

VS60

VS60 INST TST PT1
CZVSAB0

AH-9483B-MC

COPYRIGHT 76-80
FICHE 1 OF 1

JAN 1980

digital

MADE IN USA

IDENTIFICATION

B 1

SEQ 0001

PRODUCT CODE: AC-9482B-MC
PRODUCT NAME: CZVSABO VS60 INST TST PT1
DATE: JUNE 1979
MAINTAINER: DIAGNOSTIC ENGINEERING
AUTHORS: R. SHOOP & R. MOORE

COPYRIGHT (C) 1976, 1979
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE TERMS. TITLE TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.

0.0 TABLE OF CONTENTS

1.0	ABSTRACT
2.0	REQUIREMENTS
2.1	EQUIPMENT
2.2	STORAGE
3.0	LOADING PROCEDURE
4.0	STARTING PROCEDURE
4.1	STARTING ADDRESS 200
4.2	RESTART ADDRESS 204
5.0	SWITCH REGISTER SETTINGS
6.0	ERROR REPORTING
6.1	ERROR COMMENTS
6.2	ERROR DATA
7.0	MISCELLANEOUS
7.1	VS60 BUS/VECTOR ADDRESS MODIFICATION
7.2	XXDP/APT NOTES
7.3	POWER FAIL
7.4	SINGLE VS60 TESTING
8.0	EXECUTION TIME
9.0	PROGRAM TEST DESCRIPTIONS
9.1	GENERAL
9.2	TYPICAL TEST SEQUENCE
10.0	LISTING

1.0 ABSTRACT

THIS PROGRAM IS THE FIRST OF A SERIES DESIGNED TO TEST STRICTLY LOGIC FUNCTIONS OF THE VS60 DISPLAY PROCESSOR. THIS DIAGNOSTIC TESTS, READING AND WRITING OF VS60 BUS REGISTERS, NON GRAPHIC TYPE INSTRUCTIONS AND SEVERAL INTERRUPT CONDITIONS. THERE IS NO FULL SPEED VS60 TESTING IN THIS TEST. MAINTENANCE MODES 1, 2 OR 3 ARE USED.

*** REV B -- FIXES TIMING PROBLEM IN TEST 136 ***

2.0 REQUIREMENTS

2.1 EQUIPMENT

- A. PDP-11 COMPUTER WITH AT LEAST 8K OF MEMORY
- B. I/O TERMINAL (I.E. ASR33 TTY)
- C. VS60 DISPLAY PROCESSOR

2.2 STORAGE

THIS PROGRAM OCCUPIES LOWER 8K OF MEMORY.

3.0 LOADING PROCEDURE

NORMAL PROCEDURE FOR LOADING A BINARY PROGRAM INTO MEMORY SHOULD BE FOLLOWED.

4.0 STARTING PROCEDURE

4.1 STARTING ADDRESS 200

LOADING ADDRESS 200 AND STARTING WILL IDENTIFY THE TEST, INITIALIZE THE SYSTEM, AND BEGIN TESTING.

4.2 RESTART ADDRESS 204

LOADING ADDRESS 204 AND STARTING WILL INITIALIZE THE SYSTEM AND BEGIN TESTING.

5.0 SWITCH REGISTER SETTINGS

SWITCH	FUNCTION
SW15=1	HALT ON ERROR
SW14=1	LOOP ON TEST
SW13=1	INHIBIT ERROR TYPEOUTS
SW12=1	HALT AT END OF DIAGNOSTIC
SW11=1	INHIBIT ITERATIONS - NORMALLY 2000(8) AFTER 1ST PASS
SW10=1	BELL ON ERROR
SW09=1	LOOP ON ERROR
SW08=1	LOOP ON TEST IN SWR<7:0>

6.0 ERROR REPORTING

6.1 ERROR COMMENT

ALL ERRORS ARE ACCOMPANIED WITH AN ENGLISH LANGUAGE DESCRIPTIVE COMMENT AS TO THE TYPE OF FAILURE. FURTHER QUALIFICATION OF THE ERROR CAN BE OBTAINED IF NEEDED FROM THE COMMENT AT THE ERROR PC OF THE LISTING.

6.2 ERROR DATA

PC	LISTING ADDRESS WHERE THE ERROR WAS DETECTED.
TSTNUM	TEST NUMBER WHERE THE ERROR OCCURRED.
BUSADRS	DISPLAY BUS ADDRESS WHERE THE RESULTANT DATA WAS EXPECTED
EXPCT	DATA THAT WAS EXPECTED.
RCVD	DATA THAT WAS RECEIVED.

7.0 MISCELLANEOUS

7.1 VS60 BUS/VECTOR ADDRESS MODIFICATION:
 MODIFY LOCATION '\$VECT1' IF BASE VECTOR ADDRESS IS NOT 320.
 MODIFY LOCATION '\$BASE' IF BASE BUS ADDRESS IS NOT 172000.

NOTE: A RESTART IS REQUIRED AFTER THE ABOVE ADDRESS MODIFICATION.
 THE ABOVE LOCATIONS ARE LOCATED IN THE 'APT MAILBOX-ETABLE'

7.2 XXDP/APT NOTES:
 THIS DIAGNOSTIC IS CHAINABLE UNDER XXDP.
 THIS DIAGNOSTIC INCLUDES THE 'APT' SOFTWARE HOOKS HOWEVER, THEY
 HAVE NOT BEEN TESTED.

7.3 POWER FAIL:
 A POWER FAILURE WILL CAUSE A RESTART MESSAGE ON POWER UP AT
 WHICH TIME THE PROGRAM IS RESTARTED AT THE BEGINNING.

7.4 SINGLE VS60 TESTING:
 THIS DIAGNOSTIC DOES NOT TEST MULTIPLE VS60'S.

8.0 EXECUTION TIME

EXECUTION TIME RANGES FROM ABOUT 3 SECONDS WITH NO ITERATIONS
 TO ABOUT 2 MINUTES WITH ITERATIONS ENABLED. AN 'END PASS' MESSAGE
 IS TYPED EACH PASS THRU THE DIAGNOSTIC.

9.0 PROGRAM TEST DESCRIPTIONS

9.1 GENERAL

THE TABLE OF CONTENTS AT THE BEGINNING OF THE LISTING CONTAINS
 A BRIEF DESCRIPTION OF EACH TEST. IT WILL ALSO IDENTIFY WHAT MAINTENANCE
 MODE A PARTICULAR TEST USES. THE COMMENT FIELD IN EACH TEST
 ALSO CAN BE HELPFUL IN UNDERSTANDING A TEST.

9.2 TYPICAL TEST SEQUENCE

MOST TESTS ARE ORGANIZED AROUND THE FOLLOWING FORMAT:
 A MAINTENANCE MODE SWITCH IS SET (SW1, 2 OR 3). THEN A VS60 REGISTER
 BUS REGISTER ADDRESS IS LOADED INTO LOCATION '\$BDADR' WHICH
 IS WHERE THE EXPECTED DATA WILL BE READ. NEXT STEP WILL BE TO LOAD THE
 EXPECTED DATA INTO LOCATION '\$GDDAT'. THEN ONE OR MORE INSTRUCTIONS
 ARE LOADED STARTING AT LOCATION 'BUFFER'. THE DISPLAY PC IS THEN LOADED
 WITH ADDRESS 'BUFFER' WHICH CAUSES AN NPR TO 'BUFFER'. AT THE COMPLETION
 OF THE 'NPR' A NUMBER OF RESUMES COULD BE ISSUED IF ADDITIONAL 'NPRS' ARE
 REQUIRED FOR INSTRUCTIONS OR DATA. AT THIS TIME THE CONDITION
 UNDER TEST SHOULD HAVE BEEN SET UP AND AVAILABLE AT THE BUS ADDRESS
 SET UP IN '\$BDADR'. THIS CONDITION IS USUALLY LOADED INTO LOCATION
 '\$BDDAT' AND COMPARED TO THE EXPECTED RESULTS PREVIOUSLY SET UP
 IN '\$GDDAT'. IF AN ERROR IS DETECTED THEN AN ERROR TRAP (EMT) IS PREFORMED
 ACCORDING TO THE ITEM NUMBER IN THE '\$ERRTB' (ERROR TABLE).
 THIS TABLE DIRECTS OR POINTS TO THE PERTINENT 'TYPE OUT' INFORMATION.

10.0

LISTING

G 1

SEQ 0006

PROGRAM LISTING FOLLOWS.

(1)	040000	SW14=	40000
(1)	020000	SW13=	20000
(1)	010000	SW12=	10000
(1)	004000	SW11=	4000
(1)	002000	SW10=	2000
(1)	001000	SW09=	1000
(1)	000400	SW08=	400
(1)	000200	SW07=	200
(1)	000100	SW06=	100
(1)	000040	SW05=	40
(1)	000020	SW04=	20
(1)	000010	SW03=	10
(1)	000004	SW02=	4
(1)	000002	SW01=	2
(1)	000001	SW00=	1
(1)		.EQUIV	SW09,SW9
(1)		.EQUIV	SW08,SW8
(1)		.EQUIV	SW07,SW7
(1)		.EQUIV	SW06,SW6
(1)		.EQUIV	SW05,SW5
(1)		.EQUIV	SW04,SW4
(1)		.EQUIV	SW03,SW3
(1)		.EQUIV	SW02,SW2
(1)		.EQUIV	SW01,SW1
(1)		.EQUIV	SW00,SW0

;*DATA BIT DEFINITIONS (BIT00 TO BIT15)

(1)	100000	BIT15=	100000
(1)	040000	BIT14=	40000
(1)	020000	BIT13=	20000
(1)	010000	BIT12=	10000
(1)	004000	BIT11=	4000
(1)	002000	BIT10=	2000
(1)	001000	BIT09=	1000
(1)	000400	BIT08=	400
(1)	000200	BIT07=	200
(1)	000100	BIT06=	100
(1)	000040	BIT05=	40
(1)	000020	BIT04=	20
(1)	000010	BIT03=	10
(1)	000004	BIT02=	4
(1)	000002	BIT01=	2
(1)	000001	BIT00=	1
(1)		.EQUIV	BIT09,BIT9
(1)		.EQUIV	BIT08,BIT8
(1)		.EQUIV	BIT07,BIT7
(1)		.EQUIV	BIT06,BIT6
(1)		.EQUIV	BIT05,BIT5
(1)		.EQUIV	BIT04,BIT4
(1)		.EQUIV	BIT03,BIT3
(1)		.EQUIV	BIT02,BIT2
(1)		.EQUIV	BIT01,BIT1
(1)		.EQUIV	BIT00,BIT0

;*BASIC "CPU" TRAP VECTOR ADDRESSES
ERRVEC= 4 ;:TIME OUT AND OTHER ERRORS

(1)	000010	RESVEC= 10	::RESERVED AND ILLEGAL INSTRUCTIONS
(1)	000014	TBITVEC=14	::'T' BIT
(1)	000014	TRTVEC= 14	::TRACE TRAP
(1)	000014	BPTVEC= 14	::BREAKPOINT TRAP (BPT)
(1)	000020	IOTVEC= 20	::INPUT/OUTPUT TRAP (IOT) **SCOPE**
(1)	000024	PWRVEC= 24	::POWER FAIL
(1)	000030	EMTVEC= 30	::EMULATOR TRAP (EMT) **ERROR**
(1)	000034	TRAPVEC=34	::'TRAP' TRAP
(1)	000060	TKVEC= 60	::TTY KEYBOARD VECTOR
(1)	000064	TPVEC= 64	::TTY PRINTER VECTOR
(1)	000240	PIRQVEC=240	::PROGRAM INTERRUPT REQUEST VECTOR
18			
19	172000	ABASE= 172000	: 1ST REGISTER ADDRESS (DISPLAY PC).
20	100320	AVECT1= 100320	: PRI (HI BYTE) AND VECTOR (LO BYTE).

(1) 001004 000012
(1) 001006 000036
(1) 001010 000000
(1) 001012 000033
31

\$TSTM: .WORD 10. ::RUN TIM OF LONGEST TEST
\$PASTM: .WORD 30. ::RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
\$UNITM: .WORD 0 ::ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
.WORD \$ETEND-\$MAIL/2 ::LENGTH MAILBOX-ETABLE(WORDS)

(2)	001210	000000	\$PASS:	.WORD	APASS	::PASS COUNT
(2)	001212	000000	\$DEVCT:	.WORD	ADEVCT	::DEVICE COUNT
(2)	001214	000000	\$UNIT:	.WORD	AUNIT	::I/O UNIT NUMBER
(2)	001216	000000	\$MSGAD:	.WORD	AMSGAD	::MESSAGE ADDRESS
(2)	001220	000000	\$MSGLG:	.WORD	AMSGLG	::MESSAGE LENGTH
(2)	001222		\$ETABLE:			::APT ENVIRONMENT TABLE
(2)	001222	000	\$ENV:	.BYTE	AENV	::ENVIRONMENT BYTE
(2)	001223	000	\$ENVM:	.BYTE	AENVM	::ENVIRONMENT MODE BITS
(2)	001224	000000	\$SWREG:	.WORD	ASWREG	::APT SWITCH REGISTER
(2)	001226	000000	\$USWR:	.WORD	AUSWR	::USER SWITCHES
(2)	001230	000000	\$CPUOP:	.WORD	ACPUOP	::CPU TYPE,OPTIONS
(2)			*			BITS 15-11=CPU TYPE
(2)			*			11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
(2)			*			11/70=06,PDQ=07,Q=10
(2)			*			BIT 10=REAL TIME CLOCK
(2)			*			BIT 9=FLOATING POINT PROCESSOR
(2)			*			BIT 8=MEMORY MANAGEMENT
(2)	001232	000	\$MAMS1:	.BYTE	AMAMS1	::HIGH ADDRESS,M.S. BYTE
(2)	001233	000	\$MTYP1:	.BYTE	AMTYP1	::MEM. TYPE,BLK#1
(2)			*			MEM. TYPE BYTE -- (HIGH BYTE)
(2)			*			900 NSEC CORE=001
(2)			*			300 NSEC BIPOLAR=002
(2)			*			500 NSEC MOS=003
(2)	001234	000000	\$MADR1:	.WORD	AMADR1	::HIGH ADDRESS,BLK#1
(2)			*			MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF 'TYPE' ABOVE
(2)	001236	000	\$MAMS2:	.BYTE	AMAMS2	::HIGH ADDRESS,M.S. BYTE
(2)	001237	000	\$MTYP2:	.BYTE	AMTYP2	::MEM. TYPE,BLK#2
(2)	001240	000000	\$MADR2:	.WORD	AMADR2	::MEM.LAST ADDRESS,BLK#2
(2)	001242	000	\$MAMS3:	.BYTE	AMAMS3	::HIGH ADDRESS,M.S.BYTE
(2)	001243	000	\$MTYP3:	.BYTE	AMTYP3	::MEM. TYPE,BLK#3
(2)	001244	000000	\$MADR3:	.WORD	AMADR3	::MEM.LAST ADDRESS,BLK#3
(2)	001246	000	\$MAMS4:	.BYTE	AMAMS4	::HIGH ADDRESS,M.S.BYTE
(2)	001247	000	\$MTYP4:	.BYTE	AMTYP4	::MEM. TYPE,BLK#4
(2)	001250	000000	\$MADR4:	.WORD	AMADR4	::MEM.LAST ADDRESS,BLK#4
(2)	001252	100320	\$VECT1:	.WORD	AVECT1	::INTERRUPT VECTOR#1,BUS PRIORITY#1
(2)	001254	000000	\$VECT2:	.WORD	AVECT2	::INTERRUPT VECTOR#2BUS PRIORITY#2
(2)	001256	172000	\$BASE:	.WORD	ABASE	::BASE ADDRESS OF EQUIPMENT UNDER TEST
(2)	001260	000000	\$DEVN:	.WORD	ADEVN	::DEVICE MAP
(2)	001262	000000	\$CDW1:	.WORD	ACDW1	::CONTROLLER DESCRIPTION WORD#1
(2)	001264	000000	\$CDW2:	.WORD	ACDW2	::CONTROLLER DESCRIPTION WORD#2
(2)	001266	000000	\$DDW0:	.WORD	ADDW0	::DEVICE DESCRIPTOR WORD#0
(2)	001270		\$ETEND:			
(2)			.MEXIT			

```
(1) .SBTTL ERROR POINTER TABLE
(1)
(1) ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
(1) ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
(1) ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
(1) ;*NOTE1: IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
(1) ;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
(1)
(1) ;* EM ::POINTS TO THE ERROR MESSAGE
(1) ;* DH ::POINTS TO THE DATA HEADER
(1) ;* DT ::POINTS TO THE DATA
(1) ;* DF ::POINTS TO THE DATA FORMAT
(1)
(1) 001270 $ERRTB:
33
34 ;ITEM 1
35
36 001270 027216 EM1 ;MAINT. SWITCH REGISTER
37 001272 031014 DH1 ;ERRPC TSTNUM BUSADR EXPCT RCVD
38 001274 031062 DT1 ;$ERRPC TSTNUM $BDADR $GDDAT $BDDAT
39 001276 000000 0
40
41 ;ITEM 2
42
43 001300 027240 EM2 ;LOADING D.P.C. REGISTER <STATIC>
44 001302 031014 DH1 ;ERRPC TSTNUM BUSADR EXPCT RCVD
45 001304 031062 DT1 ;$ERRPC TSTNUM $BDADR $GDDAT $BDDAT
46 001306 000000 0
47
48 ;ITEM 3
49
50 001310 027274 EM3 ;LOADING MODE REGISTER
51 001312 031014 DH1 ;ERRPC TSTNUM BUSADR EXPCT RCVD
52 001314 031062 DT1 ;$ERRPC TSTNUM $BDADR $GDDAT $BDDAT
53 001316 000000 0
54
55 ;ITEM 4
56
57 001320 027315 EM4 ;SEQUENCING DPC REGISTER
58 001322 031014 DH1 ;ERRPC TSTNUM BUSADR EXPCT RCVD
59 001324 031062 DT1 ;$ERRPC TSTNUM $BDADR $GDDAT $BDDAT
60 001326 000000 0
61
62 ;ITEM 5
63
64 001330 027340 EM5 ;LOADING LINE TYPE REGISTER
65 001332 031014 DH1 ;ERRPC TSTNUM BUSADR EXPCT RCVD
66 001334 031062 DT1 ;$ERRPC TSTNUM $BDADR $GDDAT $BDDAT
67 001336 000000 0
```

69			:ITEM 6	
70				
71	001340	027366	EM6	:LOADING INTENSITY LEVEL REGISTER
72	001342	031014	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD
73	001344	031062	DT1	:\$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT
74	001346	000000	0	
75				
76			:ITEM 7	
77				
78	001350	027422	EM7	:LOADING GRAPHLOT INCREMENT REGISTER
79	001352	031014	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD
80	001354	031062	DT1	:\$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT
81	001356	000000	0	
82				
83			:ITEM 10	
84				
85	001360	027454	EM10	:DISPLAY JUMP ABSOLUTE TO AN ADDRESS
86	001362	031014	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD
87	001364	031062	DT1	:\$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT
88	001366	000000	0	
89				
90			:ITEM 11	
91				
92	001370	027510	EM11	:DISPLAY JUMP RELATIVE TO AN ADDRESS
93	001372	031014	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD
94	001374	031062	DT1	:\$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT
95	001376	000000	0	
96				
97			:ITEM 12	
98				
99	001400	027544	EM12	:DISPLAY JSR ABSOLUTE TO AN ADDRESS
100	001402	031014	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD
101	001404	031062	DT1	:\$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT
102	001406	000000	0	
103				
104			:ITEM 13	
105				
106	001410	027577	EM13	:DISPLAY JSR RELATIVE TO AN ADDRESS
107	001412	031014	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD
108	001414	031062	DT1	:\$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT
109	001416	000000	0	
110				
111			:ITEM 14	
112				
113	001420	027632	EM14	:LOADING X POSITION REGISTER
114	001422	031014	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD
115	001424	031062	DT1	:\$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT
116	001426	000000	0	

268			:ITEM 42		
269					
270	001700	030531	EM42	:L.P. INTR ENABLE - CONSOLE 0/1 ERROR	
271	001702	031014	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD	
272	001704	031062	DT1	:\$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT	
273	001706	000000	0		
274					
275			:ITEM 43		
276					
277	001710	030572	EM43	:L.P. SWITCH INTR ENABLE - CONSOLE 0/1 ERROR	
278	001712	031014	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD	
279	001714	031062	DT1	:\$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT	
280	001716	000000	0		
281					
282			:ITEM 44		
283					
284	001720	030642	EM44	:DISPLAY BUSY ERROR	
285	001722	031014	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD	
286	001724	031062	DT1	:\$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT	
287	001726	000000	0		
288					
289			:ITEM 45		
290	001730	030662	EM45	:EXTERNAL STOP LOGIC ERROR	
291	001732	031014	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD	
292	001734	031062	DT1	:\$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT	
293	001736	000000	0		
294					
295			:ITEM 46		
296					
297	001740	030711	EM46	:TIME-OUT INTERRUPT ERROR	
298	001742	031014	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD	
299	001744	031062	DT1	:\$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT	
300	001746	000000	0		
301					
302			:ITEM 47		
303					
304	001750	030732	EM47	:NAME MATCH INTERRUPT ERROR	
305	001752	031014	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD	
306	001754	031062	DT1	:\$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT	
307	001756	000000	0		
308					
309			:ITEM 50		
310					
311	001760	030755	EM50	:L.P. FLAG INTERRUPT ERROR	
312	001762	031014	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD	
313	001764	031062	DT1	:\$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT	
314	001766	000000	0		
315					
316			:ITEM 51		
317					
318	001770	030777	EM51	:STACK PTR ER	
319	001772	031014	DH1	:ERRPC TSTNUM BUSADR EXPCT RCVD	
320	001774	031062	DT1	:\$ERRPC TSTNUM \$BDADR \$GDDAT \$BDDAT	
321	001776	000000	0		

```

323
324 ;COMMON PROGRAM TAGS AND STORAGE LOCATIONS
325
326 002000 031076 DBUF: BUFFER ;FIRST WORD IN THE DISPLAY BUFFER
327 002002 031100 DBUF1: BUFFER+2 ;SECOND WORD
328 002004 031102 DBUF2: BUFFER+4 ;THIRD WORD
329 002006 031104 DBUF3: BUFFER+6 ;FOURTH WORD
330 002010 031106 DBUF4: BUFFER+10 ;FIFTH WORD
331 002012 031110 DBUF5: BUFFER+12 ;SIXTH WORD
332 002014 000000 DSAVE: 0 ;TEMP REG.
333 002016 000000 DSAVE1: 0
334 002020 000000 SIZE: 0 ;BUFFER SIZE
335 002022 000000 CNTR: 0
336 002024 000000 TSTNUM: 0 ;CONTAINS TEST NO. ON ER
337 002026 000020 DELAY: 20 ;COUNT USED FOR GENERAL PURPOSE DELAYS
350
354 ;VS60 ADDRESSES AND VECTORS POINTERS
355
356 002030 172000 DPC: 172000 ;DISPLAY PC REGISTER
357 002032 172002 SREG0: 172002 ;DISPLAY STATUS REGISTER
358 002034 172004 XPOS: 172004 ;X AXIS REGISTER AND GRAPHPLOT REG <READ ONLY>
359 002036 172006 YPOS: 172006 ;Y AXIS REGISTER AND CHARACTER REG <READ ONLY>
360 002040 172010 RLO: 172010 ;RELOCATION REGISTER
361 002042 172012 SREG1: 172012 ;MISC STATUS REGISTER #1
362 002044 172014 XDOFF: 172014 ;X DYNAMIC OFFSET REGISTER
363 002046 172016 YDOFF: 172016 ;Y DYNAMIC OFFSET REGISTER
364 002050 172020 ANAME: 172020 ;ASSOCIATIVE NAME REGISTER
365 002052 172022 CONS: 172022 ;CONSOLE STATUS REGISTER
366 002054 172024 DNAME: 172024 ;DISPLAY NAME REGISTER
367 002056 172026 STKVAL: 172026 ;VALUE OF CURRENT STACK WORD
368 002060 172030 TERMCH: 172030 ;TERMINATE CHARACTER REGISTER
369 002062 172032 STKPT: 172032 ;STACK POINTER REGISTER
370 002064 172034 ZPOS: 172034 ;Z POSITION REGISTER <READ ONLY>
371 002066 172036 ZDOFF: 172036 ;Z DYNAMIC OFFSET REGISTER
372
373 002070 000320 DDONE: 320 ;DISPLAY STOP <DONE> VECTOR
374 002072 000322 DDONE1: 322 ;
375
376 002074 000324 LPVCT: 324 ;DISPLAY LIGHT PEN VECTOR
377 002076 000326 LPVCT1: 326 ;
378
379 002100 000330 TIMEVT: 330 ;DISPLAY TIME-OUT <NXM.> ERROR VECTOR
380 002102 000332 TMEVT1: 332 ; OR "SHIFT-OUT" VECTOR
381
382 002104 000334 NAMEVT: 334 ;NAME MATCH VECTOR
383 002106 000336 NAMEV1: 336

```

```

385                                     ;SOFTWARE INITIALIZATION ROUTINE
386
388 002110                               BEGIN:
(1)                                     .SBTTL INITIALIZE THE COMMON TAGS
(1)                                     ;;CLEAR THE COMMON TAGS (%CMTAG) AREA
(1) 002110 012706 001100                MOV    #CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
(1) 002114 005026                        CLR    (R6)+         ;;CLEAR MEMORY LOCATION
(1) 002116 022706 001140                CMP    #SWR,R6     ;;DONE?
(1) 002122 001374                        BNE    -6           ;;LOOP BACK IF NO
(1) 002124 012706 001100                MOV    #STACK,SP   ;;SETUP THE STACK POINTER
(1)                                     ;;INITIALIZE A FEW VECTORS
(1) 002130 012737 024526 000020          MOV    #SCOPE,@IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
(1) 002136 012737 000340 000022          MOV    #340,@IOTVEC+2 ;;LEVEL 7
(1) 002144 012737 025252 000030          MOV    #ERROR,@EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
(1) 002152 012737 000340 000032          MOV    #340,@EMTVEC+2 ;;LEVEL 7
(1) 002160 012737 027062 000034          MOV    #TRAP,@TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
(1) 002166 012737 000340 000036          MOV    #340,@TRAPVEC+2;LEVEL 7
(1) 002174 012737 026262 000024          MOV    #PWRDN,@PWRVEC ;;POWER FAILURE VECTOR
(1) 002202 012737 000340 000026          MOV    #340,@PWRVEC+2 ;;LEVEL 7
(1) 002210 013737 024366 024360          MOV    $ENDCT,$EOPCT ;;SETUP END-OF-PROGRAM COUNTER
(1) 002216 005037 001166                  CLR    $TIMES      ;;INITIALIZE NUMBER OF ITERATIONS
(1) 002222 005037 001170                  CLR    $ESCAPE     ;;CLEAR THE ESCAPE ON ERROR ADDRESS
(1) 002226 112737 000001 001115          MOV    #1,$ERMAX   ;;ALLOW ONE ERROR PER TEST
(1) 002234 012737 002234 001106          MOV    #,$LPADR    ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
(1) 002242 012737 002242 001110          MOV    #,$LPERR    ;;SETUP THE ERROR LOOP ADDRESS
(2)                                     ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
(2)                                     ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
(2) 002250 013746 000004                  MOV    @ERRVEC,-(SP) ;;SAVE ERROR VECTOR
(2) 002254 012737 002310 000004          MOV    #64,$@ERRVEC ;;SET UP ERROR VECTOR
(2) 002262 012737 177570 001140          MOV    #DSWR,SWR   ;;SETUP FOR A HARDWARE SWICH REGISTER
(2) 002270 012737 177570 001142          MOV    #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
(2) 002276 022777 177777 176634          CMP    #-1,@SWR    ;;TRY TO REFERENCE HARDWARE SWR
(2) 002304 001012                        BNE    66$         ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
(2)                                     ;;AND THE HARDWARE SWR IS NOT = -1
(2) 002306 000403                        BR     65$         ;;BRANCH IF NO TIMEOUT
(2) 002310 012716 002316                  64$: MOV    #65$,(SP) ;;SET UP FOR TRAP RETURN
(2) 002314 000002                        RTI
(2) 002316 012737 000176 001140          65$: MOV    #SWREG,SWR ;;POINT TO SOFTWARE SWR
(2) 002324 012737 000174 001142          MOV    #DISPREG,DISPLAY
(2) 002332 012637 000004                  66$: MOV    (SP)+,@ERRVEC ;;RESTORE ERROR VECTOR
(1)
(2) 002336 005037 001210                  CLR    $PASS       ;;CLEAR PASS COUNT
(2) 002342 132737 000200 001223          BITB  #APTSIZE,$ENVM ;;TEST USER SIZE UNDER APT
(2) 002350 001403                        BEQ    67$         ;;YES,USE NON-APT SWITCH
(2) 002352 012737 001224 001140          MOV    #SSWREG,SWR ;;NO,USE APT SWITCH REGISTER
(2) 002360
389 002360 012700 002030                  67$: MOV    #DPC,R0  ;;SET UP VS60 REG ADRS POINTERS
390 002364 013701 001256                  MOV    $BASE,R1    ;;GET VS60 BASE ADRS FROM APT MAILBOX-ETABLE
391 002370 010120                        SETUP2: MOV   R1,(0)+
392 002372 062701 000002                  ADD    #2,R1
393 002376 022700 002070                  CMP    #DDONE,R0   ;;TEST FOR LAST ADDRESS
394 002402 001372                        BNE    SETUP2
395 002404 012700 002070                  MOV    #DDONE,R0   ;;SET UP VS60 VECTOR ADRS POINTERS
396 002410 013701 001252                  MOV    $VECT1,R1  ;;GET VS60 BASE VECTOR ADRS FROM APT MAILBOX-ETABLE
397 002414 042701 160000                  BIC    #160000,R1 ; CLEAR PSW BITS      ** B **
398 002420 010120                        SETUP3: MOV   R1,(0)+

```

```
399 002422 062701 000002      ADD      #2,R1
400 002426 022700 002110      CMP      #BEGIN,R0      ;TEST FOR LAST VECTOR
401 002432 001372              BNE      SETUP3
402 002434 004737 024454      JSR      PC,RSTVEC      ;GO SET UP VS60 VECTOR ADR'S WITH HLTS
403 002440 004737 026502      JSR      PC,$SIZE      ;DETERMINE LAST MEM LOCATION
404 002444 013737 026576 002020  MOV      $LSTAD,SIZE    ;LOAD LAST MEM ADDRESS
405 002452 162737 006000 002020  SUB      #6000,SIZE    ;SAVE LOADERS AND XXDP
406 002460 005737 000042      TST      @#42          ;TEST IF RUNNING CHAIN MODE UNDER XXDP
407 002464 001002              BNE      RESTRT        ;BR IF SO
408 002466 104401              TYPE
409 002470 027132              HEADER
410 002472 012737 000340 177776 RESTRT: MOV      #340,PSW      ;SET PRIORITY TO HIGHEST LEVEL
411 002500 012706 001100      MOV      #STACK,SP    ;ALWAYS SET UP STACK POINTER
412 002504 000005              RESET
413 002506 105037 001102      CLRB     $STNM        ;START TESTS WITH VS60 INITIALIZED
                          ;RESET TEST NUMBER
```

415
(3)
(3)
(2) 002512 000004
(2) 002514 012737 000001 001206
416 002522 012737 002556 001110
417 002530 012737 002576 000004
418 002536 012737 000340 000006
419 002544 013737 001256 001122
420 002552 012700 000020
421 002556 005777 176340
422 002562 062737 000002 001122
423 002570 005300
424 002572 001371
425 002574 000402
426 002576 022626
427 002600 104016
428 002602 012737 000006 000004
429 002610 005037 000006
430
431
(3)
(3)
(2) 002614 000004
(2) 002616 012737 000002 001206
432 002624 012737 002646 001110
433 002632 013737 002060 001122
434 002640 012737 000100 001124
435 002646 013700 001124
436 002652 052700 000200
437 002656 010077 177176
438 002662 017737 177172 001126
439 002670 023737 001124 001126
440 002676 001401
441 002700 104017
442 002702 006237 001124
443 002706 001357
444
445
(3)
(3)
(2) 002710 000004
(2) 002712 012737 000003 001206
446 002720 012737 002746 001110
447 002726 013737 002060 001122
448 002734 012737 000077 001124
449 002742 012700 177677
450 002746 010077 177106
451 002752 017737 177102 001126
452 002760 023737 001124 001126
453 002766 001401
454 002770 104017
455 002772 006200
456 002774 006237 001124
457 003000 052737 000100 001124
458 003006 103757

```
:::*****  
:*TEST 1 SEE IF ALL VS60 REGS ARE THERE  
:::*****  
TST1: SCOPE  
MOV #1,$STESTN ;;SET TEST NUMBER IN APT MAIL BOX  
MOV #1$, $LPERR ;;SET UP SCOPE LOOP ADRS  
MOV #2$, @#ERRVEC ;;SET UP TIME OUT TRAP  
MOV #340, @#ERRVEC+2 ;;SET UP TIME OUT PRIORITY  
MOV $BASE, $BDADR ;;PUT 1ST VS60 BUS ADRS IN LOC $BDADR  
MOV #20, R0 ;;SET UP REG COUNT  
1$: TST @ $BDADR ;;LOOK FOR VS60 REG  
ADD #2, $BDADR ;;OK-ADVANCE TO NEXT  
DEC R0 ;;HAVE WE TRIED ALL REGS?  
BNE 1$ ;;BR IF NOT  
BR 3$ ;;GO RESTORE LOCS 4 & 6  
2$: CMP (R6)+, (R6)+ ;;FIX STK SINCE NO RTI  
ERROR 16 ;;REG ADRS TIME OUT ER  
3$: MOV #ERRVEC+2, @#ERRVEC ;;POINT TO HALT IN ADRS 6  
CLR @#ERRVEC+2 ;;SET UP HALT  
  
:::*****  
:*TEST 2 FLOAT A ONE THRU TERMINATE CHARACTER REG  
:::*****  
TST2: SCOPE  
MOV #2,$STESTN ;;SET TEST NUMBER IN APT MAIL BOX  
MOV #1$, $LPERR ;;SET UP SCOPE LOOP ADRS  
MOV TERMCH, $BDADR ;;SET UP TERMCH ADRS  
MOV #100, $GDDAT ;;EXPECT 100 INITIALLY  
1$: MOV $GDDAT, R0 ;;PUT PATTERN IN R0  
BIS #200, R0 ;;SET THE ENABLE BIT  
MOV R0, @TERMCH ;;LOAD UP TERMCH REG  
MOV @TERMCH, $BDDAT ;;READ IT BACK  
CMP $GDDAT, $BDDAT ;;IS IT CORRECT?  
BEQ 2$ ;;BR IF OK  
ERROR 17 ;;TERMINATE CHARACTER READ/WRITE ER  
2$: ASR $GDDAT ;;SHIFT ONE BIT RIGHT  
BNE 1$ ;;BR IF NOT DONE  
  
:::*****  
:*TEST 3 FLOAT A ZERO THRU TERMINATE CHARACTER REG  
:::*****  
TST3: SCOPE  
MOV #3,$STESTN ;;SET TEST NUMBER IN APT MAIL BOX  
MOV #1$, $LPERR ;;SET UP SCOPE LOOP ADRS  
MOV TERMCH, $BDADR ;;SET UP TERMCH ADRS  
MOV #77, $GDDAT ;;EXPECT 77 INITIALLY  
MOV #177677, R0 ;;PUT PATTERN IN R0  
1$: MOV R0, @TERMCH ;;LOAD UP TERMCH REG - SEND HIGH BITS FOR NOISE  
MOV @TERMCH, $BDDAT ;;READ IT BACK  
CMP $GDDAT, $BDDAT ;;IS IT CORRECT?  
BEQ 2$ ;;BR IF OK  
ERROR 17 ;;TERMINATE CHARACTER READ/WRITE ER  
2$: ASR R0 ;;SHIFT ZERO BIT RIGHT  
ASR $GDDAT ;;SHIFT PATTERN IN COMPARE LOC  
BIS #100, $GDDAT ;;SET MSB  
1$: BCS 1$ ;;BR IF NOT DONE
```

