IDENTIFICATION

Product code: ZZ-ENKAB VERSION 1.5

Product name: VAX-11/730 CRD MICRO-MONITOR

Product date: 27-AUG-1982

Maintainer: BASE SYSTEMS DIAGNOSTIC ENGINEERING

Table of Contents

1 ABSTRACT

This program is the micro diagnostic monitor that is used during VAX-11/730 Customer Runable Diagnostics AUTO mode. It is similar to the standard Micmon, except that contained within the program is a pre-determined string of commands to execute particular micro diagnostic tests. This program also contains special progress and error messages that are displayed to the CRD user.

For further information about the program, and for complete documentation concering CRD AUTO mode, consult the ENSAB.DOC documentation file.

2 MAINTENANCE HISTORY

DATE	VERSION	DESCRIPTION
6-JUL-82	01.00	Initial release.
27-AUG-82	01.10	Changes for IDC Part II Version 2.0
23-MAY-83	01.20	Changes to support LCN and KERNEL 11/730 systems.
25-JUL-83	01.30	Changes to support LCN 11/725 name.
16-JAN-84	01.40	Changes for systems having TSU05 as drive 1
14-JUN-84	01.50	Changes to the Title. It will now be known as microdiagnostic section.

Object Module Synopsis

Module Name	Ident	Bytes	File	Creation Date	Creator
.MAIN.	0	8189	DRB0:[BALL.ENKAB.RELEASE]ENKAB.OBJ;1	15-JUN-1984 12:22	VAX-11 Macro V03-01

Program Section Synopsis

Psect Name	Module Name	Base	End	Length	Align	Attributes
MICMON		00004C00	00006BFC	00001FFD (8189.)	BYTE 0	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	.MAIN.	00004C00	00006BFC	00001FFD (8189.)	BYTE 0	

Symbols By Name

Symbol	Value	Symbol	Value	Symbol	Value	Symbol	Value
APT	00004C38-R	GCVECT	00004129	READ_LS	00004D02-R		
APT_PASS_CNT	00004C18-R	HEX_3	00000003	READ_WCS	00004CE7-R		
CC_ADDR	000000FE	HEX_BUF	00004C5B-R	SET_FOR_CONT	00004D2C-R		
CLEAR_CPU_ATT	00004D2F-R	INC_WORD	00004D08-R	SHIFTER	00004CE1-R		
CNTLC_TYPED	00004C39-R	LD_UPC	00004D17-R	SHIFTER1	00004D32-R		
COMPARE	00004CCF-R	LF	0000000A	SHIFT_L_2901	00004CF6-R		
CONTINUE	00004C3A-R	LOAD_CSR	00004CF0-R	SHIFT_PNT	00004C68-R		
CON_ADD_DAT	00004C3B-R	LOAD_UPC	00004CF3-R	SHIFT_R_2901	00004CF9-R		
CON_BEGIN_TEST	00004D23-R	MAKE_XD	00004D0B-R	SKIP_TEST	00004C6A-R		
CON_ERROR	00004D20-R	MCPU_A	00004D40-R	SPACE	00004CDB-R		
CON_ERR_NUM	00004C3F-R	MICRO_STEP_CPU	00004D38-R	UPC_VALUE	00004CC2-R		
CON_EXP_ADD	00004C44-R	MOVER	00004CD2-R	WCS_ADDRESS	00004C6C-R		
CON_EXP_DAT	00004C40-R	MOVER_3	00004D1A-R	WCS_BAD_PARITY	00004C6E-R		
CON_MOD_DAT	00004C46-R	MOVER_4	00004D11-R	WCS_READ	00004D26-R		
CON_MSK_DAT	00004C48-R	MSK_ERROR	00004D1D-R	WCS_SHIFT	00004C6B-R		
CON_REC_ADD	00004C53-R	MWCS_A	00004D3B-R	WCS_VALUE	00004CB4-R		
CON_REC_DAT	00004C4F-R	NA_EXP_REC	00004C5F-R	WCS_VAL_SHIFT	00004CB3-R		
CR	0000000D	NA_OTHER	00004C60-R	WCS_WRITE	00004D29-R		
CRLF	00004CDE-R	NOP_CSR	00004CE4-R	WRITE_DATA_32	00004CFF-R		
CSR_SHIFT	00004CB7-R	NOP_INST	00004C72-R	WRITE_LS	00004D05-R		
CSR_VALUE	00004CB8-R	OS_ADD	0000007C	WRITE_TAB_PNT	00004C6F-R		
DATA0	00004CA9-R	OVERLAY_RAM	00004C03-R	WRITE_WCS	00004CEA-R		
DATA1	00004CAA-R	OVERLAY_WCS	00004CC6-R	XD_ADDRS	00004C71-R		
DATA2	00004CAB-R	PARITY_JUMP	00004C61-R	X_CLR_PAGE	00004C7D-R		
DATA3	00004CAC-R	PARITY_ON	00004C64-R	X_CLR_WR0	00004CA1-R		
DATA_SHIFT	00004CA8-R	PERFORM_CSR	00004CED-R	X_COM_WR0	00004CA5-R		
DECR_WORD	00004D35-R	POWER_BRANCH	00004C65-R	X_MOV_WRWR	00004C9E-R		
DOLOOP	00004C55-R	PRINT	00004CD5-R	X_ROT_L	00004C79-R		
EOS_FLG	00004C56-R	PRINT_HEX	00004CD8-R	X_SET_PAGE	00004C81-R		
ERROR_CON	00004C57-R	PRINT_HEX_1	00004D14-R	X_SHIFT_L	00004C75-R		
FLAGS	00004C5A-R	PRINT_STRING	00004D0E-R	X_SHIFT_R	00004C85-R		
I_WORD	00004C58-R	READ_DATA_32	00004CFC-R				

Symbol	Value	Symbol	Value	Symbol	Value	Symbol	Value

Key for special characters above:

*	- Undefined
U	- Universal
R	- Relocatable
X	- External

Image Synopsis

```
Virtual memory allocated:                00004C00 00006BFF 00002000 (8192. bytes, 16. pages)
Stack size:                                     0. pages
Image binary virtual block limits:              1.       16. (   16. blocks)
Image name and identification:           ENKAB 0
Number of files:                                1.
Number of modules:                              1.
Number of program sections:                     2.
Number of global symbols:                      95.
Number of image sections:                       1.
Image type:                              SYSTEM.
Map format:                              DEFAULT in file DRB0:[BALL.ENKAB.RELEASE]ENKAB.MAP;2
Estimated map length:                    14. blocks
```

