'MENSCH COMPUTER ROM SOFTWARE'
'IRQVCTRS.ASM--IRQ VECTOR EQUATES FOR WDC65C265'

2500 A.D. 65816 Macro Assembler - Version 5.01g

Input Filename : irom2.asm
Output Filename : irom2.obj
Listing Has Been Relocated

```
1                       ; FILE: irom2.asm
2                       ; DATE: 12-02-94
3
4
5                           .TTL  'MENSCH COMPUTER ROM SOFTWARE'
6
7
8                           .PW 122
9  00:0000                  CHIP 65816
10
11                          .SYMBOLS
12                          .LINKLIST
13                          .SPACES OFF
14
15
16                      ;IROM  equ 0  ;FOR EPROM ASSEMBLY
17          00:0001     IROM  EQU 1  ;FOR ROM ASSEMBLY
18
19
20 00:0000                      INCLUDE COPYRITE.ASM
21
       ;*******************************************************************************
22                      ;*                                                             *
23                      ;* (C) Copyright 1994, 1995   No part of this program may be reproduced        *
24                      ;* in any form without the express written approval of the Western        *
25                      ;* Design Center.   [L. A. Hittel author]                      *
26                      ;*                                                             *
27
       ;*******************************************************************************
28
29
30 00:0000              INCLUDE EQU265.H
31                       .STTL 'EQU265.ASM--GLOBAL EQUATES FOR WDC65C265'
32                       .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'EQU265.ASM--GLOBAL EQUATES FOR WDC65C265'

```
33                      ;08-06-1994
34
35      00:0708         POWER_DOWN_COUNT EQU 1800     ;TIME OUT IN SECONDS
36
37      00:000F         LOWNIB  EQU $0F
38      00:00F0         HINIB   EQU $F0
39
40              ;CR         EQU $0D     ;CARRIAGE RETURN
41              ;LF         EQU $0A     ;LINE FEED
42              ;ESC        EQU $1B     ;ESCAPE
43              ;BKSP       EQU $08     ;BACKSPACE
44              ;
45              ;ACK        EQU $06
46              ;BELL_KEY   EQU $07
47              ;BKSP_KEY   EQU $08
48              ;TAB_KEY    EQU $09
49              ;L_FEED     EQU $0A
50              ;FORM_FEED  EQU $0C
51              ;C_RETURN   EQU $0D
52              ;
53              ;XON        EQU $11     ;DC1/^Q
54              ;XOFF       EQU $13     ;DC3/^S
55              ;
56              ;NAK        EQU $15
57              ;
58              ;ESC_KEY    EQU $1B
59              ;
60              ;ALT_D      EQU $84 ;Delete Line
61              ;ALT_K      EQU $8B ;Clear to end of Line
62              ;
63              ;INS_KEY    EQU $9B ;Toggles Insert Mode ON/OFF
64              ;DEL_KEY    EQU $9C ;Deletes Character
65              ;
66              ;HOME_KEY   EQU $9D ;Move to start of line.. Then to top of page
67              ;END_KEY    EQU $9E ;Move to end of line.. Then to bottom of page
68              ;
69              ;PAGE_UP    EQU $9F
70              ;PAGE_DWN   EQU $A0
71              ;
72              ;UP_ARROW   EQU $A1
73              ;DWN_ARROW  EQU $A2
74              ;LFT_ARROW  EQU $A3
75              ;RT_ARROW   EQU $A4
76              ;
77              ;
```

```
78              ;F1         EQU $A5 ;Edit Mode
79              ;F2         EQU $A6 ;Graphics Editor
80              ;F3         EQU $A7 ;Mensch Works Filer
81              ;F4         EQU $A8 ;Scroll Toggle ON/OFF
82              ;
83              ;F5         EQU $A9
84              ;F6         EQU $AA
85              ;F7         EQU $AB
86              ;F8         EQU $AC
87              ;
88              ;F9         EQU $AD
89              ;F10        EQU $AE
```

'MENSCH COMPUTER ROM SOFTWARE'
'EQU265.ASM--GLOBAL EQUATES FOR WDC65C265'

```
90                ;F11         EQU $AF
91                ;F12         EQU $B0
92                ;
93                ;PrtScrn     EQU $F0
94                ;PrtScrnShft EQU $F1
95                ;
96
97
98
99       00:0001          Bit0   EQU 1
100      00:0002          Bit1   EQU 2
101      00:0004          Bit2   EQU 4
102      00:0008          Bit3   EQU 8
103      00:0010          Bit4   EQU 16
104      00:0020          Bit5   EQU 32
105      00:0040          Bit6   EQU 64
106      00:0080          Bit7   EQU 128
107
108               ********************************************
109               *                                          *
110               *   THIS IS THE INTERNAL RAM MAP FOR       *
111               *     ALL WDC W65C265 PROGRAMS             *
112               *                                          *
113               ********************************************
114
115               ;576 BYTES
116
117               ;$00:0000 - $00:01FF
118               ;$00:DF80 - $00:DFBF
119
120
121               ********************************************
122               *                                          *
123               *   THIS IS THE GLOBAL EQUATES FOR         *
124               *     ALL WDC W65C265 PROGRAMS             *
125               *                                          *
126               ********************************************
127
128      00:0010          X8     EQU  $10
129      00:0020          M8     EQU  $20
130
131      00:FD00          PD0    EQU  $FD00  PORT 0 DATA REGISTER
132      00:FD01          PD1    EQU  $FD01  PORT 1 DATA REGISTER
133      00:FD02          PD2    EQU  $FD02  PORT 2 DATA REGISTER
134      00:FD03          PD3    EQU  $FD03  PORT 3 DATA REGISTER
```

```
00:DF04          PDD0    EQU  $DF04  PORT 0 DATA DIRECTION REGISTER
00:DF05          PDD1    EQU  $DF05  PORT 1 DATA DIRECTION REGISTER
00:DF06          PDD2    EQU  $DF06  PORT 2 DATA DIRECTION REGISTER
00:DF07          PDD3    EQU  $DF07  PORT 3 DATA DIRECTION REGISTER


00:DF08          RVD08   EQU  $DF08  RESERVED $FD08-FD1F

00:DF10          VCS0 EQU $00DF10  ;LCD CONTROL VIA

00:DF10          VIA_PB       EQU VCS0      ;PB0-PB7 IS FOR DATA
00:DF11          VIA_PA       EQU VCS0+1    ;PA  IS USED FOR CONTROL
```

'MENSCH COMPUTER ROM SOFTWARE'
'EQU265.ASM--GLOBAL EQUATES FOR WDC65C265'

```
00:DF12          VIA_PDDB     EQU VCS0+2
00:DF13          VIA_PDDA     EQU VCS0+3
00:DF14          V_PT1CL      EQU VCS0+4
00:DF15          V_PT1CH      EQU VCS0+5
00:DF16          V_PT1LL      EQU VCS0+6
00:DF17          V_PT1LH      EQU VCS0+7
00:DF18          V_PT2CL      EQU VCS0+8
00:DF19          V_PT2CH      EQU VCS0+9
00:DF1A          VIA_PSR      EQU VCS0+10
00:DF1B          VIA_ACR      EQU VCS0+11
00:DF1C          VIA_PCR      EQU VCS0+12
00:DF1D          VIA_PIFR     EQU VCS0+13
00:DF1E          VIA_PIER     EQU VCS0+14
00:DF1F          VIA_PORA     EQU VCS0+15


00:DF20          PD4    EQU  $DF20  PORT 4 DATA REGISTER
00:DF21          PD5    EQU  $DF21  PORT 5 DATA REGISTER
00:DF22          PD6    EQU  $DF22  PORT 6 DATA REGISTER
00:DF23          PD7    EQU  $DF23  PORT 7 DATA REGISTER
00:DF24          PDD4   EQU  $DF24  PORT 4 DATA DIRECTION REGISTER
00:DF25          PDD5   EQU  $DF25  PORT 5 DATA DIRECTION REGISTER
00:DF26          PDD6   EQU  $DF26  PORT 6 DATA DIRECTION REGISTER

00:DF27          PCS7   EQU  $DF27  PORT 7 CHIP SELECT ENABLE REGISTER
                    ;BIT 0-PCS70  SEL PORT REPLACEMENT & EXPANSION
                    ;BIT 1-PCS71  SEL COPROCESSOR EXPANSION
                    ;BIT 2-PCS72  SEL ON CHIP STUFF
                    ;BIT 3-PCS73  SEL 000200-007FFF  'CACHE' MEMORY
                    ;BIT 4-PCS74  SEL 8000-DEFF & E000-FFFF ROM
                    ;BIT 5-PCS75  SEL 4Meg ie BANKS 00-3F
                    ;BIT 6-PCS76  SEL 8Meg ie BANKS 40-BF
                    ;BIT 7-PCS77  SEL 4Meg ie BANKS C0-CF

00:DF28          RVD28  EQU  $DF28  RESERVED $DF28-$DF3F

00:DF40          BCR    EQU  $DF40  BUS CONTROL REGISTER
                     ;134
                    ;BIT 0-EXTERNAL MEM BUS ENABLE
                    ;BIT 1-PORT 44-47 EDGE SENS IRQ
                    ;BIT 2-ALWAYS 0
                    ;BIT 3-ICE ENABLE=1
                    ;BIT 4-PORT 50-53 EDGE SENS IRQ
                    ;BIT 5-PORT 54-57 EDGE SENS IRQ
                    ;BIT 6-NMI,IRQ1,IRQ2 ENABLE = 1
```

```
192                     ;BIT 7-EXTERNAL $F000-$FFFF = 1
193                      ;265
194                     ;BIT 0-EXTERNAL MEM BUS ENABLE
195                     ;BIT 1-TONE GEN 0 ENABLE
196                     ;BIT 2-TONE GEN 1 ENABLE
197                     ;BIT 3-ICE ENABLE=1
198                     ;BIT 4-MONITOR "WATCH DOG" ENABLE
199                     ;BIT 5-ABORT ENABLE = 1 ON P40
200                     ;BIT 6-NMI ENABLE = 1 ON P40
201                     ;BIT 7-EXTERNAL $E000-$FFFF = 1
202
203
```

'MENSCH COMPUTER ROM SOFTWARE'
'EQU265.ASM--GLOBAL EQUATES FOR WDC65C265'

```
204       00:DF41          SSCR    EQU  $DF41  SYSTEM SPEED CONTROL REGISTER
205                           ;BIT 0-FCLK START/STOP     1 = START FCLK
206                           ;BIT 1-PHI2               1 = PHI2 CLK AS FCLK/4
207                           ;BIT 2-EXTERNAL RAM SELECT  1 = EXT $0000-$01FF
208                           ;BIT 3-SYS CS0-CS7 SPEED SEL 1 = FAST FCLK
209                           ;BIT 4-CS4 SPEED SEL       1 = FAST
210                           ;BIT 5-CS5 SPEED SEL       1 = FAST
211                           ;BIT 6-CS6 SPEED SEL       1 = FAST
212                           ;BIT 7-CS7 SPEED SEL       1 = FAST
213
214       00:DF42          TCR     EQU  $DF42  TIMER CONTROL REGISTER
215                           ;BIT 0-TIMER 4 INPUT CLOCK 0 = FCLK
216                           ;BIT 1-TIMER 4 OUTPUT ENABLE 1 = OUTPUT ON P61
217                           ;BIT 2 & 3-PWM  00 = DISABLE,      01 = POSITIVE EDGE
218                           ;BIT 3-       10 = NEGATIVE EDGE, 11 = BOTH EDGES
219                           ;BIT 4-UART0 TIMER SELECT 0 = TIMER 3, 1 = T4
220                           ;BIT 5-UART1 TIMER SELECT 0 = TIMER 3, 1 = T4
221                           ;BIT 6-UART2 TIMER SELECT 0 = TIMER 3, 1 = T4
222                           ;BIT 7-UART3 TIMER SELECT 0 = TIMER 3, 1 = T4
223
224
225                 ;T0 = MONITOR "WATCH DOG"
226                 ;T1 = TIME OF DAY CLOCK
227                 ;T2 = PRESCALED INTERRUPT (UP COUNTER )
228                 ;T3 = UART #2
229                 ;T4 = UART #3
230                 ;T5 = TONE GENERATOR
231                 ;T6 = TONE GENERATOR
232                 ;T7 = PWM
233
234
235
236       00:DF43          TER     EQU  $DF43  TIMER ENABLE REGISTER
237                           ;BIT 0-TIMER 0  1 = ENABLE
238                           ;BIT 1-TIMER 1  1 = ENABLE
239                           ;BIT 2-TIMER 2  1 = ENABLE
240                           ;BIT 3-TIMER 3  1 = ENABLE
241                           ;BIT 4-TIMER 4  1 = ENABLE
242                           ;BIT 5-TIMER 5  1 = ENABLE
243                           ;BIT 6-TIMER 6  1 = ENABLE
244                           ;BIT 7-TIMER 7  1 = ENABLE
245       00:0001          T0FLG   EQU $01
246       00:0002          T1FLG   EQU $02
247       00:0004          T2FLG   EQU $04
248       00:0008          T3FLG   EQU $08
```

```
249        00:0010          T4FLG   EQU $10
250        00:0020          T5FLG   EQU $20
251        00:0040          T6FLG   EQU $40
252        00:0080          T7FLG   EQU $80
253
254
255        00:DF44          TIFR    EQU   $DF44   TIMER INTERRUPT FLAG REGISTER
256                            ;BIT 0-TIMER 0  1 = INTERRUPT PENDING
257                            ;BIT 1-TIMER 1  1 = INTERRUPT PENDING
258                            ;BIT 2-TIMER 2  1 = INTERRUPT PENDING
259                            ;BIT 3-TIMER 3  1 = INTERRUPT PENDING
260                            ;BIT 4-TIMER 4  1 = INTERRUPT PENDING
```

'MENSCH COMPUTER ROM SOFTWARE'
'EQU265.ASM--GLOBAL EQUATES FOR WDC65C265'

```
261                         ;BIT 5-TIMER 5  1 = INTERRUPT PENDING
262                         ;BIT 6-TIMER 6  1 = INTERRUPT PENDING
263                         ;BIT 7-TIMER 7  1 = INTERRUPT PENDING
264
265     00:DF45         EIFR    EQU  $DF45  EDGE INTERRUPT FLAG REGISTER
266                         ;BIT 0-PE56 EDGE IRQ
267                         ;BIT 1-NE57 EDGE IRQ
268                         ;BIT 2-PE60 EDGE IRQ
269                         ;BIT 3-PWM  PROGRAMABLE EDGE IRQ (P62)
270                         ;BIT 4-NE64 EDGE IRQ
271                         ;BIT 5-NE66 EDGE IRQ
272                         ;BIT 6-PIB      IRQ
273                         ;BIT 7-IRQ  LEVEL IRQ
274
275     00:DF46         TIER    EQU  $DF46  TIMER INTERRUPT ENABLE REGISTER
276                         ;BIT 0-TIMER 0  1 = IRQ ENABLE
277                         ;BIT 1-TIMER 1  1 = IRQ ENABLE
278                         ;BIT 2-TIMER 2  1 = IRQ ENABLE
279                         ;BIT 3-TIMER 3  1 = IRQ ENABLE
280                         ;BIT 4-TIMER 4  1 = IRQ ENABLE
281                         ;BIT 5-TIMER 5  1 = IRQ ENABLE
282                         ;BIT 6-TIMER 6  1 = IRQ ENABLE
283                         ;BIT 7-TIMER 7  1 = IRQ ENABLE
284
285     00:DF47         EIER    EQU  $DF47  EDGE INTERRUPT ENABLE REGISTER
286                         ;BIT 0-PE56 EDGE IRQ            1 = IRQ ENABLE
287                         ;BIT 1-NE57 EDGE IRQ            1 = IRQ ENABLE
288                         ;BIT 2-PE60 EDGE IRQ            1 = IRQ ENABLE
289                         ;BIT 3-PWM  PEOGRAMABLE EDGE IRQ (P62) 1 = IRQ ENABLE
290                         ;BIT 4-NE64 EDGE IRQ            1 = IRQ ENABLE
291                         ;BIT 5-NE66 EDGE IRQ            1 = IRQ ENABLE
292                         ;BIT 6-PIB  IRQ               1 = IRQ ENABLE
293                         ;BIT 7-IRQ  LEVEL IRQ           1 = IRQ ENABLE
294     00:0001         PE56ENABLE  EQU $01
295     00:0002         NE57ENABLE  EQU $02
296     00:0004         PE60ENABLE  EQU $04
297     00:0008         PWMENABLE   EQU $08
298     00:0010         NE64ENABLE  EQU $10
299     00:0020         NE66ENABLE  EQU $20
300     00:0040         PIBIRQENABLE EQU $40
301
302     00:DF48         UIFR    EQU  $DF48  UART INTERRUPT FLAG REGISTER
303                         ;BIT 0-UART0 RECEIVE   1 = INTERRUPT PENDING
304                         ;BIT 1-UART0 TRANSMIT  1 = INTERRUPT PENDING
305                         ;BIT 2-UART1 RECEIVE   1 = INTERRUPT PENDING
```

```
306                         ;BIT 3-UART1 TRANSMIT  1 = INTERRUPT PENDING
307                         ;BIT 4-UART2 RECEIVE   1 = INTERRUPT PENDING
308                         ;BIT 5-UART2 TRANSMIT  1 = INTERRUPT PENDING
309                         ;BIT 6-UART3 RECEIVE   1 = INTERRUPT PENDING
310                         ;BIT 7-UART3 TRANSMIT  1 = INTERRUPT PENDING
311       00:0001         UART0R  EQU $01
312       00:0002         UART0T  EQU $02
313       00:0004         UART1R  EQU $04
314       00:0008         UART1T  EQU $08
315       00:0010         UART2R  EQU $10
316       00:0020         UART2T  EQU $20
317       00:0040         UART3R  EQU $40
```

'MENSCH COMPUTER ROM SOFTWARE'
'EQU265.ASM--GLOBAL EQUATES FOR WDC65C265'