459
460
(3)
(3)
(2) 003010 000004
(2) 003012 012737 000004 001206
461 003020 013737 002060 001122
462 003026 012737 000177 001124
463 003034 012777 177777 177016
464 003042 005077 177012
465 003046 017737 177006 001126
466 003054 023737 001124 001126
467 003062 001401
468 003064 104017
469
470
(3)
(3)
(2) 003066 000004
(2) 003070 012737 000005 001206
471 003076 012737 003120 001110
472 003104 013737 002046 001122
473 003112 012737 004000 001124
474 003120 013777 001124 176720
475 003126 017737 176714 001126
476 003134 042737 170000 001126
477 003142 023737 001124 001126
478 003150 001401
479 003152 104020
480 003154 006237 001124
481 003160 001357
482
483
(3)
(3)
(2) 003162 000004
(2) 003164 012737 000006 001206
484 003172 012737 003212 001110
485 003200 013737 002046 001122
486 003206 012700 173777
487 003212 010037 001124
488 003216 042737 170000 001124
489 003224 010077 176616
490 003230 017737 176612 001126
491 003236 042737 170000 001126
492 003244 023737 001124 001126
493 003252 001401
494 003254 104020
495 003256 006200
496 003260 103754
497 003262 005077 176560
498
499
(3)
(3)
(2) 003266 000004

*TEST 4 TEST TERMINATE CHARACTER REG IS NOT WRITEABLE WHEN CHANGE ENA=0

TST4: SCOPE
MOV #4,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV TERMCH,\$BDADR ;;SET UP TERMCH ADRS
MOV #177,\$GDDAT ;;SET UP DATA PATTERN
MOV #177777,@TERMCH ;LOAD UP ALL WRITEABLE BITS
CLR @TERMCH ;TRY TO CLEAR ALL BITS
MOV @TERMCH,\$BDDAT ;READ IT BACK
CMP \$GDDAT,\$BDDAT ;SEE IF ANY BITS WERE CLEARED
BEQ TST5 ;ADVANCE TO NEXT TEST IF OK
ERROR 17 ;TERMINATE CHARACTER ENABLE CHANGE ER

*TEST 5 FLOAT A ONE THRU Y DYNAMIC OFFSET REG

TST5: SCOPE
MOV #5,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #1\$,\$LPERR ;SET UP SCOPE LOOP ADRS
MOV YDOFF,\$BDADR ;SET UP YDOFF ADRS
MOV #4000,\$GDDAT ;SET UP MSB TO A ONE
1\$: MOV \$GDDAT,@YDOFF ;LOAD UP YDOFF REG
MOV @YDOFF,\$BDDAT ;READ IT BACK
BIC #170000,\$BDDAT ;SAVE ONLY OFFSET VALUE
CMP \$GDDAT,\$BDDAT ;IS IT CORRECT?
BEQ 2\$;BR IF OK
ERROR 20 ;Y OFFSET READ/WRITE ER
2\$: ASR \$GDDAT ;SHIFT ONE BIT RIGHT
BNE 1\$;BR IF NOT DONE

*TEST 6 FLOAT A ZERO THRU Y DYNAMIC OFFSET REG

TST6: SCOPE
MOV #6,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #1\$,\$LPERR ;SET UP SCOPE LOOP ADRS
MOV YDOFF,\$BDADR ;SET UP YDOFF ADRS
MOV #173777,R0 ;SET UP MSB TO A ZERO
1\$: MOV R0,\$GDDAT ;SAVE PATTERN FOR COMPARE
BIC #170000,\$GDDAT ;GET RID OF HIGH BITS THAT ARE NOT WRITEABLE
MOV R0,@YDOFF ;LOAD UP YDOFF REG - SEND HIGH BITS FOR NOISE
MOV @YDOFF,\$BDDAT ;READ IT BACK
BIC #170000,\$BDDAT ;SAVE ONLY OFFSET VALUE
CMP \$GDDAT,\$BDDAT ;IS IT CORRECT?
BEQ 2\$;BR IF OK
ERROR 20 ;Y OFFSET READ/WRITE ER
2\$: ASR R0 ;SHIFT ZERO BIT RIGHT
BCS 1\$;BR IF NOT DONE
CLR @YDOFF ;INSURE 0 Y OFFSET BEFORE ADVANCING

*TEST 7 FLOAT A ONE THRU X DYNAMIC OFFSET REG

TST7: SCOPE

```

(2) 003270 012737 000007 001206      MOV      #7,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
500 003276 012737 003320 001110      MOV      #1$, $LPERR    ;SET UP SCOPE LOOP ADRS
501 003304 013737 002044 001122      MOV      XDOFF,$BDADR   ;SET UP XDOFF ADRS
502 003312 012737 004000 001124      MOV      #4000,$GDDAT   ;SET UP MSB TO A ONE
503 003320 013777 001124 176516 1$:  MOV      $GDDAT,@XDOFF  ;LOAD UP XDOFF REG
504 003326 017737 176512 001126      MOV      @XDOFF,$BDDAT ;READ IT BACK
505 003334 042737 170000 001126      BIC      #170000,$BDDAT ;SAVE ONLY OFFSET VALUE
506 003342 023737 001124 001126      CMP      $GDDAT,$BDDAT ;IS IT CORRECT?
507 003350 001401                BEQ      2$             ;BR IF OK
508 003352 104021                ERROR    21            ;X OFFSET READ/WRITE ER
509 003354 006237 001124 2$:  ASR      $GDDAT        ;SHIFT ONE BIT RIGHT
510 003360 001357                BNE      1$            ;BR IF NOT DONE

```

```

*****
;*TEST 10      FLOAT A ZERO THRU X DYNAMIC OFFSET REG
*****

```

```

(2) 003362 000004      TST10:  SCOPE
(2) 003364 012737 000010 001206      MOV      #10,$TESTN    ;;SET TEST NUMBER IN APT MAIL BOX
513 003372 012737 003412 001110      MOV      #1$, $LPERR   ;SET UP SCOPE LOOP ADRS
514 003400 013737 002044 001122      MOV      XDOFF,$BDADR  ;SET UP XDOFF ADRS
515 003406 012700 173777                MOV      #173777,R0    ;SET MBB TO A ZERO
516 003412 010037 001124 1$:  MOV      R0,$GDDAT     ;SAVE PATTERN FOR COMPARE
517 003416 042737 170000 001124      BIC      #170000,$GDDAT ;GET RID IF HIGH BITS THAT ARE NOT WRITEABLE
518 003424 010077 176414                MOV      R0,@XDOFF    ;LOAD UP XDOFF REG - SEND HIGH BITS FOR NOISE
519 003430 017737 176410 001126      MOV      @XDOFF,$BDDAT ;READ IT BACK
520 003436 042737 170000 001126      BIC      #170000,$BDDAT ;SAVE ONLY OFFSET VALUE
521 003444 023737 001124 001126      CMP      $GDDAT,$BDDAT ;IS IT CORRECT?
522 003452 001401                BEQ      2$             ;BR IF OK
523 003454 104021                ERROR    21            ;X OFFSET READ/WRITE ER
524 003456 006200 2$:  ASR      R0            ;SHIFT ZERO BIT RIGHT
525 003460 103754                BCS     1$             ;BR IF NOT DONE
526 003462 005077 176356                CLR      @XDOFF        ;INSURE 0 X OFFSET BEFORE ADVANCING

```

```

*****
;*TEST 11      TEST THAT THE Y OFFSET POLARITY BIT CAN SET & RESET
*****

```

```

(2) 003466 000004      TST11:  SCOPE
(2) 003470 012737 000011 001206      MOV      #11,$TESTN    ;;SET TEST NUMBER IN APT MAIL BOX
529 003476 012737 003524 001110      MOV      #1$, $LPERR   ;SET UP SCOPE LOOP ADRS
530 003504 013737 002066 001122      MOV      ZDOFF,$BDADR  ;SET UP REG 36 ADRS
531 003512 012700 020000                MOV      #20000,R0     ;R0 HAS BIT
532 003516 012737 040000 001124      MOV      #40000,$GDDAT ;LD EXPCTED
533 003524 010077 176316 1$:  MOV      R0,@YDOFF    ;SEND IT OUT
534 003530 017737 176332 001126      MOV      @ZDOFF,$BDDAT ;READ AT REG 36
535 003536 042737 037777 001126      BIC      #37777,$BDDAT ;SAVE X & Y POLARITY BITS
536 003544 023737 001124 001126      CMP      $GDDAT,$BDDAT ;CORRECT?
537 003552 001401                BEQ      2$             ;BR IF SO
538 003554 104020                ERROR    20            ;Y OFFSET POLARITY READ/WRITE ER
539 003556 005037 001124 2$:  CLR      $GDDAT        ;EXPECT ZERO
540 003562 162700 020000                SUB     #20000,R0      ;NOW RESET STATE
541 003566 100356                BPL     1$             ;BR IF NOT YET TESTED

```

```

*****
;*TEST 12      TEST THAT THE X OFFSET POLARITY BIT CAN SET & RESET
*****

```

```

(2) 003570 000004      TST12:  SCOPE

```

```
(2) 003572 012737 000012 001206      MOV      #12,$STESTN      ;;SET TEST NUMBER IN APT MAIL BOX
544 003600 012737 003626 001110      MOV      #1$, $LPERR      ;;SET UP SCOPE LOOP ADRS
545 003606 013737 002066 001122      MOV      ZDOFF,$BDADR     ;;SET UP REG 34 ADRS
546 003614 012700 020000      MOV      #20000,R0        ;;R0 HAS BIT
547 003620 012737 100000 001124      MOV      #100000,$GDDAT   ;;LD EXPECTED
548 003626 010077 176212      1$:     MOV      R0,@XDOFF     ;;SEND IT OUT
549 003632 017737 176230 001126      MOV      @ZDOFF,$BDDAT   ;;READ IT BACK
550 003640 042737 037777 001126      BIC      #37777,$BDDAT   ;;SAVE X & Y POLARITY BITS
551 003646 023737 001124 001126      CMP      $GDDAT,$BDDAT   ;;IS IT CORRECT?
552 003654 001401      BEQ      2$              ;;BR IF SO
553 003656 104021      ERROR   21              ;;X OFFSET POLARITY READ/WRITE ER
554 003660 005037 001124      2$:     CLR      $GDDAT      ;;EXPECT ZERO
555 003664 162737 020000 001124      SUB      #20000,$GDDAT   ;;ADVANCE TO RESET STATE
556 003672 100355      B-PL    1$              ;;BR IF NOT YET TESTED
```

*TEST 13 FLOAT A ONE THRU RELOCATE REG

```
TST13: SCOPE
(2) 003674 000004      MOV      #13,$STESTN     ;;SET TEST NUMBER IN APT MAIL BOX
(2) 003676 012737 000013 001206      MOV      #1$, $LPERR     ;;SET UP SCOPE LOOP ADRS
559 003704 012737 003726 001110      MOV      RLO,$BDADR      ;;SET UP RLO ADRS
560 003712 013737 002040 001122      MOV      #4000,$GDDAT    ;;SET UP MSB TO A ONE
561 003720 012737 004000 001124      1$:     MOV      $GDDAT,@RLO    ;;LOAD UP RLO REG
562 003726 013777 001124 176104      MOV      @RLO,$BDDAT    ;;READ IT BACK
563 003734 017737 176100 001126      CMP      $GDDAT,$BDDAT  ;;IS IT CORRECT?
564 003742 023737 001124 001126      BEQ      2$              ;;BR IF OK
565 003750 001401      ERROR   22              ;;RELOCATE REG READ/WRITE ER
566 003752 104022      2$:     ASR      $GDDAT      ;;SHIFT ONE BIT RIGHT
567 003754 006237 001124      BNE     1$              ;;BR IF NOT DONE
568 003760 001362
```

*TEST 14 FLOAT A ZERO THRU RELOCATE REG

```
TST14: SCOPE
(2) 003762 000004      MOV      #14,$STESTN     ;;SET TEST NUMBER IN APT MAIL BOX
(2) 003764 012737 000014 001206      MOV      #1$, $LPERR     ;;SET UP SCOPE LOOP ADRS
571 003772 012737 004012 001110      MOV      RLO,$BDADR      ;;SET UP RLO ADRS
572 004000 013737 002040 001122      MOV      #173777,R0      ;;SET MSB TO A ZERO
573 004006 012700 173777      1$:     MOV      R0,$GDDAT      ;;SAVE PATTERN FOR COMPARE
574 004012 010037 001124      BIC      #170000,$GDDAT  ;;GET RID OF HIGH BITS THAT DO NOT READ/WRITE
575 004016 042737 170000 001124      MOV      R0,@RLO        ;;LOAD UP RLO REG - SEND HIGH BITS FOR NOISE
576 004024 010077 176010      MOV      @RLO,$BDDAT    ;;READ IT BACK
577 004030 017737 176004 001126      CMP      $GDDAT,$BDDAT  ;;IS IT CORRECT?
578 004036 023737 001124 001126      BEQ      2$              ;;BR IF OK
579 004044 001401      ERROR   22              ;;RELOCATE REG READ/WRITE ER
580 004046 104022      2$:     ASR      R0              ;;SHIFT ZERO BIT RIGHT
581 004050 006200      BCS     1$              ;;BR IF NOT DONE
582 004052 103757      CLR     @RLO            ;;CLR RELOCATE REG BEFORE ADVANCING
583 004054 005077 175760
```

*TEST 15 TEST THAT ALL MAINT TYPE BITS ARE READ/WRITEABLE

```
TST15: SCOPE
(2) 004060 000004      MOV      #15,$STESTN     ;;SET TEST NUMBER IN APT MAIL BOX
(2) 004062 012737 000015 001206      MOV      STKPT,$BDADR    ;;SET UP REG 32 ADRS
586 004070 013737 002062 001122
```

587	004076	012737	004112	001110		MOV	#1\$,SLPERR	:SET UP SCOPE LOOP ADRS
588	004104	012737	170000	001124		MOV	#170000,\$GDDAT	:EXPECT ALL BITS INITIALLY
589	004112	013777	001124	175742	1\$:	MOV	\$GDDAT,@STKPT	:LOAD UP MAINT BITS
590	004120	017737	175736	001126		MOV	@STKPT,\$BDDAT	:READ THEM BACK
591	004126	042737	007777	001126		BIC	#7777,\$BDDAT	:SAVE ONLY MAINT SWITCHES
592	004134	023737	001124	001126		CMP	\$GDDAT,\$BDDAT	:ARE THEY CORRECT?
593	004142	001401				BEQ	2\$:BR IF SO
594	004144	104001				ERROR	1	:MAINT SWITCH READ/WRITE ERROR
595	004146	162737	010000	001124	2\$:	SUB	#10000,\$GDDAT	:GO TO NEXT PATTERN
596	004154	100356				BPL	1\$:BR IF ALL STATES NOT TESTED

597
598
(3) :*****
(3) :*TEST 16 TEST THAT RESET WILL CLEAR THE RELOCATE REG
:*****

(2) 004156 000004 TST16: SCOPE
(1) 004160 012737 000010 001166 MOV #10,\$TIMES ;;DO 10 ITERATIONS
(2) 004166 012737 000016 001206 MOV #16,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
599 004174 013737 002040 001122 MOV RLO,\$BDADR ;SET UP REG 10 ADRS
600 004202 005037 001124 CLR \$GDDAT ;EXPECT ALL ZEROS
601 004206 012777 007777 175624 MOV #7777,@RLO ;LOAD IT UP
602 004214 000005 RESET ;CLR IT OUT
603 004216 017737 175616 001126 MOV @RLO,\$BDDAT ;READ IT
604 004224 001401 BEQ TST17 ;;GO TO NEXT TEST IF CLEARED
605 004226 104022 ERROR 22 ;RESET FAILED TO CLR RELOCATE REG

606
607
(3) :*****
(3) :*TEST 17 TEST THAT RESET WILL CLEAR THE X DYNAMIC OFFSET REG
:*****

(2) 004230 000004 TST17: SCOPE
(1) 004232 012737 000010 001166 MOV #10,\$TIMES ;;DO 10 ITERATIONS
(2) 004240 012737 000017 001206 MOV #17,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
608 004246 013737 002044 001122 MOV XDOFF,\$BDADR ;SET UP REG 14 ADRS
609 004254 005037 001124 CLR \$GDDAT ;EXPECT ALL ZEROS
610 004260 012777 007777 175556 MOV #7777,@XDOFF ;LOAD IT UP
611 004266 000005 RESET ;CLR IT OUT
612 004270 017737 175550 001126 MOV @XDOFF,\$BDDAT ;READ IT
613 004276 001401 BEQ TST20 ;;GO TO NEXT TEST IF CLEARED
614 004300 104021 ERROR 21 ;RESET FAILED TO CLR X DYNAMIC OFFSET REG

615
616
(3) :*****
(3) :*TEST 20 TEST THAT RESET WILL CLEAR THE Y DYNAMIC OFFSET REG
:*****

(2) 004302 000004 TST20: SCOPE
(1) 004304 012737 000010 001166 MOV #10,\$TIMES ;;DO 10 ITERATIONS
(2) 004312 012737 000020 001206 MOV #20,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
617 004320 013737 002046 001122 MOV YDOFF,\$BDADR ;SET UP REG 16 ADRS
618 004326 005037 001124 CLR \$GDDAT ;EXPECT ALL ZEROS
619 004332 012777 007777 175506 MOV #7777,@YDOFF ;LOAD IT UP
620 004340 000005 RESET ;CLR IT OUT
621 004342 017737 175500 001126 MOV @YDOFF,\$BDDAT ;READ IT
622 004350 001401 BEQ TST21 ;;GO TO NEXT TEST IF CLEARED
623 004352 104020 ERROR 20 ;RESET FAILED TO CLR Y DYNAMIC OFF SET REG

624
625
(3) :*****
(3) :*TEST 21 TEST THAT RESET WILL CLEAR X & Y OFFSET POLARITY BITS
:*****

(2) 004354 000004
(1) 004356 012737 000040 001166
(2) 004364 012737 000021 001206
626 004372 013737 002066 001122
627 004400 005037 001124
628 004404 012777 020000 175432
629 004412 012777 020000 175426
630 004420 000005
631 004422 017737 175440 001126
632 004430 042737 037777 001126
633 004436 023737 001124 001126
634 004444 001401
635 004446 104032

TST21: SCOPE
MOV #40,\$TIMES ;;DO 40 ITERATIONS
MOV #21,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV ZDOFF,\$BDADR ;;SET UP REG 36 ADRS
CLR \$GDDAT ;;EXPECT ZERO
MOV #20000,@XDOFF ;;SET X POLARIYT BIT
MOV #20000,@YDOFF ;;SET Y POLARITY BIT
RESET ;;CLR THEM
MOV @ZDOFF,\$BDDAT ;;READ REG 36
BIC #37777,\$BDDAT ;;SAVE ONLY POLARITY BITS
CMP \$GDDAT,\$BDDAT ;;CORRECT?
BEQ TST22 ;;NEXT TEST IF SO
ERROR 32 ;;RESET FAILED TO CLR OFFSET POLARIY BITS

636
637
(3)
(3)
(2) 004450 000004
(1) 004452 012737 000010 001166
(2) 004460 012737 000022 001206
638 004466 013737 002060 001122
639 004474 005037 001124
640 004500 012777 000377 175352
641 004506 000005
642 004510 017737 175344 001126
643 004516 001401
644 004520 104017
645

*TEST 22 TEST THAT RESET WILL CLEAR TERMINATE CHARACTER REG

TST22: SCOPE
MOV #10,\$TIMES ;;DO 10 ITERATIONS
MOV #22,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV TERMCH,\$BDADR ;;SET UP REG 30 ADRS
CLR \$GDDAT ;;EXPECT ALL ZEROS
MOV #377,@TERMCH ;;LOAD IT UP
RESET ;;CLR IT OUT
MOV @TERMCH,\$BDDAT ;;READ IT
BEQ TST23 ;;GO TO NEXT TEST IF CLEARED
ERROR 17 ;;RESET FAILED TO CLR TERMINATE CHAR REG

646
(3)
(3)
(2) 004522 000004
(1) 004524 012737 000100 001166
(2) 004532 012737 000023 001206
647 004540 013737 002062 001122
648 004546 005037 001124
649 004552 012777 170000 175302
650 004560 000005
651 004562 017737 175274 001126
652 004570 042737 003777 001126
653 004576 001401
654 004600 104032
655

*TEST 23 TEST THAT RESET CLEARS ALL MAINT BITS

TST23: SCOPE
MOV #100,\$TIMES ;;DO 100 ITERATIONS
MOV #23,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV STKPT,\$BDADR ;;SET UP REG ADRS 32
CLR \$GDDAT ;;LOAD EXPECTED
MOV #170000,@STKPT ;;LOAD REGISTER
RESET ;;INITILIZE
MOV @STKPT,\$BDDAT ;;READ REG.
BIC #3777,\$BDDAT ;;MASK OUT OTHER BITS
BEQ TST24 ;;BR IF ALL CLEARED
ERROR 32 ;;RESET FAILED TO CLEAR MAINT. BITS

656
(3)
(3)
(2) 004602 000004
(2) 004604 012737 000024 001206
657 004612 012737 004634 001110
658 004620 013737 002062 001122
659 004626 012737 000037 001124
660 004634 013777 001124 175220
661 004642 017737 175214 001126
662 004650 042737 177740 001126
663 004656 023737 001124 001126
664 004664 001401

*TEST 24 TEST THAT THE STACK SELECTION BITS ARE READ/WRITEABLE

TST24: SCOPE
MOV #24,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #1\$, \$LPERR ;;SET UP SCOPE LOOP ADRS
MOV STKPT,\$BDADR ;;SET UP REG ADRS 32
MOV #37,\$GDDAT ;;EXPECT ALL BITS INITIALLY
1\$: MOV \$GDDAT,@STKPT ;;SEND PATTERN OUT
MOV @STKPT,\$BDDAT ;;READ IT BACK
BIC #177740,\$BDDAT ;;SAVE ONLY STACK SELECTION BITS
CMP \$GDDAT,\$BDDAT ;;CORRECT?
BEQ 2\$;;BR IF SO

665 004666 104051
666 004670 005337 001124
667 004674 100357
668
669
(3)
(3)
(2) 004676 000004
(1) 004700 012737 000040 001166
(2) 004706 012737 000025 001206
670 004714 013737 002062 001122
671 004722 012737 000041 001124
672 004730 012777 000037 175124
673 004736 012777 000077 175116
674 004744 017737 175112 001126
675 004752 042737 177700 001126
676 004760 023737 001124 001126
677 004766 001401
678 004770 104051
679 004772 012737 000040 001124
680 005000 012777 000037 175054
681 005006 000005
682 005010 017737 175046 001126
683 005016 023737 001124 001126
684 005024 001401
685 005026 104051
686
687
688
689
690
691
(3)
(3)
(2) 005030 000004
(1) 005032 012737 000040 001166
(2) 005040 012737 000026 001206
692 005046 013737 002030 001122
693 005054 012737 000002 001124
694 005062 012777 010000 174772
695 005070 013777 001124 174732
696 005076 017737 174726 001126
697 005104 023737 001124 001126
698 005112 001402
699 005114 104002
700 005116 000403
701 005120 006337 001124
702 005124 103361
703
704
(3)
(3)
(2) 005126 000004
(1) 005130 012737 000040 001166
(2) 005136 012737 000027 001206
705 005144 012737 005204 001110

ERROR 51 ;STACK POINTER READ/WRITE ER
2\$: DEC \$GDDAT ;ADVANCE PATTERN
BPL 1\$;BR IF ALL PATTERNS NOT TESTED

*TEST 25 TEST THAT 'TOP OF STACK' AND RESET CLEARS STACK SELECTION BITS

TST25: SCOPE
MOV #40,\$TIMES ;:DO 40 ITERATIONS
MOV #25,\$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
MOV STKPT,\$BDADR ;:SET UP REG ADRS 32
MOV #41,\$GDDAT ;:LOAD EXPECTED
MOV #37,@STKPT ;:SET ALL BITS
MOV #77,@STKPT ;:CLR ALL BITS
MOV @STKPT,\$BDDAT ;:READ STACK SELECTION BITS
BIC #177700,\$BDDAT ;:SAVE SEL BITS ONLY
CMP \$GDDAT,\$BDDAT ;:CORRECT?
BEQ 1\$;:BR IF SO
ERROR 51 ;:'TOP OF STACK' FAILED TO CLR BITS
1\$: MOV #40,\$GDDAT ;:LD EXPECTED
MOV #37,@STKPT ;:SET ALL BITS
RESET ;:RESET CLRS THEM & SET 'TOP OF STACK'
MOV @STKPT,\$BDDAT ;:READ THEM
CMP \$GDDAT,\$BDDAT ;:CORRECT?
BEQ TST26 ;:NEXT TEST IF SO
ERROR 51 ;:RESET FAILED TO SET UP 'TOP OF STACK'

.SBTTL
.SBTTL THESE TESTS USE MAINT SWITCH #1
.SBTTL

*TEST 26 FLOAT A 1 ACROSS THE D.P.C.

TST26: SCOPE
MOV #40,\$TIMES ;:DO 40 ITERATIONS
MOV #26,\$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
MOV DPC,\$BDADR ;:SET UP REG ADRS 00
MOV #BIT1,\$GDDAT ;:LOAD EXPECTED
MOV #BIT12,@STKPT ;:SET MAINT SWITCH #1
1\$: MOV \$GDDAT,@DPC ;:LOAD D.P.C.
MOV @DPC,\$BDDAT ;:READ D.P.C. INTO \$BDDAT
CMP \$GDDAT,\$BDDAT ;:COMPARE VALUE EXPECTED TO READ
BEQ 2\$;:BR IF SAME
ERROR 2 ;:FAILED TO LOAD DPC WITH A FLOATING 1
BR TST27 ;:
2\$: ASL \$GDDAT ;:CHANGE DATA EXPECTED
BCC 1\$;:BR IF NOT COMPLETED

*TEST 27 TEST THAT THE RELOCATE REG WILL CORRECTLY SET UP THE DPC

TST27: SCOPE
MOV #40,\$TIMES ;:DO 40 ITERATIONS
MOV #27,\$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
MOV #1\$,\$LPERR ;:SET UP SCOPE LOOP ADRS

```
706 005152 012777 010000 174702      MOV      #BIT12,@STKPT      ;SET MAINT SWITCH #1
707 005160 013737 002030 001122      MOV      DPC,$BDADR        ;SET UP REG 00 ADRS
708 005166 005077 174636                CLR      @DPC              ;CLEAR DPC
709 005172 012737 000100 001124      MOV      #100,$GDDAT       ;EXPECT 100 INITIALLY
710 005200 012700 000001                MOV      #1,R0            ;START WITH 1
711 005204 010077 174630                1$:     MOV      R0,@RLO       ;LOAD RELOCATE REG
712 005210 017737 174614 001126      MOV      @DPC,$BDDAT       ;READ RELOCATE REG - SHOULD BE TIMES 64
713 005216 023737 001124 001126      CMP      $GDDAT,$BDDAT     ;IS IT CORRECT?
714 005224 001401                BEQ      2$               ;BR IF SO
715 005226 104022                ERROR    22               ;RELOCATE FAILED TO SET UP DPC CORRECTLY
716 005230 062737 000100 001124      2$:     ADD      #100,$GDDAT    ;SET UP NEXT EXPECTED VALUE
717 005236 001402                BEQ      TST30            ;GO TO NEXT TEST IF ALL VALUES TESTED
718 005240 005200                INC      R0               ;SET UP NEXT PATTERN
719 005242 000760                BR       1$              ;GO TRY IT
```

```
720
721
(3)
(3)
(2) 005244 000004
(1) 005246 012737 000100 001166
(2) 005254 012737 000030 001206
722 005262 013737 002030 001122
723 005270 012777 010000 174564
724 005276 005037 001124
725 005302 012777 177776 174520
726 005310 000005
727 005312 017737 174512 001126
728 005320 001401
729 005322 104032
```

```
::*****
:*TEST 30 TEST THAT RESET CLEARS THE D.P.C.
::*****
```

```
TST30: SCOPE
MOV      #100,$TIMES      ;;DO 100 ITERATIONS
MOV      #30,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
MOV      DPC,$BDADR      ;SET UP REG ADRS 00
MOV      #BIT12,@STKPT   ;SET MAINT SWITCH #1
CLR      $GDDAT          ;LOAD EXPECTED
MOV      #-2,@DPC        ;LOAD REGISTER
RESET    ;INITILIZE
MOV      @DPC,$BDDAT     ;READ D.P.C. REG.
BEQ      TST31           ;;BR IF ALL BITS CLEARED
ERROR    32             ;RESET FAILED TO CLEAR THE D.P.C.
```

```
.SBTTL
.SBTTL THESE TESTS USE MAINT. SWITCH 2
.SBTTL
```

```
730
731
(3)
(3)
(2) 005324 000004
(2) 005326 012737 000031 001206
736 005334 013737 002032 001122
737 005342 012777 020000 174512
738 005350 012737 174000 031076
739 005356 013777 002000 174444
740 005364 012737 074000 001124
741 005372 017737 174434 001126
742 005400 042737 103777 001126
743 005406 023737 001124 001126
744 005414 001401
745 005416 104003
```

```
::*****
:*TEST 31 LOAD #17 INTO THE 'MODE' REGISTERS
::*****
```

```
TST31: SCOPE
MOV      #31,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
MOV      SREG0,$BDADR    ;SET UP REG ADRS 02
MOV      #BIT13,@STKPT   ;SET MAINT SWITCH #2
MOV      #174000,BUFFER  ;LOAD MODE REGISTER=17
MOV      DBUF,@DPC       ;LOAD DISPLAY PC.
MOV      #74000,$GDDAT   ;LOAD EXPECTED
MOV      @SREG0,$BDDAT   ;READ DISPLAY STATUS REGISTER
BIC      #103777,$BDDAT  ;MASK TO BITS 14-11
CMP      $GDDAT,$BDDAT   ;COMPARE EXPCT TO RCVD
BEQ      TST32           ;;BR IF EQUAL
ERROR    3              ;MODE BITS (14-11) FAILED TO SET
```

```
746
747
748
(3)
(3)
(2) 005420 000004
(2) 005422 012737 000032 001206
```

```
::*****
:*TEST 32 LOAD #10 INTO THE 'MODE' REGISTERS
::*****
```

```
TST32: SCOPE
MOV      #32,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
```

749 005430 013737 002032 001122
750 005436 012777 020000 174416
751 005444 012737 140000 031076
752 005452 013777 002000 174350
753 005460 012737 040000 001124
754 005466 017737 174340 001126
755 005474 042737 103777 001126
756 005502 023737 001124 001126
757 005510 001401
758 005512 104003

MOV SREG0,\$BDADR ;SET UP REG ADRS 02
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
MOV #140000,BUFFER ;LOAD MODE REGISTER=10
MOV DBUF,@DPC ;LOAD DISPLAY P.C.
MOV #40000,\$GDDAT ;LOAD EXPECTED
MOV @SREG0,\$BDDAT ;READ DISPLAY STATUS REGISTER
BIC #103777,\$BDDAT ;MASK TO BITS 14-11
CMP \$GDDAT,\$BDDAT ;COMPARE EXPCT TO RCVD
BEQ TST33 ;:BR IF EQUAL
ERROR 3 ;MODE BIT 14 FAILED TO SET

759
760
(3)

::*****
:*TEST 33 LOAD #14 INTO THE 'MODE' REGISTERS
:*****

(3)
(2) 005514 000004
(2) 005516 012737 000033 001206
761 005524 013737 002032 001122
762 005532 012777 020000 174322
763 005540 012737 160000 031076
764 005546 013777 002000 174254
765 005554 012737 060000 001124
766 005562 017737 174244 001126
767 005570 042737 103777 001126
768 005576 023737 001124 001126
769 005604 001401
770 005606 104003

TST33: SCOPE
MOV #33,\$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
MOV SREG0,\$BDADR ;SET UP REG ADRS 02
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
MOV #160000,BUFFER ;LOAD MODE REGISTER=14
MOV DBUF,@DPC ;LOAD DISPLAY P.C.
MOV #60000,\$GDDAT ;LOAD EXPECTED
MOV @SREG0,\$BDDAT ;READ DISPLAY STATUS REGISTER
BIC #103777,\$BDDAT ;MASK TO BITS 14-11
CMP \$GDDAT,\$BDDAT ;COMPARE EXPCT TO RCVD
BEQ TST34 ;:BR IF EQUAL
ERROR 3 ;MODE BIT 13 FAILED TO SET

771
772
(3)

::*****
:*TEST 34 LOAD #16 INTO THE 'MODE' REGISTERS
:*****

(3)
(2) 005610 000004
(2) 005612 012737 000034 001206
773 005620 013737 002032 001122
774 005626 012777 020000 174226
775 005634 012737 170000 031076
776 005642 013777 002000 174160
777 005650 012737 070000 001124
778 005656 017737 174150 001126
779 005664 042737 103777 001126
780 005672 023737 001124 001126
781 005700 001401
782 005702 104003

TST34: SCOPE
MOV #34,\$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
MOV SREG0,\$BDADR ;SET UP REG ADRS 02
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
MOV #170000,BUFFER ;LOAD MODE REGISTER=16
MOV DBUF,@DPC ;LOAD DISPLAY P.C.
MOV #70000,\$GDDAT ;LOAD EXPECTED
MOV @SREG0,\$BDDAT ;READ DISPLAY STATUS REGISTER
BIC #103777,\$BDDAT ;MASK TO BITS 14-11
CMP \$GDDAT,\$BDDAT ;COMPARE EXPCT TO RCVD
BEQ TST35 ;:BR IF EQUAL
ERROR 3 ;MODE BIT 12 FAILED TO SET

783
(3)

::*****
:*TEST 35 SIMPLE INCREMENT D.P.C. TEST
:*****

(3)
(2) 005704 000004
(2) 005706 012737 000035 001206
784 005714 013737 002030 001122
785 005722 012777 020000 174132
786 005730 012700 031076
787 005734 012720 164000
788 005740 012720 164000
789 005744 012720 164000
790 005750 012720 164000
791 005754 013777 002000 174046
792 005762 012737 031100 001124