Link Run Statistics

Performance Indicators	Page Faults	CPU Time	Elapsed Time
Command processing:	102	00:00:00.18	00:00:00.32
Pass 1:	33	00:00:00.14	00:00:00.65
Allocation/Relocation:	6	00:00:00.13	00:00:00.33
Pass 2:	16	00:00:02.04	00:00:06.68
Map data after object module synopsis:	9	00:00:00.26	00:00:00.88
Symbol table output:	0	00:00:00.01	00:00:00.20
Total run values:	166	00:00:02.76	00:00:09.06

```
Using a working set limited to 500 pages and 30 pages of data storage (excluding image)

Total number object records read (both passes):   132
    of which 0 were in libraries and 2 were DEBUG data records containing 39 bytes

Number of modules extracted explicitly              = 0
    with 0 extracted to resolve undefined symbols

0 library searches were for symbols not in the library searched

A total of 0 global symbol table records was written

LINK/SYST=%X4C00/MAP=ENKAB/EXE=ENKAB.EXE ENKAB.OBJ
```

```
                        2 .TITLE  "ENKAB  8085 based micro-monitor for NEBULA CRD Rev 1.5"
                        3 ;
                        4 ;
                        5 ;                               COPYRIGHT (c) 1982 BY
                        6 ;                     DIGITAL EQUIPMENT CORPORATION, MAYNARD,
                        7 ;                      MASSACHUSETTS.  ALL RIGHTS RESERVED.
                        8 ;
                        9 ; THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
                       10 ; ONLY  IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE INCLUSION
                       11 ; OF THE ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE OR ANY OTHER COPIES THEREOF
                       12 ; MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON.  NO
                       13 ; TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY TRANSFERRED.
                       14 ;
                       15 ; THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE, AND
                       16 ; SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.
                       17 ;
                       18 ; DIGITAL  ASSUMES  NO  RESPONSIBILITY  FOR  THE  USE OR RELIABILITY OF ITS
                       19 ; SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.
                       20 ;
                       21
                       22 ;++
                       23 ; FACILITY:      VAX-11/730 CUSTOMER RUNABLE MICRO-DIAGNOSTIC MONITOR
                       24 ;
                       25 ; ABSTRACT:      This 8085 based program monitors and controls the execution
                       26 ;                of both 8085 based micro-diagnostics and WCS based micro-
                       27 ;                diagnostics in the Customer Runable Diagnostics environment.
                       28 ;
                       29 ; ENVIRONMENT:   Standalone
                       30 ;
                       31 ; AUTHOR:        David Mayo                6-JUL-82        VERSION 1.0
                       32 ;
                       33 ; MODIFIED BY:
                       34 ;                David Mayo                27-AUG-82       Version 1.1
                       35 ;                Changes for iDC Part II version 2.0
                       36 ;
                       37 ;                Peter Green               3-MAR-83        VERSION 1.2
                       38 ;                Changes for LCN and Kernel systems.
                       39 ;
                       40 ;
                       41 ;                Peter Green               5-JUL-83        VERSION 1.3
                       42 ;                Changes for LCN syntax.
                       43 ;
                       44 ;
                       45 ;                Darryl L. Ball            28-NOV-83       VERSION 1.4
                       46 ;                Changes for 1505 system configurations.
                       47 ;
                       48 ;
                       49 ;                Darryl L. Ball            14-Jun-84       VERSION 1.5
                       50 ;                Added a new header for distinction between the micro
                       51 ;                and macro portions of CRD.
                       52 ;
                       53 ;--
                       59
                       60
                       61
                       62
                       63
```

```
0000     64
0000     65
0000     66
0000     67
0000     68
0000     69
0000     70
0000     71
0000     72
0000     73
0000     74
0000     75
0000     76
0000     77
0000     78
0000     79
0000     80
0000     81
0000     82
0000     83
0000     84
0000     85
0000     86
0000     87
0000     88 ;8085 SOURCE CODE   CRD MICRO DIAGNOSTIC MONITOR
0000     89
0000     90
0000     91 ;*****************************************************************************
0000     92 ;
0000     93 ;       NOTE:  THE LAST ADDRESS OF THIS PROGRAM MUST NOT EXCEED 6FFF.  THE
0000     94 ;       8085 MICRO-DIAGNOSTIC OVERLAYS LOAD AT ADDRESS 7000 HEX!!!
0000     95 ;
0000     96 ;*****************************************************************************
0000     97
0000     98
0000     99 ;*****************************************************************************
0000    100 ;
0000    101 ;       LIKE THE STANDARD MICRO-DIAGNOSTIC MONITOR, THIS PROGRAM WAS DESIGNED
0000    102 ;  USING PROGRAM DESIGN LANGUAGE (PDL).  SINCE PDL IS A HIGH LEVEL LANGUAGE
0000    103 ;  THE DECISION TO WRITE THIS MONITOR USING MACROS WAS MADE.  THE MACROS USE
0000    104 ;  MANY OF THE REGISTERS, SO USE OF REGISTERS FOR PASSING PARAMETERS IS
0000    105 ;  DANGEROUS AT BEST.  ALSO THERE ARE DIFFERENCES BETWEEN THE PDL AND THE
0000    106 ;  PROGRAM.  PDL IS MEANT AS A DESIGNERS LANGUAGE AND IS USED TO PASS
0000    107 ;  INFORMATION TO THE IMPLIMENTOR.  THE USE OF THE MACROS HELPS KEEP THE TWO
0000    108 ;  SIMILAR, BUT IMPLEMENTATION IN A SLIGHTLY DIFFERENT MANNER IS SOMETIMES
0000    109 ;  NECESSARY.
0000    110 ;
0000    111 ;*****************************************************************************
0000    112
0000    113 ;*****************************************************************************
0000    114 ;  IMPLEMENTATION NOTES:
0000    115 ;
0000    116 ;       THIS PROGRAM IS CALLED "CRD MICMON" (CUSTOMER RUNABLE DIAGNOSTICS
0000    117 ;  MICRO MONITOR).  CRD MICMON IS USED TO RUN MICRO DIAGNOSTICS FOR THE
0000    118 ;  MICRO DIAGNOSTICS.  CRD MICMON RUNS TWO DIFFERENT FORMS OF DIAGNOSTICS:
0000    119 ;       A) DIAGNOSTICS WRITTEN IN CPU MICRO CODE.
0000    120 ;       B) DIAGNOSTICS WRITTEN IN 8085 CODE.
```

```
                        0000    121 ;
                        0000    122 ;       CRD MICMON PASSES CONTROL TO 8085 DIAGNOSTICS BY CALLING THEM
                        0000    123 ;  AS IF THEY WERE SUBROUTINES OF THE MICRO MONITOR.
                        0000    124 ;
                        0000    125 ;       CRD MICMON CAUSES CPU MICRO CODE DIAGNOSTICS TO BE EXECUTED. THIS IS
                        0000    126 ;  DONE BY STARTING THE CPU AT SPECIFIC ADDRESSES. (THESE VARIOUS ADDRESSES
                        0000    127 ;  ARE CONTAINED IN THE CONSTANTS LISTED IN THE CONSTANT SECTION) THE CPU
                        0000    128 ;  MICRO DIAGNOSTICS RUN CONCURRENT TO THE 8085.  THE CPU MICRO CODED
                        0000    129 ;  DIAGNOSTIC COMMUNICATES WITH MICMON BY SETTING THE CPU ATTENTION
                        0000    130 ;  SIGNAL (HARDWARE) AND CONVEYING IT'S MESSAGE THROUGH THE USE OF A
                        0000    131 ;  32 BIT COMMAND AND STATUS WORD.
                        0000    132 ;
                        0000    133 ;       CRD MICMON ISSUES A SET OF PRE-DETERMINED COMMANDS AUTOMATICALLY,
                        0000    134 ;  RATHER THAN TAKING INPUT FROM THE CONSOLE TERMINAL.
                        0000    135 ;
                        0000    136 ;  VARIABLE NAMES ENDING IN _A ARE ASCII TEXT
                        0000    137 ;
                        0000    138 ;  VARIABLE NAMES ENDING IN _R ARE GLOBAL SUBROUTINES
                        0000    139 ;
                        0000    140 ;*****************************************************************************
                        0000    141
                        0000    142
                        0000    143 ;*****************************************************************************
                        0000    144 ;
                        0000    145 ; DATA SECTION
                        0000    146 ;
                        0000    147 ;*****************************************************************************
                        0000    148
                        0000    149                     .PSECT  MICMON
                        0000    150
00000000                0000    151 HEAD                = 0                             ;CALCULATE NO OFFSET IN ASSEMBLY
                        0000    152
                        0000    153 ;ENTRY POINT TO THE PROGRAM IS FIRST ADDRESS OF OVERLAY
                        0000    154
                        0000    155                     JMP     START_UP
                  C3    0000                    .BYTE   ^0303
                'CD2'   0001                    .WORD   <START_UP-HEAD>
                        0003    156
                        0003    157
                        0003    158                     EQUATE                          ;REGISTER EQUATE MACRO
                        0003                    .LIST   ME
                        0003        ;
                        0003        ; 8085 REGISTER DEFINITIONS
                        0003        ;
                        0003
                        0003        ;
                        0003        ; REGISTER PAIRS
                        0003        ;
00000000                0003            B       =       0
00000002                0003            D       =       2
00000004                0003            H       =       4
00000006                0003            $SP     =       6
00000006                0003            $PSW    =       6
                        0003        ;
                        0003        ; NON REGISTER PAIRS
                        0003        ;
00000001                0003            C       =       1
```

```
00000003  0003              C       =       3
00000005  0003              L       =       5
00000006  0003              M       =       6
00000007  0003              A       =       7
          0003
          0003
          0003        ;  8085 ASSEMBLY LANGUAGE DEFINITION MACRO'S
          0003        ;
          0003
          0003              .MCALL  MOV,MVI,LXI,STAX,LDAX,STA,LDA,SHLD,LHLD,XCHG
          0003              .MCALL  PUSH,POP,XTHL,SPHL,JMP,JC,JNC,JZ,JNZ,JP,JM,JPE,JPO,PCHL
          0003              .MCALL  CALL,CC,CNC,CZ,CNZ,CP,CM,CPE,CPO
          0003              .MCALL  RET,RC,RNC,RZ,RNZ,RP,RM,RPE,RPO,RST
          0003              .MCALL  INR,DCR,INX,DCX,ADD,ADC,ADI,ACI,DAD,SUB,SBB,SUI,SBI
          0003              .MCALL  ANA,XRA,ORA,CMP,ADI,XRI,ORI,CPI
          0003              .MCALL  RLC,RRC,RAL,RAR,CMA,STC,CMC,NOP,HLT,RIM,SIM
          0003              .MCALL  IN,OUT
          0003    159
          0003    160
          0003    161 ;*****************************************************************************
          0003    162 ;
          0003    163 ;  THE FOLLOWING BLOCK OF MACROS DEFINE THE HARDWARE INTERFACE TO
          0003    164 ;  THE 8085 SYSTEM.
          0003    165 ;
          0003    166 ;*****************************************************************************
          0003    167
          0003    168
00000000  0003    169 UPC14A          =<^X00>                 ;(RO) UPC BIT 14 ASSERTED HIGH <MSB>
          0003    170
00000001  0003    171 MISC2           =<^X01>
00000001  0003    172 BOOTF           =<^X01>                 ;(RO) BOOT FLAG. ASSERTED LOW. <LSB>
          0003    173
00000002  0003    174 DCLO            =<^X02>                 ;(RO) DC LOW FLAG ASSERTED HIGH<MSB>
00000002  0003    175 HLTFLG          =<^X02>                 ;(RO) HALT FLAG ASSERTED LOW   <LSB>
          0003    176
00000003  0003    177 READJ1          =<^X03>                 ;(RO) JUMPER J1 FLAG ASS. HIGH <MSB>
00000003  0003    178 ALLOWP          =<^X03>                 ;(RO) ENABLE CONTROL P ASS LOW <LSB>
          0003    179
00000004  0003    180 READJ2          =<^X04>                 ;(RO) BIT 07 EQUALS J2 <MSB>
00000004  0003    181 APTFLG          =<^X04>                 ;(RO) APT PRESENT FLAG <LSB>
          0003    182
00000005  0003    183 AUTO            =<^X05>                 ;(RO) AUTO TEST FLAG ASS LOW <LSB>
          0003    184
00000006  0003    185 REMOTE          =<^X06>                 ;(RO) REMOTE FLAG ASS LOW <LSB>
          0003    186
00000007  0003    187 OKV15           =<^X07>                 ;(RO) 15 VOLTS OK FLAG ASS LOW <MSB>
00000007  0003    188 BOOTEN          =<^X07>                 ;(RO) BOOT ENABLE FLAG ASS LOW <LSB>
          0003    189
00000008  0003    190 WCSB0           =<^X08>                 ;(WO) WCS WRITE DATA REG, BYTE 0
00000009  0003    191 WCSB1           =<^X09>                 ;(WO) WCS WRITE DATA REG, BYTE 1
0000000A  0003    192 WCSB2           =<^X0A>                 ;(WO) WCS WRITE DATA REG, BYTE 2
          0003    193
0000001C  0003    194 ITDB            =<^X1C>                 ;(R/W) TIMER DATA REGISTER
0000001D  0003    195 ITCSR           =<^X1D>                 ;(R/W) TIMER STATUS REGISTER
          0003    196
00000020  0003    197 CLRCLK          =<^X20>                 ;(WO) CLEAR CPU CLOCK RUN
00000021  0003    198 SETCLK          =<^X21>                 ;(WO) SET CPU CLOCK RUN
```

```
                    0003   199
00000022            0003   200 CLRSS       =<^X22>          ;(WO) CLEAR CLOCK SINGLE STEP
00000023            0003   201 SETSS       =<^X23>          ;(WO) SET CLOCK SINGLE STEP
                    0003   202
00000024            0003   203 CLRCSK      =<^X24>          ;(WO) CLEAR CSR CLOCK
00000025            0003   204 SETCSK      =<^X25>          ;(WO) SET CSR CLOCK
                    0003   205
00000026            0003   206 CLRUPK      =<^X26>          ;(WO) CLEAR THE UPC CLOCK
00000027            0003   207 SETUPK      =<^X27>          ;(WO) SET THE UPC CLOCK
                    0003   208
00000028            0003   209 SETMCI      =<^X28>          ;(WO) SET MEMORY CONTROL INIT
00000029            0003   210 CLRMCI      =<^X29>          ;(WO) CLEAR MEMORY CONTROL INIT
                    0003   211
0000002A            0003   212 SETMCK      =<^X2A>          ;(WO) SET MEMORY CONTROL CLOCK
0000002B            0003   213 CLRMCK      =<^X2B>          ;(WO) CLEAR MEMORY CONTROL CLOCK
                    0003   214
0000002C            0003   215 CLRTMR      =<^X2C>          ;(WO) STOP TIMER
0000002D            0003   216 SETTMR      =<^X2D>          ;(WO) START TIMER COUNTING
                    0003   217
0000002E            0003   218 CLRBSY      =<^X2E>          ;(WO) CLEAR UNIBUS BUS BUSY
0000002F            0003   219 SETBSY      =<^X2F>          ;(WO) SET UNIBUS BUS BUSY
                    0003   220
00000030            0003   221 SETDCL      =<^X30>          ;(WO) SET DC LOW
00000031            0003   222 CLRDCL      =<^X31>          ;(WO) CLEAR DC LOW
                    0003   223
00000032            0003   224 SETACL      =<^X32>          ;(WO) SET AC LOW
00000033            0003   225 CLRACL      =<^X33>          ;(WO) CLEAR AC LOW
                    0003   226
00000034            0003   227 INITL       =<^X34>          ;(WO) CLEAR UNIBUS INIT
00000035            0003   228 INITH       =<^X35>          ;(WO) SET UNIBUS INIT
                    0003   229
00000036            0003   230 CLRWCK      =<^X36>          ;(WO) CLEAR WCS WRITE CLOCK
00000037            0003   231 SETWCK      =<^X37>          ;(WO) SET WCS WRITE CLOCK
                    0003   232
00000038            0003   233 CLRCSR      =<^X38>          ;(WO) CLEAR CSR SERIAL INPUT
00000039            0003   234 SETCSR      =<^X39>          ;(WO) SET CSR SERIAL INPUT
                    0003   235
0000003C            0003   236 CLRLIT      =<^X3C>          ;(WO) CLEAR THE RUN LIGHT
0000003D            0003   237 SETLIT      =<^X3D>          ;(WO) SET THE RUN LIGHT
                    0003   238
00000044            0003   239 TUDB        =<^X44>          ;(R/W) TU-58 DATA BUFFER
00000045            0003   240 TUSR        =<^X45>          ;(RO)  TU-58 STATUS REGISTER
00000046            0003   241 TUMODE      =<^X46>          ;(R/W) TU-58 MODE REGISTER 1 + 2
                    0003   242                             ;FIRST ACCESS GET MR1,SECOND GETS MR2
                    0003   243                             ;READING THE CR FORCES THE INTERNAL
                    0003   244                             ;POINTER BACK TO MR1
00000047            0003   245 TUCR        =<^X47>          ;(R/W) TU-58 COMMAND REGISTER
                    0003   246
00000048            0003   247 TTDB        =<^X48>          ;(R/W) TERMINAL DATA BUFFER
00000049            0003   248 TTSR        =<^X49>          ;(RO)  TERMINAL STATUS REGISTER
0000004A            0003   249 TTMODE      =<^X4A>          ;(R/W) TERMINAL MODE REGISTER 1 + 2
                    0003   250                             ;FIRST ACCESS GET MR1,SECOND GETS MR2
                    0003   251                             ;READING THE CR FORCES THE INTERNAL
                    0003   252                             ;POINTER BACK TO MR1
0000004B            0003   253 TTCR        =<^X4B>          ;(R/W) TERMINAL COMMAND REGISTER
                    0003   254
00000080            0003   255 READ        =<^X80>          ;(RO) CONSOLE READ REGISTER
```

```
                     0003   256
00000082             0003   257 CPUACK          =<^X82>                 ;(RO) ACKNOWLEDGE FROM CPU <LSB>
                     0003   258
00000083             0003   259 CPATTN          =<^X83>                 ;(RO) ATTENTION FROM CPU <LSB>
                     0003   260
00000084             0003   261 UPC14           =<^X84>                 ;(RO) UPC BIT 14        <LSB>
                     0003   262
00000085             0003   263 CSR07           =<^X85>                 ;(RO) CSR BIT 7         <LSB>
                     0003   264
00000086             0003   265 CSR15           =<^X86>                 ;(RO) CSR BIT 15        <LSB>
                     0003   266
00000087             0003   267 CSR23           =<^X87>                 ;(RO) CSR BIT 23        <LSB>
                     0003   268
000000A0             0003   269 ENBPAR          =<^XA0>                 ;(WO) ENABLE STALL ON PARITY ERROR
000000A1             0003   270 DISPAR          =<^XA1>                 ;(WO) DISABLE STALL ON PARITY ERROR
                     0003   271
000000A2             0003   272 VSHIFT          =<^XA2>                 ;(WO) SET THE V-BUS TO SHIFT MODE
000000A3             0003   273 VNORML          =<^XA3>                 ;(WO) SET THE V-BUS TO NORMAL MODE
                     0003   274
000000A4             0003   275 SETTIM          =<^XA4>                 ;(WO) SET INTERVAL TIMER INTERRUPT
000000A5             0003   276 CLRTIM          =<^XA5>                 ;(WO) CLEAR INTERVAL TIMER INTRPT.
                     0003   277
000000A6             0003   278 ENBMEM          =<^XA6>                 ;(WO) ENABLE MEMORY REFERENCES
000000A7             0003   279 DISMEM          =<^XA7>                 ;(WO) DISABLE MEMORY REFERENCES
                     0003   280
000000A8             0003   281 SETACK          =<^XA8>                 ;(WO) SET CONSOLE ACKNOWLEDGE
000000A9             0003   282 CLRACK          =<^XA9>                 ;(WO) CLEAR CONSOLE ACKNOWLEDGE
                     0003   283
                     0003   284 ;NOTE:  THE CONSOLE ACKNOWLEDGE SIGNAL IS ALSO USED AS THE UPC
                     0003   285 ;SERIAL INPUT.  WHEN USED THIS WAY THE SENSE IS INVERTED.  TO SIMPLIFY
                     0003   286 ;MATTERS THE NEXT TWO EQUATES ARE INCLUDED
                     0003   287
000000A8             0003   288 CLRUPC          =<^XA8>                 ;(WO) CLEAR THE V-BUS SERIAL INPUT
000000A9             0003   289 SETUPC          =<^XA9>                 ;(WO) SET THE V-BUS SERIAL INPUT
                     0003   290
000000AA             0003   291 SETATN          =<^XAA>                 ;(WO) SET CONSOLE ATTENTION
000000AB             0003   292 CLRATN          =<^XAB>                 ;(WO) CLEAR CONSOLE ATTENTION
                     0003   293
000000AC             0003   294 SETPFI          =<^XAC>                 ;(WO) SET POWER FAIL INTERRUPT
000000AD             0003   295 CLRPFI          =<^XAD>                 ;(WO) CLEAR POWER FAIL INTERRUPT
                     0003   296
000000AE             0003   297 SETHLT          =<^XAE>                 ;(WO) SET THE CONSOLE HALT BIT
000000AF             0003   298 CLRHLT          =<^XAF>                 ;(WO) CLEAR THE CONSOLE HALT BIT
                     0003   299
000000CC             0003   300 WRITE           =<^XCC>                 ;(WO) THE CONSOLE WRITE REGISTER
                     0003   301
000000EC             0003   302 YUSRD           =<^XEC>                 ;(RO) Y-BUS READ
                     0003   303
                     0003   304
                     0003   305 ;*****************************************************************************
                     0003   306 ;
                     0003   307 ;  THE FOLLOWING EQUATES ARE USED AS PROGRAM CONSTANTS
                     0003   308 ;  THIS IS DONE WHERE EVER POSSIBLE TO CONTROL THE USAGE OF THESE
                     0003   309 ;  CONSTANTS -- SO THEY ARE DEFINED ONLY IN ONE PLACE --
                     0003   310 ;
                     0003   311 ;*****************************************************************************
                     0003   312
```

```
00000011  0003   313 ADDR            = <^X1>          ;BIT 24 COMMAND + STATUS WORD
00000049  0003   314 APT_PARAM       = <^X49>         ;LS LOCATION FOR APT PARAMETERS
00000010  0003   315 BAD_STATUS      = <^X10>         ;BIT 4 COMMAND + STATUS WORD
00000080  0003   316 BEG_TST         = <^X80>         ;BIT 15 COMMAND + STATUS WORD
00004082  0003   317 BYTSUM          = <^X4082>       ;ADDRESS OF CHECKSUM IN BOOT BLOCK
000000FE  0003   318 CC_ADDR         = <^XFE>         ;LS ADDR OF ALU CC REG
00C00008  0003   319 CLEARPAR_ERR    = <^X8>          ;BIT 27 COMMAND + STATUS WORD
00000010  0003   320 CONATT          = <^X10>         ;BIT 20 COMMAND + STATUS WORD
00000020  0003   321 CONACK          = <^X20>         ;BIT 21 COMMAND + STATUS WORD
00000002  0003   322 CONHLT          = <^X2>          ;BIT 17 COMMAND + STATUS WORD
00000004  0003   323 CON_LOOP        = <^X4>          ;BIT 10 COMMAND + STATUS WORD
00C00080  0003   324 CON_RAM         = <^X80>         ;BIT 7 ON = SECT FOR CONSOLE RAM
00007003  0003   325 CON_VER_ADD     = <^X7003>       ;ADDRESS IN 8085 BASED TEST WHERE
          0003   326                                  ; VERSION NUMBER IS FOUND (2 BYTES)
00007005  0003   327 CON_NAME_ADD    = <^X7005>       ;ADDRESS IN 8085 BASED TEST WHERE
          0003   328                                  ; FILE NAME IS FOUND (LAST 2 BYTES)
0000000D  0003   329 CR              = <^X0D>         ;HEX VALUE OF CR
000040FD  0003   330 CRD_FLAGS       = <^X40FD>       ;ADDRESS OF CRD FLAGS IN BOOT BLOCK
000040FE  0003   331 CTLDIS          = <^X40FE>       ;BOOT BLOCK DISABLE CNTL CHAR FLAG
0000411A  0003   332 DIR_VECTOR      = <^X411A>       ;ADDRESS OF DIR COMMAND IN BOOT BLOCK
0000002E  0003   333 DOT_A           = <^X2E>         ;ASCII FOR DOT
00000040  0003   334 EMEMREF         = <^X40>         ;BIT 22 COMMAND + STATUS WORD
00000040  0003   335 EOS             = <^X40>         ;BIT 14 COMMAND + STATUS WORD
00007000  0003   336 EXEC_CONTESTS   = <^X7000>       ;BEGINNING OF TESTS = RAM LOAD ADDRESS
00000048  0003   337 FLAG_LOC        = <^X48>         ;LS LOCATION OF FLAG INFO TO WRITE
00000010  0003   338 FPA_PRES        = <^X10>         ;BIT 12 COMMAND + STATUS WORD
00004129  0003   339 GCVECT          = <^X4129>       ;ADDR OF GET CHAR ROUTINE IN BOOT BLOCK
00004123  0003   340 GLVECT          = <^X4123>       ;ADDR OF GET LINE ROUTINE IN BOOT BLOCK
00004145  0003   341 GSVECT          = <^X4145>       ;ADDR OF GET SILO ROUTINE IN BOOT BLOCK
00000001  0003   342 HEX_1           = <^X1>          ;HEX VALUE OF 1
00000003  0003   343 HEX_3           = <^X3>          ;HEX VALUE OF 3
00000004  0003   344 HEX_4           = <^X4>          ;HEX VALUE OF 4
00000007  0003   345 HEX_7           = <^X7>          ;HEX VALUE OF 7
00000008  0003   346 HEX_8           = <^X8>          ;HEX VALUE OF 8
00000010  0003   347 HEX_10          = <^X10>         ;HEX VALUE OF 10
0000000D  0003   348 HEX_0D          = <^X0D>         ;HEX VALUE OF D
0000001F  0003   349 HEX_1F          = <^X1F>         ;HEX VALUE OF 1F
00000020  0003   350 HEX_20          = <^X20>         ;HEX VALUE OF 20
00000021  0003   351 HEX_21          = <^X21>         ;HEX VALUE OF 21
0000002D  0003   352 HEX_2D          = <^X2D>         ;HEX VALUE OF 2D
00000030  0003   353 HEX_30          = <^X30>         ;HEX VALUE OF 30
0000003A  0003   354 HEX_3A          = <^X3A>         ;HEX VALUE OF 3A
0000003F  0003   355 HEX_3F          = <^X3F>         ;HEX VALUE OF 3F
00000040  0003   356 HEX_40          = <^X40>         ;HEX VALUE OF 40
00000041  0003   357 HEX_41          = <^X41>         ;HEX VALUE OF 41
00000050  0003   358 HEX_50          = <^X50>         ;HEX VALUE OF 50
0000007F  0003   359 HEX_7F          = <^X7F>         ;HEX VALUE OF 7F
00000080  0003   360 HEX_80          = <^X80>         ;HEX VALUE OF 80
000000FD  0003   361 HEX_FD          = <^XFD>         ;HEX VALUE OF FD
000000FF  0003   362 HEX_FF          = <^XFF>         ;HEX VALUE OF FF
00C00080  0003   363 IDC_PRES        = <^X80>         ;BIT 31 COMMAND + STATUS WORD
00C00008  0003   364 INVTIM          = <^X8>          ;BIT 19  COMMAND + STATUS WORD
0000000A  0003   365 LF              = <^XA>          ;HEX VALUE OF LINE FEED
00000005  0003   366 LS_5            = <^X5>          ;LS LOCATION 5 (ADDRESS)
00000006  0003   367 LS_6            = <^X6>          ;LS LOCATION 6 (DATA)
00000007  0003   368 LS_7            = <^X7>          ;LS LOCATION 7 (XFER POINTER)
0000000F  0003   369 LS_F            = <^XF>          ;LS LOCATION F (T15)
```

```
00000047  0003   370 LOOP_CNT_ADD    = <^X47>          ;ADDRESS OF LOOP COUNT
00000002  0003   371 LOOPING         = <^X2>           ;BIT 25 COMMAND + STATUS WORD
00000001  0003   372 MM_OR_LS        = <^X1>           ;MSB = MM XFER
00000008  0003   373 MM_SIZE         = <^X8>           ;BIT 11 COMMAND + STATUS WORD
00000001  0003   374 MOD_PRES_FLG    = <^X1>           ;BIT 1 ON = FPA, R80 OR IDC PRESENT
00000080  0003   375 NA              = <^X80>          ;BIT 23 COMMAND + STATUS WORD
00000080  0003   376 NEG_CNT         = <^X80>          ;MSB ON FOR NEGATIVE X COM COUNT
000040FF  0003   377 NOCHK           = <^X40FF>        ;FLAG IN BOOT BLOCK TO IGNORE <CR>
000040BB  0003   378 OFLAG           = <^X40BB>        ;CONTROL O FLAG IN BOOT BLOCK
00000040  0003   379 OPTIONAL_TST    = <^X40>          ;BIT 6 OF SECTION FLAG
0000007C  0003   380 OS_ADD          = <^X7C>          ;LS ADDRESS OF OS REGISTER
0000001B  0003   381 POWER_SEQ       = <^X1B>          ;ADDRESS OF POWER UP SEQ IN ROM
00004080  0003   382 PRI_STACK       = <^X4080>        ;STACK AREA IN BOOT BLOCK (128 BYTES)
00000004  0003   383 PWRFAIL         = <^X4>           ;BIT 18 COMMAND + STATUS WORD
00000080  0003   384 R80_PRES        = <^X80>          ;BIT 7 COMMAND + STATUS WORD
00007000  0003   385 RAM_LOAD_ADD    = EXEC_CONTESTS   ;ADDRESS RAM OVERLAYS LOAD AT.
0000414D  0003   386 RET_VECTOR      = <^X414D>        ;RETURN VECTOR TO CONSOLE FROM CRD
00000008  0003   387 ROM_X_ADDRS     = <^X8>           ;ROM ADDRESS OF X COMMAND
0000413D  0003   388 RSVECT          = <^X413D>        ;ADDR OF READ SILO ROUTINE IN BOOT BLOCK
00000006  0003   389 SEC_LEN         = 6               ;LENGTH OF A SECTION NAME
00004138  0003   390 SCVECT          = <^X4138>        ;ADDR OF SEND CHAR ROUTINE IN BOOT BLOCK
00000002  0003   391 SETPAR_ERR      = <^X02>          ;BIT 9 COMMAND + STATUS WORD
00000004  0003   392 SINGLE_ERR      = <^X4>           ;BIT 26 COMMAND + STATUS WORD
00000001  0003   393 SIGNAL          = <^X1>           ;BIT 16 COMMAND + STATUS WORD
0000412E  0003   394 SLVECT          = <^X412E>        ;ADDR OF SEND LINE ROUTINE IN BOOT BLOCK
00004133  0003   395 SLVEC1          = <^X4133>        ;ADDR OF SPECIAL SEND LINE ROUTINE
00000020  0003   396 SPACE_A         = <^X20>          ;HEX VALUE OF SPACE
00004086  0003   397 SPBUFF          = <^X4086>        ;ADDRESS IN BOOT BLOCK WHERE SP STORED
00000000  0003   398 STNDRD_CON      = 0               ;BRANCH ADDRESS TO LOAD STNDARD CONSOLE
00000040  0003   399 STATUS          = <^X40>          ;ADDRESS OF COMMAND + STATUS WORD
00004117  0003   400 TU58_DRIVER     = <^X4117>        ;ADDRESS OF TU58 DRIVER
00000040  0003   401 UBE_PRES        = <^X40>          ;BIT 6 COMMAND + STATUS WORD
00000040  0003   402 UBS_BUSY        = <^X40>          ;BIT 30 COMMAND + STATUS WORD
00000020  0003   403 UBS_DC_LO       = <^X20>          ;BIT 29 COMMAND + STATUS WORD
00000010  0003   404 UBS_INIT        = <^X10>          ;BIT 28 COMMAND + STATUS WORD
00000001  0003   405 WCSERR          = <^X1>           ;BIT  8 COMMAND + STATUS WORD
00000044  0003   406 WCS_ADDR_DAT    = <^X44>          ;ADDRESS OF ADDRESS DATA WORD
00000062  0003   407 WCS_CL_FIFO     = <^X62>          ;ADDRESS OF CLEAR IDC FIFO ADDR
          0003   408                                  ; SUBROUTINE
00000600  0003   409 WCS_DAT_32_ADD  = <^X600>         ;ADDRESS OF DATA 32 XFER ROUTINE
0000005F  0003   410 WCS_DE_ICSR     = <^X5F>          ;ADDRESS OF DEPOSIT IDC CSR SUBROUTINE
00000060  0003   411 WCS_DE_IDAR     = <^X60>          ;ADDRESS OF DEPOSIT IDC DAR SUBROUTINE
00000061  0003   412 WCS_DE_DBUF     = <^X61>          ;ADDRESS OF DEPOSIT IDC DBUF SUBROUTINE
00000050  0003   413 WCS_DE_MCT      = <^X50>          ;ADDRESS OF DEPOSIT MCT SUBROUTINE
00000054  0003   414 WCS_DE_MM       = <^X54>          ;ADDRESS OF DEPOSIT MM SUBROUTINE
00000052  0003   415 WCS_DE_TB       = <^X52>          ;ADDRESS OF DEPOSIT TB SUBROUTINE
00000058  0003   416 WCS_DE_UBS      = <^X58>          ;ADDRESS OF DEPOSIT UBS SUBROUTINE
00000041  0003   417 WCS_ERR_NUM     = <^X41>          ;ADDRESS OF ERROR NUMBER DATA WORD
0000005C  0003   418 WCS_EX_DBUF     = <^X5C>          ;ADDRESS OF EXAMINE IDC DATA BUF
0000005A  0003   419 WCS_EX_ICSR     = <^X5A>          ;ADDRESS OF EXAMINE IDC CSR
0000005B  0003   420 WCS_EX_IDAR     = <^X5B>          ;ADDRESS OF EXAMINE IDC DAR
00000051  0003   421 WCS_EX_MCT      = <^X51>          ;ADDRESS OF EXAMINE MCT SUBROUTINE
00000055  0003   422 WCS_EX_MM       = <^X55>          ;ADDRESS OF EXAMINE MM SUBROUTINE
0000005D  0003   423 WCS_EX_PATT     = <^X5D>          ;ADDRESS OF EXAMINE IDC ECC PATTERN
0000005E  0003   424 WCS_EX_POSIT    = <^X5E>          ;ADDRESS OF EXAMINE IDC ECC POSITION
00000053  0003   425 WCS_EX_TB       = <^X53>          ;ADDRESS OF EXAMINE TB SUBROUTINE
00000059  0003   426 WCS_EX_UBS      = <^X59>          ;ADDRESS OF EXAMINE UBS SUBROUTINE
```

```
00000042  0003   427 WCS_EXP_DAT     = <^X42>                ;ADDRESS OF 5>          ;ADD DATA WORD
00000000  0003   428 WCS_LOAD_ADD    = <^X0>                 ;ADDRESS WCS OVERLAYS LOAD AT.
00000046  0003   429 WCS_MOD_DAT     = <^X46>                ;ADDRESS OF MODULE NUMBER DATA WORD
00000045  0003   430 WCS_MSK_DAT     = <^X45>                ;ADDRESS OF ERROR MASK DATA WORD
00000043  0003   431 WCS_REC_DAT     = <^X43>                ;ADDRESS OF RECEIVED DATA WORD
00000057  0003   432 WCS_REST_WR     = <^X57>                ;RESTORE 2901 WORKING REGS SUBR ADDR
00000056  0003   433 WCS_SAVE_WR     = <^X56>                ;SAVE 2901 WORKING REGS SUBR ADDR
00000063  0003   434 WCS_SEL_FIFOA   = <^X63>                ;ADDRESS OF SELECT IDC FIFO A
          0003   435                                         ; SUBROUTINE
00000064  0003   436 WCS_SEL_FIFOB   = <^X64>                ;ADDRESS OF SELECT IDC FIFO B
          0003   437                                         ; SUBROUTINE
00000606  0003   438 WCS_VER_ADD     = <^X606>               ;ADDRESS OF WCS BASED VERSION NUMBER
00000607  0003   439 WCS_FILE_NAME   = <^X607>               ;ADDRESS OF WCS BASED FILE NAME
00000020  0003   440 WRONG_LAB       = <^X20>                ;BIT 5 COMMAND + STATUS WORD
00000020  0003   441 XFER            = <^X20>                ;BIT 13 COMMAND + STATUS WORD
          0003   442
          0003   443 ;*****************************************************************************
          0003   444 ;
          0003   445 ;  THE FOLLOWING SET OF EQUATES DEFINE THE BITS USED IN
          0003   446 ;  THE COMMAND AND STATUS WORD
          0003   447 ;
          0003   448 ;*****************************************************************************
          0003   449
          0003   450 ;FLAG EQUATES
00000001  0003   451 LOOP_S_DEF      = <^X1>                 ;MASK USED TO SET LOOP ON ERROR
00000040  0003   452 LOOP_COM_S_DEF  = <^X40>                ;MASK USED TO SET LOOP COMAND BIT
00000080  0003   453 SPECIAL_S_DEF   = <^X80>                ;MASK USED TO SET SPECIAL BIT (USED
          0003   454                                         ; TO LOOP ON SIZING TESTS)
00000002  0003   455 NER_S_DEF       = <^X2>                 ;MASK USED TO SET NO ERROR REPORTS
00000004  0003   456 BELL_S_DEF      = <^X4>                 ;MASK USED TO SET BELL ON ERROR
00000008  0003   457 HALT_S_DEF      = <^X8>                 ;MASK USED TO SET HALT ON ERROR
00000010  0003   458 SER_S_DEF       = <^X10>                ;MASK USED TO SET SINGLE ERR. REP.
00000020  0003   459 APT_S_DEF       = <^X20>                ;MASK USED TO SET APT PRESENT
          0003   460                                         ; INDICATOR (NOT SETABLE BY SET
          0003   461                                         ; COMMAND)
          0003   462
000000BE  0003   463 LOOP_C_DEF      = <^XBE>                ;MASK USED TO CLEAR LOOP ON ERROR
          0003   464                                         ; AND LOOP COMMAND BITS
000000BF  0003   465 LOOP_COM_C_DEF  = <^XBF>                ;MASK USED TO CLEAR LOOP COMAND BIT
0000007F  0003   466 SPECIAL_C_DEF   = <^X7F>                ;MASK USED TO CLEAR SPECIAL BIT (USED
          0003   467                                         ; TO LOOP ON SIZING TESTS)
000000BD  0003   468 NER_C_DEF       = <^XBD>                ;MASK USED TO CLEAR NO ERROR REPORT
          0003   469                                         ; AND LOOP COMMAND BITS
000000FB  0003   470 BELL_C_DEF      = <^XFB>                ;MASK USED TO CLEAR BELL ON ERROR
000000F7  0003   471 HALT_C_DEF      = <^XF7>                ;MASK USED TO CLEAR HALT ON ERROR
000000EF  0003   472 SER_C_DEF       = <^XEF>                ;MASK USED TO CLEAR SINGLE ERR. REP.
000000DF  0003   473 APT_C_DEF       = <^XDF>                ;MASK USED TO CLEAR APT PRESENT
          0003   474                                         ; INDICATOR (NOT CLEARABLE BY CLEAR
          0003   475                                         ; COMMAND)
          0003   476
          0003   477 ;*****************************************************************************
          0003   478 ;
          0003   479 ;  PARAMETER BLOCK FOR TU58 DRIVER
          0003   480 ;
          0003   481 ;*****************************************************************************
          0003   482
          0003   483                                         ;TU58 DRIVER PARAMETER BLOCK
```

```
00004106  0003   484 SEC_PARMB_UNIT  = <^X4106>          ;UNIT NUMBER
00004107  0003   485 SEC_PARMB_NAM   = <^X4107>          ;FILENAME
0000410D  0003   486 SEC_PARMB_EXT   = <^X410D>          ;DOT PLUS FILE EXTENTION
00004111  0003   487 SEC_PARMB_ADD   = <^X4111>          ;LOAD ADDRESS
00004115  0003   488 SEC_PARMB_DST   = <^X4115>          ;DESTINATION 6=RAM 7=WCS
00004116  0003   489 SEC_PARMB_ERR   = <^X4116>          ;ERROR RETURN BYTE
00004150  0003   490 SEC_PARMB_DEF   = <^X4150>          ;DEFAULT DRIVE (WHERE MICMON WAS LOADED
          0003   491                                     ; FROM)
          0003   492
          0003   493 ;*****************************************************************************
          0003   494 ;
          0003   495 ;  EXTERNAL REFERENCES
          0003   496 ;  ALL EXTERNAL REFERENCES MUST BE LISTED HERE
          0003   497 ;
          0003   498 ;*****************************************************************************
          0003   499
          0003   500                 .GLOBAL COMPARE,EOS_FLG,FLAGS,HEX_BUF
          0003   501                 .GLOBAL MOVER,PRINT,PRINT_HEX,SPACE
          0003   502                 .GLOBAL MWCS_A,MCPU_A,CR,LF,HEX_3,CC_ADDR,CRLF
          0003   503                 .GLOBAL SHIFT_PNT,F_WORD,SHIFTER,DOLOOP
          0003   504                 .GLOBAL NOP_CSR,READ_WCS,WCS_ADDRESS,WCS_VALUE
          0003   505                 .GLOBAL WRITE_WCS,X_SET_PAGE,PERFORM_CSR
          0003   506                 .GLOBAL X_CLR_PAGE,WCS_VAL_SHIFT
          0003   507                 .GLOBAL CSR_VALUE,LOAD_CSR,UPC_VALUE,LOAD_UPC
          0003   508                 .GLOBAL PARITY_JUMP,PARITY_ON,X_MOV_WRWR
          0003   509                 .GLOBAL SHIFT_L_2901,SHIFT_R_2901,DATA0
          0003   510                 .GLOBAL READ_DATA_32,WRITE_DATA_32,X_CLR_WRU
          0003   511                 .GLOBAL DATA1,DATA2,DATA3,DATA_SHIFT
          0003   512                 .GLOBAL READ_LS,WRITE_LS,INC_WORD,OS_ADD
          0003   513                 .GLOBAL NOP_INST,CSR_SHIFT,X_COM_WRO
          0003   514                 .GLOBAL MAKE_XD,XD_ADDRS,X_ROT_L,X_SHIFT_L
          0003   515                 .GLOBAL WCS_BAD_PARITY,POWER_BRANCH,WCS_SHIFT
          0003   516
          0003   517                 .GLOBAL PRINT_STRING,MOVER_4,PRINT_HEX_1,LD_UPC
          0003   518                 .GLOBAL CON_ADD_DAT,CON_ERR_NUM,CON_EXP_DAT
          0003   519                 .GLOBAL CON_EXP_ADD,CON_MOD_DAT,CON_MSK_DAT
          0003   520                 .GLOBAL CON_REC_DAT,CON_REC_ADD,ERROR_CON,MOVER_3
          0003   521                 .GLOBAL NA_EXP_REC,NA_OTHER,SKIP_TEST,MSK_ERROR
          0003   522                 .GLOBAL CON_ERROR,CON_BEGIN_TEST,GCVECT,WCS_READ
          0003   523                 .GLOBAL WCS_WRITE,CONTINUE,CNTLC_TYPED,SET_FOR_CONT
          0003   524                 .GLOBAL CLEAR_CPU_ATT,SHIFTER1,X_SHIFT_R,DECR_WORD
          0003   525                 .GLOBAL MICRO_STEP_CPU,WRITE_TAB_PNT
          0003   526                 .GLOBAL APT,APT_PASS_CNT
          0003   527
          0003   528 ;*****************************************************************************
          0003   529 ;
          0003   530 ;  APT INFORMATION.... THE ADDRESS OF THESE VARIABLES MUST NOT CHANGE
          0003   531 ;  IN OFFSET FROM THE BEGINNING OF THIS PROGRAM.
          0003   532 ;
          0003   533 ;*****************************************************************************
          0003   534
00000006  0003   535 OVERLAY_RAM::    .BLKB   3               ;STARTING ADDRESS FOR RAM TESTS
          0006   536                                         ; STARTED BY APT (NOT USED FOR CRD)
00000009  0006   537 OVERLAY_WCS::    .BLKB   3               ;STARTING ADDRESS FOR WCS TESTS
          0009   538                                         ; STARTED BY APT (NOT USED FOR CRD)
     7000 0009   539 APT_RAM_LD_ADD: .WORD   EXEC_CONTESTS   ;POINTER TO RAM OVERLAY LOAD ADDRESS
00000010  0008   540                 .BLKB   5
```

```
                        0010   541 APT_MAIL_BOX:
                0000    0010   542 APT_MESSAGE_CODE:.WORD  0               ;FLAG 1=TEST START  2=ERROR
                0000    0012   543 APT_ERROR_NUMBER:.WORD  0               ;DIAGNOSTIC ERROR NUMBER
                0000    0014   544 APT_SUBTEST:    .WORD   0               ;SUBTEST NUMBER FILLER
                0000    0016   545 APT_TEST_NUMBER:.WORD   0               ;FAILING TEST NUMBER
                0000    0018   546 APT_PASS_CNT:   .WORD   0               ;# PASSES EXECUTED
            0000001C    001A   547                 .BLKB   2               ;RESERVED FILLER
                  00    001C   548 APT_MEM_SIZE:   .BYTE   0               ;APT MEMORY SIZE
                  00    001D   549 APT_HARD_FLG:   .BYTE   0               ;APT HARDWARE INFORMATION
                  00    001E   550 APT_SOFT_FLG:   .BYTE   0               ;APT SOFTWARE INFORMATION
            00000022    001F   551                 .BLKB   3               ;RESERVED FILLER
                        0022   552
                        0022   553 ;*****************************************************************************
                        0022   554 ;
                        0022   555 ;  RETURN VECTORS - THE FOLLOWING FOUR INSTRUCTIONS ARE USED AS THE RETURN
                        0022   556 ;  POINT FOR SOME BOOT BLOCK ROUTINES.  THE NEXT BYTE IS A FLAG READ BY THE
                        0022   557 ;  BOOT BLOCK TO DETERMINE IF MICMON IS LOADED.  THE NEXT TWO INSTRUCTIONS
                        0022   558 ;  ARE ALSO USED AS VECTORS FROM THE BOOT BLOCK.  THEY MUST NOT CHANGE IN
                        0022   559 ;  OFFSET FROM THE BEGINNING OF THIS PROGRAM.
                        0022   560 ;
                        0022   561 ;*****************************************************************************
                        0022   562
                        0022   563                 JMP     PARITY_ERR      ;RETURN FROM PARITY ERROR
                  C3    0022              .BYTE   ^0303
                193A'   0023              .WORD   <PARITY_ERR-HEAD>
                        0025   564                 JMP     POWER_REC       ;RETURN FROM POWER FAIL RECOVER
                  C3    0025              .BYTE   ^0303
                1C0C'   0026              .WORD   <POWER_REC-HEAD>
                        0028   565                 JMP     CNTLC_VEC       ;RETURN FROM CNTL C TYPED
                  C3    0028              .BYTE   ^0303
                1063'   0029              .WORD   <CNTLC_VEC-HEAD>
                        002B   566                 JMP     CNTLP_VEC       ;RETURN FROM CNTL P TYPED
                  C3    002B              .BYTE   ^0303
                1070'   002C              .WORD   <CNTLP_VEC-HEAD>
                  01    002E   567 MICFLAG:         .BYTE           1       ;FLAG SO BOOT BLOCK KNOWS MIC RESIDENT
                        002F   568                 JMP     TIMER_VEC       ;RETURN FROM INTERVAL TIMER INTERRUPT
                  C3    002F              .BYTE   ^0303
                1C23'   0030              .WORD   <TIMER_VEC-HEAD>
                        0032   569                 JMP     AUTO_TEST       ;ENTER MICMON IN AUTO TEST MODE
                  C3    0032              .BYTE   ^0303
                1C24'   0033              .WORD   <AUTO_TEST-HEAD>
            00000038    0035   570                 .BLKB   3               ;RESERVED FILLER FOR NEW RETURN VECTORS
                        0038   571
                        0038   572 ;*****************************************************************************
                        0038   573 ;
                        0038   574 ;       THE FOLLOWING DECLARATIONS ARE FOR GLOBAL VARIABLES.
                        0038   575 ;       DO NOT DISPLACE FROM THEIR CURRENT LOCATIONS.
                        0038   576 ;       ADD NEW VARIABLES AFTER ALL GLOBAL DECLARATIONS.
                        0038   577 ;
                        0038   578 ;*****************************************************************************
                        0038   579
                  00    0038   580 APT:             .BYTE           0       ;FLAG 1=APT PRESENT
                  00    0039   581 CNTLC_TYPED:     .BYTE           0       ;FLAG 1=CONTROL C WAS TYPED
                  00    003A   582 CONTINUE:        .BYTE           0       ;FLAG 1=CONTINUE MODE ENABLED
            0000003F    003B   583 CON_ADD_DAT:     .BLKB           4       ;OTHER DATA
                  01    003F   584 CON_ERR_NUM:     .BYTE           1       ;CONSOLE BASED ERROR NUMBER
            00000044    0040   585 CON_EXP_DAT:     .BLKB           4       ;EXPECTED DATA
```

```
00000046  0044    586 CON_EXP_ADD:    .BLKB           2               ;EXP_DAT ADDR USED BY MSK_ERR
0000004B  0046    587 CON_MOD_DAT:    .BLKB           5               ;ASCII MODULE NAME-FILLED IN
0000004F  004B    588 CON_MSK_DAT:    .BLKB           4               ;MASK DATA
00000053  004F    589 CON_REC_DAT:    .BLKB           4               ;RECEIVED DATA
00000055  0053    590 CON_REC_ADD:    .BLKB           2               ;REC_DAT ADDR USED BY MSK_ERR
       00 0055    591 DOLOOP:         .BYTE           0               ;DO MACRO LOOP COUNTER
       00 0056    592 EOS_FLG:        .BYTE           0               ;FLAG 1=END OF SECTION
       00 0057    593 ERROR_CON:      .BYTE           0               ;FLAG 1=ERROR EXISTS
     FFFF 0058    594 F_WORD:         .WORD           <^XFFFF>
       00 005A    595 FLAGS:          .BYTE           0               ;HALT,LOOP,NER,BELL,SER
0000005F  005B    596 HEX_BUF:        .BLKB           4               ;HEX PRINT OUT BUFFER
       00 005F    597 NA_EXP_REC:     .BYTE           0               ;FLAG 1=N/A FOR EXP AND REC DA
       00 0060    598 NA_OTHER:       .BYTE           0               ;FLAG 1=N/A FOR OTHER DATA
          0061    599 PARITY_JUMP:    JMP             PARITY_ERR
       C3 0061                .BYTE   ^O303
     193A' 0062                .WORD   <PARITY_ERR-HEAD>
       01 0064    600 PARITY_ON:      .BYTE           1               ;FLAG 1=CALCULATE PARITY
          0065    601 POWER_BRANCH:   JMP             POWER_REC
       C3 0065                .BYTE   ^O303
     1C0C' 0066                .WORD   <POWER_REC-HEAD>
     0000 0068    602 SHIFT_PNT:      .WORD           0               ;SHIFT ROUTINE DATA POINTER
       00 006A    603 SKIP_TEST:      .BYTE           0               ;FLAG 1=SKIP TEST # IS TO LOW
       02 006B    604 WCS_SHIFT:      .BYTE           2               ;TWO BYTES OF SHIFT DATA FOLLOW
     0000 006C    605 WCS_ADDRESS:    .WORD           0               ;WCS ADDRESS TO READ OR WRITE
       00 006E    606 WCS_BAD_PARITY: .BYTE           0               ;FLAG 1=WRITE BAD PARITY
     0000 006F    607 WRITE_TAB_PNT:  .WORD           0               ;POINTER TO UPC ADDRES TABLE
       00 0071    608 XD_ADDRS:       .BYTE           0               ;XD ADDRS FIELD OF CPU INST
          0072    609
          0072    610 ;GLOBAL CSR EXECUTABLE CODE
          0072    611
       DB 0072    612 NOP_INST:       .BYTE           <^XDB>          ;NOP INST FOR CPU CSR
       00 0073    613                 .BYTE           <^X00>
       15 0074    614                 .BYTE           <^X15>
       01 0075    615 X_SHIFT_L:      .BYTE           1
       23 0076    616                 .BYTE           <^X23>          ;CLR WR[1]
       D0 0077    617                 .BYTE           <^XD0>
       15 0078    618                 .BYTE           <^X15>
          0079    619
       01 0079    620 X_ROT_L:        .BYTE           1
       A3 007A    621                 .BYTE           <^XA3>          ;ROL WR[0]
       C0 007B    622                 .BYTE           <^XC0>          ;SHIFT LEFT 1 PLACE
       15 007C    623                 .BYTE           <^X15>
          007D    624
       01 007D    625 X_CLR_PAGE:     .BYTE           1
       90 007E    626                 .BYTE           <^X90>          ;MISC [CLR.HIGH.PAGE]
       A0 007F    627                 .BYTE           <^XA0>
       15 0080    628                 .BYTE           <^X15>
          0081    629
       01 0081    630 X_SET_PAGE:     .BYTE           1
       10 0082    631                 .BYTE           <^X10>          ;MISC [SET.HIGH.PAGE]
       B0 0083    632                 .BYTE           <^XB0>
       15 0084    633                 .BYTE           <^X15>
          0085    634
          0085    635
          0085    636 ;  This subroutine shifts the contents of WR[0] right 8 bits and leaves the
          0085    637 ;  result on the Y-bus.  The monitor can clock the console read register to
          0085    638 ;  read the result.  To be executed 3 times by the monitor for a 32 bit WR
```

```
                     0085   639 ;  read.  ROR will have to have been checked by the console.
                     0085   640
08                   0085   641 X_SHIFT_R:      .BYTE           8
23                   0086   642                 .BYTE           ^X23>           ;ROR
40                   0087   643                 .BYTE           <^X40>
15                   0088   644                 .BYTE           <^X15>
                     0089   645
23                   0089   646                 .BYTE           <^X23>          ;ROR
40                   008A   647                 .BYTE           <^X40>
15                   008B   648                 .BYTE           <^X15>
                     008C   649
23                   008C   650                 .BYTE           <^X23>          ;ROR
40                   008D   651                 .BYTE           <^X40>
15                   008E   652                 .BYTE           <^X15>
                     008F   653
23                   008F   654                 .BYTE           <^X23>          ;ROR
40                   0090   655                 .BYTE           <^X40>
15                   0091   656                 .BYTE           <^X15>
                     0092   657
23                   0092   658                 .BYTE           <^X23>          ;ROR
40                   0093   659                 .BYTE           <^X40>
15                   0094   660                 .BYTE           <^X15>
                     0095   661
23                   0095   662                 .BYTE           <^X23>          ;ROR
40                   0096   663                 .BYTE           <^X40>
15                   0097   664                 .BYTE           <^X15>
                     0098   665
23                   0098   666                 .BYTE           <^X23>          ;ROR
40                   0099   667                 .BYTE           <^X40>
15                   009A   668                 .BYTE           <^X15>
                     009B   669
23                   009B   670                 .BYTE           <^X23>          ;ROR
40                   009C   671                 .BYTE           <^X40>
15                   009D   672                 .BYTE           <^X15>
                     009E   673
A0                   009E   674 X_MOV_WRWR:     .BYTE           <^XA0>          ;MOV WR[0] TO WR[0]
00                   009F   675                 .BYTE           <^X00>          ;MOVE DATA TO Y-BUS
15                   00A0   676                 .BYTE           <^X15>          ;TO BE READ
                     00A1   677
                     00A1   678
                     00A1   679
02                   00A1   680 X_CLR_WR0:      .BYTE           2
2F                   00A2   681                 .BYTE           <^X2F>          ;CLR WR[0]
80                   00A3   682                 .BYTE           <^X80>
15                   00A4   683                 .BYTE           <^X15>
                     00A5   684
A0                   00A5   685 X_COM_WR0:      .BYTE           <^XA0>          ;COM WR[0]
C0                   00A6   686                 .BYTE           <^XC0>
15                   00A7   687                 .BYTE           <^X15>
                     00A8   688
                     00A8   689 ;*****************************************************************************
                     00A8   690 ;
                     00A8   691 ;  DATA AREA FOR READS AND WRITES TO CPU (32 BITS)
                     00A8   692 ;
                     00A8   693 ;*****************************************************************************
                     00A8   694
04                   00A8   695 DATA_SHIFT:      .BYTE           4               ;FOUR BYTE OF SHIFT DATA
```

```
                    00  00A9   696 DATA0:          .BYTE           0               ;MSB OF 32 BIT DATA AREA
                    00  00AA   697 DATA1:          .BYTE           0
                    00  00AB   698 DATA2:          .BYTE           0
                    00  00AC   699 DATA3:          .BYTE           0               ;LSB OF 32 BIT DATA AREA
             000000AF   00AD   700                 .BLKB           2               ;EXTENDED DATA FOR 3 BYTE MOVES
                        00AF   701
                    00  00AF   702 COMMAND0:       .BYTE           0               ;MSB OF 32 BIT COMMAND+STATUS
                    00  00B0   703 COMMAND1:       .BYTE           0               ;WORD
                    00  00B1   704 COMMAND2:       .BYTE           0
                    00  00B2   705 COMMAND3:       .BYTE           0               ;LSB OF 32 BIT COMMAND+STATUS
                        00B3   706
                    03  00B3   707 WCS_VAL_SHIFT:  .BYTE           3               ;SHIFT 3 BYTES OF DATA
             00000087   00B4   708 WCS_VALUE:      .BLKB           3               ;WCS WORD TO BE WRITTEN
                        00B7   709
                        00B7   710
                        00B7   711 ;*****************************************************************************
                        00B7   712 ;
                        00B7   713 ;  CSR DATA BUFFER
                        00B7   714 ;
                        00B7   715 ;*****************************************************************************
                        00B7   716
                    03  00B7   717 CSR_SHIFT:      .BYTE           3               ;3 BYTES OF DATA
             000000BB   00B8   718 CSR_VALUE:      .BLKB           3
             000000BE   00BB   719 SAVED_CSR:      .BLKB           3
             00000CC1   00BE   720 CONT_SAVE_CSR:  .BLKB           3
                        00C1   721
                        00C1   722 ;*****************************************************************************
                        00C1   723 ;
                        00C1   724 ; UPC DATA BUFFER
                        00C1   725 ;
                        00C1   726 ;*****************************************************************************
                        00C1   727
                    02  00C1   728 UPC_SHIFT:      .BYTE           2               ;2 BYTES OF DATA
             000000C4   00C2   729 UPC_VALUE:      .BLKB           2
             000000C6   00C4   730 SAVED_UPC:      .BLKB           2
             000000C8   00C6   731 UPC_SUB:        .BLKB           2
             000000CA   00C8   732 CONT_SAVE_UPC:  .BLKB           2
             000000CC   00CA   733 UPC_COMPARE:    .BLKB           2
             000000CF   00CC   734 ATT_UPC_SAVE:   .BLKB           3
                        00CF   735
                        00CF   736 ;JUMPS TO GLOBAL SUBROUTINES
                        00CF   737 COMPARE:        JMP             COMPARE_R
                      C3  00CF                   .BYTE   ^0303
                  0F55' 00D0                   .WORD   <COMPARE_R-HEAD>
                        00D2   738 MOVER:          JMP             MOVER_R
                      C3  00D2                   .BYTE   ^0303
                  0F75' 00D3                   .WORD   <MOVER_R-HEAD>
                        00D5   739 PRINT:          JMP             PRINT_R
                      C3  00D5                   .BYTE   ^0303
                  1027' 00D6                   .WORD   <PRINT_R-HEAD>
                        00D8   740 PRINT_HEX:      RET
                      C9  00D8                   .BYTE   ^0311
                        00D9   741                 RET
                      C9  00D9                   .BYTE   ^0311
                        00DA   742                 RET
                      C9  00DA                   .BYTE   ^0311
                        00DB   743 SPACE:          JMP             SPACE_R
```

```
                C3  00DB                .BYTE   ^O303
              1021' 00DC                .WORD   <SPACE_R-HEAD>
                    00DE    744 CRLF:           JMP             CRLF_R
                C3  00DE                .BYTE   ^O303
              101A' 00DF                .WORD   <CRLF_R-HEAD>
                    00E1    745 SHIFTER:        JMP             SHIFTER_R
                C3  00E1                .BYTE   ^O303
              0EB8' 00E2                .WORD   <SHIFTER_R-HEAD>
                    00E4    746 NOP_CSR:        JMP             NOP_CSR_R
                C3  00E4                .BYTE   ^O303
              0ECA' 00E5                .WORD   <NOP_CSR_R-HEAD>
                    00E7    747 READ_WCS:       JMP             READ_WCS_R
                C3  00E7                .BYTE   ^O303
              0E6D' 00E8                .WORD   <READ_WCS_R-HEAD>
                    00EA    748 WRITE_WCS:      JMP             WRITE_WCS_R
                C3  00EA                .BYTE   ^O303
              0E8A' 00EB                .WORD   <WRITE_WCS_R-HEAD>
                    00ED    749 PERFORM_CSR:    JMP             PERFORM_CSR_R
                C3  00ED                .BYTE   ^O303
              12F2' 00EE                .WORD   <PERFORM_CSR_R-HEAD>
                    00F0    750 LOAD_CSR:       JMP             LOAD_CSR_R
                C3  00F0                .BYTE   ^O303
              0EE7' 00F1                .WORD   <LOAD_CSR_R-HEAD>
                    00F3    751 LOAD_UPC:       JMP             LOAD_UPC_R
                C3  00F3                .BYTE   ^O303
              0F19' 00F4                .WORD   <LOAD_UPC_R-HEAD>
                    00F6    752 SHIFT_L_2901:   JMP             SHIFT_L_2901_R
                C3  00F6                .BYTE   ^O303
              1249' 00F7                .WORD   <SHIFT_L_2901_R-HEAD>
                    00F9    753 SHIFT_R_2901:   JMP             SHIFT_R_2901_R
                C3  00F9                .BYTE   ^O303
              1250' 00FA                .WORD   <SHIFT_R_2901_R-HEAD>
                    00FC    754 READ_DATA_32:   JMP             READ_DATA_32_R
                C3  00FC                .BYTE   ^O303
              11B4' 00FD                .WORD   <READ_DATA_32_R-HEAD>
                    00FF    755 WRITE_DATA_32:  JMP             WRITE_DATA_32_R
                C3  00FF                .BYTE   ^O303
              11D2' 0100                .WORD   <WRITE_DATA_32_R-HEAD>
                    0102    756 READ_LS:        JMP             READ_LS_R
                C3  0102                .BYTE   ^O303
              1167' 0103                .WORD   <READ_LS_R-HEAD>
                    0105    757 WRITE_LS:       JMP             WRITE_LS_R
                C3  0105                .BYTE   ^O303
              1146' 0106                .WORD   <WRITE_LS_R-HEAD>
                    0108    758 INC_WORD:       JMP             INC_WORD_R
                C3  0108                .BYTE   ^O303
              1281' 0109                .WORD   <INC_WORD_R-HEAD>
                    010B    759 MAKE_XD:        JMP             MAKE_XD_R
                C3  010B                .BYTE   ^O303
              1183' 010C                .WORD   <MAKE_XD_R-HEAD>
                    010E    760 PRINT_STRING:   RET
                C9  010E                .BYTE   ^O311
                    010F    761                 RET
                C9  010F                .BYTE   ^O311
                    0110    762                 RET
                C9  0110                .BYTE   ^O311
                    0111    763 MOVER_4:        JMP             MOVER_4_R
```

```
                C3  0111                .BYTE   ^0303
            0F73'   0112                .WORD   <MOVER_4_R-HEAD>
                    0114    764 PRINT_HEX_1:    JMP             PRINT_HEX_1_R
                C3  0114                .BYTE   ^0303
            0F94'   0115                .WORD   <PRINT_HEX_1_R-HEAD>
                    0117    765 LD_UPC:         JMP             LD_UPC_R
                C3  0117                .BYTE   ^0303
            1FCC'   0118                .WORD   <LD_UPC_R-HEAD>
                    011A    766 MOVER_3:        JMP             MOVER_3_R
                C3  011A                .BYTE   ^0303
            0F5F'   011B                .WORD   <MOVER_3_R-HEAD>
                    011D    767 MSK_ERROR:      JMP             MSK_ERROR_R
                C3  011D                .BYTE   ^0303
            1F31'   011E                .WORD   <MSK_ERROR_R-HEAD>
                    0120    768 CON_ERROR:      JMP             CON_ERROR_R
                C3  0120                .BYTE   ^0303
            1F14'   0121                .WORD   <CON_ERROR_R-HEAD>
                    0123    769 CON_BEGIN_TEST: JMP             CON_BEGIN_TEST_R
                C3  0123                .BYTE   ^0303
            1EC9'   0124                .WORD   <CON_BEGIN_TEST_R-HEAD>
                    0126    770 WCS_READ:       JMP             WCS_READ_R
                C3  0126                .BYTE   ^0303
            1F80'   0127                .WORD   <WCS_READ_R-HEAD>
                    0129    771 WCS_WRITE:      JMP             WCS_WRITE_R
                C3  0129                .BYTE   ^0303
            1FB6'   012A                .WORD   <WCS_WRITE_R-HEAD>
                    012C    772 SET_FOR_CONT:   JMP             SET_FOR_CONT_R
                C3  012C                .BYTE   ^0303
            107B'   012D                .WORD   <SET_FOR_CONT_R-HEAD>
                    012F    773 CLEAR_CPU_ATT:  JMP             CLEAR_CPU_ATT_R
                C3  012F                .BYTE   ^0303
            1A61'   0130                .WORD   <CLEAR_CPU_ATT_R-HEAD>
                    0132    774 SHIFTER1:       JMP             SHIFTER1_R
                C3  0132                .BYTE   ^0303
            0EC1'   0133                .WORD   <SHIFTER1_R-HEAD>
                    0135    775 DECR_WORD:      JMP             DECR_WORD_R
                C3  0135                .BYTE   ^0303
            128C'   0136                .WORD   <DECR_WORD_R-HEAD>
                    0138    776 MICRO_STEP_CPU: JMP             MICRO_STEP_CPU_R
                C3  0138                .BYTE   ^0303
            12A0'   0139                .WORD   <MICRO_STEP_CPU_R-HEAD>
                    013B    777
                    013B    778 ;*****************************************************************************
                    013B    779 ;
                    013B    780 ;  MODULE LITERALS
                    013B    781 ;
                    013B    782 ;*****************************************************************************
                    013B    783
                    013B    784 PRT_MOD_A:
34 39 33 38 4D      013B    785 MWCS_A:         .ASCII          'M8394'         ;WCS
30 39 33 38 4D      0140    786 MCPU_A:         .ASCII          'M8390'         ;CPU
31 39 33 38 4D      0145    787 MMCT_A:         .ASCII          'M8391'         ;MCT
39 38 33 38 4D      014A    788 MFPA_A:         .ASCII          'M8389'         ;FPA
38 32 37 38 4D      014F    789 MMEM_A:         .ASCII          'M8728'         ;MEMORY ARRAY CARD
38 38 33 38 4D      0154    790 MIDC_A:         .ASCII          'M8388'         ;IDC
20 31 41 51 44      0159    791 MRL02_A:        .ASCII          'DQA1 '         ;RL02
20 30 41 51 44      015E    792 MR80_A:         .ASCII          'DQA0 '         ;R80
```

```
                                        0163    793
                                        0163    794 ;*****************************************************************************
                                        0163    795 ;
                                        0163    796 ;       ALL PRECEDING DECLARATIONS ARE FOR GLOBAL VARIABLES.
                                        0163    797 ;       DO NOT DISPLACE ANY OF THESE FROM THEIR CURRENT LOCATIONS.
                                        0163    798 ;       ALL NEW VARIABLES MUST BE ADDED BELOW THIS POINT.
                                        0163    799 ;
                                        0163    800 ;*****************************************************************************
                                        0163    801
                                        0163    802
                                        0163    803 ;VARIABLES
                                     00 0163    804 A_ADDRS:        .BYTE       0                   ;A ADDRS FIELD OF CPU INST
                                     15 0164    805 ABORT_A:        .BYTE       21                  ;CHAR COUNT
41 20 54 53 45 54 20 4F 54 55 41 0D 0165    806                     .ASCII      <CR>''AUTO TEST ABORTED - ''
         20 2D 20 44 45 54 52 4F 42 0171
                                     00 017A    807 ACK_SAVE:       .BYTE       0                   ;SAVE CPU ACK FOR SUBROUTINES
                                     31 017B    808 ASCII_1:        .ASCII      ''1''
                                     32 017C    809 ASCII_2:        .ASCII      ''2''
                                     33 017D    810 ASCII_3:        .ASCII      ''3''
                                     1A 017E    811 BAD_A:          .BYTE       26
20 4C 4C 41 43 20 2D 20 44 41 42 20 017F    812                     .ASCII      '' BAD - CALL FIELD SERVICE''<CR>
43 49 56 52 45 53 20 44 4C 45 49 46 0188
                              0D 45 0197
                                 46 0199    813 BAD_TAPE_A:     .BYTE       166
4F 52 50 20 20 20 20 20 20 2A 2A 0D 019A    814                     .ASCII      <CR>''**      PROPER SETUP REQUIRED - CANNOT ''
45 52 20 50 55 54 45 53 20 52 45 50 01A6
4E 41 43 20 2D 20 44 45 52 49 55 51 01B2
                     20 54 4F 4E    01BE
20 20 20 20 45 55 4E 49 54 4E 4F 43 01C2    815                     .ASCII      ''CONTINUE      **''
                        2A 2A 20    01CE
44 20 4B 43 45 48 43 20 20 2A 2A 0D 01D1    816                     .ASCII      <CR>''**  CHECK DIAGNOSTIC TAPE INSERTED IN ''
41 54 20 43 49 54 53 4F 4E 47 41 49 01DD
20 44 45 54 52 45 53 4E 49 20 45 50 01E9
                     20 4E 49       01F5
30 20 45 56 49 52 44 20 38 35 55 54 01F8    817                     .ASCII      ''TU58 DRIVE 0  **''
                     2A 2A 20 20    0204
50 55 54 45 53 20 46 49 20 2A 2A 0D 0208    818                     .ASCII      <CR>''** IF SETUP CORRECT, POSSIBLE TAPE OR ''
4F 50 20 2C 54 43 45 52 52 4F 43 20 0214
20 45 50 41 54 20 45 4C 42 49 53 53 0220
                     20 52 4F       022C
45 4C 42 4F 52 50 20 45 56 49 52 44 022F    819                     .ASCII      ''DRIVE PROBLEM **''<CR>
                  0D 2A 2A 20 4D    0238
                                 00 024C    820 B_ADDRS:        .BYTE       0                   ;B ADDRS FIELD OF CPU INST
                           00000247 0241    821 BOARD_NAME:     .BLKB       6                   ;BOARD NAME AREA
                           00000247 0247    822 BOARD_BUF:      .BLKB       6                   ;BOARD NAME AREA
                                 00 024D    823 BOARD_NUM:      .BYTE       0                   ;NUMBER OF BOARD INPUT
                               0000 024E    824 BOARD_PNT:      .WORD       0                   ;POINTER TO BOARD TABLE
                                 00 0250    825 C_EXECUTE:      .BYTE       0                   ;TEMP FLAG HOLDING FOR CONTINUE
                                 00 0251    826 C_RUNNING:      .BYTE       0                   ;TEMP FLAG HOLDING FOR CONTINUE
                               0000 0252    827 CALC_ADD:       .WORD       0                   ;ADDRESS TO CALC PARITY ON
                                 00 0254    828 CNT_PAR:        .BYTE       0                   ;COUNT OF PAR ERRORS
                                 00 0255    829 CONSOLE_TEST:   .BYTE       0                   ;FLAG 1=CONSOLE 0=WCS TEST
                                 00 0256    830 CONT_DONE:      .BYTE       0                   ;FLAG 1=CONTINUE IS DONE DO RET
                                 03 0257    831 CPU_A:          .BYTE       3                   ;BYTE COUNT
                           55 50 43 0258    832                 .ASCII      ''CPU''
                                 01 025B    833 CRLF_A:         .BYTE       1                   ;BYTE COUNT
                                 0D 025C    834                 .ASCII      <CR>                ;CRLF FOR CALL CRLF ROUTINE
```

```
                                        00  025D   835 CSR_CNT:          .BYTE   0               ;COUNT OF CSR INST TO PERFORM
                                        00  025E   836 DATA_XFER_FLG:   .BYTE   0               ;FLAG 1=XFER WAS DONE
                                  00000261  025F   837 ENK_END:         .BLKB   2               ;STORAGE FOR LAST 2 CHARS OF
                                            0261   838                                         ; SECTION NAME READ FROM FILE
                                        00  0261   839                 .BYTE   0               ;LAST CHAR IS NULL
                                        66  0262   840 END_A:           .BYTE   102
20 46 4F 20 44 4E 45 20 2A 2A 2A 2A     0263   841                     .ASCII  "**** END OF VAX-11/730,725 AUTO TEST ****"<CR>
37 2C 30 33 37 2F 31 31 2D 58 41 56     026F
54 53 45 54 20 4F 54 55 41 20 35 32     027B
                  0D 2A 2A 2A 2A 20     0287
54 20 47 4E 49 4E 52 55 54 45 52 0D     028D   842                     .ASCII  <CR>"RETURNING TO VAX-11/730,725 CONSOLE, WAIT FOR"
30 33 37 2F 31 31 2D 58 41 56 20 4F     0299
45 4C 4F 53 4E 4F 43 20 35 32 37 2C     02A5
      52 4F 46 20 54 49 41 57 20 2C     02B1
50 4D 4F 52 50 20 27 3E 3E 3E 27 20     02BB   843                     .ASCII  " '>>>' PROMPT"<CR>
                        0D 54           02C7
                  43 43 4B 4E 45        02C9   844 ENKCC_A:          .ASCII  "ENKCC"
                  45 43 4B 4E 45        02CE   845 ENKCE_A:          .ASCII  "ENKCE"
                  48 43 4B 4E 45        02D3   846 ENKCH_A:          .ASCII  "ENKCH"
                  46 43 4B 4E 45        02D8   847 ENKCF_A:          .ASCII  "ENKCF"
                  47 43 4B 4E 45        02DD   848 ENKCG_A:          .ASCII  "ENKCG"
                              00        02E2   849 ERROR:            .BYTE   0               ;FLAG
                              06        02E3   850 ERROR_A:          .BYTE   6               ;CHAR COUNT
                  52 4F 52 52 45 20     02E4   851                   .ASCII  " ERROR"
                              00        02EA   852 EXECUTE:          .BYTE   0               ;FLAG 1=EXECUTE MODE
                        45 58 45 2E     02EB   853 EXT_A:            .ASCII  ".EXE"          ;ALL OVERLAYS ARE .EXE !!!!!!
                              0C        02EF   854 FAIL_A:           .BYTE   12              ;CHAR COUNT
0D 2A 44 45 4C 49 41 46 2A 20 20 20     02F0   855                   .ASCII  "   *FAILED*"<CR>
                              00        02FC   856 FAST_WRITE:       .BYTE   0               ;FLAG 1=USE FAST WRITE ROUTINE
                              00        02FD   857 FILE_LOADED:      .BYTE   0               ;SECTION LOADED FROM TAPE
                              03        02FE   858 FPA_A:            .BYTE   3               ;CHAR COUNT
                        41 50 46        02FF   859                   .ASCII  "FPA"
                              3A        0302   860 HEADER_A:         .BYTE   58              ;CHAR COUNT
                              0D        0303   861                   .ASCII  <CR>
52 45 20 54 53 54 20 20 54 43 45 53     0304   862                   .ASCII  "SECT  TST ERR   EXP      REC      OTHER    "
20 20 20 20 20 50 58 45 20 20 20 52     0310
54 4F 20 20 20 20 20 20 43 45 52 20     031C
                  20 20 20 20 52 45 48  0328
4C 55 44 4F 4D 20 20 20 20 48 53 4D     032F   863                   .ASCII  "MSK    MODULE"<CR>
                        0D 45           0338
                              00        033D   864 HAVE_CNT:         .BYTE   0               ;FLAG 1=WCS LOOP COUNT READ
                              05        033E   865 IDC_A:            .BYTE   5               ;CHAR COUNT
                  30 33 37 42 52        033F   866                   .ASCII  "RB730"
                              00        0344   867 INSTR_LEN:         .BYTE   0               ;STORAGE FOR INSTRUCTION LENGTH
                           067A'        0345   868 INSTR_PNT:         .WORD   INSTR_TABLE     ;POINTER INTO INSTRUCTION TABLE
                              00        0347   869 LOOP_COUNTER:      .BYTE   0               ;LOOP COUNTER
                              00        0348   870 LS_SETPARERR:      .BYTE   0               ;PARITY ERROR SET BY CPU ATTN
                              00        0349   871 LS_PARERR_ACT:     .BYTE   0               ;PAR ERR ACTIVE (BEING PROCESSED)
                              00        034A   872 MAX_SBE:          .BYTE   0               ;MAXIMUM NUMBER OF SINGLE BIT
                              00        034B   873                   .BYTE   0               ; ERRORS ALLOWED
                              3F        034C   874                   .BYTE   63
                              06        034D   875 MEMORY_A:         .BYTE   6
                  59 52 4F 4D 45 40     034E   876                   .ASCII  "MEMORY"
                                        0354   877 MEM_REQ:          .BYTE   0               ;FLAG 1=MEM REQ TO BE ENABLED
                                        0355   878 MENU_A:           .BYTE   52
45 52 20 4F 54 20 31 20 45 50 5-        0356   879                   .ASCII  "TYPE 1 TO RETURN TO CONSOLE,"
53 4E 4F 43 20 4F 54 20 4E 52 55 54     0362
```

```
                                2C 45 4C 4F  036E
52 41 54 53 45 52 20 4F 54 20 32 20  0372    880                     .ASCII          " 2 TO RESTART AUTO TEST "
20 54 53 45 54 20 4F 54 55 41 20 54  037E
                                00  038A    881 MICRO_STEP:         .BYTE          0                ;FLAG NON 0=MICRO STEP MODE
                                00  038B    882 MINUS:              .BYTE          0                ;FLAG 1=NEG NUMBER TYPED
                                08  038C    883 MINUTES_A:          .BYTE          8                ;CHAR COUNT
                  53 45 54 55 4E 49 4D 20  038D    884                     .ASCII          " MINUTES"
                                00  0395    885 MOD_ERR:            .BYTE          0                ;FLAG 1=MODULE CALLED OUT
                                0F  0396    886 MOD_NOT_PRES_A:     .BYTE          15               ;CHAR COUNT
42 41 4C 49 41 56 41 20 54 4F 4E 20  0397    887                     .ASCII          " NOT AVAILABLE"<CR>
                             0D 45 4C  03A3
                                08  03A6    888 MOD_PRES_A:         .BYTE          8                ;CHAR COUNT
                  54 4E 45 53 45 52 50 20  03A7    889                     .ASCII          " PRESENT"
                                00  03AF    890 MORE_DATA:          .BYTE          0                ;FLAG 1=MORE DATA TO XFER
                                09  03B0    891 NA_A:               .BYTE          9                ;CHAR COUNT
            20 20 20 20 41 2F 4E 20 20  03B1    892                     .ASCII          "  N/A    "
                             076F'  03BA    893 NAME_PNT:           .WORD          NAME_TABLE       ;POINTER INTO TEST NAME TABLE
                                00  03BC    894 NEWSUM:             .BYTE          0                ;NEW CHECKSUM VALUE
                                00  03BD    895 NOADD_FLG:          .BYTE          0                ;FLAG 1=NO ADDRESS ON EXAM+DEP
                                00  03BE    896 NOINIT:             .BYTE          0                ;FLAG 1=NO INIT OF WCS (TEST0)
                                00  03BF    897 NO_RETURN:          .BYTE          0                ;FLAG 1=NO RETURN AFTER MICMON
                                    03C0    898                                                     ; ERROR
                                00  03C0    899 NO_SAVE_UPC_CSR:    .BYTE          0                ;FLAG 1= DON'T SAVE UPC OR CSR
                                00  03C1    900 NO_SPACE:           .BYTE          0                ;FLAG 1=NO SPACE AFTER PR HEX
                                00  03C2    901 NOT_FOUND:          .BYTE          0                ;FLAG 1=INPUT PARSED NOT FOUND
                          000003CD  03C3    902 OLD_TEMP_BUF:       .BLKB          10               ;BACKUP TEMP BUFF FOR ERRORS
                              0000  03CD    903 OLD_TTBUF:          .WORD          0                ;BACKUP POINTER TO TTBUF
                                00  03CF    904 OLDSUM:             .BYTE          0                ;OLD CHECKSUM VALUE
                                04  03D0    905 OR_A:               .BYTE          4                ;CHAR COUNT
                        20 52 4F 20  03D1    906                     .ASCII          " OR "
                                00  03D5    907 PACK_ERROR:         .BYTE          0                ;FLAG 1=WRONG RL01 PACK
                                00  03D6    908 PAR_ON:             .BYTE          0                ;FLAG 1=PARITY TO BE ENABLED
                              0000  03D7    909 PAR_VECT:           .WORD          0                ;PARSER ADDRESS DEST ADDRESS
                                00  03D9    910 PARDONE:            .BYTE          0                ;FLAG 1=END OF INPUT LINE
                              0000  03DA    911 PARSE_PNT:          .WORD          0                ;POINTER TO PARSE BUFFER
                              0000  03DC    912 PARSE_TABLE:        .WORD          0                ;POINTER TO PARSE TABLE
                                0B  03DE    913 PASS_A:             .BYTE          11
   0D 44 45 53 53 41 50 20 20 20 20  03DF    914                     .ASCII          "    PASSED"<CR>
                              0000  03EA    915 PASS_CNT:           .WORD          0                ;PASS COUNTER
                                00  03EC    916 PRINT_CNT.          .BYTE          0                ;NUMBER OF CHARS TO PRINT
                                00  03ED    917 PRINT_PASS:         .BYTE          0                ;FLAG 1=PRINT PASS MESSAGE
                              0000  03EE    918 PRINT_PNT:          .WORD          0                ;POINTER TO CHARS TO BE PRINTED
                                00  03F0    919 PRINT_START:        .BYTE          0                ;FLAG 1=PRINT START MESSAGE
                                00  03F1    920 RESTART:            .BYTE          0                ;FLAG 1=RESTART PARSER
0D 4F 43 20 31 30 20 45 54 20 49 44  03F2    921 RETRY_IDC2:         .ASCII          "DI TE 01 CO"<CR>
                                04  03FE    922 RL02_A:             .BYTE          4                ;CHAR COUNT
                     31 41 51 44  03FF    923                     .ASCII          "DQA1"
                                00  0403    924 RUNNING:            .BYTE          0                ;FLAG 1=CPU RUNNING MODE
                                00  0404    925 SECTION_E_NUM:      .BYTE          0                ;BOARD NUMBER FROM SEC. TABLE
                                00  0405    926 SETACK_FLG:         .BYTE          0                ;FLAG 1=SET CONSOLE ACKNOWLEDGE
                                00  0406    927 SKIP_CNT:           .BYTE          0                ;NUMBER OF CHARS SPANED
                              00C0  0407    928 STARTING_UPC:       .WORD          0                ;STARTING UPC ADDRESS
                              0000  0409    929 STEP_CNT:           .WORD          0                ;MICRO STEP COUNTER
                                00  040B    930 TABLE_MODE:         .BYTE          0                ;TABLE MODE INDICATOR
                              0000  040C    931 TABLE_PNT:          .WORD          0                ;SECTION TABLE POINTER
                          00000418  040E    932 TEMP_BUF:           .BLKB          10               ;TEMP CHAR BUFFER
                                00  0418    933 TEMP_BYTE:          .BYTE          0                ;TEMP STORAGE BYTE
```

```
                                0000  0419   934 TEMP_WORD:       .WORD           0                   ;TEMP STORAGE WORD
                                0000  041B   935 TEMP_WORD1:      .WORD           0                   ;TEMP STORAGE WORD
                                0000  041D   936 TEMP_WORD2:      .WORD           0                   ;TEMP STORAGE WORD
                                  00  041F   937 TEMP_FLAG:       .BYTE           0                   ;TEMP STORAGE OF SECTION FLAG
                          00000426  0420   938 TEMP_SECTION:    .BLKB           6                   ;TEMP STORAGE OF SECTION NAME
                                  00  0426   939 TEST_NUM:       .BYTE           0                   ;CURRENT TEST NUMBER
                                  00  0427   940 TEST_NUM1:      .BYTE           0                   ;STARTING TEST NUMBER
                                  00  0428   941 TEST_NUM2:      .BYTE           0                   ;ENDING TEST NUMBER
                                  1D  0429   942 TEST_START_A:   .BYTE           29                  ;CHAR COUNT
52 41 54 53 20 47 4E 49 54 53 45 54  042A   943                 .ASCII          "TESTING STARTED - RUN TIME = "
49 54 20 4E 55 52 20 2D 20 44 45 54  0436
                     20 3D 20 4F 4D  0442
                                  00  0447   944 TEST_ZERO:      .BYTE           0                   ;FLAG 1=EXECUTING TEST 0
                              0802'   0448   945 TIME_PNT:       .WORD           TIME_TABLE          ;POINTER INTO TEST TIME TABLE
                                0000  044A   946 TIMOUT_CNT:     .WORD           0                   ;TIME OUT COUNTER
                                  00  044C   947 TRACE:          .BYTE           0                   ;FLAG 1=TRACE MODE
                                0000  044D   948 TTBUF_PNT:      .WORD           0                   ;POINTER TO TTY INPUT BUFFER
                                  63  044F   949 VERSION:        .BYTE           99                  ;CHAR COUNT
45 42 20 20 20 20 20 2A 2A 2A 2A 0D  0450   950                 .ASCII          <CR>"****     BEGIN VAX-11/730,725 AUTO TEST     ****"
37 2F 31 31 2D 58 41 56 20 4E 49 47  045C
20 4F 54 55 41 20 35 32 37 2C 30 33  0468
2A 2A 2A 20 20 20 20 20 54 53 45 54  0474
                                 2A  0480
35 2E 31 20 4E 4F 49 53 52 45 56 0D  0481   951                 .ASCII          <CR>"VERSION 1.5 RUN TIME APPROXIMATELY 15:00 MINUTES"<CR>
20 41 20 45 4D 49 54 20 4E 55 52 20  048D
20 59 4C 45 54 41 4D 49 58 4F 52 50  0499
45 54 55 4E 49 4D 20 30 30 3A 35 31  04A5
                              0D 53  04B1
                                 00  04B3   952 WARM_RESTART:   .BYTE           0                   ;FLAG 1=POWER FAIL RESTART
                                 D0  04B4   953 WARN_MESS:      .BYTE           208                 ;CHAR COUNT
53 20 52 45 50 4F 52 50 20 2A 2A 0D  04B5   954                 .ASCII          <CR>"** PROPER SETUP REQUIRED - TYPE <CR> TO "
45 52 49 55 51 45 52 20 50 55 54 45  04C1
52 43 3C 20 45 50 59 54 20 2D 20 44  04CD
                     20 4F 54 20 3E  04D9
   2A 2A 20 45 55 4E 49 54 4E 4F 43  04DE   955                 .ASCII          "CONTINUE **"
20 4B 43 45 48 43 20 20 20 2A 2A 0D  04E9   956                 .ASCII          <CR>"**   CHECK FOR RL02 DIAGNOSTIC PACK "
41 49 44 20 32 3C 4C 52 20 52 4F 46  04F5
4B 43 41 50 20 43 49 54 53 4F 4E 47  0501
                                 20  050D
20 20 20 27 4B 43 41 50 44 52 43 27  050E   957                 .ASCII          "'CRDPACK'    **"
                              2A 2A  051A
53 4E 49 20 20 20 20 20 20 2A 2A 0D  051D   958                 .ASCII          <CR>"**       INSERTED AND SPINNING IN DRIVE "
50 53 20 44 4E 41 20 44 45 54 52 45  0529
52 44 20 4E 49 20 47 4E 49 4E 4E 49  0535
                     20 45 56 49 56  0541
2A 2A 20 20 20 20 20 3A 31 41 51 44  0545   959                 .ASCII          "DQA1:     **"
50 55 54 45 53 20 46 49 20 2A 2A 0D  0551   960                 .ASCII          <CR>"** IF SETUP CORRECT, POSSIBLE RL02 DRIVE "
4F 50 20 2C 54 43 45 52 52 4F 43 20  055D
20 32 3C 4C 52 20 45 4C 42 49 53 53  0569
                     20 45 56 49 52  0575
      2A 2A 20 4D 45 4C 42 4F 52 50  057B   961                 .ASCII          "PROBLEM **"
                              04 57  0585   962 WCS_A:          .BYTE           4                   ;CHAR COUNT
                        20 53 43 57  0586   963                 .ASCII          "WCS "
                                 00  058A   964 WCS_CALC:       .BYTE           0                   ;FLAG 1=WCS PARITY CALCULATION
                                 00  058B   965 WCS_LOADED:     .BYTE           0                   ;FLAG 1=WCS OVERLAY LOADED
                                 00  058C   966 WCS_PNT:        .BYTE           0                   ;COUNT FOR WCS 3 BYTE LOADS
                           0000      058D   967 XFER_CNT:       .WORD           0                   ;LONG WORD XFER COUNTER
```

```
                                00000592  058F   968 XFER_DEST:      .BLKB           3               ;ADDR DATA XFER TO
                                      00  0592   969 XFER_MM:        .BYTE           0               ;FLAG FOR MM OR LS XFER
                                    0000  0593   970 XFER_PNT:       .WORD           0
                                      00  0595   971 ZERO_BYTE:      .BYTE           0
                                0000059A  0596   972 ZERO_WORD:      .BLKB           4
                                      2D  059A   973 warn_mesa:      .byte           45              ;CHAR COUNT
50 59 54 20 32 30 4C 52 20 2A 2A 0D     059B   974                    .ascii          <CR>"** RL02 TYPE 1 <CR> TO CONTINUE TESTING  "
20 4F 54 20 3E 52 43 3C 20 31 20 45     05A7
53 45 54 20 45 55 4E 49 54 4E 4F 43     05B3
                  20 20 47 4E 49 54     05BF
                        2A 2A 09        05C5   975                    .ascii          "       **"
                                  68    05C8   976 WARN_MESB:        .BYTE           104             ;CHAR COUNT
20 45 56 49 52 44 20 46 49 2A 2A 0D     05C9   977                    .ASCII          <CR>"**IF DRIVE 1 IS A TS05 THEN IGNORE ERROR "
20 35 30 53 54 20 41 20 53 49 20 31     05D5
20 45 52 4F 4E 47 49 20 4E 45 48 54     05E1
                  20 52 4F 52 52 45     05ED
      2A 2A 20 45 47 41 53 53 45 4D     05F3   978                    .ASCII          "MESSAGE **"
45 50 59 54 20 44 4E 41 20 2A 2A 0D     05FD   979                    .ASCII          <CR>"** AND TYPE 2<CR> TO CONTINUE IF DRIVE 1 "
4F 43 20 4F 54 20 3E 52 43 3C 32 20     0609
52 44 20 46 49 20 45 55 4E 49 54 4E     0615
                  20 31 20 45 56 49     0621
      2A 2A 20 20 20 20 41 20 53 49     0627   980                    .ASCII          "IS A    **"
                              31        0631   981 VERSION_1:        .BYTE           49              ;CHAR COUNT
54 53 55 43 20 20 20 2A 2A 2A 2A 0D     0632   982                    .ASCII          <CR>"****   CUSTOMER RUNNABLE MICRODIAGNOSTICS   ****"
4C 42 41 4E 4E 55 32 20 52 45 4D 4F     063E
4E 47 41 49 44 4F 52 43 49 4D 20 45     064A
2A 2A 2A 20 20 20 53 43 49 54 53 4F     0656
                              2A        0662
                                        0663   983
                                        0663   984
                                        0663   985
                                        0663   986 ;CSR EXECUTABLE CODE
                                        0663   987 ;NOTE:  THE INSTRUCTIONS THAT ARE PAIRED WITH S_XXXX ARE DYNAMICLY MODIFIED
                                        0663   988 ;       DURING EXECUTION.  THUS THE ADDRESS 0 WAS ASSEMBLED INTO THEM. THIS
                                        0663   989 ;       ADDRESS MY BE MODIFYED
                                        0663   990
                                    01  0663   991 X_READ_LS:       .BYTE           1
                                    36  0664   992                    .BYTE           <^X36>          ;MOV LS[0] TO WR[0]
                                    00  0665   993                    .BYTE           <^X00>
                                    15  0666   994                    .BYTE           <^X15>
                                        0667   995
                                    36  0667   996 S_READ_LS:       .BYTE           <^X36>          ;MOV LS[0] TO WR[0]
                                    00  0668   997                    .BYTE           <^X00>
                                    15  0669   998                    .BYTE           <^X15>
                                        066A   999
                                        066A  1000
                                    01  066A  1001 X_CLR_CPU_ATT:   .BYTE           1               ;INSTRUCTION COUNT
                                    90  066B  1002                    .BYTE           <^X90>          ;MISC [CLR.CP.ATTN.AND.ACK]
                                    C0  066C  1003                    .BYTE           <^XC0>
                                    15  066D  1004                    .BYTE           <^X15>
                                        066E  1005
                                        066E  1006
                                    01  066E  1007 X_WRITE_LS:      .BYTE           1               ;INSTRUCTION COUNT
                                    BE  066F  1008                    .BYTE           <^XBE>          ;MOV WR[0] TO LS[0]
                                    00  0670  1009                    .BYTE           <^X00>
                                    15  0671  1010                    .BYTE           <^X15>
                                        0672  1011
```

```
                     BE  0672  1012 S_WRITE_LS:     .BYTE           <^XBE>          ;MOV WR[0] TO LS[0]
                     00  0673  1013                 .BYTE           <^X00>
                     15  0674  1014                 .BYTE           <^X15>
                         0675  1015
                         0675  1016
                         0675  1017
                         0675  1018 ;*****************************************************************************
                         0675  1019 ;
                         0675  1020 ;  INPUT HEX NUMBER AREA
                         0675  1021 ;
                         0675  1022 ;*****************************************************************************
                         0675  1023
                     04  0675  1024 NUM_SHIFT:      .BYTE           4               ;4 BYTES OF HEX DIGITS
               0000067A  0676  1025 INPUT_NUM:      .BLKB           4
                         067A  1026
                         067A  1027 ;*****************************************************************************
                         067A  1028 ;
                         067A  1029 ;  INSTRUCTION STREAM TABLE
                         067A  1030 ;  THIS TABLE IS THE INSTRUCTION STREAM TO BE USED TO INPUT THE COMMANDS
                         067A  1031 ;  TO MICMON.  THE BIT 0 OF THE FIRST BYTE BEFORE THE INSTRUCTION IS USED TO
                         067A  1032 ;  INDICATE WHETHER THE CRD SHOULD PRINT A 'TESTING STARTED' MESSAGE, AND
                         067A  1033 ;  BIT 1 INDICATES WHETHER TO PRINT THE 'PASSED' MESSAGE UPON COMPLETION.
                         067A  1034 ;  THE SECOND BYTE BEFORE THE INSTRUCTION IS THE CHARACTER COUNT OF THE
                         067A  1035 ;  INSTRUCTION.
                         067A  1036 ;
                         067A  1037 ;*****************************************************************************
                         067A  1038
                     01  067A  1039 INSTR_TABLE:    .BYTE           1
                     13  067B  1040                 .BYTE           19              ;CHAR COUNT
20 41 42 4B 4E 45 20 45  067C  1041                 .ASCII          'DI SE ENKBA TE 1 4'<CR>
                  0D 34 20 31 20 45 54  0688
                     00  068F  1042                 .BYTE           0
                     0A  0690  1043                 .BYTE           10              ;CHAR COUNT
      0D 38 20 36 20 45 54 20 49 44  0691  1044     .ASCII          'DI TE 6 8'<CR>
                     00  069B  1045                 .BYTE           0
                     13  069C  1046                 .BYTE           19              ;CHAR COUNT
20 42 42 4B 4E 45 20 45  069D  1047                 .ASCII          'DI SE ENKBB TE 9 D'<CR>
                  0D 44 20 39 20 45 54  06A9
                     00  06B0  1048                 .BYTE           0
                     0A  06B1  1049                 .BYTE           10              ;CHAR COUNT
      0D 46 20 46 20 45 54 20 49 44  06B2  1050     .ASCII          'DI TE F F'<CR>
                     00  06BC  1051                 .BYTE           0
                     0C  06BD  1052                 .BYTE           12              ;CHAR COUNT
0D 43 42 4B 4E 45 20 45 53 20 49 44  06BE  1053     .ASCII          'DI SE ENKBC'<CR>
                     00  06CA  1054                 .BYTE           0
                     0C  06CB  1055                 .BYTE           12              ;CHAR COUNT
0D 44 42 4B 4E 45 20 45 53 20 49 44  06CC  1056     .ASCII          'DI SE ENKBD'<CR>
                     00  06D8  1057                 .BYTE           0
                     0C  06D9  1058                 .BYTE           12              ;CHAR COUNT
0D 45 42 4B 4E 45 20 45 53 20 49 44  06DA  1059     .ASCII          'DI SE ENKBE'<CR>
                     00  06E6  1060                 .BYTE           0
                     0C  06E7  1061                 .BYTE           12              ;CHAR COUNT
0D 41 43 4B 4E 45 20 45 53 20 49 44  06E8  1062     .ASCII          'DI SE ENKCA'<CR>
                     02  06F4  1063                 .BYTE           2
                     0C  06F5  1064                 .BYTE           12              ;CHAR COUNT
0D 42 43 4B 4E 45 20 45 53 20 49 44  06F6  1065     .ASCII          'DI SE ENKCB'<CR>
                     03  0702  1066                 .BYTE           3
```

```
                     14  0703  1067                    .BYTE           20                              ;CHAR COUNT
20 43 43 4B 4E 45 20 45 53 20 49 44  0704  1068                    .ASCII          'DI SE ENKCC TE 1 32'<CR>
               0D 32 33 20 31 20 45 54  0710
                     03  0718  1069                    .BYTE           3
                     0C  0719  1070                    .BYTE           12                              ;CHAR COUNT
0D 44 43 4B 4E 45 20 45 53 20 49 44  071A  1071                    .ASCII          'DI SE ENKCD'<CR>
                     01  0726  1072                    .BYTE           1
                     0C  0727  1073                    .BYTE           12                              ;CHAR COUNT
0D 45 43 4B 4E 45 20 45 53 20 49 44  0728  1074                    .ASCII          'DI SE ENKCE'<CR>
                     02  0734  1075                    .BYTE           2
                     0C  0735  1076                    .BYTE           12                              ;CHAR COUNT
0D 48 43 4B 4E 45 20 45 53 20 49 44  0736  1077                    .ASCII          'DI SE ENKCH'<CR>
                     03  0742  1078                    .BYTE           3
                     0C  0743  1079                    .BYTE           12                              ;CHAR COUNT
0D 46 43 4B 4E 45 20 45 53 20 49 44  0744  1080                    .ASCII          'DI SE ENKCF'<CR>
                     03  0750  1081                    .BYTE           3
                     14  0751  1082                    .BYTE           20                              ;CHAR COUNT
20 47 43 4B 4E 45 20 45 53 20 49 44  0752  1083                    .ASCII          'DI SE ENKCG TE F 10'<CR>
               0D 30 31 20 46 20 45 54  075E
                     01  0766  1084 RETURN_I:              .BYTE           1
                     07  0767  1085                    .BYTE           7                               ;CHAR COUNT
                  0D 4E 52 55 54 45 52  0768  1086                    .ASCII          'RETURN'<CR>
                                        076F  1087
                                        076F  1088 ;*****************************************************************************
                                        076F  1089 ;
                                        076F  1090 ;  TEST NAME TABLE
                                        076F  1091 ;  THIS TABLE CONTAINS THE TEXT FOR THE PROGRESS MESSAGES THAT INDICATE
                                        076F  1092 ;  THE TYPE OF TESTING THAT IS BEING DONE.  ALL STRINGS MUST BE 20
                                        076F  1093 ;  CHARACTERS IN LENGTH TO INSURE THAT THE PRINTOUT LINES UP PROPERLY.
                                        076F  1094 ;
                                        076F  1095 ;*****************************************************************************
                                        076F  1096
                     14  076F  1097 NAME_TABLE:          .BYTE           20                      ;CHAR COUNT
20 31 20 54 52 41 50 20 55 50 43 0D  0770  1098                    .ASCII          <CR>''CPU PART 1        ''
               20 20 20 20 20 20 20 20  077C
                     14  0784  1099                    .BYTE           20
20 20 20 20 20 59 52 4F 4D 45 4D 0D  0785  1100                    .ASCII          <CR>''MEMORY            ''
               20 20 20 20 20 20 20 20  0791
                     14  0799  1101                    .BYTE           20
20 32 20 54 52 41 50 20 55 50 43 0D  079A  1102                    .ASCII          <CR>''CPU PART 2        ''
               20 20 20 20 20 20 20 20  07A6
                     14  07AE  1103                    .BYTE           20
20 20 20 20 20 20 20 20 41 50 46 0D  07AF  1104                    .ASCII          <CR>''FPA               ''
               20 20 20 20 20 20 20 20  07BB
                     14  07C3  1105                    .BYTE           20
20 20 20 20 20 20 30 33 37 42 52 0D  07C4  1106                    .ASCII          <CR>''RB730             ''
               20 20 20 20 20 20 20 20  07D0
                     14  07D8  1107                    .BYTE           20
31 20 54 52 41 50 20 32 30 4C 52 0D  07D9  1108                    .ASCII          <CR>''RL02 PART 1   DQA1 ''
               20 31 41 51 44 20 20 20  07E5
                     14  07ED  1109 RETURN_N:              .BYTE           20
20 33 20 54 52 41 50 20 55 50 43 0D  07EE  1110                    .ASCII          <CR>''CPU PART 3        ''
               20 20 20 20 20 20 20 20  07FA
                                        0802  1111
                                        0802  1112 ;*****************************************************************************
                                        0802  1113 ;
                                        0802  1114 ;  TEST TIME TABLE
```

```
                        0802  1115 ;  THIS TABLE CONTAINS THE APPROXIMATE RUN TIMES FOR EACH TYPE OF CRD
                        0802  1116 ;  TESTING.  ALL TIMES MUST CONSIST OF A 4 CHARACTER STRING.
                        0802  1117 ;
                        0802  1118 ;*******************************************************************************
                        0802  1119
                     04 0802  1120 TIME_TABLE:      .BYTE           4                       ;CHAR COUNT
           30 33 3A 33  0803  1121                  .ASCII          "3:30"
                     04 0807  1122                  .BYTE           4
           30 33 3A 31  0808  1123                  .ASCII          "1:30"
                     04 080C  1124                  .BYTE           4
           35 34 3A 30  080D  1125                  .ASCII          "0:45"
                     04 0811  1126                  .BYTE           4
           35 34 3A 31  0812  1127                  .ASCII          "1:45"
                     04 0816  1128                  .BYTE           4
           35 34 3A 30  0817  1129                  .ASCII          "0:45"
                     04 081B  1130                  .BYTE           4
           35 31 3A 30  081C  1131                  .ASCII          "0:15"
                     04 0820  1132 RETURN_T:        .BYTE           4
           30 30 3A 34  0821  1133                  .ASCII          "4:00"
                        0825  1134
                        0825  1135
                        0825  1136 ;*******************************************************************************
                        0825  1137 ;
                        0825  1138 ;  CRD OPTION TABLE
                        0825  1139 ;  THIS TABLE LISTS THE ASCII TEXT THAT IS TO BE DISPLAYED IN THE EVENT THAT
                        0825  1140 ;  THE DEVICE IS FOUND TO BE BAD.  THE CPU IS NOT INCLUDED.  EACH ENTRY
                        0825  1141 ;  MUST CONSIST OF 7 BYTES.  THE FIRST INDICATES THE CHARACTER COUNT OF
                        0825  1142 ;  THE MODULE NAME.  THE ASCII TEXT COMES NEXT, FOLLOWED A BLOCK OF THE
                        0825  1143 ;  APPROPRIATE SIZE TO MAKE THE ENTRY EQUAL TO 7 BYTES.
                        0825  1144 ;
                        0825  1145 ;*******************************************************************************
                        0825  1146
                     03 0825  1147 CRD_OPT_A:       .BYTE           3
              41 50 46  0826  1148                  .ASCII          "FPA"
              0000082C  0829  1149                  .BLKB           3
                     06 082C  1150                  .BYTE           6
        59 52 4F 4D 45 4D 082D  1151                  .ASCII          "MEMORY"
                     05 0833  1152                  .BYTE           5
           30 33 37 42 52  0834  1153                  .ASCII          "RB730"
                     00 0839  1154                  .BYTE           0
                     04 083A  1155                  .BYTE           4
              31 41 51 44  083B  1156                  .ASCII          "DQA1"
              00000841     083F  1157                  .BLKB           2
                     04 0841  1158                  .BYTE           4
              30 41 51 44  0842  1159                  .ASCII          "DQA0"
                        0846  1160
                        0846  1161
                        0846  1162 ;*******************************************************************************
                        0846  1163 ;
                        0846  1164 ;  BOARD NAME TABLE
                        0846  1165 ;
                        0846  1166 ;*******************************************************************************
                        0846  1167
                     01 0846  1168 BOARD_TABLE:     .BYTE           1                       ;BOARD NUMBER
              53 43 57  0847  1169                  .ASCII          "WCS"
                     02 084A  1170                  .BYTE           2                       ;BOARD NUMBER
              50 41 44  084B  1171                  .ASCII          "DAP"
```

```
                     02  084E  1172                .BYTE           2                       ;BOARD NUMBER
                  55 50  43  084F  1173                .ASCII          "CPU"
                     04  0852  1174                .BYTE           4                       ;BOARD NUMBER
                  54 43  4D  0853  1175                .ASCII          "MCT"
                     08  0856  1176                .BYTE           8                       ;BOARD NUMBER
                  41 50  46  0857  1177                .ASCII          "FPA"
                     10  085A  1178                .BYTE           <^X10>                  ;BOARD NUMBER
                  43 44  49  085B  1179                .ASCII          "IDC"
                     FF  085E  1180                .BYTE           <^XFF>                  ;END OF BOARD TABLE
                         085F  1181
                         085F  1182 ;*************************************************************************
                         085F  1183 ;
                         085F  1184 ;  SECTION NAME TABLE (NOTE: ALL ENTRIES MUST BE THE SAME NAME LENGTH (6))
                         085F  1185 ;  AND ALL FILL CHARS MUST BE ZERO'S (BINARY ZEROS)
                         085F  1186 ;  THE BYTE BEFORE EACH NAME IS USED TO SET SECTION FLAG AS FOLLOWS
                         085F  1187 ;
                         085F  1188 ;  SECTION_FLAG           FF = END OF TABLE
                         085F  1189 ;                  BIT 7   0 = NAME IS SECTION NAME FOR WCS
                         085F  1190 ;                          1 = NAME IS SECTION NAME FOR CONSOLE RAM
                         085F  1191 ;                  BIT 6   1 = OPTIONAL TEST
                         085F  1192 ;            BITS 0 - 5   XX = CPU CLUSTER BOARD NUMBER
                         085F  1193 ;
                         085F  1194 ;
                         085F  1195 ;*************************************************************************
                         085F  1196
                     83  085F  1197 SEC_TABLE:       .BYTE           <^X83>                  ;8085 BASED CPU TESTS 1-8
            41 42 4B 4E  45  0860  1198                .ASCII          "ENKBA"
                     00  0865  1199                .BYTE           0
                         0866  1200
                     83  0866  1201                .BYTE           <^X83>                  ;8085 BASED CPU TESTS 9-F
            42 42 4B 4E  45  0867  1202                .ASCII          "ENKBB"
                     00  086C  1203                .BYTE           0
                         086D  1204
                     83  086D  1205                .BYTE           <^X83>                  ;8085 BASED CPU TESTS 10-1D
            43 42 4B 4E  45  086E  1206                .ASCII          "ENKBC"
                     00  0873  1207                .BYTE           0
                         0874  1208
                     81  0874  1209                .BYTE           <^X81>                  ;8085 BASED CPU TESTS 1E-26
            44 42 4B 4E  45  0875  1210                .ASCII          "ENKBD"
                     00  087A  1211                .BYTE           0
                         087B  1212
                     81  087B  1213                .BYTE           <^X81>                  ;8085 BASED CPU TESTS 27-2E
            45 42 4B 4E  45  087C  1214                .ASCII          "ENKBE"
                     00  0881  1215                .BYTE           0
                         0882  1216
                     C0  0882  1217                .BYTE           <^XC0>                  ;8085 BASED WCS MARCH TEST.
            46 42 4B 4E  45  0883  1218                .ASCII          "ENKBF"                 ; (OPTIONAL)
                     00  0888  1219                .BYTE           0
                         0889  1220
                     C0  0889  1221                .BYTE           <^XC0>                  ;8085 BASED APT UART TEST
            47 42 4B 4E  45  088A  1222                .ASCII          "ENKBG"                 ; (OPTIONAL)
                     00  088F  1223                .BYTE           0
                         0890  1224
                     02  0890  1225                .BYTE           <^X02>                  ;WCS BASED CPU TEST
            41 43 4B 4E  45  0891  1226                .ASCII          "ENKCA"
                     00  0896  1227                .BYTE           0
                         0897  1228
```

```
                     02  0897  1229                .BYTE           <^X02>          ;WCS BASED CPU TEST
          42 43 4B 4E 45  0898  1230                .ASCII          "ENKCB"
                     00  089D  1231                .BYTE           0
                         089E  1232
                     04  089E  1233                .BYTE           <^X04>          ;WCS BASED MCT TEST
          43 43 4B 4E 45  089F  1234                .ASCII          "ENKCC"
                     00  08A4  1235                .BYTE           0
                         08A5  1236
                     02  08A5  1237                .BYTE           <^X02>          ;WCS BASED CPU TEST
          44 43 4B 4E 45  08A6  1238                .ASCII          "ENKCD"
                     00  08AB  1239                .BYTE           0
                         08AC  1240
                     08  08AC  1241                .BYTE           <^X08>          ;WCS BASED FPA TEST
          45 43 4B 4E 45  08AD  1242                .ASCII          "ENKCE"
                     00  08B2  1243                .BYTE           0
                         08B3  1244
                     10  08B3  1245                .BYTE           <^X10>          ;WCS BASED IDC TEST
          46 43 4B 4E 45  08B4  1246                .ASCII          "ENKCF"
                     00  08B9  1247                .BYTE           0
                         08BA  1248
                     10  08BA  1249                .BYTE           <^X10>          ;WCS BASED IDC TEST
          47 43 4B 4E 45  08BB  1250                .ASCII          "ENKCG"
                     00  08C0  1251                .BYTE           0
                         08C1  1252
                     10  08C1  1253                .BYTE           <^X10>          ;WCS BASED IDC TEST
          48 43 4B 4E 45  08C2  1254                .ASCII          "ENKCH"
                     00  08C7  1255                .BYTE           0
                         08C8  1256
                     FF  08C8  1257                .BYTE           <^XFF>          ;END OF TABLE
                         08C9  1258
                     00  08C9  1259 SECTION_FLAG:    .BYTE           0               ;SECTION TYPE AND BOARD NUM
           00000008D0    08CA  1260 SECTION_NAME:    .BLKB           SEC_LEN         ;SAVE BYTES FOR NAME
                         08D0  1261
                         08D0  1262 ;*****************************************************************************
                         08D0  1263 ;
                         08D0  1264 ;  INPUT BUFFER AREA
                         08D0  1265 ;
                         08D0  1266 ;*****************************************************************************
                         08D0  1267
                     00  08D0  1268 TTBUF_CNT:       .BYTE           0               ;CHAR CNT OF INPUT LINE PUT
                         08D1  1269                                                  ; HERE BY GET LINE ROUTINE
           00000921      08D1  1270 TTBUF:           .BLKB           80              ;TTY BUFFER
                         0921  1271
                         0921  1272 ;*****************************************************************************
                         0921  1273 ;
                         0921  1274 ;  UPC VERIFY ADDRESS TABLE
                         0921  1275 ;
                         0921  1276 ;  THIS TABLE OF VALUES IS COMPARED TO ACTUAL UPC VALUES TO VERIFY
                         0921  1277 ;  THE WRITE OF DATA TO THE CPU IS WORKING CORRECTLY
                         0921  1278 ;
                         0921  1279 ;*****************************************************************************
                         0921  1280
                   0601  0921  1281 WRITE_TAB:       .WORD           <^X601>
                   0602  0923  1282                .WORD           <^X602>
                   0603  0925  1283                .WORD           <^X603>
                   0604  0927  1284                .WORD           <^X604>
                   0606  0929  1285                .WORD           <^X606>
```

```
                0603  092B  1286                    .WORD           <^X603>
                0605  092D  1287                    .WORD           <^X605>
                0606  092F  1288                    .WORD           <^X606>
                      0931  1289 ;*****************************************************************************
                      0931  1290 ;
                      0931  1291 ;  CHAR STRINGS AND JUMP ADDRESSES FOR PARSING AT COMMAND LEVEL
                      0931  1292 ;
                      0931  1293 ;*****************************************************************************
                      0931  1294
                   02 0931  1295 MAIN_TAB:         .BYTE   2
                49 44 0932  1296                    .ASCII  "DI"            ;DI MUST COME B4 D!!!!!
                      0934  1297                    CALL    DIAG_COMMAND    ;DIAGNOSE COMMAND
                   CD 0934                  .BYTE   ^O315
                0998' 0935                  .WORD   <DIAG_COMMAND-HEAD>
                      0937  1298                    RET
                   C9 0937                  .BYTE   ^O311
                      0938  1299
                   01 0938  1300                    .BYTE   1
                   43 0939  1301                    .ASCII  "C"
                      093A  1302                    CALL    CONT_COMMAND    ;CONTINUE COMMAND
                   CD 093A                  .BYTE   ^O315
                0B46' 093B                  .WORD   <CONT_COMMAND-HEAD>
                      093D  1303                    RET
                   C9 093D                  .BYTE   ^O311
                      093E  1304
                   02 093E  1305                    .BYTE   2
                45 52 093F  1306                    .ASCII  "RE"            ;RETURN COMMAND
                      0941  1307                    JMP     STNDRD_CON_COM_SUC      ;RETURN TO STANDARD CONSOLE
                   C3 0941                  .BYTE   ^O303
                0991' 0942                  .WORD   <STNDRD_CON_COM_SUC-HEAD>
                      0944  1308                    RET
                   C9 0944                  .BYTE   ^O311
                      0945  1309
                   02 0945  1310                    .BYTE   2
                44 4C 0946  1311                    .ASCII  "LD"            ;LOAD SECTION (NO EXECUTE)
                      0948  1312                    CALL    LOADER
                   CD 0948                  .BYTE   ^O315
                0B39' 0949                  .WORD   <LOADER-HEAD>
                      094B  1313                    RET
                   C9 094B                  .BYTE   ^O311
                      094C  1314
                   01 094C  1315                    .BYTE   1
                   49 094D  1316                    .ASCII  "I"             ;INIT (EXEC WCS TEST 0)
                      094E  1317                    CALL    EXEC_TEST0
                   CD 094E                  .BYTE   ^O315
                1BA1' 094F                  .WORD   <EXEC_TEST0-HEAD>
                      0951  1318                    RET
                   C9 0951                  .BYTE   ^O311
                      0952  1319
                   02 0952  1320                    .BYTE   2
                42 41 0953  1321                    .ASCII  "AB"            ;ABORT (FUTURE EXPANSION)
                      0955  1322                    JMP     START_UP
                   C3 0955                  .BYTE   ^O303
                1CD2' 0956                  .WORD   <START_UP-HEAD>
                      0958  1323                    RET
                   C9 0958                  .BYTE   ^O311
                      0959  1324
```

```
                    01  0959  1325                    .BYTE   1
                    0D  095A  1326                    .ASCII  <CR>            ;END ON LINE HAS <CR>
                        095B  1327                    RET                     ;FILLER--DO NOT REMOVE!!!
                    C9  095B                  .BYTE   ^O311
                        095C  1328                    RET                     ;FILLER--DO NOT REMOVE!!!
                    C9  095C                  .BYTE   ^O311
                        095D  1329                    RET                     ;FILLER--DO NOT REMOVE!!!
                    C9  095D                  .BYTE   ^O311
                        095E  1330                    RET
                    C9  095E                  .BYTE   ^O311
                        095F  1331
                    FF  095F  1332                    .BYTE   <^XFF>          ;END OF TABLE
                        0960  1333
                        0960  1334 ;*************************************************************************
                        0960  1335 ;
                        0960  1336 ;  CHAR STRINGS AND JUMP ADDRESSES FOR PARSING THE DIAGNOSE COMMAND
                        0960  1337 ;
                        0960  1338 ;*************************************************************************
                        0960  1339
                    02  0960  1340 DIAG_TAB:         .BYTE   2
                4F  42  0961  1341                    .ASCII  "BO"
                        0963  1342                    CALL    DIAG_BOARD      ;DIAGNOSE BOARD
                    CD  0963                  .BYTE   ^O315
                 0A0F'  0964                  .WORD   <DIAG_BOARD-HEAD>
                        0966  1343                    RET
                    C9  0966                  .BYTE   ^O311
                        0967  1344
                    02  0967  1345                    .BYTE   2
                45  53  0968  1346                    .ASCII  "SE"
                        096A  1347                    CALL    DIAG_SECTION    ;DIAGNOSE SECTION
                    CD  096A                  .BYTE   ^O315
                 0A5D'  096B                  .WORD   <DIAG_SECTION-HEAD>
                        096D  1348                    RET
                    C9  096D                  .BYTE   ^O311
                        096E  1349
                    02  096E  1350                    .BYTE   2
                45  54  096F  1351                    .ASCII  "TE"
                        0971  1352                    CALL    DIAG_TEST       ;DIAGNOSE TEST
                    CD  0971                  .BYTE   ^O315
                 0ACA'  0972                  .WORD   <DIAG_TEST-HEAD>
                        0974  1353                    RET
                    C9  0974                  .BYTE   ^O311
                        0975  1354
                    02  0975  1355                    .BYTE   2
                4F  43  0976  1356                    .ASCII  "CO"
                        0978  1357                    CALL    DIAG_CONTIN     ;DIAGNOSE CONTINUE
                    CD  0978                  .BYTE   ^O315
                 0B0E'  0979                  .WORD   <DIAG_CONTIN-HEAD>
                        097B  1358                    RET
                    C9  097B                  .BYTE   ^O311
                        097C  1359
                    02  097C  1360                    .BYTE   2
                41  50  097D  1361                    .ASCII  "PA"
                        097F  1362                    CALL    DIAG_PASS       ;DIAGNOSE PASS
                    CD  097F                  .BYTE   ^O315
                 0B14'  0980                  .WORD   <DIAG_PASS-HEAD>
                        0982  1363                    RET
```

```
                C9  0982                .BYTE   ^O311
                    0983  1364
                01  0983  1365          .BYTE   1
                00  0984  1366          .BYTE   0                   ;END OF LINE HAS 0 AFTER <CR>
                    0985  1367          CALL    SET_PARDONE
                CD  0985                .BYTE   ^O315
              1C06' 0986                .WORD   <SET_PARDONE-HEAD>
                    0988  1368          RET
                C9  0988                .BYTE   ^O311
                    0989  1369
                FF  0989  1370          .BYTE   <^XFF>              ;END OF TABLE
                    098A  1371
                    098A  1372
                    098A  1373 BEGIN_CHECKSUM:
                    098A  1374 ;*****************************************************************************
                    098A  1375 ;
                    098A  1376 ;  COMMAND SUBROUTINES
                    098A  1377 ;
                    098A  1378 ;*****************************************************************************
                    098A  1379
                    098A  1380 ;*****************************************************************************
                    098A  1381 ;
                    098A  1382 ;       STNDRD_CON_COM
                    098A  1383 ;       THIS ROUTINE IS ENTERED IF A RE, T/E, OR S/U 0 COMMAND IS ISSUED
                    098A  1384 ;       FROM THE CONSOLE DEVICE.  IT WILL INITIALIZE THE INTERVAL TIMER
                    098A  1385 ;       BY DOING A MASTER RESET TO IT AND THEN GO TO ROM ADDRESS 0.
                    098A  1386 ;
                    098A  1387 ;*****************************************************************************
                    098A  1388
                    098A  1389 STNDRD_CON_COM:
                    098A  1390                  LXI     H,CRD_FLAGS     ;SET BIT 2 OF CRD_FLAGS TO INDICATE
                21  098A                .BYTE   <H*8> ! 1
              40 FD 098B                .BYTE   <CRD_FLAGS-HEAD>&255 , <<CRD_FLAGS-HEAD>&^XFF00>/256
                    098D  1391          MVI     A,HEX_4             ; UNSUCCESSFUL CRD RUN
              04 3E 098D                .BYTE   <A*8> ! 6 , HEX_4&255
                    098F  1392          ORA     M
                B6  098F                .BYTE   ^O260 ! M
                    0990  1393          MOV     M,A
                77  0990                .BYTE   ^O100 ! <M*8> ! A
                    0991  1394
                    0991  1395 STNDRD_CON_COM_SUC:
                    0991  1396          MVI     A,HEX_FF            ;LOAD ACCUM WITH FF
              FF 3E 0991                .BYTE   <A*8> ! 6 , HEX_FF&255
                    0993  1397          OUT     ITCSR               ;SEND FF TO INTERVAL TIMER (MASTER
              1D D3 0993                .BYTE   ^O323 , ITCSR&255   ;register A
                    0995  1398                                      ; RESET)
                    0995  1399          JMP     RET_VECTOR          ;RETURN TO CONSOLE
                C3  0995                .BYTE   ^O303
              414D  0996                .WORD   <RET_VECTOR-HEAD>
                    0998  1400
                    0998  1401
                    0998  1402 ;*****************************************************************************
                    0998  1403 ;
                    0998  1404 ;  DIAG_COMMAND
                    0998  1405 ;  THE DIAGNOSE COMMAND CAUSES EXECUTION OF DIAGNOSTICS IN ONE OF THE
                    0998  1406 ;  FOLLOWING MODES
                    0998  1407 ;       1) BY THE DIAGNOSTIC NAME
```

```
                     0998  1408 ;       2) BY THE MODULE NAME
                     0998  1409 ;       3) ALL NON-OPTIONAL DIAGNOSTICS
                     0998  1410 ;
                     0998  1411 ;*****************************************************************************
                     0998  1412
                     0998  1413 DIAG_COMMAND:    IF      CONTINUE
          3A         0998                .BYTE   ^072
     003A'           0999                .WORD   <CONTINUE-HEAD>
          0F         099B                .BYTE   ^017
          D2         099C                .BYTE   ^0322
     09AF'           099D                .WORD   <2001$-HEAD>
                     099F  1414
                     099F  1415                  CLEAR   CONTINUE                ;THIS COMMAND IS ILLEGAL IN
          AF         099F                .BYTE   ^0250 ! A               ;with A
          32         09A0                .BYTE   ^062
     003A'           09A1                .WORD   <CONTINUE-HEAD>
                     09A3  1416                  SET     RESTART                 ; CONT MODE SO PUT BACK TO
          AF         09A3                .BYTE   ^0250 ! A               ;with A
          3C         09A4                .BYTE   ^04 ! <8*A>
          32         09A5                .BYTE   ^062
     03F1'           09A6                .WORD   <RESTART-HEAD>
                     09A8  1417                                          ; NORMAL MODE AND RESTART PROG
                     09A8  1418                  LXI     H,TTBUF                 ; BACK AT PARSER SO STACK
          21         09A8                .BYTE   <H*8> ! 1
     08'D1'          09A9                .BYTE   <TTBUF-HEAD>&255 , <<TTBUF-HEAD>&^XFF00>/256
                     09AB  1419                  SHLD    TTBUF_PNT               ; WILL REFLECT A NORMAL ENTRY
          22         09AB                .BYTE   ^042
     044D'           09AC                .WORD   <TTBUF_PNT-HEAD>
                     09AE  1420
                     09AE  1421                  RET
          C9         09AE                .BYTE   ^0311
                     09AF  1422
                     09AF  1423                  ENDIF
                     09AF        2001$:                                  ;GENERATE LCL 2001 NAME
                     09AF  1424
                     09AF  1425                  MVI     B,APT_C_DEF             ;GET BIT TO CLEAR FOR NO APT
     DF 06           09AF                .BYTE   <B*8> ! 6 , APT_C_DEF&255
                     09B1  1426                  CALL    CLEAR_LS_STAT           ;CLEAR LS ERROR CONTROL BIT
          CD         09B1                .BYTE   ^0315
     12D8'           09B2                .WORD   <CLEAR_LS_STAT-HEAD>
                     09B4  1427
                     09B4  1428                  LDA     APT_MEM_SIZE            ;GET MEM SIZE THAT APT SENT
          3A         09B4                .BYTE   ^072
     001C'           09B5                .WORD   <APT_MEM_SIZE-HEAD>
                     09B7  1429                  STA     DATA3                   ;PUT IN BYTE 0 STORAGE
          32         09B7                .BYTE   ^062
     00AC'           09B8                .WORD   <DATA3-HEAD>
                     09BA  1430                  LDA     APT_HARD_FLG            ;GET APT HARDWARE CONFIG
          3A         09BA                .BYTE   ^072
     001D'           09BB                .WORD   <APT_HARD_FLG-HEAD>
                     09BD  1431                  STA     DATA2                   ;PUT IN BYTE 1 STORAGE
          32         09BD                .BYTE   ^062
     00AB'           09BE                .WORD   <DATA2-HEAD>
                     09C0  1432                  LDA     APT_SOFT_FLG            ;GET APT SOFTWARE CONFIG
          3A         09C0                .BYTE   ^072
     001E'           09C1                .WORD   <APT_SOFT_FLG-HEAD>
                     09C3  1433                  STA     DATA1                   ;PUT IN BYTE 2 STORAGE
```

```
                    32  09C3                .BYTE   ^062
                 00AA'  09C4                .WORD   <DATA1-HEAD>
                        09C6  1434                  LDA     APT_SOFT_FLG+1          ;GET APT RESERVED BYTE
                    3A  09C6                .BYTE   ^072
                 001F'  09C7                .WORD   <APT_SOFT_FLG+1-HEAD>
                        09C9  1435                  STA     DATA0                   ;PUT IN BYTE 3 STORAGE
                    32  09C9                .BYTE   ^062
                 00A9'  09CA                .WORD   <DATA0-HEAD>
                        09CC  1436
                        09CC  1437                  MVI     A,APT_PARAM             ;LS LOCATION TO WRITE
                 49 3E  09CC                .BYTE   <A*8> ! 6 , APT_PARAM&255
                        09CE  1438                  CALL    WRITE_LS_R              ;WRITE INTO LS
                    CD  09CE                .BYTE   ^0315
                 1146'  09CF                .WORD   <WRITE_LS_R-HEAD>
                        09D1  1439
                        09D1  1440                  SET     EXECUTE                 ;CAUSES EXIT OF PARSER
                    AF  09D1                .BYTE   ^0250 ! A               ;with A
                    3C  09D2                .BYTE   ^04 ! <8*A>
                    32  09D3                .BYTE   ^062
                 02EA'  09D4                .WORD   <EXECUTE-HEAD>
                        09D6  1441
                        09D6  1442                  MVI     A,2
                 02 3E  09D6                .BYTE   <A*8> ! 6 , 2&255
                        09D8  1443                  STA     TABLE_MODE              ;SET TO ALL SECTION MODE(D)
                    32  09D8                .BYTE   ^062
                 040B'  09D9                .WORD   <TABLE_MODE-HEAD>
                        09DB  1444
                        09DB  1445                  LXI     H,SEC_TABLE
                    21  09DB                .BYTE   <H*8> ! 1
              08'5F'    09DC                .BYTE   <SEC_TABLE-HEAD>&255 , <<SEC_TABLE-HEAD>&^XFF00>/256
                        09DE  1446                  SHLD    TABLE_PNT               ;RESET TABLE POINTER
                    22  09DE                .BYTE   ^042
                 040C'  09DF                .WORD   <TABLE_PNT-HEAD>
                        09E1  1447
                        09E1  1448                  CALL    SET_ALL_TESTS           ;SET UP TEST NUM FOR ALL
                    CD  09E1                .BYTE   ^0315
                 1EBA'  09E2                .WORD   <SET_ALL_TESTS-HEAD>
                        09E4  1449                                                  ; TESTS AND CLEAR EOS FLG
                        09E4  1450                  XRA     A
                    AF  09E4                .BYTE   ^0250 ! A               ;with A
                        09E5  1451                  STA     PASS_CNT
                    32  09E5                .BYTE   ^062
                 03EA'  09E6                .WORD   <PASS_CNT-HEAD>
                        09E8  1452                  INR     A                       ;A NOW CONTAINS 1
                    3C  09E8                .BYTE   ^04 ! <8*A>
                        09E9  1453                  STA     PASS_CNT+1              ;SET TO 1 PASS (DEFAULT)
                    32  09E9                .BYTE   ^062
                 03EB'  09EA                .WORD   <PASS_CNT+1-HEAD>
                        09EC  1454
                        09EC  1455                  CLEAR   APT_PASS_CNT            ;INIT MAILBOX PASS CNT
                    AF  09EC                .BYTE   ^0250 ! A               ;with A
                    32  09ED                .BYTE   ^062
                 0018'  09EE                .WORD   <APT_PASS_CNT-HEAD>
                        09F0  1456                  CLEAR   APT_PASS_CNT+1          ;TO 0
                    AF  09F0                .BYTE   ^0250 ! A               ;with A
                    32  09F1                .BYTE   ^062
                 0019'  09F2                .WORD   <APT_PASS_CNT+1-HEAD>
```

```
                09F4  1457                        CLEAR   PARDONE                 ;RESET RE-ENTRY PARSING FLAG
      AF        09F4                .BYTE   ^0250 ! A               ;with A
      32        09F5                .BYTE   ^062
   03D9'        09F6                .WORD   <PARDONE-HEAD>
                09F8  1458                        CLEAR   MICRO_STEP              ;SET TO NO MICRO STEPPING
      AF        09F8                .BYTE   ^0250 ! A               ;with A
      32        09F9                .BYTE   ^062
   038A'        09FA                .WORD   <MICRO_STEP-HEAD>
                09FC  1459
                09FC  1460 1$:                    LXI     H,DIAG_TAB              ;PARSE DIAG COMMAND LINE
      21        09FC                .BYTE   <H*8> ! 1
   09'60'       09FD                .BYTE   <DIAG_TAB-HEAD>&255 , <<DIAG_TAB-HEAD>&^XFF00>/256
                09FF  1461                        CALL    PAR_TEMP_BUF            ;UNTIL ERROR OR <CR>
      CD        09FF                .BYTE   ^0315
   1D6B'        0A00                .WORD   <PAR_TEMP_BUF-HEAD>
                0A02  1462
                0A02  1463                        LDA     PARDONE                 ;GET FLAG
      3A        0A02                .BYTE   ^072
   03D9'        0A03                .WORD   <PARDONE-HEAD>
                0A05  1464                        ORA     A                       ;SET CONDITION CODES
      B7        0A05                .BYTE   ^0260 ! A
                0A06  1465                        RNZ                             ;RETURN IF PARDONE (<CR>)
      C0        0A06                .BYTE   ^0300
                0A07  1466
                0A07  1467                        LDA     NOT_FOUND               ;GET NOT FOUND FLAG
      3A        0A07                .BYTE   ^072
   03C2'        0A08                .WORD   <NOT_FOUND-HEAD>
                0A0A  1468                        ORA     A                       ;SET CONDITION CODES
      B7        0A0A                .BYTE   ^0260 ! A
                0A0B  1469                        RNZ                             ;RET IF NOT FOUND
      C0        0A0B                .BYTE   ^0300
                0A0C  1470
                0A0C  1471                        JMP     1$
      C3        0A0C                .BYTE   ^0303
   09FC'        0A0D                .WORD   <1$-HEAD>
                0A0F  1472
                0A0F  1473 ;*****************************************************************************
                0A0F  1474 ;
                0A0F  1475 ;  DIAG_BOARD
                0A0F  1476 ;  THIS ROUTINE INPUTS A BOARD NAME WHILE PARSING THE DIAGNOSE COMMAND
                0A0F  1477 ;  THIS BOARD NAME IS THEN LOOKED UP ON A TABLE AND A BOARD NUMBER IS
                0A0F  1478 ;  FOUND. TABLE MODE WILL BE SET SO THAT ONLY SECTION NAMES WITH MATCHING
                0A0F  1479 ;  BOARD NUMBERS WILL BE USED.  THE TABLE WILL BE AS FOLLOWS
                0A0F  1480 ;
                0A0F  1481 ;       NUMBER  NAME
                0A0F  1482 ;
                0A0F  1483 ;       1       WCS (M8394)
                0A0F  1484 ;       2       DAP OR CPU (M8390)
                0A0F  1485 ;       4       MCT (M8391)
                0A0F  1486 ;       8       FPA (M8389)
                0A0F  1487 ;       10      IDC (M8388)
                0A0F  1488 ;
                0A0F  1489 ;        FF (TERMINATOR)
                0A0F  1490 ;
                0A0F  1491 ;*****************************************************************************
                0A0F  1492
                0A0F  1493 DIAG_BOARD:    LHLD    TTBUF_PNT
```

```
                    2A  0A0F                .BYTE   ^052
                044D'   0A10                .WORD   <TTBUF_PNT-HEAD>
                        0A12    1494                SHLD    TEMP_WORD               ;SAVE POINTER TO BOARD NAME
                    22  0A12                .BYTE   ^042
                0419'   0A13                .WORD   <TEMP_WORD-HEAD>
                        0A15    1495                CALL    SKIP_TO_SPEC            ;SKIP OVER IT TO GET COUNT AND
                    CD  0A15                .BYTE   ^0315
                0E33'   0A16                .WORD   <SKIP_TO_SPEC-HEAD>
                        0A18    1496                LHLD    TEMP_WORD               ;MOVE BOARD NAME TO SAVE AREA
                    2A  0A18                .BYTE   ^052
                0419'   0A19                .WORD   <TEMP_WORD-HEAD>
                        0A1B    1497                LXI     D,BOARD_NAME
                    11  0A1B                .BYTE   <D*8> ! 1
              02'41'    0A1C                .BYTE   <BOARD_NAME-HEAD>&255 , <<BOARD_NAME-HEAD>&^XFF00>/256
                        0A1E    1498                CALL    MOVER_3_R
                    CD  0A1E                .BYTE   ^0315
                0F5F'   0A1F                .WORD   <MOVER_3_R-HEAD>
                        0A21    1499
                        0A21    1500                LXI     H,BOARD_TABLE
                    21  0A21                .BYTE   <H*8> ! 1
              08'46'    0A22                .BYTE   <BOARD_TABLE-HEAD>&255 , <<BOARD_TABLE-HEAD>&^XFF00>/256
                        0A24    1501                SHLD    BOARD_PNT               ;SET BOARD TABLE POINTER
                    22  0A24                .BYTE   ^042
                024E'   0A25                .WORD   <BOARD_PNT-HEAD>
                        0A27    1502
                        0A27    1503                SET     TABLE_MODE              ;BOARD MODE
                    AF  0A27                .BYTE   ^0250 ! A               ;with A
                    3C  0A28                .BYTE   ^04 ! <8*A>
                    32  0A29                .BYTE   ^062
                040B'   0A2A                .WORD   <TABLE_MODE-HEAD>
                        0A2C    1504
                        0A2C    1505 1$:            LHLD    BOARD_PNT               ;MOVE ENTRY IN BOARD TABLE
                    2A  0A2C                .BYTE   ^052
                024E'   0A2D                .WORD   <BOARD_PNT-HEAD>
                        0A2F    1506                LXI     D,BOARD_BUF             ;TO BUFFER AREA
                    11  0A2F                .BYTE   <D*8> ! 1
              02'47'    0A30                .BYTE   <BOARD_BUF-HEAD>&255 , <<BOARD_BUF-HEAD>&^XFF00>/256
                        0A32    1507                CALL    MOVER_4_R
                    CD  0A32                .BYTE   ^0315
                0F73'   0A33                .WORD   <MOVER_4_R-HEAD>
                        0A35    1508                SHLD    BOARD_PNT               ;UPDATE BOARD TABLE POINTER
                    22  0A35                .BYTE   ^042
                027E'   0A36                .WORD   <BOARD_PNT-HEAD>
                        0A38    1509
                        0A38    1510                IFEQI   BOARD_BUF,HEX_FF        ;END OF TABLE?
                        0A38                .BYTE   ^072
                        0A39                .WORD   <BOARD_BUF-HEAD>
                        0A3B                .BYTE   ^0376
                        0A3C                .BYTE   HEX_FF
                        0A3D                .BYTE   ^0302
                0A45'   0A3E                .WORD   <2002$-HEAD>
                        0A40    1511                MVI     B,<^X01>                ;BOARD NOT FOUND
                        0A40                .BYTE   <B*8> ! 6 , ^X01&255
                        0A42    1512                JMP     ERROR_ROUTINE_NO_RET    ;REPORT ERROR, DON'T RETURN
                        0A42                .BYTE   ^0303
                0A45'   0A43                .WORD   <ERROR_ROUTINE_NO_RET-HEAD>
                        0A45    1513                ENDIF
```

```
                     0A45          2002$:                                  ;GENERATE LCL 2002 NAME
                     0A45    1514
                     0A45    1515                  IFEQ    BOARD_NAME,BOARD_BUF+1,3 ;COMPARE BOARD TO NAME GIVEN
          21         0A45                  .BYTE   <H*8> ! 1
   02'41'            0A46                  .BYTE   <BOARD_NAME-HEAD>&255 , <<BOARD_NAME-HEAD>&^XFF00>/256
          11         0A48                  .BYTE   <D*8> ! 1
   02'48'            0A49                  .BYTE   <BOARD_BUF+1-HEAD>&255 , <<BOARD_BUF+1-HEAD>&^XFF00>/256
   03 0E             0A4B                  .BYTE   <C*8> ! 6 , 3&255
          CD         0A4D                  .BYTE   ^0315
    00CF'            0A4E                  .WORD   <COMPARE-HEAD>
          C2         0A50                  .BYTE   ^0302
    0A5A'            0A51                  .WORD   <2003$-HEAD>
                     0A53    1516
                     0A53    1517                  LDA     BOARD_BUF               ;EQUAL SAVE BOARD #
          3A         0A53                  .BYTE   ^072
    0247'            0A54                  .WORD   <BOARD_BUF-HEAD>
                     0A56    1518                  STA     BOARD_NUM
          32         0A56                  .BYTE   ^062
    024D'            0A57                  .WORD   <BOARD_NUM-HEAD>
                     0A59    1519                  RET                             ;DONE
          C9         0A59                  .BYTE   ^0311
                     0A5A    1520
                     0A5A    1521                  ENDIF
                     0A5A          2003$:                                  ;GENERATE LCL 2003 NAME
                     0A5A    1522
                     0A5A    1523                  JMP     1$                      ;LOOP UNTIL ERROR OR BOARD
          C3         0A5A                  .BYTE   ^0303
    0A2C'            0A5B                  .WORD   <1$-HEAD>
                     0A5D    1524                                                  ; FOUND
                     0A5D    1525
                     0A5D    1526 ;*****************************************************************************
                     0A5D    1527 ;
                     0A5D    1528 ;  DIAG_SECTION
                     0A5D    1529 ;  THIS SUBROUTINE GETS THE SECTION PARAMETER OF A DIAGNOSE COMMAND
                     0A5D    1530 ;  AND LOOKS IT UP IN A TABLE TO SEE WHETHER IT IS A CPU MICRO CODE
                     0A5D    1531 ;  OR AN 8085 DIAGNOSTIC.  IF THE NAME IS NOT FOUND IN THE TABLE IT
                     0A5D    1532 ;  IS ASSUMED TO BE A CPU MICRO CODE DIAGNOSTIC.  THE SECTION IS THEN
                     0A5D    1533 ;  LOADED INTO THE APPROPRIATE AREA.
                     0A5D    1534 ;
                     0A5D    1535 ;*****************************************************************************
                     0A5D    1536
                     0A5D    1537 DIAG_SECTION:    CLEAR   TABLE_MODE              ;SET FOR NO TABLE USAGE
          AF         0A5D                  .BYTE   ^0250 ! A               ;with A
          32         0A5E                  .BYTE   ^062
    040B'            0A5F                  .WORD   <TABLE_MODE-HEAD>
                     0A61    1538
                     0A61    1539                  ZERO    TEMP_SECTION,SEC_LEN    ;START WITH CLEAR FIELD
          21         0A61                  .BYTE   <H*8> ! 1
   04'20'            0A62                  .BYTE   <TEMP_SECTION-HEAD>&255 , <<TEMP_SECTION-HEAD>&^XFF00>/256
   06 0E             0A64                  .BYTE   <C*8> ! 6 , SEC_LEN&255
          AF         0A66                  .BYTE   ^0250 ! A               ;with A
                     0A67          2004$:  MOV     M,A
          77         0A67                  .BYTE   ^0100 ! <M*8> ! A
          23         0A68                  .BYTE   <H*8> ! 3
          0D         0A69                  .BYTE   ^05 ! <8*C>
          C2         0A6A                  .BYTE   ^0302
    0A67'            0A6B                  .WORD   <2004$-HEAD>
```

```
                     0A6D  1540
                     0A6D  1541                         LHLD    TTBUF_PNT
                2A   0A6D                       .BYTE   ^O52
             044D'   0A6E                       .WORD   <TTBUF_PNT-HEAD>
                     0A70  1542                         SHLD    TEMP_WORD               ;SAVE POINTER TO SECTION
                22   0A70                       .BYTE   ^O42
             0419'   0A71                       .WORD   <TEMP_WORD-HEAD>
                     0A73  1543                         CALL    SKIP_TO_SPEC            ;SKIP OVER IT TO GET COUNT
                CD   0A73                       .BYTE   ^O315
             0E33'   0A74                       .WORD   <SKIP_TO_SPEC-HEAD>
                     0A76  1544                         LHLD    TEMP_WORD               ;MOVE NAME TO SAVE AREA
                2A   0A76                       .BYTE   ^O52
             0419'   0A77                       .WORD   <TEMP_WORD-HEAD>
                     0A79  1545                         LXI     D,TEMP_SECTION
                11   0A79                       .BYTE   <D*8> ! 1
           04'20'    0A7A                       .BYTE   <TEMP_SECTION-HEAD>&255 , <<TEMP_SECTION-HEAD>&^XFF00>/256
                     0A7C  1546                         LDA     SKIP_CNT
                3A   0A7C                       .BYTE   ^O72
             0406'   0A7D                       .WORD   <SKIP_CNT-HEAD>
                     0A7F  1547                         MOV     C,A                     ;SECT NAME LENGTH
                4F   0A7F                       .BYTE   ^O100 ! <C*8> ! A
                     0A80  1548                         CALL    MOVER_R
                CD   0A80                       .BYTE   ^O315
             0F75'   0A81                       .WORD   <MOVER_R-HEAD>
                     0A83  1549
                     0A83  1550                         LXI     H,SEC_TABLE
                21   0A83                       .BYTE   <H*8> ! 1
           08'5F'    0A84                       .BYTE   <SEC_TABLE-HEAD>&255 , <<SEC_TABLE-HEAD>&^XFF00>/256
                     0A86  1551                         SHLD    TABLE_PNT               ;RESET TABLE POINTER
                22   0A86                       .BYTE   ^O42
             040C'   0A87                       .WORD   <TABLE_PNT-HEAD>
                     0A89  1552
                     0A89  1553 1$:                     LHLD    TABLE_PNT               ;MOVE ENTRY IN SEC TABLE TO
                2A   0A89                       .BYTE   ^O52
             040C'   0A8A                       .WORD   <TABLE_PNT-HEAD>
                     0A8C  1554                         LXI     D,SECTION_FLAG          ;BUFFER AREA
                11   0A8C                       .BYTE   <D*8> ! 1
           08'C9'    0A8D                       .BYTE   <SECTION_FLAG-HEAD>&255 , <<SECTION_FLAG-HEAD>&^XFF00>/256
                     0A8F  1555                         MVI     C,SEC_LEN+1             ;COUNT
             07 0E   0A8F                       .BYTE   <C*8> ! 6 , SEC_LEN+1&255
                     0A91  1556                         CALL    MOVER_R
                CD   0A91                       .BYTE   ^O315
             0F75'   0A92                       .WORD   <MOVER_R-HEAD>
                     0A94  1557                         SHLD    TABLE_PNT               ;UPDATE SECTION TABLE POINTER
                22   0A94                       .BYTE   ^O42
             040C'   0A95                       .WORD   <TABLE_PNT-HEAD>
                     0A97  1558
                     0A97  1559                         IFEQI   SECTION_FLAG,HEX_FF     ;END OF TABLE
                3A   0A97                       .BYTE   ^O72
             08C9'   0A98                       .WORD   <SECTION_FLAG-HEAD>
                FE   0A9A                       .BYTE   ^O376
                FF   0A9B                       .BYTE   HEX_FF
                C2   0A9C                       .BYTE   ^O302
             0AB5'   0A9D                       .WORD   <2005$-HEAD>
                     0A9F  1560
                     0A9F  1561                         CLEAR   SECTION_FLAG            ;MAKE WCS DEFAULT AREA
                AF   0A9F                       .BYTE   ^O250 ! A               ;with A
```

```
                                                .BYTE   ^062
 08C9' 0AA1                                     .WORD   <SECTION_FLAG-HEAD>
       0AA3  1562                               LXI     H,TEMP_SECTION          ;NAME IS NOT FOUND IN TABLE
    21 0AA3                                     .BYTE   <H*8> ! 1
 04'20' 0AA4                                    .BYTE   <TEMP_SECTION-HEAD>&255 , <<TEMP_SECTION-HEAD>&^XFF00>/256
       0AA6  1563                               LXI     D,SECTION_NAME          ;SO LOAD IT INTO WCS IF IT
    11 0AA6                                     .BYTE   <D*8> ! 1
 08'CA' 0AA7                                    .BYTE   <SECTION_NAME-HEAD>&255 , <<SECTION_NAME-HEAD>&^XFF00>/256
       0AA9  1564                               MVI     C,SEC_LEN               ;IS FOUND ON THE TAPE
 06 0E 0AA9                                     .BYTE   <C*8> ! 6 , SEC_LEN&255
       0AAB  1565                               CALL    MOVER_R
    CD 0AAB                                     .BYTE   ^0315
 0F75' 0AAC                                     .WORD   <MOVER_R-HEAD>
       0AAE  1566
       0AAE  1567                               CALL    LOAD_SECTION
    CD 0AAE                                     .BYTE   ^0315
 1A8B' 0AAF                                     .WORD   <LOAD_SECTION-HEAD>
       0AB1  1568                               RET                             ;EXIT FROM SEARCH LOOP
    C9 0AB1                                     .BYTE   ^031
       0AB2  1569
       0AB2  1570                               ELSE
    C3 0AB2                                     .BYTE   ^0303
 0AC7' 0AB3                                     .WORD   <2006$-HEAD>
       0AB5       2005$:                                                       ;GENERATE LOCAL SYMBOL NAME
       0AB5  1571
       0AB5  1572                               ;EQ     TEMP_SECTION,SECTION_NAME,SEC_LEN
    21 0AB5                                     .BYTE   <H*8> ! 1
 04'20' 0AB6                                    .BYTE   <TEMP_SECTION-HEAD>&255 , <<TEMP_SECTION-HEAD>&^XFF00>/256
    11 0AB8                                     .BYTE   <D*8> ! 1
 08'CA' 0AB9                                    .BYTE   <SECTION_NAME-HEAD>&255 , <<SECTION_NAME-HEAD>&^XFF00>/256
 06 0E  0ABB                                    .BYTE   <C*8> ! 6 , SEC_LEN&255
    CD  0ABD                                    .BYTE   ^0315
  00CF' 0ABE                                    .WORD   <COMPARE-HEAD>
    C2  0AC0                                    .BYTE   ^0302
 0AC7' 0AC1                                     .WORD   <2007$-HEAD>
       0AC3  1573
       0AC3  1574                               CALL    LOAD_SECTION
    CD 0AC3                                     .BYTE   ^0315
 1A8B' 0AC4                                     .WORD   <LOAD_SECTION-HEAD>
       0AC6  1575                               RET                             ;EXIT FROM SEARCH LOOP
    C9 0AC6                                     .BYTE   ^031
       0AC7  1576
       0AC7  1577                               ENDIF
       0AC7       2007$:                                                       ;GENERATE LCL 2007 NAME
       0AC7  1578
       0AC7  1579                               ENDIF
       0AC7       2006$:                                                       ;GENERATE LCL 2006 NAME
       0AC7  1580
       0AC7  1581                               JMP     1$                      ;LOOP UNTIL SECTION FOUND
    C3 0AC7                                     .BYTE   ^0303
 0A89' 0AC8                                     .WORD   <1$-HEAD>
       0ACA  1582
       0ACA  1583 ;*****************************************************************************
       0ACA  1584 ;
       0ACA  1585 ;  DIAG_TEST
       0ACA  1586 ;  THIS SUBROUTINE INPUTS THE TEST PARAMETERS OF THE DIAGNOSE KEYWORD
       0ACA  1587 ;  ONE TEST NUMBER MEANS EXECUTE THAT TEST ONLY
```

```
                     0ACA  1588 ;  TWO TEST NUMBERS MEANS EXECUTE THE TESTS BETWEEN THAT RANGE INCLUSIVE
                     0ACA  1589 ;
                     0ACA  1590 ;*****************************************************************************
                     0ACA  1591
                     0ACA  1592 DIAG_TEST:       IFN     CONSOLE_TEST            ;IF CONSOLE TEST, SKIP THIS
          3A         0ACA                .BYTE   ^072
        0255'        0ACB                .WORD   <CONSOLE_TEST-HEAD>
          0F         0ACD                .BYTE   ^017
          DA         0ACE                .BYTE   ^0332
        0AE1'        0ACF                .WORD   <2008$-HEAD>
                     0AD1  1593
                     0AD1  1594                  IFN     WCS_LOADED              ;ELSE SEE IF WCS LOADED
          3A         0AD1                .BYTE   ^072
        058B'        0AD2                .WORD   <WCS_LOADED-HEAD>
          0F         0AD4                .BYTE   ^017
          DA         0AD5                .BYTE   ^0332
        0AE1'        0AD6                .WORD   <2009$-HEAD>
                     0AD8  1595                                          ; IF NOT, REPORT ERROR
                     0AD8  1596                  CLEAR   EXECUTE         ;CLEAR EXECUTE FLAG TO PROMPT
          AF         0AD8                .BYTE   ^0250 ! A                   ;with A
          32         0AD9                .BYTE   ^062
        02EA'        0ADA                .WORD   <EXECUTE-HEAD>
                     0ADC  1597                  MVI     B,<^X28>            ;CODE FOR NO TEST CODE ERROR
        28 06        0ADE                .BYTE   <B*8> ! 6 , ^X28&255
                     0ADE  1598                  JMP     ERROR_ROUTINE_NO_RET    ;REPORT ERROR
          C3         0ADE                .BYTE   ^0303
        0C45'        0ADF                .WORD   <ERROR_ROUTINE_NO_RET-HEAD>
                     0AE1  1599                  ENDIF
                     0AE1        2009$:                                      ;GENERATE LCL 2009 NAME
                     0AE1  1600
                     0AE1  1601                  ENDIF
                     0AE1        2008$:                                      ;GENERATE LCL 2008 NAME
                     0AE1  1602
                     0AE1  1603                  CLEAR   TABLE_MODE          ;SET TO NO TABLE MODE
          AF         0AE1                .BYTE   ^0250 ! A                   ;with A
          32         0AE2                .BYTE   ^062
        040B'        0AE3                .WORD   <TABLE_MODE-HEAD>
                     0AE5  1604
                     0AE5  1605                  CALL    INPUT_NUM_IF        ;LOOK FOR FIRST TEST NUMBER
          CD         0AE5                .BYTE   ^0315
        0D9B'        0AE6                .WORD   <INPUT_NUM_IF-HEAD>
                     0AE8  1606
                     0AE8  1607                  MVI     B,<^X02>            ;NO FIRST NUMBER - ERROR
        02 06        0AE8                .BYTE   <B*8> ! 6 , ^X02&255
                     0AEA  1608                  LDA     ERROR               ;GET ERROR FLAG
          3A         0AEA                .BYTE   ^072
        02E2'        0AEB                .WORD   <ERROR-HEAD>
                     0AED  1609                  ORA     A                   ;SET CONDITION CODES
          B7         0AED                .BYTE   ^0260 ! A
                     0AEE  1610                  JNZ     ERROR_ROUTINE_NO_RET    ;REPORT IF ERROR.  DON'T RETURN
          C2         0AEE                .BYTE   ^0302
        0C45'        0AEF                .WORD   <ERROR_ROUTINE_NO_RET-HEAD>
                     0AF1  1611
                     0AF1  1612                  LDA     INPUT_NUM+3         ;FIRST NUMBER IS FIRST TEST
          3A         0AF1                .BYTE   ^072
        0679'        0AF2                .WORD   <INPUT_NUM+3-HEAD>
                     0AF4  1613                  STA     TEST_NUM1           ;SAVE IT
```

```
                     32  0AF4                .BYTE   ^062
                   0427' 0AF5                .WORD   <TEST_NUM1-HEAD>
                         0AF7  1614
                         0AF7  1615                    CAL:    INPUT_NUM_IF            ;GET SECOND NUMBER IF THERE
                     CD  0AF7                .BYTE   ^0315
                   0D9B' 0AF8                .WORD   <INPUT_NUM_IF-HEAD>
                         0AFA  1616
                         0AFA  1617                    IF      ERROR
                     3A  0AFA                .BYTE   ^072
                   02E2' 0AFB                .WORD   <ERROR-HEAD>
                     0F  0AFD                .BYTE   ^017
                     D2  0AFE                .BYTE   ^0322
                   0B07' 0AFF                .WORD   <2010$-HEAD>
                         0B01  1618                    LDA     TEST_NUM1               ;NO SECOND NUM - USE FIRST TEST
                     3A  0B01                .BYTE   ^072
                   0427' 0B02                .WORD   <TEST_NUM1-HEAD>
                         0B04  1619
                         0B04  1620                    ELSE
                     C3  0B04                .BYTE   ^0303
                   0B0A' 0B05                .WORD   <2011$-HEAD>
                         0B07        2010$:                                          ;GENERATE LOCAL SYMBOL NAME
                         0B07  1621
                         0B07  1622                    LDA     INPUT_NUM+3             ;SAVE ENDING TEST NUMBER
                     3A  0B07                .BYTE   ^072
                   0679' 0B08                .WORD   <INPUT_NUM+3-HEAD>
                         0B0A  1623                    ENDIF
                         0B0A        2011$:                                          ;GENERATE LCL 2011 NAME
                         0B0A  1624
                         0B0A  1625                    STA     TEST_NUM2
                     32  0B0A                .BYTE   ^062
                   0428' 0B0B                .WORD   <TEST_NUM2-HEAD>
                         0B0D  1626
                         0B0D  1627                    RET
                     C9  0B0D                .BYTE   ^0311
                         0B0E  1628
                         0B0E  1629
                         0B0E  1630 ;*****************************************************************************
                         0B0E  1631 ;
                         0B0E  1632 ;  DIAG_CONTIN
                         0B0E  1633 ;  THIS SUBROUTINE HANDLES THE DIAGNOSE CONTINUE KEYWORD
                         0B0E  1634 ;  A CONTINUE WILL CAUSE THE TESTING TO GO TO THE END OF THE SECTION
                         0B0E  1635 ;
                         0B0E  1636 ;  NOTE: - - TO BE EFFECTIVE THIS KEYWORD MUST BE GIVEN AFTER THE TEST
                         0B0E  1637 ;            KEYWORD AND THE TEST KEYWORD MUST HAVE ONLY 1 PARAMETER
                         0B0E  1638 ;            IF THE TEST PARAMETER IS GIVEN AFTER THE CONTINUE KEYWORD
                         0B0E  1639 ;            IT WILL BE CANCELED OUT.
                         0B0E  1640 ;
                         0B0E  1641 ;*****************************************************************************
                         0B0E  1642
                         0B0E  1643 DIAG_CONTIN:   MVI     A,HEX_50                ;MOVE LAST POSSIBLE TEST
                   50 3E 0B0E                .BYTE   <A'S> ! 6 , HEX_50&255
                         0B10  1644                    STA     TEST_NUM2               ;TO ENDING TEST NUMBER
                     32  0B10                .BYTE   ^062
                   0428' 0B11                .WORD   <TEST_NUM2-HEAD>
                         0B13  1645
                         0B13  1646                    RET
                     C9  0B13                .BYTE   ^0311
```

```
                     0B14  1647
                     0B14  1648
                     0B14  1649 ;*****************************************************************************
                     0B14  1650 ;
                     0B14  1651 ;  DIAG_PASS
                     0B14  1652 ;  THIS SUBROUTINE INPUTS THE DIAGNOSE PASS COUNT.
                     0B14  1653 ;  IF NOT SPECIFIED THE PASS COUNT WILL DEFAULT TO ONE.
                     0B14  1654 ;  IF THE PASS COUNT IS SPECIFED AS -1 THIS INDICATES A CONTINUOUS LOOP
                     0B14  1655 ;
                     0B14  1656 ;*****************************************************************************
                     0B14  1657
                     0B14  1658 DIAG_PASS:       CALL    INPUT_NUM_IF            ;GET THE PASS COUNT
          CD         0B14                .BYTE   ^O31
    0D9B'            0B15                .WORD   <INPUT_NUM_IF-HEAD>
                     0B17  1659
                     0B17  1660                  MVI     B,<^X03>                ;PASS COUNT ERROR
  C3 06              0B17                .BYTE   <B*8> ! 6 , ^X03&255
                     0B19  1661                  LDA     ERROR                   ;GET ERROR FLAG
          3A         0B19                .BYTE   ^O72
    02E2'            0B1A                .WORD   <ERROR-HEAD>
                     0B1C  1662                  ORA     A                       ;SET CONDITION CODES
          B7         0B1C                .BYTE   ^O260 ! A
                     0B1D  1663                  JNZ     ERROR_ROUTINE_NO_RET    ;REPORT IF ERROR.  DON'T RETURN
          C2         0B1D                .BYTE   ^O302
    0C45'            0B1E                .WORD   <ERROR_ROUTINE_NO_RET-HEAD>
                     0B20  1664
                     0B20  1665                  IF      MINUS
          3A         0B20                .BYTE   ^O72
    0383'            0B21                .WORD   <MINUS-HEAD>
          0F         0B23                .BYTE   ^O17
          D2         0B24                .BYTE   ^O322
    0B32'            0B25                .WORD   <2012$-HEAD>
                     0B27  1666
                     0B27  1667                  MVI     A,HEX_FF
  FF 3E              0B27                .BYTE   <A*8> ! 6 , HEX_FF&255
                     0B29  1668                  STA     PASS_CNT                ;SET TO INFINITE PASSES
          32         0B29                .BYTE   ^O62
    03EA'            0B2A                .WORD   <PASS_CNT-HEAD>
                     0B2C  1669                  STA     PASS_CNT+1
          32         0B2C                .BYTE   ^O62
    03EB'            0B2D                .WORD   <PASS_CNT+1-HEAD>
                     0B2F  1670
                     0B2F  1671                  ELSE
          C3         0B2F                .BYTE   ^O303
    0B38'            0B30                .WORD   <2013$-HEAD>
                     0B32          2012$:                                    ;GENERATE LOCAL SYMBOL NAME
                     0B32  1672
                     0B32  1673                  LHLD    INPUT_NUM+2
          2A         0B32                .BYTE   ^O52
    0678'            0B33                .WORD   <INPUT_NUM+2-HEAD>
                     0B35  1674                  SHLD    PASS_CNT                ;MOVE 2 BYTES PASS COUNT
          22         0B35                .BYTE   ^O42
    03EA'            0B36                .WORD   <PASS_CNT-HEAD>
                     0B38  1675
                     0B38  1676                  ENDIF
                     0B38          2013$:                                    ;GENERATE LCL 2013 NAME
                     0B38  1677
```

```
                0B38  1678                    RET
      C9        0B38                .BYTE     ^0311
                0B39  1679
                0B39  1680
                0B39  1681 ;*****************************************************************************
                0B39  1682 ;
                0B39  1683 ;  LOADER
                0B39  1684 ;  THIS ROUTINE LOADS A IMAGE INTO WCS OR THE CONSOLE BUT DOES
                0B39  1685 ;  NOT START IT AND DOES NOT INITIALIZE IT
                0B39  1686 ;
                0B39  1687 ;*****************************************************************************
                0B39  1688
                0B39  1689 LOADER:          SET       NOINIT
      AF        0B39                .BYTE     ^0250 ! A                   ;with A
      3C        0B3A                .BYTE     ^04 ! <8*A>
      32        0B3B                .BYTE     ^062
    03BE'       0B3C                .WORD     <NOINIT-HEAD>
                0B3E  1690                    CALL      DIAG_SECTION
      CD        0B3E                .BYTE     ^0315
    0A5D'       0B3F                .WORD     <DIAG_SECTION-HEAD>
                0B41  1691                    CLEAR     NOINIT
      AF        0B41                .BYTE     ^0250 ! A                   ;with A
      32        0B42                .BYTE     ^062
    03BE'       0B43                .WORD     <NOINIT-HEAD>
                0B45  1692                    RET
      C9        0B45                .BYTE     ^0311
                0B46  1693
                0B46  1694
                0B46  1695 ;*****************************************************************************
                0B46  1696 ;
                0B46  1697 ;  CONT_COMMAND
                0B46  1698 ;  THE CONTINUE COMMAND ALLOWS CONTINUATION OF TEST EXECUTION AFTER
                0B46  1699 ;  TESTING HAS BEEN HALTED
                0B46  1700 ;
                0B46  1701 ;*****************************************************************************
                0B46  1702
                0B46  1703 CONT_COMMAND:    MVI       B,<^X04>                  ;ERROR CODE IF CONTINUE CLEAR
   04 06        0B46                .BYTE     <B*8> ! 6 , ^X04&255
                0B48  1704
                0B48  1705                    LDA       CONTINUE                  ;GET CONTINUE FLAG
      3A        0B48                .BYTE     ^072
    003A'       0B49                .WORD     <CONTINUE-HEAD>
                0B4B  1706                    ORA       A                         ;SET CONDITION CODES
      B7        0B4B                .BYTE     ^0260 ! A
                0B4C  1707                    CZ        ERROR_ROUTINE             ;NO CONTINUE AT THIS POINT IF
      CA        0B4C                .BYTE     ^0312
    0C4A'       0B4D                .WORD     <ERROR_ROUTINE-HEAD>
                0B4F  1708                                                        ; CONTINUE FLAG IS CLEAR
                0B4F  1709                                                        ; ELSE CONTINUE ON
                0B4F  1710                    JFN       CONSOLE_TEST              ;DO WR RESTORE IF 8085
      3A        0B4F                .BYTE     ^072
    0255'       0B50                .WORD     <CONSOLE_TEST-HEAD>
      0F        0B52                .BYTE     ^017
      DA        0B53                .BYTE     ^0332
    0B59'       0B54                .WORD     <.+0145-HEAD>
                0B56  1711                    CALL      RESTORE_WR                ; TESTS ARE NOT EXECUTING
      CD        0B56                .BYTE     ^0315
```

```
                0C0A' 0B57                    .WORD   <RESTORE_WR-HEAD>
                      0B59  1712              ENDIF
                      0B59  1713      2014$:                                  ;GENERATE LCL 2014 NAME
                      0B59  1714                      OUT     SETLIT          ;TURN RUN LITE BACK ON
              3D D3   0B59                    .BYTE   ^O323 , SETLIT&255      ;register A
                      0B5B  1715
                      0B5B  1716                      LDA     C_EXECUTE       ;RESTORE FLAGS
                 3A   0B5B                    .BYTE   ^O72
                0250' 0B5C                    .WORD   <C_EXECUTE-HEAD>
                      0B5E  1717                      STA     EXECUTE
                 32   0B5E                    .BYTE   ^O62
                02EA' 0B5F                    .WORD   <EXECUTE-HEAD>
                      0B61  1718
                      0B61  1719                      LXI     H,CONT_SAVE_CSR ;RESTORE CSR
                 21   0B61                    .BYTE   <H*8> ! 1
               00'BE' 0B62                    .BYTE   <CONT_SAVE_CSR-HEAD>&255 , <<CONT_SAVE_CSR-HEAD>&^XFF00>/256
                      0B64  1720                      LXI     D,SAVED_CSR
                 11   0B64                    .BYTE   <D*8> ! 1
               00'BB' 0B65                    .BYTE   <SAVED_CSR-HEAD>&255 , <<SAVED_CSR-HEAD>&^XFF00>/256
                      0B67  1721                      CALL    MOVER_3_R
                 CD   0B67                    .BYTE   ^O315
                0F5F' 0B68                    .WORD   <MOVER_3_R-HEAD>
                      0B6A  1722
                      0B6A  1723                      LHLD    CONT_SAVE_UPC   ;RESTORE UPC
                 2A   0B6A                    .BYTE   ^O52
                00C8' 0B6B                    .WORD   <CONT_SAVE_UPC-HEAD>
                      0B6D  1724                      SHLD    SAVED_UPC
                 22   0B6D                    .BYTE   ^O42
                00C4' 0B6E                    .WORD   <SAVED_UPC-HEAD>
                      0B70  1725
                      0B70  1726                      LDA     C_RUNNING       ;GET OLD RUNNING FLAG
                 3A   0B70                    .BYTE   ^O72
                0251' 0B71                    .WORD   <C_RUNNING-HEAD>
                      0B73  1727                      ORA     A               ;SET CONDITION CODES
                 B7   0B73                    .BYTE   ^O260 ! A
                      0B74  1728                      CNZ     RESTART_CPU     ;RESTART IF SET
                 C4   0B74                    .BYTE   ^O304
                110E' 0B75                    .WORD   <RESTART_CPU-HEAD>
                      0B77  1729
                      0B77  1730                      SET     CONT_DONE
                 AF   0B77                    .BYTE   ^O250 ! A               ;with A
                 3C   0B78                    .BYTE   ^O4 ! <B*A>
                 32   0B79                    .BYTE   ^O62
                0256' 0B7A                    .WORD   <CONT_DONE-HEAD>
                      0B7C  1731
                      0B7C  1732                      RET
                 C9   0B7C                    .BYTE   ^O311
                      0B7D  1733
                      0B7D
                      0B7D    ;;  ;*****************************************************************************
                      0B7D  1736  ;
                      0B7D  1737  ;  EXEC_SUB
                      0B7D  1738  ;
                      0B7D  1739  ;  THIS ROUTINE EXECUTES A SUBROUTINE IN WCS WHICH PERFORMS A SPECIFIC
                      0B7D  1740  ;  FUNCTION.
                      0B7D  1741  ;
```

```
                     0B7D  1742 ;  INPUT CONDITIONS
                     0B7D  1743 ;        H,L  = ADDRESS OF WCS SUBROUTINE
                     0B7D  1744 ;
                     0B7D  1745 ;*****************************************************************************
                     0B7D  1746
                     0B7D  1747 EXEC_SUB:        MVI     B,<^X11>                ;ERROR CODE IF WCS NOT LOADED
          11 06      0B7D                       .BYTE   <B*8> ! 6 , ^X11&255
                     0B7F  1748                 CALL    CHECK_WCS_LOADED        ;CHECK FOR WCS LOADED SET
            CD       0B7F                       .BYTE   ^0315
         0C2A'       0B80                       .WORD   <CHECK_WCS_LOADED-HEAD>
                     0B82  1749                                                 ; RETURN IF OK
                     0B82  1750
                     0B82  1751                 MOV     A,H                     ;GET SUBROUTINE ADDR
            7C       0B82                       .BYTE   ^0100 ! <A*8> ! H
                     0B83  1752                 MOV     H,L
            65       0B83                       .BYTE   ^0100 ! <H*8> ! L
                     0B84  1753                 MOV     L,A
            6F       0B84                       .BYTE   ^0100 ! <L*8> ! A
                     0B85  1754                 SHLD    STARTING_UPC
            22       0B85                       .BYTE   ^042
         0407'       0B86                       .WORD   <STARTING_UPC-HEAD>
                     0B88  1755
                     0B88  1756                 CALL    START_CPU               ;START THE SUBROUTINE
            CD       0B88                       .BYTE   ^0315
         10FF'       0B89                       .WORD   <START_CPU-HEAD>
                     0B8B  1757                 CALL    SUB_ATTENTION           ;WAIT FOR EXIT
            CD       0B8B                       .BYTE   ^0315
         1806'       0B8C                       .WORD   <SUB_ATTENTION-HEAD>
                     0B8E  1758
                     0B8E  1759                 RET
            C9       0B8E                       .BYTE   ^0311
                     0B8F  1760
                     0B8F  1761 ;*****************************************************************************
                     0B8F  1762 ;
                     0B8F  1763 ;  CLEAR_DEFAULT
                     0B8F  1764 ;  THIS ROUTINE SETS THE FLAGS TO THEIR DEFAULT VALUES
                     0B8F  1765 ;
                     0B8F  1766 ;*****************************************************************************
                     0B8F  1767
                     0B8F  1768 CLEAR_DEFAULT:   MVI     B,BELL_C_DEF            ;CLEAR BELL
          FB 06      0B8F                       .BYTE   <B*8> ! 6 , BELL_C_DEF&255
                     0B91  1769                 CALL    CLEAR_LS_STAT
            CD       0B91                       .BYTE   ^0315
         12D8'       0B92                       .WORD   <CLEAR_LS_STAT-HEAD>
                     0B94  1770
                     0B94  1771                 MVI     B,NER_S_DEF             ;SET NER
          02 06      0B94                       .BYTE   <B*8> ! 6 , NER_S_DEF&255
                     0B96  1772                 CALL    SET_LS_STAT
            CD       0B96                       .BYTE   ^0315
         12CE'       0B97                       .WORD   <SET_LS_STAT-HEAD>
                     0B99  1773