```
318       00:0080          UART3T  EQU $80
319
320       00:DF49          UIER    EQU  $DF49  UART INTERRUPT ENABLE REGISTER
321                           ;BIT 0-UART0 RECEIVE   1 = ENABLE IRQ
322                           ;BIT 1-UART0 TRANSMIT  1 = ENABLE IRQ
323                           ;BIT 2-UART1 RECEIVE   1 = ENABLE IRQ
324                           ;BIT 3-UART1 TRANSMIT  1 = ENABLE IRQ
325                           ;BIT 4-UART2 RECEIVE   1 = ENABLE IRQ
326                           ;BIT 5-UART2 TRANSMIT  1 = ENABLE IRQ
327                           ;BIT 6-UART3 RECEIVE   1 = ENABLE IRQ
328                           ;BIT 7-UART3 TRANSMIT  1 = ENABLE IRQ
329
330
331       00:DF4A          RVD4A   EQU  $DF4A  RESERVED I/O $DF4A-$DF4F
332
333       00:DF50          T0LL    EQU  $DF50  TIMER 0 LATCH LOW
334       00:DF51          T0LH    EQU  $DF51  TIMER 0 LATCH HIGH
335       00:DF52          T1LL    EQU  $DF52  TIMER 1 LATCH LOW
336       00:DF53          T1LH    EQU  $DF53  TIMER 1 LATCH HIGH
337       00:DF54          T2LL    EQU  $DF54  TIMER 2 LATCH LOW
338       00:DF55          T2LH    EQU  $DF55  TIMER 2 LATCH HIGH
339       00:DF56          T3LL    EQU  $DF56  TIMER 3 LATCH LOW
340       00:DF57          T3LH    EQU  $DF57  TIMER 3 LATCH HIGH
341       00:DF58          T4LL    EQU  $DF58  TIMER 4 LATCH LOW
342       00:DF59          T4LH    EQU  $DF59  TIMER 4 LATCH HIGH
343       00:DF5A          T5LL    EQU  $DF5A  TIMER 5 LATCH LOW
344       00:DF5B          T5LH    EQU  $DF5B  TIMER 5 LATCH HIGH
345       00:DF5C          T6LL    EQU  $DF5C  TIMER 6 LATCH LOW
346       00:DF5D          T6LH    EQU  $DF5D  TIMER 6 LATCH HIGH
347       00:DF5E          T7LL    EQU  $DF5E  TIMER 7 LATCH LOW
348       00:DF5F          T7LH    EQU  $DF5F  TIMER 7 LATCH HIGH
349
350       00:DF60          T0CL    EQU  $DF60  TIMER 0 COUNTER LOW
351       00:DF61          T0CH    EQU  $DF61  TIMER 0 COUNTER HIGH
352       00:DF62          T1CL    EQU  $DF62  TIMER 1 COUNTER LOW
353       00:DF63          T1CH    EQU  $DF63  TIMER 1 COUNTER HIGH
354       00:DF64          T2CL    EQU  $DF64  TIMER 2 COUNTER LOW
355       00:DF65          T2CH    EQU  $DF65  TIMER 2 COUNTER HIGH
356       00:DF66          T3CL    EQU  $DF66  TIMER 3 COUNTER LOW
357       00:DF67          T3CH    EQU  $DF67  TIMER 3 COUNTER HIGH
358       00:DF68          T4CL    EQU  $DF68  TIMER 4 COUNTER LOW
359       00:DF69          T4CH    EQU  $DF69  TIMER 4 COUNTER HIGH
360       00:DF6A          T5CL    EQU  $DF6A  TIMER 5 COUNTER LOW
361       00:DF6B          T5CH    EQU  $DF6B  TIMER 5 COUNTER HIGH
362       00:DF6C          T6CL    EQU  $DF6C  TIMER 6 COUNTER LOW
```

```
363     00:DF6D         T6CH    EQU   $DF6D   TIMER 6 COUNTER HIGH
364     00:DF6E         T7CL    EQU   $DF6E   TIMER 7 COUNTER LOW
365     00:DF6F         T7CH    EQU   $DF6F   TIMER 7 COUNTER HIGH
366
367     00:DF70         ACSR0   EQU   $DF70   ASYNCH. CONTROL/STATUS REGISTER 0
368                       ;BIT 0-XMIT PORT ENABLE
369                       ;BIT 1-XMIT IRQ SOURCE
370                       ;BIT 2-7/8 BIT DATA
371                       ;BIT 3-PARITY ENABLE
372                       ;BIT 4-ODD/EVEN PARITY
373                       ;BIT 5-RECV ENABLE
374                       ;BIT 6-SOFTWARE SEMIPHORE
```

'MENSCH COMPUTER ROM SOFTWARE'
'EQU265.ASM--GLOBAL EQUATES FOR WDC65C265'

```
375                         ;BIT 7-RECV ERROR FLG
376       00:0001           SON     EQU $01
377       00:0002           DISCH   EQU $02
378
379       00:DF71           ARTD0   EQU  $DF71  ASYNCH. RECVR/TRANSMTR DATA REGISTER 0
380       00:DF72           ACSR1   EQU  $DF72  ASYNCH. CONTROL/STATUS REGISTER 1
381       00:DF73           ARTD1   EQU  $DF73  ASYNCH. RECVR/TRANSMTR DATA REGISTER 1
382       00:DF74           ACSR2   EQU  $DF74  ASYNCH. CONTROL/STATUS REGISTER 2
383       00:DF75           ARTD2   EQU  $DF75  ASYNCH. RECVR/TRANSMTR DATA REGISTER 2
384       00:DF76           ACSR3   EQU  $DF76  ASYNCH. CONTROL/STATUS REGISTER 3
385       00:DF77           ARTD3   EQU  $DF77  ASYNCH. RECVR/TRANSMTR DATA REGISTER 3
386
387
388
389                         ;UART IO LINES
390       00:0001           DTR0    EQU $01
391       00:0004           DTR1    EQU $04
392       00:0010           DTR2    EQU $10
393       00:0040           DTR3    EQU $40
394
395       00:0002           DSR0    EQU $02
396       00:0008           DSR1    EQU $08
397       00:0020           DSR2    EQU $20
398       00:0080           DSR3    EQU $80
399
400
401
402       00:DFE1           VIA2_PDA      EQU $DFE1     ;POWER CONTROL
403       00:DFE3           VIA2_PDDA     EQU $DFE3
404
405                          .END
406
407 00:0000                      INCLUDE R_EQU.H
408                       .STTL 'R_EQU.H - MENSCH EQUATES FOR ROM MONITOR
409                       .PAGE
```

```
    'MENSCH COMPUTER ROM SOFTWARE'
    'R_EQU.H - MENSCH EQUATES FOR ROM MONITOR

410                     ;11-10-1994
411
412
413
414
415
416                 ********************************************
417                 *                                          *
418                 *    THIS IS THE PORT MAP FOR THE          *
419                 *      MENSCH COMPUTER FIRMWARE            *
420                 *                                          *
421                 ********************************************
422
423
424                     ;PD0  ADDRESS REGISTER A0-A7
425                     ;PD1 ADDRESS REGISTER A8-A15
426                     ;PD2  DATA REGISTER D0-D7
427                     ;PD3  ADDRESS REGISTER A16-A23
428                     ;PD4  PORT 4
429                             ;BIT 0-NMI
430                             ;BIT 1-IRQ
431                             ;BIT 2-PCMCIA Card Sense
432                             ;BIT 3-PCMCIA Card Sense
433                             ;BIT 4-PCMCIA Card Sense
434                             ;BIT 5-PCMCIA Card Sense
435                             ;BIT 6-RES* for VIA's
436                             ;BIT 7-RES for PCMCIA slots
437                     ;PD6  PORT 5
438                             ;BIT 0-DTR0 SERIAL UART
439                             ;BIT 1-DSR0 SERIAL UART
440                             ;BIT 2-DTR1 SERIAL UART
441                             ;BIT 3-DSR1 SERIAL UART
442                             ;BIT 4-DTR2 SERIAL UART
443                             ;BIT 5-DSR2 SERIAL UART
444                             ;BIT 6-DTR3 SERIAL UART
445                             ;BIT 7-DSR3 SERIAL UART
446                     ;PD6  PORT 6
447                             ;BIT 0-RXD0 SERIAL UART
448                             ;BIT 1-TXD0 SERIAL UART
449                             ;BIT 2-RXD1 SERIAL UART
450                             ;BIT 3-TXD1 SERIAL UART
451                             ;BIT 4-RXD2 SERIAL UART
452                             ;BIT 5-TXD2 SERIAL UART
453                             ;BIT 6-RXD3 SERIAL UART
454                             ;BIT 7-TXD3 SERIAL UART
```

```
455                 ;PD7  PORT 7
456                       ;BIT 0-CS0 VIA for LCD
457                       ;BIT 1-CS1 VIA for Pwr Xtrl & Game Port
458                       ;BIT 2-CS2 Expansion Header J6
459                       ;BIT 3-CS3 RAM   $0200 - $7FFF
460                       ;BIT 4-CS4 EPROM $8000 - $FFFF
461                       ;BIT 5-CS5 Low IC Card
462                       ;BIT 6-CS6 Hi IC Card
463                       ;BIT 7-CS7 Expansion Header J6
464
465
466
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_EQU.H - MENSCH EQUATES FOR ROM MONITOR

```
467                     ;Port Replacement & Expansion
468
469
470     00:DF10         DISP_DATA_DREG EQU $DF10  ;VIA B PORT
471                        ;BIT 0-DISPLAY D0       (INPUT/OUTPUT)
472                        ;BIT 1-DISPLAY D1       (INPUT/OUTPUT)
473                        ;BIT 2-DISPLAY D2       (INPUT/OUTPUT)
474                        ;BIT 3-DISPLAY D3       (INPUT/OUTPUT)
475                        ;BIT 4-DISPLAY D4       (INPUT/OUTPUT)
476                        ;BIT 5-DISPLAY D5       (INPUT/OUTPUT)
477                        ;BIT 6-DISPLAY D6       (INPUT/OUTPUT)
478                        ;BIT 7-DISPLAY D7       (INPUT/OUTPUT)
479     00:DF12         DISP_DATA_DIR  EQU $DF12  ;DATA DIRECTION REG
480
481     00:DF11         DISP_CNTL_REG EQU $DF11 ;CONTROL REG TO DISPLAY VIA B PORT
482                        ;BIT 0-DISPLAY ENABLE     (OUTPUT)
483                        ;BIT 1-DISPLAY REG SELECT  (OUTPUT)
484                        ;BIT 2-DISPLAY READ/WRITE* (OUTPUT)
485                        ;BIT 3-Bat Voltage Detector (INPUT)
486                        ;BIT 4-Not used          (INPUT)
487                        ;BIT 5-Not used          (INPUT)
488                        ;BIT 6-DISPLAY Select     (OUTPUT)
489                        ;BIT 7-Display Reset      (OUTPUT)
490     00:DF13         DISP_CNTL_DIR EQU  $DF13  ;DATA DIRECTION REG
491
492
493
494                     ;VIA USED FOR SEGA GAME PORT & POWER XTROL
495
496     00:DFE0         SEGA_DATA_REG  EQU $DFE0  ;SEGA GAME PORT   all inputs
497                        ;BIT 0-PIN 1
498                        ;BIT 1-PIN 2
499                        ;BIT 2-PIN 3
500                        ;BIT 3-PIN 4
501                        ;BIT 4-PIN 6
502                        ;BIT 5-PIN 7
503                        ;BIT 6-PIN 9
504                        ;BIT Sega PWR Control*
505                        ;   Low = Pwr on
506     00:DFE2         SEGA_DATA_DIR  EQU $DFE2  ;DATA DIRECTION REG
507
508
509
510     00:DFE1         PWR_XTRL_REG   EQU $DFE1 ;VIA A PORT
511                        ;BIT 0-Display power*
```

```
512                             ;BIT 1-Printer Port PWR*
513                             ;BIT 2-Host Port PWR*
514                             ;BIT 3-MODEM Port PWR
515                             ;BIT 4-Spkr AMP PWR on & Expansion PIN 52
516                             ;BIT 5-Expansion PIN 54
517                             ;BIT 6-Expansion PIN 56
518                             ;BIT 7-Expansion PIN 58
519     00:DFE3         PWR_XTRL_DIR   EQU $DFE3  ;DATA DIRECTION REG
520
521
522
523             ******************************************
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_EQU.H - MENSCH EQUATES FOR ROM MONITOR

```
524                 *                               *
525                 *    THIS IS THE TIMMER MAP FOR THE      *
526                 *       MENSCH COMPUTER PROGRAMS       *
527                 *                               *
528                 ************************************************
529
530                       ;T0  TIMER #0 IS NOT USED
531                       ;T1  TIMER #1 IS FOR TIME OF DAY CLOCK TIMING
532                       ;T2  TIMER #2 IS FOR GENERAL 10 MSEC TIMING
533                       ;T3  TIMER #3 IS FOR UART BAUD RATE TIMING
534                       ;T4  TIMER #4 IS FOR UART BAUD RATE TIMING
535                       ;T5  TIMER #5 IS FOR TONE GENERATOR TIMING
536                       ;T6  TIMER #6 IS FOR TONE GENERATOR TIMING
537                       ;T7  TIMER #7 IS FOR PULSE WIDTH MEASUREMENT TIMING
538
539
540
541                         END
542
543  00:0000                      INCLUDE R_RAM.ASM
544                       .STTL 'R_RAM.ASM - MENSCH RAM ASSIGNMENTS FOR ROM MONITOR
545                       .PAGE
```

```
   'MENSCH COMPUTER ROM SOFTWARE'
   'R_RAM.ASM - MENSCH RAM ASSIGNMENTS FOR ROM MONITOR

546                     ;01-07-1995
547                     ;MODIFIED FOR MENSCH II
548
549
550
551
552                     .PAGE0
553 00:0000                     ORG   $00:0000
554
555                 *                           *
556                 *  THIS IS THE SERIAL I/O Xtrol FOR THE   *
557                 *       MENSCH COMPUTER PROGRAMS        *
558                 *                           *
559                 *********************************************
560
561 00:0000               SININDX0  .ds 2    ;SERIAL 0 IN INPUT PTR
562 00:0002               SINEND0   .ds 2      ;SERIAL 0 IN OUTPUT PTR
563 00:0004               SIN_BUF0  .ds 2    ;SERIAL 0 IN BUFFER
564 00:0006               SINCNT0   .ds 2    ;SERIAL 0 BUFFER SIZE
565
566 00:0008               SININDX1  .ds 2    ;SERIAL 1 IN INPUT PTR
567 00:000A               SINEND1   .ds 2      ;SERIAL 1 IN OUTPUT PTR
568 00:000C               SIN_BUF1  .ds 2    ;SERIAL 1 IN BUFFER
569 00:000E               SINCNT1   .ds 2    ;SERIAL 1 BUFFER SIZE
570
571 00:0010               SININDX2  .ds 2    ;SERIAL 2 IN INPUT PTR
572 00:0012               SINEND2   .ds 2      ;SERIAL 2 IN OUTPUT PTR
573 00:0014               SIN_BUF2  .ds 2    ;SERIAL 2 IN BUFFER
574 00:0016               SINCNT2   .ds 2    ;SERIAL 2 BUFFER SIZE
575
576 00:0018               SININDX3  .ds 2    ;SERIAL 3 IN INPUT PTR
577 00:001A               SINEND3   .ds 2      ;SERIAL 3 IN OUTPUT PTR
578 00:001C               SIN_BUF3  .ds 2    ;SERIAL 3 IN BUFFER
579 00:001E               SINCNT3   .ds 2    ;SERIAL 3 BUFFER SIZE
580
581
582 00:0020               SOUTINDX0  .ds 2    ;SERIAL 0 OUT INPUT PTR
583 00:0022               SOUTEND0   .ds 2    ;SERIAL 0 OUT OUTPUT PTR
584 00:0024               SOUTBUF0   .ds 2    ;SERIAL 0 OUT BUFFER
585 00:0026               SOUTCNT0   .ds 2    ;SERIAL 0 OUT BUFFER SIZE
586
587 00:0028               SOUTINDX1 .ds 2    ;SERIAL 1 OUT INPUT PTR
588 00:002A               SOUTEND1  .ds 2    ;SERIAL 1 OUT OUTPUT PTR
589 00:002C               SOUTBUF1  .ds 2    ;SERIAL 1 OUT BUFFER
590 00:002E               SOUTCNT1  .ds 2    ;SERIAL 1 OUT BUFFER SIZE
```

```
591
592 00:0030                SOUTINDX2  .ds 2    ;SERIAL 2 OUT INPUT PTR
593 00:0032                SOUTEND2   .ds 2    ;SERIAL 2 OUT OUTPUT PTR
594 00:0034                SOUTBUF2   .ds 2    ;SERIAL 2 OUT BUFFER
595 00:0036                SOUTCNT2   .ds 2    ;SERIAL 2 OUT BUFFER SIZE
596
597 00:0038                SOUTINDX3  .ds 2    ;SERIAL 3 OUT INPUT PTR
598 00:003A                SOUTEND3   .ds 2    ;SERIAL 3 OUT OUTPUT PTR
599 00:003C                SOUTBUF3   .ds 2    ;SERIAL 3 OUT BUFFER
600 00:003E                SOUTCNT3   .ds 2    ;SERIAL 3 OUT BUFFER SIZE
601
602
```

```
    'MENSCH COMPUTER ROM SOFTWARE'
    'R_RAM.ASM - MENSCH RAM ASSIGNMENTS FOR ROM MONITOR

603 00:0040             SFLAG0  .ds 1     ;SERIAL CONTROL FLAGS
604                        ;BIT 6-BEEP  SFLAG0 only
605 00:0041             SFLAG1  .ds 1
606 00:0042             SFLAG2  .ds 1
607 00:0043             SFLAG3  .ds 1
608                        ;BIT 0-SERIAL INPUT QUEUE DATA
609                        ;BIT 1-CONTROL 'C' RECEIVED, FLUSH QUEUE
610                        ;BIT 2-XON/XOFF CONTROL USED=1
611                        ;BIT 3-XON/XOFF OR HDW HS SEND OVERFLOW
612                        ;BIT 4-LAST CNTRL CHAR WAS XON=1 XOFF=0
613                        ;BIT 5-ECHO ON/OFF FLAG OFF=1
614                        ;BIT 6-OUTPUT XOFF
615                        ;BIT 7-OUTPUT XON
616       00:0001       SFLG    EQU $01
617       00:0002       CFLG    EQU $02
618       00:0004       XONOFLG  EQU $04
619       00:0008       SNDOVF   EQU $08
620       00:0010       LASTXONOF EQU $10
621       00:0020       ECHOFF   EQU $20
622       00:0040       SXOFFLG  EQU $40
623       00:0080       SXONFLG  EQU $80
624
625       00:0040       BEEP    EQU $40   ;SFLAG0  only
626
627 00:0044             SDATA_SI0 .DS 1
628 00:0045             SDATA_SI1 .DS 1
629 00:0046             SDATA_SI2 .DS 1
630 00:0047             SDATA_SI3 .DS 1
631
632 00:0048             STATUS_S0 .DS 1
633 00:0049             STATUS_S1 .DS 1
634 00:004A             STATUS_S2 .DS 1
635 00:004B             STATUS_S3 .DS 1
636 00:004C             STEMP_Sx  .DS 1
637
638 00:004D             INPUT_XTRL  .ds 1   ;INPUT PORT FLAGS
639 00:004E             OUTPUT_XTRL .ds 1   ;OUTPUT PORT FLAGS
640 00:004F             IOTEMP     .ds 1   ;OUTPUT TEMP FOR CHAR
641
642
643
644                .page
```