TST35: SCOPE
MOV #35,\$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
MOV DPC,\$BDADR ;SET UP REG ADRS 00
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
MOV #BUFFER,RO ;LOAD BUFFER POINTER
MOV #164000,(RO)+ ;LOAD DISPLAY NOP
MOV #164000,(RO)+
MOV #164000,(RO)+
MOV #164000,(RO)+
MOV DBUF,@DPC ;START THE DISPLAY
MOV #BUFFER+2,\$GDDAT ;LOAD EXPECTED

```
793 005770 017737 174034 001126      MOV    @DPC,$BDDAT      ;READ THE DISPLAY P.C.
794 005776 023737 001124 001126      CMP    $GDDAT,$BDDAT   ;TEST FOR INCREMENT
795 006004 001402                BEQ    1$               ;
796 006006 104004                ERROR  4               ;DISPLAY P.C. FAILED TO INCREMENT
797 006010 000451                BR     TST36           ;
798 006012 005277 174012          1$:  INC    @DPC          ;STEP THE DISPLAY P.C.
799 006016 062737 000002 001124      ADD    #2,$GDDAT       ;UPDATE EXPECTED
800 006024 017737 174000 001126      MOV    @DPC,$BDDAT     ;READ THE DISPLAY P.C.
801 006032 023737 001124 001126      CMP    $GDDAT,$BDDAT   ;TEST FOR INCREMENT
802 006040 001402                BEQ    2$               ;
803 006042 104004                ERROR  4               ;DISPLAY P.C. FAILED TO INCREMENT
804 006044 000433                BR     TST36           ;
805 006046                2$:  INC    @DPC          ;SINGLE STEP THE DISPLAY
(1) 006046 005277 173756          ADD    #2,$GDDAT       ;UPDATE EXPECTED
806 006052 062737 000002 001124      MOV    @DPC,$BDDAT     ;READ THE DISPLAY P.C.
807 006060 017737 173744 001126      CMP    $GDDAT,$BDDAT   ;TEST FOR INCREMENT
808 006066 023737 001124 001126      BEQ    3$               ;
809 006074 001402                ERROR  4               ;DISPLAY P.C. FAILED TO INCREMENT
810 006076 104004                BR     TST36           ;
811 006100 000415                3$:  INC    @DPC          ;SINGLE STEP THE DISPLAY
812 006102                ADD    #2,$GDDAT       ;UPDATE EXPECTED
(1) 006102 005277 173722          MOV    @DPC,$BDDAT     ;READ THE DISPLAY P.C.
813 006106 062737 000002 001124      CMP    $GDDAT,$BDDAT   ;TEST FOR INCREMENT
814 006114 017737 173710 001126      BEQ    TST36           ;:BR IF EQUAL
815 006122 023737 001124 001126      ERROR  4               ;DISPLAY P.C. FAILED TO INCREMENT
816 006130 001401                BR     TST36           ;
817 006132 104004                4:  INC    @DPC          ;SINGLE STEP THE DISPLAY
818                ADD    #2,$GDDAT       ;UPDATE EXPECTED
819                MOV    @DPC,$BDDAT     ;READ THE DISPLAY P.C.
(3)                CMP    $GDDAT,$BDDAT   ;TEST FOR INCREMENT
(3)                BEQ    TST36           ;:BR IF EQUAL
(2) 006134 000004                ERROR  4               ;DISPLAY P.C. FAILED TO INCREMENT
(1) 006136 012737 000040 001166      *****
(2) 006144 012737 000036 001206      ;*TEST 36      EXPANDED D.P.C. INCREMENT TEST
820 006152 013737 002030 001122      ;*****
821 006160 012777 020000 173674      TST36:  SCOPE
822 006166 013702 002000                MOV    #40,$TIMES      ;;DO 40 ITERATIONS
823 006172 012722 164000          1$:  MOV    #36,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
824 006176 023702 002020                MOV    DPC,$BDADR      ;SET UP REG ADRS 00
825 006202 001373                MOV    #BIT13,@STKPT   ;SET MAINT SWITCH #2
826 006204 013777 002000 173616          MOV    DBUF,R2         ;SET UP POINTER
827 006212 012737 031076 001124          MOV    #164000,(R2)+  ;MOVE DNOP INTO THE BUFFER
828 006220 013702 002020                CMP    SIZE,R2         ;FINISHED FILLING THE BUFFER?
829 006224 024242                BNE    1$              ;NO
830 006226 062737 000002 001124          2$:  MOV    DBUF,@DPC     ;YES, START THE DISPLAY
831 006234 017737 173570 001126          MOV    #BUFFER,$GDDAT ;LOAD EXPECTED
832 006242 023737 001124 001126          MOV    SIZE,R2        ;SETUP A COUNT
833 006250 001402                CMP    -(R2),-(R2)     ;DEC BY 2
834 006252 104004                ADD    #2,$GDDAT       ;UPDATE EXPECTED
835 006254 000406                MOV    @DPC,$BDDAT     ;READ DISPLAY P.C.
836                CMP    $GDDAT,$BDDAT   ;DID IT INCREMENT BY 2?
837                BEQ    3$              ;YES
838                ERROR  4               ;DISPLAY PC FAILED TO INCREMENT
839                BR     TST37           ;
840                3$:  CMP    R2,$GDDAT     ;FINISHED THE BUFFER
841                BEQ    TST37           ;:BR TO NEXT TEST
                INC    @DPC          ;SINGLE STEP THE DISPLAY
                BR     2$              ;TRY AGAIN
```

842
(3)
(3)
(2) 006272 000004
(2) 006274 012737 000037 001206
843 006302 012777 020000 173552
844 006310 012737 006324 001110
845 006316 012737 002000 001124
846 006324 012737 150000 031076
847 006332 053737 001124 031076
848 006340 012777 031076 173462
849 006346 013737 002054 001122
850 006354 017737 173474 001126
851 006362 042737 174000 001126
852 006370 023737 001124 001126
853 006376 001401
854 006400 104023
855 006402 006237 001124
856 006406 103346
857
858
(3)
(3)
(2) 006410 000004
(2) 006412 012737 000040 001206
859 006420 012777 020000 173434
860 006426 012737 006440 001110
861 006434 012700 175777
862 006440 010037 001124
863 006444 042737 174000 001124
864 006452 012737 150000 031076
865 006460 053737 001124 031076
866 006466 012777 031076 173334
867 006474 013737 002054 001122
868 006502 017737 173346 001126
869 006510 042737 170000 001126
870 006516 023737 001124 001126
871 006524 001401
872 006526 104023
873 006530 006200
874 006532 103742
875
876
(3)
(3)
(2) 006534 000004
(1) 006536 012737 000040 001166
(2) 006544 012737 000041 001206
877 006552 012777 020000 173302
878 006560 005037 001124
879 006564 012737 153777 031076
880 006572 012777 031076 173230
881 006600 013737 002054 001122
882 006606 000005
883 006610 017737 173240 001126
884 006616 001401

```
*****  
*TEST 37 TEST THAT THE NAME INSTR CAN FLOAT A ONE THRU DPU NAME REG  
*****  
TST37: SCOPE  
MOV #37,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX  
MOV #BIT13,@STKPT ;;SET MAINT SWITCH #2  
MOV #1$,$LPERR ;SET UP SCOPE LOOP ADRS  
MOV #2000,$GDDAT ;SET UP MSB  
1$: MOV #150000,BUFFER ;SET UP INSTR LOC  
BIS $GDDAT,BUFFER ;SET NAME BIT AT INSTR LOC  
MOV #BUFFER,@DPC ;START  
MOV DNAME,$BDADR ;SET UP REG ADRS 24  
MOV @DNAME,$BDDAT ;READ NAME REG  
BIC #174000,$BDDAT ;ONLY INTERESTED IN THE NAME BITS  
CMP $GDDAT,$BDDAT ;DID IT GET LOADED PROPERLY?  
BEQ 2$ ;BR IF OK  
ERROR 23 ;DPU NAME INSTR ER  
2$: ASR $GDDAT ;SHIFT 'ONE' BIT RIGHT  
BCC 1$ ;BR IF MORE BIT POSITIONS
```

```
*****  
*TEST 40 TEST THAT THE NAME INSTR CAN FLOAT A ZERO THRU DPU NAME REG  
*****  
TST40: SCOPE  
MOV #40,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX  
MOV #BIT13,@STKPT ;;SET MAINT SWITCH #2  
MOV #1$,$LPERR ;SET UP SCOPE LOOP ADRS  
MOV #175777,R0 ;SET MSB TO A ZERO  
1$: MOV R0,$GDDAT ;PUT PATTERN IN COMPARE LOC  
BIC #174000,$GDDAT ;GET RID OF EXTRA BITS  
MOV #150000,BUFFER ;SET UP INSTR  
BIS $GDDAT,BUFFER ;SET UP PATTERN AT INSTR LOC  
MOV #BUFFER,@DPC ;START  
MOV DNAME,$BDADR ;SET UP REG ADRS 24  
MOV @DNAME,$BDDAT ;READ NAME REG  
BIC #170000,$BDDAT ;ONLY INTERESTED IN THE NAME BITS  
CMP $GDDAT,$BDDAT ;DID IT GET LOADED PROPERLY?  
BEQ 2$ ;BR IF OK  
ERROR 23 ;DPU NAME INSTR ER  
2$: ASR R0 ;SHIFT ZERO BIT RIGHT  
BCS 1$ ;BR IF MORE BIT POSITIONS
```

```
*****  
*TEST 41 TEST THAT RESET WILL CLEAR DPU NAME  
*****  
TST41: SCOPE  
MOV #40,$TIMES ;;DO 40 ITERATIONS  
MOV #41,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX  
MOV #BIT13,@STKPT ;SET MAINT SWITCH 2  
CLR $GDDAT ;EXPECT ALL ZEROS  
MOV #153777,BUFFER ;SET UP NAME INSTR  
MOV #BUFFER,@DPC ;START  
MOV DNAME,$BDADR ;SET UP REG ADRS 24  
RESET ;CLR DPU NAME REG  
MOV @DNAME,$BDDAT ;READ IT  
BEQ TST42 ;;NEXT TEST IF ALL CLEARED
```

885 006620 104023 ERROR 23 ;RESET FAILED TO CLEAR DPU NAME

886
887
(3) :*****
: *TEST 42 TEST THAT THE SEARCH CODE BITS CAN BE READ AT REG 24
(3) :*****

(2) 006622 000004 TST42: SCOPE
(2) 006624 012737 000042 001206 MOV #42,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
888 006632 012737 006660 001110 MOV #1\$, \$LPERR ;;SET UP SCOPE LOOP ADRS
889 006640 013737 002054 001122 MOV DNAME,\$BDADR ;;SET UP REG 24 ADRS
890 006646 012737 030000 001124 MOV #30000,\$GDDAT ;;EXPECT BOTH BITS INITIALLY
891 006654 012700 070000 MOV #70000,\$RO ;;SET UP RO WITH ENABLE
892 006660 010077 173164 1\$: MOV RO,@ANAME ;;SEND SEARCH CODE
893 006664 017737 173164 001126 MOV @DNAME,\$BDDAT ;;READ IT BACK
894 006672 042737 147777 001126 BIC #147777,\$BDDAT ;;SAVE ONLY SEARCH CODE
895 006700 023737 001124 001126 CMP \$GDDAT,\$BDDAT ;;IS IT CORRECT?
896 006706 001401 BEQ 2\$;;BR IF OK
897 006710 104024 ERROR 24 ;;SEARCH CODE READ/WRITE ER
898 006712 162700 010000 2\$: SUB #10000,\$RO ;;SET UP NEXT SEARCH CODE
899 006716 162737 010000 001124 SUB #10000,\$GDDAT ;;SET UP NEXT SEARCH CODE
900 006724 100355 BPL 1\$;;BR IF MORE CODES TO TEST

901
902
(3) :*****
: *TEST 43 TEST THAT THE SEARCH CODE DOES NOT CHANGE WHEN ENA=0
(3) :*****

(2) 006726 000004 TST43: SCOPE
(2) 006730 012737 000043 001206 MOV #43,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
903 006736 013737 002054 001122 MOV DNAME,\$BDADR ;;SET UP REG 24 ADRS
904 006744 012737 030000 001124 MOV #30000,\$GDDAT ;;EXPECT BOTH BITS
905 006752 012777 070000 173070 MOV #70000,@ANAME ;;LOAD UP SEARCH CODE
906 006760 005077 173064 CLR @ANAME ;;TRY TO CLR SEARCH CODE WITH ENA=0
907 006764 017737 173064 001126 MOV @DNAME,\$BDDAT ;;READ IT BACK
908 006772 042737 147777 001126 BIC #147777,\$BDDAT ;;SAVE ONLY SEARCH CODE
909 007000 023737 001124 001126 CMP \$GDDAT,\$BDDAT ;;IS IT CORRECT?
910 007006 001401 BEQ TST44 ;;ADVANCE TO NEXT TEST IF OK
911 007010 104024 ERROR 24 ;;SEARCH CODE CHANGE ENABLE ER

912
913
(3) :*****
: *TEST 44 TEST THAT RESET WILL CLEAR SEARCH CODE
(3) :*****

(2) 007012 000004 TST44: SCOPE
(1) 007014 012737 000040 001166 MOV #40,\$TIMES ;;DO 40 ITERATIONS
(2) 007022 012737 000044 001206 MOV #44,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
914 007030 013737 002054 001122 MOV DNAME,\$BDADR ;;SET UP REG 24 ADRS
915 007036 005037 001124 CLR \$GDDAT ;;EXPECT RESET STATE
916 007042 012777 070000 173000 MOV #70000,@ANAME ;;SET BOTH SEARCH CODE BITS
917 007050 000005 RESET ;;CLR THEM
918 007052 017737 172776 001126 MOV @DNAME,\$BDDAT ;;READ REG
919 007060 001401 BEQ TST45 ;;NEXT TEST IF ALL CLEARED
920 007062 104024 ERROR 24 ;;RESET FAILED TO CLR SEARCH CODE

921
922
(3) :*****
: *TEST 45 TEST THAT THE NAME MATCH FLAG WILL SET ON 4 BIT COMPARES
(3) :*****

(2) 007064 000004 TST45: SCOPE
(1) 007066 012737 000010 001166 MOV #10,\$TIMES ;;DO 10 ITERATIONS
(2) 007074 012737 000045 001206 MOV #45,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX

923	007102	012777	020000	172752		MOV	#BIT13,@STKPT		:SET MAINT SWITCH #2
924	007110	012737	000340	177776		MOV	#340,PSW		:SET UP PRIORITY TO HIGHEST LEVEL
925	007116	012737	007152	001110		MOV	#2\$, \$LPERR		:SET UP SCOPE LOOP ADRS
926	007124	012737	033600	001124		MOV	#33600,\$GDDAT		:START WITH MAX VALUE
927	007132	052737	044000	001124	1\$:	BIS	#44000,\$GDDAT		:SET THE SEARCH CODE ENA BIT
928	007140	013777	001124	172702		MOV	\$GDDAT,@ANAME		:SET UP ASSOCIATIVE NAME REG
929	007146	012700	003600			MOV	#3600,R0		:SET UP R0 WITH MAX VALUE
930	007152	042737	144000	001124	2\$:	BIC	#144000,\$GDDAT		:CLR MATCH FLAG & SEARCH ENA BIT
931	007160	012737	150000	031076		MOV	#150000,BUFFER		:SET UP NAME INSTR
932	007166	050037	031076			BIS	R0,BUFFER		:SET UP NAME VALUE
933	007172	012777	031076	172630		MOV	#BUFFER,@DPC		:START
934	007200	013737	002054	001122		MOV	DNAME,\$BDADR		:SET UP REG ADRS 24
935	007206	017737	172642	001126		MOV	@DNAME,\$BDDAT		:READ NAME REG
936	007214	100410				BMI	3\$:BR IF A MATCH OCCURRED
937	007216	023737	001124	001126		CMP	\$GDDAT,\$BDDAT		:SHOULD A MATCH HAVE OCCURRED?
938	007224	001017				BNE	4\$:BR IF NOT
939	007226	052737	100000	001124		BIS	#100000,\$GDDAT		:INDICATE IT SHOULD BE SET
940	007234	104024				ERROR	24		:NAME MATCH FLAG FAILED TO SET
941	007236	042737	100000	001126	3\$:	BIC	#100000,\$BDDAT		:CLR OUT MATCH FLAG BIT
942	007244	023737	001124	001126		CMP	\$GDDAT,\$BDDAT		:MAKE SURE THERE IS A MATCH
943	007252	001404				BEQ	4\$:BR IF OK
944	007254	052737	100000	001126		BIS	#100000,\$BDDAT		:RESTORE THE MATCH FLAG BIT
945	007262	104024				ERROR	24		:NAME MATCH FLAG SET IN ER
946	007264	162700	000200		4\$:	SUB	#200,R0		:ADVANCE NAME VALUE
947	007270	100330				BPL	2\$:BR IF MORE VALUES TO TEST
948	007272	162737	000200	001124		SUB	#200,\$GDDAT		:ADVANCE VALUE FOR ASSO. NAME REG
949	007300	022737	027600	001124		CMP	#27600,\$GDDAT		:HAVE ALL VALUES BEEN TESTED?
950	007306	001311				BNE	1\$:BR IF NOT

951
952
(3)
(3)
(2) 007310 000004
(1) 007312 012737 000004 001166
(2) 007320 012737 000046 001206
953 007326 012777 020000 172526
954 007334 012737 000340 177776
955 007342 012737 007376 001110
956 007350 012737 023770 001124
957 007356 052737 044000 001124 1\$:
958 007364 013777 001124 172456
959 007372 012700 003770
960 007376 042737 144000 001124 2\$:
961 007404 012737 150000 031076
962 007412 050037 031076
963 007416 012777 031076 172404
964 007424 013737 002054 001122
965 007432 017737 172416 001126
966 007440 100410
967 007442 023737 001124 001126
968 007450 001017
969 007452 052737 100000 001124
970 007460 104024
971 007462 042737 100000 001126 3\$:
972 007470 023737 001124 001126
973 007476 001404

```
*****  
:*TEST 46 TEST THAT THE NAME MATCH FLAG WILL SET ON 8 BIT COMPARES  
*****  
TST46: SCOPE  
MOV #4,$TIMES ;;DO 4 ITERATIONS  
MOV #46,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX  
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2  
MOV #340,PSW ;SET PRIORITY TO HIGHEST LEVEL  
MOV #2$, $LPERR ;SET UP SCOPE LOOP ADRS  
MOV #23770,$GDDAT ;START WITH MAX VALUE  
1$: BIS #44000,$GDDAT ;SET SEARCH ENA BIT  
MOV $GDDAT,@ANAME ;SET UP ASSOCIATIVE NAME REG  
MOV #3770,R0 ;SET UP R0 WITH MAX VALUE  
2$: BIC #144000,$GDDAT ;CLR MATCH FLAG & SEARCH ENA  
MOV #150000,BUFFER ;SET UP NAME INSTR  
BIS R0,BUFFER ;SET UP NAME VALUE  
MOV #BUFFER,@DPC ;START  
MOV DNAME,$BDADR ;SET UP REG ADRS 24  
MOV @DNAME,$BDDAT ;READ NAME REG  
BMI 3$ ;BR IF A MATCH OCCURED  
CMP $GDDAT,$BDDAT ;SHOULD A MATCH HAVE OCCURED?  
BNE 4$ ;BR IF NOT  
BIS #100000,$GDDAT ;INDICATE IT SHOULD BE SET  
ERROR 24 ;NAME MATCH FLAG FAILED TO SET  
3$: BIC #100000,$BDDAT ;CLR OUT MATCH FLAG BIT  
CMP $GDDAT,$BDDAT ;MAKE SURE THERE IS A MATCH  
BEQ 4$ ;BR IF OK
```

```
974 007500 052737 100000 001126 BIS #100000,$BDDAT ;RESTORE THE MATCH FLAG BIT
975 007506 104024 ERROR 24 ;NAME MATCH FLAG SET IN ER
976 007510 162700 000010 4$: SUB #10,R0 ;ADVANCE NAME VALUE
977 007514 100330 BPL 2$ ;BR IF MORE VALUES TO TEST
978 007516 162737 000010 001124 SUB #10,$GDDAT ;ADVANCE VALUE FOR ASSO. NAME REG
979 007524 022737 017770 001124 CMP #17770,$GDDAT ;HAVE ALL VALUES BEEN TESTED?
980 007532 001311 BNE 1$ ;BR IF NOT
981
982
(3)
(3)
(2) 007534 000004 TST47: SCOPE
(1) 007536 012737 000002 001166 MOV #2,$TIMES ;:DO 2 ITERATIONS
(2) 007544 012737 000047 001206 MOV #47,$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
983 007552 012777 020000 172302 MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
984 007560 012737 007614 001110 MOV #2$,$LPERR ;SET UP SCOPE LOOP ADRS
985 007566 012737 010077 001124 MOV #10077,$GDDAT ;START WITH 77 - WORK ONLY LOWER ORDER BITS
986 007574 052737 044000 001124 1$: BIS #44000,$GDDAT ;SET SEARCH ENA BIT
987 007602 013777 001124 172240 MOV $GDDAT,@ANAME ;SET UP ASSOCIATIVE NAME REG
988 007610 012700 000077 MOV #77,R0 ;SET UP R0 WITH 77
989 007614 042737 144000 001124 2$: BIC #144000,$GDDAT ;CLR MATCH FLAG & SEARCH ENA
990 007622 012737 150000 031076 MOV #150000,BUFFER ;SET UP NAME INSTR
991 007630 050037 031076 BIS R0,BUFFER ;SET UP NAME VALUE
992 007634 012777 031076 172166 MOV #BUFFER,@DPC ;START
993 007642 013737 002054 001122 MOV DNAME,$BDADR ;SET UP REG ADRS 24
994 007650 000240 NOP ;ALLOW TIME FOR NPR
995 007652 017737 172176 001126 MOV @DNAME,$BDDAT ;READ NAME REG
996 007660 100410 BMI 3$ ;BR IF A MATCH OCCURED
997 007662 023737 001124 001126 CMP $GDDAT,$BDDAT ;SHOULD A MATCH HAVE OCCURED?
998 007670 001017 BNE 4$ ;BR IF NOT
999 007672 052737 100000 001124 BIS #100000,$GDDAT ;INDICATE IT SHOULD BE SET
1000 007700 104024 ERROR 24 ;NAME MATCH FLAG FAILED TO SET
1001 007702 042737 100000 001126 3$: BIC #100000,$BDDAT ;CLR OUT MATCH FLAG BIT
1002 007710 023737 001124 001126 CMP $GDDAT,$BDDAT ;MAKE SURE THERE IS A MATCH
1003 007716 001404 BEQ 4$ ;BR IF OK
1004 007720 052737 100000 001126 BIS #100000,$BDDAT ;RESTORE THE MATCH FLAG BIT
1005 007726 104024 ERROR 24 ;NAME MATCH FLAG SET IN ER
1006 007730 005300 4$: DEC R0 ;ADVANCE NAME VALUE
1007 007732 100330 BPL 2$ ;BR IF MORE VALUES TO TEST
1008 007734 005337 001124 DEC $GDDAT ;ADVANCE VALUE FOR ASSO. NAME REG
1009 007740 022737 007777 001124 CMP #7777,$GDDAT ;HAVE ALL VALUES BEEN TESTED?
1010 007746 001312 BNE 1$ ;BR IF NOT
1011
1012
(3)
(3)
(2) 007750 000004 TST50: SCOPE
(1) 007752 012737 000040 001166 MOV #40,$TIMES ;:DO 40 ITERATIONS
(2) 007760 012737 000050 001206 MOV #50,$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
1013 007766 012777 020000 172066 MOV #BIT13,@STKPT ;SET MAINT SWITCH 2
1014 007774 012777 057777 172046 MOV #57777,@ANAME ;SET ANAME FOR 11 BIT COMPARE ON 3777
1015 010002 012737 153777 031076 MOV #153777,BUFFER ;SET UP NAME INSTR
1016 010010 005037 001124 CLR $GDDAT ;EXPECT ALL ZEROS
1017 010014 012777 031076 172006 MOV #BUFFER,@DPC ;START
1018 010022 013737 002054 001122 MOV DNAME,$BDADR ;SET UP REG ADRS 24
1019 010030 000005 RESET ;CLR DPU NAME REG (MATCH)
```

1020 010032 017737 172016 001126
1021 010040 001401
1022 010042 104023
1023
1024
(3)
(3)
(2) 010044 000004
(2) 010046 012737 000051 001206
1025 010054 012777 020000 172000
1026 010062 012737 000340 177776
1027 010070 013737 002054 001122
1028 010076 012777 057777 171744
1029 010104 012777 050000 171736
1030 010112 012737 153777 031076
1031 010120 012777 031076 171702
1032 010126 012737 113777 001124
1033 010134 017737 171714 001126
1034 010142 023737 001124 001126
1035 010150 001401
1036 010152 104024
1037
1038
(3)
(3)
(2) 010154 000004
(1) 010156 012737 000002 001166
(2) 010164 012737 000052 001206
1039 010172 012777 020000 171662
1040 010200 012737 000340 177776
1041 010206 012737 010266 001110
1042 010214 013701 001252
1043 010220 042701 160000
1044 010224 012761 010342 000014
1045 010232 012761 000340 000016
1046 010240 012737 010077 001124
1047 010246 052737 044000 001124 1\$:
1048 010254 013777 001124 171566
1049 010262 012700 000077
1050 010266 042737 044000 001124 2\$:
1051 010274 012737 150000 031076
1052 010302 050037 031076
1053 010306 012777 031076 171514
1054 010314 013737 002054 001122
1055 010322 005037 177776
1056 010326 017737 171522 001126
1057 010334 100010
1058 010336 104047
1059 010340 000416
1060 010342 022626
1061 010344 017737 171504 001126 3\$:
1062 010352 100401
1063 010354 104047
1064 010356 005300 4\$:
1065 010360 100342
1066 010362 005337 001124

```
MOV @DNAME,$BDDAT ;READ IT
BEQ TST51 ;NEXT TEST IF ALL CLEARED
ERROR 23 ;RESET FAILED TO CLEAR NAME MATCH

*****
*TEST 51 TEST THAT THE ASSOCIATIVE NAME REG DOES NOT CHANGE WITH ENA=0
*****
TST51: SCOPE
MOV #51,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
MOV #340,PSW ;DON'T ALLOW INTERRUPT
MOV DNAME,$BDADR ;SET UP REG 24 ADRS
MOV #57777,@ANAME ;SET UP FOR 11 BIT CMP ON VALUE #3777
MOV #50000,@ANAME ;NOW TRY WRITING 0 WITH ENA DISABLED
MOV #153777,BUFFER ;SET UP NAME INSTR WITH VALUE #3777
MOV #BUFFER,@DPC ;START
MOV #113777,$GDDAT ;EXPECT NAME MATCH ON 11 BIT CMP VALUE #3777
MOV @DNAME,$BDDAT ;GET DPU NAME REG CONTENTS
CMP $GDDAT,$BDDAT ;IS THERE A MATCH?
BEQ TST52 ;ADVANCE TO NEXT TEST IF SO
ERROR 24 ;ASSOCIATIVE NAME CHANGE ENABLE ER

*****
*TEST 52 TEST THAT AN INTERRUPT WILL OCCUR ON 11 BIT NAME MATCH COMPARES
*****
TST52: SCOPE
MOV #2,$TIMES ;DO 2 ITERATIONS
MOV #52,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
MOV #340,PSW ;SET PSW TO 7
MOV #2,$LPERR ;SET UP SCOPE LOOP ADRS
MOV $VECT1,R1 ;GET VECTOR ADRS
BIC #160000,R1 ; CLEAR PSW BITS ** B **
MOV #3$,14(R1) ;SET UP NAME INT SER ADRS
MOV #340,16(R1) ;SET UP PSW TO 7 ON INT
MOV #10077,$GDDAT ;START WITH MAX VALUE
1$: BIS #44000,$GDDAT ;SET SEACH CHANGE & NAME CHANGE ENABLES
MOV $GDDAT,@ANAME ;SET UP ASSOCIATIVE NAME REG
MOV #77,R0 ;SET UP R0 WITH MAX VALUE
2$: BIC #44000,$GDDAT ;CLR ENA BITS
MOV #150000,BUFFER ;SET UP NAME INSTR
BIS R0,BUFFER ;SET UP NAME VALUE
MOV #BUFFER,@DPC ;START
MOV DNAME,$BDADR ;SET UP REG ADRS 24
CLR PSW ;ALLOW INT TO OCCUR
MOV @DNAME,$BDDAT ;READ THE NAME REG
BPL 4$ ;BR IF NO MATCH
ERROR 47 ;NAME MATCH FAILED TO INTERRUPT
BR 5$ ;GO LOOK FOR LOOP ON TEST SWITCH
3$: CMP (R6)+,(R6)+ ;FIX STK SINCE NO RTI
MOV @DNAME,$BDDAT ;READ NAME REG
BMI 4$ ;BR IF MATCH FLAG SET
ERROR 47 ;NAME MATCH INTERRUPTED BUT NO FLAG
4$: DEC R0 ;ADVANCE NAME VALUE IN R0
BPL 2$ ;BR IF MORE VALUES TO TEST
DEC $GDDAT ;ADVANCE VALUE FOR ASSO. NAME REG
```

1067 010366 022737 007777 001124
1068 010374 001324
1069 010376 004737 024454
1070
1071
(3)
(3)
(2) 010402 000004
(2) 010404 012737 000053 001206
1072 010412 012777 020000 171442
1073 010420 012777 100005 171352
1074 010426 012737 000001 001124
1075 010434 013777 002000 171366
1076 010442 013737 002032 001122
1077 010450 017737 171356 001126
1078 010456 042737 177774 001126
1079 010464 023737 001124 001126
1080 010472 001401
1081 010474 104005
1082
1083
(3)
(3)
(2) 010476 000004
(2) 010500 012737 000054 001206
1084 010506 012777 020000 171346
1085 010514 012777 100006 171256
1086 010522 012737 000002 001124
1087 010530 013777 002000 171272
1088 010536 013737 002032 001122
1089 010544 017737 171262 001126
1090 010552 042737 177774 001126
1091 010560 023737 001124 001126
1092 010566 001401
1093 010570 104005
1094
1095
1096
(3)
(3)
(2) 010572 000004
(2) 010574 012737 000055 001206
1097 010602 012777 020000 171252
1098 010610 012777 100007 171162
1099 010616 013777 002000 171204
1100 010624 012777 100000 171146
1101 010632 012737 000003 001124
1102 010640 013777 002000 171162
1103 010646 013737 002032 001122
1104 010654 017737 171152 001126
1105 010662 042737 177774 001126
1106 010670 023737 001124 001126
1107 010676 001401
1108 010700 104005
1109
1110

CMP #7777,\$GDDAT ;SEE IF ALL VALUES HAVE BEEN TESTED
BNE 1\$;BR IF NOT
5\$: JSR PC,RSTVEC ;RESTORE NAME MATCH VECTOR WITH HALT

::*****
:*TEST 53 LOAD #1 INTO LINE TYPE REGISTERS
:*****

TST53: SCOPE
MOV #53,\$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;:SET MAINT SWITCH #2
MOV #100005,@DBUF ;:LINE TYPE ENABLE =1 LINE TYPE =1
MOV #1,\$GDDAT ;:LOAD EXPECTED
MOV DBUF,@DPC ;:LOAD DISPLAY P.C.
MOV SREG0,\$BDADR ;:SET UP REG ADRS 02
MOV @SREG0,\$BDDAT ;:READ DISPLAY STATUS REGISTER
BIC #177774,\$BDDAT ;:MASK TO BITS 1-0
CMP \$GDDAT,\$BDDAT ;:COMPARE EXPCT TO RCVD
BEQ TST54 ;:BR IF EQUAL
ERROR 5 ;:LINE BIT 0 FAILED TO SET

::*****
:*TEST 54 LOAD #2 INTO LINE TYPE REGISTERS
:*****

TST54: SCOPE
MOV #54,\$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;:SET MAINT SWITCH #2
MOV #100006,@DBUF ;:LINE TYPE ENABLE =1 LINE TYPE =2
MOV #2,\$GDDAT ;:LOAD EXPECTED
MOV DBUF,@DPC ;:LOAD DISPLAY P.C.
MOV SREG0,\$BDADR ;:SET UP REG ADRS 02
MOV @SREG0,\$BDDAT ;:READ DISPLAY STATUS REGISTER
BIC #177774,\$BDDAT ;:MASK TO BITS 1-0
CMP \$GDDAT,\$BDDAT ;:COMPARE EXPCT TO RCVD
BEQ TST55 ;:BR IF EQUAL
ERROR 5 ;:LINE BIT 1 FAILED TO SET

::*****
:*TEST 55 CHECK LINE TYPE ENABLE GATE
:*****

TST55: SCOPE
MOV #55,\$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;:SET MAINT SWITCH #2
MOV #100007,@DBUF ;:LINE TYPE ENABLE =1 LINE TYPE =3
MOV DBUF,@DPC ;:LOAD DISPLAY P.C.
MOV #100000,@DBUF ;:LOAD EXPECTED
MOV #3,\$GDDAT ;:LOAD EXPECTED
MOV DBUF,@DPC ;:LOAD DISPLAY P.C.
MOV SREG0,\$BDADR ;:SET UP REG ADRS 02
MOV @SREG0,\$BDDAT ;:READ DISPLAY STATUS REGISTER
BIC #177774,\$BDDAT ;:MASK TO BITS 1-0
CMP \$GDDAT,\$BDDAT ;:COMPARE EXPCT TO RCVD
BEQ TST56 ;:BR IF EQUAL
ERROR 5 ;:LINE TYPE ENABLE FAILED TO INHIBIT
;CHANGING OF LINETYPE VALUE

