*FZVPS5 VPAGE 0	SET PAGE ADDRESSABILITY			
					9503	ORG *,256,0	SET STARTING ADDRESS			
		4A00			9504	FZVPG5 EQU *	START OF PROGRAM CODING			
4901					9505	ORG *-255	RESET IAR TO PAGE			
4A00					9506	ORG *,256,0	* BOUNDARY ADDRESS			
		4A00			9507	USING *,@BR	SET PAGE BASE ADDRESS			
4A00					9508	ORG FZVPG5	RESET STARTING ADDRESS			
					9509	*** END OF EXPANSION ***				
				4900	9511	USING FZVPG3,@XR	SET AUX BASE			
					9512	*****				
					9513	*				*
					9514	* SUBSCRIPT PACKING ROUTINE				*
					9515	*				*
					9516	*****				
					9517	*				
					9518	* INITIALIZE THE SUBSCRIPT PACKING BUFFER				
					9519	*				
4A00	7C	4D	CA		9520	FZV600 MVI FZVPAC(,@BR),B@LPAR	INSERT LEFT PAREN			
4A03	35	02	0D5B		9521	L I\$WRK2,@XR	LOAD PG 3 CADDR			
4A07	6C	01	C1 ED		9522	MVC FZVWSC(,@BR),FZVWS1(B@LBIN,@XR)	SHIFT SUBSC 1 TO WORK			
4A0B	D0	87	66		9523	FZV610 B FZV700(,@BR)	CONVERT SUBSC 1			
4A0E	5C	03	CE C5		9524	MVC FZVP04(,@BR),FZVSUB(FZVL04,@BR)	INSERT SUBSC 1			
4A12	7C	6B	CF		9525	MVI FZVP05(,@BR),B@CMMA	INSERT COMMA			
4A15	6C	01	C1 EF		9526	MVC FZVWSC(,@BR),FZVWS2(B@LBIN,@XR)	SHIFT SUBSCRIPT 2			
4A19	D0	87	66		9527	FZV620 B FZV700(,@BR)	CONVERT SUBSC 2			
4A1C	5C	03	D3 C5		9528	MVC FZVP09(,@BR),FZVSUB(FZVL04,@BR)	INSERT SUBSC 2			
4A20	7C	5D	D4		9529	MVI FZVP10(,@BR),B@RPAR	INSERT RIGHT PAREN			
4A23	7C	40	D5		9530	MVI FZVP11(,@BR),B@BLNK	INSERT BLANK			
					9531	*				
					9532	* PACK THE SECOND SUBSCRIPT				
					9533	*				
4A26	C2	02	0613		9534	FZV630 LA I\$FWRK+12,@XR	SET PT			
4A2A	7D	F0	D0		9535	CLI FZVP06(,@BR),B@DEC0	IS BYTE A ZERO ?			
4A2D	F2	01	0B		9536	JNE FZV650	NO, PROCESS SUBSC I			
4A30	9C	04	00 D5		9537	FZV640 MVC 0(,@XR),FZVP11(FZVL05,@BR)	SHIFT THE SUBSCRIPT 1 BYTE			
4A34	6C	04	D4 00		9538	MVC FZVP10(FZVL05,@BR),0(,@XR)	* TO PACK IT			
4A38	D0	87	26		9539	B FZV630(,@BR)	LOOP UNTIL NON ZERO FOUND			
					9540	*				
					9541	* PACK SUBSCRIPT 1				
					9542	*				
4A3B	7D	F0	CB		9543	FZV650 CLI FZVP01(,@BR),B@DEC0	IS BYTE ZERO ?			
4A3E	F2	01	0B		9544	JNE FZV670	NO, COMPLETE PROCESSING			
4A41	9C	09	00 D5		9545	FZV660 MVC 0(,@XR),FZVP11(FZVL10,@BR)	SHIFT THE SUBSCRIPT 1 BYTE			

FZVART - VARIABLE TRACE ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 413

```

4A45 6C 09 D4 00          9546          MVC    FZVP10(FZVL10,@BR),0(@XR)  * TO PACK IT
4A49 D0 87 3B             9547          B      FZV650(@BR)                LOOP UNTIL NON ZERO FOUND
                             9548 *
                             9549 * TEST FOR VECTOR SUBSCRITING
                             9550 *
4A4C 7D 6B CB             9551 FZV670 CLI    FZVP01(@BR),B@CMMMA      A COMMA ?
4A4F F2 01 08             9552          JNE    FZV680                NOS CONTINUE PROC
4A52 9C 09 00 D5          9553 FZV675 MVC    0(@XR),FZVP11(FZVL10,@BR)  SHIFT SUBSCRIPT DOWN 1 BYTE
4A56 6C 09 D4 00          9554          MVC    FZVP10(FZVL10,@BR),0(@XR)  * FOR VECTOR IMAGE
4A5A 9C 0B 01 D5          9555 FZV680 MVC    1(@XR),FZVP11(FZVL12,@BR)  SHIFT SUBSC FOR PARAM
4A5E 34 02 0D5B           9556          ST     I$WRK2,@XR          SET BASE IN PARAM
4A62 C0 87 12D3           9557          B      I$RTRN                RETURN

                             9559 *****
                             9560 *
                             9561 * CONVERT BINARY SUBSCRIPT TO DECIMAL
                             9562 *
                             9563 *****
                             9564 *
                             9565 * SAVE RETURN ADDRESS
                             9566 *
4A66 74 08 9A             9567 FZV700 ST     FZV750+@OP1(@BR),@ARR      SAVE RETURN ADDR
                             9568 *
                             9569 * CONVERT SUBSCRIPT TO DECIMAL
                             9570 *
4A69 7C 80 8C             9571 FZV705 MVI    FZV740+@Q(@BR),@NOP      SET BR SW OFF
4A6C 7C C1 78             9572          MVI    FZV720+@D1(@BR),FZVBS2    SET SUBSC DISP
4A6F 54 70 C9 BF          9573          ZAZ    FZVSAC(FZVL04*2,@BR),FZVDC1(1,@BR)  SET DEC ACCUMS TO 1
4A73 7C 01 77             9574 FZV710 MVI    FZV720+@Q(@BR),@B1      SET BIN MASK FOR 2**0 BIT
4A76 78 00 00             9575 FZV720 TBN    *-*(@BR),*-*          TEST BIN SUBSC MAGNITUDE BIT
4A79 F2 90 04             9576          JF     FZV730                * AND BRANCH IF 0
4A7C 56 03 C5 C9          9577          AZ     FZVSUB(FZVL04,@BR),FZVSAC(FZVL04,@BR)  INCR DEC SUBSC
4A80 56 03 C9 C9          9578 FZV730 AZ     FZVSAC(FZVL04,@BR),FZVSAC(FZVL04,@BR)  DOUBLE DEC ACCUM
4A84 5E 00 77 77          9579          ALC    FZV720+@Q(1,@BR),FZV720+@Q(@BR)  SHIFT MASK LEFT
4A88 D0 28 76             9580          BC     FZV720(@BR),@BNOL+@BNOZ    REPEAT UNTIL OVERFLOW
                             9581 *
                             9582 * TEST IF 2ND LOOP COMPLETED
                             9583 *
4A8B F2 00 09             9584 FZV740 JC     FZV750,*-*          BR IF TWO LOOPS COMPLETED
                             9585 *
                             9586 * INITIALIZE SECOND LOOP
                             9587 *
4A8E 7C 87 8C             9588          MVI    FZV740+@Q(@BR),@UCB      SET BR SW ON
4A91 7C C0 78             9589          MVI    FZV720+@D1(@BR),FZVBS1    SET SUBSC BYTE DISP
4A94 D0 87 73             9590          B      FZV710(@BR)                REPEAT LOOP
                             9591 *
                             9592 * RETURN
                             9593 *
4A97 C0 87 0000           9594 FZV750 B      *-*                RETURN

```

FZVART - VARIABLE TRACE ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 414
					9596	*****		
					9597	*		
					9598	* CONVERT BINARY EXPONENT TO DECIMAL		
					9599	*		
					9600	*****		
					9601	*		
				4B00	9602	USING	FZVPG4,@XR	BASE FOR PG 4 OVERLAY
4A9B	35	02	0D5B		9603	FZV170	L I\$WRK2,@XR	BASE FOR INDEY
4A9F	A4	10	D5 CA		9604	ZAZ	FZVDAC(B@LBIN,@XR),FZVD01(1,@XR)	SET DEC ACCUM TO 1
4AA3	7C	01	A7		9605	MVI	FZV175+@Q(,@BR),@B1	SET BIN MASK FOR 2**0 BIT
4AA6	B8	00	D3		9606	FZV175	TBN FZVBEX(,@XR),*-*	TEST BIN EXP MAGNITUDE BIT
4AA9	F2	90	04		9607	JF	FZV180	* AND BRANCH IF BIT IS 0
4AAC	A6	01	E2 D5		9608	AZ	FZVEXP(B@LBIN,@XR),FZVDAC(B@LBIN,@XR)	INCR DEC EXP
4AB0	5E	00	A7 A7		9609	FZV180	ALC FZV175+@Q(,@BR),FZV175+@Q(1,@BR)	SHIFT BIN MASK LEFT
4AB4	A6	01	D5 D5		9610	AZ	FZVDAC(B@LBIN,@XR),FZVDAC(B@LBIN,@XR)	DOUBLE DEC ACCUM
4AB8	D0	08	A6		9611	BNOZ	FZV175(,@BR)	REPEAT LOOP UNTIL ACCUM > 64
					9612	*		
					9613	* RETURN		
					9614	*		
4ABB	C0	87	12D3		9615	B	I\$RTRN	RETURN
					9617	*****		
					9618	*		
					9619	* VARIABLE TRACE EQUATES, CONSTANTS AND WORK AREAS (5TH VM PAGE)		
					9620	*		
					9621	*****		
					9622	*		
					9623	* 5TH INTERFACE PAGE EQUATES		
					9624	*		
				0004	9625	FZVL04	EQU 4	LNG OF WORK SUBSC TO ZERO
				0005	9626	FZVL05	EQU 5	PACK LNG 2ND SUBSC
				000A	9627	FZVL10	EQU 10	PACK LNG 2 SUBSCRIPTS
				000C	9628	FZVL12	EQU 12	LNG OF SUBSC
					9629	*		
					9630	* 5TH INTERFACE PAGE CONSTANTS		
					9631	*		
4ABF	F1			4ABF	9632	FZVDC1	DC DL1'1'	DECIMAL 1
					9633	*		
					9634	* 5TH INTERFACE PAGE WORK AREAS		
					9635	*		
4AC0				4AC1	9636	FZVWSC	DS CL2	SLBSC WORK
4AC2				4AC5	9637	FZVSUB	DS CL4	SUBSC DECIMAL HOLD
4AC6				4AC9	9638	FZVSAC	DS CL4	DECIMAL SUBSC ACCUMULATOR
				4ACA	9639	FZVPAC	EQU *	CADDR SUBSC WORK AREA
4ACA				4AD5	9640		DS CL12	SUBSC PACK WORK AREA
					9641	*		
					9642	* 5TH INTERFACE PAGE EQUATES REFERENCING PAGE		
					9643	*		
				4ACB	9644	FZVP01	EQU FZVPAC+1	PACK BFR DISP OF 1
				4ACE	9645	FZVP04	EQU FZVPAC+4	PACK BFR DISP OF 4
				4ACF	9646	FZVP05	EQU FZVPAC+5	PACK BFR DISP OF 5
				4AD0	9647	FZVP06	EQU FZVPAC+6	PACK BFR DISP OF 6
				4AD3	9648	FZVP09	EQU FZVPAC+9	PACK BFR DISP OF 9
				4AD4	9649	FZVP10	EQU FZVPAC+10	PACK BFR DISP OF 10
				4AD5	9650	FZVP11	EQU FZVPAC+11	PACK BFR DISP OF 11
				00C0	9651	FZVBS1	EQU FZVWSC-FZVPG5-1	CADDR DISP OF 1ST SUBSC WORK

FZVART - VARIABLE TRACE ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 416
					9654	*****	*****			
					9655	*				*
					9656	*****	*****			
					9657	*				*
					9658	*	4TH VM INTERFACE PAGE - STACK ELEMENT			*
					9659	*				*
					9660	*****	*****			
					9661	*				*
					9662	*****	*****			
					9663	*				
					9664	*	ESTABLISH ADDRESSABILITY FOR INTERFACE 4TH VM PAGE			
					9665	*				
4B00					9666	*FA2G4	VPAGE 0			SET PAGE ADDRESSABILITY
					9667		ORG *,256,0			SET STARTING ADDRESS
				4B00	9668	FZVPG4	EQU *			START OF PROGRAM CODING
4A01					9669		ORG *-255			RESET IAR TO PAGE
4B00					9670		ORG *,256,0			* BOUNDARY ADDRESS
				4B00	9671		USING *,@BR			SET PAGE BASE ADDRESS
4B00					9672		ORG FZVPG4			RESET STARTING ADDRESS
					9673	***	END OF EXPANSION ***			
					9674	*				
					9675	*	SAVE THE PAGE CORE ADDRESS			
					9676	*				
4B00	34	01	0D5B		9677	FZV034	ST I\$WRK2,@BR			SAVE BASE ADDR
					9678	*				
					9679	*	CLEAR DATA BUCKET TO BLANKS			
					9680	*				
4B04	7C	40	EB		9681	FZV035	MVI FZVB21(,@BR),B@BLNK			INITIALIZE THE DATA BUCKET
4B07	5C	15	EA EB		9682		MVC FZVB20(,@BR),FZVB21(FZVL22,@BR) *			TO BLANKS
					9683	*				
					9684	*	GET THE VIRTUAL MEMORY PAGE AT I\$PARM			
					9685	*				
4B0B	4C	01	15 0D57		9686	FZV036	MVC FZV037(@VADDR,@BR),I\$PARM			SET CALL INST WITH VADDR
4B10	C0	87	1330		9687		B I\$LDXR			GET VM PG AND SET XR TO VALUE
4B14				4B15	9688	FZV037	DS CL(@VADDR)			VADDR PARAM
					9689	*				
					9690	*	TEST STATUS BYTE OF VALUE FOR CHARACTER CONSTANT			
					9691	*				
4B16	B8	40	00		9692	FZV040	TBN I@STAT(,@XR),B@DTYP			A CHAR ELEMENT
4B19	F2	90	37		9693		JF FZV060			NO, STACK ARITH ELEMENT
					9695	*****	*****			
					9696	*				*
					9697	*	MOVE CHARATER ELEMENT TO THE PRINT BUFFER			*
					9698	*				*
					9699	*****	*****			
					9700	*				
					9701	*	MOVE BYTE AND TEST FOR OVERFLOW			
					9702	*				
4B1C	7C	D6	26		9703	FZV045	MVI FZV048+@D1(,@BR),FZVD00			SET PT DISP TO ZERO
4B1F	4C	00	D2 0D57		9704		MVC FZVCNT(1,@BR),I\$PARM			SET OVERFLOW COUNTER
4B24	6C	00	00 00		9705	FZV048	MVC *-*(1,@BR),0(,@XR)			SHIFT CHAR TO PRINT BFR
					9706	*				
					9707	*	TEST FOR PAGE BOUNDARY CONDITION			
					9708	*				
4B28	E2	02	01		9709		LA 1(,@XR),@XR			INCR VM PT

FZVART - VARIABLE TRACE ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 417

```

4B2B 5E 00 D2 CB          9710      ALC  FZVCNT(1,@BR),FZVHX1(,@BR)  INCR BFR OVERFLOW COUNT
4B2F F2 20 0E          9711      JNOL FZV055                      NO OVERFLOW, CONTINUE PROC
                               9712 *
                               9713 * GET NEXT CONTIGUOUS VM PAGE
                               9714 *
4B32 7C 01 3E          9715 FZV050 MVI  FZV052(,@BR),@B1          SET VADDR PG INCR
4B35 4E 00 3E 0D56      9716      ALC  FZV052(,@BR),I$PARM-1(1)  INCR VADDR PG
4B3A C0 87 1330          9717      B    I$LDXR                      GET VM PG AND SET XR TO VALUE
4B3E                      4B3E 9718 FZV052 DS    CL1                VADDR PARAM PG
4B3F 00                  4B3F 9719      DC    AL1(@ZERO)          VADDR PARAM DISP
                               9720 *
                               9721 * TEST IF LAST CHARACTER MOVED
                               9722 *
4B40 5E 00 26 CB      9723 FZV055 ALC  FZV048+@D1(1,@BR),FZVHX1(,@BR)  INCR PT DISP
4B44 7D E9 26          9724      CLI  FZV048+@D1(,@BR),FZVD00+I@LCRV  ALL BYTES PROCESSED ?
4B47 D0 01 24          9725      BNE  FZV048(,@BR)                NO, RECYCLE THE LOOP
4B4A 7C 7D D6          9726      MVI  FZVB00(,@BR),B@SQUO          SET PRECEDING QUOTE
4B4D 7C 7D E9          9727      MVI  FZVB19(,@BR),B@SQUO          SET TRAILING QUOTE
4B50 F2 87 73          9728      J    FZV190                      YES, DETERMINE VAR SYMBOL

                               9730 *****
                               9731 *
                               9732 * MOVE ARITHMETIC DATA ELEMENT TO THE PRINT BUFFER
                               9733 *
                               9734 *****
                               9735 *
                               9736 * PREPARE WORK AREA FOR MANTISSA UNPACKING
                               9737 *
4B53 54 70 DE D1      9738 FZV060 ZAZ  FZVB01+I@LUFV-1(I@LUFV,@BR),FZVDC0(1,@BR)  TO DEC 0'S
4B57 B8 10 00          9739 FZV069 TBN  I@STAT(,@XR),B@SIGN          IS SIGN NEGATIVE
4B5A 7C 40 D6          9740      MVI  FZVB00(,@BR),B@BLNK          SET SIGN BLANK
4B5D F2 90 03          9741      JF   FZV070                      NO, SHIFT 1ST NO.
4B60 7C 60 D6          9742      MVI  FZVB00(,@BR),B@MINS          SET SIGN MINUS
4B63 68 03 D7 00      9743 FZV070 MNN  FZVB01(,@BR),I@PMN1(,@XR)  MOVE 1ST NO. IN MANTISSA
4B67 7C 4B D8          9744      MVI  FZVB02(,@BR),B@DPNT          SET DECIMAL POINT
                               9745 *
                               9746 * UNPACK REMAINDER OF THE MANTISSA
                               9747 *
4B6A 7C D9 8F          9748 FZV075 MVI  FZV077+@D1(,@BR),FZVD03      SET DISP FOR 1ST DIGIT MOVE
4B6D 7C DA 93          9749      MVI  FZV078+@D1(,@BR),FZVD04      SET DISP FOR 2ND DIGIT MOVE
4B70 4C 00 D2 0D57      9750      MVC  FZVCNT(1,@BR),I$PARM          SET OVERFLOW COUNTER
                               9751 *
                               9752 * TEST FOR PAGE BOUNDARY CONDITION
                               9753 *
4B75 E2 02 01          9754 FZV079 LA    1(,@XR),@XR                INCR VM PT
4B78 5E 00 D2 CB      9755      ALC  FZVCNT(,@BR),FZVHX1(1,@BR)  INCR COUNT, TEST OVERFLOW
4B7C F2 20 0E          9756      JNOL FZV077                      NO OVERFLOW, CONTINUE PROC
4B7F 7C 01 8B          9757 FZV080 MVI  FZV082(,@BR),@B1          SET PG INCR
4B82 4E 00 8B 0D56      9758      ALC  FZV082(1,@BR),I$PARM-1      INCR TO NEXT VADDR PG
4B87 C0 87 1330          9759      B    I$LDXR                      GET NEXT PG
4B8B                      4B8B 9760 FZV082 DS    CL1                VADDR PARAM PG
4B8C 00                  4B8C 9761      DC    AL1(@ZERO)          VADDR PARAM DISP
                               9762 *
                               9763 * SHIFT MANTISSA ONE BYTE AT A TIME
                               9764 *
4B8D 68 02 00 00      9765 FZV077 MNZ  *-*(,@BR),0(,@XR)          SHIFT MANTISSA DIGIT

```


ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE STATEMENT	VER 15,	MOD 00	31/05/21	PAGE 418
	4B91	68	03	00	00	9766 FZV078 MNN *-*(,@BR),0(,@XR)			SHIFT MANTISSA DIGIT	
						9767 *				
						9768 * INCREMENT POINTERS AND TEST FOR LAST DIGIT MOVED				
						9769 *				
	4B95	5E	00	8F	CC	9770 FZV085 ALC FZV077+@D1(,@BR),FZVHX2(1,@BR) INCR PT DISP				
	4B99	5E	00	93	CC	9771 ALC FZV078+@D1(,@BR),FZVHX2(1,@BR) INCR PT DISP				
	4B9D	7D	E1	8F		9772 CLI FZV077+@D1(,@BR),FZVD03+I@PREC+1 LAST CHAR MOVED				1-3
	4BA0	D0	01	75		9773 BNE FZV079(,@BR) NO, UNPACK NEXT DIGIT				
						9774 *				
						9775 * SAVE EXPONENT				
						9776 *				
	4BA3	6C	00	D3	00	9777 FZV090 MVC FZVBEX(,@BR),0(1,@XR) SAVE EXPONENT				1-3
	4BA7	5F	00	D3	CB	9778 SLC FZVBEX(1,@BR),FZVHX1(,@BR) ADJUST EXP FOR SHIFT				
						9779 *				
						9780 * INITIALIZE OUTPUT FORM OF EXPONENT - TEST FOR EXPONENT SIGN				
						9781 *				
	4BAB	5C	03	E2	D1	9782 MVC FZVEXP(,@BR),FZVXIM(FZVUXL,@BR) SHIFT IMAGE TO WORK				
	4BAF	5F	00	D3	CD	9783 SLC FZVBEX(1,@BR),FZVNZX(,@BR) TEST FOR SIGN				
	4BB3	F2	02	0A		9784 JNL FZV168 BR IF NOT NEG				
						9785 *				
						9786 * NEGATIVE EXPONENT - MODIFY SIGN AND RECOMPUTE BINARY EXPONENT				
						9787 *				
	4BB6	7C	60	E0		9788 FZV160 MVI FZVEXP-2(,@BR),B@MINS MAKE EXP SIGN NEG				
	4BB9	7C	81	D3		9789 MVI FZVBEX(,@BR),B@NXZR+1 RESTORE EXP TO NORMALIZED 0				
	4BBC	6F	00	D3	00	9790 FZV165 SLC FZVBEX(,@BR),0(1,@XR) FIND BINARY MAGNITUDE				1-3
						9791 *				
						9792 * CONVERT BINARY EXPONENT TO ZONED DECIMAL				
						9793 *				
	4BC0	C0	87	12	B1	9794 FZV168 B I\$CALL CALL RTN TO CONVERT EXP TO DEC				
	4BC4	4A9B				9795 DC AL(@VADDR)(FZV170) VADDR PARAM				
						9796 *				
						9797 * RETURN TO 1ST INTERFACE PAGE TO PRINT THE BUFFER				
						9798 *				
	4BC6	C0	87	12	D3	9799 FZV190 B I\$RTRN RETURN TO THE CALLING RTN				
						9801 *****				*
						9802 *				*
						9803 * VARIABLE TRACE EQUATES, CONSTANTS AND WORK AREAS (4TH VM PAGE)				*
						9804 *				*
						9805 *****				*
						9806 *				
						9807 * 4TH INTERFACE PAGE EQUATES				
						9808 *				
					0004	9809 FZVUXL EQU 4				
					0016	9810 FZVL22 EQU 22			LNG OF THE FORMAT BFR	
						9811 *				
						9812 * 4TH INTERFACE PAGE CONSTANTS				
						9813 *				
	4BCA	F1			4BCA	9814 FZVD01 DC DL1'1'			DECIMAL 1	
	4BCB	01			4BCB	9815 FZVHX1 DC XL1'01'			INCR OF 1	
	4BCC	02			4BCC	9816 FZVHX2 DC XL1'02'			INCR OF 2	
	4BCD	80			4BCD	9817 FZVNZX DC AL1(B@NXZR)			ZERO NORMALIZED EYP	
	4BCE	C54EF0F0			4BD1	9818 FZVXIM DC CL(FZVUXL)'E+00'			EXP IMAGE FOR OUTPUT	
					4BD1	9819 FZVDC0 EQU *-1			USES IMAGE LAST BYTE FOR 0	
						9820 *				
						9821 * 4TH INTERFACE PAGE WORK AREAS				

FZVART - VARIABLE TRACE ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 419
					9822	*				
4BD2				4BD2	9823	FZVCNT DS	CL1			OVERFLOW COUNT AREA
4BD3				4BD3	9824	FZVBEX DS	CL1			EXPONENT WORK AREA
4BD4				4BD5	9825	FZVDAC DS	CL2			BIN TO DECIMAL ACCUM
				4BD6	9826	FZVFBR EQU	*			CADDR FORMAT BFR
4BD6				4BEB	9827	DS	CL22			FORMAT BFR
					9828	*				
					9829	* 4TH INTERFACE PAGE	EQUATES REFERENCING PAGE			
					9830	*				
				4BD6	9831	FZVB00 EQU	FZVFBR			1ST BYTE OF FORMAT BFR
				4BD7	9832	FZVB01 EQU	FZVFBR+1			2ND BYTE OF FORMAT BFR
				4BD8	9833	FZVB02 EQU	FZVFBR+2			3RD BYTE OF FORMAT BFR
				4BD9	9834	FZVB03 EQU	FZVFBR+3			4TH BYTE OF FORMAT BFR
				4BDA	9835	FZVB04 EQU	FZVFBR+4			5TH BYTE OF FORMAT BFR
				4BE9	9836	FZVB19 EQU	FZVFBR+19			20TH BYTE OF FORMAT BFR
				4BEA	9837	FZVB20 EQU	FZVFBR+20			21ST BYTE OF BFR
				4BEB	9838	FZVB21 EQU	FZVFBR+21			22ND BYTE OF BFR
				00D6	9839	FZVD00 EQU	FZVB00-FZVPG4			BFR DISP OF 0
				00D9	9840	FZVD03 EQU	FZVB03-FZVPG4			BFR DISP OF 2ND MANTISSA DIGIT
				00DA	9841	FZVD04 EQU	FZVB04-FZVPG4			BFR DISP OF 3RD MANTISSA DIGIT
					9843	*****				
					9844	*				*
					9845	*****				
					9846	*				*
					9847	* FZVART EQUATES REFERENCING PROGRAM				*
					9848	*				*
					9849	*****				
					9850	*				*
					9851	*****				
					9852	*				
				4BE2	9853	FZVEXP EQU	FZVFBR+I@PREC+5			LAST BYTE OF EXP IMAGE
				070C	9854	FZVLVT EQU	FZVCAD+B@DL06+1			CADDR 1ST ENTRY LVT
				098A	9855	FZVCVT EQU	FZVCAD+B@DL10+1			CADDR 1ST ENTRY CVT
				09C4	9856	FZVNAT EQU	FZVCAD+B@DL11+1			CADDR 1ST ENTRY NAT
				09FE	9857	FZVCAT EQU	FZVCAD+B@DL12+1			CADDR 1ST ENTRY CAT
					9858	*				
					9859	*				
							END MODULE FZVART *****			

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 420
		9861		*****			
		9862	*	5703-XM1 COPYRIGHT IBM CORP. 1970			*
		9863	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083			*
		9864	*				*
		9865		*****			*
		9866	*	*STATUS			*
		9867	*	VERSION 1 MODIFICATION 0			*
		9868	*				*
		9869	*	*FUNCTION			*
		9870	*	* FZZVMP EXECUTION CAUSES ALL MODIFIED CORE VIRTUAL MEMORY PAGES			*
		9871	*	TO BE WRITTEN BACK TO DISK (PUSHED) OR ALL UNLOCKED CORE			*
		9872	*	VIRTUAL MEMORY PAGES TO BE LOADED INTO CORE (PULLED).			*
		9873	*	* OPERATION OF THIS ROUTINE DEPENDS UPON THE ENTRY POINT SELECTED			*
		9874	*	FOR EXECUTION -			*
		9875	*	* ENTRY POINT FZZVPS - ALL CORE VIRTUAL MEMORY PAGES REFER-			*
		9876	*	ENCED WITH A 'MODIFY' INDICATOR IN THE PAGING MODULE 'LOCK			*
		9877	*	AND READ ONLY' INDICATOR TABLE ARE WRITTEN INTO DISK			*
		9878	*	VIRTUAL MEMORY. THE 'MODIFY' INDICATOR IS UNSET IN THE			*
		9879	*	INDICATOR TABLE. THIS 'PUSH' IS AUTOMATICALLY ADJUSTED			*
		9880	*	TO PROCESS AN EXPANDED TABLE AND CORE PAGE REGION FOR			*
		9881	*	EXTENDED CORE CONFIGURATIONS.			*
		9882	*	* ENTRY POINT FZZVPL - ALL CORE VIRTUAL MEMORY PAGES REFER-			*
		9883	*	ENCED WITH A 'LOCK' INDICATOR IN THE PAGING MODULE 'LOCK			*
		9884	*	AND READ ONLY' INDICATOR TABLE ARE REPLACED WITH THE			*
		9885	*	CORRESPONDING PAGE FROM DISK VIRTUAL MEMORY. THIS 'PULL'			*
		9886	*	IS AUTOMATICALLY ADJUSTED TO PROCESS AN EXPANDED TABLE AND			*
		9887	*	CORE PAGE REGION FOR EXTENDED CORE CONFIGURATIONS.			*
		9888	*				*
		9889	*	*ENTRY POINTS			*
		9890	*	* ENTRY FZZVPS - FOR PERFORMING THE 'PUSH' OPERATION.			*
		9891	*	CALLING SEQUENCE IS			*
		9892	*	B IPGCAL			*
		9893	*	DC AL2(V\$VMPS)			*
		9894	*	WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL			*
		9895	*	ADDRESS OF ENTRY POINT FZZVPS.			*
		9896	*	* ENTRY FZZVPL - FOR PERFORMING THE 'PULL' OPERATION.			*
		9897	*	CALLING SEQUENCE IS			*
		9898	*	B IPGCAL			*
		9899	*	DC AL2(V\$VMPL)			*
		9900	*	WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL			*
		9901	*	ADDRESS OF ENTRY POINT FXXVPL.			*
		9902	*	* IN EACH CASE, EXECUTION IS SUBJECT TO THE INPUT CONDITIONS			*
		9903	*	DESCRIBED BELOW.			*
		9904	*				*
		9905	*	*INPUT			*
		9906	*	* \$EXFTR - 1 BYTE, FOR THE SYSTEM CORE EXTENSION FACTOR. THIS			*
		9907	*	CONTAINS THE NUMBER OF CORE PAGES (256-BYTE REGIONS) AVAILABLE			*
		9908	*	FOR GENERAL USE BEYOND THE 8K MINIMUM CONFIGURATION.			*
		9909	*	* PAGE INDICATOR TABLE - 10 BYTES (MINIMUM), FOR THE PAGING			*
		9910	*	MODULE 'LOCK AND READ ONLY' CORE VIRTUAL MEMORY INDICATORS.			*
		9911	*	THIS TABLE, WHICH IS EXPANDED TO (10+\$EXFTE-1) BYTES WHEN			*
		9912	*	\$EXFTR IS NON-ZERO, CONTAINS A SINGLE BYTE ENTRY CORRESPONDING			*
		9913	*	TO EACH CORE PAGE. BIT 6 (MASK X'02') IN EACH ENTRY INDICATES			*
		9914	*	THE MODIFICATION STATUS OF A CORE PAGE (1 = MODIFIED).			*
		9915	*	BIT 7 (MASK X'01') IN EACH ENTRY INDICATES THE LOCKED STATUS			*
		9916	*	OF A CORE PAGE (1 = LOCKED).			*

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 421
		9917	*	* PAGE REFERENCE TABLE - 256 BYTES, FOR THE PAGING MODULE CORE	*		
		9918	*	VIRTUAL MEMORY MAP. EACH BYTE IN THIS TABLE IS ASSOCIATED WITH	*		
		9919	*	A SPECIFIC VIRTUAL MEMORY PAGE, AND CONTAINS EITHER A VALUE OF	*		
		9920	*	ZERO OR THE NUMBER OF THE CORE PAGE CURRENTLY FILLED WITH THAT	*		
		9921	*	VIRTUAL MEMORY PAGE.	*		
		9922	*		*		
		9923	*	*OUTPUT	*		
		9924	*	* DISK VIRTUAL MEMORY - FOR ENTRY POINT FZZVPS ONLY, EACH CORE	*		
		9925	*	VIRTUAL MEMORY PAGE, FOR WHICH A 'PAGE MODIFY' BIT IS SET IS	*		
		9926	*	WRITTEN BACK TO DISK VIRTUAL MEMORY SO THAT DISK V.M. PAGES	*		
		9927	*	REFLECT THE CURRENT PROCESSING STATUS.	*		
		9928	*	* CORE VIRTUAL MEMORY - FOR ENTRY POINT FZZVPL ONLY, EACH CORE	*		
		9929	*	VIRTUAL MEMORY PAGE, FOR WHICH A 'PAGE LOCKED' BIT IS NOT SET,	*		
		9930	*	IS REPLACED WITH THE CORRESPONDING DISK VIRTUAL MEMORY PAGE	*		
		9931	*	SO THAT CORE V.M. PAGES REFLECT CURRENT DISK STATUS.	*		
		9932	*		*		
		9933	*	*EXTERNAL REFERENCES	*		
		9934	*	* \$DISKN - ENTRY POINT FOR THE SYSTEM PHYSICAL DISK IOCS.	*		
		9935	*	* \$WAITF - CORE ADDRESS OF 'WAIT' FUNCTION DISK PARAMETER LIST.	*		
		9936	*	* I\$RTRN - ENTRY POINT FOR PAGING MODULE V.M. RETURN CONTROL RTN.	*		
		9937	*	* \$EXFTR - 1 BYTE, FOR THE SYSTEM CORE EXTENSION FACTOR.	*		
		9938	*	* I\$CSXA - CORE ADDRESS OF 1ST BYTE IN CORE EXTENSION PAST 8K.	*		
		9939	*	* ISPLAT - CORE ADDRESS OF PAGE INDICATOR TABLE BASE ENTRY.	*		
		9940	*	* I\$PSTB - CORE ADDRESS OF PAGE REFERENCE TABLE BASE ENTRY.	*		
		9941	*		*		
		9942	*	*EXITS, NORMAL	*		
		9943	*	CONTROL IS ALWAYS PASSED TO THE PAGING ROUTINE AT ENTRY POINT.	*		
		9944	*	I\$RTRN (IPGRTN) FOR A RETURN TO THE CALLING PROGRAM.	*		
		9945	*		*		
		9946	*	*EXITS, ERROR	*		
		9947	*	N/A	*		
		9948	*		*		
		9949	*	*TABLES/WORK AREAS	*		
		9950	*	* DISK ADDRESS CONVERSION WORK AREAS - TWO 2-BYTE AREAS USED TO	*		
		9951	*	CONVERT LOGICAL DISK ADDRESSES TO PHYSICAL (A LA DL4ICS).	*		
		9952	*	* DISK PARAMETER LIST - 6 BYTES, FOR VIRTUAL PAGE READ/WRITE	*		
		9953	*	OPERATIONS.	*		
		9954	*		*		
		9955	*	*ATTRIBUTES	*		
		9956	*	* REUSABLE	*		
		9957	*	* NATURALLY RELOCATABLE	*		
		9958	*		*		
		9959	*	*CHARACTER CODE DEONENCY	*		
		9960	*	THE OPERATION OR THIS MODULE DOES NOT DEPEND UPON A PARTICULAR	*		
		9961	*	INTERNAL REPRESENTATION OF THE EXTERNAL CHARACTER SET.	*		
		9962	*		*		
		9963	*	*NOTES	*		
		9964	*	ERROR PROCEDURES	*		
		9965	*	NONE	*		
		9966	*		*		
		9967	*	REGISTER USAGE	*		
		9968	*	* REGISTER @BR IS TO CONTAIN THE CORE PAGE BASE ADDRESS	*		
		9969	*	ESTABLISHED THROUGH PAGING MODULE CONTROL FOR THE PAGE WHICH	*		
		9970	*	INCLUDES FZZVMP, AND IS RESTORED THROUGH THE PAGING MODULE.	*		
		9971	*	* REGISTER @XR IS NOT SAVED. IT IS USED IN FZZVMP FPR GENERAL	*		
		9972	*	PURPOSE INDEXING OPERATIONS.	*		

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 422	
		9973	*						*
		9974	*	SAVED/RESTORED AREAS					*
		9975	*	NONE					*
		9976	*						*
		9977	*	MODIFICATION CONSIDERATIONS					*
		9978	*	NONE					*
		9979	*						*
		9980	*	REQUIRED MODULES					*
		9981	*	* @SYSEQ - COMMON SYSTEM EQUATES					*
		9982	*	* @FXDEQ - SYSTEM NUCLEUS ADDRESSES AND INDICATOR EQUATES.					*
		9983	*	* \$B@EQU - COMPILER PARAMETER AND CONSTANT EQUATES.					*
		9984	*	* \$I\$EQU - INTERPRETER FIXED LOCATION ADDRESS EQUATES.					*
		9985	*	* \$I@SEQ - INTERPRETER PARAMETER EQUATES (FOR STD PREC. ONLY)					*
		9986	*	* \$I@LEQ - INTERPRETER DARANETER EQUATES (FOR LNG PREC. ONLY)					*
		9987	*						*
		9988	*	OTHER					*
		9989	*	NONE					*
		9990	*	*****					*

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 423
					9992	*****	*****			
					9993	*	START OF VIRTUAL MEMORY PUSH/PULL EXECUTION ROUTINE			*
					9994	*****	*****			
					9995	*				
					9996	*	ESTABLISH VIRTUAL PAGE ADDRESSABILTY			
					9997	*				
					9998	*FZPGB	VPAGE 0			SET PAGE ADDRESSABILITY
4C00					9999		ORG *,256,0			SET STARTING ADDRESS
				4C00		FZZPGB	EQU *			START OF PROGRAM CODING
4B01					1		ORG *-255			RESET IAR TO PAGE
4C00					2		ORG *,256,0			* BOUNDARY ADDRESS
				4C00	3		USING *,@BR			SET PAGE BASE ADDRESS
4C00					4		ORG FZZPGB			RESET STARTING ADDRESS
					5	***	END OF EXPANSION ***			
					6	*				
					7	*****	*****			

```

          9 *
         10 * ENTRY POINT FZZVPS - SET VIRTUAL PAGE PUSH FUNCTION.
         11 *
4C00 7C 02 BD      4C00 12 FZZVPS EQU      *          VM PUSH ROUTINE ENTRY POINT
4C03 F2 87 03      13          MVI      FZZDPL+@DCTRL(,@BR),@DPUT SET DISK OUTPUT PARAMETER
                                14          J      FZZ005          GO PERFORM THE PUCH OPERATION
         15 *
         16 * ENTRY POINT FZZVPL - SET VIRTUAL PAGE PULL FUNCTION.
         17 *
4C06 7C 01 BD      4C06 18 FZZVPL EQU      *          VM PULLH ROUTINE ENTRY POINT
                                19          MVI      FZZDPL+@DCTRL(,@BR),@DGET SET DISK OUTPUT PARAMETER
         21 *
         22 * INITIALIZE PUSH/PULL ROUTINE FOR 8K SYSTEM ENVIRONMENT.
         23 *
4C09 7C 0A 2B      24 FZZ005 MVI      FZZ020+@D1(,@BR),I@NCPG  SET MAX CORE PAGE COUNT FOR 8K
4C0C 5C 01 BA B5    25          MVC      FZZHCA(,@BR),FZZSXA(@CADDR,@BR) SET HIGH CORE ADDR FOR 8K
         26 *
         27 * TEST FOR CORE AVAILABILITY BEYOND 8K - RE-INITIALIZE IF EXTENDED CORE
         28 *
4C10 3D 00 043B    29          CLI      $EXFTR,@ZERO          TEST FOR NULL CORE EXTENSION
4C14 F2 81 0E      30          JE      FZZ010          BRANCH IF ONLY 8K SYSTEM CONFIG.
         31 *
4C17 4E 00 2B 043B 32          ALC      FZZ020+@D1(,@BR),$EXFTR(1) ADD 1 LESS THAN EXTRA NO. OF
4C1C 5F 00 2B B3    33          SLC      FZZ020+@D1(,@BR),FZZBN1(1,@BR) * PAGES TO CORE PAGE COUNT
4C20 4E 00 B9 043B 34          ALC      FZZHCA-1(,@BR),$EXFTR(1) SET EXTENDED SYSTEM HIGH CADDR
         35 *
         36 *****
```

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER	15,	MOD	00	31/05/21	PAGE	425
					38	*								
					39	*	ACCESS A CORE PAGE ENTRY IN THE PAGING MODULE 'LOCK AND READ ONLY'							
					40	*	INDICATOR TABLE							
					41	*								
4C25	C2	02	15E1		42	FZZ010	LA I\$PLRT-1,@XR						LOAD CORE PAGE INDR TABLE BASE	
4C29	E2	02	00		43	FZZ020	LA *-*(,@XR),@XR						INCR POINTER TO CORE PAGE ENTRY	
					44	*								
					45	*	TEST FOR PUSH OR PULL FUNCTION EXECUTION							
					46	*								
4C2C	7D	01	BD		47		CLI FZZDPL+@DCTRL(@BR),@DGET IF DISK PARAM SET FOR INPUT							
4C2F	F2	81	0C		48		JE FZZ025 * BRANCH TO EXECUTE PAGE PULL							
					49	*								
					50	*	PUSH FUNCTION - TEST THE CURRENTLY REFERENCED CORE PAGE INDICATOR							
					51	*	FOR MODIFY BIT SET ON, AND PUSH THE CORE PAGE ONLY IF MODIFIED							
					52	*								
4C32	B8	02	00		53		TBN FZZLRT(@XR),FZZMDY IF CORE PAGE IS NOT MODIFIED							
4C35	F2	90	6A		54		JF FZZ090 * GO DECREMENT CORE PAGE COUNT							
4C38	BB	02	00		55		SBF FZZLRT(@XR),FZZMDY PAGE MODIFIED - SET INDICATOR							
4C3B	F2	87	06		56		J FZZ030 * OFF AND GO PERFORM PAGE PUSH							
					57	*								
					58	*	PULL FUNCTION - TEST THE CURRENTLY REFERENCED CORE PAGE INDICATOR							
					59	*	FOR LOCK BIT SET ON, AND PULL THE CORE PAGE ONLY IF NOT LOCKED							
					60	*								
4C3E	B8	01	00		61	FZZ025	TBN FZZLRT(@XR),FZZLOK IF THE CORE PAGE IS LOCKED							
4C41	F2	10	5E		62		JT FZZ090 * GO DECREMENT CORE PAGE COUNT							
					63	*								
					64	*	PUSH OR PULL CURRENTLY REFERENCED CORE PAGE - SEARCH THE PAGE							
					65	*	REFERENCE TABLE TO DETERMINE THE ACTUAL VIRTUAL PAGE NUMBER							
					66	*								
4C44	7C	FF	51		67	FZZ030	MVI FZZ040+@D1(@BR),FZZBM1 SET VIRTUAL PAGE NO. = MINUS 1							
4C47	C2	02	14CA		68		LA I\$PGTB,@XR LOAD PAGE REFERENCE TABLE BASE							
4C4B	5E	00	51 B3		69	FZZ035	ALC FZZ040+@D1(@BR),FZZBN1(1,@BR) INCREMENT VIRTUAL PAGE NO.							
4C4F	9D	00	00 2B		70	FZZ040	CLC *-*(,@XR),FZZ020+@D1(1,@BR) COMPARE REF TBL ENTRY W/ CORE							
4C53	D0	01	4B		71		BNE FZZ035(@BR) * PAGE NO. AND LOOP IF NO MATCH							
					72	*								
					73	*	*****							

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER	15,	MOD	00	31/05/21	PAGE	426
					75		*****							
					76		* CONVERT VIRTUAL PAGE NUMBER TO A PHYSICAL DISK ADDRESS							*
					77		*****							
					78		*							
					79		* ESTABLISH LOGICAL DISK ADDRESS IN THE DISK PARAMETER LIST							
					80		*							
	4C56	7C	07	BE	81	MVI	FZZDPL+@DCYL(,@BR),B@DVCY SET VIRTUAL MEMORY BASE CYL NO.							
	4C59	5C	00	BF 51	82	MVC	FZZDPL+@DSAD(,@BR),FZZ040+@D1(1,@BR) SET RELATIVE SECTOR							
					83		* ADDRESS EQUAL VIRT PAGE NO.							
					84		*							
					85		* DETERMINE THE TRACK SECTOR COUNT (= LOGICAL SECTOR ADDRESS, MOD 24).							
					86		* INCREMENT THE CYLINDER/DISK/TRACK INDICATOR DURING EACH PASS THROUGH							
					87		* THE SUBTRACTION (DIVISION) LOOP.							
					88		*							
	4C5D	5C	01	BC B8	89	MVC	FZZCNT(,@BR),FZZCDT(@DADDR,@BR) INITLZ CYL/DISK/TRACK CNT							
	4C61	5F	01	BC B8	90	FZZ050 SLC	FZZCNT(,@BR),FZZCDT(@DADDR,@BR) INCR CYL/DISK/TRACK COUNT							
	4C65	5F	00	BF B6	91	SLC	FZZDPL+@DSAD(,@BR),FZZNST(1,@BR) DECR LOGICAL SECTOR ADDR							
	4C69	D0	02	61	92	BNM	FZZ050(,@BR) REPEAT UNTIL SADDR IS NEGATIVE							
	4C6C	5E	00	BF B6	93	ALC	FZZDPL+@DSAD(,@BR),FZZNST(1,@BR) RESTORE POSITIVE SADDR							
					94		*							
					95		* THE DISK PARAMETER LIST NOW CONTAINS THE PHYSICAL SECTOR COUNT -							
					96		* THE CYLINDER CORRECTION COUNT CONTAINS THE INCREMENT WITH WHICH TO							
					97		* ADJUST THE LOGICAL CYLINDER ADDRESS, AND BITS 0 AND 1 OF THE DISK/							
					98		* TRACK INDICATOR BYTE ARE SET RESPECTIVELY TO THE CORRECT PHYSICAL							
					99		* DISK AND TRACK STATUS CONDITIONS.							
					100		*							
					101		* CONVERT THE LOGICAL (BASE) CYLINDER ADDRESS TO A PHYSICAL ADDRESS							
					102		*							
	4C70	5E	00	BE BB	103	ALC	FZZDPL+@DCYL(,@BR),FZZCNT-1(1,@BR) ADJUST THE CYL ADDR							
					104		*							
					105		* SHIFT SECTOR COUNT 2 BITS LEFT (MULTIPLY BY 4)							
					106		*							
	4C74	5E	00	BF BF	107	ALC	FZZDPL+@DSAD(,@BR),FZZDPL+@DSAD(1,@BR) SHIFT COUNT (2X)							
	4C78	5E	00	BF BF	108	ALC	FZZDPL+@DSAD(,@BR),FZZDPL+@DSAD(1,@BR) SHIFT COUNT (4X)							
					109		*							
					110		* SET THE SECTOR ADDRESS DISK (REMOVABLE OR FIXED) INDICATOR BIT							
					111		*							
	4C7C	78	80	BC	112	TBN	FZZCNT(,@BR),FZZIDM TEST INDICATOR DISK BIT							
	4C7F	F2	90	03	113	JF	FZZ060 * AND BRANCH IF NOT EQUAL 1							
	4C82	7A	01	BF	114	SBN	FZZDPL+@DSAD(,@BR),FZZSDM SET SADDR FOR FIXED DISK							
					115		*							
					116		* SET THE SECTOR ADDRESS TRACK (UPPER OR LOWER) INDICATOR BIT							
					117		*							
	4C85	78	40	BC	118	FZZ060 TBN	FZZCNT(,@BR),FZZITM TEST INDICATOR TRACK BIT							
	4C88	F2	90	03	119	JF	FZZ070 * AND BRANCH IF NOT EQUAL 1							
	4C8B	7A	80	BF	120	SBN	FZZDPL+@DSAD(,@BR),FZZSTM SET SADDR FOR LOWER TRACK							
					121		*							
					122		*****							