```
   'MENSCH COMPUTER ROM SOFTWARE'
   'R_RAM.ASM - MENSCH RAM ASSIGNMENTS FOR ROM MONITOR

                ********************************************
                *                                          *
                *    THIS IS THE DISPLAY MAP FOR THE       *
                *       MENSCH COMPUTER PROGRAMS           *
                *                                          *
                ********************************************

 00:0050                DISP_PTR  .DS 3     ;INDIRECT POINTER FOR LCD


 00:0053                DISPTYP  .ds 1       ;DISPLAY TYPE AND IF
                           ;TOD DISPLAY IS ON
                           ;BIT 0-2 TYPE OF DISPLAY
                           ;BIT 3-
                           ;BIT 4-A MENSCH PLATFORM
                           ;BIT 5-POWER UP IN PROGRESS FLG
                           ;BIT 6-DISPLAY NOT WORKING
                           ;BIT 7-TOD ON DISPLAY FLG
       00:000F         DTYPMSK  EQU $0F      ;UP TO 8 TYPES OF DISPLAYS   >7=Default
       00:0010         A_MENSCH EQU $10
       00:0020         PUFLG   EQU $20       ;POWER UP
       00:0040         NO_DISPLAY  EQU $40   ;DISPLAY NOT WORKING
       00:0080         DISP_TOD_FLG EQU $80   ;DISPLAY TOD


                ********************************************
                *                                          *
                *    THIS IS THE TEMP RAM MAP FOR THE      *
                *       MENSCH COMPUTER PROGRAMS           *
                *                                          *
                ********************************************

 00:0054                TMPRY_PTR .ds 3       ;G.P. TEMPORARY POINTER

 00:0057                TMPC    .ds 2         ;COUNT DOWN CTR FOR S28


 00:0059                WRAP    .ds 1         ;$FF:FFFF WRAP AROUND

 00:005A                DIFF    .ds 3        ;EA-SA = DIFF (3 BYTES)
 00:005D                TMP0    .ds 3        ;START ADDR (3 BYTES)
 00:0060                TMP1    .ds 3
 00:0063                TMP2    .ds 3
 00:0066                TMP4    .ds 3
 00:0069                TMP6    .ds 3
```

```
690 00:006C             TMP8    .ds 3       ;USED IN REAL-TIME CLOCK DISPLAY
691 00:006F
692 00:006F             ERRORS  .ds 1       ;S28 DOWNLOAD ERROR COUNT
693 00:0070             TEMP    .ds 2
694
695 00:0072             R_TYPE  .ds 1               ;USED IN LOADERS
696
697
698
699                 ;JUMPS FOR ROM I/O
700 00:0073             GET_CHR_JMP   .ds 2
701 00:0075             PUT_CHR_JMP   .ds 2
```

```
    'MENSCH COMPUTER ROM SOFTWARE'
    'R_RAM.ASM - MENSCH RAM ASSIGNMENTS FOR ROM MONITOR

702  00:0077              GET|PUT_CHR_JMP .ds 2
703  00:0079              CLR_LCD_JMP    .ds 2
704  00:007B              DISP_LCD_JMP   .ds 2
705  00:007D              TXT_CUR_JMP    .ds 2
706  00:007F              SND_BEEP_JMP   .ds 2
707  00:0081              LO_PWR_JMP     .ds 2
708
709
710                 ******************************************
711                 *                                        *
712                 *    THIS IS THE DTMF GENERATION FOR OUT *
713                 *        DIALING MAP FOR THE             *
714                 *       MENSCH COMPUTER PROGRAMS         *
715                 *                                        *
716                 ******************************************
717
718
719
720       00:0006           DTMF    EQU $06      ;DTMF CONTROL TG0 & TG1
721       00:0002           SNGL    EQU $02      ;SINGLE TONE CONTROL TG0
722
723       00:0000           ATG     EQU $0  ;AUDIBLE TONE GENERATOR
724  00:0083              T_TIME   .ds 2        ;tone duration timer
725
726  00:0085              INTKNT1  .ds 1
727
728                 .page
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_RAM.ASM - MENSCH RAM ASSIGNMENTS FOR ROM MONITOR

```
                ********************************************
                *                                          *
                *  THIS IS THE OTHER FEATURES MAP FOR THE  *
                *       MENSCH COMPUTER PROGRAMS           *
                *                                          *
                ********************************************

00:0086                 DUMP_FLGS .DS 1 ;(format control byte)
      00:0001           Flag1 equ Bit0   ; output S28+byte-count
      00:0002           Flag2 equ Bit1   ;add space after address
      00:0004           Flag3 equ Bit2   ;add spaces between data bytes
      00:0008           Flag4 equ Bit3   ;add checksum
      00:0010           Flag5 equ Bit4   ;add : after bank addr
      00:0020           Flag6 equ Bit5   ;add printer page header
      00:0040           Flag7 equ Bit6   ;ASCII not Hex data

                ;   Working Variable Locations & Definitions

00:0087                 OUTBUF     .ds 40          ;A STRING OUTPUT BUFFER

00:00AF                 LINE_CNT    .ds 2    ;LINE COUNT FOR OUTPUT DEVICE
00:00B1                 LINE_MAX    .ds 2    ;MAX LINES PER PAGE OR SCREEN

                ;       ******************

                .page
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_RAM.ASM - MENSCH RAM ASSIGNMENTS FOR ROM MONITOR

```
755                     ;       **** Misc **************
756
757  00:00B3                    COUNT       .ds 2     ;RECORD COUNTER IN DUMP
758  00:00B5                    CARD        .ds 1     ;1 = HI  0 = LO
759
760  00:00B6                    OUTPUT_TMP  .ds 1     ;TEMPORARY OUTPUT SOURCE REG
761  00:00B7                    INPUT_SRC   .ds 1     ;LAST INPUT SOURCE IN GET_CHR
762
763  00:00B8                    source      .ds 4
764  00:00BC                    dest        .ds 4
765
766
767         00:00C0             PZLASTBYTE EQU *
768
769         00:003F             PZSPACE EQU $00FF-PZLASTBYTE    ;gives space left in Page 0
770
771
772
773                         .ENDS       ;Ends page 0 declarations
774
775
776
777                         .STTL 'M_RAM.ASM - MENSCH Vectors & Buffers 100H'
778                         .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'M_RAM.ASM - MENSCH Vectors & Buffers 100H'

```
779
780
781                     .DATA
782 00:0100                     .ORG $00:0100
783
784
785 00:0100             UBRK    .ds 4       ;USER BREAK
786 00:0104             UNMI    .ds 4       ;USER NMI VECTOR
787 00:0108             UNIRQ   .ds 4       ;USER IRQ VECTOR
788 00:010C             COPIRQ  .ds 4       ;USER CO-PROCESSOR IRQ
789 00:0110             IABORT  .ds 4               ;USER ABORT POINTER
790 00:0114             PIBIRQ  .ds 4       ;PERIPHERAL INTERFACE IRQ
791 00:0118             EDGEIRQS .ds 4      ;ALL EDGE IRQS
792 00:011C             UNIRQT7 .ds 4       ;USER TIMER 7 IRQ
793 00:0120             UNIRQT2 .ds 4       ;USER TIMER 2 IRQ
794 00:0124             UNIRQT1 .ds 4       ;USER TIMER 1 IRQ
795 00:0128             UNIRQT0 .ds 4       ;USER TIMER 0 IRQ
796 00:012C             USER_CMD .ds 4      ;USER COMMAND
797 00:0130             URESTART .ds 4      ;USER PWR UP RESTART VECT
798 00:0134             UALRMIRQ .ds 4      ;USER --ALARM WAKEUP CALL
799
800             ;*********************************************
801
802 00:0138             FORMAT_FLAGS .ds 1  ;FLAG BITS FOR LCD SCREEN FORMAT & UPDATES
803
804     00:0001         TIME_CHK    .equ Bit0 ;TIME TO UPDATE TIME DISPLAY
805     00:0002         DATE_CHK    .equ Bit1 ;TIME TO UPDATE DATE DISPLAY
806
807             ;*********************************************
808
809             ;SERIAL BUFFERS FOR ROM ONLY
810
811 00:0139             ROM_IBUF0 .ds 10
812 00:0143             ROM_IBUF1 .ds 10
813 00:014D             ROM_IBUF2 .ds 10
814 00:0157             ROM_IBUF3 .ds 10
815
816 00:0161             ROM_OBUF0 .ds 10
817 00:016B             ROM_OBUF1 .ds 10
818 00:0175             ROM_OBUF2 .ds 10
819 00:017F             ROM_OBUF3 .ds 10
820
821 00:0189             STR_BUF_PTR  .ds 2      ;POINTER FOR INPUT BUFFER
822 00:018B             STR_BUF_HDR  .ds 3  ;A PLACE FOR HI/LO:
823 00:018E             STR_BUF             .ds 40;A STRING INPUT BUFFER
```

```
824 00:01B6                 D|TBUF     .ds 10    ;SHORT BUFFER FOR DATE & TIME
825
826
827
828
829
830       00:01C0           P1LASTBYTE EQU *
831
832       00:003F           P1SPACE EQU $01FF-P1LASTBYTE    ;gives space left in Page 1
833
834
835                 ;The ROM only Stack will start at 00:01FFh
```

'MENSCH COMPUTER ROM SOFTWARE'
'M_RAM.ASM - MENSCH Vectors & Buffers 100H'

```
836
837                     .STTL 'M_RAM.ASM - High RAM for special variables'
838                     .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'M_RAM.ASM - High RAM for special variables'

```
839
840                 ********************************************
841                 *                                          *
842                 *    THIS IS THE INTERNAL RAM FOR $DF80    *
843                 *       ALL WDC W65C265 PROGRAMS           *
844                 *                                          *
845                 ********************************************
846
847
848 00:DF80                           .org $00:DF80
849
850
851                 ******************************************************
852                 ******************************************************
853                 * WARNING These 12 bytes Below  MUST be kept in order *
854                 ******************************************************
855                 ******************************************************
856
857
858 00:DF80               ACC     .ds 2       ;TEMP ACC REG
859 00:DF82               XREG    .ds 2       ;TEMP X REG
860 00:DF84               YREG    .ds 2       ;TEMP Y REG
861 00:DF86               STK|PTR .ds 2       ;TEMP STACK POINTER
862
863 00:DF88               DIRREG  .ds 2       ;DIRECT PAGE REG
864 00:DF8A               DBREG   .ds 1       ;DATA BANK REG
865
866 00:DF8B               FLGS    .ds 1       ;CONDITIONAL CODE REG
867                          ;BIT 0--CARRY BIT/EMULATION FLAG
868                          ;BIT 1--ZERO BIT
869                          ;BIT 2--INTERRUPT ENABLE BIT
870                          ;BIT 3--BINARY CODED DECIMAL
871                          ;BIT 4--INDEX REG SELECT/BRK
872                          ;BIT 5--ACC REG SELECT/E-BIT
873                          ;BIT 6--OVERFLOW
874                          ;BIT 7--MINUS
875
876       00:DF8C           EBIT    EQU *
877
878                 ******************************************************
879                 ******************************************************
880                 * WARNING These 13 bytes Above  MUST be kept in order *
881                 ******************************************************
882                 ******************************************************
883
```

```
884
885
886 00:DF8C                PCL     .ds 1       ;PROGRAM COUNTER LOW
887 00:DF8D                PCH     .ds 1       ;PROGRAM COUNTER HIGH
888 00:DF8E                TPBR    .ds 1       ;PROGRAM BANK REG
889 00:DF8F                SB_SENTL .ds 1      ;SENTINAL BYTE FOR ROMULATOR
890
891
892
893
894
895
```

'MENSCH COMPUTER ROM SOFTWARE'
'M_RAM.ASM - High RAM for special variables'

```
896
897
898                 ******************************************************
899                 ******************************************************
900                 * WARNING These 7 bytes Below  MUST be kept in order *
901                 ******************************************************
902                 ******************************************************
903
904 00:DF90               DAYWK   .ds 1       ;DAY OF WEEK 1 = SUNDAY
905                           ;7 = SATURDAY
906 00:DF91               MONTH   .ds 1       ;MONTH 1= JAN 12= DEC
907 00:DF92               DAY     .ds 1       ;DAY     1 TO 28,29,30,31
908 00:DF93               YR      .ds 1       ;91
909
910 00:DF94               HR      .ds 1       ;HOURS    0 TO 23
911 00:DF95               MIN     .ds 1       ;MINUTES  0 TO 59
912 00:DF96               SEC     .ds 1       ;SECONDS  0 TO 59
913
914
915                 ******************************************************
916                 ******************************************************
917                 * WARNING These 7 bytes Above  MUST be kept in order *
918                 ******************************************************
919                 ******************************************************
920
921
922
923                 ******************************************************
924                 ******************************************************
925                 * WARNING These 7 bytes Below  MUST be kept in order *
926                 ******************************************************
927                 ******************************************************
928
929 00:DF97               ADAYWK  .ds 1       ;DAY OF WEEK 1 = SUNDAY
930                           ;7 = SATURDAY
931 00:DF98               AMONTH  .ds 1       ;MONTH 1= JAN 12= DEC
932 00:DF99               ADAY    .ds 1       ;DAY     1 TO 28,29,30,31
933 00:DF9A               AYR     .ds 1       ;91
934
935 00:DF9B               AHR     .ds 1       ;HOURS    0 TO 23
936 00:DF9C               AMIN    .ds 1       ;MINUTES  0 TO 59
937 00:DF9D               ASEC    .ds 1       ;SECONDS  0 TO 59
938
939
940                 ******************************************************
```

```
941                 ******************************************************
942                 * WARNING These 7 bytes Above  MUST be kept in order *
943                 ******************************************************
944                 ******************************************************
945
946  00:DF9E                H100HZ  .ds 1       ;0.01 SEC (ie 10 MSEC)
947  00:DF9F                H10HZ   .ds 1
948  00:DFA0                H1HZ    .ds 1
949  00:DFA1                TENTHSEC .ds 1      ;0.1 SEC
950
951  00:DFA2                DAYLIT  .ds 1       ;DAY LIGHT SAVINGS TIME
952                            ;BIT 0--ENABLED =1
```

'MENSCH COMPUTER ROM SOFTWARE'
'M_RAM.ASM - High RAM for special variables'

```
953                             ;BIT 1-6 NU
954                             ;BIT 7--IN PROCESS OF
955                             ;       MODIFYING TOD
956         00:0001             DAYLITFLG EQU $01
957         00:0080             DAYLPROG  EQU $80
958
959  00:DFA3                    TODCKS   .ds 2        ;clock checksum
960
961
962  00:DFA5                    DOWNT0   .ds 2        ;COUNT DOWN TIMER 0
963  00:DFA7                    DOWNT1   .ds 2        ;COUNT DOWN TIMER 1
964  00:DFA9                    DOWNT2   .ds 2        ;COUNT DOWN TIMER 2
965  00:DFAB                    DOWNT3   .ds 2        ;COUNT DOWN TIMER 3
966  00:DFAD                    DOWNT4   .ds 2        ;COUNT DOWN TIMER 4
967
968  00:DFAF                    UPT0     .ds 4        ;COUNT UP TIMER 0 FOR
969
970  00:DFB3                    PWD_CELLS .ds 2                    ;POWER DOWN SENTINALS
971
972                                     ;STOP WATCH
973
974  00:DFB5                    SPEED    .ds 1        ;MAIN XTAL SPEED
975                             ;0 = 1.843200MHZ
976                             ;1 = 2.457600MHZ
977                             ;2 = 3.686400MHZ
978                             ;3 = 4.915200MHZ
979                             ;4 = 6.144000MHZ
980
981
982                     .page
```

'MENSCH COMPUTER ROM SOFTWARE'
'M_RAM.ASM - High RAM for special variables'

```
983
984  00:DFB6                FLAGS   .ds 1  ;SYSTEM FLAGS FOR ALARM & SPEED DIALING
985                         ; BIT 0 EQU ALARM ENABLE
986                         ; BIT 1 EQU ALARM IRQ
987                         ; BIT 2 EQU ALARM RESET
988                         ; BIT 3 EQU RESET COMPENSATION
989                         ; BIT 4 EQU DELAY ON DTMF DIALING
990                         ; BIT 5 EQU SPEED DIAL ACTIVE
991                         ; BIT 6 EQU GOT A TIMER INTERRUPT FLAG
992                         ; BIT 7 EQU
993
994        00:0040          TMRIFLG  EQU $40  ;TIMER INTERRUPT OCCURED FLG
995        00:0020          SPDFLG   EQU $20  ;SPEED DIAL FLG
996        00:0010          DIALDELY EQU $10  ;DELAY ON DTMF
997        00:0008          RES_COMP EQU $08  ;ADD RESET TIME TO TOD CLOCK
998        00:0004          ALRMRST  EQU $04  ;RESET ALARM
999        00:0002          ALRMIRQ  EQU $02  ;ALARM IS ACTIVE
1000       00:0001          ALRMENAB EQU $01  ;ALARM IS SET
1001
1002
1003  00:DFB7               DTMFTMR  .ds 1       ;DTMF DURATION COUNTER
1004
1005
1006  00:DFB8               PD_TIMER .ds 2
1007
1008
1009
1010       00:DFBA          DFLASTBYTE EQU *
1011
1012       00:0005          DFSPACE EQU $DFBF-DFLASTBYTE    ;gives space left in the INTERNAL RAM
1013
1014                     .ENDS
1015
1016                .END
1017
1018  00:0000                    INCLUDE ASCII.H
1019                ;FILE:  ASCII.H
1020                ;DATE:  11-10-94
1021
1022
1023                   .STTL 'ASCII CODES USED IN MENSCH ROM'
1024                   .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'ASCII CODES USED IN MENSCH ROM'

```
1025
1026    00:0000         NULL    EQU   0
1027    00:0001         SOH     EQU   01    ;START OF HEADER
1028    00:0002         STX     EQU   02    ;START of TEXT
1029    00:0003         ETX     EQU   03    ;END of TEXT
1030    00:0004         EOT     EQU   04    ;END OF TRANSMISSION
1031    00:0005         ENQ     EQU   05    ;ENQUIRE
1032    00:0006         ACK     EQU   06    ;ACKNOWLEDGEMENT
1033    00:0007         BELL    EQU   07
1034
1035    00:0008         BKSP    EQU   08    ;BACKSPACE
1036    00:0009         HTAB    EQU   09    ;HORIZONTAL TAB
1037    00:000A         L_FEED  EQU   $0A   ;LINE FEED
1038    00:000B         VTAB    EQU   $0B   ;VERTICAL TAB
1039    00:000C         FORM_FEED EQU $0C
1040    00:000D         C_RETURN EQU  $0D
1041    00:000E         SO      EQU   $0E   ;SHIFT_OUT
1042    00:000F         SI      EQU   $0F   ;SHIFT IN
1043
1044    00:0011         DC1     EQU   $11   ;DEVICE CONTROL 1
1045    00:0012         DC2     EQU   $12   ;DEVICE CONTROL 2
1046    00:0013         DC3     EQU   $13   ;DEVICE CONTROL 3
1047    00:0014         DC4     EQU   $14   ;DEVICE CONTROL 4
1048    00:0015         NAK     EQU   $15   ;NEGATIVE ACKNOWLEDGEMENT
1049    00:0016         SYN     EQU   $16   ;SYNC CHARACTER
1050    00:0017         ETB     EQU   $17   ;END of TRANSMISSION BLOCK
1051
1052    00:0018         CAN     EQU   $18   ;CANCEL
1053    00:0019         EM      EQU   $19   ;END of MESSAGE
1054    00:001A         SUB     EQU   $1A   ;SUBSITUTE
1055    00:001B         ESC     EQU   $1B   ;ESCAPE
1056    00:001C         FS      EQU   $1C   ;FIELD SEPARATOR
1057    00:001D         GS      EQU   $1D   ;GROUP SEPARATOR
1058    00:001E         RS      EQU   $1E   ;RECORD SEPARATOR
1059    00:001F         US      EQU   $1F   ;UNIT SEPARATOR
1060
1061
1062    00:0011         XON     EQU   DC1   ;PUNCH (SEND) ON
1063    00:0013         XOFF    EQU   DC3   ;PUNCH (SEND) OFF
1064
1065                    .END
1066
1067
1068
1069                    LONGA OFF
```

```
1070                    LONGI ON
1071
1072                    .CODE
1073
1074    [01]            .IFZ IROM
1075                    .ORG $00:8000
1076                    .BYTE 'WDC',0
1077                     LDA #$89      ;ENABLE EXT ROM, ICE & EXT BUS
1078                     STA BCR
1079                     LDA #$13      ;ENABLE CS4, CS1 & CS0
1080                     STA PCS7
1081
```