1111
(3)
(3)
(2) 010702 000004
(2) 010704 012737 000056 001206
1112 010712 012777 020000 171142
1113 010720 012777 100020 171052
1114 010726 005037 001124
1115 010732 013777 002000 171070
1116 010740 013737 002032 001122
1117 010746 017737 171060 001126
1118 010754 042737 177767 001126
1119 010762 023737 001124 001126
1120 010770 001401
1121 010772 104027
1122
(3)
(3)
(2) 010774 000004
(2) 010776 012737 000057 001206
1123 011004 012777 020000 171050
1124 011012 012737 100030 031076
1125 011020 012737 000010 001124
1126 011026 013777 002000 170774
1127 011034 013737 002032 001122
1128 011042 017737 170764 001126
1129 011050 042737 177767 001126
1130 011056 023737 001124 001126
1131 011064 001401
1132 011066 104027
1133
1134
1135
(3)
(3)
(2) 011070 000004
(2) 011072 012737 000060 001206
1136 011100 012777 020000 170754
1137 011106 012777 100030 170664
1138 011114 013777 002000 170706
1139 011122 012737 000010 001124
1140 011130 012777 100000 170642
1141 011136 013777 002000 170664
1142 011144 013737 002032 001122
1143 011152 017737 170654 001126
1144 011160 042737 177767 001126
1145 011166 023737 001124 001126
1146 011174 001401
1147 011176 104027
1148
1149
1150
(3)
(3)
(2) 011200 000004
(2) 011202 012737 000061 001206

```
*****  
*TEST 56 LOAD #0 INTO BLINK REGISTERS  
*****  
TST56: SCOPE  
MOV #56,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX  
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2  
MOV #100020,@DBUF ;BLINK ENABLE =1 BLINK =0  
CLR $GDDAT ;LOAD EXPECTED  
MOV DBUF,@DPC ;LOAD DISPLAY P.C.  
MOV SREG0,$BDADR ;SET UP REG ADRS 02  
MOV @SREG0,$BDDAT ;READ DISPLAY STATUS REGISTER  
BIC #177767,$BDDAT ;MASK TO BIT 3  
CMP $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD  
BEQ TST57 ;;BR IF EQUAL  
ERROR 27 ;BLINK BIT FAILED TO RESET  
*****  
*TEST 57 LOAD #1 INTO BLINK REGISTERS  
*****  
TST57: SCOPE  
MOV #57,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX  
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2  
MOV #100030,BUFFER ;BLINK ENABLE =1 BLINK =1  
MOV #10,$GDDAT ;LOAD EXPECTED  
MOV DBUF,@DPC ;LOAD DISPLAY P.C.  
MOV SREG0,$BDADR ;SET UP REG ADRS 02  
MOV @SREG0,$BDDAT ;READ DISPLAY STATUS REGISTER  
BIC #177767,$BDDAT ;MASK TO BIT 3  
CMP $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD  
BEQ TST60 ;;BR IF EQUAL  
ERROR 27 ;BLINK BIT FAILED TO SET  
*****  
*TEST 60 CHECK BLINK ENABLE GATE  
*****  
TST60: SCOPE  
MOV #60,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX  
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2  
MOV #100030,@DBUF ;LOAD BLINK ON  
MOV DBUF,@DPC ;LOAD EXPECTED  
MOV #BIT3,$GDDAT ;LOAD EXPECTED  
MOV #100000,@DBUF ;BLINK ENABLE =0 BLINK =0  
MOV DBUF,@DPC ;LOAD DISPLAY P.C.  
MOV SREG0,$BDADR ;SET UP REG ADRS 02  
MOV @SREG0,$BDDAT ;READ DISPLAY STATUS REGISTER  
BIC #177767,$BDDAT ;MASK TO BIT 3  
CMP $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD  
BEQ TST61 ;;BR IF EQUAL  
ERROR 27 ;BLINK ENABLE FAILED TO INHIBIT  
;CHANGING OF THE BLINK BIT  
*****  
*TEST 61 TEST FOR FALSE L.P. STATUS FLAG (L.P. - NOT ENABLED)  
*****  
TST61: SCOPE  
MOV #61,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
```

```
1151 011210 012777 020000 170644      MOV      #BIT13,@STKPT      ;SET MAINT SWITCH #2
1152 011216 012737 100100 001124      MOV      #100100,$GDDAT    ;LP ENABLE =1 LP=0
1153 011224 013777 001124 170546      MOV      $GDDAT,@DBUF      ;LOAD BUFFER
1154 011232 013777 002000 170570      MOV      DBUF,@DPC         ;LOAD DISPLAY P.C.
1155 011240 013737 002032 001122      MOV      SREG0,$BDADR     ;SET UP REG ADRS 02
1156 011246 017737 170560 001126      MOV      @SREG0,$BDDAT    ;READ STATUS
1157 011254 032737 000200 001126      BIT      #200,$BDDAT      ;
1158 011262 001401                BEQ      TST62             ;;BR IF EQUAL
1159 011264 104034                ERROR    34              ;LIGHT PEN FLAG SET IN ERROR
```

```
::*****
:*TEST 62      TEST FOR FALSE L.P. STATUS FLAG (L.P. - ENABLED)
::*****
```

```
TST62: SCOPE
(2) 011266 000004
(2) 011270 012737 000062 001206      MOV      #62,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
1162 011276 012777 020000 170556      MOV      #BIT13,@STKPT    ;SET MAINT SWITCH #2
1163 011304 012737 100140 001124      MOV      #100140,$GDDAT  ;LP ENABLE =1 LP=1
1164 011312 013777 001124 170460      MOV      $GDDAT,@DBUF    ;LOAD BUFFER
1165 011320 013777 002000 170502      MOV      DBUF,@DPC       ;LOAD DISPLAY P.C.
1166 011326 013737 002032 001122      MOV      SREG0,$BDADR    ;SET UP REG ADRS 02
1167 011334 017737 170472 001126      MOV      @SREG0,$BDDAT  ;READ STATUS
1168 011342 032737 000200 001126      BIT      #200,$BDDAT    ;
1169 011350 001401                BEQ      TST63             ;;BR IF EQUAL
1170 011352 104034                ERROR    34              ;LIGHT PEN FLAG SET IN ERROR
```

```
::*****
:*TEST 63      LOAD #4 INTO INT LEVEL REGISTER
::*****
```

```
TST63: SCOPE
(2) 011354 000004
(2) 011356 012737 000063 001206      MOV      #63,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
1173 011364 012777 020000 170470      MOV      #BIT13,@STKPT    ;SET MAINT SWITCH #2
1174 011372 012777 103000 170400      MOV      #103000,@DBUF   ;INTENSITY LEVEL ENABLE =1 LEVEL =4
1175 011400 012737 002000 001124      MOV      #2000,$GDDAT    ;LOAD EXPECTED
1176 011406 013777 002000 170414      MOV      DBUF,@DPC       ;LOAD DISPLAY P.C.
1177 011414 013737 002032 001122      MOV      SREG0,$BDADR    ;SET UP REG ADRS 02
1178 011422 017737 170404 001126      MOV      @SREG0,$BDDAT  ;READ DISPLAY STATUS REGISTER
1179 011430 042737 174377 001126      BIC      #174377,$BDDAT  ;MASK TO BITS 8-10
1180 011436 023737 001124 001126      CMP      $GDDAT,$BDDAT   ;COMPARE EXPCT TO RCVD
1181 011444 001401                BEQ      TST64             ;;BR IF EQUAL
1182 011446 104006                ERROR    6              ;INTENSITY LEVEL BIT 10 FAILED
```

```
::*****
:*TEST 64      LOAD #2 INTO INT LEVEL REGISTER
::*****
```

```
TST64: SCOPE
(2) 011450 000004
(2) 011452 012737 000064 001206      MOV      #64,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
1185 011460 012777 020000 170374      MOV      #BIT13,@STKPT    ;SET MAINT SWITCH #2
1186 011466 012777 102400 170304      MOV      #102400,@DBUF   ;INTENSITY LEVEL ENABLE =1 LEVEL =2
1187 011474 012737 001000 001124      MOV      #1000,$GDDAT    ;LOAD EXPECTED
1188 011502 013777 002000 170320      MOV      DBUF,@DPC       ;LOAD DISPLAY P.C.
1189 011510 013737 002032 001122      MOV      SREG0,$BDADR    ;SET UP REG ADRS 02
1190 011516 017737 170310 001126      MOV      @SREG0,$BDDAT  ;READ DISPLAY STATUS REGISTER
1191 011524 042737 174377 001126      BIC      #174377,$BDDAT  ;MASK TO BITS 8-10
1192 011532 023737 001124 001126      CMP      $GDDAT,$BDDAT   ;COMPARE EXPCT TO RCVD
1193 011540 001401                BEQ      TST65             ;;BR IF EQUAL
1194 011542 104006                ERROR    6              ;INTENSITY LEVEL BIT 9 FAILED
```

C 4

1195
1196
(3)
(3)
(2) 011544 000004
(2) 011546 012737 000065 001206
1197 011554 012777 020000 170300
1198 011562 012777 102200 170210
1199 011570 012737 000400 001124
1200 011576 013777 002000 170224
1201 011604 013737 002032 001122
1202 011612 017737 170214 001126
1203 011620 042737 174377 001126
1204 011626 023737 001124 001126
1205 011634 001401
1206 011636 104006
1207
1208
1209
(3)
(3)
(2) 011640 000004
(2) 011642 012737 000066 001206
1210 011650 012777 020000 170204
1211 011656 012777 103200 170114
1212 011664 012777 100000 170110
1213 011672 013777 002000 170130
1214 011700 005277 170124
1215 011704 012737 002400 001124
1216 011712 013737 002032 001122
1217 011720 017737 170106 001126
1218 011726 042737 174377 001126
1219 011734 023737 001124 001126
1220 011742 001401
1221 011744 104006
1222
1223
1224
(3)
(3)
(2) 011746 000004
(2) 011750 012737 000067 001206
1225 011756 012777 020000 170076
1226 011764 012777 170040 170006
1227 011772 005037 001124
1228 011776 013777 002000 170024
1229 012004 013737 002032 001122
1230 012012 017737 170014 001126
1231 012020 042737 177757 001126
1232 012026 023737 001124 001126
1233 012034 001401
1234 012036 104030
1235
1236
1237
(3)

*TEST 65 LOAD #1 INTO INT LEVEL REGISTER

TST65: SCOPE
MOV #65,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
MOV #102200,@DBUF ;INTENSITY LEVEL ENABLE =1 LEVEL =1
MOV #400,\$GDDAT ;LOAD EXPECTED
MOV DBUF,@DPC ;LOAD DISPLAY P.C.
MOV SREG0,\$BDADR ;SET UP REG ADRS 02
MOV @SREG0,\$BDDAT ;READ DISPLAY STATUS REGISTER
BIC #174377,\$BDDAT ;MASK TO BITS 8-10
CMP \$GDDAT,\$BDDAT ;COMPARE EXPCT TO RCVD
BEQ TST66 ;;BR IF EQUAL
ERROR 6 ;INTENSITY LEVEL BIT 8 FAILED

*TEST 66 CHECK INTENSITY LEVEL ENABLE GATE

TST66: SCOPE
MOV #66,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
MOV #103200,@DBUF ;LOAD LEVEL
MOV #100000,@DBUF1 ;INTENSITY LEVEL ENABLE =0 LEVEL =0
MOV DBUF,@DPC ;LOAD DISPLAY P.C.
INC @DPC ;SINGLE STEP THE DISPLAY
MOV #2400,\$GDDAT ;LOAD EXPECTED
MOV SREG0,\$BDADR ;SET UP REG ADRS 02
MOV @SREG0,\$BDDAT ;READ DISPLAY STATUS REGISTER
BIC #174377,\$BDDAT ;MASK TO BITS 8-10
CMP \$GDDAT,\$BDDAT ;COMPARE EXPCT TO RCVD
BEQ TST67 ;;BR IF EQUAL
ERROR 6 ;INTENSITY LEVEL ENABLE FAILED TO INHIBIT
;INTENSITY LEVEL CHANGE

*TEST 67 LOAD #0 INTO ITALIC REGISTER

TST67: SCOPE
MOV #67,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
MOV #170040,@DBUF ;ITALICS ENABLE=1 ITALICS=0
CLR \$GDDAT ;LOAD EXPECTED
MOV DBUF,@DPC ;LOAD DISPLAY P.C.
MOV SREG0,\$BDADR ;SET UP REG ADRS 02
MOV @SREG0,\$BDDAT ;READ DISPLAY STATUS REGISTER
BIC #177757,\$BDDAT ;MASK TO BIT 4
CMP \$GDDAT,\$BDDAT ;COMPARE EXPCT TO RCVD
BEQ TST70 ;;BR IF EQUAL
ERROR 30 ;ITALICS BIT FAILED TO RESET

*TEST 70 LOAD #1 INTO ITALIC REGISTER

(3)
(2) 012040 000004
(2) 012042 012737 000070 001206
1238 012050 012777 020000 170004
1239 012056 012777 170060 167714
1240 012064 012737 000020 001124
1241 012072 013777 002000 167730
1242 012100 013737 002032 001122
1243 012106 017737 167720 001126
1244 012114 042737 177757 001126
1245 012122 023737 001124 001126
1246 012130 001401
1247 012132 104030

```
*****  
TST70: SCOPE  
MOV #70,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX  
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2  
MOV #170060,@DBUF ;ITALICS ENABLE=1 ITALICS=1  
MOV #BIT4,$GDDAT ;LOAD EXPECTED  
MOV DBUF,@DPC ;LOAD DISPLAY P.C.  
MOV SREG0,$BDADR ;SET UP REG ADRS 02  
MOV @SREG0,$BDDAT ;READ DISPLAY STATUS REGISTER  
BIC #177757,$BDDAT ;MASK TO BIT 4  
CMP $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD  
BEQ TST71 ;;BR IF EQUAL  
ERROR 30 ;ITALICS BIT FAILED TO SET
```

1248
1249
1250
(3)
(3)
(2) 012134 000004
(2) 012136 012737 000071 001206
1251 012144 012777 020000 167710
1252 012152 012777 170060 167620
1253 012160 012777 170000 167614
1254 012166 013777 002000 167634
1255 012174 012737 000020 001124
1256 012202 005277 167622
1257 012206 013737 002032 001122
1258 012214 017737 167612 001126
1259 012222 042737 177757 001126
1260 012230 023737 001124 001126
1261 012236 001401
1262 012240 104030

```
*****  
*TEST 71 CHECK ITALIC ENABLE GATE  
*****  
TST71: SCOPE  
MOV #71,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX  
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2  
MOV #170060,@DBUF ;LOAD ENABLE  
MOV #170000,@DBUF1 ;ITALICS ENABLE=0 ITALICS=0  
MOV DBUF,@DPC  
MOV #BIT4,$GDDAT ;LOAD EXPECTED  
INC @DPC ;SINGLE STEP THE DISPLAY  
MOV SREG0,$BDADR ;SET UP REG ADRS 02  
MOV @SREG0,$BDDAT ;READ DISPLAY STATUS REGISTER  
BIC #177757,$BDDAT ;MASK TO BITS 4  
CMP $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD  
BEQ TST72 ;;BR IF EQUAL  
ERROR 30 ;ITALICS ENABLE FAILED TO INHIBIT  
;CLEARING OF ITALICS BIT
```

1263
1264
1265
(3)
(3)
(2) 012242 000004
(2) 012244 012737 000072 001206
1266 012252 013737 002032 001122
1267 012260 012777 020000 167574
1268 012266 012737 172000 031076
1269 012274 013777 002000 167526
1270 012302 012737 100000 001124
1271 012310 017737 167516 001126
1272 012316 100401
1273 012320 104037
1274 012322 012737 160000 031076
1275 012330 012777 031076 167472
1276 012336 005037 001124
1277 012342 017737 167464 001126
1278 012350 100001
1279 012352 104040

```
*****  
*TEST 72 TEST INTERNAL STOP FLAG CAN SET AND RESET  
*****  
TST72: SCOPE  
MOV #72,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX  
MOV SREG0,$BDADR ;SET UP REG ADRS 02  
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2  
MOV #172000,BUFFER ;'STOP' BIT =1  
MOV DBUF,@DPC ;LOAD DISPLAY P.C.  
MOV #BIT15,$GDDAT ;LOAD EXPECTED  
MOV @SREG0,$BDDAT ;READ DISPLAY STATUS REGISTER  
BMI 1$ ;;BR IF SET  
ERROR 37 ;'INTERNAL-STOP' BIT FAILED TO SET  
1$: MOV #160000,BUFFER ;LOAD 'NOP'  
MOV #BUFFER,@DPC ;START DISPLAY  
CLR $GDDAT ;CLEAR EXPECTED  
MOV @SREG0,$BDDAT ;READ DISPLAY STATUS REGISTER  
BPL TST73 ;;BR IF CLEARED  
ERROR 40 ;'INTERNAL-STOP' BIT FAILED TO RESET
```

1280
1281
(3)

```
*****  
*TEST 73 TEST THAT RESET WILL CLEAR STOP, MODE, INTENSITY, ITALICS, BLINK & LINE
```

(3)
 (2) 012354 000004
 (1) 012356 012737 000010 001166
 (2) 012364 012737 000073 001206
 1282 012372 012777 020000 167462
 1283 012400 012737 102637 031076
 1284 012406 012737 170060 031100
 1285 012414 012777 031076 167406
 1286 012422 012737 002000 001124
 1287 012430 005277 167374
 1288 012434 012737 172000 031102
 1289 012442 005277 167362
 1290 012446 013737 002032 001122
 1291 012454 000005
 1292 012456 017737 167350 001126
 1293 012464 023737 001124 001126
 1294 012472 001401
 1295 012474 104032
 1296
 1297

```

:*****
TST73: SCOPE
MOV #10,$TIMES ;;DO 10 ITERATIONS
MOV #73,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;SET UP MAINT SW 2
MOV #102637,BUFFER ;SET UP CHAR
MOV #170060,BUFFER+2 ;SET UP ITALICS INSTR
MOV #BUFFER,@DPC ;START
MOV #2000,$GDDAT ;EXPECT INTENSITY ONLY
INC @DPC ;PICK UP ITALICS INSTR
MOV #172000,BUFFER+4 ;SET UP STOP INSTR
INC @DPC ;PICK UP STOP INSTR
MOV SREG0,$BDADR ;SET UP REG ADRS 02
RESET ;CLR REG 02
MOV @SREG0,$BDDAT ;READ REG
CMP $GDDAT,$BDDAT ;ALL BUT THE MSB OF INTENSITY SHOULD HAVE BEEN CLEARED
BEQ TST74 ;NEXT TEST IF OK
ERROR 32 ;RESET FAILED TO SET UP REG 02
  
```

(3)
 (3)
 (2) 012476 000004
 (1) 012500 012737 000100 001166
 1298 012506 012737 000340 177776
 1299 012514 013737 002032 001122
 1300 012522 012737 172000 001124
 1301 012530 013700 001252
 1302 012534 042700 160000
 1303 012540 012710 012620
 1304 012544 012760 000340 000002
 1305 012552 012777 020040 167302
 1306 012560 012737 173400 031076
 1307 012566 012777 031076 167234
 1308 012574 005037 177776
 1309 012600 021616
 1310 012602 005700
 1311 012604 001430
 1312 012606 017737 167220 001126
 1313 012614 104000
 1314 012616 000423
 1315 012620 022626
 1316 012622 017737 167204 001126
 1317 012630 005700
 1318 012632 001002
 1319 012634 104033
 1320 012636 000413
 1321 012640 023737 001124 001126
 1322 012646 001402
 1323 012650 104033
 1324 012652 000405
 1325 012654 012737 173000 031076
 1326 012662 005000
 1327 012664 000740
 1328 012666 004737 024454
 1329

```

:*****
*TEST 74 TEST THAT INTERNAL STOP WILL CAUSE AN INTERRUPT
:*****
TST74: SCOPE
MOV #100,$TIMES ;;DO 100 ITERATIONS
MOV #340,PSW ;HIGHEST PRIORITY
MOV SREG0,$BDADR ;SET UP REG ADRS 02
MOV #172000,$GDDAT ;EXPECT STOP, MODE AND INTENSITY LEVEL ONLY
MOV $VECT1,R0 ;GET INTR ADRS
BIC #160000,R0 ; CLEAR PSW BITS ** B **
MOV #2$(R0) ;SET UP RETURN ADRS
MOV #340,2(R0) ;HIGHEST PRIORITY ON INTR
MOV #20040,@STKPT ;SET MAINT SW 2
MOV #173400,BUFFER ;ENABLE STOP INTR + STOP
1$: MOV #BUFFER,@DPC ;START
CLR PSW ;ALLOW INTR
CMP (SP),(SP) ;FUMBLE
TST R0 ;EXPECT INTR?
BEQ 5$ ;BR IF NOT
MOV @SREG0,$BDDAT ;READ STATUS
ERROR ;INTERNAL STOP FAILED TO INTR
BR 5$ ;LOOK FOR LOOP ON TEST SW
2$: CMP (SP)+,(SP)+ ;FIX STACK SINCE NO RTI
MOV @SREG0,$BDDAT ;READ REG 02
TST R0 ;EXPECT INTR?
BNE 3$ ;BR IF SO
ERROR 33 ;INTERNAL STOP INTR'ED WHEN NOT ENABLED
BR 5$ ;GO LOOK FOR LOOP ON TEST SW
3$: CMP $GDDAT,$BDDAT ;STATUS CORRECT?
BEQ 4$ ;BR IF SO
ERROR 33 ;STOP FLAG NOT SET ON INTR
BR 5$ ;GO LOOK FOR LOOP ON TEST SW
4$: MOV #173000,BUFFER ;SET UP NOW WITH INTR NOT ENABLED
CLR R0 ;INDICATE NO INTR
BR 1$ ;REPEAT TEST
5$: JSR PC,RSTVEC ;GO RESTORE VECTORS
  
```

1330
(3)
(3)
(2) 012672 000004
(2) 012674 012737 000075 001206
1331 012702 012777 020000 167152
1332 012710 012777 170002 167062
1333 012716 005037 001124
1334 012722 013777 002000 167100
1335 012730 013737 002042 001122
1336 012736 017737 167100 001126
1337 012744 042737 177677 001126
1338 012752 023737 001124 001126
1339 012760 001401
1340 012762 104031
1341
1342
1343
(3)
(3)
(2) 012764 000004
(2) 012766 012737 000076 001206
1344 012774 012777 020000 167060
1345 013002 012777 170003 166770
1346 013010 012737 000100 001124
1347 013016 013777 002000 167004
1348 013024 013737 002042 001122
1349 013032 017737 167004 001126
1350 013040 042737 177677 001126
1351 013046 023737 001124 001126
1352 013054 001401
1353 013056 104031
1354
1355
1356
(3)
(3)
(2) 013060 000004
(2) 013062 012737 000077 001206
1357 013070 013737 002042 001122
1358 013076 012777 020000 166756
1359 013104 012737 170002 031076
1360 013112 012777 031076 166710
1361 013120 012737 170001 031100
1362 013126 005277 166676
1363 013132 005037 001124
1364 013136 017737 166700 001126
1365 013144 042737 177677 001126
1366 013152 023737 001124 001126
1367 013160 001401
1368 013162 104031
1369
1370
1371
(3)
(3)

```
*****  
*TEST 75 LOAD #0 INTO MENU REGISTER  
*****  
TST75: SCOPE  
MOV #75,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX  
MOV #BIT13,@STKPT ;;SET MAINT SWITCH #2  
MOV #170002,@DBUF ;MENU ENABLE=1 MENU=0  
CLR $GDDAT ;LOAD EXPECTED  
MOV DBUF,@DPC ;LOAD DISPLAY P.C.  
MOV SREG1,$BDADR ;SET UP REG ADRS12  
MOV @SREG1,$BDDAT ;READ DISP STATUS REGISTER  
BIC #177677,$BDDAT ;MASK TO BIT 2  
CMP $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD  
BEQ TST76 ;;BR IF EQUAL  
ERROR 31 ;MENU BIT FAILED TO RESET
```

```
*****  
*TEST 76 LOAD #1 INTO MENU REGISTER  
*****  
TST76: SCOPE  
MOV #76,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX  
MOV #BIT13,@STKPT ;;SET MAINT SWITCH #2  
MOV #170003,@DBUF ;MENU ENABLE=1 MENU=1  
MOV #BIT6,$GDDAT ;LOAD EXPECTED  
MOV DBUF,@DPC ;LOAD DISPLAY P.C.  
MOV SREG1,$BDADR ;SET UP REG ADRS 12  
MOV @SREG1,$BDDAT ;READ DISPLAY STATUS REGISTER  
BIC #177677,$BDDAT ;MASK TO BIT 2  
CMP $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD  
BEQ TST77 ;;BR IF EQUAL  
ERROR 31 ;MENU BIT FAILED TO SET
```

```
*****  
*TEST 77 CHECK MENU ENABLE GATE  
*****  
TST77: SCOPE  
MOV #77,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX  
MOV SREG1,$BDADR ;SET UP REG ADRS 12  
MOV #BIT13,@STKPT ;;SET MAINT SWITCH #2  
MOV #170002,BUFFER ;MENU ENABLE = 1 MENU = 0  
MOV #BUFFER,@DPC ;START DISPLAY  
MOV #170001,BUFFER+2 ;MENU ENABLE=0 MENU=1  
INC @DPC ;RESUME  
CLR $GDDAT ;LOAD EXPECTED  
MOV @SREG1,$BDDAT ;READ DISPLAY STATUS REGISTER  
BIC #177677,$BDDAT ;MASK TO BIT 2  
CMP $GDDAT,$BDDAT ;COMPARE EXPCT TO RCVD  
BEQ TST100 ;;BR IF EQUAL  
ERROR 31 ;MENU ENABLE FAILED TO INHIBIT  
;RESETTING OF MENU BIT
```

```
*****  
*TEST 100 TEST THAT LOAD STATUS C INSTR CAN SET THE VECTOR SCALE  
*****
```

(2) 013164 000004
(2) 013166 012737 000100 001206
1372 013174 012777 020000 166660
1373 013202 012737 013216 001110
1374 013210 012737 000017 001124
1375 013216 012737 154020 031076
1376 013224 053737 001124 031076
1377 013232 012777 031076 166570
1378 013240 013737 002042 001122
1379 013246 017737 166570 001126
1380 013254 042737 177760 001126
1381 013262 023737 001124 001126
1382 013270 001401
1383 013272 104025
1384 013274 005337 001124
1385 013300 100346
1386
1387
(3)
(3)
(2) 013302 000004
(2) 013304 012737 000101 001206
1388 013312 012777 020000 166542
1389 013320 012737 154037 031076
1390 013326 012737 000017 001124
1391 013334 012777 031076 166466
1392 013342 012737 154000 031100
1393 013350 005277 166454
1394 013354 013737 002042 001122
1395 013362 017737 166454 001126
1396 013370 042737 177760 001126
1397 013376 023737 001124 001126
1398 013404 001401
1399 013406 104025
1400
1401
(3)
(3)
(2) 013410 000004
(2) 013412 012737 000102 001206
1402 013420 012777 020000 166434
1403 013426 012737 013450 001110
1404 013434 012737 154340 031076
1405 013442 012737 001400 001124
1406 013450 012777 031076 166352
1407 013456 013737 002042 001122
1408 013464 017737 166352 001126
1409 013472 042737 176377 001126
1410 013500 023737 001124 001126
1411 013506 001402
1412 013510 001411
1413 013512 104026
1414 013514 162737 000400 001124
1415 013522 100404
1416 013524 162737 000040 031076
1417 013532 000746

TST100: SCOPE
MOV #100,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
MOV #1\$,\$LPERR ;SET UP SCOPE LOOP ADRS
MOV #17,\$GDDAT ;START WITH ALL ONES
1\$: MOV #154020,BUFFER ;SET UP INSTR AT BUFFER
BIS \$GDDAT,BUFFER ;SET UP VECTOR SCALE AT INSTR LOC
MOV #BUFFER,@DPC ;START
MOV SREG1,\$BDADR ;SET UP REG ADRS 12
MOV @SREG1,\$BDDAT ;READ VECTOR SCALE
BIC #177760,\$BDDAT ;ONLY INTERESTED IN VECTOR SCALE
CMP \$GDDAT,\$BDDAT ;ARE THEY CORRECT?
BEQ 2\$;BR IF OK
ERROR 25 ;VECTOR SCALE INSTR ER
2\$: DEC \$GDDAT ;ADVANCE PATTERN
BPL 1\$;BR IF NOT DONE

:*****
:*TEST 101 TEST THAT THE VECTOR SCALE DOES NOT CHANGE WHEN CHANGE ENA=0
:*****
TST101: SCOPE
MOV #101,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
MOV #154037,BUFFER ;SET UP INSTR AT BUFFER-VECTOR SCALE ENA SET
MOV #17,\$GDDAT ;SET UP VECTOR SCALE EXPECTED
MOV #BUFFER,@DPC ;START
MOV #154000,BUFFER+2 ;SET UP INSTR AT BUFFER-NO VECTOR SCALE ENA
INC @DPC ;RESUME
MOV SREG1,\$BDADR ;SET UP REG ADRS 12
MOV @SREG1,\$BDDAT ;READ VECTOR SCALE
BIC #177760,\$BDDAT ;SAVE VECTOR SCALE ONLY
CMP \$GDDAT,\$BDDAT ;DOES IT STILL EQ 17
BEQ TST102 ;ADVANCE TO NEXT TEST IF OK
ERROR 25 ;VECTOR SCALE CHANGE ENABLE INSTR ER

:*****
:*TEST 102 TEST THAT LOAD STATUS C INSTR CAN SET THE CHARACTER SCALE
:*****
TST102: SCOPE
MOV #102,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
MOV #1\$,\$LPERR ;SET UP SCOPE LOOP ADRS
MOV #154340,BUFFER ;SET UP INSTR AT BUFFER
MOV #1400,\$GDDAT ;START WITH ALL ONES
1\$: MOV #BUFFER,@DPC ;START
MOV SREG1,\$BDADR ;SET UP REG ADRS 12
MOV @SREG1,\$BDDAT ;READ CHAR SCALE
BIC #176377,\$BDDAT ;SAVE CHAR SCALE ONLY
CMP \$GDDAT,\$BDDAT ;IS IT CORRECT?
BEQ 2\$;BR IF OK
BEQ TST103 ;ADVANCE TO NEXT TEST IF OK
ERROR 26 ;CHARACTER SCALE INSTR ER
2\$: SUB #400,\$GDDAT ;ADVANCE PATTERN
BMI TST103 ;ADVANCE TO NEXT TEST IF DONE
SUB #40,BUFFER ;ADVANCE PATTERN AT INSTR LOC
BR 1\$;DO NEXT CHAR SCALE

1418
1419
(3)
(3)
(2) 013534 000004
(2) 013536 012737 000103 001206
1420 013544 012777 020000 166310
1421 013552 012737 154340 031076
1422 013560 012737 001400 001124
1423 013566 012777 031076 166234
1424 013574 012737 154000 031100
1425 013602 005277 166222
1426 013606 013737 002042 001122
1427 013614 017737 166222 001126
1428 013622 042737 176377 001126
1429 013630 023737 001124 001126
1430 013636 001401
1431 013640 104026

```
*****
:*TEST 103 TEST THAT THE CHAR SCALE DOES NOT CHANGE WHEN CHANGE ENA=0
*****
TST103: SCOPE
MOV #103,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;;SET MAINT SWITCH #2
MOV #154340,BUFFER ;SET UP INSTR AT BUFFER-CHAR SCALE ENA SET
MOV #1400,$GDDAT ;SET UP CHAR SCALE EXPECTED
MOV #BUFFER,@DPC ;START
MOV #154000,BUFFER+2 ;SET UP INSTR AT BUFFER-NO CHAR SCALE ENA
INC @DPC ;RESUME
MOV SREG1,$BDADR ;SET UP REG ADRS 12
MOV @SREG1,$BDDAT ;READ CHAR SCALE
BIC #176377,$BDDAT ;SAVE CHAR SCALE ONLY
CMP $GDDAT,$BDDAT ;IS IT CORRECT?
BEQ TST104 ;;GO TO NEXT TEST IF NO CHANGE
ERROR 26 ;CHARACTER SCALE CHANGE ENABLE INSTR ER
```

1432
1433
(3)
(3)
(2) 013642 000004
(2) 013644 012737 000104 001206
1434 013652 012777 020000 166202
1435 013660 012737 013702 001110
1436 013666 012737 155400 031076
1437 013674 012737 002000 001124
1438 013702 012777 031076 166120
1439 013710 013737 002042 001122
1440 013716 017737 166120 001126
1441 013724 042737 175777 001126
1442 013732 023737 001124 001126
1443 013740 001401
1444 013742 104026
1445 013744 162737 002000 001124
1446 013752 100404
1447 013754 162737 000400 031076
1448 013762 000747
1449

```
*****
:*TEST 104 TEST THAT LOAD STATUS C INSTR CAN SET THE CHARACTER ROTATE
*****
TST104: SCOPE
MOV #104,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;;SET MAINT SWITCH #2
MOV #1$, $LPERR ;SET UP SCOPE LOOP ADRS
MOV #155400,BUFFER ;SET UP INSTR AT BUFFER
MOV #2000,$GDDAT ;EXPECT CHAR ROTATE BIT
1$: MOV #BUFFER,@DPC ;START
MOV SREG1,$BDADR ;SET UP REG ADRS 12
MOV @SREG1,$BDDAT ;READ CHAR ROTATE BIT
BIC #175777,$BDDAT ;SAVE CHAR ROTATE BIT ONLY
CMP $GDDAT,$BDDAT ;IS IT CORRECT?
BEQ 2$ ;BR IF OK
ERROR 26 ;CHARACTER ROTATE INSTR ER
2$: SUB #2000,$GDDAT ;ADVANCE PATTERN
BMI TST105 ;;ADVANCE TO NEXT TEST IF OK
SUB #400,BUFFER ;ADVANCE PATTERN AT INSTR LOC
BR 1$ ;NOW WRITE A ZERO INTO CHAR ROTATE
```

1450
(3)
(3)
(2) 013764 000004
(2) 013766 012737 000105 001206
1451 013774 012777 020000 166060
1452 014002 012737 155400 031076
1453 014010 012737 002000 001124
1454 014016 012777 031076 166004
1455 014024 012737 154000 031100
1456 014032 005277 165772
1457 014036 013737 002042 001122
1458 014044 017737 165772 001126
1459 014052 042737 175777 001126
1460 014060 023737 001124 001126
1461 014066 001401

```
*****
:*TEST 105 TEST THAT THE CHARACTER ROTATE DOES NOT CHANGE WHEN CHANGE ENA=0
*****
TST105: SCOPE
MOV #105,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;;SET MAINT SWITCH #2
MOV #155400,BUFFER ;SET UP INSTR AT BUFFER CHAR ROTATE ENA SET
MOV #2000,$GDDAT ;EXPECT CHAR ROTATE BIT
MOV #BUFFER,@DPC ;START
MOV #154000,BUFFER+2 ;SET UP INSTR AT BUFFER-WITHOUT CHAR ROTATE ENA
INC @DPC ;RESUME
MOV SREG1,$BDADR ;SET UP REG ADRS 12
MOV @SREG1,$BDDAT ;READ CHAR ROTATE BIT
BIC #175777,$BDDAT ;SAVE CHAR ROTATE BIT
CMP $GDDAT,$BDDAT ;IS IT SET?
BEQ TST106 ;;ADVANCE TO NEXT IF OK
```

1462 014070 104026
1463
1464
1465
(3)
(3)
(2) 014072 000004
(1) 014074 012737 000040 001166
(2) 014102 012737 000106 001206
1466 014110 012777 020000 165744
1467 014116 012737 155733 031076
1468 014124 012737 170003 031100
1469 014132 012777 031076 165670
1470 014140 012737 000404 001124
1471 014146 005277 165656
1472 014152 013737 002042 001122
1473 014160 000005
1474 014162 017737 165654 001126
1475 014170 023737 001124 001126
1476 014176 001401
1477 014200 104032
1478
1479
1480
1481
(3)
(3)
(2) 014202 000004
(2) 014204 012737 000107 001206
1482 014212 012737 014240 001110
1483 014220 012777 020000 165634
1484 014226 012737 174100 031076
1485 014234 005037 001124
1486 014240 012777 031076 165562
1487 014246 013737 002034 001122
1488 014254 017737 165554 001126
1489 014262 042737 001777 001126
1490 014270 023737 001124 001126
1491 014276 001401
1492 014300 104007
1493 014302 005237 031076
1494 014306 062737 002000 001124
1495 014314 103351
1496
1497
(3)
(3)
(2) 014316 000004
(2) 014320 012737 000110 001206
1498 014326 013737 002034 001122
1499 014334 012777 020000 165520
1500 014342 012777 174100 165430
1501 014350 013777 002000 165452
1502 014356 012737 174077 031100
1503 014364 005277 165440
1504 014370 005037 001124