                     0B99  1774                 MVI     B,LOOP_COM_C_DEF        ;BIT TO CLEAR LOOP COMMAND
          BF 06      0B99                       .BYTE   <B*8> ! 6 , LOOP_COM_C_DEF&255
                     0B9B  1775                 CALL    CLEAR_LS_STAT
            CD       0B9B                       .BYTE   ^0315
         12D8'       0B9C                       .WORD   <CLEAR_LS_STAT-HEAD>
                     0B9E  1776                 MVI     B,HALT_S_DEF            ;SET HALT
```

```
08 06  0B9E                    .BYTE   <B*8> ! 6 , HALT_S_DEF&255
       0BA0  1777              CALL    SET_LS_STAT
  CD   0BA0                    .BYTE   ^0315
 12CE' 0BA1                    .WORD   <SET_LS_STAT-HEAD>
       0BA3  1778
       0BA3  1779              CLEAR   ERROR_CON                       ;CLEAR LOOP
  AF   0BA3                    .BYTE   ^0250 ! A               ;with A
  32   0BA4                    .BYTE   ^062
 0057' 0BA5                    .WORD   <ERROR_CON-HEAD>
       0BA7  1780              MVI     B,LOOP_C_DEF
BE 06  0BA7                    .BYTE   <B*8> ! 6 , LOOP_C_DEF&255
       0BA9  1781              CALL    CLEAR_LS_STAT
  CD   0BA9                    .BYTE   ^0315
 12D8' 0BAA                    .WORD   <CLEAR_LS_STAT-HEAD>
       0BAC  1782
       0BAC  1783              CLEAR   TRACE                           ;CLEAR TRACE
  AF   0BAC                    .BYTE   ^0250 ! A               ;with A
  32   0BAD                    .BYTE   ^062
 044C' 0BAE                    .WORD   <TRACE-HEAD>
       0BB0  1784
       0BB0  1785              MVI     B,SER_C_DEF                     ;CLEAR SER
EF 06  0BB0                    .BYTE   <B*8> ! 6 , SER_C_DEF&255
       0BB2  1786              CALL    CLEAR_LS_STAT
  CD   0BB2                    .BYTE   ^0315
 12D8' 0BB3                    .WORD   <CLEAR_LS_STAT-HEAD>
       0BB5  1787
       0BB5  1788              MVI     B,SPECIAL_C_DEF                 ;CLEAR SP
7F 06  0BB5                    .BYTE   <B*8> ! 6 , SPECIAL_C_DEF&255
       0BB7  1789              CALL    CLEAR_LS_STAT
  CD   0BB7                    .BYTE   ^0315
 12D8' 0BB8                    .WORD   <CLEAR_LS_STAT-HEAD>
       0BBA  1790
       0BBA  1791              RET
  C9   0BBA                    .BYTE   ^0311
       0BBB  1792
       0BBB  1793 ;*****************************************************************************
       0BBB  1794 ;
       0BBB  1795 ;  INT_CLEAR_PAR
       0BBB  1796 ;  THIS SUBROUTINE IS CALLED FROM BOTH THE COMMAND CLEAR_PARITY AND FROM
       0BBB  1797 ;  THE INTERNAL CLEAR PARITY REQUEST
       0BBB  1798 ;
       0BBB  1799 ;  INPUT CONDITION:
       0BBB  1800 ;       ERROR IS SET IF NO ADDRESS IS FOUND
       0BBB  1801 ;
       0BBB  1802 ;*****************************************************************************
       0BBB  1803
       0BBB  1804 INT_CLEAR_PAR:  LDA     ERROR                        ;GET ERROR FLAG
  3A   0BBB                    .BYTE   ^072
02E2' 0BBC                    .WORD   <ERROR-HEAD>
       0BBE  1805              ORA     A                            ;SET CONDITION CODES
  B7   0BBE                    .BYTE   ^0260 ! A
       0BBF  1806              CZ      CHANGE_WCS                   ;CHANGE PARITY TO GOOD IF
  CC   0BBF                    .BYTE   ^0314
0C1D' 0BC0                    .WORD   <CHANGE_WCS-HEAD>
       0BC2  1807                                                   ; ADDRESS GIVEN
       0BC2  1808
       0BC2  1809              SET     PAR_ON                       ;SET FOR HALT ON PAR ERR
```

```
                AF  0BC2                .BYTE   ^0250 ! A               ;with A
                3C  0BC3                .BYTE   ^04 ! <8*A>
                32  0BC4                .BYTE   ^062
            03D6'   0BC5                .WORD   <PAR_ON-HEAD>
                    0BC7  1810
                    0BC7  1811                  RET
                C9  0BC7                .BYTE   ^0311
                    0BC8  1812
                    0BC8  1813
```

```
                    0BC8  1815 ;*****************************************************************************
                    0BC8  1816 ;
                    0BC8  1817 ;  MISC SUBROUTINES
                    0BC8  1818 ;
                    0BC8  1819 ;*****************************************************************************
                    0BC8  1820 ;*****************************************************************************
                    0BC8  1821 ;
                    0BC8  1822 ;  PARITY_CALC
                    0BC8  1823 ;  THIS ROUTINE CALCULATES PARITY FOR THE WCS WORDS
                    0BC8  1824 ;  AND PLACES IT IN BIT 23 OF WCS_WORD IF THE PARITY SELECTOR IS SET
                    0BC8  1825 ;  TO BAD PARITY AS USED IN STOP ON MICRO MATCH THE PARITY WILL BE
                    0BC8  1826 ;  CHANGED TO BAD PARITY IN BIT 23 OF CSR_VALUE
                    0BC8  1827 ;
                    0BC8  1828 ;  INPUT CONDITIONS:
                    0BC8  1829 ;       H+L              ADDRESS OF 24 BIT WORD TO CALC PARITY ON
                    0BC8  1830 ;       PARITY_ON        PARITY CALCULATING ENABLE FLAG
                    0BC8  1831 ;       WCS_BAD_PARITY   INDICATES THAT THIS WCS WORD IS TO HAVE BAD PARITY
                    0BC8  1832 ;
                    0BC8  1833 ;  OUTPUT CONDITION:
                    0BC8  1834 ;       WCS_BAD_PARITY   IS CLEARED IF BAD PARITY AND WCS CALC WERE SET
                    0BC8  1835 ;       PARITY_ON        SET FOR PARITY CALCULATION
                    0BC8  1836 ;
                    0BC8  1837 ;*****************************************************************************
                    0BC8  1838
                    0BC8  1839 PARITY_CALC:    SHLD    CALC_ADD                ;SAVE INPUT POINTER
                 22 0BC8                       .BYTE   ^042
               0252' 0BC9                      .WORD   <CALC_ADD-HEAD>
                    0BCB  1840
                    0BCB  1841                 IF      PARITY_ON               ;IF PARITY CALC ENABLED
                 3A 0BCB                       .BYTE   ^072
               0064' 0BCC                      .WORD   <PARITY_ON-HEAD>
                 0F 0BCE                       .BYTE   ^017
                 D2 0BCF                       .BYTE   ^0322
               0C04' 0BD0                      .WORD   <2015$-HEAD>
                    0BD2  1842
                    0BD2  1843                 LHLD    CALC_ADD
                 2A 0BD2                       .BYTE   ^052
               0252' 0BD3                      .WORD   <CALC_ADD-HEAD>
                    0BD5  1844                 MOV     A,M                     ;GET FIRST BYTE
                 7E 0BD5                       .BYTE   ^0100 ! <A*8> ! M
                    0BD6  1845                 ANI     HEX_7F                  ;AND OUT CURRENT PARITY BIT
                 E6 0BD6                       .BYTE   ^0346
                 7F 0BD7                       .BYTE   HEX_7F
                    0BD8  1846                 INX     H
                 23 0BD8                       .BYTE   <H*8> ! 3
                    0BD9  1847                 XRA     M                       ;XOR WITH SECOND AND THIRD
                 AE 0BD9                       .BYTE   ^0250 ! M               ;with A
                    0BDA  1848                 INX     H                       ;BYTES OF 24 BIT WORD
                 23 0BDA                       .BYTE   <H*8> ! 3
                    0BDB  1849                 XRA     M
                 AE 0BDB                       .BYTE   ^0250 ! M               ;with A
                    0BDC  1850
                    0BDC  1851                 DCX     H                       ;MOVE POINTER BACK AND
                 2B 0BDC                       .BYTE   <H*8> ! 11
                    0BDD  1852                 DCX     H                       ;GET THE HIGH BYTE
                 2B 0BDD                       .BYTE   <H*8> ! 11
                    0BDE  1853                 MOV     A,M
```

```
                7E  0BDE                .BYTE   ^0100 ! <A*8> ! M
                    0BDF  1854
                    0BDF  1855          IFPO                            ;IF PARITY ODD
                EA  0BDF                .BYTE   ^0352
            0BE7'   0BE0                .WORD   <2016$-HEAD>
                    0BE2  1856
                    0BE2  1857          ANI     HEX_7F                  ;AND OUT PARITY BIT
                E6  0BE2                .BYTE   ^0346
                7F  0BE3                .BYTE   HEX_7F
                    0BE4  1858
                    0BE4  1859          ELSE
                C3  0BE4                .BYTE   ^0303
            0BE9'   0BE5                .WORD   <2017$-HEAD>
                    0BE7        2016$:                                  ;GENERATE LOCAL SYMBOL NAME
                    0BE7  1860
                    0BE7  1861          ORI     HEX_80                  ;OR IN PARITY BIT
                F6  0BE7                .BYTE   ^0366
                80  0BE8                .BYTE   HEX_80
                    0BE9  1862
                    0BE9  1863          ENDIF
                    0BE9        2017$:                                  ;GENERATE LCL 2017 NAME
                    0BE9  1864
                    0BE9  1865          MOV     M,A                     ;PUT DATA BACK IN CSR_VALUE
                77  0BE9                .BYTE   ^0100 ! <M*8> ! A
                    0BEA  1866                                          ; OR WCS_VALUE
                    0BEA  1867          IF      WCS_CALC                ;THIS ROUTINE WHEN WCS PARITY
                3A  0BEA                .BYTE   ^072
            058A'   0BEB                .WORD   <WCS_CALC-HEAD>
                0F  0BED                .BYTE   ^017
                D2  0BEE                .BYTE   ^0322
            0C04'   0BEF                .WORD   <2018$-HEAD>
                    0BF1  1868                                          ; IS CALCULATED
                    0BF1  1869          IF      WCS_BAD_PARITY
                3A  0BF1                .BYTE   ^072
            0068'   0BF2                .WORD   <WCS_BAD_PARITY-HEAD>
                0F  0BF4                .BYTE   ^017
                D2  0BF5                .BYTE   ^0322
            0C04'   0BF6                .WORD   <2019$-HEAD>
                    0BF8  1870          LHLD    CALC_ADD
                2A  0BF8                .BYTE   ^052
            0252'   0BF9                .WORD   <CALC_ADD-HEAD>
                    0BFB  1871          MOV     A,M
                7F  0BFB                .BYTE   ^0100 ! <A*8> ! M
                    0BFC  1872          RAL
                17  0BFC                .BYTE   ^027
                    0BFD  1873          CMC                             ;FLIP PARITY BIT IN CARRY
                3F  0BFD                .BYTE   ^077
                    0BFE  1874          RAR                             ;TO MAKE BAD PARITY
                1F  0BFE                .BYTE   ^037
                    0BFF  1875          MOV     M,A
                77  0BFF                .BYTE   ^0100 ! <M*8> ! A
                    0C00  1876          CLEAR   WCS_BAD_PARITY          ;CLEAR BAD PARITY FLAG
                AF  0C00                .BYTE   ^0250 ! A               ;with A
                3E  0C01                .BYTE   ^062
            006E'   0C02                .WORD   <WCS_BAD_PARITY-HEAD>
                    0C04  1877          ENDIF
                    0C04        2019$:                                  ;GENERATE LCL 2019 NAME
```

```
                 0C04  1878                     ENDIF
                 0C04        2018$:                                          ;GENERATE LCL 2018 NAME
                 0C04  1879
                 0C04  1880                     ENDIF
                 0C04        2015$:                                          ;GENERATE LCL 2015 NAME
                 0C04  1881
                 0C04  1882                     SET     PARITY_ON               ;TURN PARITY CALC BACK ON
      AF         0C04                .BYTE    ^0250 ! A                ;with A
      3C         0C05                .BYTE    ^04 ! <8*A>
      32         0C06                .BYTE    ^062
    0064'        0C07                .WORD    <PARITY_ON-HEAD>
                 0C09  1883
                 0C09  1884                     RET
      C9         0C09                .BYTE    ^0311
                 0C0A  1885
                 0C0A  1886
                 0C0A  1887 ;*****************************************************************************
                 0C0A  1888 ;
                 0C0A  1889 ;  RESTORE_WR
                 0C0A  1890 ;  THIS ROUTINE CALLS A WCS SUBROUTINE THAT RESTORES THE WORKING REGS
                 0C0A  1891 ;  OF THE 2901 FROM A SAVED AREA
                 0C0A  1892 ;
                 0C0A  1893 ;*****************************************************************************
                 0C0A  1894
                 0C0A  1895 RESTORE_WR:          LXI     H,WCS_REST_WR           ;ADDRESS OF SUBROUTINE
      21         0C0A                .BYTE    <H*8> ! 1
   00 57         0C0B                .BYTE    <WCS_REST_WR-HEAD>&255 , <<WCS_REST_WR-HEAD>&^XFF00>/256
                 0C0D  1896                     CALL    EXEC_SUB                ;DO THE SUBROUTINE
      CD         0C0D                .BYTE    ^0315
    0B7D'        0C0E                .WORD    <EXEC_SUB-HEAD>
                 0C10  1897
                 0C10  1898                     RET
      C9         0C10                .BYTE    ^0311
                 0C11  1899
                 0C11  1900
                 0C11  1901 ;*****************************************************************************
                 0C11  1902 ;
                 0C11  1903 ;  SAVE_WR
                 0C11  1904 ;  THIS ROUTINE CALLS A WCS SUBROUTINE THAT SAVES THE CONTENTS OF THE
                 0C11  1905 ;  2901 IN A SPECIAL SAVE AREA SO THEY MAY BE RESTORED LATER.
                 0C11  1906 ;
                 0C11  1907 ;*****************************************************************************
                 0C11  1908
                 0C11  1909 SAVE_WR:             SET     NO_SAVE_UPC_CSR         ;SET FLAG SO SAVED CSR AND
      AF         0C11                .BYTE    ^0250 ! A                ;with A
      3C         0C12                .BYTE    ^04 ! <8*A>
      32         0C13                .BYTE    ^062
    03C0'        0C14                .WORD    <NO_SAVE_UPC_CSR-HEAD>
                 0C16  1910                                                      ; UPC ARE NOT DESTROYED
                 0C16  1911                     LXI     H,WCS_SAVE_WR           ;ADDRESS OF SUBROUTINE
      21         0C16                .BYTE    <H*8> ! 1
   00 56         0C17                .BYTE    <WCS_SAVE_WR-HEAD>&255 , <<WCS_SAVE_WR-HEAD>&^XFF00>/256
                 0C19  1912                     CALL    EXEC_SUB                ;DO THE SUBROUTINE
      CD         0C19                .BYTE    ^0315
    0B7D'        0C1A                .WORD    <EXEC_SUB-HEAD>
                 0C1C  1913
                 0C1C  1914                     RET
```

```
                C9  0C1C                .BYTE   ^0311
                    0C1D  1915
                    0C1D  1916
                    0C1D  1917 ;*****************************************************************************
                    0C1D  1918 ;
                    0C1D  1919 ;  CHANGE_WCS
                    0C1D  1920 ;  THIS ROUTINE READS A WCS WORD, THEN CALCULATES PARITY ON IT AND
                    0C1D  1921 ;  WRITES IT BACK.  THIS IS USED FOR SETTING AND CLEARING SOMM
                    0C1D  1922 ;  ADDRESSES AND SETING BAD PARITY
                    0C1D  1923 ;
                    0C1D  1924 ;  INPUT CONDITIONS:
                    0C1D  1925 ;       WCS_BAD_PARITY MUST BE SET OR CLEARED
                    0C1D  1926 ;       BECAUSE IT IS USED IN A CALLED ROUTINE
                    0C1D  1927 ;
                    0C1D  1928 ;       INPUT_NUM MUST HAVE THE ADDRESS TO BE CHANGED
                    0C1D  1929 ;
                    0C1D  1930 ;*****************************************************************************
                    0C1D  1931
                    0C1D  1932 CHANGE_WCS:      LHLD    INPUT_NUM+2
                  2A  0C1D                .BYTE   ^052
              0678' 0C1E                .WORD   <INPUT_NUM+2-HEAD>
                    0C20  1933                  SHLD    WCS_ADDRESS             ;PUT ADDRESS IN UPC TO READ
                  22  0C20                .BYTE   ^042
              006C' 0C21                .WORD   <WCS_ADDRESS-HEAD>
                    0C23  1934                  CALL    READ_WCS_R              ;FROM WCS
                  CD  0C23                .BYTE   ^0315
              0E6D' 0C24                .WORD   <READ_WCS_R-HEAD>
                    0C26  1935
                    0C26  1936                  CALL    WRITE_WCS_R             ;WRITE WORD BACK WITH
                  CD  0C26                .BYTE   ^0315
              0E8A' 0C27                .WORD   <WRITE_WCS_R-HEAD>
                    0C29  1937                                                  ;PARITY
                    0C29  1938                  RET
                C9  0C29                .BYTE   ^0311
                    0C2A  1939
                    0C2A  1940
                    0C2A  1941 ;*****************************************************************************
                    0C2A  1942 ;
                    0C2A  1943 ;  CHECK_WCS LOADED
                    0C2A  1944 ;
                    0C2A  1945 ;  THIS ROUTINE CHECKS THE WCS LOADED FLAG AND REPORTS AN ERROR IF SET.  IT
                    0C2A  1946 ;  WILL POP THE STACK IF ERROR IS SET, SO THAT THE RETURN IN THE ERROR ROUTINE
                    0C2A  1947 ;  WILL SKIP THE REST OF THE ROUTINE THAT HAD THE ERROR.
                    0C2A  1948 ;
                    0C2A  1949 ;*****************************************************************************
                    0C2A  1950
                    0C2A  1951 CHECK_WCS_LOADED:
                    0C2A  1952                  LDA     WCS_LOADED              ;GET WCS LOADED FLAG
                  3A  0C2A                .BYTE   ^072
              0E83' 0C2B                .WORD   <WCS_LOADED-HEAD>
                    0C2D  1953                  ORA     A                       ;SET CONDITION CODES
                  B7  0C2D                .BYTE   ^0260 ! A
                    0C2E  1954                  RNZ                             ;RETURN IF WCS LOADED
                  C0  0C2E                .BYTE   ^0300
                    0C2F  1955
                    0C2F  1956                  CLEAR   NOADD_FLG               ;ELSE CLEAR FLAG IF SET
                  AF  0C2F                .BYTE   ^0250 ! A               ;with A
```

```
                 32  0C30                .BYTE   ^062
              03BD'  0C31                .WORD   <NOADD_FLG-HEAD>
                     0C33  1957                  POP     $PSW                    ;POP THE STACK TO REMOVE
                F1   0C33                .BYTE   ^0301 ! <8*$PSW>
                     0C34  1958                                                  ; RETURN ADDRESS TO CALLING
                     0C34  1959                                                  ; ROUTINE
                     0C34  1960                  JMP     ERROR_ROUTINE           ;GO REPORT ERROR BELOW
                C3   0C34                .BYTE   ^0303
              0C4A'  0C35                .WORD   <ERROR_ROUTINE-HEAD>
                     0C37  1961
                     0C37  1962 ;*****************************************************************************
                     0C37  1963 ;
                     0C37  1964 ;  CHECK_ERR_FLG:
                     0C37  1965 ;
                     0C37  1966 ;  THIS ROUTINE CHECKS THE ERROR FLAG AND REPORTS AN ERROR IF SET.  IT WILL
                     0C37  1967 ;  POP THE STACK IF ERROR IS SET, SO THAT THE RETURN IN THE ERROR ROUTINE
                     0C37  1968 ;  WILL SKIP THE REST OF THE ROUTINE THAT HAD THE ERROR.
                     0C37  1969 ;
                     0C37  1970 ;*****************************************************************************
                     0C37  1971
                     0C37  1972 INPUT_NUM_CHECK:CALL     INPUT_NUM_IF            ;GET INPUT DATA
                CD   0C37                .BYTE   ^0315
              0D9B'  0C38                .WORD   <INPUT_NUM_IF-HEAD>
                     0C3A  1973
                     0C3A  1974 CHECK_ERR_10:    MVI     B,<^X10>                ;ERROR CODE IF ERROR SET
             10 06   0C3A                .BYTE   <B*8> ! 6 , ^X10&255
                     0C3C  1975                                                  ; (NO INPUT ADDRESS OR DATA)
                     0C3C  1976
                     0C3C  1977 CHECK_ERR_FLG:   LDA     ERROR                   ;GET ERROR FLAG
                3A   0C3C                .BYTE   ^072
              02E2'  0C3D                .WORD   <ERROR-HEAD>
                     0C3F  1978                  ORA     A                       ;SET CONDITION CODES
                B7   0C3F                .BYTE   ^0260 ! A
                     0C40  1979                  RZ                              ;RETURN IF NO ERROR
                C8   0C40                .BYTE   ^0310
                     0C41  1980
                     0C41  1981                  POP     $PSW                    ;POP THE STACK TO REMOVE
                F1   0C41                .BYTE   ^0301 ! <8*$PSW>
                     0C42  1982                                                  ; RETURN ADDRESS TO CALLING
                     0C42  1983                                                  ; ROUTINE
                     0C42  1984                  JMP     ERROR_ROUTINE           ;GO REPORT ERROR BELOW
                C3   0C42                .BYTE   ^0303
              0C4A'  0C43                .WORD   <ERROR_ROUTINE-HEAD>
                     0C45  1985
                     0C45  1986 ;*****************************************************************************
                     0C45  1987 ;
                     0C45  1988 ;  ERROR_ROUTINE
                     0C45  1989 ;  THIS ROUTINE WILL REPORT AN ERROR AND ITS ERROR NUMBER
                     0C45  1990 ;
                     0C45  1991 ;  INPUT CONDITIONS:
                     0C45  1992 ;       B = ERROR CODE
                     0C45  1993 ;
                     0C45  1994 ;*****************************************************************************
                     0C45  1995
                     0C45  1996 ERROR_ROUTINE_NO_RET:
                     0C45  1997                  SET     NO_RETURN               ;SET FLAG, JUMP PARSER AFTER
                AF   0C45                .BYTE   ^0250 ! A               ;with A
```

```
                3C  0C46                .BYTE   ^O4 ! <8*A>
                32  0C47                .BYTE   ^O62
             038F'  0C48                .WORD   <NO_RETURN-HEAD>
                    0C4A  1998                                          ; ERROR REPORTED
                    0C4A  1999 ERROR_ROUTINE:   MOV     A,B             ;ERROR CODE FROM B TO AC
                78  0C4A                .BYTE   ^O100 ! <A*8> ! B
                    0C4B  2000                  STA     HEX_BUF         ;SAVE ERROR CODE
                32  0C4B                .BYTE   ^O62
              005B' 0C4C                .WORD   <HEX_BUF-HEAD>
                    0C4E  2001                  STA     APT_ERROR_NUMBER+1      ;SAVE IN APT MAILBOX
                32  0C4E                .BYTE   ^O62
              0013' 0C4F                .WORD   <APT_ERROR_NUMBER+1-HEAD>
                    0C51  2002
                    0C51  2003                  MVI     B,<^X25>        ;CHECK FOR TU58 ERROR
             25 06  0C51                .BYTE   <B*8> ! 6 , ^X25&255
                    0C53  2004                  XRA     B
                A8  0C53                .BYTE   ^O250 ! B               ;with A
                    0C54  2005                  IFZ                     ;REPORT BAD TU58 IF ?25 ERROR
                C2  0C54                .BYTE   ^O302
              0C75' 0C55                .WORD   <2020$-HEAD>
                    0C57  2006                  CALL    CRLF_R
                CD  0C57                .BYTE   ^O315
              101A' 0C58                .WORD   <CRLF_R-HEAD>
                    0C5A  2007                  LXI     H,BAD_TAPE_A
                21  0C5A                .BYTE   <H*8> ! 1
             01'99' 0C5B                .BYTE   <BAD_TAPE_A-HEAD>&255 , <<BAD_TAPE_A-HEAD>&^XFF00>/256
                    0C5D  2008                  CALL    PRINT_STRING_R
                CD  0C5D                .BYTE   ^O315
              100E' 0C5E                .WORD   <PRINT_STRING_R-HEAD>
                    0C60  2009                  CALL    CRLF_R
                CD  0C60                .BYTE   ^O315
              101A' 0C61                .WORD   <CRLF_R-HEAD>
                    0C63  2010                  LXI     H,ABORT_A
                21  0C63                .BYTE   <H*8> ! 1
             01'64' 0C64                .BYTE   <ABORT_A-HEAD>&255 , <<ABORT_A-HEAD>&^XFF00>/256
                    0C66  2011                  CALL    PRINT_STRING_R
                CD  0C66                .BYTE   ^O315
              100E' 0C67                .WORD   <PRINT_STRING_R-HEAD>
                    0C69  2012                  CALL    CRLF_R
                CD  0C69                .BYTE   ^O315
              101A' 0C6A                .WORD   <CRLF_R-HEAD>
                    0C6C  2013                  LXI     H,END_A
                21  0C6C                .BYTE   <H*8> ! 1
             02'62' 0C6D                .BYTE   <END_A-HEAD>&255 , <<END_A-HEAD>&^XFF00>/256
                    0C6F  2014                  CALL    PRINT_STRING_R
                CD  0C6F                .BYTE   ^O315
              100E' 0C70                .WORD   <PRINT_STRING_R-HEAD>
                    0C72  2015                  JMP     STNDRD_CON_COM          ;CANNOT CONTINUE CRD
                C3  0C72                .BYTE   ^O303
              098A' 0C73                .WORD   <STNDRD_CON_COM-HEAD>
                    0C75  2016                  ENDIF
                    0C75        2020$:                                  ;GENERATE LCL 2020 NAME
                    0C75  2017
                    0C75  2018                  MVI     A,1             ;1 IN ACCUM
             01 3E  0C75                .BYTE   <A*8> ! 6 , 1&255
                    0C77  2019                  STA     APT_ERROR_NUMBER        ;1 IN HIGH BYTE TO INDICATE
                32  0C77                .BYTE   ^O62
```

```
                0012' 0C78                .WORD   <APT_ERROR_NUMBER-HEAD>
                      0C7A  2020                                                ; MONITOR ERROR
                      0C7A  2021
                      0C7A  2022                CLEAR   ERROR                   ;RESET ERROR CONDITON
                  AF  0C7A                .BYTE   ^O250 ! A               ;with A
                  32  0C7B                .BYTE   ^O62
                02E2' 0C7C                .WORD   <ERROR-HEAD>
                      0C7E  2023                CLEAR   NOADD_FLG               ;MAKE SURE EXAM FLAG IS CLEAR
                  AF  0C7E                .BYTE   ^O250 ! A               ;with A
                  32  0C7F                .BYTE   ^O62
                03BD' 0C80                .WORD   <NOADD_FLG-HEAD>
                      0C82  2024                CLEAR   CTLDIS                  ;ENABLE CONTROL CHAR CHECK
                  AF  0C82                .BYTE   ^O250 ! A               ;with A
                  32  0C83                .BYTE   ^O62
                40FE  0C84                .WORD   <CTLDIS-HEAD>
                      0C86  2025                                                ; IF DISABLED
                      0C86  2026                MVI     A,2                     ;INDICATE ERROR
               02 3E  0C86                .BYTE   <A*8> ! 6 , 2&255
                      0C88  2027                STA     APT_MESSAGE_CODE+1      ;STORE IN MAILBOX
                  32  0C88                .BYTE   ^O62
                0011' 0C89                .WORD   <APT_MESSAGE_CODE+1-HEAD>
                      0C8B  2028
                      0C8B  2029                LXI     H,FAIL_A                ;PRINT *FAIL* MESSAGE
                  21  0C8B                .BYTE   <H*8> ! 1
              02'EF'  0C8C                .BYTE   <FAIL_A-HEAD>&255 , <<FAIL_A-HEAD>&^XFF00>/256
                      0C8E  2030                CALL    PRINT_STRING_R
                  CD  0C8E                .BYTE   ^O315
                100E' 0C8F                .WORD   <PRINT_STRING_R-HEAD>
                      0C91  2031                CALL    CRLF_R                  ;PRINT CRLF
                  CD  0C91                .BYTE   ^O315
                101A' 0C92                .WORD   <CRLF_R-HEAD>
                      0C94  2032                LXI     H,ABORT_A               ;PRINT AUTOTEST ABORT MESSAGE
                  21  0C94                .BYTE   <H*8> ! 1
              01'64'  0C95                .BYTE   <ABORT_A-HEAD>&255 , <<ABORT_A-HEAD>&^XFF00>/256
                      0C97  2033                CALL    PRINT_STRING_R
                  CD  0C97                .BYTE   ^O315
                100E' 0C98                .WORD   <PRINT_STRING_R-HEAD>
                      0C9A  2034
                      0C9A  2035                LXI     H,CPU_A                 ;CPU IS POSSIBLY BAD
                  21  0C9A                .BYTE   <H*8> ! 1
              02'57'  0C9B                .BYTE   <CPU_A-HEAD>&255 , <<CPU_A-HEAD>&^XFF00>/256
                      0C9D  2036                CALL    PRINT_STRING_R          ;PRINT
                  CD  0C9D                .BYTE   ^O315
                100E' 0C9E                .WORD   <PRINT_STRING_R-HEAD>
                      0CA0  2037
                      0CA0  2038                MVI     C,5                     ;BYTE COUNT
               05 0E  0CA0                .BYTE   <C*8> ! 6 , 5&255
                      0CA2  2039                LXI     H,SECTION_NAME          ;LOAD SECTION NAME INTO DE
                  21  0CA2                .BYTE   <H*8> ! 1
              08'CA'  0CA3                .BYTE   <SECTION_NAME-HEAD>&255 , <<SECTION_NAME-HEAD>&^XFF00>/256
                      0CA5  2040                XCHG
                  EB  0CA5                .BYTE   ^O353
                      0CA6  2041
                      0CA6  2042                LXI     H,ENKCC_A               ;SEE IF ENKCC IS BEING RUN
                  21  0CA6                .BYTE   <H*8> ! 1
              02'C9'  0CA7                .BYTE   <ENKCC_A-HEAD>&255 , <<ENKCC_A-HEAD>&^XFF00>/256
                      0CA9  2043                CALL    COMPARE_R
```

```
                CD  0CA9                .BYTE   ^O315
              0F55' 0CAA                .WORD   <COMPARE_R-HEAD>
                    0CAC  2044          IFZ                             ;IF IT IS
                C2  0CAC                .BYTE   ^O302
              0CBB' 0CAD                .WORD   <2021$-HEAD>
                    0CAF  2045          LXI     H,OR_A                  ;THEN PRINT MEMORY BAD
                21  0CAF                .BYTE   <H*8> ! 1
             03'D0' 0CB0                .BYTE   <OR_A-HEAD>&255 , <<OR_A-HEAD>&^XFF00>/256
                    0CB2  2046          CALL    PRINT_STRING_R
                CD  0CB2                .BYTE   ^O315
              100E' 0CB3                .WORD   <PRINT_STRING_R-HEAD>
                    0CB5  2047          LXI     H,MEMORY_A
                21  0CB5                .BYTE   <H*8> ! 1
             03'4D' 0CB6                .BYTE   <MEMORY_A-HEAD>&255 , <<MEMORY_A-HEAD>&^XFF00>/256
                    0CB8  2048          CALL    PRINT_STRING_R
                CD  0CB8                .BYTE   ^O315
              100E' 0CB9                .WORD   <PRINT_STRING_R-HEAD>
                    0CBB  2049          ENDIF
                    0CBB        2021$:                                  ;GENERATE LCL 2021 NAME
                    0CBB  2050
                    0CBB  2051          LXI     H,ENKCE_A               ;SEE IF ENKCE IS BEING RUN
                21  0CBB                .BYTE   <H*8> ! 1
             02'CE' 0CBC                .BYTE   <ENKCE_A-HEAD>&255 , <<ENKCE_A-HEAD>&^XFF00>/256
                    0CBE  2052          CALL    COMPARE_R
                CD  0CBE                .BYTE   ^O315
              0F55' 0CBF                .WORD   <COMPARE_R-HEAD>
                    0CC1  2053          IFZ                             ;IF IT IS
                C2  0CC1                .BYTE   ^O302
              0CD0' 0CC2                .WORD   <2022$-HEAD>
                    0CC4  2054          LXI     H,OR_A                  ;THEN PRINT FPA BAD
                21  0CC4                .BYTE   <H*8> ! 1
             03'D0' 0CC5                .BYTE   <OR_A-HEAD>&255 , <<OR_A-HEAD>&^XFF00>/256
                    0CC7  2055          CALL    PRINT_STRING_R
                CD  0CC7                .BYTE   ^O315
              100E' 0CC8                .WORD   <PRINT_STRING_R-HEAD>
                    0CCA  2056          LXI     H,FPA_A
                21  0CCA                .BYTE   <H*8> ! 1
             02'FE' 0CCB                .BYTE   <FPA_A-HEAD>&255 , <<FPA_A-HEAD>&^XFF00>/256
                    0CCD  2057          CALL    PRINT_STRING_R
                CD  0CCD                .BYTE   ^O315
              100E' 0CCE                .WORD   <PRINT_STRING_R-HEAD>
                    0CD0  2058          ENDIF
                    0CD0        2022$:                                  ;GENERATE LCL 2022 NAME
                    0CD0  2059
                    0CD0  2060          LXI     H,ENKCH_A               ;SEE IF ENKCH IS BEING RUN
                21  0CD0                .BYTE   <H*8> ! 1
             02'D3' 0CD1                .BYTE   <ENKCH_A-HEAD>&255 , <<ENKCH_A-HEAD>&^XFF00>/256
                    0CD3  2061          CALL    COMPARE_R
                CD  0CD3                .BYTE   ^O315
              0F55' 0CD4                .WORD   <COMPARE_R-HEAD>
                    0CD6  2062          IFZ                             ;IF IT IS
                C2  0CD6                .BYTE   ^O302
              0CE5' 0CD7                .WORD   <2023$-HEAD>
                    0CD9  2063          LXI     H,OR_A                  ;THEN PRINT IDC BAD
                21  0CD9                .BYTE   <H*8> ! 1
             03'D0' 0CDA                .BYTE   <OR_A-HEAD>&255 , <<OR_A-HEAD>&^XFF00>/256
                    0CDC  2064          CALL    PRINT_STRING_R
```

```
                CD  0CDC                .BYTE   ^O315
              100E' 0CDD                .WORD   <PRINT_STRING_R-HEAD>
                    0CDF  2065                  LXI     H,IDC_A
                21  0CDF                .BYTE   <H*8> ! 1
             03'3E' 0CE0                .BYTE   <IDC_A-HEAD>&255 , <<IDC_A-HEAD>&^XFF00>/256
                    0CE2  2066                  CALL    PRINT_STRING_R
                CD  0CE2                .BYTE   ^O315
              100E' 0CE3                .WORD   <PRINT_STRING_R-HEAD>
                    0CE5  2067                  ENDIF
                    0CE5        2023$:                                      ;GENERATE LCL 2023 NAME
                    0CE5  2068
                    0CE5  2069                  LXI     H,ENKCF_A           ;SEE IF ENKCF IS BEING RUN
                21  0CE5                .BYTE   <H*8> ! 1
             02'D8' 0CE6                .BYTE   <ENKCF_A-HEAD>&255 , <<ENKCF_A-HEAD>&^XFF00>/256
                    0CE8  2070                  CALL    COMPARE_R
                CD  0CE8                .BYTE   ^O315
              0F55' 0CE9                .WORD   <COMPARE_R-HEAD>
                    0CEB  2071                  IFZ                         ;IF IT IS
                C2  0CEB                .BYTE   ^O302
              0CFA' 0CEC                .WORD   <2024$-HEAD>
                    0CEE  2072                  LXI     H,OR_A              ;THEN PRINT IDC BAD
                21  0CEE                .BYTE   <H*8> ! 1
             03'D0' 0CEF                .BYTE   <OR_A-HEAD>&255 , <<OR_A-HEAD>&^XFF00>/256
                    0CF1  2073                  CALL    PRINT_STRING_R
                CD  0CF1                .BYTE   ^O315
              100E' 0CF2                .WORD   <PRINT_STRING_R-HEAD>
                    0CF4  2074                  LXI     H,IDC_A
                21  0CF4                .BYTE   <H*8> ! 1
             03'3E' 0CF5                .BYTE   <IDC_A-HEAD>&255 , <<IDC_A-HEAD>&^XFF00>/256
                    0CF7  2075                  CALL    PRINT_STRING_R
                CD  0CF7                .BYTE   ^O315
              100E' 0CF8                .WORD   <PRINT_STRING_R-HEAD>
                    0CFA  2076                  ENDIF
                    0CFA        2024$:                                      ;GENERATE LCL 2024 NAME
                    0CFA  2077
                    0CFA  2078                  LXI     H,ENKCG_A           ;SEE IF ENKCF IS BEING RUN
                21  0CFA                .BYTE   <H*8> ! 1
             02'DD' 0CFB                .BYTE   <ENKCG_A-HEAD>&255 , <<ENKCG_A-HEAD>&^XFF00>/256
                    0CFD  2079                  CALL    COMPARE_R
                CD  0CFD                .BYTE   ^O315
              0F55' 0CFE                .WORD   <COMPARE_R-HEAD>
                    0D00  2080                  IFZ                         ;IF IT IS
                C2  0D00                .BYTE   ^O302
              0D18' 0D01                .WORD   <2025$-HEAD>
                    0D03  2081                  LXI     H,OR_A              ;THEN PRINT IDC BAD
                21  0D03                .BYTE   <H*8> ! 1
             03'D0' 0D04                .BYTE   <OR_A-HEAD>&255 , <<OR_A-HEAD>&^XFF00>/256
                    0D06  2082                  CALL    PRINT_STRING_R
                CD  0D06                .BYTE   ^O315
              100E' 0D07                .WORD   <PRINT_STRING_R-HEAD>
                    0D09  2083                  LXI     H,IDC_A
                21  0D09                .BYTE   <H*8> ! 1
             03'3E' 0D0A                .BYTE   <IDC_A-HEAD>&255 , <<IDC_A-HEAD>&^XFF00>/256
                    0D0C  2084                  CALL    PRINT_STRING_R
                CD  0D0C                .BYTE   ^O315
              100E' 0D0D                .WORD   <PRINT_STRING_R-HEAD>
                    0D0F  2085                  LXI     H,OR_A              ;PRINT RL02 BAD
```

```
                21  0D0F                .BYTE   <H*8> ! 1
         03'D0' 0D10                    .BYTE   <OR_A-HEAD>&255 , <<OR_A-HEAD>&^XFF00>/256
                0D12  2086              CALL    PRINT_STRING_R
                CD  0D12                .BYTE   ^O315
                100E' 0D13              .WORD   <PRINT_STRING_R-HEAD>
                0D15  2087              LXI     H,RL02_A
                21  0D15                .BYTE   <H*8> ! 1
         03'FE' 0D16                    .BYTE   <RL02_A-HEAD>&255 , <<RL02_A-HEAD>&^XFF00>/256
                0D18  2088              CALL    PRINT_STRING_R
                CD  0D18                .BYTE   ^O315
                100E' 0D19              .WORD   <PRINT_STRING_R-HEAD>
                0D1B  2089              ENDIF
                0D1B        2025$:                                  ;GENERATE LCL 2025 NAME
                0D1B  2090
                0D1B  2091              LXI     H,BAD_A
                21  0D1B                .BYTE   <H*8> ! 1
         01'7E' 0D1C                    .BYTE   <BAD_A-HEAD>&255 , <<BAD_A-HEAD>&^XFF00>/256
                0D1E  2092              CALL    PRINT_STRING_R
                CD  0D1E                .BYTE   ^O315
                100E' 0D1F              .WORD   <PRINT_STRING_R-HEAD>
                0D21  2093
                0D21  2094 ERR_MENU:    CALL    CRLF_R
                CD  0D21                .BYTE   ^O315
                101A' 0D22              .WORD   <CRLF_R-HEAD>
                0D24  2095
                0D24  2096              LXI     H,MENU_A                ;PRINT OPTIONS
                21  0D24                .BYTE   <H*8> ! 1
         03'55' 0D25                    .BYTE   <MENU_A-HEAD>&255 , <<MENU_A-HEAD>&^XFF00>/256
                0D27  2097              CALL    PRINT_STRING_R
                CD  0D27                .BYTE   ^O315
                100E' 0D28              .WORD   <PRINT_STRING_R-HEAD>
                0D2A  2098
                0D2A  2099              CALL    INPUT_LINE              ;GET DESIRED OPTION
                CD  0D2A                .BYTE   ^O315
                102E' 0D2B              .WORD   <INPUT_LINE-HEAD>
                0D2D  2100
                0D2D  2101              MVI     C,1
         01 0E  0D2D                    .BYTE   <C*8> ! 6 , 1&255
                0D2F  2102              LXI     H,TTBUF                 ;POINT TO USER RESPONSE
                21  0D2F                .BYTE   <H*8> ! 1
         08'D1' 0D30                    .BYTE   <TTBUF-HEAD>&255 , <<TTBUF-HEAD>&^XFF00>/256
                0D32  2103              XCHG
                EB  0D32                .BYTE   ^O353
                0D33  2104
                0D33  2105              LXI     H,ASCII_1               ;WAS A 1 TYPED?
                21  0D33                .BYTE   <H*8> ! 1
         01'7B' 0D34                    .BYTE   <ASCII_1-HEAD>&255 , <<ASCII_1-HEAD>&^XFF00>/256
                0D36  2106              CALL    COMPARE_R
                CD  0D36                .BYTE   ^O315
                0F55' 0D37              .WORD   <COMPARE_R-HEAD>
                0D39  2107
                0D39  2108              IFZ                             ;IF 1 WAS TYPED
                C2  0D39                .BYTE   ^O302
                0D4B' 0D3A              .WORD   <2026$-HEAD>
                0D3C  2109              CALL    CRLF_R
                CD  0D3C                .BYTE   ^O315
                101A' 0D3D              .WORD   <CRLF_R-HEAD>
```

```
                 0D3F  2110                     LXI     H,END_A
          21     0D3F                  .BYTE    <H*8> ! 1
  02'62'         0D40                  .BYTE    <END_A-HEAD>&255 , <<END_A-HEAD>&^XFF00>/256
                 0D42  2111                     CALL    PRINT_STRING_R
          CD     0D42                  .BYTE    ^O315
   100E'         0D43                  .WORD    <PRINT_STRING_R-HEAD>
                 0D45  2112                     CALL    CRLF_R
          CD     0D45                  .BYTE    ^O315
   101A'         0D46                  .WORD    <CRLF_R-HEAD>
                 0D48  2113                     JMP     STNDRD_CON_COM          ;THEN GO BACK TO CONSOLE
          C3     0D48                  .BYTE    ^O303
   098A'         0D49                  .WORD    <STNDRD_CON_COM-HEAD>
                 0D4B  2114                     ENDIF
                 0D4B        2026$:                                             ;GENERATE LCL 2026 NAME
                 0D4B  2115
                 0D4B  2116                     LXI     H,ASCII_2
          21     0D4B                  .BYTE    <H*8> ! 1
  01'7C'         0D4C                  .BYTE    <ASCII_2-HEAD>&255 , <<ASCII_2-HEAD>&^XFF00>/256
                 0D4E  2117                     CALL    COMPARE_R
          CD     0D4E                  .BYTE    ^O315
   0F55'         0D4F                  .WORD    <COMPARE_R-HEAD>
                 0D51  2118
                 0D51  2119                     IFZ                             ;IF 2 WAS TYPED
          C2     0D51                  .BYTE    ^O302
   0D57'         0D52                  .WORD    <2027$-HEAD>
                 0D54  2120                     JMP     START_UP                ;THEN START AUTO MODE
          C3     0D54                  .BYTE    ^O303
   1CD2'         0D55                  .WORD    <START_UP-HEAD>
                 0D57  2121                     ENDIF
                 0D57        2027$:                                             ;GENERATE LCL 2027 NAME
                 0D57  2122
                 0D57  2123                     LXI     H,ASCII_3
          21     0D57                  .BYTE    <H*8> ! 1
  01'7D'         0D58                  .BYTE    <ASCII_3-HEAD>&255 , <<ASCII_3-HEAD>&^XFF00>/256
                 0D5A  2124                     CALL    COMPARE_R
          CD     0D5A                  .BYTE    ^O315
   0F55'         0D5B                  .WORD    <COMPARE_R-HEAD>
                 0D5D  2125
                 0D5D  2126                     IFZ                             ;IF 3 WAS TYPED
          C2     0D5D                  .BYTE    ^O302
   0D76'         0D5E                  .WORD    <2028$-HEAD>
                 0D60  2127
                 0D60  2128                     LXI     H,CRLF_A                ;POINT TO CR
          21     0D60                  .BYTE    <H*8> ! 1
  02'5B'         0D61                  .BYTE    <CRLF_A-HEAD>&255 , <<CRLF_A-HEAD>&^XFF00>/256
                 0D63  2129                     CALL    PRINT_STRING_APT        ;PRINT CRLF
          CD     0D63                  .BYTE    ^O315
   1014'         0D64                  .WORD    <PRINT_STRING_APT-HEAD>
                 0D66  2130                     MVI     A,HEX_3F                ;QUESTION MARK
3F 3E            0D66                  .BYTE    <A*8> ! 6 , HEX_3F&255
                 0D68  2131                     CALL    TYPE_CHAR
          CD     0D68                  .BYTE    ^O315
   1027'         0D69                  .WORD    <TYPE_CHAR-HEAD>
                 0D6B  2132
                 0D6B  2133                     MVI     C,1                     ;PRINT 1 BYTE (2 DIGITS)
01 0E            0D6B                  .BYTE    <C*8> ! 6 , 1&255
                 0D6D  2134                     CALL    PRINT_HEX_R             ;PRINT ERROR CODE IN HEX BUF
```

```
                CD  0D6D                .BYTE   ^0315
              0F96' 0D6E                .WORD   <PRINT_HEX_R-HEAD>
                    0D70  2135
                    0D70  2136                  LXI     H,ERROR_A               ;PRINT '' ERROR''
                21  0D70                .BYTE   <H*8> ! 1
             02'E3' 0D71                .BYTE   <ERROR_A-HEAD>&255 , <<ERROR_A-HEAD>&^XFF00>/256
                    0D73  2137                  CALL    PRINT_STRING_APT
                CD  0D73                .BYTE   ^0315
              1014' 0D74                .WORD   <PRINT_STRING_APT-HEAD>
                    0D76  2138                  ENDIF
                    0D76        2028$:                                          ;GENERATE LCL 2028 NAME
                    0D76  2139                  JMP     ERR_MENU
                C3  0D76                .BYTE   ^0303
              0D21' 0D77                .WORD   <ERR_MENU-HEAD>
                    0D79  2140
                    0D79  2141                  LDA     NO_RETURN               ;SEE IF CAN'T RETURN
                3A  0D79                .BYTE   ^072
              03BF' 0D7A                .WORD   <NO_RETURN-HEAD>
                    0D7C  2142                  ORA     A                       ;SET CONDITION CODES
                B7  0D7C                .BYTE   ^0260 ! A
                    0D7D  2143                  RZ                              ;RETURN IF FLAG CLEAR
                C8  0D7D                .BYTE   ^0310
                    0D7E  2144
                    0D7E  2145                  CLEAR   NO_RETURN               ;ELSE CLEAR FLAG
                AF  0D7E                .BYTE   ^0250 ! A               ;with A
                32  0D7F                .BYTE   ^062
              03BF' 0D80                .WORD   <NO_RETURN-HEAD>
                    0D82  2146                  CLEAR   CONTINUE                ;NO CONTINUE ALLOWED
                AF  0D82                .BYTE   ^0250 ! A               ;with A
                32  0D83                .BYTE   ^062
              003A' 0D84                .WORD   <CONTINUE-HEAD>
                    0D86  2147                  LXI     $SP,PRI_STACK           ;RESET STACK POINTER
                31  0D86                .BYTE   <$SP*8> ! 1
             40 80  0D87                .BYTE   <PRI_STACK-HEAD>&255 , <<PRI_STACK-HEAD>&^XFF00>/256
                    0D89  2148                  JMP     PARSER                  ;GO TO PARSER
                C3  0D89                .BYTE   ^0303
              1DAE' 0D8A                .WORD   <PARSER-HEAD>
                    0D8C  2149
                    0D8C  2150 ;*****************************************************************************
                    0D8C  2151 ;
                    0D8C  2152 ;  PRINT_CSR_VALUE
                    0D8C  2153 ;  THIS ROUTINE PRINTS THE VALUE IN THE FIELD CSR_VALUE
                    0D8C  2154 ;
                    0D8C  2155 ;*****************************************************************************
                    0D8C  2156
                    0D8C  2157 PRINT_CSR_VAL:   LXI     H,CSR_VALUE
                21  0D8C                .BYTE   <H*8> ! 1
             00'B8' 0D8D                .BYTE   <CSR_VALUE-HEAD>&255 , <<CSR_VALUE-HEAD>&^XFF00>/256
                    0D8F  2158                  LXI     D,HEX_BUF               ;MOVE CSR_VALUE TO HEX NUMBER
                11  0D8F                .BYTE   <D*8> ! 1
             00'5B' 0D90                .BYTE   <HEX_BUF-HEAD>&255 , <<HEX_BUF-HEAD>&^XFF00>/256
                    0D92  2159                  CALL    MOVER_3_R               ;PRINT AREA
                CD  0D92                .BYTE   ^0315
              0F5F' 0D93                .WORD   <MOVER_3_R-HEAD>
                    0D95  2160
                    0D95  2161                  MVI     C,3
             03 0E  0D95                .BYTE   <C*8> ! 6 , 3&255
```

```
                        0D97  2162                    CALL    PRINT_HEX_R             ;PRINT 3 BYTES OF CSR
              CD        0D97                    .BYTE   ^0315
          0F96'         0D98                    .WORD   <PRINT_HEX_R-HEAD>
                        0D9A  2163
                        0D9A  2164                    RET
              C9        0D9A                    .BYTE   ^0311
                        0D9B  2165
                        0D9B  2166
                        0D9B  2167 ;*****************************************************************************
                        0D9B  2168 ;
                        0D9B  2169 ;  INPUT_NUM_IF
                        0D9B  2170 ;  THIS ROUTINE TAKES A NUMBER FROM THE INPUT LINE
                        0D9B  2171 ;
                        0D9B  2172 ;  INPUT CONDITIONS
                        0D9B  2173 ;       TTBUF_PNT = POINTER TO INPUT STRING
                        0D9B  2174 ;
                        0D9B  2175 ;  OUTPUT CONDITIONS
                        0D9B  2176 ;       MINUS = 1 IF MINUS SIGN FOUND,0 IF NOT
                        0D9B  2177 ;       ERROR IS SET IF AN ILLEGAL HEX CHAR IS FOUND
                        0D9B  2178 ;
                        0D9B  2179 ;*****************************************************************************
                        0D9B  2180
                        0D9B  2181 INPUT_NUM_IF:    CLEAR   ERROR
              AF        0D9B                    .BYTE   ^0250 ! A               ;with A
              32        0D9C                    .BYTE   ^062
          02E2'         0D9D                    .WORD   <ERROR-HEAD>
                        0D9F  2182
                        0D9F  2183                    IFEQI   TEMP_BUF,HEX_2D         ;IF MINUS SIGN
              3A        0D9F                    .BYTE   ^072
          040E'         0DA0                    .WORD   <TEMP_BUF-HEAD>
              FE        0DA2                    .BYTE   ^0376
              2D        0DA3                    .BYTE   HEX_2D
              C2        0DA4                    .BYTE   ^0302
          0DB5'         0DA5                    .WORD   <2029$-HEAD>
                        0DA7  2184
                        0DA7  2185                    SET     MINUS                   ;MINUS = INFINITE LOOP
              AF        0DA7                    .BYTE   ^0250 ! A               ;with A
              3C        0DA8                    .BYTE   ^04 ! <8*A>
              32        0DA9                    .BYTE   ^062
          0388'         0DAA                    .WORD   <MINUS-HEAD>
                        0DAC  2186                    CALL    SKIP_TO_SPEC            ;SKIP - SIGN AND THEN
              CD        0DAC                    .BYTE   ^0315
          0F33'         0DAD                    .WORD   <SKIP_TO_SPEC-HEAD>
                        0DAF  2187                    CALL    SKIP_TO_SPEC            ;SKIP THE NUMBER
              CD        0DAF                    .BYTE   ^0315
          0E33'         0DB0                    .WORD   <SKIP_TO_SPEC-HEAD>
                        0DB2  2188
                        0DB2  2189                    ELSE
              C3        0DB2                    .BYTE   ^0303
          0DFC'         0DB3                    .WORD   <2030$-HEAD>
                        0DB5        2029$:                                          ;GENERATE LOCAL SYMBOL NAME
                        0DB5  2190
                        0DB5  2191                    CLEAR   MINUS                   ;NO MINUS SIGN FOUND
              AF        0DB5                    .BYTE   ^0250 ! A               ;with A
              32        0DB6                    .BYTE   ^062
          0388'         0DB7                    .WORD   <MINUS-HEAD>
                        0DB9  2192
```

```
                 0DB9  2193                    LXI     H,NUM_SHIFT             ;SET UP SHIFT POINTER
      21         0DB9                  .BYTE   <H*8> ! 1
 06'75'          0DBA                  .BYTE   <NUM_SHIFT-HEAD>&255 , <<NUM_SHIFT-HEAD>&^XFF00>/256
                 0DBC  2194                    SHLD    SHIFT_PNT               ;TO NUMBER BUFFER
      22         0DBC                  .BYTE   ^042
  0068'          0DBD                  .WORD   <SHIFT_PNT-HEAD>
                 0DBF  2195                    CLEAR   MINUS
      AF         0DBF                  .BYTE   ^0250 ! A               ;with A
      32         0DC0                  .BYTE   ^062
  038B'          0DC1                  .WORD   <MINUS-HEAD>
                 0DC3  2196                    ZERO    INPUT_NUM,4             ;CLEAR INPUT NUMBER AREA
      21         0DC3                  .BYTE   <H*8> ! 1
 06'76'          0DC4                  .BYTE   <INPUT_NUM-HEAD>&255 , <<INPUT_NUM-HEAD>&^XFF00>/256
 04 0E           0DC6                  .BYTE   <C*8> ! 6 , 4&255
      AF         0DC8                  .BYTE   ^0250 ! A               ;with A
                 0DC9          2031$:          MOV     M,A
      77         0DC9                  .BYTE   ^0100 ! <M*8> ! A
      23         0DCA                  .BYTE   <H*8> ! 3
      0D         0DCB                  .BYTE   ^05 ! <8*C>
      C2         0DCC                  .BYTE   ^0302
  0DC9'          0DCD                  .WORD   <2031$-HEAD>
                 0DCF  2197
                 0DCF  2198                    CALL    SKIP_TO_SPEC            ;SPAN AND GET COUNT
      CD         0DCF                  .BYTE   ^0315
  0E33'          0DD0                  .WORD   <SKIP_TO_SPEC-HEAD>
                 0DD2  2199
                 0DD2  2200                    LHLD    OLD_TTBUF
      2A         0DD2                  .BYTE   ^052
  03CD'          0DD3                  .WORD   <OLD_TTBUF-HEAD>
                 0DD5  2201                    SHLD    TEMP_WORD1
      22         0DD5                  .BYTE   ^042
  041B'          0DD6                  .WORD   <TEMP_WORD1-HEAD>
                 0DD8  2202
                 0DD8  2203 1$:                ORA     A                       ;CLEAR THE CARRY SO SHIFT IN
      B7         0DD8                  .BYTE   ^0260 ! A
                 0DD9  2204                                                    ;WILL BE ZERO'S
                 0DD9  2205                    CALL    SHIFTER_R               ;SHIFT NUMBER BUFF LEFT NIBBLE
      CD         0DD9                  .BYTE   ^0315
  0EB8'          0DDA                  .WORD   <SHIFTER_R-HEAD>
                 0DDC  2206                    CALL    SHIFTER_R
      CD         0DDC                  .BYTE   ^0315
  0EB8'          0DDD                  .WORD   <SHIFTER_R-HEAD>
                 0DDF  2207                    CALL    SHIFTER_R
      CD         0DDF                  .BYTE   ^0315
  0EB8'          0DE0                  .WORD   <SHIFTER_R-HEAD>
                 0DE2  2208                    CALL    SHIFTER_R
      CD         0DE2                  .BYTE   ^0315
  0EB8'          0DE3                  .WORD   <SHIFTER_R-HEAD>
                 0DE5  2209
                 0DE5  2210                    LHLD    TEMP_WORD1
      2A         0DE5                  .BYTE   ^052
  041B'          0DE6                  .WORD   <TEMP_WORD1-HEAD>
                 0DE8  2211                    MOV     A,M                     ;GET A CHAR AND BUMP POINTER
      7E         0DE8                  .BYTE   ^0100 ! <A*8> ! M
                 0DE9  2212                    INX     H
      23         0DE9                  .BYTE   <H*8> ! 3
                 0DEA  2213                    SHLD    TEMP_WORD1
```

```
                                                22  0DEA                .BYTE   ^042
                                              041B' 0DEB                .WORD   <TEMP_WORD1-HEAD>
                                                    0DED  2214
                                                    0DED  2215                CALL    ASCII_TO_BIN              ;CONVERT TO BINARY
                                                CD  0DED                .BYTE   ^0315
                                              0E15' 0DEE                .WORD   <ASCII_TO_BIN-HEAD>
                                                    0DF0  2216
                                                    0DF0  2217                LXI     H,INPUT_NUM+3             ;OR BINARY VALUE AND NIBBLE
                                                21  0DF0                .BYTE   <H*8> ! 1
                                              06'79' 0DF1                .BYTE   <INPUT_NUM+3-HEAD>&255 , <<INPUT_NUM+3-HEAD>&^XFF00>/256
                                                    0DF3  2218                ORA     M
                                                B6  0DF3                .BYTE   ^0260 ! M
                                                    0DF4  2219                MOV     M,A
                                                77  0DF4                .BYTE   ^0100 ! <M*8> ! A
                                                    0DF5  2220
                                                    0DF5  2221                LXI     H,SKIP_CNT
                                                21  0DF5                .BYTE   <H*8> ! 1
                                              04'06' 0DF6                .BYTE   <SKIP_CNT-HEAD>&255 , <<SKIP_CNT-HEAD>&^XFF00>/256
                                                    0DF8  2222                DCR     M                         ;DEC CHAR COUNT
                                                35  0DF8                .BYTE   ^05 ! <8*M>
                                                    0DF9  2223                JNZ     1$                        ;END?
                                                C2  0DF9                .BYTE   ^0302
                                              0DD8' 0DFA                .WORD   <1$-HEAD>
                                                    0DFC  2224
                                                    0DFC  2225                ENDIF
                                                    0DFC        2030$:                                        ;GENERATE LCL 2030 NAME
                                                    0DFC  2226
                                                    0DFC  2227                IF      ERROR
                                                3A  0DFC                .BYTE   ^072
                                              02E2' 0DFD                .WORD   <ERROR-HEAD>
                                                0F  0DFF                .BYTE   ^017
                                                D2  0E00                .BYTE   ^0322
                                              0E14' 0E01                .WORD   <2032$-HEAD>
                                                    0E03  2228
                                                    0E03  2229                LHLD    OLD_TTBUF                 ;NO NUMBERS FOUND MOVE POINTER
                                                2A  0E03                .BYTE   ^052
                                              03CD' 0E04                .WORD   <OLD_TTBUF-HEAD>
                                                    0E06  2230                SHLD    TTBUF_PNT                 ;BACK TO WHERE IT WAS ON ENTRY
                                                22  0E06                .BYTE   ^042
                                              044D' 0E07                .WORD   <TTBUF_PNT-HEAD>
                                                    0E09  2231
                                                    0E09  2232                LXI     H,OLD_TEMP_BUF
                                                21  0E09                .BYTE   <H*8> ! 1
                                              03'C3' 0E0A                .BYTE   <OLD_TEMP_BUF-HEAD>&255 , <<OLD_TEMP_BUF-HEAD>&^XFF00>/256
                                                    0E0C  2233                LXI     D,TEMP_BUF
                                                11  0E0C                .BYTE   <D*8> ! 1
                                              04'0E' 0E0D                .BYTE   <TEMP_BUF-HEAD>&255 , <<TEMP_BUF-HEAD>&^XFF00>/256
                                                    0E0F  2234                MVI     C,10
                                              0A 0E  0E0F                .BYTE   <C*8> ! 6 , 10&255
                                                    0E11  2235                CALL    MOVER_R
                                                CD  0E11                .BYTE   ^0315
                                              0F75' 0E12                .WORD   <MOVER_R-HEAD>
                                                    0E14  2236
                                                    0E14  2237                ENDIF
                                                    0E14        2032$:                                        ;GENERATE LCL 2032 NAME
                                                    0E14  2238
                                                    0E14  2239                RET
```

```
                C9  0E14                .BYTE   ^O311
                    0E15  2240
                    0E15  2241
                    0E15  2242 ;*****************************************************************************
                    0E15  2243 ;
                    0E15  2244 ;  ASCII_TO_BIN
                    0E15  2245 ;  THIS ROUTINE CONVERTS AN ASCII CHAR TO A BINARY NUMBER
                    0E15  2246 ;  IF THE CHAR IS OUT OF RANGE ERROR IS SET
                    0E15  2247 ;
                    0E15  2248 ;*****************************************************************************
                    0E15  2249
                    0E15  2250 ASCII_TO_BIN:    CPI     <^X30>
                FE  0E15                .BYTE   ^O376
                30  0E16                .BYTE   ^X30
                    0E17  2251                  JP      1$
                F2  0E17                .BYTE   ^O362
            0E1F'   0E18                .WORD   <1$-HEAD>
                    0E1A  2252                  SET     ERROR                   ;CHAR BELOW RANGE
                AF  0E1A                .BYTE   ^O250 ! A                   ;with A
                3C  0E1B                .BYTE   ^O4 ! <8*A>
                32  0E1C                .BYTE   ^O062
            02E2'   0E1D                .WORD   <ERROR-HEAD>
                    0E1F  2253 1$:              SUI     <^X30>
                D6  0E1F                .BYTE   ^O326
                30  0E20                .BYTE   ^X30
                    0E21  2254                  CPI     <^XA>
                FE  0E21                .BYTE   ^O376
                0A  0E22                .BYTE   ^XA
                    0E23  2255                  JM      3$
                FA  0E23                .BYTE   ^O372
            0E32'   0E24                .WORD   <3$-HEAD>
                    0E26  2256                  SUI     7
                D0  0E26                .BYTE   ^O326
                07  0E27                .BYTE   7
                    0E28  2257                  CPI     <^X10>
                FE  0E28                .BYTE   ^O376
                10  0E29                .BYTE   ^X10
                    0E2A  2258                  JM      3$
                FA  0E2A                .BYTE   ^O372
            0E32'   0E2B                .WORD   <3$-HEAD>
                    0E2D  2259                  SET     ERROR                   ;CHAR ABOVE RANGE
                AF  0E2D                .BYTE   ^O250 ! A                   ;with A
                3C  0E2E                .BYTE   ^O4 ! <8*A>
                32  0E2F                .BYTE   ^O062
            02E2'   0E30                .WORD   <ERROR-HEAD>
                    0E32  2260 3$:              RET
                C9  0E32                .BYTE   ^O311
                    0E33  2261
                    0E33  2262
                    0E33  2263 ;*****************************************************************************
                    0E33  2264 ;
                    0E33  2265 ;  SKIP_TO_SPEC
                    0E33  2266 ;  THIS ROUTINE SKIPS TO THE NEXT SPECIAL CHAR IN THE INPUT STRING
                    0E33  2267 ;  THEN MOVES ONE MORE.(WHICH SHOULD BE THE BEGINING OF THE NEXT WORD.)
                    0E33  2268 ;  IT ALSO LEAVES A COUNT IN SKIP_CNT FOR THE NUMBER OF CHARS SPANNED
                    0E33  2269 ;  AT END IT MOVES 10 CHARS FROM THE NOW POINTER INTO TEMP_BUF
                    0E33  2270 ;
```

```
                        0E33  2271 ;  INPUT CONDITIONS:
                        0E33  2272 ;       TTBUF_PNT        = POINTS TO CHARS IN THE INPUT LINE
                        0E33  2273 ;
                        0E33  2274 ;  OUTPUT CONDITIONS:
                        0E33  2275 ;       SKIP_CNT         = # CHARS SPANNED  (SEE NOTE)
                        0E33  2276 ;       TEMP_BUF         = 10 CHARS FROM POINTER
                        0E33  2277 ;       TTBUF_PNT        = NEW POINTER
                        0E33  2278 ;       OLD_TTBUF        = CHAR POINTER ON ENTRY (FOR RESET WHEN NO # FOUND)
                        0E33  2279 ;
                        0E33  2280 ;  NOTE:  SPANNED CHARS COUNT DOES NOT INCLUDE THE SPECIAL CHAR !!!
                        0E33  2281 ;
                        0E33  2282 ;*******************************************************************************
                        0E33  2283
                        0E33  2284 SKIP_TO_SPEC:   CLEAR   SKIP_CNT
              AF        0E33                       .BYTE   ^0250 ! A                ;with A
              32        0E34                       .BYTE   ^062
           0406'        0E35                       .WORD   <SKIP_CNT-HEAD>
                        0E37  2285
                        0E37  2286                 LHLD    TTBUF_PNT               ;POINTER TO INPUT CHARS
              2A        0E37                       .BYTE   ^052
           044D'        0E38                       .WORD   <TTBUF_PNT-HEAD>
                        0E3A  2287                 SHLD    OLD_TTBUF               ;SAVE OLD POINTER FOR ERRORS
              22        0E3A                       .BYTE   ^042
           03CD'        0E3B                       .WORD   <OLD_TTBUF-HEAD>
                        0E3D  2288
                        0E3D  2289                 LXI     H,TEMP_BUF              ;SAVE OLD BUFFER FOR ERRORS
              21        0E3D                       .BYTE   <H*8> ! 1
        04'0E'          0E3E                       .BYTE   <TEMP_BUF-HEAD>&255 , <<TEMP_BUF-HEAD>&^XFF00>/256
                        0E40  2290                 LXI     D,OLD_TEMP_BUF
              11        0E40                       .BYTE   <D*8> ! 1
        03'C3'          0E41                       .BYTE   <OLD_TEMP_BUF-HEAD>&255 , <<OLD_TEMP_BUF-HEAD>&^XFF00>/256
                        0E43  2291                 MVI     C,10
           0A 0E        0E43                       .BYTE   <C*8> ! 6 , 10&255
                        0E45  2292                 CALL    MOVER_R
              CD        0E45                       .BYTE   ^0315
           0F75'        0E46                       .WORD   <MOVER_R-HEAD>
                        0E48  2293
                        0E48  2294 1$:             LHLD    TTBUF_PNT
              2A        0E48                       .BYTE   ^052
           044D'        0E49                       .WORD   <TTBUF_PNT-HEAD>
                        0E4B  2295                 MOV     A,M
              7E        0E4B                       .BYTE   ^0100 ! <A*8> ! M
                        0E4C  2296                 CPI     HEX_30                  ;SPEC CHAR ARE <30 HEX
              FE        0E4C                       .BYTE   ^0376
              30        0E4D                       .BYTE   HEX_30
                        0E4E  2297
                        0E4E  2298                 INX     H                       ;MOVE TO NEXT INPUT CHAR
              23        0E4E                       .BYTE   <H*8> ! 3
                        0E4F  2299                 SHLD    TTBUF_PNT               ;STORE NEW POINTER.  NOTE:
              22        0E4F                       .BYTE   ^042
           044D'        0E50                       .WORD   <TTBUF_PNT-HEAD>
                        0E52  2300                                                 ; THESE 2 INSTRUCTIONS DO
                        0E52  2301                                                 ; NOT AFFECT CONDITION CODES
                        0E52  2302                 IFPOS                           ;IF NOT SPECIAL CHAR
              FA        0E52                       .BYTE   ^0372
           0E61'        0E53                       .WORD   <2033$-HEAD>
                        0E55  2303                 CPI     HEX_3A                  ;SEE IF : (ALSO SPECIAL CHAR)
```

```
                FE  0E55                .BYTE   ^0376
                3A  0E56                .BYTE   HEX_3A
                    0E57  2304                  JZ      2$                          ;JUMP IF :
                CA  0E57                .BYTE   ^0312
             0E61'  0E58                .WORD   <2$-HEAD>
                    0E5A  2305                  LXI     H,SKIP_CNT
                21  0E5A                .BYTE   <H*8> ! 1
           04'06'   0E5B                .BYTE   <SKIP_CNT-HEAD>&255 , <<SKIP_CNT-HEAD>&^XFF00>/256
                    0E5D  2306                  INR     M                           ;INCREMENT SPANNED CHAR CNT
                34  0E5D                .BYTE   ^04 ! <8*M>
                    0E5E  2307
                    0E5E  2308                  ELSE                                ;IF SPECIAL CHARACTER
                C3  0E5E                .BYTE   ^0303
             0E6A'  0E5F                .WORD   <2034$-HEAD>
                    0E61        2033$:                                              ;GENERATE LOCAL SYMBOL NAME
                    0E61  2309  2$:             MVI     C,10
             0A 0E  0E61                .BYTE   <C*8> ! 6 , 10&255
                    0E63  2310                  LXI     D,TEMP_BUF
                11  0E63                .BYTE   <D*8> ! 1
           04'0E'   0E64                .BYTE   <TEMP_BUF-HEAD>&255 , <<TEMP_BUF-HEAD>&^XFF00>/256
                    0E66  2311                  CALL    MOVER_R                     ;PUT 10 CHARS INTO TEMP_BUF
                CD  0E66                .BYTE   ^0315
             0F75'  0E67                .WORD   <MOVER_R-HEAD>
                    0E69  2312                  RET                                 ;GET OUT OF LOOP
                C9  0E69                .BYTE   ^0311
                    0E6A  2313
                    0E6A  2314                  ENDIF
                    0E6A        2034$:                                              ;GENERATE LCL 2034 NAME
                    0E6A  2315
                    0E6A  2316                  JMP     1$                          ;LOOP UNTIL SPECIAL CHAR
              C3    0E6A                .BYTE   ^0303
           0E48'    0E6B                .WORD   <1$-HEAD>
                    0E6D  2317
                    0E6D  2318
                    0E6D  2319  ;*****************************************************************************
                    0E6D  2320  ;
                    0E6D  2321  ;  READ_WCS_R
                    0E6D  2322  ;  THIS ROUTINE READS 3 BYTES OF DATA FROM THE WCS TO CSR_VALUE USING
                    0E6D  2323  ;  THE ADDRESS IN WCS_ADDRESS. ALSO A NOP IS PLACED IN THE CSR TO PREVENT
                    0E6D  2324  ;  THE MACHINE FROM DOING NASTY THINGS
                    0E6D  2325  ;
                    0E6D  2326  ;  INPUT CONDITIONS:
                    0E6D  2327  ;       WCS_ADDRESS HAS WCS ADDRESS TO READ
                    0E6D  2328  ;
                    0E6D  2329  ;  OUTPUT CONDITIONS:
                    0E6D  2330  ;       CSR_VALUE AND WCS_VALUE CONTAIN THE VALUE OF THE WORD READ
                    0E6D  2331  ;
                    0E6D  2332  ;*****************************************************************************
                    0E6D  2333
                    0E6D  2334  READ_WCS_R:     LHLD    WCS_ADDRESS                 ;GET ADDRESS TO READ
                2A  0E6D                .BYTE   ^052
             006C'  0E6E                .WORD   <WCS_ADDRESS-HEAD>
                    0E70  2335                  SHLD    UPC_VALUE
                22  0E70                .BYTE   ^042
             00C2'  0E71                .WORD   <UPC_VALUE-HEAD>
                    0E73  2336                  CALL    LOAD_UPC_R
                CD  0E73                .BYTE   ^0315
```

```
                     OF19' 0E74                 .WORD   <LOAD_UPC_R-HEAD>
                           0E76  2337
                           0E76  2338                   CALL    NOP_CSR_R
                       CD  0E76                 .BYTE   ^O315
                     OECA' 0E77                 .WORD   <NOP_CSR_R-HEAD>
                           0E79  2339
                           0E79  2340                   OUT     SETSS                   ;SING STEP MACHINE TO GET
                   23 D3   0E79                 .BYTE   ^O323 , SETSS&255       ;register A
                           0E7B  2341                   OUT     CLRSS                   ;WCS VALUE INTO CSR
                   22 D3   0E7B                 .BYTE   ^O323 , CLRSS&255       ;register A
                           0E7D  2342
                           0E7D  2343                   CALL    LOAD_CSR_R              ;READ IT OUT
                       CD  0E7D                 .BYTE   ^O315
                     OEE7' 0E7E                 .WORD   <LOAD_CSR_R-HEAD>
                           0E80  2344
                           0E80  2345                   LXI     H,CSR_VALUE
                       21  0E80                 .BYTE   <H*8> ! 1
                   00'B8'  0E81                 .BYTE   <CSR_VALUE-HEAD>&255 , <<CSR_VALUE-HEAD>&^XFF00>/256
                           0E83  2346                   LXI     D,WCS_VALUE
                       11  0E83                 .BYTE   <D*8> ! 1
                   00'B4'  0E84                 .BYTE   <WCS_VALUE-HEAD>&255 , <<WCS_VALUE-HEAD>&^XFF00>/256
                           0E86  2347                   CALL    MOVER_3_R
                       CD  0E86                 .BYTE   ^O315
                     OF5F' 0E87                 .WORD   <MOVER_3_R-HEAD>
                           0E89  2348
                           0E89  2349                   RET
                       C9  0E89                 .BYTE   ^O311
                           0E8A  2350
                           0E8A  2351 ;*****************************************************************************
                           0E8A  2352 ;
                           0E8A  2353 ;  WRITE_WCS_R
                           0E8A  2354 ;  THIS ROUTINE WRITES 3 BYTES OF DATA TO THE WCS FROM WCS_VALUE
                           0E8A  2355 ;  BEFORE THE WRITE OCCURES PARITY IS CALCULATED ON THAT 3 BYTE WCS WORD
                           0E8A  2356 ;  AND PLACED IN BIT 23.
                           0E8A  2357 ;
                           0E8A  2358 ;  INPUT CONDITIONS:
                           0E8A  2359 ;       WCS_ADDRESS HAS WCS ADDRESS TO READ
                           0E8A  2360 ;       WCS_BAD_PARITY = 0 FOR GOOD PARITY, 1 FOR BAD PARITY
                           0E8A  2361 ;       PARITY_ON = 0 FOR NO PARITY CALC, 1 TO CALCULATE PARITY
                           0E8A  2362 ;
                           0E8A  2363 ;*****************************************************************************
                           0E8A  2364
                           0E8A  2365 WRITE_WCS_R:    LHLD    WCS_ADDRESS             ;GET ADDRESS TO WRITE
                       2A  0E8A                 .BYTE   ^O52
                     006C' 0E8B                 .WORD   <WCS_ADDRESS-HEAD>
                           0E8D  2366                   SHLD    UPC_VALUE
                       22  0E8D                 .BYTE   ^O42
                     00C2' 0E8E                 .WORD   <UPC_VALUE-HEAD>
                           0E90  2367                   CALL    LOAD_UPC_R
                       CD  0E90                 .BYTE   ^O315
                     OF19' 0E91                 .WORD   <LOAD_UPC_R-HEAD>
                           0E93  2368
                           0E93  2369                   CALL    NOP_CSR_R               ;KEEP MACHINE IDLE.
                       CD  0E93                 .BYTE   ^O315
                     OECA' 0E94                 .WORD   <NOP_CSR_R-HEAD>
                           0E96  2370
                           0E96  2371                   SET     WCS_CALC                ;PARITY CALC ON WCS
```

```
                AF  0E96                .BYTE   ^0250 ! A                       ;with A
                3C  0E97                .BYTE   ^04 ! <8*A>
                32  0E98                .BYTE   ^062
            058A'   0E99                .WORD   <WCS_CALC-HEAD>
                    0E9B  2372                  LXI     H,WCS_VALUE             ;CALC PARITY ON WCS WORD
                21  0E9B                .BYTE   <H*8> ! 1
            00'B4'  0E9C                .BYTE   <WCS_VALUE-HEAD>&255 , <<WCS_VALUE-HEAD>&^XFF00>/256
                    0E9E  2373                  CALL    PARITY_CALC
                CD  0E9E                .BYTE   ^0315
            0BC8'   0E9F                .WORD   <PARITY_CALC-HEAD>
                    0EA1  2374                  CLEAR   WCS_CALC                ;CLEAR FLAG
                AF  0EA1                .BYTE   ^0250 ! A                       ;with A
                32  0EA2                .BYTE   ^062
            058A'   0EA3                .WORD   <WCS_CALC-HEAD>
                    0EA5  2375
                    0EA5  2376                  LXI     H,WCS_VALUE+2
                21  0EA5                .BYTE   <H*8> ! 1
            00'B6'  0EA6                .BYTE   <WCS_VALUE+2-HEAD>&255 , <<WCS_VALUE+2-HEAD>&^XFF00>/256
                    0EA8  2377                  MOV     A,M                     ;GET LOW BYTE
                7E  0EA8                .BYTE   ^0100 ! <A*8> ! M
                    0EA9  2378
                    0EA9  2379                  OUT     WCSB0                   ;CLOCK INTO BYTE 0 OF LATCH
            08 D3   0EA9                .BYTE   ^0323 , WCSB0&255       ;register A
                    0EAB  2380                  DCX     H                       ;INC POINTER TO NEXT BYTE
                2B  0EAB                .BYTE   <H*8> ! 11
                    0EAC  2381
                    0EAC  2382                  MOV     A,M                     ;GET MIDDLE BYTE
                7E  0EAC                .BYTE   ^0100 ! <A*8> ! M
                    0EAD  2383                  OUT     WCSB1                   ;CLOCK INTO BYTE 1 OF LATCH
            09 D3   0EAD                .BYTE   ^0323 , WCSB1&255       ;register A
                    0EAF  2384                  DCX     H                       ;INC POINTER TO NEXT BYTE
                2B  0EAF                .BYTE   <H*8> ! 11
                    0EB0  2385
                    0EB0  2386                  MOV     A,M                     ;GET HIGH BYTE
                7E  0EB0                .BYTE   ^0100 ! <A*8> ! M
                    0EB1  2387                  OUT     WCSB2                   ;CLOCK INTO BYTE 2 OF LATCH
            0A D3   0EB1                .BYTE   ^0323 , WCSB2&255       ;register A
                    0EB3  2388
                    0EB3  2389                  OUT     SETWCK                  ;CLOCK 24 BITS INTO WCS
            37 D3   0EB3                .BYTE   ^0323 , SETWCK&255      ;register A
                    0EB5  2390                  OUT     CLRWCK                  ;CLEAR WCS CLOCK
            36 D3   0EB5                .BYTE   ^0323 , CLRWCK&255      ;register A
                    0EB7  2391                  RET
                C9  0EB7                .BYTE   ^0311
                    0EB8  2392
                    0EB8  2393
                    0EB8  2394 ;*****************************************************************************
                    0EB8  2395 ;
                    0EB8  2396 ;   SHIFTER_R
                    0EB8  2397 ;   THIS ROUTINE SHIFTS N BYTES OF DATA LEFT 1 BIT AT THE ADDRESS POINTED
                    0EB8  2398 ;   TO BY  SHIFT_PNT.
                    0EB8  2399 ;
                    0EB8  2400 ;   INPUT CONDITIONS:
                    0EB8  2401 ;        SHIFT_PNT = POINTS TO COUNT OF NUMBER OF BYTES TO SHIFT
                    0EB8  2402 ;                    DATA TO BE SHIFTED FOLLOWS IN NEXT N BYTES
                    0EB8  2403 ;
                    0EB8  2404 ;   CARRY BIT GETS MSB, CARRY MAY ALSO BE USED AS
```

```
                     0EB8  2405 ;  INPUT TO THE SHIFT
                     0EB8  2406 ;
                     0EB8  2407 ;*****************************************************************************
                     0EB8  2408
                     0EB8  2409 SHIFTER_R:       PUSH    $PSW            ;SAVE CARRY
                  F5 0EB8                .BYTE   ^0305 ! <8*$PSW>
                     0EB9  2410                 LHLD    SHIFT_PNT       ;POINTER TO SHIFT DATA
                  2A 0EB9                .BYTE   ^052
              0068' 0EBA                .WORD   <SHIFT_PNT-HEAD>
                     0EBC  2411                 MOV     C,M             ;GET COUNT IN BYTES
                  4E 0EBC                .BYTE   ^0100 ! <C*8> ! M
                     0EBD  2412                 MVI     B,0             ;MAKE COUNT IN REG PAIR
                00 06 0EBD                .BYTE   <B*8> ! 6 , 0&255
                     0EBF  2413                 DAD     B               ;ADD COUNT , GOTO LAST BYTE
                  09 0EBF                .BYTE   <B*8> ! 9
                     0EC0  2414                 POP     $PSW
                  F1 0EC0                .BYTE   ^0301 ! <8*$PSW>
                     0EC1  2415
                     0EC1  2416 SHIFTER1_R:      MOV     A,M             ;MOVE BYTE 1 LEFT
                  7E 0EC1                .BYTE   ^0100 ! <A*8> ! M
                     0EC2  2417                 RAL                     ;CARRY = ROTATE IN BIT NEXT BTYE
                  17 0EC2                .BYTE   ^027
                     0EC3  2418                 MOV     M,A
                  77 0EC3                .BYTE   ^0100 ! <M*8> ! A
                     0EC4  2419                 DCR     C               ;DEC BYTE COUNT
                  0D 0EC4                .BYTE   ^05 ! <8*C>
                     0EC5  2420                 RZ                      ;RETURN IF ZERO
                  C8 0EC5                .BYTE   ^0310
                     0EC6  2421                 DCX     H               ;MOVE POINTER TO NEXT BYTE
                  2B 0EC6                .BYTE   <H*8> ! 11
                     0EC7  2422
                     0EC7  2423                 JMP     SHIFTER1_R      ;REPEAT UNTIL BYTE COUNT=0
                  C3 0EC7                .BYTE   ^0303
              0EC1' 0EC8                .WORD   <SHIFTER1_R-HEAD>
                     0ECA  2424
                     0ECA  2425 ;*****************************************************************************
                     0ECA  2426 ;
                     0ECA  2427 ;  NOP_CSR_R:
                     0ECA  2428 ;  THIS ROUTINE LOAD A NOP INTO THE CSR
                     0ECA  2429 ;
                     0ECA  2430 ;*****************************************************************************
                     0ECA  2431
                     0ECA  2432 NOP_CSR_R:       LXI     H,NOP_INST
                  21 0ECA                .BYTE   <H*8> ! 1
              00'72' 0ECB                .BYTE   <NOP_INST-HEAD>&255 , <<NOP_INST-HEAD>&^XFF00>/256
                     0ECD  2433                 LXI     D,CSR_VALUE
                  11 0ECD                .BYTE   <D*8> ! 1
              00'B8' 0ECE                .BYTE   <CSR_VALUE-HEAD>&255 , <<CSR_VALUE-HEAD>&^XFF00>/256
                     0ED0  2434                 CALL    MOVER_3_R               ;MOVE NOP TO CSR SAVE AREA
                  CD 0ED0                .BYTE   ^0315
              0F5F' 0ED1                .WORD   <MOVER_3_R-HEAD>
                     0ED3  2435                 LDA     PARITY_ON               ;SAVE PARITY ON FLAG
                  3A 0ED3                .BYTE   ^072
              0064' 0ED4                .WORD   <PARITY_ON-HEAD>
                     0ED6  2436                 STA     TEMP_BYTE               ;STORE IN TEMP BYTE
                  32 0ED6                .BYTE   ^062
              0418' 0ED7                .WORD   <TEMP_BYTE-HEAD>
```

```
                     0ED9  2437                    CLEAR   PARITY_ON               ;CALC NO PARITY ON NOP
          AF         0ED9                .BYTE    ^0250 ! A              ;with A
          32         0EDA                .BYTE    ^062
       0064'         0EDB                .WORD    <PARITY_ON-HEAD>
                     0EDD  2438
                     0EDD  2439                    CALL    LOAD_CSR_R              ;LOAD IT
          CD         0EDD                .BYTE    ^0315
       0EE7'         0EDE                .WORD    <LOAD_CSR_R-HEAD>
                     0EE0  2440
                     0EE0  2441                    LDA     TEMP_BYTE               ;GET OLD PARITY_ON FLAG
          3A         0EE0                .BYTE    ^072
       0418'         0EE1                .WORD    <TEMP_BYTE-HEAD>
                     0EE3  2442                    STA     PARITY_ON               ;RESTORE FLAG
          32         0EE3                .BYTE    ^062
       0064'         0EE4                .WORD    <PARITY_ON-HEAD>
                     0EE6  2443
                     0EE6  2444                    RET
          C9         0EE6                .BYTE    ^0311
                     0EE7  2445
                     0EE7  2446
                     0EE7  2447 ;*****************************************************************************
                     0EE7  2448 ;
                     0EE7  2449 ;  LOAD_CSR_R
                     0EE7  2450 ;  THIS ROUTINE LOADS THE CSR FROM THE CSR_VALUE AREA,
                     0EE7  2451 ;  ALSO CSR_VALUE AREA GETS THE VALUE OF THE CSR
                     0EE7  2452 ;
                     0EE7  2453 ;*****************************************************************************
                     0EE7  2454
                     0EE7  2455 LOAD_CSR_R:       OUT     VSHIFT                  ;SET V-BUS SHIFT MODE
       A2 D3         0EE7                .BYTE    ^0323 , VSHIFT&255      ;register A
                     0EE9  2456
                     0EE9  2457                    LXI     H,CSR_VALUE             ;CALC PARITY ON NEW CSR VALUE
          21         0EE9                .BYTE    <H*8> ! 1
    00'B8'           0EEA                .BYTE    <CSR_VALUE-HEAD>&255 , <<CSR_VALUE-HEAD>&^XFF00>/256
                     0EEC  2458                    CALL    PARITY_CALC
          CD         0EEC                .BYTE    ^0315
       0BC8'         0EED                .WORD    <PARITY_CALC-HEAD>
                     0EEF  2459
                     0EEF  2460                    DO      24                      ;SHIFT 24 BITS
          21         0EEF                .BYTE    <H*8> ! 1
    00'55'           0EF0                .BYTE    <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
          46         0EF2                .BYTE    ^0100 ! <B*8> ! M
          C5         0EF3                .BYTE    ^0305 ! <8*8>
       18 3E         0EF4                .BYTE    <A*8> ! 6 , 24&255
          77         0EF6                .BYTE    ^0100 ! <M*8> ! A
                     0EF7           2035$:
                     0EF7  2461
                     0EF7  2462                    IN      CSR23                   ;GET OLD CSR DATA
       87 DB         0EF7                .BYTE    ^0333 , CSR23&255       ;to register A
                     0EF9  2463                    RAR                             ;PUT CSR 23 IN CARRY
          1F         0EF9                .BYTE    ^037
                     0EFA  2464                    LXI     H,CSR_VALUE+2           ;POINTER TO LAST CSR DATA BYTE
          21         0EFA                .BYTE    <H*8> ! 1
    00'BA'           0EFB                .BYTE    <CSR_VALUE+2-HEAD>&255 , <<CSR_VALUE+2-HEAD>&^XFF00>/256
                     0EFD  2465                    MVI     C,3                     ;BYTES TO SHIFT
       03 0E         0EFD                .BYTE    <C*8> ! 6 , 3&255
                     0EFF  2466                    CALL    SHIFTER1_R              ;CARRY = MSB OF NEW CSR DATA ON
```

```
                CD  0EFF                .BYTE   ^0315
 0EC1'          0F00                    .WORD   <SHIFTER1_R-HEAD>
                0F02  2467                                              ; RETURN
                0F02  2468
                0F02  2469                      OUT     CLRCSR          ;CLEAR CSR INPUT
             38 D3  0F02                .BYTE   ^0323 , CLRCSR&255      ;register A
                0F04  2470                      JNC     1$              ;DON'T SET INPUT IF CARRY CLEAR
                D2  0F04                .BYTE   ^0322
 0F09'          0F05                    .WORD   <1$-HEAD>
                0F07  2471                      OUT     SETCSR          ;ELSE SET CSR INPUT
             39 D3  0F07                .BYTE   ^0323 , SETCSR&255      ;register A
                0F09  2472
                0F09  2473 1$:                  OUT     SETCSK          ;SET UPC CLOCK
             25 D3  0F09                .BYTE   ^0323 , SETCSK&255      ;register A
                0F0B  2474                      OUT     CLRCSK          ;CLEAR UPC CLOCK
             24 D3  0F0B                .BYTE   ^0323 , CLRCSK&255      ;register A
                0F0D  2475
                0F0D  2476                      ENDDO
                21  0F0D                .BYTE   <H*8> ! 1
 00'55'         0F0E                    .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
                35  0F10                .BYTE   ^05 ! <8*M>
                C2  0F11                .BYTE   ^0302
 0EF7'          0F12                    .WORD   <2035$-HEAD>
                C1  0F14                .BYTE   ^0301 ! <8*B>
                70  0F15                .BYTE   ^0100 ! <M*8> ! B
                0F16  2477
                0F16  2478                      OUT     VNORML          ;SET V-BUS TO NORMAL MODE
             A3 D3  0F16                .BYTE   ^0323 , VNORML&255      ;register A
                0F18  2479
                0F18  2480                      RET
                C9  0F18                .BYTE   ^0311
                0F19  2481
                0F19  2482
                0F19  2483 ;*****************************************************************************
                0F19  2484 ;
                0F19  2485 ;  LOAD_UPC_R
                0F19  2486 ;  THIS ROUTINE LOADS THE UPC FROM THE UPC_VALUE AREA,
                0F19  2487 ;  ALSO UPC_VALUE AREA GETS THE VALUE OF THE UPC
                0F19  2488 ;
                0F19  2489 ;*****************************************************************************
                0F19  2490
                0F19  2491 LOAD_UPC_R:          OUT     VSHIFT          ;SET V-BUS SHIFT MODE
             A2 D3  0F19                .BYTE   ^0323 , VSHIFT&255      ;register A
                0F1B  2492
                0F1B  2493                      LXI     H,UPC_SHIFT     ;POINTER TO UPC DATA BUFF
                21  0F1B                .BYTE   <H*8> ! 1
 00'C1'         0F1C                    .BYTE   <UPC_SHIFT-HEAD>&255 , <<UPC_SHIFT-HEAD>&^XFF00>/256
                0F1E  2494                      SHLD    SHIFT_PNT
                22  0F1E                .BYTE   ^042
  0068'         0F1F                    .WORD   <SHIFT_PNT-HEAD>
                0F21  2495
                0F21  2496                      ORA     A               ;CLEAR CARRY
                B7  0F21                .BYTE   ^0260 ! A
                0F22  2497
                0F22  2498                      CALL    SHIFTER_R       ;LEFT JUST. 15 BITS
                CD  0F22                .BYTE   ^0315
 0E88'          0F23                    .WORD   <SHIFTER_R-HEAD>
```

```
                        0F25  2499
                        0F25  2500                      DO      15                      ;SHIFT 15 BITS
                  21    0F25                    .BYTE   <H*8> ! 1
            00'55'      0F26                    .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
                  46    0F28                    .BYTE   ^0100 ! <B*8> ! M
                  C5    0F29                    .BYTE   ^0305 ! <8*B>
               0F 3E    0F2A                    .BYTE   <A*8> ! 6 , 15&255
                  77    0F2C                    .BYTE   ^0100 ! <M*8> ! A
                        0F2D            2036$:
                        0F2D  2501
                        0F2D  2502                      IN      UPC14
               84 DB    0F2D                    .BYTE   ^0333 , UPC14&255       ;to register A
                        0F2F  2503                      RAR                             ;PUT UPC14 IN CARRY
                  1F    0F2F                    .BYTE   ^037
                        0F30  2504                      CALL    SHIFTER_R               ;CARRY = MSB ON RET
                  CD    0F30                    .BYTE   ^0315
             0EB8'      0F31                    .WORD   <SHIFTER_R-HEAD>
                        0F33  2505
                        0F33  2506                      OUT     CLRUPC                  ;SET = 0 UPC SER IN
               A8 D3    0F33                    .BYTE   ^0323 , CLRUPC&255      ;register A
                        0F35  2507                      IFC
                  D2    0F35                    .BYTE   ^0322
             0F3A'      0F36                    .WORD   <2037$-HEAD>
                        0F38  2508                      OUT     SETUPC                  ;SET = 1 UPC SER IN
               A9 D3    0F38                    .BYTE   ^0323 , SETUPC&255      ;register A
                        0F3A  2509                      ENDIF
                        0F3A            2037$:                                  ;GENERATE LCL 2037 NAME
                        0F3A  2510
                        0F3A  2511                      OUT     SETUPK                  ;SET UPC CLOCK
               27 D3    0F3A                    .BYTE   ^0323 , SETUPK&255      ;register A
                        0F3C  2512                      OUT     CLRUPK                  ;CLEAR UPC CLOCK
               26 D3    0F3C                    .BYTE   ^0323 , CLRUPK&255      ;register A
                        0F3E  2513
                        0F3E  2514                      ENDDO
                  21    0F3E                    .BYTE   <H*8> ! 1
            00'55'      0F3F                    .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
                  35    0F41                    .BYTE   ^05 ! <8*M>
                  C2    0F42                    .BYTE   ^0302
             0F2D'      0F43                    .WORD   <2036$-HEAD>
                  C1    0F45                    .BYTE   ^0301 ! <8*B>
                  70    0F46                    .BYTE   ^0100 ! <M*8> ! B
                        0F47  2515
                        0F47  2516                      OUT     VNORML                  ;SET V-BUS TO NORMAL MODE
               A3 D3    0F47                    .BYTE   ^0323 , VNORML&255      ;register A
                        0F49  2517
                        0F49  2518                      OUT     CLRACK                  ;CLEAR CONSOLE ACKNOWLEDGE
               A9 D3    0F49                    .BYTE   ^0323 , CLRACK&255      ;register A
                        0F4B  2519                      IF      SETACK_FLG              ;MUST SET UP CONSOLE ACKNOWLEDE
                  3A    0F4B                    .BYTE   ^072
             0405'      0F4C                    .WORD   <SETACK_FLG-HEAD>
                  0F    0F4E                    .BYTE   ^017
                  D2    0F4F                    .BYTE   ^0322
             0F54'      0F50                    .WORD   <2038$-HEAD>
                        0F52  2520                                              ;BEFORE RETURNING
                        0F52  2521                      OUT     SETACK                  ;SET CONSOLE ACKNOWLEDGE IF
               A8 D3    0F52                    .BYTE   ^0323 , SETACK&255      ;register A
                        0F54  2522                                              ; FLAG SET
```

```
                        0F54  2523                ENDIF
                        0F54        2038$:                                  ;GENERATE LCL 2038 NAME
                        0F54  2524
                        0F54  2525                RET
                     C9 0F54                .BYTE   ^0311
                        0F55  2526
                        0F55  2527
                        0F55  2528 ;*****************************************************************************
                        0F55  2529 ;
                        0F55  2530 ;  COMPARE_R
                        0F55  2531 ;  THIS ROUTINE IS USED BY THE STRUCTURED MACRO SET AND MAY ALSO
                        0F55  2532 ;  BE CALLED DIRECTLY.
                        0F55  2533 ;
                        0F55  2534 ;  INPUT CONDITIONS:
                        0F55  2535 ;       H + L    = POINTER TO DATA FIELD 1
                        0F55  2536 ;       D + E    = POINTER TO DATA FIELD 2
                        0F55  2537 ;       C        = COUNT IN BYTES FOR COMPARISON
                        0F55  2538 ;
                        0F55  2539 ;  OUTPUT CONDITIONS:
                        0F55  2540 ;       CONDITIONS CODES ARE SET
                        0F55  2541 ;
                        0F55  2542 ;*****************************************************************************
                        0F55  2543
                        0F55  2544 COMPARE_R:       LDAX    D               ;FIELD 2 BYTE
                     1A 0F55                .BYTE   <D*8> ! 10
                        0F56  2545                CMP     M               ;COMPARED TO FIELD 1 BYTE
                     BE 0F56                .BYTE   ^0270 ! M
                        0F57  2546                RNZ                     ;RETURN WITH CC NON ZERO IF NOT SAME
                     C0 0F57                .BYTE   ^0300
                        0F58  2547                INX     H               ;INCREMENT FIELD 1 POINTER
                     23 0F58                .BYTE   <H*8> ! 3
                        0F59  2548                INX     D               ;INCREMENT FIELD 2 POINTER
                     13 0F59                .BYTE   <D*8> ! 3
                        0F5A  2549                DCR     C               ;DECREMENT COUNT
                     0D 0F5A                .BYTE   ^05 ! <8*C>
                        0F5B  2550                JNZ     COMPARE_R       ;LOOP UNTIL COUNT GONE
                     C2 0F5B                .BYTE   ^0302
                0F55' 0F5C                .WORD   <COMPARE_R-HEAD>
                        0F5E  2551                RET                     ;DONE - RETURN ZERO CC
                     C9 0F5E                .BYTE   ^0311
                        0F5F  2552
                        0F5F  2553
                        0F5F  2554 ;*****************************************************************************
                        0F5F  2555 ;
                        0F5F  2556 ;  MOVER_R
                        0F5F  2557 ;
                        0F5F  2558 ;  THIS ROUTINE MOVES DATA FROM THE ADDRESS GIVEN IN H+L TO THE ADDRESS
                        0F5F  2559 ;  GIVEN IN D+E FOR THE COUNT IN C
                        0F5F  2560 ;
                        0F5F  2561 ;  INPUT CONDITIONS:
                        0F5F  2562 ;       H+L      = SOURCE ADDRESS
                        0F5F  2563 ;       D+E      = DEST. ADDRESS
                        0F5F  2564 ;       C        = LENGTH
                        0F5F  2565 ;
                        0F5F  2566 ;  OUTPUT CONDITIONS:
                        0F5F  2567 ;       REGISTERS ARE INCREMENTED DURING THE MOVE, SO THEY
                        0F5F  2568 ;       ARE POINTING TO THE LOCATION AFTER THE DATA
```

```
                     0F5F  2569 ;
                     0F5F  2570 ;*****************************************************************************
                     0F5F  2571
                     0F5F  2572 MOVER_3_R:      MVI     C,3                     ;MOVE 3 TO COUNT
03 0E                0F5F                       .BYTE   <C*8> ! 6 , 3&255
                     0F61  2573                 JMP     MOVER_R                 ;GO MOVE THREE BYTES
   C3                0F61                       .BYTE   ^0303
 0F75'               0F62                       .WORD   <MOVER_R-HEAD>
                     0F64  2574
                     0F64  2575 MOV_INPUT_DATA0:
                     0F64  2576                 LXI     H,INPUT_NUM             ;SOURCE IN INPUT_NUM
   21                0F64                       .BYTE   <H*8> ! 1
06'76'               0F65                       .BYTE   <INPUT_NUM-HEAD>&255 , <<INPUT_NUM-HEAD>&^XFF00>/256
                     0F67  2577                 LXI     D,DATA0                 ;DEST IS DATA0-DATA3
   11                0F67                       .BYTE   <D*8> ! 1
00'A9'               0F68                       .BYTE   <DATA0-HEAD>&255 , <<DATA0-HEAD>&^XFF00>/256
                     0F6A  2578                 JMP     MOVER_4_R               ;MOVE 4 BYTES
   C3                0F6A                       .BYTE   ^0303
 0F73'               0F6B                       .WORD   <MOVER_4_R-HEAD>
                     0F6D  2579
                     0F6D  2580 MOV_DATA0_HEXBUF:
                     0F6D  2581                 LXI     D,HEX_BUF               ;DESTINATION IS HEX_BUF
   11                0F6D                       .BYTE   <D*8> ! 1
00'5B'               0F6E                       .BYTE   <HEX_BUF-HEAD>&255 , <<HEX_BUF-HEAD>&^XFF00>/256
                     0F70  2582 MOV_FROM_DATA0: LXI     H,DATA0                 ;SOURCE IS DATA0-3
   21                0F70                       .BYTE   <H*8> ! 1
00'A9'               0F71                       .BYTE   <DATA0-HEAD>&255 , <<DATA0-HEAD>&^XFF00>/256
                     0F73  2583
                     0F73  2584 MOVER_4_R:      MVI     C,4                     ;MOVE 4 TO COUNT
04 0E                0F73                       .BYTE   <C*8> ! 6 , 4&255
                     0F75  2585 MOVER_R:        MOV     A,M                     ;GET SOURCE DATA
   7E                0F75                       .BYTE   ^0100 ! <A*8> ! M
                     0F76  2586                 STAX    D                       ;PUT AT DESTINATION
   12                0F76                       .BYTE   <D*8> ! 2
                     0F77  2587                 INX     H
   23                0F77                       .BYTE   <H*8> ! 3
                     0F78  2588                 INX     D                       ;INCREMENT POINTERS
   13                0F78                       .BYTE   <D*8> ! 3
                     0F79  2589                 DCR     C                       ;DECREMENT COUNT
   0D                0F79                       .BYTE   ^05 ! <8*C>
                     0F7A  2590                 RZ                              ;RETURN IF DONE
   C8                0F7A                       .BYTE   ^0310
                     0F7B  2591                 JMP     MOVER_R                 ;LOOP OTHERWISE
   C3                0F7B                       .BYTE   ^0303
 0F75'               0F7C                       .WORD   <MOVER_R-HEAD>
                     0F7E  2592
                     0F7E  2593
                     0F7E  2594
                     0F7E  2595 ;*****************************************************************************
                     0F7E  2596 ;
                     0F7E  2597 ;  PRINT_FLG
                     0F7E  2598 ;  THIS ROUTINE PRINTS 5 CHARS AND A SPACE FROM THE POINTER TEMP_WORD
                     0F7E  2599 ;
                     0F7E  2600 ;*****************************************************************************
                     0F7E  2601
                     0F7E  2602 PRINT_FLG:      LHLD    TEMP_WORD               ;POINTER TO FLAG TO PRINT
   2A                0F7E                       .BYTE   ^052
```

```
                0419' 0F7F                      .WORD   <TEMP_WORD-HEAD>
                      0F81  2603                MVI     C,5
             05 0E    0F81                      .BYTE   <C*8> ! 6 , 5&255
                      0F83  2604                CALL    PRINTER                 ;PRINT THE 5 CHAR
                CD    0F83                      .BYTE   ^0315
                1007' 0F84                      .WORD   <PRINTER-HEAD>
                      0F86  2605                CALL    SPACE_R                 ;AND THE SPACE
                CD    0F86                      .BYTE   ^0315
                1021' 0F87                      .WORD   <SPACE_R-HEAD>
                      0F89  2606
                      0F89  2607                RET
                C9    0F89                      .BYTE   ^0311
                      0F8A  2608
                      0F8A  2609
                      0F8A  2610 ;*****************************************************************************
                      0F8A  2611 ;
                      0F8A  2612 ;  PRINT_HEX_R
                      0F8A  2613 ;  THIS ROUTINE PRINTS HEX NUMBERS
                      0F8A  2614 ;
                      0F8A  2615 ;  INPUT CONDITIONS:
                      0F8A  2616 ;       C = NUMBER OF BYTES TO BE PRINTED
                      0F8A  2617 ;       DATA IS IN HEX_BUF
                      0F8A  2618 ;
                      0F8A  2619 ;*****************************************************************************
                      0F8A  2620
                      0F8A  2621 PRINT_HEX_4:   MVI     C,4                     ;LOAD BYTE COUNT WITH 4
             04 0E    0F8A                      .BYTE   <C*8> ! 6 , 4&255
                      0F8C  2622                JMP     PRINT_HEX_R             ;PRINT FOUR CHARS
                C3    0F8C                      .BYTE   ^0303
                0F96' 0F8D                      .WORD   <PRINT_HEX_R-HEAD>
                      0F8F  2623
                      0F8F  2624 PRINT_HEX_2:   MVI     C,2                     ,BYTE COUNT OF 2
             02 0E    0F8F                      .BYTE   <C*8> ! 6 , 2&255
                      0F91  2625                JMP     PRINT_HEX_R             ;PRINT 2 CHARS
                C3    0F91                      .BYTE   ^0303
                0F96' 0F92                      .WORD   <PRINT_HEX_R-HEAD>
                      0F94  2626
                      0F94  2627 PRINT_HEX_1_R: MVI     C,1                     ;PRINT 1 CHAR
             01 0E    0F94                      .BYTE   <C*8> ! 6 , 1&255
                      0F96  2628 PRINT_HEX_R:   LXI     H,HEX_BUF
                21    0F96                      .BYTE   <H*8> ! 1
             00'5B'   0F97                      .BYTE   <HEX_BUF-HEAD>&255 , <<HEX_BUF-HEAD>&^XFF00>/256
                      0F99  2629                SHLD    PRINT_PNT               ;POINTER TO PRINT DATA
                22    0F99                      .BYTE   ^042
                03EE' 0F9A                      .WORD   <PRINT_PNT-HEAD>
                      0F9C  2630                MOV     A,C
                79    0F9C                      .BYTE   ^0100 ! <A*8> ! C
                      0F9D  2631                STA     PRINT_CNT               ;PRINT COUNT
                32    0F9D                      .BYTE   ^062
                03EC' 0F9E                      .WORD   <PRINT_CNT-HEAD>
                      0FA0  2632
                      0FA0  2633 1$:            LHLD    PRINT_PNT
                2A    0FA0                      .BYTE   ^052
                03EE' 0FA1                      .WORD   <PRINT_PNT-HEAD>
                      0FA3  2634                MOV     A,M                     ;GET FIRST BYTE
                7E    0FA3                      .BYTE   ^0100 ! <A*8> ! M
                      0FA4  2635                RAR
```

```
                     1F  0FA4                .BYTE   ^037
                         0FA5  2636                  RAR
                     1F  0FA5                .BYTE   ^037
                         0FA6  2637                  RAR
                     1F  0FA6                .BYTE   ^037
                         0FA7  2638                  RAR
                     1F  0FA7                .BYTE   ^037
                         0FA8  2639                  ANI     <^XF>                   ;GET FIRST NIBBLE
                     E6  0FA8                .BYTE   ^0346
                     0F  0FA9                .BYTE   ^XF
                         0FAA  2640                  CALL    PR_HEX_DIGIT            ;PRINT HEX DIGIT
                     CD  0FAA                .BYTE   ^0315
                0FC9'    0FAB                .WORD   <PR_HEX_DIGIT-HEAD>
                         0FAD  2641
                         0FAD  2642                  LHLD    PRINT_PNT
                     2A  0FAD                .BYTE   ^052
                03EE'    0FAE                .WORD   <PRINT_PNT-HEAD>
                         0FB0  2643                  MOV     A,M
                     7E  0FB0                .BYTE   ^0100 ! <A*8> ! M
                         0FB1  2644                  INX     H
                     23  0FB1                .BYTE   <H*8> ! 3
                         0FB2  2645                  SHLD    PRINT_PNT
                     22  0FB2                .BYTE   ^042
                03EE'    0FB3                .WORD   <PRINT_PNT-HEAD>
                         0FB5  2646                  ANI     <^XF>                   ;GET SECOND NIBBLE
                     E6  0FB5                .BYTE   ^0346
                     0F  0FB6                .BYTE   ^XF
                         0FB7  2647                  CALL    PR_HEX_DIGIT
                     CD  0FB7                .BYTE   ^0315
                0FC9'    0FB8                .WORD   <PR_HEX_DIGIT-HEAD>
                         0FBA  2648
                         0FBA  2649                  LXI     H,PRINT_CNT             ;DECR PRINT COUNT
                     21  0FBA                .BYTE   <H*8> ! 1
               03'EC'    0FBB                .BYTE   <PRINT_CNT-HEAD>&255 , <<PRINT_CNT-HEAD>&^XFF00>/256
                         0FBD  2650                  DCR     M
                     35  0FBD                .BYTE   ^05 ! <8*M>
                         0FBE  2651                  JNZ     1$                      ;RETURN IF DONE
                     C2  0FBE                .BYTE   ^0302
                0FA0'    0FBF                .WORD   <1$-HEAD>
                         0FC1  2652
                         0FC1  2653                  LDA     NO_SPACE                ;GET FLAG
                     3A  0FC1                .BYTE   ^072
                03C1'    0FC2                .WORD   <NO_SPACE-HEAD>
                         0FC4  2654                  ORA     A                       ;SET CONDITION CODES
                     B7  0FC4                .BYTE   ^0260 ! A
                         0FC5  2655                  CZ      SPACE_R                 ;PRINT SPACE IF NO FLAG
                     CC  0FC5                .BYTE   ^0314
                1021'    0FC6                .WORD   <SPACE_R-HEAD>
                         0FC8  2656                  RET
                     C9  0FC8                .BYTE   ^0311
                         0FC9  2657
                         0FC9  2658
                         0FC9  2659 ;*****************************************************************************
                         0FC9  2660 ;
                         0FC9  2661 ;  PR_HEX_DIGIT
                         0FC9  2662 ;  THIS ROUTINE PRINTS A HEX DIGIT
                         0FC9  2663 ;  THE HEX NUMBER IS IN THE AC.
```

```
                        0FC9  2664 ;
                        0FC9  2665 ;*****************************************************************************
                        0FC9  2666
                        0FC9  2667 PR_HEX_DIGIT:    ADI     HEX_30                  ;ADD FOR 0-9 TO ASCII
                     C6 0FC9                .BYTE   ^0306
                     30 0FCA                .BYTE   HEX_30
                        0FCB  2668                  CPI     HEX_3A                  ;GREATER THAN 9 ?
                     FE 0FCB                .BYTE   ^0376
                     3A 0FCC                .BYTE   HEX_3A
                        0FCD  2669                  JM      1$
                     FA 0FCD                .BYTE   ^0372
                  0FD2' 0FCE                .WORD   <1$-HEAD>
                        0FD0  2670                  ADI     HEX_7                   ;ADD FOR A-F
                     C6 0FD0                .BYTE   ^0306
                     07 0FD1                .BYTE   HEX_7
                        0FD2  2671 1$:              CALL    TYPE_CHAR               ;PRINT ASCII CHAR
                     CD 0FD2                .BYTE   ^0315
                   1027' 0FD3                .WORD   <TYPE_CHAR-HEAD>
                        0FD5  2672                  RET
                     C9 0FD5                .BYTE   ^0311
                        0FD6  2673
                        0FD6  2674
                        0FD6  2675 ;*****************************************************************************
                        0FD6  2676 ;
                        0FD6  2677 ;  PRINT_PROMPT
                        0FD6  2678 ;
                        0FD6  2679 ;  THIS ROUTINE PRINTS THE PROMPT ON THE CONSOLE PORT
                        0FD6  2680 ;  THIS ROUTINE ALSO CALCULATES A CHECKSUM AND COMPARES IT TO THE CHECKSUM
                        0FD6  2681 ;  CALCULATED WHEN THE PROGRAM WAS STARTED. IF THEY ARE NOT THE SAME
                        0FD6  2682 ;  AN ERROR IS PRINTED
                        0FD6  2683 ;
                        0FD6  2684 ;*****************************************************************************
                        0FD6  2685
                        0FD6  2686 PRINT_PROMPT:    CLEAR   OFLAG                   ;CLEAR CONTROL O FLAG IN BOOT
                     AF 0FD6                .BYTE   ^0250 ! A               ;with A
                     32 0FD7                .BYTE   ^062
                   40BB 0FD8                .WORD   <OFLAG-HEAD>
                        0FDA  2687                                          ; BLOCK
                        0FDA  2688                  CLEAR   ERROR_CON       ;CLEAR 8085 ERROR LOOP FLAG
                     AF 0FDA                .BYTE   ^0250 ! A               ;with A
                     32 0FDB                .BYTE   ^062
                   0057' 0FDC                .WORD   <ERROR_CON-HEAD>
                        0FDE  2689
                        0FDE  2690                  IN      APTFLG                  ;READ APT PRESENT FLAG
                  04 DB 0FDE                .BYTE   ^0333 , APTFLG&255      ;to register A
                        0FE0  2691                  CMA                             ;INVERT VALUE (HIGH=SET NOW)
                     2F 0FE0                .BYTE   ^057
                        0FE1  2692                  MOV     B,A                     ;STORE IN B REG
                     47 0FE1                .BYTE   ^0100 ! <B*8> ! A
                        0FE2  2693                  IN      REMOTE                  ;GET REMOTE FLAG
                  06 DB 0FE2                .BYTE   ^0333 , REMOTE&255      ;to register A
                        0FE4  2694                  CMA                             ;INVERT VALUE
                     2F 0FE4                .BYTE   ^057
                        0FE5  2695                  ANA     B                       ;AND TWO FLAGS
                     A0 0FE5                .BYTE   ^0240 ! B
                        0FE6  2696                  ANI     1                       ;MASK ALL BUT BIT 0
                     E6 0FF6                .BYTE   ^0346
```

```
                    01  0FE7                       .BYTE   1
                        0FE8  2697                  STA     APT                         ;SAVE APT PRESENT FLAG
                    32  0FE8                       .BYTE   ^062
                0038'   0FE9                       .WORD   <APT-HEAD>
                        0FEB  2698
                        0FEB  2699                  CALL    CHECKSUM                    ;CALC CHECKSUM
                    CD  0FEB                       .BYTE   ^0315
                1BE9'   0FEC                       .WORD   <CHECKSUM-HEAD>
                        0FEE  2700
                        0FEE  2701                  IFNEQ   NEWSUM,OLDSUM,1
                    21  0FEE                       .BYTE   <H*8> ! 1
            03'BC'      0FEF                       .BYTE   <NEWSUM-HEAD>&255 , <<NEWSUM-HEAD>&^XFF00>/256
                    11  0FF1                       .BYTE   <D*8> ! 1
            03'CF'      0FF2                       .BYTE   <OLDSUM-HEAD>&255 , <<OLDSUM-HEAD>&^XFF00>/256
            01 0E       0FF4                       .BYTE   <C*8> ! 6 , 1&255
                    CD  0FF6                       .BYTE   ^0315
                00CF'   0FF7                       .WORD   <COMPARE-HEAD>
                    CA  0FF9                       .BYTE   ^0312
                1001'   0FFA                       .WORD   <2039$-HEAD>
                        0FFC  2702                  MVI     B,<^X27>
            27 06       0FFC                       .BYTE   <B*8> ! 6 , ^X27&255
                        0FFE  2703                  CALL    ERROR_ROUTINE               ;CHECKSUM ERROR IN MICMON
                    CD  0FFE                       .BYTE   ^0315
                0C4A'   0FFF                       .WORD   <ERROR_ROUTINE-HEAD>
                        1001  2704                  ENDIF
                        1001        2039$:                                              ;GENERATE LCL 2039 NAME
                        1001  2705
                        1001  2706                  CLEAR   CNT_PAR                     ;CLEAR PARITY ERROR COUNT
                    AF  1001                       .BYTE   ^0250 ! A                   ;with A
                    32  1002                       .BYTE   ^062
                0254'   1003                       .WORD   <CNT_PAR-HEAD>
                        1005  2707
                        1005  2708                  EI                                  ;ENABLE INTERUPTS
                    FB  1005                       .BYTE   ^0373
                        1006  2709
                        1006  2710                  RET
                    C9  1006                       .BYTE   ^0311
                        1007  2711
                        1007  2712
                        1007  2713 ;*****************************************************************************
                        1007  2714 ;
                        1007  2715 ;  PRINTER
                        1007  2716 ;  THIS ROUTINE PRINTS A STRING OF CHARS
                        1007  2717 ;  INPUT CONDITIONS
                        1007  2718 ;       H+L = POINTER TO THE STRING
                        1007  2719 ;       C   = NUMBER OF CHARS IN THE STRING
                        1007  2720 ;
                        1007  2721 ;*****************************************************************************
                        1007  2722
                        1007  2723 PRINTER:          MOV     A,C                         ;LD A WITH CHAR COUNT
                    79  1007                       .BYTE   ^0100 ! <A*8> ! C
                        1008  2724                  CALL    SLVEC1                      ;CALL SEND LINE ROUTINE IN BOOT
                    CD  1008                       .BYTE   ^0315
                4133    1009                       .WORD   <SLVEC1-HEAD>
                        1008  2725                                                      ; BLOCK.
                        1008  2726                  JMP     CHECK_CNTLC                 ;IF ^C TYPED, GO TO PARSER
                    C3  1008                       .BYTE   ^0303
```

```
1073' 100C                      .WORD   <CHECK_CNTLC-HEAD>
      100E  2727
      100E  2728 ;*****************************************************************************
      100E  2729 ;
      100E  2730 ;  PRINT_STRING_R
      100E  2731 ;  THIS ROUTINE PRINTS A STRING OF CHARS WITH THE CHAR COUNT AS THE FIRST BYTE
      100E  2732 ;  OF THE STRING TO THE TERMINAL IF APT IS NOT PRESENT.
      100E  2733 ;
      100E  2734 ;  INPUT CONDITIONS
      100E  2735 ;       H+L = POINTER TO THE STRING BEGINNING WITH CHAR COUNT
      100E  2736 ;
      100E  2737 ;*****************************************************************************
      100E  2738
      100E  2739 PRINT_STRING_R:
      100E  2740                      CALL    SLVECT                  ;CALL SEND LINE ROUTINE IN BOOT
   CD 100E                    .BYTE   ^O315
412E 100F                     .WORD   <SLVECT-HEAD>
      1011  2741                                                      ; BLOCK, IF APT NOT PRESENT,
      1011  2742                                                      ; THAT EXPECTS CHAR COUNT
      1011  2743                                                      ; AS FIRST CHAR IN STRING
      1011  2744                      JMP     CHECK_CNTLC             ;IF ^C TYPED, GO TO PARSER
   C3 1011                    .BYTE   ^O303
1073' 1012                    .WORD   <CHECK_CNTLC-HEAD>
      1014  2745
      1014  2746 ;*****************************************************************************
      1014  2747 ;
      1014  2748 ;  PRINT_STRING_APT
      1014  2749 ;  THIS ROUTINE PRINTS A STRING OF CHARS WITH THE CHAR COUNT AS THE FIRST BYTE
      1014  2750 ;  OF THE STRING TO THE TERMINAL OR TO APT IF PRESENT.
      1014  2751 ;
      1014  2752 ;  INPUT CONDITIONS
      1014  2753 ;       H+L = POINTER TO THE STRING BEGINNING WITH CHAR COUNT
      1014  2754 ;
      1014  2755 ;*****************************************************************************
      1014  2756
      1014  2757 PRINT_STRING_APT:
      1014  2758                      CALL    SLVECT                  ;CALL SEND LINE ROUTINE IN BOOT
   CD 1014                    .BYTE   ^O315
412E  015                     .WORD   <SLVECT-HEAD>
      1017  2759                                                      ; BLOCK THAT EXPECTS CHAR COUNT
      1017  2760                                                      ; AS FIRST CHAR IN STRING
      1017  2761                      JMP     CHECK_CNTLC             ;IF ^C TYPED, GO TO PARSER
   C3 1017                    .BYTE   ^O303
1073' 1018                    .WORD   <CHECK_CNTLC-HEAD>
      101A  2762
      101A  2763
      101A  2764 ;*****************************************************************************
      101A  2765 ;
      101A  2766 ;  CRLF_R
      101A  2767 ;  THIS ROUTINE PRINTS A RETURN AND LINE FEED (THE BOOT BLOCK PRINT LINE
      101A  2768 ;  ROUTINE SUPPLIES THE LINE FEED).
      101A  2769 ;
      101A  2770 ;*****************************************************************************
      101A  2771
      101A  2772 CRLF_R:              LXI     H,CRLF_A                ;POINT TO CR
   21 101A                    .BYTE   <H*8> ! 1
02'5B' 101B                   .BYTE   <CRLF_A-HEAD>&255 , <<CRLF_A-HEAD>&^XFF00>/256
```

```
                101D  2773                      CALL    PRINT_STRING_R          ;PRINT IT
             CD 101D                    .BYTE   ^O315
          100E' 101E                    .WORD   <PRINT_STRING_R-HEAD>
                1020  2774
                1020  2775                      RET
             C9 1020                    .BYTE   ^O311
                1021  2776
                1021  2777 ;*****************************************************************************
                1021  2778 ;
                1021  2779 ;  SPACE_R
                1021  2780 ;  THIS ROUTINE PRINTS A SPACE
                1021  2781 ;
                1021  2782 ;*****************************************************************************
                1021  2783
                1021  2784 SPACE_R:                 MVI     A,SPACE_A               ;PRINT SPACE
          20 3E 1021                    .BYTE   <A*8> ! 6 , SPACE_A&255
                1023  2785                      CALL    TYPE_CHAR
             CD 1023                    .BYTE   ^O315
          1027' 1024                    .WORD   <TYPE_CHAR-HEAD>
                1026  2786                      RET
             C9 1026                    .BYTE   ^O311
                1027  2787
                1027  2788 ;*****************************************************************************
                1027  2789 ;
                1027  2790 ;  TYPE_CHAR
                1027  2791 ;  THIS ROUTINE LOADS THE ACCUMULATOR WITH AN ASCII CHARACTER AND CALLS THE
                1027  2792 ;  SEND CHAR ROUTINE IN THE BOOT BLOCK.
                1027  2793 ;
                1027  2794 ;*****************************************************************************
                1027  2795
                1027  2796 TYPE_CHAR:
                1027  2797 PRINT_R:                CALL    SCVECT                  ;CALL SEND CHAR ROUTINE IN
             CD 1027                    .BYTE   ^O315
           4138 1028                    .WORD   <SCVECT-HEAD>
                102A  2798                                                      ; BOOT BLOCK
                102A  2799
                102A  2800                      CALL    CHECK_CNTLC             ;IF ^C TYPED, GO TO PARSER
             CD 102A                    .BYTE   ^O315
          1073' 102B                    .WORD   <CHECK_CNTLC-HEAD>
                102D  2801                      RET
             C9 102D                    .BYTE   ^O311
                102E  2802
                102E  2803
                102E  2804 ;*****************************************************************************
                102E  2805 ;
                102E  2806 ;  INPUT_LINE
                102E  2807 ;
                102E  2808 ;  THIS ROUTINE INPUTS A LINE OF DATA FROM EITHER THE CONSOLE UART OR THE
                102E  2809 ;  APT-RD UART AND PUTS THEM INTO BUFFERS.
                102E  2810 ;
                102E  2811 ;                       IF APT IS PRESENT THE APT-RD PORT WILL RECEIVE
                102E  2812 ;               A PROMPT AND WILL BE MONITORED.
                102E  2813 ;
                102E  2814 ;                       IF APT IS NOT PRESENT ONLY THE TERMINAL PORT
                102E  2815 ;               WILL RECEIVE A PROMPT AND BE MONITORED.  THE STANDARD
                102E  2816 ;               CONSOLE AND MICRO MONITOR WILL EXPECT THE TU58 TO BE PRESENT
                102E  2817 ;               IF ANY COMMAND REQUIRING IT IS USED.  IN THE CASE OF DUMP
```

```
                        102E  2818 ;                       MODE ISSUING COMMANDS REQUIRING THE TU58 ARE ILLEGAL.
                        102E  2819 ;
                        102E  2820 ;*****************************************************************************
                        102E  2821
                        102E  2822 INPUT_LINE:          CLEAR   ERROR
                     AF 102E                    .BYTE   ^0250 ! A                       ;with A
                     32 102F                    .BYTE   ^062
                  02E2' 1030                    .WORD   <ERROR-HEAD>
                        1032  2823
                        1032  2824                       ZERO    TTBUF,80                        ;ZERO BUFFER AREA
                     21 1032                    .BYTE   <H*8> ! 1
                  08'D1' 1033                   .BYTE   <TTBUF-HEAD>&255 , <<TTBUF-HEAD>&^XFF00>/256
                  50 0E 1035                    .BYTE   <C*8> ! 6 , 80&255
                     AF 1037                    .BYTE   ^0250 ! A                       ;with A
                        1038            2040$:          MOV     M,A
                     77 1038                    .BYTE   ^0100 ! <M*8> ! A
                     23 1039                    .BYTE   <H*8> ! 3
                     0D 103A                    .BYTE   ^05 ! <8*C>
                     C2 103B                    .BYTE   ^0302
                   1038' 103C                   .WORD   <2040$-HEAD>
                        103E  2825                      LXI     H,TTBUF_CNT                     ;INIT TTY BUFFER POINTER
                     21 103E                    .BYTE   <H*8> ! 1
                  08'D0' 103F                   .BYTE   <TTBUF_CNT-HEAD>&255 , <<TTBUF_CNT-HEAD>&^XFF00>/256
                        1041  2826                      CALL    GLVECT                          ;CALL GET LINE ROUTINE IN BOOT
                     CD 1041                    .BYTE   ^0315
                   4123 1042                    .WORD   <GLVECT-HEAD>
                        1044  2827                                                              ; BLOCK
                        1044  2828 1$:                  IF      CNTLC_TYPED
                     3A 1044                    .BYTE   ^072
                   0039' 1045                   .WORD   <CNTLC_TYPED-HEAD>
                     0F 1047                    .BYTE   ^017
                     D2 1048                    .BYTE   ^0322
                   1055' 1049                   .WORD   <2041$-HEAD>
                        104B  2829                      CLEAR   CNTLC_TYPED                     ;CLEAR FLAG
                     AF 104B                    .BYTE   ^0250 ! A                       ;with A
                     32 104C                    .BYTE   ^062
                   0039' 104D                   .WORD   <CNTLC_TYPED-HEAD>
                        104F  2830                      CALL    PRINT_PROMPT                    ;PROMPT FOR NEW INPUT
                     CD 104F                    .BYTE   ^0315
                   0FD6' 1050                   .WORD   <PRINT_PROMPT-HEAD>
                        1052  2831                      JMP     INPUT_LINE
                     C3 1052                    .BYTE   ^0303
                   102E' 1053                   .WORD   <INPUT_LINE-HEAD>
                        1055  2832                      ENDIF
                        1055            2041$:                                              ;GENERATE LCL 2041 NAME
                        1055  2833
                        1055  2834                      LXI     H,TTBUF                         ;INIT TTY BUFFER POINTER
                     21 1055                    .BYTE   <H*8> ! 1
                  08'D1' 1056                   .BYTE   <TTBUF-HEAD>&255 , <<TTBUF-HEAD>&^XFF00>/256
                        1058  2835                      SHLD    TTBUF_PNT
                     22 1058                    .BYTE   ^042
                   044D' 1059                   .WORD   <TTBUF_PNT-HEAD>
                        105B  2836
                        105B  2837                      RET
                     C9 105B                    .BYTE   ^0311
                        105C  2838
                        105C  2839 ;*****************************************************************************
```

```
                        105C  2840 ;
                        105C  2841 ;  INPUT_CHAR_IF
                        105C  2842 ;  THIS ROUTINE INPUTS A CHAR FROM THE APT-RD PORT
                        105C  2843 ;
                        105C  2844 ;  OUTPUT CONDTIONS
                        105C  2845 ;       ACCUMULATOR  = CHAR THAT WAS INPUT
                        105C  2846 ;
                        105C  2847 ;*****************************************************************************
                        105C  2848
                        105C  2849 INPUT_CHAR_IF:  CALL    GSVECT                  ;CALL GET SILO ROUTINE IN
              CD        105C                       .BYTE   ^0315
            4145        105D                       .WORD   <GSVECT-HEAD>
                        105F  2850                                                 ; BOOT BLOCK.  IF CHAR FOUND,
                        105F  2851                                                 ; CARRY IS SET AND CHAR IS IN
                        105F  2852                                                 ; ACCUMULATOR
                        105F  2853
                        105F  2854                 JNC     INPUT_CHAR_IF           ;LOOP UNTIL CHAR RECEIVED
              D2        105F                       .BYTE   ^0322
           105C'        1060                       .WORD   <INPUT_CHAR_IF-HEAD>
                        1062  2855
                        1062  2856                 RET
              C9        1062                       .BYTE   ^0311
                        1063  2857
                        1063  2858 ;*****************************************************************************
                        1063  2859 ;
                        1063  2860 ;  CNTLC_VEC
                        1063  2861 ;  THIS ROUTINE IS ENTERED AFTER CALLING THE BOOT BLOCK GET CHAR OR GET LINE
                        1063  2862 ;  ROUTINE IF A ^C WAS TYPED.  IT WILL SET A FLAG, RESTORE THE STACK POINTER,
                        1063  2863 ;  AND RETURN TO THE CALLING ROUTINE.
                        1063  2864 ;
                        1063  2865 ;*****************************************************************************
                        1063  2866
                        1063  2867 CNTLC_VEC:        SET     CNTLC_TYPED             ;SET FLAG INDICATING ^C
              AF        1063                       .BYTE   ^0250 ! A               ;with A
              3C        1064                       .BYTE   ^04 ! <8*A>
              32        1065                       .BYTE   ^062
           0039'        1066                       .WORD   <CNTLC_TYPED-HEAD>
                        1068  2868                 LHLD    SPBUFF                  ;HL GETS OLD SP
              2A        1068                       .BYTE   ^052
            4086        1069                       .WORD   <SPBUFF-HEAD>
                        106B  2869                 SPHL                            ;RESTORE SP
              F9        106B                       .BYTE   ^0371
                        106C  2870                 STC                             ;SET CARRY FOR INPUT_CHAR_IF
              37        106C                       .BYTE   ^067
                        106D  2871                 MVI     A,<^X3>                 ;PUT HEX FOR ^C IN A
           03 3E        106D                       .BYTE   <A*8> ! 6 , ^X3&255
                        106F  2872                 RET                             ;RETURN WILL NOW WORK
              C9        106F                       .BYTE   ^0311
                        1070  2873
                        1070  2874 ;*****************************************************************************
                        1070  2875 ;
                        1070  2876 ;  CNTLP_VEC
                        1070  2877 ;  THIS ROUTINE IS ENTERED AFTER CALLING THE BOOT BLOCK GET CHAR OR GET LINE
                        1070  2878 ;  ROUTINE IF A ^P WAS TYPED.  IT WILL CAUSE CRD TO ABORT AND RETURN TO THE
                        1070  2879 ;  CONSOLE.
                        1070  2880 ;
                        1070  2881 ;*****************************************************************************
```

```
                     1070  2882
                     1070  2883 CNTLP_VEC:
                     1070  2884                 JMP     MENU                    ;EITHER EXIT OR RESTART
          C3         1070                 .BYTE   ^0303
        1616'        1071                 .WORD   <MENU-HEAD>
                     1073  2885
                     1073  2886
                     1073  2887 ;*****************************************************************************
                     1073  2888 ;
                     1073  2889 ;  CHECK_CNTLC
                     1073  2890 ;
                     1073  2891 ;  THIS ROUTINE CHECKS THE CNTLC_TYPED FLAG TO SEE IF A CNTLC HAS BEEN
                     1073  2892 ;  DETECTED.  IF SO, IT CALLS THE SET_FOR_CONT_R ROUTINE.
                     1073  2893 ;
                     1073  2894 ;*****************************************************************************
                     1073  2895
                     1073  2896 CHECK_CNTLC:      LDA     CNTLC_TYPED             ;GET FLAG
          3A         1073                 .BYTE   ^072
        0039'        1074                 .WORD   <CNTLC_TYPED-HEAD>
                     1076  2897                 ORA     A                       ;SET CONDITION CODES
          B7         1076                 .BYTE   ^0260 ! A
                     1077  2898                 CNZ     SET_FOR_CONT_R          ;CALL SET FOR CONT IF ^C SET
          C4         1077                 .BYTE   ^0304
        107B'        1078                 .WORD   <SET_FOR_CONT_R-HEAD>
                     107A  2899                 RET                             ;ELSE RETURN
          C9         107A                 .BYTE   ^0311
                     1078  2900
                     1078  2901 ;*****************************************************************************
                     1078  2902 ;
                     1078  2903 ;  SET_FOR_CONT_R:
                     1078  2904 ;
                     1078  2905 ;  THIS ROUTINE SETS UP FOR A CONTINUE COMMAND TO BE EXECUTED IF IT IS
                     1078  2906 ;  ASKED FOR
                     1078  2907 ;
                     1078  2908 ;*****************************************************************************
                     1078  2909
                     1078  2910 SET_FOR_CONT_R: CLEAR   CNTLC_TYPED
          AF         1078                 .BYTE   ^0250 ! A               ;with A
          32         107C                 .BYTE   ^062
        0039'        107D                 .WORD   <CNTLC_TYPED-HEAD>
                     107F  2911                 CLEAR   NO_SAVE_UPC_CSR         ;CLEAR FLAG
          AF         107F                 .BYTE   ^0250 ! A               ;with A
          32         1080                 .BYTE   ^062
        03C0'        1081                 .WORD   <NO_SAVE_UPC_CSR-HEAD>
                     1083  2912
                     1083  2913                 LDA     EXECUTE                 ;SAVE EXECUTION FLAGS
          3A         1083                 .BYTE   ^072
        02EA'        1084                 .WORD   <EXECUTE-HEAD>
                     1086  2914                 STA     C_EXECUTE
          32         1086                 .BYTE   ^062
        0250'        1087                 .WORD   <C_EXECUTE-HEAD>
                     1089  2915                 LDA     RUNNING
          3A         1089                 .BYTE   ^072
        0403'        108A                 .WORD   <RUNNING-HEAD>
                     108C  2916                 STA     C_RUNNING
          32         108C                 .BYTE   ^062
        0251'        108D                 .WORD   <C_RUNNING-HEAD>
```

```
                    108F  2917
                    108F  2918                    IF      RUNNING
                      3A  108F                    .BYTE   ^O72
                  0403' 1090                    .WORD   <RUNNING-HEAD>
                      0F  1092                    .BYTE   ^O17
                      D2  1093                    .BYTE   ^O322
                  109E' 1094                    .WORD   <2042$-HEAD>
                        1096  2919                CALL    STOP_CPU                ;HALT CPU
                      CD  1096                    .BYTE   ^O315
                  10C9' 1097                    .WORD   <STOP_CPU-HEAD>
                        1099  2920                SET     MICRO_STEP              ;SET FLAG TO PRINT UPC-1
                      AF  1099                    .BYTE   ^O250 ! A               ;with A
                      3C  109A                    .BYTE   ^O4 ! <8*A>
                      32  109B                    .BYTE   ^O62
                  038A' 109C                    .WORD   <MICRO_STEP-HEAD>
                        109E  2921                ENDIF
                        109E            2042$:                                  ;GENERATE LCL 2042 NAME
                        109E  2922
                        109E  2923                CLEAR   MICRO_STEP              ;CLEAR MICRO STEP MODE
                      AF  109E                    .BYTE   ^O250 ! A               ;with A
                      32  109F                    .BYTE   ^O62
                  038A' 10A0                    .WORD   <MICRO_STEP-HEAD>
                        10A2  2924                LHLD    SAVED_UPC               ;SAVE CSR AND UPC FOR CONT
                      2A  10A2                    .BYTE   ^O52
                  00C4' 10A3                    .WORD   <SAVED_UPC-HEAD>
                        10A5  2925                SHLD    CONT_SAVE_UPC
                      22  10A5                    .BYTE   ^O42
                  00C8' 10A6                    .WORD   <CONT_SAVE_UPC-HEAD>
                        10A8  2926
                        10A8  2927                LXI     H,SAVED_CSR
                      21  10A8                    .BYTE   <H*8> ! 1
                00'BB' 10A9                     .BYTE   <SAVED_CSR-HEAD>&255 , <<SAVED_CSR-HEAD>&^XFF00>/256
                        10AB  2928                LXI     D,CONT_SAVE_CSR
                      11  10AB                    .BYTE   <D*8> ! 1
                00'BE' 10AC                     .BYTE   <CONT_SAVE_CSR-HEAD>&255 , <<CONT_SAVE_CSR-HEAD>&^XFF00>/256
                        10AE  2929                CALL    MOVER_3_R
                      CD  10AE                    .BYTE   ^O315
                  0F5F' 10AF                    .WORD   <MOVER_3_R-HEAD>
                        10B1  2930
                        10B1  2931                CLEAR   EXECUTE                 ;CLEAR EXECUTION FLAG TO ENTER
                      AF  10B1                    .BYTE   ^O250 ! A               ;with A
                      32  10B2                    .BYTE   ^O62
                  02EA' 10B3                    .WORD   <EXECUTE-HEAD>
                        10B5  2932                                              ; PARSER
                        10B5  2933
                        10B5  2934                IFN     CONSOLE_TEST            ;IF 8085 TEST IS NOT RUNNING
                      3A  10B5                    .BYTE   ^O72
                  0255' 10B6                    .WORD   <CONSOLE_TEST-HEAD>
                      0F  10B8                    .BYTE   ^O17
                      DA  10B9                    .BYTE   ^O332
                  10C4' 10BA                    .WORD   <2043$-HEAD>
                        10BC  2935                SET     CONTINUE                ;CONTINUE ALWAYS LEGAL FOR WCS
                      AF  10BC                    .BYTE   ^O250 ! A               ;with A
                      3C  10BD                    .BYTE   ^O4 ! <8*A>
                      32  10BE                    .BYTE   ^O62
                  003A' 10BF                    .WORD   <CONTINUE-HEAD>
                        10C1  2936                                              ; TESTS IF THIS ROUTINE CALLED
```

```
                        10C1  2937                    ELSE
          C3            10C1                .BYTE   ^O303
       10C6'            10C2                .WORD   <2044$-HEAD>
                        10C4          2043$:                                    ;GENERATE LOCAL SYMBOL NAME
                        10C4  2938                    OUT     CLRLIT            ;IF 8085 TEST, TURN OFF RUN
       3C D3            10C4                .BYTE   ^O323 , CLRLIT&255      ;register A
                        10C6  2939                                              ; LIGHT
                        10C6  2940                    ENDIF
                        10C6          2044$:                                    ;GENERATE LCL 2044 NAME
                        10C6  2941
                        10C6  2942                    JMP     MENU
          C3            10C6                .BYTE   ^O303
       1616'            10C7                .WORD   <MENU-HEAD>
                        10C9  2943
                        10C9  2944 ;*****************************************************************************
                        10C9  2945 ;
                        10C9  2946 ;  STOP_CPU
                        10C9  2947 ;  THIS ROUTINE STOPS THE CPU. IT SAVES THE UPC AND CSR SO THAT
                        10C9  2948 ;  EXECUTION MAY RESUME AT THE NEXT INSTRUCTION
                        10C9  2949 ;
                        10C9  2950 ;*****************************************************************************
                        10C9  2951
                        10C9  2952 STOP_CPU:          CLEAR   RUNNING
          AF            10C9                .BYTE   ^O250 ! A               ;with A
          32            10CA                .BYTE   ^O62
        0403'           10CB                .WORD   <RUNNING-HEAD>
                        10CD  2953                    OUT     CLRLIT                    ;CLEAR RUNLITE
       3C D3            10CD                .BYTE   ^O323 , CLRLIT&255      ;register A
                        10CF  2954
                        10CF  2955                    OUT     CLRCLK                    ;STOP CPU CLOCK
       20 D3            10CF                .BYTE   ^O323 , CLRCLK&255      ;register A
                        10D1  2956                    OUT     DISMEM                    ;DISABLE MEM REF
       A7 D3            10D1                .BYTE   ^O323 , DISMEM&255      ;register A
                        10D3  2957                    OUT     DISPAR                    ;DISABLE PARITY
       A1 D3            10D3                .BYTE   ^O323 , DISPAR&255      ;register A
                        10D5  2958
                        10D5  2959                    IFN     NO_SAVE_UPC_CSR           ;DON'T SAVE UPC AND CSR
          3A            10D5                .BYTE   ^O72
        03C0'           10D6                .WORD   <NO_SAVE_UPC_CSR-HEAD>
          0F            10D8                .BYTE   ^O17
          DA            10D9                .BYTE   ^O332
       10F7'            10DA                .WORD   <2045$-HEAD>
                        10DC  2960                                              ; IF FLAG SET
                        10DC  2961                    CALL    LOAD_UPC_R        ;GET UPC AND SAVE UPC
          CD            10DC                .BYTE   ^O315
       0F19'            10DD                .WORD   <LOAD_UPC_R-HEAD>
                        10DF  2962                    LHLD    UPC_VALUE
          2A            10DF                .BYTE   ^O52
       00C2'            10E0                .WORD   <UPC_VALUE-HEAD>
                        10E2  2963                    SHLD    SAVED_UPC
          22            10E2                .BYTE   ^O42
       00C4'            10E3                .WORD   <SAVED_UPC-HEAD>
                        10E5  2964                    CALL    LOAD_UPC_R        ;RETURN VALUE
          CD            10E5                .BYTE   ^O315
       0F19'            10E6                .WORD   <LOAD_UPC_R-HEAD>
                        10E8  2965
                        10F8  2966                    CALL    LOAD_CSR_R        ;GET CSR AND SAVE CSR
```

```
                CD  10E8                .BYTE   ^O315
            0EE7'   10E9                .WORD   <LOAD_CSR_R-HEAD>
                    10EB  2967                  LXI     H,CSR_VALUE
                21  10EB                .BYTE   <H*8> ! 1
            00'B8'  10EC                .BYTE   <CSR_VALUE-HEAD>&255 , <<CSR_VALUE-HEAD>&^XFF00>/256
                    10EE  2968                  LXI     D,SAVED_CSR
                11  10EE                .BYTE   <D*8> ! 1
            00'BB'  10EF                .BYTE   <SAVED_CSR-HEAD>&255 , <<SAVED_CSR-HEAD>&^XFF00>/256
                    10F1  2969                  CALL    MOVER_3_R
                CD  10F1                .BYTE   ^O315
            0F5F'   10F2                .WORD   <MOVER_3_R-HEAD>
                    10F4  2970                  CALL    LOAD_CSR_R              ;RETURN VALUE
                CD  10F4                .BYTE   ^O315
            0EE7'   10F5                .WORD   <LOAD_CSR_R-HEAD>
                    10F7  2971
                    10F7  2972                  ENDIF
                    10F7        2045$:                                          ;GENERATE LCL 2045 NAME
                    10F7  2973
                    10F7  2974                  CLEAR   NO_SAVE_UPC_CSR         ;CLEAR FLAG
                AF  10F7                .BYTE   ^O250 ! A               ;with A
                32  10F8                .BYTE   ^O62
            03C0'   10F9                .WORD   <NO_SAVE_UPC_CSR-HEAD>
                    10FB  2975                  CALL    CLEAR_CPU_ATT_R
                CD  10FB                .BYTE   ^O315
            1A61'   10FC                .WORD   <CLEAR_CPU_ATT_R-HEAD>
                    10FE  2976
                    10FE  2977                  RET
                C9  10FE                .BYTE   ^O311
                    10FF  2978
                    10FF  2979
                    10FF  2980 ;*****************************************************************************
                    10FF  2981 ;
                    10FF  2982 ;  START CPU
                    10FF  2983 ;  THIS ROUTINE STARTS THE CPU AT THE ADDRESS GIVEN BY THE RAM
                    10FF  2984 ;  LOCATION STARTING_ADD
                    10FF  2985 ;  IF MICRO STEP IS SET THE CPU IS SET UP BUT NOT STARTED
                    10FF  2986 ;
                    10FF  2987 ;*****************************************************************************
                    10FF  2988
                    10FF  2989 START_CPU:       LHLD    STARTING_UPC
                2A  10FF                .BYTE   ^O52
            0407'   1100                .WORD   <STARTING_UPC-HEAD>
                    1102  2990                  SHLD    UPC_VALUE               ;MOVE STARTING ADDRESS
                22  1102                .BYTE   ^O42
            00C2'   1103                .WORD   <UPC_VALUE-HEAD>
                    1105  2991
                    1105  2992                  CALL    LOAD_UPC_R
                CD  1105                .BYTE   ^O315
            0F19'   1106                .WORD   <LOAD_UPC_R-HEAD>
                    1108  2993                  CALL    NOP_CSR_R
                CD  1108                .BYTE   ^O315
            0EC4'   1109                .WORD   <NOP_CSR_R-HEAD>
                    1108  2994
                    1108  2995                  JMP     START_CPU_COM           ;GO TO ROUTINE COMMON TO
                C3  1108                .BYTE   ^O303
            1123'   110C                .WORD   <START_CPU_COM-HEAD>
                    110E  2996                                                  ; START OR RESTART
```

```
                        110E  2997
                        110E  2998
                        110E  2999 ;*****************************************************************************
                        110E  3000 ;
                        110E  3001 ;  THIS SUBROUTINE IS USED TO RESTART THE CPU AND IT RESTORES THE
                        110E  3002 ;  CSR AND UPC.  IF MICRO STEP IS SET THE CPU IS SET UP BUT NOT STARTED
                        110E  3003 ;
                        110E  3004 ;*****************************************************************************
                        110E  3005
                        110E  3006 RESTART_CPU:    LHLD    SAVED_UPC                ;RESTORE UPC
                  2A    110E                       .BYTE   ^052
                00C4'   110F                       .WORD   <SAVED_UPC-HEAD>
                        1111  3007                 SHLD    UPC_VALUE
                  22    1111                       .BYTE   ^042
                00C2'   1112                       .WORD   <UPC_VALUE-HEAD>
                        1114  3008
                        1114  3009                 CALL    LOAD_UPC_R
                  CD    1114                       .BYTE   ^0315
                0F19'   1115                       .WORD   <LOAD_UPC_R-HEAD>
                        1117  3010                 LXI     H,SAVED_CSR              ;LOAD CSR
                  21    1117                       .BYTE   <H*8> ! 1
              00'B8'    1118                       .BYTE   <SAVED_CSR-HEAD>&255 , <<SAVED_CSR-HEAD>&^XFF00>/256
                        111A  3011                 LXI     D,CSR_VALUE
                  11    111A                       .BYTE   <D*8> ! 1
              00'B8'    111B                       .BYTE   <CSR_VALUE-HEAD>&255 , <<CSR_VALUE-HEAD>&^XFF00>/256
                        111D  3012                 CALL    MOVER_3_R
                  CD    111D                       .BYTE   ^0315
                0F5F'   111E                       .WORD   <MOVER_3_R-HEAD>
                        1120  3013                 CALL    LOAD_CSR_R
                  CD    1120                       .BYTE   ^0315
                0EE7'   1121                       .WORD   <LOAD_CSR_R-HEAD>
                        1123  3014
                        1123  3015 START_CPU_COM:
                        1123  3016                 IF      PAR_ON
                  3A    1123                       .BYTE   ^072
                03D6'   1124                       .WORD   <PAR_ON-HEAD>
                  0F    1126                       .BYTE   ^017
                  D2    1127                       .BYTE   ^0322
                112C'   1128                       .WORD   <2046$-HEAD>
                        112A  3017                 OUT     ENBPAR                   ;ENABLE PARITY
               A0 D3    112A                       .BYTE   ^0323 , ENBPAR&255       ;register A
                        112C  3018                 ENDIF
                        112C        2046$:                                          ;GENERATE LCL 2046 NAME
                        112C  3019
                        112C  3020                 IF      MEM_REQ
                  3A    112C                       .BYTE   ^072
                0354'   112D                       .WORD   <MEM_REQ-HEAD>
                  0F    112F                       .BYTE   ^017
                  D2    1130                       .BYTE   ^0322
                1135'   1131                       .WORD   <2047$-HEAD>
                        1133  3021                 OUT     ENBMEM                   ;ENABLE MEM REF
               A6 D3    1133                       .BYTE   ^0323 , ENBMEM&255       ;register A
                        1135  3022                 ENDIF
                        1135        2047$:                                          ;GENERATE LCL 2047 NAME
                        1135  3023
                        1135  3024                 IFN     MICRO_STEP
                  3A    1135                       .BYTE   ^072
```

```
038A' 1136                .WORD   <MICRO_STEP-HEAD>
   0F 1138                .BYTE   ^017
   DA 1139                .BYTE   ^0332
113E' 113A                .WORD   <2048$-HEAD>
      113C 3025                   OUT     SETCLK
21 D3 113C                .BYTE   ^0323 , SETCLK&255      ;register A
      113E 3026                   ENDIF
      113E       2048$:                                   ;GENERATE LCL 2048 NAME
      113E 3027
      113E 3028                   SET     RUNNING
   AF 113E                .BYTE   ^0250 ! A              ;with A
   3C 113F                .BYTE   ^04 ! <8*A>
   32 1140                .BYTE   ^062
0403' 1141                .WORD   <RUNNING-HEAD>
      1143 3029                   OUT     SETLIT                  ;SET RUN LITE
3D D3 1143                .BYTE   ^0323 , SETLIT&255      ;register A
      1145 3030
      1145 3031                   RET
   C9 1145                .BYTE   ^0311
      1146 3032
      1146 3033
      1146 3034 ;*****************************************************************************
      1146 3035 ;
      1146 3036 ;  WRITE_LS_R
      1146 3037 ;  THIS RUTINE WRITES A 32 BIT LS VALUE
      1146 3038 ;  INPUT CONDITIONS:
      1146 3039 ;       AC = LS ADDRESS THAT IS TO BE WRITTEN
      1146 3040 ;       DATA0 THRU DATA3 CONTAIN THE 32 BITS OF DATA TO BE WRITTEN
      1146 3041 ;
      1146 3042 ;*****************************************************************************
      1146 3043
      1146 3044 WRITE_LS_R:       STA     XD_ADDRS                ;SAVE ADDRESS TO BE WRITTEN
   32 1146                .BYTE   ^062
0071' 1147                .WORD   <XD_ADDRS-HEAD>
      1149 3045
      1149 3046                   LXI     H,S_WRITE_LS            ;RESET MOVE INSTRUCTION
   21 1149                .BYTE   <H*8> ! 1
06'72' 114A               .BYTE   <S_WRITE_LS-HEAD>&255 , <<S_WRITE_LS-HEAD>&^XFF00>/256
      114C 3047                   LXI     D,X_WRITE_LS+1
   11 114C                .BYTE   <D*8> ! 1
06'6F' 114D               .BYTE   <X_WRITE_LS+1-HEAD>&255 , <<X_WRITE_LS+1-HEAD>&^XFF00>/256
      114F 3048                   CALL    MOVER_3_R
   CD 114F                .BYTE   ^0315
0F5F' 1150                .WORD   <MOVER_3_R-HEAD>
      1152 3049
      1152 3050                   LXI     H,X_WRITE_LS+1          ;MODIFY MOVE INSTRUCTION
   21 1152                .BYTE   <H*8> ! 1
06'6F' 1153               .BYTE   <X_WRITE_LS+1-HEAD>&255 , <<X_WRITE_LS+1-HEAD>&^XFF00>/256
      1155 3051                   CALL    MAKE_XD_R
   CD 1155                .BYTE   ^0315
1183' 1156                .WORD   <MAKE_XD_R-HEAD>
      1158 3052
      1158 3053                   CALL    WRITE_DATA_32_R         ;WRITE THE DATA TO 2901
   CD 1158                .BYTE   ^0315
11D2' 1159                .WORD   <WRITE_DATA_32_R-HEAD>
      1158 3054
      1158 3055                   LXI     H,X_WRITE_LS            ;PERFORM THE WRITE
```

```
                21  115B                .BYTE   <H*8> ! 1
          06'6E'    115C                .BYTE   <X_WRITE_LS-HEAD>&255 , <<X_WRITE_LS-HEAD>&^XFF00>/256
                    115E  3056                  CALL    PERFORM_CSR_R
                CD  115E                .BYTE   ^O315
           12F2'    115F                .WORD   <PERFORM_CSR_R-HEAD>
                    1161  3057
                    1161  3058                  CALL    GCVECT                  ;SEE IF INPUT FOR SILO
                CD  1161                .BYTE   ^O315
           4129     1162                .WORD   <GCVECT-HEAD>
                    1164  3059                  RET
                C9  1164                .BYTE   ^O311
                    1165  3060
                    1165  3061 ;*****************************************************************************
                    1165  3062 ;
                    1165  3063 ;  READ_LS_R
                    1165  3064 ;  THIS ROUTINE READS A 32 BIT LS VALUE
                    1165  3065 ;  INPUT CONDITIONS:
                    1165  3066 ;       AC = LS ADDRESS THAT IS TO BE READ
                    1165  3067 ;  OUTPUT CONDTIONS:
                    1165  3068 ;       DATA0 THRU DATA3 CONTAIN THE 32 BITS READ
                    1165  3069 ;
                    1165  3070 ;*****************************************************************************
                    1165  3071
                    1165  3072 READ_LS_7:       MVI     A,LS_7                  ;LS ADDRESS 7
          07 3E     1165                .BYTE   <A*8> ! 6 , LS_7&255
                    1167  3073
                    1167  3074 READ_LS_R:       STA     XD_ADDRS                ;SAVE ADDRESS TO BE READ
                32  1167                .BYTE   ^O62
           0071'    1168                .WORD   <XD_ADDRS-HEAD>
                    116A  3075
                    116A  3076                  LXI     H,S_READ_LS             ;RESET MOVE INSTRUCTION
                21  116A                .BYTE   <H*8> ! 1
          06'67'    116B                .BYTE   <S_READ_LS-HEAD>&255 , <<S_READ_LS-HEAD>&^XFF00>/256
                    116D  3077                  LXI     D,X_READ_LS+1
                11  116D                .BYTE   <D*8> ! 1
          06'64'    116E                .BYTE   <X_READ_LS+1-HEAD>&255 , <<X_READ_LS+1-HEAD>&^XFF00>/256
                    1170  3078                  CALL    MOVER_3_R
                CD  1170                .BYTE   ^O315
           0F5F'    1171                .WORD   <MOVER_3_R-HEAD>
                    1173  3079
                    1173  3080                  LXI     H,X_READ_LS+1           ;AND MODIFY MOVE TO DO IT
                21  1173                .BYTE   <H*8> ! 1
          06'64'    1174                .BYTE   <X_READ_LS+1-HEAD>&255 , <<X_READ_LS+1-HEAD>&^XFF00>/256
                    1176  3081                  CALL    MAKE_XD_R
                CD  1176                .BYTE   ^O315
           1183'    1177                .WORD   <MAKE_XD_R-HEAD>
                    1179  3082
                    1179  3083                  LXI     H,X_READ_LS             ;PERFORM THE READ
                21  1179                .BYTE   <H*8> ! 1
          06'63'    117A                .BYTE   <X_READ_LS-HEAD>&255 , <<X_READ_LS-HEAD>&^XFF00>/256
                    117C  3084                  CALL    PERFORM_CSR_R
                CD  117C                .BYTE   ^O315
           12F2'    117D                .WORD   <PERFORM_CSR_R-HEAD>
                    117F  3085                  CALL    READ_DATA_32_R
                CD  117F                .BYTE   ^O315
           1184'    1180                .WORD   <READ_DATA_32_R-HEAD>
                    1182  3086
```

```
                1182  3087                    RET
          C9    1182                    .BYTE   ^0311
                1183  3088
                1183  3089
                1183  3090 ;*****************************************************************************
                1183  3091 ;
                1183  3092 ;  MAKE_XD_R
                1183  3093 ;  THIS ROUTINE MAKES AN INSTRUCTION HAVE THE PROPER XD ADDRESS FIELD
                1183  3094 ;
                1183  3095 ;  INPUT CONDITIONS:
                1183  3096 ;       XD_ADDRS = DATA TO BE INSERTED INTO THE XD ADDRESS FIELD (BITS 9-16)
                1183  3097 ;       H + L    = POINTS TO THE 3 BYTE INSTRUCTION TO MODIFY
                1183  3098 ;
                1183  3099 ;*****************************************************************************
                1183  3100
                1183  3101 MAKE_XD_R:       LDA     XD_ADDRS                ;PUT BIT 7 OF XD_ADDRS
          3A    1183                    .BYTE   ^072
      0071'     1184                    .WORD   <XD_ADDRS-HEAD>
                1186  3102                    RAL                             ;INTO BIT 16 OF INSTRUCTION
          17    1186                    .BYTE   ^027
                1187  3103                    MVI     A,0
       00 3E    1187                    .BYTE   <A*8> ! 6 , 0&255
                1189  3104                    RAL
          17    1189                    .BYTE   ^027
                118A  3105                    ORA     M
          B6    118A                    .BYTE   ^0260 ! M
                118B  3106                    MOV     M,A
          77    118B                    .BYTE   ^0100 ! <M*8> ! A
                118C  3107                    INX     H
          23    118C                    .BYTE   <H*8> ! 3
                118D  3108
                118D  3109                    LDA     XD_ADDRS                ;PUT BITS 0 - 6 INTO BITS
          3A    118D                    .BYTE   ^072
      0071'     118E                    .WORD   <XD_ADDRS-HEAD>
                1190  3110                                                    ;15 - 9 OF INSTRUCTION
                1190  3111                    ORA     A                       ;CLEAR CARRY
          B7    1190                    .BYTE   ^0260 ! A
                1191  3112                    RAL
          17    1191                    .BYTE   ^027
                1192  3113                    ORA     M
          B6    1192                    .BYTE   ^0260 ! M
                1193  3114                    MOV     M,A
          77    1193                    .BYTE   ^0100 ! <M*8> ! A
                1194  3115
                1194  3116                    RET
          C9    1194                    .BYTE   ^0311
                1195  3117
                1195  3118
                1195  3119 ;*****************************************************************************
                1195  3120 ;
                1195  3121 ;  MAKE_B
                1195  3122 ;  THIS ROUTINE MAKES AN INSTRUCTION HAVE THE PROPER B ADDRESS FIELD
                1195  3123 ;
                1195  3124 ;  INPUT CONDITIONS:
                1195  3125 ;       B_ADDRS  = DATA TO BE INSERTED INTO THE B ADDRESS FIELD (BITS 7-8)
                1195  3126 ;       H + L    = POINTS TO THE 3 BYTE INSTRUCTION TO MODIFY
                1195  3127 ;
```

```
                     1193  3128 ;*****************************************************************************
                     1195  3129
                     1195  3130 MAKE_B:          INX     H                       ;MOVE TO SECOND BYTE
                  23 1195                .BYTE   <H*8> ! 3
                     1196  3131
                     1196  3132                LDA     B_ADDRS                 ;PUT BIT 2 OF B_ADDRS INTO
                  3A 1196                .BYTE   ^072
                0240' 1197               .WORD   <B_ADDRS-HEAD>
                     1199  3133                RAR                             ;BIT 8 OF THE INSTRUCTION
                  1F 1199                .BYTE   ^037
                     119A  3134                RAR
                  1F 119A                .BYTE   ^037
                     119B  3135                MVI     A,0
               00 3E 119B                .BYTE   <A*8> ! 6 , 0&255
                     119D  3136                RAL
                  17 119D                .BYTE   ^027
                     119E  3137                ORA     M
                  B6 119E                .BYTE   ^0260 ! M
                     119F  3138                MOV     M,A
                  77 119F                .BYTE   ^0100 ! <M*8> ! A
                     11A0  3139                INX     H
                  23 11A0                .BYTE   <H*8> ! 3
                     11A1  3140
                     11A1  3141                LDA     B_ADDRS                 ;PUT BIT 1 OF B_ADDRS INTO
                  3A 11A1                .BYTE   ^072
                0240' 11A2               .WORD   <B_ADDRS-HEAD>
                     11A4  3142                RAR                             ;BIT 7 OF THE INSTRUCTION
                  1F 11A4                .BYTE   ^037
                     11A5  3143                MVI     A,0
               00 3E 11A5                .BYTE   <A*8> ! 6 , 0&255
                     11A7  3144                RAR
                  1F 11A7                .BYTE   ^037
                     11A8  3145                ORA     M
                  B6 11A8                .BYTE   ^0260 ! M
                     11A9  3146                MOV     M,A
                  77 11A9                .BYTE   ^0100 ! <M*8> ! A
                     11AA  3147
                     11AA  3148                RET
                  C9 11AA                .BYTE   ^0311
                     11AB  3149
                     11AB  3150
                     11AB  3151 ;*****************************************************************************
                     11AB  3152 ;
                     11AB  3153 ;  MAKE_A
                     11AB  3154 ;  THIS ROUTINE MAKES AN INSTRUCTION HAVE THE PROPER A ADDRESS FIELD
                     11AB  3155 ;
                     11AB  3156 ;  INPUT CONDITIONS:
                     11AB  3157 ;       A_ADDRS   = DATA TO BE INSERTED INTO THE A ADDRESS FIELD (BITS 9-10)
                     11AB  3158 ;       H + L     = POINTS TO THE 3 BYTE INSTRUCTION TO MODIFY
                     11AB  3159 ;
                     11AB  3160 ;*****************************************************************************
                     11AB  3161
                     11AB  3162 MAKE_A:          INX     H                       ;MOVE TO SECOND BYTE
                  23 11AB                .BYTE   <H*8> ! 3
                     11AC  3163
                     11AC  3164                ORA     A                       ;CLEAR THE CARRY FOR ROTATES
                  B7 11AC                .BYTE   ^0260 ! A
```

```
                    11AD  3165
                    11AD  3166                  LDA     A_ADDRS                 ;PUT BIT 2 OF B_ADDRS INTO
              3A    11AD                  .BYTE   ^O72
          0163'     11AE                  .WORD   <A_ADDRS-HEAD>
                    11B0  3167                  RAL                             ;MOVE TO BITS 9 - 10
              17    11B0                  .BYTE   ^O27
                    11B1  3168                  ORA     M
              B6    11B1                  .BYTE   ^O260 ! M
                    11B2  3169                  MOV     M,A                     ;OR IT INTO INSTRUCTION
              77    11B2                  .BYTE   ^O100 ! <M*8> ! A
                    11B3  3170
                    11B3  3171                  RET
              C9    11B3                  .BYTE   ^O311
                    11B4  3172
                    11B4  3173
                    11B4  3174 ;*****************************************************************************
                    11B4  3175 ;
                    11B4  3176 ;  READ_DATA_32_R
                    11B4  3177 ;  THIS ROUTINE IS USED TO READ 32 BITS OF DATA FROM THE
                    11B4  3178 ;  CPU 8 BITS AT A TIME VIA THE 2901
                    11B4  3179 ;
                    11B4  3180 ;*****************************************************************************
                    11B4  3181
                    11B4  3182 READ_DATA_32_R: LXI     D,DATA3                 ;DE POINTS TO DATA AREA
              11    11B4                  .BYTE   <D*8> ! 1
          00'AC'    11B5                  .BYTE   <DATA3-HEAD>&255 , <<DATA3-HEAD>&^XFF00>/256
                    11B7  3183                  CALL    1$                      ;READ BYTE AND STORE
              CD    11B7                  .BYTE   ^O315
           11C9'    11B8                  .WORD   <1$-HEAD>
                    11BA  3184
                    11BA  3185                  LXI     D,DATA2                 ;DE POINTS TO DATA AREA
              11    11BA                  .BYTE   <D*8> ! 1
          00'AB'    11BB                  .BYTE   <DATA2-HEAD>&255 , <<DATA2-HEAD>&^XFF00>/256
                    11BD  3186                  CALL    1$                      ;READ NEXT BYTE AND STORE
              CD    11BD                  .BYTE   ^O315
           11C9'    11BE                  .WORD   <1$-HEAD>
                    11C0  3187