'MENSCH COMPUTER ROM SOFTWARE'
'ASCII CODES USED IN MENSCH ROM'

```
1082                            JMP RESET
1083
1084          [00]              .ENDIF
1085
1086  00:E000                           .org $00:E000
1087
1088                            .sttl 'E VECTORS'
1089  00:0000                           include e_vector.h
1090                        ;FILE:E_VECTOR
1091
1092                        ;DATE:01-05-95
1093
1094
1095
1096
1097  00:E000  4C 96 EB                 JMP       Alter_Memory
1098
1099  00:E003  4C EA F1                 JMP       BACKSPACE
1100
1101  00:E006  4C 06 E0                 JMP       *
1102
1103  00:E009  4C 23 F4                 JMP       CONTROL_TONES
1104
1105  00:E00C  4C 61 E6                 JMP       DO_LOW_PWR_PGM
1106
1107  00:E00F  4C 84 E7                 JMP       DUMPREGS
1108
1109  00:E012  4C 9A EC                 JMP       DumpS28
1110
1111  00:E015  4C 73 EC                 JMP       Dump_1_line_to_Output
1112
1113  00:E018  4C 93 EC                 JMP       Dump_1_line_to_Screen
1114
1115  00:E01B  4C 6B EC                 JMP       Dump_to_Output
1116
1117  00:E01E  4C 5C EC                 JMP       Dump_to_Printer
1118
1119  00:E021  4C 85 EC                 JMP       Dump_to_Screen
1120
1121  00:E024  4C 8C EC                 JMP       Dump_to_Screen_ASCII
1122
1123  00:E027  4C 53 ED                 JMP       Dump_It
1124
1125  00:E02A  4C 1E EB                 JMP       FILL_Memory
1126
```

```
1127  00:E02D  4C A0 F2              JMP     GET_3BYTE_ADDR
1128
1129  00:E030  4C EE F5              JMP     GET_ALARM_STATUS
1130
1131  00:E033  4C F9 FB              JMP     GET_BYTE_FROM_PC
1132
1133  00:E036  4C B2 F1              JMP     GET_CHR
1134
1135  00:E039  4C 7D F0                              JMP     GET_HEX
1136
1137  00:E03C  4C C6 F1              JMP     GET_PUT_CHR
1138
```

'MENSCH COMPUTER ROM SOFTWARE'
'E VECTORS'

```
1139  00:E03F  4C D7 F3          JMP     GET_STR
1140
1141  00:E042  4C 04 F2          JMP     Get_Address
1142
1143  00:E045  4C 8C F2          JMP     Get_E_Address
1144
1145  00:E048  4C 6F F2          JMP     Get_S_Address
1146
1147  00:E04B  4C BC F1          JMP     PUT_CHR
1148
1149  00:E04E  4C A1 F3          JMP     PUT_STR
1150
1151  00:E051  4C 1E F6          JMP     READ_ALARM
1152
1153  00:E054  4C E7 F7          JMP     READ_DATE
1154
1155  00:E057  4C 1E F8          JMP     READ_TIME
1156
1157  00:E05A  4C C7 F5          JMP     RESET_ALARM
1158
1159  00:E05D  4C D6 E6          JMP     SBREAK
1160
1161  00:E060  4C 21 FE          JMP     SELECT_COMMON_BAUD_RATE
1162
1163  00:E063  4C BC FD          JMP     SEND_BYTE_TO_PC
1164
1165  00:E066  4C D6 F1          JMP     SEND_CR
1166
1167  00:E069  4C DF F1          JMP     SEND_SPACE
1168
1169  00:E06C  4C 70 F3          JMP     SEND_HEX_OUT
1170
1171  00:E06F  4C 75 F5          JMP     SET_ALARM
1172
1173  00:E072  4C 09 EB          JMP     SET_Breakpoint
1174
1175  00:E075  4C 95 F4          JMP     SET_DATE
1176
1177  00:E078  4C 28 F5          JMP     SET_TIME
1178
1179  00:E07B  4C 55 E6          JMP     VERSION
1180
1181  00:E07E  4C 51 F3          JMP     WR_3_ADDRESS
1182
1183  00:E081  4C C3 E9          JMP     XS28IN
```

```
1184
1185  00:E084  4C B0 E0              JMP      RESET
1186                      ;*********
1187
1188
1189  00:E087  20 BF F0              JSR      ASCBIN
1190  00:E08A  6B                    RTL
1191
1192  00:E08B  20 29 F1              JSR      BIN2DEC
1193  00:E08E  6B                    RTL
1194
1195  00:E08F  20 D7 F0              JSR      BINASC
```

'MENSCH COMPUTER ROM SOFTWARE'
'E VECTORS'

```
1196  00:E092  6B                RTL
1197
1198  00:E093  20 F0 F0           JSR     HEXIN
1199  00:E096  6B                RTL
1200
1201  00:E097  20 16 F1           JSR     IFASC
1202  00:E09A  6B                RTL
1203
1204  00:E09B  20 0D F1           JSR     ISDECIMAL
1205  00:E09E  6B                RTL
1206
1207  00:E09F  20 03 F1           JSR     ISHEX
1208  00:E0A2  6B                RTL
1209
1210  00:E0A3  20 1D F1           JSR     UPPER_CASE
1211  00:E0A6  6B                RTL
1212
1213
1214
1215  00:E0A7                    .DS 8   ;RESERVED FOR EXPANSION
1216
1217
1218
1219                       .sttl 'initialization routine'
1220                       .page
```

'MENSCH COMPUTER ROM SOFTWARE'
'initialization routine'

```
1221  00:E0AF                         include rom_init.asm
1222                        ;FILE:  'ROM_INIT.ASM - Initial Code  MENSCH'
1223                        ;DATE: 02-06-1995
1224
1225
1226
1227                        .LONGA OFF
1228                        .LONGI OFF
1229  00:E0B0                           .ORG $00:E0B0
1230
1231        00:E0B0               RESET    .EQU *
1232
1233        [01]                .IFNZ IROM
1234
1235  00:E0B0  A9 01                    LDA #$01      ;ENABLE EXT BUS          ONLY
1236  00:E0B2  8D 40 DF                 STA BCR
1237  00:E0B5  80 52                    BRA START
1238
1239        [01]                .ELSE
1240
1241                            LDA #$C9      ;ENABLE EXT ROM, NMI, ICE & EXT BUS
1242                            STA BCR
1243                            LDA #$3B      ;ENABLE CS5, CS4, CS3, CS1 & CS0
1244                            STA PCS7
1245                            BRA START
1246
1247        [00]                .ENDIF
1248
1249
1250  00:E0B7                       MONVER:
1251  00:E0B7  4D 45 4E 53 43                .BYTE 'MENSCH ROM Version '
          48 20 52 4F 4D
          20 56 65 72 73
          69 6F 6E 20
1252  00:E0CA  32 2E 30 37 0D          MONVRS   .BYTE '2.07',$0D
1253  00:E0CF  20 28 43 29 20                .BYTE ' (C) Copyright 1995',$0D
          43 6F 70 79 72
          69 67 68 74 20
          31 39 39 35 0D
1254  00:E0E3  41 73 73 65 6D                   .BYTE 'Assembled '
          62 6C 65 64 20
1255  00:E0ED  4D 6F 6E 20 46          MONDATE  DATE
          65 62 20 20 36
          20 31 30 3A 30
          33 3A 34 32 20
```

```
          31 39 39 35 00
1256  00:E106                   MONVEND:
1257
1258  00:E106  57 44 43         WDC      .byte 'WDC'
1259
1260
1261
1262                      ;*  START  is the normal restart point for the
1263                      ;*    Mensch computer software. All previous
1264                      ;*    setup conditions are lost.
1265                      ;*
1266
```

'MENSCH COMPUTER ROM SOFTWARE'
'initialization routine'

```
1267
1268        00:E109              START    EQU *
1269  00:E109  78                        SEI
1270  00:E10A  18                        CLC         ;GOTO 65C265 NATIVE MODE
1271  00:E10B  FB                        XCE
1272  00:E10C  C2 18                     REP #X8+$08    ;SET Index & CLEAR DECIMAL MODE (CLD)
1273
1274                       .LONGI ON
1275
1276  00:E10E  A2 FF 01                  LDX #$01FF     ;SETUP  STACK
1277  00:E111  9A                        TXS          ;CHANGE STACK
1278
1279        [01]                 IFNZ IROM
1280
1281  00:E112  A9 20                     LDA #$20      ;ENABLE EXTERNAL RAM $40
1282                                      ;BIT 5-PCS75  SEL 4Meg ie BANKS 00-3F
1283  00:E114  8D 27 DF                 STA PCS7
1284                         CHKAGAIN:                ;CHECK PCMCIA CARD
1285  00:E117  A2 00 00                  LDX #$00
1286  00:E11A  BD 00 80          CHKPCMLP LDA !$8000,X  ;CHK FOR EXTERNAL ROM
1287  00:E11D  DD 06 E1                  CMP !WDC,X
1288  00:E120  D0 09                     BNE CHKLROM
1289  00:E122  E8                        INX
1290  00:E123  E0 03 00                  CPX #3
1291  00:E126  D0 F2                     BNE CHKPCMLP
1292  00:E128  4C 04 80                  JMP $8004      ;JMP TO EXTERNAL ROM
1293
1294
1295
1296  00:E12B                    CHKLROM:
1297  00:E12B  A9 10                     LDA #Bit4      ;HAVE WE CHECKED 00:8000  CS4 ?
1298  00:E12D  0C 27 DF                  TSB PCS7
1299  00:E130  F0 E5                     BEQ CHKAGAIN   ;NO
1300
1301                                      ;ENABLE $8000 & RAM ($0200-$7FFF)
1302                                      ;BIT 0-PCS70  SEL PORT REPLACEMENT
1303                                      ;BIT 3-PCS73  SEL 000200-007FFF  'CACHE' MEMORY
1304                                      ;BIT 4-PCS74  SEL 8000-DEFF & E000-FFFF ROM
1305                                      ;BIT 5-PCS75  SEL 4Meg ie BANKS 00-3F
1306                                      ;SO WE CAN CK THE 'WDC'
1307
1308  00:E132  A9 08                     LDA #Bit3
1309
1310        [01]                 .ELSE
1311
```

```
1312                            LDA #Bit4+Bit3
1313
1314        [00]            .ENDIF
1315
1316  00:E134  8D 27 DF             STA PCS7
1317
1318                    CHKRAM:             ;NOW WE CHECK 00:0800  CS3
1319  00:E137  A2 00 00             LDX #$00
1320
1321  00:E13A  BD 00 08       CHKRAM1  LDA !$800,X     ;CHK FOR EXTERNAL RAM
1322  00:E13D  DD 06 E1             CMP !WDC,X
1323  00:E140  D0 09                BNE NOEXTMEM
```

```
     'MENSCH COMPUTER ROM SOFTWARE'
     'initialization routine'

 1324  00:E142  E8                 INX
 1325  00:E143  E0 03 00           CPX #3
 1326  00:E146  D0 F2              BNE CHKRAM1
 1327  00:E148  4C 04 08           JMP $804      ;JMP TO EXTERNAL RAM
 1328
 1329
 1330                    ;
 1331                    ;      Reset all regs to reset values (in case we had a JMP reset rather
 1332                    ;      than a hard reset.  Exception is TCR1 because the chip will die if
 1333                    ;      we switch to slow clock and shut off fast clock simultaniously,
 1334
 1335  00:E14B                 NOEXTMEM:
 1336
 1337  00:E14B  A9 FB            LDA #$FB      ;ENABLE $8000 & RAM ($0200-$7FFF)
 1338                                    ;BIT 0-PCS70  SEL DF00-DF1F VIA1
 1339                                    ;BIT 1-PCS71  SEL DFC0-DFEF VIA2
 1340                                    ;BIT 2-PCS72  NOT USED
 1341                                     ;BIT 3-PCS73  SEL 000200-007FFF  'CACHE' MEMORY
 1342                                     ;BIT 4-PCS74  SEL 8000-DEFF & E000-FFFF ROM
 1343                                     ;BIT 5-PCS75  SEL 4Meg ie BANKS 00-3F
 1344                                    ;BIT 6-PCS76  SEL 8Meg ie BANKS 40-BF
 1345                                    ;BIT 7-PCS77  SEL 4Meg ie BANKS CO-FF
 1346  00:E14D  8D 27 DF           STA PCS7
 1347                                     ;  START FAST CLOCK BUT DO NOT USE YET
 1348  00:E150  A9 F9              LDA #$F9     ;AND HAVE ALL CHIP SELECTS AS 1 TO 1
DIVIDE
 1349  00:E152  8D 41 DF           STA SSCR
 1350
 1351  00:E155  A9 C0              LDA #$C0
 1352  00:E157  8D 24 DF           STA PDD4
 1353  00:E15A  A9 80              LDA #Bit7
 1354  00:E15C  8D 20 DF           STA PD4      ;BIT 6 = RESET*,  BIT 7 = RESET
 1355  00:E15F  EA                 NOP
 1356  00:E160  EA                 NOP
 1357  00:E161  A9 40              LDA #Bit6    ;END RESET OF PERIPHERALS
 1358  00:E163  8D 20 DF           STA PD4
 1359
 1360  00:E166  A9 55              LDA #Bit0+Bit2+Bit4+Bit6
 1361  00:E168  8D 25 DF           STA PDD5
 1362  00:E16B  9C 21 DF           STZ PD5      ;ALL DTR'S TO BE LOW UNTIL PWR ON
 1363
 1364  00:E16E  A9 AA              LDA #Bit1+Bit3+Bit5+Bit7
 1365  00:E170  8D 26 DF           STA PDD6     ;PLACE TXD'S IN OUTPUT MODE
 1366  00:E173  9C 22 DF           STZ PD6      ;ALL TXD'S TO BE LOW UNTIL PWR ON
 1367
```

```
1368
1369  00:E176  A9 FF              LDA #$FF
1370  00:E178  8D 23 DF           STA PD7
1371
1372  00:E17B  9C B8 DF           STZ PD_TIMER        ;TIMES OUT FOR POWER DOWN
1373  00:E17E  9C B9 DF           STZ PD_TIMER+1
1374
1375  00:E181  9C 43 DF           STZ TER
1376  00:E184  9C 44 DF           STZ TIFR
1377  00:E187  9C 46 DF           STZ TIER
1378  00:E18A  9C 42 DF           STZ TCR
1379  00:E18D  9C 49 DF           STZ UIER
1380  00:E190  9C 48 DF           STZ UIFR
```

'MENSCH COMPUTER ROM SOFTWARE'
'initialization routine'

```
1381  00:E193  9C 45 DF          STZ EIFR
1382  00:E196  9C 8F DF          STZ SB_SENTL
1383
1384  00:E199  9C 47 DF          STZ EIER
1385  00:E19C
1386  00:E19C  A2 00 00             LDX #0              ;RESET PWR DOWN SENTINALS
1387  00:E19F  8E B3 DF          STX PWD_CELLS
1388  00:E1A2  80 3E                BRA MONSUP1
1389
1390
1391        00:E1A4          MXTALCALC EQU *       ;MAX VALUE OF T1CL ALLOWABLE
1392  00:E1A4  97                   .BYTE $97     ;1.8432MHZ      X=0
1393  00:E1A5  F2                   .BYTE $F2     ;2.4576MHZ      X=1
1394  00:E1A6  4D                   .BYTE $4D     ;3.6864MHZ      X=2
1395  00:E1A7  7B                   .BYTE $7B     ;4.9152MHZ      X=3
1396  00:E1A8  12                   .BYTE $12     ;6.1440MHZ      X=4
1397
1398        00:E1A9          MINXTALCALC EQU *     ;MIN VALUE OF T1CL ALLOWABLE
1399  00:E1A9  92                   .BYTE $92     ;1.8432MHZ
1400  00:E1AA  EE                   .BYTE $EE     ;2.4576MHZ
1401  00:E1AB  48                   .BYTE $48     ;3.6864MHZ
1402  00:E1AC  77                   .BYTE $77     ;4.9152MHZ
1403  00:E1AD  0A                   .BYTE $0A     ;6.1440MHZ
1404        00:E1AE          MXTLEND  EQU *
1405
1406
1407
1408        00:E1AE          MONIRQTBL EQU *       ;USER INTERRUPTS
1409  00:E1AE  5C D6 E6 00        JML SBREAK       ;UBRK
1410  00:E1B2  5C E4 E6 00        JML NMIBRK       ;UNMI
1411  00:E1B6  5C B0 E0 00        JML RESET       ;UIRQ
1412  00:E1BA  5C B0 E0 00        JML RESET       ;USER CO-PROCESSOR
1413  00:E1BE  5C B0 E0 00           JML RESET       ;ABORT INTERRUPT
1414  00:E1C2  5C B0 E0 00        JML RESET       ;PIBIRQ
1415  00:E1C6  5C B0 E0 00        JML RESET       ;EDGEIRQS
1416  00:E1CA  5C B0 E0 00        JML RESET       ;UIRQT7
1417  00:E1CE  5C B0 E0 00        JML RESET       ;UIRQT2
1418  00:E1D2  5C 9F F6 00        JML TODIRQ       ;UIRQT1
1419  00:E1D6  5C B0 E0 00        JML RESET       ;UIRQT0
1420  00:E1DA  5C B0 E0 00        JML RESET       ;USER COMMAND
1421  00:E1DE  5C B0 E0 00        JML RESET       ;USER SUPPLIED RESTART
1422        00:E1E2          MONIRQEND EQU *
1423
1424                    ; UALRMIRQ DOESNOT GET INITIALIZED BY THIS ROUTINE
1425                    ;  ITS A USER PROBLEM!
```

```
1426
1427          00:E1E2               MONSUP1  EQU *
1428
1429  00:E1E2  C2 10                     REP #X8
1430                              .LONGI ON
1431
1432  00:E1E4  A2 00 01                  LDX #$100     ;WAIT FOR FCLOCK TO START
1433  00:E1E7  CA                 DLY0     DEX
1434  00:E1E8  D0 FD                     BNE DLY0
1435  00:E1EA  A9 0A                     LDA #$02+$08   ;ENABLE FAST CLOCK
1436  00:E1EC  0C 41 DF                  TSB SSCR
1437
```