ERROR 26 ;CHARACTER ROTATE CHANGE ENABLE INSTR ER

*TEST 106 TEST THAT RESET WILL CLEAR ROTATE, CHAR SCALE, MENU & VECTOR SCALE

TST106: SCOPE
MOV #40,\$TIMES ;DO 40 ITERATIONS
MOV #106,\$TESTN ;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;SET MAINT SW 2
MOV #155733,BUFFER ;SET UP 'LOAD STATUS C' INSTR
MOV #170003,BUFFER+2 ;SET UP MENU INSTR
MOV #BUFFER,@DPC ;START
MOV #404,\$GDDAT ;EXPECT MSB OF VECTOR SCALE ONLY
INC @DPC ;PICK UP MENU INSTR
MOV SREG1,\$BDADR ;SET UP REG ADRS 12
RESET ;CLR REG 12
MOV @SREG1,\$BDDAT ;READ STATUS REG 1
CMP \$GDDAT,\$BDDAT ;IS IT CORRECT?
BEQ TST107 ;NEXT TEST IF OK
ERROR 32 ;RESET FAILED TO SET UP REG 12

;GRAPHPLOT INCREMENT REGISTER TESTS

*TEST 107 LOAD 0-77 INTO GRAPHPLOT INCREMENT REGISTER

TST107: SCOPE
MOV #107,\$TESTN ;SET TEST NUMBER IN APT MAIL BOX
MOV #1\$,\$LPERR ;SET UP SCOPE LOOP ADRS
MOV #BIT13,@STKPT ;SET MAINT SW 2
MOV #174100,BUFFER ;LOAD GRAPHPLOT INSTR
CLR \$GDDAT ;LD EXPECTED
1\$: MOV #BUFFER,@DPC ;START
MOV XPOS,\$BDADR ;SET UP REG ADRS 04
MOV @XPOS,\$BDDAT ;READ GRAPHPLOT
BIC #1777,\$BDDAT ;SAVE ONLY GRAPHPLOT
CMP \$GDDAT,\$BDDAT ;CORRECT?
BEQ 2\$;BR IF SO
ERROR 7 ;GRAPHPLOT FAILED TO LOAD CORRECTLY
2\$: INC BUFFER ;ADVANCE GRAPHPLOT AT INSTR
ADD #2000,\$GDDAT ;ADVANCE EXPECTED
BCC 1\$;BR IF NOT COMBS TESTED

*TEST 110 CHECK THE GRAPHPLOT INCREMENT ENABLE REGISTER

TST110: SCOPE
MOV #110,\$TESTN ;SET TEST NUMBER IN APT MAIL BOX
MOV XPOS,\$BDADR ;SET UP REG ADRS 04
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
MOV #174100,@DBUF ;LOAD GRAPHPLOT COUNTER WITH 0
MOV DBUF,@DPC ;START DISPLAY
MOV #174077,BUFFER+2 ;LOAD GRAPHPLOT NO ENABLE
INC @DPC ;RESUME
CLR \$GDDAT ;LOAD EXPECTED

1505 014374 017737 165434 001126
1506 014402 042737 001777 001126
1507 014410 023737 001124 001126
1508 014416 001401
1509 014420 104007
1510
1511
1512

MOV @XPOS,\$BDDAT ;READ INCREMENT REGISTER
BIC #1777,\$BDDAT ;MASK TO BITS 15-10
CMP \$GDDAT,\$BDDAT ;COMPARE EXPCT TO RCVD
BEQ TST111 ;:BR IF EQUAL
ERROR 7 ;GRAPHPLOT REGISTER CHANGED WITHOUT
; THE ENABLE BEING SET

(3)
(3)

::*****
:*TEST 111 TEST THAT RESET WILL CLEAR GRAPHPLOT INCREMENT
::*****

(2) 014422 000004
(1) 014424 012737 000040 001166
(2) 014432 012737 000111 001206
1513 014440 013737 002034 001122
1514 014446 012777 020000 165406
1515 014454 012737 174177 031076
1516 014462 012777 031076 165340
1517 014470 005037 001124
1518 014474 000005
1519 014476 017737 165332 001126
1520 014504 001401
1521 014506 104032
1522

TST111: SCOPE
MOV #40,\$TIMES ;:DO 40 ITERATIONS
MOV #111,\$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
MOV XPOS,\$BDADR ;:SET UP REG ADRS 04
MOV #BIT13,@STKPT ;:SET MAINT SW 2
MOV #174177,BUFFER ;:SET UP GRAPHPLOT INCREMENT INSTR
MOV #BUFFER,@DPC ;:START
CLR \$GDDAT ;:EXPECT ZERO
RESET ;:CLR REG 04
MOV @XPOS,\$BDDAT ;:READ REG 04
BEQ TST112 ;:NEXT TEST IF ZERO
ERROR 32 ;:RESET FAILED TO CLR GRAPHPLOT INC

(3)
(3)

::*****
:*TEST 112 TEST THAT LOAD STATUS B INSTR CAN SET THE COLOR LEVEL
::*****

(2) 014510 000004
(2) 014512 012737 000112 001206
1524 014520 012777 020000 165334
1525 014526 012737 014550 001110
1526 014534 012737 175600 031076
1527 014542 012737 000014 001124
1528 014550 012777 031076 165252
1529 014556 013737 002052 001122
1530 014564 017737 165262 001126
1531 014572 042737 177763 001126
1532 014600 023737 001124 001126
1533 014606 001401
1534 014610 104035
1535 014612 162737 000004 001124
1536 014620 100404
1537 014622 162737 000200 031076
1538 014630 000747
1539
1540

TST112: SCOPE
MOV #112,\$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;:SET MAINT SWITCH #2
MOV #1,\$LPERR ;:SET UP SCOPE LOOP ADRS
MOV #175600,BUFFER ;:SET UP INSTR AT BUFFER
MOV #14,\$GDDAT ;:EXPECT ALL ONES AT START
1\$: MOV #BUFFER,@DPC ;:START
MOV CONS,\$BDADR ;:SET UP REG ADRS 22
MOV @CONS,\$BDDAT ;:READ COLOR LEVEL
BIC #177763,\$BDDAT ;:ONLY INTERESTED IN THE COLOR LEVEL
CMP \$GDDAT,\$BDDAT ;:IS IT CORRECT?
BEQ 2\$;:BR IF OK
ERROR 35 ;:COLOR LEVEL INSTR ER
2\$: SUB #4,\$GDDAT ;:ADVANCE PATTERN
BMI TST113 ;:ADVANCE TO NEXT TEST IF ALL PATTERNS TESTED
SUB #200,BUFFER ;:SET UP NEXT PATTERN AT INSTR LOC
BR 1\$;:TRY NEXT PATTERN

(3)
(3)

::*****
:*TEST 113 TEST THAT THE COLOR LEVEL DOES NOT CHANGE WHEN CHANGE ENA=0
::*****

(2) 014632 000004
(2) 014634 012737 000113 001206
1541 014642 012777 020000 165212
1542 014650 012737 175600 031076
1543 014656 012737 000014 001124
1544 014664 012777 031076 165136
1545 014672 012737 174000 031100
1546 014700 005277 165124
1547 014704 013737 002052 001122

TST113: SCOPE
MOV #113,\$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;:SET MAINT SWITCH #2
MOV #175600,BUFFER ;:SET UP INSTR AT BUFFER-COLOR LEVEL ENA SET
MOV #14,\$GDDAT ;:SET UP COLOR LEVEL EXPECTED
MOV #BUFFER,@DPC ;:START
MOV #174000,BUFFER+2 ;:SET UP INSTR AT BUFFER-NO COLOR LEVEL ENA
INC @DPC ;:RESUME
MOV CONS,\$BDADR ;:SET UP REG ADRS 22

```

1548 014712 017737 165134 001126      MOV    @CONS,$BDDAT    ;READ COLOR LEVEL
1549 014720 042737 177763 001126      BIC    #177763,$BDDAT ;ONLY INTERESTED IN THE COLOR LEVEL
1550 014726 023737 001124 001126      CMP    $GDDAT,$BDDAT  ;IS IT CORRECT?
1551 014734 001401                                BEQ    TST114          ;ADVANCE TO NEXT TEST IF OK
1552 014736 104035                                ERROR  35              ;COLOR LEVEL CHANGE ENABLE INSTR ER
1553
1554                                     ;*****
(3)                                     ;*TEST 114      TEST THAT NOP INSTR CAN SET & RESET INTENSITY ENABLED-CONSOLE 0
(3)                                     ;*****
(2) 014740 000004                                TST114: SCOPE
(2) 014742 012737 000114 001206      MOV    #114,$TESTN    ;;SET TEST NUMBER IN APT MAIL BOX
1555 014750 012777 020000 165104      MOV    #BIT13,@STKPT ;SET MAINT SWITCH #2
1556 014756 012737 015000 001110      MOV    #1$,$LPERR    ;SET UP SCOPE LOOP ADRS
1557 014764 012737 100000 001124      MOV    #100000,$GDDAT ;EXPECT IT TO BE SET INITIALLY
1558 014772 012737 164300 031076      MOV    #164300,BUFFER ;SET UP INSTR LOC
1559 015000 012777 031076 165022 1$:  MOV    #BUFFER,@DPC   ;START
1560 015006 013737 002052 001122      MOV    CONS,$BDADR   ;SET UP REG ADRS 22
1561 015014 017737 165032 001126      MOV    @CONS,$BDDAT ;READ CONSOLE 0 INT ENA REG
1562 015022 042737 077777 001126      BIC    #77777,$BDDAT ;SAVE ONLY INTENSITY ENABLE BIT
1563 015030 023737 001124 001126      CMP    $GDDAT,$BDDAT ;IS IT CORRECT?
1564 015036 001401                                BEQ    2$             ;BR IF SO
1565 015040 104036                                ERROR  36             ;CONSOLE 0 INTENSITY ENABLE INSTR ER
1566 015042 162737 100000 001124 2$:  SUB    #100000,$GDDAT ;GO TO RESET STATE
1567 015050 100404                                BMI    TST115        ;GO TO NEXT TEST IF BOTH STATES TESTED
1568 015052 042737 000100 031076      BIC    #100,BUFFER   ;CLR INT ENABLED BIT AT INSTR LOC
1569 015060 000747                                BR     1$             ;GO TRY RESET STATE
1570
1571                                     ;*****
(3)                                     ;*TEST 115      TEST THAT CONSOLE 0 INTENSITY ENABLED DOES NOT RESET WHEN ENA=0
(3)                                     ;*****
(2) 015062 000004                                TST115: SCOPE
(2) 015064 012737 000115 001206      MOV    #115,$TESTN    ;;SET TEST NUMBER IN APT MAIL BOX
1572 015072 012777 020000 164762      MOV    #BIT13,@STKPT ;SET MAINT SWITCH #2
1573 015100 012737 100000 001124      MOV    #100000,$GDDAT ;EXPECT INTENSITY TO BE SET
1574 015106 012737 164300 031076      MOV    #164300,BUFFER ;SET UP INSTR
1575 015114 012777 031076 164706      MOV    #BUFFER,@DPC   ;START
1576 015122 012737 164000 031100      MOV    #164000,BUFFER+2 ;SET UP INSTR WITH ENA=0
1577 015130 005277 164674                                INC    @DPC           ;ADVANCE TO NEXT INSTR
1578 015134 013737 002052 001122      MOV    CONS,$BDADR   ;SET UP REG ADRS 22
1579 015142 005777 164704                                TST    @CONS         ;IS INTENSITY STILL SET?
1580 015146 100403                                BMI    TST116        ;ADVANCE TO NEXT TEST IF STILL SET
1581 015150 005037 001126      CLR    $BDDAT        ;INDICATE IT WAS CLEARED
1582 015154 104036                                ERROR  36             ;CONSOLE 0 INTENSITY CHANGE ENABLE INSTR ER
1583
1584                                     ;*****
(3)                                     ;*TEST 116      TEST THAT NOP INSTR CAN SET & RESET INTENSITY ENABLED-CONSOLE 1
(3)                                     ;*****
(2) 015156 000004                                TST116: SCOPE
(2) 015160 012737 000116 001206      MOV    #116,$TESTN    ;;SET TEST NUMBER IN APT MAIL BOX
1585 015166 012777 020000 164666      MOV    #BIT13,@STKPT ;SET MAINT SWITCH #2
1586 015174 013737 015216 001110      MOV    1$,$LPERR     ;SET UP SCOPE LOOP ADRS
1587 015202 012737 001000 001124      MOV    #1000,$GDDAT  ;EXPECT IT TO BE SET INITIALLY
1588 015210 012737 164700 031076      MOV    #164700,BUFFER ;SET UP INSTR LOCATION
1589 015216 012777 031076 164604 1$:  MOV    #BUFFER,@DPC   ;START
1590 015224 013737 002052 001122      MOV    CONS,$BDADR   ;SET UP REG ADRS 22
1591 015232 017737 164614 001126      MOV    @CONS,$BDDAT ;READ CONSOLE 1 INT ENA REG

```

1592 015240 042737 176777 001126
1593 015246 023737 001124 001126
1594 015254 001401
1595 015256 104036
1596 015260 162737 001000 001124
1597 015266 100404
1598 015270 042737 000100 031076
1599 015276 000747

BIC #176777,\$BDDAT ;SAVE ONLY INT ENA
CMP \$GDDAT,\$BDDAT ;IS IT CORRECT?
BEQ 2\$;BR IF OK
ERROR 36 ;CONSOLE 1 INTENSITY ENABLE INSTR ER
2\$: SUB #1000,\$GDDAT ;GO TO RESET STATE
BMI TST117 ;GO TO NEXT TEST IF BOTH STATES TESTED
BIC #100,BUFFER ;CLR INT ENABLED BIT AT INSTR LOC
BR 1\$;GO TRY RESET STATE

1600
1601
(3)
(3)

*TEST 117 TEST THAT CONSOLE 1 INTENSITY ENABLED DOES NOT RESET WHEN ENA=0

(2) 015300 000004
(2) 015302 012737 000117 001206
1602 015310 012777 020000 164544
1603 015316 012737 001000 001124
1604 015324 012737 164700 031076
1605 015332 012777 031076 164470
1606 015340 012737 164400 031100
1607 015346 005277 164456
1608 015352 013737 002052 001122
1609 015360 032777 001000 164464
1610 015366 001003
1611 015370 005037 001126
1612 015374 104036

TST117: SCOPE
MOV #117,\$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
MOV #1000,\$GDDAT ;EXPECT INTENSITY TO BE SET
MOV #164700,BUFFER ;SET UP INSTR
MOV #BUFFER,@DPC ;START
MOV #164400,BUFFER+2 ;SET UP INSTR WITH ENA=0
INC @DPC ;ADVANCE TO NEXT INSTR
MOV CONS,\$BDDADR ;SET UP REG ADRS 22
BIT #1000,@CONS ;IS INTENSITY STILL SET?
BNE TST120 ;ADVANCE TO NEXT TEST IF STILL SET
CLR \$BDDAT ;INDICATE IT WAS CLEARED
ERROR 36 ;CONSOLE 1 INTENSITY CHANGE ENABLE INSTR ER

1613
1614
(3)
(3)

*TEST 120 TEST THAT NOP INSTR CAN SET & RESET L.P. INTR ENABLED-CONSOLE 0

(2) 015376 000004
(2) 015400 012737 000120 001206
1615 015406 012777 020000 164446
1616 015414 012737 015436 001110
1617 015422 012737 004000 001124
1618 015430 012737 164060 031076
1619 015436 012777 031076 164364
1620 015444 013737 002052 001122
1621 015452 017737 164374 001126
1622 015460 042737 173777 001126
1623 015466 023737 001124 001126
1624 015474 001401
1625 015476 104042
1626 015500 162737 004000 001124
1627 015506 100404
1628 015510 042737 000020 031076
1629 015516 000747

TST120: SCOPE
MOV #120,\$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
MOV #1\$,\$LPERR ;SET UP SCOPE LOOP ADRS
MOV #4000,\$GDDAT ;EXPECT IT TO BE SET INITIALLY
MOV #164060,BUFFER ;SET UP INSTR LOC
1\$: MOV #BUFFER,@DPC ;START
MOV CONS,\$BDDADR ;SET UP REG ADRS 22
MOV @CONS,\$BDDAT ;READ CONSOLE 0 LP INTR ENABLED REG
BIC #173777,\$BDDAT ;SAVE ONLY LP INTR ENABLED BIT
CMP \$GDDAT,\$BDDAT ;IS IT CORRECT?
BEQ 2\$;BR IF OK
ERROR 42 ;CONSOLE 0 L.P. INTR ENABLE INSTR ER
2\$: SUB #4000,\$GDDAT ;GO TO RESET STATE
BMI TST121 ;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
BIC #20,BUFFER ;CLR THE ENABLED BIT AT INSTR LOC
BR 1\$;GO TRY RESET STATE

1630
1631
(3)
(3)

*TEST 121 TEST THAT CONSOLE 0 L.P. INTR ENABLED DOES NOT RESET WHEN ENA=0

(2) 015520 000004
(2) 015522 012737 000121 001206
1632 015530 012777 020000 164324
1633 015536 012737 004000 001124
1634 015544 012737 164060 031076
1635 015552 012777 031076 164250

TST121: SCOPE
MOV #121,\$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
MOV #4000,\$GDDAT ;EXPECT L.P. INTR ENABLED TO BE SET
MOV #164060,BUFFER ;SET UP INSTR LOC
MOV #BUFFER,@DPC ;START

```

1636 015560 012737 164000 031100 MOV #164000,BUFFER+2 ;SET UP INSTR WITH ENA=0
1637 015566 005277 164236 INC @DPC ;ADVANCE TO NEXT INSTR
1638 015572 013737 002052 001122 MOV CONS,$BDADR ;SET UP REG ADRS 22
1639 015600 032777 004000 164244 BIT #4000,@CONS ;IS L.P. INTR ENABLED STILL SET?
1640 015606 001003 BNE TST122 ;ADVANCE TO NEXT TEST IF STILL SET
1641 015610 005037 001126 CLR $BDDAT ;INDICATE IT WAS CLEARED
1642 015614 104042 ERROR 42 ;CONSOLE 0 L.P. CHANGE INTR ENABLE INSTR ER
  
```

```

1643
1644 ::*****
  (3) ;*TEST 122 TEST THAT NOP INSTR CAN SET & RESET L.P. INTR ENABLED-CONSOLE 1
  (3) ::*****
  
```

```

(2) 015616 000004 TST122: SCOPE
(2) 015620 012737 000122 001206 MOV #122,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
1645 015626 012777 020000 164226 MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
1646 015634 012737 015656 001110 MOV #1,$LPERR ;SET UP SCOPE LOOP ADRS
1647 015642 012737 000040 001124 MOV #40,$GDDAT ;EXPECT IT TO BE SET INITIALLY
1648 015650 012737 164460 031076 MOV #164460,BUFFER ;SET UP INSTR LOC
1649 015656 012777 031076 164144 1$: MOV #BUFFER,@DPC ;START
1650 015664 013737 002052 001122 MOV CONS,$BDADR ;SET UP REG ADRS 22
1651 015672 017737 164154 001126 MOV @CONS,$BDDAT ;READ CONSOLE 1 LP INTR ENABLED REG
1652 015700 042737 177737 001126 BIC #177737,$BDDAT ;SAVE ONLY LP INTR ENABLED BIT
1653 015706 023737 001124 001126 CMP $GDDAT,$BDDAT ;IS IT CORRECT?
1654 015714 001401 BEQ 2$ ;BR IF OK
1655 015716 104042 ERROR 42 ;CONSOLE 1 L.P. INTR ENABLE INSTR ER
1656 015720 162737 000040 001124 2$: SUB #40,$GDDAT ;GO TO RESET STATE
1657 015726 100404 BMI TST123 ;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
1658 015730 042737 000020 031076 BIC #20,BUFFER ;CLR THE ENABLED BIT AT INSTR LOC
1659 015736 000747 BR 1$ ;GO TRY RESET STATE
  
```

```

1660
1661 ::*****
  (3) ;*TEST 123 TEST THAT CONSOLE 1 L.P. INTR ENABLED DOES NOT RESET WHEN ENA=0
  (3) ::*****
  
```

```

(2) 015740 000004 TST123: SCOPE
(2) 015742 012737 000123 001206 MOV #123,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
1662 015750 012777 020000 164104 MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
1663 015756 012737 000040 001124 MOV #40,$GDDAT ;EXPECT L.P. INTR ENABLED TO BE SET
1664 015764 012737 164460 031076 MOV #164460,BUFFER ;SET UP INSTR LOC
1665 015772 012777 031076 164030 MOV #BUFFER,@DPC ;START
1666 016000 012737 164400 031100 MOV #164400,BUFFER+2 ;SET UP INSTR WITH ENA=0
1667 016006 005277 164016 INC @DPC ;ADVANCE TO NEXT INSTR
1668 016012 013737 002052 001122 MOV CONS,$BDADR ;SET UP REG ADRS 22
1669 016020 032777 000040 164024 BIT #40,@CONS ;IS L.P. INTR ENABLED STILL SET?
1670 016026 001003 BNE TST124 ;ADVANCE TO NEXT TEST IF STILL SET
1671 016030 005037 001126 CLR $BDDAT ;INDICATE IT WAS CLEARED
1672 016034 104042 ERROR 42 ;CONSOLE 1 L.P. CHANGE INTR ENABLE INSTR ER
  
```

```

1673
1674 ::*****
  (3) ;*TEST 124 TEST THAT NOP INSTR CAN SET & RESET L.P. SWITCH INTR ENABLED-CONSOLE 0
  (3) ::*****
  
```

```

(2) 016036 000004 TST124: SCOPE
(2) 016040 012737 000124 001206 MOV #124,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
1675 016046 012777 020000 164006 MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
1676 016054 012737 016076 001110 MOV #1,$LPERR ;SET UP SCOPE LOOP ADRS
1677 016062 012737 002000 001124 MOV #2000,$GDDAT ;EXPECT LP SW INTR ENABLED BIT
1678 016070 012737 164014 031076 MOV #164014,BUFFER ;SET UP INSTR
1679 016076 012777 031076 163724 1$: MOV #BUFFER,@DPC ;START
  
```

1680 016104 013737 002052 001122
1681 016112 017737 163734 001126
1682 016120 042737 175777 001126
1683 016126 023737 001124 001126
1684 016134 001401
1685 016136 104043
1686 016140 162737 002000 001124 2\$:
1687 016146 100404
1688 016150 042737 000004 031076
1689 016156 000747

MOV CONS,\$BDADR ;SET UP REG ADRS 22
MOV @CONS,\$BDDAT ;READ L.P. SWITCH INTR ENABLED REG
BIC #175777,\$BDDAT ;SAVE ONLY L.P. SWITCH INTR ENABLED BIT
CMP \$GDDAT,\$BDDAT ;IS IT CORRECT?
BEQ 2\$;BR IF OK
ERROR 43 ;CONSOLE 0 L.P. SW INTR ENABLE INSTR ER
SUB #2000,\$GDDAT ;GO TO RESET STATE
BMI TST125 ;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
BIC #4,\$BUFFER ;CLR THE ENABLED BIT AT INSTR LOC
BR 1\$;GO TRY RESET STATE

1690
1691
(3)
(3)

*TEST 125 TEST THAT L.P. SWITCH INTR ENABLED DOES NOT RESET WHEN ENA=0

(2) 016160 000004
(2) 016162 012737 000125 001206
1692 016170 012777 020000 163664
1693 016176 012737 002000 001124
1694 016204 012737 164014 031076
1695 016212 012777 031076 163610
1696 016220 012737 164000 031100
1697 016226 005277 163576
1698 016232 013737 002052 001122
1699 016240 032777 002000 163604
1700 016246 001003
1701 016250 005037 001126
1702 016254 104043

TST125: SCOPE
MOV #125,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
MOV #2000,\$GDDAT ;EXPECT L.P. SWITCH INTR ENABLED TO BE SET
MOV #164014,\$BUFFER ;SET UP INSTR LOC
MOV #BUFFER,@DPC ;START
MOV #164000,\$BUFFER+2 ;SET UP INSTR WITH ENA=0
INC @DPC ;ADVANCE TO NEXT INSTR
MOV CONS,\$BDADR ;SET UP REG ADRS 22
BIT #2000,@CONS ;IS L.P. SWITCH INTR ENABLED STILL SET?
BNE TST126 ;ADVANCE TO NEXT TEST IF STILL SET
CLR \$BDDAT ;INDICATE IT WAS CLEARED
ERROR 43 ;CONSOLE 0 L.P. SW CHANGE INTR ENABLE INSTR ER

1703
1704
(3)
(3)

*TEST 126 TEST THAT NOP INSTR CAN SET & RESET L.P. SWITCH INTR ENABLED-CONSOLE 1

(2) 016256 000004
(2) 016260 012737 000126 001206
1705 016266 012777 020000 163566
1706 016274 012737 016316 001110
1707 016302 012737 000020 001124
1708 016310 012737 164414 031076
1709 016316 012777 031076 163504 1\$:
1710 016324 013737 002052 001122
1711 016332 017737 163514 001126
1712 016340 042737 177757 001126
1713 016346 023737 001124 001126
1714 016354 001401
1715 016356 104043
1716 016360 162737 000020 001124 2\$:
1717 016366 100404
1718 016370 042737 000004 031076
1719 016376 000747

TST126: SCOPE
MOV #126,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
MOV #1\$,\$LPERR ;SET UP SCOPE LOOP ADRS
MOV #20,\$GDDAT ;EXPECT L.P. SWITCH INTR ENABLED BIT
MOV #164414,\$BUFFER ;SET UP INSTR
MOV #BUFFER,@DPC ;START
MOV CONS,\$BDADR ;SET UP REG ADRS 22
MOV @CONS,\$BDDAT ;READ L.P. SWITCH INTR ENABLED REG
BIC #177757,\$BDDAT ;SAVE ONLY L.P. SWITCH INTR ENABLED BIT
CMP \$GDDAT,\$BDDAT ;IS IT CORRECT?
BEQ 2\$;BR IF OK
ERROR 43 ;CONSOLE 1 L.P. SW INTR ENABLE INSTR ER
SUB #20,\$GDDAT ;GO TO RESET STATE
BMI TST127 ;ADVANCE TO NEXT TEST IF BOTH STATES TESTED
BIC #4,\$BUFFER ;CLR THE ENABLED BIT AT INSTR LOC
BR 1\$;GO TRY RESET STATE

1720
1721
(3)
(3)

*TEST 127 TEST THAT L.P. SWITCH INTR ENABLED DOES NOT RESET WHEN ENA=0

(2) 016400 000004
(2) 016402 012737 000127 001206
1722 016410 012777 020000 163444
1723 016416 012737 000020 001124

TST127: SCOPE
MOV #127,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
MOV #20,\$GDDAT ;EXPECT L.P. SW INTR ENABLED TO BE SET

1724	016424	012737	164414	031076	MOV	#164414,BUFFER	;SET UP INSTR LOC
1725	016432	012777	031076	163370	MOV	#BUFFER,@DPC	;START
1726	016440	012737	164400	031100	MOV	#164400,BUFFER+2	;SET UP INSTR LOC WITH ENA=0
1727	016446	005277	163356		INC	@DPC	;ADVANCE TO NEXT INSTR
1728	016452	013737	002052	001122	MOV	CONS,\$BDADR	;SET UP REG ADRS 22
1729	016460	032777	000020	163364	BIT	#20,@CONS	;IS L.P. SW INTR ENABLED STILL SET?
1730	016466	001003			BNE	TST130	;ADVANCE TO NEXT TEST IF STILL SET
1731	016470	005037	001126		CLR	\$BDDAT	;INDICATE IT WAS CLEARED
1732	016474	104043			ERROR	43	;CONSOLE 1 L.P. SW CHANGE INTR ENABLE INSTR ER

1733
1734
1735

*TEST 130 TEST THAT RESET WILL CLEAR L.P. INTR ENABLES & COLOR

(3)				
(3)				
(2)	016476	000004		
(1)	016500	012737	000040	001166
(2)	016506	012737	000130	001206
1736	016514	013737	002052	001122
1737	016522	012777	020000	163332
1738	016530	012737	164274	031076
1739	016536	012777	031076	163264
1740	016544	012737	164774	031100
1741	016552	005277	163252	
1742	016556	012737	175600	031102
1743	016564	005277	163240	
1744	016570	012737	100000	001124
1745	016576	000005		
1746	016600	017737	163246	001126
1747	016606	023737	001124	001126
1748	016614	001401		
1749	016616	104032		

TST130: SCOPE
MOV #40,\$TIMES ;;DO 40 ITERATIONS
MOV #130,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV CONS,\$BDADR ;;SET UP REG ADRS 22
MOV #BIT13,@STKPT ;;SET MAINT SW 2
MOV #164274,BUFFER ;;SET UP INSTR FOR CONSOLE 0 L.P. INTR ENAB
MOV #BUFFER,@DPC ;;START
MOV #164774,BUFFER+2 ;;CONS 1 L.P. INTR ENA
INC @DPC ;;RESUME
MOV #175600,BUFFER+4 ;;COLOR LEVEL
INC @DPC ;;RESUME
MOV #BIT15,\$GDDAT ;;EXPECT INTENSITY ENABLE CONSOLE 0 ONLY
RESET ;;CLR FLAGS
MOV @CONS,\$BDDAT ;;READ FLAGS
CMP \$GDDAT,\$BDDAT ;;IS REG 22 CORRECT?
BEQ TST131 ;;NEXT TEST IF SO
ERROR 32 ;;RESET FAILED TO SET UP REG 22

1750
1751
(3)

*TEST 131 JUMP TEST TO LOC. 25252

(3)				
(2)	016620	000004		
(2)	016622	012737	000131	001206
1752	016630	012777	020000	163224
1753	016636	012777	160000	163134
1754	016644	012737	025252	001124
1755	016652	013777	002000	163150
1756	016660	013737	001124	031100
1757	016666	005277	163136	
1758	016672	013737	002030	001122
1759	016700	017737	163124	001126
1760	016706	023737	001124	001126
1761	016714	001401		
1762	016716	104010		

TST131: SCOPE
MOV #131,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;;SET MAINT SWITCH #2
MOV #160000,@DBUF ;;MOVE DJUMP INTO THE BUFFER
MOV #25252,\$GDDAT ;;MOVE 25252 INTO THE NEXT LOC.
MOV DBUF,@DPC ;;START THE DISPLAY
MOV \$GDDAT,BUFFER+2 ;;LD ADRS
INC @DPC ;;SINGLE STEP THE DISPLAY
MOV DPC,\$BDADR ;;SET UP REG ADRS 00
MOV @DPC,\$BDDAT ;;READ THE DISPLAY P.C.
CMP \$GDDAT,\$BDDAT ;;COMPARE EXPCT TO RCVD
BEQ TST132 ;;BR IF EQUAL
ERROR 10 ;;DJUMP FAILED TO LOAD THE NEW
;;DISPLAY P.C. PROPERLY

1763
1764
1765
(3)

*TEST 132 JUMP TEST TO LOC. 12524

(3)				
(2)	016720	000004		
(2)	016722	012737	000132	001206
1766	016730	012777	020000	163124

TST132: SCOPE
MOV #132,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;;SET MAINT SWITCH #2

1767	016736	012777	160000	163034	MOV	#160000,@DBUF	:MOVE DJUMP INTO THE BUFFER
1768	016744	012737	012524	001124	MOV	#12524,\$GDDAT	:MOVE 12524 INTO THE NEXT LOC.
1769	016752	013777	002000	163050	MOV	DBUF,@DPC	:START THE DISPLAY
1770	016760	013737	001124	031100	MOV	\$GDDAT,BUFFER+2	:LD ADRS
1771	016766	005277	163036		INC	@DPC	:SINGLE STEP THE DISPLAY
1772	016772	013737	002030	001122	MOV	DPC,\$BDADR	:SET UP REG ADRS 00
1773	017000	017737	163024	001126	MOV	@DPC,\$BDDAT	:READ THE DISPLAY P.C.
1774	017006	023737	001124	001126	CMP	\$GDDAT,\$BDDAT	:COMPARE EXPT TO RCVD
1775	017014	001401			BEQ	TST133	::BR IF EQUAL
1776	017016	104010			ERROR	10	:DJUMP FAILED TO LOAD THE NEW
1777							:DISPLAY P.C. PROPERLY
1778							
1779							

::*****
:*TEST 133 JUMP TO EACH EXISTING LOCATION IN LOWER BANK
:*****

(3)					TST133: SCOPE		
(2)	017020	000004			MOV	#20,\$TIMES	::DO 20 ITERATIONS
(1)	017022	012737	000020	001166	MOV	#133,\$TESTN	::SET TEST NUMBER IN APT MAIL BOX
(2)	017030	012737	000133	001206	MOV	\$LSTAD,\$GDDAT	:LOAD JUMP DESTINATION
1780	017036	013737	026576	001124	MOV	#BIT13,@STKPT	:SET MAINT SWITCH #2
1781	017044	012777	020000	163010	MOV	#160000,BUFFER	:LOAD 'DISPLAY JUMP' INST
1782	017052	012737	160000	031076	1\$: MOV	#BUFFER,@DPC	:START DISPLAY
1783	017060	012777	031076	162742	MOV	\$GDDAT,BUFFER+2	:SET UP ADRS
1784	017066	013737	001124	031100	INC	@DPC	:SINGLE STEP THE DISPLAY
1785	017074	005277	162730		MOV	DPC,\$BDADR	:SET UP REG ARS 00
1786	017100	013737	002030	001122	MOV	@DPC,\$BDDAT	:READ DESTINATION D.P.C.
1787	017106	017737	162716	001126	CMP	\$GDDAT,\$BDDAT	:TEST IF EXPECTED
1788	017114	023737	001124	001126	BEQ	2\$::BR IF SAME
1789	017122	001401			ERROR	10	:JUMP TO LOCATION 'N' FAILED
1790	017124	104010					:N=ADDRESS IN \$GDDAT
1791							:ADJUST EXPECTED ADDRESS
1792	017126	162737	000002	001124	2\$: SUB	#2,\$GDDAT	:BR IF MORE LOCATIONS TO TEST
1793	017134	103351			BCC	1\$	
1794							
1795							