FZZVMP - S/3 BASIC INTERPRETER V.M. PUSH/PULL EXEC RTN									
ERR	LOC	OBJECT CODE		ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 427	
					124	*****			
					125	* PERFORM READ/WRITE BETWEEN CORE PAGE AND DISK VIRTUAL MEMORY			*
					126	*****			
					127	*			
					128	* CALCULATE THE AFFECTED CORE PAGE ACTUAL CORE ADDRESS			
					129	*			
4C8E	5C	01	C2	BA	130	FZZ070	MVC	FZZDPL+@DBFR2(,@BR),FZZHCA(@CADDR,@BR)	SET HIGH CORE ADDR
4C92	5F	00	C1	2B	131		SLC	FZZDPL+@DBFR1(,@BR),FZZ020+@D1(1,@BR)	SUB CORE PAGE NO.
					132	*			
					133	* PERFORM THE CORE PAGE - VIRTUAL MEMORY DISK OPERATION			
					134	*			
4C96	D2	02	BD		135		LA	FZZDPL(,@BR),@XR	LOAD PARAMETER LIST CORE ADDR
4C99	74	02	A1		136		ST	FZZ080(,@BR),@XR	STORE DPL CORE ADOR FOR CALL
4C9C	C0	87	0025		137		B	\$DISKN	LINK TO READ/WRITE THE CORE PAGE
4CA0				4CA1	138	FZZ080	DS	CL(@CADDR)	PARAMETER LIST CORE ADDRESS
					140	*			
					141	* SET NEXT CORE PAGE PROCESSING - EXIT IF NO MORE CORE PAGES			
					142	*			
4CA2	5F	00	2B	B3	143	FZZ090	SLC	FZZ020+@D1(,@BR),FZZBN1(1,@BR)	DECR THE CORE PAGE NUMBER
4CA6	D0	84	25		144		BP	FZZ010(,@BR)	GO PROCESS NEW PAGE UNLESS ZERO
					145	*			
					146	* EXIT - RETURN TO THE CALLING ROUTINE			
					147	*			
4CA9	C0	87	0025		148		B	\$DISKN	LINK TO WAIT I/O COMPLETED
4CAD	057F			4CAE	149		DC	AL(@CADDR)(\$WAITF)	'WAIT' FUNCTION PARAM CADDR
					150	*			
4CAF	C0	87	12D3		151		B	I\$RTRN	RETURN TO CALLING ROUTINE
					152	*			
					153	*****			

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15,	MOD 00	31/05/21	PAGE 427
					124		*****				
					125	*	PERFORM READ/WRITE BETWEEN CORE PAGE AND DISK VIRTUAL MEMORY				*
					126		*****				
					127	*					
					128	*	CALCULATE THE AFFECTED CORE PAGE ACTUAL CORE ADDRESS				
					129	*					
4C8E	5C	01	C2	BA	130	FZZ070	MVC FZZDPL+@DBFR2(,@BR),FZZHCA(@CADDR,@BR)	SET	HIGH	CORE	ADDR
4C92	5F	00	C1	2B	131		SLC FZZDPL+@DBFR1(,@BR),FZZ020+@D1(1,@BR)	SUB	CORE	PAGE	NO.
					132	*					
					133	*	PERFORM THE CORE PAGE - VIRTUAL MEMORY DISK OPERATION				
					134	*					
4C96	D2	02	BD		135		LA FZZDPL(,@BR),@XR	LOAD	PARAMETER	LIST	CORE ADDR
4C99	74	02	A1		136		ST FZZ080(,@BR),@XR	STORE	DPL	CORE	ADOR FOR CALL
4C9C	C0	87	0025		137		B \$DISKN	LINK	TO	READ/WRITE	THE CORE PAGE
4CA0				4CA1	138	FZZ080	DS CL(@CADDR)	PARAMETER	LIST	CORE	ADDRESS
					140	*					
					141	*	SET NEXT CORE PAGE PROCESSING - EXIT IF NO MORE CORE PAGES				
					142	*					
4CA2	5F	00	2B	B3	143	FZZ090	SLC FZZ020+@D1(,@BR),FZZBN1(1,@BR)	DECR	THE	CORE	PAGE NUMBER
4CA6	D0	84	25		144		BP FZZ010(,@BR)	GO	PROCESS	NEW	PAGE UNLESS ZERO
					145	*					
					146	*	EXIT - RETURN TO THE CALLING ROUTINE				
					147	*					
4CA9	C0	87	0025		148		B \$DISKN	LINK	TO	WAIT	I/O COMPLETED
4CAD	057F			4CAE	149		DC AL(@CADDR)(\$WAITF)	'WAIT'	FUNCTION	PARAM	CADDR
					150	*					
4CAF	C0	87	12D3		151		B I\$RTRN	RETURN	TO	CALLING	ROUTINE
					152	*					
					153		*****				

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 428

```

155 *****
156 * VIRTUAL MEMORY PUSH/PULL ROUTINE CONSTANTS *
157 *****
158 *
4CB3 01      4CB3 159 FZZBN1 DC      IL1'1'          BINARY INTEGER +1
160 *
4CB4 2000    4CB5 161 FZZSXA DC      AL(@CADDR)(I$CSXA)  CORE EXTENSION STARTING ADDRESS
162 *
4CB6 18      4CB6 163 FZZNST DC      AL1(@DTRSZ)          NO. OF SECTORS PER DISK TRACK
4CB7 FFC0    4CB8 164 FZZCDT DC      XL(@DADDR)'FFC0'      CYLINDER/DISK/TRACK DECREMENT
166 *****
167 * VIRTUAL MEMORY PUSH/PULL ROUTINE WORK AREAS *
168 *****
169 *
4CB9      4CBA 170 FZZHCA DS      CL(@CADDR)          HIGHEST AVAILABLE CADDR + 1
171 *
4CBB      4CBC 172 FZZCNT DS      CL(@DADDR)          CYLINDER/DISK/TRACK COUNTER
173 *
174 *FZZDPL DPL      CNT-1          VM I/O DISK PARAMETER LIST
4CBD      4CBD 175 FZZDPL EQU      *          DISK PARAMETER LIST
4CBD 00      4CBD 176          DC      AL1(*-*)          REQUESTED FUNCTION
4CBE 00      4CBE 177          DC      AL1(*-*)          CYLINDER ADDRESS
4CBF 00      4CBF 178          DC      AL1(*-*)          HEAD/SECTOR/DRIVE/DISK SPEC
4CC0 01      4CC0 179          DC      AL1(1)          SECTOR COUNT
4CC1 0000    4CC2 180          DC      AL2(*-*)          BUFFER ADDRESS
181 *** END OF EXPANSION ***

183 *****
184 * VIRTUAL MEMORY PUSH/PULL ROUTINE EQUATES REFERENCING CONSTANTS *
185 *****
186 *
00FF      187 FZZBM1 EQU      X'FF'          BINARY INTEGER -1
188 *
0000      189 FZZLRT EQU      0          DISP FOR PAGE INDR TABLE ENTRY
0001      190 FZZLOK EQU      X'01'        CORE PAGE INDICATOR LOCK MASK
0002      191 FZZMDY EQU      X'02'        CORE PAGE INDICATOR MODIFY MASK
192 *
0080      193 FZZIDM EQU      X'80'        INDICATOR DISK BIT MASK
0040      194 FZZITM EQU      X'40'        INDICATOR TRACE BIT MASK
0001      195 FZZSDM EQU      X'01'        SECTOR ADDR DISK BIT MASK
0080      196 FZZSTM EQU      X'80'        SECTOR ADDR TRACK BIT MASK
197 *
198 *** END OF VIRTUAL MEMORY PUSH/PULL ROUTINE CODING ***
199 *

```

DLFPRT - LINE PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00 31/05/21 PAGE 429
		201		*****	
		202	*	5703-XM1 COPYRIGHT IBM CORP. 1970	*
		203	*	REFER TO INSTRUCTIONS ON COPY RIGHT NOTICE, 120-2083	*
		204	*		*
		205		*****	
		206	*	*STATUS	*
		207	*	VERSION 1 MODIFICATION 0	*
		208	*		*
		209	*	*FUNCTION	*
		210	*	* DLFprt EXECUTION CAUSES DATA OUTPUT AND/OR CARRIER POSITIONING	*
		211	*	ON THE SYSTEM PRINT DEVICE UNDER CONTROL OF CODES RECEIVED FROM	*
		212	*	THE CALLING ROUTINE, PRINTING IS DONE BIDIRECTIONALLY	*
		213	*	* THE FOLLOWING ACTIONS ARE PERFORMED DEPENDING ON THE CODE AND	*
		214	*	CARRIER POSITION:	*
		215	*	* INDEX, PRINT AND INDEX & TAB, PRINT AND INDEX	*
		216	*	* INPUT CODES	*
		217	*	* PRINT X'40' WILL CAUSE THE DATA TO BE PRINTED TO	*
		218	*	BE MOVED INTO THE LINE PRINTER BUFFER	*
		219	*	* PRINT & RETRN X'C0' WILL CAUSE THE DATA TO BE MOVED INTO	*
		220	*	THE BUFFER, AND THE CONTENTS PRINTED	*
		221	*	* CARRAGE RETRN X'80' WILL CAUSE AN INDEX IF THE BUFFER IS	*
		222	*	EMPTY OR THE BUFFER PRINTED IF NOT	*
		223	*		*
		224	*	*ENTRY POINTS	*
		225	*	THIS ROUTINE HAS A SINGLE CALLING ENTRY POINT - DLFprt - WHOSE	*
		226	*	FUNCTION IS DEFINED ABOVE. THE CALLING SEQUENCE IS:	*
		227	*	B I\$LDXR	*
		228	*	DC AL2(V\$LPRT)	*
		229	*	WHERE THE ADDRESS CONSTANT PARAMETER DEFINES THE VIRTUAL ADDRESS	*
		230	*	OF ENTRY POINT DLFprt.	*
		231	*		*
		232	*	*INPUT	*
		233	*	* \$PRPOS - 1 BYTE CARRIER POSITION RELATIVE TO HARDWARE LEFTMGN	*
		234	*	* \$LMRGN - 1 BYTE SOFTWARE LEFT MARGIN INDICATOR	*
		235	*		*
		236	*	*OUTPUT	*
		237	*	* PRINTED OUTPUT AND CARRIER POSITIONING	*
		238	*	* \$PRPOS - 1 BYTE 'DUMMY' CARRIER POSITION INDICATING WHERE THE	*
		239	*	CARRIER SHOULD BE. SET EQUAL TO \$LMRGN AFTER PRINTING.	*
		240	*	* \$BUFPT - 1 BYTE POINTS AT NEXT AVAIL BYTE IN LINE PRINT BUFFER	*
		241	*	* \$LPRP3 - 1 BYTE LINE PRINTER INDICATORS	*
		242	*	* 3LPRI0 - 2 BYTES ONE FOR BUFFER INCREMENT ONE FOR PDAR DISP.	*
		243	*		*
		244	*	*EXTERNAL REFERENCES	*
		245	*	* V\$LPRT2 - VIRTUAL ENTRY SECOND PAGE OF LINE PRINTER ROUTINE	*
		246	*	* V\$LPRTB - VIRTUAL ADDRESS OF THE LINE PRINTER BUFFER	*
		247	*	* I\$LDXR - ENTRY POINT FOR PAGING MODULE V.M. LOAD XR ROUTINE	*
		248	*	* \$LPRI0 - ENTRY POINT FOR PAGING MODULE V.M. CONVERT ADDRESS	*
		249	*		*
		250	*	*EXITS, NORMAL	*
		251	*	EXIT IS TO THE CALLING ROUTINE VIA A BRANCH TO THE V.M. PAGING	*
		252	*	ROUTINE.	*
		253	*		*
		254	*	*EXITS, ERROR	*
		255	*	NONE	*
		256	*		*

DLFPRT - LINE PRINTER ROUTINE

ERR LOC	OBJECT CODE	ADDR	STMT	SOURCE STATEMENT	VER 15, MOD 00	31/05/21	PAGE 430
		257	*	*TABLES/WORKAREAS			*
		258	*	* N/A			*
		259	*				*
		260	*	*ATTRIBLTES			*
		261	*	* NATURALLY RELOCATABLE AND REUSABLE			*
		262	*				*
		263	*	*CHARACTLR CODE DEPENDENCY			*
		264	*	* THE OPERATION OF THIS MODULE DEPENDS UPON AN INTERNAL REPRESENTATION OF THE EXTERNAL CHARACTER SET WHICH IS EQUIVALENT TO THE			*
		265	*	* ONE USED AT ASSEMBLY TIME.			*
		266	*				*
		267	*				*
		268	*	*NOTES			*
		269	*	* ERROR PROCEDLRES			*
		270	*	* IF A PRINTER UNIT CHECK OCCURES. THE LINE IN WHICH THE CHECK			*
		271	*	* OCCURED WILL BE REPRINTED			*
		272	*				*
		273	*	* REGISTER USAGE			*
		274	*	* REGISTER 1 (@BR) IS USED AS A BASE REGISTER FOR DFPRNT			*
		275	*	* REGISTER 2 (@XR) IS USED AS A BASE REGISTER FOR: THE FIRST			*
		276	*	* PAGE OF DLFPRT, LINE PRINTER BUFFER, OR IN THE CASE OF A UNIT			*
		277	*	* CHECK, THE PRINTER ERROR HANDELING ROUTINE 'DFPNDX'.			*
		278	*				*
		279	*	* SAVED/RESTORED AREAS			*
		280	*	* NONE			*
		281	*				*
		282	*	* MODIFICATION CONSIDERATIONS			*
		283	*	* CHANGES TO EITHER DLFPRT OR DFPRNT MAY DIRECTLY AFFECT THE			*
		284	*	* INTERFACE BETWEEN THE TWO MODULES.			*
		285	*				*
		286	*	* REQUIRED MODULES			*
		287	*	* @SYSEQ			*
		288	*	* @FXDEQ			*
		289	*	* @HDWEQ			*
		290	*	* \$V\$EQU			*
		291	*	* \$I\$EQU			*
		292	*	* DFPRNT			*
		293	*				*
		294	*	* OTHER			*
		295	*	* NONE			*
		296	*	*****			*

DLFPRT - LINE PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 431
					298	*****				
	4D00				299		ORG *,256,0			SET STARTING ADDRESS
				2800	300		USING DFPASE,@BR			SET PAGE BASE ADDRESS - DFPRNT
				4D00	301		USING DLFPRT,@XR			SET PAGE BASE ADDRESS
					302	*				
				4D00	303	DLFPRT EQU *				ENTRY BIDIR PRINT
	4D00	7C	87	BC	304	MVI	DFP330+@Q(,@BR),@UCB			SET BRANCH TO LINE PRINTER PAGE
	4D03	B4	02	66	305	ST	DLF155+@OP1(,@XR),@XR			SAVE XR
	4D06	3A	40	03E4	306	SBN	\$LPRP3,@PRINT			SET LINE PRINTER FLAG
	4D0A	2C	01	144A D7	307	MVC	I\$VADR,DLFVD1(@VADDR,@XR)			GET PRINTER BUFFER VADDR
	4D0F	C0	87	1349	308	B	I\$MDFY			LOAD BUFFER & SET PAGE MDFY BIT
	4D13	8C	01	D9 144C	309	MVC	BUFADR(2,@XR),I\$CADR			SAVE BUFFER ADDR
				4D18	310	DLF050 EQU *				PROCESS PRINTER UNIT CHECK
	4D18	7C	25	BD	311	MVI	DFP330+@D1(,@BR),DENTRY			SET ENTRY DISPLACEMENT
	4D1B	BC	87	A9	312	MVI	DLF360+@Q(,@XR),@UCB			FORCE RETURN ENTRY
	4D1E	6C	02	BA F6	313	MVC	DFP333(3,@BR),DLFEOR(,@XR)			SET DLFPRT ERROR ENTRY
					314	*				
	4D22	D0	87	A2	315	B	DFP280(,@BR)			GO CHECK FOR PREV. ERROR
					317	*****				
					318	*				
					319	*	FIND FUNCTION			
					320	*				
					321	*****				
				4D25	322	DLF100 EQU *				RETURN FROM ERROR CHECK
	4D25	BC	80	A9	323	MVI	DLF360+@Q(,@XR),@NOP			RESET ENTRY INDICATOR
	4D28	78	40	F5	324	TBN	DLFIST+@PCTRL(,@BR),@PRINT			IS OP A PRINT ?
	4D2B	F2	90	4A	325	JF	DLF170			CHECK IF BUFFER FULL
					326	*****				
					327	*				
					328	*	ENTRY TO FILL BUFFER			
					329	*				
					330	*****				
	4D2E	39	01	03E4	331	TBF	\$LPRP3,@INDEX			TEST DUMMY PRINT
	4D32	F2	90	0A	332	JF	DLF140			SKIP IF IN USE
	4D35	3A	01	03E4	333	SBN	\$LPRP3,@INDEX			SET DUMMY PRINT POS. USED
	4D39	0C	00	03E5 03C2	334	MVC	\$LPROS(1),\$PRPOS			SAVE TRUE POSITION
				4D3F	335	DLF140 EQU *				UPDATE BUFFER POINTER
					336	*				
					337	*****				
					338	*				
	4D3F	1E	00	03E3 F6	339	ALC	\$BUFPT,DLFIST+@PRCNT(1,@BR)			ADD NEXT COUNT TO BUFFER PTR
	4D44	1E	00	03C2 F6	340	ALC	\$PRPOS(1),DLFIST+@PRCNT(,@BR)			UPDATE HEAD POSITION
					341	*				
					342	*	INCREMENT BUFFER POINTER			
					343	*				
	4D49	2C	01	144A ED	344	MVC	I\$VADR,DLFPCH(@VADDR,@XR)			V.M. PATCH PAGE ENTRY ADDR 1-5
	4D4E	C0	87	1358	345	DLF143 B	I\$CVAD			LOAD PATCH PAGE 1-5
	4D52	8C	01	5A 144C	346	MVC	DLF145+@OP1(@CADDR,@XR),I\$CADR			MOVE CADDR TO BRANCH 1-5
	4D57	C0	87	0000	347	DLF145 B	*-*			1-5
					348	*				
					349	*	MOVE DATA TO BUFFER			
					350	*				
	4D5B	B5	02	D9	351	DLF146 L	BUFADR(,@XR),@XR			XR - BUFFER CADDR
	4D5E	8C	00	00 0000	352	DLF150 MVC	*-*(@VQ,@XR),*-*			MOVE DATA INTO BUFFER
					353	*				

DLFPRT - LINE PRINTER ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 432

```

4D63 C2 02 0000          354 DLF155 LA      *-*,@XR          RESTORE DLFPRT BASE ADDR
                        355 *
                        356 *          TEST FOR CARRAGE RETURN
                        357 *
4D67 7D C0 F5          358          CLI      DLFIST+@PCTRL(,@BR),@PRETR  TEST CARRAGE RETURN ON
4D6A F2 01 4C          359          JNE      DLF175          JUMP TO RETURN IF NO C.R.
                        4D6D 360 DLF160 EQU      *          LOAD PAGE2 LINE PRINTER
4D6D 7C 88 BD          361          MVI      DFP330+@D1(,@BR),DERROR  SET ERROR ENTRY DISP.
4D70 2C 01 144A EB      362 DLF165 MVC      I$VADR,DLFVD2(@VADDR,@XR) VADDR VLPRT2
4D75 E0 87 93          363          B        DLF400(,@XR)          LOAD BASE
                        4D78 365 DLF170 EQU      *          CHECK IF BUFFER EMPTY
4D78 3D 00 03E3          366          CLI      $BUFPT,@ZERO          IS BUFFER EMPTY ?
4D7C E0 01 6D          367          BNE      DLF160(,@XR)          GO TO PRINT EXIT
4D7F 7C 01 DE          368          MVI      DLFPCF(,@BR),@INDEX      SET INDEX ONLY
4D82 7C 87 A0          369          MVI      DFP270+@Q(,@BR),@UCB      FORCE RETURN
4D85 D0 87 92          370          B        DFP240(,@BR)          GO DO I/O
                        372 *
                        373 *          NO ERROR, CHECK FOR PREVIOUS ERROR
                        374 *
4D88 F2 00 1D          375 DLF350 JC      DLF360,*-*          JUMP NO PREVIOUS ERROR
4D89          376          ORG      DLF350+@Q          * INITIALIZE
4D89 87          377          DC      AL1(@UCB)          * TO INDICATE
4D8B          378          ORG      DLF350+@INST3          * NO PREVIOUS PRINTER ERROR
4D8B BC 87 89          379          MVI      DLF350+@Q(,@XR),@UCB      RESET ERROR INDICATOR
4D8E 2C 01 144A E3      380 DLF355 MVC      I$VADR,DLFRTY(@VADDR,@XR) VADDR RETRY ENTRY VLPRT2
                        4D93 381 DLF400 EQU      *          PREPARE TO EXIT LINE PTR PAGE1
4D93 3C 80 12B6          382          MVI      I$LBFR,@NOP          FORCE LINE PRINTER UNLOCK
4D97 C0 87 1358          383          B        I$CVAD          LOAD LINE PRINTER PAGE2
4D9B 8C 01 A7 144C      384          MVC      DLF425+@OP1(@CADDR,@XR),I$CADDR  MOVE CADDR TO BR
4DA0 C0 87 1354          385          B        I$LOCK          LOCK PAGE VLPRT2          1-5
4DA4 C0 87 0000          386 DLF425 B        *-*          BRANCH TO PAGE2
                        4DA8 E0 00 25          388 DLF360 BC      DLF100(,@XR),*-*          FORMAT NEXT LINE / GO TO ENTRY
4DA9          389          ORG      DLF360+@Q          * INITIALIZE
4DA9 80          390          DC      AL1(@NOP)          * TO FORMAT
4DAB          391          ORG      DLF360+@INST3          * NEXT LINE TO BE PRINTED
4DAB 2C 01 144A EF      392          MVC      I$VADR,DLFPC1(@VADDR,@XR) V.M. PATCH PAGE ENTRY ADDR 1-5
4DB0 E0 87 4E          393 DLF375 B        DLF143(,@XR)          BRANCH TO MV CADDR TO BRANCH 1-5
                        395 *****
                        396 *****          RETURN TO CALLER          *****
                        397 *****
                        4DB3 398 RETURN EQU      *          LINE PRINTER RETURN AREA
4DB3 0C 00 03C2 03C1      399          MVC      $PRPOS(1),$LMRGN          SET DUMMY POSITION LEFT MGN
                        4DB9 400 DLF175 EQU      *          RETURN FROM DLFPRT
4DB9 7C 80 BC          401          MVI      DFP330+@Q(,@BR),@NOP          RESET BRANCH TO LINR PRINTER
4DBC 7C 80 A0          402          MVI      DFP270+@Q(,@BR),@NOP          RESET DFPRNT EXIT
4DBF 6C 02 BA F3          403          MVC      DFP333(3,@BR),DFPEOR(,@XR)  RESTORE DFPRNT ERROR TEST
4DC3 7C 11 E0          404          MVI      DLFPCF+2(,@BR),@TBLIX          RESTORE MATRIX PRINTER END
4DC6 3B 40 03E4          405          SBF      $LPRP3,@PRINT          RESET LINE PRINTER FLAG
4DCA D0 87 CA          406          B        DFP300(,@BR)          RETURN TO CALLER
                        407 *
                        408 *****
4DCD          409 DLFPRPE EQU      *          PRINTER UNIT CHECK ENTRY

```


DLFPRT - LINE PRINTER ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 433

4DCD	C0 87 1330		410	B	I\$LDXR	BR TO FORCE DLFPRT TO BE MOST	
			411	*		* RECENTLY USED PAGE	
4DD1	4D00	4DD2	412	DC	AL2(V\$LPRT)	DLFPRT VADDR	
4DD3	D0 87 D3		413	B	DFPRPE-DFPRNT(,@BR)	GO PROCESS LOAD ERP SECTION	
			414	*			
			415		*****		
4DD6	4F00	4DD7	416	DLFVD1 DC	AL(@VADDR)(V\$LPRB)	LINE PRINTER BUFFER PAGE	
4DD8	0000	4DD9	417	BUFADR DC	XL2'00'	SAVED BUFFER ADDR	
			418	*			
4DDA	0000	4ddb	419	DFPWIDTH DC	XL2'00'	LINE WIDTH	
4DDC	00	4DDC	420	DFPRES DC	XL1'00'	LINE COUNT	
4DDD	0000	4DDE	421	BUFRWK DC	XL2'00'	BUFFER POINTER	
4DDF	00	4DDF	422	DLFBPT DC	XL1'00'	BUFFER INCREMENT	
			423	*			
4DE0	0025	4DE1	424	DLFMAR DC	AL2(DLF500-VLPRT2)	DISPLACENENT TO FORMAT LINE	
4DE2	4E49	4DE3	425	DLFRTY DC	AL2(V\$LPRT2+DLF700-VLPRT2)	RETRY ENTRY POINT	
			426	*			
4DE4	00	4DE4	427	DFPPOS DC	XL1'00'	CHARACTER POSITION ON LINE	
4DE5	8080C00001	4DE9	428	LPRCMD DC	XL5'8080C00001'	LINE PRINTER CMDS.	
4DEA	4E00	4DEB	429	DLFVD2 DC	AL2(V\$LPRT2)	LINE PRINTER PAGE2	
		004E	430	DLFX4E EQU	X'4E'	VLPRT2 LOCK BIT	1-5
		0053	431	DLFX53 EQU	X'53'	VLPRT3 LOCK BIT	1-5
		0090	432	DLTABL EQU	X'90'	TAB LEFT AND CHAIN	
4DEC	5391	4DED	433	DLFPCH DC	AL2(V\$PCH2+DLF400-@D1-DLFPRT)	PATCH PAGE ENTRY ADDR	1-5
4DEE	53B6	4DEF	434	DLFPC1 DC	AL2(V\$PCH2+DLF175-@DD2-DLFPRT)	PATCH PAGE ENTRY ADDR	1-5
4DF0	00	4DF0	435	DLFSWC DC	XL1'00'	RETURN CARRIAGE SWITCH	1-5
		00A0	436	DLTABR EQU	X'A0'	TAB RIGHT AND CHAIN	
		0088	437	DERROR EQU	DLF350-DLFPRT	ERROR CHECK ENTRY DISP.	
		0025	438	DENTRY EQU	DLF100-DLFPRT	ENTRY RETURN DISP.	
		0001	439	DLFRTN EQU	X'01'	RETURN CARRIAGE INDICATOR	1-5
			440	*			
			441	*	INSTRUCTION MODIFICATION TP DFPRNT AT DFP335		
			442	*			
4DF1	D1 E0 D3		443	TIO	DFPRPE-DFPRNT(,@BR),@PERR	FORCE BRANCH TO DFPRNT ERROR	
		4DF3	444	DFPEOR EQU	*-1	LAST BYTE OF FORCE DFPRNT ERROR	
4DF4	E1 E0 CD		445	TIO	DLFRPE(,@XR),@PERR	FORCE BRANCH TO DLFPRT ERROR	
		4DF6	446	DLFEOR EQU	*-1	LAST BYTE DLFPRT FORCE ERROR	
			447		*****		
			448		*****	END V\$LPRT	*****
			449		*****		

DLFPRT - LINE PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 434
					451	*****				
					452	*				
					453	*	ENTRY TO FORMAT PRINT LINE			
					454	*				
					455	*****				
4E00					456	ORG	*,256,0 SET STARTING ADDRESS			
				2800	457	USING	DFPASE,@BR SET PAGE BASE ADDRESS - DFPRNT			
				4D00	458	USING	DLFPRT,@XR SET PAGE BASE ADDRESS			
				4E00	459	VLPRT2 EQU	*			
4E00	2C	01	144A	D7	460	MVC	I\$VADR,DLFVD1(@VADDR,@XR) GET BUFFER ADDR			
4E05	C0	87	1354		461	B	I\$LOCK LOCK PRINT BUFFER			
4E09	8C	01	D9	144C	462	MVC	BUFADR(2,@XR),I\$CADR SAVE LINE PRINTER BUFFER CADDR			
4E0E	8C	01	DE	144C	463	MVC	BUFRWK(2,@XR),I\$CADR SAVE BUFFER ADDRESS			
					464	*****				
					465	*				
					466	*	DETERMINE ANY MARGIN COMPUTATION REQUIRED			
					467	*				
					468	*****				
4E13	8C	00	DC	03E3	469	MVC	DFPRES(1,@XR),\$BUFPT SAVE COUNT			
4E18	8C	00	DB	03C0	470	MVC	DFPWT(1,@XR),\$RMRGN SET RIGHT MARGIN VALUE			
4E1D	8F	00	DB	03C1	471	SLC	DFPWT(1,@XR),\$LMRGN CALCULATE WIDTH			
4E22	F2	87	04		472	J	DLF525 CONTINUE			
					473	*				
				4E25	474	DLF500 EQU	* FORMAT LINE			
4E25	AE	01	DE	DB	475	ALC	BUFRWK(2,@XR),DFPWT(,@XR) GET NEXT PDAR ADDR			
				4E29	476	DLF525 EQU	*			
4E29	AD	00	DB	DC	477	CLC	DFPWT(1,@XR),DFPRES(,@XR) COMPARE WIDTH TO LINE LENGTH			
4E2D	F2	02	0C		478	JNL	DLF550 JUMP LENGTH < WIDTH			
					479	*****				
					480	*				
					481	*	COMPUTE MARGIN AND FORMAT DATA			
					482	*				
					483	*****				
4E30	AF	00	DC	DB	484	SLC	DFPRES(1,@XR),DFPWT(,@XR) NEXT LINE = RESIDUAL			
4E34	2C	00	03E3	DB	485	MVC	\$BUFPT(1),DFPWT(,@XR) SET NEW LINE - WIDTH			
4E39	F2	87	08		486	J	DLF600 GO TO FORMAT NEXT LINE			
					487	*				
					488	*	COUNT < WIDTH			
					489	*				
				4E3C	490	DLF550 EQU	* \$BUFPT RESIDUAL			
4E3C	2C	00	03E3	DC	491	MVC	\$BUFPT(1),DFPRES(,@XR)			
4E41	7C	87	A0		492	MVI	DFP270+@Q(,@BR),@UCB FORCE LINE PRINT EXIT			
					493	*				
				4E44	494	DLF600 EQU	* FORMAT LINE			
4E44	8C	00	DF	03E3	495	MVC	DLFBPT(1,@XR),\$BUFPT SAVE BUFFER POINTER			
				4E49	496	DLF700 EQU	* PRINT RETRY ENTRY POINT			
4E49	B1	E4	DE		497	LIO	BUFRWK(,@XR),@PDAR SET DATA ADDR			
4E4C	6C	04	E2	E9	498	MVC	DFPPCO(5,@BR),LPRCMD(,@XR) SET LINE PRINTER CMDS.			
					499	*				
					500	*	COMMON MARGIN ENTRY			
					501	*				
4E50	7C	00	9E		502	MVI	DFP260-DFPRNT+@D1(,@BR),@ZERO SET TO PRINT RIGHT			
4E53	8C	00	E4	03E5	503	MVC	DFPPOS(1,@XR),\$LPROS GET ACTUAL POSITION			
4E58	0C	00	03E5	03C1	504	MVC	\$LPROS(1),\$LMRGN SET REFERENCE			
4E5E	0E	00	03E5	03E3	505	ALC	\$LPROS(1),\$BUFPT UPDATE PRINT POSITION			
					506	*				

DLFPRT - LINE PRINTER ROUTINE

ERR LOC OBJECT CODE ADDR STMT SOURCE STATEMENT VER 15, MOD 00 31/05/21 PAGE 435

4E64	1F	00	03E3	E7	507	SLC	\$BUFPT(1),DLF001(,@BR)	COUNT LESS ONE
4E69	4C	00	E1	03E3	508	MVC	DLFPCF+3(1,@BR),\$BUFPT	MOVE DATA COUNT TO PCF
4E6E	2D	00	03C1	E4	509	CLC	\$LMRGN(1),DFPPOS(,@XR)	AT LEFT MARGIN ?
4E73	F2	81	61		510	JE	DLF950	JUM IF AT LEFT MARGIN
					512	*****		
					513	*		
					514	* CALCULATE TAB		
					515	***	IS PRINT POSITION < HALF OF DATA COUNT ?	
					516	*	TAKE ONE-HALF OF COUNT ROUTINE (DIVIDE)	
					517	*		
					518	*****		
4E76	7C	00	E4		519	MVI	DLFORK-1(,@BR),@ZERO	
4E79	4C	00	E5	03E3	520	MVC	DLFORK(1,@BR),\$BUFPT	MOVE COUNT TO WORK AREA
4E7E	5E	01	E5	E5	521	ALC	DLFORK(2,@BR),DLFORK(,@BR)	ADD THREE TIMES
4E82	5E	01	E5	E5	522	ALC	DLFORK(2,@BR),DLFORK(,@BR)	
4E86	5E	01	E5	E5	523	ALC	DLFORK(2,@BR),DLFORK(,@BR)	
4E8A	58	01	E4	E4	524	MZN	DLFORK-1(,@BR),DLFORK-1(,@BR)	MOVE ZONE NUM
4E8E	58	02	E4	E5	525	MNZ	DLFORK-1(,@BR),DLFORK(,@BR)	DLFORK-1=1/2 NEXT LINE CNT
					526	*		
					527	*	MOVE CARRAGE TO LEFT MARGIN OR TAB	
					528	*		
4E92	8F	00	E4	03C1	529	SLC	DFPPOS(1,@XR),\$LMRGN	PRPOS WITH IN WIDTH
4E97	9D	00	E4	E4	530	CLC	DFPPOS(1,@XR),DLFORK-1(,@BR)	IS PRPOS > 1/2 NEXT LINE ?
4E9B	F2	82	2E		531	JL	DLF900	SET TO GO TO LEFT MARGIN
					533	*****		
					534	*	DETERMINE TAB DIRECTION	
					535	*****		
4E9E	1E	00	03E3	E7	536	ALC	\$BUFPT(1),DLF001(,@BR)	COUNT PLUS ONE
4EA3	0C	00	03E5	03C1	537	MVC	\$LPROS(1),\$LMRGN	SET POSITION TO LEFT MARGIN
4EA9	7C	01	9E		538	MVI	DFP260-DFPRNT+2(,@BR),@B1	SET TO PRINT LEFT
4EAC	8D	00	E4	03E3	539	CLC	DFPPOS(1,@XR),\$BUFPT	COMPARE PRINT POS. TO LINE LNG
4EB1	F2	81	23		540	JE	DLF950	JUMP EQUAL LINE & POSITION
4EB4	F2	84	10		541	JH	DLF800	JUMP TO TAB LEFT

DLPRT - LINE PRINTER ROUTINE

				543	*				
				544	*	COMPUTE TAB RIGHT			
				545	*				
4EB7	2F	00	03E3 E4	546		SLC \$BUFPT(1),DFPPOS(,@XR)	GET TAB DISTANCE		
4EBC	8C	00	E4 03E3	547		MVC DFPPOS(1,@XR),\$BUFPT	SAVE BUFFER POINTER		
4EC1	7C	A0	DE	548		MVI DLFPCF(,@BR),DLTABR	SET TAB RIGHT OP		
4EC4	F2	87	08	549		J DLF920	JUMP TO SET TAB COUNT		

DLFPRT - LINE PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 437
					551	*				
					552	*	COMPUTE LEFT TAB			
					553	*				
				4EC7	554	DLF800	EQU *			FIND TAB LEFT COUNT
4EC7	8F	00	E4 03E3		555		SLC DFPPOS(1,@XR), \$BUFPT			GET TAB DISTANCE
				4ECC	556	DLF900	EQU *			SET TAB LEFT
4ECC	7C	90	DE		557		MVI DLFPCF(, @BR), DLTABL			SET TAB LEFT OP
				4ECF	558	DLF920	EQU *			HARDWARE REQUIREMENT
4ECF	9F	00	E4 E7		559		SLC DFPPOS(1,@XR), DLF001(, @BR)			ONE LESS
4ED3	6C	00	DF E4		560		MVC DLFPCF+1(, @BR), DFPPOS(, @XR)			SET TAB COUNT
				4ED7	561	DLF950	EQU *			SET AT LEFT MARGIN INDICATION
4ED7	2C	01	03EA DF		562		MVC \$LPRIO, DLFBPT(2, @XR)			SAVE PDAR ADDR & BUFR. INCR.
4EDC	74	02	E5		563		ST DLFORK(, @BR), @XR			SAVE XR
4EDF	B5	02	D9		564		L BUFADR(, @XR), @XR			XR = CADDR LINE PRINTER BUFFER
4EE2	74	02	DD		565		ST DFPAPC(, @BR), @XR			SAVE BUFFER ADDR
4EE5	7C	FB	DD		566		MVI DFPAPC(, @BR), DLFCAR			GET DISP. TO COMMANDS
4EE8	9C	04	FF E2		567		MVC BFPCRO-LPBUFR(5, @XR), DFPPCO(, @BR)			MOVE COMMANDS TO PCAR
4EEC	75	02	E5		568		L DLFORK(, @BR), @XR			RESTORE XR TO VLPRT2
4EEF	3C	00	03E3		569		MVI \$BUFPT, @ZERO			SET BUFFER PTR = 0
4EF3	D0	87	99		570		B DFP250(, @BR)			GO TO DFPRNT TO DO I/O
					571	*				
					572	*****				
					573	*****	END V\$LPR2			*****
					574	*****				

DLPRT - LINE PRINTER ROUTINE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 438
					576	*****	*****			
					577	*	LINE DRINTER BUFFER AREA			
					578	*****	*****			
4F00					579	ORG	*,256,0			
				4F00	580	USING	LPBUFR,@XR	SET	BASE	FOR
				4F00	581	LPBUFR	EQU *	LINE	PRINTER	BUFFER
4F00				4FFA	582	DS	CL251	LINE	PRINTER	BUFFER
					584	*****	LINE PRINTER COMMANDS PCAR *****			
				4FFB	585	BFPCAR	EQU *	LINE	PRINTER	COMMANDS
4FFB	0000000000			4FFF	586	DC	XL5'00'	LINE	PRINTER	COMMANDS
				4FFF	587	BFPCRO	EQU *-1	LAST	BYTE	OF
				00FB	588	DLFCAR	EQU BFPCAR-LPBUFR	DISPLACEMENT	TO	PCAR
					589	*****	*****			

VLPRT3 - BI-DIRECTIONAL PRINT ROUTINE CORRECTION PAGE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00 31/05/21 PAGE 439
5300				591		ORG	X'5300'	PATCH AREA 1-5
				592			*****	
				593		*		*
				594		*	THIS PAGE 15 USED BY THE BI-DIRECTIONAL PRINT ROUTINES TO CORRECT	*
				595		*	PROBLEMS CONNECTED WITS APAR NUMBERS 968 AND 972. THE ROUTINES	*
				596		*	USING THIS PAGE AND THEIR ENTRY POINTS ARE:	*
				597		*	DFPRNT - VLPRT3, DFPENT	*
				598		*	FZSPRT - VLPRT4	*
				599		*	DLPRT - VLPRT5, VLPRT6	*
				600		*		*
				601			*****	
				602	5300	VLPRT3 EQU	*	DFPRNT INTERFACE 1-5
				603	5300	DFPCHK EQU	*	1-5
				604	2800		USING DFPASE,@BR	1-5
				605	4D00		USING DLPRT,@XR	1-5
5300	7D	00	F6	606		CLI	DFPIST+@PRCNT(,@BR),@ZERO	ANOTHER LINE TO PRINT ? 1-5
5303	F2	01	0B	607		JNE	DFPENT	CONTINUE PROCESSING LINE 1-5
5306	F2	87	30	608		J	DFPULK	GO TO UNLOCK ROUTINE 1-5
5309	C0	87	1354	609		B	I\$LOCK	LOCK PAGE VLPRT3 1-5
530D	6C	03	F8 03	610		MVC	DFPIST+@PLNGH-1(@PLNGH,@BR),@PLNGH-1(,@XR)	MOVE THE PRT1-5
				611		*		* PARAMETER LIST TO WRK AREA 1-5
5311	5C	02	F4 F8	612		DFPENT MVC	DFPDSV(@CADDR+1,@BR),DFPIST+@PDATA(,@BR)	MOVE THE PRT 1-5
				613		*		* CNT AND DATA ADDRESS 1-5
5315	4C	00	FB 03C2	614		MVC	DFPSYC+@SYCNT(1,@BR),\$PRPOS	SAVE HD POSITION FOR SYNC 1-5
531A	5C	01	DF F6	615		MVC	DFPPCF+@PRCNT(2,@BR),DFPIST+@PRCNT(,@BR)	SET CTRL+CNT 1-5
531E	39	1E	03E4	616		TBF	\$LPRP3,@KENAB	TEST FOR MATRIX PRINT MODE 1-5
5322	D0	90	23	617		BF	DFP115(,@BR)	BRANCH IF MATRIX PRINT 1-5
5325	38	80	03D2	618		TBN	\$IOIND,\$LNPTR	IS LINE PRINTER REQUESTED ? 1-5
5329	D0	90	23	619		BF	DFP115(,@BR)	BRANCH IF NOT 1-5
532C	C0	87	1330	620		B	I\$LDXR	BRANCH TO LOAD PAGE 1-5
5330	4D00			621	5331	DC	AL(@VADDR)(V\$LPRT)	LINE PRINTER PAGE 1-5
5332	C0	87	1354	622		B	I\$LOCK	GO LOCK PAGE 1-5
5336	E0	87	00	623		B	@ZERO(,@XR)	BRANCH TO LINE PRINTER LINK 1-5
				624		*		1-5
				625	5339	DFPULK EQU	*	UNLOCK ALL LINE PRINTER 1-5
				626		*		* ROUTINE PAGES 1-5
5339	7C	80	A3	627		MVI	DFP280+@Q-DFPASE(,@BR),@NOP	SET ERP INDR OFF 1-5
533C	1C	01	144A 1F	628		MVC	I\$VADR,DFP105(2,@BR)	DLPRT VM ADDR 1-5
5341	C0	87	1350	629		B	I\$UNLK	UNLOCK PAGE 1-5
5345	3C	4E	1449	630		MVI	I\$VADR-1,DLFX4E	VLPRT2 VM ADDR 1-5
5349	C0	87	1350	631		B	I\$UNLK	UNLOCK PAGE 1-5
534D	3C	53	1449	632		MVI	I\$VADR-1,DLFX53	VLPRT3 VM ADDR 1-5
5351	C0	87	1350	633		B	I\$UNLK	UNLOCK PAGE 1-5
5355	C0	87	12D3	634		B	I\$RTRN	BRANCH TO CALLING PGM-FZPRNT 1-5
				635		*		1-5
				636	5359	VLPRT4 EQU	*	FZSPRT INTERFACE 1-5
				637	3600	USING	FZSP3B,@BR	1-5
5359	4E	00	DB 03C2	638	FZS991	ALC	FZS3CC(,@BR),\$PRPOS(1)	ADD PRT ZONE LNG TO CURRENT 1-5
535E	5D	00	DB 6A	639		CLC	FZS3CC(,@BR),FZS3RM(1,@BR)	* CARRIER POSITION - BRANCH 1-5
5362	F2	84	03	640		JH	FZS992	* IF RIGHT MGN IS EXCEEDED 1-5
5365	D0	87	B9	641		B	FZS710(,@BR)	BRANCH BACK IF NOT 1-5
				643	FZS992	TBN	\$IOIND,\$LNPTR	IS LINE PRINTER REQUESTED ? 1-5
536C	F2	90	03	644		JF	FZS993	NO, DON'T SET CARRIAGE RTN 1-5
536F	7C	C0	F2	645		MVI	FZS3PF(,@BR),@PRETR	SET CARRIAGE RETURN INDR 1-5
5372	D2	02	F2	646	FZS993	LA	FZS3PL(,@BR),@XR	LOAD DATA OUTDUT PPL CADDR 1-5