'MENSCH COMPUTER ROM SOFTWARE'
'initialization routine'

```
1438  00:E1EF  A2 00 40          LDX #$4000
1439  00:E1F2  CA            DLY1  DEX        ;ALLOW TIME TO STABILIZE ON FCLOCK
1440  00:E1F3  D0 FD              BNE DLY1
1441
1442                       ;SETUP TIMER #1 SO WE CAN USE IT TO DETERMINE
1443                       ;THE FAST CLOCK SPEED
1444
1445  00:E1F5  A9 FF              LDA #<32767   ;SET TIMER 1 FOR
1446  00:E1F7  8D 62 DF           STA T1CL      ;1 SECOND IRQ
1447  00:E1FA  A9 7F              LDA #>32767
1448  00:E1FC  8D 63 DF           STA T1CH
1449  00:E1FF  A9 02              LDA #T1FLG    ;ENABLE TIMER 1 IRQS
1450  00:E201  0C 43 DF           TSB TER
1451  00:E204  0C 46 DF           TSB TIER
1452
1453
1454  00:E207  A2 00 10           LDX #$1000    ;CK MAIN XTAL SPEED
1455  00:E20A
1456  00:E20A  AD 63 DF      T1ZERO  LDA T1CH
1457  00:E20D  0D 62 DF           ORA T1CL      ;WAIT UNTIL TOD CLOCK
1458  00:E210  D0 F8              BNE T1ZERO    ;READY TO LOAD
1459  00:E212
1460  00:E212  CA            T1DELAY  DEX       ;NOW WAIT A PREDETERMINED
1461  00:E213  D0 FD              BNE T1DELAY   ;AMT OF TIME TO CALC XTAL
1462
1463  00:E215  E2 10              SEP #X8
1464                        .LONGI OFF
1465
1466  00:E217  AD 62 DF           LDA T1CL      ;CK A RANGE OF #'S
1467  00:E21A
1468  00:E21A  A2 05              LDX #MINXTALCALC-MXTALCALC
1469  00:E21C  DD A3 E1      TRYMXTAL CMP !MXTALCALC-1,X
1470  00:E21F  90 03              BLT TRYMINXTAL
1471  00:E221  CA                 DEX
1472  00:E222  D0 F8              BNE TRYMXTAL
1473
1474  00:E224  DD A8 E1      TRYMINXTAL CMP !MINXTALCALC-1,X
1475  00:E227  B0 05              BGE MXTALFND
1476  00:E229  CA                 DEX
1477  00:E22A  D0 F0              BNE TRYMXTAL
1478
1479  00:E22C  A2 03              LDX #3        ;DEFAULT 3.6864 MHZ
1480
1481        00:E22E          MXTALFND EQU *
1482  00:E22E  CA                 DEX
```

```
1483  00:E22F  8E B5 DF              STX SPEED     ;SAVE MAIN XTAL SPEED
1484
1485  00:E232  C2 10                 REP #X8
1486                              .LONGI ON
1487
1488
1489                              ;SETUP ALL COMMON USER INTERRUPTS
1490                              ;IN LOWER PART OF PAGE 0
1491
1492  00:E234  A2 34 00              LDX #MONIRQEND-MONIRQTBL
1493  00:E237  BD AE E1           FUIRQS   LDA !MONIRQTBL,X
1494  00:E23A  9F 00 01 00           STA UBRK,X
```

'MENSCH COMPUTER ROM SOFTWARE'
'initialization routine'

```
1495  00:E23E  CA                 DEX
1496  00:E23F  10 F6              BPL FUIRQS
1497
1498                        ;SETUP TIME OF DAY CLOCK
1499
1500  00:E241  A2 07 00           LDX #DFLTSEND-DFLTS-1
1501  00:E244  A9 00              LDA #00
1502  00:E246  18                 CLC
1503  00:E247  7D 8F DF      CKTODLP  ADC !DAYWK-1,X    ;CK IF VALID TOD CLOCK
1504  00:E24A  CA                 DEX
1505  00:E24B  D0 FA              BNE CKTODLP
1506  00:E24D  6D 58 DF           ADC T4LL
1507  00:E250  6D 59 DF           ADC T4LH      ;ADD SERIAL BAUD RATE CLOCK
1508  00:E253
1509  00:E253  49 FF              EOR #$FF
1510  00:E255  CF A3 DF 00        CMP TODCKS
1511  00:E259  F0 1B              BEQ GDTOD
1512
1513                        ;USE DEFAULT SETTING IF CHKSUM IS NG
1514
1515  00:E25B  A2 07 00           LDX #DFLTSEND-DFLTS-1 ;LOAD TOD DEFAULT
1516  00:E25E  BD 4A FF      ICLK1    LDA !DFLTS-1,X
1517
1518  00:E261  9D 8F DF           STA !DAYWK-1,X
1519  00:E264  CA                 DEX
1520  00:E265  D0 F7              BNE ICLK1
1521  00:E267  A2 07 00           LDX #DFLTSEND-DFLTS-1 ;RESET ALARM CLOCK ALSO
1522  00:E26A  9E 96 DF      ICLK2    STZ !ADAYWK-1,X
1523  00:E26D  CA                 DEX
1524  00:E26E  D0 FA              BNE ICLK2
1525
1526  00:E270  20 7F F6           JSR CLOCK_CK_SUM  ;MAKE NEW CLOCK CHECKSUM
1527  00:E273  9C B6 DF           STZ FLAGS     ;CLEAR ALARM FLAGS ON RAM LOSS
1528
1529                              ;SETUP ALL 4 SERIAL PORTS
1530  00:E276  A9 08         GDTOD    LDA #8        ;9600
1531  00:E278  22 21 FE 00        JSL SELECT_COMMON_BAUD_RATE ;INITIALIZE ACIA'S.
0,1,3
1532                              ;hardware handshake is OFF
1533  00:E27C  64 6F              STZ ERRORS
1534  00:E27E  A2 00 00           LDX #0
1535  00:E281  86 83              STX T_TIME
1536
1537  00:E283  A9 FB              LDA #$FF-XONOFLG
1538  00:E285  14 40              TRB SFLAG0     ;reset all of SFLAG except XON/XOFF,
```

```
1539  00:E287  14 41              TRB SFLAG0+1   ;leave it in old state.
1540  00:E289  14 42              TRB SFLAG0+2
1541  00:E28B  14 43              TRB SFLAG0+3
1542
1543  00:E28D  20 71 FE           JSR SIOPORTS
1544
1545  00:E290  A9 08              LDA #RES_COMP  ;WE NEED RESET COMPENSATION
1546  00:E292  0C B6 DF           TSB FLAGS      ;ADD 2 SEC'S TO TOD CLOCK TO
1547                               ;COMPENSATE FOR IRQ DEAD TIME
1548  00:E295
1549
1550  00:E295                RET|TO|MENSCH:
1551
```

'MENSCH COMPUTER ROM SOFTWARE'
'initialization routine'

```
1552  00:E295  A2 00 00          LDX #$00
1553  00:E298  BD F8 DE     CHK_MROM   LDA !$DEF8,X    ;CHK FOR MENSCH EPROM
1554  00:E29B  DD 06 E1          CMP !WDC,X
1555  00:E29E  D0 09             BNE DO_STD
1556  00:E2A0  E8                INX
1557  00:E2A1  E0 03 00          CPX #3
1558  00:E2A4  D0 F2             BNE CHK_MROM
1559  00:E2A6  4C FC DE          JMP $DEFC      ;JMP TO MENSCH COMPUTER EPROM
1560
1561  00:E2A9                DO_STD:
1562
1563  00:E2A9  A2 03 F2             LDX #RTL_EXIT
1564  00:E2AC  86 79             STX CLR_LCD_JMP
1565  00:E2AE  86 7B             STX DISP_LCD_JMP
1566  00:E2B0  86 7D                STX TXT_CUR_JMP
1567  00:E2B2  86 7F             STX SND_BEEP_JMP
1568  00:E2B4  AE 61 E6          LDX DO_LOW_PWR_PGM
1569  00:E2B7  86 81             STX LO_PWR_JMP
1570
1571  00:E2B9  20 A2 F1             JSR SET_GET|PUT_CHR  ;ROM SETUP FOR I/O
1572
1573
1574  00:E2BC  A2 0A 00             LDX #10        ;SET BUFFER SIZES
1575  00:E2BF  86 06                STX SINCNT0
1576  00:E2C1  86 0E                STX SINCNT1
1577  00:E2C3  86 16                STX SINCNT2
1578  00:E2C5  86 1E                STX SINCNT3
1579  00:E2C7  86 26                STX SOUTCNT0
1580  00:E2C9  86 2E                STX SOUTCNT1
1581  00:E2CB  86 36                STX SOUTCNT2
1582  00:E2CD  86 3E                STX SOUTCNT3
1583
1584  00:E2CF  A2 39 01             LDX #ROM_IBUF0    ;LOCATE BUFFERS IN 265 RAM
1585  00:E2D2  86 04                STX SIN_BUF0
1586  00:E2D4  A2 43 01             LDX #ROM_IBUF1
1587  00:E2D7  86 0C                STX SIN_BUF1
1588  00:E2D9  A2 4D 01             LDX #ROM_IBUF2
1589  00:E2DC  86 14                STX SIN_BUF2
1590  00:E2DE  A2 57 01             LDX #ROM_IBUF3
1591  00:E2E1  86 1C                STX SIN_BUF3
1592
1593
1594  00:E2E3  A2 61 01             LDX #ROM_OBUF0   ;LOCATE BUFFERS IN 265 RAM
1595  00:E2E6  86 24                STX SOUTBUF0
1596  00:E2E8  A2 6B 01             LDX #ROM_OBUF1
```

```
1597  00:E2EB  86 2C                STX SOUTBUF1
1598  00:E2ED  A2 75 01             LDX #ROM_OBUF2
1599  00:E2F0  86 34                STX SOUTBUF2
1600  00:E2F2  A2 7F 01             LDX #ROM_OBUF3
1601  00:E2F5  86 3C                STX SOUTBUF3
1602
1603
1604
1605                    ; Initialize Port3 (HOST)
1606
1607  00:E2F7  A2 00 00             LDX #0              ;CLR input buffer
1608  00:E2FA  86 1A                STX SINEND3
```

'MENSCH COMPUTER ROM SOFTWARE'
'initialization routine'

```
1609  00:E2FC  86 18              STX SININDX3
1610  00:E2FE  86 38              STX SOUTINDX3      ;clear output buffer
1611  00:E300  86 3A              STX SOUTEND3
1612  00:E302  A9 04              LDA #Bit2      ;xon-xoff
1613  00:E304  04 43              TSB SFLAG3
1614
1615  00:E306  A9 08              LDA #Bit3      ;PORT 3 OUTPUT
1616  00:E308  85 4E            STA OUTPUT_XTRL
1617
1618  00:E30A  AD E4 DF         LDA SEGA_DATA_REG+4   ;VIA COUNTER
1619  00:E30D  CD E4 DF         CMP SEGA_DATA_REG+4
1620  00:E310  F0 0E            BEQ ?NM
1621  00:E312  A9 08            LDA #Bit3
1622  00:E314  1C E1 DF         TRB PWR_XTRL_REG  ;LOW = ON
1623  00:E317  0C E3 DF         TSB PWR_XTRL_DIR
1624  00:E31A  85 4D            STA INPUT_XTRL
1625
1626  00:E31C  A9 10            LDA #A_MENSCH  ;SET A FLAG FOR PWR DWN
1627  00:E31E  04 53            TSB DISPTYP
1628
1629  00:E320  A9 80         ?NM    LDA #Bit7
1630  00:E322  0C 26 DF         TSB PDD6      ;PUTS TXD IN MARK HOLD
1631  00:E325  0C 22 DF         TSB PD6
1632
1633  00:E328  A9 40              LDA #Bit6      ;set DTR
1634  00:E32A  1C 21 DF           TRB PD5        ;LOW = TRUE
1635
1636  00:E32D  A9 20              LDA #Bit5      ;turn on receive
1637  00:E32F  0C 76 DF           TSB ACSR3
1638  00:E332  A9 C0              LDA #Bit6+Bit7 ;Rec & xmit interrupt
1639  00:E334  0C 49 DF           TSB UIER
1640
1641  00:E337  A9 02            LDA #ALRMIRQ
1642  00:E339  14 53            TRB DISPTYP
1643  00:E33B  D0 0B            BNE ?NA
1644  00:E33D  AF 34 01 00      LDA UALRMIRQ
1645  00:E341  C9 5C            CMP #$5C      ;JUMP LONG IN-PLACE
1646  00:E343  D0 03            BNE ?NA
1647  00:E345  4C 34 01         JMP UALRMIRQ
1648
1649  00:E348  58            ?NA    CLI
1650  00:E349  22 D6 F1 00      JSL SEND_CR
1651
1652  00:E34D  A2 B7 E0         LDX #MONVER
1653  00:E350  A9 00            LDA #0
```

```
1654  00:E352  22 A1 F3 00                JSL PUT_STR
1655
1656  00:E356  00 00                      BRK    ;ENTER MONITOR
1657
1658                                 .END
1659
1660
1661                                 .sttl 'main routine'
1662                              .page
```

'MENSCH COMPUTER ROM SOFTWARE'
'main routine'

```
1663  00:E356                       include r_main.asm
1664                      ;File: R_MAIN.asm
1665                      ;Date: 01-07-95
1666
1667
1668
1669  00:E358                   ROM|START:
1670  00:E358  C2 10                REP #X8         ;SET X & Y Long
1671  00:E35A  E2 20                SEP #M8       ;SET Acc SHORT
1672                      .LONGA OFF
1673                      .LONGI ON
1674
1675
1676  00:E35C  22 D6 F1 00          JSL SEND_CR
1677  00:E360  A9 3E                LDA #'>'      ;TYPE PROMPTING '>'
1678  00:E362  22 BC F1 00          JSL PUT_CHR
1679
1680  00:E366  22 B2 F1 00    S00     JSL GET_CHR
1681  00:E36A  C9 0A                CMP #$0A      ;IGNORE LINE FEED - LEFT OVER
1682  00:E36C  F0 F8                BEQ S00       ;FROM PREVIOUS CR
1683  00:E36E  22 BC F1 00    ?00     JSL PUT_CHR
1684  00:E372  B0 FA                BCS ?00
1685
1686                      ;
1687                      ;       Raw character is in A.  May be wrong case, etc.  We will JSR to
1688                      ;       alt. parsing if it exists at this point.  The alternate parser
1689                      ;       will jump through the vector table to START if it completes the
1690                      ;       command, and will do an RTS to the regular parser if it does not
1691                      ;       have the command in its table.
1692
1693  00:E374                     S0:
1694  00:E374  C2 10                REP #X8         ;SET X & Y Long
1695  00:E376  E2 20                SEP #M8       ;SET Acc SHORT
1696                      .LONGA OFF
1697                      .LONGI ON
1698
1699  00:E378                   DFLTPRSR:
1700  00:E378  20 1D F1             JSR UPPER_CASE ;IN ACC/ MAKE SURE
1701                                 ;UPPERCASE
1702  00:E37B  A2 15 00             LDX #ADRS-CMDS-1     ;LENGTH OF CMD TABLE
1703  00:E37E  DD B4 E3       S1      CMP !CMDS,X
1704  00:E381  F0 0B                BEQ S2
1705  00:E383  CA                   DEX
1706  00:E384  10 F8                BPL S1        ;LOOP FOR ALL CMDS
1707
```

```
1708  00:E386  A9 3F              LDA #'?'      ;OPERATOR ERR, TYPE'?'
1709  00:E388  22 BC F1 00        JSL PUT_CHR
1710  00:E38C  80 0F              BRA RDY       ;GOTO READY
1711
1712  00:E38E  C2 20         S2   REP #M8
1713                        LONGA ON
1714
1715  00:E390  8A                 TXA
1716  00:E391  0A                 ASL A         ;X2
1717  00:E392  AA                 TAX
1718
1719  00:E393  E2 20                SEP #M8
```

'MENSCH COMPUTER ROM SOFTWARE'
'main routine'

```
1720                            LONGA OFF
1721
1722  00:E395  22 DB F1 00            JSL SEND_SPACE2
1723  00:E399  22 B1 E3 00               JSL IJMP
1724  00:E39D  A9 00            RDY     LDA #0
1725  00:E39F  A2 A8 E3                 LDX #Ready_Now
1726  00:E3A2  22 A1 F3 00            JSL PUT_STR
1727  00:E3A6  80 BE                     BRA S00
1728
1729
1730  00:E3A8  0D 52 45 41 44          Ready_Now .byte $0D,'READY',$0D,'>',00
          59 0D 3E 00
1731
1732
1733  00:E3B1  7C CA E3       IJMP    JMP (ADRS,X)
1734
1735
1736                            .STTL 'R_MAIN.ASM - Command Tables/pointers'
1737                            .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_MAIN.ASM - Command Tables/pointers'