::*****
:*TEST 134 TEST 'JUMP RELATIVE' INSTR (BIT 8=0) AT LOC 'BUFFER' & VERIFY NEW DPC AD
:*****

(3)					TST134: SCOPE		
(2)	017136	000004			MOV	#134,\$TESTN	::SET TEST NUMBER IN APT MAIL BOX
(2)	017140	012737	000134	001206	MOV	#1\$,\$LPERR	:SET UP SCOPE LOOP ADRS
1796	017146	012737	017170	001110	MOV	#BIT13,@STKPT	:SET MAINT SWITCH #2
1797	017154	012777	020000	162700	MOV	#400,R0	:SET UP MSB FIRST
1798	017162	012700	070400		BR	2\$:GO CALCULATE ADRS
1799	017166	000423			1\$: MOV	#161000,BUFFER	:SET UP JUMP RELATIVE INSTR
1800	017170	012737	161000	031076	BIS	R0,BUFFER	:SET UP RELATIVE ADRS PATTERN
1801	017176	050037	031076		MOV	#BUFFER,@DPC	:START
1802	017202	012777	031076	162620	MOV	DPC,\$BDADR	:SET UP REG ADRS 00
1803	017210	013737	002030	001122	MOV	@DPC,\$BDDAT	:READ NEW DPC ADRS
1804	017216	017737	162606	001126	CMP	\$GDDAT,\$BDDAT	:DOES IT EQUAL THE CALCULATED ADRS?
1805	017224	023737	001124	001126	BEQ	2\$:BR IF OK
1806	017232	001401			ERROR	11	:JUMP RELATIVE POSITIVE FAILED
1807	017234	104011			2\$: ASR	R0	:SHIFT RIGHT TO NEXT BIT
1808	017236	006200			BCS	TST135	::ADVANCE TO NEXT TEST IF DONE
1809	017240	103410			MOV	R0,\$GDDAT	:SET UP OFFSET VALUE
1810	017242	010037	001124		ASL	\$GDDAT	:MAKE INTO WORD OFFSET
1811	017246	006337	001124		ADD	#BUFFER+2,\$GDDAT	:ADD IN ADRS OF INSTR PLUS 2
1812	017252	062737	031100	001124	BR	1\$:TRY THIS RELATIVE ADRS
1813	017260	000743					

1814
1815
(3)
(3)
(2) 017262 000004
(2) 017264 012737 000135 001206
1816 017272 012737 017314 001110
1817 017300 012777 020000 162554
1818 017306 012700 000400
1819 017312 000423
1820 017314 012737 161400 031076 1\$:
1821 017322 050037 031076
1822 017326 012777 031076 162474
1823 017334 013737 002030 001122
1824 017342 017737 162462 001126
1825 017350 023737 001124 001126
1826 017356 001401
1827 017360 104011
1828 017362 006200 2\$:
1829 017364 103413
1830 017366 012737 031100 001124
1831 017374 010001
1832 017376 005401
1833 017400 006301
1834 017402 042701 177000
1835 017406 160137 001124
1836 017412 000740

```
*****  
*TEST 135 TEST 'JUMP RELATIVE' INSTR (BIT 8=1) AT LOC 'BUFFER' & VERIFY NEW DPC  
*****  
TST135: SCOPE  
MOV #135,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX  
MOV #1$, $LPERR ;;SET UP SCOPE LOOP ADRS  
MOV #BIT13,@STKPT ;;SET MAINT SWITCH #2  
MOV #400,R0 ;;SET UP MSB FIRST  
BR 2$ ;;GO CALCULATE ADRS  
1$: MOV #161400,BUFFER ;;SET UP JUMP RELATIVE INSTR  
BIS R0,BUFFER ;;SET UP RELATIVE ADRS PATTERN  
MOV #BUFFER,@DPC ;;START  
MOV DPC,$BDADR ;;SET UP REG ADRS 00  
MOV @DPC,$BDDAT ;;READ NEW DPC ADRS  
CMP $GDDAT,$BDDAT ;;DOES IT EQUAL THE CALCULATED VALUE?  
BEQ 2$ ;;BR IF OK  
ERROR 11 ;;JUMP RELATIVE NEGATIVE FAILED  
2$: ASR R0 ;;SHIFT RIGHT TO NEXT BIT  
BCS TST136 ;;ADVANCE TO NEXT TEST IF DONE  
MOV #BUFFER+2,$GDDAT ;;SET UP CURRENT ADRS  
MOV R0,R1 ;;GET RELATIVE ADRS VALUE  
NEG R1 ;;MAKE NEG  
ASL R1 ;;CORRECT FOR WORD OFFSET  
BIC #177000,R1 ;;GET RID OF MSBS  
SUB R1,$GDDAT ;;SUB OFFSET FROM DPC VALUE  
BR 1$ ;;TRY NEXT RELATIVE ADRS
```

1837
1838
(3)
(3)
(2) 017414 000004
(2) 017416 012737 000136 001206
1839 017424 012777 020000 162430
1840 017432 012737 160000 031076
1841 017440 005037 001124
1842 017444 012737 000001 031100
1843 017452 012777 031076 162350 1\$:
1844 017460 012737 017452 001110
1845 017466 005277 162336
1846 017472 013737 002030 001122
1847 017500 000240
1848 017502 017737 162322 001126
1849 017510 023737 001124 001126
1850 017516 001401
1851 017520 104010
1852 017522 006337 031100 2\$:
1853 017526 103404
1854 017530 013737 031100 001124
1855 017536 000745

```
*****  
*TEST 136 TEST THAT JUMP ABS INSTR WILL SETUP DPC WITH A FLOATING ONE  
*****  
TST136: SCOPE  
MOV #136,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX  
MOV #BIT13,@STKPT ;;SET MAINT SWITCH #2  
MOV #160000,BUFFER ;;SET UP JUMP ABS INSTRUCTION AT 'BUFFER'  
CLR $GDDAT ;;EXPECT ZEROS FIRST TIME  
1$: MOV #1,BUFFER+2 ;;SET UP ABS ADRS TO #1  
MOV #BUFFER,@DPC ;;START  
MOV #1$, $LPERR ;;SET UP SCOPE LOOP ADRS  
INC @DPC ;;PICK UP ABS ADRS ** B **  
MOV DPC,$BDADR ;;SET UP REG ADRS 00 ** B **  
NOP ;;ALLOW TIME FOR 2ND NPR  
MOV @DPC,$BDDAT ;;GET ADRS JUMPED TO  
CMP $GDDAT,$BDDAT ;;IS IT CORRECT?  
BEQ 2$ ;;BR IF OK  
ERROR 10 ;;JUMP ABSOLUTE FAILED (FLOAT A 1 ADDRESS)  
2$: ASL BUFFER+2 ;;ADVANCE ONE BIT LEFT  
BCS TST137 ;;ADVANCE TO NEXT TEST IF ALL BITS TESTED  
MOV BUFFER+2,$GDDAT ;;SET UP EXPECTED ADRS  
BR 1$ ;;TRY NEXT BIT
```

1856
1857
(3)
(3)
(2) 017540 000004
(2) 017542 012737 000137 001206

```
*****  
*TEST 137 TEST THAT JUMP ABS INSTR WILL SETUP DPC WITH A FLOATING ZERO  
*****  
TST137: SCOPE  
MOV #137,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
```

1858 017550 012737 017600 001110
1859 017556 012777 020000 162276
1860 017564 012737 160000 031076
1861 017572 012737 177774 001124
1862 017600 012777 031076 162222
1863 017606 013737 001124 031100
1864 017614 005277 162210
1865 017620 013737 002030 001122
1866 017626 017737 162176 001126
1867 017634 023737 001124 001126
1868 017642 001401
1869 017644 104010
1870 017646 006337 001124
1871 017652 103004
1872 017654 062737 000002 001124
1873 017662 000746
1874
1875

MOV #1\$, \$LPERR ;SET UP SCOPE LOOP ADRS
MOV #BIT13, @STKPT ;SET MAINT SWITCH #2
MOV #160000, BUFFER ;SET UP JUMP ABS INSTR AT 'BUFFER'
MOV #177774, \$GDDAT ;SET LSB TO ZERO
1\$: MOV #BUFFER, @DPC ;START
MOV \$GDDAT, BUFFER+2 ;SET UP ADRS
INC @DPC ;PICK UP ABS ADRS
MOV DPC, \$BDADR ;SET UP REG ADRS 00
MOV @DPC, \$BDDAT ;GET ADRS JUMPED TO
CMP \$GDDAT, \$BDDAT ;IS IT CORRECT?
BEQ 2\$;BR IF OK
ERROR 10 ;JUMP ABSOLUTE FAILED (TO A FLOATING ZERO)
2\$: ASL \$GDDAT ;SHIFT ZERO BIT LEFT
BCC TST140 ;ADVANCE TO NEXT TEST IF ALL BITS TESTED
ADD #2, \$GDDAT ;REPLACE LSB
BR 1\$;TRY NEXT BIT

*TEST 140 TEST THAT THE 'JSR ABSOLUTE' INSTR WILL SET 'JSR' BIT IN REG 32

(3)
(3)
(2) 017664 000004
(2) 017666 012737 000140 001206
1876 017674 012777 020000 162160
1877 017702 012737 162000 031076
1878 017710 005037 031100
1879 017714 012737 001000 001124
1880 017722 012777 031076 162100
1881 017730 013737 002062 001122
1882 017736 032777 001000 162116
1883 017744 001003
1884 017746 005037 001126
1885 017752 104012
1886
1887

TST140: SCOPE
MOV #140, \$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13, @STKPT ;SET MAINT SWITCH #2
MOV #162000, BUFFER ;SET UP JSR ABS INSTR
CLR BUFFER+2 ;CLR ABS ADRS LOC
MOV #BIT9, \$GDDAT ;EXPECT JSR BIT
MOV #BUFFER, @DPC ;START
MOV STKPT, \$BDADR ;SET UP REG ARS 32
BIT #BIT9, @STKPT ;IS JSR BIT SET?
BNE TST141 ;ADVANCE TO NEXT TEST IF OK
CLR \$BDDAT ;SHOW THAT IT WAS ZERO
ERROR 12 ;CALL ERROR ROUTINE USING MSG 23

*TEST 141 TEST 'JSR RELATIVE' INSTR (BIT 8=0) AT LOC 'BUFFER' & VERIFY NEW DPC ADR

(3)
(3)
(2) 017754 000004
(2) 017756 012737 000141 001206
1888 017764 012777 020000 162070
1889 017772 012737 020006 001110
1890 020000 012700 000400
1891 020004 000426
1892 020006 052777 000040 162046
1893 020014 012737 163000 031076
1894 020022 050037 031076
1895 020026 012777 031076 161774
1896 020034 013737 002030 001122
1897 020042 017737 161762 001126
1898 020050 023737 001124 001126
1899 020056 001401
1900 020060 104013
1901 020062 006200
1902 020064 103410
1903 020066 010037 001124
1904 020072 006337 001124
1905 020076 062737 031100 001124

TST141: SCOPE
MOV #141, \$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13, @STKPT ;SET MAINT SWITCH #2
MOV #1\$, \$LPERR ;SET UP SCOPE LOOP ADRS
MOV #400, R0 ;SET UP MSB FIRST
BR 2\$;GO CALCULATE ADRS
1\$: BIS #40, @STKPT ;RESET STK PTR
MOV #163000, BUFFER ;SET UP JSR RELATIVE INSTR
BIS R0, BUFFER ;SET UP RELATIVE ADRS PATTERN
MOV #BUFFER, @DPC ;START
MOV DPC, \$BDADR ;SET UP REG ADRS 00
MOV @DPC, \$BDDAT ;READ NEW DPC ADRS
CMP \$GDDAT, \$BDDAT ;DOES IT EQUAL THE CALCULATED ADRS?
BEQ 2\$;BR IF OK
ERROR 13 ;JSR RELATIVE 'POSITIVE' FAILED
2\$: ASR R0 ;SHIFT RIGHT TO NEXT BIT
BCS TST142 ;ADVANCE TO NEXT TEST IF DONE
MOV R0, \$GDDAT ;SET UP OFFSET VALUE
ASL \$GDDAT ;MAKE INTO WORD OFFSET
ADD #BUFFER+2, \$GDDAT ;ADD IN ADRS OF INSTR PLUS 2

1906 020104 000740
1907
1908
(3)
(3)
(2) 020106 000004
(2) 020110 012737 000142 001206
1909 020116 012777 020000 161736
1910 020124 012737 020140 001110
1911 020132 012700 000400
1912 020136 000426
1913 020140 052777 000040 161714
1914 020146 012737 163400 031076
1915 020154 050037 031076
1916 020160 012777 031076 161642
1917 020166 013737 002030 001122
1918 020174 017737 161630 001126
1919 020202 023737 001124 001126
1920 020210 001401
1921 020212 104013
1922 020214 006200
1923 020216 103413
1924 020220 012737 031100 001124
1925 020226 010001
1926 020230 005401
1927 020232 006301
1928 020234 042701 177000
1929 020240 160137 001124
1930 020244 000735
1931
1932
(3)
(3)
(2) 020246 000004
(2) 020250 012737 000143 001206
1933 020256 012777 020000 161576
1934 020264 012737 162000 031076
1935 020272 005037 001124
1936 020276 012737 000001 031100
1937 020304 052777 000040 161550
1938 020312 012777 031076 161510
1939 020320 012737 020304 001110
1940 020326 005277 161476
1941 020332 013737 002030 001122
1942 020340 017737 161464 001126
1943 020346 023737 001124 001126
1944 020354 001401
1945 020356 104012
1946 020360 006337 031100
1947 020364 103404
1948 020366 013737 031100 001124
1949 020374 000743
1950
1951
(3)
(3)

```
BR 1$ ;TRY THIS RELATIVE ADRS  
*****  
*TEST 142 TEST 'JSR RELATIVE' INSTR (BIT 8=1) AT LOC 'BUFFER' & VERIFY NEW DPC ADR  
*****  
TST142: SCOPE  
MOV #142,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX  
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2  
MOV #1$,$LPERR ;SET UP SCOPE LOOP ADRS  
MOV #400,R0 ;SET UP MSB FIRST  
BR 2$ ;GO CALCULATE ADRS  
1$: BIS #40,@STKPT ;RESET STK PTR  
MOV #163400,BUFFER ;SET UP JSR RELATIVE INSTR  
BIS R0,BUFFER ;SET UP RELATIVE ADRS PATTERN  
MOV #BUFFER,@DPC ;START  
MOV DPC,$BDADR ;SET UP REG ADRS 00  
MOV @DPC,$BDDAT ;READ NEW DPC ADRS  
CMP $GDDAT,$BDDAT ;DOES IT EQUAL THE CALCULATED VALUE?  
BEQ 2$ ;BR IF OK  
ERROR 13 ;JSR RELATIVE NEGATIVE TO ADDRESS FAILED  
2$: ASR R0 ;SHIFT RIGHT TO NEXT BIT  
BCS TST143 ;ADVANCE TO NEXT TEST IF DONE  
MOV #BUFFER+2,$GDDAT ;SET UP CURRENT ADRS  
MOV R0,R1 ;GET RELATIVE ADRS VALUE  
NEG R1 ;MAKE NEG  
ASL R1 ;CORRECT FOR WORD OFFSET  
BIC #177000,R1 ;GET RID OF MSBS  
SUB R1,$GDDAT ;SUB OFFSET FROM DPC VALUE  
BR 1$ ;TRY NEXT RELATIVE ADRS  
*****  
*TEST 143 TEST THAT 'JSR ABS' INSTR WILL SET UP DPC WITH A FLOATING ONE  
*****  
TST143: SCOPE  
MOV #143,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX  
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2  
MOV #162000,BUFFER ;SET UP JSR ABSOLUTE INSTR AT 'BUFFER'  
CLR $GDDAT ;EXPECT ZEROS FIRST TIME  
MOV #1,BUFFER+2 ;SET UP ABS ADRS TO 1  
1$: BIS #40,@STKPT ;RESET STK PTR  
MOV #BUFFER,@DPC ;START  
MOV #1$,$LPERR ;SET UP SCOPE LOOP ADRS  
INC @DPC ;ADVANCE TO ABS ADRS  
MOV DPC,$BDADR ;SET UP REG ADRS 00  
MOV @DPC,$BDDAT ;GET ADRS JUMPED TO  
CMP $GDDAT,$BDDAT ;IS IT CORRECT?  
BEQ 2$ ;BR IF OK  
ERROR 12 ;JSR ABSOLUTE FAILED (TO A FLOATING ONE)  
2$: ASL BUFFER+2 ;ADVANCE ONE BIT LEFT  
BCS TST144 ;ADVANCE TO NEXT TEST IF ALL BITS TESTED  
MOV BUFFER+2,$GDDAT ;SET UP EXPECTED ADRS  
BR 1$ ;TRY NEXT BIT  
*****  
*TEST 144 TEST THAT JSR ABS INSTR WILL SET UP DPC WITH A FLOATING ZERO  
*****
```

(2) 020376 000004
(2) 020400 012737 000144 001206
1952 020406 012777 020000 161446
1953 020414 012737 020436 001110
1954 020422 012737 162000 031076
1955 020430 012737 177774 001124
1956 020436 052777 000040 161416
1957 020444 012777 031076 161356
1958 020452 013737 001124 031100
1959 020460 005277 161344
1960 020464 013737 002030 001122
1961 020472 017737 161332 001126
1962 020500 023737 001124 001126
1963 020506 001401
1964 020510 104012
1965 020512 006337 001124
1966 020516 103004
1967 020520 062737 000002 001124
1968 020526 000743

TST144: SCOPE
MOV #144,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
MOV #1\$, \$LPERR ;SET UP SCOPE LOOP ADRS
MOV #162000,BUFFER ;SET UP JSR ABSOLUTE AT 'BUFFER'
MOV #177774,\$GDDAT ;SET LSB TO ZERO
1\$: BIS #40,@STKPT ;RESET STK PTR
MOV #BUFFER,@DPC ;START
MOV \$GDDAT,BUFFER+2 ;SET UP ADRS
INC @DPC ;ADVANCE TO ABS ADRS
MOV DPC,\$BDADR ;SET UP REG ADRS 00
MOV @DPC,\$BDDAT ;GET ADRS JUMPED TO
CMP \$GDDAT,\$BDDAT ;IS IT CORRECT?
BEQ 2\$;BR IF OK
ERROR 12 ;JSR ABSOLUTE FAILED (TO A FLOATING ZERO)
2\$: ASL \$GDDAT ;SHIFT ZERO BIT LEFT
BCC TST145 ;ADVANCE TO NEXT TEST IF ALL BITS TESTED
ADD #2,\$GDDAT ;REPLACE LSB
BR 1\$;TRY NEXT BIT

*TEST 145 TEST THAT GRAPHPLT X CAN LOAD X POSITION REGISTER WITH #1252

(3)
(3)
(2) 020530 000004
(2) 020532 012737 000145 001206
1971 020540 013737 002034 001122
1972 020546 012777 020000 161306
1973 020554 012700 031076
1974 020560 012720 122000
1975 020564 012720 001252
1976 020570 013777 002000 161232
1977 020576 012737 001252 001124
1978 020604 005277 161220
1979 020610 004737 024514
1980 020614 017737 161214 001126
1981 020622 023737 001124 001126
1982 020630 001401
1983 020632 104014

TST145: SCOPE
MOV #145,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV XPOS,\$BDADR ;SET UP REG ADRS 04
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
MOV #BUFFER,R0 ;LOAD BUFFER POINTER
MOV #122000,(R0)+ ;LOW INTENSITY - SET GRAPH PLOT X MODE
MOV #1252,(R0)+ ;SET X POSITION
MOV DBUF,@DPC ;LOAD DISPLAY P.C.
MOV #1252,\$GDDAT ;LOAD EXPECTED
INC @DPC ;SINGLE STEP THE DISPLAY
JSR PC,DLAY ;STALL
MOV @XPOS,\$BDDAT ;READ X POSITION
CMP \$GDDAT,\$BDDAT ;COMPARE EXPCT TO RCVD
BEQ TST146 ;BR IF EQUAL
ERROR 14 ;X POSITION REGISTER FAILED TO LOAD
;PROPERLY USING GRAPH PLOT X MODE

*TEST 146 TEST THAT GRAPHPLT X CAN LOAD X POSITION REGISTER WITH #525

(3)
(3)
(2) 020634 000004
(2) 020636 012737 000146 001206
1987 020644 013737 002034 001122
1988 020652 012777 020000 161202
1989 020660 012700 031076
1990 020664 012720 122000
1991 020670 012720 000525
1992 020674 012777 031076 161126
1993 020702 012737 000525 001124
1994 020710 005277 161114
1995 020714 004737 024514
1996 020720 017737 161110 001126
1997 020726 023737 001124 001126

TST146: SCOPE
MOV #146,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV XPOS,\$BDADR ;SET UP REG ADRS 04
MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
MOV #BUFFER,R0 ;LOAD BUFFER POINTER
MOV #122000,(R0)+ ;LOW INTENSITY - SET GRAPH PLOT X MODE
MOV #525,(R0)+ ;SET X POSITION
MOV #BUFFER,@DPC ;STAR DISPLAY
MOV #525,\$GDDAT ;LOAD EXPECTED
INC @DPC ;SINGLE STEP THE DISPLAY
JSR PC,DLAY ;STALL
MOV @XPOS,\$BDDAT ;READ X POSITION
CMP \$GDDAT,\$BDDAT ;COMPARE EXPCT TO RCVD

1998 020734 001401 BEQ TST147 ;;BR IF EQUAL
1999 020736 104014 ERROR 14 ;X POSITION REGISTER FAILED TO LOAD
2000 ;PROPERLY USING GRAPH PLOT X MODE
2001
2002

2003 (3) *****
(3) *TEST 147 TEST THAT GRAPH PLOT Y CAN LOAD Y POSITION REGISTER WITH #1252

(2) 020740 000004 TST147: SCOPE
(2) 020742 012737 000147 001206 MOV #147,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
2003 020750 013737 002036 001122 MOV YPOS,\$BDADR ;SET UP REG ADRS 06
2004 020756 012777 020000 161076 MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
2005 020764 012700 031076 MOV #BUFFER,R0 ;LOAD BUFFER POINTER
2006 020770 012720 126000 MOV #126000,(R0)+ ;LOW INTENSITY - SET GRAPH PLOT Y
2007 020774 012720 001252 MOV #1252,(R0)+ ;SET Y POSITION
2008 021000 013777 002000 161022 MOV DBUF,@DPC ;LOAD THE DISPLAY P.C.
2009 021006 012737 001252 001124 MOV #1252,\$GDDAT ;LOAD EXPECTED
2010 021014 005277 161010 INC @DPC ;SINGLE STEP THE DISPLAY
2011 021020 004737 024514 JSR PC,DLAY ;STALL
2012 021024 017737 161006 001126 MOV @YPOS,\$BDDAT ;READ Y POSITION
2013 021032 023737 001124 001126 CMP \$GDDAT,\$BDDAT ;COMPARE EXPCT TO RCVD
2014 021040 001401 BEQ TST150 ;;BR IF EQUAL
2015 021042 104015 ERROR 15 ;Y POSITION REGISTER FAILED TO LOAD
2016 ;PROPERLY USING GRAPH PLOT Y MODE
2017

2018 (3) *****
(3) *TEST 150 TEST THAT GRAPH PLOT Y CAN LOAD Y POSITION REGISTER WITH #525

(2) 021044 000004 TST150: SCOPE
(2) 021046 012737 000150 001206 MOV #150,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
2019 021054 013737 002036 001122 MOV YPOS,\$BDADR ;SET UP REG ADRS 06
2020 021062 012777 020000 160772 MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
2021 021070 012700 031076 MOV #BUFFER,R0 ;LOAD BUFFER POINTER
2022 021074 012720 126000 MOV #126000,(R0)+ ;LOW INTENSITY - SET GRAPH PLOT Y MODE
2023 021100 012720 000525 MOV #525,(R0)+ ;SET Y POSITION
2024 021104 013777 002000 160716 MOV DBUF,@DPC ;LOAD THE DISPLAY P.C.
2025 021112 012737 000525 001124 MOV #525,\$GDDAT ;LD EXPECTED
2026 021120 005277 160704 INC @DPC ;SINGLE STEP THE DISPLAY
2027 021124 004737 024514 JSR PC,DLAY ;STALL
2028 021130 017737 160702 001126 MOV @YPOS,\$BDDAT ;READ Y POSITION
2029 021136 023737 001124 001126 CMP \$GDDAT,\$BDDAT ;COMPARE EXPCT TO RCVD
2030 021144 001401 BEQ TST151 ;;BR IF EQUAL
2031 021146 104015 ERROR 15 ;Y POSITION REGISTER FAILED TO LOAD
2032 ;PROPERLY USING GRAPH PLOT Y MODE
2033

2034 (3) *****
2035 (3) *TEST 151 TEST FOR PROPER SELECTION OF X AND Y REGISTERS

(2) 021150 000004 TST151: SCOPE
(2) 021152 012737 000151 001206 MOV #151,\$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
2036 021160 013737 002034 001122 MOV XPOS,\$BDADR ;SET UP REG ADRS 04
2037 021166 012777 020000 160666 MOV #BIT13,@STKPT ;SET MAINT SWITCH #2
2038 021174 012700 031076 MOV #BUFFER,R0 ;LOAD BUFFER POINTER
2039 021200 012720 122000 MOV #122000,(R0)+ ;LOW INTENSITY - SET GRAPH PLOT X MODE
2040 021204 012720 001234 MOV #1234,(R0)+ ;SET X POSITION
2041 021210 012720 126000 MOV #126000,(R0)+ ;SET GRAPH PLOT Y MODE

```

2042 021214 012720 001432      MOV      #1432,(R0)+      ;SET Y POSITION
2043 021220 013777 002000 160602  MOV      DBUF,@DPC      ;LOAD THE DISPLAY P.C.
2044 021226 012737 001234 001124  MOV      #1234,$GDDAT   ;LD EXPECTED
2045 021234 005277 160570      INC      @DPC           ;SINGLE STEP THE DISPLAY
2046 021240 004737 024514      JSR      PC,DLAY        ;STALL
2047 021244 017737 160564 001126  MOV      @XPOS,$BDDAT   ;READ X POSITION
2048 021252 023737 001124 001126  CMP      $GDDAT,$BDDAT  ;COMPARE EXPCT TO RCVD
2049 021260 001402      BEQ      1$
2050 021262 104014      ERROR   14             ;GRAPHPLOT X MODE FAILED TO SELECT
2051 021264 000424      BR       TST152        ;;
2052
2053 021266      1$:
(1) 021266 005277 160536      INC      @DPC           ;SINGLE STEP THE DISPLAY
2054 021272 013737 002036 001122  MOV      YPOS,$BDADR    ;SET UP REG ADRS 06
2055 021300 005277 160524      INC      @DPC           ;SINGLE STEP THE DISPLAY
2056 021304 012737 001432 001124  MOV      #1432,$GDDAT   ;LOAD EXPECTED
2057 021312 004737 024514      JSR      PC,DLAY        ;STALL
2058 021316 017737 160514 001126  MOV      @YPOS,$BDDAT   ;READ Y POSITION
2059 021324 023737 001124 001126  CMP      $GDDAT,$BDDAT  ;COMPARE EXPCT TO RCVD
2060 021332 001401      BEQ      TST152        ;;BR IF EQUAL
2061 021334 104015      ERROR   15             ;Y POSITION REGISTER FAILED TO LOAD
2062                                     ;PROPERLY USING GRAPHPLOT Y MODE
2063
2064
(3)
(3)
(2) 021336 000004
(1) 021340 012737 000040 001166  MOV      #40,$TIMES     ;;DO 40 ITERATIONS
(2) 021346 012737 000152 001206  MOV      #152,$TESTN    ;;SET TEST NUMBER IN APT MAIL BOX
2065 021354 013737 002034 001122  MOV      XPOS,$BDADR    ;SET UP REG ADRS 04
2066 021362 012737 114000 031076  MOV      #114000,BUFFER ;LOAD 'POINT' INST.
2067 021370 012777 020000 160464  MOV      #BIT13,@STKPT  ;SET MAINT SWITCH #2
2068 021376 012777 031076 160424  MOV      #BUFFER,@DPC   ;START DISPLAY
2069 021404 012737 001777 031100  MOV      #1777,BUFFER+2 ;X=1777
2070 021412 005277 160412      INC      @DPC           ;SINGLE STEP THE DISPLAY
2071 021416 012737 001777 031102  MOV      #1777,BUFFER+4 ;Y=1777
2072 021424 005277 160400      INC      @DPC           ;SINGLE STEP THE DISPLAY
2073 021430 004737 024514      JSR      PC,DLAY        ;STALL
2074
2075 021434 000005      RESET          ;INITILIZE
2076 021436 005037 001124      CLR      $GDDAT        ;CLEAR EXPECTED
2077 021442 017737 160366 001126  MOV      @XPOS,$BDDAT   ;READ X POSITION
2078 021450 001401      BEQ      1$           ;;BR IF CLEARED
2079 021452 104032      ERROR   32           ;RESET FAILED TO CLEAR X POS. REGISTER
2080
2081 021454 013737 002036 001122 1$:  MOV      YPOS,$BDADR    ;SET UP REG ADRS 06
2082 021462 017737 160350 001126  MOV      @YPOS,$BDDAT   ;READ Y POSITION
2083 021470 001401      BEQ      TST153        ;;BR IF CLARED
2084 021472 104032      ERROR   32           ;RESET FAILED TO CLEAR Y POS. REGISTER
2085
(3)
(3)
(2) 021474 000004
(2) 021476 012737 000153 001206  MOV      #153,$TESTN    ;;SET TEST NUMBER IN APT MAIL BOX
2086 021504 012777 020000 160350  MOV      #BIT13,@STKPT  ;SET MAINT SW 2
2087 021512 012737 100000 001124  MOV      #100000,$GDDAT ;EXPECT DISPLAY BUSY INITIALLY
    
```

```

*****
*TEST 152      TEST THAT RESET CLEARS X AND Y POSITION REG.
*****
TST152: SCOPE
*****
*TEST 153      TEST THAT DISPLAY BUSY WILL SET & THEN CLR ON INTERNAL STOP
*****
TST153: SCOPE
    
```

2088	021520	012737	164000	031076	MOV	#164000,BUFFER	;SET UP A NOP INSTR
2089	021526	012737	172000	031100	MOV	#172000,BUFFER+2	;SET UP LD STATUS A INSTR WITH STOP
2090	021534	012777	031076	160266	MOV	#BUFFER,@DPC	;START
2091	021542	013737	002042	001122	MOV	SREG1,\$BDADR	;SET UP REG ADRS 12
2092	021550	005777	160266		TST	@SREG1	;SEE THAT DISPLAY BUSY IS SET
2093	021554	100403			BMI	1\$;BR IF SO
2094	021556	005037	001126		CLR	\$BDDAT	;INDICATE DISPLAY BUSY NOT SET
2095	021562	104044			ERROR	44	;DISPLAY BUSY NOT SET
2096	021564	005277	160240		1\$: INC	@DPC	;ADVANCE TO STOP INSTR
2097	021570	005037	001124		CLR	\$GDDAT	;EXPECTE ZERO
2098	021574	005777	160242		TST	@SREG1	;SEE THAT DISPLAY BUSY HAS CLEARED
2099	021600	100004			BPL	TST154	;GO TO NEXT TEST IF OK
2100	021602	012737	100000	001126	MOV	#100000,\$BDDAT	;INDICATE DISPLAY BUSY WAS STILL SET
2101	021610	104044			ERROR	44	;DISPLAY BUSY FAILED TO CLR ON STOP INSTR

2102
2103
2104

*TEST 154 TEST THAT RESET WILL CLEAR DISPLAY BUSY

(3)
(3)
(2) 021612 000004
(1) 021614 012737 000040 001166
(2) 021622 012737 000154 001206
2105 021630 013737 002042 001122
2106 021636 012777 020000 160216
2107 021644 012737 164000 031076
2108 021652 012777 031076 160150
2109 021660 012737 000404 001124
2110 021666 000005
2111 021670 017737 160146 001126
2112 021676 023737 001124 001126
2113 021704 001401
2114 021706 104044

TST154: SCOPE
MOV #40,\$TIMES ;:DO 40 ITERATIONS
MOV #154,\$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
MOV SREG1,\$BDADR ;:SET UP REGADRS 12
MOV #BIT13,@STKPT ;:SET MAINT SW 2
MOV #164000,BUFFER ;:SET UP NOP INSTR
MOV #BUFFER,@DPC ;:START
MOV #404,\$GDDAT ;:EXPECT ONLY CHAR & VECTOR SCALE BITS
RESET ;:CLR BUSY
MOV @SREG1,\$BDDAT ;:READ REG 12
CMP \$GDDAT,\$BDDAT ;:IS IT CORRECT?
BEQ TST155 ;:NEXT TEST IF OK
ERROR 44 ;:RESET FAILED TO CLR DISPLAY BUSY

2115
2116
(3)
(3)

*TEST 155 TEST THAT EXTERNAL STOP WILL SET & RESET WITH DISPLAY OFF

(2) 021710 000004
(1) 021712 012737 000040 001166
(2) 021720 012737 000155 001206
2117 021726 012777 020000 160126
2118 021734 012737 000340 177776
2119 021742 000005
2120 021744 013737 002042 001122
2121 021752 012737 000200 001124
2122 021760 013777 001124 160054
2123 021766 105777 160050
2124 021772 100403
2125 021774 005037 001126
2126 022000 104045
2127 022002 012777 031076 160020 1\$:
2128 022010 105777 160026
2129 022014 100006
2130 022016 005037 001124
2131 022022 012737 000200 001126
2132 022030 104045
2133

TST155: SCOPE
MOV #40,\$TIMES ;:DO 40 ITERATIONS
MOV #155,\$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;:SET MAINT SWITCH #2
MOV #340,\$PSW ;:SET PRIORITY TO HIGHEST LEVEL
RESET ;:MAKE SURE DISPLAY IS OFF
MOV SREG1,\$BDADR ;:SET UP REG 12 ADRS
MOV #200,\$GDDAT ;:EXPECT EXTERNAL STOP TO BE SET
MOV \$GDDAT,@SREG1 ;:SET EXTERNAL STOP
TSTB @SREG1 ;:SEE IF BIT SET
BMI 1\$;:BR IF IT DID SET
CLR \$BDDAT ;:INDICATE IT WAS NOT SET
ERROR 45 ;:EXTERNAL STOP FAILED TO SET
MOV #BUFFER,@DPC ;:CLR EXTERNAL STOP BIT
TSTB @SREG1 ;:CHECK THAT IT WAS CLEARED
BPL TST156 ;:BR TO NEXT TEST IF OK
CLR \$GDDAT ;:EXPECTED ZERO
MOV #200,\$BDDAT ;:INDICATE THAT IT DID NOT CLR
ERROR 45 ;:EXTERNAL STOP FAILED TO CLR ON START