VLPRT3 - BI-DIRECTIONAL PRINT ROUTINE CORRECTION PAGE

ERR	LOC	OBJECT	CODE	ADDR	STMT	SOURCE	STATEMENT	VER 15, MOD 00	31/05/21	PAGE 440
	5375	C0	87	12B1	647	B	I\$CALL			LINK TO EXECUTE PRINTER IOCR 1-5
	5379	2800		537A	648	DC	AL(@VADDR)(V\$SPRT)			MATRIX PRINTER IOCR VADDR 1-5
	537B	7C	40	F2	649	MVI	FZS3PF(, @BR), @PRINT			SET INDR TO PRINT ONLY 1-5
	537E	0D	00	044A 0D5A	650	CLC	\$PRDEV-1, I\$WRK2-1(1)			IF CRT IS NOT A SYSTEM PRINT 1-5
	5384	F2	82	06	651	JL	FZS994			* DEVICE, EXIT ROUTINE 1-5
	5387	C0	87	12B1	652	B	I\$CALL			LINK TO EXCUTE PRINT ON CRT 1-5
	538B	3700		538C	653	DC	AL(@VADDR)(FZS800)			PRINT CRT RTN VADDR 1-5
	538D	C0	87	12D3	654	FZS994 B	I\$RTRN			RETURN TO 1ST PRINT RTN PAGE 1-5
				5391	656	VLPRT5 EQU *				DLFPRT INTERFACE NO. 1 1-5
				2800	657		USING DFPASE, @BR			1-5
				4D00	658		USING DLFPRT, @XR			1-5
	5391	5F	01	F2 E7	659	SLC	DLFDSV-2(2, @BR), DLF001(, @BR) COUNT LESS ONE			1-5
	5395	BD	01	F0	660	CLI	DLFSWC(, @XR), DLFRTN IS SWITCH SET FOR RTN CARRAGE			1-5
	5398	F2	81	04	661	JE	DLF960 YES, DO NOT INCR DATA PTR			1-5
	539B	5E	01	F8 F2	662	ALC	DLFIST+@PDATA(2, @BR), DLFDSV-2(, @BR) GET DATA ADDR PTR			1-5
	539F	9C	01	62 F8	663	DLF960 MVC	DLF150+@DOP2(2, @XR), DLFIST+@PDATA(, @BR) SET DATA ADDR			1-5
	53A3	9C	00	5F F2	664	MVC	DLF150+@VQ(1, @XR), DLFDSV-2(, @BR) GET COUNT FOR MVC			1-5
	53A7	8C	00	60 03E3	665	MVC	DLF150+@D1(1, @XR), \$BUFPT MOVE BUFFER DISP. INTO INST.			1-5
	53AC	9F	00	60 E7	666	SLC	DLF150+@D1(1, @XR), DLF001(, @BR) DISP. LESS ONE			1-5
	53B0	BC	00	F0	667	MVI	DLFSWC(, @XR), X'00' SET CARRAGE RETURN SW OFF			1-5
	53B3	E0	87	5B	668	B	DLF146(, @XR) CONTINUE			1-5
					669	*				
				53B6	670	VLPRT6 EQU *				DLFPRT INTERFACE NO. 2 1-5
	53B6	7C	40	F5	671	MVI	DLFIST+@PCTRL(, @BR), @PRINT SET PRINT ONLY			1-5
	53B9	6C	00	F6 DC	672	MVC	DLFIST+@PRCNT(, @BR), DFPRES(1, @XR) BUF PTR - RESIDUAL			1-5
	53BD	6C	00	F2 DC	673	MVC	DLFDSV-2(, @BR), DFPRES(1, @XR) DATA COUNT - RESIDUAL			1-5
	53C1	0C	00	03C2 03C1	674	MVC	\$PRPOS(1), \$LMRGN SET DUMMY POSITION-LEFT MGN.			1-5
	53C7	BC	01	F0	675	MVI	DLFSWC(, @XR), DLFRTN SET SWITCH FOR RTN CARRIAGE			1-5
	53CA	E0	87	25	676	B	DLF100(, @XR) CONTINUE PROCESSING			1-5
					678	*##### X'5400' #####				
					679	* NOT S C A N N E D (GENERAL PURPOSE BUFFERS 1 & 2.)				
					680	*##### X'55FF' #####				
				FFFF	681	END				

TOTAL STATEMENTS IN ERROR IN THIS ASSEMBLY = 0

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 441

SYMBOL	LEN	VALUE	DEFN	REFERENCES
\$\$\$CMD	001	0020	0660	
\$\$\$DAT	001	0040	0659	
\$\$\$EPL	001	0091	0656	
\$\$\$ERN	001	0080	0710	
\$\$\$FUN	001	0010	0661	
\$\$\$NLN	001	00A0	0706	
\$\$\$STD	001	0081	0655	
\$\$\$001	040	2DF8	2361	
\$\$BNLN	001	0605	0636	0638
\$\$CDBS	001	08C0	0686	
\$\$CDND	001	0666	0645	
\$\$CDRD	001	0890	0684	0686
\$\$CKEY	001	0603	0634	
\$\$CKFF	001	0B3D	0666	
\$\$COFF	001	0B44	0665	
\$\$CSNS	001	209C	0695	
\$\$DATB	001	0BBF	0667	
\$\$EOSA	001	0AFE	0664	
\$\$ERSK	001	1C00	0705	
\$\$FITS	001	1D00	0713	
\$\$FLIB	001	06FF	0712	
\$\$ILEN	001	0601	0630	0632 0636
\$\$ILHD	001	0600	0628	0630
\$\$INLN	001	0607	0643	0645 0647
\$\$INND	001	06FA	0647	
\$\$KBDT	001	09E1	0654	0658
\$\$KBSN	001	09E2	0658	0663
\$\$KLD1	001	0600	0718	9833 0564
\$\$KLD2	001	0700	0720	
\$\$KLD3	001	0C00	0722	
\$\$LPOS	001	09EB	0663	
\$\$PCNT	001	07E9	0679	
\$\$PLYN	001	2004	0693	9632 1122
\$\$PRES	001	0890	0652	0654 0664 0665 0666 0667 0684
\$\$PRFL	001	2143	0697	
\$\$PRNT	001	0707	0673	0674 0678 0679 9631
\$\$PRTN	001	0782	0674	
\$\$PSIO	001	07CE	0678	
\$\$PYCD	001	2200	0699	
\$\$PYMP	001	2000	0691	0693 0695 0697 0699
\$\$SLIB	001	1C00	0708	
\$\$TPCD	001	0606	0638	0643
\$\$UPAR	001	0602	0632	0634
\$\$WSPB	001	1E00	0711	
\$\$XIND	001	06FF	0709	0712
\$\$ZERO	001	0000	0224	0225 0227 0228 0229 0233 0691
\$ABORT	001	0010	0337	
\$BASIC	001	0080	0395	
\$BIGCD	001	0080	0471	9901
\$BLDPL	001	0579	0604	0606 8412*
\$BLNOE	001	0569	0594	8410*
\$BLOAD	001	0522	0585	0587 0590 0603 0604 9781 0430 8408 8409
\$BLRTN	001	0550	0593	0594 8413
\$BRSAV	001	03C5	0282	0283 0871* 0891
\$BSADR	001	0587	0609	0611 8378 8404
\$BUFPT	001	03E3	0490	0491 0686 1474* 1488* 0339* 0366 0469 0485* 0491* 0495 0505 0507*

CROSS REFERENCE																	
SYMBOL	LEN	VALUE	DEFN	REFERENCES											VER 15, MOD 00 31/05/21 PAGE 442		
				0508	0520	0536*	0539	0546*	0547	0555	0569*	0665					
\$CABLD	001	04B4	0563	0564													
\$CAERK	001	0469	0540	0543													
\$CAERR	001	03CD	0288	0290													
\$CAIPL	001	049D	0559	0561													
\$CALLI	001	0008	0480														
\$CARDI	001	0001	0251														
\$CARPL	001	04A1	0561	0563													
\$CIENT	001	0483	0550	0551	0998	1007											
\$CIEXT	001	0480	0549	0550													
\$CIMSK	001	0476	0546	0549													
\$CISUS	001	0496	0554	0559	1960	6591											
\$CLBFR	001	0010	0438														
\$CMDKY	001	0008	0350														
\$CMODE	001	0002	0400														
\$CONFIG	001	03DD	0463	0473	9901												
\$CRPOS	001	03E2	0489	0490	4764	4771	4878	5534	7250	8659	8985						
\$CRTAD	001	044D	0528	0529													
\$CRTAV	001	0002	0344														
\$CRTDN	001	0002	0368														
\$CRTIN	001	03D3	0365	0372													
\$CRTNO	001	0004	0347														
\$CRTPU	001	0004	0369														
\$CRTSP	001	0008	0370														
\$CRTUP	001	0001	0367														
\$CRUSH	001	0080	0476														
\$CSDPL	001	050E	0575	0576													
\$C0001	001	0464	0532	0538													
\$DATE	001	043A	0513	0514													
\$DBGUF	001	03E0	0475	0484	9367												
\$DBLOK	001	0001	0425														
\$DFDET	001	03E8	0496	0497													
\$DISKN	001	0025	0227	9772	9792	0419	0444	8393	8395	8926	8933	8935	9003	9005 0137			
				0148													
\$DKERR	001	0008	0406														
\$DKSIZ	001	03D7	0450	0458	0499												
\$DK100	001	0001	0452														
\$DK200	001	0002	0453														
\$DK400	001	0004	0454														
\$DK600	001	0008	0455														
\$DK800	001	0010	0456														
\$DPLSV	001	0449	0524	0526	9788	0438											
\$DTNMB	001	0040	0271														
\$DTRDR	001	0040	0359														
\$ENDNU	001	0600	0618	0628	0652	0673	0709	0718	0720	0722	2753						
\$ERDPL	001	046F	0543	0545</													

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 443

SYMBOL	LEN	VALUE	DEFN	REFERENCES
\$ER050	001	0363	0233	
\$ER1N2	001	0050	0300	
\$EXADR	001	0517	0578	0580
\$EXCMD	001	0001	0332	
\$EXFTR	001	043B	0514	0519 9548 0879 2193 4135 5639 8373 8375 8635 9034 0029 0032 0034
\$FCIND	001	0010	0410	
\$FDIND	001	0040	0417	
\$FEARR	001	0004	0225	
\$FEMAP	001	0588	0611	0612
\$FILIB	001	03DA	0461	0462
\$FITIN	001	0010	0386	
\$FUIND	001	0020	0415	
\$GUFIO	001	0583	0608	0609
\$GUFIR	001	0008	0260	
\$HISTE	001	042E	0511	0512 1433* 1483*
\$HIST1	001	0435	0512	0513 1248* 1428*
\$HRDER	001	0020	0356	1245 1482
\$INDR1	001	03D4	0372	0398
\$INDR2	001	03D5	0398	0423 1253* 1426 1429*
\$INDR3	001	03D6	0423	0450
\$INLNO	001	03CF	0290	0292 0304 0311 2098 8609
\$INRPT	001	0020	0268	
\$IOIND	001	03D2	0339	0365 1245* 1482* 0618 0643
\$IOPGS	001	0010	0479	9367
\$IOYES	001	0002	0254	
\$IPLDV	001	05FF	0615	0618
\$IRKEY	001	0020	0478	
\$KEYBD	001	03E1	0484	0489
\$KEYCD	001	03C3	0248	0282
\$KEYDT	001	0040	0392	
\$KE090	001	00DE	0228	
\$KE130	001	01D5	0229	
\$KYBSY	001	0010	0265	
\$LDRTN	001	0571	0603	8411*
\$LEVEL	001	03DF	0473	0475
\$LIST	001	0002	0427	
\$LMRGN	001	03C1	0243	0245 0884 1314 1317 1445 5532 7248 8983 0399 0471 0504 0509 0529 0537 0674
\$LNPTR	001	0080	0362	0618 0643
\$LOADB	001	054A	0587	
\$LOADR	001	051A	0580	0583
\$LPRIO	001	03EA	0497	1473 0562*
\$LPROS	001	03E5	0492	0494 0696 1422 1463* 0334* 0503 0504* 0505* 0537*
\$LPRP3	001	03E4	0491	0492 0585* 0662* 0694 0697* 1127* 1130* 1420 1423* 1440 1460 1462* 1484 1949* 2013* 2214* 6664* 0306* 0331 0333* 0405* 0616
\$MOUNT	001	0020	0441	
\$MPDWN	001	0001	0341	1482
\$NEXTB	001	03E6	0494	0495
\$NEXTL	001	03E7	0495	0496
\$NOENB	001	0008	0433	
\$NOLST	001	0004	0257	
\$NUCBS	001	03C0	0240	0241
\$NWRKF	001	0080	0446	
\$NWRKR	001	0040	0443	
\$PASWD	001	042D	0510	0511

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 444

SYMBOL	LEN	VALUE	DEFN	REFERENCES
\$PAUSD	001	04BA	0564	0566
\$PAUSE	001	0002	0334	
\$PGMDT	001	0020	0389	
\$PGMST	001	0010	0353	
\$PKERT	001	0419	0508	0510
\$PLST1	001	0454	0529	0530
\$PLST2	001	045B	0530	0531
\$PLST3	001	0462	0531	0532
\$PRDEV	001	044B	0526	0528 9540 9543* 9547* 9548* 9563* 1120 1125 2197 2209 4406 4580 5643 5655 8639 8689 9038 9050 0650
\$PRESN	001	0002	0377	
\$PROCI	001	0001	0374	
\$PRPOS	001	03C2	0245	0248 0696* 1288* 1294 1305* 1312 1317* 1422* 1456* 1463 4481 4488 4604 5532 7248 8649 8983 0334 0340* 0399* 0614 0638 0674*
\$PSDBR	001	04FA	0569	
\$PSDXR	001	04F2	0568	0569
\$PSTEP	001	0004	0335	
\$PSTMT	001	0008	0336	
\$PTCH1	001	03F5	0499	0503
\$READY	001	0080	0419	
\$REORD	001	0040	0477	
\$RLOAD	001	051E	0583	0585
\$RMRGN	001	03C0	0241	0243 0883 1295 4411 8650 0470
\$RSTR	001	04D6	0566	0568 0570 0575
\$RUNIT	001	0001	0313	
\$SFAID	001	050D	0571	
\$SPRNT	001	0465	0538	0540
\$SRTRN	001	04FE	0570	0571
\$STEPT	001	0002	0314	
\$SWPCR	001	0511	0576	0578
\$TABLN	001	03CB	0285	0288
\$TFLOW	001	0008	0320	
\$TRACE	001	0004	0315	
\$TRALL	001	0010	0321	
\$TROVR	001	054E	0590	0593
\$TRUNK	001	0080	0273	
\$TRVAR	001	0020	0322	
\$UNMSK	001	048D	0551	0554
\$USRDR	001	03DC	0462	0463
\$VMDEF	001	0080	0326	
\$VOLF1	001	03FE	0505	0506
\$VOLF2	001	040E	0507	
\$VOLID	001	03F6	0503	0504 0508
\$VOLR1	001	03F6	0504	0505
\$VOLR2	001	0406	0506	0507
\$WAITF	001	057F	0606	0608 9793 0445 8396 8936 9006 0149
\$WFDEF	001	0040	0520	
\$WFLOK	001	0008	0383	
\$WFNME	001	0443	0519	0524
\$WSIND	001	0004	0380	
\$XIND1	001	03D0	0311	0330
\$XIND2	001	03D1	0330	0339
\$XIND3	001	03D8	0458	0461
\$XPREC	001	0040	0323	
\$XRSAB	001	03C7	0283	0285
\$ZTRAD	001	05A2	0612	

CROSS REFERENCE

SYMBOL LEN VALUE DEFN REFERENCES VER 15, MOD 00 31/05/21 PAGE 445

\$12K 001 0004 0467
\$16CKY 001 0008 0469
\$16K 001 0002 0466
\$22IMP 001 0001 0464
###BL 001 0000 1437
###CK 001 0000 1565
###CN 001 0000 1533
###CO 001 0000 1325
###CS 001 0000 1385
###DR 001 0000 1129
###ER 001 0000 1329
###FS 001 0000 1425
###IN 001 0000 1569
###PW 001 0000 1573
###RS 001 0000 1405
###SA 001 0000 1393
###SS 001 0000 1389
###VU 001 0600 1349
###0T 001 0700 1121
###1T 001 0000 1125
###BCO 001 0600 1137
###BOV 001 0800 1409
###DPR 001 0700 1145
###DRE 001 0889 1161
###DSP 001 2800 1181
###ECM 001 0C00 1441
###EFK 001 0C00 1461
###ERR 001 0C00 1433
###EXM 001 0C00 1321
###FIL 001 0E00 1401
###FIS 001 0E00 1397
###FML 001 0200 1529
###FMS 001 0200 1369
###GRA 001 0889 1293
###GUF 001 0C00 1429
###INL 001 0600 1509
###INS 001 0600 1133
###KAL 001 0C00 1297
###KCA 001 0C00 1513
###KCH 001 0C00 1265
###KCN 001 0C00 1381
###KCT 001 0C00 1233
###KDE 001 0C00 1229
###KDI 001 0D00 1309
###KDN 001 0C00 1217
###KDO 001 0E00 1313
###KED 001 0C00 1153
###KEN 001 0C00 1157
###KEX 001 0C00 1177
###KGO 001 0C00 1149
###KHE 001 0C00 1333
###KKE 001 0C00 1561
###KLI 001 0C00 1237
###KLL 001 0920 1537
###KLO 001 0C00 1241
###KME 001 0D00 1221

8423 8431

3965

9813 0540

CROSS REFERENCE

SYMBOL LEN VALUE DEFN REFERENCES VER 15, MOD 00 31/05/21 PAGE 446

###KMO 001 0C00 1165
###KNA 001 0C00 1277
###KOV 001 0E00 1197
###KPA 001 0C00 1173
###KPO 001 0C00 1261
###KPR 001 0C00 1285
###KRE 001 0C00 1205
###KRL 001 0700 1301
###KRM 001 0C00 1169
###KRN 001 0700 1189
###KRO 001 0D00 1193
###KRS 001 0C00 1517
###KRU 001 0C00 1213
###KRV 001 0800 1305
###KSA 001 0C00 1249
###KSE 001 0E00 1289
###KSO 001 0C20 1341
###KSS 001 0C00 1273
###KSV 001 0980 1269
###KSY 001 0C00 1281
###KWI 001 0C00 1209
###KWR 001 0C00 1201
###LOA 001 0600 1141
###MIP 001 0C00 1337
###SDS 001 0C00 1449
###SFF 001 0E00 1453
###SFL 001 0F00 1445
###SFO 001 1500 1417
###SFS 001 0C00 1413
###SPA 001 0C00 1253
###SPO 001 0806 1257
###SPS 001 0C00 1245
###STR 001 1600 1421
###TDC 001 1000 1225
###TSY 001 1000 1185
###TVK 001 0FC0 1361
###UAL 001 0C00 1377
###UAT 001 0900 1473
###UCD 001 0900 1481
###UCN 001 0C00 1465
###UCP 001 0700 1469
###UDE 001 0C00 1485
###UDI 001 0C00 1489
###UEX 001 0C00 1373
###UIN 001 0C00 1477
###UPA 001 0C00 1457
###UPO 001 0C00 1525
###UPT 001 0C00 1521
###VCR 001 2000 1317
###VLO 001 0600 1353
###VOD 001 0600 1357
###VVM 001 0000 1365
###VXI 001 0600 1345
###ZDU 001 1100 1497
###ZLB 001 1100 1541
###ZLO 001 1100 1501

9823 0552

CROSS REFERENCE

SYMBOL LEN VALUE DEFN REFERENCES VER 15, MOD 00 31/05/21 PAGE 447

###ZLV	001	0F00	1557	
###ZL1	001	0F00	1545	
###ZL2	001	0F00	1549	
###ZL3	001	0C00	1553	
###ZTR	001	1000	1493	
###ZUT	001	0C00	1505	
##BLN	001	18D4	1436	
##CKT	001	2118	1564	
##CNF	001	2000	1532	
##COR	001	0800	1324	
##CSA	001	1000	1384	
##DRT	001	0000	1128	
##ERM	001	0928	1328	
##FSP	001	1880	1424	
##INV	001	212C	1568	8439
##PWR	001	2300	1572	
##RSP	001	1780	1404	
##SAV	001	1180	1392	
##SSA	001	1128	1388	
##VUF	001	0B08	1348	
##0TR	001	0000	1120	
##1TR	001	0080	1124	
##@BL	001	0001	1438	
##@CK	001	0004	1566	
##@CN	001	0001	1534	
##@CO	001	003A	1326	
##@CS	001	003A	1386	
##@DR	001	0008	1130	
##@ER	001	0032	1330	
##@FS	001	0030	1426	
##@IN	001	003A	1570	
##@PW	001	00C0	1574	
##@RS	001	0030	1406	
##@SA	001	0108	1394	
##@SS	001	0001	1390	
##@VU	001	0002	1350	
##@0T	001	0018	1122	
##@1T	001	0018	1126	
##@BCO	001	0018	1138	
##@BOV	001	0018	1410	
##@DPR	001	0005	1146	
##@DRE	001	0001	1162	
##@DSP	001	0004	1182	
##@ECM	001	0006	1442	
##@EFK	001	0002	1462	
##@ERR	001	0003	1434	
##@EXM	001	0003	1322	
##@FIL	001	0009	1402	
##@FIS	001	0009	1398	8430
##@FML	001	0052	1530	
##@FMS	001	0052	1370	
##@GRA	001	0003	1294	
##@GUF	001	0010	1430	
##@INL	001	0010	1510	
##@INS	001	0010	1134	
##@KAL	001	000F	1298	

CROSS REFERENCE

SYMBOL LEN VALUE DEFN REFERENCES VER 15, MOD 00 31/05/21 PAGE 448

#\$@KCA	001	000C	1514	
#\$@KCH	001	000C	1266	
#\$@KCN	001	0010	1382	
#\$@KCT	001	0009	1234	
#\$@KDE	001	0010	1230	
#\$@KDI	001	0005	1310	
#\$@KDN	001	0010	1218	
#\$@KDO	001	000C	1314	
#\$@KED	001	000E	1154	
#\$@KEN	001	0006	1158	
#\$@KEX	001	0003	1178	
#\$@KGO	001	0002	1150	
#\$@KHE	001	000C	1334	
#\$@KKE	001	0006	1562	
#\$@KLI	001	0011	1238	
#\$@KLL	001	0001	1538	
#\$@KLO	001	0008	1242	
#\$@KME	001	0003	1222	
#\$@KMO	001	0004	1166	
#\$@KNA	001	0008	1278	
#\$@KOV	001	0009	1198	
#\$@KPA	001	0005	1174	
#\$@KPO	001	000D	1262	
#\$@KPR	001	0009	1286	
#\$@KRE	001	0002	1206	
#\$@KRL	001	0004	1302	
#\$@KRM	001	0003	1170	
#\$@KRN	001	0003	1190	
#\$@KRO	001	000A	1194	
#\$@KRS	001	000A	1518	
#\$@KRU	001	0003	1214	
#\$@KRV	001	000D	1306	
#\$@KSA	001	0011	1250	
#\$@KSE	001	0004	1290	
#\$@KSO	001	0005	1342	
#\$@KSS	001	000B	1274	
#\$@KSV	001	0002	1270	
#\$@KSY	001	000F	1282	
#\$@KWI	001	0002	1210	
#\$@KWR	001	0002	1202	
#\$@LOA	001	0013	1142	
#\$@MIP	001	000D	1338	
#\$@SDS	001	0004	1450	
#\$@SFF	001	0008	1454	
#\$@SFL	001	0005	1446	
#\$@SFO	001	0003	1418	
#\$@SFS	001	0011	1414	
#\$@SPA	001	0004	1254	
#\$@SPO	001	0003	1258	
#\$@SPS	001	0001	1246	
#\$@STR	001	0002	1422	
#\$@TDC	001	0003	1226	
#\$@TSY	001	0003	1186	
#\$@TVK	001	0001	1362	
#\$@UAL	001	0011	1378	
#\$@UAT	001	000C	1474	

CROSS REFERENCE

SYMBOL LEN VALUE DEFN REFERENCES VER 15, MOD 00 31/05/21 PAGE 449

#\$@UCD	001	000B	1482	
#\$@UCN	001	0009	1466	
#\$@UCP	001	000F	1470	
#\$@UDE	001	000E	1486	
#\$@UDI	001	0008	1490	
#\$@UEX	001	000E	1374	
#\$@UIN	001	000F	1478	
#\$@UPA	001	0004	1458	
#\$@UPO	001	0005	1526	
#\$@UPT	001	0012	1522	
#\$@VCR	001	0008	1318	
#\$@VLO	001	0002	1354	
#\$@VOD	001	0016	1358	
#\$@VVM	001	0030	1366	
#\$@VXI	001	0002	1346	
#\$@ZDU	001	0008	1498	
#\$@ZLB	001	0002	1542	
#\$@ZLO	001	000C	1502	
#\$@ZLV	001	0006	1558	
#\$@ZL1	001	0007	1546	
#\$@ZL2	001	000D	1550	
#\$@ZL3	001	000A	1554	
#\$@ZTR	001	0001	1494	
#\$@ZUT	001	0014	1506	
#\$BCOM	001	0080	1136	
#\$BOLV	001	1780	1408	
#\$DPRI	001	014C	1144	
#\$DREA	001	0200	1160	
#\$DSPL	001	0240	1180	
#\$ECMA	001	1900	1440	
#\$EFKE	001	1990	1460	
#\$ERRP	001	18C0	1432	
#\$EXMS	001	07D4	1320	
#\$FILN	001	1724	1400	8438
#\$FIST	001	1700	1396	8429
#\$FMLN	001	1E00	1528	
#\$FMST	001	0D00	1368	
#\$GRAP	001	0690	1292	
#\$GUFU	001	1880	1428	
#\$INLN	001	1C84	1508	
#\$INST	001	0020	1132	
#\$KALL	001	06A4	1296	
#\$KCAL	001	1CC4	1512	
#\$KCHA	001	053C	1264	
#\$KCND	001	0F80	1380	
#\$KCTL	001	03BC	1232	
#\$KDEL	001	035C	1228	
#\$KDIS	001	0744	1308	
#\$KDNT	001	0300	1216	
#\$KDOV	001	0780	1312	
#\$KEDI	001	0188	1152	
#\$KENA	001	01C4	1156	
#\$KEXT	001	0234	1176	
#\$KGOS	001	0180	1148	
#\$KHEL	001	0A30	1332	
#\$KKEY	001	2100	1560	

CROSS REFERENCE

SYMBOL LEN VALUE DEFN REFERENCES VER 15, MOD 00 31/05/21 PAGE 450

#\$KLIS	001	0400	1236
#\$KLLA	001	2004	1536
#\$KLOG	001	0444	1240
#\$KMER	001	030C	1220
#\$KMOU	001	0204	1164
#\$KNAM	001	05C0	1276
#\$KOVN	001	0290	1196
#\$KPAS	001	0220	1172
#\$KPOO	001	0508	1260
#\$KPRT	001	063C	1284
#\$KREA	001	02BC	1204
#\$KRLA	001	0700	1300
#\$KRMO	001	0214	1168
#\$KRNU	001	0280	1188
#\$KROV	001	028C	1192
#\$KRSU	001	1D24	1516
#\$KRUN	001	02CC	1212
#\$KRVL	001	0710	1304
#\$KSAV	001	0488	1248
#\$KSET	001	0680	1288
#\$KSOV	001	0AC8	1340
#\$KSSP	001	0594	1272
#\$KSVL	001	058C	1268
#\$KSYM	001	0600	1280
#\$KWID	001	02C4	1208
#\$KWRI	001	02B4	1200
#\$LOAD	001	0100	1140
#\$MIPP	001	0A80	1336
#\$SDSY	001	192C	1448
#\$SFFI	001	193C	1452
#\$SFLO	001	1918	1444
#\$SFOV	001	1844	1416
#\$SFSY	001	1800	1412
#\$SPAC	001	04CC	1252
#\$SPOV	001	04DC	1256
#\$SPSY	001	0484	1244
#\$STRO	001	1850	1420
#\$TDCK	001	0350	1224
#\$TSYK	001	0250	1184
#\$TVKB	001	0BAC	1360
#\$UALL	001	0F00	1376
#\$UATR	001	1A38	1472
#\$UCDI	001	1AD8	1480
#\$UCNF	001	19B8	1464
#\$UCPL	001	19DC	1468
#\$UDEL	001	1B24	1484
#\$UDIS	001	1B5C	1488
#\$UEXL	001	0EA8	1372
#\$UINI	001	1A88	1476
#\$UPAC	001	1980	1456
#\$UPOV	001	1D24	1524
#\$UPTF	001	1D5C	1520
#\$VCRT	001	07B4	1316
#\$VLOA	001	0B80	1352
#\$VODK	001	0B88	1356
#\$VVMR	001	0C00	1364

CROSS REFERENCE

SYMBOL LEN VALUE DEFN REFERENCES VER 15, MOD 00 31/05/21 PAGE 451

#\$VXIT	001	0B00	1344	
#\$ZDUM	001	1BA4	1496	
#\$ZLBM	001	2008	1540	
#\$ZLOA	001	1BC4	1500	
#\$ZLVR	001	20B0	1556	
#\$ZL1M	001	2010	1544	
#\$ZL2M	001	2030	1548	
#\$ZL3M	001	2088	1552	
#\$ZTRA	001	1B9C	1492	
#\$ZUTM	001	1C14	1504	
#@#BAD	001	0455	0880	
#@#IO1	001	0459	0888	
#@#IO2	001	045D	0889	
#@#TAT	001	0941	0916	
#@#TBA	001	09A1	0920	
#@#TFS	001	0941	0914	
#@#TSY	001	0941	0918	
#@#VFP	001	0700	0906	
#@#VLP	001	093D	0909	
#@#WDB	001	050C	0901	
#@#WFT	001	0500	0899	
###BA	001	0001	0881	
###IO	001	0001	0893	
###SC	001	0002	0890	
###TA	001	0010	0917	
###TB	001	0010	0921	
###TS	001	0005	0919	
###TW	001	0020	0915	
###VM	001	0100	0910	
###WD	001	00BD	0902	
###WF	001	0003	0900	
###04	001	0004	0892	
###08	001	0008	0891	
###BOV	001	0018	0869	
###ECM	001	0006	0883	
###ERR	001	0003	0877	
###GUF	001	0010	0873	
###LDS	001	0002	0879	
###SDS	001	0004	0875	
###SFF	001	0008	0887	
###SFL	001	0005	0885	9822 0551
###SFO	001	0005	0895	
###SFS	001	0011	0871	
###VSF	001	0010	0923	
###VSL	001	000F	0924	9812 0539
###VTR	001	0001	0908	
#@BOVL	001	0400	0868	
#@CORS	001	0005	0774	
#@ECMA	001	0481	0882	
#@ERRP	001	0441	0876	
#@GUFU	001	0401	0872	
#@LDSV	001	044D	0878	
#@MVSD	001	0001	0782	
#@NERO	001	0003	0776	
#@OBRA	001	0002	0778	
#@PTFL	001	0006	0797	

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 452

SYMBOL	LEN	VALUE	DEFN	REFERENCES
#@PTFS	001	0001	0796	
#@SDSY	001	04AD	0874	
#@SFFI	001	04BD	0886	
#@SFLO	001	0499	0884	9821 0550
#@SFOV	001	04C4	0894	
#@SFSY	001	0480	0870	
#@VCNT	001	0002	0794	
#@VLAB	001	0001	0789	
#@VLSD	001	0001	0780	
#@VSFI	001	09A1	0922	9811 0538
#@VTRL	001	0708	0907	
#@WAF1	001	0401	0867	
#@WAR1	001	0400	0866	
#CNDIS	001	0001	0749	
#CNFIG	001	0005	0785	
#CORSV	001	0010	0773	
#DKEXT	001	0002	0756	
#FIGSC	001	0001	0786	
#FMSTD	001	0000	0002	
#HISCT	001	0006	0763	
#HISDX	001	0003	0758	
#HISLN	001	0008	0755	0756 1248 1428
#HISN1	001	0003	0761	
#HISN2	001	0005	0762	
#HISTC	001	0007	0765	
#HISTN	001	0009	0767	
#HISTQ	001	0000	0759	
#HISTR	001	0001	0760	
#HISTS	001	0008	0766	
#HISTV	001	000F	0768	
#HSEND	001	0007	0764	
#HSENT	001	0001	0757	
#IOSDR	001	0019	0784	
#MVSDR	001	000D	0781	
#NEROV	001	009C	0775	
#OBRAD	001	001D	0777	
#PKCNT	001	0002	0742	
#PKMRW	001	002B	0743	
#PKRDD	001	0003	0740	
#PKRTD	001	0003	0739	
#PKRTL	001	0004	0746	
#PKVRD	001	000B	0744	
#PKVWD	001	0007	0745	
#PKWTD	001	0001	0741	
#PTFDA	001	00DC	0795	
#RDWTL	001	0004	0747	
#SDRDK	001	0011	0783	
#VLSDR	001	000C	0779	
#VLTBE	001	0008	0734	
#VOLF1	001	0009	0787	
#VOLNG	001	0006	0732	0734 0756
#VOLOC	001	0005	0733	
#VOLR1	001	0008	0788	
#VTCF1	001	0025	0791	
#VTCF2	001	0027	0793	
#VTCR1	001	0024	0790	

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 453

SYMBOL	LEN	VALUE	DEFN	REFERENCES
#VTCR2	001	0026	0792	
@\$D1BF	001	0008	2230	9356 9360
@\$D1DC	001	0000	2229	
@\$D1DF	001	001E	2234	
@\$D1DP	001	0016	2233	
@\$D1DV	001	000E	2232	
@\$D1E1	001	0000	2223	
@\$D1FS	001	000A	2231	
@\$D1SW	001	001F	2236	9355
@\$D2AS	001	0002	2241	
@\$D2BS	001	0003	2248	9644 9752 0410
@\$D2CB	001	0005	2251	9653 9965* 9966* 9967* 0116 0119* 0406 0453* 0457* 0484* 0497*
@\$D2CF	001	0001	2240	9382* 9383 9527 0098 0605* 0607 0656*
@\$D2CP	001	0005	2249	9649 9711* 9796 9842 0274 0312* 0327 0462 0493* 0634 0634*
@\$D2CS	001	0004	2250	9644 9695 9752 0408 0410 0471* 0645
@\$D2CY	001	0006	2252	
@\$D2DA	001	0007	2253	
@\$D2DC	001	0000	2245	9384 9541 9545 9573 0111 0611 0642
@\$D2DD	001	0009	2254	9693 0635 0635*
@\$D2EE	001	000F	2257	0645 0645*
@\$D2E1	001	0040	2244	9401 9412 0605
@\$D2FS	001	000B	2255	9692
@\$D2IO	001	0001	2246	9528 9530 9532 9536* 9552 9582 9654 9656 9754 9845 9933 0099 0101 0103 0107* 0268 0451 0613 0615 0617* 0625* 0641* 0644 0644*
@\$D2LC	001	000D	2256	0256 0272 0313* 0326 0454* 0499* 0636 0636*
@\$D2PN	001	000A	2242	
@\$D2SF	001	000B	2243	
@\$D2VB	001	0002	2247	9650 9797 9843 0120 0275 0328 0463
@\$L1BF	001	0008	2263	
@\$L1DC	001	0001	2262	
@\$L1DF	001	0008	2265	9334 9360
@\$L1DP	001	0008	2266	
@\$L1DV	001	0006	2267	
@\$L1E	001	0020	2261	9365
@\$L1FS	001	0002	2264	
@\$L2AS	001	0001	2273	
@\$L2BS	001	0001	2280	
@\$L2CB	001	0001	2283	9653 9850 9900 9919 9965 9966 9967 0484 0497 0561 0562
@\$L2CF	001	0002	2272	
@\$L2CP	001	0002	2281	9649 9711 9796 9842 0274 0312 0327 0371 0493 0634
@\$L2CS	001	0001	2282	9695 0410 0471 0560
@\$L2DA	001	0002	2284	
@\$L2DC	001	0001	2277	0644
@\$L2DD	001	0002	2285	9693 0635
@\$L2E	001	0010	2276	9413 0645
@\$L2FS	001	0002	2286	9692 9694 9696
@\$L2HD	001	0040	2271	
@\$L2IO	001	0001	2278	9654 0644
@\$L2LC	001	0002	2287	0256 0272 0313 0326 0378 0454 0499 0559 0636
@\$L2PN	001	0008	2275	
@\$L2SF	001	0002	2274	
@\$L2VB	001	0001	2279	9644 9650 9752 9797 9843 9939 0120 0275 0328 0463
@\$MBCD	001	0020	2301	9573 0111
@\$MBCR	001	0008	2303	9545
@\$MBEN	001	000C	2291	0676
@\$MBND	001	0000	2298	

CROSS REFERENCE											
SYMBOL	LEN	VALUE	DEFN	REFERENCES	VER 15, MOD 00 31/05/21 PAGE 454						
@\$MBPD	001	0080	2299								
@\$MBPT	001	0010	2302	9541							
@\$MBPU	001	0001	2294								
@\$MBSD	001	0040	2300	0642							
@\$M2CI	001	0008	2318	9530	0099	0107	0613	0641			
@\$M2CO	001	0004	2319	9528	9536	0101	0615	0641			
@\$M2EF	001	0002	2293	9552	9582	9656	9754	9845	9933	0617	0625
@\$M2FI	001	0080	2307	0103							
@\$M2FO	001	0040	2308	9532							
@\$M2FP	001	0020	2309	9667	0268						
@\$M2FT	001	0010	2312	0451							
@\$M2NS	001	00FF	2292								
@@E001	001	0000	2111	2113							
@@E003	001	0001	2113	2115							
@@E004	001	0002	2115	2117							
@@E005	001	0003	2117	2119							
@@E006	001	0004	2119	2121							
@@E007	001	0005	2121	2123							
@@E008	001	0006	2123	2125							
@@E009	001	0007	2125	2127							
@@E010	001	0008	2127	2129							
@@E011	001	0009	2129	2131							
@@E012	001	000A	2131	2133							
@@E013	001	000B	2133	2135							
@@E014	001	000C	2135	2137							
@@E015	001	000D	2137	2139							
@@E016	001	000E	2139	2141							
@@E017	001	000F	2141	2143							
@@E018	001	0010	2143	2145							
@@E019	001	0011	2145	2147							
@@E020	001	0012	2147	2149							
@@E021	001	0013	2149	2151							
@@E023	001	0014	2151	2153							
@@E024	001	0015	2153	2155							
@@E025	001	0016	2155	2157							
@@E026	001	0017	2157	2159							
@@E027	001	0018	2159	2161							
@@E028	001	0019	2161	2163							
@@E029	001	001A	2163	2165							
@@E030	001	001B	2165	2167							
@@E031	001	001C	2167	2169							
@@E032	001	001D	2169	2171							
@@E035	001	001E	2171	2173							
@@E036	001	001F	2173	2175							
@@E037	001	0020	2175	2177							
@@E038	001	0021	2177	2179							
@@E039	001	0022	2179	2181							
@@E040	001	0023	2181	2183							
@@E041	001	0024	2183	2185							
@@E042	001	0025	2185	2187							
@@E043	001	0026	2187	2189							
@@E044	001	0027	2189	2191							
@@E045	001	0028	2191	2193							
@@E046	001	0029	2193	2195							

VER 15, MOD 00 31/05/21 PAGE 454

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 455

SYMBOL	LEN	VALUE	DEFN	REFERENCES
@@E100	001	0000	1583	1585
@@E101	001	0001	1585	1587
@@E102	001	0002	1587	1589
@@E103	001	0003	1589	1591
@@E110	001	0004	1591	1593
@@E112	001	0005	1593	1595
@@E113	001	0006	1595	1597
@@E114	001	0007	1597	1599
@@E115	001	0008	1599	1601
@@E116	001	0009	1601	1603
@@E117	001	000A	1603	1605
@@E120	001	000B	1605	1607
@@E122	001	000C	1607	1609
@@E123	001	000D	1609	1611
@@E124	001	000E	1611	1613
@@E129	001	000F	1613	1615
@@E130	001	0010	1615	1617
@@E131	001	0011	1617	1619
@@E133	001	0012	1619	1621
@@E134	001	0013	1621	1623
@@E135	001	0014	1623	1625
@@E136	001	0015	1625	1627
@@E137	001	0016	1627	1629
@@E138	001	0017	1629	1631
@@E139	001	0018	1631	1633
@@E142	001	0019	1633	1635
@@E143	001	001A	1635	1637
@@E150	001	001B	1637	1639
@@E151	001	001C	1639	1641
@@E160	001	001D	1641	1643
@@E162	001	001E	1643	1645
@@E163	001	001F	1645	1647
@@E164	001	0020	1647	1649
@@E200	001	0021	1649	1651
@@E205	001	0022	1651	1653
@@E210	001	0023	1653	1655
@@E211	001	0024	1655	1657
@@E212	001	0025	1657	1659
@@E213	001	0026	1659	1661
@@E215	001	0027	1661	1663
@@E216	001	0028	1663	1665
@@E217	001	0029	1665	1667
@@E220	001	002A	1667	1669
@@E221	001	002B	1669	1671
@@E222	001	002C	1671	1673
@@E223	001	002D	1673	1675
@@E225	001	002E	1675	1677
@@E226	001	002F	1677	1679
@@E227	001	0030	1679	1681
@@E228	001	0031	1681	1683
@@E229	001	0032	1683	1685
@@E230	001	0033	1685	1687
@@E232	001	0034	1687	1689
@@E234	001	0035	1689	1691
@@E237	001	0036	1691	1693
@@E240	001	0037	1693	1695

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 456

SYMBOL	LEN	VALUE	DEFN	REFERENCES
@@E241	001	0038	1695	1697 2709
@@E242	001	0039	1697	1699
@@E248	001	003A	1699	1701
@@E249	001	003B	1701	1703
@@E250	001	003C	1703	1705
@@E251	001	003D	1705	1707
@@E252	001	003E	1707	1709
@@E253	001	003F	1709	1711
@@E254	001	0040	1711	1713
@@E255	001	0041	1713	1715
@@E256	001	0042	1715	1717
@@E300	001	0043	1717	1719
@@E301	001	0044	1719	1721
@@E302	001	0045	1721	1723
@@E303	001	0046	1723	1725
@@E304	001	0047	1725	1727
@@E305	001	0048	1727	1729
@@E308	001	0049	1729	1731
@@E310	001	004A	1731	1733
@@E315	001	004B	1733	1735
@@E316	001	004C	1735	1737
@@E320	001	004D	1737	1739
@@E325	001	004E	1739	1741
@@E330	001	004F	1741	1743
@@E335	001	0050	1743	1745
@@E338	001	0051	1745	1747
@@E340	001	0052	1747	1749
@@E350	001	0053	1749	1751
@@E351	001	0054	1751	1753
@@E352	001	0055	1753	1755
@@E360	001	0056	1755	1757
@@E361	001	0057	1757	1759
@@E362	001	0058	1759	1761
@@E371	001	0059	1761	1763
@@E380	001	005A	1763	1765
@@E390	001	005B	1765	1767
@@E400	001	005C	1767	1769
@@E410	001	005D	1769	1771
@@E415	001	005E	1771	1773
@@E417	001	005F	1773	1775
@@E420	001	0060	1775	1777
@@E430	001	0061	1777	1779
@@E432	001	0062	1779	1781
@@E433	001	0063	1781	1783
@@E450	001	0064	1783	1785
@@E451	001	0065	1785	1787
@@E460	001	0066	1787	1789
@@E461	001	0067	1789	1791
@@E464	001	0068	1791	1793
@@E465	001	0069	1793	1795
@@E466	001	006A	1795	1797
@@E467	001	006B	1797	1799
@@E469	001	006C	1799	1801
@@E470	001	006D	1801	1803
@@E471	001	006E	1803	1805
@@E473	001	006F	1805	1807

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 457

SYMBOL	LEN	VALUE	DEFN	REFERENCES
@@E474	001	0070	1807	1809
@@E475	001	0071	1809	1811
@@E476	001	0072	1811	1813
@@E477	001	0073	1813	1815
@@E478	001	0074	1815	1817
@@E479	001	0075	1817	1819
@@E480	001	0076	1819	1821
@@E481	001	0077	1821	1823
@@E482	001	0078	1823	1825
@@E483	001	0079	1825	1827
@@E484	001	007A	1827	1829
@@E485	001	007B	1829	1831
@@E486	001	007C	1831	1833
@@E487	001	007D	1833	1835
@@E488	001	007E	1835	1837
@@E489	001	007F	1837	1839
@@E490	001	0080	1839	1841
@@E491	001	0081	1841	1843
@@E492	001	0082	1843	1845
@@E493	001	0083	1845	1847
@@E494	001	0084	1847	1849
@@E495	001	0085	1849	1851
@@E496	001	0086	1851	1853
@@E497	001	0087	1853	1855
@@E498	001	0088	1855	1857
@@E500	001	0089	1857	1859
@@E501	001	008A	1859	1861
@@E530	001	008B	1861	1863
@@E531	001	008C	1863	1865
@@E535	001	008D	1865	1867
@@E540	001	008E	1867	1869
@@E541	001	008F	1869	1871
@@E542	001	0090	1871	1873
@@E543	001	0091	1873	1875
@@E544	001	0092	1875	1877
@@E545	001	0093	1877	1879
@@E546	001	0094	1879	1881
@@E547	001	0095	1881	1883
@@E548	001	FFFF	2087	
@@E549	001	0096	1883	1885
@@E550	001	0097	1885	1887
@@E551	001	0098	1887	1889
@@E552	001	0099	1889	1891
@@E553	001	009A	1891	1893
@@E554	001	009B	1893	1895
@@E555	001	009C	1895	1897
@@E556	001	009D	1897	1899
@@E558	001	009E	1899	1901
@@E570	001	009F	1901	1903
@@E571	001	00A0	1903	1905
@@E572	001	00A1	1905	1907
@@E573	001	00A2	1907	1909
@@E574	001	00A3	1909	1911
@@E575	001	FFFF	2089	
@@E578	001	00A4	1911	1913
@@E579	001	FFFF	2091	