```
1738
1739                    ;COMMANDS USED
1740                    ;A,B,D,F,G,H,J,M,N
1741                    ;R,S,T,U,W,X
1742                    ;<,>, ,?,/,|
1743
1744
1745                    ;COMMANDS NOT YET USED
1746                    ;C,E,I,K,L,O,P,Q,V,Y,Z
1747                    ;+,-,!,@,%,^,&,[,],\,{,},~,',",`
1748
1749
1750
1751
1752  00:E3B4  41             CMDS    .BYTE 'A'     ;ALTER REGISTERS
1753  00:E3B5  4D                     .BYTE 'M'     ;CHANGE A MEMORY LOC
1754  00:E3B6  3C                     .BYTE '<'     ;DEC TO NXT MEMORY LOC
1755  00:E3B7  3E                     .BYTE '>'     ;INC TO NXT MEMORY LOC
1756  00:E3B8  20                     .BYTE ' '     ;REDISPLAY OLD LOCATION
1757
1758  00:E3B9  52                     .BYTE 'R'     ;DISPLAY REGISTERS
1759  00:E3BA  47                     .BYTE 'G'     ;GO/JML
1760  00:E3BB  4A                     .BYTE 'J'     ;JSL
1761  00:E3BC  44                     .BYTE 'D'     ;DUMP MEMORY IN HEX
1762  00:E3BD  46                     .BYTE 'F'     ;FILL MEMORY
1763
1764  00:E3BE  3F                     .BYTE '?'     ;HELP MENU
1765  00:E3BF  48                     .BYTE 'H'     ;HELP MENU
1766  00:E3C0  54                     .BYTE 'T'     ;DISPLAY AND/OR MODIFY TIME
1767  00:E3C1  4E                     .BYTE 'N'     ;DISPLAY AND/OR MODIFY DATE
1768  00:E3C2  53                     .BYTE 'S'     ;S28 LOADER FROM MONITOR
1769
1770  00:E3C3  57                     .BYTE 'W'     ;S28 DUMPER
1771  00:E3C4  58                     .BYTE 'X'     ;POWER DOWN
1772  00:E3C5  2F                     .BYTE '/'     ;QUICK ACCESS TO MEM FOR HOSTS
1773  00:E3C6  7C                     .BYTE '|'     ;QUICK ACCESS TO REGISTERS FOR HOSTS
1774  00:E3C7  55                     .BYTE 'U'     ;USER COMMAND
1775  00:E3C8  42                     .BYTE 'B'     ;SET BREAKPOINT
1776
1777  00:E3C9  2A                     .BYTE '*'     ;returnto mensch computer
1778
1779
1780  00:E3CA  ACE8           ADRS    .WORD ALTER_REGS     ;CHANGE CURRENT REGS
1781  00:E3CC  96EB                   .WORD Alter_Memory   ;ALTER MEMORY LOCATIONS
1782  00:E3CE  68F0                   .WORD DSPLYDEC       ;DEC ADDR & DISPLAY
```

```
1783  00:E3D0  6DF0              .WORD DSPLYINC      ;INC ADDR & DISPLAY
1784  00:E3D2  70F0              .WORD DSPLYOLD      ;DISPLAY CURRENT ADDR
1785
1786  00:E3D4  84E7              .WORD DUMPREGS      ;DISPLAY REGS
1787  00:E3D6  B6E5              .WORD GO_JML        ;GO/JML
1788  00:E3D8  A1E5              .WORD GO_JSL        ;JSL
1789  00:E3DA  6BEC              .WORD Dump_to_Output ;DUMP MEMORY IN HEX
1790  00:E3DC  1EEB              .WORD FILL_Memory   ;FILL MEMORY WITH A CONSTANT

1791
1792  00:E3DE  F6E3              .WORD HELP          ;HELP MENU
1793  00:E3E0  F6E3              .WORD HELP          ;HELP MENU
1794  00:E3E2  FBE5              .WORD DTIME         ;DISPLAY TIME OF DAY
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_MAIN.ASM - Command Tables/pointers'

```
1795  00:E3E4  28E6              .WORD DDATE         ;DISPLAY DATE
1796  00:E3E6  F9E9              .WORD XS28ROM       ;MOTOROLA S28 LOADER FROM
MONITOR
1797
1798  00:E3E8  9AEC              .WORD DumpS28       ;MOTOROLA S28 DUMP
1799  00:E3EA  5EE6              .WORD ENTER_LOW_POWER_MODE
1800  00:E3EC  A1EE              .WORD SLASH         ;HOST MEMORY ACCESS
1801  00:E3EE  76EF              .WORD PIPE          ;HOST REGISTER ACCESS
1802  00:E3F0  2C01              .WORD USER_CMD      ;USER COMMAND
1803  00:E3F2  09EB              .WORD SET_Breakpoint
1804
1805  00:E3F4  95E2              .WORD RET|TO|MENSCH
1806
1807
1808                       .STTL 'DBGSPCL.ASM - 65C816 Display All Registers'
1809                       .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'DBGSPCL.ASM - 65C816 Display All Registers'

```
1810
1811
1812
1813                ;       8 bits
1814                ; _____________________
1815                ;|                     |
1816                ;|  Data Bank Reg      |
1817                ;|     (DBR)           |
1818                ;|_____________________|
1819
1820
1821                ;                       8 bits                8 bits
1822                ;                   _____________________  _____________________
1823                ;                  |                     ||                     |
1824                ;                  |   X Reg High        ||    X Reg Low        |
1825                ;                  |     (XH)            ||      (XL)           |
1826                ;                  |_____________________||_____________________|
1827
1828
1829                ;                   _____________________  _____________________
1830                ;                  |                     ||                     |
1831                ;                  |   Y Reg High        ||    Y Reg Low        |
1832                ;                  |     (YH)            ||      (YL)           |
1833                ;                  |_____________________||_____________________|
1834
1835
1836                ; _____________________  _____________________  _____________________
1837                ;|                     ||                     ||                     |
1838                ;|                     || Stack Reg High      ||  Stack Reg Low      |
1839                ;|       00            ||      (SH)           ||      (SL)           |
1840                ;|_____________________||_____________________||_____________________|
1841
1842
1843                ;                   _____________________  _____________________
1844                ;                  |                     ||                     |
1845                ;                  |  Accumulator        ||   Accumulator       |
1846                ;                  |     (B)             ||      (A)            |
1847                ;                  |_____________________||_____________________|
1848
1849
1850
1851                ; _____________________  _____________________  _____________________
1852                ;|                     ||                     ||                     |
1853                ; | Program Bank Reg    ||  Program Cntr Hi    ||  Program Cntr Low   |
1854                ;|     (PBR)           ||     (PCH)           ||      (PCL)          |
```

```
                        ; |_____________________|    |_____________________|    |_____________________|
                        ;
00:E3F6                 ;
                        ; _____________________    _____________________    _____________________
                        ; |                   |    |                   |    |                   |
                        ; |                   |    | Direct Reg High   |    | Direct Reg Low    |
                        ; |        00         |    |       (DH)        |    |       (DL)        |
                        ; |___________________|    |___________________|    |___________________|
                        ;
                        ; _____________________
                        ; |                   |
```

'MENSCH COMPUTER ROM SOFTWARE'
'DBGSPCL.ASM - 65C816 Display All Registers'

```
1867                    ; |   Status Reg    |
1868                    ; |     (FLAGS)     |
1869                    ; |_________________|
1870
1871                    ;  N  V  M  X  D  I  Z  C
1872                    ;  |  |  |  |  |  |  |  |_Carry            1 = True
1873                    ;  |  |  |  |  |  |  |____Zero             1 = Result Zero
1874                    ;  |  |  |  |  |  |_______IRQ Disable      1 = Disable
1875                    ;  |  |  |  |  |__________Decimal Mode     1 = True
1876                    ;  |  |  |  |_____________Index Reg Select  1 = 8 bit  0 = 16 bit mode
1877                    ;  |  |  |________________Memory Select     1 = 8 bit  0 = 16 bit mode
1878                    ;  |  |___________________Overflow          1 = True
1879                    ;  |______________________Negative          1 = Negative
1880
1881
1882                    ;*******************
1883                    ;*
1884                    ;NEW FORM
1885                    ;
1886                    ;    PCntr   Acc    X      Y      S      Dir    F   B
1887                    ;    00:8000 00 EA  00 01  00 00  01 FD  00 00  4F  00
1888                    ;    Status Reg
1889                    ;    N  V  M  X  D  I  Z  C
1890                    ;    0  1  0  0  1  1  1  1
1891
1892
1893                     .STTL 'R_MAIN .. HELP LIST'
1894                     .PAGE
```

```
    'MENSCH COMPUTER ROM SOFTWARE'
    'R_MAIN .. HELP LIST'

1895
1896
1897
1898  00:E3F6                   HELP:
1899
1900  00:E3F6  A9 00                  LDA #0       ;BANK
1901  00:E3F8  A2 04 E4               LDX #HELPMENU
1902  00:E3FB  22 A1 F3 00            JSL PUT_STR   ;WILL RETURN TO DISPATCHER
1903  00:E3FF  22 D6 F1 00            JSL SEND_CR
1904  00:E403  6B                     RTL
1905
1906
1907
1908  00:E404  0D               HELPMENU .BYTE $0D
1909  00:E405  4D 20 20 20 20                .BYTE 'M      Alter memory',$0D
          20 20 41 6C 74
          65 72 20 6D 65
          6D 6F 72 79 0D
1910  00:E419  53 50 41 43 45          .BYTE 'SPACE  Display memory address',$0D
          20 20 44 69 73
          70 6C 61 79 20
          6D 65 6D 6F 72
          79 20 61 64 64
          72 65 73 73 0D
1911  00:E437  3C 2C 3E 20 20                .BYTE '<,>    Decrement, Increment memory address',$0D
          20 20 44 65 63
          72 65 6D 65 6E
          74 2C 20 49 6E
          63 72 65 6D 65
          6E 74 20 6D 65
          6D 6F 72 79 20
          61 64 64 72 65
          73 73 0D
1912  00:E462  44 20 20 20 20             .BYTE 'D      Dump memory',$0D
          20 20 44 75 6D
          70 20 6D 65 6D
          6F 72 79 0D
1913  00:E475  52 20 20 20 20          .BYTE 'R      Display registers',$0D
          20 20 44 69 73
          70 6C 61 79 20
          72 65 67 69 73
          74 65 72 73 0D
1914  00:E48E  42 20 20 20 20          .BYTE 'B      SET Breakpoint',$0D
          20 20 53 45 54
          20 42 72 65 61
```

```
          6B 70 6F 69 6E
          74 0D
1915  00:E4A4  47 2C 4A 20 20          .BYTE 'G,J    JML, JSL, to PC [location]',$0D
          20 20 4A 4D 4C
          2C 20 4A 53 4C
          2C 20 74 6F 20
          50 43 20 5B 6C
          6F 63 61 74 69
          6F 6E 5D 0D
1916  00:E4C6  46 20 20 20 20          .BYTE 'F      Block Fill',$0D
          20 20 42 6C 6F
          63 6B 20 46 69
```

```
    'MENSCH COMPUTER ROM SOFTWARE'
    'R_MAIN .. HELP LIST'

          6C 6C 0D
1917  00:E4D8  53 2C 57 20 20          .BYTE 'S,W    S28 Input, Output',$0D
          20 20 53 32 38
          20 49 6E 70 75
          74 2C 20 4F 75
          74 70 75 74 0D
1918  00:E4F1  3F 2C 48 20 20          .BYTE '?,H    HELP',$0D
          20 20 48 45 4C
          50 0D
1919  00:E4FD  54 20 20 20 20          .BYTE 'T      Display & Change Time',$0D
          20 20 44 69 73
          70 6C 61 79 20
          26 20 43 68 61
          6E 67 65 20 54
          69 6D 65 0D
1920  00:E51A  4E 20 20 20 20          .BYTE 'N      Display & Change Date',$0D
          20 20 44 69 73
          70 6C 61 79 20
          26 20 43 68 61
          6E 67 65 20 44
          61 74 65 0D
1921  00:E537  58 20 20 20 20       .BYTE 'X      EXIT to Low Power Mode',$0D,$0D
          20 20 45 58 49
          54 20 74 6F 20
          4C 6F 77 20 50
          6F 77 65 72 20
          4D 6F 64 65 0D
          0D
1922  00:E556  2F 20 20 20 20       .BYTE '/      Host memory access',$0D
          20 20 48 6F 73
          74 20 6D 65 6D
          6F 72 79 20 61
          63 63 65 73 73
          0D
1923  00:E570  7C 20 20 20 20          .BYTE '|      Host register access',$0D
          20 20 48 6F 73
          74 20 72 65 67
          69 73 74 65 72
          20 61 63 63 65
          73 73 0D
1924  00:E58C  55 20 20 20 20          .BYTE 'U      USER command',$0D
          20 20 55 53 45
          52 20 63 6F 6D
          6D 61 6E 64 0D
1925  00:E5A0  00                   .BYTE $0      ;string termination
```

```
1926
1927
1928
1929                    .STTL 'R_MAIN .. COMMANDS
1930                    .PAGE
```

```
    'MENSCH COMPUTER ROM SOFTWARE'
    'R_MAIN .. COMMANDS

1931
1932                        GO_JSL:             ;LEAVE RETURN ON STACK
1933  00:E5A1  20 C3 E5            JSR GET_SAVE_PC
1934  00:E5A4  90 01                BCC ?G
1935  00:E5A6  6B                   RTL
1936
1937  00:E5A7  FA               ?G    PLX     ;POP STACK
1938  00:E5A8  68                   PLA
1939  00:E5A9  A9 00                LDA #0  ;SETUP RETURN TO BREAK SOFTWARE
1940  00:E5AB  48                   PHA
1941  00:E5AC  A0 F8 E6             LDY #JSL|RTL_IN-1
1942  00:E5AF  5A                   PHY
1943  00:E5B0  BA                   TSX
1944  00:E5B1  8E 86 DF             STX STK|PTR
1945  00:E5B4  80 0A                BRA go8
1946
1947  00:E5B6                   GO_JML:
1948  00:E5B6  20 C3 E5             JSR GET_SAVE_PC
1949  00:E5B9  90 05                BCC go8
1950
1951  00:E5BB  C9 0D                    CMP #C_RETURN       ;ENTER KEY
1952  00:E5BD  F0 01                BEQ go8
1953  00:E5BF  6B                   RTL
1954
1955  00:E5C0  4C 4E E7         go8     JMP GO_AGAIN            ;RESTART FROM OLD PC VALUE
1956
1957
1958
1959  00:E5C3                   GET_SAVE_PC:
1960  00:E5C3  22 04 F2 00          JSL Get_Address  ;starting address
1961  00:E5C7  B0 0B                    BCS ?X
1962  00:E5C9  A6 63                LDX TMP2
1963  00:E5CB  8E 8C DF             STX PCL
1964  00:E5CE  A5 65                LDA TMP2+2
1965  00:E5D0  8F 8E DF 00          STA TPBR
1966  00:E5D4  60               ?X      RTS
1967
1968  00:E5D5  0D 45 4E 54 45           ENTR_TIME .BYTE $0D,'ENTER NEW TIME  ',$0D,0
          52 20 4E 45 57
          20 54 49 4D 45
          20 20 0D 00
1969  00:E5E8  0D 45 4E 54 45           ENTR_DATE .BYTE $0D,'ENTER NEW DATE  ',$0D,0
          52 20 4E 45 57
          20 44 41 54 45
          20 20 0D 00
```

```
1970
1971
1972  00:E5FB                   DTIME:
1973  00:E5FB  22 D6 F1 00          JSL SEND_CR
1974  00:E5FF  A2 87 00             LDX #OUTBUF
1975  00:E602  22 1E F8 00          JSL READ_TIME
1976  00:E606  A9 00                LDA #0
1977  00:E608  A2 87 00             LDX #OUTBUF
1978  00:E60B  22 A1 F3 00          JSL PUT_STR
1979  00:E60F  A9 00                    LDA #0
1980  00:E611  A2 D5 E5             LDX #ENTR_TIME
1981  00:E614  22 A1 F3 00          JSL PUT_STR
```

```
    'MENSCH COMPUTER ROM SOFTWARE'
    'R_MAIN .. COMMANDS

1982  00:E618  A9 00                LDA #0
1983  00:E61A  A2 8E 01          LDX #STR_BUF
1984  00:E61D  22 D7 F3 00       JSL GET_STR
1985  00:E621  B0 04             BCS ?X
1986  00:E623  22 28 F5 00          JSL SET_TIME
1987
1988  00:E627  6B            ?X   RTL
1989
1990
1991  00:E628                DDATE:
1992  00:E628  22 D6 F1 00       JSL SEND_CR
1993  00:E62C  A2 87 00          LDX #OUTBUF
1994  00:E62F  22 E7 F7 00       JSL READ_DATE
1995  00:E633  A9 00             LDA #0
1996  00:E635  A2 87 00          LDX #OUTBUF
1997  00:E638  22 A1 F3 00       JSL PUT_STR
1998  00:E63C  A9 00                LDA #0
1999  00:E63E  A2 E8 E5          LDX #ENTR_DATE
2000  00:E641  22 A1 F3 00       JSL PUT_STR
2001  00:E645  A9 00                LDA #0
2002  00:E647  A2 8E 01          LDX #STR_BUF
2003  00:E64A  22 D7 F3 00       JSL GET_STR
2004  00:E64E  B0 04             BCS ?X
2005  00:E650  22 95 F4 00          JSL SET_DATE
2006
2007  00:E654  6B            ?X   RTL
2008
2009                  ;************************************************8
2010                  ;*    Version returns pointers to
2011                  ;*       1) a 4 ASCII chr version # in the Xreg
2012                  ;*          ie  "2.01"
2013                  ;*
2014                  ;*       2) a formated ASCII string of the last
2015                  ;*      assembly date in Yreg.
2016                  ;*          ie  "SAT DEC  3 12:16:05 1994"
2017                  ;*
2018                  ;*    The Areg is set to 0 upon return.
2019                  ;*
2020
2021
2022
2023  00:E655             VERSION:
2024  00:E655  A9 00            LDA #0
2025  00:E657  A2 CA E0         LDX #MONVRS
2026  00:E65A  A0 ED E0         LDY #MONDATE
```

```
2027  00:E65D  6B                    RTL
2028
2029
2030
2031                        .END
2032
2033
2034                          .sttl 'powerdown control'
2035                          .page
```

'MENSCH COMPUTER ROM SOFTWARE'
'powerdown control'