2134
(3)
(3)
(2) 022032 000004
(2) 022034 012737 000156 001206
2135 022042 012777 020000 160012
2136 022050 012737 000340 177776
2137 022056 012737 000200 001124
2138 022064 012737 164000 031076
2139 022072 012777 031076 157730
2140 022100 012737 164000 031100
2141 022106 013777 001124 157726
2142 022114 005277 157710
2143 022120 013737 002042 001122
2144 022126 032777 000200 157706
2145 022134 001004
2146 022136 005037 001126
2147 022142 104045
2148 022144 000415
2149 022146 012777 031076 157654 1\$:
2150 022154 032777 000200 157660
2151 022162 001406
2152 022164 005037 001124
2153 022170 012737 000200 001126
2154 022176 104045
2155
2156
(3)
(3)
(2) 022200 000004
(1) 022202 012737 000010 001166
(2) 022210 012737 000157 001206
2157 022216 000005
2158 022220 013737 002042 001122
2159 022226 005037 001124
2160 022232 012777 000200 157602
2161 022240 000005
2162 022242 032777 000200 157572
2163 022250 001404
2164 022252 012737 000200 001126
2165 022260 104045
2166
2167
(3)
(3)
(2) 022262 000004
(1) 022264 012737 000040 001166
(2) 022272 012737 000160 001206
2168 022300 012777 020000 157554
2169 022306 000005
2170 022310 012737 000340 177776
2171 022316 013737 002042 001122
2172 022324 012737 000200 001124
2173 022332 013701 001252
2174 022336 042701 160000
2175 022342 012711 022406

```
*****
:*TEST 156 TEST THAT EXTERNAL STOP WILL SET & RESET WITH DISPLAY BUSY
*****
TST156: SCOPE
MOV #156,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;;SET MAINT SWITCH #2
MOV #340,PSW ;;SET PSW TO 7
MOV #200,$GDDAT ;;EXPECT EXTERNAL STOP TO BE SET INITIALLY
MOV #164000,BUFFER ;;SET UP A NOP INSTR
MOV #BUFFER,@DPC ;;START
MOV #164000,BUFFER+2 ;;SET UP ANOTHER NOP
MOV $GDDAT,@SREG1 ;;ISSUE A SOFTWARE STOP
INC @DPC ;;THIS RESUME SHOULD ALLOW EXTERNAL STOP TO SET
MOV SREG1,$BDADR ;;SET UP REG ADRS 12
BIT #200,@SREG1 ;;SEE IF IT SET
BNE 1$ ;;BR IF SO
CLR $BDDAT ;;INDICATE IT DID NOT SET
ERROR 45 ;;EXTERNAL STOP FAILED TO SET WHEN DISPLAY BUSY
BR TST157 ;;GO LOOK FOR LOOP ON TEST SW
MOV #BUFFER,@DPC ;;START - THIS SHOULD CLR EXT STOP
BIT #200,@SREG1 ;;CHECK THAT IT DID CLEAR
BEQ TST157 ;;GO TO NEXT TEST IF EXTERNAL STOP CLEARED
CLR $GDDAT ;;EXPECTED ZERO
MOV #200,$BDDAT ;;INDICATE IT DID NOT CLR
ERROR 45 ;;EXTERNAL STOP FAILED TO CLR ON START
```

```
*****
:*TEST 157 TEST THAT RESET WILL CLEAR EXTERNAL STOP
*****
TST157: SCOPE
MOV #10,$TIMES ;;DO 10 ITERATIONS
MOV #157,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
RESET ;;MAKE SURE DISPLAY IS OFF
MOV SREG1,$BDADR ;;SET UP REG 12 ADRS
CLR $GDDAT ;;EXPECT RESET STATE
MOV #200,@SREG1 ;;SET EXTERNAL STOP
RESET ;;CLR IT
BIT #200,@SREG1 ;;DID IT CLR?
BEQ TST160 ;;NEXT TEST IF SO
MOV #200,$BDDAT ;;SHOW THAT IT WAS STILL SET
ERROR 45 ;;RESET FAILED TO CLR EXTERNAL STOP
```

```
*****
:*TEST 160 TEST THAT EXTERNAL STOP WILL CAUSE AN INTERRUPT
*****
TST160: SCOPE
MOV #40,$TIMES ;;DO 40 ITERATIONS
MOV #160,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
MOV #BIT13,@STKPT ;;SET MAINT SWITCH #2
RESET ;;MAKE SURE DISPLAY IS OFF
MOV #340,PSW ;;SET PSW TO 7
MOV SREG1,$BDADR ;;SET UP REG 12 ADRS
MOV #200,$GDDAT ;;EXPECT EXTERNAL STOP FLAG
MOV $VECT1,R1 ;;GET BASIC VECTOR ADRS
BIC #160000,R1 ;;CLEAR PSW BITS ** B **
MOV #1$(,R1) ;;SET UP STOP INT ADRS
```

```

2176 022346 012761 000340 000002      MOV      #340,2(R1)      ;SET UP PSW TO 7 ON INT
2177 022354 012777 000200 157460      MOV      #200,@SREG1    ;CAUSE AN EXTERNAL STOP INT
2178 022362 005037 177776              CLR      PSW            ;ALLOW INT TO OCCUR
2179 022366 017737 157450 001126      MOV      @SREG1,$BDDAT  ;READ REG 12
2180 022374 042737 177577 001126      BIC      #177577,$BDDAT ;SAVE ONLY STOP FLAG
2181 022402 104033              ERROR   33             ;EXTERNAL STOP FAILED TO INTERRUPT
2182 022404 000410              BR       2$            ;GO LOOK FOR LOOP ON TEST SWITCH
2183 022406 022626              1$:  CMP      (R6)+,(R6)+  ;FIX STK SINCE NO RETURN
2184 022410 032777 000200 157424      BIT      #200,@SREG1    ;CK THAT STOP FLAG IS SET
2185 022416 001003              BNE     2$            ;BR IF SO
2186 022420 005037 001126      CLR      $BDDAT        ;INDICATE FLAG WAS NOT SET
2187 022424 104033              ERROR   33             ;EXTERNAL STOP INTERRUPTED BUT FLAG NOT SET
2188 022426 004737 024454              2$:  JSR      PC,RSTVEC   ;RESTORE EXTERNAL STOP VECTOR WITH HALT
2189
2190

```

```

(3)
(3)
(2) 022432 000004
(1) 022434 012737 000040 001166      MOV      #40,$TIMES     ;:DO 40 ITERATIONS
(2) 022442 012737 000161 001206      MOV      #161,$TESTN    ;:SET TEST NUMBER IN APT MAIL BOX
2191 022450 012777 020000 157404      MOV      #BIT13,@STKPT  ;SET MAINT SWITCH #2
2192 022456 012737 000340 177776      MOV      #340,PSW      ;SET PRIORITY TO HIGHEST LEVEL
2193 022464 013701 001252              MOV      $VECT1,R1     ;GET BASIC VECTOR ADRS
2194 022470 042701 160000              BIC      #160000,R1    ; CLEAR PSW BITS      ** B **
2195 022474 012761 022566 000010      MOV      #1$,10(R1)    ;SET UP TIMEOUT INT SERVICE ADRS
2196 022502 012761 000340 000012      MOV      #340,12(R1)   ;SET UP PRIORITY TO 7 ON INT
2197 022510 013737 002042 001122      MOV      SREG1,$BDADR  ;SET UP REG 12 ADRS
2198 022516 012737 004060 001124      MOV      #4060,$GDDAT  ;EXPECT TIMEOUT & DPC 16,17
2199 022524 012777 007600 157306      MOV      #7600,@RLO    ;RELOCATE TO ADRS 760000
2200 022532 005037 177776              CLR      PSW           ;ALLOW INTERRUPT TO OCCUR
2201 022536 012777 000000 157264      MOV      #0,@DPC       ;DISPLAY WILL TIMEOUT AT ADRS 760000
2202 022544 021616              CMP      (SP),(SP)     ;ALLOW TIME
2203 022546 017737 157270 001126      MOV      @SREG1,$BDDAT ;READ REG 12
2204 022554 042737 173717 001126      BIC      #173717,$BDDAT ;SAVE ONLY TIMEOUT BIT & DPC 16,17
2205 022562 104046              ERROR   46             ;VS60 FAILED TO INTERRUPT ON TIME OUT
2206 022564 000414              BR       2$            ;GO LOOK FOR LOOP ON TEST SWITCH
2207 022566 022626              1$:  CMP      (R6)+,(R6)+  ;FIX STACK SINCE NO RETURN
2208 022570 017737 157246 001126      MOV      @SREG1,$BDDAT ;READ STATUS
2209 022576 042737 173717 001126      BIC      #173717,$BDDAT ;SAVE TIMEOUT & DPC 16,17
2210 022604 023737 001124 001126      CMP      $GDDAT,$BDDAT ;CORRECT?
2211 022612 001401              BEQ     2$            ;BR IF SO
2212 022614 104046              ERROR   46             ;TIMEOUT BIT FAILED TO SET ON INTERRUPT
2213 022616 004737 024454              2$:  JSR      PC,RSTVEC   ;RESET TIMEOUT VECTOR WITH HALT
2214 022622 000005              RESET                  ;CLR RELOCATE REG BEFORE ADVANCING
2215
2216

```

```

(3)
(3)
(2) 022624 000004
(1) 022626 012737 000040 001166      MOV      #40,$TIMES     ;:DO 40 ITERATIONS
(2) 022634 012737 000162 001206      MOV      #162,$TESTN    ;:SET TEST NUMBER IN APT MAIL BOX
2217 022642 013737 002042 001122      MOV      SREG1,$BDADR  ;SET L.P. REG ADRS 12
2218 022650 012777 020000 157204      MOV      #BIT13,@STKPT ;SET MAINT SW 2
2219 022656 012737 000340 177776      MOV      #340,PSW      ;DON'T LET INTR OCCUR.
2220 022664 012737 000404 001124      MOV      #404,$GDDAT   ;EXPECT ONLY CHAR & VECTOR SCALE BITS
2221 022672 012777 007600 157140      MOV      #7600,@RLO    ;RELOCATE TO ADRS 760000

```

2222 022700 012777 000000 157122
2223 022706 021616
2224 022710 000005
2225 022712 017737 157124 001126
2226 022720 023737 001124 001126
2227 022726 001401
2228 022730 104032
2229
2230

MOV #0,@DPC ;DISPLAY SHOULD TIMEOUT AT ADRS 760000
CMP (SP),(SP) ;ALLOW TIME
RESET ;CLR REG REG 12
MOV @SREG1,\$BDDAT ;READ REG 12
CMP \$GDDAT,\$BDDAT ;IS IT CORRECT?
BEQ TST163 ;NEXT TEST IF OK
ERROR 32 ;RESET FAILED TO CLR TIMEOUT

(3)
(3)
(2) 022732 000004
(1) 022734 012737 000040 001166
(2) 022742 012737 000163 001206
2231 022750 012737 000340 177776
2232 022756 012777 020000 157076
2233 022764 012737 144000 001124
2234 022772 012737 164360 031076
2235 023000 012777 031076 157022
2236 023006 052777 040000 157046
2237 023014 012737 122000 031100
2238 023022 005277 157002
2239 023026 012737 001777 031102
2240 023034 005277 156770
2241 023040 013737 002052 001122
2242 023046 052777 040000 156776
2243 023054 017737 156772 001126
2244 023062 023737 001124 001126
2245 023070 001401
2246 023072 104034
2247 023074 000005
2248 023076 012737 100000 001124
2249 023104 017737 156742 001126
2250 023112 023737 001124 001126
2251 023120 001401
2252 023122 104032
2253
2254

*TEST 163 TEST THAT LIGHT PEN FLAG WILL SET THEN CLEAR ON RESET - CONS 0

TST163: SCOPE
MOV #40,\$TIMES ;DO 40 ITERATIONS
MOV #163,\$TESTN ;SET TEST NUMBER IN APT MAIL BOX
MOV #340,PSW ;HIGHEST PRIORITY
MOV #20000,@STKPT ;SET MAINT SW 2
MOV #144000,\$GDDAT ;EXPECT L.P. FLAG INITIALLY
MOV #164360,BUFFER ;ENABLE L.P. FLG INTR
MOV #BUFFER,@DPC ;START
BIS #40000,@STKPT ;SET MAINT SW 3
MOV #122000,BUFFER+2 ;GRAPH X MODE
INC @DPC ;RESUME
MOV #1777,BUFFER+4 ;X=1777
INC @DPC ;RESUME
MOV CONS,\$BDADR ;SET UP REG ADRS 22
BIS #40000,@CONS ;SET L.P. FLAG
MOV @CONS,\$BDDAT ;READ FLAGS REG 22
CMP \$GDDAT,\$BDDAT ;IS IT CORRECT?
BEQ 1\$;BR IF SO
ERROR 34 ;LIGHT PEN FLAG FAILED TO SET - CONSOLE 0
1\$:
RESET ;CLR FLAGS
MOV #100000,\$GDDAT ;EXPECT ONLY INT ENA CONSOLE 0
MOV @CONS,\$BDDAT ;READ REG 22
CMP \$GDDAT,\$BDDAT ;DID REG 22 RESET?
BEQ TST164 ;NEXT TEST IF SO
ERROR 32 ;RESET FAILED TO CLEAR FLAGS - CONSOLE 0

(3)
(3)
(2) 023124 000004
(1) 023126 012737 000040 001166
(2) 023134 012737 000164 001206
2255 023142 012737 000340 177776
2256 023150 012777 020000 156704
2257 023156 012737 101440 001124
2258 023164 012737 164760 031076
2259 023172 012777 031076 156630
2260 023200 052777 040000 156654
2261 023206 012737 122000 031100
2262 023214 005277 156610
2263 023220 012737 001777 031102
2264 023226 005277 156576
2265 023232 013737 002052 001122
2266 023240 052777 000400 156604
2267 023246 017737 156600 001126

*TEST 164 TEST THAT LIGHT PEN FLAG WILL SET THEN CLEAR ON RESET - CONS 1

TST164: SCOPE
MOV #40,\$TIMES ;DO 40 ITERATIONS
MOV #164,\$TESTN ;SET TEST NUMBER IN APT MAIL BOX
MOV #340,PSW ;HIGHEST PRIORITY
MOV #20000,@STKPT ;SET MAINT SW 2
MOV #101440,\$GDDAT ;EXPECT L.P. FLAG INITIALLY
MOV #164760,BUFFER ;ENABLE L.P. FLG
MOV #BUFFER,@DPC ;START
BIS #40000,@STKPT ;SET MAINT SW 3
MOV #122000,BUFFER+2 ;GRAPH X MODE
INC @DPC ;RESUME
MOV #1777,BUFFER+4 ;X=1777
INC @DPC ;RESUME
MOV CONS,\$BDADR ;SET UP REG ADRS 22
BIS #400,@CONS ;SET L.P. FLAG
MOV @CONS,\$BDDAT ;READ IT BACK

```

2268 023254 023737 001124 001126
2269 023262 001401
2270 023264 104034
2271 023266 000005
2272 023270 012737 100000 001124
2273 023276 017737 156550 001126
2274 023304 023737 001124 001126
2275 023312 001401
2276 023314 104032
2277
2278
(3)
(3)
(2) 023316 000004
(1) 023320 012737 000100 001166
(2) 023326 012737 000165 001206
2279 023334 012737 000340 177776
2280 023342 013701 001252
2281 023346 042701 160000
2282 023352 012761 023522 000004
2283 023360 012761 000340 000006
2284 023366 012737 144000 001124
2285 023374 012700 000001
2286 023400 012777 020040 156454
2287 023406 012737 164360 031076
2288 023414 012777 031076 156406
2289 023422 052777 040000 156432
2290 023430 012737 122000 031100
2291 023436 005277 156366
2292 023442 012737 001777 031102
2293 023450 005277 156354
2294 023454 013737 002052 001122
2295 023462 052777 040000 156362
2296 023470 005037 177776
2297 023474 012701 000040
2298 023500 005301
2299 023502 001376
2300 023504 005700
2301 023506 001451
2302 023510 017737 156336 001126
2303 023516 104050
2304 023520 000454
2305 023522 022626
2306 023524 017737 156322 001126
2307 023532 005700
2308 023534 001002
2309 023536 104050
2310 023540 000444
2311 023542 023737 001124 001126
2312 023550 001402
2313 023552 104050
2314 023554 000436
2315 023556 032777 000200 156246
2316 023564 001012
2317 023566 013737 002032 001122
2318 023574 012737 000200 001124

```

```

CMP $GDDAT,$BDDAT ;CORRECT?
BEQ 1$ ;BR IF SO
ERROR 34 ;LIGHT PEN FLAG FAILED TO SET - CONSOLE 1
1$: RESET ;CLR FLAGS
MOV #100000,$GDDAT ;EXPECT ONLY INT EN FOR CONSOLE 0
MOV @CONS,$BDDAT ;READ REG 22
CMP $GDDAT,$BDDAT ;CORRECT?
BEQ TST165 ;NEXT TEST IF SO
ERROR 32 ;RESET FAILED TO CLR FLAGS - CONSOLE 1

;*****
;*TEST 165 TEST THAT LIGHT PEN FLAG - CONSOLE 0 WILL CAUSE AN INTERRUPT
;*****
TST165: SCOPE
MOV #100,$TIMES ;DO 100 ITERATIONS
MOV #165,$TESTN ;SET TEST NUMBER IN APT MAIL BOX
MOV #340,$PSW ;SET PRIORITY TO 7
MOV $VECT1,R1 ;GET BASIC VS60 VECTOR ADRS
BIC #160000,R1 ; CLEAR PSW BITS ** B **
MOV #3$,4(R1) ;SET UP INTR SERVICE ADRS
MOV #340,6(R1) ;SET UP PRIORITY TO 7 ON INT
MOV #144000,$GDDAT ;EXPECT INT EN, L.P. FLG, L.P. INT EN
MOV #1,R0 ;R0 WHEN NON-ZERO MEANS INT EXPECTED
MOV #20040,@STKPT ;SET MAINT SW 2
MOV #164360,BUFFER ;SET UP TO ENAB L.P. INT & INTENSITY EN
1$: MOV #BUFFER,@DPC ;START
BIS #40000,@STKPT ;SET MAINT SW 3
MOV #122000,BUFFER+2 ;GRAPH X MODE
INC @DPC ;PICK UP GRAPH X INSTR - INT WILL OCCUR IN DATA MODE
MOV #1777,BUFFER+4 ;X=1777
INC @DPC ;RESUME FOR X
MOV CONS,$BDADR ;SET UP REG ADRS 22
BIS #40000,@CONS ;SET L.P. FLAG
CLR PSW ;ALLOW INTR
MOV #40,R1 ;SET UP DELAY
2$: DEC R1 ;COUNT AWAY
BNE 2$ ;STAY UNTIL DONE
TST R0 ;DO WE EXPECT INT THIS PASS - NO IF ZERO
BEQ 7$ ;BR IF NOT
MOV @CONS,$BDDAT ;READ L.P. STATUS
ERROR 50 ;LIGHT PEN CONSOLE 0 INTERRUPT FAILURE
BR 8$ ;GO LOOK FOR LOOP ON TEST SW
3$: CMP (R6)+,(R6)+ ;FIX STACK SINCE NO RTI
MOV @CONS,$BDDAT ;GET CONTENTS OF REG 22
TST R0 ;DID WE EXPECT AN L.P. INTERRUPT THIS PASS?
BNE 4$ ;BR IF SO
ERROR 50 ;L.P. FLAG INTERRUPTED WHEN NOT ENABLED
BR 8$ ;GO LOOK FOR LOOP ON TEST SW
4$: CMP $GDDAT,$BDDAT ;SEE IF STATUS IS CORRECT AFTER INT
BEQ 5$ ;BR IF SO
ERROR 50 ;INCORRECT INTERRUPT STATUS
BR 8$ ;GO LOOK FOR LOOP ON TEST SW
5$: BIT #200,@SREG0 ;L.P. FLG SET IN REG 02 ALSO?
BNE 6$ ;BR IF SO
MOV SREG0,$BDADR ;SET UP REG ADRS 02
MOV #200,$GDDAT ;EXPECTED L.P. FLAG

```

```
2319 023602 005037 001126 CLR $BDDAT ;SHOW IT WAS NOT SET
2320 023606 104050 ERROR 50 ;L.P. FLAG FAILED TO SET AT REG 02
2321 023610 000420 BR 8$ ;GO LOOK FOR LOOP ON TEST SW
2322 023612 012737 100000 001124 6$: MOV #100000,$GDDAT ;SET UP EXPECTED WITH L.P. FLAG INT DISABLED
2323 023620 012737 164340 031076 MOV #164340,BUFFER ;SET UP INSTR WITH L.P. FLAG INT DISABLED
2324 023626 005000 CLR R0 ;SET TEST SWITCH TO NO INTERRUPT EXPECTED
2325 023630 000671 BR 1$ ;GO REPEAT TEST WITH INT EN DISABLED
2326 023632 017737 156214 001126 7$: MOV @CONS,$BDDAT ;READ REG 22
2327 023640 023737 001126 001124 CMP $BDDAT,$GDDAT ;FLAG CLR ON START?
2328 023646 001401 BEQ 8$ ;BR IF SO
2329 023650 104034 ERROR 34 ;START FAILED TO CLR L.P. FLAG
2330 023652 004737 024454 8$: JSR PC,RSTVEC ;RESET L.P. INTERRUPT VECTOR WITH HALT
2331
2332
(3)
(3)
(2) 023656 000004 TST166: SCOPE
(1) 023660 012737 000100 001166 MOV #100,$TIMES ;:DO 100 ITERATIONS
(2) 023666 012737 000166 001206 MOV #166,$TESTN ;:SET TEST NUMBER IN APT MAIL BOX
2333 023674 012737 000340 177776 MOV #340,PSW ;:SET PRIORITY TO 7
2334 023702 013701 001252 MOV $VECT1,R1 ;:GET BASIC VS60 VECTOR ADRS
2335 023706 042701 160000 BIC #160000,R1 ;:CLEAR PSW BITS ** B **
2336 023712 012761 024062 000004 MOV #3$,4(R1) ;:SET UP INTR SERVICE ADRS
2337 023720 012761 000340 000006 MOV #340,6(R1) ;:SET UP PRIORITY TO 7 ON INT
2338 023726 012737 101440 001124 MOV #101440,$GDDAT ;:EXPECT INT EN, L.P. FLG, L.P. INT EN
2339 023734 012700 000001 MOV #1,R0 ;:R0 WHEN NON-ZERO MEANS INT EXPECTED
2340 023740 012777 020040 156114 MOV #20040,@STKPT ;:SET MAINT SW 2
2341 023746 012737 164760 031076 MOV #164760,BUFFER ;:SET UP TO ENAB L.P. INT & INTENSITY EN
2342 023754 012777 031076 156046 1$: MOV #BUFFER,@DPC ;:START
2343 023762 052777 040000 156072 BIS #40000,@STKPT ;:SET MAINT SW 3
2344 023770 012737 122000 031100 MOV #122000,BUFFER+2 ;:GRAPH X MODE
2345 023776 005277 156026 INC @DPC ;:PICK UP GRAPH X INSTR - INT WILL OCCUR IN DATA MODE
2346 024002 012737 001777 031102 MOV #1777,BUFFER+4 ;:X=1777
2347 024010 005277 156014 INC @DPC ;:RESUME FOR X
2348 024014 013737 002052 001122 MOV CONS,$BDADR ;:SET UP REG ADRS 22
2349 024022 052777 000400 156022 BIS #400,@CONS ;:SET L.P. FLAG
2350 024030 005037 177776 CLR PSW ;:ALLOW INTR
2351 024034 012701 000040 MOV #40,R1 ;:SET UP DELAY
2352 024040 005301 2$: DEC R1 ;:COUNT AWAY
2353 024042 001376 BNE 2$ ;:STAY UNTIL DONE
2354 024044 005700 TST R0 ;:DO WE EXPECT INT THIS PASS - NO IF ZERO
2355 024046 001433 BEQ 6$ ;:BR IF NOT
2356 024050 017737 155776 001126 MOV @CONS,$BDDAT ;:READ L.P. STATUS
2357 024056 104050 ERROR 50 ;:LIGHT PEN CONSOLE 1 INTERRUPT FAILURE
2358 024060 000436 BR 7$ ;:GO LOOK FOR LOOP ON TEST SW
2359 024062 022626 3$: CMP (R6)+,(R6)+ ;:FIX STACK SINCE NO RTI
2360 024064 017737 155762 001126 MOV @CONS,$BDDAT ;:GET L.P. STATUS
2361 024072 005700 TST R0 ;:DID WE EXPECT AN L.P. INTERRUPT THIS PASS?
2362 024074 001002 BNE 4$ ;:BR IF SO
2363 024076 104050 ERROR 50 ;:L.P. FLAG INTERRUPTED WHEN NOT ENABLED
2364 024100 000426 BR 7$ ;:GO LOOK FOR LOOP ON TEST SW
2365 024102 023737 001124 001126 4$: CMP $GDDAT,$BDDAT ;:SEE IF STATUS IS CORRECT AFTER INT
2366 024110 001402 BEQ 5$ ;:BR IF SO
2367 024112 104050 ERROR 50 ;:INCORRECT L.P. STATUS
2368 024114 000420 BR 7$ ;:GO LOOK FOR LOOP ON TEST SW
2369 024116 012737 101000 001124 5$: MOV #101000,$GDDAT ;:SET UP TEST WITH L.P. FLAG INT DISABLED
```

```
2370 024124 012737 164740 031076      MOV    #164740,BUFFER ;SET UP INSTR WITH L.P. FLAG DISABLED
2371 024132 005000                    CLR    R0              ;SET TEST SWITCH TO NO INTERRUPT EXPECTED
2372 024134 000707                    BR     1$             ;GO REPEAT TEST WITH INT EN DISABLED
2373 024136 017737 155710 001126 6$:   MOV    @CONS,$BDDAT  ;READ REG 22
2374 024144 023737 001124 001126      CMP    $GDDAT,$BDDAT ;L.P. FLG CLR ON START?
2375 024152 001401                    BEQ    7$             ;BR IF SO
2376 024154 104034                    ERROR  34             ;START FAILED TO CLR L.P. FLAG
2377 024156 004737 024454 7$:   JSR    PC,RSTVEC    ;RESET L.P. INTERRUPT VECTOR WITH HALT
2378
2379
```

```
(3)
(3)
(2) 024162 000004
(2) 024164 012737 000167 001206
2380 024172 012777 020000 155662
2381 024200 012737 000100 001124
2382 024206 012737 110000 031076
2383 024214 005037 031100
2384 024220 005037 031102
2385 024224 012777 031076 155576
2386 024232 000402
2387 024234 005277 155570 1$:   INC    @DPC           ;START
2388 024240 013737 002062 001122 2$:   MOV    STKPT,$BDADR  ;GO ALLOW TIME FOR INSTR NPR
2389 024246 017737 155610 001126      MOV    @STKPT,$BDDAT ;PICK UP DATA WORD
2390 024254 042737 177077 001126      BIC    #177077,$BDDAT ;SET UP REG ADRS 32
2391 024262 023737 001124 001126      CMP    $GDDAT,$BDDAT ;READ REG 32
2392 024270 001401                    BEQ    3$             ;SAVE ONLY WORD INDICATORS
2393 024272 104041                    ERROR  41             ;IS IT CORRECT?
2394 024274 006337 001124 3$:   ASL    $GDDAT        ;BR IF SO
2395 024300 022737 001000 001124      CMP    #1000,$GDDAT  ;WORD INDICATOR FAILURE
2396 024306 001352                    BNE    1$             ;ADVANCE TO NEXT WORD
2397
2398
```

```
*****
;*TEST 167 TEST THAT WORD 0,1 & 2 INDICATORS WILL SET
*****
TST167: SCOPE
MOV    #167,$TESTN    ;;SET TEST NUMBER IN APT MAIL BOX
MOV    #BIT13,@STKPT  ;SET MAINT SWITCH #2
MOV    #100,$GDDAT    ;EXPECT WORD 0 INDICATION INITIALLY
MOV    #110000,BUFFER ;SET UP LONG VECTOR INSTR
CLR    BUFFER+2       ;CLR X,Y
CLR    BUFFER+4
MOV    #BUFFER,@DPC   ;START
BR     2$             ;GO ALLOW TIME FOR INSTR NPR
1$:   INC    @DPC       ;PICK UP DATA WORD
2$:   MOV    STKPT,$BDADR ;SET UP REG ADRS 32
MOV    @STKPT,$BDDAT ;READ REG 32
BIC    #177077,$BDDAT ;SAVE ONLY WORD INDICATORS
CMP    $GDDAT,$BDDAT ;IS IT CORRECT?
BEQ    3$             ;BR IF SO
ERROR  41             ;WORD INDICATOR FAILURE
3$:   ASL    $GDDAT        ;ADVANCE TO NEXT WORD
CMP    #1000,$GDDAT  ;HAVE ALL BEEN TESTED?
BNE    1$             ;BR IF NOT
```

```
*****
;*TEST 170 TEST FOR HALT AT END OF DIAGNOSTIC (SW12 SET)
*****
```

```
TST170: SCOPE
MOV    #170,$TESTN    ;;SET TEST NUMBER IN APT MAIL BOX
BIT    #BIT12,@SWR    ;IS SW12 SET?
BEQ    $EOP           ;BR IF NOT
HALT                    ;COMPLETED PASS - SW12 SAYS HALT
```

```
2400 024326 001401
2401 024330 000000
```



```
2406 ::*****  
2407 ;THIS CODE RESETS ALL VS60 VECTORS WITH HALTS  
2408 ::*****  
2409 024454 012737 000340 177776 RSTVEC: MOV #340,PSW ;RESET PRIORITY TO HIGHEST LEVEL  
2410 024462 012701 000004 MOV #4,R1 ;SET UP VECTOR LOCATION COUNT  
2411 024466 013700 001252 MOV $VECT1,R0 ;GET 1ST VS60 VECTOR ADRS  
2412 024472 042700 160000 BIC #160000,R0 ; CLEAR PSW BITS ** B **  
2413 024476 010010 1$: MOV R0,(R0) ;SET UP ADRS TO HALT  
2414 024500 062720 000002 ADD #2,(R0)+ ;POINT IT TO HALT  
2415 024504 005020 CLR (R0)+ ;SET UP HALT  
2416 024506 005301 DEC R1 ;COUNT THIS VECTOR RESTORE  
2417 024510 001372 BNE 1$ ;BR IF MORE TO RESTORE  
2418 024512 000207 RTS PC ;RETURN IF ALL DONE  
2419  
2420 ::*****  
2421 ;THIS IS A GENERAL PURPOSE DELAY ROUTINE (REG CNT OF 20)  
2422 ::*****  
2423 024514 013700 002026 DLAY: MOV DELAY,R0 ;GET COUNT  
2424 024520 005300 1$: DEC R0 ;COUNT AWAY  
2425 024522 001376 BNE 1$ ;BR TILL DONE  
2426 024524 000207 RTS PC ;EXIT  
2427
```



```
(1) 024750 011637 001110      MOV      (SP), $LPERR      ;;SAVE ERROR LOOP ADDRESS
(1) 024754 005037 001170      CLR      $ESCAPE          ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
(1) 024760 112737 000001 001115  MOVVB   #1, $ERMAX        ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
(1) 024766 013777 001102 154146 $OVER:  MOV      $TSTNM, @DISPLAY ;;DISPLAY TEST NUMBER
(1) 024774 013716 001106      MOV      $LPADR, (SP)     ;;FUDGE RETURN ADDRESS
(1) 025000 000002      RTI                      ;;FIXES PS
(1) 025002 003720      $MXCNT: 2000.           ;;MAX. NUMBER OF ITERATIONS
2430 .SBTTL  APT COMMUNICATIONS ROUTINE
(1)
(2)
(1) 025004 112737 000001 025250 $ATY1:  MOVVB   #1, $FFLG      ;;TO REPORT FATAL ERROR
(1) 025012 112737 000001 025246 $ATY3:  MOVVB   #1, $MFLG      ;;TO TYPE A MESSAGE
(1) 025020 000403      BR      $ATYC
(1) 025022 112737 000001 025250 $ATY4:  MOVVB   #1, $FFLG      ;;TO ONLY REPORT FATAL ERROR
(1) 025030 $ATYC:
(3) 025030 010046      MOV      R0, -(SP)        ;;PUSH R0 ON STACK
(3) 025032 010146      MOV      R1, -(SP)        ;;PUSH R1 ON STACK
(1) 025034 105737 025246      TSTB   $MFLG            ;;SHOULD TYPE A MESSAGE?
(1) 025040 001450      BEQ     5$              ;;IF NOT: BR
(1) 025042 122737 000001 001222 CMPB   #APTENV, $ENV      ;;OPERATING UNDER APT?
(1) 025050 001031      BNE     3$              ;;IF NOT: BR
(1) 025052 132737 000100 001223 BITB   #APTPOOL, $ENVM    ;;SHOULD SPOOL MESSAGES?
(1) 025060 001425      BEQ     3$              ;;IF NOT: BR
(1) 025062 017600 000004      MOV      @4(SP), R0       ;;GET MESSAGE ADDR.
(1) 025066 062766 000002 000004 ADD     #2, 4(SP)         ;;BUMP RETURN ADDR.
(1) 025074 005737 001202 1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
(1) 025100 001375      BNE     1$              ;;IF NOT: WAIT
(1) 025102 010037 001216      MOV      R0, $MSGAD       ;;PUT ADDR IN MAILBOX
(1) 025106 105720 2$:     TSTB   (R0)+            ;;FIND END OF MESSAGE
(1) 025110 001376      BNE     2$
(1) 025112 163700 001216      SUB     $MSGAD, R0        ;;SUB START OF MESSAGE
(1) 025116 006200      ASR     R0               ;;GET MESSAGE LNTH IN WORDS
(1) 025120 010037 001220      MOV      R0, $MSGGLT      ;;PUT LENGTH IN MAILBOX
(1) 025124 012737 000004 001202 MOV     #4, $MSGTYPE      ;;TELL APT TO TAKE MSG.
(1) 025132 000413      BR      5$
(1) 025134 017637 000004 025160 3$:     MOV     @4(SP), 4$        ;;PUT MSG ADDR IN JSR LINKAGE
(1) 025142 062766 000002 000004 ADD     #2, 4(SP)         ;;BUMP RETURN ADDRESS
(3) 025150 013746 177776      MOV     177776, -(SP)    ;;PUSH 177776 ON STACK
(1) 025154 004737 026600      JSR     PC, $TYPE        ;;CALL TYPE MACRO
(1) 025160 000000 4$:     .WORD  0
(1) 025162 5$:
(1) 025162 105737 025250 10$:    TSTB   $FFLG            ;;SHOULD REPORT FATAL ERROR?
(1) 025166 001416      BEQ     12$            ;;IF NOT: BR
(1) 025170 005737 001222      TST     $ENV             ;;RUNNING UNDER APT?
(1) 025174 001413      BEQ     12$            ;;IF NOT: BR
(1) 025176 005737 001202 11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
(1) 025202 001375      BNE     11$            ;;IF NOT: WAIT
(1) 025204 017637 000004 001204 MOV     @4(SP), $FATAL    ;;GET ERROR #
(1) 025212 062766 000002 000004 ADD     #2, 4(SP)         ;;BUMP RETURN ADDR.
(1) 025220 005237 001202      INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
(1) 025224 105037 025250 12$:    CLRB   $FFLG            ;;CLEAR FATAL FLAG
(1) 025230 105037 025247      CLRB   $LFLG            ;;CLEAR LOG FLAG
(1) 025234 105037 025246      CLRB   $MFLG            ;;CLEAR MESSAGE FLAG
(3) 025240 012601      MOV     (SP)+, R1        ;;POP STACK INTO R1
(3) 025242 012600      MOV     (SP)+, R0        ;;POP STACK INTO R0
(1) 025244 000207      RTS     PC              ;;RETURN
```

(1) 025246 000
(1) 025247 000
(1) 025250 000
(1) 025252
(1) 000200
(1) 000001
(1) 000100
(1) 000040

\$MFLG: .BYTE 0 ;;MESSG. FLAG
\$LFLG: .BYTE 0 ;;LOG FLAG
\$FFLG: .BYTE 0 ;;FATAL FLAG
 .EVEN
APTSIZE=200
APTENV=001
APTSPOOL=100
APTCSUP=040
.SBTTL ERROR HANDLER ROUTINE

2431
(1)
(2)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)

```
*****  
*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,  
*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL  
*AND GO TO $ERRTYP ON ERROR  
*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:  
*SW15=1 HALT ON ERROR  
*SW13=1 INHIBIT ERROR TYPEOUTS  
*SW10=1 BELL ON ERROR  
*SW09=1 LOOP ON ERROR  
*CALL  
* ERROR N ;;ERROR=EMT AND N=ERROR ITEM NUMBER
```

(1) 025252
(2) 025252 113737 001102 002024
(1) 025260 105237 001103
(1) 025264 001775
(1) 025266 013777 001102 153646
(1) 025274 032777 002000 153636
(1) 025302 001402
(1) 025304 104401 001172
(1) 025310 005237 001112
(1) 025314 011637 001116
(1) 025320 162737 000002 001116
(1) 025326 117737 153564 001114
(1) 025334 032777 020000 153576
(1) 025342 001004
(1) 025344 004737 025454
(1) 025350 104401 001177
(1) 025354
(1) 025354 122737 000001 001222
(1) 025362 001007
(1) 025364 113737 001114 025376
(1) 025372 004737 025022
(1) 025376 000
(1) 025377 000
(1) 025400 000777
(1) 025402 005777 153532
(1) 025406 100001
(1) 025410 000000
(1) 025412 032777 001000 153520
(1) 025420 001402
(1) 025422 013716 001110
(1) 025426 005737 001170
(1) 025432 001402
(1) 025434 013716 001170
(1) 025440

```
$ERROR:  
7$: MOV $STNM,TSTNUM  
 INCB $ERFLG ;;SET THE ERROR FLAG  
 BEQ 7$ ;;DON'T LET THE FLAG GO TO ZERO  
 MOV $STNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG  
 BIT #BIT10,@SWR ;;BELL ON ERROR?  
 BEQ 1$ ;;NO - SKIP  
 TYPE $BELL ;;RING BELL  
1$: INC $ERTTL ;;COUNT THE NUMBER OF ERRORS  
 MOV (SP),$ERRPC ;;GET ADDRESS OF ERROR INSTRUCTION  
 SUB #2,$ERRPC  
 MOV $ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE  
 BIT #BIT13,@SWR ;;SKIP TYPEOUT IF SET  
 BNE 20$ ;;SKIP TYPEOUTS  
 JSR PC,$ERRTYP ;;GO TO USER ERROR ROUTINE  
 TYPE $CRLF  
20$: CMPB #APTENV,$ENV ;;RUNNING IN APT MODE  
 BNE 2$ ;;NO,SKIP APT ERROR REPORT  
 MOV $ITEMB,21$ ;;SET ITEM NUMBER AS ERROR NUMBER  
 JSR PC,$ATY4 ;;REPORT FATAL ERROR TO APT  
21$: .BYTE 0  
 .BYTE 0  
22$: BR 22$ ;;APT ERROR LOOP  
2$: TST @SWR ;;HALT ON ERROR  
 BPL 3$ ;;SKIP IF CONTINUE  
 HALT ;;HALT ON ERROR!  
3$: BIT #BIT09,@SWR ;;LOOP ON ERROR SWITCH SET?  
 BEQ 4$ ;;BR IF NO  
 MOV $LPERR,(SP) ;;FUDGE RETURN FOR LOOPING  
4$: TST $ESCAPE ;;CHECK FOR AN ESCAPE ADDRESS  
 BEQ 5$ ;;BR IF NONE  
 MOV $ESCAPE,(SP) ;;FUDGE RETURN ADDRESS FOR ESCAPE  
5$:
```

(1)	025440	022737	024420	000042	CMP	#SENDAD,@#42	::ACT-11 AUTO-ACCEPT?
(1)	025446	001001			BNE	6\$::BRANCH IF NO
(1)	025450	000000			HALT		::YES
(1)	025452						
(1)	025452	000002			RTI		::RETURN

2433
 2434

.SBTTL ERROR MESSAGE TYPEOUT ROUTINE

 *THIS ROUTINE USES THE "ITEM CONTROL BYTE" (\$ITEMB) TO DETERMINE WHICH
 *ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" (\$ERRTB),
 *AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.