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 458

SYMBOL	LEN	VALUE	DEFN	REFERENCES
@@E580	001	FFFF	2093	
@@E585	001	00A5	1913	1915
@@E595	001	FFFF	2095	
@@E597	001	FFFF	2097	
@@E598	001	FFFF	2099	
@@E600	001	00A6	1915	1917
@@E601	001	00A7	1917	1919
@@E602	001	00A8	1919	1921
@@E603	001	00A9	1921	1923
@@E604	001	00AA	1923	1925
@@E606	001	00AB	1925	1927
@@E607	001	00AC	1927	1929
@@E608	001	00AD	1929	1931
@@E609	001	00AE	1931	1933
@@E610	001	00AF	1933	1935
@@E611	001	00B0	1935	1937
@@E612	001	00B1	1937	1939
@@E613	001	00B2	1939	1941
@@E614	001	00B3	1941	1943
@@E700	001	00B4	1943	1945
@@E701	001	00B5	1945	1947 9247
@@E710	001	00B6	1947	1949 9336 9340 9358
@@E712	001	00B7	1949	1951 9534 0105
@@E713	001	00B8	1951	1953
@@E714	001	00B9	1953	1955 0361
@@E715	001	00BA	1955	1957 9698
@@E716	001	00BB	1957	1959
@@E717	001	00BC	1959	1961
@@E718	001	00BD	1961	1963 0213 2509
@@E720	001	00BE	1963	1965 3647
@@E721	001	00BF	1965	1967 3707
@@E723	001	00C0	1967	1969
@@E724	001	00C1	1969	1971 5314 5493
@@E725	001	00C2	1971	1973
@@E726	001	00C3	1973	1975
@@E727	001	00C4	1975	1977
@@E728	001	00C5	1977	1979 9213
@@E729	001	00C6	1979	1981
@@E730	001	00C7	1981	1983
@@E732	001	00C8	1983	1985 9223
@@E752	001	00C9	1985	1987 7701 7924 8186
@@E753	001	00CA	1987	1989 7704
@@E754	001	00CB	1989	1991 8189 8356 8382
@@E755	001	00CC	1991	1993 7439 7717 7792 8070
@@E756	001	00CD	1993	1995 8192
@@E757	001	00CE	1995	1997 8366
@@E758	001	00CF	1997	1999 7714
@@E759	001	00D0	1999	2001 7709
@@E760	001	00D1	2001	2003
@@E761	001	00D2	2003	2005 8437
@@E762	001	00D3	2005	2007 7436
@@E763	001	00D4	2007	2009
@@E764	001	00D5	2009	2011 8401
@@E765	001	00D6	2011	2013 8363
@@E766	001	00D7	2013	2015 7928 8359 8386
@@E767	001	00D8	2015	2017 8197

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 459

SYMBOL	LEN	VALUE	DEFN	REFERENCES
@@E768	001	00D9	2017	2019
@@E769	001	00DA	2019	2021
@@E770	001	00DB	2021	2023
@@E771	001	00DC	2023	2025
@@E772	001	00DD	2025	2027
@@E773	001	00DE	2027	2029
@@E774	001	00DF	2029	2031 5160
@@E775	001	00E0	2031	2033 5364
@@E776	001	00E1	2033	2035 4806
@@E777	001	00E2	2035	2037 4128
@@E778	001	00E3	2037	2039 4108
@@E779	001	00E4	2039	2041 4119
@@E780	001	00E5	2041	2043 6193
@@E781	001	00E6	2043	2045 7325
@@E782	001	00E7	2045	2047 7349
@@E783	001	00E8	2047	2049 6629
@@E784	001	00E9	2049	2051 6617
@@E785	001	00EA	2051	2053 6913
@@E786	001	00EB	2053	2055 6863
@@E790	001	00EC	2055	2057 5124
@@E791	001	00ED	2057	2059 5235
@@E792	001	00EE	2059	2061
@@E793	001	00EF	2061	2063 6205
@@E794	001	00F0	2063	2065 6196
@@E795	001	00F1	2065	2067 5613
@@E796	001	00F2	2067	2069 5599
@@E797	001	00F3	2069	2071 7317
@@E798	001	00F4	2071	2073 7341
@@E800	001	FFFF	2101	
@@E801	001	FFFF	2103	
@@E802	001	FFFF	2105	
@@E803	001	FFFF	2107	
@@E804	001	FFFF	2109	
@@E900	001	00F5	2073	2075 2705
@@E901	001	00F6	2075	2077 2707
@@E902	001	00F7	2077	2079 2706
@@E903	001	00F8	2079	2081 2708
@@E905	001	00F9	2081	2083
@@E906	001	00FA	2083	2085
@@E910	001	00FB	2085	2704
@@M250	001	2D00	2316	2125
@@M251	001	2D04	2320	2134
@@M256	001	2D08	2324	2146
@@M257	001	2D0C	2328	
@@M258	001	2D10	2332	
@@M259	001	2D14	2336	
@@M260	001	2D18	2340	
@@T250	001	2D1C	2344	2318
@@T251	001	2D24	2346	2322
@@T256	001	2D2D	2348	2326
@@T257	001	2D52	2350	2330
@@T258	001	2D75	2352	2334
@@T259	001	2D8C	2354	2338
@@T260	001	2DAC	2356	2342
@ALTFL	001	0001	0963	
@ARR	001	0008	0017	7112 8422 8755 9720 1097 1104 1118 1137 2188 2541 2584 2759

CROSS REFERENCE

SYMBOL LEN VALUE DEFN REFERENCES VER 15, MOD 00 31/05/21 PAGE 460

				2959 9567	3168	3401	3421	4600	4617	4874	4892	5578	5634	8704	9029
@ASIGN	001	007C	0072												
@ASTER	001	005C	0070												
@BCRDL	001	0050	0089	9903											
@BE	001	0081	0044	2444											
@BF	001	0090	0053												
@BH	001	0084	0042												
@BKSPC	001	0010	1060												
@BL	001	0082	0043												
@BLANK	001	0040	0066	9326	0143										
@BM	001	0082	0055												
@BNE	001	0001	0047	0129	0130	2443									
@BNH	001	0004	0045												
@BNL	001	0002	0046	8645											
@BNM	001	0002	0058												
@BNOL	001	0020	0051	9580											
@BNOZ	001	0008	0050	9580											
@BNP	001	0004	0057												
@BNZ	001	0001	0059												
@BOL	001	00A0	0049												
@BOZ	001	0088	0048												
@BP	001	0084	0054												
@BR	001	0001	0014	4099	4107	4107	4108	4109	4118	4118	4119	4120	4121	4127	4128
				4134	4136	4138	4175	4176	4177	4178	4180	4181	4186	4187	4212
				4213	4244	4253	4254	4258	4267	4267	4268	4273	4273	4279	4280
				4297	4301	4301	4305	4306	4310	4321	4327	4327	4332	4332	4333
				4333	4334	4479	4487	4488	4491	4504	4511	4620	4627	4629	4643
				4645	4645	4652	4653	4656	4667	4668	4790	4805	4807	4821	4827
				4828	4832	4833	4834	4835	4836	4836	4837	4846	4847	4850	4854
				4855	4856	4857	4857	4858	4884	4884	4885	4895	4905	4906	4907
				4909	4909	4916	4916	4917	4926	4928	4936	4936	4939	4981	4984
				4986	4987	5099	5113	5118	5123	5125	5130	5132	5133	5142	5144
				5145	5145	5146	5147	5149	5151	5152	5153	5159	5161	5168	5170
				5175	5175	5176	5180	5181	5190	5192	5193	5201	5202	5203	5208
				5209	5210	5220	5226	5234	5236	5240	5357	5363	5365	5367	5375
				5381	5385	5398	5407	5408	5408	5409	5410	5411	5412	5413	5422
				5423	5424	5424	5425	5426	5431	5432	5439	5440	5440	5441	5441
				5443	5584	5591	5596	5599	5600	5611	5612	5613	5615	5616	5623
				5625	5629	5635	5647	5651	5652	5653	5654	5655	5656	5657	5658
				5669	5685	5686	5687	5688	5689	5690	5693	5694	5699	5700	5701
				5730	5740	5741	5744	5745	5745	5746	5746	5751	5751	5752	5753
				5759	5761	5762	5762	5763	5764	5892	5910	5919	5920	5920	5921
				6035	6042	6049	6051	6058	6058	6059	6186	6192	6193	6194	6196
				6197	6198	6199	6205	6207	6209	6210	6214	6223	6337	6343	6344
				6351	6352	6353	6357	6358	6363	6388	6389	6390	6396	6422	6427
				6427	6428	6429	6436	6436	6437	6439	6446	6447	6448	6449	6450
				6452	6453	6454	6469	6469	6470	6470	6472	6473	6474	6476	6477
				6477	6478	6488	6491	6611	6616	6617	6619	6623	6628	6629	6634
				6638	6640	6641	6642	6642	6643	6644	6648	6649	6650	6655	6657
				6658	6663	6667	6668	6671	6682	6683	6684	6685	6689	6690	6692
				6699	6700	6702	6707	6718	6719	6721	6849	6863	6864	6877	6878
				6879	6882	6883	6893	6913	6914	6919	6921	6930	6949	6951	6970
				6972	6976	6982	6987	7036	7048	7052	7053	7059	7064	7080	7084
				7085	7092	7097	7112	7118	7124	7128	7133	7138	7140	7144	7145
				7150	7152	7156	7157	7162	7300	7364	7472	7593	7595	7596	7602

CROSS REFERENCE

SYMBOL LEN VALUE DEFN REFERENCES VER 15, MOD 00 31/05/21 PAGE 461

	7716	7727	7840	7855	7972	7988	7988	7999	8003	8003	8008	8008
	8009	8009	8013	8013	8018	8018	8019	8023	8023	8027	8027	8031
	8031	8035	8035	8036	8040	8045	8055	8056	8057	8061	8061	8062
	8070	8282	8291	8292	8293	8303	8327	8339	8349	8353	8353	8355
	8356	8364	8365	8371	8377	8379	8379	8380	8386	8391	8398*	8422
	8709	8718	8719	8720	8729	8755	8792	8818	8831	8832	8833	8842
	8864	8881	8894	8903	8916	8926	8947	8964	9189	9205	9209	9218
	9254	9255	9266	9270	9275	9275	9276	9311	9315	9315	9316	9322
	9324	9328	9328	9329	9330	9334	9338	9345	9346	9347	9348	9352
	9355	9362	9362	9363	9366	9369	9373	9374	9376	9380	9382	9398
	9400	9401	9520	9523	9540	9543	9547	9563	9567	9577	9581	9595
	9597	9597	9600	9604	9622	9637	9646	9652	9653	9654	9655	9660
	9661	9662	9667	9669	9676	9683	9684	9687	9689	9689	9690	9690
	9692	9693	9694	9694	9695	9696	9696	9701	9701	9702	9703	9704
	9704	9705	9705	9706	9706	9711	9712	9712	9713	9714	9714	9715
	9716	9720	9723	9748	9749	9751	9764	9765	9769	9769	9770	9776
	9776	9777	9778	9778	9779	9789*	9837	9841	9842	9843	9849	9850
	9862	9863	9869	9870	9871	9876	9876	9877	9880	9881	9881	9882
	9882	9883	9895	9899	9899	9900	9900	9903	9904	9920	9934	9935
	9939	9939	9944	9951	9951	9955	9955	9956	9956	9964	9965	9966
	9967	0092	0094	0118	0120	0125	0130	0138	0139	0152	0165	0172
	0173	0179	0181	0186	0190	0191	0195	0197	0209	0214	0223	0244
	0256	0266	0267	0270	0271	0271	0272	0273	0274	0275	0283	0286
	0288	0288	0289	0289	0291	0291	0293	0294	0294	0295	0295	0297
	0299	0301	0301	0312	0313	0314	0314	0326	0327	0328	0331	0331
	0332	0365	0392	0416	0416	0417	0424	0424	0425	0427	0427	0428
	0439*	0454	0461	0462	0463	0470	0471	0472	0476	0477	0478	0484
	0485	0485	0490	0492	0493	0494	0496	0497	0498	0498	0499	0507
	0512	0518	0519	0524	0527	0577	0583	0584	0590	0600	0618	0643
	0651	0651	0653	0656	0657	0661	0663	0665	0666	0684	0688	0689
	0699	0867	0871	0872	0872	0873	0877	0878	0879	0880	0881	0882
	0883	0884	0885	0885	0886	0887	0888	0889	0890	0893	0895	0899
	0900	0904	0906	0908	0994	0995	0996	0996	0997	1058	1060	1060
	1066	1069	1070	1073	1075	1076	1077	1079	1079	1080	1082	1083
	1096	1097	1098	1098	1099	1100	1104	1105	1105	1106	1106	1107
	1108	1118	1119	1120	1125	1131	1137	1138	1138	1152	1153	1154
	1156	1158	1160	1161	1163	1165	1166	1172	1176	1177	1179	1179
	1180	1180	1182	1192	1192	1195	1196	1196	1197	1206	1209	1211
	1211	1212	1218	1219	1219	1220	1223	1224	1237	1238	1239	1248
	1254	1269	1271	1273	1276	1278	1283	1285	1286	1288	1293	1294
	1295	1297	1299	1299	1301	1302	1302	1303	1303	1305	1306	1306
	1309	1309	1310	1312	1314	1316	1318	1318	1319	1319	1320	1320
	1322	1338	1338	1339	1341	1342	1343	1344	1348	1356	1358	1359
	1363	1364	1417	1433	1443	1443	1445	1448	1448	1450	1450	1452
	1453	1455	1456	1457	1457	1459	1464	1467	1472	1477	1491	1919
	1933	1963	1966	1967*	1986	2010	2052	2066	2075	2077	2081	2083
	2084	2098	2099	2099	2100	2100	2101	2102	2104	2104	2105	2105
	2107	2107	2108	2109	2109	2110	2114	2118	2121	2126	2128	2129
	2135	2137	2138	2139	2140	2145	2148	2149	2150	2156	2179	2183
	2188	2192	2192	2193	2197	2202	2209	2297	2396	2412	2413	2414
	2415	2428	2429	2440	2441	2444	2445	2447	2448	2449	2457	2462
	2463	2464	2479	2483	2484	2486	2500	2516	2541	2550	2550	2557
	2558	2558	2560	2561	2563	2564	2566	2566	2567	2569	2584	2588
	2647	2653	2654	2672	2684	2709	2712	2715	2726	2726	2728	2731
	2731	2732	2733	2734	2735	2739	2739	2741	2741	2743	2743	2759
	2763	2816	2822	2822	2823	2824	2825	2835	2836	2841	2842	2844

CROSS REFERENCE

SYMBOL LEN VALUE DEFN REFERENCES VER 15, MOD 00 31/05/21 PAGE 462

	2844	2845	2846	2858	2858	2859	2860	2865	2867	2868	2887	2888	
	2893	2894	2903	2904	2906	2906	2907	2908	2912	2913	2914	2914	
	2915	2915	2916	2916	2917	2917	2918	2918	2919	2924	2924	2926	
	2926	2936	2936	2938	2938	2959	2963	2967	3023	3035	3037	3047	
	3048	3053	3055	3067*	3082	3083	3083	3084	3098	3099	3101	3101	
	3102	3103	3103	3104	3109	3113	3115	3129	3130	3131	3135	3137	
	3138	3150	3152	3156*	3157	3168	3172	3256	3262	3263	3263	3264	
	3265	3276	3277	3281	3288	3288	3289	3290	3294	3303	3303	3305	
	3307	3308	3308	3310	3311	3317	3318	3322	3324	3337	3338	3343	
	3344	3344	3345	3346	3347	3352	3352	3353	3353	3354	3359	3359	
	3361	3361	3366	3368	3369	3373	3375	3401	3403	3408	3421	3425	
	3636	3654	3661	3662	3663	3664	3669	3689	3713	3726	3728	3984	
	3995	3998	4003	4013	4016	4018	4025	4025	4037	4038	4038	4039	
	4044	4044	4045	4046	4051	4051	4061	4067	4083	4085	4086	4086	
	4089	4091	4091	4098	4101	4115	4116	4116	4124	4125	4134	4227	
	4234	4237	4242	4244	4252	4253	4260	4261	4261	4262	4262	4267	
	4268	4268	4270	4274	4290	4296	4297	4297	4299	4305	4309	4309	
	4311	4312	4312	4318	4319	4320	4324	4324	4325	4326	4328	4328	
	4329	4329	4330	4330	4331	4335	4337	4337	4338	4342	4343	4343	
	4344	4400	4411	4415	4416	4417	4422	4422	4424	4428	4439	4441	
	4443	4452	4465	4467	4467	4469	4469	4470	4471	4473	4473	4481	
	4482	4488	4489	4489	4500	4505	4514	4523	4527	4527	4528	4529	
	4529	4541	4550	4551	4560	4561	4571	4575	4576	4600	4604	4605	
	4610	4617	4685	4691	4695	4699	4700	4701	4706	4706	4708	4712	
	4723	4735	4748	4750	4750	4752	4752	4753	4754	4756	4756	4764	
	4765	4771	4772	4772	4783	4788	4797	4806	4810	4810	4811	4812	
	4812	4824	4833	4834	4843	4844	4854	4858	4860	4874	4878	4879	
	4884	4892	4894	5221	5227	5228	5229	5230	5231	5243	5253	5272	
	5273	5275	5279*	5280	5281	5296	5315	5321	5332	5333	5333	5334	
	5334	5335*	5337	5338*	5350	5352	5382	5390	5459	5470	5471	5473	
	5483	5494	5499	5506	5527	5531	5533	5535	5546	5548	5549	5553	
	5578	5610	5634	5638	5638	5639	5643	5648	5655	5720	5733	5733	
	5735	5745	5746	5748	5748	5754	5759	5759	5763	5771	5773	5777	
	5783	5787	5787	5792	5817	5822	5824	5824	5825	5825	5833	5844	
	5849	5850	5851	5857	5865	5871	5871	5872	5872	5873	5873	5876*	
	5877	5877*	5879	5934	5962	5965	5978	5992	6002	6011	6012	6014	
	6014	6017	6023	6023	6024	6024	6026	6033	6051	6053	6060	6070	
	6071	6071	6075	6082	6089	6091	6092	6092	6127	6132	6143	6145	
	6147	6148	6148	6155	6156	6160	6160	6161	6162	6164	6165	6165	
	6166	6166	6167	6178	6178	6186	6186	6187	6188	6188	6191	6191	
	6194	6194	6199*	6200	6201*	6207	6207	6208	6210	6211	6223	6234	
	6242	6244	6248	6248	6252	6252	6258	6258	6259*	6260	6261*	6265	
	6270	6270	6271	6271	6516	6529	6530	6530	6534	6536	6543	6552	
	6552	6562	6562	6563	6563	6571	6572	6572	6583	6596*	6616	6617	
	6628	6628	6632	6640	6645	6645	6650	6651	6652	6656	6656	6657	
	6661	6862	6872	6881	6890	6905	6910	6910	6914	6914	6918	6922	
	6922	6923	6927	6927	6928	6950	6955	6958	6974	6979	6981	6992	
	7003	7005	7201	7210	7219	7221	7222	7222	7231	7232	7233	7233	
	7244	7269	7269	7274	7274	7279	7289	7289	7302	7302	7303	7307	
	7307	7308	7420	7425	7425	7431	7431	7436	7439	7445	7446	7447	
	7449	7453	7453	7457	7457	7461	7468	7474	7482	7487	7487	7488	
	7492	7492	7493	7591	7603	7604	7604	7605	7605	7606	7606	7607	
	7607	7608	7613	7613	7614	7614	7615	7615	7619	7619	7620	7620	
	7621	7621	7622	7622	7623	7630	7630	7631	7634	7638	7638	7640	
	7648	7648	7649	7650	7650	7651	7657	7657	7658	7662	7662	7663	
	7696	7701	7702	7704	7705	7707	7709	7714	7717	7797	7798	7802	

CROSS REFERENCE

SYMBOL LEN VALUE DEFN REFERENCES VER 15, MOD 00 31/05/21 PAGE 463

				7802	7806	7806	7810	7816	7824	7829	7829	7830	7834	7834	7835
				7915	7920	7920	7922	7934	7934	7940	7940	7941	7941	7943	7947
				7947	7951	7952	7952	7957	7961	7961	7962	7967	7967	7968	7972
				7972	7973	8057	8075	8076	8080	8080	8084	8084	8085	8087	8092
				8092	8093	8097	8097	8098	8177	8186	8187	8189	8190	8192	8197
				8203	8204	8204	8205	8205	8206	8206	8207	8211	8211	8212	8212
				8213	8213	8217	8217	8218	8220	8222	8222	8226	8226	8227	8231
				8232	8349	8353	8355	8357	8369	8371	8372	8375	8377	8377	8378
				8379	8380	8383	8398	8398	8399	8404	8406	8411	8412	8596	8609
				8613	8613	8614	8614	8615	8616	8618	8618	8619	8619	8621	8621
				8622	8623	8623	8624	8628	8629	8634	8634	8635	8639	8645	8646
				8648	8649	8650	8655	8656	8658	8659	8660	8668	8669	8674	8675
				8679	8680	8684	8685	8689	8691	8704	8717	8911	8919	8923	8924
				8925	8931	8932	8968	8971	8971	8972	8973	8973	8990	8991	8995
				8996	9000	9001	9002	9029	9033	9033	9034	9038	9043	9050	9139
				9151	9152	9153	9162	9162	9163	9163	9164	9174	9175	9176	9184
				9184	9185	9185	9186	9196	9196	9197	9198	9205	9205	9206	9206
				9207	9208	9208	9209	9218	9219	9225	9226	9226	9234	9236	9240
				9307	9319	9320	9321	9327	9332	9332	9333	9333	9334	9344	9345
				9346	9352	9357	9357	9358	9358	9359	9369	9370	9370	9371	9376
				9376	9377	9378	9378	9382	9382	9383	9384	9392	9394	9394	9395
				9399	9399	9400	9401	9402	9412	9416	9417	9417	9418	9422	9422
				9423	9424	9432	9432	9433	9434	9435	9439	9445	9454	9455	9507
				9520	9522	9523	9524	9524	9525	9526	9527	9528	9528	9529	9530
				9535	9537	9538	9539	9543	9545	9546	9547	9551	9553	9554	9555
				9567	9571	9572	9573	9573	9574	9575	9577	9577	9578	9578	9579
				9579	9580	9588	9589	9590	9605	9609	9609	9611	9671	9677	9681
				9682	9682	9686	9703	9704	9705	9710	9710	9715	9716	9723	9723
				9724	9725	9726	9727	9738	9738	9740	9742	9743	9744	9748	9749
				9750	9755	9755	9757	9758	9765	9766	9770	9770	9771	9771	9772
				9773	9777	9778	9778	9782	9782	9783	9783	9788	9789	9790	0003
				0013	0019	0024	0025	0025	0032	0033	0033	0034	0047	0067	0069
				0069	0070	0071	0081	0082	0082	0089	0089	0090	0090	0091	0091
				0092	0093	0093	0103	0103	0107	0107	0108	0108	0112	0114	0118
				0120	0130	0130	0131	0131	0135	0136	0143	0143	0144	0300	0304
				0311	0313	0315	0324	0339	0340	0358	0361	0368	0369	0370	0401
				0402	0403	0404	0406	0413	0443	0457	0492	0498	0502	0507	0508
				0519	0520	0521	0521	0522	0522	0523	0523	0524	0524	0525	0525
				0530	0536	0538	0548	0557	0559	0560	0563	0565	0566	0567	0568
				0570	0604	0606	0610	0612	0612	0614	0615	0615	0617	0619	0627
				0628	0637	0638	0639	0639	0641	0645	0646	0649	0657	0659	0659
				0662	0662	0663	0664	0666	0671	0672	0673				
@BT	001	0010	0052												
@BZ	001	0081	0056												
@BZ37B	001	00F2	1073												
@B1	001	0001	0064	4238	4354	4679	4954	5257	5472	5778	6049	6493	6628	6689	7011
				7184	7186	7188	7366	7583	7723	9355	9596	9602	9673	9679	9690
				9704	9705	9706	9712	9714	9877	9950	0159	0167	0185	0347	0457
				2101	2108	2586	2705	2710	2761	2961	3088	3091	3170	3423	4018
				4105	4325	5609	5745	5761	5771	5777	5877	6161	6223	6575	8615
				9574	9605	9715	9757	0538							
@CADDR	001	0002	0143	2658	2685	3502	3503	3504	8426	8447	8448	8449	8994	8995	8996
				8997	8998	8999	9000	9001	9002	9203	9218	9257	9266	9291	9318
				9540	9543	9547	9563	9610	9626	9702	9723	9749	9767	9788	9803
				9981	0231	0283	0379	0381	0382	0383	0385	0438	0519	0885	1029
				1060	1105	1106	1107	1125	1138	1192	1196	1211	1216	1219	1271

CROSS REFERENCE														
SYMBOL	LEN	VALUE	DEFN	REFERENCES								VER 15, MOD 00 31/05/21 PAGE 464		
				1273	1276	1303	1320	1377	1450	1453	1495	1935	1963	1969
				2020	2192	2205	2209	2231	2318	2322	2326	2330	2334	2342
				2447	2457	2558	3745	3746	3747	4134	4158	4406	4417	4691
				4701	4896	5256	5324	5330	5470	5471	5527	5638	5651	5673
				6536	7425	7431	7515	7516	7520	8394	8396	8404	8411	8436
				8437	8438	8634	8639	8720	8734	8927	8934	8936	9004	9033
				9046	9050	9085	9196	9205	9278	9383	9479	0025	0130	0149
				0161	0170	0346	0384	0612						
@CARDL	001	0060	0088	0645	9906	9921*	9922	9922	9922*	0128	0137*			
@CC37B	001	0000	1069											
@CD37B	001	00F0	1087											
@CHARA	001	00C1	0073											
@CHARF	001	00C6	0074											
@CHARR	001	00D9	0075											
@CHARZ	001	00E9	0076											
@CKY01	001	0001	1021											
@CKY02	001	0002	1022											
@CKY03	001	0003	1023											
@CKY04	001	0004	1024											
@CKY05	001	0005	1025											
@CKY06	001	0006	1026											
@CKY07	001	0007	1027											
@CKY08	001	0008	1028											
@CKY09	001	0009	1029											
@CKY10	001	000A	1030											
@CKY11	001	000B	1031											
@CKY12	001	000C	1032											
@CKY13	001	000D	1033											
@CKY14	001	000E	1034											
@CKY15	001	000F	1035											
@CKY16	001	0010	1036											
@CLOFF	001	0010	0095											
@CLON	001	0011	0094											
@CMLON	001	0001	1039											
@CMOFF	001	0000	1038											
@COMMA	001	006B	0067	0146	0176									
@CPLUS	001	004E	0080											
@CP37B	001	0004	1100											
@CRERR	001	0090	1055											
@CRPRY	001	0004	1059											
@CRTDS	001	0092	1052											
@CRTQ	001	0090	1054											
@CURSR	001	0040	1056											
@DADDR	001	0002	0141	8377	8378	8398	8439	0089	0090	0164	0172			
@DBFR1	001	0004	0130											

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 465

SYMBOL	LEN	VALUE	DEFN	REFERENCES
@DCWID	001	0001	0951	
@DCYL	001	0001	0127	0081* 0103*
@DCYMV	001	0001	0942	
@DD2	001	0003	0031	5746 5746* 5751* 6427 6427* 6436* 8291* 8718* 3662* 5228* 5229* 5332* 5333 5333* 5871* 5872* 5873 5873* 6186 6186* 6187* 6188* 6191 6191* 6194 6207* 6208 6223* 6248 6265 6270 6299 0434
@DEFLG	001	0002	0964	
@DERCE	001	0020	0994	
@DERD2	001	0008	0986	
@DEREQ	001	0010	0985	
@DERIN	001	0040	0983	
@DERMA	001	0020	0984	
@DERNR	001	0004	0987	
@DERR	001	0000	0958	
@DERSC	001	0001	0989	
@DERTC	001	0002	0988	
@DFCR	001	0006	0944	
@DFDR	001	0004	0945	
@DGET	001	0001	0135	9820 0426 0549 8428 9002 9115 0019 0047
@DHARD	001	0000	0972	
@DLNCT	001	000F	1058	
@DLNLG	001	0040	1057	4695 8660
@DOLAR	001	005B	0069	
@DOP2	001	0004	0029	9960 1400 2604 0663*
@DPLNG	001	0006	0133	8412
@DPOS	001	0000	0134	
@DPUT	001	0002	0136	9768 9810 0537 8420 8925 0013
@DREAD	001	0001	0948	
@DSAD	001	0002	0128	8377* 8378* 8398* 8404* 0082* 0091* 0093* 0107 0107* 0108 0108* 0114* 0120*
@DSBCY	001	0004	0107	3269
@DSBSY	001	0092	1053	
@DSCS1	001	0000	0108	3270
@DSEEK	001	0000	0947	
@DSIVF	001	0003	0139	
@DSPIN	001	0002	0132	
@DTRSZ	001	0018	0086	0163
@DUNSF	001	0080	0990	
@DVBCY	001	0007	0109	3328
@DVERY	001	0003	0953	
@DVRFY	001	0031	0137	
@DVST1	001	0002	0959	
@DVST2	001	0003	0960	
@DWAIT	001	00FF	0138	
@DWBCY	001	0005	0104	3325
@DWBIT	001	0002	0949	
@DWSIZ	001	00C0	0106	
@DWTB1	001	0003	0105	3326
@DZERO	001	00F0	0065	4133 4172 4251 4489 4636 4866 4927 5112 5117 5129 5366 5380 5387 5407 5663 5900 6191 6484 6698
@D1	001	0002	0027	8293* 8720* 9958 0138* 0172 0173* 0179* 0190* 1353 1464* 3084* 3098 3103* 3182 3264* 3305 3308* 3436 3664* 4039* 4044* 4045 4051 4085* 4086* 4091 4260* 4261* 4262* 4267* 4268* 4274 4296* 4297* 4305 4342* 4343 4343* 4428* 4712* 5230* 5272* 5273* 5275* 5891 6002* 6011* 6012* 6014* 6017 6023 6024* 6194* 6210 6872* 6881* 6890* 8968* 8971* 8973 9151* 9152* 9162* 9163* 9174* 9175* 9184* 9185* 9197* 9208* 9218* 9319*

CROSS REFERENCE																	
SYMBOL	LEN	VALUE	DEFN	REFERENCES											VER 15, MOD 00	31/05/21	PAGE 466
				9333*	9344*	9358*	9572*	9589*	9703*	9723*	9724	9748*	9749*	9770*	9771*		
				9772	0024*	0032*	0033*	0067*	0069*	0070	0082	0131	0143*	0311*	0361*		
				0433	0502*	0665*	0666*										
@EOF	001	001C	0078	9659	0278	0620											
@EOFTC	001	0075	0163														
@EOS	001	001E	0077	3341	0137	0149	0161	1210	5281	5484	5584						
@ER37B	001	00F0	1074														
@FDDBC	001	0000	0196														
@FDE1	001	000C	0201														
@FDFNA	001	000B	0199														
@FDHLN	001	0002	0209														
@FDLNC	001	0002	0194														
@FDNSC	001	0003	0211														
@FDSD	001	0000	0207														
@FLACE	001	0009	0198														
@FLDBC	001	0001	0197														
@FLDIN	001	0012	1046														
@FLENT	001	0004	0202														
@FLFNA	001	0002	0200														
@FLHLN	001	0002	0210														
@FLLNC	001	0002	0195														
@FLNSC	001	0001	0212														
@FLSD	001	0001	0208														
@HCEPK	001	003C	0829														
@HCOPS	001	001C	0836														
@HCOPY	001	081C	0831														
@HCRHE	001	7858	0852														
@HDNRY	001	1008	0817														
@HDRHE	001	7854	0850														
@HDRLN	001	0007	0093	0673													
@HDRV1	001	7840	0842														
@HDRV2	001	7844	0844														
@HDTRD	001	1040	0813														
@HDTRJ	001	1010	0815														
@HERPG	001	087C	0819														
@HFEHT	001	0804	0834														
@HIPLE	001	006C	0826														
@HKBER	001	2040	0809	1252													
@HKBHE	001	7848	0846														
@HLOGE	001	1844	0821														
@HPRER	001	0070	0811	1432													
@HPRHE	001	784C	0848														
@HSTAD	001	0009	0970														
@HSTEN	001	0007	0969														
@HSTPE	001	0006	0968	1433*	1483*												

CROSS REFERENCE

SYMBOL LEN VALUE DEFN REFERENCES VER 15, MOD 00 31/05/21 PAGE 467

@IP37B	001	00C0	1109												
@I1IAR	001	00C0	0021	0900*	1058*										
@KCMDK	001	0020	1020												
@KELOK	001	001B	1019												
@KENAB	001	001E	1017	0585	0662	0901	1009	1127	1130	1949	2013	2214	6664	0616	
@KEXIT	001	001F	1018												
@KEYBD	001	0010	1037	0901	0991	0995	1008	1059	1063	1065	1140	1178	1182		
@KFUNK	001	0010	1040	1028	1073										
@KHARD	001	0011	1045												
@KLEAR	001	000D	1041												
@LINSZ	001	00F4	0085	0647											
@LO37B	001	00F0	1078												
@MAPEN	001	0005	0090												
@MINCR	001	2000	0084												
@MINUS	001	0060	0081												
@NOP	001	0080	0041	3844	4120	5203	6357	6914	8339	8769	8833	9677	0118	0152	0267
				0341	0351	0483	0496	0517	0524	0583	0584	0629	0661	1078	1171
				1189	1249	1326	1328	1352	1475	1960	1965	2410	2415	2428	2445
				2823	2842	2865	2887	2967	3281	3322	3403	6543	6583	6591	6595
				6898	8410	8646	8655	9571	0323	0382	0390	0401	0402	0627	
@NORFL	001	0000	0965												
@NTRDY	001	00A0	1102												
@NUMBR	001	007B	0071												
@OPD2	001	0004	0030	6428*	6469*	6470*	6477	6477*	1079*	4125*					
@OP1	001	0003	0028	4273*	5919*	5920*	5921*	7112*	8422*	8755*	9347*	9380*	9615	9686	9708
				9720*	9751*	9778*	9779*	9841*	9964	0181*	0304	0311	0440	0442	0618*
				0655	0688*	0872*	0873*	1097*	1104*	1106*	1107*	1118*	1137*	1273*	1966*
				2121*	2145	2188*	2192*	2193*	2197	2209	2234	2541*	2584*	2759*	2959*
				3035*	3047*	3113*	3131*	3168*	3373*	3401*	3421*	3661*	4443*	4600*	4617*
				4691*	4874*	4892*	5334*	5578*	5634*	5638*	5639*	5643	5655	6178*	6258*
				6536*	6632	7425*	7431*	8634*	8635*	8639	8689	8704*	8919*	9029*	9033*
				9034*	9038	9050	9196*	9205*	9567*	0305*	0346*	0384*			
@OP2	001	0005	0032	4267	4267*	4327*	4884	4884*	4916*	9710	9764*	9765*	0305		
@OVRUN	001	0004	0995												
@PBUSY	001	00E2	1007	0685	0698	1282	1340	8389							
@PCAR	001	00E6	1004	1322*	1454*										
@PCNT	001	0003	0939												
@PCTRL	001	0000	0150	1188*	1190*	1283	1286	1301*	1310	1316*	1455*	1459*	4647	4926	8756
				0324	0358	0671*									
@PCYL	001	0001	0937												
@PC37B	001	00F2	1094												
@PDAR	001	00E4	1003	1293*	0497*										
@PDATA	001	0003	0152	1020	1105*	1138	1215*	1216*	1293	1303*	1450*	2077*	2138*	2183*	4649
				4928	5687	8758	0612	0662*	0663						
@PD37B	001	0080	1108												
@PERR	001	00E0	1010	0699	1344	1346	1348	0443	0445						
@PFLAG	001	0000	0936												
@PFORM	001	00E1	1008	1342											
@PGCSZ	001	0020	0083	0084											
@PLITE	001	00E2	1009	1343*	1363*										
@PLNGH	001	0004	1000	0610	0610	0610*									
@PMGCK	001	0020	1011	1434											
@PN37B	001	00F0	1093												
@PPLNG	001	0004	0149	2146	2147										
@PRCNT	001	0001	0151	1285*	1294*	1295*	1297*	1299	1299*	1302	1305	1306*	1309*	4648	4927
				5686	8757	0339	0340	0606	0615	0615*	0672*				

CROSS REFERENCE

SYMBOL	LEN	VALUE	DEFN	REFERENCES	VER 15, MOD 00	31/05/21	PAGE 468
@PRETR	001	00C0	0155	2257 2324 2328 2332 2336 2340 4505 4788 5681 8668 9091 0358			
				0645			
@PRINT	001	0040	0153	0155 1283 1440 1460 1484 2264 2271 2316 2320 4415 4699 8679			
				0306 0324 0405 0649 0671			
@PRITY	001	0080	1044	1070			
@PSAD	001	0002	0938				
@PSIOQ	001	00E0	1006	1323 1496			
@PSIOR	001	0000	1005	1323 1497			
@PSNSQ	001	00E2	1012	1425			
@PSR	001	0004	0016				
@PWAIT	001	00FF	0159	9373			
@PLIAR	001	0020	0019	0994* 1066*			
@P2IAR	001	0040	0020				
@Q	001	0001	0025	4108* 4119* 4120* 4127* 4128* 5203* 5209* 5599* 5613* 6049* 6058 6058*			
				6193* 6196* 6205* 6352* 6357* 6617* 6629* 6863* 6864* 6913* 6914* 7595*			
				7596* 8327* 8339* 8768 8831* 8833* 9373* 9400* 9709 9877* 9881 9881*			
				9905 9959 0118* 0130* 0139* 0152* 0267* 0270* 0295* 0492* 0496* 0524*			
				0527* 0583* 0584* 0590* 0661* 0665* 0666* 1072* 1078* 1171* 1191* 1249*			
				1325 1339* 1351 1475* 2101* 2105 2105* 2108* 2412* 2413* 2414* 2415*			
				2428* 2429* 2440* 2441* 2444* 2445* 2516* 2560* 2563* 2569* 2653* 2654*			
				2823* 2824* 2825* 2841* 2842* 2865* 2887* 2893* 2967* 3281* 3294* 3322			
				3403* 3408* 4013* 4016* 4037* 4038* 4115* 4116* 4201 4252* 4253* 4325*			
				4329 4329* 4637 4638 4658 4916 4917 5337 5735* 5746* 5824* 5825*			
				6023* 6070* 6071* 6091* 6092* 6111 6161* 6165 6165* 6248* 6252* 6270*			
				6271* 6290 6543* 6583* 6616* 7210* 7219* 7221* 7231* 7232* 7436* 7439*			
				7701* 7704* 7709* 7714* 7717* 8186* 8189* 8192* 8197* 8353* 8399* 8615*			
				8621 8621* 8645* 8646* 8655* 8656* 9571* 9574* 9579 9579* 9588* 9605*			
				9609 9609* 0304* 0312* 0323* 0369* 0376 0379* 0389 0401* 0402* 0492*			
				0627*			
@RD37B	001	00F1	1088				
@REGL	001	0002	0013	5256 9285 0996 5893 6108 6286 6287 7690			
@RETRN	001	0080	0154	0155 1301 1310 1386 1455 1459 2278 4653 4932 5690 5691 8762			
				9098			
@RLDWN	001	004F	0160				
@RTCNT	001	0003	1002	1312* 1314* 1318*			
@RTRNC	001	0080	0162	2279 4654 4933 8763 9099			
@RT37B	001	0005	1101				
@SBLN	001	0005	0171				
@SBLNL	001	0002	0185				
@SCTSZ	001	0100	0101				
@SDFLN	001	0007	0091				
@SDF0	001	0000	0167	0467			
@SDF1	001	0001	0168	0485* 0493 0498* 0499			
@SDF2	001	0002	0169	0478			
@SDF3	001	0003	0170				
@SECCY	001	0030	0087				
@SIST	001	0001	0182				
@SKCTL	001	0000	0952				
@SLASH	001	0061	0068				
@SLAST	001	0002	0184	0478			
@SMIDL	001	0003	0183				
@SNSB0	001	0000	0976				
@SNSB1	001	0001	0977				
@SNSB2	001	0002	0978				
@SNSB3	001	0003	0979				
@SNULL	001	0080	0174	0467			

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 469

SYMBOL	LEN	VALUE	DEFN	REFERENCES
@SN37B	001	00F2	1082	
@SONLY	001	0000	0181	
@SPINA	001	00A0	0961	
@SPINB	001	00B0	0962	
@STEXT	001	0007	0173	0565
@STYPE	001	0006	0172	0490
@SYCNT	001	0002	1001	1445* 1456 1457* 0614*
@SYLVL	001	0005	2740	
@TBCNT	001	0000	0161	
@TBLEF	001	0010	0156	0158 1286
@TBLIX	001	0011	0158	0404
@TJ37B	001	0040	1099	
@TYPAM	001	0002	1043	1176 1206
@TYPO	001	001C	1042	
@UCB	001	0087	0040	3821 4127 5209 6352 6864 8327 8831 9677 0139 0197 0270 0351
				0492 0527 0590 0604 0650 0665 0666 1072 1191 1244 1339 2412
				2413 2414 2440 2508 2516 2824 2825 2841 2893 3294 3408 6616
				6932 8656 9588 0304 0312 0369 0377 0379 0492
@UPARW	001	005A	0079	2723 0931
@VADDR	001	0002	0142	2659 2685 3062 3498 3510 3511 3512 3512 3526 3529 3531 3555
				3556 3557 3595 3598 3601 3604 3607 3610 3613 3622 3625 3628
				3631 3634 4154 4162 4207 4502 4513 4798 4815 4878 5215 5232
				5640 5665 5697 6371 6383 6395 6675 6713 6869 6926 6966 6992
				7311 7335 8391 8461 8792 8795 8801 8805 8838 8860 8877 8899
				8921 8943 8960 8977 9197 9208 9229 9237 9254 9255 9256 9269
				9270 9275 9276 9297 9315 9316 9352 9374 9397 9398 9407 9408
				9415 9523 9567 9576 9618 9722 9830 9916 9928 9941 0094 0122
				0135 0202 0207 0209 0219 0223 0229 0262 0277 0324 0330 0365
				0462 0465 0510 0522 0599 0623 0653 0663 0691 0904 0906 1279
				1356 1469 1933 1947 1955 1964 1975 1984 2002 2010 2020 2021
				2118 2156 2216 2233 2469 2687 3070 3143 3212 3216 3220 3224
				3228 3232 3654 3657 3689 4059 4140 4441 4586 4619 5253 5294
				5321 5343 5363 5382 5390 5403 5404 5504 5518 5661 5815 6080
				6530 6534 6541 6552 6557 6563 6586 6594 6605 6614 6628 6635
				6640 6661 6675 6676 6692 6914 6945 6955 6969 6979 6992 6998
				7014 7222 7233 7255 7259 7267 7274 7279 7295 7321 7323 7324
				7445 7446 7447 7457 7461 7468 7482 7505 7506 7507 7518 7597
				7604 7613 7614 7620 7621 7622 7630 7631 7634 7638 7650 7651
				7675 7679 7683 7688 7710 7712 7798 7806 7810 7824 7846 7847
				7853 7961 7962 7987 7997 8076 8084 8085 8087 8110 8111 8115
				8198 8204 8205 8211 8212 8217 8218 8220 8222 8239 8241 8242
				8246 8252 8364 8391 8415 8711 8938 8948 8953 9056 9157 9179
				9200 9214 9249 9252 9377 9416 9423 9441 9686 9688 9795 0307
				0344 0362 0380 0392 0416 0460 0621 0648 0653
@VENTA	001	0056	0114	3329 3584 9206 9242 3642 3702
@VMDDV	001	00FE	0115	
@VMFD1	001	0000	0110	
@VMFD2	001	0001	0111	
@VMRS3	001	0002	0113	
@VMTRL	001	0001	0112	
@VOLID	001	0006	0092	
@VQ	001	0001	0026	6437 6453 7598 9707 9878 9957 0303 4119 4255 5827 5879 6026
				6094 6260 6272 0352 0664*
@WA37B	001	00FF	1107	
@WSFIT	001	0500	0102	
@WSTBL	001	0503	0103	