```
2036  00:E65D                       include R_pwrdwn.asm
2037                        ; FILE: R_PWRDWN.ASM     power down routine
2038                        ; DATE: 12-02-94
2039
2040
2041  00:E65E                   ENTER_LOW_POWER_MODE:
2042
2043  00:E65E  6C 81 00             JMP (LO_PWR_JMP)
2044
2045                        ;*****************************************
2046
2047
2048  00:E661                   DO_LOW_PWR_PGM:
2049
2050  00:E661  A2 FF 01               LDX #$01FF    ;RESET STACK POINTER
2051  00:E664  9A                   TXS
2052
2053  00:E665  58                   CLI    ;LET THE CLOCK RUN
2054
2055  00:E666  A9 10                LDA #A_MENSCH
2056  00:E668  24 53                BIT DISPTYP
2057  00:E66A  F0 05                BEQ ?1        ;NOT A MENSCH PLATFORM
2058  00:E66C  A9 08                LDA #Bit3     ;TURN OFF HOST PORT PWR
2059  00:E66E  0C E1 DF             TSB PWR_XTRL_REG
2060
2061  00:E671  A9 FF            ?1    LDA #$FF      ;set all busses low
2062  00:E673  8D 04 DF            STA PDD0
2063  00:E676  8D 05 DF            STA PDD1
2064  00:E679  8D 06 DF            STA PDD2
2065  00:E67C  8D 07 DF            STA PDD3
2066  00:E67F  9C 00 FD             STZ PD0
2067  00:E682  9C 01 FD             STZ PD1
2068  00:E685  9C 02 FD             STZ PD2
2069  00:E688  9C 03 FD             STZ PD3
2070
2071        [01]              .IFZ IROM
2072
2073                          LDA #$FF      ;SET ALL EXTERNAL SELECTS HI
2074                          STA PD7
2075                          LDA #Bit4     ;LEAVE EPROM SELECT ONLY
2076                          STA PCS7
2077                          LDA #Bit7+Bit3+Bit0 ;FOR TEST ONLY
2078                          STA BCR
2079
2080        [01]              .ELSE
```

```
2081
2082  00:E68B  A9 FF              LDA #$FF       ;SET ALL EXTERNAL SELECTS HI
2083  00:E68D  8D 23 DF           STA PD7
2084  00:E690  9C 27 DF              STZ PCS7
2085  00:E693  A9 09              LDA #Bit3+Bit0
2086  00:E695  8D 40 DF              STA BCR
2087
2088          [00]            .ENDIF
2089
2090  00:E698  A9 0E              LDA #Bit1+Bit2+Bit3  ;goto slow clock & Internal RAM
2091  00:E69A  1C 41 DF           TRB SSCR
2092  00:E69D  A9 01              LDA #Bit0      ;TURN OFF  FAST CLOCK
```

```
    'MENSCH COMPUTER ROM SOFTWARE'
    'powerdown control'

2093  00:E69F  1C 41 DF          TRB SSCR
2094
2095  00:E6A2  A2 00 00          LDX #0        ;WAIT FOR THINGS TO SETTLE
2096  00:E6A5  E8            ?DLY   INX         ;BEFORE TESTING POWER UP.
2097  00:E6A6  D0 FD             BNE ?DLY
2098
2099
2100
2101  00:E6A8                TRY_RESTART:
2102
2103  00:E6A8  AF B6 DF 00        ?3    LDA FLAGS      ;LOOK FOR REASON TO RESTART
2104  00:E6AC  89 01             BIT #ALRMENAB
2105  00:E6AE  F0 07             BEQ ?4
2106  00:E6B0  89 02             BIT #ALRMIRQ
2107  00:E6B2  F0 03             BEQ ?4
2108  00:E6B4  4C 30 01          JMP URESTART
2109
2110
2111
2112  00:E6B7  A9 10         ?4   LDA #A_MENSCH
2113  00:E6B9  24 53             BIT DISPTYP
2114  00:E6BB  F0 0A             BEQ ?6        ;NOT A MENSCH PLATFORM
2115  00:E6BD  AD 22 DF          LDA PD6
2116  00:E6C0  89 01             BIT #Bit0     ;CK KEYBOARD DATA IN
2117  00:E6C2  D0 03             BNE ?6        ;MARKING
2118  00:E6C4  4C 30 01      ?5   JMP URESTART
2119
2120  00:E6C7  AE B3 DF      ?6     LDX PWD_CELLS   ;USER SENTINAL
2121  00:E6CA  E0 55 AA             CPX #$AA55
2122  00:E6CD  D0 D9             BNE TRY_RESTART      ;NO PAGE 1 PGM
2123  00:E6CF  20 C0 01             JSR $00:01C0
2124  00:E6D2  70 D4                BVS TRY_RESTART  ;KEEP CHECKING
2125  00:E6D4  80 EE             BRA ?5              ;RESTART
2126
2127
2128
2129
2130                         END
2131
2132
2133
2134                         .sttl 'NMI & software break'
2135                         .page
```

'MENSCH COMPUTER ROM SOFTWARE'
'NMI & software break'

```
2136  00:E6D4                       include r_sbreak.asm
2137                        ; FILE R_SBREAK       ..... ROMULATOR INTERFACE
2138                        ; DATE 12-17-94
2139
2140
2141
2142
2143                        ; ROM-U-CODE
2144                        ;
2145
2146                        ;  This is the code needed in a TARGET computer to
2147                        ;interface with the COM LOG  Romutator.
2148                        ;
2149                        ; Include SBREAK as a vector for the BRK interrupt.
2150                        ;
2151                        ; Include NMIBRK  as a vector for the NMI interrupt vector
2152
2153
2154         00:FFB4              WRT_ERR EQU $FFB4
2155
2156         00:0001              SBRK    EQU $01
2157         00:0002              NMIFLG  EQU $02
2158
2159
2160
2161
2162                           .global SBREAK,NMIBRK
2163
2164
2165  00:E6D6  48               SBREAK  PHA       ;SOFTWARE BREAK INTERRUPT
2166  00:E6D7  08                       PHP       ;SAVE  OLD Areg SIZE
2167  00:E6D8  E2 20                  SEP #$20
2168                             .LONGA OFF
2169
2170  00:E6DA  A9 01                    LDA #SBRK
2171  00:E6DC  0C 8F DF               TSB SB_SENTL
2172  00:E6DF  F0 2D                    BEQ NMIS
2173  00:E6E1  28                       PLP       ;RESTORE Areg SIZE
2174  00:E6E2  68                       PLA       ;WE ARE STILL WORKING ON A BRK!
2175  00:E6E3  40                       RTI
2176
2177
2178
2179
2180  00:E6E4  48               NMIBRK  PHA       ;NMI INTERRUPT FROM HOST
```

```
2181  00:E6E5  08                  PHP
2182  00:E6E6  E2 20              SEP #$20
2183                      .LONGA OFF
2184
2185  00:E6E8  A9 02              LDA #NMIFLG
2186  00:E6EA  0C 8F DF           TSB SB_SENTL
2187  00:E6ED  D0 07              BNE NMIX      ;RE-ENTERED!  GET-OUT
2188  00:E6EF  A9 01              LDA #SBRK
2189  00:E6F1  2C 8F DF           BIT SB_SENTL
2190  00:E6F4  F0 18              BEQ NMIS      ;SOFTWARE INTERRUPT NOT IN PROCESS
2191  00:E6F6  28               NMIX  PLP
2192  00:E6F7  68                 PLA
```

'MENSCH COMPUTER ROM SOFTWARE'
'NMI & software break'

```
2193  00:E6F8  40                RTI
2194
2195  00:E6F9                JSL|RTL_IN:
2196  00:E6F9  8F 80 DF 00       STA ACC          ;SAVES 8 OR 16 BITS
2197
2198  00:E6FD  C2 10             REP #$10         ;NEED 16 BIT INDEX & 8 BIT Areg
2199                          .LONGI ON
2200  00:E6FF  E2 20             SEP #$20
2201                          .LONGA OFF
2202
2203  00:E701  EB                XBA              ;SAVE Breg IF Areg WAS 8 BITS
2204  00:E702  8F 81 DF 00       STA ACC+1        ; IF Areg WAS 16 BITS DOESNT MATTER
2205  00:E706  08                PHP          ;GET FLAGS
2206  00:E707  68                PLA
2207  00:E708  8F 8B DF 00       STA FLGS
2208  00:E70C  80 23             BRA NMIS_J     ;SAVE REGISTERS BUT NOT P.C.
2209
2210                                   ;SAVE ALL THE REGISTERS
2211  00:E70E  28             NMIS   PLP
2212  00:E70F  68                PLA              ;RESTORE Areg SIZE
2213  00:E710  8F 80 DF 00       STA ACC          ;SAVES 8 OR 16 BITS
2214
2215  00:E714  C2 10             REP #$10         ;NEED 16 BIT INDEX & 8 BIT Areg
2216                          .LONGI ON
2217  00:E716  E2 20             SEP #$20
2218                          .LONGA OFF
2219
2220  00:E718  EB                XBA              ;SAVE Breg IF Areg WAS 8 BITS
2221  00:E719  8F 81 DF 00       STA ACC+1        ; IF Areg WAS 16 BITS DOESNT MATTER
2222
2223  00:E71D  68                PLA
2224  00:E71E  8F 8B DF 00       STA FLGS       ;P REG
2225  00:E722  68                PLA
2226  00:E723  8F 8C DF 00       STA PCL        ;PROGRAM COUNTER LO
2227  00:E727  68                PLA
2228  00:E728  8F 8D DF 00       STA PCH        ;HI
2229  00:E72C  68                PLA
2230  00:E72D  8F 8E DF 00       STA TPBR       ;PROGRAM BANK REG
2231
2232  00:E731                NMIS_J:
2233  00:E731  8E 82 DF          STX XREG
2234  00:E734  8C 84 DF          STY YREG
2235  00:E737
2236  00:E737  BA                TSX
2237  00:E738  8E 86 DF          STX STK|PTR    ;STACK POINTER
```

```
2238  00:E73B  0B                PHD
2239  00:E73C  FA                PLX
2240  00:E73D  8E 88 DF          STX DIRREG     ;DIRECT PAGE
2241  00:E740  8B                PHB
2242  00:E741  68                PLA
2243  00:E742  8F 8A DF 00       STA DBREG      ;DATA BANK
2244
2245  00:E746  58             CLI
2246
2247  00:E747  22 84 E7 00       JSL DUMPREGS
2248
2249  00:E74B  4C 58 E3       JMP ROM|START   ;GOTO MONITOR
```

'MENSCH COMPUTER ROM SOFTWARE'
'NMI & software break'

```
2250
2251
2252  00:E74E                    GO_AGAIN:
2253  00:E74E  9C 8F DF              STZ SB_SENTL
2254  00:E751  AF 8A DF 00           LDA DBREG     ;RESTORE ALL REGISTERS
2255  00:E755  48                    PHA
2256  00:E756  AB                    PLB
2257  00:E757  AE 88 DF              LDX DIRREG
2258  00:E75A  DA                    PHX
2259  00:E75B  2B                    PLD
2260  00:E75C  AE 86 DF            LDX STK|PTR
2261  00:E75F  9A                    TXS
2262  00:E760  AF 8E DF 00           LDA TPBR
2263  00:E764  48                    PHA
2264  00:E765  AF 8D DF 00           LDA PCH
2265  00:E769  48                    PHA
2266  00:E76A  AF 8C DF 00                 LDA PCL
2267  00:E76E  48                    PHA
2268
2269  00:E76F  AF 8B DF 00           LDA FLGS
2270  00:E773  48                    PHA
2271  00:E774  AC 84 DF              LDY YREG
2272  00:E777  AE 82 DF              LDX XREG
2273
2274  00:E77A  AF 81 DF 00           LDA ACC+1     ;RESUME PROCESSING
2275  00:E77E  EB                    XBA
2276  00:E77F  AF 80 DF 00           LDA ACC
2277
2278  00:E783  40                    RTI
2279
2280
2281                     .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'NMI & software break'

```
2282
2283                    ;THIS ROUTINE FORMS AN OUTPUT LINE THAT
2284                    ;DISPLAYS THE VARIOUS REGISTERS
2285                    ;FOR THE 65C816 IE A,X,Y,S,PC,DP ETC.
2286                    ;  The format is:
2287                    ;
2288
2289                    ;* Routine:  DUMPREGS DISPLAY REG CMD'
2290                    ;*
2291                    ;* Reg Used: ACC,Y,X
2292                    ;* Var Used: TMPC,TMP0
2293                    ;* Routines Called: WRPC16,SETR,SPAC,WROB
2294                    ;* Returned Reg: NONE
2295                    ;*
2296
2297 00:E784                  DUMPREGS:
2298 00:E784 48                    PHA
2299 00:E785 DA                    PHX
2300 00:E786 5A                    PHY
2301 00:E787 08                    PHP             ;SAVE CPU MODES
2302 00:E788 E2 20                 SEP #M8            ;SET Acc SHORT
2303 00:E78A C2 10                 REP #X8
2304
2305                    .LONGA OFF
2306                    .LONGI ON
2307 00:E78C A9 03                 LDA #3
2308 00:E78E 22 F1 F1 00           JSL CLEAR_LCD_DISPLAY
2309
2310 00:E792 20 A1 E7              JSR DISP_REGS       ;PRINT OUT REGISTERS
2311                                    ;Acc,X,Y,S,D,B, & P
2312 00:E795 20 F3 E7              JSR DISP_FLAGS
2313 00:E798 22 D6 F1 00           JSL SEND_CR
2314
2315 00:E79C 28                    PLP             ;RESTORE CPU MODES
2316 00:E79D 7A                    PLY
2317 00:E79E FA                    PLX
2318 00:E79F 68                    PLA
2319 00:E7A0 6B                    RTL
2320
2321
2322
2323                                    ;PRINT OUT REGISTERS
2324                                    ;Acc,X,Y,S,D,B, & P
2325 00:E7A1                  DISP_REGS:
2326 00:E7A1 20 1E E8              JSR REGTTL1    ;PRINT OUT HEADER FOR REGISTERS
```

```
2327  00:E7A4  20 36 E8          JSR WR_PC      ;WRITE Program Counter
2328  00:E7A7  22 DB F1 00       JSL SEND_SPACE2
2329  00:E7AB  22 DF F1 00       JSL SEND_SPACE
2330
2331  00:E7AF  A2 80 DF          LDX #ACC       ;SET TO ACCESS REGS
2332  00:E7B2  86 63             STX TMP2
2333  00:E7B4  A9 08             LDA #DIRREG-ACC
2334  00:E7B6  85 57             STA TMPC       ;SAVE BYTE COUNT IE LENGTH OF REG
TBL
2335  00:E7B8  64 58             STZ TMPC+1
2336  00:E7BA  20 CF E7          JSR REG_OUT
2337
2338
```

'MENSCH COMPUTER ROM SOFTWARE'
'NMI & software break'

```
 2339  00:E7BD  20 2A E8          JSR REGTTL2    ;PRINT OUT HEADER FOR MORE
REGISTERS
 2340  00:E7C0  A2 88 DF          LDX #DIRREG      ;SET TO ACCESS REGS
 2341  00:E7C3  86 63             STX TMP2
 2342  00:E7C5  A9 04             LDA #EBIT-DIRREG
 2343  00:E7C7  85 57             STA TMPC       ;SAVE BYTE COUNT IE LENGTH OF REG
TBL
 2344  00:E7C9  64 58             STZ TMPC+1
 2345  00:E7CB  20 CF E7          JSR REG_OUT
 2346  00:E7CE  60                RTS
 2347
 2348
 2349  00:E7CF                REG_OUT:
 2350  00:E7CF  A0 00 00          LDY #0
 2351  00:E7D2  B1 63         ?1    LDA (TMP2),Y   ;GET ADDR OF NXT REG
 2352  00:E7D4  48                PHA          ;SAVE LSB ON STACK
 2353  00:E7D5  C8                INY
 2354  00:E7D6  B1 63             LDA (TMP2),Y   ;GET ADDR OF NXT REG
 2355  00:E7D8  22 70 F3 00       JSL SEND_HEX_OUT      ;OUTPUT BYTE STRING (2 ASCII
CHAR)
 2356  00:E7DC  C8                INY
 2357  00:E7DD  22 DF F1 00       JSL SEND_SPACE
 2358  00:E7E1  68                PLA          ;GET LSB OFF OF STACK
 2359  00:E7E2  22 70 F3 00       JSL SEND_HEX_OUT      ;OUTPUT BYTE STRING (2 ASCII
CHAR)
 2360  00:E7E6  22 DB F1 00       JSL SEND_SPACE2
 2361  00:E7EA  C4 57             CPY TMPC
 2362  00:E7EC  90 E4             BLT ?1
 2363  00:E7EE  22 D6 F1 00          JSL SEND_CR
 2364  00:E7F2  60                RTS
 2365
 2366
 2367  00:E7F3                DISP_FLAGS:
 2368  00:E7F3  64 5F             STZ TMP0+2         ;SAVE BANK OF STRING
 2369  00:E7F5  A9 00             LDA #0
 2370  00:E7F7  A2 4D E8          LDX #ACCCCTBL
 2371  00:E7FA  22 A1 F3 00       JSL PUT_STR
 2372                                ;NOW GET STATUS REGISTER BITS
 2373  00:E7FE  AF 8B DF 00       LDA FLGS          ;GET ORIG STATUS REG
 2374  00:E802  48                PHA
 2375  00:E803  A0 08 00          LDY #8
 2376  00:E806  68            DMP_FLGS PLA
 2377  00:E807  0A                ASL A
 2378  00:E808  48                PHA
 2379  00:E809  B0 04             BCS DMP_FLG1
```

```
2380  00:E80B  A9 30              LDA #'0'
2381  00:E80D  80 02              BRA DMP_FLGX
2382
2383  00:E80F  A9 31        DMP_FLG1 LDA #'1'
2384  00:E811  22 BC F1 00  DMP_FLGX JSL PUT_CHR
2385  00:E815  22 DB F1 00        JSL SEND_SPACE2
2386  00:E819  88                 DEY
2387  00:E81A  D0 EA              BNE DMP_FLGS
2388  00:E81C  68                 PLA
2389  00:E81D  60                 RTS
2390
2391
2392  00:E81E               REGTTL1:
2393  00:E81E  64 5F              STZ TMP0+2     ;BANK OF STRING
2394  00:E820  A9 00              LDA #0         ;PRINT OUT REG INFO HEADER
2395  00:E822  A2 71 E8           LDX #REGTBL1
```

'MENSCH COMPUTER ROM SOFTWARE'
'NMI & software break'

```
2396  00:E825  22 A1 F3 00          JSL PUT_STR
2397  00:E829  60                   RTS
2398
2399  00:E82A                REGTTL2:
2400  00:E82A  64 5F                STZ TMP0+2     ;BANK OF STRING
2401  00:E82C  A9 00                LDA #0         ;PRINT OUT REG INFO HEADER
2402  00:E82E  A2 98 E8             LDX #REGTBL2
2403  00:E831  22 A1 F3 00          JSL PUT_STR
2404  00:E835  60                   RTS
2405
2406
2407
2408
2409                    WR_PC:  ;WRITE OUT PC AS A 3 BYTE ADDRESS
2410  00:E836  AF 8E DF 00          LDA TPBR
2411  00:E83A  85 5F                STA TMP0+2
2412  00:E83C  AF 8D DF 00                LDA PCH
2413  00:E840  85 5E                STA TMP0+1
2414  00:E842  AF 8C DF 00          LDA PCL
2415  00:E846  85 5D                STA TMP0
2416  00:E848  22 51 F3 00          JSL WR_3_ADDRESS
2417  00:E84C  60                   RTS
2418
2419                    .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'NMI & software break'