                    11C0  3188                  LXI     D,DATA1                 ;DE POINTS TO DATA AREA
              11    11C0                  .BYTE   <D*8> ! 1
          00'AA'    11C1                  .BYTE   <DATA1-HEAD>&255 , <<DATA1-HEAD>&^XFF00>/256
                    11C3  3189                  CALL    1$                      ;READ NEXT BYTE AND STORE
              CD    11C3                  .BYTE   ^O315
           11C9'    11C4                  .WORD   <1$-HEAD>
                    11C6  3190
                    11C6  3191                  LXI     D,DATA0                 ;DE POINTS TO DATA AREA
              11    11C6                  .BYTE   <D*8> ! 1
          00'A9'    11C7                  .BYTE   <DATA0-HEAD>&255 , <<DATA0-HEAD>&^XFF00>/256
                    11C9  3192                                                  ; READ NEXT BYTE AND STORE
                    11C9  3193
                    11C9  3194 1$:              OUT     YBUSRD
           EC D3    11C9                  .BYTE   ^O323 , YBUSRD&255      ;register A
                    11CB  3195                  IN      READ                    ;READ MOST SIG. BYTE
           80 DB    11CB                  .BYTE   ^O333 , READ&255        ;to register A
                    11CD  3196                  STAX    D                       ;STORE IN DATA BUFFER
              12    11CD                  .BYTE   <D*8> ! 2
                    11CE  3197                  CALL    SHIFT_R_2901_R          ;READ NEXT BYTE
              CD    11CE                  .BYTE   ^O315
```

```
1250' 11CF                      .WORD   <SHIFT_R_2901_R-HEAD>
      11D1  3198                RET
  C9  11D1                      .BYTE   ^O311
      11D2  3199
      11D2  3200
      11D2  3201 ;*****************************************************************************
      11D2  3202 ;
      11D2  3203 ;  WRITE_DATA_32_R
      11D2  3204 ;  THIS ROUTINE IS USED TO WRITE 32 BITS OF DATA TO THE CPU
      11D2  3205 ;  DATA WRITES TO THE CPU ARE DONE IN TWO DIFFERENT MODES
      11D2  3206 ;  SLOW MODE: AND FAST MODE:  SLOW MODE USES NO SUPPORT MICRO CODE
      11D2  3207 ;  IN THE WCS MEMORY, AND THE FAST MODE DOES. THE MODE IS INITIALLY
      11D2  3208 ;  SLOW MODE UNTIL A WCS FILE IMAGE IS LOADED.  THEN IT IS SWITCHED TO
      11D2  3209 ;  FAST MODE UNLESS THERE IS AN ERROR IN THE FAST MODE VERIFY.
      11D2  3210 ;
      11D2  3211 ;*****************************************************************************
      11D2  3212
      11D2  3213 WRITE_DATA_32_R:
      11D2  3214                      IFN     FAST_WRITE
  3A  11D2                      .BYTE   ^O72
02FC' 11D3                      .WORD   <FAST_WRITE-HEAD>
  0F  11D5                      .BYTE   ^O17
  DA  11D6                      .BYTE   ^O332
120A' 11D7                      .WORD   <2049$-HEAD>
      11D9  3215                                                      ;SLOW WRITE
      11D9  3216                LXI     H,X_CLR_WRO                   ;CLEAR WORKING REG AND COMPLIM.
  21  11D9                      .BYTE   <H*8> ! 1
00'A1' 11DA                     .BYTE   <X_CLR_WRO-HEAD>&255 , <<X_CLR_WRO-HEAD>&^XFF00>/256
      11DC  3217                CALL    PERFORM_CSR_R                 ;TO MAKE ALL 1'S THEN ROTATE
  CD  11DC                      .BYTE   ^O315
12F2' 11DD                      .WORD   <PERFORM_CSR_R-HEAD>
      11DF  3218                                                      ;LEFT TO MAKE A 1 AND ASH LEFT
      11DF  3219                                                      ;TO MAKE A ZERO
      11DF  3220
      11DF  3221                DO      32
  21  11DF                      .BYTE   <H*8> ! 1
00'55' 11E0                     .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
  46  11E2                      .BYTE   ^O100 ! <B*8> ! M
  C5  11E3                      .BYTE   ^O305 ! <8*B>
20 3E 11E4                      .BYTE   <A*8> ! 6 , 32&255
  77  11E6                      .BYTE   ^O100 ! <M*8> ! A
      11E7          2050$:
      11E7  3222
      11E7  3223                LXI     H,DATA_SHIFT+4                ;SHIFT DATA INTO CPU
  21  11E7                      .BYTE   <H*8> ! 1
00'AC' 11E8                     .BYTE   <DATA_SHIFT+4-HEAD>&255 , <<DATA_SHIFT+4-HEAD>&^XFF00>/256
      11EA  3224                MVI     C,4
04 0E 11EA                      .BYTE   <C*8> ! 6 , 4&255
      11EC  3225                CALL    SHIFTER1_R
  CD  11EC                      .BYTE   ^O315
0EC1' 11ED                      .WORD   <SHIFTER1_R-HEAD>
      11EF  3226
      11EF  3227                IFC
  D2  11EF                      .BYTE   ^O322
11FB' 11F0                      .WORD   <2051$-HEAD>
      11F2  3228                LXI     H,X_ROT_L                     ;ADDR OF CSR INSTS TO EXECUTE
  21  11F2                      .BYTE   <H*8> ! 1
```

```
                    00'79' 11F3                  .BYTE   <X_ROT_L-HEAD>&255 , <<X_ROT_L-HEAD>&^XFF00>/256
                           11F5  3229                    CALL    PERFORM_CSR_R           ;MAKE A 1 BY END AROUND CARRY
                      CD   11F5                  .BYTE   ^0315
                    12F2'  11F6                  .WORD   <PERFORM_CSR_R-HEAD>
                           11F8  3230
                           11F8  3231                    ELSE
                      C3   11F8                  .BYTE   ^0303
                    11FE'  11F9                  .WORD   <2052$-HEAD>
                           11FB        2051$:                                        ;GENERATE LOCAL SYMBOL NAME
                           11FB  3232                    CALL    SHIFT_L_2901_R          ;MAKE A 0 BY SHIFT IN ZERO
                      CD   11FB                  .BYTE   ^0315
                    1249'  11FC                  .WORD   <SHIFT_L_2901_R-HEAD>
                           11FE  3233                    ENDIF
                           11FE        2052$:                                        ;GENERATE LCL 2052 NAME
                           11FE  3234
                           11FE  3235                    ENDDO
                      21   11FE                  .BYTE   <H*8> ! 1
                    00'55' 11FF                  .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
                      35   1201                  .BYTE   ^05 ! <8*M>
                      C2   1202                  .BYTE   ^0302
                    11E7'  1203                  .WORD   <2050$-HEAD>
                      C1   1205                  .BYTE   ^0301 ! <8*B>
                      70   1206                  .BYTE   ^0100 ! <M*8> ! B
                           1207  3236
                           1207  3237                    ELSE
                      C3   1207                  .BYTE   ^0303
                    1248'  1208                  .WORD   <2053$-HEAD>
                           120A        2049$:                                        ;GENERATE LOCAL SYMBOL NAME
                           120A  3238
                           120A  3239                    LXI     H,WCS_DAT_32_ADD        ;FAST WRITE
                      21   120A                  .BYTE   <H*8> ! 1
                    06 00  120B                  .BYTE   <WCS_DAT_32_ADD-HEAD>&255 , <<WCS_DAT_32_ADD-HEAD>&^XFF00>/256
                           120D  3240                    MOV     A,H
                      7C   120D                  .BYTE   ^0100 ! <A*8> ! H
                           120E  3241                    MOV     H,L
                      65   120E                  .BYTE   ^0100 ! <H*8> ! L
                           120F  3242                    MOV     L,A
                      6F   120F                  .BYTE   ^0100 ! <L*8> ! A
                           1210  3243                    SHLD    UPC_VALUE
                      22   1210                  .BYTE   ^042
                    00C2'  1211                  .WORD   <UPC_VALUE-HEAD>
                           1213  3244                    CALL    LOAD_UPC_R
                      CD   1213                  .BYTE   ^0315
                    0F19'  1214                  .WORD   <LOAD_UPC_R-HEAD>
                           1216  3245
                           1216  3246                    CALL    NOP_CSR_R               ;SET UP SUBROUTINE
                      CD   1216                  .BYTE   ^0315
                    0ECA'  1217                  .WORD   <NOP_CSR_R-HEAD>
                           1219  3247
                           1219  3248                    CALL    MICRO_STEP_CPU_R        ;EXEC INIT FOR SUB.
                      CD   1219                  .BYTE   ^0315
                    12A0'  121A                  .WORD   <MICRO_STEP_CPU_R-HEAD>
                           121C  3249                    CALL    MICRO_STEP_CPU_R
                      CD   121C                  .BYTE   ^0315
                    12A0'  121D                  .WORD   <MICRO_STEP_CPU_R-HEAD>
                           121F  3250
                           121F  3251                    DO      32
```

```
                    21  121F                .BYTE   <H*8> ! 1
                 00'55' 1220                .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
                    46  1222                .BYTE   ^O100 ! <B*8> ! M
                    C5  1223                .BYTE   ^O305 ! <8*B>
                 20 3E  1224                .BYTE   <A*8> ! 6 , 32&255
                    77  1226                .BYTE   ^O100 ! <M*8> ! A
                        1227        2054$:
                        1227  3252
                        1227  3253                LXI     H,DATA_SHIFT+4          ;POINT TO END OF DATA AREA
                    21  1227                .BYTE   <H*8> ! 1
                 00'AC' 1228                .BYTE   <DATA_SHIFT+4-HEAD>&255 , <<DATA_SHIFT+4-HEAD>&^XFF00>/256
                        122A  3254                MVI     C,4                     ;BYTES TO SHIFT IN REG C
                 04 0E  122A                .BYTE   <C*8> ! 6 , 4&255
                        122C  3255                CALL    SHIFTER1_R
                    CD  122C                .BYTE   ^O315
                  0EC1' 122D                .WORD   <SHIFTER1_R-HEAD>
                        122F  3256
                        122F  3257                OUT     CLRATN                  ;SET A ZERO TO BE READ
                 AB D3  122F                .BYTE   ^O323 , CLRATN&255      ;register A
                        1231  3258
                        1231  3259                IFC
                    D2  1231                .BYTE   ^O322
                  1236' 1232                .WORD   <2055$-HEAD>
                        1234  3260                OUT     SETATN                  ;SET A ONE TO BE READ
                 AA D3  1234                .BYTE   ^O323 , SETATN&255      ;register A
                        1236  3261                ENDIF
                        1236        2055$:                                  ;GENERATE LCL 2055 NAME
                        1236  3262
                        1236  3263                CALL    MICRO_STEP_CPU_R        ;CYCLE THROUGH LOOP IN SUB.
                    CD  1236                .BYTE   ^O315
                 12A0' 1237                .WORD   <MICRO_STEP_CPU_R-HEAD>
                        1239  3264                CALL    MICRO_STEP_CPU_R
                    CD  1239                .BYTE   ^O315
                 12A0' 123A                .WORD   <MICRO_STEP_CPU_R-HEAD>
                        123C  3265                CALL    MICRO_STEP_CPU_R
                    CD  123C                .BYTE   ^O315
                 12A0' 123D                .WORD   <MICRO_STEP_CPU_R-HEAD>
                        123F  3266
                        123F  3267                ENDDO
                    21  123F                .BYTE   <H*8> ! 1
                 00'55' 1240                .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
                    35  1242                .BYTE   ^O5 ! <8*M>
                    C2  1243                .BYTE   ^O302
                  1227' 1244                .WORD   <2054$-HEAD>
                    C1  1246                .BYTE   ^O301 ! <8*B>
                    70  1247                .BYTE   ^O100 ! <M*8> ! B
                        1248  3268
                        1248  3269                ENDIF
                        1248        2053$:                                  ;GENERATE LCL 2053 NAME
                        1248  3270
                        1248  3271                RET
                    C9  1248                .BYTE   ^O311
                        1249  3272
                        1249  3273
                        1249  3274 ;*****************************************************************************
                        1249  3275 ;
                        1249  3276 ;  SHIFT_L_2901_R
```

```
                     1249  3277 ;  THIS ROUTINE SHIFTS THE DATA IN THE 2901 LEFT 1 BIT THIS INCLUDES
                     1249  3278 ;  SHIFTING A ZERO INTO THE LSB
                     1249  3279 ;
                     1249  3280 ;*******************************************************************************
                     1249  3281
                     1249  3282 SHIFT_L_2901_R: LXI     H,X_SHIFT_L             ;ADDR OF CSR INSTS TO EXECUTE
          21         1249                .BYTE   <H*8> ! 1
     00'75'          124A                .BYTE   <X_SHIFT_L-HEAD>&255 , <<X_SHIFT_L-HEAD>&^XFF00>/256
                     124C  3283                  CALL    PERFORM_CSR_R
          CD         124C                .BYTE   ^O315
     12F2'           124D                .WORD   <PERFORM_CSR_R-HEAD>
                     124F  3284
                     124F  3285                  RET
          C9         124F                .BYTE   ^O311
                     1250  3286
                     1250  3287 ;*******************************************************************************
                     1250  3288 ;
                     1250  3289 ;  SHIFT_R_2901_R
                     1250  3290 ;  THIS ROUTINE SHIFTS THE DATA IN THE 2901 RIGHT 8 BITS AND LEAVES
                     1250  3291 ;  THE LEAST SIG. BYTE ON THE Y-BUS TO BE READ
                     1250  3292 ;
                     1250  3293 ;*******************************************************************************
                     1250  3294
                     1250  3295 SHIFT_R_2901_R: LXI     H,X_SHIFT_R             ;ADDR OF CSR INSTS TO EXECUTE
          21         1250                .BYTE   <H*8> ! 1
     00'85'          1251                .BYTE   <X_SHIFT_R-HEAD>&255 , <<X_SHIFT_R-HEAD>&^XFF00>/256
                     1253  3296                  CALL    PERFORM_CSR_R
          CD         1253                .BYTE   ^O315
     12F2'           1254                .WORD   <PERFORM_CSR_R-HEAD>
                     1256  3297
                     1256  3298                  RET
          C9         1256                .BYTE   ^O311
                     1257  3299
                     1257  3300 ;*******************************************************************************
                     1257  3301 ;
                     1257  3302 ;  MICRO_STEPER
                     1257  3303 ;  THIS ROUTINE SINGLE STEPS THE CPU. IF THE MACHINE IS IN SPACE BAR
                     1257  3304 ;  MODE ANY CHAR OTHER THAN SPACE WILL CLEAR EXECUTE MODE AND STOP
                     1257  3305 ;  MICRO STEPING.  IF THE MACHINE IS IN COUNT MODE IT WILL DECREMENT
                     1257  3306 ;  THE COUNT AND STOP MICRO STEPING AT 0 COUNT
                     1257  3307 ;
                     1257  3308 ;*******************************************************************************
                     1257  3309
                     1257  3310 MICRO_STEPER:    IFEQI   MICRO_STEP,1            ;SPACE BAR MODE
          3A         1257                .BYTE   ^O72
     038A'           1258                .WORD   <MICRO_STEP-HEAD>
          FE         125A                .BYTE   ^O376
          01         125B                .BYTE   1
          C2         125C                .BYTE   ^O302
     1285'           125D                .WORD   <2056$-HEAD>
                     125F  3311
                     125F  3312 1$:                CALL    RSVECT                  ;CALL READ SILO ROUTINE IN
          CD         125F                .BYTE   ^O315
     413D            1260                .WORD   <RSVECT-HEAD>
                     1262  3313                                                  ; BOOT BLOCK.  IF CHAR FOUND,
                     1262  3314                                                  ; CARRY IS SET AND CHAR IS IN
                     1262  3315                                                  ; ACCUMULATOR
```

```
                     1262  3316
                     1262  3317                JNC     1$                      ;LOOP UNTIL CHAR RECEIVED
          D2         1262                .BYTE   ^0322
       125F'         1263                .WORD   <1$-HEAD>
                     1265  3318
                     1265  3319                ANI     HEX_7F                  ;STRIP ASCII PARITY BIT
          E6         1265                .BYTE   ^0346
          7F         1266                .BYTE   HEX_7F
                     1267  3320                CPI     SPACE_A                 ;SPACE?
          FE         1267                .BYTE   ^0376
          20         1268                .BYTE   SPACE_A
                     1269  3321                JZ      2$                      ;JUMP TO SINGLE STEP IF SPACE
          CA         1269                .BYTE   ^0312
       127C'         126A                .WORD   <2$-HEAD>
                     126C  3322
                     126C  3323                CALL    STOP_CPU                ;STOP CPU, SAVE CSR AND UPC
          CD         126C                .BYTE   ^0315
       10C9'         126D                .WORD   <STOP_CPU-HEAD>
                     126F  3324                CLEAR   MICRO_STEP              ;CLEAR FLAG
          AF         126F                .BYTE   ^0250 ! A               ;with A
          32         1270                .BYTE   ^062
       038A'         1271                .WORD   <MICRO_STEP-HEAD>
                     1273  3325                CALL    SAVE_WR                 ;SAVE CPU WORKING REGS
          CD         1273                .BYTE   ^0315
       0C11'         1274                .WORD   <SAVE_WR-HEAD>
                     1276  3326                CALL    SET_FOR_CONT_R          ;NO SPACE, END MICRO STEPPING
          CD         1276                .BYTE   ^0315
       107B'         1277                .WORD   <SET_FOR_CONT_R-HEAD>
                     1279  3327                JMP     RESTART_CPU             ;CONTINUE COMES HERE
          C3         1279                .BYTE   ^0303
       110E'         127A                .WORD   <RESTART_CPU-HEAD>
                     127C  3328
                     127C  3329 2$:              CALL    GSVECT                  ;CALL GET SILO TO POP OUT SPACE
          CD         127C                .BYTE   ^0315
        4145         127D                .WORD   <GSVECT-HEAD>
                     127F  3330                CALL    MICRO_STEP_CPU_R        ;STEP MACHINE AND PRINT ADDRESS
          CD         127F                .BYTE   ^0315
       12A0'         1280                .WORD   <MICRO_STEP_CPU_R-HEAD>
                     1282  3331
                     1282  3332                ELSE
          C3         1282                .BYTE   ^0303
       129F'         1283                .WORD   <2057$-HEAD>
                     1285        2056$:                                      ;GENERATE LOCAL SYMBOL NAME
                     1285  3333
                     1285  3334                CALL    MICRO_STEP_CPU_R        ;COUNT MODE
          CD         1285                .BYTE   ^0315
       12A0'         1286                .WORD   <MICRO_STEP_CPU_R-HEAD>
                     1288  3335                LXI     H,STEP_CNT              ;DECREMENT STEP COUNT
          21         1288                .BYTE   <H*8> ! 1
     04'09'          1289                .BYTE   <STEP_CNT-HEAD>&255 , <<STEP_CNT-HEAD>&^XFF00>/256
                     128B  3336                CALL    DECR_WORD_R
          CD         128B                .BYTE   ^0315
       12BC'         128C                .WORD   <DECR_WORD_R-HEAD>
                     128E  3337                RNZ                             ;RETURN  UNTIL COUNT EXHAUSTED
          C0         128E                .BYTE   ^0300
                     128F  3338
                     128F  3339                CALL    STOP_CPU                ;STOP CPU, SAVE CSR AND UPC
```

```
                CD  128F                .BYTE   ^0315
              10C9' 1290                .WORD   <STOP_CPU-HEAD>
                    1292  3340                  CLEAR   MICRO_STEP                ;CLEAR FLAG
                AF  1292                .BYTE   ^0250 ! A               ;with A
                32  1293                .BYTE   ^062
              038A' 1294                .WORD   <MICRO_STEP-HEAD>
                    1296  3341                  CALL    SAVE_WR                   ;SAVE CPU WORKING REGS
                CD  1296                .BYTE   ^0315
              0C11' 1297                .WORD   <SAVE_WR-HEAD>
                    1299  3342                  CALL    SET_FOR_CONT_R            ;NO SPACE, END MICRO STEPPING
                CD  1299                .BYTE   ^0315
              107B' 129A                .WORD   <SET_FOR_CONT_R-HEAD>
                    129C  3343                  JMP     RESTART_CPU               ;CONTINUE COMES HERE
                C3  129C                .BYTE   ^0303
              110E' 129D                .WORD   <RESTART_CPU-HEAD>
                    129F  3344
                    129F  3345                  ENDIF
                    129F      2057$:                                              ;GENERATE LCL 2057 NAME
                    129F  3346
                    129F  3347                  RET
                C9  129F                .BYTE   ^0311
                    12A0  3348
                    12A0  3349
                    12A0  3350 ;*****************************************************************************
                    12A0  3351 ;
                    12A0  3352 ;  MICRO_STEP_CPU_R
                    12A0  3353 ;  THIS ROUTINE CAUSES THE MACHINE TO DO ONE MICRO INSTRUCTION
                    12A0  3354 ;
                    12A0  3355 ;*****************************************************************************
                    12A0  3356
                    12A0  3357 MICRO_STEP_CPU_R:
                    12A0  3358                  OUT     SETSS
              23 D3 12A0                .BYTE   ^0323 , SETSS&255         ;register A
                    12A2  3359                  OUT     CLRSS                     ;SINGLE STEP CPU
              22 D3 12A2                .BYTE   ^0323 , CLRSS&255         ;register A
                    12A4  3360
                    12A4  3361                  RET
                C9  12A4                .BYTE   ^0311
                    12A5  3362
                    12A5  3363
                    12A5  3364 ;*****************************************************************************
                    12A5  3365 ;
                    12A5  3366 ;  INC_TEMPW_5
                    12A5  3367 ;  THIS ROUTINE INCREMENTS THE VALUE OF TEMP_WORD BY 5
                    12A5  3368 ;
                    12A5  3369 ;*****************************************************************************
                    12A5  3370
                    12A5  3371 INC_TEMPW_5:     MVI     B,0
              00 06 12A5                .BYTE   <B*8> ! 6 , 0&255
                    12A7  3372                  MVI     C,5
              05 0E 12A7                .BYTE   <C*8> ! 6 , 5&255
                    12A9  3373                  LHLD    TEMP_WORD
                2A  12A9                .BYTE   ^052
              0419' 12AA                .WORD   <TEMP_WORD-HEAD>
                    12AC  3374                  DAD     B                         ;ADD 5 TO TEMP_WORD
                09  12AC                .BYTE   <B*8> ! 9
                    12AD  3375                  SHLD    TEMP_WORD                 ;TO MOVE TO NEXT FLAG NAME
```

```
                22  12AD                .BYTE   ^042
              0419' 12AE                .WORD   <TEMP_WORD-HEAD>
                    12B0  3376
                    12B0  3377                  RET
                C9  12B0                .BYTE   ^0311
                    12B1  3378
                    12B1  3379
                    12B1  3380 ;*****************************************************************************
                    12B1  3381 ;
                    12B1  3382 ;  INC_WORD_R
                    12B1  3383 ;  THIS ROUTINE INCREMENTS A WORD OF DATA
                    12B1  3384 ;
                    12B1  3385 ;  INPUT CONDITIONS:
                    12B1  3386 ;       H+L = POINTER TO THE WORD TO INCREMENT
                    12B1  3387 ;  OUTPUT CONDITIONS:
                    12B1  3388 ;       CONDITION CODES
                    12B1  3389 ;
                    12B1  3390 ;*****************************************************************************
                    12B1  3391
                    12B1  3392 INC_WORD_R:      INX     H                       ;POINT TO LOW ORDER PART
                23  12B1                .BYTE   <H*8> ! 3
                    12B2  3393
                    12B2  3394                  MOV     A,M
                7E  12B2                .BYTE   ^0100 ! <A*8> ! M
                    12B3  3395                  ADI     1
                C6  12B3                .BYTE   ^0306
                01  12B4                .BYTE   1
                    12B5  3396                  MOV     M,A                     ;INC VALUE IN LOW ORD. PART
                77  12B5                .BYTE   ^0100 ! <M*8> ! A
                    12B6  3397                  DCX     H                       ;MOVE TO HIGH ORDER PART
                2B  12B6                .BYTE   <H*8> ! 11
                    12B7  3398
                    12B7  3399                  MOV     A,M
                7E  12B7                .BYTE   ^0100 ! <A*8> ! M
                    12B8  3400                  ACI     0                       ;ADD ZERO + CARRY
                CE  12B8                .BYTE   ^0316
                00  12B9                .BYTE   0
                    12BA  3401                  MOV     M,A
                77  12BA                .BYTE   ^0100 ! <M*8> ! A
                    12BB  3402
                    12BB  3403                  RET
                C9  12BB                .BYTE   ^0311
                    12BC  3404
                    12BC  3405
                    12BC  3406 ;*****************************************************************************
                    12BC  3407 ;
                    12BC  3408 ;  DECR_WORD_R
                    12BC  3409 ;  THIS ROUTINE DECREMENTS A WORD OF DATA
                    12BC  3410 ;
                    12BC  3411 ;  INPUT CONDITIONS:
                    12BC  3412 ;       H+L = POINTER TO THE WORD TO DECREMENT
                    12BC  3413 ;  OUTPUT CONDITIONS:
                    12BC  3414 ;       CONDITION CODES
                    12BC  3415 ;
                    12BC  3416 ;*****************************************************************************
                    12BC  3417
                    12BC  3418 DECR_WORD_R:     INX     H                       ;MOVE TO SECOND BYTE
```

```
                                                .BYTE   <H*8> ! 3
                        12BD    3419
                        12BD    3420            MOV     A,M
                7E      12BD                    .BYTE   ^0100 ! <A*8> ! M
                        12BE    3421            SUI     1                       ;DEC VALUE IN LOW ORD. PART
                D6      12BE                    .BYTE   ^0326
                01      12BF                    .BYTE   1
                        12C0    3422            MOV     M,A
                77      12C0                    .BYTE   ^0100 ! <M*8> ! A
                        12C1    3423            DCX     H                       ;MOVE TO HIGH ORD. PART
                2B      12C1                    .BYTE   <H*8> ! 11
                        12C2    3424
                        12C2    3425            MOV     A,M
                7E      12C2                    .BYTE   ^0100 ! <A*8> ! M
                        12C3    3426            SBI     0                       ;SUB CARRY
                DE      12C3                    .BYTE   ^0336                   ;with borrow
                00      12C4                    .BYTE   0
                        12C5    3427            MOV     M,A
                77      12C5                    .BYTE   ^0100 ! <M*8> ! A
                        12C6    3428
                        12C6    3429            CPI     0
                FE      12C6                    .BYTE   ^0376
                00      12C7                    .BYTE   0
                        12C8    3430            RNZ                             ;CHECK FOR ZERO RESULT
                C0      12C8                    .BYTE   ^0300
                        12C9    3431            INX     H
                23      12C9                    .BYTE   <H*8> ! 3
                        12CA    3432            MOV     A,M
                7E      12CA                    .BYTE   ^0100 ! <A*8> ! M
                        12CB    3433            CPI     0
                FE      12CB                    .BYTE   ^0376
                00      12CC                    .BYTE   0
                        12CD    3434            RET                             ;SEND BACK COND. CODE
                C9      12CD                    .BYTE   ^0311
                        12CE    3435
                        12CE    3436
                        12CE    3437
                        12CE    3438 ;*******************************************************************************
                        12CE    3439 ;
                        12CE    3440 ;  WRITE_LS_STAT
                        12CE    3441 ;  THIS ROUTINE WRITES THE LS ERROR CONTROL WORD ONLY IF THE
                        12CE    3442 ;  WCS IMAGE IS LOADED
                        12CE    3443 ;
                        12CE    3444 ;  THE ROUTINE CAN BE ENTERED AT THE SET BIT OR CLEAR BIT ENTRY POINT.
                        12CE    3445 ;  IN THAT CASE, THE B REGISTER MUST BE LOADED WITH THE BIT TO SET OR CLEAR
                        12CE    3446 ;  BEFORE ENTRY TO THIS ROUTINE.
                        12CE    3447 ;
                        12CE    3448 ;*******************************************************************************
                        12CE    3449
                        12CE    3450 SET_LS_STAT:       LDA     FLAGS           ;GET CURRENT FLAGS
                3A      12CE                    .BYTE   ^072
              005A'     12CF                    .WORD   <FLAGS-HEAD>
                        12D1    3451            ORA     B                       ;OR IN THIS BIT TO SET FLAG
                B0      12D1                    .BYTE   ^0260 ! B
                        12D2    3452            STA     FLAGS                   ;STORE NEW FLAGS
                32      12D2                    .BYTE   ^062
              005A'     12D3                    .WORD   <FLAGS-HEAD>
```

```
                12D5  3453                    JMP     WRITE_LS_STAT           ;WRITE FLAGS IN LS
          C3    12D5                  .BYTE   ^O303
      12DF'     12D6                  .WORD   <WRITE_LS_STAT-HEAD>
                12D8  3454
                12D8  3455 CLEAR_LS_STAT:
                12D8  3456                    LDA     FLAGS                   ;GET CURRENT FLAGS
          3A    12D8                  .BYTE   ^O72
      005A'     12D9                  .WORD   <FLAGS-HEAD>
                12DB  3457                    ANA     B                       ;AND IN THIS BIT TO CLEAR FLAG
          A0    12DB                  .BYTE   ^O240 ! B
                12DC  3458                    STA     FLAGS                   ;STORE NEW FLAGS
          32    12DC                  .BYTE   ^O62
      005A'     12DD                  .WORD   <FLAGS-HEAD>
                12DF  3459
                12DF  3460 WRITE_LS_STAT:     IF      WCS_LOADED
          3A    12DF                  .BYTE   ^O72
      058B'     12E0                  .WORD   <WCS_LOADED-HEAD>
          0F    12E2                  .BYTE   ^O17
          D2    12E3                  .BYTE   ^O322
      12F1'     12E4                  .WORD   <2058$-HEAD>
                12E6  3461
                12E6  3462                    LDA     FLAGS                   ;GET STATUS FLAGS
          3A    12E6                  .BYTE   ^O72
      005A'     12E7                  .WORD   <FLAGS-HEAD>
                12E9  3463                    STA     DATA3
          32    12E9                  .BYTE   ^O62
      00AC'     12EA                  .WORD   <DATA3-HEAD>
                12EC  3464
                12EC  3465                    MVI     A,FLAG_LOC              ;WRITE TO LS ERROR CONTROL LOC.
       48 3E    12EC                  .BYTE   <A*8> ! 6 , FLAG_LOC&255
                12EE  3466                    CALL    WRITE_LS_R
          CD    12EE                  .BYTE   ^O315
       1146'    12EF                  .WORD   <WRITE_LS_R-HEAD>
                12F1  3467
                12F1  3468                    ENDIF
                12F1            2058$:                                        ;GENERATE LCL 2058 NAME
                12F1  3469
                12F1  3470                    RET
          C9    12F1                  .BYTE   ^O311
                12F2  3471
                12F2  3472
                12F2  3473 ;*****************************************************************************
                12F2  3474 ;
                12F2  3475 ;  PERFORM_CSR_R
                12F2  3476 ;  THIS ROUTINE WILL PUT INSTRUCTIONS IN TO THE CSR AND EXECUTE THEM
                12F2  3477 ;
                12F2  3478 ;  INPUT CONDITIONS:
                12F2  3479 ;       H+L = POINTER TO INSTRUCTION TABLE
                12F2  3480 ;                TABLE = CNT,INST,INST,.....
                12F2  3481 ;
                12F2  3482 ;*****************************************************************************
                12F2  3483
                12F2  3484 PERFORM_CSR_R:
                12F2  3485                    MOV     A,M                     ;GET COUNT
          7E    12F2                  .BYTE   ^O100 ! <A*8> ! M
                12F3  3486                    STA     CSR_CNT                 ;AND SAVE
          32    12F3                  .BYTE   ^O62
```

```
                025D' 12F4                .WORD   <CSR_CNT-HEAD>
                      12F6  3487
                      12F6  3488                  INX     H
                  23  12F6                .BYTE   <H*8> ! 3
                      12F7  3489                  SHLD    TEMP_WORD
                  22  12F7                .BYTE   ^042
                0419' 12F8                .WORD   <TEMP_WORD-HEAD>
                      12FA  3490
                      12FA  3491 1$:              LHLD    TEMP_WORD
                  2A  12FA                .BYT    ^052
                0419' 12FB                .WORD   <TEMP_WORD-HEAD>
                      12FD  3492                  LXI     D,CSR_VALUE             ;MOVE 3 BYTES OF INST TO CSR_VALUE
                  11  12FD                .BYTE   <D*8> ! 1
              00'B8'  12FE                .BYTE   <CSR_VALUE-HEAD>&255 , <<CSR_VALUE-HEAD>&^XFF00>/256
                      1300  3493                  CALL    MOVER_3_R
                  CD  1300                .BYTE   ^0315
                0F5F' 1301                .WORD   <MOVER_3_R-HEAD>
                      1303  3494                  SHLD    TEMP_WORD
                  22  1303                .BYTE   ^042
                0419' 1304                .WORD   <TEMP_WORD-HEAD>
                      1306  3495
                      1306  3496                  CALL    LOAD_CSR_R              ;PUT INST IN CSR
                  CD  1306                .BYTE   ^0315
                0EE7' 1307                .WORD   <LOAD_CSR_R-HEAD>
                      1309  3497
                      1309  3498                  OUT     SETSS
              23 D3   1309                .BYTE   ^0323 , SETSS&255       ;register A
                      130B  3499                  OUT     CLRSS                   ;SINGLE STEP CPU
              22 D3   130B                .BYTE   ^0323 , CLRSS&255       ;register A
                      130D  3500
                      130D  3501                  LXI     H,CSR_CNT
                  21  130D                .BYTE   <H*8> ! 1
              02'5D'  130E                .BYTE   <CSR_CNT-HEAD>&255 , <<CSR_CNT-HEAD>&^XFF00>/256
                      1310  3502                  DCR     M                       ;DECREMENT THE COUNT
                  35  1310                .BYTE   ^05 ! <8*M>
                      1311  3503                  JNZ     1$                      ;LOOP UNITL ZERO
                  C2  1311                .BYTE   ^0302
                12FA' 1312                .WORD   <1$-HEAD>
                      1314  3504
                      1314  3505                  LXI     H,X_MOV_WRWR            ;LEAVE MOV INST IN CSR TO
                  21  1314                .BYTE   <H*8> ! 1
              00'9E'  1315                .BYTE   <X_MOV_WRWR-HEAD>&255 , <<X_MOV_WRWR-HEAD>&^XFF00>/256
                      1317  3506                  LXI     D,CSR_VALUE             ;ENABLE Y-BUS
                  11  1317                .BYTE   <D*8> ! 1
              00'B8'  1318                .BYTE   <CSR_VALUE-HEAD>&255 , <<CSR_VALUE-HEAD>&^XFF00>/256
                      131A  3507                  CALL    MOVER_3_R
                  CD  131A                .BYTE   ^0315
                0F5F' 131B                .WORD   <MOVER_3_R-HEAD>
                      131D  3508
                      131D  3509                  CALL    LOAD_CSR_R              ;LOAD MOV
                  CD  131D                .BYTE   ^0315
                0EE7' 131E                .WORD   <LOAD_CSR_R-HEAD>
                      1320  3510
                      1320  3511                  RET
                  C9  1320                .BYTE   ^0311
                      1321  3512
                      1321  3513
```

```
                     1321   3514 ;*****************************************************************************
                     1321   3515 ;
                     1321   3516 ;  ATTENTION
                     1321   3517 ;  THIS ATTENTION ROUTINE IS USED WHEN THE CPU IS EXECUTING TESTS(RUNNING)
                     1321   3518 ;  IT IS BASICLY AN IDLE LOOP FOR THE 8085.  DURING THE IDLE LOOP
                     1321   3519 ;  THE 8085 CHECKS FOR SPECIAL OCCURANCES SUCH AS ^C,^P,OR CPATTN
                     1321   3520 ;  IF CPATTN SIGNAL COMES HIGH IT MEANS THAT THE CPU MICRO CODED
                     1321   3521 ;  DIAGNOSTIC IS REQUESTING SOMETHING.  MICRO STEPPING IS DONE ONCE
                     1321   3522 ;  FOR EACH ITTERATION OF THE IDLE LOOP
                     1321   3523 ;
                     1321   3524 ;*****************************************************************************
                     1321   3525
                     1321   3526 ATTENTION:       CLEAR   DATA_XFER_FLG
       AF            1321                .BYTE   ^O250 ! A                   ;with A
       32            1322                .BYTE   ^O62
  025E'              1323                .WORD   <DATA_XFER_FLG-HEAD>
                     1325   3527                 CALL    RESET_TIMOUT               ;INIT TIMOUTS AND XFER INFO
       CD            1325                .BYTE   ^O315
  1932'              1326                .WORD   <RESET_TIMOUT-HEAD>
                     1328   3528
                     1328   3529                 WHILE   RUNNING
                     1328        4001$:  LDA     RUNNING
       3A            1328                .BYTE   ^O72
  0403'              1329                .WORD   <RUNNING-HEAD>
       0F            132B                .BYTE   ^O17
       D2            132C                .BYTE   ^O322
  137E'              132D                .WORD   <2059$-HEAD>
                     132F   3530
                     132F   3531                 LDA     MICRO_STEP                 ;GET MICRO_STEP FLAG
       3A            132F                .BYTE   ^O72
  038A'              1330                .WORD   <MICRO_STEP-HEAD>
                     1332   3532                 ORA     A                          ;SET CONDITION CODES
       B7            1332                .BYTE   ^O260 ! A
                     1333   3533                 CNZ     MICRO_STEPER               ;CALL IF MICRO STEP
       C4            1333                .BYTE   ^O304
  1257'              1334                .WORD   <MICRO_STEPER-HEAD>
                     1336   3534
                     1336   3535                 IN      CPATTN
    83 DB            1336                .BYTE   ^O333 , CPATTN&255      ;to register A
                     1338   3536                 RAR
       1F            1338                .BYTE   ^O37
                     1339   3537                 IFC                                ;IF CPU_ATTENTION
       D2            1339                .BYTE   ^O322
  1375'              133A                .WORD   <2060$-HEAD>
                     133C   3538
                     133C   3539                 MVI     A,1
    01 3E            133C                .BYTE   <A*8> ! 6 , 1&255
                     133E   3540                 STA     APT_MESSAGE_CODE+1         ;SET APT START FLAG
       32            133E                .BYTE   ^O62
  0011'              133F                .WORD   <APT_MESSAGE_CODE+1-HEAD>
                     1341   3541
                     1341   3542                 CALL    RESET_TIMOUT               ;CPU HEARD, SO RESET TIMOUT
       CD            1341                .BYTE   ^O315
  1932'              1342                .WORD   <RESET_TIMOUT-HEAD>
                     1344   3543                 CALL    STOP_CPU
       CD            1344                .BYTE   ^O315
  10C9'              1345                .WORD   <STOP_CPU-HEAD>
```

```
                     1347  3544
                     1347  3545                    LHLD    SAVED_UPC               ;SAVE UPC WHEN GOING TO
          2A         1347                  .BYTE   ^052
       00C4'         1348                  .WORD   <SAVED_UPC-HEAD>
                     134A  3546                    SHLD    UPC_SUB                 ;SUBROUTINE
          22         134A                  .BYTE   ^042
       00C6'         134B                  .WORD   <UPC_SUB-HEAD>
                     134D  3547
                     134D  3548                    CALL    SAVE_WR
          CD         134D                  .BYTE   ^0315
       0C11'         134E                  .WORD   <SAVE_WR-HEAD>
                     1350  3549
                     1350  3550                    CALL    CHECK_CNTLC             ;IF CONTROL C TYPED STOP
          CD         1350                  .BYTE   ^0315
       1073'         1351                  .WORD   <CHECK_CNTLC-HEAD>
                     1353  3551
                     1353  3552                    MVI     A,STATUS
       40 3E         1353                  .BYTE   <A*8> ! 6 , STATUS&255
                     1355  3553                    CALL    READ_LS_R               ;READ LS COMMAND + STATUS
          CD         1355                  .BYTE   ^0315
       1167'         1356                  .WORD   <READ_LS_R-HEAD>
                     1358  3554                    CALL    DO_COMMANDS
          CD         1358                  .BYTE   ^0315
       137F'         1359                  .WORD   <DO_COMMANDS-HEAD>
                     135B  3555
                     135B  3556                    IFN     EOS_FLG                 ;IF EOS DONT RESTART
          3A         135B                  .BYTE   ^072
       0056'         135C                  .WORD   <EOS_FLG-HEAD>
          0F         135E                  .BYTE   ^017
          DA         135F                  .BYTE   ^0332
       1375'         1360                  .WORD   <2061$-HEAD>
                     1362  3557
                     1362  3558                    IFN     TEST_ZERO               ;IF TEST 0 DONT RESTART
          3A         1362                  .BYTE   ^072
       0447'         1363                  .WORD   <TEST_ZERO-HEAD>
          0F         1365                  .BYTE   ^017
          DA         1366                  .BYTE   ^0332
       1375'         1367                  .WORD   <2062$-HEAD>
                     1369  3559                    CALL    RESTORE_WR
          CD         1369                  .BYTE   ^0315
       0C0A'         136A                  .WORD   <RESTORE_WR-HEAD>
                     136C  3560                    LHLD    UPC_SUB                 ;RETURN FROM SUBROUTINE
          2A         136C                  .BYTE   ^052
       00C6'         136D                  .WORD   <UPC_SUB-HEAD>
                     136F  3561                    SHLD    STARTING_UPC            ;START CPU AGAIN
          22         136F                  .BYTE   ^042
       0407'         1370                  .WORD   <STARTING_UPC-HEAD>
                     1372  3562                    CALL    START_CPU
          CD         1372                  .BYTE   ^0315
       10FF'         1373                  .WORD   <START_CPU-HEAD>
                     1375  3563                    ENDIF
                     1375        2062$:                                          ;GENERATE LCL 2062 NAME
                     1375  3564
                     1375  3565                    ENDIF
                     1375        2061$:                                          ;GENERATE LCL 2061 NAME
                     1375  3566
                     1375  3567                    ENDIF
```

```
                1375        2060$:                                          ;GENERATE LCL 2060 NAME
                1375  3568
                1375  3569                  CALL    GCVECT                  ;CHECK FOR OPERATOR REQUEST (^P)
      CD        1375                  .BYTE   ^O315
    4129        1376                  .WORD   <GCVECT-HEAD>
                1378  3570                  CALL    TIMOUT                  ;CHECK FOR TIMOUT
      CD        1378                  .BYTE   ^O315
    1907'       1379                  .WORD   <TIMOUT-HEAD>
                137B  3571
                137B  3572                  ENDWHILE
      C3        137B                  .BYTE   ^O303
    1328'       137C                  .WORD   <4001$-HEAD>
                137E        2059$:                                          ;GENERATE LCL 2059 NAME
                137E  3573
                137E  3574                  RET
      C9        137E                  .BYTE   ^O311
                137F  3575
                137F  3576
                137F  3577  ;*****************************************************************************
                137F  3578  ;
                137F  3579  ;  DO_COMMANDS
                137F  3580  ;  THIS ROUTINE DECODES THE LS COMMAND AND STATUS WORD AND CALLS ROUTINES
                137F  3581  ;  TO PERFORM THE FUNCTIONS WHERE NECESSARY.
                137F  3582  ;
                137F  3583  ;*****************************************************************************
                137F  3584
                137F  3585  DO_COMMANDS:    LXI     D,COMMAND0              ;DESTINATION
      11        137F                  .BYTE   <D*8> ! 1
   00'AF'       1380                  .BYTE   <COMMAND0-HEAD>&255 , <<COMMAND0-HEAD>&^XFF00>/256
                1382  3586                  CALL    MOV_FROM_DATA0          ;MOVE 4 BYTES OF DATA FROM
      CD        1382                  .BYTE   ^O315
    0F70'       1383                  .WORD   <MOV_FROM_DATA0-HEAD>
                1385  3587                                                  ; DATA0
                1385  3588                  IFMBI   COMMAND3,BAD_STATUS     ;BIT 4 = BAD RL02 DISK STATUS
      3A        1385                  .BYTE   ^O72
    00B2'       1386                  .WORD   <COMMAND3-HEAD>
      E6        1388                  .BYTE   ^O346
      10        1389                  .BYTE   BAD_STATUS
      CA        138A                  .BYTE   ^O312
    1395'       138B                  .WORD   <2063$-HEAD>
                138D  3589                  SET     PACK_ERROR              ;SET FLAG INDICATING PACK ERROR
      AF        138D                  .BYTE   ^O250 ! A               ;with A
      3C        138E                  .BYTE   ^O4 ! <8*A>
      32        138F                  .BYTE   ^O062
    03D5'       1390                  .WORD   <PACK_ERROR-HEAD>
                1392  3590                  CALL    PACK_WARNING
      CD        1392                  .BYTE   ^O315
    165B'       1393                  .WORD   <PACK_WARNING-HEAD>
                1395  3591
                1395  3592                  ENDIF
                1395        2063$:                                          ;GENERATE LCL 2063 NAME
                1395  3593
                1395  3594                  IFMBI   COMMAND3,WRONG_LAB      ;BIT 5 = CRD PACK NOT IN RL02
      3A        1395                  .BYTE   ^O72
    00B2'       1396                  .WORD   <COMMAND3-HEAD>
      E6        1398                  .BYTE   ^O346
      20        1399                  .BYTE   WRONG_LAB
```

```
                      CA  139A                .BYTE   ^O312
                    13A5' 139B                .WORD   <2064$-HEAD>
                          139D  3595                  SET     PACK_ERROR              ;SET FLAG INDICATING PACK ERROR
                      AF  139D                .BYTE   ^O250 ! A               ;with A
                      3C  139E                .BYTE   ^O04 ! <8*A>
                      32  139F                .BYTE   ^O62
                    03D5' 13A0                .WORD   <PACK_ERROR-HEAD>
                          13A2  3596                  CALL    PACK_WARNING
                      CD  13A2                .BYTE   ^O315
                    165B' 13A3                .WORD   <PACK_WARNING-HEAD>
                          13A5  3597
                          13A5  3598                  ENDIF
                          13A5        2064$:                                      ;GENERATE LCL 2064 NAME
                          13A5  3599
                          13A5  3600                  IFMBI   COMMAND3,UBE_PRES       ;BIT 6 = PRINT UBE PRESENT
                      3A  13A5                .BYTE   ^O72
                    00B2' 13A6                .WORD   <COMMAND3-HEAD>
                      E6  13A8                .BYTE   ^O346
                      40  13A9                .BYTE   UBE_PRES
                      CA  13AA                .BYTE   ^O312
                    13AD' 13AB                .WORD   <2065$-HEAD>
                          13AD  3601                  ENDIF
                          13AD        2065$:                                      ;GENERATE LCL 2065 NAME
                          13AD  3602
                          13AD  3603                  IFMBI   COMMAND3,R80_PRES       ;BIT 7 = PRINT R80 PRESENT
                      3A  13AD                .BYTE   ^O72
                    00B2' 13AE                .WORD   <COMMAND3-HEAD>
                      E6  13B0                .BYTE   ^O346
                      80  13B1                .BYTE   R80_PRES
                      CA  13B2                .BYTE   ^O312
                    13B5' 13B3                .WORD   <2066$-HEAD>
                          13B5  3604                  ENDIF
                          13B5        2066$:                                      ;GENERATE LCL 2066 NAME
                          13B5  3605
                          13B5  3606                  IFMBI   COMMAND2,WCSERR         ;BIT 8 = ERROR FOUND BY WCS
                      3A  13B5                .BYTE   ^O72
                    00B1' 13B6                .WORD   <COMMAND2-HEAD>
                      E6  13B8                .BYTE   ^O346
                      01  13B9                .BYTE   WCSERR
                      CA  13BA                .BYTE   ^O312
                    13C0' 13BB                .WORD   <2067$-HEAD>
                          13BD  3607                  CALL    WCS_ERROR
                      CD  13BD                .BYTE   ^O315
                    1442' 13BE                .WORD   <WCS_ERROR-HEAD>
                          13C0  3608                  ENDIF
                          13C0        2067$:                                      ;GENERATE LCL 2067 NAME
                          13C0  3609
                          13C0  3610                  IFMBI   COMMAND2,SETPAR_ERR     ;BIT 9 = SET PARITY ERROR
                      3A  13C0                .BYTE   ^O72
                    00B1' 13C1                .WORD   <COMMAND2-HEAD>
                      F6  13C3                .BYTE   ^O346
                      02  13C4                .BYTE   SETPAR_ERR
                      CA  13C5                .BYTE   ^O312
                    13D0' 13C6                .WORD   <2068$-HEAD>
                          13C8  3611                  SET     LS_SETPARERR            ;SET FLAG FOR USE LATER
                      AF  13C8                .BYTE   ^O250 ! A               ;with A
                      3C  13C9                .BYTE   ^O04 ! <8*A>
```

```
      32  13CA              .BYTE   ^062
   0348'  13CB              .WORD   <LS_SETPARERR-HEAD>
          13CD                      CALL    SET_PAR_INT             ;SET UP PARITY ERROR IN SPECIFIED
      CD  13CD              .BYTE   ^0315
   16C7'  13CE              .WORD   <SET_PAR_INT-HEAD>
          13D0                                                      ; WCS LOCATION
          13D0                      ENDIF
          13D0      2068$:                                          ;GENERATE LCL 2068 NAME
          13D0
          13D0                      IFMBI   COMMAND2,MM_SIZE        ;BIT 11 = PRINT MM SIZE
      3A  13D0              .BYTE   ^072
   00B1'  13D1              .WORD   <COMMAND2-HEAD>
      E6  13D3              .BYTE   ^0346
      08  13D4              .BYTE   MM_SIZE
      CA  13D5              .BYTE   ^0312
   13D8'  13D6              .WORD   <2069$-HEAD>
          13D8                      ENDIF
          13D8      2069$:                                          ;GENERATE LCL 2069 NAME
          13D8
          13D8                      IFMBI   COMMAND2,FPA_PRES       ;BIT 12 = PRINT FPA PRESENT
      3A  13D8              .BYTE   ^072
   00B1'  13D9              .WORD   <COMMAND2-HEAD>
      E6  13DB              .BYTE   ^0346
      10  13DC              .BYTE   FPA_PRES
      CA  13DD              .BYTE   ^0312
   13E3'  13DE              .WORD   <2070$-HEAD>
          13E0                      CALL    PRINT_MOD_PRES          ;PRINT PRESENT OR NOT PRESENT
      CD  13E0              .BYTE   ^0315
   1723'  13E1              .WORD   <PRINT_MOD_PRES-HEAD>
          13E3                      ENDIF
          13E3      2070$:                                          ;GENERATE LCL 2070 NAME
          13E3
          13E3                      IFMBI   COMMAND2,XFER           ;BIT 13 = DATA TRANSFER REQ
      3A  13E3              .BYTE   ^072
   00B1'  13E4              .WORD   <COMMAND2-HEAD>
      E6  13E6              .BYTE   ^0346
      20  13E7              .BYTE   XFER
      CA  13E8              .BYTE   ^0312
   13EE'  13E9              .WORD   <2071$-HEAD>
          13EB                      CALL    DATA_XFER
      CD  13EB              .BYTE   ^0315
   1996'  13EC              .WORD   <DATA_XFER-HEAD>
          13EE                      ENDIF
          13EE      2071$:                                          ;GENERATE LCL 2071 NAME
          13EE
          13EE                      IFMBI   COMMAND2,EOS            ;BIT 14 = END OF SECTION
      3A  13EE              .BYTE   ^072
   00B1'  13EF              .WORD   <COMMAND2-HEAD>
      E6  13F1              .BYTE   ^0346
      40  13F2              .BYTE   EOS
      CA  13F3              .BYTE   ^0312
   13FB'  13F4              .WORD   <2072$-HEAD>
          13F6                      SET     EOS_FLG
      AF  13F6              .BYTE   ^0250 ! A               ;with A
      3C  13F7              .BYTE   ^04 ! <8*A>
      32  13F8              .BYTE   ^062
   0056'  13F9              .WORD   <EOS_FLG-HEAD>
```

```
                    13FB  3629                          ENDIF
                    13FB        2072$:                                          ;GENERATE LCL 2072 NAME
                    13FB  3630
                    13FB  3631                          IFMBI   COMMAND2,BEG_TST        ;BIT 15 = BEGINING OF TEST
      3A            13FB                .BYTE   ^072
   00B1'            13FC                .WORD   <COMMAND2-HEAD>
      E6            13FE                .BYTE   ^0346
      80            13FF                .BYTE   BEG_TST
      CA            1400                .BYTE   ^0312
    1406'           1401                .WORD   <2073$-HEAD>
                    1403  3632                          CALL    BEGIN_OF_TEST
      CD            1403                .BYTE   ^0315
    1881'           1404                .WORD   <BEGIN_OF_TEST-HEAD>
                    1406  3633                          ENDIF
                    1406        2073$:                                          ;GENERATE LCL 2073 NAME
                    1406  3634
                    1406  3635                          IFMBI   COMMAND1,SIGNAL         ;BIT 16 = SET UP OF SIGNAL AS
      3A            1406                .BYTE   ^072
   00B0'            1407                .WORD   <COMMAND1-HEAD>
      E6            1409                .BYTE   ^0346
      01            140A                .BYTE   SIGNAL
      CA            140B                .BYTE   ^0312
    1411'           140C                .WORD   <2074$-HEAD>
                    140E  3636                          CALL    SET_SIGNALS
      CD            140E                .BYTE   ^0315
    1760'           140F                .WORD   <SET_SIGNALS-HEAD>
                    1411  3637                          ENDIF
                    1411        2074$:                                          ;GENERATE LCL 2074 NAME
                    1411  3638
                    1411  3639                          IFMBI   COMMAND0,LOOPING        ;SPECIAL CPU LOOPING CONTROL
      3A            1411                .BYTE   ^072
   00AF'            1412                .WORD   <COMMAND0-HEAD>
      E6            1414                .BYTE   ^0346
      02            1415                .BYTE   LOOPING
      CA            1416                .BYTE   ^0312
    141C'           1417                .WORD   <2075$-HEAD>
                    1419  3640                          CALL    LOOPER
      CD            1419                .BYTE   ^0315
    17DF'           141A                .WORD   <LOOPER-HEAD>
                    141C  3641                          ENDIF
                    141C        2075$:                                          ;GENERATE LCL 2075 NAME
                    141C  3642
                    141C  3643                          IFMBI   COMMAND0,SINGLE_ERR     ;BIT 26 = PRINT SINGLE BIT ERR
      3A            141C                .BYTE   ^072
   00AF'            141D                .WORD   <COMMAND0-HEAD>
      E6            141F                .BYTE   ^0346
      04            1420                .BYTE   SINGLE_ERR
      CA            1421                .BYTE   ^0312
    1427'           1422                .WORD   <2076$-HEAD>
                    1424  3644                          CALL    PRINT_SINGLE
      CD            1424                .BYTE   ^0315
    16EA'           1425                .WORD   <PRINT_SINGLE-HEAD>
                    1427  3645                          ENDIF
                    1427        2076$:                                          ;GENERATE LCL 2076 NAME
                    1427  3646
                    1427  3647                          IFMBI   COMMAND0,CLEARPAR_ERR   ;BIT 27 = CLEAR PARITY ERROR
      3A            1427                .BYTE   ^072
```

```
00AF'  1428                    .WORD   <COMMAND0-HEAD>
  E6   142A                    .BYTE   ^0346
  08   142B                    .BYTE   CLEARPAR_ERR
  CA   142C                    .BYTE   ^0312
1436'  142D                    .WORD   <2077$-HEAD>
       142F  3648                      CLEAR   LS_SETPARERR            ;CLEAR LS SET PARITY ERROR FLG
  AF   142F                    .BYTE   ^0250 ! A                ;with A
  32   1430                    .BYTE   ^062
0348'  1431                    .WORD   <LS_SETPARERR-HEAD>
       1433  3649                      CALL    CLEAR_PAR_INT           ;RESTORE GOOD PARITY AT ADDRESS
  CD   1433                    .BYTE   ^0315
16D9'  1434                    .WORD   <CLEAR_PAR_INT-HEAD>
       1436  3650                                                      ; SPECIFIED IN LS 7
       1436  3651                      ENDIF
       1436        2077$:                                            ;GENERATE LCL 2077 NAME
       1436  3652
       1436  3653                      IFMBI   COMMAND0,IDC_PRES       ;BIT 31 = PRINT IDC PRESENT
  3A   1436                    .BYTE   ^072
00AF'  1437                    .WORD   <COMMAND0-HEAD>
  E6   1439                    .BYTE   ^0346
  80   143A                    .BYTE   IDC_PRES
  CA   143B                    .BYTE   ^03T2
1441'  143C                    .WORD   <2078$-HEAD>
       143E  3654                      CALL    PRINT_MOD_PRES_IDC
  CD   143E                    .BYTE   ^0315
1742'  143F                    .WORD   <PRINT_MOD_PRES_IDC-HEAD>
       1441  3655                      ENDIF
       1441        2078$:                                            ;GENERATE LCL 2078 NAME
       1441  3656
       1441  3657                              RET
  C9   1441                            .BYTE   ^0311
       1442  3658
       1442  3659
       1442  3660
       1442  3661 ;*****************************************************************************
       1442  3662 ;
       1442  3663 ;  WCS_ERROR
       1442  3664 ;  THIS ROUTINE IS EXECUTED WHEN THE COMMAND AND STATUS WORD INDICATES
       1442  3665 ;  THAT THERE HAS BEEN AN ERROR.  IF BELL IS SET THE BELL IS RUNG
       1442  3666 ;  IT ALSO DECODES THE MODULES NAME FROM THE BITS AS FOLLOWS
       1442  3667 ;
       1442  3668 ;       BIT 0 - WCS M8395
       1442  3669 ;           1 - CPU M8390
       1442  3670 ;           2 - MCT M8391
       1442  3671 ;           3 - FPA M8389
       1442  3672 ;           4 - MEMORY ARRAY CARD
       1442  3673 ;           5 - IDC M8388
       1442  3674 ;
       1442  3675 ;  THIS ROUTINE IS ALSO USED BY THE 8085 BASED TESTS.  IN THAT CASE, THE
       1442  3676 ;  CONSOLE_TEST FLAG IS SET AND SOME DIFFERENT BRANCHES ARE TAKEN.
       1442  3677 ;
       1442  3678 ;*****************************************************************************
       1442  3679
       1442  3680 WCS_ERROR:
       1442  3681                      LXI     H,FAIL_A                ;PRINT *FAIL* MESSAGE
  21   1442                            .BYTE   <H*8> ! 1
02'EF' 1443                            .BYTE   <FAIL_A-HEAD>&255 , <<FAIL_A-HEAD>&^XFF00>/256
```

```
                     1445  3682                CALL    PRINT_STRING_R
                CD   1445                .BYTE   ^O315
            100E'    1446                .WORD   <PRINT_STRING_R-HEAD>
                     1448  3683
                     1448  3684                CALL    CRLF_R
                CD   1448                .BYTE   ^O315
            101A'    1449                .WORD   <CRLF_R-HEAD>
                     144B  3685                LXI     H,ABORT_A               ;PRINT ABORT MESSAGE
                21   144B                .BYTE   <H*8> ! 1
          01'64'     144C                .BYTE   <ABORT_A-HEAD>&255 , <<ABORT_A-HEAD>&^XFF00>/256
                     144E  3686                CALL    PRINT_STRING_R
                CD   144E                .BYTE   ^O315
            100E'    144F                .WORD   <PRINT_STRING_R-HEAD>
                     1451  3687
                     1451  3688                IFN     CONSOLE_TEST            ;GET VALUES FROM LS IF NOT
                3A   1451                .BYTE   ^O72
            0255'    1452                .WORD   <CONSOLE_TEST-HEAD>
                OF   1454                .BYTE   ^O17
                DA   1455                .BYTE   ^O332
            14B1'    1456                .WORD   <2079$-HEAD>
                     1458  3689                                            ; 8085 BASED CONSOLE TEST
                     1458  3690                MVI     A,WCS_ERR_NUM           ;GET ERROR NUMBER FROM LS
             41 3E   1458                .BYTE   <A*8> ! 6 , WCS_ERR_NUM&255
                     145A  3691                CALL    READ_LS_R
                CD   145A                .BYTE   ^O315
            1167'    145B                .WORD   <READ_LS_R-HEAD>
                     145D  3692                LDA     DATA3
                3A   145D                .BYTE   ^O72
            00AC'    145E                .WORD   <DATA3-HEAD>
                     1460  3693                STA     CON_ERR_NUM             ;STORE IN MEM
                32   1460                .BYTE   ^O62
            003F'    1461                .WORD   <CON_ERR_NUM-HEAD>
                     1463  3694
                     1463  3695                MVI     A,WCS_EXP_DAT           ;GET EXPECTED VALUE FROM LS
             42 3E   1463                .BYTE   <A*8> ! 6 , WCS_EXP_DAT&255
                     1465  3696                CALL    READ_LS_R
                CD   1465                .BYTE   ^O315
            1167'    1466                .WORD   <READ_LS_R-HEAD>
                     1468  3697                LXI     D,CON_EXP_DAT           ;DESTINATION
                11   1468                .BYTE   <D*8> ! 1
          00'40'     1469                .BYTE   <CON_EXP_DAT-HEAD>&255 , <<CON_EXP_DAT-HEAD>&^XFF00>/256
                     146B  3698                CALL    MOV_FROM_DATA0          ;MOVE 4 BYTES OF DATA FROM
                CD   146B                .BYTE   ^O315
            0F70'    146C                .WORD   <MOV_FROM_DATA0-HEAD>
                     146E  3699                                              ; DATA0
                     146E  3700
                     146E  3701                MVI     A,WCS_REC_DAT           ;GET RECEIVED VALUE FROM LS
             43 3E   146E                .BYTE   <A*8> ! 6 , WCS_REC_DAT&255
                     1470  3702                CALL    READ_LS_R
                CD   1470                .BYTE   ^O315
            1167'    1471                .WORD   <READ_LS_R-HEAD>
                     1473  3703                LXI     D,CON_REC_DAT           ;DESTINATION
                11   1473                .BYTE   <D*8> ! 1
          00'4F'     1474                .BYTE   <CON_REC_DAT-HEAD>&255 , <<CON_REC_DAT-HEAD>&^XFF00>/256
                     1476  3704                CALL    MOV_FROM_DATA0          ;MOVE 4 BYTES OF DATA FROM
                CD   1476                .BYTE   ^O315
            0F70'    1477                .WORD   <MOV_FROM_DATA0-HEAD>
```

```
                     1479  3705                                        ; DATA0
                     1479  3706
                     1479  3707                    MVI     A,WCS_ADDR_DAT          ;GET OTHER DATA FROM LS
          44 3E      1479                  .BYTE    <A*8> ! 6 , WCS_ADDR_DAT&255
                     147B  3708                    CALL    READ_LS_R
            CD       147B                  .BYTE    ^O315
         1167'       147C                  .WORD    <READ_LS_R-HEAD>
                     147E  3709                    LXI     D,CON_ADD_DAT           ;DESTINATION
            11       147E                  .BYTE    <D*8> ! 1
        00'3B'       147F                  .BYTE    <CON_ADD_DAT-HEAD>&255 , <<CON_ADD_DAT-HEAD>&^XFF00>/256
                     1481  3710                    CALL    MOV_FROM_DATA0          ;MOVE 4 BYTES OF DATA FROM
            CD       1481                  .BYTE    ^O315
         0F70'       1482                  .WORD    <MOV_FROM_DATA0-HEAD>
                     1484  3711                                        ; DATA0
                     1484  3712
                     1484  3713                    MVI     A,WCS_MSK_DAT           ;GET ERROR MASK FROM LS
          45 3E      1484                  .BYTE    <A*8> ! 6 , WCS_MSK_DAT&255
                     1486  3714                    CALL    READ_LS_R
            CD       1486                  .BYTE    ^O315
         1167'       1487                  .WORD    <READ_LS_R-HEAD>
                     1489  3715                    LXI     D,CON_MSK_DAT           ;DESTINATION
            11       1489                  .BYTE    <D*8> ! 1
        00'4B'       148A                  .BYTE    <CON_MSK_DAT-HEAD>&255 , <<CON_MSK_DAT-HEAD>&^XFF00>/256
                     148C  3716                    CALL    MOV_FROM_DATA0          ;MOVE 4 BYTES OF DATA FROM
            CD       148C                  .BYTE    ^O315
         0F70'       148D                  .WORD    <MOV_FROM_DATA0-HEAD>
                     148F  3717                                        ; DATA0
                     148F  3718
                     148F  3719                    CLEAR   NA_EXP_REC
            AF       148F                  .BYTE    ^O250 ! A               ;with A
            32       1490                  .BYTE    ^O62
         005F'       1491                  .WORD    <NA_EXP_REC-HEAD>
                     1493  3720                    IFMBI   COMMAND1,NA             ;SEE IF EXP AND REC ARE N/A
            3A       1493                  .BYTE    ^O72
         00B0'       1494                  .WORD    <COMMAND1-HEAD>
            E6       1496                  .BYTE    ^O346
            80       1497                  .BYTE    NA
            CA       1498                  .BYTE    ^O312
         14A0'       1499                  .WORD    <2080$-HEAD>
                     149B  3721                    SET     NA_EXP_REC              ;SET FLAG IF SO
            AF       149B                  .BYTE    ^O250 ! A               ;with A
            3C       149C                  .BYTE    ^O04 ! <8*A>
            32       149D                  .BYTE    ^O62
         005F'       149E                  .WORD    <NA_EXP_REC-HEAD>
                     14A0  3722                    ENDIF
                     14A0       2080$:                                     ;GENERATE LCL 2080 NAME
                     14A0  3723
                     14A0  3724                    SET     NA_OTHER
            AF       14A0                  .BYTE    ^O250 ! A               ;with A
            3C       14A1                  .BYTE    ^O04 ! <8*A>
            32       14A2                  .BYTE    ^O62
         0060'       14A3                  .WORD    <NA_OTHER-HEAD>
                     14A5  3725                    IFMBI   COMMAND0,ADDR           ;SEE IF OTHER DATA IS N/A
            3A       14A5                  .BYTE    ^O72
         00AF'       14A6                  .WORD    <COMMAND0-HEAD>
            E6       14A8                  .BYTE    ^O346
            01       14A9                  .BYTE    ADDR
```

```
                CA  14AA                .BYTE   ^0312
          14B1' 14AB                    .WORD   <2081$-HEAD>
                14AD  3726              CLEAR   NA_OTHER                ;CLEAR FLAG IF NOT N/A
                AF  14AD                .BYTE   ^0250 ! A               ;with A
                32  14AE                .BYTE   ^062
          0060' 14AF                    .WORD   <NA_OTHER-HEAD>
                14B1  3727              ENDIF
                14B1        2081$:                                      ;GENERATE LCL 2081 NAME
                14B1  3728
                14B1  3729              ENDIF
                14B1        2079$:                                      ;GENERATE LCL 2079 NAME
                14B1  3730
                14B1  3731              IF      CONSOLE_TEST            ;CALL OUT FAILING OPTION
                3A  14B1                .BYTE   ^072
          0255' 14B2                    .WORD   <CONSOLE_TEST-HEAD>
                0F  14B4                .BYTE   ^017
                D2  14B5                .BYTE   ^0322
          14C1' 14B6                    .WORD   <2082$-HEAD>
                14B8  3732
                14B8  3733              LXI     H,CPU_A                 ;IF 8085 BASED CONSOLE TEST
                21  14B8                .BYTE   <H*8> ! 1
          02'57' 14B9                   .BYTE   <CPU_A-HEAD>&255 , <<CPU_A-HEAD>&^XFF00>/256
                14BB  3734              CALL    PRINT_STRING_R          ; THEN CALL OUT CPU BAD
                CD  14BB                .BYTE   ^0315
          100E' 14BC                    .WORD   <PRINT_STRING_R-HEAD>
                14BE  3735
                14BE  3736              ELSE                            ;FOR WCS BASED TESTS
                C3  14BE                .BYTE   ^0303
          1541' 14BF                    .WORD   <2083$-HEAD>
                14C1        2082$:                                      ;GENERATE LOCAL SYMBOL NAME
                14C1  3737
                14C1  3738              MVI     A,WCS_MOD_DAT           ;READ MODULE DATA FROM LS
          46 3E 14C1                    .BYTE   <A*8> ! 6 , WCS_MOD_DAT&255
                14C3  3739              CALL    READ_LS_R
                CD  14C3                .BYTE   ^0315
          1167' 14C4                    .WORD   <READ_LS_R-HEAD>
                14C6  3740              LDA     DATA3
                3A  14C6                .BYTE   ^072
          00AC' 14C7                    .WORD   <DATA3-HEAD>
                14C9  3741              STA     TEMP_BYTE               ;STORE MODULE DATA
                32  14C9                .BYTE   ^062
          0418' 14CA                    .WORD   <TEMP_BYTE-HEAD>
                14CC  3742              LXI     H,CRD_OPT_A             ;POINT TO OPTION CALL OUT TABLE
                21  14CC                .BYTE   <H*8> ! 1
          08'25' 14CD                   .BYTE   <CRD_OPT_A-HEAD>&255 , <<CRD_OPT_A-HEAD>&^XFF00>/256
                14CF  3743              SHLD    TEMP_WORD               ;STORE THE POINTER
                22  14CF                .BYTE   ^042
          0419' 14D0                    .WORD   <TEMP_WORD-HEAD>
                14D2  3744              CLEAR   MOD_ERR
                AF  14D2                .BYTE   ^0250 ! A               ;with A
                32  14D3                .BYTE   ^062
          0395' 14D4                    .WORD   <MOD_ERR-HEAD>
                14D6  3745
                14D6  3746              DO      3                       ;FIRST 3 BITS CALL OUT CPU
                21  14D6                .BYTE   <H*8> ! 1
          00'55' 14D7                   .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
                46  14D9                .BYTE   ^0100 ! <B*8> ! M
```

```
                C5  14DA                .BYTE   ^O305 ! <8*B>
          03 3E  14DB                .BYTE   <A*8> ! 6 , 3&255
             77  14DD                .BYTE   ^O100 ! <M*8> ! A
                 14DE        2084$:
                 14DE  3747
                 14DE  3748                  LDA     TEMP_BYTE               ;CHECK FOR BIT SET
                3A  14DE                .BYTE   ^O72
              0418' 14DF                .WORD   <TEMP_BYTE-HEAD>
                 14E1  3749                  RRC
                OF  14E1                .BYTE   ^O17
                 14E2  3750                  STA     TEMP_BYTE
                32  14E2                .BYTE   ^O62
              0418' 14E3                .WORD   <TEMP_BYTE-HEAD>
                 14E5  3751
                 14E5  3752                  IFC
                D2  14E5                .BYTE   ^O322
              14FA' 14E6                .WORD   <2085$-HEAD>
                 14E8  3753
                 14E8  3754                  IFN     MOD_ERR                 ;CALL OUT BAD CPU
                3A  14E8                .BYTE   ^O72
              0395' 14E9                .WORD   <MOD_ERR-HEAD>
                OF  14EB                .BYTE   ^O17
                DA  14EC                .BYTE   ^O332
              14F5' 14ED                .WORD   <2086$-HEAD>
                 14EF  3755                  LXI     H,CPU_A                 ;IF WE HAVEN'T ALREADY DONE SO
                21  14EF                .BYTE   <H*8> ! 1
            02'57' 14F0                .BYTE   <CPU_A-HEAD>&255 , <<CPU_A-HEAD>&^XFF00>/256
                 14F2  3756                  CALL    PRINT_STRING_R
                CD  14F2                .BYTE   ^O315
              100E' 14F3                .WORD   <PRINT_STRING_R-HEAD>
                 14F5  3757                  ENDIF
                 14F5        2086$:                                          ;GENERATE LCL 2086 NAME
                 14F5  3758
                 14F5  3759                  SET     MOD_ERR                 ;AT LEAST 1 CPU MODULE IS BAD
                AF  14F5                .BYTE   ^O250 ! A               ;with A
                3C  14F6                .BYTE   ^O04 ! <8*A>
                32  14F7                .BYTE   ^O62
              0395' 14F8                .WORD   <MOD_ERR-HEAD>
                 14FA  3760
                 14FA  3761                  ENDIF
                 14FA        2085$:                                          ;GENERATE LCL 2085 NAME
                 14FA  3762                  ENDDO
                21  14FA                .BYTE   <H*8> ! 1
            00'55' 14FB                .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
                35  14FD                .BYTE   ^O05 ! <8*M>
                C2  14FE                .BYTE   ^O302
              14DE' 14FF                .WORD   <2084$-HEAD>
                C1  1501                .BYTE   ^O301 ! <8*B>
                70  1502                .BYTE   ^O100 ! <M*8> ! B
                 1503  3763
                 1503  3764                  DO      5                       ;EXAMINE REMAINING BITS
                21  1503                .BYTE   <H*8> ! 1
            00'55' 1504                .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
                46  1506                .BYTE   ^O100 ! <B*8> ! M
                C5  1507                .BYTE   ^O305 ! <8*B>
          05 3E  1508                .BYTE   <A*8> ! 6 , 5&255
             77  150A                .BYTE   ^O100 ! <M*8> ! A
```

```
                     150B        2087$:
                     150B  3765
                     150B  3766                    LDA     TEMP_BYTE               ;CHECK FOR BIT SET
                       3A  150B              .BYTE   ^O72
                  0418'    150C              .WORD   <TEMP_BYTE-HEAD>
                     150E  3767                    RRC
                       0F  150E              .BYTE   ^O17
                     150F  3768                    STA     TEMP_BYTE
                       32  150F              .BYTE   ^O62
                  0418'    1510              .WORD   <TEMP_BYTE-HEAD>
                     1512  3769
                     1512  3770                    IFC
                       D2  1512              .BYTE   ^O322
                  152D'    1513              .WORD   <2088$-HEAD>
                     1515  3771
                     1515  3772                    IF      MOD_ERR                 ;PRINT 'OR' IF OTHER MODULES
                       3A  1515              .BYTE   ^O72
                  0395'    1516              .WORD   <MOD_ERR-HEAD>
                       0F  1518              .BYTE   ^O17
                       D2  1519              .BYTE   ^O322
                  1522'    151A              .WORD   <2089$-HEAD>
                     151C  3773                    LXI     H,OR_A                  ;HAVE ALREADY BEEN CALLED OUT
                       21  151C              .BYTE   <H*8> ! 1
                  03'D0'   151D              .BYTE   <OR_A-HEAD>&255 , <<OR_A-HEAD>&^XFF00>/256
                     151F  3774                    CALL    PRINT_STRING_R
                       CD  151F              .BYTE   ^O315
                  100E'    1520              .WORD   <PRINT_STRING_R-HEAD>
                     1522  3775                    ENDIF
                     1522        2089$:                                          ;GENERATE LCL 2089 NAME
                     1522  3776
                     1522  3777                    LHLD    TEMP_WORD               ;PRINT OUT MODULE NAME
                       2A  1522              .BYTE   ^O52
                  0419'    1523              .WORD   <TEMP_WORD-HEAD>
                     1525  3778                    CALL    PRINT_STRING_R
                       CD  1525              .BYTE   ^O315
                  100E'    1526              .WORD   <PRINT_STRING_R-HEAD>
                     1528  3779                    SET     MOD_ERR
                       AF  1528              .BYTE   ^O250 ! A               ;with A
                       3C  1529              .BYTE   ^O4 ! <8*A>
                       32  152A              .BYTE   ^O62
                  0395'    152B              .WORD   <MOD_ERR-HEAD>
                     152D  3780
                     152D  3781                    ENDIF
                     152D        2088$:                                          ;GENERATE LCL 2088 NAME
                     152D  3782
                     152D  3783                    MVI     B,0                     ;INCREMENT TO POINT TO NEXT
                    00 06  152D              .BYTE   <B*8> ! 6 , 0&255
                     152F  3784                    MVI     C,7                     ; MODULE NAME
                    07 0E  152F              .BYTE   <C*8> ! 6 , 7&255
                     1531  3785                    LHLD    TEMP_WORD
                       2A  1531              .BYTE   ^O52
                  0419'    1532              .WORD   <TEMP_WORD-HEAD>
                     1534  3786                    DAD     B
                       09  1534              .BYTE   <B*8> ! 9
                     1535  3787                    SHLD    TEMP_WORD
                       22  1535              .BYTE   ^O42
                  0419'    1536              .WORD   <TEMP_WORD-HEAD>
```

```
                     1538  3788
                     1538  3789                    ENDDO
          21         1538                  .BYTE   <H*8> ! 1
   00'55'            1539                  .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
          35         153B                  .BYTE   ^05 ! <8*M>
          C2         153C                  .BYTE   ^0302
    150B'            153D                  .WORD   <2087$-HEAD>
          C1         153F                  .BYTE   ^0301 ! <8*B>
          70         1540                  .BYTE   ^0100 ! <M*8> ! B
                     1541  3790
                     1541  3791                    ENDIF
                     1541        2083$:                                          ;GENERATE LCL 2083 NAME
                     1541  3792
                     1541  3793                    LXI     H,BAD_A                 ;PRINT 'BAD'
          21         1541                  .BYTE   <H*8> ! 1
   01'7E'            1542                  .BYTE   <BAD_A-HEAD>&255 , <<BAD_A-HEAD>&^XFF00>/256
                     1544  3794                    CALL    PRINT_STRING_R
          CD         1544                  .BYTE   ^0315
    100E'            1545                  .WORD   <PRINT_STRING_R-HEAD>
                     1547  3795
                     1547  3796                    JMP     MENU                    ;WHAT TO DO FROM HERE?
          C3         1547                  .BYTE   ^0303
    1616'            1548                  .WORD   <MENU-HEAD>
                     154A  3797
                     154A  3798 ;*****************************************************************************
                     154A  3799 ;
                     154A  3800 ;  FULL_ERROR
                     154A  3801 ;  THIS ROUTINE PRINTS OUT THE FULL ERROR REPORT IF THAT OPTION IS DESIRED.
                     154A  3802 ;
                     154A  3803 ;*****************************************************************************
                     154A  3804
                     154A  3805 FULL_ERROR:      LXI     H,HEADER_A              ;PRINT HEADER LINE FOR ERROR
          21         154A                  .BYTE   <H*8> ! 1
   03'02'            154B                  .BYTE   <HEADER_A-HEAD>&255 , <<HEADER_A-HEAD>&^XFF00>/256
                     154D  3806                    CALL    PRINT_STRING_R
          CD         154D                  .BYTE   ^0315
    100E'            154E                  .WORD   <PRINT_STRING_R-HEAD>
                     1550  3807
                     1550  3808                    LXI     H,SECTION_NAME          ;PRINT SECTION NAME LOADED
          21         1550                  .BYTE   <H*8> ! 1
   08'CA'            1551                  .BYTE   <SECTION_NAME-HEAD>&255 , <<SECTION_NAME-HEAD>&^XFF00>/256
                     1553  3809                    MVI     C,SEC_LEN
   06 0E             1553                  .BYTE   <C*8> ! 6 , SEC_LEN&255
                     1555  3810                    CALL    PRINTER
          CD         1555                  .BYTE   ^0315
    1007'            1556                  .WORD   <PRINTER-HEAD>
                     1558  3811
                     1558  3812                    CALL    SPACE_R
          CD         1558                  .BYTE   ^0315
    1021'            1559                  .WORD   <SPACE_R-HEAD>
                     155B  3813
                     155B  3814                    LDA     TEST_NUM                ;PRINT TEST NUMBER
          3A         155B                  .BYTE   ^072
    0426'            155C                  .WORD   <TEST_NUM-HEAD>
                     155E  3815                    STA     HEX_BUF
          32         155E                  .BYTE   ^062
    005B'            155F                  .WORD   <HEX_BUF-HEAD>
```

```
                     1561  3816                       CALL    PRINT_HEX_1_R
          CD         1561                       .BYTE   ^0315
      0F94'          1562                       .WORD   <PRINT_HEX_1_R-HEAD>
                     1564  3817                       CALL    SPACE_R
          CD         1564                       .BYTE   ^0315
      1021'          1565                       .WORD   <SPACE_R-HEAD>
                     1567  3818
                     1567  3819                       LDA     CON_ERR_NUM             ;ERROR NUMBER LOADED PREVIOUSLY
          3A         1567                       .BYTE   ^072
      003F'          1568                       .WORD   <CON_ERR_NUM-HEAD>
                     156A  3820                       STA     HEX_BUF
          32         156A                       .BYTE   ^062
      005B'          156B                       .WORD   <HEX_BUF-HEAD>
                     156D  3821                       CALL    PRINT_HEX_1_R           ;PRINT ERROR NUMBER
          CD         156D                       .BYTE   ^0315
      0F94'          156E                       .WORD   <PRINT_HEX_1_R-HEAD>
                     1570  3822                       CALL    SPACE_R
          CD         1570                       .BYTE   ^0315
      1021'          1571                       .WORD   <SPACE_R-HEAD>
                     1573  3823
                     1573  3824                       IF      NA_EXP_REC              ;IF EXPECTED AND RECEIVED DATA
          3A         1573                       .BYTE   ^072
      005F'          1574                       .WORD   <NA_EXP_REC-HEAD>
          0F         1576                       .BYTE   ^017
          D2         1577                       .BYTE   ^0322
      1589'          1578                       .WORD   <2090$-HEAD>
                     157A  3825                                               ; ARE NOT APPLICABLE
                     157A  3826                       LXI     H,NA_A                  ;POINT TO ASCII N/A
          21         157A                       .BYTE   <H*8> ! 1
      03'B0'         157B                       .BYTE   <NA_A-HEAD>&255 , <<NA_A-HEAD>&^XFF00>/256
                     157D  3827                       CALL    PRINT_STRING_R          ;PRINT UNDER EXPECTED DATA
          CD         157D                       .BYTE   ^0315
      100E'          157E                       .WORD   <PRINT_STRING_R-HEAD>
                     1580  3828                       LXI     H,NA_A                  ;POINT TO ASCII N/A
          21         1580                       .BYTE   <H*8> ! 1
      03'B0'         1581                       .BYTE   <NA_A-HEAD>&255 , <<NA_A-HEAD>&^XFF00>/256
                     1583  3829                       CALL    PRINT_STRING_R          ;PRINT UNDER RECEIVED DATA
          CD         1583                       .BYTE   ^0315
      100E'          1584                       .WORD   <PRINT_STRING_R-HEAD>
                     1586  3830
                     1586  3831                       ELSE
          C3         1586                       .BYTE   ^0303
      15A1'          1587                       .WORD   <2091$-HEAD>
                     1589            2090$:                                      ;GENERATE LOCAL SYMBOL NAME
                     1589  3832
                     1589  3833                       LXI     H,CON_EXP_DAT           ;POINT TO EXPECTED DATA
          21         1589                       .BYTE   <H*8> ! 1
      00'40'         158A                       .BYTE   <CON_EXP_DAT-HEAD>&255 , <<CON_EXP_DAT-HEAD>&^XFF00>/256
                     158C  3834                       LXI     D,HEX_BUF
          11         158C                       .BYTE   <D*8> ! 1
      00'5B'         158D                       .BYTE   <HEX_BUF-HEAD>&255 , <<HEX_BUF-HEAD>&^XFF00>/256
                     158F  3835                       CALL    MOVER_4_R
          CD         158F                       .BYTE   ^0315
      0F73'          1590                       .WORD   <MOVER_4_R-HEAD>
                     1592  3836                       CALL    PRINT_HEX_4             ;PRINT EXPECTED DATA
          CD         1592                       .BYTE   ^0315
      0F8A'          1593                       .WORD   <PRINT_HEX_4-HEAD>
```

```
                     1595  3837
                     1595  3838                    L       H,CON_REC_DAT           ;POINT TO RECEIVED DATA
          21         1595                  .BYTE   <H*8> ! 1
  00'4F'             1596                  .BYTE   <CON_REC_DAT-HEAD>&255 , <<CON_REC_DAT-HEAD>&^XFF00>/256
                     1598  3839                    LXI     D,HEX_BUF
          11         1598                  .BYTE   <D*8> ! 1
  00'5B'             1599                  .BYTE   <HEX_BUF-HEAD>&255 , <<HEX_BUF-HEAD>&^XFF00>/256
                     159B  3840                    CALL    MOVER_4_R
          CD         159B                  .BYTE   ^O315
  0F73'              159C                  .WORD   <MOVER_4_R-HEAD>
                     159E  3841                    CALL    PRINT_HEX_4             ;PRINT RECEIVED DATA
          CD         159E                  .BYTE   ^O315
  0F8A'              159F                  .WORD   <PRINT_HEX_4-HEAD>
                     15A1  3842
                     15A1  3843                    ENDIF
                     15A1          2091$:                                          ;GENERATE LCL 2091 NAME
                     15A1  3844
                     15A1  3845                    IF      NA_OTHER                ;IF OTHER DATA IS NOT
          3A         15A1                  .BYTE   ^O72
  0060'              15A2                  .WORD   <NA_OTHER-HEAD>
          0F         15A4                  .BYTE   ^O17
          D2         15A5                  .BYTE   ^O322
  15B1'              15A6                  .WORD   <2092$-HEAD>
                     15A8  3846                                                    ; APPLICABLE
                     15A8  3847                    LXI     H,NA_A                  ;POINT TO ASCII N/A
          21         15A8                  .BYTE   <H*8> ! 1
  03'B0'             15A9                  .BYTE   <NA_A-HEAD>&255 , <<NA_A-HEAD>&^XFF00>/256
                     15AB  3848                    CALL    PRINT_STRING_R          ;PRINT UNDER OTHER DATA
          CD         15AB                  .BYTE   ^O315
  100E'              15AC                  .WORD   <PRINT_STRING_R-HEAD>
                     15AE  3849
                     15AE  3850                    ELSE
          C3         15AE                  .BYTE   ^O303
  15BD'              15AF                  .WORD   <2093$-HEAD>
                     15B1          2092$:                                          ;GENERATE LOCAL SYMBOL NAME
                     15B1  3851                    LXI     H,CON_ADD_DAT           ;POINT TO OTHER DATA
          21         15B1                  .BYTE   <H*8> ! 1
  00'3B'             15B2                  .BYTE   <CON_ADD_DAT-HEAD>&255 , <<CON_ADD_DAT-HEAD>&^XFF00>/256
                     15B4  3852                    LXI     D,HEX_BUF
          11         15B4                  .BYTE   <D*8> ! 1
  00'5B'             15B5                  .BYTE   <HEX_BUF-HEAD>&255 , <<HEX_BUF-HEAD>&^XFF00>/256
                     15B7  3853                    CALL    MOVER_4_R
          CD         15B7                  .BYTE   ^O315
  0F73'              15B8                  .WORD   <MOVER_4_R-HEAD>
                     15BA  3854                    CALL    PRINT_HEX_4             ;PRINT OTHER DATA
          CD         15BA                  .BYTE   ^O315
  0F8A'              15BB                  .WORD   <PRINT_HEX_4-HEAD>
                     15BD  3855                    ENDIF
                     15BD          2093$:                                          ;GENERATE LCL 2093 NAME
                     15BD  3856
                     15BD  3857                    LXI     H,CON_MSK_DAT           ;POINT TO ERROR MASK
          21         15BD                  .BYTE   <H*8> ! 1
  00'4B'             15BE                  .BYTE   <CON_MSK_DAT-HEAD>&255 , <<CON_MSK_DAT-HEAD>&^XFF00>/256
                     15C0  3858                    LXI     D,HEX_BUF
          11         15C0                  .BYTE   <D*8> ! 1
  00'5B'             15C1                  .BYTE   <HEX_BUF-HEAD>&255 , <<HEX_BUF-HEAD>&^XFF00>/256
                     15C3  3859                    CALL    MOVER_4_R
```

```
                  CD  15C3                .BYTE   ^0315
              0F73'   15C4                .WORD   <MOVER_4_R-HEAD>
                      15C6  3860                  CALL    PRINT_HEX_4             ;PRINT ERROR MASK
                  CD  15C6                .BYTE   ^0315
              0F8A'   15C7                .WORD   <PRINT_HEX_4-HEAD>
                      15C9  3861
                      15C9  3862                  IFN     CONSOLE_TEST
                  3A  15C9                .BYTE   ^072
              0255'   15CA                .WORD   <CONSOLE_TEST-HEAD>
                  0F  15CC                .BYTE   ^017
                  DA  15CD                .BYTE   ^0332
              1608'   15CE                .WORD   <2094$-HEAD>
                      15D0  3863
                      15D0  3864                  MVI     A,WCS_MOD_DAT           ;PRINT MODULE NUMBER
               46 3E  15D0                .BYTE   <A*8> ! 6 , WCS_MOD_DAT&255
                      15D2  3865                  CALL    READ_LS_R               ;READ FROM LS
                  CD  15D2                .BYTE   ^0315
              1167'   15D3                .WORD   <READ_LS_R-HEAD>
                      15D5  3866                  LDA     DATA3
                  3A  15D5                .BYTE   ^072
              00AC'   15D6                .WORD   <DATA3-HEAD>
                      15D8  3867                  STA     TEMP_BYTE
                  32  15D8                .BYTE   ^062
              0418'   15D9                .WORD   <TEMP_BYTE-HEAD>
                      15DB  3868                  LXI     H,PRT_MOD_A
                  21  15DB                .BYTE   <H*8> ! 1
            01'3B'    15DC                .BYTE   <PRT_MOD_A-HEAD>&255 , <<PRT_MOD_A-HEAD>&^XFF00>/256
                      15DE  3869                  SHLD    TEMP_WORD
                  22  15DE                .BYTE   ^042
              0419'   15DF                .WORD   <TEMP_WORD-HEAD>
                      15E1  3870
                      15E1  3871                  DO      8                       ;EIGHT POSS. MODULE NAMES
                  21  15E1                .BYTE   <H*8> ! 1
            00'55'    15E2                .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
                  46  15E4                .BYTE   ^0100 ! <B*8> ! M
                  C5  15E5                .BYTE   ^0305 ! <8*B>
               08 3E  15E6                .BYTE   <A*8> ! 6 , 8&255
                  77  15E8                .BYTE   ^0100 ! <M*8> ! A
                      15E9          2095$:
                      15E9  3872
                      15E9  3873                  LDA     TEMP_BYTE
                  3A  15E9                .BYTE   ^072
              0418'   15EA                .WORD   <TEMP_BYTE-HEAD>
                      15EC  3874                  RRC
                  0F  15EC                .BYTE   ^017
                      15ED  3875                  STA     TEMP_BYTE
                  32  15ED                .BYTE   ^062
              0418'   15EE                .WORD   <TEMP_BYTE-HEAD>
                      15F0  3876
                      15F0  3877                  IFC
                  D2  15F0                .BYTE   ^0322
              15F9'   15F1                .WORD   <2096$-HEAD>
                      15F3  3878                  CALL    PRINT_FLG
                  CD  15F3                .BYTE   ^0315
              0F7E'   15F4                .WORD   <PRINT_FLG-HEAD>
                      15F6  3879                  CALL    SPACE_R
                  CD  15F6                .BYTE   ^0315
```

```
1021' 15F7                      .WORD   <SPACE_R-HEAD>
      15F9  3880                ENDIF
      15F9        2096$:                                      ;GENERATE LCL 2096 NAME
      15F9  3881
      15F9  3882                CALL    INC_TEMPW_5           ;INCREMENT TO NEXT FLAG
  CD  15F9                      .BYTE   ^O315
12A5' 15FA                      .WORD   <INC_TEMPW_5-HEAD>
      15FC  3883
      15FC  3884                ENDDO
  21  15FC                      .BYTE   <H*8> ! 1
00'55' 15FD                     .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
  35  15FF                      .BYTE   ^O5 ! <8*M>
  C2  1600                      .BYTE   ^O302
15E9' 1601                      .WORD   <2095$-HEAD>
  C1  1603                      .BYTE   ^O301 ! <8*B>
  70  1604                      .BYTE   ^O100 ! <M*8> ! B
      1605  3885
      1605  3886                ELSE                          ;IF CONSOLE TEST
  C3  1605                      .BYTE   ^O303
1610' 1606                      .WORD   <2097$-HEAD>
      1608        2094$:                                      ;GENERATE LOCAL SYMBOL NAME
      1608  3887                LXI     H,CON_MOD_DAT
  21  1608                      .BYTE   <H*8> ! 1
00'46' 1609                     .BYTE   <CON_MOD_DAT-HEAD>&255 , <<CON_MOD_DAT-HEAD>&^XFF00>/256
      160B  3888                MVI     C,5
05 0E 160B                      .BYTE   <C*8> ! 6 , 5&255
      160D  3889                CALL    PRINTER               ;PRINT MODULE #
  CD  160D                      .BYTE   ^O315
1007' 160E                      .WORD   <PRINTER-HEAD>
      1610  3890                ENDIF
      1610        2097$:                                      ;GENERATE LCL 2097 NAME
      1610  3891
      1610  3892                CALL    CRLF_R
  CD  1610                      .BYTE   ^O315
101A' 1611                      .WORD   <CRLF_R-HEAD>
      1613  3893
      1613  3894                JMP     MENU
  C3  1613                      .BYTE   ^O303
1616' 1614                      .WORD   <MENU-HEAD>
      1616  3895
      1616  3896
      1616  3897 ;*****************************************************************************
      1616  3898 ;
      1616  3899 ;  MENU
      1616  3900 ;  THIS ROUTINE PRINTS THE MENU PROMPT AFTER AN ERROR OR CONTROL_C.  IT
      1616  3901 ;  THEN TAKES THE APPROPRIATE ACTION.
      1616  3902 ;
      1616  3903 ;*****************************************************************************
      1616  3904
      1616  3905 MENU:           CALL    CRLF_R
  CD  1616                      .BYTE   ^O315
101A' 1617                      .WORD   <CRLF_R-HEAD>
      1619  3906
      1619  3907                LXI     H,MENU_A              ;PRINT OPTIONS
  21  1619                      .BYTE   <H*8> ! 1
03'55' 161A                     .BYTE   <MENU_A-HEAD>&255 , <<MENU_A-HEAD>&^XFF00>/256
      161C  3908                CALL    PRINT_STRING_R
```

```
                      CD  161C                .BYTE   ^O315
                    100E' 161D                .WORD   <PRINT_STRING_R-HEAD>
                          161F  3909
                          161F  3910                  CALL    INPUT_LINE              ;GET DESIRED OPTION
                      CD  161F                .BYTE   ^O315
                    102E' 1620                .WORD   <INPUT_LINE-HEAD>
                          1622  3911
                          1622  3912                  MVI     C,1
                   01 0E  1622                .BYTE   <C*8> ! 6 , 1&255
                          1624  3913                  LXI     H,TTBUF                 ;POINT TO USER RESPONSE
                      21  1624                .BYTE   <H*8> ! 1
                    08'D1' 1625                .BYTE   <TTBUF-HEAD>&255 , <<TTBUF-HEAD>&^XFF00>/256
                          1627  3914                  XCHG
                      EB  1627                .BYTE   ^O353
                          1628  3915
                          1628  3916                  LXI     H,ASCII_1               ;WAS A 1 TYPED?
                      21  1628                .BYTE   <H*8> ! 1
                    01'7B' 1629                .BYTE   <ASCII_1-HEAD>&255 , <<ASCII_1-HEAD>&^XFF00>/256
                          162B  3917                  CALL    COMPARE_R
                      CD  162B                .BYTE   ^O315
                    0F55' 162C                .WORD   <COMPARE_R-HEAD>
                          162E  3918
                          162E  3919                  IFZ                             ;IF 1 WAS TYPED
                      C2  162E                .BYTE   ^O302
                    1640' 162F                .WORD   <2098$-HEAD>
                          1631  3920                  CALL    CRLF_R
                      CD  1631                .BYTE   ^O315
                    101A' 1632                .WORD   <CRLF_R-HEAD>
                          1634  3921                  LXI     H,END_A
                      21  1634                .BYTE   <H*8> ! 1
                    02'62' 1635                .BYTE   <END_A-HEAD>&255 , <<END_A-HEAD>&^XFF00>/256
                          1637  3922                  CALL    PRINT_STRING_R
                      CD  1637                .BYTE   ^O315
                    100E' 1638                .WORD   <PRINT_STRING_R-HEAD>
                          163A  3923                  CALL    CRLF_R
                      CD  163A                .BYTE   ^O315
                    101A' 163B                .WORD   <CRLF_R-HEAD>
                          163D  3924                  JMP     STNDRD_CON_COM          ;THEN GO BACK TO CONSOLE
                      C3  163D                .BYTE   ^O303
                    098A' 163E                .WORD   <STNDRD_CON_COM-HEAD>
                          1640  3925                  ENDIF
                          1640        2098$:                                        ;GENERATE LCL 2098 NAME
                          1640  3926
                          1640  3927                  LXI     H,ASCII_2
                      21  1640                .BYTE   <H*8> ! 1
                    01'7C' 1641                .BYTE   <ASCII_2-HEAD>&255 , <<ASCII_2-HEAD>&^XFF00>/256
                          1643  3928                  CALL    COMPARE_R
                      CD  1643                .BYTE   ^O315
                    0F55' 1644                .WORD   <COMPARE_R-HEAD>
                          1646  3929
                          1646  3930                  IFZ                             ;IF 2 WAS TYPED
                      C2  1646                .BYTE   ^O302
                    164C' 1647                .WORD   <2099$-HEAD>
                          1649  3931                  JMP     START_UP                ;THEN START AUTO MODE
                      C3  1649                .BYTE   ^O303
                    1CD2' 164A                .WORD   <START_UP-HEAD>
                          164C  3932                  ENDIF
```

```
                        164C        2099$:                                          ;GENERATE LCL 2099 NAME
                        164C  3933
                        164C  3934                LXI     H,ASCII_3
                  21    164C                .BYTE   <H*8> ! 1
               01'7D'   164D                .BYTE   <ASCII_3-HEAD>&255 , <<ASCII_3-HEAD>&^XFF00>/256
                        164F  3935                CALL    COMPARE_R
                  CD    164F                .BYTE   ^O315
               0F55'    1650                .WORD   <COMPARE_R-HEAD>
                        1652  3936
                        1652  3937                IFZ                                     ;IF 3 WAS TYPED
                  C2    1652                .BYTE   ^O302
               1658'    1653                .WORD   <2100$-HEAD>
                        1655  3938                JMP     FULL_ERROR                      ;PRINT FULL ERROR REPORT
                  C3    1655                .BYTE   ^O303
               154A'    1656                .WORD   <FULL_ERROR-HEAD>
                        1658  3939                ENDIF
                        1658        2100$:                                          ;GENERATE LCL 2100 NAME
                        1658  3940
                        1658  3941                JMP     MENU
                  C3    1658                .BYTE   ^O303
               1616'    1659                .WORD   <MENU-HEAD>
                        165B  3942
                        165B  3943
                        165B  3944 ;*****************************************************************************
                        165B  3945 ;
                        165B  3946 ;  PACK_WARNING
                        165B  3947 ;  THIS ROUTINE PRINTS THE IMPROPER RL02 SETUP WARNING MESSAGE AS DISCOVERED
                        165B  3948 ;  BY THE IDC PART II DIAGNOSTIC.
                        165B  3949 ;
                        165B  3950 ;*****************************************************************************
                        165B  3951
                        165B  3952 PACK_WARNING:    CALL    CRLF_R
                  CD    165B                .BYTE   ^O315
               101A'    165C                .WORD   <CRLF_R-HEAD>
                        165E  3953                LXI     H,WARN_MESB                     ;PRINT WARNING MESSAGE
                  21    165E                .BYTE   <H*8> ! 1
               05'C8'   165F                .BYTE   <WARN_MESB-HEAD>&255 , <<WARN_MESB-HEAD>&^XFF00>/256
                        1661  3954                CALL    PRINT_STRING_R
                  CD    1661                .BYTE   ^O315
               100E'    1662                .WORD   <PRINT_STRING_R-HEAD>
                        1664  3955                LXI     H,WARN_MESA                     ;PRINT WARNING MESSAGE
                  21    1664                .BYTE   <H*8> ! 1
               05'9A'   1665                .BYTE   <WARN_MESA-HEAD>&255 , <<WARN_MESA-HEAD>&^XFF00>/256
                        1667  3956                CALL    PRINT_STRING_R
                  CD    1667                .BYTE   ^O315
               100E'    1668                .WORD   <PRINT_STRING_R-HEAD>
                        166A  3957
                        166A  3958                CALL    INPUT_LINE                      ;WAIT FOR RESPONSE
                  CD    166A                .BYTE   ^O315
               102E'    166B                .WORD   <INPUT_LINE-HEAD>
                        166D  3959
                        166D  3960                MVI     C,1
               01 0E    166D                .BYTE   <C*8> ! 6 , 1&255
                        166F  3961                LXI     H,TTBUF
                  21    166F                .BYTE   <H*8> ! 1
               08'D1'   1670                .BYTE   <TTBUF-HEAD>&255 , <<TTBUF-HEAD>&^XFF00>/256
                        1672  3962                XCHG
```

```
                EB  1672                .BYTE   ^O353
                    1673  3963
                    1673  3964                  LXI     H,ASCII_1
                21  1673                .BYTE   <H*8> ! 1
          01'7B'    1674                .BYTE   <ASCII_1-HEAD>&255 , <<ASCII_1-HEAD>&^XFF00>/256
                    1676  3965                  CALL    COMPARE_R
                CD  1676                .BYTE   ^O315
              0F55' 1677                .WORD   <COMPARE_R-HEAD>
                    1679  3966
                    1679  3967                  IFZ
                C2  1679                .BYTE   ^O302
              16A7' 167A                .WORD   <2101$-HEAD>
                    167C  3968                  CALL    CRLF_R
                CD  167C                .BYTE   ^O315
              101A' 167D                .WORD   <CRLF_R-HEAD>
                    167F  3969                  LXI     H,WARN_MESS             ;PRINT WARNING MESSAGE
                21  167F                .BYTE   <H*8> ! 1
          04'B4'    1680                .BYTE   <WARN_MESS-HEAD>&255 , <<WARN_MESS-HEAD>&^XFF00>/256
                    1682  3970                  CALL    PRINT_STRING_R
                CD  1682                .BYTE   ^O315
              100E' 1683                .WORD   <PRINT_STRING_R-HEAD>
                    1685  3971
                    1685  3972                  CALL    INPUT_LINE
                CD  1685                .BYTE   ^O315
              102E' 1686                .WORD   <INPUT_LINE-HEAD>
                    1688  3973
                    1688  3974                  CLEAR   PRINT_START             ;DON'T PRINT START OR PASSED
                AF  1688                .BYTE   ^O250 ! A               ;with A
                32  1689                .BYTE   ^O62
              03F0' 168A                .WORD   <PRINT_START-HEAD>
                    168C  3975                  CLEAR   PRINT_PASS              ; MESSAGES
                AF  168C                .BYTE   ^O250 ! A               ;with A
                32  168D                .BYTE   ^O62
              03ED' 168E                .WORD   <PRINT_PASS-HEAD>
                    1690  3976
                    1690  3977                  DO      60                      ;SET UP CURSER TO PRINT
                21  1690                .BYTE   <H*8> ! 1
          00'55'    1691                .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
                46  1693                .BYTE   ^O100 ! <B*8> ! M
                C5  1694                .BYTE   ^O305 ! <8*B>
          3C 3E     1695                .BYTE   <A*8> ! 6 , 60&255
                77  1697                .BYTE   ^O100 ! <M*8> ! A
                    1698          2102$:
                    1698  3978                  CALL    SPACE_R                 ; PASSED OR FAILED MESSAGE
                CD  1698                .BYTE   ^O315
              1021' 1699                .WORD   <SPACE_R-HEAD>
                    169B  3979                  ENDDO
                21  169B                .BYTE   <H*8> ! 1
          00'55'    169C                .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
                35  169E                .BYTE   ^O5 ! <8*M>
                C2  169F                .BYTE   ^O302
              1698' 16A0                .WORD   <2102$-HEAD>
                C1  16A2                .BYTE   ^O301 ! <8*B>
                70  16A3                .BYTE   ^O100 ! <M*8> ! B
                    16A4  3980
                    16A4  3981                  JMP     PARSER                  ;TRY TEST AGAIN
                C3  16A4                .BYTE   ^O303
```

```
                1DAE' 16A5                .WORD   <PARSER-HEAD>
                      16A7  3982                  ENDIF
                      16A7        2101$:                                      ;GENERATE LCL 2101 NAME
                      16A7  3983
                      16A7  3984                  LXI     H,ASCII_2
                   21 16A7                .BYTE   <H*8> ! 1
                01'7C' 16A8                .BYTE   <ASCII_2-HEAD>&255 , <<ASCII_2-HEAD>&^XFF00>/256
                      16AA  3985                  CALL    COMPARE_R
                   CD 16AA                .BYTE   ^O315
                0F55' 16AB                .WORD   <COMPARE_R-HEAD>
                      16AD  3986
                      16AD  3987                  IFZ
                   C2 16AD                .BYTE   ^O302
                16C6' 16AE                .WORD   <2103$-HEAD>
                      16B0  3988                  CLEAR   PRINT_PASS                  ; MESSAGES
                   AF 16B0                .BYTE   ^O250 ! A                   ;with A
                   32 16B1                .BYTE   ^O62
                03ED' 16B2                .WORD   <PRINT_PASS-HEAD>
                      16B4  3989                  CLEAR   PACK_ERROR
                   AF 16B4                .BYTE   ^O250 ! A                   ;with A
                   32 16B5                .BYTE   ^O62
                03D5' 16B6                .WORD   <PACK_ERROR-HEAD>
                      16B8  3990                  LXI     H,TEST_NUM
                   21 16B8                .BYTE   <H*8> ! 1
                04'26' 16B9                .BYTE   <TEST_NUM-HEAD>&255 , <<TEST_NUM-HEAD>&^XFF00>/256
                      16BB  3991                  INR     M                           ;INCREMENT TEST NUM
                   34 16BB                .BYTE   ^O4 ! <8*M>
                      16BC  3992                  SET     EOS_FLG
                   AF 16BC                .BYTE   ^O250 ! A                   ;with A
                   3C 16BD                .BYTE   ^O4 ! <8*A>
                   32 16BE                .BYTE   ^O62
                0056' 16BF                .WORD   <EOS_FLG-HEAD>
                      16C1  3993                  CLEAR   PRINT_PASS                  ; MESSAGES
                   AF 16C1                .BYTE   ^O250 ! A                   ;with A
                   32 16C2                .BYTE   ^O62
                03ED' 16C3                .WORD   <PRINT_PASS-HEAD>
                      16C5  3994                  RET
                   C9 16C5                .BYTE   ^O311
                      16C6  3995                  ENDIF
                      16C6        2103$:                                      ;GENERATE LCL 2103 NAME
                      16C6  3996
                      16C6  3997                  RET
                   C9 16C6                .BYTE   ^O311
                      16C7  3998
                      16C7  3999 ;*****************************************************************************
                      16C7  4000 ;
                      16C7  4001 ;  SET_PAR_INT
                      16C7  4002 ;  THIS ROUTINE SETS A PARITY ERROR IN WCS ON REQUEST FROM THE CPU DIAG.
                      16C7  4003 ;  THE ADDRESS TO SET THE PARITY ERROR ON IS IN LS 7
                      16C7  4004 ;
                      16C7  4005 ;*****************************************************************************
                      16C7  4006
                      16C7  4007 SET_PAR_INT:     CALL    READ_LS_7               ;READ ADDRESS TO SET BAD PARITY
                   CD 16C7                .BYTE   ^O315
                1165' 16C8                .WORD   <READ_LS_7-HEAD>
                      16CA  4008
                      16CA  4009                  LXI     D,INPUT_NUM             ;DESTINATION
```

```
                 11  16CA                .BYTE   <D*8> ! 1
               06'76' 16CB                .BYTE   <INPUT_NUM-HEAD>&255 , <<INPUT_NUM-HEAD>&^XFF00>/256
                      16CD  4010                   CALL    MOV_FROM_DATA0          ;MOVE 4 BYTES OF DATA FROM
                 CD   16CD                .BYTE   ^O315
                0F70' 16CE                .WORD   <MOV_FROM_DATA0-HEAD>
                      16D0  4011                                                   ; DATA0
                      16D0  4012
                      16D0  4013                   SET     WCS_BAD_PARITY          ;CHANGE PARITY TO BAD
                 AF   16D0                .BYTE   ^O250 ! A               ;with A
                 3C   16D1                .BYTE   ^O4 ! <8*A>
                 32   16D2                .BYTE   ^O62
                006E' 16D3                .WORD   <WCS_BAD_PARITY-HEAD>
                      16D5  4014                   CALL    CHANGE_WCS              ;WRITE WCS WITH BAD PARITY
                 CD   16D5                .BYTE   ^O315
                0C1D' 16D6                .WORD   <CHANGE_WCS-HEAD>
                      16D8  4015
                      16D8  4016                   RET
                 C9   16D8                .BYTE   ^O311
                      16D9  4017
                      16D9  4018
                      16D9  4019 ;*****************************************************************************
                      16D9  4020 ;
                      16D9  4021 ;  CLEAR_PAR_INT
                      16D9  4022 ;  THIS ROUTINE CLEARS A PARITY ERROR IN WCS ON REQUEST FROM THE CPU DIAG.
                      16D9  4023 ;  THIS ALSO ENABLES HALT ON PARITY ERROR
                      16D9  4024 ;  THE ADDRESS TO CLEAR THE PARITY ERROR ON IS IN LS 7
                      16D9  4025 ;
                      16D9  4026 ;*****************************************************************************
                      16D9  4027
                      16D9  4028 CLEAR_PAR_INT:
                      16D9  4029                   CALL    READ_LS_7               ;READ ADDRESS TO SET GOOD PARITY
                 CD   16D9                .BYTE   ^O315
               1165'  16DA                .WORD   <READ_LS_7-HEAD>
                      16DC  4030
                      16DC  4031                   LXI     D,INPUT_NUM             ;DESTINATION
                 11   16DC                .BYTE   <D*8> ! 1
               06'76' 16DD                .BYTE   <INPUT_NUM-HEAD>&255 , <<INPUT_NUM-HEAD>&^XFF00>/256
                      16DF  4032                   CALL    MOV_FROM_DATA0          ;MOVE 4 BYTES OF DATA FROM
                 CD   16DF                .BYTE   ^O315
                0F70' 16E0                .WORD   <MOV_FROM_DATA0-HEAD>
                      16E2  4033                                                   ; DATA0
                      16E2  4034
                      16E2  4035                   CLEAR   ERROR
                 AF   16E2                .BYTE   ^O250 ! A               ;with A
                 32   16E3                .BYTE   ^O62
                02E2' 16E4                .WORD   <ERROR-HEAD>
                      16E6  4036                   CALL    INT_CLEAR_PAR           ;SET THE PARITY GOOD
                 CD   16E6                .BYTE   ^O315
                0B8B' 16E7                .WORD   <INT_CLEAR_PAR-HEAD>
                      16E9  4037
                      16E9  4038                   RET
                 C9   16E9                .BYTE   ^O311
                      16EA  4039
                      16EA  4040
                      16EA  4041 ;*****************************************************************************
                      16EA  4042 ;
                      16FA  4043 ;  PRINT_SINGLE
```

```
                        16EA  4044 ;  THIS ROUTINES COMPARES THE NUMBER OF SINGLE BIT ERRORS IN A 256 K BLOCK
                        16EA  4045 ;  OF MEMORY WITH THE PREDETERMINED MAXIMUM.  IT WILL REPORT A BAD MEMORY
                        16EA  4046 ;  IF THE MAXIMUM IS MET OR EXCEEDED
                        16EA  4047 ;
                        16EA  4048 ;*****************************************************************************
                        16EA  4049
                        16EA  4050 PRINT_SINGLE:    CALL    READ_LS_7                   ;READ DATA
              CD        16EA                .BYTE   ^O315
          1165'         16EB                .WORD   <READ_LS_7-HEAD>
                        16ED  4051
                        16ED  4052                  IFGT    DATA1,MAX_SBE,3             ;IF SINGLE BIT ERRORS FOUND ARE
              21        16ED                .BYTE   <H*8> ! 1
         00'AA'         16EE                .BYTE   <DATA1-HEAD>&255 , <<DATA1-HEAD>&^XFF00>/256
              11        16F0                .BYTE   <D*8> ! 1
         03'4A'         16F1                .BYTE   <MAX_SBE-HEAD>&255 , <<MAX_SBE-HEAD>&^XFF00>/256
         03 0E          16F3                .BYTE   <C*8> ! 6 , 3&255
              CD        16F5                .BYTE   ^O315
          00CF'         16F6                .WORD   <COMPARE-HEAD>
              F2        16F8                .BYTE   ^O362
          1722'         16F9                .WORD   <2104$-HEAD>
                        16FB  4053                  LXI     H,FAIL_A                    ; GREATER THAN TOLERATED
              21        16FB                .BYTE   <H*8> ! 1
         02'EF'         16FC                .BYTE   <FAIL_A-HEAD>&255 , <<FAIL_A-HEAD>&^XFF00>/256
                        16FE  4054                  CALL    PRINT_STRING_R              ;PRINT FAILURE
              CD        16FE                .BYTE   ^O315
          100E'         16FF                .WORD   <PRINT_STRING_R-HEAD>
                        1701  4055                  CALL    CRLF_R
              CD        1701                .BYTE   ^O315
          101A'         1702                .WORD   <CRLF_R-HEAD>
                        1704  4056                  LXI     H,ABORT_A                   ;PRINT ABORT MESSAGE
              21        1704                .BYTE   <H*8> ! 1
         01'64'         1705                .BYTE   <ABORT_A-HEAD>&255 , <<ABORT_A-HEAD>&^XFF00>/256
                        1707  4057                  CALL    PRINT_STRING_R
              CD        1707                .BYTE   ^O315
          100E'         1708                .WORD   <PRINT_STRING_R-HEAD>
                        170A  4058                  LXI     H,MEMORY_A                  ;PRINT MEMORY BAD
              21        170A                .BYTE   <H*8> ! 1
         03'4D'         170B                .BYTE   <MEMORY_A-HEAD>&255 , <<MEMORY_A-HEAD>&^XFF00>/256
                        170D  4059                  CALL    PRINT_STRING_R
              CD        170D                .BYTE   ^O315
          100E'         170E                .WORD   <PRINT_STRING_R-HEAD>
                        1710  4060                  LXI     H,BAD_A
              21        1710                .BYTE   <H*8> ! 1
         01'7E'         1711                .BYTE   <BAD_A-HEAD>&255 , <<BAD_A-HEAD>&^XFF00>/256
                        1713  4061                  CALL    PRINT_STRING_R
              CD        1713                .BYTE   ^O315
          100E'         1714                .WORD   <PRINT_STRING_R-HEAD>
                        1716  4062                  LXI     H,END_A                     ;PRINT END MESSAGE
              21        1716                .BYTE   <H*8> ! 1
         02'62'         1717                .BYTE   <END_A-HEAD>&255 , <<END_A-HEAD>&^XFF00>/256
                        1719  4063                  CALL    PRINT_STRING_R
              CD        1719                .BYTE   ^O315
          100E'         171A                .WORD   <PRINT_STRING_R-HEAD>
                        171C  4064                  CALL    CRLF_R
              CD        171C                .BYTE   ^O315
          101A'         171D                .WORD   <CRLF_R-HEAD>
                        171F  4065                  JMP     STNDRD_CON_COM
```

```
                C3  171F                .BYTE   ^0303
              098A' 1720                .WORD   <STNDRD_CON_COM-HEAD>
                    1722  4066                  ENDIF
                    1722        2104$:                                      ;GENERATE LCL 2104 NAME
                    1722  4067
                    1722  4068                  RET
                C9  1722                .BYTE   ^0311
                    1723  4069
                    1723  4070 ;*****************************************************************************
                    1723  4071 ;
                    1723  4072 ;  PRINT_MOD_PRES
                    1723  4073 ;  THIS ROUTINE PRINTS WHETHER THE MICRO DIAGNOSITIC FINDS THE FPA, UBE, IDC
                    1723  4074 ;  OR R80
                    1723  4075 ;
                    1723  4076 ;*****************************************************************************
                    1723  4077
                    1723  4078 PRINT_MOD_PRES: CALL    READ_LS_7               ;LS 7 CONTAINS 1 FOR MOD PRES
                CD  1723                .BYTE   ^0315
              1165' 1724                .WORD   <READ_LS_7-HEAD>
                    1726  4079                                                ; AND 0 FOR NOT PRESENT
                    1726  4080                  IF      DATA3                 ;IF MODULE IS PRESENT
                3A  1726                .BYTE   ^072
              00AC' 1727                .WORD   <DATA3-HEAD>
                0F  1729                .BYTE   ^017
                D2  172A                .BYTE   ^0322
              172E' 172B                .WORD   <2105$-HEAD>
                    172D  4081                  RET
                C9  172D                .BYTE   ^0311
                    172E  4082                  ENDIF
                    172E        2105$:                                      ;GENERATE LCL 2105 NAME
                    172E  4083
                    172E  4084                  CALL    CRLF_R
                CD  172E                .BYTE   ^0315
              101A' 172F                .WORD   <CRLF_R-HEAD>
                    1731  4085                  LXI     H,FPA_A               ;PRINT FPA NOT PRESENT
                21  1731                .BYTE   <H*8> ! 1
            02'FE' 1732                .BYTE   <FPA_A-HEAD>&255 , <<FPA_A-HEAD>&^XFF00>/256
                    1734  4086                  CALL    PRINT_STRING_R
                CD  1734                .BYTE   ^0315
              100E' 1735                .WORD   <PRINT_STRING_R-HEAD>
                    1737  4087                  LXI     H,MOD_NOT_PRES_A      ;POINT TO MODULE NOT PRESENT
                21  1737                .BYTE   <H*8> ! 1
            03'96' 1738                .BYTE   <MOD_NOT_PRES_A-HEAD>&255 , <<MOD_NOT_PRES_A-HEAD>&^XFF00>/256
                    173A  4088                  CALL    PRINT_STRING_R
                CD  173A                .BYTE   ^0315
              100E' 173B                .WORD   <PRINT_STRING_R-HEAD>
                    173D  4089
                    173D  4090                  CLEAR   PRINT_PASS            ;DO NOT PRINT 'PASS'
                AF  173D                .BYTE   ^0250 ! A               ;with A
                32  173E                .BYTE   ^062
              03ED' 173F                .WORD   <PRINT_PASS-HEAD>
                    1741  4091
                    1741  4092                  RET
                C9  1741                .BYTE   ^0311
                    1742  4093
                    1742  4094 PRINT_MOD_PRES_IDC:
                    1742  4095                  CALL    READ_LS_7             ;LS 7 CONTAINS 1 FOR MOD PRES
```

```
                      CD  1742                .BYTE   ^0315
                    1165' 1743                .WORD   <READ_LS_7-HEAD>
                          1745  4096
                          1745  4097                  IF      DATA3                   ;IF MODULE IS PRESENT
                      3A  1745                .BYTE   ^072
                    00AC' 1746                .WORD   <DATA3-HEAD>
                      0F  1748                .BYTE   ^017
                      D2  1749                .BYTE   ^0322
                    174D' 174A                .WORD   <2106$-HEAD>
                          174C  4098                  RET
                      C9  174C                .BYTE   ^0311
                          174D  4099                  ENDIF
                          174D        2106$:                                      ;GENERATE LCL 2106 NAME
                          174D  4100
                          174D  4101                  LXI     H,RETURN_I          ;NO IDC RELOAD CONSOLE
                      21  174D                .BYTE   <H*8> ! 1
                    07'66' 174E               .BYTE   <RETURN_I-HEAD>&255 , <<RETURN_I-HEAD>&^XFF00>/256
                          1750  4102                  SHLD    INSTR_PNT           ;
                      22  1750                .BYTE   ^042
                    0345' 1751                .WORD   <INSTR_PNT-HEAD>
                          1753  4103
                          1753  4104                  LXI     H,RETURN_N          ;THEN PRINT
                      21  1753                .BYTE   <H*8> ! 1
                    07'ED' 1754               .BYTE   <RETURN_N-HEAD>&255 , <<RETURN_N-HEAD>&^XFF00>/256
                          1756  4105                  SHLD    NAME_PNT            ;CPU PART 3
                      22  1756                .BYTE   ^042
                    03BA' 1757                .WORD   <NAME_PNT-HEAD>
                          1759  4106
                          1759  4107                  LXI     H,RETURN_T          ;AND
                      21  1759                .BYTE   <H*8> ! 1
                    08'20' 175A               .BYTE   <RETURN_T-HEAD>&255 , <<RETURN_T-HEAD>&^XFF00>/256
                          175C  4108                  SHLD    TIME_PNT            ;SET TIME
                      22  175C                .BYTE   ^042
                    0448' 175D                .WORD   <TIME_PNT-HEAD>
                          175F  4109
                          175F  4110                  RET
                      C9  175F                .BYTE   ^0311
                          1760  4111
                          1760  4112
                          1760  4113 ;*****************************************************************************
                          1760  4114 ;
                          1760  4115 ;  SET_SIGNALS
                          1760  4116 ;  THIS ROUTINE SETS UP THE SIGNALS AS THEY ARE DESCRIBED IN THE LS
                          1760  4117 ;  COMMAND AND STATUS WORD BITS
                          1760  4118 ;
                          1760  4119 ;*****************************************************************************
                          1760  4120
                          1760  4121 SET_SIGNALS:    OUT     CLRHLT              ;CLEAR CONSOLE HALT
                    AF D3 1760                .BYTE   ^0323 , CLRHLT&255      ;register A
                          1762  4122
                          1762  4123                  IFMBI   COMMAND1,CONHLT     ;BIT 17
                      3A  1762                .BYTE   ^072
                    00B0' 1763                .WORD   <COMMAND1-HEAD>
                      E6  1765                .BYTE   ^0346
                      02  1766                .BYTE   CONHLT
                      CA  1767                .BYTE   ^0312
                    176C' 1768                .WORD   <2107$-HEAD>
```

```
                    176A  4124                    OUT     SETHLT                  ;SET CONSOLE HALT
AE D3               176A                  .BYTE   ^O323 , SETHLT&255      ;register A
                    176C  4125                    ENDIF
                    176C        2107$:                                      ;GENERATE LCL 2107 NAME
                    176C  4126
                    176C  4127                    OUT     CLRPFI                  ;CLEAR POWER FAIL INIT
AD D3               176C                  .BYTE   ^O323 , CLRPFI&255      ;register A
                    176E  4128
                    176E  4129                    IFMBI   COMMAND1,PWRFAIL        ;BIT 18
   3A               176E                  .BYTE   ^O72
 00B0'              176F                  .WORD   <COMMAND1-HEAD>
   E6               1771                  .BYTE   ^O346
   04               1772                  .BYTE   PWRFAIL
   CA               1773                  .BYTE   ^O312
 1778'              1774                  .WORD   <2108$-HEAD>
                    1776  4130                    OUT     SETPFI                  ;SET POWER FAIL INIT
AC D3               1776                  .BYTE   ^O323 , SETPFI&255      ;register A
                    1778  4131                    ENDIF
                    1778        2108$:                                      ;GENERATE LCL 2108 NAME
                    1778  4132
                    1778  4133                    OUT     CLRTIM                  ;CLEAR INTERVAL TIMER INIT
A5 D3               1778                  .BYTE   ^O323 , CLRTIM&255      ;register A
                    177A  4134
                    177A  4135                    IFMBI   COMMAND1,INVTIM         ;BIT 19
   3A               177A                  .BYTE   ^O72
 00B0'              177B                  .WORD   <COMMAND1-HEAD>
   E6               177D                  .BYTE   ^O346
   08               177E                  .BYTE   INVTIM
   CA               177F                  .BYTE   ^O312
 1784'              1780                  .WORD   <2109$-HEAD>
                    1782  4136                    OUT     SETTIM                  ;SET INTERVAL TIMER INIT
A4 D3               1782                  .BYTE   ^O323 , SETTIM&255      ;register A
                    1784  4137                    ENDIF
                    1784        2109$:                                      ;GENERATE LCL 2109 NAME
                    1784  4138
                    1784  4139                    OUT     CLRATN                  ;CLEAR CONSOLE ATTENTION
AB D3               1784                  .BYTE   ^O323 , CLRATN&255      ;register A
                    1786  4140
                    1786  4141                    IFMBI   COMMAND1,CONATT         ;BIT 20
   3A               1786                  .BYTE   ^O72
 00B0'              1787                  .WORD   <COMMAND1-HEAD>
   E6               1789                  .BYTE   ^O346
   10               178A                  .BYTE   CONATT
   CA               178B                  .BYTE   ^O312
 1790'              178C                  .WORD   <2110$-HEAD>
                    178E  4142                    OUT     SETATN                  ;SET CONSOLE ATTENTION
AA D3               178E                  .BYTE   ^O323 , SETATN&255      ;register A
                    1790  4143                    ENDIF
                    1790        2110$:                                      ;GENERATE LCL 2110 NAME
                    1790  4144
                    1790  4145                    CLEAR   SETACK_FLG              ;CLEAR FLAG TO CLEAR CONSOLE
   AF               1790                  .BYTE   ^O250 ! A               ;with A
   32               1791                  .BYTE   ^O62
 0405'              1792                  .WORD   <SETACK_FLG-HEAD>
                    1794  4146                                                ; ACKNOWLEDGE BEFORE RETURNING
                    1794  4147                                                ; TO WCS
                    1794  4148
```

```
                     1794  4149                    IFMBI   COMMAND1,CONACK         ;BIT 21
          3A         1794                  .BYTE   ^072
      00B0'          1795                  .WORD   <COMMAND1-HEAD>
          E6         1797                  .BYTE   ^0346
          20         1798                  .BYTE   CONACK
          CA         1799                  .BYTE   ^0312
      17A1'          179A                  .WORD   <2111$-HEAD>
                     179C  4150                    SET     SETACK_FLG              ;SET FLAG TO SET CONSOLE
          AF         179C                  .BYTE   ^0250 ! A               ;with A
          3C         179D                  .BYTE   ^04 ! <8*A>
          32         179E                  .BYTE   ^062
      0405'          179F                  .WORD   <SETACK_FLG-HEAD>
                     17A1  4151                                                    ; ACKNOWLEDGE BEFORE RETURNING
                     17A1  4152                                                    ; TO WCS
                     17A1  4153                    ENDIF
                     17A1          2111$:                                          ;GENERATE LCL 2111 NAME
                     17A1  4154
                     17A1  4155                    SET     MEM_REQ                 ;ALLOW MEMORY REQEST WHEN
          AF         17A1                  .BYTE   ^0250 ! A               ;with A
          3C         17A2                  .BYTE   ^04 ! <8*A>
          32         17A3                  .BYTE   ^062
      0354'          17A4                  .WORD   <MEM_REQ-HEAD>
                     17A6  4156
                     17A6  4157                    IFMBI   COMMAND1,EMEMREF        ;BIT 22
          3A         17A6                  .BYTE   ^072
      00B0'          17A7                  .WORD   <COMMAND1-HEAD>
          E6         17A9                  .BYTE   ^0346
          40         17AA                  .BYTE   EMEMREF
          CA         17AB                  .BYTE   ^0312
      17B2'          17AC                  .WORD   <2112$-HEAD>
                     17AE  4158                    CLEAR   MEM_REQ                 ;DONT ALLOW MEM REQ WHEN
          AF         17AE                  .BYTE   ^0250 ! A               ;with A
          32         17AF                  .BYTE   ^062
      0354'          17B0                  .WORD   <MEM_REQ-HEAD>
                     17B2  4159                    ENDIF
                     17B2          2112$:                                          ;GENERATE LCL 2112 NAME
                     17B2  4160
                     17B2  4161                    OUT     INITL                   ;CLEAR UNIBUS INIT
      34 D3          17B2                  .BYTE   ^0323 , INITL&255       ;register A
                     17B4  4162
                     17B4  4163                    IFMBI   COMMAND0,UBS_INIT       ;BIT 28
          3A         17B4                  .BYTE   ^072
      00AF'          17B5                  .WORD   <COMMAND0-HEAD>
          E6         17B7                  .BYTE   ^0346
          10         17B8                  .BYTE   UBS_INIT
          CA         17B9                  .BYTE   ^0312
      17BE'          17BA                  .WORD   <2113$-HEAD>
                     17BC  4164                    OUT     INITH                   ;SET UNIBUS INIT
      35 D3          17BC                  .BYTE   ^0323 , INITH&255       ;register A
                     17BE  4165                    ENDIF
                     17BE          2113$:                                          ;GENERATE LCL 2113 NAME
                     17BE  4166
                     17BE  4167                    OUT     CLRDCL                  ;CLEAR UNIBUS DC LO
      31 D3          17BE                  .BYTE   ^0323 , CLRDCL&255      ;register A
                     17C0  4168
                     17C0  4169                    IFMBI   COMMAND0,UBS_DC_LO      ;BIT 29
          3A         17C0                  .BYTE   ^072
```

```
                00AF' 17C1                .WORD   <COMMAND0-HEAD>
                  E6  17C3                .BYTE   ^0346
                  20  17C4                .BYTE   UBS_DC_LO
                  CA  17C5                .BYTE   ^0312
                17D2' 17C6                .WORD   <2114$-HEAD>
                      17C8  4170                  CALL    NOP_CSR_R               ;LOAD CSR WITH A NOP
                  CD  17C8                .BYTE   ^0315
                0ECA' 17C9                .WORD   <NOP_CSR_R-HEAD>
                      17CB  4171                  OUT     SETDCL                  ;SET UNIBUS DC LO
               30 D3  17CB                .BYTE   ^0323 , SETDCL&255      ;register A
                      17CD  4172                  CALL    MICRO_STEP_CPU_R        ;CLOCK THE SIGNAL INTO MCT
                  CD  17CD                .BYTE   ^0315
                12A0' 17CE                .WORD   <MICRO_STEP_CPU_R-HEAD>
                      17D0  4173                  OUT     CLRDCL                  ;CLEAR SIGNAL
               31 D3  17D0                .BYTE   ^0323 , CLRDCL&255      ;register A
                      17D2  4174                  ENDIF
                      17D2        2114$:                                      ;GENERATE LCL 2114 NAME
                      17D2  4175
                      17D2  4176                  OUT     CLRBSY                  ;CLEAR UNIBUS BUSY
               2E D3  17D2                .BYTE   ^0323 , CLRBSY&255      ;register A
                      17D4  4177
                      17D4  4178                  IFMBI   COMMAND0,UBS_BUSY       ;BIT 30
                  3A  17D4                .BYTE   ^072
                00AF' 17D5                .WORD   <COMMAND0-HEAD>
                  E6  17D7                .BYTE   ^0346
                  40  17D8                .BYTE   UBS_BUSY
                  CA  17D9                .BYTE   ^0312
                17DE' 17DA                .WORD   <2115$-HEAD>
                      17DC  4179                  OUT     SETBSY                  ;SET UNIBUS BUSY
               2F D3  17DC                .BYTE   ^0323 , SETBSY&255      ;register A
                      17DE  4180                  ENDIF
                      17DE        2115$:                                      ;GENERATE LCL 2115 NAME
                      17DE  4181
                      17DE  4182                  RET
                  C9  17DE                .BYTE   ^0311
                      17DF  4183
                      17DF  4184
                      17DF  4185  ;*****************************************************************************
                      17DF  4186  ;
                      17DF  4187  ;  LOOPER
                      17DF  4188  ;  THIS ROUTINE READS A COUNT FROM THE CPU AND CAUSES THE CPU TO LOOP
                      17DF  4189  ;  THAT NUMBER OF TIMES BY SETTING THE ADDRESS TO ITS LOOP INSTRUCTION
                      17DF  4190  ;  WHEN CALLED THE FIRST TIME THIS ROUTINE READS THE COUNT. ON ALL OTHER
                      17DF  4191  ;  CALLS UNTIL THE COUNT = 0 THE COUNT IS DECREMENTED.  WHEN THE COUNT = 0
                      17DF  4192  ;  THE ADDRESS IS SET FOR NO LOOPING AND THE NEXT CALL IS SET TO READ A
                      17DF  4193  ;  NEW COUNT
                      17DF  4194  ;
                      17DF  4195  ;*****************************************************************************
                      17DF  4196
                      17DF  4197  LOOPER:
                      17DF  4198                  IFN     HAVE_CNT
                  3A  17DF                .BYTE   ^072
                033D' 17E0                .WORD   <HAVE_CNT-HEAD>
                  0F  17E2                .BYTE   ^017
                  DA  17E3                .BYTE   ^0332
                17F6' 17E4                .WORD   <2116$-HEAD>
                      17F6  4199
```

```
                     17E6  4200                    MVI     A,LOOP_CNT_ADD          ;READ LOOP COUNT (ONCE ONLY)
          47 3E      17E6                  .BYTE   <A*8> ! 6 , LOOP_CNT_ADD&255
                     17E8  4201                    CALL    READ_LS_R
            CD       17E8                  .BYTE   ^0315
          1167'      17E9                  .WORD   <READ_LS_R-HEAD>
                     17EB  4202                    LDA     DATA3                   ;GET COUNT FROM DATA WORD
            3A       17EB                  .BYTE   ^072
          00AC'      17EC                  .WORD   <DATA3-HEAD>
                     17EE  4203                    STA     LOOP_COUNTER            ;SAVE IT
            32       17EE                  .BYTE   ^062
          0347'      17EF                  .WORD   <LOOP_COUNTER-HEAD>
                     17F1  4204
                     17F1  4205                    SET     HAVE_CNT
            AF       17F1                  .BYTE   ^0250 ! A               ;with A
            3C       17F2                  .BYTE   ^04 ! <8*A>
            32       17F3                  .BYTE   ^062
          033D'      17F4                  .WORD   <HAVE_CNT-HEAD>
                     17F6  4206
                     17F6  4207                    ENDIF
                     17F6          2116$:                                  ;GENERATE LCL 2116 NAME
                     17F6  4208
                     17F6  4209                    LXI     H,LOOP_COUNTER
            21       17F6                  .BYTE   <H*8> ! 1
          03'47'     17F7                  .BYTE   <LOOP_COUNTER-HEAD>&255 , <<LOOP_COUNTER-HEAD>&^XFF00>/256
                     17F9  4210                    DCR     M                       ;DECREMENT LOOP COUNT
            35       17F9                  .BYTE   ^05 ! <8*M>
                     17FA  4211                    RNZ                             ;EXIT IF COUNT NOT EXPIRED
            C0       17FA                  .BYTE   ^0300
                     17FB  4212
                     17FB  4213                    LXI     H,UPC_SUB               ;MOVE UPC ONE EXTRA WORD
            21       17FB                  .BYTE   <H*8> ! 1
          00'C6'     17FC                  .BYTE   <UPC_SUB-HEAD>&255 , <<UPC_SUB-HEAD>&^XFF00>/256
                     17FE  4214                    CALL    INC_WORD_R              ;AND CAUSE IT TO START THERE
            CD       17FE                  .BYTE   ^0315
          1281'      17FF                  .WORD   <INC_WORD_R-HEAD>
                     1801  4215
                     1801  4216                    CLEAR   HAVE_CNT                ;SET TO READ A NEW COUNT
            AF       1801                  .BYTE   ^0250 ! A               ;with A
            32       1802                  .BYTE   ^062
          033D'      1803                  .WORD   <HAVE_CNT-HEAD>
                     1805  4217
                     1805  4218                    RET
            C9       1805                  .BYTE   ^0311
                     1806  4219
                     1806  4220
                     1806  4221 ;*****************************************************************************
                     1806  4222 ;
                     1806  4223 ;  SUB_ATTENTION
                     1806  4224 ;  THIS ROUTINE IS USED AS AN IDLE ROUTINE WHEN EXECUTING A SUBROUTINE IN
                     1806  4225 ;  WCS.  THERE IS NO READ OF THE COMMAND AND STATUS WORD AFTER AN ATTENTION
                     1806  4226 ;  SIGNAL IS FOUND BECAUSE IT IS KNOWN THAT THIS MEANS END OF SUBROUTINE.
                     1806  4227 ;  SEE ATTENTION ROUTINE FOR MORE INFO
                     1806  4228 ;
                     1806  4229 ;*****************************************************************************
                     1806  4230
                     1806  4231 SUB_ATTENTION:
                     1806  4232                    CALL    RESET_TIMOUT            ;RESET TIMOUT COUNTER
```

```
                                                .BYTE   ^0315
1932' 1807                                      .WORD   <RESET_TIMOUT-HEAD>
      1809  4233
      1809  4234                                WHILE   RUNNING
      1809        4002$:                      LDA     RUNNING
  3A  1809                                      .BYTE   ^072
0403' 180A                                      .WORD   <RUNNING-HEAD>
  0F  180C                                      .BYTE   ^017
  D2  180D                                      .BYTE   ^0322
183D' 180E                                      .WORD   <2117$-HEAD>
      1810  4235
      1810  4236                                LDA     MICRO_STEP              ;GET MICRO STEP FLAG
  3A  1810                                      .BYTE   ^072
038A' 1811                                      .WORD   <MICRO_STEP-HEAD>
      1813  4237                                ORA     A                       ;SET CONDITION CODES
  B7  1813                                      .BYTE   ^0260 ! A
      1814  4238                                CNZ     MICRO_STEPER            ;MICRO STEP CPU IF SET
  C4  1814                                      .BYTE   ^0304
1257' 1815                                      .WORD   <MICRO_STEPER-HEAD>
      1817  4239
      1817  4240                                IN      CPATTN
83 DB 1817                                      .BYTE   ^0333 , CPATTN&255      ;to register A
      1819  4241                                RAR
  1F  1819                                      .BYTE   ^037
      181A  4242
      181A  4243                                IFC                             ;IF CPU_ATTENTION
  D2  181A                                      .BYTE   ^0322
1834' 181B                                      .WORD   <2118$-HEAD>
      181D  4244                                IN      CPUACK
82 DB 181D                                      .BYTE   ^0333 , CPUACK&255      ;to register A
      181F  4245                                STA     ACK_SAVE
  32  181F                                      .BYTE   ^062
017A' 1820                                      .WORD   <ACK_SAVE-HEAD>
      1822  4246                                CALL    RESET_TIMOUT
  CD  1822                                      .BYTE   ^0315
1932' 1823                                      .WORD   <RESET_TIMOUT-HEAD>
      1825  4247                                CALL    STOP_CPU                ;STOP SUBROUTINE
  CD  1825                                      .BYTE   ^0315
10C9' 1826                                      .WORD   <STOP_CPU-HEAD>
      1828  4248
      1828  4249                                MVI     B,<^X22>                ;ERROR CODE IF CPU ACK SET
22 06 1828                                      .BYTE   <B*8> ! 6 , ^X22&255
      182A  4250                                LDA     ACK_SAVE
  3A  182A                                      .BYTE   ^072
017A' 182B                                      .WORD   <ACK_SAVE-HEAD>
      182D  4251                                RAR
  1F  182D                                      .BYTE   ^037
      182E  4252                                JC      ERROR_ROUTINE_NO_RET    ;GO TO ERROR ROUTINE IF CPU
  DA  182E                                      .BYTE   ^0332
0C45' 182F                                      .WORD   <ERROR_ROUTINE_NO_RET-HEAD>
      1831  4253                                                                ; ACK (CARRY) IS SET. DON'T
      1831  4254                                                                ; RETURN
      1831  4255
      1831  4256                                CALL    CHECK_CNTLC             ;ELSE IF ^C TYPED, GO TO PARSER
  CD  1831                                      .BYTE   ^0315
1073' 1832                                      .WORD   <CHECK_CNTLC-HEAD>
      1834  4257
```

```
                        1834  4258                    ENDIF
                        1834        2118$:                                      ;GENERATE LCL 2118 NAME
                        1834  4259
                        1834  4260                    CALL    GCVECT              ;CHECK FOR OPERATOR REQUEST
                  CD    1834                .BYTE   ^0315
                4129    1835                .WORD   <GCVECT-HEAD>
                        1837  4261                    CALL    TIMOUT
                  CD    1837                .BYTE   ^0315
               1907'    1838                .WORD   <TIMOUT-HEAD>
                        183A  4262
                        183A  4263                    ENDWHILE
                  C3    183A                .BYTE   ^0303
               1809'    183B                .WORD   <4002$-HEAD>
                        183D        2117$:                                      ;GENERATE LCL 2117 NAME
                        183D  4264
                        183D  4265                    RET
                  C9    183D                .BYTE   ^0311
                        183E  4266
                        183E  4267
                        183E  4268 ;*************************************************************************
                        183E  4269 ;
                        183E  4270 ;  EOP
                        183E  4271 ;  THIS ROUTINE IS EXECUTED AT END OF PASS
                        183E  4272 ;
                        183E  4273 ;*************************************************************************
                        183E  4274
                        183E  4275 EOP:
                        183E  4276                    IF      PRINT_PASS
                  3A    183E                .BYTE   ^072
               03ED'    183F                .WORD   <PRINT_PASS-HEAD>
                  0F    1841                .BYTE   ^017
                  D2    1842                .BYTE   ^0322
               184B'    1843                .WORD   <2119$-HEAD>
                        1845  4277
                        1845  4278                    LXI     H,PASS_A
                  21    1845                .BYTE   <H*8> ! 1
               03'DE'   1846                .BYTE   <PASS_A-HEAD>&255 , <<PASS_A-HEAD>&^XFF00>/256
                        1848  4279                    CALL    PRINT_STRING_R
                  CD    1848                .BYTE   ^0315
               100E'    1849                .WORD   <PRINT_STRING_R-HEAD>
                        184B  4280
                        184B  4281                    ENDIF
                        184B        2119$:                                      ;GENERATE LCL 2119 NAME
                        184B  4282
                        184B  4283                    LXI     H,SEC_TABLE
                  21    184B                .BYTE   <H*8> ! 1
               08'5F'   184C                .BYTE   <SEC_TABLE-HEAD>&255 , <<SEC_TABLE-HEAD>&^XFF00>/256
                        184E  4284                    SHLD    TABLE_PNT           ;RESET TABLE POINTER
                  22    184E                .BYTE   ^042
               040C'    184F                .WORD   <TABLE_PNT-HEAD>
                        1851  4285
                        1851  4286                    CLEAR   EOS_FLG             ;RESET END OF SECTION
                  AF    1851                .BYTE   ^0250 ! A               ;with A
                  32    1852                .BYTE   ^062
                0056'   1853                .WORD   <EOS_FLG-HEAD>
                        1855  4287
                        1855  4288                    IFNEQ   PASS_CNT,F_WORD,2       ;-1 PASS CNT =INFINITE,SKIP OUT
```

```
                21  1855                .BYTE   <H*8> ! 1
          03'EA'    1856                .BYTE   <PASS_CNT-HEAD>&255 , <<PASS_CNT-HEAD>&^XFF00>/256
                11  1858                .BYTE   <D*8> ! 1
          00'58'    1859                .BYTE   <F_WORD-HEAD>&255 , <<F_WORD-HEAD>&^XFF00>/256
          02 0E     185B                .BYTE   <C*8> ! 6 , 2&255
                CD  185D                .BYTE   ^O315
            00CF'   185E                .WORD   <COMPARE-HEAD>
                CA  1860                .BYTE   ^O312
            1880'   1861                .WORD   <2120$-HEAD>
                    1863  4289
                    1863  4290                    LXI     H,PASS_CNT
                21  1863                .BYTE   <H*8> ! 1
          03'EA'    1864                .BYTE   <PASS_CNT-HEAD>&255 , <<PASS_CNT-HEAD>&^XFF00>/256
                    1866  4291                    CALL    DECR_WORD_R             ;DECREMENT PASS COUNT
                CD  1866                .BYTE   ^O315
            12BC'   1867                .WORD   <DECR_WORD_R-HEAD>
                    1869  4292
                    1869  4293                    IFEQ    PASS_CNT,ZERO_WORD,2    ;IF PASS CNT = 0 CAUSE
                21  1869                .BYTE   <H*8> ! 1
          03'EA'    186A                .BYTE   <PASS_CNT-HEAD>&255 , <<PASS_CNT-HEAD>&^XFF00>/256
                11  186C                .BYTE   <D*8> ! 1
          05'96'    186D                .BYTE   <ZERO_WORD-HEAD>&255 , <<ZERO_WORD-HEAD>&^XFF00>/256
          02 0E     186F                .BYTE   <C*8> ! 6 , 2&255
                CD  1871                .BYTE   ^O315
            00CF'   1872                .WORD   <COMPARE-HEAD>
                C2  1874                .BYTE   ^O302
            1880'   1875                .WORD   <2121$-HEAD>
                    1877  4294                                            ;RETURN TO PARSER
                    1877  4295                    SET     FILE_LOADED             ;CAUSE EXIT OF MAIN LOAD LOOP
              AF    1877                .BYTE   ^O250 ! A               ;with A
              3C    1878                .BYTE   ^O04 ! <8*A>
              32    1879                .BYTE   ^O062
          02FD'     187A                .WORD   <FILE_LOADED-HEAD>
                    187C  4296                    CLEAR   EXECUTE
              AF    187C                .BYTE   ^O250 ! A               ;with A
              32    187D                .BYTE   ^O062
          02EA'     187E                .WORD   <EXECUTE-HEAD>
                    1880  4297                    ENDIF
                    1880        2121$:                                      ;GENERATE LCL 2121 NAME
                    1880  4298
                    1880  4299                    ENDIF
                    1880        2120$:                                      ;GENERATE LCL 2120 NAME
                    1880  4300
                    1880  4301                    RET
              C9    1880                .BYTE   ^O311
                    1881  4302
                    1881  4303 ;*****************************************************************************
                    1881  4304 ;
                    1881  4305 ;  BEGIN_OF_TEST
                    1881  4306 ;  THIS ROUTINE HANDLES BEGINING OF TEST SENT FROM THE CPU
                    1881  4307 ;  IF ALSO VERIFIES THE TEST SEQUENCE OF WCS TESTING
                    1881  4308 ;
                    1881  4309 ;*****************************************************************************
                    1881  4310
                    1881  4311 BEGIN_OF_TEST:
                    1881  4312                    LXI     H,TEST_NUM
              21    1881                .BYTE   <H*8> ! 1
```

```
                04'26' 1882                .BYTE   <TEST_NUM-HEAD>&255 , <<TEST_NUM-HEAD>&^XFF00>/256
                       1884  4313                   INR     M                       ;INCREMENT TEST NUM
                   34  1884                .BYTE   ^04 ! <8*M>
                       1885  4314
                       1885  4315                   IFGT    TEST_NUM,TEST_NUM2,1
                   21  1885                .BYTE   <H*8> ! 1
                04'26' 1886                .BYTE   <TEST_NUM-HEAD>&255 , <<TEST_NUM-HEAD>&^XFF00>/256
                   11  1888                .BYTE   <D*8> ! 1
                04'28' 1889                .BYTE   <TEST_NUM2-HEAD>&255 , <<TEST_NUM2-HEAD>&^XFF00>/256
                01 0E  188B                .BYTE   <C*8> ! 6 , 1&255
                   CD  188D                .BYTE   ^0315
                00CF'  188E                .WORD   <COMPARE-HEAD>
                   F2  1890                .BYTE   ^0362
                1898'  1891                .WORD   <2122$-HEAD>
                       1893  4316                   SET     EOS_FLG                 ;FORCE END OF SECTION
                   AF  1893                .BYTE   ^0250 ! A               ;with A
                   3C  1894                .BYTE   ^04 ! <8*A>
                   32  1895                .BYTE   ^062
                0056'  1896                .WORD   <EOS_FLG-HEAD>
                       1898  4317                   ENDIF
                       1898        2122$:                                  ;GENERATE LCL 2122 NAME
                       1898  4318
                       1898  4319                   CALL    VERIFY_SEQ
                   CD  1898                .BYTE   ^0315
                18A1'  1899                .WORD   <VERIFY_SEQ-HEAD>
                       189B  4320                   SET     MEM_REQ
                   AF  189B                .BYTE   ^0250 ! A               ;with A
                   3C  189C                .BYTE   ^04 ! <8*A>
                   32  189D                .BYTE   ^062
                0354'  189E                .WORD   <MEM_REQ-HEAD>
                       18A0  4321                   RET
                   C9  18A0                .BYTE   ^0311
                       18A1  4322
                       18A1  4323 ;*****************************************************************************
                       18A1  4324 ;
                       18A1  4325 ;  VERIFY_SEQ
                       18A1  4326 ;  THIS ROUTINE VERIFIES THE TESTS RUN IN SEQUENCE BY READING THE UPC
                       18A1  4327 ;  AND COMPARING IT TO A KNOW VALUE
                       18A1  4328 ;
                       18A1  4329 ;*****************************************************************************
                       18A1  4330
                       18A1  4331 VERIFY_SEQ:     LHLD    UPC_SUB
                   2A  18A1                .BYTE   ^052
                00C6'  18A2                .WORD   <UPC_SUB-HEAD>
                       18A4  4332                   SHLD    TEMP_WORD2              ;SAVE GET UPC VALUE
                   22  18A4                .BYTE   ^042
                041D'  18A5                .WORD   <TEMP_WORD2-HEAD>
                       18A7  4333
                       18A7  4334                   DO      4
                   21  18A7                .BYTE   <H*8> ! 1
                00'55' 18A8                .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
                   46  18AA                .BYTE   ^0100 ! <B*8> ! M
                   C5  18AB                .BYTE   ^0305 ! <8*B>
                04 3E  18AC                .BYTE   <A*8> ! 6 , 4&255
                   77  18AE                .BYTE   ^0100 ! <M*8> ! A
                       18AF        2123$:
                       18AF  4335
```

```
                    18AF  4336                  LXI     H,TEMP_WORD2            ;MASK OFF WCS HIGH PAGE
          21        18AF                .BYTE   <H*8> ! 1
    04'1D'          18B0                .BYTE   <TEMP_WORD2-HEAD>&255 , <<TEMP_WORD2-HEAD>&^XFF00>/256
                    18B2  4337                  MOV     A,M
          7E        18B2                .BYTE   ^0100 ! <A*8> ! M
                    18B3  4338                  ANI     HEX_3F
          E6        18B3                .BYTE   ^0346
          3F        18B4                .BYTE   HEX_3F
                    18B5  4339                  MOV     M,A
          77        18B5                .BYTE   ^0100 ! <M*8> ! A
                    18B6  4340                  CALL    DECR_WORD_R             ;INC VALUE BY 4
          CD        18B6                .BYTE   ^0315
    12BC'           18B7                .WORD   <DECR_WORD_R-HEAD>
                    18B9  4341
                    18B9  4342                  ENDDO
          21        18B9                .BYTE   <H*8> ! 1
    00'55'          18BA                .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
          35        18BC                .BYTE   ^05 ! <8*M>
          C2        18BD                .BYTE   ^0302
    18AF'           18BE                .WORD   <2123$-HEAD>
          C1        18C0                .BYTE   ^0301 ! <8*B>
          70        18C1                .BYTE   ^0100 ! <M*8> ! B
                    18C2  4343
                    18C2  4344                  CLEAR   WCS_ADDRESS
          AF        18C2                .BYTE   ^0250 ! A               ;with A
          32        18C3                .BYTE   ^062
    006C'           18C4                .WORD   <WCS_ADDRESS-HEAD>
                    18C6  4345                  LDA     TEST_NUM                ;GET KNOW VALUE FOR THIS TEST
          3A        18C6                .BYTE   ^072
    0426'           18C7                .WORD   <TEST_NUM-HEAD>
                    18C9  4346                  STA     WCS_ADDRESS+1           ;FROM WCS
          32        18C9                .BYTE   ^062
    006D'           18CA                .WORD   <WCS_ADDRESS+1-HEAD>
                    18CC  4347                  CALL    READ_WCS_R
          CD        18CC                .BYTE   ^0315
    0E6D'           18CD                .WORD   <READ_WCS_R-HEAD>
                    18CF  4348
                    18CF  4349                  DO      4
          21        18CF                .BYTE   <H*8> ! 1
    00'55'          18D0                .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
          46        18D2                .BYTE   ^0100 ! <B*8> ! M
          C5        18D3                .BYTE   ^0305 ! <8*B>
    04    3E        18D4                .BYTE   <A*8> ! 6 , 4&255
          77        18D6                .BYTE   ^0100 ! <M*8> ! A
                    18D7          2124$:
                    18D7  4350                  LXI     H,WCS_VALUE+2           ;POINT TO WCS DATA
          21        18D7                .BYTE   <H*8> ! 1
    00'B6'          18D8                .BYTE   <WCS_VALUE+2-HEAD>&255 , <<WCS_VALUE+2-HEAD>&^XFF00>/256
                    18DA  4351                  MVI     C,3
    03    0E        18DA                .BYTE   <C*8> ! 6 , 3&255
                    18DC  4352                  CALL    SHIFTER1_R              ;SHIFT 17-4(OF JUMP) TO 21-8
          CD        18DC                .BYTE   ^0315
    0EC1'           18DD                .WORD   <SHIFTER1_R-HEAD>
                    18DF  4353                  ENDDO
          21        18DF                .BYTE   <H*8> ! 1
    00'55'          18E0                .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
          35        18F2                .BYTE   ^05 ! <8*M>
```

```
                C2  18E3                .BYTE   ^O302
              18D7' 18E4                .WORD   <2124$-HEAD>
                C1  18E6                .BYTE   ^O301 ! <8*B>
                70  18E7                .BYTE   ^O100 ! <M*8> ! B
                    18E8  4354
                    18E8  4355                  LDA     WCS_VALUE
                3A  18E8                .BYTE   ^O72
              00B4' 18E9                .WORD   <WCS_VALUE-HEAD>
                    18EB  4356                  ANI     HEX_3F                      ;AND OUT TO ONLY ADDRESS
                E6  18EB                .BYTE   ^O346
                3F  18EC                .BYTE   HEX_3F
                    18ED  4357                  STA     WCS_VALUE
                32  18ED                .BYTE   ^O62
              00B4' 18EE                .WORD   <WCS_VALUE-HEAD>
                    18F0  4358
                    18F0  4359                  IFNEQ   TEMP_WORD2,WCS_VALUE,2
                21  18F0                .BYTE   <H*8> ! 1
              04'1D' 18F1               .BYTE   <TEMP_WORD2-HEAD>&255 , <<TEMP_WORD2-HEAD>&^XFF00>/256
                11  18F3                .BYTE   <D*8> ! 1
              00'B4' 18F4               .BYTE   <WCS_VALUE-HEAD>&255 , <<WCS_VALUE-HEAD>&^XFF00>/256
              02 0E 18F6                .BYTE   <C*8> ! 6 , 2&255
                CD  18F8                .BYTE   ^O315
              00CF' 18F9                .WORD   <COMPARE-HEAD>
                CA  18FB                .BYTE   ^O312
              1906' 18FC                .WORD   <2125$-HEAD>
                    18FE  4360                  MVI     B,<^X23>                    ;SEQUENCE ERROR IN WCS TEST
              23 06 18FE                .BYTE   <B*8> ! 6 , ^X23&255
                    1900  4361                  CALL    ERROR_ROUTINE
                CD  1900                .BYTE   ^O315
              0C4A' 1901                .WORD   <ERROR_ROUTINE-HEAD>
                    1903  4362                  CALL    SET_FOR_CONT_R              ;ALLOW CONTINUE
                CD  1903                .BYTE   ^O315
              107B' 1904                .WORD   <SET_FOR_CONT_R-HEAD>
                    1906  4363                  ENDIF
                    1906        2125$:                                              ;GENERATE LCL 2125 NAME
                    1906  4364
                    1906  4365                  RET
                C9  1906                .BYTE   ^O311
                    1907  4366
                    1907  4367 ;*****************************************************************************
                    1907  4368 ;
                    1907  4369 ;  TIMOUT
                    1907  4370 ;  THIS ROUTINE IS USED TO TELL IF THE CPU IS NOT RESPONDING
                    1907  4371 ;  IN A REASONABLE TIME PERIOD.  IF THE COUNTER OVERFLOWS A TIMOUT
                    1907  4372 ;  HAS OCCURED.
                    1907  4373 ;
                    1907  4374 ;*****************************************************************************
                    1907  4375
                    1907  4376 TIMOUT:           IFNMBI  FLAGS,LOOP_S_DEF
                3A  1907                .BYTE   ^O72
              005A' 1908                .WORD   <FLAGS-HEAD>
                E6  190A                .BYTE   ^O346
                01  190B                .BYTE   LOOP_S_DEF
                C2  190C                .BYTE   ^O302
              192E' 190D                .WORD   <2126$-HEAD>
                    190F  4377
                    190F  4378                  LXI     H,TIMOUT_CNT                ;COUNTER FOR TIMING
```

```
                21  190F                .BYTE   <H*8> ! 1
          04'4A'    1910                .BYTE   <TIMOUT_CNT-HEAD>&255 , <<TIMOUT_CNT-HEAD>&^XFF00>/256
                    1912  4379          CALL    INC_WORD_R                ;INCREMENT COUNTER
                CD  1912                .BYTE   ^O315
           12B1'    1913                .WORD   <INC_WORD_R-HEAD>
                    1915  4380
                    1915  4381          IFEQ    TIMOUT_CNT,ZERO_WORD,2
                21  1915                .BYTE   <H*8> ! 1
          04'4A'    1916                .BYTE   <TIMOUT_CNT-HEAD>&255 , <<TIMOUT_CNT-HEAD>&^XFF00>/256
                11  1918                .BYTE   <D*8> ! 1
          05'96'    1919                .BYTE   <ZERO_WORD-HEAD>&255 , <<ZERO_WORD-HEAD>&^XFF00>/256
          02 0E     191B                .BYTE   <C*8> ! 6 , 2&255
                CD  191D                .BYTE   ^O315
           00CF'    191E                .WORD   <COMPARE-HEAD>
                C2  1920                .BYTE   ^O302
           192B'    1921                .WORD   <2127$-HEAD>
                    1923  4382          MVI     B,<^X24>                  ;OVERFLOW, PRINT ERROR
          24 06     1923                .BYTE   <B*8> ! 6 , ^X24&255
                    1925  4383          CALL    ERROR_ROUTINE
                CD  1925                .BYTE   ^O315
           0C4A'    1926                .WORD   <ERROR_ROUTINE-HEAD>
                    1928  4384          CALL    SET_FOR_CONT_R            ;GO TO COMMAND LEVEL
                CD  1928                .BYTE   ^O315
           107B'    1929                .WORD   <SET_FOR_CONT_R-HEAD>
                    192B  4385          ENDIF
                    192B        2127$:                                    ;GENERATE LCL 2127 NAME
                    192B  4386
                    192B  4387          ELSE
                C3  192B                .BYTE   ^O303
           1931'    192C                .WORD   <2128$-HEAD>
                    192E        2126$:                                    ;GENERATE LOCAL SYMBOL NAME
                    192E  4388          CALL    CHECK_CNTLC               ;IF ERROR LOOPING, CHECK
                CD  192E                .BYTE   ^O315
           1073'    192F                .WORD   <CHECK_CNTLC-HEAD>
                    1931  4389                                            ; FOR ^C FROM OPERATOR
                    1931  4390          ENDIF
                    1931        2128$:                                    ;GENERATE LCL 2128 NAME
                    1931  4391