CROSS REFERENCE

SYMBOL LEN VALUE DEFN REFERENCES VER 15, MOD 00 31/05/21 PAGE 470

@XR	001	0002	0015	4129*	4133	4135	4143	4172	4173	4173	4174	4176	4177	4179	4181
				4186	4188	4188	4194	4196	4197	4198	4198	4213	4245	4274*	4288
				4312*	4328	4328*	4485*	4486	4489	4490	4490	4498	4499	4499	4500
				4500	4503	4511	4626*	4627	4628	4636	4637	4637	4642	4644	4644*
				4654*	4655	4664	4665*	4666	4667	4668	4804	4820	4827	4833	4835
				4846	4848	4849	4849	4855	4865	4894	4908	4910	4910*	4925*	4937
				4938	4939	4982*	4983	4985	4987	5103*	5112	5117	5122	5129	5131
				5131	5132	5143	5146*	5149	5150*	5151	5158	5167	5169	5174	5177
				5189	5191	5210	5220	5241	5361*	5362	5366	5374	5379	5379	5380
				5385	5386	5386	5387	5391	5391	5392	5393	5393	5394	5394	5395
				5395	5396	5396	5397	5397	5398	5409	5410	5422	5425	5439	5442
				5442*	5447*	5590*	5595	5597	5597	5598	5606*	5610	5614	5617	5622
				5629	5634	5647	5651	5653	5654	5656	5658	5663	5669	5685	5690
				5691	5695	5734*	5741	5742	5742	5744	5752	5753	5759	5761	5764
				5898*	5899	5900	5901	5901	5919	5922*	5927	5928	5929*	5930	6041*
				6042	6050	6052	6052	6057	6057	6063	6063	6190*	6191	6195	6198
				6204*	6206	6210	6213	6214	6217	6222	6223	6226	6341*	6342	6344
				6345	6350	6358	6359	6361	6362	6363	6365	6376	6384	6389	6390
				6396	6437	6438	6471	6472	6473	6488	6615*	6618	6627*	6633	6638
				6639	6640	6648	6655	6656	6657	6667	6668	6669	6669	6671	6673
				6690	6698	6700	6701	6706	6706	6707	6720	6720	6721	6862*	6877
				6878	6879	6883	6885	6893	6894	6895	6912*	6918	6920	6943*	6947
				6949	6950	6955	6955	6960	6970	6972	6976	6977	6982	7052	7053
				7058	7058	7059	7064	7084	7085	7086	7091	7091	7092	7116	7118
				7122	7122	7124	7128	7133	7138	7140	7144	7145	7150	7152	7156
				7157	7162	7322*	7323	7346*	7347	7360	7360	7364	7365	7366	7479*
				7483	7582*	7583	7588	7593	7594	7596	7598	7598	7602	7607	7608
				7608	7609	7710*	7711	7716	7720	7721	7721	7722	7723	7727	7839*
				7840	7854*	7855	7979*	7998	7999	8040	8041	8046	8050	8055	8056
				8057	8066	8070	8071	8287*	8291	8292*	8293	8297*	8298	8314*	8349
				8356	8365	8371	8377	8386	8427*	8714*	8718	8719*	8720	8724*	8725
				8766*	8792	8800*	8801	8805	8805*	8806	8817*	8855	8893	8915	8925*
				8981*	9196*	9197	9204*	9205*	9206	9208	9228*	9229	9236*	9242	9254
				9255	9256	9258*	9267*	9268	9269	9270	9325*	9326	9354*	9355	9356
				9360	9365	9365*	9380	9381*	9382	9383	9383*	9384	9389	9390*	9391
				9526*	9527	9527*	9528	9530	9532	9536	9541	9545	9552	9556*	9557
				9573	9581	9582	9588*	9589	9594	9596	9596*	9599	9601	9601	9602
				9602*	9603	9605	9609	9614*	9644	9644	9649	9650	9653	9654	9655
				9656	9658*	9659	9660*	9664*	9665	9673	9673	9674	9678	9679	9679
				9680	9680	9684	9685*	9692	9693	9695	9711	9752	9752	9754	9779
				9790*	9796	9797	9841	9842	9843	9845	9848*	9849	9851	9857	9857
				9858	9862	9863	9870	9871	9872	9878	9880	9887	9888	9890	9891
				9920*	9921	9922	9922	9926*	9932*	9933	9949	9950	9950*	9957	9964*
				9965	9966	9967	0097*	0098	0098*	0099	0101	0103	0107	0111	0116
				0119	0120	0125	0128	0137	0143	0146	0149	0156	0159	0159*	0161
				0164	0167	0181	0185	0185*	0195*	0256	0268	0272	0273	0274	0275
				0278	0284	0297*	0299	0310*	0312	0313	0326	0327	0328	0337*	0338
				0346	0347	0347	0348	0348	0355	0356	0356	0406	0408	0410	0410
				0428	0441*	0451	0453	0454	0457	0461	0462	0463	0467	0470*	0471
				0476	0477*	0484	0493	0497	0499	0507	0512*	0518	0519	0600	0605
				0607	0607*	0611	0613	0615	0617	0618	0619*	0620	0624*	0625	0634
				0634	0635	0635	0636	0636	0641	0642	0644	0644	0645	0645	0654*
				0656	0688	0689*	0692*	0868	0880	0881	0882	0886*	0887	0889*	0890
				0891*	0892	0892*	0893	0894	0894*	0895	0899	0908*	1071	1072	1074
				1078	1080*	1081	1099*	1107	1119	1131*	1139	1171	1181	1183	1188
				1190	1191	1194	1194*	1198	1207	1208	1208*	1215	1215	1216	1216

CROSS REFERENCE

SYMBOL LEN VALUE DEFN REFERENCES VER 15, MOD 00 31/05/21 PAGE 471

				1217	1217*	1237*	1246	1249	1280	1324	1346	1350	1369	1418	1425
				1428	1434	1452	1453	1454	1464	1465	1470	1472	1473	1474	1475
				1489	1940*	1941	1942	1942	1953*	1999*	2007	2056	2066*	2075*	2077
				2083*	2120*	2121	2125	2125*	2128*	2133*	2134	2134*	2137*	2138	2139*
				2145*	2146	2146*	2147	2147*	2179*	2183	2202	2456	2456	2457	2458
				2458*	2476	2485	2495	2586	2586*	2587	2658	2669	2671	2673	2684*
				2705	2705*	2706	2708	2710	2710*	2711	2714	2729	2732	2761	2761*
				2762	2850	2882	2889	2891	2898	2903	2907	2961	2961*	2962	2965
				3035	3036	3036*	3041	3047	3049	3051	3056	3088	3088*	3089	3091
				3091*	3092	3102	3108	3113	3114*	3115	3116*	3130	3131	3135*	3136
				3136*	3137	3143	3144*	3149*	3151	3157	3170	3170*	3171	3282	3295
				3307	3315	3334	3345	3373	3374*	3375	3380*	3404	3406	3423	3423*
				3424	3427	3661	3662	3663*	3664	3665*	3681	3684*	3702	3712	3991*
				4014	4017	4023	4030	4032	4037	4042	4065	4077	4088	4100	4101
				4105	4105*	4118	4119	4119	4123	4124*	4126	4234	4236	4237	4241
				4243	4253	4255	4255	4260	4263	4269	4288	4288	4289	4290	4295
				4298	4300	4311	4320	4344	4541*	4575*	4610*	4824*	4858*	4884*	4894
				5227*	5230	5267*	5268	5274	5280	5336*	5337	5484	5531*	5549*	5579*
				5584	5586	5588	5590	5593	5595	5597	5601	5603	5609	5609*	5615
				5648	5731	5739	5744	5753	5758	5761	5761*	5762	5768	5779	5783
				5788	5788*	5793	5822*	5823	5827	5827	5875*	5879	5944*	5945	5946
				5947*	5952	5954	5959	5969	5971	5977	5989*	5991	6011	6026	6031
				6031	6032	6033	6050	6052	6054	6058	6059	6070	6088*	6089*	6090
				6094	6094	6142*	6143*	6145	6154	6164	6177*	6187	6200	6221	6224
				6229	6233*	6234*	6243	6260	6268	6272	6272	6528*	6529	6570*	6571
				6574	6579*	6581	6582	6582	6632*	6639*	6904*	6905	6954*	6978*	6991*
				7243*	7244	7278*	7432*	7437	7437	7440	7440	7445	7446	7447	7449
				7467	7467*	7474*	7603	7623	7624	7628	7628*	7629	7635	7635*	7640*
				7641	7700*	7702	7705	7707	7710	7710	7712	7712	7715	7715	7718
				7718	7720	7720	7789*	7790	7790	7797	7798	7816	7921*	7922	7926
				7926	7942*	7943	7951	7957	8064*	8068	8068	8075	8076	8182*	8187
				8190	8193	8193	8195	8195	8198	8198	8203	8354*	8357	8360	8360
				8364	8364	8367	8367	8379*	8380	8381*	8383	8387	8387	8400*	8402
				8406*	8407	8408*	8409	8410	8411	8412	8628*	8629	8674*	8684*	8717
				8919	8923*	8924	8931*	8932	8957	8958*	8961	8963	8967*	8969	8990*
				8995*	9000*	9001	9010*	9043	9155*	9156	9156*	9157	9177*	9178	9178*
				9179	9199*	9200	9234*	9235	9235*	9236	9250	9253	9325*	9326	9326*
				9327*	9350*	9351	9351*	9352*	9369*	9371	9377	9392	9400	9412*	9416
				9433	9447	9448*	9449	9450	9511	9521*	9522	9526	9534*	9537	9538
				9545	9546	9553	9554	9555	9556	9602	9603*	9604	9604	9606	9608
				9608	9610	9610	9692	9705	9709	9709*	9739	9743	9754	9754*	9765
				9766	9777	9790	0042*	0043	0043*	0053	0055	0061	0068*	0070	0135*
				0136	0301	0305	0305	0307	0309	0312	0313	0323	0344	0346	0351
				0351*	0352	0354*	0362	0363	0367	0379	0380	0384	0388	0392	0393
				0403	0445	0458	0460	0462	0463	0469	0470	0471	0475	0475	0477
				0477	0484	0484	0485	0491	0495	0497	0498	0503	0509	0529	0530
				0539	0546	0547	0555	0559	0560	0562	0563	0564	0564*	0565	0567
				0568*	0580	0605	0610	0623	0646*	0658	0660	0663	0664	0665	0666
				0667	0668	0672	0673	0675	0676						
@ZERO	001	0000	0063	4268	4832	4885	5142	5392	5918	6377	6616	6682	6718	9322	9326
				9338	9356	9384	9599	9603*	9605*	9652	9659*	9687	9773	9782	9869
				9935	9944	9949*	0143	0146	0149	0156	0161	0164	0278	0284	0406
				0408	0420	0431	0453	0611	0686	0697	0888	1108	1285	1297	1341
				1357	1483	1488	2449	2451	2733	2919	3343	3727	3995	4439	4500
				4723	4783	5257	5534	6238	7250	8355	8373	8985	9719	9761	0029
				0366	0502	0519	0569	0606	0623						

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 472

SYMBOL	LEN	VALUE	DEFN	REFERENCES
@4K	001	0010	1061	
B\$ADMK	001	0001	2966	
B\$ADSW	001	159D	2965	
B\$ARMK	001	0001	2951	
B\$ARSW	001	0A45	2950	
B\$BABF	001	1D00	2756	
B\$BCKT	001	1590	2878	
B\$BDPL	001	19E8	2830	
B\$BDSA	001	19EA	2831	
B\$BINO	001	1A6A	2894	
B\$BRLN	001	19F1	2829	
B\$BROP	001	1AF7	2935	
B\$BRVA	001	19EF	2828	
B\$BRVP	001	19EE	2827	
B\$BTAB	001	1996	2826	
B\$CADR	001	1AF9	2936	
B\$CASA	001	0000	2771	
B\$CASC	001	0671	2775	
B\$CASM	001	0608	2773	
B\$CBAS	001	14BB	2901	
B\$CBFA	001	0CBC	2856	
B\$CCGT	001	0600	2781	
B\$CCLS	001	0695	2787	
B\$CCON	001	001F	2854	
B\$CDAT	001	0600	2767	
B\$CDEF	001	0600	2768	
B\$CDIM	001	0673	2769	
B\$CDUM	001	0000	2805	
B\$CEND	001	0600	2803	2804
B\$CEOF	001	0600	2804	
B\$CFOR	001	0600	2776	
B\$CGET	001	06A3	2784	
B\$CGSB	001	0690	2782	
B\$CGTO	001	06B3	2780	
B\$CIFA	001	0600	2778	
B\$CIFC	001	0600	2779	
B\$CIMG	001	0600	2793	
B\$CINP	001	0600	2788	
B\$CLTA	001	0000	2770	
B\$CLTC	001	0669	2774	
B\$CLTM	001	0600	2772	
B\$CMAT	001	0600	2794	
B\$CMGT	001	0665	2795	
B\$CMIN	001	06D3	2796	
B\$CMPR	001	069B	2799	
B\$CMPT	001	069B	2798	
B\$CMPU	001	0600	2800	
B\$CMRD	001	06D0	2797	
B\$CNXT	001	0600	2777	
B\$CPCT	001	0CA8	2859	
B\$CPRT	001	0600	2791	
B\$CPRU	001	0600	2792	
B\$CPSE	001	06E7	2801	
B\$CPUT	001	0600	2785	
B\$CPWA	001	0CA6	2930	
B\$CRAD	001	150D	2900	

CROSS REFERENCE																			
SYMBOL	LEN	VALUE	DEFN	REFERENCES													VER 15, MOD 00	31/05/21	PAGE 473
B\$CRBS	001	1509	2902																
B\$CREA	001	06CF	2789																
B\$CREM	001	0000	2766																
B\$CRMK	001	0001	2978																
B\$CRSR	001	06E3	2790																
B\$CRST	001	06A6	2786																
B\$CRSW	001	0E42	2977																
B\$CRTN	001	06CF	2783																
B\$CSBF	001	0600	2753	2767	2768	2769	2772	2773	2774	2775	2776	2777	2778	2779	2780				
				2781	2782	2783	2784	2785	2786	2787	2788	2789	2790	2791	2792				
				2793	2794	2795	2796	2797	2798	2799	2800	2801	2802	2803	2806				
				2807	2808	2809	2810												
B\$CSCN	001	14B0	2875																
B\$CSMK	001	0007	2981																
B\$CSSW	001	14BC	2980																
B\$CSTP	001	06D6	2802																
B\$CSTR	001	14CC	2899																
B\$CSXA	001	2000	2759																
B\$CTYP	001	0A5F	2853																
B\$CVPD	001	0C5D	2858																
B\$CVPG	001	0CA5	2857																
B\$CWRK	001	F500	2927																
B\$DIST	001	0700	2819																
B\$DLNK	001	1B37	2925																
B\$DL4T	001	1A6B	2896																
B\$DPWA	001	0E46	2931																
B\$DST2	001	073A	2820																
B\$ERMK	001	0007	2954																
B\$ERSW	001	0993	2953																
B\$FACA	001	0E53	2862																
B\$FAIS	001	15AC	2879																
B\$FAIW	001	15A0	2880																
B\$FCON	001	0A46	2852																
B\$FORT	001	1B0E	2921																
B\$FPWA	001	15AC	2932																
B\$FRMK	001	0007	2972																
B\$FRSW	001	16CC	2971																
B\$FSC1	001	0E4C	2863																
B\$FSC2	001	0E4D	2864																
B\$FSMK	001	0007	2963																
B\$FSSW	001	0E5C	2962																
B\$FSVA	001	0E4F	2865																
B\$FTND	001	1B0B	2923																
B\$FTPT	001	1B0D	2922																
B\$FVME	001	15A2	2884																
B\$FVMP	001	15A4	2885																
B\$FVMS	001	15A6	2886																
B\$FVPE	001	15A8	2881																
B\$FVPP	001	15AA	2882																
B\$FVPS	001	15AC	2883																
B\$GBSW	001	08AF	2956																
B\$GBWK	001	0001	2957																
B\$GETC	001	0867	2833																
B\$GPTR	001	0878	2835																

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 474

SYMBOL	LEN	VALUE	DEFN	REFERENCES
B\$IFSW	001	16E5	2974	
B\$INVT	001	1B38	2915	
B\$KWMK	001	0001	2969	
B\$KWSW	001	159E	2968	
B\$LBAS	001	185E	2906	
B\$LBSV	001	18E7	2904	
B\$LDRP	001	1A00	2754	
B\$LINE	001	07D0	2821	
B\$LIST	001	1853	2888	
B\$LRTN	001	18EB	2905	
B\$LSTR	001	1862	2903	
B\$LTYP	001	18F2	2889	
B\$MATR	001	18F3	2891	
B\$MBMK	001	0007	2990	
B\$MBSW	001	1903	2989	
B\$MFBK	001	1B8F	2917	
B\$MGMK	001	0007	2987	
B\$MGSW	001	18FF	2986	
B\$MPMK	001	0007	2993	
B\$MPSW	001	1981	2992	
B\$MRMK	001	0007	2984	
B\$MRSW	001	0DDE	2983	
B\$NUMC	001	0873	2834	
B\$NXMK	001	0007	2960	
B\$NXSW	001	071D	2959	
B\$PARP	001	0A41	2842	
B\$PBNL	001	0A01	2848	
B\$PCAD	001	0A40	2843	
B\$PCDL	001	09D3	2847	
B\$PCPG	001	0A35	2846	
B\$PECT	001	0A44	2850	
B\$PERC	001	0A39	2849	
B\$PFAE	001	0033	2840	
B\$PFCL	001	009D	2841	
B\$PFNC	001	094E	2838	
B\$PFWP	001	0015	2839	
B\$PNBY	001	0A41	2844	
B\$PPWA	001	0A35	2929	
B\$PRM1	001	1AF3	2933	
B\$PTBF	001	1F00	2758	
B\$PUTC	001	093A	2837	
B\$PVAD	001	0A43	2845	
B\$RMRK	001	1AE6	2898	
B\$RTRN	001	1AF5	2934	
B\$SABF	001	1C00	2755	
B\$SCAN	001	1514	2877	
B\$SCAT	001	13C8	2872	
B\$SCON	001	001B	2855	
B\$SCVT	001	12E0	2870	
B\$SDPL	001	07DA	2823	
B\$SFAB	001	0E48	2867	
B\$SFNT	001	143C	2873	
B\$SLDT	001	109C	2869	
B\$SLVT	001	1062	2868	
B\$SNAT	001	131A	2871	
B\$SPAT	001	07E0	2824	

CROSS REFERENCE																
SYMBOL	LEN	VALUE	DEFN	REFERENCES	VER 15, MOD 00 31/05/21 PAGE 475											
B\$SSTA	001	1BAC	2919													
B\$STAS	001	061B	2808													
B\$STIF	001	0606	2810													
B\$STMA	001	061B	2809													
B\$STML	001	0600	2807													
B\$STRL	001	0600	2806													
B\$SVRB	001	0E46	2866													
B\$SYMB	001	0DBC	2861													
B\$TCD2	001	0001	2939													
B\$TLTH	001	0002	2940	2941												
B\$TOD1	001	0000	2938													
B\$TOTB	001	1AF8	2941													
B\$TTAB	001	1AFA	2937	2941												
B\$TYPE	001	0739	2822													
B\$WORK	001	15A0	2926													
B\$ZDBN	001	19F2	2893													
B@ABAS	001	0007	3526	6690	7026	7336	7445	7446	7447	7603	7710	7710	7712	7712	7798	
				7943	8075	8076	8198	8198	8203	8364	8364	9377				
B@ACD1	001	0001	3523	3524	6687	7023	7333	7702	7705	7707	7715	7718	7718	7922	7926	
				8187	8190	8193	8195	8357	8360	8383	8387	9371	9400			
B@ACD2	001	0003	3524	3525	8371*	8377	6688	7024	7334	7437	7437	7440	7440	7449	7715	
				7720	7720	7790	7790	7926	8068	8068	8193	8195	8360	8367	8367	
				8387	8402	9392										
B@AFLG	001	0000	3518													
B@ALLA	001	005C	3343													
B@AMAX	001	0005	3525	3526	8386	6689	7025	7335								
B@BLNK	001	0040	3352	9887	9921	1941	2587	2714	2762	2962	3108	3171	3190	3424	4013	
				4088	4118	5735	5823	5954	5959	6581	8942	8959	8969	9153	9530	
				9681	9740											
B@BLSZ	001	0100	3477	3616	3619	3622	3637	3640	9576	9618	9722	0219	0262	0324	0559	
B@BREQ	001	0084	3132													
B@BRHI	001	0088	3133													
B@BRLO	001	0082	3131													
B@BRNE	001	0094	3135													
B@BRNH	001	0098	3136													
B@BRNL	001	0092	3134													
B@CADD	001	0006	3001													
B@CADF	001	0058	3042													
B@CBAS	001	0003	3529	9416												
B@CBNX	001	004A	3035													
B@CBRA	001	0046	3033													
B@CBRC	001	0044	3032													
B@CBRD	001	0048	3034													
B@CBRS	001	004C	3036													
B@CCLS	001	005E	3045													
B@CCMC	001	0042	3031													
B@CCMF	001	0040	3030													
B@CCNT	001	001F	3455	4100	5268											
B@CCSA	001	003E	3029													
B@CDCA	001	006A	3051													
B@CDDL	001	006C	3052													
B@CDIV	001	000C	3004													
B@CDMN	001	0001	3528	3529	9433											
B@CDWA	001	006E	3053													

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 476

SYMBOL	LEN	VALUE	DEFN	REFERENCES
B@CFCI	001	0016	3009	
B@CFN0	001	0012	3007	
B@CFN1	001	0014	3008	
B@CFOR	001	004E	3037	
B@CGET	001	0052	3039	8719
B@CHAR	001	0000	3468	2007 2476 2485 2495 2587 2658 2669 2671 2673 2706 2708 2711 2714 2729 2732 2762 2850 2882 2889 2891 2898 2903 2907 2962 2965 3041 3049 3051 3056 3089 3092 3102 3108 3130 3151 3171 3282 3295 3307 3315 3334 3345 3404 3406 3424 3427 5484 5584 5586 5588 5590 5593 5595 5597 5601 5603 5739 5753 5758* 5762 5768 5952 5954* 5959 5969 5971* 5977 6243*
B@CHLT	001	0004	3000	
B@CIEX	001	00C5	3428	3219 3231
B@CIMH	001	0066	3049	
B@CINI	001	0056	3041	
B@CIP1	001	00D7	3431	3215 3227
B@CIS2	001	00E2	3434	3211 3223
B@CMF1	001	0018	3010	
B@CMF2	001	001A	3011	
B@CMF3	001	001C	3012	
B@CMA	001	006B	3363	9949 2476 9525 9551
B@CMPY	001	000A	3003	
B@CMSM	001	001E	3013	
B@CNEG	001	0010	3006	
B@CNXT	001	0050	3038	
B@COLN	001	007A	3365	
B@CPMK	001	00FF	3273	3277 3281 3282 3316
B@CPRS	001	0060	3046	
B@CPRU	001	0062	3047	
B@CPUT	001	0054	3040	
B@CPWR	001	000E	3005	
B@CRSR	001	005A	3043	
B@CRST	001	005C	3044	
B@CSA1	001	0036	3025	
B@CSA2	001	0038	3026	
B@CSB1	001	003A	3027	
B@CSC1	001	002A	3019	
B@CSD0	001	002E	3021	8292
B@CSD1	001	0030	3022	
B@CSD2	001	0032	3023	
B@CSF1	001	0022	3015	
B@CSF2	001	0024	3016	
B@CSTA	001	0034	3024	
B@CSTC	001	0028	3018	
B@CSTF	001	0020	3014	
B@CSTH	001	0064	3048	
B@CSTX	001	003C	3028	
B@CSUB	001	0008	3002	
B@CSVC	001	0002	2999	
B@CTYP	001	0020	3453	4077
B@CUSC	001	002C	3020	
B@CUSF	001	0026	3017	
B@CVAR	001	005B	3342	9176 9346 9445
B@DAMK	001	0080	3521	
B@DASA	001	00FF	3282	
B@DASC	001	0040	3286	

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 477

SYMBOL	LEN	VALUE	DEFN	REFERENCES
B@DASM	001	0038	3284	
B@DCGT	001	0050	3292	
B@DCLS	001	0054	3298	
B@DDAT	001	0024	3278	
B@DDEF	001	0034	3279	
B@DDIM	001	0004	3280	
B@DDUM	001	00FF	3316	
B@DEC0	001	00F0	3411	5612 7323 7347 7594 7607 7711 7720 8046 8050 2898 2965 3295 3366 3427 4023 4042 4269 4298 4335 5746 5991 6224 6268 9535 9543
B@DEC1	001	00F1	3412	4254 4868 4935 5411 6378 7365 7722 4236 6032
B@DEC2	001	00F2	3413	5412 5591 6977 7086
B@DEC3	001	00F3	3414	
B@DEC4	001	00F4	3415	5616
B@DEC5	001	00F5	3416	6895
B@DEC6	001	00F6	3417	
B@DEC7	001	00F7	3418	
B@DEC8	001	00F8	3419	
B@DEC9	001	00F9	3420	5413 5431
B@DEND	001	0058	3314	3315
B@DEOF	001	0058	3315	
B@DFOR	001	0028	3287	
B@DGET	001	0040	3295	
B@DGSB	001	0020	3293	
B@DGTO	001	0044	3291	
B@DIFA	001	0048	3289	
B@DIFC	001	004C	3290	
B@DIGS	001	007B	3345	5586 5593 5597 5603 5753 5762
B@DIMG	001	003C	3304	
B@DINP	001	0000	3299	
B@DIVD	001	0061	3362	
B@DLTA	001	00FF	3281	
B@DLTC	001	0040	3285	
B@DLTM	001	0038	3283	
B@DL01	001	0001	3596	3599
B@DL02	001	0003	3599	3602
B@DL03	001	0005	3602	3605
B@DL04	001	0007	3605	3608
B@DL05	001	0009	3608	3611
B@DL06	001	000B	3611	3614 9854
B@DL07	001	0045	3614	3617
B@DL08	001	0145	3617	3620
B@DL09	001	0245	3620	3623
B@DL10	001	0289	3623	3626 9855
B@DL11	001	02C3	3626	3629 9856
B@DL12	001	02FD	3629	3632 9857
B@DL13	001	0337	3632	3635
B@DL14	001	0371	3635	3638
B@DL15	001	0471	3638	3641
B@DL16	001	0507	3641	9479
B@DMAT	001	0008	3305	
B@DMGT	001	0044	3306	
B@DMIN	001	0038	3307	
B@DMPR	001	0048	3310	
B@DMPT	001	004C	3309	
B@DMPU	001	0054	3311	

CROSS REFERENCE																			
SYMBOL	LEN	VALUE	DEFN	REFERENCES													VER 15, MOD 00	31/05/21	PAGE 478
B@DMRD	001	003C	3308																
B@DNXT	001	0044	3288																
B@DPNT	001	004B	3353	9858	2850	3282	3315	4263	4289	5595	5601	5739	5768	5977	9744				
B@DPRT	001	002C	3302																
B@DPRU	001	0030	3303																
B@DPSE	001	0050	3312																
B@DPUT	001	0040	3296																
B@DREA	001	000C	3300																
B@DREM	001	00FF	3277																
B@DRSR	001	005C	3301																
B@DRST	001	0050	3297																
B@DRTN	001	005C	3294																
B@DSCY	001	0004	3269																
B@DSIF	001	001C	3318																
B@DSLT	001	0010	3317																
B@DSML	001	0010	3319																
B@DSNS	001	0018	3271																
B@DSS1	001	0000	3270																
B@DSTP	001	0054	3313																
B@DTBN	001	0010	3335																
B@DTB1	001	0050	3334																
B@DTCY	001	0009	3331																
B@DTSN	001	0010	3333																
B@DTS1	001	0040	3332																
B@DTYP	001	0040	3447	8806	9557	9589	9665	0284	0338	3082	9692								
B@DVCY	001	0007	3328	0081															
B@DVC1	001	0056	3329																
B@DWCY	001	0005	3325																
B@DWT1	001	0003	3326																
B@D1MK	001	0080	3519																
B@D2MK	001	00C0	3520																
B@EOST	001	001E	3341	2007	2485	2495	2706												
B@EQUL	001	007E	3367	9154	9220														
B@EXPC	001	00C5	3344	2882	3334														
B@FOFL	001	005C	3346	6090															
B@FVAD	001	0001	3531																
B@GETC	001	0001	3470																
B@GETE	001	00FF	3471																
B@GETS	001	0000	3469																
B@GRTR	001	006E	3364																
B@ICON	001	0050	3426	2673	3056														
B@LADD	001	0001	3070																
B@LADF	001	0002	3111																
B@LADV	001	0008	3555	3576	8299	8392	8397	6529	6529	6685	6905	6905	7021	7244	7244				
				7331	7437	7437	7440	7446	7447	7603	7603	7603	7705	7707	7707				
				7710	7712	7712	7715	7715	7715	7718	7720	7720	7790	7797	7797				
				7798	7943	8068	8075	8075	8190	8193	8195	8198	8203	8203	8357				
				8360	8360	8364	8367												
B@LBIN	001	0002	3480	3481	3487	9370	9371	9378	9382	9392	9394	9399	9400	9422	9432				
				9433	9522	9526	9604	9608	9608	9610	9610								
B@LBNX	001	0003	3104																
B@LBRA	001	0003	3102	9287															
B@LBRC	001	0004	3101																
B@LBRD	001	0003	3103																

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 479

SYMBOL	LEN	VALUE	DEFN	REFERENCES
B@LCCC	001	0001	3063	3101
B@LCDV	001	0004	3556	3577
B@LCER	001	0001	3061	3125
B@LCFN	001	0004	3512	
B@LCLN	001	0002	3066	3117 3118 3125
B@LCLS	001	0001	3114	
B@LCMC	001	0001	3100	
B@LCMF	001	0001	3099	
B@LCNA	001	0006	3510	
B@LCNN	001	0001	3064	3089 3098 3110 3122 2448 2566 2605
B@LCOP	001	0001	3060	3068 3069 3070 3071 3072 3073 3074 3075 3076 3077 3078 3079 3080 3081 3082 3083 3084 3085 3086 3087 3088 3089 3090 3091 3092 3093 3094 3095 3096 3097 3098 3099 3100 3101 3102 3103 3104 3105 3106 3107 3108 3109 3110 3111 3112 3113 3114 3115 3116 3117 3118 3119 3120 3121 3122 3123 8291 8718 3662
B@LCRV	001	0013	3554	3574
B@LCSA	001	0002	3098	
B@LCVA	001	0002	3062	3076 3077 3078 3079 3080 3081 3082 3083 3084 3085 3087 3088 3090 3091 3092 3093 3094 3095 3096 3101 3102 3103 3104 3106 3107 3108 3120 3121 8298 8349 3681 3712
B@LCXX	001	0001	3065	3097 3109 3111 3115 3116 2550 2557 2607
B@LDAT	001	0004	3224	
B@LDCA	001	0003	3120	3737
B@LDDL	001	0003	3121	
B@LDDM	001	0004	3484	
B@LDEF	001	0003	3225	
B@LDIM	001	0003	3226	
B@LDIN	001	0004	3483	3484 3485
B@LDIV	001	0001	3073	
B@LDMN	001	0002	3481	3510 3511 3523 3524 3525 3528 3555 3556 8353 8356 8365 8371 8377 8379 8386 8462 8463 6562 6571 6572 6645 6656 6678 6910 6922 6927 7013 7269 7289 7302 7307 7320 7437 7440 7449 7453 7487 7492 7509 7510 7511 7522 7605 7606 7607 7615 7619 7648 7657 7662 7672 7673 7674 7676 7677 7678 7680 7681 7682 7687 7702 7705 7707 7715 7718 7720 7725 7790 7802 7829 7834 7844 7845 7849 7854 7922 7926 7947 7952 7967 7972 7988 7994 7995 7996 8068 8080 8092 8097 8107 8108 8109 8116 8187 8190 8193 8195 8206 8213 8226 8231 8240 8243 8244 8245 8250 8357 8360 8367 8383 8387
B@LDSN	001	0004	3485	2099 2100 2104 2104 2109 2109 2249 2252 2265 8613 8614 8618 8618 8619 8619 8648 8658 8669 8680 8742 8746
B@LDWA	001	0002	3122	9287
B@LELP	001	0010	3553	
B@LEND	001	0003	3253	
B@LEOF	001	0001	3123	
B@LEOP	001	0001	3119	
B@LERC	001	0003	3125	
B@LESP	001	0008	3552	
B@LESS	001	004C	3354	
B@LET\$	001	005B	3374	
B@LET#	001	007B	3375	
B@LET@	001	007C	3376	
B@LETA	001	00C1	3378	2729 3151
B@LETB	001	00C2	3380	
B@LETC	001	00C3	3381	
B@LETD	001	00C4	3382	

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 480

SYMBOL	LEN	VALUE	DEFN	REFERENCES
B@LETE	001	00C5	3383	
B@LETF	001	00C6	3384	
B@LETG	001	00C7	3385	
B@LETH	001	00C8	3386	
B@LETI	001	00C9	3387	
B@LETJ	001	00D1	3388	
B@LETK	001	00D2	3389	
B@LETL	001	00D3	3390	
B@LETM	001	00D4	3391	
B@LETN	001	00D5	3392	
B@LETO	001	00D6	3393	
B@LETP	001	00D7	3394	
B@LETQ	001	00D8	3395	
B@LETR	001	00D9	3396	
B@LETS	001	00E2	3397	
B@LETT	001	00E3	3398	
B@LETU	001	00E4	3399	
B@LETV	001	00E5	3400	
B@LETW	001	00E6	3401	
B@LETX	001	00E7	3402	
B@LETY	001	00E8	3403	
B@LETZ	001	00E9	3404	
B@LEXP	001	0008	3443	
B@LFCI	001	0003	3078	9288
B@LFNA	001	0002	3557	3578
B@LFN0	001	0003	3076	
B@LFN1	001	0003	3077	
B@LFOR	001	0003	3106	
B@LFRT	001	0004	3498	3499
B@LGET	001	0003	3108	
B@LGSB	001	0005	3232	
B@LGTO	001	0004	3231	
B@LHLT	001	0001	3069	
B@LIEX	001	0002	3429	
B@LIFN	001	0003	3492	
B@LILP	001	0009	3551	3569 3570 3571
B@LIMG	001	0001	3243	
B@LIMH	001	0003	3118	
B@LINI	001	0002	3110	
B@LINP	001	0005	3238	
B@LIPI	001	0003	3432	
B@LISP	001	0005	3550	3558 3564 3565 3566
B@LIS2	001	0005	3435	
B@LIVT	001	0001	3508	
B@LKCL	001	0005	3237	
B@LKFR	001	0003	3228	
B@LKGT	001	0003	3234	
B@LKIF	001	0002	3230	
B@LKON	001	0002	3263	
B@LKPT	001	0003	3235	
B@LKPU	001	000A	3242	
B@LKRR	001	0007	3240	
B@LKRT	001	0005	3236	
B@LKTO	001	0002	3257	
B@LLET	001	0003	3227	
B@LL01	001	0002	3595	3596

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 481

SYMBOL	LEN	VALUE	DEFN	REFERENCES
B@LL02	001	0002	3598	3599
B@LL03	001	0002	3601	3602
B@LL04	001	0002	3604	3605
B@LL05	001	0002	3607	3608
B@LL06	001	0002	3610	3611
B@LL07	001	003A	3613	3614
B@LL08	001	0100	3616	3617
B@LL09	001	0100	3619	3620
B@LL10	001	0044	3622	3623
B@LL11	001	003A	3625	3626
B@LL12	001	003A	3628	3629
B@LL13	001	003A	3631	3632
B@LL14	001	003A	3634	3635
B@LL15	001	0100	3637	3638
B@LL16	001	0096	3640	3641
B@LMAT	001	0003	3244	
B@LMF1	001	0003	3079	
B@LMF2	001	0003	3080	
B@LMF3	001	0003	3081	
B@LMGT	001	0006	3245	
B@LMIN	001	0008	3246	
B@LMPR	001	0008	3249	
B@LMPT	001	0006	3248	
B@LMPU	001	000D	3250	
B@LMPY	001	0001	3072	
B@LMRD	001	0007	3247	
B@LMSM	001	0003	3082	
B@LNEG	001	0001	3075	
B@LNEX	001	0004	3229	
B@LNXT	001	0003	3107	
B@LPAR	001	004D	3355	9321 9520
B@LPRS	001	0002	3115	
B@LPRT	001	0005	3241	
B@LPRU	001	0002	3116	
B@LPSE	001	0005	3251	
B@LPUT	001	0002	3109	
B@LPWR	001	0001	3074	
B@LREA	001	0004	3239	
B@LREM	001	0003	3223	
B@LRSR	001	0001	3112	
B@LRST	001	0001	3113	
B@LRTN	001	0006	3233	
B@LSA1	001	0003	3094	
B@LSA2	001	0003	3095	
B@LSB1	001	0003	3096	
B@LSC1	001	0003	3088	
B@LSDF	001	0004	3478	
B@LSD0	001	0003	3090	
B@LSD1	001	0003	3091	
B@LSD2	001	0003	3092	
B@LSF1	001	0003	3084	
B@LSF2	001	0003	3085	
B@LSKW	001	0002	3494	
B@LSNO	001	0002	3487	2098 2248 8609 8741
B@LSPT	001	0003	3502	3505
B@LSTA	001	0003	3093	

VER 15, MOD 00 31/05/21 PAGE 482

B@LSTC	001	0003	3087													
B@LSTE	001	0004	3258													
B@LSTF	001	0003	3083													
B@LSTH	001	0003	3117													
B@LSTP	001	0004	3252													
B@LSTX	001	0002	3097													
B@LSUB	001	0001	3071													
B@LSVC	001	0001	3068													
B@LTHN	001	0004	3259													
B@LTYP	001	0001	3488													
B@LUFN	001	0002	3495													
B@LUSC	001	0002	3089													
B@LUSF	001	0001	3086													
B@LVPG	001	0100	3582	3585	4098	4243	4377	4789	4980	4992	4993	5098	5356	5583	5729	
				5797	6185	6236	6237	6238	6239	6336	6421	6507	6508	6610	8281	
				8708	9188	9309	9518	9635	9746	9830	9835	0090	0242	0390	0576	
				1941*	1942	1942	1942*	3635	5220	5458	5719	5933	6126	6581*	6582	
				6582	6582*	6861	7200	7419	7590	7597	7695	7784	7914	8176	9253	
B@MINS	001	0060	3361	9872	9890	2671	2891	3051	3053	3222	3226	3230	3406	4016	4318	
				5590	5952	5971	6154	6229	6243	9742	9788					
B@MULT	001	005C	3358													
B@NAAR	001	001D	3546	3576	3628											
B@NCAR	001	001D	3547	3577	3631											
B@NCRV	001	001D	3545	3574	3625											
B@NDGT	001	000A	3538	3544												
B@NEQL	001	007F	3368													
B@NFRT	001	000A	3497	3499												
B@NICN	001	0006	3540	3542												
B@NIEL	001	0007	3542	3558	3564	3569										
B@NIFN	001	0018	3491													
B@NIVR	001	0001	3541	3542												
B@NIVT	001	0057	3507													
B@NLDV	001	0122	3544	3566	3571	3622										
B@NLRV	001	001D	3543	3565	3570	3613										
B@NLTR	001	001D	3537	3543	3544	3545	3546	3547	3548							
B@NSKW	001	0004	3493													
B@NSPT	001	0028	3501													
B@NUFN	001	001D	3548	3578	3634											
B@NVPG	001	0100	3581	3585												
B@NXHI	001	00E3	3462	2981												
B@NXLO	001	001E	3461	4655	4848	5598	7609	8066	2982	3368	7624	7983				
B@NXZR	001	0080	3460	3461	3462	4233	4235	4486	4498	4518	4629	4652	4804	4938	4985	
				5152	5167	5201	5258	5595	5610	5622	5634	5717	5721	5723	5782	
				5784	5787	5789	5792	5909	5939	6195	6206	6342	6350	6415	6618	
				6633	6730	6732	6894	6918	6920	6947	6960	7011	7116	7184	7186	
				7188	7190	7366	7583	7588	7595	7723	7870	8041	8071	9979	2980	
				3262	4030	4032	4154	4241	4243	4252	4319	4360	5990	6052	6058	
				6155	6221	7986	9789	9817								
B@PLUS	001	004E	3356	2669	2889	3048	3049	3210	3214	3218	3404	5588	5969			
B@POWR	001	005A	3357													
B@PREC	001	0020	3449	9674	9678	0346	0355									
B@PROD	001	0023	3558													
B@PRPL	001	0002	3145	9560	7219											
B@PRPN	001	0001	3144													
B@PRPR	001	0004	3147	4156	7221											
B@PRPS	001	0003	3146	7210												

CROSS REFERENCE																
SYMBOL	LEN	VALUE	DEFN	REFERENCES				VER 15, MOD 00 31/05/21 PAGE 483								
B@PRRC	001	0007	3150	9554	7253	7257	7264									
B@PRRL	001	0008	3151	4155	4156											
B@PRSL	001	0005	3148	8893	3997											
B@PRSS	001	0006	3149													
B@PTAB	001	0000	3503													
B@PTAD	001	0001	3504													
B@PTSA	001	0002	3505													
B@PUD1	001	0006	3161	7231	7232											
B@PUD2	001	0007	3162													
B@PUI0	001	0001	3155													
B@PUI1	001	0004	3156													
B@PUI2	001	0005	3157													
B@PUNL	001	0002	3159													
B@PUNS	001	0003	3160	5806	5831											
B@PURE	001	0020	3165	5229	5374	7232										
B@PUTM	001	0010	3164	5229	5373	5522	7232									
B@RPAR	001	005D	3359	9529												
B@SADV	001	00E8	3576	3579												
B@SAVL	001	0B76	3572	3589												
B@SAVS	001	065E	3567	3588												
B@SCDV	001	0074	3577	3579												
B@SCLN	001	005E	3360													
B@SCRV	001	0227	3574	3588	3589											
B@SDMK	001	0080	3489	0490												
B@SEXP	001	0004	3442													
B@SFAT	001	0196	3579	3588	3589	3640										
B@SFNA	001	003A	3578	3579												
B@SFRT	001	0028	3499													
B@SIEL	001	003F	3569	3572												
B@SIES	001	0023	3564	3567												
B@SIGN	001	0010	3451	9739												
B@SLDL	001	0A32	3571	3572												
B@SLDS	001	05AA	3566	3567												
B@SLVL	001	0105	3570	3572												
B@SLVS	001	0091	3565	3567												
B@SQUO	001	007D	3366	9594	9599	9603	9605	9851	0156	0164	0167	2658	2708	2711	3041	
				3089	3092	9726	9727									
B@STAT	001	0000	3441													
B@TASA	001	0012	3176													
B@TASC	001	001E	3182													
B@TASM	001	0018	3178													
B@TASS	001	007B	3183													
B@TCGT	001	0030	3191													
B@TCLS	001	0042	3197													
B@TDAT	001	0006	3172													
B@TDEF	001	0009	3173													
B@TDIM	001	000C	3174													
B@TDUM	001	0078	3215													
B@TEND	001	0072	3213													
B@TEOF	001	0075	3214													
B@TFOR	001	0021	3185													
B@TGET	001	0039	3194													
B@TGSB	001	0033	3192													
B@TGTO	001	002D	3190													

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 484

SYMBOL	LEN	VALUE	DEFN	REFERENCES
B@TIFS	001	007D	3189	
B@TIMG	001	0054	3203	
B@TINP	001	0045	3198	
B@TLTA	001	000F	3175	
B@TLTC	001	001B	3179	
B@TLTM	001	0015	3177	
B@TLTS	001	0079	3180	
B@TMAS	001	007C	3184	
B@TMAT	001	0057	3204	
B@TMGT	001	005A	3205	
B@TMIN	001	005D	3206	
B@TMLS	001	007A	3181	
B@TMPR	001	0066	3209	
B@TMPT	001	0063	3208	
B@TMPU	001	0069	3210	
B@TMRD	001	0060	3207	
B@TNXT	001	0024	3186	
B@TPRT	001	004E	3201	
B@TPRU	001	0051	3202	
B@TPSE	001	006C	3211	
B@TPUT	001	003C	3195	
B@TRAC	001	0080	3445	
B@TREA	001	0048	3199	
B@TREM	001	0003	3171	
B@TRSR	001	004B	3200	
B@TRST	001	003F	3196	
B@TRTN	001	0036	3193	
B@TSTP	001	006F	3212	
B@VMC1	001	0056	3584	
B@VMLB	001	F0CD	3589	
B@VMSB	001	F5E5	3588	
B@VMSZ	001	0000	3585	3587 3588 3589
B@VMTB	001	0000	3587	
B@ZNEG	001	00D0	3458	3324
B@ZPOS	001	00F0	3457	4135 4628 4820 4983 5158 5191 5362 5617 5928 6639 6643 6649 6656 6673 6684 7483 7998 9888 9891 2912 2913 3324 3369 4014 4017 4295 4300 5945 5946 9225
BFPCAR	001	4FFB	0585	0588
BFPCRO	001	4FFF	0587	0567*
BUFADR	002	4DD9	0417	0309* 0351 0462* 0564
BUFRWK	002	4DDE	0421	1472* 0463* 0475* 0497
CBFAD1	001	0C70	5889	5893
CBFEXP	001	0002	5938	5920 5921 5939
CBFPZD	004	0C70	5898	
CBFSFT	002	0CB1	5939	5920
CBF100	004	0C97	5922	5919* 5920* 5921*
CBF900	004	0CAC	5934	5910
CCZAD1	001	04AD	4617	4621
CCZDC1	001	04FB	4679	4645
CCZDFP	004	04AD	4626	
CCZEXP	001	04FA	4674	4629* 4645* 4652 4668
CCZONE	001	0001	4678	4644
CCZSGN	001	04F9	4673	4627* 4667
CCZ020	003	04C2	4642	4653
CCZ100	005	04DF	4664	4643
CCZ900	004	04F5	4669	4656

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 485

SYMBOL	LEN	VALUE	DEFN	REFERENCES
CDBACC	001	0004	6068	6052* 6063
CDBADD	001	0003	6069	6052 6052 6057 6057 6063
CDBAD1	001	0CB2	6032	6036
CDBNZD	004	0CB2	6041	
CDBONE	001	0CDA	6073	6042
CDB010	003	0CBD	6050	6049* 6058 6058* 6059
CDB100	004	0CC7	6057	6051
CENAD1	001	0470	4476	4480
CENXZD	004	0470	4485	
CENZRO	001	04AC	4518	4511
CEN100	003	0487	4498	4488
CEN150	003	0498	4503	
CEN200	004	049E	4511	4487
CEN900	004	04A8	4514	4491 4504
DENTRY	001	0025	0438	0311
DERROR	003	0088	0437	0361
DFKACK	001	0010	1047	1188
DFKATA	001	261C	1010	1079 1223*
DFKATC	001	2733	1175	1198 1207
DFKBLE	002	2617	1006	0895* 1080
DFKBSP	001	0016	1041	1154
DFKBS2	001	2600	0990	0875 0877 0886 0889 1031 1149 1211* 1219* 1223*
DFKBS3	001	2700	1150	0868 0897 1032 1078* 1107* 1235
DFKCNT	001	2624	1018	1098*
DFKC01	002	2621	1013	1015 1180 1196 1237
DFKDIO	001	0065	0902	0873
DFKDLP	001	2696	1103	1069 1197
DFKDTK	001	0040	1054	1075
DFKEMS	001	0002	1046	1163
DFKENB	001	0012	1051	1065
DFKENT	001	2653	1068	0889
DFKERA	001	2789	1214	1159
DFKERS	001	0003	1043	1158
DFKEUD	001	001D	1049	
DFKEXL	001	0019	1053	0991
DFKEYN	001	2500	0870	0867 0875 0892 0897 0902 0912 0989 1031 1032
DFKIAR	002	2615	1005	0887* 0900
DFKIET	002	2619	1007	1058
DFKIME	002	262C	1023	1179* 1180*
DFKIRK	001	2634	1028	0996
DFKIST	002	2621	1015	1248
DFKKIX	001	0011	1048	1190
DFKLKA	001	25F9	0971	0979
DFKLMG	002	2628	1021	0880* 0908 1192 1211 1219
DFKLNK	001	0039	0979	1223
DFKLOK	001	0018	1050	1059 1178
DFKMCT	002	262E	1024	1179
DFKMSD	002	27B1	1235	1216
DFKNAB	001	264D	1064	1061
DFKNPS	002	261F	1012	0883* 0884* 0885 0888* 1096* 1098 1105 1108*
DFKNSK	001	261D	1011	0995* 0996 1070 1073 1075 1152 1154 1156 1158 1161 1163 1165
				1176 1182* 1206
DFKNTR	001	2603	0993	0886
DFKPG2	002	263A	1031	0906
DFKPG3	002	263C	1032	0904
DFKPL1	001	2770	1200	1188* 1190* 1194