(1)
 (2)
 (1)
 (1)
 (1)
 (1)
 (1)
 (1) 025454
 (1) 025454 104401 001177
 (1) 025460 010046
 (1) 025462 005000
 (1) 025464 153700 001114
 (1) 025470 001004
 (1)
 (2) 025472 013746 001116
 (2)
 (2) 025476 104402
 (1) 025500 000426
 (1) 025502 005300
 (1) 025504 006300
 (1) 025506 006300
 (1) 025510 006300
 (1) 025512 062700 001270
 (1) 025516 012037 025526
 (1) 025522 001404
 (1) 025524 104401
 (1) 025526 000000
 (1) 025530 104401 001177
 (1) 025534 012037 025544
 (1) 025540 001404
 (1) 025542 104401
 (1) 025544 000000
 (1) 025546 104401 001177
 (1) 025552 011000
 (1) 025554 001004
 (1) 025556 012600
 (1) 025560 104401 001177
 (1) 025564 000207
 (1) 025566
 (2) 025566 013046
 (2) 025570 104402
 (1) 025572 005710
 (1) 025574 001770
 (1) 025576 104401 025604
 (1) 025602 000771
 (1) 025604 020040 000
 (1) 025610

\$ERRTYP:
 TYPE , \$CRLF ;:'CARRIAGE RETURN' & 'LINE FEED'
 MOV RO,-(SP) ;:SAVE RO
 CLR RO ;:PICKUP THE ITEM INDEX
 BISB @#\$ITEMB,RO
 BNE 1\$;:IF ITEM NUMBER IS ZERO, JUST
 ;:TYPE THE PC OF THE ERROR
 MOV \$ERRPC,-(SP) ;:SAVE \$ERRPC FOR TYPEOUT
 ;:ERROR ADDRESS
 TYPOC ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
 BR 6\$;:GET OUT
 1\$: DEC RO ;:ADJUST THE INDEX SO THAT IT WILL
 ASL RO ;:WORK FOR THE ERROR TABLE
 ASL RO
 ASL RO
 ADD # \$ERRTB,RO ;:FORM TABLE POINTER
 MOV (RO)+,2\$;:PICKUP "ERROR MESSAGE" POINTER
 BEQ 3\$;:SKIP TYPEOUT IF NO POINTER
 TYPE ;:TYPE THE "ERROR MESSAGE"
 2\$: .WORD 0 ;:"ERROR MESSAGE" POINTER GOES HERE
 TYPE , \$CRLF ;:'CARRIAGE RETURN' & 'LINE FEED'
 3\$: MOV (RO)+,4\$;:PICKUP "DATA HEADER" POINTER
 BEQ 5\$;:SKIP TYPEOUT IF 0
 TYPE ;:TYPE THE "DATA HEADER"
 4\$: .WORD 0 ;:"DATA HEADER" POINTER GOES HERE
 TYPE , \$CRLF ;:'CARRIAGE RETURN' & 'LINE FEED'
 5\$: MOV (RO),RO ;:PICKUP "DATA TABLE" POINTER
 BNE 7\$;:GO TYPE THE DATA
 6\$: MOV (SP)+,RO ;:RESTORE RO
 TYPE , \$CRLF ;:'CARRIAGE RETURN' & 'LINE FEED'
 RTS PC ;:RETURN
 7\$: MOV @ (RO)+,-(SP) ;:SAVE @ (RO)+ FOR TYPEOUT
 TYPOC ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
 TST (RO) ;:IS THERE ANOTHER NUMBER?
 BEQ 6\$;:BR IF NO
 TYPE ,8\$;:TYPE TWO(2) SPACES
 BR 7\$;:LOOP
 8\$: .ASCIIZ / / ;:TWO(2) SPACES
 .EVEN

(1)	026126	001002			BNE	5\$::FALL THROUGH IF 0
(1)	026130	105716			TSTB	(SP)	::STILL DOING LEADING 0'S?
(1)	026132	100407			BMI	7\$::BR IF YES
(1)	026134	106316			ASLB	(SP)	::MSD?
(1)	026136	103003			BCC	6\$::BR IF NO
(1)	026140	116663	000001	177777	MOVB	1(SP),-1(R3)	::YES--SET THE SIGN
(1)	026146	052702	000060		BIS	#'0,R2	::MAKE THE BCD DIGIT ASCII
(1)	026152	052702	000040		BIS	#',R2	::MAKE IT A SPACE IF NOT ALREADY A DIGIT
(1)	026156	110223			MOVB	R2,(R3)+	::PUT THIS CHARACTER IN THE OUTPUT BUFFER
(1)	026160	005720			TST	(R0)+	::JUST INCREMENTING
(1)	026162	020027	000010		CMP	R0,#10	::CHECK THE TABLE INDEX
(1)	026166	002746			BLT	2\$::GO DO THE NEXT DIGIT
(1)	026170	003002			BGT	8\$::GO TO EXIT
(1)	026172	010502			MOV	R5,R2	::GET THE LSD
(1)	026174	000764			BR	6\$::GO CHANGE TO ASCII
(1)	026176	105726			TSTB	(SP)+	::WAS THE LSD THE FIRST NON-ZERO?
(1)	026200	100003			BPL	9\$::BR IF NO
(1)	026202	116663	177777	177776	MOVB	-1(SP),-2(R3)	::YES--SET THE SIGN FOR TYPING
(1)	026210	105013			CLRB	(R3)	::SET THE TERMINATOR
(3)	026212	012605			MOV	(SP)+,R5	::POP STACK INTO R5
(3)	026214	012603			MOV	(SP)+,R3	::POP STACK INTO R3
(3)	026216	012602			MOV	(SP)+,R2	::POP STACK INTO R2
(3)	026220	012601			MOV	(SP)+,R1	::POP STACK INTO R1
(3)	026222	012600			MOV	(SP)+,R0	::POP STACK INTO R0
(1)	026224	104401	026252		TYPE	,\$DBLK	::NOW TYPE THE NUMBER
(1)	026230	016666	000002	000004	MOV	2(SP),4(SP)	::ADJUST THE STACK
(1)	026236	012616			MOV	(SP)+,(SP)	
(1)	026240	000002			RTI		::RETURN TO USER
(1)	026242	023420			\$DTBL:	10000.	
(1)	026244	001750				1000.	
(1)	026246	000144				100.	
(1)	026250	000012				10.	
(1)	026252	000004			\$DBLK:	.BLKW 4	


```

(1) 026760 105337 027056          DECB  $CHARCNT      ;;DO NOT COUNT AS A COUNT
(1) 026764 000770                BR    7$             ;;LOOP
(1)                                ;HORIZONTAL TAB PROCESSOR
(1)                                8$:  MOVB  #' (SP)      ;;REPLACE TAB WITH SPACE
(1) 026766 112716 000040          9$:  JSR   PC,$TYPEC  ;;TYPE A SPACE
(1) 026772 004737 027012          BITB  #7,$CHARCNT  ;;BRANCH IF NOT AT
(1) 026776 132737 000007 027056  BNE   9$           ;;TAB STOP
(1) 027004 001372                TST   (SP)+        ;;POP SPACE OFF STACK
(1) 027006 005726                BR    2$           ;;GET NEXT CHARACTER
(1) 027010 000724                $TYPEC: TSTB @ $TPS ;;WAIT UNTIL PRINTER IS READY
(1) 027012 105777 152132          BPL   $TYPEC
(1) 027016 100375                MOVB  2(SP),@$TPB  ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1) 027020 116677 000002 152124  CMPB  #CR,2(SP)   ;;IS CHARACTER A CARRIAGE RETURN?
(1) 027026 122766 000015 000002  BNE   1$           ;;BRANCH IF NO
(1) 027034 001003                CLRB  $CHARCNT    ;;YES--CLEAR CHARACTER COUNT
(1) 027036 105037 027056          BR    $TYPEX      ;;EXIT
(1) 027042 000406                1$:  CMPB #LF,2(SP) ;;IS CHARACTER A LINE FEED?
(1) 027044 122766 000012 000002  BEQ   $TYPEX      ;;BRANCH IF YES
(1) 027052 001402                INCB  (PC)+       ;;COUNT THE CHARACTER
(1) 027054 105227                $CHARCNT: .WORD  0 ;;CHARACTER COUNT STORAGE
(1) 027056 000000                $TYPEX: .RTS     PC
(1) 027060 000207
  
```

2449


```
2454 .SBTTL ASCII MESSAGES
2455 027132 005015 051412 040524 HEADER: .ASCIZ <15><12><12>/START OF CZVSA-B VS60 INSTRUCTION TEST PART I/<15><12>
2456
2457 027216 040515 047111 027124 EM1: .ASCIZ /MAINT. SWITCH REG/
2458 027240 047514 042101 047111 EM2: .ASCIZ /LOADING D.P.C. REG <STATIC>/
2459 027274 047514 042101 047111 EM3: .ASCIZ /LOADING MODE REG/
2460 027315 123 050505 042525 EM4: .ASCIZ /SEQUENCING DPC REG/
2461 027340 047514 042101 047111 EM5: .ASCIZ /LOADING LINE TYPE REG/
2462 027366 047514 042101 047111 EM6: .ASCIZ /LOADING INTENSITY LEVEL REG/
2463 027422 047514 042101 047111 EM7: .ASCIZ /LOADING GRAPHPLOT INC REG/
2464 027454 044504 050123 040514 EM10: .ASCIZ /DISPLAY JUMP ABS TO AN ADRS/
2465 027510 044504 050123 040514 EM11: .ASCIZ /DISPLAY JUMP REL TO AN ADRS/
2466 027544 044504 050123 040514 EM12: .ASCIZ /DISPLAY JSR ABS TO AN ADRS/
2467 027577 104 051511 046120 EM13: .ASCIZ /DISPLAY JSR REL TO AN ADRS/
2468 027632 047514 042101 047111 EM14: .ASCIZ /LOADING X POS REG/
2469 027654 047514 042101 047111 EM15: .ASCIZ /LOADING Y POS REG/
2470 027676 052502 020123 044524 EM16: .ASCIZ /BUS TIME-OUT ER/
2471 027716 044103 051101 052040 EM17: .ASCIZ /CHAR TERMINATE ER/
2472 027740 020131 054504 040516 EM20: .ASCIZ /Y DYNAMIC OFFSET ER/
2473 027764 020130 054504 040516 EM21: .ASCIZ /X DYNAMIC OFFSET ER/
2474 030010 042522 047514 040503 EM22: .ASCIZ /RELOCATE REG ER/
2475 030030 027104 027120 027125 EM23: .ASCIZ /D.P.U. NAME REG ER/
2476 030053 123 040505 041522 EM24: .ASCIZ /SEARCH CODE OF ASSOC. NAME REG ER/
2477 030115 126 041505 047524 EM25: .ASCIZ /VECTOR SCALE ER/
2478 030135 103 040510 020122 EM26: .ASCIZ /CHAR SCALE ER/
2479 030153 102 044514 045516 EM27: .ASCIZ /BLINK LOGIC ER/
2480 030172 052111 046101 041511 EM30: .ASCIZ /ITALIC LOGIC ER/
2481 030212 042515 052516 046040 EM31: .ASCIZ /MENU LOGIC ER/
2482 030230 042522 042523 020124 EM32: .ASCIZ /RESET FAILED TO CLEAR A REG/
2483 030264 052123 050117 044440 EM33: .ASCIZ /STOP INTR ER/
2484 030301 114 050056 020056 EM34: .ASCIZ /L.P. FLAG ER/
2485 030316 047503 047514 020122 EM35: .ASCIZ /COLOR LEVEL ER/
2486 030335 111 042524 051516 EM36: .ASCIZ /ITENSITY ENABLE - CONS 0 + 1 ER/
2487 030375 111 052116 051105 EM37: .ASCIZ /INTERNAL STOP FLAG FAILED TO SET/
2488 030436 047111 042524 047122 EM40: .ASCIZ /INTERNAL STOP FLAG FAILED TO CLEAR/
2489 030501 127 051117 020104 EM41: .ASCIZ /WORD 0-1-2 INDICATOR ER/
2490 030531 114 050056 020056 EM42: .ASCIZ /L.P. INTR ENABLE - CONS 0 + 1 ER/
2491 030572 027114 027120 051440 EM43: .ASCIZ /L.P. SWITCH INTR ENABLE - CONS 0 + 1 ER/
2492 030642 044504 050123 040514 EM44: .ASCIZ /DISPLAY BUSY ER/
2493 030662 054105 042524 047122 EM45: .ASCIZ /EXTERNAL STOP LOGIC ER/
2494 030711 124 046511 026505 EM46: .ASCIZ /TIME-OUT INTR ER/
2495 030732 040516 042515 046440 EM47: .ASCIZ /NAME MATCH INTR ER/
2496 030755 114 050056 020056 EM50: .ASCIZ /L.P. FLAG INTR ER/
2497 030777 123 040524 045503 EM51: .ASCIZ /STACK PTR ER/
2498 031014 051105 050122 020103 DH1: .ASCIZ /ERRPC TSTNUM BUSADR EXPCT RCVD/
2499 031062 031062 .EVEN
2500 031062 001116 002024 001122 DT1: $ERRPC,TSTNUM,$BDADR,$GDDAT,$BDDAT,0
2501
2502 ;:*****
2503 ;THIS IS THE WORKING AREA FOR ALL VS60 NPR'S (INSTRS & DATA)
2504 ;FROM HERE TO THE END OF 28K
2505 ;:*****
2506 031076 000000 BUFFER: 0
2507
2508 000001 .END
```


TST147	020740	1998	2002#			
TST15	004060	585#				
TST150	021044	2014	2018#			
TST151	021150	2030	2035#			
TST152	021336	2051	2060	2064#		
TST153	021474	2083	2085#			
TST154	021612	2099	2104#			
TST155	021710	2113	2116#			
TST156	022032	2129	2134#			
TST157	022200	2148	2151	2156#		
TST16	004156	598#				
TST160	022262	2163	2167#			
TST161	022432	2190#				
TST162	022624	2216#				
TST163	022732	2227	2230#			
TST164	023124	2251	2254#			
TST165	023316	2275	2278#			
TST166	023656	2332#				
TST167	024162	2379#				
TST17	004230	604	607#			
TST170	024310	2398#				
TST2	002614	431#				
TST20	004302	613	616#			
TST21	004354	622	625#			
TST22	004450	634	637#			
TST23	004522	643	646#			
TST24	004602	653	656#			
TST25	004676	669#				
TST26	005030	684	691#			
TST27	005126	700	704#			
TST3	002710	445#				
TST30	005244	717	721#			
TST31	005324	728	735#			
TST32	005420	744	748#			
TST33	005514	757	760#			
TST34	005610	769	772#			
TST35	005704	781	783#			
TST36	006134	797	804	811	816	819#
TST37	006272	835	838	842#		
TST4	003010	460#				
TST40	006410	858#				
TST41	006534	876#				
TST42	006622	884	887#			
TST43	006726	902#				
TST44	007012	910	913#			
TST45	007064	919	922#			
TST46	007310	952#				
TST47	007534	982#				
TST5	003066	467	470#			
TST50	007750	1012#				
TST51	010044	1021	1024#			
TST52	010154	1035	1038#			
TST53	010402	1071#				
TST54	010476	1080	1083#			
TST55	010572	1092	1096#			
TST56	010702	1107	1111#			

937	941*	942	944*	965*	967	971*	972	974*	995*	997	1001*	1002
1004*	1020*	1033*	1034	1056*	1061*	1077*	1078*	1079	1089*	1090*	1091	1104*
1105*	1106	1117*	1118*	1119	1128*	1129*	1130	1143*	1144*	1145	1156*	1157
1167*	1168	1173*	1179*	1180	1190*	1191*	1192	1202*	1203*	1204	1217*	1218*
1219	1230*	1231*	1232	1243*	1244*	1245	1258*	1259*	1260	1271*	1277*	1292*
1293	1312*	1316*	1321	1336*	1337*	1338	1349*	1350*	1351	1364*	1365*	1366
1379*	1380*	1381	1395*	1396*	1397	1408*	1409*	1410	1427*	1428*	1429	1440*
1441*	1442	1458*	1459*	1460	1474*	1475	1488*	1489*	1490	1505*	1506*	1507
1519*	1530*	1531*	1532	1548*	1549*	1550	1561*	1562*	1563	1581*	1591*	1592*
1593	1611*	1621*	1622*	1623	1641*	1651*	1652*	1653	1671*	1681*	1682*	1683
1701*	1711*	1712*	1713	1731*	1746*	1747	1759*	1760	1773*	1774	1787*	1788
1804*	1805	1824*	1825	1848*	1849	1866*	1867	1884*	1897*	1898	1918*	1919
1942*	1943	1961*	1962	1980*	1981	1996*	1997	2012*	2013	2028*	2029	2047*
2048	2058*	2059	2077*	2082*	2094*	2100*	2111*	2112	2125*	2131*	2146*	2153*
2164*	2179*	2180*	2186*	2203*	2204*	2208*	2209*	2210	2225*	2226	2243*	2244
2249*	2250	2267*	2268	2273*	2274	2302*	2306*	2311	2319*	2326*	2327	2356*
2360*	2365	2373*	2374	2389*	2390*	2391	2500					

\$BELL 001172
 \$CDW1 001262
 \$CDW2 001264
 \$CHARC 027056
 \$CKSWR= ***** U
 \$CMTAG 001100
 \$CM1 = 000002
 \$CM2 = 000004
 \$CM3 = 000002
 \$CPUOP 001230
 \$CRLF 001177
 \$DBLK 026252
 \$DDWO 001266
 \$DEVCT 001212
 \$DEVMI 001260
 \$DOAGN 024430
 \$DTBL 026242
 \$ENDAD 024420
 \$ENDCT 024366
 \$ENDMG 024437
 \$ENULL 024434
 \$ENV 001222
 \$ENVM 001223
 \$EOP 024332
 \$EOPCT 024760
 \$ERFLG 001103
 \$ERMAX 001115
 \$ERROR 025252
 \$ERRPC 001116
 \$ERRTB 001270
 \$ERRTY 025454
 \$ERTTL 001112
 \$ESCAP 001170
 \$ETABL 001222
 \$ETEND 001270
 \$FATAL 001204
 \$FFLG 025250
 \$FILLC 001156
 \$FILLS 001155

32#
 32#
 32#
 2448#*
 2451
 32# 388
 32#
 32#
 32#
 32#
 32#
 2431 2431 2434 2448
 2438#
 32#
 32#
 2403#
 2438#
 27 2403# 2431
 388 2403#
 2403#
 2403#
 32# 2430 2431 2448
 32# 388 2430 2448
 2400 2403#
 388* 2403#
 32# 2429* 2431*
 32# 388* 2429*
 388 2431#
 32# 2431* 2434 2500
 32# 2434
 2431 2434#
 32# 2431*
 32# 388* 2429* 2431
 30 32#
 32# 2430*
 2430#*
 32# 2448
 32# 2448

\$GDADR 001120
 \$GDDAT 001124

32#													
32#	434*	435	439	442*	448*	452	456*	457*	462*	466	473*	474	
477	480*	487*	488*	492	502*	503	506	509*	516*	517*	521	532*	
536	539*	547*	551	554*	555*	561*	562	564	567*	574*	575*	578	
588*	589	592	595*	600*	609*	618*	627*	633	639*	648*	659*	660	
663	666*	671*	676	679*	683	693*	695	697	701*	709*	713	716*	
724*	740*	743	753*	756	765*	768	777*	780	792*	794	799*	801	
806*	808	813*	815	827*	830*	832	837	845*	847	852	855*	862*	
863*	865	870	878*	890*	895	899*	904*	909	915*	926*	927*	928	
930*	937	939*	942	948*	949	956*	957*	958	960*	967	969*	972	
978*	979	985*	986*	987	989*	997	999*	1002	1008*	1009	1016*	1032*	
1034	1046*	1047*	1048	1050*	1066*	1067	1074*	1079	1086*	1091	1101*	1106	
1114*	1119	1125*	1130	1139*	1145	1152*	1153	1163*	1164	1175*	1180	1187*	
1192	1199*	1204	1215*	1219	1227*	1232	1240*	1245	1255*	1260	1270*	1276*	
1286*	1293	1300*	1321	1333*	1338	1346*	1351	1363*	1366	1374*	1376	1381	
1384*	1390*	1397	1405*	1410	1414*	1422*	1429	1437*	1442	1445*	1453*	1460	
1470*	1475	1485*	1490	1494*	1504*	1507	1517*	1527*	1532	1535*	1543*	1550	
1557*	1563	1566*	1573*	1587*	1593	1596*	1603*	1617*	1623	1626*	1633*	1647*	
1653	1656*	1663*	1677*	1683	1686*	1693*	1707*	1713	1716*	1723*	1744*	1747	
1754*	1756	1760	1768*	1770	1774	1780*	1784	1788	1792*	1805	1810*	1811*	
1812*	1825	1830*	1835*	1841*	1849	1854*	1861*	1863	1867	1870*	1872*	1879*	
1898	1903*	1904*	1905*	1919	1924*	1929*	1935*	1943	1948*	1955*	1958	1962	
1965*	1967*	1977*	1981	1993*	1997	2009*	2013	2025*	2029	2044*	2048	2056*	
2059	2076*	2087*	2097*	2109*	2112	2121*	2122	2130*	2137*	2141	2152*	2159*	
2172*	2198*	2210	2220*	2226	2233*	2244	2248*	2250	2257*	2268	2272*	2274	
2284*	2311	2318*	2322*	2327	2338*	2365	2369*	2374	2381*	2391	2394*	2395	

\$GET42 024410
 \$GTSWR= ***** U
 \$HD = 000000
 \$HIBTS 001000
 \$ICNT 001104
 \$ILLUP 026426
 \$INTAG 001135
 \$ITEMB 001114
 \$LF 001200
 \$LFLG 025247
 \$LPADR 001106
 \$LPERR 001110

2403#													
2451													
13													
30#													
32#	2429*												
2441#													
32#													
32#	2431*	2434											
32#	2431	2448											
2430#*													
32#	388*	2429*											
32#	388*	416*	432*	446*	471*	484*	500*	513*	529*	544*	559*	571*	
587*	657*	705*	844*	860*	888*	925*	955*	984*	1041*	1373*	1403*	1435*	
1482*	1525*	1556*	1586*	1616*	1646*	1676*	1706*	1796*	1816*	1844*	1858*	1889*	
1910*	1939*	1953*	2429*	2431									
404	1780	2445#*											

\$LSTAD 026576
 \$MADR1 001234
 \$MADR2 001240
 \$MADR3 001244
 \$MADR4 001250
 \$MAIL 001202

30	32#	388	415	431	445	460	470	483	499	512	528	543	
558	570	585	598	607	616	625	637	646	656	669	691	704	
721	735	748	760	772	783	819	842	858	876	887	902	913	
922	952	982	1012	1024	1038	1071	1083	1096	1111	1122	1135	1150	
1161	1172	1184	1196	1209	1224	1237	1250	1265	1281	1330	1343	1356	
1371	1387	1401	1419	1433	1450	1465	1481	1497	1512	1523	1540	1554	
1571	1584	1601	1614	1631	1644	1661	1674	1691	1704	1721	1735	1751	
1765	1779	1795	1815	1838	1857	1875	1887	1908	1932	1951	1970	1986	
2002	2018	2035	2064	2085	2104	2116	2134	2156	2167	2190	2216	2230	

	2254	2278	2332	2379	2398	2429	2431	2448										
\$MAMS1 001232	32#																	
\$MAMS2 001236	32#																	
\$MAMS3 001242	32#																	
\$MAMS4 001246	32#																	
\$MBADR 001002	30#																	
\$MFLG 025246	2430#*																	
\$MSGAD 001216	32#	2430*																
\$MSGLG 001220	32#	2430*																
\$MSGTY 001202	32#	2430*																
\$MTYP1 001233	32#																	
\$MTYP2 001237	32#																	
\$MTYP3 001243	32#																	
\$MTYP4 001247	32#																	
\$MXCNT 025002	2429#																	
\$NULL 001154	32#	2448																
\$NWTST= 000001	415#	431#	445#	460#	470#	483#	499#	512#	528#	543#	558#	570#	585#					
	598#	607#	616#	625#	637#	646#	656#	669#	691#	704#	721#	735#	748#					
	760#	772#	783#	819#	842#	858#	876#	887#	902#	913#	922#	952#	982#					
	1012#	1024#	1038#	1071#	1083#	1096#	1111#	1122#	1135#	1150#	1161#	1172#	1184#					
	1196#	1209#	1224#	1237#	1250#	1265#	1281#	1297#	1330#	1343#	1356#	1371#	1387#					
	1401#	1419#	1433#	1450#	1465#	1481#	1497#	1512#	1523#	1540#	1554#	1571#	1584#					
	1601#	1614#	1631#	1644#	1661#	1674#	1691#	1704#	1721#	1735#	1751#	1765#	1779#					
	1795#	1815#	1838#	1857#	1875#	1887#	1908#	1932#	1951#	1970#	1986#	2002#	2018#					
	2035#	2064#	2085#	2104#	2116#	2134#	2156#	2167#	2190#	2216#	2230#	2254#	2278#					
	2332#	2379#	2398#															
\$OCNT 026032	2437#*																	
\$OMODE 026034	2437#*																	
\$OVER 024766	2429#																	
\$PASS 001210	32#	388*	2403*	2429														
\$PASTM 001006	30#																	
\$PWRAD 026422	2441#																	
\$PWRDN 026262	388	2441#																
\$PWRMG 026416	2441#																	
\$PWRUP 026334	2441#																	
\$QUES 001176	32#	2431	2448															
\$RDCHR= ***** U	2451																	
\$RDDEC= ***** U	2451																	
\$RDLIN= ***** U	2451																	
\$RDOCT= ***** U	2451																	
\$REGAD 001160	32#																	
\$REGO 001162	32#																	
\$REG1 001164	32#																	
\$RTNAD 024432	2403#																	
\$R2A = ***** U	2451																	
\$SAVRE= ***** U	2451																	
\$SAVR6 026432	2441#*																	
\$SCOPE 024526	388	2429#																
\$SETUP= 000037	387#	388	2403	2429	2431													
\$SIZE 026502	403	2445#																
\$STUP = 177777	387#																	
\$SVLAD 024732	2429#																	
\$SVPC = 000210	27#																	
\$SWR = 167400	5#	13	22	32	388	415	431	445	460	470	485	499	512					
	528	543	558	570	585	598	607	616	625	637	646	656	669					
	691	704	721	735	748	760	772	783	819	842	858	876	887					

COMMEN	17#														
DELAY	347#														
DELAY1	351#														
ENDCOM	17#														
ERROR	17#	427	441	454	468	479	494	508	523	538	553	566	580	594	605
	614	623	635	644	654	665	678	685	699	715	729	745	758	770	782
	796	803	810	817	834	854	872	885	897	911	920	940	945	970	975
	1000	1005	1022	1036	1058	1063	1081	1093	1108	1121	1132	1147	1159	1170	1182
	1194	1206	1221	1234	1247	1262	1273	1279	1295	1313	1319	1323	1340	1353	1368
	1383	1399	1413	1431	1444	1462	1477	1492	1509	1521	1534	1552	1565	1582	1595
	1612	1625	1642	1655	1672	1685	1702	1715	1732	1749	1762	1776	1790	1807	1827
	1851	1869	1885	1900	1921	1945	1964	1983	1999	2015	2031	2050	2061	2079	2084
	2095	2101	2114	2126	2132	2147	2154	2165	2181	2187	2205	2212	2228	2246	2252
	2270	2276	2303	2309	2313	2320	2329	2357	2363	2367	2376	2393			
ESCAPE	17#														
GETPRI	17#	2445													
GETSWR	17#														
MAINT1	338#	694	706	723											
MAINT2	341#	737	750	762	774	785	821	843	859	923	953	983	1025	1039	1072
	1084	1097	1112	1123	1136	1151	1162	1173	1185	1197	1210	1225	1238	1251	1267
	1331	1344	1358	1372	1388	1402	1420	1434	1451	1499	1524	1541	1555	1572	1585
	1602	1615	1632	1645	1662	1675	1692	1705	1722	1752	1766	1781	1797	1817	1839
	1859	1876	1888	1909	1933	1952	1972	1988	2004	2020	2037	2067	2117	2135	2168
	2191	2380													
MULT	17#														
NEWTST	17#	415	431	445	460	470	483	499	512	528	543	558	570	585	598
	607	616	625	637	646	656	669	691	704	721	735	748	760	772	783
	819	842	858	876	887	902	913	922	952	982	1012	1024	1038	1071	1083
	1096	1111	1122	1135	1150	1161	1172	1184	1196	1209	1224	1237	1250	1265	1281
	1297	1330	1343	1356	1371	1387	1401	1419	1433	1450	1465	1481	1497	1512	1523
	1540	1554	1571	1584	1601	1614	1631	1644	1661	1674	1691	1704	1721	1735	1751
	1765	1779	1795	1815	1838	1857	1875	1887	1908	1932	1951	1970	1986	2002	2018
	2035	2064	2085	2104	2116	2134	2156	2167	2190	2216	2230	2254	2278	2332	2379
	2398														
POP	17#	2430	2438	2441											
PUSH	17#	2430	2438	2441											
REPORT	17#														
RESUME	344#	805	812	839	1214	1256	1757	1771	1785	1978	1994	2010	2026	2045	2053
	2055	2070	2072												
SCOPE	17#	415	431	445	460	470	483	499	512	528	543	558	570	585	598
	607	616	625	637	646	656	669	691	704	721	735	748	760	772	783
	819	842	858	876	887	902	913	922	952	982	1012	1024	1038	1071	1083
	1096	1111	1122	1135	1150	1161	1172	1184	1196	1209	1224	1237	1250	1265	1281
	1297	1330	1343	1356	1371	1387	1401	1419	1433	1450	1465	1481	1497	1512	1523
	1540	1554	1571	1584	1601	1614	1631	1644	1661	1674	1691	1704	1721	1735	1751
	1765	1779	1795	1815	1838	1857	1875	1887	1908	1932	1951	1970	1986	2002	2018
	2035	2064	2085	2104	2116	2134	2156	2167	2190	2216	2230	2254	2278	2332	2379
	2398	2403													
SETPRI	17#														
SETTRA	2451#														
SETUP	17#	388													
SKIP	17#	467	604	613	622	634	643	653	684	698	700	717	728	744	757
	769	781	797	804	811	816	835	838	840	884	910	919	1021	1035	1080
	1092	1107	1120	1131	1146	1158	1169	1181	1193	1205	1220	1233	1246	1261	1272
	1278	1294	1339	1352	1367	1398	1412	1415	1430	1446	1461	1476	1508	1520	1536
	1551	1567	1580	1597	1610	1627	1640	1657	1670	1687	1700	1717	1730	1748	1761

.\$EOP	7#	2403	
.\$ERRO	7#	2431	
.\$ERRT	9#	2434	
.\$SPARM	8#		
.\$POWE	8#	2441	
.\$READ	8#		
.\$SAVE	8#		
.\$SCOP	8#	2429	
.\$SIZE	9#	2445	
.\$SPAC	8#		
.\$SWDO	8#		
.\$STRAP	8#	2451	
.\$TYPD	9#	2438	
.\$TYPE	7#	8#	2448
.\$TYPO	7#	2437	

. ABS. 031100 000 CON RW ABS GBL I

ERRORS DETECTED: 0

CZVSAB,CZVSAB/CRF=CZVSAB
RUN-TIME: 36 23 2 SECONDS
RUN-TIME RATIO: 134/62=2.1
CORE USED: 26K (51 PAGES)