                    1931  4392          RET
                C9  1931                .BYTE   ^O311
                    1932  4393
                    1932  4394 ;*****************************************************************************
                    1932  4395 ;
                    1932  4396 ;  RESET_TIMOUT
                    1932  4397 ;  THIS ROUTINE RESETS THE TIMEOUT COUNTERS TO ZERO FOR A FULL
                    1932  4398 ;  TIMOUT PERIOD
                    1932  4399 ;
                    1932  4400 ;*****************************************************************************
                    1932  4401
                    1932  4402 RESET_TIMOUT:    LXI     H,TIMOUT_CNT
                21  1932                .BYTE   <H*8> ! 1
          04'4A'    1933                .BYTE   <TIMOUT_CNT-HEAD>&255 , <<TIMOUT_CNT-HEAD>&^XFF00>/256
                    1935  4403          XRA     A
                AF  1935                .BYTE   ^O250 ! A                 ;with A
                    1936  4404          MOV     M,A
                77  1936                .BYTE   ^O100 ! <M*8> ! A
                    1937  4405          INX     H
```

```
                      23  1937                .BYTE   <H*8> ! 3
                          1938  4406                  MOV     M,A
                      77  1938                .BYTE   ^0100 ! <M*8> ! A
                          1939  4407                  RET
                      C9  1939                .BYTE   ^0311
                          193A  4408
                          193A  4409
                          193A  4410 ;*****************************************************************************
                          193A  4411 ;
                          193A  4412 ;  PARITY_ERR
                          193A  4413 ;  THIS ROUTINE GETS CONTROL ON A PARITY ERROR INTERRUPT
                          193A  4414 ;  THE INTERRUPT COULD ALSO BE KNOWN PARITY ERRORS SUCH AS A SOMM
                          193A  4415 ;
                          193A  4416 ;*****************************************************************************
                          193A  4417
                          193A  4418 PARITY_ERR:      PUSH    $PSW                    ;SAVE FLAGS AND ACCUMULATOR
                      F5  193A                .BYTE   ^0305 ! <8*$PSW>
                          193B  4419                  PUSH    B                       ;SAVE BC PAIR
                      C5  193B                .BYTE   ^0305 ! <8*B>
                          193C  4420                  PUSH    D                       ;SAVE DE PAIR
                      D5  193C                .BYTE   ^0305 ! <8*D>
                          193D  4421                  PUSH    H                       ;SAVE HL PAIR
                      E5  193D                .BYTE   ^0305 ! <8*H>
                          193E  4422                  CALL    STOP_CPU                ;STOP THE CLOCK
                      CD  193E                .BYTE   ^0315
                    10C9' 193F                .WORD   <STOP_CPU-HEAD>
                          1941  4423                  LDA     CNT_PAR                 ;GET COUNT OF PARITY ERRORS
                      3A  1941                .BYTE   ^072
                    0254' 1942                .WORD   <CNT_PAR-HEAD>
                          1944  4424                  INR     A                       ;INC COUNT
                      3C  1944                .BYTE   ^04 ! <8*A>
                          1945  4425                  STA     CNT_PAR                 ;UPDATE COUNT
                      32  1945                .BYTE   ^062
                    0254' 1946                .WORD   <CNT_PAR-HEAD>
                          1948  4426
                          1948  4427                  CPI     2                       ;SEE IF PARITY COUNT=2 (SECOND
                      FE  1948                .BYTE   ^0376
                      02  1949                .BYTE   2
                          194A  4428                                                  ; PARITY ERROR OCCURED)
                          194A  4429                  JZ      1$                      ;DON'T TRY SAVE_WR IF SO
                      CA  194A                .BYTE   ^0312
                    1951' 194B                .WORD   <1$-HEAD>
                          194D  4430
                          194D  4431                  EI                              ;REENABLE INTERRUPTS
                      FB  194D                .BYTE   ^0373
                          194E  4432                  CALL    SAVE_WR
                      CD  194E                .BYTE   ^0315
                    0C11' 194F                .WORD   <SAVE_WR-HEAD>
                          1951  4433 1$:               LHLD    SAVED_UPC
                      2A  1951                .BYTE   ^052
                    00C4' 1952                .WORD   <SAVED_UPC-HEAD>
                          1954  4434                  SHLD    HEX_BUF                 ;SAVE UPC
                      22  1954                .BYTE   ^042
                    005B' 1955                .WORD   <HEX_BUF-HEAD>
                          1957  4435                  LXI     H,HEX_BUF               ;POINT TO STORED UPC
                      21  1957                .BYTE   <H*8> ! 1
                    00'5B' 1958                .BYTE   <HEX_BUF-HEAD>&255 , <<HEX_BUF-HEAD>&^XFF00>/256
```

```
                        195A  4436                    CALL    DECR_WORD_R             ;BACK UP UPC..PARITY ERROR
      CD                195A                  .BYTE   ^0315
   12BC'                195B                  .WORD   <DECR_WORD_R-HEAD>
                        195D  4437
                        195D  4438                                                    ; HAS ALREADY REACHED CSR
                        195D  4439                    CALL    CHECK_LS_SETPAR         ;CHECK TO SEE IF PARITY ERROR
      CD                195D                  .BYTE   ^0315
   197D'                195E                  .WORD   <CHECK_LS_SETPAR-HEAD>
                        1960  4440                                                    ; WAS INTENTIONAL
                        1960  4441
                        1960  4442                    IF      LS_PARERR_ACT
      3A                1960                  .BYTE   ^072
    0349'               1961                  .WORD   <LS_PARERR_ACT-HEAD>
      0F                1963                  .BYTE   ^017
      D2                1964                  .BYTE   ^0322
    1970'               1965                  .WORD   <2129$-HEAD>
                        1967  4443                    LXI     H,NOP_INST              ;SET UP TO PUT NOP IN CSR
      21                1967                  .BYTE   <H*8> ! 1
   00'72'               1968                  .BYTE   <NOP_INST-HEAD>&255 , <<NOP_INST-HEAD>&^XFF00>/256
                        196A  4444                    LXI     D,SAVED_CSR
      11                196A                  .BYTE   <D*8> ! 1
   00'BB'               196B                  .BYTE   <SAVED_CSR-HEAD>&255 , <<SAVED_CSR-HEAD>&^XFF00>/256
                        196D  4445                    CALL    MOVER_3_R
      CD                196D                  .BYTE   ^0315
    0F5F'               196E                  .WORD   <MOVER_3_R-HEAD>
                        1970  4446                    ENDIF
                        1970      2129$:                                            ;GENERATE LCL 2129 NAME
                        1970  4447
                        1970  4448                    CLEAR   LS_PARERR_ACT         ;CLEAR FLG IN CASE IT IS SET
      AF                1970                  .BYTE   ^0250 ! A                   ;with A
      32                1971                  .BYTE   ^062
    0349'               1972                  .WORD   <LS_PARERR_ACT-HEAD>
                        1974  4449                    CALL    RESTART_CPU
      CD                1974                  .BYTE   ^0315
    110E'               1975                  .WORD   <RESTART_CPU-HEAD>
                        1977  4450
                        1977  4451                    POP     H                       ;RESTORE HL PAIR
      E1                1977                  .BYTE   ^0301 ! <8*H>
                        1978  4452                    POP     D                       ;RESTORE DE PAIR
      D1                1978                  .BYTE   ^0301 ! <8*D>
                        1979  4453                    POP     B                       ;RESTORE BC PAIR
      C1                1979                  .BYTE   ^0301 ! <8*B>
                        197A  4454                    POP     $PSW                    ;RESTORE FLAGS AND ACCUMULATOR
      F1                197A                  .BYTE   ^0301 ! <8*$PSW>
                        197B  4455
                        197B  4456                    EI                            ;REENABLE INTERRUPTS
      FB                197B                  .BYTE   ^0373
                        197C  4457                    RET
      C9                197C                  .BYTE   ^0311
                        197D  4458
                        197D  4459 ;
                        197D  4460 ; THIS ROUTINE CHECKS TO SEE IF PARITY ERROR WAS CAUSED BY INTENTIONALLY
                        197D  4461 ; PUTTING A PARITY ERROR IN WCS VIA A COMMAND FROM THE CPU MICRO-CODE
                        197D  4462 ;
                        197D  4463
                        197D  4464 CHECK_LS_SETPAR:
                        197D  4465                    IF      LS_SETPARERR
```

```
                3A  197D                .BYTE   ^072
              0348' 197E                .WORD   <LS_SETPARERR-HEAD>
                0F  1980                .BYTE   ^017
                D2  1981                .BYTE   ^0322
              198C' 1982                .WORD   <2130$-HEAD>
                    1984  4466                   SET     LS_PARERR_ACT           ;SET LS PARITY ERROR ACTIVE FLG
                AF  1984                .BYTE   ^0250 ! A               ;with A
                3C  1985                .BYTE   ^04 ! <8*A>
                32  1986                .BYTE   ^062
              0349' 1987                .WORD   <LS_PARERR_ACT-HEAD>
                    1989  4467
                    1989  4468                   ELSE
                C3  1989                .BYTE   ^0303
              1995' 198A                .WORD   <2131$-HEAD>
                    198C        2130$:                                      ;GENERATE LOCAL SYMBOL NAME
                    198C  4469
                    198C  4470                   MVI     B,<^X26>                ;ERROR - WCS PARITY ERROR
             26 06  198C                .BYTE   <B*8> ! 6 , ^X26&255
                    198E  4471                   CALL    ERROR_ROUTINE
                CD  198E                .BYTE   ^0315
              0C4A' 198F                .WORD   <ERROR_ROUTINE-HEAD>
                    1991  4472                   EI                              ;REENABLE INTERRUPTS
                FB  1991                .BYTE   ^0373
                    1992  4473                   CALL    SET_FOR_CONT_R          ;ALLOW CONTINUE FROM HERE
                CD  1992                .BYTE   ^0315
              107B' 1993                .WORD   <SET_FOR_CONT_R-HEAD>
                    1995  4474
                    1995  4475                   ENDIF
                    1995        2131$:                                      ;GENERATE LCL 2131 NAME
                    1995  4476                   RET                             ;BACK TO PAR ERR ROUTINE
                C9  1995                .BYTE   ^0311
                    1996  4477
                    1996  4478 ;*****************************************************************************
                    1996  4479 ;
                    1996  4480 ;  DATA_XFER
                    1996  4481 ;  THIS ROUTINE TRANSFERS DATA FROM THE WCS MEMORY TO LS OR MAIN MEMORY
                    1996  4482 ;  A POINTER IS USED AND INCREMENTED AFTER EACH XFER.  THIS POINTER POINTS
                    1996  4483 ;  TO ANOTHER POINTER THAT POINTS TO A DATA CONTROL BLOCK.  THE DATA
                    1996  4484 ;  CONTROL BLOCK CONTAINS THE BYTE COUNT AND  DESTINATION ADDRESS OF THE
                    1996  4485 ;  DATA.  THE POINTER IN WCS ALSO CONTAINS ONE BIT THAT TELLS IF THE
                    1996  4486 ;  TRANSFER IS FOR LS OR MAIN MEMORY.
                    1996  4487 ;
                    1996  4488 ;*****************************************************************************
                    1996  4489
                    1996  4490 DATA_XFER:       SET     DATA_XFER_FLG           ;SET XFER FLAG
                AF  1996                .BYTE   ^0250 ! A               ;with A
                3C  1997                .BYTE   ^04 ! <8*A>
                32  1998                .BYTE   ^062
              025E' 1999                .WORD   <DATA_XFER_FLG-HEAD>
                    199B  4491
                    199B  4492                   CALL    READ_LS_7               ;READ THE POINTER
                CD  199B                .BYTE   ^0315
              1165' 199C                .WORD   <READ_LS_7-HEAD>
                    199E  4493
                    199E  4494                   LXI     H,DATA2
                21  199E                .BYTE   <H*8> ! 1
            00'AB'  199F                .BYTE   <DATA2-HEAD>&255 , <<DATA2-HEAD>&^XFF00>/256
```

```
                    19A1  4495                    MOV     A,M
                7E  19A1                .BYTE   ^O100 ! <A*8> ! M
                    19A2  4496                    ANI     HEX_3F
                E6  19A2                .BYTE   ^O346
                3F  19A3                .BYTE   HEX_3F
                    19A4  4497                    STA     WCS_ADDRESS
                32  19A4                .BYTE   ^O62
            006C'   19A5                .WORD   <WCS_ADDRESS-HEAD>
                    19A7  4498                    INX     H
                23  19A7                .BYTE   <H*8> ! 3
                    19A8  4499                    MOV     A,M
                7E  19A8                .BYTE   ^O100 ! <A*8> ! M
                    19A9  4500                    STA     WCS_ADDRESS+1           ;SAVE POINTER FOR READ
                32  19A9                .BYTE   ^O62
            006D'   19AA                .WORD   <WCS_ADDRESS+1-HEAD>
                    19AC  4501
                    19AC  4502                    CALL    READ_WCS_R              ;READ FIRST DATA WORD(COUNT)
                CD  19AC                .BYTE   ^O315
            0E6D'   19AD                .WORD   <READ_WCS_R-HEAD>
                    19AF  4503
                    19AF  4504                    LHLD    WCS_VALUE+1
                2A  19AF                 BYTE   ^O52
            00B5'   1980                .WORD   <WCS_VALUE+1-HEAD>
                    1982  4505                    SHLD    XFER_CNT                ;NUMBER OF BYTES TO XFER
                22  1982                .BYTE   ^O42
            058D'   1983                .WORD   <XFER_CNT-HEAD>
                    1985  4506
                    1985  4507                    CLEAR   XFER_MM                 ;SET TRANSFER TO LS TO START
                AF  1985                .BYTE   ^O250 ! A               ;with A
                32  1986                .BYTE   ^O62
            0592'   1987                .WORD   <XFER_MM-HEAD>
                    1989  4508
                    1989  4509                    IFMBI   WCS_VALUE,MM_OR_LS
                3A  1989                .BYTE   ^O72
            00B4'   198A                .WORD   <WCS_VALUE-HEAD>
                E6  198C                .BYTE   ^O346
                01  198D                .BYTE   MM_OR_LS
                CA  198E                .BYTE   ^O312
            1906'   198F                .WORD   <2132$-HEAD>
                    1991  4510                    SET     XFER_MM                 ;TRANSFER IS TO MM, SET FLAG
                AF  1991                .BYTE   ^O250 ! A               ;with A
                3C  1992                .BYTE   ^O04 ! <8*A>
                32  1993                .BYTE   ^O62
            0592'   1994                .WORD   <XFER_MM-HEAD>
                    1996  4511                    ENDIF
                    1996        2132$:                                      ;GENERATE LCL 2132 NAME
                    1996  4512
                    1996  4513                    LXI     H,WCS_ADDRESS
                21  1996                .BYTE   <H*8> ! 1
            00'6C'  1997                .BYTE   <WCS_ADDRESS-HEAD>&255 , <<WCS_ADDRESS-HEAD>&^XFF00>/256
                    1999  4514                    CALL    INC_WORD_R
                CD  1999                .BYTE   ^O315
            12B1'   199A                .WORD   <INC_WORD_R-HEAD>
                    199C  4515                    CALL    READ_WCS_R              ;READ DEST ADDRESS
                CD  199C                .BYTE   ^O315
            0E6D'   199D                .WORD   <READ_WCS_R-HEAD>
                    199F  4516
```

```
                    19CF  4517                      LXI     H,WCS_VALUE
                 21 19CF                    .BYTE   <H*8> ! 1
               00'B4' 19D0                  .BYTE   <WCS_VALUE-HEAD>&255 , <<WCS_VALUE-HEAD>&^XFF00>/256
                    19D2  4518                      LXI     D,XFER_DEST
                 11 19D2                    .BYTE   <D*8> ! 1
               05'8F' 19D3                  .BYTE   <XFER_DEST-HEAD>&255 , <<XFER_DEST-HEAD>&^XFF00>/256
                    19D5  4519                      CALL    MOVER_3_R               ;SAVE DEST ADDRESS
                 CD 19D5                    .BYTE   ^O315
               0F5F' 19D6                   .WORD   <MOVER_3_R-HEAD>
                    19D8  4520
                    19D8  4521                      LXI     H,WCS_VALUE             ;SAVE DEST ADDRESS IN DATA
                 21 19D8                    .BYTE   <H*8> ! 1
               00'B4' 19D9                  .BYTE   <WCS_VALUE-HEAD>&255 , <<WCS_VALUE-HEAD>&^XFF00>/256
                    19DB  4522                      LXI     D,DATA1                 ;AREA FOR WRITE TO LS ON
                 11 19DB                    .BYTE   <D*8> ! 1
               00'AA' 19DC                  .BYTE   <DATA1-HEAD>&255 , <<DATA1-HEAD>&^XFF00>/256
                    19DE  4523                                                      ;MM XFER
                    19DE  4524                      CALL    MOVER_3_R
                 CD 19DE                    .BYTE   ^O315
               0F5F' 19DF                   .WORD   <MOVER_3_R-HEAD>
                    19E1  4525                      XRA     A
                 AF 19E1                    .BYTE   ^O250 ! A               ;with A
                    19E2  4526                      STA     DATA0                   ;ZERO OUT TOP BYTE
                 32 19E2                    .BYTE   ^O62
               00A9' 19E3                   .WORD   <DATA0-HEAD>
                    19E5  4527
                    19E5  4528                      IF      XFER_MM
                 3A 19E5                    .BYTE   ^O72
               0592' 19E6                   .WORD   <XFER_MM-HEAD>
                 0F 19E8                    .BYTE   ^O17
                 D2 19E9                    .BYTE   ^O322
               19F1' 19EA                   .WORD   <2133$-HEAD>
                    19EC  4529
                    19EC  4530                      MVI     A,LS_5                  ;LS ADDRESS TO WRITE
              05 3E 19EC                    .BYTE   <A*8> ! 6 , LS_5&255
                    19EE  4531                      CALL    WRITE_LS_R              ;WITH MM ADDRESS
                 CD 19EE                    .BYTE   ^O315
               1146' 19EF                   .WORD   <WRITE_LS_R-HEAD>
                    19F1  4532
                    19F1  4533                      ENDIF
                    19F1        2133$:                                              ;GENERATE LCL 2133 NAME
                    19F1  4534
                    19F1  4535                      LHLD    WCS_ADDRESS
                 2A 19F1                    .BYTE   ^O52
               006C' 19F2                   .WORD   <WCS_ADDRESS-HEAD>
                    19F4  4536                      SHLD    XFER_PNT
                 22 19F4                    .BYTE   ^O42
               0593' 19F5                   .WORD   <XFER_PNT-HEAD>
                    19F7  4537
                    19F7  4538                      WHILER  MORE_DATA               ;LOOP CONTROL BYTE
                 AF 19F7                    .BYTE   ^O250 ! A               ;with A
                 3C 19F8                    .BYTE   ^O4 ! <8*A>
                 32 19F9                    .BYTE   ^O62
               03AF' 19FA                   .WORD   <MORE_DATA-HEAD>
                    19FC        4003$:      LDA     MORE_DATA
                 3A 19FC                    .BYTE   ^O72
               03AF' 19FD                   .WORD   <MORE_DATA-HEAD>
```

```
                    OF  19FF                .BYTE   ^017
                    D2  1A00                .BYTE   ^0322
                1A28' 1A01                  .WORD   <2134$-HEAD>
                      1A03  4539                    CALL    FOUR_BYTES              ;READ LONG WORD OF DATA (WCS)
                    CD  1A03                .BYTE   ^0315
                1A29' 1A04                  .WORD   <FOUR_BYTES-HEAD>
                      1A06  4540
                      1A06  4541                    IF      XFER_MM
                    3A  1A06                .BYTE   ^072
                0592' 1A07                  .WORD   <XFER_MM-HEAD>
                    OF  1A09                .BYTE   ^017
                    D2  1A0A                .BYTE   ^0322
                1A1B' 1A0B                  .WORD   <2135$-HEAD>
                      1A0D  4542                    MVI     A,LS_6                  ;LS DATA TO WRITE
                06 3E 1A0D                .BYTE   <A*8> ! 6 , LS_6&255
                      1A0F  4543                    CALL    WRITE_LS_R              ;WITH DATA FOR MM
                    CD  1A0F                .BYTE   ^0315
                1146' 1A10                  .WORD   <WRITE_LS_R-HEAD>
                      1A12  4544
                      1A12  4545                    LXI     H,WCS_DE_MM
                    21  1A12                .BYTE   <H*8> ! 1
                00 54 1A13                .BYTE   <WCS_DE_MM-HEAD>&255 , <<WCS_DE_MM-HEAD>&^XFF00>/256
                      1A15  4546                    CALL    EXEC_SUB                ;DO THE SUBROUTINE
                    CD  1A15                .BYTE   ^0315
                0B7D' 1A16                  .WORD   <EXEC_SUB-HEAD>
                      1A18  4547
                      1A18  4548                    ELSE
                    C3  1A18                .BYTE   ^0303
                1A25' 1A19                  .WORD   <2136$-HEAD>
                      1A1B        2135$:                                          ;GENERATE LOCAL SYMBOL NAME
                      1A1B  4549
                      1A1B  4550                    LDA     XFER_DEST+2             ;ADDRESS TO WRITE IN LS
                    3A  1A1B                .BYTE   ^072
                0591' 1A1C                  .WORD   <XFER_DEST+2-HEAD>
                      1A1E  4551                    CALL    WRITE_LS_R
                    CD  1A1E                .BYTE   ^0315
                1146' 1A1F                  .WORD   <WRITE_LS_R-HEAD>
                      1A21  4552
                      1A21  4553                    LXI     H,XFER_DEST+2           ;INC DEST POINTER FOR LS
                    21  1A21                .BYTE   <H*8> ! 1
                05'91' 1A22                 .BYTE   <XFER_DEST+2-HEAD>&255 , <<XFER_DEST+2-HEAD>&^XFF00>/256
                      1A24  4554                    INR     M                       ;WRITE(MM IS AUTOMATIC)
                    34  1A24                .BYTE   ^04 ! <8*M>
                      1A25  4555
                      1A25  4556                    ENDIF
                      1A25        2136$:                                          ;GENERATE LCL 2136 NAME
                      1A25  4557
                      1A25  4558                    ENDWHILE
                    C3  1A25                .BYTE   ^0303
                19FC' 1A26                  .WORD   <4003$-HEAD>
                      1A28        2134$:                                          ;GENERATE LCL 2134 NAME
                      1A28  4559
                      1A28  4560                    RET
                    C9  1A28                .BYTE   ^0311
                      1A29  4561
                      1A29  4562
                      1A29  4563
```

```
                     1A29  4564
                     1A29  4565 ;*****************************************************************************
                     1A29  4566 ;
                     1A29  4567 ;  FOUR_BYTES
                     1A29  4568 ;  THIS ROUTINE READS FOUR BYTES OF DATA FROM WCS USING XFER_PNT
                     1A29  4569 ;  THE DATA IS MOVED TO THE DATA0,1,2,3 AREAS.  IF THE COUNT XFER_CNT
                     1A29  4570 ;  RUNS OUT THE LOOP CONTROL FLAG MORE_DATA IS CLEARED
                     1A29  4571 ;
                     1A29  4572 ;*****************************************************************************
                     1A29  4573
                     1A29  4574 FOUR_BYTES:      CALL    READ_WCS_DATA           ;GET 2 BYTES OF DATA
          CD         1A29                .BYTE   ^0315
      1A4E'          1A2A                .WORD   <READ_WCS_DATA-HEAD>
                     1A2C  4575                  SHLD    DATA2
          22         1A2C                .BYTE   ^042
      00AB'          1A2D                .WORD   <DATA2-HEAD>
                     1A2F  4576
                     1A2F  4577                  CALL    READ_WCS_DATA           ;GET 2 BYTES OF DATA
          CD         1A2F                .BYTE   ^0315
      1A4E'          1A30                .WORD   <READ_WCS_DATA-HEAD>
                     1A32  4578                  SHLD    DATA0
          22         1A32                .BYTE   ^042
      00A9'          1A33                .WORD   <DATA0-HEAD>
                     1A35  4579
                     1A35  4580                  LXI     H,XFER_CNT
          21         1A35                .BYTE   <H*8> ! 1
    05'8D'           1A36                .BYTE   <XFER_CNT-HEAD>&255 , <<XFER_CNT-HEAD>&^XFF00>/256
                     1A38  4581                  CALL    DECR_WORD_R
          CD         1A38                .BYTE   ^0315
      12BC'          1A39                .WORD   <DECR_WORD_R-HEAD>
                     1A3B  4582
                     1A3B  4583                  IFEQ    XFER_CNT,ZERO_WORD,2
          21         1A3B                .BYTE   <H*8> ! 1
    05'8D'           1A3C                .BYTE   <XFER_CNT-HEAD>&255 , <<XFER_CNT-HEAD>&^XFF00>/256
          11         1A3E                .BYTE   <D*8> ! 1
    05'96'           1A3F                .BYTE   <ZERO_WORD-HEAD>&255 , <<ZERO_WORD-HEAD>&^XFF00>/256
    02 0E            1A41                .BYTE   <C*8> ! 6 , 2&255
          CD         1A43                .BYTE   ^0315
      00CF'          1A44                .WORD   <COMPARE-HEAD>
          C2         1A46                .BYTE   ^0302
      1A4D'          1A47                .WORD   <2137$-HEAD>
                     1A49  4584
                     1A49  4585                  CLEAR   MORE_DATA
          AF         1A49                .BYTE   ^0250 ! A               ;with A
          32         1A4A                .BYTE   ^062
      03AF'          1A4B                .WORD   <MORE_DATA-HEAD>
                     1A4D  4586
                     1A4D  4587                  ENDIF
                     1A4D        2137$:                                  ;GENERATE LCL 2137 NAME
                     1A4D  4588
                     1A4D  4589                  RET
          C9         1A4D                .BYTE   ^0311
                     1A4E  4590
                     1A4E  4591
                     1A4E  4592 READ_WCS_DATA:
                     1A4E  4593                  LXI     H,XFER_PNT              ;INC POINTER AND READ DATA
          21         1A4E                .BYTE   <H*8> ! 1
```

```
                05'93' 1A4F                     .BYTE   <XFER_PNT-HEAD>&255 , <<XFER_PNT-HEAD>&^XFF00>/256
                       1A51  4594                       CALL    INC_WORD_R              ;TO TRANSFER TO WCS
                  CD   1A51                       .BYTE   ^0315
                12B1'  1A52                       .WORD   <INC_WORD_R-HEAD>
                       1A54  4595                       LHLD    XFER_PNT
                  2A   1A54                       .BYTE   ^052
                0593'  1A55                       .WORD   <XFER_PNT-HEAD>
                       1A57  4596                       SHLD    WCS_ADDRESS             ;HIGH VALUE
                  22   1A57                       .BYTE   ^042
                006C'  1A58                       .WORD   <WCS_ADDRESS-HEAD>
                       1A5A  4597                       CALL    READ_WCS_R
                  CD   1A5A                       .BYTE   ^0315
                0E6D'  1A5B                       .WORD   <READ_WCS_R-HEAD>
                       1A5D  4598
                       1A5C  4599                       LHLD    WCS_VALUE+1
                  2A   1A5D                       .BYTE   ^052
                00B5'  1A5E                       .WORD   <WCS_VALUE+1-HEAD>
                       1A60  4600                       RET
                  C9   1A60                       .BYTE   ^0311
                       1A61  4601
                       1A61  4602
                       1A61  4603 ;*****************************************************************************
                       1A61  4604 ;
                       1A61  4605 ;  CLEAR_CPU_ATT_R
                       1A61  4606 ;  THIS ROUTINE EXECUTES AN INSTRUCTION IN THE CSR THAT CLEARS
                       1A61  4607 ;  BOTH CPU ATTENTION AND CPU ACKNOWLEDGE
                       1A61  4608 ;
                       1A61  4609 ;*****************************************************************************
                       1A61  4610
                       1A61  4611 CLEAR_CPU_ATT_R:
                       1A61  4612                       CALL    LOAD_UPC_R              ;SAVE UPC IN ATT_UPC_SAVE
                  CD   1A61                       .BYTE   ^0315
                0F19'  1A62                       .WORD   <LOAD_UPC_R-HEAD>
                       1A64  4613                       LXI     H,UPC_VALUE
                  21   1A64                       .BYTE   <H*8> ! 1
                00'C2' 1A65                       .BYTE   <UPC_VALUE-HEAD>&255 , <<UPC_VALUE-HEAD>&^XFF00>/256
                       1A67  4614                       LXI     D,ATT_UPC_SAVE
                  11   1A67                       .BYTE   <D*8> ! 1
                00'CC' 1A68                       .BYTE   <ATT_UPC_SAVE-HEAD>&255 , <<ATT_UPC_SAVE-HEAD>&^XFF00>/256
                       1A6A  4615                       CALL    MOVER_3_R
                  CD   1A6A                       .BYTE   ^0315
                0F5F'  1A6B                       .WORD   <MOVER_3_R-HEAD>
                       1A6D  4616
                       1A6D  4617                       LXI     H,X_CLR_CPU_ATT         ;ADDRESS OF INSTRUCTION
                  21   1A6D                       .BYTE   <H*8> ! 1
                06'6A' 1A6E                       .BYTE   <X_CLR_CPU_ATT-HEAD>&255 , <<X_CLR_CPU_ATT-HEAD>&^XFF00>/256
                       1A70  4618                       CALL    PERFORM_CSR_R
                  CD   1A70                       .BYTE   ^0315
                12F2'  1A71                       .WORD   <PERFORM_CSR_R-HEAD>
                       1A73  4619
                       1A73  4620                       LXI     H,ATT_UPC_SAVE
                  21   1A73                       .BYTE   <H*8> ! 1
                00'CC' 1A74                       .BYTE   <ATT_UPC_SAVE-HEAD>&255 , <<ATT_UPC_SAVE-HEAD>&^XFF00>/256
                       1A76  4621                       LXI     D,UPC_VALUE
                  11   1A76                       .BYTE   <D*8> ! 1
                00'C2' 1A77                       .BYTE   <UPC_VALUE-HEAD>&255 , <<UPC_VALUE-HEAD>&^XFF00>/256
                       1A79  4622                       CALL    MOVER_3_R
```

```
                CD  1A79                .BYTE   ^0315
          0F5F' 1A7A                    .WORD   <MOVER_3_R-HEAD>
                1A7C  4623              CALL    LOAD_UPC_R
                CD  1A7C                .BYTE   ^0315
          0F19' 1A7D                    .WORD   <LOAD_UPC_R-HEAD>
                1A7F  4624
                1A7F  4625              RET
                C9  1A7F                .BYTE   ^0311
                1A80  4626
                1A80  4627
                1A80  4628 ;*****************************************************************************
                1A80  4629 ;
                1A80  4630 ;  LOAD_SECTION
                1A80  4631 ;  THIS ROUTINE LOADS A SECTION INTO WCS OR INTO CONSOLE RAM
                1A80  4632 ;  IF THE LOAD IS TO WCS A FLAG IS SET AND THE DATA TRANSFER BRANCH
                1A80  4633 ;  AT ADDRESS 0 OF THE WCS IS TAKEN.
                1A80  4634 ;
                1A80  4635 ;*****************************************************************************
                1A80  4636
                1A80  4637 MOV_LOAD_SECTION:
                1A80  4638                      LXI     H,TEMP_FLAG             ;POINTER TO TEMP SECTION BUFFER
                  21  1A80                .BYTE   <H*8> ! 1
          04'1F' 1A81                   .BYTE   <TEMP_FLAG-HEAD>&255 , <<TEMP_FLAG-HEAD>&^XFF00>/256
                1A83  4639              LXI     D,SECTION_FLAG          ;POINTER TO SECTION BUFFER
                  11  1A83                .BYTE   <D*8> ! 1
          08'C9' 1A84                   .BYTE   <SECTION_FLAG-HEAD>&255 , <<SECTION_FLAG-HEAD>&^XFF00>/256
                1A86  4640              MVI     C,SEC_LEN+1             ;COUNT
          07 0E 1A86                    .BYTE   <C*8> ! 6 , SEC_LEN+1&255
                1A88  4641              CALL    MOVER_R                 ;MOVE ENTRIES TO LOCAL BUFFER
                CD  1A88                .BYTE   ^0315
          0F75' 1A89                    .WORD   <MOVER_R-HEAD>
                1A8B  4642
                1A8B  4643 LOAD_SECTION:        LXI     H,SECTION_NAME
                  21  1A8B                .BYTE   <H*8> ! 1
          08'CA' 1A8C                   .BYTE   <SECTION_NAME-HEAD>&255 , <<SECTION_NAME-HEAD>&^XFF00>/256
                1A8E  4644              LXI     D,SEC_PARMB_NAM
                  11  1A8E                .BYTE   <D*8> ! 1
          41 07 1A8F                    .BYTE   <SEC_PARMB_NAM-HEAD>&255 , <<SEC_PARMB_NAM-HEAD>&^XFF00>/256
                1A91  4645              MVI     C,SEC_LEN               ;MOVE SECTION NAME TO TU58
          06 0E 1A91                    .BYTE   <C*8> ! 6 , SEC_LEN&255
                1A93  4646              CALL    MOVER_R                 ;PARAMETER BLOCK
                CD  1A93                .BYTE   ^0315
          0F75' 1A94                    .WORD   <MOVER_R-HEAD>
                1A96  4647
                1A96  4648              LXI     H,EXT_A                 ;MOVE EXT TO PARM BLOCK
                  21  1A96                .BYTE   <H*8> ! 1
          02'EB' 1A97                   .BYTE   <EXT_A-HEAD>&255 , <<EXT_A-HEAD>&^XFF00>/256
                1A99  4649              LXI     D,SEC_PARMB_EXT
                  11  1A99                .BYTE   <D*8> ! 1
          41 0D 1A9A                    .BYTE   <SEC_PARMB_EXT-HEAD>&255 , <<SEC_PARMB_EXT-HEAD>&^XFF00>/256
                1A9C  4650              CALL    MOVER_4_R
                CD  1A9C                .BYTE   ^0315
          0F73' 1A9D                    .WORD   <MOVER_4_R-HEAD>
                1A9F  4651
                1A9F  4652              CLEAR   SEC_PARMB_ADD+2
                  AF  1A9F                .BYTE   ^0250 ! A               ;with A
                  32  1AA0                .BYTE   ^062
```

```
                4113  1AA1                .WORD   <SEC_PARMB_ADD+2-HEAD>
                      1AA3  4653                  CLEAR   SEC_PARMB_ADD+3         ;CLR MOST SIG. 2BYTES(UNUSED)
                  AF  1AA3                .BYTE   ^O250 ! A               ;with A
                  32  1AA4                .BYTE   ^O62
                4114  1AA5                .WORD   <SEC_PARMB_ADD+3-HEAD>
                      1AA7  4654
                      1AA7  4655                  CLEAR   FAST_WRITE
                  AF  1AA7                .BYTE   ^O250 ! A               ;with A
                  32  1AA8                .BYTE   ^O62
              02FC'   1AA9                .WORD   <FAST_WRITE-HEAD>
                      1AAB  4656                  CLEAR   WCS_LOADED
                  AF  1AAB                .BYTE   ^O250 ! A               ;with A
                  32  1AAC                .BYTE   ^O62
              058B'   1AAD                .WORD   <WCS_LOADED-HEAD>
                      1AAF  4657                  CLEAR   CONSOLE_TEST
                  AF  1AAF                .BYTE   ^O250 ! A               ;with A
                  32  1AB0                .BYTE   ^O62
              0255'   1AB1                .WORD   <CONSOLE_TEST-HEAD>
                      1AB3  4658
                      1AB3  4659                  IFMBI   SECTION_FLAG,CON_RAM
                  3A  1AB3                .BYTE   ^O72
              08C9'   1AB4                .WORD   <SECTION_FLAG-HEAD>
                  E6  1AB6                .BYTE   ^O346
                  80  1AB7                .BYTE   CON_RAM
                  CA  1AB8                .BYTE   ^O312
              1AC6'   1AB9                .WORD   <2138$-HEAD>
                      1ABB  4660
                      1ABB  4661                  MVI     A,6                     ;DRIVER HAS 6 = RAM LOAD
            06 3E     1ABB                .BYTE   <A*8> ! 6 , 6&255
                      1ABD  4662                  STA     SEC_PARMB_DST           ;STORE IN DESTINATION CODE
                  32  1ABD                .BYTE   ^O62
                4115  1ABE                .WORD   <SEC_PARMB_DST-HEAD>
                      1AC0  4663                  LXI     H,RAM_LOAD_ADD
                  21  1AC0                .BYTE   <H*8> ! 1
            70 00     1AC1                .BYTE   <RAM_LOAD_ADD-HEAD>&255 , <<RAM_LOAD_ADD-HEAD>&^XFF00>/256
                      1AC3  4664
                      1AC3  4665                  ELSE
                  C3  1AC3                .BYTE   ^O303
              1ACE'   1AC4                .WORD   <2139$-HEAD>
                      1AC6          2138$:                                        ;GENERATE LOCAL SYMBOL NAME
                      1AC6  4666
                      1AC6  4667                  MVI     A,7                     ;DRIVER HAS 7 = WCS LOAD
            07 3E     1AC6                .BYTE   <A*8> ! 6 , 7&255
                      1AC8  4668                  STA     SEC_PARMB_DST           ;STORE IN DESTINATION CODE
                  32  1AC8                .BYTE   ^O62
                4115  1AC9                .WORD   <SEC_PARMB_DST-HEAD>
                      1ACB  4669                  LXI     H,WCS_LOAD_ADD
                  21  1ACB                .BYTE   <H*8> ! 1
            00 00     1ACC                .BYTE   <WCS_LOAD_ADD-HEAD>&255 , <<WCS_LOAD_ADD-HEAD>&^XFF00>/256
                      1ACE  4670
                      1ACE  4671                  ENDIF
                      1ACE          2139$:                                        ;GENERATE LCL 2139 NAME
                      1ACE  4672
                      1ACE  4673                  SHLD    SEC_PARMB_ADD           ;STORE LOAD ADDR IN PARM BLOCK
                  22  1ACE                .BYTE   ^O42
                4111  1ACF                .WORD   <SEC_PARMB_ADD-HEAD>
                      1AD1  4674
```

```
                 1AD1  4675                    LDA     SEC_PARMB_DEF           ;GET DEFAULT TU58 UNIT (THIS
      3A         1AD1                  .BYTE   ^O72
    4150         1AD2                  .WORD   <SEC_PARMB_DEF-HEAD>
                 1AD4  4676                                                    ; UNIT)
                 1AD4  4677                    STA     SEC_PARMB_UNIT          ;PARM BLOCK ADDRESS FOR UNIT
      32         1AD4                  .BYTE   ^O62
    4106         1AD5                  .WORD   <SEC_PARMB_UNIT-HEAD>
                 1AD7  4678
                 1AD7  4679                    CALL    TU58_DRIVER             ;GET THE FILE
      CD         1AD7                  .BYTE   ^O315
    4117         1AD8                  .WORD   <TU58_DRIVER-HEAD>
                 1ADA  4680                    OUT     DISMEM                  ;DISABLE MEM REF (TU58 DRIVER
   A7 D3         1ADA                  .BYTE   ^O323 , DISMEM&255      ;register A
                 1ADC  4681                                                    ; ENABLED IT AFTER LOAD)
                 1ADC  4682
                 1ADC  4683                    IFNEQI  SEC_PARMB_ERR,0         ;CHECK FOR TU58 ERROR
      3A         1ADC                  .BYTE   ^O72
    4116         1ADD                  .WORD   <SEC_PARMB_ERR-HEAD>
      FE         1ADF                  .BYTE   ^O376
      00         1AE0                  .BYTE   0
      CA         1AE1                  .BYTE   ^O312
   1AF5'         1AE2                  .WORD   <2140$-HEAD>
                 1AE4  4684
                 1AE4  4685                    CLEAR   FAST_WRITE
      AF         1AE4                  .BYTE   ^O250 ! A               ;with A
      32         1AE5                  .BYTE   ^O62
   02FC'         1AE6                  .WORD   <FAST_WRITE-HEAD>
                 1AE8  4686                    CLEAR   WCS_LOADED
      AF         1AE8                  .BYTE   ^O250 ! A               ;with A
      32         1AE9                  .BYTE   ^O62
   058B'         1AEA                  .WORD   <WCS_LOADED-HEAD>
                 1AEC  4687                    CLEAR   EXECUTE                 ;ALLOW PARSER TO PROMPT
      AF         1AEC                  .BYTE   ^O250 ! A               ;with A
      32         1AED                  .BYTE   ^O62
   02EA'         1AEE                  .WORD   <EXECUTE-HEAD>
                 1AF0  4688                    MVI     B,<^X25>
   25 06         1AF0                  .BYTE   <B*8> ! 6 , ^X25&255
                 1AF2  4689                    JMP     ERROR_ROUTINE_NO_RET    ;TU58 ERROR.  DON'T RETURN
      C3         1AF2                  .BYTE   ^O303
   0C45'         1AF3                  .WORD   <ERROR_ROUTINE_NO_RET-HEAD>
                 1AF5  4690                    ENDIF
                 1AF5        2140$:                                            ;GENERATE LCL 2140 NAME
                 1AF5  4691
                 1AF5  4692                    SET     FILE_LOADED             ;SECTION WAS LOADED OK
      AF         1AF5                  .BYTE   ^O250 ! A               ;with A
      3C         1AF6                  .BYTE   ^O04 ! <8*A>
      32         1AF7                  .BYTE   ^O62
   02FD'         1AF8                  .WORD   <FILE_LOADED-HEAD>
                 1AFA  4693
                 1AFA  4694                    IFMBI   SECTION_FLAG,CON_RAM
      3A         1AFA                  .BYTE   ^O72
   08C9'         1AFB                  .WORD   <SECTION_FLAG-HEAD>
      E6         1AFD                  .BYTE   ^O346
      80         1AFE                  .BYTE   CON_RAM
      CA         1AFF                  .BYTE   ^O312
   1B0A'         1B00                  .WORD   <2141$-HEAD>
                 1B02  4695                    SET     CONSOLE_TEST
```

```
                     AF  1B02                .BYTE   ^O250 ! A               ;with A
                     3C  1B03                .BYTE   ^O4 ! <8*A>
                     32  1B04                .BYTE   ^O62
                   0255' 1B05                .WORD   <CONSOLE_TEST-HEAD>
                         1B07  4696
                         1B07  4697                    ELSE
                     C3  1B07                .BYTE   ^O303
                   1B15' 1B08                .WORD   <2142$-HEAD>
                         1B0A        2141$:                                  ;GENERATE LOCAL SYMBOL NAME
                         1B0A  4698                    SET     WCS_LOADED
                     AF  1B0A                .BYTE   ^O250 ! A               ;with A
                     3C  1B0B                .BYTE   ^O4 ! <8*A>
                     32  1B0C                .BYTE   ^O62
                   058B' 1B0D                .WORD   <WCS_LOADED-HEAD>
                         1B0F  4699                    CALL    EXEC_TEST0
                     CD  1B0F                .BYTE   ^O315
                   1BA1' 1B10                .WORD   <EXEC_TEST0-HEAD>
                         1B12  4700                    CALL    NOP_CSR_R
                     CD  1B12                .BYTE   ^O315
                   0ECA' 1B13                .WORD   <NOP_CSR_R-HEAD>
                         1B15  4701
                         1B15  4702                    ENDIF
                         1B15        2142$:                                  ;GENERATE LCL 2142 NAME
                         1B15  4703
                         1B15  4704                    RET
                     C9  1B15                .BYTE   ^O311
                         1B16  4705
                         1B16  4706
                         1B16  4707 ;*****************************************************************************
                         1B16  4708 ;
                         1B16  4709 ;  VERIFY_WRITER
                         1B16  4710 ;  THIS ROUTINE SINGLE STEPS THROUGH THE WRITE DATA 32 MICRO CODE
                         1B16  4711 ;  AND COMPARES IT TO AN EXPECTED VALUE TABLE
                         1B16  4712 ;
                         1B16  4713 ;*****************************************************************************
                         1B16  4714
                         1B16  4715 VERIFY_WRITER:
                         1B16  4716                    SET     FAST_WRITE              ;ERROR WILL CLEAR FAST WRITE
                     AF  1B16                .BYTE   ^O250 ! A               ;with A
                     3C  1B17                .BYTE   ^O4 ! <8*A>
                     32  1B18                .BYTE   ^O62
                   02FC' 1B19                .WORD   <FAST_WRITE-HEAD>
                         1B1B  4717                    LXI     H,WCS_DAT_32_ADD
                     21  1B1B                .BYTE   <H*8> ! 1
                  06 00  1B1C                .BYTE   <WCS_DAT_32_ADD-HEAD>&255 , <<WCS_DAT_32_ADD-HEAD>&^XFF00>/256
                         1B1E  4718                    MOV     A,H
                     7C  1B1E                .BYTE   ^O100 ! <A*8> ! H
                         1B1F  4719                    MOV     H,L
                     65  1B1F                .BYTE   ^O100 ! <H*8> ! L
                         1B20  4720                    MOV     L,A
                     6F  1B20                .BYTE   ^O100 ! <L*8> ! A
                         1B21  4721                    SHLD    UPC_VALUE
                     22  1B21                .BYTE   ^O42
                   00C2' 1B22                .WORD   <UPC_VALUE-HEAD>
                         1B24  4722                    CALL    LOAD_UPC_R
                     CD  1B24                .BYTE   ^O315
                   0F19' 1B25                .WORD   <LOAD_UPC_R-HEAD>
```

```
                1B27  4723
                1B27  4724                      CALL    NOP_CSR_R
          CD    1B27                    .BYTE   ^0315
       0ECA'    1B28                    .WORD   <NOP_CSR_R-HEAD>
                1B2A  4725                      LXI     H,WRITE_TAB
          21    1B2A                    .BYTE   <H*8> ! 1
       09'21'   1B2B                    .BYTE   <WRITE_TAB-HEAD>&255 , <<WRITE_TAB-HEAD>&^XFF00>/256
                1B2D  4726                      SHLD    WRITE_TAB_PNT
          22    1B2D                    .BYTE   ^042
        006F'   1B2E                    .WORD   <WRITE_TAB_PNT-HEAD>
                1B30  4727                      OUT     CLRATN
       AB D3    1B30                    .BYTE   ^0323 , CLRATN&255      ;register A
                1B32  4728
                1B32  4729                      DO      5
          21    1B32                    .BYTE   <H*8> ! 1
       00'55'   1B33                    .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
          46    1B35                    .BYTE   ^0100 ! <B*8> ! M
          C5    1B36                    .BYTE   ^0305 ! <8*B>
       05 3E    1B37                    .BYTE   <A*8> ! 6 , 5&255
          77    1B39                    .BYTE   ^0100 ! <M*8> ! A
                1B3A          2143$:
                1B3A  4730
                1B3A  4731                      CALL    MICRO_STEP_CPU_R
          CD    1B3A                    .BYTE   ^0315
       12A0'    1B3B                    .WORD   <MICRO_STEP_CPU_R-HEAD>
                1B3D  4732                      CALL    CHECK_UPC
          CD    1B3D                    .BYTE   ^0315
       1B73'    1B3E                    .WORD   <CHECK_UPC-HEAD>
                1B40  4733
                1B40  4734                      LDA     FAST_WRITE              ;GET FAST WRITE FLAG
          3A    1B40                    .BYTE   ^072
       02FC'    1B41                    .WORD   <FAST_WRITE-HEAD>
                1B43  4735                      ORA     A                       ;SET CONDITION CODES
          B7    1B43                    .BYTE   ^0260 ! A
                1B44  4736                      JZ      LEAVE_DO                ;LEAVE DO LOOP IF FAST_WRITE
          CA    1B44                    .BYTE   ^0312
       1B71'    1B45                    .WORD   <LEAVE_DO-HEAD>
                1B47  4737                                                      ; CLEAR
                1B47  4738                      ENDDO
          21    1B47                    .BYTE   <H*8> ! 1
       00'55'   1B48                    .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
          35    1B4A                    .BYTE   ^05 ! <8*M>
          C2    1B4B                    .BYTE   ^0302
       1B3A'    1B4C                    .WORD   <2143$-HEAD>
          C1    1B4E                    .BYTE   ^0301 ! <8*B>
          70    1B4F                    .BYTE   ^0100 ! <M*8> ! B
                1B50  4739
                1B50  4740                      OUT     SETATN
       AA D3    1B50                    .BYTE   ^0323 , SETATN&255      ;register A
                1B52  4741
                1B52  4742                      DO      3
          21    1B52                    .BYTE   <H*8> ! 1
       00'55'   1B53                    .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
          46    1B55                    .BYTE   ^0100 ! <B*8> ! M
          C5    1B56                    .BYTE   ^0305 ! <8*B>
       03 3E    1B57                    .BYTE   <A*8> ! 6 , 3&255
          77    1B59                    .BYTE   ^0100 ! <M*8> ! A
```

```
                 1B5A          2144$:
                 1B5A  4743
                 1B5A  4744                    CALL    MICRO_STEP_CPU_R
      CD         1B5A                  .BYTE   ^0315
   12A0'         1B5B                  .WORD   <MICRO_STEP_CPU_R-HEAD>
                 1B5D  4745                    CALL    CHECK_UPC
      CD         1B5D                  .BYTE   ^0315
   1B73'         1B5E                  .WORD   <CHECK_UPC-HEAD>
                 1B60  4746
                 1B60  4747                    LDA     FAST_WRITE              ;GET FAST WRITE FLAG
      3A         1B60                  .BYTE   ^072
   02FC'         1B61                  .WORD   <FAST_WRITE-HEAD>
                 1B63  4748                    ORA     A                       ;SET CONDITION CODES
      B7         1B63                  .BYTE   ^0260 ! A
                 1B64  4749                    JZ      LEAVE_DO                ;LEAVE DO LOOP IF FAST_WRITE
      CA         1B64                  .BYTE   ^0312
   1B71'         1B65                  .WORD   <LEAVE_DO-HEAD>
                 1B67  4750                                                    ; CLEAR
                 1B67  4751                    ENDDO
      21         1B67                  .BYTE   <H*8> ! 1
   00'55'        1B68                  .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
      35         1B6A                  .BYTE   ^05 ! <8*M>
      C2         1B6B                  .BYTE   ^0302
   1B5A'         1B6C                  .WORD   <2144$-HEAD>
      C1         1B6E                  .BYTE   ^0301 ! <8*B>
      70         1B6F                  .BYTE   ^0100 ! <M*8> ! B
                 1B70  4752
                 1B70  4753                    RET
      C9         1B70                  .BYTE   ^0311
                 1B71  4754
                 1B71  4755 LEAVE_DO:          POP     B                       ;POP STACK SINCE DO-LOOP DID
      C1         1B71                  .BYTE   ^0301 ! <8*B>
                 1B72  4756                                                    ; A PUSH
                 1B72  4757                    RET                             ;AND RETURN
      C9         1B72                  .BYTE   ^0311
                 1B73  4758
                 1B73  4759 ;*****************************************************************************
                 1B73  4760 ;
                 1B73  4761 ;  CHECK_UPC:
                 1B73  4762 ;  THIS ROUTINE COMPARE THE UPC VALUE FROM THE TABLE AND THE VALUE IN
                 1B73  4763 ;  THE UPC AT THE TIME IT IS CALLED.
                 1B73  4764 ;
                 1B73  4765 ;  INPUTCONDITION:
                 1B73  4766 ;       WRITE_TAB_PNT = POINTS TO THE TABLE VALUE
                 1B73  4767 ;
                 1B73  4768 ;  OUTPUT CONDITION:
                 1B73  4769 ;       WRITE_TAB_PNT IS INCREMENTED TO THE NEXT TABLE VALUE
                 1B73  4770 ;
                 1B73  4771 ;*****************************************************************************
                 1B73  4772
                 1B73  4773 CHECK_UPC:         CALL    LOAD_UPC_R
      CD         1B73                  .BYTE   ^0315
   0F19'         1B74                  .WORD   <LOAD_UPC_R-HEAD>
                 1B76  4774                    LHLD    WRITE_TAB_PNT
      2A         1B76                  .BYTE   ^052
   006F'         1B77                  .WORD   <WRITE_TAB_PNT-HEAD>
                 1B79  4775                    MOV     A,M
```

```
                7E  1B79                    .BYTE   ^0100 ! <A*8> ! M
                    1B7A  4776                      STA     UPC_COMPARE+1
                32  1B7A                    .BYTE   ^062
             00CB'  1B7B                    .WORD   <UPC_COMPARE+1-HEAD>
                    1B7D  4777                      INX     H
                23  1B7D                    .BYTE   <H*8> ! 3
                    1B7E  4778                      MOV     A,M
                7E  1B7E                    .BYTE   ^0100 ! <A*8> ! M
                    1B7F  4779                      STA     UPC_COMPARE
                32  1B7F                    .BYTE   ^062
             00CA'  1B80                    .WORD   <UPC_COMPARE-HEAD>
                    1B82  4780                      INX     H
                23  1B82                    .BYTE   <H*8> ! 3
                    1B83  4781                      SHLD    WRITE_TAB_PNT
                22  1B83                    .BYTE   ^042
             006F'  1B84                    .WORD   <WRITE_TAB_PNT-HEAD>
                    1B86  4782
                    1B86  4783                      IFNEQ   UPC_COMPARE,UPC_VALUE,2
                21  1B86                    .BYTE   <H*8> ! 1
          00'CA'    1B87                    .BYTE   <UPC_COMPARE-HEAD>&255 , <<UPC_COMPARE-HEAD>&^XFF00>/256
                11  1B89                    .BYTE   <D*8> ! 1
          00'C2'    1B8A                    .BYTE   <UPC_VALUE-HEAD>&255 , <<UPC_VALUE-HEAD>&^XFF00>/256
          02 0E     1B8C                    .BYTE   <C*8> ! 6 , 2&255
                CD  1B8E                    .BYTE   ^0315
             00CF'  1B8F                    .WORD   <COMPARE-HEAD>
                CA  1B91                    .BYTE   ^0312
             1B9D'  1B92                    .WORD   <2145$-HEAD>
                    1B94  4784                      MVI     B,<^X29>
          29 06     1B94                    .BYTE   <B*8> ! 6 , ^X29&255
                    1B96  4785                      CALL    ERROR_ROUTINE
                CD  1B96                    .BYTE   ^0315
             0C4A'  1B97                    .WORD   <ERROR_ROUTINE-HEAD>
                    1B99  4786                      CLEAR   FAST_WRITE
                AF  1B99                    .BYTE   ^0250 ! A                       ;with A
                32  1B9A                    .BYTE   ^062
             02FC'  1B9B                    .WORD   <FAST_WRITE-HEAD>
                    1B9D  4787
                    1B9D  4788                      ENDIF
                    1B9D        2145$:                                              ;GENERATE LCL 2145 NAME
                    1B9D  4789
                    1B9D  4790                      CALL    LOAD_UPC_R
                CD  1B9D                    .BYTE   ^0315
             0F19'  1B9E                    .WORD   <LOAD_UPC_R-HEAD>
                    1BA0  4791
                    1BA0  4792                      RET
                C9  1BA0                    .BYTE   ^0311
                    1BA1  4793
                    1BA1  4794 ;*****************************************************************************
                    1BA1  4795 ;
                    1BA1  4796 ;  EXEC_TEST0
                    1BA1  4797 ;  THIS ROUTINE EXECUTES TEST ZERO OF A WCS IMAGE ONLY IF FILE LOADED
                    1BA1  4798 ;  IS SET.  TEST ZERO IS CONTINUED EACH TIME A DATA REQUEST IS RETURNED
                    1BA1  4799 ;  WHEN A DATA REQUEST IS NOT RETURNED THEN EXECUTION OF TEST 0 STOPS
                    1BA1  4800 ;
                    1BA1  4801 ;*****************************************************************************
                    1BA1  4802
                    1BA1  4803 EXEC_TEST0:      MVI     B,<^X2A>                    ;ERROR CODE IF WCS NOT LOADED
```

```
                2A 06  1BA1                  .BYTE   <B*8> ! 6 , ^X2A&255
                       1BA3  4804                    CALL    CHECK_WCS_LOADED        ;CHECK FOR WCS LOADED SET
                 CD    1BA3                  .BYTE   ^O315
               0C2A'   1BA4                  .WORD   <CHECK_WCS_LOADED-HEAD>
                       1BA6  4805                                                    ; RETURN IF OK
                       1BA6  4806
                       1BA6  4807                    IFN     NOINIT                  ;ELSE CHECK NO INIT FLAG
                 3A    1BA6                  .BYTE   ^O72
               03BE'   1BA7                  .WORD   <NOINIT-HEAD>
                 0F    1BA9                  .BYTE   ^O17
                 DA    1BAA                  .BYTE   ^O332
               1BDF'   1BAB                  .WORD   <2146$-HEAD>
                       1BAD  4808
                       1BAD  4809                    CALL    VERIFY_WRITER           ;CHECK DATA XFER ROUTINE
                 CD    1BAD                  .BYTE   ^O315
               1B16'   1BAE                  .WORD   <VERIFY_WRITER-HEAD>
                       1BB0  4810
                       1BB0  4811                    CLEAR   STARTING_UPC
                 AF    1BB0                  .BYTE   ^O250 ! A               ;with A
                 32    1BB1                  .BYTE   ^O62
               0407'   1BB2                  .WORD   <STARTING_UPC-HEAD>
                       1BB4  4812                    CLEAR   STARTING_UPC+1          ;SET TO EXECUTE DATA XFER
                 AF    1BB4                  .BYTE   ^O250 ! A               ;with A
                 32    1BB5                  .BYTE   ^O62
               0408'   1BB6                  .WORD   <STARTING_UPC+1-HEAD>
                       1BB8  4813
                       1BB8  4814                    WHILER  DATA_XFER_FLG           ;FLAG SET IF DATA XFER EVENT
                 AF    1BB8                  .BYTE   ^O250 ! A               ;with A
                 3C    1BB9                  .BYTE   ^O04 ! <8*A>
                 32    1BBA                  .BYTE   ^O62
               025E'   1BBB                  .WORD   <DATA_XFER_FLG-HEAD>
                       1BBD          4004$:  LDA     DATA_XFER_FLG
                 3A    1BBD                  .BYTE   ^O72
               025E'   1BBE                  .WORD   <DATA_XFER_FLG-HEAD>
                 0F    1BC0                  .BYTE   ^O17
                 D2    1BC1                  .BYTE   ^O322
               1BDF'   1BC2                  .WORD   <2147$-HEAD>
                       1BC4  4815                                                    ;LOOP STOPS IF NO EVENT
                       1BC4  4816                    CALL    START_CPU
                 CD    1BC4                  .BYTE   ^O315
               10FF'   1BC5                  .WORD   <START_CPU-HEAD>
                       1BC7  4817                    SET     TEST_ZERO               ;SET FOR NO RESTART
                 AF    1BC7                  .BYTE   ^O250 ! A               ;with A
                 3C    1BC8                  .BYTE   ^O04 ! <8*A>
                 32    1BC9                  .BYTE   ^O62
               0447'   1BCA                  .WORD   <TEST_ZERO-HEAD>
                       1BCC  4818                    CALL    ATTENTION               ;WAIT FOR CPU ATTENTIONS
                 CD    1BCC                  .BYTE   ^O315
               1321'   1BCD                  .WORD   <ATTENTION-HEAD>
                       1BCF  4819                    CLEAR   TEST_ZERO
                 AF    1BCF                  .BYTE   ^O250 ! A               ;with A
                 32    1BD0                  .BYTE   ^O62
               0447'   1BD1                  .WORD   <TEST_ZERO-HEAD>
                       1BD3  4820                    CALL    RESTORE_WR
                 CD    1BD3                  .BYTE   ^O315
               0C0A'   1BD4                  .WORD   <RESTORE_WR-HEAD>
                       1BD6  4821
```

```
                1BD6  4822                    LHLD    UPC_SUB                 ;SET FOR CONTINUE AT NEXT ADDR.
      2A        1BD6                  .BYTE   ^052
  0006'         1BD7                  .WORD   <UPC_SUB-HEAD>
                1BD9  4823                    SHLD    STARTING_UPC
      22        1BD9                  .BYTE   ^042
  0407'         1BDA                  .WORD   <STARTING_UPC-HEAD>
                1BDC  4824
                1BDC  4825                    ENDWHILE
      C3        1BDC                  .BYTE   ^0303
  18BD'         1BDD                  .WORD   <4004$-HEAD>
                1BDF        2147$:                                            ;GENERATE LCL 2147 NAME
                1BDF  4826
                1BDF  4827                    ENDIF
                1BDF        2146$:                                            ;GENERATE LCL 2146 NAME
                1BDF  4828
                1BDF  4829                    RET
      C9        1BDF                  .BYTE   ^0311
                1BE0  4830
                1BE0  4831
                1BE0  4832 ;*****************************************************************************
                1BE0  4833 ;
                1BE0  4834 ;  SYNTAX_ERROR
                1BE0  4835 ;  THIS ROUTINE PRINTS "SYNTAX ERROR"
                1BE0  4836 ;
                1BE0  4837 ;*****************************************************************************
                1BE0  4838
                1BE0  4839 SYNTAX_ERROR:
                1BE0  4840                    CLEAR   NOT_FOUND
      AF        1BE0                  .BYTE   ^0250 ! A                       ;with A
      32        1BE1                  .BYTE   ^062
  03C2'         1BE2                  .WORD   <NOT_FOUND-HEAD>
                1BE4  4841                    CLEAR   EXECUTE                         ;PARSER WILL PROMPT FOR COMMAND
      AF        1BE4                  .BYTE   ^0250 ! A                       ;with A
      32        1BE5                  .BYTE   ^062
  02FA'         1BE6                  .WORD   <EXECUTE-HEAD>
                1BE8  4842                    RET
      C9        1BE8                  .BYTE   ^0311
                1BE9  4843
                1BE9  4844
                1BE9  4845 ;*****************************************************************************
                1BE9  4846 ;
                1BE9  4847 ;  CHECKSUM
                1BE9  4848 ;  THIS ROUTINE CHECKSUMS THE PROGRAM AREA  (NOT THE VARIABLE AREA)
                1BE9  4849 ;  THE CHECKSUM IS LEFT IN NEWSUM
                1BE9  4850 ;
                1BE9  4851 ;*****************************************************************************
                1BE9  4852
                1BE9  4853 CHECKSUM:          LXI     H,BEGIN_CHECKSUM
      21        1BE9                  .BYTE   <H*8> ! 1
  09'8A'        1BEA                  .BYTE   <BEGIN_CHECKSUM-HEAD>&255 , <<BEGIN_CHECKSUM-HEAD>&^XFF00>/256
                1BEC  4854                    LXI     D,END_CHECKSUM
      11        1BEC                  .BYTE   <D*8> ! 1
  1F'FD'        1BED                  .BYTE   <END_CHECKSUM-HEAD>&255 , <<END_CHECKSUM-HEAD>&^XFF00>/256
                1BEF  4855                    CLEAR   NEWSUM
      AF        1BEF                  .BYTE   ^0250 ! A                       ;with A
      32        1BF0                  .BYTE   ^062
  03BC'         1BF1                  .WORD   <NEWSUM-HEAD>
```

```
                 1BF3  4856 1$:              LDA     NEWSUM
      3A         1BF3                .BYTE   ^072
   03BC'         1BF4                .WORD   <NEWSUM-HEAD>
                 1BF6  4857                  ADD     M
      86         1BF6                .BYTE   ^0200 ! M
                 1BF7  4858                  INX     H
      23         1BF7                .BYTE   <H*8> ! 3
                 1BF8  4859                  STA     NEWSUM
      32         1BF8                .BYTE   ^062
   03BC'         1BF9                .WORD   <NEWSUM-HEAD>
                 1BFB  4860                  MOV     A,H
      7C         1BFB                .BYTE   ^0100 ! <A*8> ! H
                 1BFC  4861                  CMP     D
      BA         1BFC                .BYTE   ^0270 ! D
                 1BFD  4862                  JNZ     1$
      C2         1BFD                .BYTE   ^0302
   1BF3'         1BFE                .WORD   <1$-HEAD>
                 1C00  4863                  MOV     A,L
      7D         1C00                .BYTE   ^0100 ! <A*8> ! L
                 1C01  4864                  CMP     E
      BB         1C01                .BYTE   ^0270 ! E
                 1C02  4865                  JNZ     1$
      C2         1C02                .BYTE   ^0302
   1BF3'         1C03                .WORD   <1$-HEAD>
                 1C05  4866                  RET
      C9         1C05                .BYTE   ^0311
                 1C06  4867
                 1C06  4868
                 1C06  4869 ;*****************************************************************************
                 1C06  4870 ;
                 1C06  4871 ;  SET_PARDONE
                 1C06  4872 ;  THIS ROUTINE SETS THE FLAG PARDONE SO THAT RE-ENTRANT PARSING
                 1C06  4873 ;  ROUTINES(SUCH AS DIAGNOSE) KNOW THEY HAVE HAVE END OF LINE.
                 1C06  4874 ;
                 1C06  4875 ;*****************************************************************************
                 1C06  4876
                 1C06  4877 SET_PARDONE:     SET     PARDONE
      AF         1C06                .BYTE   ^0250 ! A               ;with A
      3C         1C07                .BYTE   ^04 ! <8*A>
      32         1C08                .BYTE   ^062
   03D9'         1C09                .WORD   <PARDONE-HEAD>
                 1C0B  4878                  RET
      C9         1C0B                .BYTE   ^0311
                 1C0C  4879
                 1C0C  4880
                 1C0C  4881 ;*****************************************************************************
                 1C0C  4882 ;
                 1C0C  4883 ;  POWER_REC
                 1C0C  4884 ;  THIS ROUTINE IS BRANCHED TO ON A POWER RECOVERY
                 1C0C  4885 ;  IT WILL CALL THE ROM CODE THAT DOES POWER UP SEQUENCING AND WILL
                 1C0C  4886 ;  DO AN INITIALIZE IF THERE IS WCS CODE LOADED
                 1C0C  4887 ;
                 1C0C  4888 ;*****************************************************************************
                 1C0C  4889
                 1C0C  4890 POWER_REC:       CALL    POWER_SEQ               ;CALL ROM POWER SEQ ROUTINE
      CD         1C0C                .BYTE   ^0315
   001B          1C0D                .WORD   <POWER_SEQ-HEAD>
```

```
                        1C0F  4891
                        1C0F  4892                                      ;THE FOLLOWING DELAY LOOP
                        1C0F  4893                                      ; IS TO ALLOW AC LOW TO
                        1C0F  4894                                      ; SETTLE BEFORE RE-ENABLING
                        1C0F  4895                                      ; INTERRUPTS
                        1C0F  4896              LXI     H,0             ;INIT COUNTER IN H&L TO 0
     21                 1C0F                    .BYTE   <H*8> ! 1
  00 00                 1C10                    .BYTE   0&255 , <0&^XFF00>/256
                        1C12  4897 1$:          DCX     H               ;DECREMENT COUNTER
     2B                 1C12                    .BYTE   <H*8> ! 11
                        1C13  4898              MOV     A,H             ;GET HIGH BYTE RESULT
     7C                 1C13                    .BYTE   ^0100 ! <A*8> ! H
                        1C14  4899              ORA     L               ;'OR' WITH LOW BYTE
     B5                 1C14                    .BYTE   ^0260 ! L
                        1C15  4900              JNZ     1$              ;REPEAT IF NOT 0
     C2                 1C15                    .BYTE   ^0302
   1C12'                1C16                    .WORD   <1$-HEAD>
                        1C18  4901
                        1C18  4902              MVI     A,HEX_10        ;A GETS A 10(H)
  10 3E                 1C18                    .BYTE   <A*8> ! 6 , HEX_10&255
                        1C1A  4903              SIM                     ;RESET 7.5
     30                 1C1A                    .BYTE   ^060
                        1C1B  4904
                        1C1B  4905              SET     WARM_RESTART    ;INDICATE POWER FAIL RESTART
     AF                 1C1B                    .BYTE   ^0250 ! A               ;with A
     3C                 1C1C                    .BYTE   ^04 ! <8*A>
     32                 1C1D                    .BYTE   ^062
  04B3'                 1C1E                    .WORD   <WARM_RESTART-HEAD>
                        1C20  4906              JMP     START_UP
     C3                 1C20                    .BYTE   ^0303
  1CD2'                 1C21                    .WORD   <START_UP-HEAD>
                        1C23  4907
                        1C23  4908 ;*****************************************************************************
                        1C23  4909 ;
                        1C23  4910 ;  TIMER_VEC
                        1C23  4911 ;
                        1C23  4912 ;       THIS ROUTINE WILL HANDLE A INTERVAL TIMER INTERRUPT IF THE INTERRUPT
                        1C23  4913 ;       IS NOT MASKED OUT.  AT PRESENT, THIS ROUTINE WILL NOT BE ENTERED AND
                        1C23  4914 ;       DOES NOTHING.  THIS ROUTINE WILL BE WRITTEN AT A LATER TIME IF NECES-
                        1C23  4915 ;       SARY.
                        1C23  4916 ;
                        1C23  4917 ;*****************************************************************************
                        1C23  4918
                        1C23  4919 TIMER_VEC:   RET                     ;DO A RETURN
     C9                 1C23                    .BYTE   ^0311
                        1C24  4920
                        1C24  4921 ;*****************************************************************************
                        1C24  4922 ;
                        1C24  4923 ;  AUTO_TEST
                        1C24  4924 ;
                        1C24  4925 ;       THIS ROUTINE WILL HANDLE INSTUCTION FLOW WHILE RUNNING MICMON UNDER
                        1C24  4926 ;       CUSTOMER RUNABLE DIAGNOSTICS AUTO MODE
                        1C24  4927 ;
                        1C24  4928 ;*****************************************************************************
                        1C24  4929
                        1C24  4930 AUTO_TEST:   CLEAR   ERROR
     AF                 1C24                    .BYTE   ^0250 ! A               ;with A
```

```
                                                .BYTE   ^062
02E2' 1C26                                      .WORD   <ERROR-HEAD>
      1C28  4931
      1C28  4932                                IF      PACK_ERROR                ;IF WRONG RL02 PACK WAS FOUND
   3A 1C28                                      .BYTE   ^072
03D5' 1C29                                      .WORD   <PACK_ERROR-HEAD>
   0F 1C2B                                      .BYTE   ^017
   D2 1C2C                                      .BYTE   ^0322
1C47' 1C2D                                      .WORD   <2148$-HEAD>
      1C2F  4933                                LXI     H,RETRY_IDC2              ; THEN RESTART TEST
   21 1C2F                                      .BYTE   <H*8> ! 1
03'F2' 1C30                                     .BYTE   <RETRY_IDC2-HEAD>&255 , <<RETRY_IDC2-HEAD>&^XFF00>/256
      1C32  4934                                LXI     D,TTBUF                   ;MOVE INSTRUCTION INTO TTBUF
   11 1C32                                      .BYTE   <D*8> ! 1
08'D1' 1C33                                     .BYTE   <TTBUF-HEAD>&255 , <<TTBUF-HEAD>&^XFF00>/256
      1C35  4935                                MVI     C,12
0C 0E 1C35                                      .BYTE   <C*8> ! 6 , 12&255
      1C37  4936                                CALL    MOVER_R
   CD 1C37                                      .BYTE   ^0315
0F75' 1C38                                      .WORD   <MOVER_R-HEAD>
      1C3A  4937
      1C3A  4938                                LXI     H,TTBUF                   ;INIT TTY BUFFER POINTER
   21 1C3A                                      .BYTE   <H*8> ! 1
08'D1' 1C3B                                     .BYTE   <TTBUF-HEAD>&255 , <<TTBUF-HEAD>&^XFF00>/256
      1C3D  4939                                SHLD    TTBUF_PNT
   22 1C3D                                      .BYTE   ^042
044D' 1C3E                                      .WORD   <TTBUF_PNT-HEAD>
      1C40  4940                                CLEAR   PACK_ERROR
   AF 1C40                                      .BYTE   ^0250 ! A                 ;with A
   32 1C41                                      .BYTE   ^062
03D5' 1C42                                      .WORD   <PACK_ERROR-HEAD>
      1C44  4941
      1C44  4942                                ELSE
   C3 1C44                                      .BYTE   ^0303
1CCA' 1C45                                      .WORD   <2149$-HEAD>
      1C47        2148$:                                                          ;GENERATE LOCAL SYMBOL NAME
      1C47  4943                                LHLD    INSTR_PNT                 ;MOVE FIRST BYTE OF INSTRUCTION
   2A 1C47                                      .BYTE   ^052
0345' 1C48                                      .WORD   <INSTR_PNT-HEAD>
      1C4A  4944                                MOV     A,M                       ; TO A
   7E 1C4A                                      .BYTE   ^0100 ! <A*8> ! M
      1C4B  4945                                RAR                               ;ROTATE FIRST BIT INTO CARRY
   1F 1C4B                                      .BYTE   ^037
      1C4C  4946
      1C4C  4947                                IFC                               ;BIT 0 SET TELLS TO PRINT
   D2 1C4C                                      .BYTE   ^0322
1C57' 1C4D                                      .WORD   <2150$-HEAD>
      1C4F  4948                                SET     PRINT_START               ;'TESTING STARTED' MESSAGE
   AF 1C4F                                      .BYTE   ^0250 ! A                 ;with A
   3C 1C50                                      .BYTE   ^04 ! <8*A>
   32 1C51                                      .BYTE   ^062
03F0' 1C52                                      .WORD   <PRINT_START-HEAD>
      1C54  4949                                ELSE
   C3 1C54                                      .BYTE   ^0303
1C5B' 1C55                                      .WORD   <2151$-HEAD>
      1C57        2150$:                                                          ;GENERATE LOCAL SYMBOL NAME
      1C57  4950                                CLEAR   PRINT_START               ;BIT 0 CLEAR TELLS TO NOT PRINT
```

```
                                                .BYTE   ^0250 ! A               ;with A
                                                .BYTE   ^062
                                                .WORD   <PRINT_START-HEAD>
                                                ENDIF
                                        2151$:                                  ;GENERATE LCL 2151 NAME
                                                LHLD    INSTR_PNT               ;LOOK AT SECOND BIT
                                                .BYTE   ^052
                                                .WORD   <INSTR_PNT-HEAD>
                                                MOV     A,M
                                                .BYTE   ^0100 ! <A*8> ! M
                                                RAR
                                                .BYTE   ^037
                                                RAR
                                                .BYTE   ^037
                                                IFC                             ;BIT 1 SET TELLS TO PRINT
                                                .BYTE   ^0322
                                                .WORD   <2152$-HEAD>
                                                SET     PRINT_PASS              ; 'PASSED' MESSAGE
                                                .BYTE   ^0250 ! A               ;with A
                                                .BYTE   ^04 ! <8*A>
                                                .BYTE   ^062
                                                .WORD   <PRINT_PASS-HEAD>
                                                ELSE
                                                .BYTE   ^0303
                                                .WORD   <2153$-HEAD>
                                        2152$:                                  ;GENERATE LOCAL SYMBOL NAME
                                                CLEAR   PRINT_PASS              ;BIT 1 CLEAR TELLS TO NOT PRINT
                                                .BYTE   ^0250 ! A               ;with A
                                                .BYTE   ^062
                                                .WORD   <PRINT_PASS-HEAD>
                                                ENDIF
                                        2153$:                                  ;GENERATE LCL 2153 NAME
                                                IF      PRINT_START             ;PRINT START OF TEST IF SET
                                                .BYTE   ^072
                                                .WORD   <PRINT_START-HEAD>
                                                .BYTE   ^017
                                                .BYTE   ^0322
                                                .WORD   <2154$-HEAD>
                                                LHLD    NAME_PNT                ;PRINT TEST NAME
                                                .BYTE   ^052
                                                .WORD   <NAME_PNT-HEAD>
                                                CALL    PRINT_STRING_R
                                                .BYTE   ^0315
                                                .WORD   <PRINT_STRING_R-HEAD>
                                                LXI     H,TEST_START_A          ;PRINT 'TESTING STARTED'
                                                .BYTE   <H*8> ! 1
                                                .BYTE   <TEST_START_A-HEAD>&255 , <<TEST_START_A-HEAD>&^XFF00>/256
                                                CALL    PRINT_STRING_R
                                                .BYTE   ^0315
                                                .WORD   <PRINT_STRING_R-HEAD>
                                                LHLD    TIME_PNT                ;PRINT RUN TIME
```

```
                2A  1C83                .BYTE   ^052
             0448'  1C84                .WORD   <TIME_PNT-HEAD>
                    1C86  4973                  CALL    PRINT_STRING_R
                CD  1C86                .BYTE   ^0315
             100E'  1C87                .WORD   <PRINT_STRING_R-HEAD>
                    1C89  4974
                    1C89  4975                  LXI     H,MINUTES_A             ;PRINT 'MINUTES' MESSAGE
                21  1C89                .BYTE   <H*8> ! 1
           03'8C'   1C8A                .BYTE   <MINUTES_A-HEAD>&255 , <<MINUTES_A-HEAD>&^XFF00>/256
                    1C8C  4976                  CALL    PRINT_STRING_R
                CD  1C8C                .BYTE   ^0315
             100E'  1C8D                .WORD   <PRINT_STRING_R-HEAD>
                    1C8F  4977
                    1C8F  4978                  MVI     B,0                     ;POINT TO NEXT RUN TIME
             00 06  1C8F                .BYTE   <B*8> ! 6 , 0&255
                    1C91  4979                  MVI     C,5
             05 0E  1C91                .BYTE   <C*8> ! 6 , 5&255
                    1C93  4980                  LHLD    TIME_PNT
                2A  1C93                .BYTE   ^052
             0448'  1C94                .WORD   <TIME_PNT-HEAD>
                    1C96  4981                  DAD     B
                09  1C96                .BYTE   <B*8> ! 9
                    1C97  4982                  SHLD    TIME_PNT
                22  1C97                .BYTE   ^042
             0448'  1C98                .WORD   <TIME_PNT-HEAD>
                    1C9A  4983
                    1C9A  4984                  MVI     B,0                     ;POINT TO NEXT TEST NAME
             00 06  1C9A                .BYTE   <B*8> ! 6 , 0&255
                    1C9C  4985                  MVI     C,21
             15 0E  1C9C                .BYTE   <C*8> ! 6 , 21&255
                    1C9E  4986                  LHLD    NAME_PNT
                2A  1C9E                .BYTE   ^052
             03BA'  1C9F                .WORD   <NAME_PNT-HEAD>
                    1CA1  4987                  DAD     B
                09  1CA1                .BYTE   <B*8> ! 9
                    1CA2  4988                  SHLD    NAME_PNT
                22  1CA2                .BYTE   ^042
             03BA'  1CA3                .WORD   <NAME_PNT-HEAD>
                    1CA5  4989
                    1CA5  4990                  ENDIF
                    1CA5        2154$:                                      ;GENERATE LCL 2154 NAME
                    1CA5  4991
                    1CA5  4992                  LHLD    INSTR_PNT           ;POINT TO LENGTH OF INSTRUCTION
                2A  1CA5                .BYTE   ^052
             0345'  1CA6                .WORD   <INSTR_PNT-HEAD>
                    1CA8  4993                  INX     H
                23  1CA8                .BYTE   <H*8> ! 3
                    1CA9  4994                  MOV     A,M                     ;STORE COUNT IN INSTR_LEN
                7E  1CA9                .BYTE   ^0100 ! <A*8> ! M
                    1CAA  4995                  STA     INSTR_LEN
                32  1CAA                .BYTE   ^062
             0344'  1CAB                .WORD   <INSTR_LEN-HEAD>
                    1CAD  4996
                    1CAD  4997                  MOV     C,A                     ;STORE COUNT IN C
                4F  1CAD                .BYTE   ^0100 ! <C*8> ! A
                    1CAE  4998                  INX     H                       ;HL POINTS TO START OF INSTRUCTION
                23  1CAE                .BYTE   <H*8> ! 3
```

```
                 1CAF  4999                     LXI     D,TTBUF                 ;DE POINTS TO TTBUF
      11         1CAF                   .BYTE   <D*8> ! 1
08'D1'           1CB0                   .BYTE   <TTBUF-HEAD>&255 , <<TTBUF-HEAD>&^XFF00>/256
                 1CB2  5000                     CALL    MOVER_R                 ;MOVE INSTRUCTION INTO TTBUF
      CD         1CB2                   .BYTE   ^O315
 0F75'           1CB3                   .WORD   <MOVER_R-HEAD>
                 1CB5  5001
                 1CB5  5002                     LXI     H,TTBUF                 ;INIT TTY BUFFER POINTER
      21         1CB5                   .BYTE   <H*8> ! 1
08'D1'           1CB6                   .BYTE   <TTBUF-HEAD>&255 , <<TTBUF-HEAD>&^XFF00>/256
                 1CB8  5003                     SHLD    TTBUF_PNT
      22         1CB8                   .BYTE   ^O42
 044D'           1CB9                   .WORD   <TTBUF_PNT-HEAD>
                 1CBB  5004
                 1CBB  5005                     MVI     B,0                     ;POINT TO NEXT INSTRUCTION
   00 06         1CBB                   .BYTE   <B*8> ! 6 , 0&255
                 1CBD  5006                     LDA     INSTR_LEN
      3A         1CBD                   .BYTE   ^O72
 0344'           1CBE                   .WORD   <INSTR_LEN-HEAD>
                 1CC0  5007                     ADI     2
      C6         1CC0                   .BYTE   ^O306
      02         1CC1                   .BYTE   2
                 1CC2  5008                     MOV     C,A
      4F         1CC2                   .BYTE   ^O100 ! <C*8> ! A
                 1CC3  5009                     LHLD    INSTR_PNT
      2A         1CC3                   .BYTE   ^O52
 0345'           1CC4                   .WORD   <INSTR_PNT-HEAD>
                 1CC6  5010                     DAD     B
      09         1CC6                   .BYTE   <B*8> ! 9
                 1CC7  5011                     SHLD    INSTR_PNT
      22         1CC7                   .BYTE   ^O42
 0345'           1CC8                   .WORD   <INSTR_PNT-HEAD>
                 1CCA  5012
                 1CCA  5013                     ENDIF
                 1CCA            2149$:                                          ;GENERATE LCL 2149 NAME
                 1CCA  5014
                 1CCA  5015                     SET     RESTART
      AF         1CCA                   .BYTE   ^O250 ! A               ;with A
      3C         1CCB                   .BYTE   ^O04 ! <8*A>
      32         1CCC                   .BYTE   ^O062
 03F1'           1CCD                   .WORD   <RESTART-HEAD>
                 1CCF  5016                     JMP     PARSER
      C3         1CCF                   .BYTE   ^O303
 1DAE'           1CD0                   .WORD   <PARSER-HEAD>
                 1CD2  5017
```

```
                        1CD2  5019 ;*****************************************************************************
                        1CD2  5020 ;
                        1CD2  5021 ;  MAIN LINE ROUTINES
                        1CD2  5022 ;
                        1CD2  5023 ;*****************************************************************************
                        1CD2  5024
                        1CD2  5025 ;*****************************************************************************
                        1CD2  5026 ;
                        1CD2  5027 ; START_UP
                        1CD2  5028 ; THINGS THAT ARE ONLY DONE AT THE BEGINING
                        1CD2  5029 ;
                        1CD2  5030 ;*****************************************************************************
                        1CD2  5031
                        1CD2  5032 START_UP:        LXI     $SP,PRI_STACK
                  31    1CD2                .BYTE   <$SP*8> ! 1
                40 80   1CD3                .BYTE   <PRI_STACK-HEAD>&255 , <<PRI_STACK-HEAD>&^XFF00>/256
                        1CD5  5033
                        1CD5  5034                  DI                                  ;DISABLE INTERUPTS
                  F3    1CD5                .BYTE   ^0363
                        1CD6  5035
                        1CD6  5036                  CALL    STOP_CPU                    ;STOP CPU IN CASE IT IS RUNNING
                  CD    1CD6                .BYTE   ^0315
                10C9'   1CD7                .WORD   <STOP_CPU-HEAD>
                        1CD9  5037                  CLEAR   MICRO_STEP                  ;CLEAR MICRO STEP MODE
                  AF    1CD9                .BYTE   ^0250 ! A               ;with A
                  32    1CDA                .BYTE   ^062
                038A'   1CDB                .WORD   <MICRO_STEP-HEAD>
                        1CDD  5038                  SET     MEM_REQ                     ;ENABLE MEM REQ FOR LATER
                  AF    1CDD                .BYTE   ^0250 ! A               ;with A
                  3C    1CDE                .BYTE   ^04 ! <8*A>
                  32    1CDF                .BYTE   ^062
                0354'   1CE0                .WORD   <MEM_REQ-HEAD>
                        1CE2  5039
                        1CE2  5040                  CALL    CHECKSUM
                  CD    1CE2                .BYTE   ^0315
                18E9'   1CE3                .WORD   <CHECKSUM-HEAD>
                        1CE5  5041                  LDA     NEWSUM                      ;SAVE CHECKSUM VALUE
                  3A    1CE5                .BYTE   ^072
                03BC'   1CE6                .WORD   <NEWSUM-HEAD>
                        1CE8  5042                  STA     OLDSUM
                  32    1CE8                .BYTE   ^062
                03CF'   1CE9                .WORD   <OLDSUM-HEAD>
                        1CEB  5043
                        1CEB  5044                  LXI     H,CRD_FLAGS                 ;CLEAR BIT 1 OF CRD_FLAGS TO
                  21    1CEB                .BYTE   <H*8> ! 1
                40 FD   1CEC                .BYTE   <CRD_FLAGS-HEAD>&255 , <<CRD_FLAGS-HEAD>&^XFF00>/256
                        1CEE  5045                  MVI     A,HEX_FD                    ; INDICATE CRD HAS STARTED
                FD 3E   1CEE                .BYTE   <A*8> ! 6 , HEX_FD&255
                        1CF0  5046                  ANA     M
                  A6    1CF0                .BYTE   ^0240 ! M
                        1CF1  5047                  MOV     M,A
                  77    1CF1                .BYTE   ^0100 ! <M*8> ! A
                        1CF2  5048
                        1CF2  5049                  CLEAR   OFLAG                       ;ENABLE PRINTOUTS
                  AF    1CF2                .BYTE   ^0250 ! A               ;with A
                  32    1CF3                .BYTE   ^062
                4088    1CF4                .WORD   <OFLAG-HEAD>
```

```
                        1CF6  5050
                        1CF6  5051                      LXI     H,VERSION_1             ;PRINT VERSION NUMBER
              21        1CF6                    .BYTE   <H*8> ! 1
      06'31'            1CF7                    .BYTE   <VERSION_1-HEAD>&255 , <<VERSION_1-HEAD>&^XFF00>/256
                        1CF9  5052                      CALL    PRINT_STRING_R
              CD        1CF9                    .BYTE   ^0315
       100E'            1CFA                    .WORD   <PRINT_STRING_R-HEAD>
                        1CFC  5053                      LXI     H,VERSION               ;PRINT VERSION NUMBER
              21        1CFC                    .BYTE   <H*8> ! 1
      04'4F'            1CFD                    .BYTE   <VERSION-HEAD>&255 , <<VERSION-HEAD>&^XFF00>/256
                        1CFF  5054                      CALL    PRINT_STRING_R
              CD        1CFF                    .BYTE   ^0315
       100E'            1D00                    .WORD   <PRINT_STRING_R-HEAD>
                        1D02  5055                      CALL    CRLF_R
              CD        1D02                    .BYTE   ^0315
       101A'            1D03                    .WORD   <CRLF_R-HEAD>
                        1D05  5056
                        1D05  5057                      SET     PARITY_ON               ;INIT PARITY CALC
              AF        1D05                    .BYTE   ^0250 ! A               ;with A
              3C        1D06                    .BYTE   ^04 ! <8*A>
              32        1D07                    .BYTE   ^062
        0064'           1D08                    .WORD   <PARITY_ON-HEAD>
                        1D0A  5058                      CLEAR   CONTINUE                ;INIT CONTINUE FLAG
              AF        1D0A                    .BYTE   ^0250 ! A               ;with A
              32        1D0B                    .BYTE   ^062
        003A'           1D0C                    .WORD   <CONTINUE-HEAD>
                        1D0E  5059                      CLEAR   ERROR                   ;INIT ERROR FLAG
              AF        1D0E                    .BYTE   ^0250 ! A               ;with A
              32        1D0F                    .BYTE   ^062
        02E2'           1D10                    .WORD   <ERROR-HEAD>
                        1D12  5060
                        1D12  5061                      CLEAR   EXECUTE
              AF        1D12                    .BYTE   ^0250 ! A               ;with A
              32        1D13                    .BYTE   ^062
        02EA'           1D14                    .WORD   <EXECUTE-HEAD>
                        1D16  5062                      CLEAR   WCS_BAD_PARITY          ;SET FOR GOOD PARITY
              AF        1D16                    .BYTE   ^0250 ! A               ;with A
              32        1D17                    .BYTE   ^062
        006E'           1D18                    .WORD   <WCS_BAD_PARITY-HEAD>
                        1D1A  5063
                        1D1A  5064                      LXI     H,X_CLR_PAGE            ;CLR WCS HIGH PAGE
              21        1D1A                    .BYTE   <H*8> ! 1
      00'7D'            1D18                    .BYTE   <X_CLR_PAGE-HEAD>&255 , <<X_CLR_PAGE-HEAD>&^XFF00>/256
                        1D1D  5065                      CALL    PERFORM_CSR_R
              CD        1D1D                    .BYTE   ^0315
       12F2'            1D1E                    .WORD   <PERFORM_CSR_R-HEAD>
                        1D20  5066
                        1D20  5067                      ZERO    DATA0,4                 ;RESET OUT OF COMPAT. MODE
              21        1D20                    .BYTE   <H*8> ! 1
      00'A9'            1D21                    .BYTE   <DATA0-HEAD>&255 , <<DATA0-HEAD>&^XFF00>/256
      04 0E             1D23                    .BYTE   <C*8> ! 6 , 4&255
              AF        1D25                    .BYTE   ^0250 ! A               ;with A
                        1D26            2155$:          MOV     M,A
              77        1D26                    .BYTE   ^0100 ! <M*8> ! A
              23        1D27                    .BYTE   <H*8> ! 3
              0D        1D28                    .BYTE   ^05 ! <8*C>
              C2        1D29                    .BYTE   ^0302
```

```
                1D26' 1D2A                .WORD   <2155$-HEAD>
                      1D2C  5068                  MVI     A,HEX_FF
              FF 3E   1D2C                .BYTE   <A*8> ! 6 , HEX_FF&255
                      1D2E  5069                  CALL    WRITE_LS_R
                CD    1D2E                .BYTE   ^0315
              1146'   1D2F                .WORD   <WRITE_LS_R-HEAD>
                      1D31  5070
                      1D31  5071                  SET     PAR_ON                  ;ENABLE WCS PARITY FOR LATER
                AF    1D31                .BYTE   ^0250 ! A               ;with A
                3C    1D32                .BYTE   ^04 ! <8*A>
                32    1D33                .BYTE   ^062
              03D6'   1D34                .WORD   <PAR_ON-HEAD>
                      1D36  5072
                      1D36  5073                  RIM                             ;SET INTERRUPT MASK
                20    1D36                .BYTE   ^040
                      1D37  5074                  ORI     HEX_8
                F6    1D37                .BYTE   ^0366
                08    1D38                .BYTE   HEX_8
                      1D39  5075                  ORI     HEX_1                   ;DISABLE 5.5 INTERRUPT
                F6    1D39                .BYTE   ^0366
                01    1D3A                .BYTE   HEX_1
                      1D3B  5076                  ANI     HEX_0D
                E6    1D3B                .BYTE   ^0346
                0D    1D3C                .BYTE   HEX_0D
                      1D3D  5077                  SIM
                30    1D3D                .BYTE   ^060
                      1D3E  5078
                      1D3E  5079                  EI                              ;TURN ON INTERRUPTS
                FB    1D3E                .BYTE   ^0373
                      1D3F  5080
                      1D3F  5081                  IF      WARM_RESTART            ;IF POWER FAIL RESTART
                3A    1D3F                .BYTE   ^072
              04B3'   1D40                .WORD   <WARM_RESTART-HEAD>
                0F    1D42                .BYTE   ^017
                D2    1D43                .BYTE   ^0322
              1D53'   1D44                .WORD   <2156$-HEAD>
                      1D46  5082                  CALL    WRITE_LS_STAT           ;WRITE CURRENT FLAGS IN LS
                CD    1D46                .BYTE   ^0315
              12DF'   1D47                .WORD   <WRITE_LS_STAT-HEAD>
                      1D49  5083                                              ; ERROR CONTROL WORD
                      1D49  5084                  LDA     WCS_LOADED          ;GET WCS LOADED FLAG
                3A    1D49                .BYTE   ^072
              058B'   1D4A                .WORD   <WCS_LOADED-HEAD>
                      1D4C  5085                  ORA     A                       ;SET CONDITION CODES
                B7    1D4C                .BYTE   ^0260 ! A
                      1D4D  5086                  CNZ     EXEC_TEST0              ;DO WCS INIT IF WCS IS LOADED
                C4    1D4D                .BYTE   ^0304
              1BA1'   1D4E                .WORD   <EXEC_TEST0-HEAD>
                      1D50  5087
                      1D50  5088                  ELSE
                C3    1D50                .BYTE   ^0303
              1D56'   1D51                .WORD   <2157$-HEAD>
                      1D53        2156$:                                      ;GENERATE LOCAL SYMBOL NAME
                      1D53  5089
                      1D53  5090                  CALL    CLEAR_DFFAULT           ;SET FLAGS TO DEFAULT
                CD    1D53                .BYTE   ^0315
              0B8F'   1D54                .WORD   <CLEAR_DEFAULT-HEAD>
```

```
                        1D56  5091                    ENDIF
                        1D56        2157$:                                      ;GENERATE LCL 2157 NAME
                        1D56  5092
                        1D56  5093                    LXI     H,INSTR_TABLE       ;POINT TO FIRST INSTRUCTION
                  21    1D56                .BYTE     <H*8> ! 1
              06'7A'    1D57                .BYTE     <INSTR_TABLE-HEAD>&255 , <<INSTR_TABLE-HEAD>&^XFF00>/256
                        1D59  5094                    SHLD    INSTR_PNT
                  22    1D59                .BYTE     ^042
                0345'   1D5A                .WORD     <INSTR_PNT-HEAD>
                        1D5C  5095
                        1D5C  5096                    LXI     H,NAME_TABLE        ;POINT TO FIRST TEST NAME
                  21    1D5C                .BYTE     <H*8> ! 1
              07'6F'    1D5D                .BYTE     <NAME_TABLE-HEAD>&255 , <<NAME_TABLE-HEAD>&^XFF00>/256
                        1D5F  5097                    SHLD    NAME_PNT
                  22    1D5F                .BYTE     ^042
                03BA'   1D60                .WORD     <NAME_PNT-HEAD>
                        1D62  5098
                        1D62  5099                    LXI     H,TIME_TABLE        ;POINT TO FIRST TEST TIME
                  21    1D62                .BYTE     <H*8> ! 1
              08'02'    1D63                .BYTE     <TIME_TABLE-HEAD>&255 , <<TIME_TABLE-HEAD>&^XFF00>/256
                        1D65  5100                    SHLD    TIME_PNT
                  22    1D65                .BYTE     ^042
                0448'   1D66                .WORD     <TIME_PNT-HEAD>
                        1D68  5101
                        1D68  5102                    JMP     PARSER              ;GO
                  C3    1D68                .BYTE     ^0303
                1DAE'   1D69                .WORD     <PARSER-HEAD>
                        1D6B  5103
                        1D6B  5104
                        1D6B  5105
                        1D6B  5106 ;*****************************************************************************
                        1D6B  5107 ;
                        1D6B  5108 ;  PAR
                        1D6B  5109 ;  THIS ROUTINE PARSES COMMANDS AND DISPATCHES TO THE ADDRESS GIVEN
                        1D6B  5110 ;
                        1D6B  5111 ;  INPUT CONDITONS
                        1D6B  5112 ;       H+L = PARSE NAME TABLE
                        1D6B  5113 ;             EACH ENTRY = LEN,ASCII,JMP ADDRESS
                        1D6B  5114 ;             LAST TABLE ENTRY HAS LENTH FF
                        1D6B  5115 ;       D+E = BUFFER TO COMPARE ASCII WITH
                        1D6B  5116 ;
                        1D6B  5117 ;*****************************************************************************
                        1D6B  5118
                        1D6B  5119 PAR_TEMP_BUF:    LXI     D,TEMP_BUF          ;POINTER TO CHAR BUFFER
                  11    1D6B                .BYTE     <D*8> ! 1
              04'0E'    1D6C                .BYTE     <TEMP_BUF-HEAD>&255 , <<TEMP_BUF-HEAD>&^XFF00>/256
                        1D6E  5120
                        1D6E  5121 PAR:             SHLD    PARSE_TABLE
                  22    1D6E                .BYTE     ^042
                03DC'   1D6F                .WORD     <PARSE_TABLE-HEAD>
                        1D71  5122                    XCHG
                  EB    1D71                .BYTE     ^0353
                        1D72  5123                    SHLD    PARSE_PNT           ;SAVE BUFFER TO COMPARE WITH
                  22    1D72                .BYTE     ^042
                03DA'   1D73                .WORD     <PARSE_PNT-HEAD>
                        1D75  5124                    CLEAR   NOT_FOUND           ;RESET NOT_FOUND FLAG
                  AF    1D75                .BYTE     ^0250 ! A           ;with A
```

```
                    32  1D76                .BYTE   ^062
                03C2'   1D77                .WORD   <NOT_FOUND-HEAD>
                        1D79  5125
                        1D79  5126 2$:              LHLD    PARSE_PNT
                    2A  1D79                .BYTE   ^052
                03DA'   1D7A                .WORD   <PARSE_PNT-HEAD>
                        1D7C  5127                  XCHG                        ;ONE POINTER IN D+E
                    EB  1D7C                .BYTE   ^0353
                        1D7D  5128                  LHLD    PARSE_TABLE         ;OTHER IN H+L
                    2A  1D7D                .BYTE   ^052
                03DC'   1D7E                .WORD   <PARSE_TABLE-HEAD>
                        1D80  5129                  MOV     C,M                 ;GET COUNTER
                    4E  1D80                .BYTE   ^0100 ! <C*8> ! M
                        1D81  5130                  MOV     A,C
                    79  1D81                .BYTE   ^0100 ! <A*8> ! C
                        1D82  5131                  CPI     HEX_FF              ;END OF TABLE??
                    FE  1D82                .BYTE   ^0376
                    FF  1D83                .BYTE   HEX_FF
                        1D84  5132                  JZ      1$
                    CA  1D84                .BYTE   ^0312
                1DA8'   1D85                .WORD   <1$-HEAD>
                        1D87  5133                  INX     H
                    23  1D87                .BYTE   <H*8> ! 3
                        1D88  5134                  CALL    COMPARE_R           ;COMPARE TABLE AND BUFFER
                    CD  1D88                .BYTE   ^0315
                0F55'   1D89                .WORD   <COMPARE_R-HEAD>
                        1D8B  5135
                        1D8B  5136                  IFZ
                    C2  1D8B                .BYTE   ^0302
                1D9B'   1D8C                .WORD   <2158$-HEAD>
                        1D8E  5137
                        1D8E  5138                  SHLD    PAR_VECT
                    22  1D8E                .BYTE   ^042
                03D7'   1D8F                .WORD   <PAR_VECT-HEAD>
                        1D91  5139                  CALL    SKIP_TO_SPEC        ;SKIP ALL NONPARSED CHARS
                    CD  1D91                .BYTE   ^0315
                0E33'   1D92                .WORD   <SKIP_TO_SPEC-HEAD>
                        1D94  5140                  LHLD    PAR_VECT
                    2A  1D94                .BYTE   ^052
                03D7'   1D95                .WORD   <PAR_VECT-HEAD>
                        1D97  5141
                        1D97  5142                  PCHL                        ;BRANCH TO ROUTINES
                    E9  1D97                .BYTE   ^0351
                        1D98  5143
                        1D98  5144                  ELSE
                    C3  1D98                .BYTE   ^0303
                1DA5'   1D99                .WORD   <2159$-HEAD>
                        1D9B            2158$:                              ;GENERATE LOCAL SYMBOL NAME
                        1D9B  5145
                        1D9B  5146                  MOV     A,C             ;USE LEFTOVER COUNT FROM
                    79  1D9B                .BYTE   ^0100 ! <A*8> ! C
                        1D9C  5147                  ADI     4               ;COMPARE TO SKIP OVER DATA
                    C6  1D9C                .BYTE   ^0306
                    04  1D9D                .BYTE   4
                        1D9E  5148                  MOV     C,A
                    4F  1D9E                .BYTE   ^0100 ! <C*8> ! A
                        1D9F  5149                  MVI     B,0
```

```
                00 06  1D9F                    .BYTE   <B*8> ! 6 , 0&255
                       1DA1  5150               DAD     B                       ;SPIP TO NEXT COMMAND
                  09   1DA1                    .BYTE   <B*8> ! 9
                       1DA2  5151               SHLD    PARSE_TABLE
                  22   1DA2                    .BYTE   ^042
               03DC'   1DA3                    .WORD   <PARSE_TABLE-HEAD>
                       1DA5  5152
                       1DA5  5153               E_IF
                       1DA5        2159$:                                      ;GENERATE LCL 2159 NAME
                       1DA5  5154
                       1DA5  5155               JMP     2$
                  C3   1DA5                    .BYTE   ^0303
               1D79'   1DA6                    .WORD   <2$-HEAD>
                       1DA8  5156
                       1DA8  5157 1$:           SET     NOT_FOUND               ;NOT FOUND
                  AF   1DA8                    .BYTE   ^0250 ! A               ;with A
                  3C   1DA9                    .BYTE   ^074 ! <8*A>
                  32   1DAA                    .BYTE   ^062
               03C2'   1DAB                    .WORD   <NOT_FOUND-HEAD>
                       1DAD  5158               RET
                  C9   1DAD                    .BYTE   ^0311
                       1DAE  5159
                       1DAE  5160 ;*****************************************************************************
                       1DAE  5161 ;
                       1DAE  5162 ;  PARSER
                       1DAE  5163 ;  THIS ROUTINE PARSES THE COMMAND LINE GIVEN AFTER THE PROMPT AND DISPATCHES
                       1DAE  5164 ;  TO THE PROPER SUBROUTINE TO HANDLE EACH KEYWORD
                       1DAE  5165 ;
                       1DAE  5166 ;*****************************************************************************
                       1DAE  5167
                       1DAE  5168 PARSER:       CLEAR   MICRO_STEP
                  AF   1DAE                    .BYTE   ^0250 ! A               ;with A
                  32   1DAF                    .BYTE   ^062
               038A'   1DB0                    .WORD   <MICRO_STEP-HEAD>
                       1DB2  5169
                       1DB2  5170               WHILENR EXECUTE
                  AF   1DB2                    .BYTE   ^0250 ! A               ;with A
                  32   1DB3                    .BYTE   ^062
               02E4'   1DB4                    .WORD   <EXECUTE-HEAD>
                       1DB6        4005$:       LDA     EXECUTE
                  3A   1DB6                    .BYTE   ^072
               02EA'   1DB7                    .WORD   <EXECUTE-HEAD>
                  0F   1DB9                    .BYTE   ^017
                  DA   1DBA                    .BYTE   ^0332
               1DF8'   1DBB                    .WORD   <2160$-HEAD>
                       1DBD  5171
                       1DBD  5172               IFN     RESTART
                  3A   1DBD                    .BYTE   ^072
               03F1'   1DBE                    .WORD   <RESTART-HEAD>
                  0F   1DC0                    .BYTE   ^017
                  DA   1DC1                    .BYTE   ^0332
               1DCD'   1DC2                    .WORD   <2161$-HEAD>
                       1DC4  5173
                       1DC4  5174               CALL    PRINT_PROMPT
                  CD   1DC4                    .BYTE   ^0315
               0FDE'   1DC5                    .WORD   <PRINT_PROMPT-HEAD>
                       1DC7  5175               CALL    AUTO_TEST
```

```
                      CD  1DC7                .BYTE   ^0315
                    1C24' 1DC8                .WORD   <AUTO_TEST-HEAD>
                          1DCA  5176
                          1DCA  5177                    ELSE
                      C3  1DCA                .BYTE   ^0303
                    1DD4' 1DCB                .WORD   <2162$-HEAD>
                          1DCD        2161$:                                    ;GENERATE LOCAL SYMBOL NAME
                          1DCD  5178
                          1DCD  5179                    LXI     $SP,PRI_STACK       ;SET UP STACK POINTER
                       31 1DCD                .BYTE   <$SP*8> ! 1
                    40 80 1DCE                .BYTE   <PRI_STACK-HEAD>&255 , <<PRI_STACK-HEAD>&^XFF00>/256
                          1DD0  5180                    CLEAR   PESTART
                       AF 1DD0                .BYTE   ^0250 ! A                   ;with A
                       32 1DD1                .BYTE   ^062
                    03F1' 1DD2                .WORD   <RESTART-HEAD>
                          1DD4  5181
                          1DD4  5182                    ENDIF
                          1DD4        2162$:                                    ;GENERATE LCL 2162 NAME
                          1DD4  5183
                          1DD4  5184
                          1DD4  5185                    LXI     H,MAIN_TAB          ;POINTER TO PARSE TABLE
                       21 1DD4                .BYTE   <H*8> ! 1
                    09'31' 1DD5                .BYTE   <MAIN_TAB-HEAD>&255 , <<MAIN_TAB-HEAD>&^XFF00>/256
                          1DD7  5186                    LXI     D,TTBUF             ;POINTER TO CHAR BUFFER
                       11 1DD7                .BYTE   <D*8> ! 1
                    08'D1' 1DD8                .BYTE   <TTBUF-HEAD>&255 , <<TTBUF-HEAD>&^XFF00>/256
                          1DDA  5187                    CALL    PAR
                      CD  1DDA                .BYTE   ^0315
                    1D6E' 1DDB                .WORD   <PAR-HEAD>
                          1DDD  5188
                          1DDD  5189                    IF      CONT_DONE           ;CONTINUE PARSED.. BACKUP
                      3A  1DDD                .BYTE   ^072
                    0256' 1DDE                .WORD   <CONT_DONE-HEAD>
                       0F 1DE0                .BYTE   ^017
                       22 1DE1                .BYTE   ^0322
                    1DEE' 1DE2                .WORD   <2163$-HEAD>
                          1DE4  5190                                            ; INTO STACK ONE LEVEL
                          1DE4  5191                    CLEAR   CONT_DONE
                       AF 1DE4                .BYTE   ^0250 ! A                   ;with A
                       32 1DE5                .BYTE   ^062
                    0256' 1DE6                .WORD   <CONT_DONE-HEAD>
                          1DE8  5192                    MVI     A,1
                    01 3E 1DE8                .BYTE   <A*8> ! 6 , 1&255
                          1DEA  5193                    STA     APT_MESSAGE_CODE+1  ;SET APT START FLAG
                       32 1DEA                .BYTE   ^062
                    0011' 1DEB                .WORD   <APT_MESSAGE_CODE+1-HEAD>
                          1DED  5194                    RET
                       C9 1DED                .BYTE   ^0311
                          1DEE  5195                    ENDIF
                          1DEE        2163$:                                    ;GENERATE LCL 2163 NAME
                          1DEE  5196
                          1DEE  5197                    LDA     NOT_FOUND           ;GET NOT FOUND FLAG
                      3A  1DEE                .BYTE   ^072
                    0302' 1DEF                .WORD   <NOT_FOUND-HEAD>
                          1DF1  5198                    ORA                         ;SET CONDITION CODES
                      B7  1DF1                .BYTE   ^0260 ! A
                          1DF2  5199                    CNZ     SYNTAX_ERROR        ;REPORT SYNTAX ERROR IF SET
```

```
                C4  1DF2                .BYTE   ^0304
              1BE0' 1DF3                .WORD   <SYNTAX_ERROR-HEAD>
                    1DF5  5200
                    1DF5  5201                  ENDWHILE
                C3  1DF5                .BYTE   ^0303
              1DB6' 1DF6                .WORD   <4005$-HEAD>
                    1DF8        2160$:                                  ;GENERATE LCL 2160 NAME
                    1DF8  5202
                    1DF8  5203
                    1DF8  5204
                    1DF8  5205 ;*****************************************************************************
                    1DF8  5206 ;
                    1DF8  5207 ;  THIS ROUTINE PARSES THE SECTION TABLE
                    1DF8  5208 ;
                    1DF8  5209 ;  THERE IS A FLAG WITH EACH SECTION NAME (SECTION_FLAG)
                    1DF8  5210 ;
                    1DF8  5211 ;  SECTION_FLAG:              FF = END OF TABLE
                    1DF8  5212 ;                      BIT 7   0 = NAME IS SECTION NAME FOR WCS
                    1DF8  5213 ;                              1 = NAME IS SECTION NAME FOR CONSOLE RAM
                    1DF8  5214 ;                      BIT 6   1 = OPTIONAL TEST
                    1DF8  5215 ;                BITS 0 - 5   XX = CPU CLUSTER BOARD NUMBER
                    1DF8  5216 ;
                    1DF8  5217 ;  THERE IS A FLAG STATING HOW THE TABLE IS TO BE PARSED (TABLE_MODE)
                    1DF8  5218 ;
                    1DF8  5219 ;  TABLE_MODE    0 = TABLE NOT IN USE
                    1DF8  5220 ;                 1 = BOARD MODE
                    1DF8  5221 ;                 2 = ALL SECTION MODE
                    1DF8  5222 ;
                    1DF8  5223 ;*****************************************************************************
                    1DF8  5224
                    1DF8  5225                  WHILE   EXECUTE
                    1DF8        4006$:  LDA     EXECUTE
                3A  1DF8                .BYTE   ^072
              02EA' 1DF9                .WORD   <EXECUTE-HEAD>
                0F  1DFB                .BYTE   ^017
                D2  1DFC                .BYTE   ^0322
              1E87' 1DFD                .WORD   <2164$-HEAD>
                    1DFF  5226
                    1DFF  5227                  IFNEQI  TABLE_MODE,0
                3A  1DFF                .BYTE   ^072
              040B' 1E00                .WORD   <TABLE_MODE-HEAD>
                FE  1E02                .BYTE   ^0376
                00  1E03                .BYTE   0
                CA  1E04                .BYTE   ^0312
              1E63' 1E05                .WORD   <2165$-HEAD>
                    1E07  5228
                    1E07  5229                  CALL    SET_ALL_TESTS           ;SET UP TEST NUM FOR ALL
                CD  1E07                .BYTE   ^0315
              1EBA' 1E08                .WORD   <SET_ALL_TESTS-HEAD>
                    1E0A  5230                                              ; TESTS AND CLEAR EOS FLG
                    1E0A  5231                  UNTILR  FILE_LOADED
                AF  1E0A                .BYTE   ^0250 ! A               ;with A
                32  1E0B                .BYTE   ^062
              02FD' 1E0C                .WORD   <FILE_LOADED-HEAD>
                    1E0E        4007$:  LDA     FILE_LOADED
                3A  1E0E                .BYTE   ^072
              02FD' 1E0F                .WORD   <FILE_LOADED-HEAD>
```

```
                     OF  1E11                .BYTE   ^017
                     DA  1E12                .BYTE   ^0332
                1E60'    1E13                .WORD   <2166$-HEAD>
                         1E15  5232
                         1E15  5233                  LHLD    TABLE_PNT               ;POINTER TO SECTION TABLE
                     2A  1E15                .BYTE   ^052
                040C'    1E16                .WORD   <TABLE_PNT-HEAD>
                         1E18  5234                  LXI     D,TEMP_FLAG             ;POINTER TO TEMP DATA BUFFER
                     11  1E18                .BYTE   <D*8> ! 1
               04'1F'    1E19                .BYTE   <TEMP_FLAG-HEAD>&255 , <<TEMP_FLAG-HEAD>&^XFF00>/256
                         1E1B  5235                  MVI     C,SEC_LEN+1             ;COUNT
                07 0E    1E1B                .BYTE   <C*8> ! 6 , SEC_LEN+1&255
                         1E1D  5236                  CALL    MOVER_R                 ;MOVE ENTRIES TO LOCAL BUFFER
                     CD  1E1D                .BYTE   ^0315
                0F75'    1E1E                .WORD   <MOVER_R-HEAD>
                         1E20  5237                  SHLD    TABLE_PNT               ;UPDATE SECTION TABLE POINTER
                     22  1E20                .BYTE   ^042
                040C'    1E21                .WORD   <TABLE_PNT-HEAD>
                         1E23  5238
                         1E23  5239                  LDA     TEMP_FLAG
                     3A  1E23                .BYTE   ^072
                041F'    1E24                .WORD   <TEMP_FLAG-HEAD>
                         1E26  5240                  ANI     HEX_3F
                     E6  1E26                .BYTE   ^0346
                     3F  1E27                .BYTE   HEX_3F
                         1E28  5241                  STA     SECTION_B_NUM           ;SAVE BOARD THAT OWNS THIS SEC
                     32  1E28                .BYTE   ^062
                0404'    1E29                .WORD   <SECTION_B_NUM-HEAD>
                         1E2B  5242
                         1E2B  5243                  IFEQI   TEMP_FLAG,HEX_FF        ;END OF TABLE??
                     3A  1E2B                .BYTE   ^072
                041F'    1E2C                .WORD   <TEMP_FLAG-HEAD>
                     FE  1E2E                .BYTE   ^0376
                     FF  1E2F                .BYTE   HEX_FF
                     C2  1E30                .BYTE   ^0302
                1E39'    1E31                .WORD   <2167$-HEAD>
                         1E33  5244
                         1E33  5245                  CALL    EOP
                     CD  1E33                .BYTE   ^0315
                183E'    1E34                .WORD   <EOP-HEAD>
                         1E36  5246
                         1E36  5247                  ELSE
                     C3  1E36                .BYTE   ^0303
                1E5D'    1E37                .WORD   <2168$-HEAD>
                         1E39        2167$:                                      ;GENERATE LOCAL SYMBOL NAME
                         1E39  5248
                         1E39  5249
                         1E39  5250                  IFNMBI  TEMP_FLAG,OPTIONAL_TST  ;OPTIONAL TEST??
                     3A  1E39                .BYTE   ^072
                041F'    1E3A                .WORD   <TEMP_FLAG-HEAD>
                     E6  1E3C                .BYTE   ^0346
                     40  1E3D                .BYTE   OPTIONAL_TST
                     C2  1E3E                .BYTE   ^0302
                1E5D'    1E3F                .WORD   <2169$-HEAD>
                         1E41  5251
                         1E41  5252                  IFEQI   TABLE_MODE,1            ;BOARD MODE??
                     3A  1E41                .BYTE   ^072
```

```
040B' 1E42                .WORD   <TABLE_MODE-HEAD>
  FE  1E44                .BYTE   ^0376
  01  1E45                .BYTE   1
  C2  1E46                .BYTE   ^0302
1E5A' 1E47                .WORD   <2170$-HEAD>
      1E49  5253
      1E49  5254                    IFMB    BOARD_NUM,SECTION_B_NUM ;SEC BELONG TO THIS BOARD?
  3A  1E49                .BYTE   ^072
0404' 1E4A                .WORD   <SECTION_B_NUM-HEAD>
  47  1E4C                .BYTE   ^0100 ! <B*8> ! A
  3A  1E4D                .BYTE   ^072
024D' 1E4E                .WORD   <BOARD_NUM-HEAD>
  A0  1E50                .BYTE   ^0240 ! B
  CA  1E51                .BYTE   ^0312
1E57' 1E52                .WORD   <2171$-HEAD>
      1E54  5255                    CALL    MOV_LOAD_SECTION        ;MOVE NAME TO SECTION_NAME AND
  CD  1E54                .BYTE   ^0315
1A80' 1E55                .WORD   <MOV_LOAD_SECTION-HEAD>
      1E57  5256                                                    ; LOAD SECTION
      1E57  5257                    ENDIF
      1E57      2171$:                                            ;GENERATE LCL 2171 NAME
      1E57  5258
      1E57  5259                    ELSE
  C3  1E57                .BYTE   ^0303
1E5D' 1E58                .WORD   <2172$-HEAD>
      1E5A      2170$:                                            ;GENERATE LOCAL SYMBOL NAME
      1E5A  5260                    CALL    MOV_LOAD_SECTION        ;IF DI MODE, MOVE NAME TO
  CD  1E5A                .BYTE   ^0315
1A80' 1E5B                .WORD   <MOV_LOAD_SECTION-HEAD>
      1E5D  5261                                                    ; SECTION_NAME AND LOAD SECTION
      1E5D  5262                    ENDIF
      1E5D      2172$:                                            ;GENERATE LCL 2172 NAME
      1E5D  5263
      1E5D  5264                    ENDIF
      1E5D      2169$:                                            ;GENERATE LCL 2169 NAME
      1E5D  5265
      1E5D  5266                    ENDIF
      1E5D      2168$:                                            ;GENERATE LCL 2168 NAME
      1E5D  5267
      1E5D  5268                    ENDUNTIL
  C3  1E5D                .BYTE   ^0303
1E0E' 1E5E                .WORD   <4007$-HEAD>
      1E60      2166$:                                            ;GENERATE LCL 2166 NAME
      1E60  5269
      1E60  5270                    ELSE
  C3  1E60                .BYTE   ^0303
1E6A' 1E61                .WORD   <2173$-HEAD>
      1E63      2165$:                                            ;GENERATE LOCAL SYMBOL NAME
      1E63  5271
      1E63  5272                    LDA     EOS_FLG                 ;GET EOS FLAG
  3A  1E63                .BYTE   ^072
0056' 1E64                .WORD   <EOS_FLG-HEAD>
      1E66  5273                    ORA     A                       ;SET CONDITION CODES
  B7  1E66                .BYTE   ^0260 ! A
      1E67  5274                    CNZ     EOP                     ;CALL EOP IF SET
  C4  1E67                .BYTE   ^0304
183E' 1E68                .WORD   <EOP-HEAD>
```

```
                        1E6A  5275
                        1E6A  5276                      ENDIF
                        1E6A        2173$:                                      ;GENERATE LCL 2173 NAME
                        1E6A  5277
                        1E6A  5278                      IF      EXECUTE
                  3A    1E6A                  .BYTE     ^072
              02EA'     1E6B                  .WORD     <EXECUTE-HEAD>
                  0F    1E6D                  .BYTE     ^017
                  D2    1E6E                  .BYTE     ^0322
              1EB4'     1E6F                  .WORD     <2174$-HEAD>
                        1E71  5279
                        1E71  5280                      LDA     APT_PASS_CNT+1          ;GET PASS CNT
                  3A    1E71                  .BYTE     ^072
              0019'     1E72                  .WORD     <APT_PASS_CNT+1-HEAD>
                        1E74  5281                      ORA     A                       ;SET CONDITION CODES
                  B7    1E74                  .BYTE     ^0260 ! A
                        1E75  5282                      JNZ     1$                      ;JUMP OUT IF LSB NOT 0
                  C2    1E75                  .BYTE     ^0302
              1E7C'     1E76                  .WORD     <1$-HEAD>
                        1E78  5283                      LDA     APT_PASS_CNT            ;GET MSB OF PASS CNT
                  3A    1E78                  .BYTE     ^072
              0018'     1E79                  .WORD     <APT_PASS_CNT-HEAD>
                        1E7B  5284                      ORA     A                       ;SET CONDITION CODES
                  B7    1E7B                  .BYTE     ^0260 ! A
                        1E7C  5285
                        1E7C  5286 1$:                  IF      CONSOLE_TEST
                  3A    1E7C                  .BYTE     ^072
              0255'     1E7D                  .WORD     <CONSOLE_TEST-HEAD>
                  0F    1E7F                  .BYTE     ^017
                  D2    1E80                  .BYTE     ^0322
              1E9E'     1E81                  .WORD     <2175$-HEAD>
                        1E83  5287                      CLEAR   FAST_WRITE
                  AF    1E83                  .BYTE     ^0250 ! A                   ;with A
                  32    1E84                  .BYTE     ^062
              02FC'     1E85                  .WORD     <FAST_WRITE-HEAD>
                        1E87  5288                      ZERO    DATA0,4                 ;RESET OUT OF COMPAT. MODE
                  21    1E87                  .BYTE     <H*8> ! 1
            00'A9'      1E88                  .BYTE     <DATA0-HEAD>&255 , <<DATA0-HEAD>&^XFF00>/256
            04 0E       1E8A                  .BYTE     <C*8> ! 6 , 4&255
                  AF    1E8C                  .BYTE     ^0250 ! A                   ;with A
                        1E8D        2176$:              MOV     M,A
                  77    1E8D                  .BYTE     ^0100 ! <M*8> ! A
                  23    1E8E                  .BYTE     <H*8> ! 3
                  0D    1E8F                  .BYTE     ^05 ! <8*C>
                  C2    1E90                  .BYTE     ^0302
              1E8D'     1E91                  .WORD     <2176$-HEAD>
                        1E93  5289                      MVI     A,HEX_FF
            FF 3E       1E93                  .BYTE     <A*8> ! 6 , HEX_FF&255
                        1E95  5290                      CALL    WRITE_LS_R
                  CD    1E95                  .BYTE     ^0315
              1146'     1E96                  .WORD     <WRITE_LS_R-HEAD>
                        1E98  5291                      CALL    EXEC_CONTESTS
                  CD    1E98                  .BYTE     ^0315
              7000      1E99                  .WORD     <EXEC_CONTESTS-HEAD>
                        1E9B  5292
                        1E9B  5293                      ELSE
                  C3    1E9B                  .BYTE     ^0303
```

```
1EB4' 1E9C                .WORD   <2177$-HEAD>
      1E9E      2175$:
      1E9E  5294
      1E9E  5295                LDA     TEST_NUM1               ;USE TEST NUMBER AS STARTING
  3A  1E9E                .BYTE   ^072
0427' 1E9F                .WORD   <TEST_NUM1-HEAD>
      1EA1  5296                MOV     H,A                     ;ADDRESS IN WCS
  67  1EA1                .BYTE   ^0100 ! <H*8> ! A
      1EA2  5297                XRA     A
  AF  1EA2                .BYTE   ^0250 ! A               ;with A
      1EA3  5298                MOV     L,A
  6F  1EA3                .BYTE   ^0100 ! <L*8> ! A
      1EA4  5299                SHLD    STARTING_UPC
  22  1EA4                .BYTE   ^042
0407' 1EA5                .WORD   <STARTING_UPC-HEAD>
      1EA7  5300
      1EA7  5301                LDA     TEST_NUM1               ;SET UP FOR INC ON BEG OF TEST
  3A  1EA7                .BYTE   ^072
0427' 1EA8                .WORD   <TEST_NUM1-HEAD>
      1EAA  5302                DCR     A
  3D  1EAA                .BYTE   ^05 ! <8*A>
      1EAB  5303                STA     TEST_NUM
  32  1EAB                .BYTE   ^062
0426' 1EAC                .WORD   <TEST_NUM-HEAD>
      1EAE  5304
      1EAE  5305                CALL    START_CPU
  CD  1EAE                .BYTE   ^0315
10FF' 1EAF                .WORD   <START_CPU-HEAD>
      1EB1  5306                CALL    ATTENTION
  CD  1EB1                .BYTE   ^0315
1321' 1EB2                .WORD   <ATTENTION-HEAD>
      1EB4  5307
      1EB4  5308                ENDIF
      1EB4      2177$:                                          ;GENERATE LCL 2177 NAME
      1EB4  5309
      1EB4  5310                ENDIF
      1EB4      2174$:                                          ;GENERATE L L 2174 NAME
      1EB4  5311
      1EB4  5312                ENDWHILE
  C3  1EB4                .BYTE   ^0303
1DF8' 1EB5                .WORD   <4006$-HEAD>
      1EB7      2164$:                                          ;GENERATE LCL 2164 NAME
      1EB7  5313
      1EB7  5314                JMP     PARSER
  C3  1EB7                .BYTE   ^0303
1DAE' 1EB8                .WORD   <PARSER-HEAD>
      1EBA  5315
      1EBA  5316 ;*****************************************************************************
      1EBA  5317 ;
      1EBA  5318 ;  SET_ALL_TESTS
      1EBA  5319 ;
      1EBA  5320 ;  THIS ROUTINE SETS STARTING TEST NUM TO 1, ENDING TEST NUM TO 50(H) AND
      1EBA  5321 ;  CLEARS THE EOS FLAG.
      1EBA  5322 ;
      1EBA  5323 ;*****************************************************************************
      1EBA  5324
      1EBA  5325 SET_ALL_TESTS:
```

```
                1EBA  5326                      MVI     A,1
01 3E           1EBA                    .BYTE   <A*8> ! 6 , 1&255
                1EBC  5327                      STA     TEST_NUM1               ;SET TO START AT TEST 1(D)
  32            1EBC                    .BYTE   ^O62
 0427'          1EBD                    .WORD   <TEST_NUM1-HEAD>
                1EBF  5328
                1EBF  5329                      MVI     A,HEX_50
50 3E           1EBF                    .BYTE   <A*8> ! 6 , HEX_50&255
                1EC1  5330                      STA     TEST_NUM2               ;SET UP ENDING TEST
  32            1EC1                    .BYTE   ^O62
 0428'          1EC2                    .WORD   <TEST_NUM2-HEAD>
                1EC4  5331
                1EC4  5332                      CLEAR   EOS_FLG
  AF            1EC4                    .BYTE   ^O250 ! A               ;with A
  32            1EC5                    .BYTE   ^O62
 0056'          1EC6                    .WORD   <EOS_FLG-HEAD>
                1EC8  5333
                1EC8  5334                      RET                             ;DONE
  C9            1EC8                    .BYTE   ^O311
                1EC9  5335
                1EC9  5336 ;
                1EC9  5337 ;CONSOLE TEST SUBROUTINES
                1EC9  5338 ;
                1EC9  5339
                1EC9  5340 ;*****************************************************************************
                1EC9  5341 ;
                1EC9  5342 ;  CON_BEGIN_TEST_R  THIS ROUTINE IS USED FOR BEGINNING OF TEST FOR CONSOLE
                1EC9  5343 ;                    BASED CPU TESTS ONLY.
                1EC9  5344 ;
                1EC9  5345 ;                    INPUT CONDITIONS:
                1EC9  5346 ;                        A=TEST NUMBER
                1EC9  5347 ;
                1EC9  5348 ;*****************************************************************************
                1EC9  5349
                1EC9  5350 CON_BEGIN_TEST_R:
                1EC9  5351                      STA     TEST_NUM                ;SAVE TEST NUMBER
  32            1EC9                    .BYTE   ^O62
 0426'          1ECA                    .WORD   <TEST_NUM-HEAD>
                1ECC  5352                      SET     CONTINUE                ;SET UP TO CONTINUE IF TEST
  AF            1ECC                    .BYTE   ^O250 ! A               ;with A
  3C            1ECD                    .BYTE   ^O4 ! <8*A>
  32            1ECE                    .BYTE   ^O62
 003A'          1ECF                    .WORD   <CONTINUE-HEAD>
                1ED1  5353                                                      ; STOPPED BY OTHER THAN ^P
                1ED1  5354                      CALL    GCVECT                  ;CHECK FOR OPERATOR REQUEST
  CD            1ED1                    .BYTE   ^O315
4129            1ED2                    .WORD   <GCVECT-HEAD>
                1ED4  5355
                1ED4  5356                      CALL    CHECK_CNTLC             ;IF ^C, GO BACK TO PARSER AND
  CD            1ED4                    .BYTE   ^O315
1073'           1ED5                    .WORD   <CHECK_CNTLC-HEAD>
                1ED7  5357                                                      ; ALLOW CONTINUE
                1ED7  5358
                1ED7  5359                      MVI     A,1
01 3E           1ED7                    .BYTE   <A*8> ! 6 , 1&255
                1ED9  5360                      STA     APT_MESSAGE_CODE+1      ;SET APT START FLAG
  32            1ED9                    .BYTE   ^O62
```

```
                0011' 1EDA                .WORD   <APT_MESSAGE_CODE+1-HEAD>
                      1EDC  5361
                      1EDC  5362                  IFGT    TEST_NUM1,TEST_NUM,1
                  21  1EDC                .BYTE   <H*8> ! 1
               04'27' 1EDD                .BYTE   <TEST_NUM1-HEAD>&255 , <<TEST_NUM1-HEAD>&^XFF00>/256
                  11  1EDF                .BYTE   <D*8> ! 1
               04'26' 1EE0                .BYTE   <TEST_NUM-HEAD>&255 , <<TEST_NUM-HEAD>&^XFF00>/256
               01 0E  1EE2                .BYTE   <C*8> ! 6 , 1&255
                  CD  1EE4                .BYTE   ^0315
                00CF' 1EE5                .WORD   <COMPARE-HEAD>
                  F2  1EE7                .BYTE   ^0362
                1EF2' 1EE8                .WORD   <2178$-HEAD>
                      1EEA  5363                  SET     SKIP_TEST
                  AF  1EEA                .BYTE   ^0250 ! A                       ;with A
                  3C  1EEB                .BYTE   ^04 ! <8*A>
                  32  1EEC                .BYTE   ^062
                006A' 1EED                .WORD   <SKIP_TEST-HEAD>
                      1EEF  5364
                      1EEF  5365                  ELSE
                  C3  1EEF                .BYTE   ^0303
                1F13' 1EF0                .WORD   <2179$-HEAD>
                      1EF2        2178$:                                        ;GENERATE LOCAL SYMBOL NAME
                      1EF2  5366
                      1EF2  5367                  CLEAR   SKIP_TEST
                  AF  1EF2                .BYTE   ^0250 ! A                       ;with A
                  32  1EF3                .BYTE   ^062
                006A' 1EF4                .WORD   <SKIP_TEST-HEAD>
                      1EF6  5368
                      1EF6  5369                  IFGT    TEST_NUM,TEST_NUM2,1
                  21  1EF6                .BYTE   <H*8> ! 1
               04'26' 1EF7                .BYTE   <TEST_NUM-HEAD>&255 , <<TEST_NUM-HEAD>&^XFF00>/256
                  11  1EF9                .BYTE   <D*8> ! 1
               04'28' 1EFA                .BYTE   <TEST_NUM2-HEAD>&255 , <<TEST_NUM2-HEAD>&^XFF00>/256
               01 0E  1EFC                .BYTE   <C*8> ! 6 , 1&255
                  CD  1EFE                .BYTE   ^0315
                00CF' 1EFF                .WORD   <COMPARE-HEAD>
                  F2  1F01                .BYTE   ^0362
                1F0D' 1F02                .WORD   <2180$-HEAD>
                      1F04  5370                  SET     EOS_FLG                 ;FORCE END OF SECTION
                  AF  1F04                .BYTE   ^0250 ! A                       ;with A
                  3C  1F05                .BYTE   ^04 ! <8*A>
                  32  1F06                .BYTE   ^062
                0056' 1F07                .WORD   <EOS_FLG-HEAD>
                      1F09  5371                  RET
                  C9  1F09                .BYTE   ^0311
                      1F0A  5372
                      1F0A  5373                  ELSE
                  C3  1F0A                .BYTE   ^0303
                1F13' 1F0B                .WORD   <2181$-HEAD>
                      1F0D        2180$:                                        ;GENERATE LOCAL SYMBOL NAME
                      1F0D  5374                  LDA     TEST_NUM                ;GET TEST NUMBER
                  3A  1F0D                .BYTE   ^072
                0426' 1F0E                .WORD   <TEST_NUM-HEAD>
                      1F10  5375                  STA     APT_TEST_NUMBER+1       ;SAVE IN APT MAILBOX
                  32  1F10                .BYTE   ^062
                0017' 1F11                .WORD   <APT_TEST_NUMBER+1-HEAD>
                      1F13  5376                  ENDIF
```

```
                1F13        2181$:                                  ;GENERATE LCL 2181 NAME
                1F13  5377
                1F13  5378                  ENDIF
                1F13        2179$:                                  ;GENERATE LCL 2179 NAME
                1F13  5379
                1F13  5380                  RET
             C9 1F13                .BYTE   ^0311
                1F14  5381
                1F14  5382
                1F14  5383
                1F14  5384 ;*****************************************************************************
                1F14  5385 ;
                1F14  5386 ;  CON_ERROR_R  THIS ROUTINE IS EXECUTED TO PRINT AN ERROR FOUND BY A
                1F14  5387 ;                CONSOLE BASED CPU TEST.
                1F14  5388 ;
                1F14  5389 ;*****************************************************************************
                1F14  5390
                1F14  5391
                1F14  5392 CON_ERROR_R:     OUT     CLRLIT          ;TURN RUN LITE OFF
          3C D3 1F14                .BYTE   ^0323 , CLRLIT&255      ;register A
                1F16  5393                  CALL    MSK_ERROR_R     ;MASK THE EXPECTED AND RECEIVED
             CD 1F16                .BYTE   ^0315
           1F31' 1F17                .WORD   <MSK_ERROR_R-HEAD>
                1F19  5394                                          ; DATA TO ELIMINATE ANY BITS
                1F19  5395                                          ; WHICH ARE NOT OF INTEREST.
                1F19  5396                  CALL    WCS_ERROR       ;CALL MICMON ROUTINE TO PRINT ERROR
             CD 1F19                .BYTE   ^0315
           1442' 1F1A                .WORD   <WCS_ERROR-HEAD>
                1F1C  5397                                          ; REPORT
                1F1C  5398                  LDA     FLAGS           ;DO NOT SET ERROR CON FLAG IF LOOP
             3A 1F1C                .BYTE   ^072
          005A' 1F1D                .WORD   <FLAGS-HEAD>
                1F1F  5399                  RAR                     ; ON ERROR IS NOT SET
             1F 1F1F                .BYTE   ^037
                1F20  5400                  JNC     1$
             D2 1F20                .BYTE   ^0322
          1F2E' 1F21                .WORD   <1$-HEAD>
                1F23  5401
                1F23  5402                  SET     ERROR_CON       ;SET FLAG SAYING ERROR HAPPENED AND
             AF 1F23                .BYTE   ^0250 ! A               ;with A
             3C 1F24                .BYTE   ^04 ! <8*A>
             32 1F25                .BYTE   ^062
          0057' 1F26                .WORD   <ERROR_CON-HEAD>
                1F28  5403                                          ; LOOP ON ERROR WAS SET
                1F28  5404                  CALL    GCVECT          ;CHECK FOR OPERATOR INPUT
             CD 1F28                .BYTE   ^0315
           4129 1F29                .WORD   <GCVECT-HEAD>
                1F2B  5405                  CALL    CHECK_CNTLC     ;CHECK FOR CONTROL C INPUT
             CD 1F2B                .BYTE   ^0315
           1073' 1F2C                .WORD   <CHECK_CNTLC-HEAD>
                1F2E  5406 1$:              OUT     SETLIT          ;TURN RUN LITE BACK ON
          3D D3 1F2E                .BYTE   ^0323 , SETLIT&255      ;register A
                1F30  5407                  RET
             C9 1F30                .BYTE   ^0311
                1F31  5408
                1F31  5409
                1F31  5410 ;*****************************************************************************
```

```
                        1F31  5411 ;
                        1F31  5412 ;  MSK_ERROR_R  THIS ROUTINE MASKS DATA FROM BOTH THE CON_EXP_DAT AND
                        1F31  5413 ;               CON_REC_DAT AREAS BY USING THE ERROR MASK DATA. A ONE
                        1F31  5414 ;               IN THE ERROR MASK INDICATES THAT THAT PARTICULAR BIT IS
                        1F31  5415 ;               NOT OF INTEREST AND SHOULD BE SET TO ZERO.
                        1F31  5416 ;
                        1F31  5417 ;*****************************************************************************
                        1F31  5418
                        1F31  5419 MSK_ERROR_R:    LXI     H,CON_EXP_DAT   ;MOVE MEMORY ADDRESS CON_EXP_DAT
                  21    1F31                .BYTE   <H*8> ! 1
               00'40'   1F32                .BYTE   <CON_EXP_DAT-HEAD>&255 , <<CON_EXP_DAT-HEAD>&^XFF00>/256
                        1F34  5420                                          ; INTO THE H,L PAIR.
                        1F34  5421                      SHLD    CON_EXP_ADD     ;STORE THE ADDRESS OF THE EXPECTED
                  22    1F34                .BYTE   ^042
                0044'   1F35                .WORD   <CON_EXP_ADD-HEAD>
                        1F37  5422                                          ; DATA IN CON_EXP_DAT.
                        1F37  5423                      LXI     D,CON_MSK_DAT   ;MOVE MEMORY ADDRESS CON_MSK_DAT
                  11    1F37                .BYTE   <D*8> ! 1
               00'48'   1F38                .BYTE   <CON_MSK_DAT-HEAD>&255 , <<CON_MSK_DAT-HEAD>&^XFF00>/256
                        1F3A  5424                                          ; INTO THE D,E PAIR.
                        1F3A  5425
                        1F3A  5426                      DO      4               ;LOOP 4 TIMES TO MASK 4 BYTES OF
                  21    1F3A                .BYTE   <H*8> ! 1
               00'55'   1F3B                .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
                  46    1F3D                .BYTE   ^0100 ! <B*8> ! M
                  C5    1F3E                .BYTE   ^0305 ! <8*B>
               04 3E    1F3F                .BYTE   <A*8> ! 6 , 4&255
                  77    1F41                .BYTE   ^0100 ! <M*8> ! A
                        1F42          2182$:
                        1F42  5427                                          ; EXPECTED DATA.
                        1F42  5428                      LHLD    CON_EXP_ADD     ;MOVE THE MEMORY ADDRESS OF THE BYTE
                  2A    1F42                .BYTE   ^052
                0044'   1F43                .WORD   <CON_EXP_ADD-HEAD>
                        1F45  5429                                          ; OF EXPECTED DATA INTO THE H,L PAIR.
                        1F45  5430                      CALL    MSK_BYTE        ;CALL ROUTINE TO MASK 1 BYTE
                  CD    1F45                .BYTE   ^0315
               1F78'    1F46                .WORD   <MSK_BYTE-HEAD>
                        1F48  5431                      SHLD    CON_EXP_ADD     ;STORE THE MEMORY ADDRESS OF THE
                  22    1F48                .BYTE   ^042
                0044'   1F49                .WORD   <CON_EXP_ADD-HEAD>
                        1F4B  5432                                          ; EXPECTED DATA INTO CON_EXP_ADD.
                        1F4B  5433                      ENDDO
                  21    1F4B                .BYTE   <H*8> ! 1
               00'55'   1F4C                .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
                  35    1F4E                .BYTE   ^05 ! <8*M>
                  C2    1F4F                .BYTE   ^0302
               1F42'    1F50                .WORD   <2182$-HEAD>
                  C1    1F52                .BYTE   ^0301 ! <8*B>
                  70    1F53                .BYTE   ^0100 ! <M*8> ! B
                        1F54  5434
                        1F54  5435                      LXI     H,CON_REC_DAT   ;MOVE MEMORY ADDRESS CON_REC_DAT
                  21    1F54                .BYTE   <H*8> ! 1
               00'4F'   1F55                .BYTE   <CON_REC_DAT-HEAD>&255 , <<CON_REC_DAT-HEAD>&^XFF00>/256
                        1F57  5436                                          ; INTO THE H,L PAIR.
                        1F57  5437                      SHLD    CON_REC_ADD     ;STORE THE ADDRESS OF THE RECEIVED
                  22    1F57                .BYTE   ^042
                0053'   1F58                .WORD   <CON_REC_ADD-HEAD>
```

```
                 1F5A  5438                                     ; DATA IN CON_REC_ADD.
                 1F5A  5439                     LXI     D,CON_MSK_DAT   ;MOV MEMORY ADDRESS CON_MSK_DAT
      11         1F5A                   .BYTE   <D*8> ! 1
00'4B'           1F5B                   .BYTE   <CON_MSK_DAT-HEAD>&255 , <<CON_MSK_DAT-HEAD>&^XFF00>/256
                 1F5D  5440                                     ; INTO THE D,E PAIR.
                 1F5D  5441
                 1F5D  5442                     DO      4               ;LOOP 4 TIMES TO MASK 4 BYTES OF
      21         1F5D                   .BYTE   <H*8> ! 1
00'55'           1F5E                   .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
      46         1F60                   .BYTE   ^0100 ! <B*8> ! M
      C5         1F61                   .BYTE   ^0305 ! <8*B>
   04 3E         1F62                   .BYTE   <A*8> ! 6 , 4&255
      77         1F64                   .BYTE   ^0100 ! <M*8> ! A
                 1F65           2183$:
                 1F65  5443                                     ; RECEIVED DATA.
                 1F65  5444                     LHLD    CON_REC_ADD     ;LOAD THE ADDRESS OF THE RECEIVED
      2A         1F65                   .BYTE   ^052
 0053'           1F66                   .WORD   <CON_REC_ADD-HEAD>
                 1F68  5445                                     ; DATA INTO THE H,L PAIR.
                 1F68  5446                     CALL    MSK_BYTE        ;CALL ROUTINE TO MASK ONE BYTE
      CD         1F6E                   .BYTE   ^0315
 1F78'           1F69                   .WORD   <MSK_BYTE-HEAD>
                 1F6B  5447                     SHLD    CON_REC_ADD     ;STORE THE ADDRESS OF THE RECEIVED.
      22         1F6B                   .BYTE   ^042
 0053'           1F6C                   .WORD   <CON_REC_ADD-HEAD>
                 1F6E  5448                                     ; DATA IN CON_REC_ADD.
                 1F6E  5449                     ENDDO
      21         1F6E                   .BYTE   <H*8> ! 1
00'55'           1F6F                   .BYTE   <DOLOOP-HEAD>&255 , <<DOLOOP-HEAD>&^XFF00>/256
      35         1F71                   .BYTE   ^05 ! <8*M>
      C2         1F72                   .BYTE   ^0302
 1F65'           1F73                   .WORD   <2183$-HEAD>
      C1         1F75                   .BYTE   ^0301 ! <8*B>
      70         1F76                   .BYTE   ^0100 ! <M*8> ! B
                 1F77  5450
                 1F77  5451                     RET
      C9         1F77                   .BYTE   ^0311
                 1F78  5452
                 1F78  5453 MSK_BYTE:           MOV     B,M             ;MOVE A BYTE OF EXPECTED DATA INTO
      46         1F78                   .BYTE   ^0100 ! <B*8> ! M
                 1F79  5454                                     ; REGISTER B.
                 1F79  5455                     LDAX    D               ;MOVE A BYTE OF MASK DATA INTO
      1A         1F79                   .BYTE   <D*8> ! 10
                 1F7A  5456                                     ; THE ACCUMULATOR.
                 1F7A  5457                     CMA                     ;CHANGE THE MASK POLARITY BY
      2F         1F7A                   .BYTE   ^057
                 1F7B  5458                                     ;COMPLEMENTING THE ACCUMULATOR.
                 1F7B  5459                     ANA     B               ;''AND'' THE COMPLEMENTED MASK WITH
      A0         1F7B                   .BYTE   ^0240 ! B
                 1F7C  5460                                     ; THE REGISTER B CONTENTS.
                 1F7C  5461                     MOV     M,A             ;MOVE THE MASKED EXPECTED DATA BACK
      77         1F7C                   .BYTE   ^0100 ! <M*8> ! A
                 1F7D  5462                                     ; INTO THE EXPECTED DATA LOCATION.
                 1F7D  5463                     INX     H               ;INCREMENT THE D,E AND H,L REGISTER
      23         1F7D                   .BYTE   <H*8> ! 3
                 1F7E  5464                     INX     D               ; PAIRS TO POINT AT THE NEXT BYTES
      13         1F7F                   .BYTE   <D*8> ! 3
```

```
                        1F7F  5465                                      ; OF EXPECTED AND RECEIVED DATA.
                        1F7F  5466                      RET             ;DONE
                  C9    1F7F                    .BYTE   ^O311
                        1F80  5467
                        1F80  5468
                        1F80  5469 ;*****************************************************************************
                        1F80  5470 ;
                        1F80  5471 ;  WCS_READ_R  THIS ROUTINE READS DATA FROM WCS AT THE LOCATION GIVEN BY
                        1F80  5472 ;              WCS_ADDRESS. THE DATA IS PUT INTO THE CSR AND SHIFTED OUT
                        1F80  5473 ;              INTO WCS_VALUE FOR CHECKING.  AS THE CSR IS READ, IT IS
                        1F80  5474 ;              LOADED WITH A NOP INSTRUCTION.
                        1F80  5475 ;
                        1F80  5476 ;              THIS ROUTINE IS USED BY THE CONSOLE TESTS AND IS TAYLORED
                        1F80  5477 ;              FOR FAST EXECUTION RATHER THAN VERSITILITY.
                        1F80  5478 ;
                        1F80  5479 ;*****************************************************************************
                        1F80  5480
                        1F80  5481
                        1F80  5482 WCS_READ_R:  CALL    LD_UPC_R        ;LOAD UPC WITH WCS ADDRESS
                     CD 1F80                    .BYTE   ^O315
                1FCC' 1F81                      .WORD   <LD_UPC_R-HEAD>
                        1F83  5483              OUT     SETSS           ;SINGLE STEP MACHINE TO LOAD
                  23 D3 1F83                    .BYTE   ^O323 , SETSS&255       ;register A
                        1F85  5484              OUT     CLRSS           ; WCS CONTENTS INTO THE CSR.
                  22 D3 1F85                    .BYTE   ^O323 , CLRSS&255       ;register A
                        1F87  5485
                        1F87  5486              OUT     VSHIFT
                  A2 D3 1F87                    .BYTE   ^O323 , VSHIFT&255      ;register A
                        1F89  5487
                        1F89  5488              LXI     H,NOP_INST      ;POINT TO NOP INSTRUCTION
                     21 1F89                    .BYTE   <H*8> ! 1
                00'72' 1F8A                     .BYTE   <NOP_INST-HEAD>&255 , <<NOP_INST-HEAD>&^XFF00>/256
                        1F8C  5489              LXI     D,WCS_VALUE     ;POINT TO WCS VALUE AREA
                     11 1F8C                    .BYTE   <D*8> ! 1
                00'B4' 1F8D                     .BYTE   <WCS_VALUE-HEAD>&255 , <<WCS_VALUE-HEAD>&^XFF00>/256
                        1F8F  5490              CALL    MOVER_3_R       ;LOAD WCS_VALUE WITH NOP INST
                     CD 1F8F                    .BYTE   ^O315
                 0F5F' 1F90                     .WORD   <MOVER_3_R-HEAD>
                        1F92  5491
                        1F92  5492              MVI     B,24            ;LOOP COUNT IN REGISTER B
                  18 06 1F92                    .BYTE   <B*8> ! 6 , 24&255
                        1F94  5493 1$:          LXI     H,WCS_VALUE+2   ;POINT TO END OF WCS_VALUE AREA
                     21 1F94                    .BYTE   <H*8> ! 1
                00'B6' 1F95                     .BYTE   <WCS_VALUE+2-HEAD>&255 , <<WCS_VALUE+2-HEAD>&^XFF00>/256
                        1F97  5494              MVI     C,3             ;BYTE COUNT
                  03 0E 1F97                    .BYTE   <C*8> ! 6 , 3&255
                        1F99  5495
                        1F99  5496              IN      CSR23           ;GET CSR BIT 23
                  87 DB 1F99                    .BYTE   ^O333 , CSR23&255       ;to register A
                        1F9B  5497              RAR                     ;ROTATE CSR 23 INTO CARRY
                     1F 1F9B                    .BYTE   ^O037
                        1F9C  5498 2$:          MOV     A,M             ;MOV WCS_VALUE INTO ACCUMULATOR
                     7E 1F9C                    .BYTE   ^O100 ! <A*8> ! M
                        1F9D  5499              RAL                     ;CARRY GOES IN BIT 0
                     17 1F9D                    .BYTE   ^O027
                        1F9E  5500              MOV     M,A             ;MOV ACCUMULATOR BACK TO MEM
                     77 1F9E                    .BYTE   ^O100 ! <M*8> ! A
```

```
                 1F9F  5501                     DCX     H               ;POINT TO NEXT BYTE
      2B         1F9F                   .BYTE   <H*8> ! 11
                 1FA0  5502                     DCR     C               ;DEC BYTE COUNT
      0D         1FA0                   .BYTE   ^05 ! <8*C>
                 1FA1  5503                     JNZ     2$              ;REPEAT UNTIL 3 BYTES DONE
      C2         1FA1                   .BYTE   ^0302
   1F9C'         1FA2                   .WORD   <2$-HEAD>
                 1FA4  5504
                 1FA4  5505                     OUT     CLRCSR          ;CLEAR CSR INPUT
   38 D3         1FA4                   .BYTE   ^0323 , CLRCSR&255      ;register A
                 1FA6  5506                     JNC     6$              ;DON'T SET INPUT IF CARRY CLEAR
      D2         1FA6                   .BYTE   ^0322
   1FAB'         1FA7                   .WORD   <6$-HEAD>
                 1FA9  5507                     OUT     SETCSR          ;ELSE SET CSR INPUT
   39 D3         1FA9                   .BYTE   ^0323 , SETCSR&255      ;register A
                 1FAB  5508
                 1FA9  5509 6$:                 OUT     SETCSK          ;CLOCK CSR TO SHIFT IT AND ALSO LOAD
   25 D3         1FAB                   .BYTE   ^0323 , SETCSK&255      ;register A
                 1FAD  5510                                             ; NOP INSTRUCTION
                 1FAD  5511                     OUT     CLRCSK          ;CLEAR UPC CLOCK
   24 D3         1FAD                   .BYTE   ^0323 , CLRCSK&255      ;register A
                 1FAF  5512
                 1FAF  5513                     DCR     B               ;DECREMENT SHIFT COUNT
      05         1FAF                   .BYTE   ^05 ! <8*B>
                 1FB0  5514                     JNZ     1$              ;REPEAT UNTIL 24 BITS SHIFTED
      C2         1FB0                   .BYTE   ^0302
   1F94'         1FB1                   .WORD   <1$-HEAD>
                 1FB3  5515
                 1FB3  5516                     OUT     VNORML          ;RETURN TO NORMAL MODE.
   A3 D3         1FB3                   .BYTE   ^0323 , VNORML&255      ;register A
                 1FB5  5517
                 1FB5  5518                     RET
      C9         1FB5                   .BYTE   ^0311
                 1FB6  5519
                 1FB6  5520
                 1FB6  5521 ;*****************************************************************************
                 1FB6  5522 ;
                 1FB6  5523 ;  WCS_WRITE_R  THIS ROUTINE WRITES DATA FROM WCS_VALUE TO WCS AT THE
                 1FB6  5524 ;                LOCATION GIVEN BY WCS_ADDRESS.
                 1FB6  5525 ;
                 1FB6  5526 ;*****************************************************************************
                 1FB6  5527
                 1FB6  5528 WCS_WRITE_R:        CALL    LD_UPC_R        ;LOAD UPC WITH WCS ADDRESS
      CD         1FB6                   .BYTE   ^0315
   1FCC'         1FB7                   .WORD   <LD_UPC_R-HEAD>
                 1FB9  5529
                 1FB9  5530                     LXI     H,WCS_VALUE+2   ;POINT AT THE LOW BYTE.
      21         1FB9                   .BYTE   <H*8> ! 1
00'B6'           1FBA                   .BYTE   <WCS_VALUE+2-HEAD>&255 , <<WCS_VALUE+2-HEAD>&^XFF00>/256
                 1FBC  5531
                 1FBC  5532                     MOV     A,M             ;MOVE THE LOW BYTE TO WCSB0.
      7E         1FBC                   .BYTE   ^0100 ! <A*8> ! M
                 1FBD  5533                     OUT     WCSB0
   08 D3         1FBD                   .BYTE   ^0323 , WCSB0&255       ;register A
                 1FBF  5534                     DCX     H
      2B         1FBF                   .BYTE   <H*8> ! 11
                 1FC0  5535
```

```
                 1FC0  5536                    MOV     A,M             ;MOVE THE NEXT BYTE TO WCSB1.
    7E           1FC0                  .BYTE   ^0100 ! <A*8> ! M
                 1FC1  5537                    OUT     WCSB1
 09 D3           1FC1                  .BYTE   ^0323 , WCSB1&255       ;register A
                 1FC3  5538                    DCX     H
    2B           1FC3                  .BYTE   <H*8> ! 11
                 1FC4  5539
                 1FC4  5540                    MOV     A,M             ;MOVE THE HIGH BYTE TO WCSB2.
    7E           1FC4                  .BYTE   ^0100 ! <A*8> ! M
                 1FC5  5541                    OUT     WCSB2
 0A D3           1FC5                  .BYTE   ^0323 , WCSB2&255       ;register A
                 1FC7  5542
                 1FC7  5543                    OUT     SETWCK          ;WRITE TO WCS.
 37 D3           1FC7                  .BYTE   ^0323 , SETWCK&255      ;register A
                 1FC9  5544                    OUT     CLRWCK
 36 D3           1FC9                  .BYTE   ^0323 , CLRWCK&255      ;register A
                 1FCB  5545
                 1FCB  5546                    RET
    C9           1FCB                  .BYTE   ^0311
                 1FCC  5547
                 1FCC  5548 ;*****************************************************************************
                 1FCC  5549 ;
                 1FCC  5550 ;  LD_UPC_R     THIS ROUTINE LOADS THE UPC WITH WCS_ADDRESS
                 1FCC  5551 ;
                 1FCC  5552 ;*****************************************************************************
                 1FCC  5553
                 1FCC  5554 LD_UPC_R:            LHLD    WCS_ADDRESS     ;UPC DATA
    2A           1FCC                  .BYTE   ^052
 006C'           1FCD                  .WORD   <WCS_ADDRESS-HEAD>
                 1FCF  5555                    SHLD    UPC_VALUE       ;STORE DATA IN UPC_VALUE
    22           1FCF                  .BYTE   ^042
 00C2'           1FD0                  .WORD   <UPC_VALUE-HEAD>
                 1FD2  5556
                 1FD2  5557                    LXI     H,UPC_VALUE+1   ;POINT TO END OF UPC_VALUE AREA
    21           1FD2                  .BYTE   <H*8> ! 1
 00'C3'          1FD3                  .BYTE   <UPC_VALUE+1-HEAD>&255 , <<UPC_VALUE+1-HEAD>&^XFF00>/256
                 1FD5  5558                    MVI     C,2             ;BYTE COUNT
 02 0E           1FD5                  .BYTE   <C*8> ! 6 , 2&255
                 1FD7  5559                    CALL    SHIFTER1_R      ;LEFT JUST. 15 BITS WITH COUNT
    CD           1FD7                  .BYTE   ^0315
 0EC1'           1FD8                  .WORD   <SHIFTER1_R-HEAD>
                 1FDA  5560                                            ; IN REGISTER C
                 1FDA  5561
                 1FDA  5562                    OUT     VSHIFT          ;SET V-BUS SHIFT MODE
 A2 D3           1FDA                  .BYTE   ^0323 , VSHIFT&255      ;register A
                 1FDC  5563                    MVI     B,15            ;SHIFT COUNT=SHIFT 15 BITS
 0F 06           1FDC                  .BYTE   <B*8> ! 6 , 15&255
                 1FDE  5564
                 1FDE  5565 3$:                LXI     H,UPC_VALUE+1   ;POINTER TO END OF UPC DATA
    21           1FDE                  .BYTE   <H*8> ! 1
 00'C3'          1FDF                  .BYTE   <UPC_VALUE+1-HEAD>&255 , <<UPC_VALUE+1-HEAD>&^XFF00>/256
                 1FE1  5566                    MVI     C,2             ;LOAD BYTE COUNT WITH 2
 02 0E           1FE1                  .BYTE   <C*8> ! 6 , 2&255
                 1FE3  5567
                 1FE3  5568 1$:                MOV     A,M             ;MOVE BYTE 1 LEFT
    7E           1FE3                  .BYTE   ^0100 ! <A*8> ! M
                 1FE4  5569                    RAL                     ;CARRY = ROTATE IN BIT NEXT BTYE
```

```
                    17  1FE4                .BYTE   ^027
                        1FE5  5570                  MOV     M,A
                    77  1FE5                .BYTE   ^0100 ! <M*8> ! A
                        1FE6  5571                  DCX     H                   ;MOVE POINTER TO NEXT BYTE
                    28  1FE6                .BYTE   <H*8> ! 11
                        1FE7  5572                  DCR     C                   ;DEC BYTE COUNT
                    0D  1FE7                .BYTE   ^05 ! <8*C>
                        1FE8  5573                  JNZ     1$                  ;REPEAT UNTIL 2 BYTES SHIFTED
                    C2  1FE8                .BYTE   ^0302
                1FE3'   1FE9                .WORD   <1$-HEAD>
                        1FEB  5574
                        1FEB  5575                  OUT     CLRUPC              ;SET UPC SER IN=0 IN CASE CARRY CLR
                A8 D3   1FEB                .BYTE   ^0323 , CLRUPC&255          ;register A
                        1FED  5576                  JNC     2$                  ;SKIP NEXT INSRUCTION IF CARRY CLR
                    D2  1FED                .BYTE   ^0322
                1FF2'   1FEE                .WORD   <2$-HEAD>
                        1FF0  5577                  OUT     SETUPC              ;ELSE SET UPC SER IN=1
                A9 D3   1FF0                .BYTE   ^0323 , SETUPC&255          ;register A
                        1FF2  5578
                        1FF2  5579 2$:              OUT     SETUPK              ;SET UPC CLOCK
                27 D3   1FF2                .BYTE   ^0323 , SETUPK&255          ;register A
                        1FF4  5580                  OUT     CLRUPK              ;CLEAR UPC CLOCK
                26 D3   1FF4                .BYTE   ^0323 , CLRUPK&255          ;register A
                        1FF6  5581
                        1FF6  5582                  DCR     B                   ;DECREMENT SHIFT COUNT
                    05  1FF6                .BYTE   ^05 ! <8*8>
                        1FF7  5583                  JNZ     3$                  ;REPEAT UNTIL 15 BITS SHIFTED
                    C2  1FF7                .BYTE   ^0302
                1FDE'   1FF8                .WORD   <3$-HEAD>
                        1FFA  5584
                        1FFA  5585                  OUT     VNORML              ;SET V-BUS TO NORMAL MODE
                A3 D3   1FFA                .BYTE   ^0323 , VNORML&255          ;register A
                        1FFC  5586
                        1FFC  5587                  RET
                    C9  1FFC                .BYTE   ^0311
                        1FFD  5588
                        1FFD  5589 END_CHECKSUM:
                        1FFD  5590
                        1FFD  5591 ;*****************************************************************************
                        1FFD  5592 ;
                        1FFD  5593 ;       NOTE:  THE LAST ADDRESS OF THIS PROGRAM MUST NOT EXCEED 6FFF.  THE
                        1FFD  5594 ;       8085 BASED MICRODIAGNOSTIC OVERLAYS LOAD STARTING AT 7000 HEX!!!!
                        1FFD  5595 ;
                        1FFD  5596 ;*****************************************************************************
                        1FFD  5597                  .END
```

```
$PSW                            = 00000006
$SP                             = 00000006
A                               = 00000007
ABORT_A                           00000164 R      01
ACK_SAVE                          0000017A R      01
ADDR                            = 00000001
ALLOWP                          = 00000003
APT                               00000038 RG     01
APTFLG                          = 00000004
APT_C_DEF                       = 000000DF
APT_ERROR_NUMBER                  00000012 R      01
APT_HARD_FLG                      0000001D R      01
APT_MAIL_BOX                      00000010 R      01
APT_MEM_SIZE                      0000001C R      01
APT_MESSAGE_CODE                  00000010 R      01
APT_PARAM                       = 00000049
APT_PASS_CNT                      00000018 RG     01
APT_RAM_LD_ADD                    00000009 R      01
APT_SOFT_FLG                      0000001E R      01
APT_SUBTEST                       00000014 R      01
APT_S_DEF                       = 00000020
APT_TEST_NUMBER                   00000016 R      01
ASCII_1                           00000178 R      01
ASCII_2                           0000017C R      01
ASCII_3                           0000017D R      01
ASCII_TO_BIN                      00000E15 R      01
ATTENTION                         00001321 R      01
ATT_UPC_SAVE                      000000CC R      01
AUTO                            = 00000005
AUTO_TEST                         00001C24 R      01
A_ADDRS                           00000163 R      01
B                               = 00000000
BAD_A                             0000017E R      01
BAD_STATUS                      = 00000010
BAD_TAPE_A                        00000199 R      01
BEGIN_CHECKSUM                    0000098A R      01
BEGIN_OF_TEST                     00001881 R      01
BEG_TST                         = 00000080
BELL_C_DEF                      = 000000FB
BELL_S_DEF                      = 00000004
BOARD_BUF                         00000247 R      01
BOARD_NAME                        00000241 R      01
BOARD_NUM                         0000024D R      01
BOARD_PNT                         0000024E R      01
BOARD_TABLE                       00000846 R      01
BOOTEN                          = 00000007
BOOTF                           = 00000001
BYTSUM                          = 00004082
B_ADDRS                           00000240 R      01
C                               = 00000001
CALC_ADD                          00000252 R      01
CC_ADDR                         = 000000FE   G
CHANGE_WCS                        00000C1D R      01
CHECKSUM                          00001BE9 R      01
CHECK_CNTLC                       00001073 R      01
CHECK_ERR_10                      00000C3A R      01
CHECK_ERR_FLG                     00000C3C R      01
CHECK_LS_SETPAR                   0000197D R      01
CHECK_UPC                         00001B73 R      01
CHECK_WCS_LOADED                  00000C2A R      01
CLEARPAR_ERR                    = 00000008
CLEAR_CPU_ATT                     0000012F RG     01
CLEAR_CPU_ATT_R                   00001A61 R      01
CLEAR_DEFAULT                     00000B8F R      01
CLEAR_LS_STAT                     000012D8 R      01
CLEAR_PAR_INT                     000016D9 R      01
CLRACK                          = 000000A9
CLRACL                          = 00000033
CLRATN                          = 000000AB
CLRBSY                          = 0000002E
CLRCLK                          = 00000020
CLRCSK                          = 00000024
CLRCSR                          = 00000038
CLRDCL                          = 00000031
CLRHLT                          = 000000AF
CLRLIT                          = 0000003C
CLRMCI                          = 00000029
CLRMCK                          = 0000002B
CLRPFI                          = 000000AD
CLRSS                           = 00000022
CLRTIM                          = 00000UA5
CLRTMR                          = 0000002C
CLRUPC                          = 000000A8
CLRUPK                          = 00000026
CLRWCK                          = 00000036
CNT                             = 00000887
CNTA                            = 00000FA7
CNTLC_TYPED                       00000039 RG     01
CNTLC_VEC                         00001063 R      01
CNTLP_VEC                         00001070 R      01
CNT_PAR                           00000254 R      01
COMMAND0                          000000AF R      01
COMMAND1                          000000B0 R      01
COMMAND2                          000000B1 R      01
COMMAND3                          000000B2 R      01
COMPARE                           000000CF RG     01
COMPARE_R                         00000F55 R      01
CONACK                          = 00000020
CONATT                          = 00000010
CONHLT                          = 00000002
CONSOLE_TEST                      00000255 R      01
CONTINUE                          0000003A RG     01
CONT_COMMAND                      00000B46 R      01
CONT_DONE                         00C00256 R      01
CONT_SAVE_CSR                     000000BE R      01
CONT_SAVE_UPC                     000000C8 R      01
CON_ADD_DAT                       0000003B RG     01
CON_BEGIN_TEST                    00000123 RG     01
CON_BEGIN_TEST_R                  00001EC9 R      01
CON_ERROR                         00000120 RG     01
CON_ERROR_R                       00001F14 R      01
CON_ERR_NUM                       0000003F RG     01
CON_EXP_ADD                       00000044 RG     01
CON_EXP_DAT                       00000040 RG     01
```