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 486

SYMBOL	LEN	VALUE	DEFN	REFERENCES
DFKPL2	001	27A3	1226	1208
DFKPL3	001	27A5	1228	1215* 1216* 1217
DFKPPL	001	2623	1016	1020 1099 1105* 1138
DFKPRT	001	26AC	1117	1100 1195 1209 1218 1238
DFKP10	001	26BD	1123	0878* 0879* 1120 1125
DFKP20	002	26BF	1124	1119*
DFKRET	002	2630	1025	0893* 1060* 1066
DFKRKY	001	0011	1045	1027 1152
DFKRMG	002	262A	1022	0882* 0885* 1138
DFKROR	001	27BB	1243	1071
DFKROS	002	2632	1026	0890* 0994
DFKRTN	001	0015	1042	1156
DFKRT1	001	2683	1095	1082 1172
DFKSGL	001	0007	1234	1230
DFKSG1	001	27A9	1232	1234 1235
DFKSPA	001	279D	1222	1162
DFKSPB	001	274D	1187	1155
DFKSPC	001	0040	1044	1161
DFKSTN	001	2626	1020	0881* 1106 1107 1192 1196* 1211* 1219*
DFKTAB	001	0005	1040	1165
DFKTBL	001	25C0	0913	0894 0979
DFKTST	001	26DD	1136	1077 1160
DFKULK	001	001C	1052	1063 1140
DFKXDP	002	2638	1030	1060
DFKXIT	001	264A	1062	1076 1083 1166 1177 1220 1239 1254
DFKXRS	002	2636	1029	0899* 1131
DFK001	001	0001	1039	1096
DFK100	004	2565	0903	0872* 0873* 0892 0894 1030
DFK120	005	2569	0904	1030
DFK140	004	257E	0909	
DFK160	003	2600	0991	0997
DFK180	003	263D	1058	1153 1212
DFK200	004	2671	1079	1224
DFK220	005	2678	1081	1079* 1106*
DFK240	004	2699	1105	1101
DFK260	004	26A8	1109	1097* 1104*
DFK280	004	26C8	1127	1121
DFK300	003	26D6	1131	1126
DFK320	004	26D9	1132	1118*
DFK340	004	26EA	1141	1137*
DFK350	001	2700	1151	1074 1183
DFK360	004	2740	1180	1181
DFK380	003	2750	1189	1078* 1171* 1191*
DFK400	004	2759	1192	1189
DFK420	003	276D	1198	1193
DFK440	003	2772	1206	1139
DFK460	003	2778	1208	1157
DFK480	004	277E	1210	1107* 1246
DFK500	003	27B2	1237	1164
DFK520	003	27BB	1244	1072* 1249*
DFK540	005	27C5	1248	1244
DFPAPC	002	28DD	1377	1319* 1320* 1322 0565* 0566*
DFPASE	001	2800	1375	1269 1339* 1387 1391 1400 1417 0300 0457 0604 0627* 0657
DFPASY	002	29D5	1495	1452* 1453* 1454
DFPCFD	002	28EB	1387	1320
DFPCHK	001	5300	0603	

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 487

SYMBOL	LEN	VALUE	DEFN	REFERENCES
DFPDSV	004	28F4	1392	1409 1450 0612*
DFPENT	004	5311	0612	0607
DFPEOR	001	4DF3	0444	0403
DFPERC	001	28EE	1390	1338
DFPERR	004	29DD	1499	
DFPETN	001	28E9	1386	1309
DFPEXT	001	29D3	1494	1464
DFPGCT	001	0000	1501	1443*
DFPIOR	001	29D7	1497	
DFPIST	001	28F5	1393	1283 1285* 1293 1294* 1295* 1297* 1299 1303* 1395 1407 1450* 0606 0610* 0612 0615
DFPITE	002	28E7	1402	1363
DFPLBU	002	29D2	1493	1465
DFPMCK	001	2939	1442	1435
DFPNDX	001	2900	1419	1368 1418
DFPOFF	001	28E3	1383	1343
DFPOGE	001	29DD	1500	1428
DFPORK	002	28E5	1384	1302* 1303 1408
DFPPCF	001	28DE	1378	1286 1299* 1301* 1302 1305 1306* 1309* 1310 1312* 1314* 1316* 1318* 1387 1411 0615*
DFPPCH	002	28FD	1400	1271 1276 1356
DFPPCO	001	28E2	1382	0498* 0567
DFPPOS	001	4DE4	0427	0503* 0509 0529* 0530 0539 0546 0547* 0555* 0559* 0560
DFPRCK	001	28A5	1337	
DFPRCL	001	0002	1376	1338 1389
DFPRCT	002	28ED	1389	1338* 1443* 1448*
DFPRES	001	4DDC	0420	0469* 0477 0484* 0491 0672 0673
DFPRNT	001	2800	1270	1368 1375 0413 0443 0502* 0538*
DFPRPE	001	28D3	1366	1348 0413 0443
DFPRSN	002	29D9	1498	1425* 1434 1470
DFPSCK	001	2932	1436	
DFPSC2	001	2948	1447	1441
DFPSYC	001	28F9	1397	1391 1445* 1455* 1456 1457* 1459* 0614*
DFPULK	001	5339	0625	0608
DFPVCK	001	0004	1502	1470
DFPWITH	002	4DDB	0419	0470* 0471* 0475 0477 0484 0485
DFPX39	001	0039	1401	1277
DFPYCD	002	28F0	1391	1453
DFPYCT	001	0001	1403	1448*
DFP001	002	28E7	1385	1288 1306 1318 1402 1410 1433 1443 1448 1457
DFP100	004	2805	1272	1278 1358 1400
DFP101	004	280E	1274	1273*
DFP102	005	2812	1276	1359
DFP105	002	281F	1279	0628
DFP115	001	2823	1281	0617 0619
DFP120	003	283D	1293	1284
DFP140	004	2853	1299	1296
DFP160	005	2862	1305	1298
DFP180	003	2872	1310	1287 1308
DFP200	005	2878	1312	
DFP220	006	2888	1317	1315
DFP240	003	2892	1319	1289 1311 0370
DFP250	001	2899	1321	0570
DFP260	003	289C	1323	1341* 1477 0502* 0538*
DFP270	003	289F	1324	1325 1327 0369* 0402* 0492*
DFP280	003	28A2	1328	1339* 0315 0627*

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 488

SYMBOL	LEN	VALUE	DEFN	REFERENCES
DFP300	003	28CA	1359	1491 0406
DFP320	003	28AC	1340	1328
DFP330	003	28BB	1350	1351 1353 1355 1464* 0304* 0311* 0361* 0401*
DFP333	001	28BA	1349	0313* 0403*
DFP335	003	28B8	1344	1345 1347
DFP340	003	28B2	1342	1364
DFP360	003	28CD	1363	1342
DFP378	001	2911	1424	1421
DFP380	005	2927	1433	1427
DFP400	004	29B2	1482	1444 1449
DFP420	003	2953	1452	1446
DFP440	004	296F	1460	1458
DFP480	001	29CE	1490	1485
DLFBPT	001	4DDF	0422	1473* 1474 0495* 0562
DLFCAR	001	00FB	0588	0566
DLFDSV	004	28F4	1409	0659* 0662 0664 0673*
DLFEOR	001	4DF6	0446	0313
DLFIST	001	28F5	1407	0324 0339 0340 0358 0662* 0663 0671* 0672*
DLFMAR	002	4DE1	0424	
DLFORK	002	28E5	1408	1467* 1472 0519* 0520* 0521 0521* 0522 0522* 0523 0523* 0524 0524*
				0525 0525* 0530 0563* 0568
DLFPCF	001	28DE	1411	0368* 0404* 0508* 0548* 0557* 0560*
DLFPCH	002	4DED	0433	0344
DLFPC1	002	4DEF	0434	0392
DLFPRT	001	4D00	0303	1324 1346 1354 1472* 1473* 1474 1475* 1489 1494 0301 0433 0434
				0437 0438 0458 0605 0658
DLFRPE	001	4DCD	0409	1346 0445
DLFRTN	001	0001	0439	0660 0675
DLFRTY	002	4DE3	0425	0380
DLFSWC	001	4DF0	0435	0660 0667* 0675*
DLFVD1	002	4DD7	0416	0307 0460
DLFVD2	002	4DEB	0429	0362
DLFX4E	001	004E	0430	0630
DLFX53	001	0053	0431	0632
DLF001	002	28E7	1410	0507 0536 0559 0659 0666
DLF050	001	4D18	0310	
DLF100	001	4D25	0322	1354 0388 0438 0676
DLF140	001	4D3F	0335	0332
DLF143	004	4D4E	0345	0393
DLF145	004	4D57	0347	0346*
DLF146	003	4D5B	0351	0668
DLF150	005	4D5E	0352	0663* 0664* 0665* 0666*
DLF155	004	4D63	0354	0305*
DLF160	001	4D6D	0360	0367
DLF165	005	4D70	0362	
DLF170	001	4D78	0365	0325
DLF175	001	4DB9	0400	0359 0434
DLF350	003	4D88	0375	1475* 1494 0376 0378 0379* 0437
DLF355	005	4D8E	0380	
DLF360	003	4DA8	0388	0312* 0323* 0375 0389 0391
DLF375	003	4DB0	0393	
DLF400	001	4D93	0381	0363 0433
DLF425	004	4DA4	0386	0384*
DLF450	001	29AF	1476	1461 1471
DLF500	001	4E25	0474	0424
DLF525	001	4E29	0476	0472

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 489

SYMBOL	LEN	VALUE	DEFN	REFERENCES
DLF550	001	4E3C	0490	0478
DLF600	001	4E44	0494	0486
DLF700	001	4E49	0496	0425
DLF800	001	4EC7	0554	0541
DLF900	001	4ECC	0556	0531
DLF920	001	4ECF	0558	0549
DLF950	001	4ED7	0561	0510 0540
DLF960	004	539F	0663	0661
DLTABL	001	0090	0432	0557
DLTABR	001	00A0	0436	0548
FABABS	001	1761	7478	
FABAD1	001	1761	7469	7473
FAB100	003	1765	7483	
FBSADA	008	009F	6503	6428
FBSATA	008	129F	6497	6503
FBSATN	004	1100	6341	6503
FBSAT1	008	1188	6414	6389
FBSBN1	001	1297	6493	6474
FBSINS	005	128E	6491	6477
FBSINZ	004	1285	6488	6427
FBSLNR	001	0009	6405	6376 6384 6448 6453 6491
FBSLNT	001	128D	6490	6470
FBSLNW	001	000B	6406	6376 6378* 6438 6438 6449 6449 6452 6452 6471 6471 6472 6473 6482 6483
FBSLST	001	12B6	6502	6491
FBSMDS	004	1296	6492	6469
FBSMDZ	004	128C	6489	6436
FBSONE	007	1190	6416	6358 6363
FBSRND	001	1180	6413	6390
FBSRRR	001	0614	6407	6384 6448* 6453* 6491*
FBSSGN	001	117F	6412	6344* 6396
FBSWWW	001	061F	6408	6376* 6378* 6438 6449* 6452* 6471
FBSZER	001	0005	6404	6376 6376
FBSZZZ	011	1278	6482	6437* 6449 6452 6488*
FBS10Y	011	1283	6483	6472* 6473
FBS100	003	1111	6350	
FBS110	003	111D	6357	6351
FBS190	004	1148	6370	6353
FBS200	005	114E	6376	
FBS400	004	120A	6436	6478
FBS405	004	120E	6437	6427* 6429 6436* 6450
FBS420	005	1212	6438	6454 6476
FBS425	005	1224	6448	6428* 6470*
FBS430	005	1229	6449	
FBS440	005	1231	6452	6446
FBS450	005	1236	6453	6447 6469* 6477*
FBS600	004	123E	6458	6439
FBS800	003	1166	6388	6352* 6357*
FBS810	004	116D	6390	6388
FBS900	004	117B	6400	6343
FCSACS	004	1400	6615	
FCSARG	008	14E5	6735	6668* 6684 6690 6707
FCSASN	004	1413	6627	
FCSONE	007	14DC	6733	6640 6657 6671
FCSPI2	007	14D4	6731	6638* 6642 6643* 6648 6649* 6700 6721
FCSSGN	001	14DD	6734	6642* 6655* 6667

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 490

SYMBOL	LEN	VALUE	DEFN	REFERENCES
FCSSWH	001	14E6	6736	6616* 6628* 6682 6689* 6718
FCS050	004	143B	6648	6623 6641
FCS100	004	1445	6655	6619
FCS125	004	144C	6657	6644
FCS150	004	1453	6662	6617* 6629*
FCS200	004	145A	6667	6658
FCS300	003	1495	6698	6683 6685
FCS350	004	14A5	6706	6699
FCS400	004	14B1	6712	6692
FCS500	004	14BD	6720	6702
FCS900	004	14C9	6726	6634 6650 6663 6719
FEBAAD	002	409B	7516	7431
FEBBN1	002	40A5	7522	7487 7492
FEBDEC	002	40A3	7520	7474
FEBDM1	002	4093	7509	7453* 7487*
FEBDM2	002	4095	7510	7453
FEBDM3	002	4097	7511	7449* 7492*
FEBIVA	002	40A1	7518	7457
FEBMAD	001	4007	7430	
FEBMSB	001	4000	7424	
FEBSAD	002	4099	7515	7425
FEBVAA	002	408D	7505	7446* 7461
FEBVAB	002	408F	7506	7447* 7468
FEBVAC	002	4091	7507	7445* 7457* 7482
FEB020	004	400B	7432	7426
FEB100	004	4023	7445	
FEB200	004	4033	7453	7493
FEB300	004	4037	7457	7488
FEB400	004	405B	7475	7425* 7431*
FEB450	004	4084	7499	7436* 7438 7439* 7441
FEB500	004	4088	7500	7477 7497
FGSBN1	001	05CF	4954	4836 4857 4909 4936
FGSEVP	004	0500	4797	
FGSFVE	001	05D0	4955	4926
FGSINL	001	0005	4949	4884 4916 4956
FGSINS	006	05F5	4965	4884
FGSITN	001	05FC	4970	4885* 4906 4909*
FGSMNN	010	05E9	4959	4846
FGSMOD	005	05D5	4956	4916
FGSNNL	001	000A	4950	4957 4959 4961
FGSNNN	010	05DF	4957	4827
FGSONE	001	0001	4947	4827 4835 4855 4865 4867* 4894 4908 4961
FGSSFZ	002	0619	4990	4987
FGSTEN	011	05F4	4961	4833 4835 4855
FGSTHR	001	0003	4948	4827 4833 4835 4846 4855 4950
FGSXM1	001	05FB	4969	4832* 4836* 4854* 4857* 4936* 4939
FGS001	004	0600	4982	
FGS004	004	0614	4988	4984 4986
FGS005	004	050C	4806	4828
FGS010	004	0513	4814	4805
FGS100	004	0529	4833	4837
FGS110	004	053B	4846	4821
FGS115	003	054C	4854	4847
FGS120	004	054F	4855	4858
FGS210	005	055D	4865	4834 4856
FGS220	005	057D	4894	4905 4917

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 491

SYMBOL	LEN	VALUE	DEFN	REFERENCES
FGS250	006	0585	4901	4884* 4916*
FGS260	003	058E	4906	4895
FGS300	004	05A7	4925	4907
FGS305	005	05BF	4937	4928
FGS900	004	05CB	4943	4807 4850
FHSD00	001	15E7	7010	6893
FHSHCS	001	1500	6861	
FHSHFN	001	1500	6846	6850 6907 6938
FHSHSN	001	1557	6911	
FHSHTN	001	1593	6942	
FHSI01	007	16B8	7185	7053 7085
FHSKNA	007	16C0	7187	7133
FHSKNB	007	16C8	7189	7128
FHSKNC	007	16D0	7191	7145
FHSMK	001	000F	7006	6950
FHSONE	007	15EF	7012	6878 6949 6970 6976
FHSPG2	001	1600	7033	7037
FHSWK1	008	15F7	7016	6877* 6879 6883 6972* 6982
FHSWK2	008	16D8	7195	7052* 7059 7064 7084* 7092 7118* 7138 7140* 7150 7156 7157
FHSWK3	008	16E0	7196	7124* 7144 7152* 7162
FHS010	004	150A	6868	6919
FHS015	004	1510	6870	
FHS017	004	1517	6872	6863* 6882 6913*
FHS020	004	151E	6877	6871
FHS025	004	152A	6880	
FHS030	003	1539	6884	6864* 6914*
FHS035	004	153F	6889	6884
FHS040	004	1543	6893	6930
FHS045	004	154D	6896	
FHS050	004	1551	6900	6873 6921 6951
FHS100	003	1561	6918	
FHS170	004	156D	6925	
FHS175	003	1600	7048	6926
FHS180	004	1603	7052	
FHS190	004	160B	7054	
FHS200	004	160F	7058	
FHS210	004	1617	7060	
FHS220	004	161B	7064	
FHS230	003	1573	6930	
FHS300	003	1597	6947	
FHS305	004	15A7	6955	6948
FHS310	003	15AF	6960	
FHS320	004	15B5	6965	
FHS330	004	15BB	6970	
FHS340	004	15C7	6976	
FHS350	004	15D2	6982	
FHS360	003	15DA	6987	
FHS370	004	15DD	6991	6961
FHS380	003	1627	7080	6992
FHS385	004	162A	7084	
FHS390	004	1639	7091	
FHS400	004	15E3	6996	6987
FHS800	003	1648	7112	7048 7080
FHS805	003	164B	7116	
FHS810	004	1655	7122	
FHS820	004	1661	7128	

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 492

SYMBOL	LEN	VALUE	DEFN	REFERENCES
FHS830	004	1669	7133	
FHS840	004	1671	7138	
FHS850	004	167D	7144	
FHS870	004	1695	7156	
FHS880	004	16A1	7162	
FHS890	004	16A9	7167	7112* 7117
FHS960	004	16AD	7174	7097
FITMPQ	001	3F00	7203	
FJBINT	001	176C	7578	
FJBSGN	001	17A6	7619	7593* 7602
FJB100	004	176C	7582	
FJB150	003	1770	7583	
FJB200	003	1776	7588	
FJB300	004	177C	7593	
FJB400	004	178D	7598	7595* 7596*
FJB500	004	1791	7602	7597
FJB600	003	1798	7607	7584
FJB650	003	179F	7609	
FJB700	004	17A2	7613	7589 7603
FKSADD	001	0002	4224	4194
FKSARG	008	037C	4347	4253* 4254* 4258* 4279* 4321* 4332 4333*
FKSCNT	008	0093	4369	4894 4908
FKSCNV	008	02A3	4229	4107* 4118* 4213
FKSCON	008	0393	4359	4288 4369
FKSDCR	001	02BC	4238	4177 4186
FKSINC	001	038A	4354	4301
FKSINS	006	036F	4343	4267
FKSINT	001	0005	4338	4267
FKSITN	001	0384	4349	4268* 4301* 4305
FKSLGT	004	0200	4107	4117 4126
FKSLOG	003	0219	4127	
FKSLTW	004	020B	4118	
FKSMDY	005	0389	4353	4327
FKSMOD	001	0005	4339	4327 4353
FKSONE	001	0001	4223	4173 4173 4176 4179 4181 4186* 4188* 4194* 4196 4197 4198 4251* 4252 4288 4310 4333*
FKSRND	001	038B	4355	4310
FKSSFT	001	0002	4225	4197*
FKSSHT	007	0383	4348	4332* 4333
FKSTEN	007	02AB	4234	4107
FKSTNE	008	02BB	4237	4176 4181
FKSTWO	007	02B3	4236	4118
FKS010	003	0212	4120	4109
FKS020	004	021F	4129	4121
FKS025	004	022F	4137	4108* 4119* 4128* 4134
FKS030	005	0236	4143	4136
FKS090	004	0300	4251	4369
FKS095	004	0321	4274	4273*
FKS100	005	0325	4279	4297 4334
FKS120	006	0332	4296	4267* 4327*
FKS150	004	033B	4301	4280
FKS175	005	0358	4321	4306
FKS205	003	024E	4174	4178
FKS210	003	025F	4179	4175 4187
FKS220	004	0270	4188	4180
FKS600	003	028D	4212	4120* 4127*

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 493

SYMBOL	LEN	VALUE	DEFN	REFERENCES
FKS700	004	0298	4219	4138 4212
FLBADA	002	43F0	8111	8076* 8084* 8087
FLBADB	002	43EE	8110	8075* 8085
FLBAD1	001	43A0	8054	8058
FLBCOL	002	43EC	8109	8080* 8092*
FLBLEL	002	43F4	8115	8084
FLBMAS	001	43A0	8063	
FLBONE	002	43F6	8116	8092 8097
FLBROW	002	43E8	8107	8097*
FLBSAV	002	43EA	8108	8080
FLB050	004	43B3	8075	8069
FLB100	004	43BB	8080	8098
FLB200	004	43BF	8084	8093
FLB800	004	43E3	8102	8071
FMBABC	002	41B7	7672	7604* 7613
FMBACB	002	41C3	7678	7614 7620* 7630* 7631
FMBACC	002	41CB	7682	7622* 7634 7638*
FMBADA	002	41BD	7675	7650* 7651
FMBADB	002	41C5	7679	7614* 7620
FMBADC	002	41CD	7683	7603* 7604 7613* 7621* 7622
FMBCLG	002	41BB	7674	7605 7605* 7606* 7638
FMBDEC	002	41D4	7690	7640
FMBDZR	001	41D2	7689	7623
FMBLEL	002	41D1	7688	7606 7621 7630 7650
FMBLIC	002	41C7	7680	7619* 7648*
FMBL1K	002	41C1	7677	7619
FMBL2C	002	41C9	7681	7607* 7615* 7657*
FMBL2K	002	41B9	7673	7615
FMBL3C	002	41BF	7676	7662*
FMBMPY	001	4100	7595	7597
FMBONE	002	41CF	7687	7607 7648 7657 7662
FMBZER	002	4257	7725	7702 7705 7707
FMB100	004	4119	7606	7608
FMB500	004	4127	7613	7663
FMB550	004	412F	7615	7609
FMB600	004	4133	7619	7658
FMB700	003	414A	7628	7649
FMB800	004	41B2	7667	7599 7644
FMB900	004	424E	7723	7701* 7703 7704* 7706 7708 7709* 7711 7713 7714* 7716 7717* 7719
FMB950	004	4252	7724	7721 7722
FNBBN1	001	08EC	5257	5145
FNBCNT	001	08E0	5250	5142* 5145* 5151* 5152 5201
FNBD1C	001	08EE	5260	5175
FNBDGT	001	08E1	5251	5149* 5175* 5192
FNBF1P	001	08EE	5259	5132
FNBMK1	001	0002	5246	5132
FNBMN1	002	08EB	5256	5146
FNBPWR	004	0800	5103	
FNBST1R	008	08E9	5252	5210* 5220
FNB005	003	0810	5122	
FNB010	003	081D	5129	5113
FNB030	003	082E	5142	5130
FNB200	003	0831	5143	5147
FNB250	004	0841	5149	5144
FNB275	004	0859	5160	5118

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 494

SYMBOL	LEN	VALUE	DEFN	REFERENCES
FNB300	003	0860	5167	5153
FNB350	004	0871	5175	5181
FNB400	003	088B	5189	5168 5170
FNB500	003	08A6	5209	5159 5190 5193 5202
FNB800	004	08A9	5210	5208
FNB880	003	08D6	5240	5203* 5209* 5234
FNB900	004	08DC	5242	5123 5125 5133 5161 5176 5180 5226 5236 5240
FPBCON	001	17EA	7869	7840 7855
FPBDEG	001	17DA	7850	
FPBRAD	001	17CB	7835	
FPB100	004	17CB	7839	
FPB200	004	17CF	7840	
FPB300	004	17D3	7844	
FPB400	004	17DA	7854	
FPB450	004	17DE	7855	
FPB500	004	17E2	7859	
FPB600	004	17E6	7863	7845
FQSCNT	001	18B0	8100	8035*
FQSCT2	001	18B1	8103	8045* 8061*
FQSI01	002	188E	8091	8057 8061
FQSI64	001	188F	8092	8035
FQSNUL	001	0000	8087	7993
FQSRND	001	1800	7969	7973
FQSSVP	008	1897	8093	8013 8018 8070
FQSSVX	008	189F	8094	
FQSSVY	008	18A7	8095	
FQSSVZ	008	18AF	8096	7988 8003
FQSWKU	008	18B9	8104	7999* 8008* 8009* 8013 8018* 8031 8040 8055* 8056
FQSWKX	008	18C1	8105	8008 8023*
FQSWKY	008	18C9	8106	8009 8023 8027*
FQSWKZ	008	18D1	8107	7988* 8003* 8027 8031*
FQS005	004	1800	7978	
FQS010	004	1808	7983	
FQS020	004	180F	7988	
FQS030	004	1817	7993	7984
FQS040	003	181E	7998	
FQS050	004	1825	8003	
FQS100	004	182C	8008	7994 8036
FQS110	004	1834	8013	8004 8019
FQS120	004	183B	8018	
FQS130	004	1842	8023	8014
FQS140	004	1846	8027	
FQS150	004	184A	8031	
FQS160	004	184E	8035	
FQS170	004	1855	8040	
FQS180	003	185C	8045	
FQS190	003	1862	8050	8062
FQS200	004	1868	8055	
FQS210	004	1874	8061	
FQS220	003	187B	8066	
FQS230	004	187E	8070	8051
FQS240	004	1885	8072	
FQS250	004	1889	8076	
FRBACC	001	0001	5454	5425*
FRBBN1	001	09B4	5472	5385
FRBDC1	001	09B5	5473	5425

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 495

SYMBOL	LEN	VALUE	DEFN	REFERENCES
FRBEVN	001	0001	5453	5379*
FRBEXP	001	0002	5456	5393 5394 5395
FRBFC1	009	09AC	5462	5412* 5424
FRBFC2	007	09B3	5468	5407* 5408 5408* 5409 5410 5413* 5431 5440 5441 5441*
FRBLNG	001	0000	5457	5422* 5439*
FRBNRM	001	09B6	5474	5398
FRBONE	001	0001	5452	5391*
FRBSQR	004	0900	5361	
FRBSUB	009	09A3	5461	5411* 5422 5424* 5439 5440*
FRBTWO	001	0002	5455	
FRB005	003	0911	5366	5363
FRB010	003	0917	5374	
FRB020	004	0927	5385	5375
FRB030	004	0932	5391	5381
FRB100	004	0969	5422	5426 5443
FRB150	003	097B	5431	5423
FRB400	003	097E	5432	
FRB850	004	0993	5447	5432
FRB900	004	0997	5448	5365 5367
FSSADD	001	0003	5708	5647
FSSCOF	007	0B70	5783	5745
FSSCOS	004	0A00	5590	5584
FSSDCO	001	0B67	5777	5751
FSSEQ8	001	0001	5706	
FSSFP1	007	0AC8	5718	5629
FSSHLF	007	0AD6	5722	5669 5685
FSSINP	008	0B66	5773	5741* 5764
FSSINT	001	0003	5709	5651 5653 5653 5654 5656 5656 5719 5720
FSSLOP	001	0B5E	5772	5740* 5762*
FSSMDY	001	0AD8	5724	5690
FSSMN1	001	0B68	5778	5762
FSSMOD	001	0002	5707	5690
FSSOCT	001	0AC0	5713	5591* 5612* 5616* 5647 5658* 5686 5688 5693 5699
FSSONE	001	0001	5705	5690*
FSSRST	008	0B5D	5771	5745* 5753 5761*
FSSSIN	004	0A1A	5606	
FSSSQD	008	0B55	5770	5744* 5752
FSS008	003	0ACE	5720	5654 5656
FSS050	003	0A14	5599	5596
FSS064	003	0ACB	5719	5651 5653
FSS100	003	0A33	5617	5600
FSS150	003	0A36	5622	5615
FSS160	004	0A3C	5624	5599* 5613*
FSS200	004	0A43	5629	5623
FSS205	003	0A4B	5634	
FSS225	004	0A5B	5651	
FSS230	004	0A66	5654	5652 5657
FSS260	004	0A74	5658	5655
FSS300	004	0A81	5669	5635
FSS360	003	0A9D	5691	5700
FSS370	004	0AA0	5692	5701
FSS380	003	0AA4	5693	5689
FSS400	004	0AAD	5696	5694
FSS425	004	0AB3	5698	5611 5625
FSS450	003	0AB7	5699	5687
FSS900	004	0B00	5734	5697

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 496

SYMBOL	LEN	VALUE	DEFN	REFERENCES
FSS905	004	0B17	5745	5746
FSS910	004	0B1F	5751	5763
FSS920	004	0B2F	5759	5746* 5751*
FTBAD1	001	1700	7297	7301
FTBCON	001	1760	7377	7364
FTBCSC	001	1725	7330	
FTB100	004	1700	7310	
FTB125	004	1706	7315	
FTB150	004	1714	7322	7316
FTB160	003	1718	7323	
FTB170	004	171E	7325	
FTB200	004	1725	7334	
FTB300	004	172B	7339	
FTB400	004	1739	7346	7340
FTB450	003	173D	7347	
FTB475	004	1743	7349	
FTB500	004	174A	7360	7324 7348
FTB550	004	174E	7364	
FTB600	004	1758	7367	
FTB700	004	175C	7371	7318 7326 7342 7350
FTSSEC	001	1700	7306	
FUBSGN	001	17A7	7706	
FUBSSA	001	17CA	7737	7716* 7727
FUB100	004	17A7	7710	
FUB150	003	17AB	7711	
FUB200	004	17B1	7716	
FUB300	003	17B5	7720	
FUB350	003	17BC	7722	
FUB400	004	17C2	7727	
FUB500	004	17C6	7731	7712
FVBADD	002	45E2	8434	8355* 8371* 8411 8435
FVBDET	003	4543	8372	8369
FVBDIS	001	0017	8442	8371
FVBDP@	002	4589	8394	8380*
FVBBDPL	001	45D5	8419	8372* 8375* 8377* 8378* 8379
FVBBDP2	001	45DB	8427	8398* 8404* 8412
FVBIDS	002	45E8	8439	8377
FVBINV	001	4500	8346	8350 8370
FVBLGM	001	001C	8443	8353 8399
FVBLNG	002	45E6	8438	8398
FVBBSAV	001	2A00	8445	8421
FVBSTM	001	0023	8444	8402
FVBZRO	002	45E4	8437	8357 8383
FVB010	004	4552	8377	8374
FVB015	003	4597	8399	8353*
FVB020	004	459A	8400	8397
FVB021	003	45A2	8402	8399*
FVB025	004	45CB	8414	8406
FVB030	004	45D1	8416	8358 8362 8365 8368 8384 8388 8403
FVB08K	001	0012	8441	8372 8422
FWSCOT	004	0D00	6190	
FWSLRG	001	0003	6230	6195 6206
FWSPCH	120	0DFB	6231	
FWSSAV	008	0D27	6200	6198* 6210* 6214 6223
FWSTAN	004	0D28	6204	
FWS005	003	0D10	6195	6192

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 497

SYMBOL	LEN	VALUE	DEFN	REFERENCES
FWS007	003	0D2F	6206	
FWS009	004	0D35	6208	6193* 6194 6196* 6197 6205*
FWS030	004	0D3C	6210	6207
FWS040	004	0D62	6220	6199
FWS900	004	0D80	6228	6209
FXBADA	002	448D	8242	8211* 8212
FXBADB	002	4495	8246	8203* 8217* 8218
FXBADC	002	4487	8239	8212* 8220 8222*
FXBCIN	002	4489	8240	8206*
FXBCOL	002	4493	8245	8213* 8226*
FXBLEL	002	449A	8252	8205 8211 8217
FXBONE	001	4498	8251	8206 8226 8231
FXBRDP	002	448B	8241	8204* 8205* 8222
FXBROW	002	448F	8243	8231*
FXBSAV	002	4491	8244	8213
FXBTRN	001	4400	8181	
FXBZER	002	4497	8250	8187 8190 8204
FXB100	004	443B	8205	8207
FXB200	004	4446	8211	8232
FXB300	004	4452	8217	8227
FXB800	004	447E	8234	8186* 8188 8189* 8191 8192* 8194 8196 8197* 8199
FXB900	004	4482	8235	8233
FYBAMA	002	42C8	7847	7806* 7824
FYBAMB	002	42C6	7846	7798* 7810
FYBCOL	002	42D2	7849	7802* 7829*
FYBDEC	002	42D8	7854	7829 7834
FYBINC	002	42D6	7853	7806
FYBROW	002	42C2	7844	7834*
FYBSAV	002	42C4	7845	7802
FYBSCL	008	42D0	7848	7797* 7816
FYBSMM	001	4264	7788	
FYB100	004	4276	7797	7791
FYB200	004	427E	7802	7835
FYB300	004	4282	7806	7830
FYB400	004	42BD	7839	7793 7819
FZABN1	002	3E90	7013	6922 6927
FZADM2	001	3E98	7025	6910* 6922* 6927*
FZADVA	001	3E93	7020	7023 7024 7025 7026
FZADVR	008	3E9A	7021	6905*
FZAD2S	001	3E96	7024	6910
FZAMGT	001	3E06	6879	
FZAMIO	001	3E00	6864	6872 6881 6890
FZAMPT	001	3E0C	6888	
FZAMRD	001	3E00	6870	
FZAPFL	002	3E92	7014	6914
FZAVAD	001	3E9A	7026	6914* 6955 6979 6992
FZA010	004	3E0F	6898	6873 6882
FZA020	004	3E13	6904	
FZA030	004	3E1B	6910	6928
FZA040	004	3E1F	6914	6923
FZA041	001	3E94	7023	
FZA050	003	3E23	6918	6872* 6881* 6890*
FZA060	004	3E26	6922	6958 6981 7005
FZA070	004	3E2D	6927	
FZA080	004	3E34	6932	6950 6974 7003
FZA200	004	3E3C	6944	6872

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 498

SYMBOL	LEN	VALUE	DEFN	REFERENCES
FZA210	004	3E42	6949	
FZA220	004	3E49	6954	
FZA300	004	3E59	6968	6881
FZA310	004	3E5F	6973	
FZA320	004	3E66	6978	
FZA400	004	3E72	6991	6890
FZA410	004	3E7F	6997	
FZA420	004	3E85	7002	
FZBADR	002	438C	7997	7943* 7961* 7962
FZBBN1	002	437A	7988	7967 7972
FZBCOL	002	438A	7996	7947* 7952 7967*
FZBCON	001	4324	7933	
FZBC00	001	4371	7983	7940
FZBC01	001	4376	7986	7920 7934
FZBDAG	005	4384	7993	7920* 7934* 7940* 7941 7957
FZBELE	005	437F	7992	7941* 7951
FZBIDN	001	4300	7919	
FZBLNG	002	4378	7987	7961
FZBROW	002	4386	7994	7952 7972*
FZBSAV	002	4388	7995	7947
FZBZER	001	432B	7939	
FZBZRO	003	4375	7985	7922
FZB100	004	4316	7926	7923
FZB200	004	432F	7941	7935
FZB300	004	4337	7943	7927
FZB400	004	433B	7947	7973
FZB500	004	433F	7951	7968
FZB600	004	434E	7961	7953
FZB700	004	4369	7977	7925 7929
FZCAPR	002	3FA3	7323	7222
FZCAPU	002	3FA5	7324	7233
FZCBN1	002	3F9F	7320	7289 7302 7307
FZCDM1	001	3FA7	7333	7307*
FZCDM2	001	3FAB	7335	7269* 7289 7302*
FZCDVA	001	3FA6	7330	7333 7334 7335 7336
FZCDVR	008	3FAD	7331	7244*
FZCD2S	001	3FA9	7334	7269
FZCMPL	001	3F06	7217	
FZCMPS	001	3F00	7208	
FZCMPU	001	3F13	7229	
FZCPFL	002	3FA1	7321	7274
FZCVAD	001	3FAD	7336	7274* 7279
FZC005	003	3F09	7221	7211
FZC010	004	3F1D	7243	7223
FZC020	006	3F25	7248	
FZC025	004	3F35	7253	7249
FZC030	004	3F3F	7257	7251
FZC040	004	3F49	7264	7308
FZC060	004	3F5B	7274	7303
FZC070	004	3F5F	7278	
FZC080	004	3F70	7287	7210* 7219* 7231*
FZC085	004	3F7B	7291	7221* 7232*
FZC090	004	3F7F	7293	7290
FZC095	002	3F84	7295	7222* 7233*
FZC100	004	3F8C	7302	
FZC110	004	3F93	7307	

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 499

SYMBOL	LEN	VALUE	DEFN	REFERENCES
FZC120	004	3F9A	7312	7298
FZDBN1	002	3DE0	6673	6645 6650 6656
FZDBVA	002	3DE4	6676	6534 6661
FZDDLML	002	3DE6	6678	6572
FZDDM1	001	3DE8	6687	6656*
FZDDM2	001	3DEC	6689	6562* 6571 6572 6645*
FZDDTM	001	0080	6698	6574 6678
FZDDVA	001	3DE7	6684	6687 6688 6689 6690
FZDDVR	008	3DEE	6685	6529*
FZDD2S	001	3DEA	6688	6562
FZDMIP	001	3D00	6520	
FZDPFL	002	3DE2	6675	6628
FZDPGB	001	3D00	6513	6517
FZDVAD	001	3DEE	6690	6530 6552 6563* 6628* 6640
FZDVAS	002	3DF0	6692	6530* 6552* 6563
FZD005	004	3D2A	6552	6657
FZD010	004	3D2E	6556	6617
FZD020	004	3D34	6562	6544
FZD030	004	3D3C	6570	6652
FZD040	004	3D4E	6575	6573
FZD050	004	3D52	6579	6536* 6632
FZD052	003	3D56	6580	6543* 6583* 6616*
FZD055	003	3D60	6583	6580
FZD060	004	3D82	6604	6592
FZD070	004	3D9B	6628	6611 6651
FZD080	004	3DC7	6656	6646
FZLBLN	002	46BC	8741	8609* 8616 8623 8623*
FZLDAC	004	46C0	8742	8613* 8614 8618 8619 8619*
FZLDC1	001	46B7	8732	8613
FZLDLN	004	46C6	8746	8614* 8618*
FZLINT	001	4600	8608	
FZLPAD	001	46CB	8758	8629*
FZLPAL	001	46BA	8740	8648* 8649* 8650 8658* 8659* 8660
FZLPAR	001	46C1	8744	8628
FZLPCT	001	46C9	8757	8669* 8680*
FZLPDA	002	46B9	8734	8634
FZLPFN	001	46C8	8756	8668* 8679*
FZLPPL	001	46C8	8750	8684 8756 8757 8758
FZLP1B	001	4600	8593	8597
FZLRPL	001	46CC	8761	8674
FZL010	004	4605	8613	
FZL020	003	460D	8615	8624
FZL030	003	4610	8616	8615* 8621 8621* 8622
FZL040	004	461A	8619	8617
FZL050	003	462F	8628	8620
FZL052	003	4646	8645	
FZL054	003	465C	8655	8640 8691
FZL058	003	466D	8662	8651
FZL060	003	4673	8668	
FZL070	003	467D	8674	8662
FZL080	003	4683	8679	8663
FZL090	003	4689	8684	8670
FZL100	005	468F	8689	
FZL110	003	4694	8691	8645* 8655*
FZL120	004	4697	8695	
FZL200	003	469B	8704	8675 8685

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 500

SYMBOL	LEN	VALUE	DEFN	REFERENCES
FZL210	003	469E	8706	8646* 8656*
FZL220	003	46AA	8717	8706
FZL230	004	46AD	8719	8634* 8635* 8639 8689
FZL240	002	46B2	8720	8717*
FZL250	004	46B3	8724	8704* 8713
FZRBAT	001	336D	3743	3663
FZRBN1	001	336B	3736	3726
FZRDCA	001	332F	3677	3746
FZRDDL	001	3349	3698	3747
FZREAD	001	3300	3640	3745 3746 3747
FZREOP	001	335F	3721	3745
FZRLDA	001	336C	3737	3689
FZR010	004	3307	3647	
FZR020	005	330F	3654	3643 3713 3728
FZR030	002	3319	3657	3654*
FZR040	003	331A	3661	
FZR050	004	3324	3664	3662*
FZR060	004	3328	3665	3661*
FZR070	003	332C	3669	3664*
FZR080	005	332F	3681	
FZR090	005	3340	3689	
FZR100	003	3349	3702	
FZR110	004	334F	3707	
FZR120	005	3357	3712	3703
FZR130	005	335F	3726	
FZSBN1	001	34DF	4152	4025 4044 4086
FZSCAJ	001	34E1	4155	4083
FZSCAT	001	34E5	4173	4124
FZSCNT	004	34C7	4201	3995* 4018* 4025* 4051* 4061* 4091* 4101* 4116
FZSDAC	002	35D8	4373	4324* 4328 4330 4330*
FZSDC1	001	35CB	4357	4324
FZSDC5	001	35CC	4358	4234
FZSEXB	004	35D1	4363	4309
FZSLXB	001	0004	4362	4304 4309 4342 4344 4363 4370
FZSLXM	001	0002	4372	4318* 4324 4328 4328 4330 4330 4373
FZSNXZ	001	34E0	4154	4038
FZSPAL	001	0000	4199	4065* 4289* 4295* 4300*
FZSPCH	002	36FB	4658	4441
FZSPDA	002	34E4	4158	4134
FZSPRT	001	3400	3990	
FZSP1B	001	3400	3981	3985
FZSP2B	001	3500	4224	4228
FZSP3B	001	3600	4397	4401 4658 0637
FZSP4B	001	3700	4682	4686
FZSSAJ	001	34E2	4156	4098
FZSXWK	004	35D6	4370	4309* 4318* 4328* 4335 4337 4337* 4344
FZS010	003	3404	3995	
FZS020	004	340E	4002	
FZS030	003	3415	4013	
FZS035	003	3418	4014	
FZS040	003	3424	4018	4015
FZS050	003	3427	4023	
FZS060	003	3434	4030	4024
FZS070	004	3440	4037	
FZS080	003	344B	4042	4039* 4044* 4045 4046 4051
FZS090	003	3455	4045	4037* 4038*

VER 15, MOD 00 31/05/21 PAGE 501

[illegible]

VER 15, MOD 00 31/05/21 PAGE 502

FZS450	004	3574	4309																
FZS460	003	3586	4318																
FZS470	004	3590	4324	4314															
FZS472	003	3597	4326	4325*	4329	4329*	4331												
FZS474	004	35A1	4329	4327															
FZS480	003	35AC	4335	4313															
FZS490	003	35BB	4342	4336															
FZS500	004	35C2	4344	4342*	4343*														
FZS510	004	35C6	4348																
FZS600	006	3600	4406	4140															
FZS605	003	362B	4429	4428*															
FZS610	003	362E	4439	4175	4175	4176	4177	4178	4180	4181	4182	4183	4185	4185	4186				
				4187	4187	4188	4190	4190	4191	4192	4192	4193							
FZS615	004	3642	4444	4443*															
FZS620	003	3646	4452	4176															
FZS630	003	364C	4465	4177															
FZS632	004	364F	4467	4471															
FZS633	003	365A	4470	4658															
FZS634	004	3660	4473	4468															
FZS636	005	3664	4481	4453															
FZS638	003	3669	4482	4637															
FZS640	005	366F	4488																
FZS650	003	367B	4500	4178	4188	4193													
FZS655	003	3681	4505	4483															
FZS660	003	3687	4514	4180															
FZS670	003	368D	4523	4181															
FZS675	004	3690	4527	4515															
FZS680	003	36A1	4541	4182	4501														
FZS690	003	36A7	4550	4183															
FZS695	003	36B0	4560	4186															
FZS700	003	36B6	4571	4191															
FZS710	003	36B9	4575	4490	4506	4530	0641												
FZS720	003	36BC	4576	4542															
FZS730	006	36BF	4580	4552															
FZS740	004	36C8	4585	4407															
FZS750	004	36CE	4590	4440	4531	4581													
FZS760	003	36D2	4600	4424	4528	4551	4561												
FZS770	003	36DA	4605	4638															
FZS780	003	36E3	4617	4576															
FZS790	004	36EC	4623	4600*	4606	4617*													
FZS800	005	3700	4691	4586	0653														
FZS805	003	3725	4713	4712*															
FZS810	003	3728	4723																
FZS820	003	3740	4735																
FZS830	003	3746	4748																
FZS832	004	3749	4750	4754															
FZS834	004	375A	4756	4751															
FZS836	005	375E	4764	4736															
FZS838	003	3763	4765	4916															
FZS840	005	3769	4771																
FZS850	003	3775	4783																
FZS855	003	377B	4788	4766															
FZS860	003	3781	4797																
FZS870	003	3787	4806																
FZS875	004	378A	4810	4798															
FZS880	003	379B	4824	4784															

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 503

SYMBOL	LEN	VALUE	DEFN	REFERENCES
FZS890	003	37A1	4833	
FZS895	003	37AA	4843	
FZS900	003	37B0	4854	
FZS910	003	37B3	4858	4724 4773 4789 4813
FZS920	003	37B6	4860	4825
FZS950	004	37B9	4864	4725 4814 4835
FZS960	003	37BD	4874	4708 4811 4834 4844
FZS970	003	37C5	4879	4917
FZS980	003	37CE	4892	4860
FZS982	004	37D4	4895	4691*
FZS984	002	37D9	4896	4894*
FZS990	004	37DA	4900	4874* 4880 4892*
FZS991	005	5359	0638	
FZS992	004	5368	0643	0640
FZS993	003	5372	0646	0644
FZS994	004	538D	0654	0651
FZUAB1	002	38F4	5403	5253 5390
FZUAB2	002	38F6	5404	5321 5382
FZUB11	001	3CEF	6285	6207
FZUCAT	001	38F7	5410	5227
FZUCIN	001	3AFA	5910	5759
FZUCNT	001	3AFD	5915	5733 5733* 5759* 5792
FZUCSP	001	0000	6311	5823* 5827 5827* 6090* 6094 6094* 6260* 6268* 6272 6272*
FZUDAC	002	3CFB	6302	6160* 6164 6166 6166*
FZUDCT	001	3AFC	5913	
FZUDIN	001	3AF9	5908	
FZUECS	004	3AF7	5899	5783
FZUEXB	004	3CF9	6293	6145
FZUFCT	001	3AFD	5914	5915
FZUFIN	001	3AFA	5909	5745* 5771 5777 5910
FZUIXP	001	0004	6312	6145* 6154* 6164*
FZULCF	001	3AF2	5895	5849 5850
FZULDD	001	0002	6309	6186 6207
FZULNX	001	3AF3	5898	5787
FZULQD	001	0002	6308	5333 5873
FZULXB	001	0004	5897	5783 5783 5788 5898 5899 6145 6293
FZULXM	001	0002	6301	6154* 6160 6164 6164 6166 6166 6302
FZUNCT	001	0003	5905	5733 5759 5792
FZUPAD	001	39F8	5687	5470* 5471 5527*
FZUPCT	001	39F6	5686	5548*
FZUPDA	002	39F4	5673	5638
FZUPPL	001	39F5	5680	5549 5686 5687
FZUPP2	001	3CF5	6291	6271
FZUPRT	001	3800	5226	5411 5412 5413 5414 5416 5417 5418 5419
FZURPL	001	39F9	5689	5531
FZUSCT	001	3AFB	5912	5748* 5787* 5824 5851* 5857 5865* 5871
FZUSIN	001	3AF8	5907	5748
FZU010	004	380B	5230	5228* 5229*
FZU020	003	380F	5231	5230*
FZU030	004	3812	5241	5412
FZU040	004	3819	5251	5416
FZU050	004	3830	5267	5417
FZU055	003	3837	5272	
FZU060	004	3849	5279	
FZU065	004	384D	5280	5272* 5273*
FZU070	003	3851	5281	5275*

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 504

SYMBOL	LEN	VALUE	DEFN	REFERENCES
FZU080	004	3857	5293	5413
FZU090	004	3860	5306	5414 5418
FZU100	004	3867	5312	
FZU110	004	3875	5319	5313
FZU120	004	3888	5329	5350
FZU130	004	38A6	5337	5332* 5333 5333*
FZU135	004	38AA	5338	5334*
FZU140	004	38AE	5342	5307
FZU145	004	38B4	5349	
FZU150	004	38BE	5362	5419
FZU160	004	38C4	5373	5352
FZU170	004	38CF	5379	5411
FZU180	004	38DF	5387	5296 5380
FZU190	004	38EF	5395	5243 5283 5315 5375 5388
FZU200	005	3900	5470	5294
FZU210	003	390D	5483	5343
FZU220	004	3916	5491	
FZU230	004	3924	5498	5492
FZU240	004	392B	5503	5485
FZU250	004	3934	5517	5363
FZU260	004	393A	5522	5506
FZU270	005	3941	5527	5499
FZU280	003	3946	5531	5473
FZU290	004	3959	5540	
FZU3B1	003	3AE6	5891	5872
FZU3B2	001	3AEF	5892	5825
FZU3M2	002	3AF1	5893	5822
FZU300	004	3960	5545	
FZU310	003	396F	5553	5541
FZU320	004	3972	5558	5494 5523
FZU330	003	3976	5578	5483 5546
FZU340	003	397D	5584	5610
FZU350	003	3995	5593	5589
FZU360	003	39AA	5601	5591
FZU370	003	39B6	5609	5596 5599 5602
FZU380	004	39BC	5615	5585 5587 5594 5598 5604
FZU390	004	39C0	5616	5578*
FZU4B1	001	3BFB	6106	5962 6024 6033
FZU4B2	001	3BFC	6107	6092
FZU4D5	001	3BFF	6109	6026
FZU4M2	002	3BFE	6108	6089
FZU4XZ	003	3BB8	6111	6014 6071
FZU400	003	39C4	5634	5533 5535 5553
FZU410	004	39C7	5638	
FZU420	005	39D0	5643	
FZU430	003	39D8	5648	
FZU440	004	39DB	5650	5638* 5639* 5643 5655
FZU450	002	39E0	5651	5648*
FZU460	005	39E1	5655	
FZU470	004	39E9	5660	5644
FZU480	004	39EF	5665	5634* 5656
FZU5B1	001	3CEE	6284	6252
FZU5D1	001	3CF4	6288	6160
FZU5M2	002	3CF1	6286	6234
FZU5M5	002	3CF3	6287	6143
FZU5XM	004	3CCD	6299	6147* 6148* 6155* 6156* 6162

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 505

SYMBOL	LEN	VALUE	DEFN	REFERENCES
FZU5XZ	003	3C8B	6290	6148 6188
FZU500	004	3A00	5731	5504
FZU510	003	3A0F	5739	
FZU520	004	3A15	5744	5773
FZU530	004	3A1F	5748	5754
FZU540	003	3A26	5753	5740
FZU550	003	3A2C	5758	5735* 5746* 5763
FZU560	003	3A33	5761	5749
FZU570	003	3A3C	5768	
FZU580	003	3A42	5771	
FZU590	003	3A4B	5777	5769
FZU595	004	3A55	5783	5778
FZU600	005	3A67	5792	5772 5784
FZU610	004	3A70	5804	
FZU620	004	3A82	5814	
FZU630	003	3A8B	5822	5807 5809
FZU640	004	3A9C	5827	5824* 5825*
FZU650	004	3AA0	5831	5826
FZU660	004	3AAA	5843	5518
FZU670	005	3AB1	5849	
FZU680	003	3AC0	5857	5832
FZU690	003	3ACD	5865	5859
FZU700	004	3AD0	5871	5863
FZU705	003	3AE4	5877	5891
FZU710	004	3AE7	5879	5871* 5872* 5873 5873*
FZU720	004	3AEB	5883	5817 5833 5844
FZU730	004	3B00	5944	5815
FZU735	003	3B04	5945	
FZU740	003	3B11	5952	
FZU750	003	3B1D	5959	5948
FZU760	003	3B32	5969	5960
FZU770	003	3B3E	5977	5970
FZU780	006	3B44	5983	5953 5955 5963 5972
FZU785	004	3B50	5989	
FZU790	005	3B5E	6002	
FZU800	004	3B6A	6011	
FZU810	003	3B7D	6017	6005
FZU820	004	3B83	6023	
FZU825	004	3B8B	6026	6002* 6011* 6012* 6014* 6017 6023 6023* 6024*
FZU828	004	3B92	6031	
FZU830	004	3B9D	6043	6013 6015 6018 6027
FZU835	005	3BA4	6050	
FZU840	003	3BB7	6058	6111
FZU845	004	3BC2	6070	6044
FZU846	004	3BCD	6074	6070* 6071*
FZU850	004	3BD4	6079	5992 6072
FZU860	004	3BDD	6088	5965 5978 6051 6053 6060 6075
FZU865	004	3BF3	6094	6091* 6092*
FZU870	004	3BF7	6098	6082 6093
FZU880	004	3C00	6131	6080
FZU885	004	3C07	6142	
FZU888	003	3C21	6154	
FZU890	004	3C2C	6160	6150
FZU892	003	3C33	6162	6161* 6165 6165* 6167
FZU894	004	3C3D	6165	6163
FZU900	004	3C48	6177	6132 6149

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 506

SYMBOL	LEN	VALUE	DEFN	REFERENCES
FZU902	004	3C4F	6186	
FZU904	004	3C65	6194	6189
FZU910	004	3C6C	6199	6211
FZU912	004	3C70	6200	6186 6186* 6191* 6194* 6207* 6208 6210
FZU914	004	3C74	6201	6178*
FZU916	004	3C78	6207	6192 6195
FZU920	003	3C8A	6221	6209 6290
FZU925	003	3C96	6229	6222
FZU930	004	3C99	6233	
FZU940	004	3CA0	6238	
FZU950	004	3CB4	6248	6239
FZU960	003	3CC3	6258	6251
FZU962	004	3CCA	6260	6187* 6188* 6191 6194 6223* 6248 6248* 6252* 6265 6270 6299
FZU964	004	3CCE	6261	6258*
FZU970	003	3CD2	6265	
FZU972	004	3CE6	6272	6270* 6271*
FZU990	004	3CEA	6276	6242 6244 6266 6269
FZVAAC	002	49E9	9479	9369 9412
FZVALN	002	49E5	9477	9382
FZVART	001	4700	8918	
FZVATB	001	48C5	9287	9234
FZVBAS	004	49F3	9488	9377* 9382* 9383 9416* 9422* 9423
FZVBEX	001	4BD3	9824	9606 9777* 9778* 9783* 9789* 9790*
FZVBS1	002	00C0	9651	9589
FZVBS2	002	00C1	9652	9572
FZVB00	001	4BD6	9831	9726* 9740* 9742* 9839
FZVB01	001	4BD7	9832	9738* 9743*
FZVB02	001	4BD8	9833	9744*
FZVB03	001	4BD9	9834	9840
FZVB04	001	4BDA	9835	9841
FZVB19	001	4BE9	9836	9727*
FZVB20	001	4BEA	9837	9682*
FZVB21	001	4BEB	9838	8961 8963 9681* 9682
FZVCAD	001	0700	9079	9110 9119 9479 9854 9855 9856 9857
FZVCAT	001	09FE	9857	9350
FZVCLN	002	49E7	9478	9422
FZVCNT	001	4BD2	9823	9704* 9710* 9750* 9755*
FZVCVT	001	098A	9855	9177 9278
FZVDAC	002	4BD5	9825	9604* 9608 9610 9610*
FZVDC0	001	4BD1	9819	9738
FZVDC1	001	4ABF	9632	9573
FZVDGT	001	48C2	9282	9198* 9206* 9225* 9226
FZVD00	001	00D6	9839	9703 9724
FZVD01	001	4BCA	9814	9604
FZVD03	001	00D9	9840	9748 9772
FZVD04	001	00DA	9841	9749
FZVEND	001	0039	9267	9151 9174 9319 9344
FZVEXP	001	4BE2	9853	9608* 9782* 9788*
FZVFBR	001	4BD6	9826	9831 9832 9833 9834 9835 9836 9837 9838 9853
FZVHLD	001	49EB	9485	9320* 9332* 9345* 9357* 9454
FZVHX1	001	4BCB	9815	9710 9723 9755 9778
FZVHX2	001	4BCC	9816	9770 9771
FZVH00	001	0000	9072	
FZVH01	001	48BD	9276	9162 9184 9206 9208
FZVH02	002	48BF	9277	9163 9185 9205
FZVH09	001	0009	9265	9198

VER 15, MOD 00 31/05/21 PAGE 507

FZVH28	001	001C	9266	9152	9175	9197	9320	9345			
FZVIN1	002	49E1	9475	9332	9357	9370	9378	9394	9399	9432	
FZVI00	002	49DF	9474	9327	9352	9371					
FZVI01	001	47DA	9084	8971							
FZVI02	002	49E3	9476	9333	9358						
FZVLDT	002	48C1	9278	9196							
FZVLST	001	FFFF	9269	9252							
FZVLVT	001	070C	9854	9155							
FZVL01	001	0001	9469	9418							
FZVL02	001	0002	9470	9376							
FZVL04	001	0004	9625	9417	9524	9528	9573	9577	9577	9578	9578
FZVL05	001	0005	9626	9537	9538						
FZVL08	001	0008	9074								
FZVL10	001	000A	9627	9545	9546	9553	9554				
FZVL12	001	000C	9628	9449	9450	9555					
FZVL22	001	0016	9810	8961	8963	9682					
FZVL32	001	0020	9075								
FZVL33	001	0021	9076	8943							
FZVL34	001	0022	9077	8968							
FZVNAT	001	09C4	9856	9325							
FZVNZX	001	4BCD	9817	9783							
FZVPAC	001	4ACA	9639	9520*	9644	9645	9646	9647	9648	9649	9650
FZVPDA	002	47DC	9085	9033							
FZVPG1	001	4700	8908	8912							
FZVPG2	001	4800	9136	8948	9140						
FZVPG3	001	4900	9304	9214	9308	9511					
FZVPG4	001	4B00	9668	8953	8957	9602	9672	9839	9840	9841	
FZVPG5	001	4A00	9504	9441	9447	9508	9651	9652			
FZVPPL	001	47DD	9090	8973*	8995						
FZVP01	001	4ACB	9644	9543	9551						
FZVP04	001	4ACE	9645	9524*							
FZVP05	001	4ACF	9646	9525*							
FZVP06	001	4AD0	9647	9535							
FZVP09	001	4AD3	9648	9528*							
FZVP10	001	4AD4	9649	9529*	9538*	9546*	9554*				
FZVP11	001	4AD5	9650	9449*	9450	9530*	9537	9545	9553	9555	
FZVRPL	001	47E1	9097	8990							
FZVSAC	004	4AC9	9638	9573*	9577	9578	9578*				
FZVSAV	001	47E5	9106	8923	8925*	9000	9002*				
FZVSCN	001	0006	9073	9109	9118						
FZVSDA	001	0959	9080	9108							
FZVSLN	001	0002	9264	9240	9283						
FZVSSA	002	48C4	9283	9153*	9176*	9219*	9226*	9236*	9240		
FZVSUB	004	4AC5	9637	9524	9528	9577*					
FZVSYM	001	47EB	9114	8931							
FZVTBD	001	0041	9078	9117							
FZVTYP	001	49EA	9484	9321*	9346*	9445	9455				
FZVUXL	001	0004	9809	9782	9818						
FZVWSC	002	4AC1	9636	9522*	9526*	9651	9652				
FZVWS1	002	49ED	9486	9370*	9376	9376*	9399*	9400	9522		
FZVWS2	002	49EF	9487	9378*	9392	9394*	9417	9417*	9418*	9432*	9433
FZVXIM	004	4BD1	9818	9782							
FZV010	003	4703	8923								
FZV015	002	4711	8927	8924*							
FZV020	003	4712	8931								
FZV025	002	471D	8934	8932*							

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 508

SYMBOL	LEN	VALUE	DEFN	REFERENCES
FZV030	004	472A	8942	
FZV034	004	4B00	9677	
FZV035	003	4B04	9681	
FZV036	005	4B0B	9686	
FZV037	002	4B15	9688	9686*
FZV040	003	4B16	9692	
FZV045	003	4B1C	9703	
FZV048	004	4B24	9705	9703* 9723* 9724 9725
FZV050	003	4B32	9715	
FZV052	001	4B3E	9718	9715* 9716*
FZV055	004	4B40	9723	9711
FZV060	004	4B53	9738	9693
FZV069	003	4B57	9739	
FZV070	004	4B63	9743	9741
FZV075	003	4B6A	9748	
FZV077	004	4B8D	9765	9748* 9756 9770* 9772
FZV078	004	4B91	9766	9749* 9771*
FZV079	003	4B75	9754	9773
FZV080	003	4B7F	9757	
FZV082	001	4B8B	9760	9757* 9758*
FZV085	004	4B95	9770	
FZV090	004	4BA3	9777	
FZV095	004	4734	8947	
FZV1LN	001	0095	9272	9250
FZV1PG	001	0B07	9270	9250*
FZV100	003	4800	9151	
FZV105	004	480D	9155	9164
FZV110	003	4811	9156	9151* 9163*
FZV115	004	481C	9162	
FZV120	003	4827	9174	
FZV125	004	4830	9177	9186
FZV130	003	4834	9178	9174* 9185*
FZV135	004	483F	9184	
FZV140	003	4890	9234	9158 9180 9221
FZV145	003	4893	9235	9152* 9162* 9175* 9184* 9197* 9208* 9218*
FZV150	005	489A	9240	
FZV155	004	48A3	9248	8938
FZV160	003	4BB6	9788	
FZV165	004	4BBC	9790	
FZV168	004	4BC0	9794	9784
FZV170	004	4A9B	9603	9795
FZV175	003	4AA6	9606	9605* 9609 9609* 9611
FZV180	004	4AB0	9609	9607
FZV190	004	4BC6	9799	9728
FZV2LS	001	FEFF	9268	9249
FZV2PG	001	0C07	9271	9253*
FZV200	004	484A	9196	
FZV205	003	4851	9198	9209
FZV210	004	4854	9199	9196* 9205* 9207
FZV215	003	4889	9225	9201
FZV220	004	4860	9205	
FZV230	004	4872	9213	
FZV240	005	4878	9218	
FZV300	003	4900	9319	
FZV305	004	4909	9325	9334
FZV310	003	490D	9326	9319* 9333*

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 509

SYMBOL	LEN	VALUE	DEFN	REFERENCES
FZV320	004	4916	9332	9384 9401
FZV330	003	4921	9344	
FZV335	004	492A	9350	9359
FZV340	003	492E	9351	9344* 9358*
FZV350	004	4937	9357	9424 9434
FZV400	003	4942	9369	9328
FZV410	004	4950	9376	
FZV415	004	4954	9377	9372
FZV420	004	4958	9378	9402
FZV430	004	495C	9382	9395
FZV440	004	496B	9392	
FZV450	004	4972	9394	
FZV460	004	4979	9399	9393
FZV470	003	4987	9412	9353
FZV480	004	498A	9416	
FZV490	004	4995	9422	9435
FZV510	004	49A4	9432	
FZV520	004	49B2	9439	9385 9425
FZV525	003	49BC	9445	
FZV530	005	49D0	9454	9446
FZV540	004	49DA	9459	
FZV600	003	4A00	9520	
FZV610	003	4A0B	9523	
FZV620	003	4A19	9527	
FZV630	004	4A26	9534	9539
FZV640	004	4A30	9537	
FZV650	003	4A3B	9543	9536 9547
FZV660	004	4A41	9545	
FZV670	003	4A4C	9551	9544
FZV675	004	4A52	9553	
FZV680	004	4A5A	9555	9552
FZV700	003	4A66	9567	9523 9527
FZV705	003	4A69	9571	
FZV710	003	4A73	9574	9590
FZV720	003	4A76	9575	9572* 9574* 9579 9579* 9580 9589*
FZV730	004	4A80	9578	9576
FZV740	003	4A8B	9584	9571* 9588*
FZV750	004	4A97	9594	9567* 9584
FZV775	004	473A	8952	
FZV780	004	4744	8959	
FZV785	005	4753	8963	8960
FZV790	004	4758	8967	8962
FZV795	003	475F	8969	8968* 8971* 8972 8973
FZV798	004	476C	8973	8970
FZV800	006	4770	8983	
FZV810	003	4782	8990	8984
FZV820	003	4788	8995	8986
FZV830	003	478B	8996	
FZV840	003	478E	9000	
FZV850	004	4797	9003	
FZV855	002	479C	9004	9001*
FZV860	004	47A3	9010	8919*
FZV900	003	47AB	9029	8991 8996
FZV910	004	47AE	9033	
FZV920	005	47B7	9038	
FZV930	003	47BF	9043	

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 510

SYMBOL	LEN	VALUE	DEFN	REFERENCES
FZV940	004	47C2	9045	9033* 9034* 9038 9050
FZV950	002	47C7	9046	9043*
FZV960	005	47C8	9050	
FZV970	004	47D0	9055	9039
FZV980	004	47D6	9060	9029* 9051
FZXBCA	001	0DC8	3461	1935* 1940 1953 1999
FZXBKT	001	31CF	3188	3192 3193 3194 3196
FZXBLK	001	31E2	3190	3083
FZXBLN	002	2CE5	2248	2098* 2102 2107 2107*
FZXBPT	001	00FF	3463	2456* 2457* 2458 3036 3157*
FZXBVA	002	2B91	2021	1933 2010
FZXB10	001	32E6	3437	3352
FZXCNT	001	0D56	2615	2451* 2464*
FZXCNV	001	3100	3033	6635
FZXCRP	001	2CFB	2277	2179
FZXCRR	001	31E1	3194	3083* 3098 3115
FZXCR1	001	31D0	3193	3084
FZXDAC	004	2CE9	2249	2100* 2104 2109 2109*
FZDXLN	004	2CED	2252	2099* 2104*
FZXDTC	001	2EF9	2607	2449* 2550* 2557* 2561 2564*
FZXTM	001	0080	2608	2561 2564
FZDXL	001	0002	2990	2903 2991 3444
FZXECA	004	2C68	2234	2183
FZXELN	001	2CEA	2251	2137
FZXER0	001	00F0	3465	2488 2560 2663
FZXER1	001	00F1	3466	2563 2699
FZXER2	001	00F2	3467	2501 2569
FZXER3	001	00F3	3468	2441
FZXER4	001	00F4	3469	2429
FZXETS	001	0D58	2610	2410* 2443* 2653 2654
FZXEVA	002	2CE0	2233	2118 2156
FZXEXP	001	32EA	3450	3262* 3288* 3303* 3359* 3361* 3368*
FZXGCS	001	2E00	2406	
FZXICA	001	0003	3206	3143
FZXICB	001	31D0	3196	3048* 3053* 3130* 3137
FZXICC	001	2FB6	2789	2732*
FZXICL	001	0004	2775	2726 2731 2739 2741 2743 2778 2781 2790
FZXICN	001	0001	3205	3137
FZXICR	004	2FB9	2791	2733
FZXICT	001	31E3	3208	3135
FZXICW	004	2FB9	2790	2726 2726* 2731 2731* 2739 2741 2743 2791
FZXIEX	003	2FAD	2778	2739
FZXIPI	002	2FB1	2781	2741
FZXIP1	001	2B00	1929	
FZXIP2	001	2B6E	1995	
FZXIS2	004	2FB5	2783	2743
FZXITL	001	0004	3204	3135 3136
FZXLVA	001	2B8F	2020	1963
FZXMIS	001	2E17	2424	6605
FZXMNR	001	32F1	3452	3263* 3305 3375
FZXMN1	001	32EB	3451	3264 3366
FZXPDA	002	2CDE	2231	2192
FZXPEM	001	2C18	2094	6614
FZXPNP	001	2CF7	2270	2128
FZXPQ1	001	2C00	2064	6541
FZXPQ2	001	2C06	2073	6557

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 511

SYMBOL	LEN	VALUE	DEFN	REFERENCES
FZXPRP	001	2CEF	2256	2077* 2083
FZXPSA	002	2B56	1969	1966
FZXPSP	001	2CF3	2263	2138* 2139
FZXP1B	001	2B00	1916	1920
FZXP2B	001	2C00	2049	2053
FZXP3B	001	2D00	2294	2056 2233 2298
FZXP4B	001	2E00	2393	2397
FZXP5B	001	2F00	2644	2648
FZXP6B	001	3000	2813	2817
FZXP7B	001	3100	3020	3024 3084 3098
FZXP8B	001	3200	3253	3257 3264 3305
FZXQML	001	0002	2240	2258
FZXQM1	001	2CE2	2238	2066
FZXQM2	001	2CE1	2236	2075
FZXSEC	001	00FF	2613	2411 2748 2943
FZXSER	001	0D59	2612	2450* 2471 2748* 2943*
FZXSGN	001	32F1	3453	3324* 3369*
FZXSTC	001	2EF8	2605	2448* 2566*
FZXSTP	005	2EC0	2604	2447* 2558*
FZXSTS	001	31CF	3192	3082* 3101*
FZXXCL	001	0002	2979	2822 2844 2858 2924 2926 2936 2938 2980 2981 2982 2988
FZXXCT	002	30EA	2988	2822* 2844* 2858* 2924* 2926* 2936 2938
FZXXHI	002	30E6	2981	2936
FZXXLO	002	30E8	2982	2938
FZXXZR	002	30E4	2980	2822
FZX010	004	2B1E	1946	1986
FZX020	004	2B57	1974	1961
FZX030	004	2B8B	2014	2008
FZX050	003	2C00	2066	1947
FZX060	003	2C06	2075	
FZX070	003	2C09	2077	2067
FZX080	005	2C18	2098	1984
FZX090	003	2C28	2102	2101* 2105 2105* 2108* 2110
FZX100	004	2C32	2105	2103
FZX110	004	2C40	2109	2106
FZX120	004	2C66	2133	2121* 2145 2234
FZX130	003	2C82	2147	2149
FZX140	004	2C9D	2161	2086
FZX150	003	2CA1	2179	2081 2114
FZX160	004	2CA4	2183	2126 2135 2150
FZX170	003	2CA8	2188	2084 2129 2140
FZX180	004	2CBF	2204	2192* 2193* 2197 2209
FZX190	002	2CC4	2205	2202*
FZX2D0	001	2CDB	2228	2099
FZX2D1	001	2CDC	2229	2100 2148
FZX200	004	2CCD	2214	2198
FZX210	004	2CD7	2220	2188* 2210
FZX250	004	2E00	2410	
FZX260	003	2E17	2428	
FZX270	003	2E20	2440	1975
FZX280	004	2E26	2443	2430
FZX290	005	2E30	2447	2416
FZX300	003	2E4F	2462	2479
FZX310	005	2E55	2464	2486
FZX320	003	2E6D	2479	2412* 2428* 2440*
FZX330	003	2E86	2495	2477

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 512

SYMBOL	LEN	VALUE	DEFN	REFERENCES
FZX340	003	2E93	2502	2413* 2444*
FZX350	004	2E96	2505	2429* 2441*
FZX360	004	2EA4	2514	2489 2502
FZX370	003	2E9A	2508	2415* 2472 2491 2496 2516*
FZX375	003	2EA8	2516	2508 2510
FZX380	003	2EAF	2541	2462 2483 2500
FZX390	003	2EB2	2542	2414* 2445*
FZX4B1	002	2EF7	2598	2457 2464 2550 2558 2566
FZX400	004	2EB5	2550	2567
FZX410	005	2EBC	2557	2604
FZX420	004	2ED4	2566	2562
FZX430	004	2EDE	2573	2551 2560* 2563* 2569*
FZX440	004	2EE2	2575	2541* 2542
FZX450	003	2EE6	2584	2463 2484
FZX460	003	2EE9	2586	2588
FZX470	004	2EF2	2590	2584*
FZX5M1	002	2FA9	2773	2684
FZX500	005	2F00	2653	2469
FZX510	003	2F14	2664	2653*
FZX520	003	2F20	2672	2670
FZX530	004	2F35	2699	2659
FZX540	003	2F39	2700	2654*
FZX550	003	2F3C	2705	2709 2712
FZX560	004	2F5D	2726	2674
FZX570	003	2F61	2728	2734
FZX580	004	2F7B	2739	2730
FZX590	004	2F90	2748	2664 2700 2707
FZX6BX	002	30EC	2992	2924 2926
FZX6B1	002	30E2	2977	2844 2858
FZX6DX	002	30EC	2991	2903* 2906 2906* 2907* 2912* 2913* 2914 2914* 2915 2915* 2916 2916*
				2917 2917* 2918 2918* 2919* 2992
FZX600	004	2F94	2750	2689 2716 2740 2742 2744
FZX610	003	2F98	2759	2672 2715 2728 2735
FZX620	003	2F9B	2761	2763
FZX630	004	2FA4	2765	2759*
FZX650	004	3000	2822	2687
FZX660	003	300D	2835	2836
FZX670	003	3016	2841	
FZX675	004	301C	2844	2846
FZX680	003	3026	2850	2837
FZX690	003	302F	2857	2823* 2841* 2860
FZX7B1	003	31C4	3182	3101 3103
FZX700	003	3036	2859	2852 2857
FZX710	003	3042	2867	2868
FZX720	003	3048	2872	2824* 2851 2861 2967*
FZX730	003	3066	2894	2890
FZX740	003	3069	2898	2892
FZX750	003	3084	2912	2905
FZX780	003	30A1	2923	2887* 2893*
FZX790	004	30AB	2926	2923
FZX8BK	001	32EA	3447	3450 3451 3452 3453
FZX8BX	002	32E9	3445	3359 3361
FZX8B1	003	32D8	3436	3288 3303 3308 3353
FZX8DX	002	32E9	3444	3343* 3344 3344* 3345* 3352* 3353* 3445
FZX8D0	001	32E7	3438	3263
FZX800	003	30AF	2931	2825* 2842* 2865* 2883 2925

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 513

SYMBOL	LEN	VALUE	DEFN	REFERENCES
FZX810	004	30C0	2943	2872 2899 2937
FZX820	004	30C4	2945	2931 2939
FZX830	003	30C8	2959	2835 2845 2859 2867 2888 2894 2904 2908
FZX840	003	30CB	2961	2963
FZX850	004	30DD	2969	2959* 2966
FZX860	003	3100	3035	2002
FZX863	003	3124	3055	3050
FZX866	003	3127	3056	3052
FZX870	004	312D	3067	3047*
FZX873	003	313A	3082	3042
FZX876	003	3144	3088	3099 3104
FZX880	003	3156	3098	3090
FZX883	004	3160	3102	3084* 3098 3103*
FZX886	003	316B	3108	3093
FZX890	004	317C	3116	3113*
FZX893	003	3183	3129	3057
FZX896	003	3190	3136	3138
FZX900	004	31A7	3149	3131*
FZX903	003	31AB	3150	3152
FZX906	004	31B4	3156	3035* 3072 3117
FZX910	003	31BF	3168	3037 3055 3109 3129 3150
FZX913	003	31C2	3170	3172 3182
FZX916	004	31CB	3174	3168*
FZX920	003	3200	3262	3070
FZX923	003	320D	3276	3277
FZX926	003	3213	3281	
FZX930	004	321F	3288	3290
FZX933	003	3223	3289	3287
FZX936	003	3229	3294	3318
FZX940	003	322C	3295	3283
FZX943	003	3232	3302	3281* 3294* 3311
FZX946	003	3239	3305	3302
FZX950	004	323F	3307	3264* 3305 3308*
FZX953	003	3247	3310	3306
FZX956	003	3259	3322	3296 3316
FZX960	003	3262	3334	3323
FZX963	004	3271	3344	3347
FZX966	004	3282	3352	3354
FZX970	004	3286	3353	3348
FZX973	003	328D	3358	3322 3403* 3408*
FZX976	004	3297	3361	3358
FZX980	003	329B	3366	3335 3360
FZX984	003	32A7	3373	3367
FZX986	004	32B6	3380	3373*
FZX990	003	32BE	3401	3265 3338
FZX992	003	32D3	3421	3276 3289 3310 3317 3337 3346
FZX994	003	32D6	3423	3405 3425 3436
FZX996	003	32DF	3427	3407
FZX998	004	32E2	3428	3401* 3421*
FZZBM1	001	00FF	0187	0067
FZZBN1	001	4CB3	0159	0033 0069 0143
FZZCDT	002	4CB8	0164	0089 0090
FZZCNT	002	4CBC	0172	0089* 0090* 0103 0112 0118
FZZDPL	001	4CBD	0175	0013* 0019* 0047 0081* 0082* 0091* 0093* 0103* 0107 0107* 0108 0108*
				0114* 0120* 0130* 0131* 0135
FZZHCA	002	4CBA	0170	0025* 0034* 0130

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 514

SYMBOL	LEN	VALUE	DEFN	REFERENCES
FZZIDM	001	0080	0193	0112
FZZITM	001	0040	0194	0118
FZZLOK	001	0001	0190	0061
FZZLRT	001	0000	0189	0053 0055* 0061
FZZMDY	001	0002	0191	0053 0055
FZZNST	001	4CB6	0163	0091 0093
FZZPGB	001	4C00	0000	0004
FZZSDM	001	0001	0195	0114
FZZSTM	001	0080	0196	0120
FZZSXA	002	4CB5	0161	0025
FZZVPL	001	4C06	0018	
FZZVPS	001	4C00	0012	
FZZ005	003	4C09	0024	0014
FZZ010	004	4C25	0042	0030 0144
FZZ020	003	4C29	0043	0024* 0032* 0033* 0070 0131 0143*
FZZ025	003	4C3E	0061	0048
FZZ030	003	4C44	0067	0056
FZZ035	004	4C4B	0069	0071
FZZ040	004	4C4F	0070	0067* 0069* 0082
FZZ050	004	4C61	0090	0092
FZZ060	003	4C85	0118	0113
FZZ070	004	4C8E	0130	0119
FZZ080	002	4CA1	0138	0136*
FZZ090	004	4CA2	0143	0054 0062
GHS860	004	1689	7150	
I\$ADJX	001	0D56	3718	5990* 6054* 6147 6156
I\$ADST	001	0C9D	3673	8399
I\$BASE	001	0C60	3675	8398 1967 6596
I\$BRCN	001	117B	3727	
I\$BSET	001	119D	3726	1968 6597
I\$B1SW	001	0040	3783	5251 5387
I\$B2SW	001	0020	3785	5306 5319 5379 7270
I\$CADR	001	144C	3764	9266* 9267 9318 9354 9526 9658 9702 9723* 9751 9764 9765 9803
				9926 0097 1273 1467 1935 2120 4443 5256 5324 6536 0309 0346
				0384 0462 0463
I\$CALL	001	12B1	3758	4153 4161 4206 4501 4512 4797 4814 5214 5231 5639 5664 5696
				6211 6215 6220 6224 6370 6382 6394 6674 6712 6868 6925 6965
				6991 7310 7334 8794 8837 8859 8876 8898 8920 8959 8976 9396
				9561 9575 9617 9721 9927 0134 0201 0206 0218 0261 0323 0509
				0521 0622 0690 1128 1946 1954 1974 1983 2001 2215 2468 2686
				3069 4058 4139 4585 4618 5293 5342 5362 5503 5517 5660 5814
				6079 6540 6556 6585 6604 6613 6634 6944 6968 6997 7254 7258
				7266 7293 7596 8390 8414 8710 8937 8947 8952 9055 9213 9440
				9794 0647 0652
I\$CBM1	001	0D43	3694	
I\$CBN1	001	0D3E	3690	
I\$CBN2	001	0D3F	3691	
I\$CBN3	001	0D40	3692	
I\$CBN4	001	0D41	3693	
I\$CFBS	001	0AE3	3741	8428
I\$CLFA	001	0D4A	3700	
I\$CLVA	001	0D49	3699	8799 9203
I\$CL1C	001	0D46	3697	8776 9323
I\$CL1F	001	0D44	3695	8426 8764 9257
I\$CL2C	001	0D47	3698	
I\$CL2F	001	0D45	3696	

VER 15, MOD 00 31/05/21 PAGE 515

I\$CPG1	001	1600	3655													
I\$CPUF	001	0A27	3737	9559	9856	7281	7463	7470	7633	7637	7812					
I\$CSCT	001	0D5A	3713	5792*												
I\$CSSW	001	0010	3787	5491	5804											
I\$CSXA	001	2000	3654	2231	4158	5673	8734	9085	0161							
I\$CUPF	001	0A85	3739	8767	9259	3376	7481	7652	7823							
I\$CVAD	001	1358	3752	9353	9750	1272	4442	0345	0383							
I\$DATA	001	0D53	3681	8943*	3642	3654	3689*	3712*	3726*	3727*						
I\$DAT1	001	0D55	3682	8943												
I\$DMSW	001	0BC1	3735	6898*	6932*											
I\$ECSW	001	0004	3791	5732	5786	6004	6043	6131								
I\$ERRC	001	0CBC	3680	4137*	4806*	5124*	5160*	5179	5225	5233	5235*	5364*	5624*	6208*	6662*	
				6870	6872*	6881	7315	7317*	7325*	7339	7341*	7349*	8432	8437*	8812	
				9213*	9223*	9247*	9336*	9340*	9358*	9534*	9698*	0105*	0213*	0361*	1980	
				2148*	2152*	2273	2411*	2488	2501	2505*	2509*	2514*	2573*	2663	2699	
				3647*	3707*	5314*	5493*	6610	6949	6973	7002	7297	7476	7499*	7598	
				7643	7723*	7792*	7818	7924*	7928*	8070*	8234*	8356*	8359*	8363*	8366*	
				8382*	8386*	8401*	8405*									
I\$FACT	001	0DD1	3720													
I\$FADD	001	075D	3743	5692	5760	6360	6889	6956	6971	7054	7065	7087	7134	7146	7158	
				7516	7642											
I\$FATE	001	0DE6	3721	9291												
I\$FATP	001	0DE8	3722	9203*	9204	9218	9228									
I\$FDVD	001	0919	3748	6218	6227	6366	6691	6708	6880	6983	7060	7093	7163	7367	7859	
				8072												
I\$FMPY	001	082A	3746	4214	5178	5224	5630	5670	5743	5754	5765	6670	6896	7123	7129	
				7139	7844	7639	7817									
I\$FSUB	001	0751	3744	6364	6672	6722	6978	7151	7515							
I\$FWRK	001	0607	3664	4143*	4194	4196*	4197	4251*	4252	4252*	4253	4258	4279	4288*	4296*	
				4310*	4311	4311*	4321	4343	4343*	4664*	4666	4865*	4866*	4867	4867*	
				4868*	4894*	4901*	4908*	4926*	4927	4935*	4937	4965	4965*	5174*	5177	
				5899*	5909	5918*	5921	5927	5930	6213*	6217	6222*	6226	6359*	6361*	
				6362	6365	6377*	6407	6408	6445	6458	6474*	6475	8942*	8943	8943*	
				8959	8961*	8963*	8967	9093	9154*	9220*	9240*	9449	9450*	9534		
I\$IMC1	001	0DCE	3711	5731*	5849*	5875	5947									
I\$IMLN	001	0DC6	3707	5257*	5273	5274*	5275	5332								
I\$IMPT	001	0DCC	3710	5330*	5471*	5548	5579	5615*	5793*	6088	6142					
I\$INDR	001	0DC5	3706	5241*	5251*	5306	5312	5319*	5329*	5349	5379	5387	5491	5498*	5540	
				5545	5732*	5786*	5804*	5843	5861*	5961*	6004	6043	6131	6241	6250	
				7270*												
I\$INIT	001	0607	3663													
I\$INTR	001	0C5C	3667													
I\$IRSW	001	0CDE	3687													
I\$I700	001	0E24	3749	1936	6527											
I\$LBFR	001	12B6	3759	0382*												
I\$LDBR	001	1329	3756	0874												
I\$LDXR	001	1330	3757	4877	9915	9940	0121	0276	0329	0464	0598	0896	1367	1468	1486	
				3656	9248	9251	9687	9717	9759	0410	0620					
I\$LOCK	001	1354	3754	9317	9524	0096	0127	0621	0876	0898	1466	1934	2119	5254	5322	
				6535	0385	0461	0609	0622								
I\$MDFY	001	1349	3753	7978	9525	9651	9798	9943	0095	0126	0664	0308				
I\$MOD4	001	130B	3750	1966*												
I\$NCPG	001	000A	3775													
I\$NDSW	001	0002	3793	5329	5349	5498	5545									
I\$NISW	001	0080	3781	5241	5312	5540										
I\$NPAG	001	0C68	3668													

CROSS REFERENCE

SYMBOL LEN VALUE DEFN REFERENCES VER 15, MOD 00 31/05/21 PAGE 516

I\$PARM	001	0D57	3683	7993	8725*	9554*	9560*	2448	2615	3997	4061	4083*	4098*	4125	4126*
				4130*	4274*	4304*	4305*	4338*	4416	4428	4700	4712	5228	5373	5374
				5522	5806	5831	6575*	6650*	7253*	7257*	7264*	7287*	7291*	9157	9179
				9200	9383	9423	9686	9704	9716	9750	9758				

I\$PGDS	001	144A	3762												
---------	-----	------	------	--	--	--	--	--	--	--	--	--	--	--	--

I\$PGNO	001	1449	3761												
---------	-----	------	------	--	--	--	--	--	--	--	--	--	--	--	--

I\$PGTB	001	14CA	3765	0068											
---------	-----	------	------	------	--	--	--	--	--	--	--	--	--	--	--

I\$PLRT	001	15E2	3766	0042											
---------	-----	------	------	------	--	--	--	--	--	--	--	--	--	--	--

I\$PSTK	001	15CA	3767	1969											
---------	-----	------	------	------	--	--	--	--	--	--	--	--	--	--	--

I\$PUB1	001	0DC8	3708	3461	5256*	5279	5335	5470							
---------	-----	------	------	------	-------	------	------	------	--	--	--	--	--	--	--

I\$PUB2	001	0DCA	3709	5324*	5330	5336	5527								
---------	-----	------	------	-------	------	------	------	--	--	--	--	--	--	--	--

I\$RESW	001	0CE9	3688	1965*	6595*										
---------	-----	------	------	-------	-------	--	--	--	--	--	--	--	--	--	--

I\$RNMK	001	0001	3703	7983	7989										
---------	-----	------	------	------	------	--	--	--	--	--	--	--	--	--	--

I\$RNSW	001	0D5C	3702	7983	7989*										
---------	-----	------	------	------	-------	--	--	--	--	--	--	--	--	--	--

I\$RTRN	001	12D3	3760	4219	4317	4514	4669	4943	4988	5242	5448	5698	5766	5934	6064
---------	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------

				6219	6228	6400	6459	6726	6900	6996	7069	7174	7371	7487	7613
--	--	--	--	------	------	------	------	------	------	------	------	------	------	------	------

				7731	7863	8076	8401	8438	8985	9214	9224	9248	9277	9402	9569
--	--	--	--	------	------	------	------	------	------	------	------	------	------	------	------

				9699	9804	9968	0225	0367	0503	0529	0668	0700	0909	1981	2014
--	--	--	--	------	------	------	------	------	------	------	------	------	------	------	------

				2161	2517	2750	2945	3159	3382	3648	3690	3708	4144	4276	4348
--	--	--	--	------	------	------	------	------	------	------	------	------	------	------	------

				4590	4864	5395	5558	5883	6098	6276	6665	6934	7312	7500	7667
--	--	--	--	------	------	------	------	------	------	------	------	------	------	------	------

				7724	7839	7977	8102	8235	8416	8695	9011	9241	9254	9459	9557
--	--	--	--	------	------	------	------	------	------	------	------	------	------	------	------

				9615	9799	0151	0634	0654							
--	--	--	--	------	------	------	------	------	--	--	--	--	--	--	--

I\$SDCT	001	0D59	3715	5962*	5983	6002									
---------	-----	------	------	-------	------	------	--	--	--	--	--	--	--	--	--

I\$SDPT	001	0DD0	3712	5744*	5779*	6199	6233								
---------	-----	------	------	-------	-------	------	------	--	--	--	--	--	--	--	--

I\$SFCT	001	0D5A	3716	5984	6012	6210									
---------	-----	------	------	------	------	------	--	--	--	--	--	--	--	--	--

I\$SFFO	001	0D5D	3724												
---------	-----	------	------	--	--	--	--	--	--	--	--	--	--	--	--

I\$SICT	001	0D5B	3717	5983*	5984*	6050	6059	6074	6238						
---------	-----	------	------	-------	-------	------	------	------	------	--	--	--	--	--	--

I\$SLLC	001	0BA1	3731	8759	4002	4423	4707	5808							
---------	-----	------	------	------	------	------	------	------	--	--	--	--	--	--	--

I\$SLNG	001	0BA2	3730	8299*	9237*	3682*									
---------	-----	------	------	-------	-------	-------	--	--	--	--	--	--	--	--	--

I\$SNSW	001	0001	3795	5732	5961	6241	6250								
---------	-----	------	------	------	------	------	------	--	--	--	--	--	--	--	--

I\$SSCT	001	0D58	3714	5850*	5851	6091									
---------	-----	------	------	-------	------	------	--	--	--	--	--	--	--	--	--

I\$STAK	001	0D4E	3676	4129	4312	4485	4626	4654	4665	4925	4982	5103	5150	5361	5447
---------	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------

				5590	5606	5734	5898	5929	6041	6190	6204	6341	6615	6627	6862
--	--	--	--	------	------	------	------	------	------	------	------	------	------	------	------

				6912	6943	7322	7346	7479	7582	7710	7839	7854	7979	8314	8426*
--	--	--	--	------	------	------	------	------	------	------	------	------	------	------	-------

				8427	8764*	8766	8776*	8799*	8800	8855*	9236	9257*	9258	9323*	9324*
--	--	--	--	------	-------	------	-------	-------	------	-------	------	-------	------	-------	-------

				9325	9329*	9345*	9346*	9347	9348*	9556	9588	9610	9664	9848	0283
--	--	--	--	------	-------	-------	-------	------	-------	------	------	------	------	------	------

				0337	0619	1963*	2447	3114	3144	3374	3684	3991	5267	5876	5944
--	--	--	--	------	------	-------	------	------	------	------	------	------	------	------	------

				5989	6177	6259	6528	6570	6639	6904	6954	6978	6991	7243	7278
--	--	--	--	------	------	------	------	------	------	------	------	------	------	------	------

				7432	7629*	7641*	7700	7789	7921	7942	8064	8182	8354	8381	8400
--	--	--	--	------	-------	-------	------	------	------	------	------	------	------	------	------

I\$STCK	001	0B50	3729	8315	8370	9238	3145	3685	6993	7280	7462	7469	7632	7636	7811
---------	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------

				8086	8219										
--	--	--	--	------	------	--	--	--	--	--	--	--	--	--	--

I\$STHA	001	0D51	3686	1964	6594										
---------	-----	------	------	------	------	--	--	--	--	--	--	--	--	--	--

I\$STKB	001	0639	3665												
---------	-----	------	------	--	--	--	--	--	--	--	--	--	--	--	--

I\$STKI	001	0D4F	3677	8397*											
---------	-----	------	------	-------	--	--	--	--	--	--	--	--	--	--	--

I\$STSW	001	0008	3789	5804	5843	5861									
---------	-----	------	------	------	------	------	--	--	--	--	--	--	--	--	--

I\$TFSW	001	0D28	3689												
---------	-----	------	------	--	--	--	--	--	--	--	--	--	--	--	--

I\$ULNG	001	0C3A	3734	8392*	8808*										
---------	-----	------	------	-------	-------	--	--	--	--	--	--	--	--	--	--

I\$UNLK	001	1350	3755	9375	9399	9568	0210	0224	0366	0667	0905	0907	2011	2157	5383
---------	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------