Symbol table

```
CON_LOOP                =  00000004
CON_MOD_DAT                00000046 RG     01
CON_MSK_DAT                0000004B RG     01
CON_NAME_ADD            =  00007005
CON_RAM                 =  00000080
CON_REC_ADD                00000053 RG     01
CON_REC_DAT                0000004F RG     01
CON_VER_ADD             =  00007003
CPATTN                  =  00000083
CPUACK                  =  00000082
CPU_A                      00000257 R      01
CR                      =  0000000D   G
CRD_FLAGS               =  000040FD
CRD_OPT_A                  00000825 R      01
CRLF                       000000DE RG     01
CRLF_A                     00000258 R      01
CRLF_R                     0000101A R      01
CSR07                   =  00000085
CSR15                   =  00000086
CSR23                   =  00000087
CSR_CNT                    0000025D R      01
CSR_SHIFT                  000000B7 RG     01
CSR_VALUE                  000000B8 RG     01
CTLDIS                  =  000040FE
C_EXECUTE                  00000250 R      01
C_RUNNING                  00000251 R      01
D                       =  00000002
DATA0                      000000A9 RG     01
DATA1                      000000AA RG     01
DATA2                      000000AB RG     01
DATA3                      000000AC RG     01
DATA_SHIFT                 000000A8 RG     01
DATA_XFER                  00001996 R      01
DATA_XFER_FLG              0000025E R      01
DCLO                    =  00000002
DECR_WORD                  00000135 RG     01
DECR_WORD_R                000012BC R      01
DIAG_BOARD                 00000A0F R      01
DIAG_COMMAND               00000998 R      01
DIAG_CONTIN                00000B0E R      01
DIAG_PASS                  00000B14 R      01
DIAG_SECTION               00000A5D R      01
DIAG_TAB                   00000960 R      01
DIAG_TEST                  00000ACA R      01
DIR_VECTOR              =  0000411A
DISMEM                  =  000000A7
DISPAR                  =  000000A1
DOLOOP                     00000055 RG     01
DO_A                    =  0000002E
DO_COMMANDS                0000137F R      01
E                       =  00000003
EMEMREF                 =  00000040
ENBMEM                  =  000000A6
ENBPAR                  =  000000A0
END_A                      00000262 R      01
END_CHECKSUM               00001FFD R      01
ENKCC_A                    000002C9 R      01
ENKCE_A                    000002CE R      01
ENKCF_A                    000002D8 R      01
ENKCG_A                    000002DD R      01
ENKCH_A                    000002D3 R      01
ENK_END                    0000025F R      01
EOP                        0000183E R      01
EOS                     =  00000040
EOS_FLG                    00000056 RG     01
ERROR                      000002E2 R      01
ERROR_A                    000002E3 R      01
ERROR_CON                  00000057 RG     01
ERROR_ROUTINE              00000C4A R      01
ERROR_ROUTINE_NO_RET       00000C45 R      01
ERR_MENU                   00000D21 R      01
EXECUTE                    000002EA R      01
EXEC_CONTESTS           =  00007000
EXEC_SUB                   00000B7D R      01
EXEC_TEST0                 00001BA1 R      01
EXT_A                      000002EB R      01
FAIL_A                     000002EF R      01
FAST_WRITE                 000002FC R      01
FILE_LOADED                000002FD R      01
FLAGS                      0000005A RG     01
FLAG_LOC                =  00000048
FOUR_BYTES                 00001A29 R      01
FPA_A                      000002FE R      01
FPA_PRES                =  00000010
FULL_ERROR                 0000154A R      01
F_WORD                     00000058 RG     01
GCVECT                  =  00004129   G
GLVECT                  =  00004123
GSVECT                  =  00004145
H                       =  00000004
HALT_C_DEF              =  000000F7
HALT_S_DEF              =  00000008
HAVE_CNT                   0000033D R      01
HEAD                    =  00000000
HEADER_A                   00000302 R      01
HEX_0D                  =  0000000D
HEX_1                   =  00000001
HEX_10                  =  00000010
HEX_1F                  =  0000001F
HEX_20                  =  00000020
HEX_21                  =  00000021
HEX_2D                  =  0000002D
HEX_3                   =  00000003   G
HEX_30                  =  00000030
HEX_3A                  =  0000003A
HEX_3F                  =  0000003F
HEX_4                   =  00000004
HEX_40                  =  00000040
HEX_41                  =  00000041
HEX_50                  =  00000050
HEX_7                   =  00000007
HEX_7F                  =  0000007F
HEX_8                   =  00000008
HEX_80                  =  00000080
```