				5391	6662	0629	0631	0633							
--	--	--	--	------	------	------	------	------	--	--	--	--	--	--	--

I\$USTK	001	0BB0	3733	8393	8813	9260	6641	6956	7483	7653	7825	7963	8088	8221	
---------	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	--

I\$VADR	001	144A	3763	8298*	8391*	8801*	9197*	9208	9229	9256*	9316*	9352*	9374*	9398*	9523*
---------	-----	------	------	-------	-------	-------	-------	------	------	-------	-------	-------	-------	-------	-------

				9567*	9649*	9650*	9749*	9796*	9797*	0094*	0209*	0223*	0365*	0653*	0663*
--	--	--	--	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

				0904*	0906*	1271*	1276*	1277*	1356*	1357*	1465*	1933*	2010*	2118*	2156*
--	--	--	--	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

				3143*	3681*	4441*	5253*	5321*	5382*	5390*	6534*	6640*	6661*	6955*	6979*
--	--	--	--	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

				6992*	7279*	7461*	7468*	7482*	7631*	7634*	7651*	7810*	7824*	7962*	8085*
--	--	--	--	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

				8087*	8218*	8220*	0307*	0344*	0362*	0380*	0392*	0460*	0628*	0630*	0632*
--	--	--	--	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

CROSS REFERENCE

SYMBOL LEN VALUE DEFN REFERENCES VER 15, MOD 00 31/05/21 PAGE 517

I\$WRK1	001	0D59	3684	9276*	9318*	9381	9389*	9609*	9610*	9766*	9768*	9850	0426*	1964*	2610
				2612	4123*	4417	4701	6594*							
I\$WRK2	001	0D5B	3685	9391*	9767*	4134*	4135*	4406	4580	4691	8407*	8958	9218	9219	9439*
				9448	9454*	9455*	9521	9556*	9603	9677*	0650				
I\$XAD1	001	0C89	3672	8981											
I\$XAD2	001	0C82	3671	8925											
I\$XAD3	001	0C7B	3670	8817											
I\$XAD4	001	0C74	3669												
I\$XERR	001	0CAB	3674												
I\$XIAR	001	0D4C	3679	8287	8297	8714	8724	9196	9269						
I\$XPAG	001	0C61	3678	9268											
I@APRC	001	0006	3857	6893	4032	4234	4234*	4243	4252	4255	4255*	4267	4288	4288	4288*
				4296											
I@APRL	001	000B	3834												
I@APRS	001	0006	3811	3857											
I@ASTA	001	0000	3869	7981	7984										
I@ASTL	001	0020	3845												
I@ASTS	001	0000	3822	3869											
I@CMEQ	001	0004	3926												
I@CMHI	001	0008	3927												
I@CMLO	001	0002	3925												
I@DEXP	001	0000	3904	4486	4499	4500*	4511*	4655*	4668*	4804	4848*	4938*	4939*	4985	5167
				5374	5391	5392*	5396*	5397*	5398*	5595	5598*	5610	5622	5634	6195
				6206	6342	6350	6618	6633	6920	4030	4032	4037	4237*	4241	4243
				4253	4260	4290*	4311	4320	6011	6033*	6050*	6052	6054	6058*	6059*
				6070	6187	6221	6229*								
I@ICBA	001	F500	3871	3212	3216	3220	3224	3228	3232						
I@ICBL	001	F000	3847												
I@ICBS	001	F500	3824	3871											
I@IVBA	001	F531	3872												
I@IVBL	001	F049	3848												
I@IVBS	001	F531	3825	3872											
I@LCRF	001	0012	3886	3887	9595	9601	9601	9601*	3083	3194	4085	5272	5280	5857	5865
				5895											
I@LCRV	001	0013	3887	8759	8808	9683	0286	3115	3115*	3189	3682	4002	4423	4707	5280
				5808	9478	9724									
I@LFPZ	001	0012	3956	4115	4118*	4119	4119*	4452	4470	4514	4550	4560	4571	4735	4753
				4797	4833	4843	4854								
I@LPFL	001	0009	3836	3839	3840	9676	9743	0372	0376						
I@LPFS	001	0005	3813	3816	3817	3859	9669	9743	0266	0372	0376				
I@LPFV	001	0005	3859	9289	3212	3216	3220	3224	3228	3232	6675	7014	7321	7518	7688
				7853	7920	7934	7940	7941	7951	7951*	7957	7957*	7982	7985	7987
				7992	7993	8115	8252	9477							
I@LPPZ	001	0003	3955	4465	4523	4632	4748	4806	4910						
I@LPSW	001	0080	3844												
I@LSFV	001	0007	3861	5411*	5412*	5413*	5422	5424	5439	5440	5440	5457			
I@LUFL	001	0010	3837	3841											
I@LUFS	001	0008	3814	3818	3860										
I@LUFV	001	0008	3860	3861	3896	3898	3899	3901	3902	4107	4118	4143*	4143*	4173	4176
				4176	4181	4181	4194	4194	4196	4196*	4197	4197	4213	4229	4237
				4251*	4252	4252	4252*	4253	4258	4258	4258	4279	4279	4279	4279
				4288	4288	4288*	4296	4296	4296*	4296*	4310	4310*	4311	4321	4321
				4321	4321	4328	4343	4343	4343	4343	4343*	4343*	4347	4359	4360
				4361	4362	4363	4364	4365	4366	4367	4652	4833	4835	4835	4855
				4855	4865	4865*	4865*	4894	4894	4894*	4894*	4901	4901	4908	4908
				4908*	4908*	4910	4926	4965	4965	5174	5177	5210	5220	5252	5408

CROSS REFERENCE

SYMBOL LEN VALUE DEFN REFERENCES VER 15, MOD 00 31/05/21 PAGE 518

				5408	5408	5410	5422	5424	5439	5441	5461	5462	5629	5669	5685
				5741	5742	5744	5745	5752	5753	5759	5761	5764	5770	5771	5773
				5899	5899*	5901	5927	5930	6063	6198	6200	6210	6213	6213*	6213*
				6214	6217	6217	6217	6222	6222*	6222*	6223	6226	6226	6226	6358
				6359	6359*	6359*	6361	6361*	6361*	6361*	6362	6362	6362	6363	6365
				6365	6365	6365	6384	6384	6389	6389	6389	6390	6390	6405	6406
				6407	6407	6408	6408	6408	6414	6437	6448	6458	6488	6488	6490
				6497	6498	6499	6500	6640	6648	6657	6668	6669	6671	6690	6700
				6706	6707	6720	6721	6735	6877	6878	6879	6883	6955	6970	6972
				6976	6982	7016	7052	7053	7058	7059	7064	7084	7085	7091	7092
				7118	7122	7124	7128	7133	7138	7140	7144	7145	7150	7152	7156
				7157	7162	7195	7196	7360	7840	7855	7988	8003	8008	8009	8009
				8013	8018	8018	8023	8027	8031	8070	8093	8094	8095	8096	8104
				8105	8106	8107	3375	3375*	3448	7467	7520	7628	7635	7690	7790
				7797	7798	7816	7848	9738	9738*						
I@LXPT	001	0060	3947												
I@MANL	001	0001	3905	4133	4172*	4176*	4642	4987*	5112	5117	5129	5132*	5366	6191	6698
				4023	4236*	5991	6032*	6224*							
I@MANR	001	0007	3906	5122	5143	5149	5158	5189	5191*	5241*	6031	6031*			
I@NCPG	001	000A	3949	0024											
I@NERR	001	0000	3958	5179	5225	5233	6870	6881	7315	7339	8432	8812	1980	2152	2450
				2471	2514	6610	6949	6973	7002	7297	7476	7598	7643	7818	8405
I@NXPG	001	0020	3946	3947											
I@NXPT	001	0003	3945	3947											
I@PEXL	001	0008	3840	9673	9679*	9680	9680*	0347*	0348	0348*	0356				
I@PEXP	001	0004	3864												
I@PEXS	001	0004	3817	3864	9673*	9679	0347	0356*							
I@PMNR	001	0003	3863												
I@PMN1	001	0000	3883	9743											
I@PMRL	001	0007	3839												
I@PMRS	001	0003	3816	3863											
I@PRCL	001	000F	3833	3837	7871										
I@PRCS	001	0007	3810	3814	3856										
I@PREC	001	0007	3856	4143	4194	4234	4236	4253	4254*	4258	4305	4311*	4332	4333	4348
				4490	4629	4637	4664	4664*	4666	4666	4827	4846	4849	4849	4866*
				4867	4867	4867*	4901*	4906	4926*	4937	4937	4950	4965	4965*	5131
				5131	5152	5174*	5177	5201	5379	5386	5409	5441	5457	5468	5595
				5597	5597	5610	5718	5722	5777	5783	5785	5788	5790	5793	5901
				5909	5927	6042	6416	6491	6618	6731	6733	6949	7012	7185	7187
				7189	7191	7364	7588	7595	7608	7721	7840	7855	7999	7999	8040
				8045	8055	8056	9857	9857	9857*	9862*	9872*	9880*	9888	9891*	9895
				3263	4039	6017	6031	6208	6265	6291	7623	9772	9853		
I@PRSW	001	0087	3868	9672	9677	0342	0351	8397							
I@PRTE	001	0000	3943												
I@RSE1	001	0007	3896	4135	4143	4174	4177*	4179	4186*	4188	4489*	4490	4490*	4503*	4664
				4666*	4667*	4849	4849*	4937*	4983	5131	5131*	5174	5379	5386	5409*
				5597	5597*	5614	5617*	5647*	5651	5653*	5654	5656*	5658	5663*	5691*
				5695*	5741	5742	5744	5752*	5761	5899	6042*	6052	6057	6057*	6063*
				6198	6210	6213	6214*	6222	6223*	6359	6361	6362*	6396*	6640	6648*
				6657	6668	6669	6700*	6701*	6706	6707*	6720	6721*	6877	6878*	6949*
				6950*	6955	6972	7052	7058	7059*	7084	7091	7092*	7118	7122	7124
				7140	7145*	7152	7156*	7360	8040*	8046*					
I@RSE2	001	000F	3899	4197*	4198*	4213*	4627	4628*	4820	4827	4833	4835*	4846	4855*	4865
				5169	5177*	5210	5220*	5379*	5380*	5386*	5422*	5439*	5629*	5669*	5685*
				5742*	5753*	5759*	5764*	5900*	5901	5901*	5927*	5928*	5930*	6217*	6226*
				6358*	6363*	6365*	6384*	6389*	6390*	6438*	6471*	6472	6473*	6488	6669*

CROSS REFERENCE															
SYMBOL	LEN	VALUE	DEFN	REFERENCES	VER 15, MOD 00 31/05/21 PAGE 519										
				6671* 6690* 6706* 6720* 6879* 6883* 6893* 6955* 6970* 6976* 6982* 7053* 7058* 7064* 7085* 7091* 7122* 7128* 7133* 7138* 7144* 7150* 7157* 7162* 7360* 7840* 7855* 8070* 7816*											
I@RSE3	001	0017	3902	4636* 4637 4637* 5410*											
I@SCOD	001	0000	3936	6571* 6574*											
I@SGNL	001	000F	3842												
I@SGNS	001	0007	3819	3866											
I@SIDX	001	0001	3937	8356 8365											
I@SIGN	001	0007	3866	5362 6344 6345* 6638 6639* 6655 6656* 6667* 6673* 6885* 7483* 7593											
				7602* 7716 7720* 7727* 7998* 3453 4014 4017* 5945 5946*											
I@SPSW	001	0087	3821	3868											
I@STAT	001	0000	3880	8806 9557 9589 9594* 9665 9674* 9678* 9851 9858* 9863* 9870 9871*											
				9878 9887* 9890* 0338 0346* 0355* 0620* 3192 3193 4077 4100* 4101											
				5268* 5274 9692 9739											
I@SVAD	001	0001	3935	8801 9242 9254* 9255 9256											
I@UEXP	001	0000	3882	3904 6918 6947 6960 7116 7366* 7583 7588 7596 7609* 7723* 8041*											
				8057* 8066* 8071* 3450 7624*											
I@UMNR	001	0007	3865	3906 7364* 7594* 7598 7598* 7607* 7608 7608* 7721 7721* 7999 8055											
				8056* 3452 7623* 7797											
I@UMN1	001	0001	3884	3905 6977* 7086* 7323 7347 7365* 7711 7722* 8050 3451											
I@UMRL	001	000F	3841	3842											
I@UMRS	001	0007	3818	3819 3865											
I@XBRC	001	0003	3919												
I@XCNT	001	0001	3917	8855											
I@XCOD	001	0001	3918	8725 8893 8915											
I@XLNO	001	0002	3915												
I@XOPC	001	0000	3914	3915 3916 3917 3918 3919 8291 8718 3662											
I@XVAD	001	0002	3916	8298 8349 8792 9197 9268* 9269* 9270* 3681 3702 3712											
I@1SE1	001	0000	3895	3896 3898 4133 4486 4498* 4499 4499* 4500 4500* 4511* 4642 4655* 4668* 4804 4848* 4938* 4939* 4985 4987* 5112 5132* 5158 5189 5191* 5241* 5362 5366 5374 5385* 5391 5391* 5392* 5393 5393* 5394 5394* 5395 5395* 5396 5396* 5397 5397* 5398* 5425* 5595 5598* 5610 5622 5634 6050 6052* 6063 6191 6195 6206 6342 6344 6345* 6350 6376											
				6618 6633 6638 6639* 6655 6656* 6667* 6673* 6698 6885* 6920											
I@1SE2	001	0008	3898	3899 3901 5117 5122 5129 5143 5149 5151 5167 5387* 5690* 6894* 6895* 6977* 7086* 8071*											
I@1SE3	001	0010	3901	3902 4172* 4173 4173* 4176* 4181* 4188* 4194* 4196 4198											
IBR810	003	1ACF	8964												
IDDBAT	001	19AA	8446	8292											
IDDBN1	002	19B1	8455	8379											
IDDSAV	002	19B3	8461	8349* 8391											
IDDSD0	001	191F	8310	8447											
IDDSD1	001	192A	8322	8448											
IDDSD2	001	1930	8334	8449											
IDDVST	001	1900	8286	8447 8448 8449											
IDDWK1	002	19B5	8462	8353 8353* 8365* 8379*											
IDDWK2	002	19B7	8463	8356* 8371 8377* 8386											
IDD010	004	1904	8291												
IDD020	004	190B	8293	8291*											
IDD030	004	190F	8297												
IDD040	003	191C	8303	8293*											
IDD050	004	191F	8314												
IDD060	003	192A	8327												
IDD070	003	1930	8339												
IDD080	004	1933	8349	8328											
IDD090	004	1937	8353												

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 520

SYMBOL	LEN	VALUE	DEFN	REFERENCES
IDD100	003	1942	8360	8327* 8339*
IDD110	003	1945	8364	
IDD120	004	194C	8370	8360
IDD130	003	1954	8375	
IDD140	004	1957	8377	8380
IDD150	004	195E	8379	8375
IDD160	004	1965	8386	
IDD170	005	196C	8391	
IDD180	004	1979	8397	8316
IDD200	001	1989	8420	8355 8364
IDD210	006	198C	8426	
IDD220	004	199A	8432	
IDD230	004	199E	8433	8422*
IDD240	004	19A2	8437	8378 8387
IDFADF	001	1A95	8872	8997
IDFBAT	001	1AE0	8993	8719
IDFCLS	001	1AD2	8972	9000
IDFGET	001	1A40	8788	8994
IDFILE	001	1A00	8713	8994 8995 8996 8997 8998 8999 9000 9001 9002
IDFINI	001	1A87	8850	8996
IDFPRS	001	1A9E	8889	9001
IDFPRU	001	1AAD	8911	9002
IDFPUT	001	1A75	8826	8995
IDFRSR	001	1AC0	8939	8998
IDFRST	001	1AC9	8955	8999
IDFSMK	001	000C	8931	8915
IDF010	004	1A04	8718	
IDF020	004	1A0B	8720	8718*
IDF030	004	1A0F	8724	
IDF040	003	1A18	8729	8720*
IDF050	003	1A1B	8755	8832 8894 8916
IDF055	004	1A1E	8759	
IDF060	006	1A25	8764	
IDF065	004	1A2F	8767	8768 8770 8831* 8833*
IDF070	006	1A36	8776	8760
IDF075	004	1A3C	8780	8755* 8772
IDF100	004	1A40	8792	
IDF110	002	1A49	8795	8792*
IDF120	006	1A4A	8799	
IDF130	003	1A59	8805	
IDF140	004	1A66	8812	8807
IDF150	004	1A6E	8817	
IDF200	003	1A75	8831	
IDF220	003	1A84	8842	
IDF300	005	1A87	8855	
IDF310	004	1A8C	8859	
IDF320	003	1A92	8864	
IDF420	004	1A95	8876	
IDF430	003	1A9B	8881	
IDF500	003	1A9E	8893	
IDF510	004	1AA4	8898	
IDF520	003	1AAA	8903	
IDF600	003	1AAD	8915	
IDF610	004	1AB3	8920	
IDF620	004	1AB9	8925	8842 8864 8881 8903
IDF700	006	1AC0	8943	

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 521

SYMBOL	LEN	VALUE	DEFN	REFERENCES
IDF710	003	1AC6	8947	
IDF800	004	1AC9	8959	
IDF900	004	1AD2	8976	
IDF910	004	1AD8	8981	8947 8964
IDF990	004	1ADC	8985	8818 8926
IDIBM2	002	1BA3	9285	9205
IDIFNC	001	1B00	9195	
IDIFTE	002	1BA8	9291	9218
IDIFVA	001	0001	9303	9206 9208 9229*
IDILBI	001	1BA4	9287	9254 9266
IDILFI	001	1BA5	9288	9270
IDILPV	001	1BA6	9289	9275
IDIVAD	002	1BAA	9297	9255* 9275* 9276
IDI010	006	1B09	9203	
IDI020	003	1B13	9205	9209
IDI030	004	1B24	9213	
IDI040	005	1B2C	9218	9207
IDI050	004	1B34	9223	
IDI060	004	1B3C	9228	9219
IDI070	004	1B45	9236	
IDI080	003	1B51	9242	
IDI090	004	1B57	9247	
IDI100	004	1B5F	9254	9243
IDI110	005	1B7E	9266	
IDI130	004	1B95	9275	
IDP210	004	1A7E	8837	
LPBUFR	001	4F00	0581	0567* 0580 0588
LPRCMD	005	4DE9	0428	0498
RETURN	001	4DB3	0398	1324 1489
SFACTR	001	1CF6	9414	9322* 9328* 9334 9338 9345
SFADFR	001	1C00	9310	9311
SFAD2D	001	1CF4	9412	9362* 9363 9382
SFAVD1	002	1CEE	9407	9315 9374
SFAVD2	002	1CF0	9408	9316
SFAWK1	002	1CF8	9415	9315* 9352 9355* 9398
SFA0B0	001	00B0	9406	9363 9400
SFA001	001	1CF1	9409	9324 9328 9329
SFA007	001	1CF2	9410	9346
SFA008	001	1CF3	9411	9348
SFA010	004	1C21	9325	9330
SFA020	003	1C37	9334	9327 9401*
SFA030	003	1C44	9338	9335
SFA032	001	1CF5	9413	9362
SFA040	005	1C51	9345	9339
SFA050	005	1C65	9352	9376
SFA060	003	1C76	9356	9366
SFA065	004	1C7C	9358	9369
SFA070	005	1C83	9360	9347* 9357
SFA075	003	1C8F	9363	9373* 9400*
SFA080	004	1C9B	9367	9364
SFA090	003	1CA5	9373	9368
SFA100	003	1CB4	9380	9361
SFA110	004	1CCC	9390	9380*
SFA115	005	1CDA	9398	9359 9385
SFA120	003	1CE3	9400	9337 9341
SFGBLK	003	003D	0236	0172 0179

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 522

SYMBOL	LEN	VALUE	DEFN	REFERENCES
SFGBS1	001	2100	0091	0092 0202 0207
SFGBS2	001	2200	0243	0244
SFGBS3	001	2300	0391	0392 0543 0555
SFGBVA	002	214B	0122	0120* 0123 0209
SFGCBA	002	21FC	0231	0125* 0195
SFGCBP	001	00FF	0234	0128
SFGCBV	002	2368	0465	0462* 0463*
SFGCNL	002	22E8	0379	0266* 0271* 0286* 0288 0289 0314* 0380
SFGDEH	001	0006	0566	0476
SFGDLS	001	22E3	0372	0271
SFGDRL	001	00E9	0555	0425
SFGDWL	001	00E3	0543	0417
SFGD2P	004	2276	0311	0273*
SFGELS	001	0004	0376	0348
SFGETR	001	2100	0093	
SFGHDL	001	0007	0565	0476 0566 0569
SFGICR	003	0040	0235	0138 0190
SFGLEH	001	23F4	0568	0478 0485* 0490 0493 0498* 0499
SFGMFA	006	2272	0305	0299*
SFGMLQ	002	22EC	0383	0293* 0294* 0295 0297 0301 0384
SFGMS1	001	00FF	0375	0293
SFGMTA	006	2270	0304	0283* 0301* 0331*
SFGNFM	001	00FF	0233	0116 0119
SFGONE	001	22E4	0373	0331
SFGPAF	001	23F1	0560	0471
SFGPCL	002	22EA	0382	0288* 0291* 0294 0312 0313 0314
SFGPLR	001	23E9	0548	0555
SFGPLW	001	23E3	0536	0543
SFGPSL	001	23F3	0562	0497 0498
SFGRPL	004	2334	0434	0424* 0425*
SFGRST	003	003A	0237	0173
SFGSA0	001	0F00	0564	
SFGSBR	004	233A	0440	0427*
SFGSB2	007	23FA	0570	0507* 0512 0518* 0519
SFGSDF	002	22E6	0378	0272* 0289 0291 0326*
SFGSHD	007	23FA	0569	0476* 0570
SFGSSL	001	23F2	0561	0484 0485
SFGSSZ	002	23F0	0559	0454
SFGSXR	004	233E	0442	0428* 0461* 0470 0477
SFGVCB	002	2234	0277	0274* 0275*
SFGVD2	002	21FA	0229	0094 0223
SFGVNB	002	229D	0330	0327* 0328* 0365
SFGWPL	004	231E	0423	0416* 0417*
SFGXRD	001	00FE	0571	0519*
SFGZRO	002	22E2	0371	0256
SFG120	004	2126	0105	0102
SFG150	003	212D	0107	0104
SFG200	003	2130	0111	0100
SFG205	004	2142	0120	0117
SFG210	003	215A	0129	0118* 0130*
SFG215	004	2160	0134	0197
SFG220	003	216F	0143	0186
SFG225	003	2172	0144	0138* 0172 0173* 0179* 0190*
SFG227	003	2175	0146	0235 0236 0237
SFG230	003	218A	0159	0165
SFG235	003	219F	0172	0157

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 523

SYMBOL	LEN	VALUE	DEFN	REFERENCES
SFG240	004	21A8	0176	0181*
SFG245	003	21AF	0179	0237
SFG250	003	21B2	0181	0236
SFG255	003	21B5	0185	0144 0168 0174 0177 0191 0235
SFG260	003	21BB	0190	0147
SFG265	003	21C1	0195	0150
SFG270	003	21C4	0197	0139* 0152*
SFG280	004	21CD	0206	0129
SFG282	005	21D3	0209	0214
SFG285	004	21DF	0213	0162
SFG290	004	21E6	0218	0112
SFG295	005	21EC	0223	0106 0211
SFG450	003	220D	0266	0257
SFG470	004	2220	0272	0269
SFG500	004	2249	0288	0285 0332
SFG520	003	2258	0293	0290
SFG550	006	226D	0303	0295* 0304 0305
SFG555	004	2273	0310	0311
SFG570	004	22A5	0337	0315
SFG575	003	22AF	0341	0267* 0270*
SFG585	003	22C3	0351	0341
SFG690	004	22D0	0361	0279
SFG695	005	22D4	0365	0339 0342 0349 0351 0357
SFG750	003	2300	0406	0494
SFG760	004	230C	0410	0472
SFG780	003	2313	0416	0409
SFG785	004	2319	0419	0423
SFG790	004	232F	0430	0434
SFG795	004	2337	0439	0440
SFG800	004	233B	0441	0442
SFG810	003	2345	0451	0411
SFG825	003	2355	0457	0452
SFG830	003	2358	0461	0407
SFG840	004	2379	0476	0468
SFG850	003	2386	0483	0492* 0496*
SFG860	003	2394	0490	0479
SFG870	004	239D	0493	0483
SFG880	003	23A4	0496	0491
SFG890	004	23AF	0499	0486
SFG900	004	23B3	0503	0455
SFG920	003	23B7	0507	0202
SFG930	003	23C6	0517	0207 0524* 0527*
SFG935	004	23D0	0521	0517
SFG940	003	23DC	0527	0513
SFG945	004	23DF	0529	0525
SFPBFR	006	1EC8	9708	9702* 9706*
SFPBS1	001	1D00	9519	9520
SFPBS2	001	1E00	9636	9637
SFPBS3	001	1F00	9747	9748 9816 9826
SFPBS4	001	2000	9836	9837
SFPCBP	002	2094	9917	9843* 9939
SFPCBV	002	2095	9916	9917 9918
SFPCFL	005	20DE	9959	9850* 9895* 9900 9955 9956 9966
SFPCNL	001	1EF2	9728	9669* 9676* 9683* 9690 9701 9712* 9729 9766
SFPCPT	002	20C0	9942	9899 9920 9935* 9944 9951* 9965
SFPCPW	002	20FD	9983	9899* 9900* 9904

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 524

SYMBOL	LEN	VALUE	DEFN	REFERENCES
SFPCRT	002	1DF0	9632	9547
SFPCXI	004	1DE0	9615	9581*
SFPC01	002	1EFC	9738	9696 9714
SFPDAC	002	20FD	9981	9876* 9880 9882 9882* 9982
SFPDCA	005	20DF	9958	9956*
SFPDEV	002	1DEB	9626	9540* 9563 9628
SFPDIC	002	1DEB	9628	9595* 9597*
SFPDLS	001	0004	9743	9680
SFPDP1	001	1F7E	9809	9816
SFPDP2	001	1F84	9819	9826
SFPD1D	001	007E	9816	9770
SFPD2D	001	0084	9826	9777
SFPENC	001	0005	9974	9895
SFPEXI	004	20FA	9978	9862
SFPEZR	001	20FB	9979	9863
SFPLEX	001	0004	9972	9862 9974 9978
SFPLXM	001	0002	9973	9876 9880 9880 9882 9882
SFPMPT	002	1DEE	9631	9543
SFPMS1	001	00FF	9744	9703
SFPMVL	006	1EC6	9709	9703* 9704* 9705 9706
SFPNGE	002	20FD	9982	9869* 9870* 9871 9983
SFPONE	001	1DEC	9630	9597
SFPPRT	002	1EF8	9734	9689* 9701* 9704 9711 9712 9735
SFPRT2	002	1F8B	9830	9749
SFPSAO	001	0F00	9833	
SFPSCA	002	1EFA	9737	9723 9803*
SFPSIO	002	1EF6	9742	9654* 9667
SFPSTC	005	20E1	9960	9849* 9955*
SFPSTK	006	1ECA	9710	9684* 9705* 9714* 9767
SFPUTR	001	1D00	9522	
SFPVCA	002	20C0	9941	9842* 9939* 9942
SFPVD2	002	1DF2	9633	9523 9567
SFPWK2	002	1EF6	9733	9692* 9694* 9695* 9696* 9742
SFPWRK	001	1EF4	9732	9652* 9653* 9687 9689 9690* 9693* 9694
SFPXR1	004	1E7C	9686	9655* 9660
SFPX01	001	20F5	9976	9951 9967
SFPZD1	001	20F6	9977	9876
SFP050	004	1D26	9534	9531
SFP075	003	1D2D	9536	9533
SFP100	005	1D30	9540	9529
SFP120	003	1D43	9545	9542
SFP130	003	1D54	9552	9544
SFP133	004	1D61	9556	9553
SFP135	004	1D6F	9560	9558
SFP140	004	1D73	9561	9555
SFP150	005	1D7E	9567	9535 9577 9622
SFP175	003	1D8B	9573	9546
SFP200	003	1D9A	9581	9574
SFP220	004	1DA3	9588	
SFP230	003	1DB3	9596	9600 9604
SFP250	003	1DD0	9605	9598
SFP320	004	1DDD	9614	9583 9590 9615
SFP350	005	1E0D	9649	9645
SFP370	003	1E1B	9652	9647
SFP385	004	1E3F	9664	9657
SFP400	003	1E5F	9676	9668

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 525

SYMBOL	LEN	VALUE	DEFN	REFERENCES
SFP410	003	1E65	9678	
SFP430	003	1E73	9683	9666
SFP450	003	1E76	9684	9672 9675 9677 9681
SFP460	004	1E79	9685	9686
SFP480	004	1EA5	9698	
SFP490	004	1EA9	9699	9662 9713
SFP5	001	0005	9832	9769* 9770* 9776* 9777*
SFP500	004	1EAD	9701	9688 9691 9716
SFP510	005	1EB1	9702	9697
SFP550	006	1EC5	9707	9708 9709 9710
SFP560	004	1ED6	9714	
SFP580	003	1EE0	9720	9646 9661 9715
SFP590	004	1EEE	9724	9720*
SFP610	005	1F1E	9764	9753 9755
SFP625	006	1F28	9766	9764*
SFP630	006	1F2E	9767	9765*
SFP635	004	1F3E	9772	9769* 9770*
SFP640	004	1F50	9781	9776* 9777*
SFP650	004	1F58	9789	9778*
SFP655	004	1F5C	9790	9779*
SFP675	005	1F66	9796	9760
SFP680	006	1F74	9803	9751*
SFP720	004	204A	9876	9865
SFP725	003	2051	9878	9877* 9881 9881* 9883
SFP730	004	205B	9881	9879
SFP750	003	2066	9887	9864
SFP760	003	2075	9895	9889
SFP785	004	2078	9899	9852
SFP790	003	208A	9904	9902 9903* 9905 9907
SFP800	004	2090	9915	9846
SFP830	004	20AA	9932	9841* 9964
SFP850	004	20B7	9939	9908
SFP865	004	20D5	9955	9945
SFP875	005	20DD	9957	9958 9959 9960
SFP950	004	20F1	9968	9934
SFRBS1	001	2400	0578	0577 0684
SFRCAL	001	2400	0582	
SFRCLS	001	240A	0589	
SFRIXR	004	2484	0655	0600*
SFRLPR	003	24B7	0688	0699
SFRNOE	001	24AB	0674	0651* 0675
SFRONE	001	24AA	0672	0651
SFRSET	001	240D	0594	
SFRVD2	002	2412	0599	0653 0663
SFRX10	001	24AC	0677	0656
SFR100	004	240D	0598	
SFR110	003	2416	0604	0584* 0665*
SFR115	003	241C	0607	0604 0657
SFR130	003	241F	0611	
SFR135	004	2448	0624	0618*
SFR140	003	244F	0629	0590* 0614 0661*
SFR200	004	2452	0634	0643
SFR300	003	2461	0641	0629
SFR900	003	2472	0650	0583* 0612 0616 0637 0666*
SFR950	004	2481	0654	0655
SFR995	003	248C	0661	0650 0652

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 526

SYMBOL	LEN	VALUE	DEFN	REFERENCES
SFR996	004	24C3	0692	0688*
SFR997	004	24CA	0694	0687
SFR998	006	24D1	0696	0693
SFR999	004	24D7	0697	0695
SF1000	001	24E5	0702	0689
V\$APWR	001	0800	2366	2512
V\$BFR1	001	5400	2429	2620 2021 5403 6676
V\$BFR2	001	5500	2430	2621 5404
V\$CBNZ	001	0CB2	2438	2519 4502 4513
V\$CCON	001	5120	2445	2617
V\$CDCV	001	3100	2442	2572 0522
V\$CDSY	001	2E00	2441	2569 0510
V\$CFPZ	001	0C70	2436	2518 4815 5640 6371
V\$CNXZ	001	0470	2439	2507 4162
V\$CSSR	001	5100	2444	2616
V\$CZFP	001	04AD	2437	2508 4207 5665 6395
V\$DTLN	001	4600	2451	2604
V\$DTVR	001	4700	2452	2605
V\$FABS	001	1761	2337	2536
V\$FACS	001	1400	2353	2528
V\$FASN	001	1413	2352	2529
V\$FATN	001	1100	2351	2525 6383 6713
V\$FCOS	001	0A00	2348	2514 6212 6225 7311
V\$FCOT	001	0D00	2346	2520
V\$FCSC	001	1725	2350	2535
V\$FDEG	001	17DA	2357	2540
V\$FDET	001	4540	2360	2603
V\$FEXP	001	0500	2344	2509 4798 5232 6869 6966
V\$FHCS	001	1500	2356	2530
V\$FHSN	001	1557	2355	2531
V\$FHTN	001	1593	2354	2532
V\$FINT	001	176C	2338	2537
V\$FLGT	001	0200	2342	2502 4154 4878
V\$FLOG	001	0219	2341	2504 5215
V\$FLTW	001	020B	2343	2503
V\$FRAD	001	17CB	2358	2539
V\$FRND	001	1800	2359	2541
V\$FSEC	001	1700	2349	2534
V\$FSGN	001	17A7	2339	2538
V\$FSIN	001	0A1A	2347	2515 6216 6221 7335
V\$FSQR	001	0900	2340	2513 6675
V\$FTAN	001	0D28	2345	2521
V\$IFCI	001	1B00	2329	2545
V\$IFIO	001	1A00	2331	2544
V\$ISDN	001	1900	2330	2542
V\$KBTL	001	1EAC	2473	
V\$KBTS	001	0DAC	2472	
V\$LPRB	001	4F00	2427	2614 1493 0416
V\$LPRT	001	4D00	2425	2612 1279 1469 1487 0412 0621
V\$LPR2	001	4E00	2426	2613 0425 0429
V\$MADD	001	4007	2374	2592
V\$MASN	001	43A0	2372	2599
V\$MCON	001	4324	2379	2597
V\$MIDN	001	4300	2380	2596
V\$MINV	001	4500	2384	2602
V\$MMPY	001	4100	2376	2593

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 527

SYMBOL	LEN	VALUE	DEFN	REFERENCES
V\$MSMY	001	4264	2377	2595
V\$MSUB	001	4000	2375	2591
V\$MTRN	001	4400	2383	2601
V\$MZER	001	432B	2381	2598
V\$PCH1	001	5200	2465	2618
V\$PCH2	001	5300	2466	2619 1400 4658 0433 0434
V\$SCDI	001	2A00	2422	2563 0135
V\$SCDO	001	2A96	2423	2564 9928
V\$SFA2	001	5000	2407	2615 9397
V\$SFD1	001	0000	2417	2500 9407
V\$SFD2	001	0100	2418	2501 9408 9633 0229 0599
V\$SKEY	001	2500	2421	2558 0875 0897 1031 1032 1955 6586
V\$SPRT	001	2800	2420	2561 0691 1129 1368 2216 4619 5661 8711 9056 0648
V\$VMPL	001	4C06	2459	2611 8415
V\$VMPS	001	4C00	2458	2610 8391
V\$XKAF	001	1C00	2406	2546 8877
V\$XKCA	001	2400	2410	2554
V\$XKCL	001	240A	2409	2555 8977
V\$XKIN	001	2B00	2405	2565 8860
V\$XKLP	001	24AD	2411	
V\$XKRS	001	240D	2408	2556 8960
V\$XMGT	001	3E06	2399	2586
V\$XMIN	001	3D00	2398	2584
V\$XMPL	001	3F06	2402	2589
V\$XMPS	001	3F00	2401	2588
V\$XMPT	001	3E0C	2400	2587
V\$XMPU	001	3F13	2403	2590
V\$XMRD	001	3E00	2397	2585
V\$XSGT	001	2100	2392	2551 0202 0207 0219 0262 0324 6969
V\$XSIN	001	2B6E	2391	2566
V\$XSPR	001	3400	2394	2575 8899 9562 7255 7259 7267 7323
V\$XSPT	001	1D00	2393	2547 8838 9576 9618 9722 9830 0623 6998
V\$XSPU	001	3800	2395	2579 8921 7324
V\$XSRD	001	3300	2390	2574 6945
V\$00E1	001	0000	2500	
V\$01E1	001	0100	2501	
V\$02E1	001	0200	2502	
V\$02E2	001	020B	2503	
V\$02F3	001	0219	2504	
V\$03CC	001	0300	2505	
V\$04CC	001	0400	2506	
V\$04E1	001	0470	2507	
V\$04E2	001	04AD	2508	
V\$05E1	001	0500	2509	
V\$06CC	001	0600	2510	
V\$07CC	001	0700	2511	
V\$08E1	001	0800	2512	
V\$09E1	001	0900	2513	
V\$10E1	001	0A00	2514	
V\$10E2	001	0A1A	2515	
V\$11CC	001	0B00	2516	
V\$12CC	001	0C00	2517	
V\$12E1	001	0C70	2518	
V\$12E2	001	0CB2	2519	
V\$13E1	001	0D00	2520	
V\$13E2	001	0D28	2521	

CROSS REFERENCE

SYMBOL LEN VALUE DEFN REFERENCES VER 15, MOD 00 31/05/21 PAGE 528

V\$14CC 001 0E00 2522
V\$15CC 001 0F00 2523
V\$16CC 001 1000 2524
V\$17E1 001 1100 2525
V\$18CC 001 1200 2526
V\$19CC 001 1300 2527
V\$20E1 001 1400 2528
V\$20E2 001 1413 2529
V\$21E1 001 1500 2530
V\$21E2 001 1557 2531
V\$21E3 001 1593 2532
V\$22CC 001 1600 2533
V\$23E1 001 1700 2534
V\$23E2 001 1725 2535
V\$23E3 001 1761 2536
V\$23E4 001 176C 2537
V\$23E5 001 17A7 2538
V\$23E6 001 17CB 2539
V\$23E7 001 17DA 2540
V\$24E1 001 1800 2541
V\$25E1 001 1900 2542
V\$26E1 001 1A00 2544
V\$27E1 001 1B00 2545
V\$28E1 001 1C00 2546
V\$29E1 001 1D00 2547
V\$30CC 001 1E00 2548
V\$31CC 001 1F00 2549
V\$32CC 001 2000 2550
V\$33E1 001 2100 2551
V\$34CC 001 2200 2552
V\$35CC 001 2300 2553
V\$36CC 001 2400 2557
V\$36E1 001 2400 2554
V\$36E2 001 240A 2555
V\$36E3 001 240D 2556
V\$37E1 001 2500 2558
V\$38CC 001 2600 2559
V\$39CC 001 2700 2560
V\$40E1 001 2800 2561
V\$41CC 001 2900 2562
V\$42E1 001 2A00 2563
V\$42E2 001 2A96 2564
V\$43E1 001 2B00 2565
V\$43E2 001 2B6E 2566
V\$44CC 001 2C00 2567
V\$45CC 001 2D00 2568
V\$46E1 001 2E00 2569
V\$47CC 001 2F00 2570
V\$48CC 001 3000 2571
V\$49E1 001 3100 2572
V\$50CC 001 3200 2573
V\$51E1 001 3300 2574
V\$52E1 001 3400 2575
V\$53CC 001 3500 2576
V\$54CC 001 3600 2577
V\$55CC 001 3700 2578

CROSS REFERENCE

SYMBOL LEN VALUE DEFN REFERENCES VER 15, MOD 00 31/05/21 PAGE 529

V\$56E1 001 3800 2579
V\$57CC 001 3900 2580
V\$58CC 001 3A00 2581
V\$59CC 001 3B00 2582
V\$60CC 001 3C00 2583
V\$61E1 001 3D00 2584
V\$62E1 001 3E00 2585
V\$62E2 001 3E06 2586
V\$62E3 001 3E0C 2587
V\$63E1 001 3F00 2588
V\$63E2 001 3F06 2589
V\$63E3 001 3F13 2590
V\$64E1 001 4000 2591
V\$64E2 001 4007 2592
V\$65E1 001 4100 2593
V\$66CC 001 4200 2594
V\$66E1 001 4264 2595
V\$67E1 001 4300 2596
V\$67E2 001 4324 2597
V\$67E3 001 432B 2598
V\$67E4 001 43A0 2599
V\$68E1 001 4400 2601
V\$69E1 001 4500 2602
V\$69E2 001 4540 2603
V\$70E1 001 4600 2604
V\$71E1 001 4700 2605
V\$72CC 001 4800 2606
V\$73CC 001 4900 2607
V\$74CC 001 4A00 2608
V\$75CC 001 4B00 2609
V\$76E1 001 4C00 2610
V\$76E2 001 4C06 2611
V\$77CC 001 4D00 2612
V\$78CC 001 4E00 2613
V\$79CC 001 4F00 2614
V\$80E1 001 5000 2615
V\$81E2 001 5100 2616
V\$81E3 001 5120 2617
V\$82E1 001 5200 2618
V\$83E2 001 5300 2619
V\$84E1 001 5400 2620
V\$85E2 001 5500 2621
V@CDPT 001 0007 2632
V@CHGH 001 0008 2737
V@CMIC 001 0002 2633
V@CMNI 001 00FF 2630
V@CMUL 001 0007 2738
V@CNIX 001 0080 2631
V@COEX 001 001E 2628
V@CPLS 001 00F0 2635
V@CPRC 001 000A 2637
V@CSQR 001 0003 2735
V@CSTR 001 0002 2736
V@CTTA 001 0027 2638
V@DCAD 001 0002 2658
V@DEXP 001 0000 2663

2659

CROSS REFERENCE

VER 15, MOD 00 31/05/21 PAGE 530

SYMBOL	LEN	VALUE	DEFN	REFERENCES
V@DMAN	001	000D	2665	2666
V@DMN1	001	0001	2664	
V@DPDF	001	0002	2653	
V@DSAD	001	0001	2654	
V@DSGN	001	000D	2666	
V@DVAD	001	0004	2659	
V@EART	001	0001	2636	
V@ECRT	001	0038	2709	
V@EFUL	001	00F8	2708	
V@EINV	001	00FB	2704	
V@EIPR	001	00F5	2705	
V@ENSV	001	00F7	2706	
V@ENUL	001	0000	2703	
V@ERPC	001	0020	2634	
V@ESAV	001	00F6	2707	
V@FEHN	001	0002	2733	
V@FEPL	001	0091	2729	
V@FERS	001	0003	2732	
V@FPGS	001	0081	2728	
V@FRET	001	0015	2731	
V@FSPC	001	0040	2730	
V@FTAB	001	0000	2734	
V@KADD	001	004E	2719	
V@KCLE	001	006E	2716	
V@KDIV	001	0061	2722	
V@KEMN	001	006C	2714	
V@KEPL	001	006B	2713	
V@KMUL	001	005C	2721	
V@KPER	001	004B	2724	
V@KPST	001	007B	2718	
V@KPWR	001	005A	2723	
V@KSQR	001	006F	2715	
V@KSTO	001	006D	2717	
V@KSUB	001	0060	2720	
V@LAIP	001	0003	2684	2685
V@LDEX	001	0002	2687	
V@LETE	001	0003	2691	
V@LEXP	001	0001	2681	2683
V@LFKO	001	0006	2686	
V@LINI	001	0200	2690	
V@LLKS	001	0010	2683	
V@LMAN	001	000F	2682	2683
V@LNOP	001	0015	2688	
V@LTBE	001	0007	2685	
V@LVPG	001	0100	2689	2690
V@MCHS	001	00C0	2670	
V@MCRD	001	0010	2646	
V@MDEF	001	0008	2647	
V@MEXC	001	0080	2644	
V@MEXT	001	0004	2673	
V@MICC	001	0010	2629	
V@MIPC	001	0080	2671	
V@MIPL	001	0020	2677	
V@MLST	001	0040	2645	
V@MPND	001	0000	2676	
V@MPOF	001	0080	2674	

CROSS REFERENCE

SYMBOL LEN VALUE DEFN REFERENCES VER 15, MOD 00 31/05/21 PAGE 531

V@MPRC	001	0020	2643	
V@MSFU	001	0002	2648	
V@MSTN	001	0004	2642	
V@OALL	001	00F4	2699	
V@ONUL	001	00F0	2695	2696
V@OPM1	001	00F2	2697	2698
V@ORTN	001	00F1	2696	2697
V@OSTK	001	00F3	2698	2699
V@PEOF	001	0002	2672	
V@PSQ2	001	0014	2675	
VLPR2	001	4E00	0459	0424 0425
VLPR3	001	5300	0602	
VLPR4	001	5359	0636	
VLPR5	001	5391	0656	
VLPR6	001	53B6	0670	

TOTAL STATEMENTS IN ERROR IN THIS ASSEMBLY = 0

OL105 I THE CODE LENGTH OF #FMSTD IS 21453 DECIMAL.
OL103 I TOTAL NUMBER OF LIBRARY SECTORS REQUIRED IS 89
 NAME-#FMSTD,PACK-R1R1R1,UNIT-R1,RETAIN-P,LIBRARY-R,CATEGORY-000

START ADDRESS	CATEGORY	NAME AND ENTRY	CODE LENGTH	HEXADECIMAL	DECIMAL
0200	0	#FMSTD	53CD	21453	
OL100 I THE TOTAL CORE USED BY #FMSTD IS 21453 DECIMAL.					
OL101 I THE START CONTROL ADDRESS OF THIS MODULE IS 0200.					
OL104 I TOTAL NUMBER OF LIBRARY SECTORS REQUIRED IS 84					
NAME-#FMSTD,PACK-R1R1R1,UNIT-R1,RETAIN-P,LIBRARY-O					
2698					
V@ORTN	001	00F1	2696	2697	
V@OSTK	001	00F3	2698	2699	
V@PEOF	001	0002	2672		
V@PSQ2	001	0014	2675		
VLPRT2	001	4E00	0467	0432	0433
VLPRT3	001	5300	0610		
VLPRT4	001	5359	0644		
VLPRT5	001	5391	0664		
VLPRT6	001	53B6	0678		
TOTAL STATEMENTS IN ERROR IN THIS ASSEMBLY = 4					