HEX_BUF		0000005B	RG	01	MACSTK4	=	00000878				
HEX_FD	=	000000FD			MACSTK5	=	00000879				
HEX_FF	=	000000FF			MACSTK6	=	0000087C				
HLTFLG	=	00000002			MACSTK7	=	0000087B				
IDC_A		0000033E	R	01	MAIN_TAB		00000931	R	01		
IDC_PRES	=	00000080			MAKE_A		000011AB	R	01		
INC_TEMPW_5		000012A5	R	01	MAKE_B		00001195	R	01		
INC_WORD		00000108	RG	01	MAKE_XD		0000010B	RG	01		
INC_WORD_R		000012B1	R	01	MAKE_XD_R		00001183	R	01		
INITH	=	00000035			MAX_SBE		0000034A	R	01		
INITL	=	00000034			MCPU_A		00000140	RG	01		
INPUT_CHAR_IF		0000105C	R	01	MEMORY_A		0000034D	R	01		
INPUT_LINE		0000102E	R	01	MEM_REQ		00000354	R	01		
INPUT_NUM		00000676	R	01	MENU		00001616	R	01		
INPUT_NUM_CHECK		00000C37	R	01	MENU_A		00000355	R	01		
INPUT_NUM_IF		00000D9B	R	01	MFPA_A		0000014A	R	01		
INSTR_LEN		00000344	R	01	MICFLAG		0000002E	R	01		
INSTR_PNT		00000345	R	01	MICRO_STEP		0000038A	R	01		
INSTR_TABLE		0000067A	R	01	MICRO_STEPER		00001257	R	01		
INT_CLEAR_PAR		00000BBB	R	01	MICRO_STEP_CPU		00000138	RG	01		
INVTIM	=	00000008			MICRO_STEP_CPU_R		000012A0	R	01		
ITCSR	=	0000001D			MIDC_A		00000154	R	01		
ITDB	=	0000001C			MINUS		0000038B	R	01		
L	=	00000005			MINUTES_A		0000038C	R	01		
LD_UPC		00000117	RG	01	MISC2	=	00000001				
LD_UPC_R		00001FCC	R	01	MMCT_A		00000145	R	01		
LEAVE_DO		00001B71	R	01	MMEM_A		0000014F	R	01		
LF	=	0000000A	G		MM_OR_LS	=	00000001				
LITERAL	=	00000001			MM_SIZE	=	00000008				
LOADER		00000B39	R	01	MOD_ERR		00000395	R	01		
LOAD_CSR		000000F0	RG	01	MOD_NOT_PRES_A		00000396	R	01		
LOAD_CSR_R		00000EE7	R	01	MOD_PRES_A		000003A6	R	01		
LOAD_SECTION		00001A8B	R	01	MOD_PRES_FLG	=	00000001				
LOAD_UPC		000000F3	RG	01	MORE_DATA		000003AF	R	01		
LOAD_UPC_R		00000F19	R	01	MOVER		000000D2	RG	01		
LOOPER		000017DF	R	01	MOVER_3		0000011A	RG	01		
LOOPING	=	00000002			MOVER_3_R		00000F5F	R	01		
LOOP_CNT_ADD	=	00000047			MOVER_4		00000111	RG	01		
LOOP_COM_C_DEF	=	000000BF			MOVER_4_R		00000F73	R	01		
LOOP_COM_S_DEF	=	00000040			MOVER_R		00000F75	R	01		
LOOP_COUNTER		00000347	R	01	MOV_DATA0_HEXBUF		00000F6D	R	01		
LOOP_C_DEF	=	000000BE			MOV_FROM_DATA0		00000F70	R	01		
LOOP_S_DEF	=	00000001			MOV_INPUT_DATA0		00000F64	R	01		
LS_5	=	00000005			MOV_LOAD_SECTION		00001A80	R	01		
LS_6	=	00000006			MR80_A		0000015E	R	01		
LS_7	=	00000007			MRL02_A		00000159	R	01		
LS_F	=	0000000F			MSK_BYTE		00C01F78	R	01		
LS_PARERR_ACT		00000349	R	01	MSK_ERROR		0000011D	RG	01		
LS_SETPARERR		00000348	R	01	MSK_ERROR_R		00001F31	R	01		
LVL	=	00000000			MWCS_A		0000013B	RG	01		
M	=	00000006			NA	=	00000080				
M1STK1	=	00000FA6			NAME_PNT		000003BA	R	01		
M1STK2	=	00000FA4			NAME_TABLE		0000076F	R	01		
M1STK3	=	00000FA7			NA_A		000003B0	R	01		
MACSTK1	=	00000887			NA_EXP_REC		0000005F	RG	01		
MACSTK2	=	00000885			NA_OTHER		00000060	RG	01		
MACSTK3	=	00000881			NEG_CNT	=	00000080				

NER_C_DEF	=	000000BD			PRINT_MOD_PRES_IDC		00001742	R	01
NER_S_DEF	=	00000002			PRINT_PASS		000003ED	R	01
NEWSUM		000003BC	R	01	PRINT_PNT		000003EE	R	01
NOADD_FLG		000003BD	R	01	PRINT_PROMPT		00000FD6	R	01
NOCHK	=	000040FF			PRINT_R		00001027	R	01
NOINIT		000003BE	R	01	PRINT_SINGLE		000016EA	R	01
NOP_CSR		000000E4	RG	01	PRINT_START		000003F0	R	01
NOP_CSR_R		00000ECA	R	01	PRINT_STRING		0000010E	RG	01
NOP_INST		00000072	RG	01	PRINT_STRING_APT		00001014	R	01
NOT_FOUND		000003C2	R	01	PRINT_STRING_R		0000100E	R	01
NO_RETURN		000003BF	R	01	PRI_STACK	=	00004080		
NO_SAVE_UPC_CSR		000003C0	R	01	PRT_MOD_A		0000013B	R	01
NO_SPACE		000003C1	R	01	PR_HEX_DIGIT		00000FC9	R	01
NUM_SHIFT		00000675	R	01	PWRFAIL	=	00000004		
OFLAG	=	000040BB			R80_PRES	=	00000080		
OKV15	=	00000007			RAM_LOAD_ADD	=	00007000		
OLDSUM		000003CF	R	01	READ	=	00000080		
OLD_TEMP_BUF		000003C3	R	01	READJ1	=	00000003		
OLD_TTBUF		000003CD	R	01	READJ2	=	00000004		
OPTIONAL_TST	=	00000040			READ_DATA_32		000000FC	RG	01
OR_A		000003D0	R	01	READ_DATA_32_R		000011B4	R	01
OS_ADD	=	0000007C	G		READ_LS		00000102	RG	01
OVERLAY_RAM		00000003	RG	01	READ_LS_7		00001165	R	01
OVERLAY_WCS		00000006	RG	01	READ_LS_R		00001167	R	01
PACK_ERROR		000003D5	R	01	READ_WCS		000000E7	RG	01
PACK_WARNING		0000165B	R	01	READ_WCS_DATA		00001A4E	R	01
PAR		00001D6E	R	01	READ_WCS_R		00000E6D	R	01
PARDONE		000003D9	R	01	REMOTE	=	00000006		
PARITY_CALC		00000BC8	R	01	RESET_TIMOUT		00001932	R	01
PARITY_ERR		0000193A	R	01	RESTART		000003F1	R	01
PARITY_JUMP		00000061	RG	01	RESTART_CPU		0000110E	R	01
PARITY_ON		00000064	RG	01	RESTORE_WR		00000C0A	R	01
PARSER		00001DAE	R	01	RETRY_IDC2		000003F2	R	01
PARSE_PNT		000003DA	R	01	RETURN_I		00000766	R	01
PARSE_TABLE		000003DC	R	01	RETURN_N		000007ED	R	01
PAR_ON		000003D6	R	01	RETURN_T		00000820	R	01
PAR_TEMP_BUF		00001D6F	R	01	RET_VECTOR	=	0000414D		
PAR_VECT		000003D	R	01	RL02_A		000003FE	R	01
PASS_A		000003	R	01	ROM_X_ADDRS	=	0000000B		
PASS_CNT		000003	R	01	RSVECT	=	0000413D		
PERFORM_CSR		00000	RG	01	RUNNING		00000403	R	01
PERFORM_CSR_R		00001	R	01	SAVED_CSR		000000BB	R	01
POWER_BRANCH		00000	RG	01	SAVED_UPC		000000C4	R	01
POWER_REC		00001C0	R	01	SAVE_WR		00000C11	R	01
POWER_SEQ	=	00000018			SCVECT	=	00004138		
PRINT		000000D5	RG		SECTION_B_NUM		00000404	R	01
PRINTER		00001007	R	01	SECTION_FLAG		00C008C9	R	01
PRINT_CNT		000003EC	R	01	SECTION_NAME		000008CA	R	01
PRINT_CSR_VAL		00000D8C	R	01	SEC_LEN	=	00000006		
PRINT_FLG		00000F7E	R	01	SEC_PARMB_ADD	=	00004111		
PRINT_HEX		000000D8	RG	01	SEC_PARMB_DEF	=	00004150		
PRINT_HEX_1		00000114	RG	01	SEC_PARMB_DST	=	00004115		
PRINT_HEX_1_R		00000F94	R	01	SEC_PARMB_ERR	=	00004116		
PRINT_HEX_2		00000F8F	R	01	SEC_PARMB_EXT	=	0000410D		
PRINT_HEX_4		00000F8A	R	01	SEC_PARMB_NAM	=	00004107		
PRINT_HEX_R		00000F96	R	01	SEC_PARMB_UNIT	=	00004106		
PRINT_MOD_PRES		00001723	R	01	SEC_TABLE		0000085F	R	01

```
SER_C_DEF                       = 000000EF
SER_S_DEF                       = 00000010
SETACK                          = 000000A8
SETACK_FLG                        00000405 R      01
SETACL                          = 00000032
SETATN                          = 000000AA
SETBSY                          = 0000002F
SETCLK                          = 00000021
SETCSK                          = 00000025
SETCSR                          = 00000039
SETDCL                          = 00000030
SETHLT                          = 000000AE
SETLIT                          = 0000003D
SETMCI                          = 00000028
SETMCK                          = 0000002A
SETPAR_ERR                      = 00000002
SETPFI                          = 000000AC
SETSS                           = 00000023
SETTIM                          = 000000A4
SETTMR                          = 0000002D
SETUPC                          = 000000A9
SETUPK                          = 00000027
SETWCK                          = 00000037
SET_ALL_TESTS                     00001EBA R      01
SET_FOR_CONT                      0000012C RG     01
SET_FOR_CONT_R                    0000107B R      01
SET_LS_STAT                       000012CE R      01
SET_PARDONE                       00001C06 R      01
SET_PAR_INT                       000016C7 R      01
SET_SIGNALS                       00001760 R      01
SHIFTER                           000000E1 RG     01
SHIFTER1                          00000132 RG     01
SHIFTER1_R                        00000EC1 R      01
SHIFTER_R                         00000EB8 R      01
SHIFT_L_2901                      000000F6 RG     01
SHIFT_L_2901_R                    00001249 R      01
SHIFT_PN                          00000068 RG     01
SHIFT_R_2901                      000000F9 RG     01
SHIFT_R_2901_R                    00001250 R      01
SIGNAL                          = 00000001
SINGLE_ERR                      = 00000004
SKIP_CNT                          00000406 R      01
SKIP_TEST                         0000006A RG     01
SKIP_TO_SPEC                      00000E33 R      01
SLVEC1                          = 00004133
SLVECT                          = 0000412E
SPACE                             000000DB RG     01
SPACE_A                         = 00000020
SPACE_R                           00001021 R      01
SPBUFF                          = 00004086
SPECIAL_C_DEF                   = 0000007F
SPECIAL_S_DEF                   = 00000080
STARTING_UPC                      00000407 R      01
START_CPU                         000010FF R      01
START_CPU_COM                     00001123 R      01
START_UP                          00001CD2 R      01
STATUS                          = 00000040
STEP_CNT                          00000409 R      01
STNDRD_CON                      = 00000000
STNDRD_CON_COM                    0000098A R      01
STNDRD_CON_COM_SUC                00000991 R      01
STOP_CPU                          000010C9 R      01
SUB_ATTENTION                     00001806 R      01
SYM                             = 00000887
SYMA                            = 00000FA6
SYNTAX_ERROR                      00001BE0 R      01
S_READ_LS                         00000667 R      01
S_WRITE_LS                        00000672 R      01
TABLE_MODE                        0000040B R      01
TABLE_PNT                         0000040C R      01
TEMP_BUF                          0000040E R      01
TEMP_BYTE                         00000418 R      01
TEMP_FLAG                         0000041F R      01
TEMP_SECTION                      00000420 R      01
TEMP_WORD                         00000419 R      01
TEMP_WORD1                        0000041B R      01
TEMP_WORD2                        0000041D R      01
TEST_NUM                          00000426 R      01
TEST_NUM1                         00000427 R      01
TEST_NUM2                         00000428 R      01
TEST_START_A                      00000429 R      01
TEST_ZERO                         00000447 R      01
TIMER_VEC                         00001C23 R      01
TIME_PNT                          00000448 R      01
TIME_TABLE                        00000802 R      01
TIMOUT                            00001907 R      01
TIMOUT_CNT                        0000044A R      01
TRACE                             0000044C R      01
TTBUF                             000008D1 R      01
TTBUF_CNT                         000008D0 R      01
TTBUF_PNT                         0000044D R      01
TTCR                            = 0000004B
TTDB                            = 00000048
TTMODE                          = 0000004A
TTSR                            = 00000049
TU58_DRIVER                     = 00004117
TUCR                            = 00000047
TUDB                            = 00000044
TUMODE                          = 00000046
TUSR                            = 00000045
TYPE_CHAR                         00001027 R      01
UBE_PRES                        = 00000040
UBS_BUSY                        = 00000040
UBS_DC_LO                       = 00C00020
UBS_INIT                        = 00000010
UPC14                           = 00000084
UPC14A                          = 00000000
UPC_COMPARE                       000000CA R      01
UPC_SHIFT                         000000C1 R      01
UPC_SUB                           000000C6 R      01
UPC_VALUE                         000000C2 RG     01
VERIFY_SEQ                        000018A1 R      01
VERIFY_WRITER                     00001B16 R      01
VERSION                           0000044F R      01
```

```
VERSION_1                       00000631 R      01
VNORML                        = 000000A3
VSHIFT                        = 000000A2
WARM_RESTART                    000004B3 R      01
WARN_MESA                       0000059A R      01
WARN_MESB                       000005C8 R      01
WARN_MESS                       000004B4 R      01
WCSB0                         = 00000008
WCSB1                         = 00000009
WCSB2                         = 0000000A
WCSERR                        = 00000001
WCS_A                           00000585 R      01
WCS_ADDRESS                     0000006C RG     01
WCS_ADDR_DAT                  = 00000044
WCS_BAD_PARITY                  0000006E RG     01
WCS_CALC                        0000058A R      01
WCS_CL_FIFO                   = 00000062
WCS_DAT_32_ADD                = 00000600
WCS_DE_DBUF                   = 00000061
WCS_DE_ICSR                   = 0000005F
WCS_DE_IDAR                   = 00000060
WCS_DE_MCT                    = 00000050
WCS_DE_MM                     = 00000054
WCS_DE_TB                     = 00000052
WCS_DE_UBS                    = 00000058
WCS_ERROR                       00001442 R      01
WCS_ERR_NUM                   = 00000041
WCS_EXP_DAT                   = 00000042
WCS_EX_DBUF                   = 0000005C
WCS_EX_ICSR                   = 0000005A
WCS_EX_IDAR                   = 0000005B
WCS_EX_MCT                    = 00000051
WCS_EX_MM                     = 00000055
WCS_EX_PATT                   = 0000005D
WCS_EX_POSIT                  = 0000005E
WCS_EX_TB                     = 00000053
WCS_EX_UBS                    = 00000059
WCS_FILE_NAME                 = 00000607
WCS_LOADED                      0000058B R      01
WCS_LOAD_ADD                  = 00000000
WCS_MOD_DAT                   = 00000046
WCS_MSK_DAT                   = 00000045
WCS_PNT                         0000058C R      01
WCS_READ                        00000126 RG     01
WCS_READ_R                      00001F80 R      01
WCS_REC_DAT                   = 00000043
WCS_REST_WR                   = 00000057
WCS_SAVE_WR                   = 00000056
WCS_SEL_FIFOA                 = 00000063
WCS_SEL_FIFOB                 = 00000064
WCS_SHIFT                       0000006B RG     01
WCS_VALUE                       000000B4 RG     01
WCS_VAL_SHIFT                   000000B3 RG     01
WCS_VER_ADD                   = 00000606
WCS_WRITE                       00000129 RG     01
WCS_WRITE_R                     00001FB6 R      01
WRITE                         = 000000CC
WRITE_DATA_32                   000000FF RG     01
WRITE_DATA_32_R                 000011D2 R      01
WRITE_LS                        00000105 RG     01
WRITE_LS_R                      00001146 R      01
WRITE_LS_STAT                   000012DF R      01
WRITE_TAB                       00000921 R      01
WRITE_TAB_PNT                   0000006F RG     01
WRITE_WCS                       000000EA RG     01
WRITE_WCS_R                     00000E8A R      01
WRONG_LAB                     = 00000020
XD_ADDRS                        00000071 RG     01
XFER                          = 00000020
XFER_CNT                        0000058D R      01
XFER_DEST                       0000058F R      01
XFER_MM                         00000592 R      01
XFER_PNT                        00000593 R      01
X_CLR_CPU_ATT                   0000066A R      01
X_CLR_PAGE                      0000007D RG     01
X_CLR_WRO                       000000A1 RG     01
X_COM_WRO                       000000A5 RG     01
X_MOV_WRWR                      0000009E RG     01
X_READ_LS                       00000663 R      01
X_ROT_L                         00000079 RG     01
X_SET_PAGE                      00000081 RG     01
X_SHIFT_L                       00000075 RG     01
X_SHIFT_R                       00000085 RG     01
X_WRITE_LS                      0000066E R      01
Y                             = 00000055
YBUSRD                        = 000000EC
ZERO_BYTE                       00000595 R      01
ZERO_WORD                       00000596 R      01
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
MICMON	00001FFD (8189.)	01 (1.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	104	00:00:00.50	00:00:03.94
Command processing	119	00:00:00.50	00:00:03.67
Pass 1	3277	00:01:01.88	00:06:38.08
Symbol table sort	13	00:00:01.33	00:00:04.92
Pass 2	3729	00:00:25.50	00:01:47.52
Symbol table output	92	00:00:00.60	00:00:03.89
Psect synopsis output	13	00:00:00.01	00:00:00.07
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	7354	00:01:30.34	00:08:42.09

The working set limit was 500 pages.
620993 bytes (1213 pages) of virtual memory were used to buffer the intermediate code.
There were 40 pages of symbol table space allocated to hold 658 non-local and 219 local symbols.
7627 source lines were read in Pass 1, producing 65 object records in Pass 2.
151 pages of virtual memory were used to define 150 macros.

Macro library statistics

Macro library name	Macros defined
SYS$SYSROOT:[SYSLIB]STARLET.MLB;1	0

0 GETS were required to define 0 macros.

There were no errors, warnings or information messages.

MACRO/SHOW=(BIN)/LIST=ENKAB.LIS/OBJECT=ENKAB.OBJ 8085MAC.MAR+MACDEF.MAC+ENKAB.MAC

Fiche	Frame	Sequence		
1	B1	1	Documentation	
1	E1	4	Map	
1	H1	7	.MAIN.	14-JUN-1984
1	D15	185	Symbol table	14-JUN-1984
1	J15	191	Psect synopsis	14-JUN-1984

B	1	Documentation
C	1	Documentation
D	1	Documentation
E	1	Map
F	1	Map
G	1	Map
H	1	.MAIN.
I	1	.MAIN.
J	1	.MAIN.
K	1	.MAIN.
L	1	.MAIN.
M	1	.MAIN.
N	1	.MAIN.
B	2	.MAIN.
C	2	.MAIN.
D	2	.MAIN.
E	2	.MAIN.
F	2	.MAIN.
G	2	.MAIN.
H	2	.MAIN.
I	2	.MAIN.
J	2	.MAIN.
K	2	.MAIN.
L	2	.MAIN.
M	2	.MAIN.
N	2	.MAIN.
B	3	.MAIN.
C	3	.MAIN.
D	3	.MAIN.
E	3	.MAIN.
F	3	.MAIN.
G	3	.MAIN.
H	3	.MAIN.
I	3	.MAIN.
J	3	.MAIN.
K	3	.MAIN.
L	3	.MAIN.
M	3	.MAIN.
N	3	.MAIN.
B	4	.MAIN.
C	4	.MAIN.
D	4	.MAIN.
E	4	.MAIN.
F	4	.MAIN.
G	4	.MAIN.
H	4	.MAIN.
I	4	.MAIN.
J	4	.MAIN.
K	4	.MAIN.
L	4	.MAIN.
M	4	.MAIN.
N	4	.MAIN.
B	5	.MAIN.
C	5	.MAIN.
D	5	.MAIN.
E	5	.MAIN.
F	5	.MAIN.
G	5	.MAIN.
H	5	.MAIN.
I	5	.MAIN.
J	5	.MAIN.
K	5	.MAIN.
L	5	.MAIN.
M	5	.MAIN.
N	5	.MAIN.
B	6	.MAIN.
C	6	.MAIN.
D	6	.MAIN.
E	6	.MAIN.
F	6	.MAIN.
G	6	.MAIN.
H	6	.MAIN.
I	6	.MAIN.
J	6	.MAIN.
K	6	.MAIN.
L	6	.MAIN.
M	6	.MAIN.
N	6	.MAIN.
B	7	.MAIN.
C	7	.MAIN.
D	7	.MAIN.
E	7	.MAIN.
F	7	.MAIN.
G	7	.MAIN.
H	7	.MAIN.
I	7	.MAIN.
J	7	.MAIN.
K	7	.MAIN.
L	7	.MAIN.
M	7	.MAIN.
N	7	.MAIN.
B	8	.MAIN.
C	8	.MAIN.
D	8	.MAIN.
E	8	.MAIN.
F	8	.MAIN.
G	8	.MAIN.
H	8	.MAIN.
I	8	.MAIN.
J	8	.MAIN.
K	8	.MAIN.
L	8	.MAIN.
M	8	.MAIN.
N	8	.MAIN.
B	9	.MAIN.
C	9	.MAIN.
D	9	.MAIN.
E	9	.MAIN.
F	9	.MAIN.
G	9	.MAIN.
H	9	.MAIN.
I	9	.MAIN.
J	9	.MAIN.
K	9	.MAIN.
L	9	.MAIN.
M	9	.MAIN.
N	9	.MAIN.
B	10	.MAIN.
C	10	.MAIN.
D	10	.MAIN.
E	10	.MAIN.
F	10	.MAIN.
G	10	.MAIN.
H	10	.MAIN.
I	10	.MAIN.
J	10	.MAIN.
K	10	.MAIN.
L	10	.MAIN.
M	10	.MAIN.
N	10	.MAIN.
B	11	.MAIN.
C	11	.MAIN.
D	11	.MAIN.
E	11	.MAIN.
F	11	.MAIN.
G	11	.MAIN.
H	11	.MAIN.
I	11	.MAIN.
J	11	.MAIN.
K	11	.MAIN.
L	11	.MAIN.
M	11	.MAIN.
N	11	.MAIN.
B	12	.MAIN.
C	12	.MAIN.
D	12	.MAIN.
E	12	.MAIN.
F	12	.MAIN.
G	12	.MAIN.
H	12	.MAIN.
I	12	.MAIN.
J	12	.MAIN.
K	12	.MAIN.
L	12	.MAIN.
M	12	.MAIN.
N	12	.MAIN.
B	13	.MAIN.
C	13	.MAIN.
D	13	.MAIN.
E	13	.MAIN.
F	13	.MAIN.
G	13	.MAIN.
H	13	.MAIN.
I	13	.MAIN.
J	13	.MAIN.
K	13	.MAIN.
L	13	.MAIN.
M	13	.MAIN.
N	13	.MAIN.
B	14	.MAIN.
C	14	.MAIN.
D	14	.MAIN.
E	14	.MAIN.
F	14	.MAIN.
G	14	.MAIN.
H	14	.MAIN.
I	14	.MAIN.
J	14	.MAIN.
K	14	.MAIN.
L	14	.MAIN.

```
M 14  .MAIN.
N 14  .MAIN.
B 15  .MAIN.
C 15  .MAIN.
D 15  Symbol table
E 15  Symbol table
F 15  Symbol table
G 15  Symbol table
H 15  Symbol table
I 15  Symbol table
  15  Psect synopsis
K 15  Directory
```