```
2420                    ;*******************************************************
2421
2422
2423  00:E84D                   ACCCCTBL:
2424  00:E84D  0D                       .BYTE C_RETURN
2425  00:E84E  0D 53 74 61 74           .BYTE C_RETURN,'Status Reg',C_RETURN
          75 73 20 52 65
          67 0D
2426  00:E85A  4E 20 20 56 20           .BYTE 'N  V  M  X  D  I  Z  C'
          20 4D 20 20 58
          20 20 44 20 20
          49 20 20 5A 20
          20 43
2427  00:E870  8D                       .DC C_RETURN
2428  00:E871                   ACCCCEND:
2429
2430
2431
        ;*************************************************************
2432  00:E871                   REGTBL1:
2433  00:E871  0D                       .BYTE C_RETURN
2434  00:E872  50 43 6E 74 72           .BYTE 'PCntr     Acc    Xreg   Yreg   Stack'
          20 20 20 20 20
          41 63 63 20 20
          20 20 58 72 65
          67 20 20 20 59
          72 65 67 20 20
          20 53 74 61 63
          6B
2435  00:E896  0D 00                    .BYTE C_RETURN,0
2436
2437  00:E898                   REGTBL2:
2438  00:E898  0D                       .BYTE C_RETURN
2439  00:E899  20 20 44 69 72           .BYTE '  DirRg  F  DBk'
          52 67 20 20 46
          20 20 44 42 6B
2440  00:E8A8  0D 20 20 00              .BYTE C_RETURN,'  ',0
2441
2442
        ;*************************************************************
2443
2444                    .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'NMI & software break'

```
2445
2446                    ;THIS ROUTINE FORMS AN OUTPUT LINE THAT
2447                    ;DISPLAYS THE VARIOUS REGISTERS
2448                    ;FOR THE 65C816 IE A,X,Y,S,PC,DP ETC.
2449                    ; AND THEN ALLOWS THE ABILITY TO CHANGE
2450                    ;ANY REGISTER.
2451
2452                    ;* Routine:  ALTER_REGS
2453                    ;*
2454                    ;* Reg Used: ACC,Y,X
2455                    ;* Var Used: TMPC,TMP0
2456                    ;* Routines Called: WRPC16,SETR,SPAC,WROB
2457                    ;* Returned Reg: NONE
2458                    ;*
2459
2460  00:E8AC                   ALTER_REGS:
2461
2462  00:E8AC  20 1E E8             JSR REGTTL1    ;PRINT OUT HEADER FOR REGISTERS
2463  00:E8AF  20 36 E8             JSR WR_PC      ;WRITE Program Counter
2464  00:E8B2  22 DB F1 00          JSL SEND_SPACE2
2465  00:E8B6  22 DF F1 00          JSL SEND_SPACE
2466  00:E8BA  A2 80 DF             LDX #ACC       ;SET TO ACCESS REGS
2467  00:E8BD  86 63                STX TMP2
2468  00:E8BF  A9 08                LDA #DIRREG-ACC
2469  00:E8C1  85 57                STA TMPC       ;SAVE BYTE COUNT IE LENGTH OF REG
TBL
2470  00:E8C3  64 58                STZ TMPC+1
2471  00:E8C5  20 CF E7             JSR REG_OUT    ;Print out current values of reg's
2472
2473
2474                    PC_CNTR_IN:    ;get new value for pc counter
2475  00:E8C8
2476  00:E8C8  22 9C F0 00          JSL GET|BYTE
2477  00:E8CC  90 10                BCC ?4
2478  00:E8CE  C9 20                CMP #' '
2479  00:E8D0  F0 03                BEQ ?3          ;MUST BE A SPACE
2480  00:E8D2  82 EC 00             BRL ALTER|ERR  ;NON HEX
2481
2482  00:E8D5  AF 8E DF 00   ?3        LDA TPBR
2483  00:E8D9  20 8C F3             JSR WRT2OUT
2484  00:E8DC  80 04                   BRA ?5
2485
2486  00:E8DE  8F 8E DF 00   ?4      STA TPBR
2487  00:E8E2  A9 3A            ?5      LDA #':'
2488  00:E8E4  22 BC F1 00          JSL PUT_CHR
```

```
2489
2490  00:E8E8  22 9C F0 00          JSL GET|BYTE
2491  00:E8EC  90 10               BCC ?7
2492  00:E8EE  C9 20               CMP #' '
2493  00:E8F0  F0 03               BEQ ?6          ;MUST BE A SPACE
2494  00:E8F2  82 CC 00            BRL ALTER|ERR  ;NON HEX
2495
2496  00:E8F5  AF 8D DF 00   ?6       LDA PCH
2497  00:E8F9  20 8C F3            JSR WRT2OUT
2498  00:E8FC  80 04                  BRA ?8
2499
2500  00:E8FE  8F 8D DF 00   ?7     STA PCH
2501
```

'MENSCH COMPUTER ROM SOFTWARE'
'NMI & software break'

```
2502  00:E902  22 9C F0 00    ?8     JSL GET|BYTE
2503  00:E906  90 10                  BCC ?A
2504  00:E908  C9 20                  CMP #' '
2505  00:E90A  F0 03                  BEQ ?9          ;MUST BE A SPACE
2506  00:E90C  82 B2 00               BRL ALTER|ERR  ;NON HEX
2507
2508  00:E90F  AF 8C DF 00    ?9        LDA PCL
2509  00:E913  20 8C F3               JSR WRT2OUT
2510  00:E916  80 04                     BRA ?B
2511
2512  00:E918  8F 8C DF 00    ?A      STA PCL
2513
2514  00:E91C  22 DB F1 00    ?B      JSL SEND_SPACE2
2515  00:E920  22 DF F1 00              JSL SEND_SPACE
2516
2517  00:E924                 REG_IN:
2518
2519  00:E924  A0 00 00               LDY #0
2520  00:E927
2521  00:E927  C8             ?1        INY                ;POINT TO MOST SIGNIFICANT BYTE
2522  00:E928  22 9C F0 00            JSL GET|BYTE     ;DO 1ST BYTE
2523  00:E92C  90 0E                  BCC ?4
2524  00:E92E  C9 20                  CMP #' '
2525  00:E930  F0 03                  BEQ ?3          ;MUST BE A SPACE
2526  00:E932  82 8C 00               BRL ALTER|ERR  ;NON HEX
2527
2528  00:E935  B1 63          ?3        LDA (TMP2),Y            ;REPEAT ORIGIONAL BYTE
2529  00:E937  20 8C F3               JSR WRT2OUT
2530  00:E93A  80 02                     BRA ?5
2531
2532  00:E93C  91 63          ?4      STA (TMP2),Y
2533  00:E93E  22 DF F1 00    ?5        JSL SEND_SPACE
2534  00:E942  88                     DEY                ;POINT TO LEAST SIGNIFICANT BYTE
2535
2536  00:E943  22 9C F0 00            JSL GET|BYTE     ;DO 2ND BYTE
2537  00:E947  90 0E                  BCC ?7
2538  00:E949  C9 20                  CMP #' '
2539  00:E94B  F0 03                  BEQ ?6          ;MUST BE A SPACE
2540  00:E94D  82 71 00               BRL ALTER|ERR  ;NON HEX
2541
2542  00:E950  B1 63          ?6        LDA (TMP2),Y
2543  00:E952  20 8C F3                 JSR WRT2OUT
2544  00:E955  80 02                    BRA ?8
2545
2546  00:E957  91 63          ?7      STA (TMP2),Y
```

```
2547
2548  00:E959  C8              ?8      INY
2549  00:E95A  C8                      INY          ;POINT TO NEXT REG
2550  00:E95B  22 DB F1 00             JSL SEND_SPACE2
2551  00:E95F  C4 57                   CPY TMPC
2552  00:E961  90 C4                   BLT ?1
2553
2554  00:E963                  REG_IN_II:
2555
2556  00:E963  64 5F                   STZ TMP0+2     ;BANK OF STRING
2557  00:E965  A9 00                   LDA #0         ;PRINT OUT REG INFO HEADER
2558  00:E967  A2 98 E8                LDX #REGTBL2
```

'MENSCH COMPUTER ROM SOFTWARE'
'NMI & software break'

```
2559  00:E96A  22 A1 F3 00          JSL PUT_STR
2560
2561  00:E96E  A2 88 DF             LDX #DIRREG      ;SET TO ACCESS REGS
2562  00:E971  86 63                STX TMP2
2563  00:E973  A9 04                LDA #EBIT-DIRREG
2564  00:E975  85 57                STA TMPC      ;SAVE BYTE COUNT IE LENGTH OF REG
TBL
2565  00:E977  64 58                STZ TMPC+1
2566  00:E979  20 CF E7             JSR REG_OUT
2567
2568  00:E97C  22 DB F1 00             JSL SEND_SPACE2
2569  00:E980  A0 00 00             LDY #0
2570  00:E983
2571  00:E983  C8              ?1      INY               ;POINT TO MOST SIGNIFICANT BYTE
2572  00:E984  22 9C F0 00           JSL GET|BYTE     ;DO 1ST BYTE
2573  00:E988  90 0E                 BCC ?4
2574  00:E98A  C9 20                 CMP #' '
2575  00:E98C  F0 03                 BEQ ?3         ;MUST BE A SPACE
2576  00:E98E  82 30 00              BRL ALTER|ERR  ;NON HEX
2577
2578  00:E991  B1 63           ?3      LDA (TMP2),Y           ;REPEAT ORIGIONAL BYTE
2579  00:E993  20 8C F3              JSR WRT2OUT
2580  00:E996  80 02                    BRA ?5
2581
2582  00:E998  91 63           ?4     STA (TMP2),Y
2583  00:E99A  22 DF F1 00     ?5      JSL SEND_SPACE
2584  00:E99E  88                    DEY               ;POINT TO LEAST SIGNIFICANT BYTE
2585
2586  00:E99F  22 9C F0 00           JSL GET|BYTE     ;DO 2ND BYTE
2587  00:E9A3  90 0E                 BCC ?7
2588  00:E9A5  C9 20                 CMP #' '
2589  00:E9A7  F0 03                 BEQ ?6         ;MUST BE A SPACE
2590  00:E9A9  82 15 00              BRL ALTER|ERR  ;NON HEX
2591
2592  00:E9AC  B1 63           ?6      LDA (TMP2),Y
2593  00:E9AE  20 8C F3              JSR WRT2OUT
2594  00:E9B1  80 02                 BRA ?8
2595
2596  00:E9B3  91 63           ?7     STA (TMP2),Y
2597
2598  00:E9B5  C8              ?8     INY
2599  00:E9B6  C8                    INY          ;POINT TO NEXT REG
2600  00:E9B7  22 DB F1 00           JSL SEND_SPACE2
2601  00:E9BB  C4 57                 CPY TMPC
2602  00:E9BD  90 C4                 BLT ?1
```

```
2603
2604  00:E9BF  18                    CLC
2605  00:E9C0  6B                    RTL
2606
2607  00:E9C1                  ALTER|ERR:
2608  00:E9C1  38                  SEC
2609  00:E9C2  6B                  RTL
2610
2611
2612                       .END
2613
2614
2615
```

'MENSCH COMPUTER ROM SOFTWARE'
'NMI & software break'

```
2616                    .sttl 'Debug routines, Loads & Dumps'
2617                    .page
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
2618  00:E9C2                       include r_debug.asm
2619                        ;FILE = R_DEBUG.ASM
2620                        ;DATE = 1-06-94
2621
2622
2623
2624
2625                        ;*    Routine:  XS28IN
2626                        ;*
2627                        ;*  This routine must be called using a JSL cmd!
2628                        ;*
2629                        ;* XS28IN Reads S28 formated data from the
2630                        ;*   input selected by the CONTROL_INPUT routine
2631                        ;*   and places it into memory.  This routine
2632                        ;*   outputs a "." each time a record is transfered
2633                        ;*   to memory with out error.  A "?" is returned
2634                        ;*   if the checksum does not agree. After receiving
2635                        ;*   the final record a Cy = 0  is returned if no
2636                        ;*   errors had been encountered.  Likewise, if errors
2637                        ;*   occurred, a Cy = 1  will be returned.
2638                        ;*
2639                        ;* Reg Used: ACC,Y,X
2640                        ;* Var Used: TMP0,TMP2,TMP4,TMP6,TMPC,ERRORS
2641                        ;* Routines Called: GET_CHR,DADD,RDOB,BYTE
2642                        ;*
2643                        ;* Returned Reg: NO registers are saved!
2644                        ;*
2645
2646         00:E9C3              XS28IN   EQU *
2647
2648
2649  00:E9C3  A5 4D                   LDA INPUT_XTRL
2650  00:E9C5  29 0D                AND #Bit0+Bit2+Bit3
2651  00:E9C7  F0 29                   BEQ XS28EEE          ;NO INPUT SOURCE
2652
2653  00:E9C9  A9 03                LDA #3
2654  00:E9CB  22 F1 F1 00          JSL CLEAR_LCD_DISPLAY
2655
2656  00:E9CF  A9 00                LDA #0
2657  00:E9D1  A2 D5 EA             LDX #S28_Loader
2658  00:E9D4  22 FA F1 00          JSL DISP_LCD_STRNG
2659
2660  00:E9D8  A2 00 00             LDX #0        ;Record #
2661  00:E9DB  A9 03                LDA #03       ;on the LCD
2662  00:E9DD  22 FD F1 00          JSL POSITION_TEXT_CURSOR
```

```
2663
2664  00:E9E1  A9 00              LDA #0
2665  00:E9E3  A2 E3 EA           LDX #RecordNo
2666  00:E9E6  22 FA F1 00        JSL DISP_LCD_STRNG
2667
2668  00:E9EA  22 B2 F1 00        JSL GET_CHR
2669  00:E9EE  C9 1B              CMP #ESC
2670  00:E9F0  D0 03              BNE XS28A
2671  00:E9F2  82 DE 00       XS28EEE  BRL XLS_BAD
2672
2673  00:E9F5  C9 53          XS28A    CMP #'S'      ;FIND NEXT RCD MARK (S)
2674  00:E9F7  D0 CA              BNE XS28IN
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
2675
2676  00:E9F9                  XS28ROM:
2677  00:E9F9  64 6F                STZ ERRORS
2678  00:E9FB  A2 00 00             LDX #0
2679  00:E9FE  86 B3                STX COUNT     ;GOOD RECORD COUNT
2680
2681        00:EA00             XLSS    EQU *         ;LOAD SINGLE S28 RECORD
2682                                     ;CHECKSUM USED, BUT
2683                                     ;REQUIRED SO THAT IT
2684                                     ;IS COMPATABLE
2685  00:EA00  E6 6F                INC ERRORS
2686  00:EA02  22 B2 F1 00          JSL GET_CHR    ;GET S RECORD TYPE
2687  00:EA06  48                   PHA           ;SAVE S RECORD TYPE
2688  00:EA07  64 66                STZ TMP4
2689  00:EA09  64 67                STZ TMP4+1    ;CLR CKSUM REG
2690  00:EA0B  22 7D F0 00          JSL GET_HEX    ;GET BYTE COUNT
2691  00:EA0F  85 57                STA TMPC      ;SAVE BYTE COUNT
2692  00:EA11  20 46 F1             JSR DADD
2693  00:EA14  C6 57                DEC TMPC      ;DEC FOR S28 ADDR
2694  00:EA16  C6 57                DEC TMPC
2695  00:EA18  C6 57                DEC TMPC
2696  00:EA1A  68                   PLA           ;GET RECORD TYPE
2697  00:EA1B  C9 32                CMP #'2'      ;ONLY ALLOW S28 RECORDS
2698  00:EA1D  F0 07                BEQ XS28LA1
2699  00:EA1F  C9 38                CMP #'8'
2700  00:EA21  D0 44                BNE XLSS0     ;UNKNOWN RECORD TYPE
2701  00:EA23  82 99 00             BRL XLSSDONE   ;ITS LAST LINE OF RECORD
2702
2703  00:EA26  C6 57             XS28LA1  DEC TMPC
2704  00:EA28  22 7D F0 00          JSL GET_HEX    ;GET BANK ADDR
2705  00:EA2C  85 5F                STA TMP0+2    ;SAVE 3 BYTE ADDRESS
2706  00:EA2E  20 46 F1             JSR DADD      ;ADD TO CKSM
2707  00:EA31  22 7D F0 00          JSL GET_HEX    ;SA HO TO TMP0+1
2708  00:EA35  85 5E                STA TMP0+1
2709  00:EA37  20 46 F1             JSR DADD      ;ADD TO CKSM
2710  00:EA3A  22 7D F0 00          JSL GET_HEX    ;SA LO TO TMP0
2711  00:EA3E  85 5D                STA TMP0
2712  00:EA40  20 46 F1             JSR DADD      ;ADD TO CHKSM
2713
2714  00:EA43  A5 57                LDA TMPC      ;CK IF # OF BYTES ZERO
2715  00:EA45  F0 13                BEQ XS28G2
2716  00:EA47  20 EC EA          XS28GD1  JSR BYTE      ;BYTE SUB/R DECRS LCNT
2717  00:EA4A  90 0C                BCC XS28G3
2718  00:EA4C  E6 6F                INC ERRORS    ;DEC COUNTER & INC ADDR
2719  00:EA4E  E6 66                INC TMP4      ;MESS UP CKSUM SO WILL PRINT ERR
```

```
2720  00:EA50  20 46 F1            JSR DADD      ;INCR CKSUM
2721  00:EA53  20 54 F1            JSR INCTMP0   ;GO INCR TMP0 ADR
2722  00:EA56  C6 57               DEC TMPC
2723                                ;BYTE ENDING TOO SOON
2724  00:EA58  D0 ED          XS28G3  BNE XS28GD1   ;ON EXIT
2725  00:EA5A  22 7D F0 00    XS28G2  JSL GET_HEX   ;CKSUM FROM HEX RCD>TMP0
2726  00:EA5E  20 46 F1            JSR DADD
2727  00:EA61  A5 66               LDA TMP4      ;GET CHKSUM
2728
2729  00:EA63  C9 FF               CMP #$FF
2730  00:EA65  D0 40               BNE XS28ERR   ;BAD RECORD LOAD
2731
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
2732 00:EA67 C6 6F          XLSS0   DEC ERRORS    ;A GOOD LOAD
2733
2734 00:EA69 22 B2 F1 00    XLSFIN  JSL GET_CHR   ;GET CR OR LF
2735 00:EA6D C9 0D              CMP #C_RETURN     ;CR
2736 00:EA6F F0 04              BEQ XLSFIN1
2737 00:EA71 C9 0A              CMP #L_FEED   ;LF/NEW LINE
2738 00:EA73 D0 F4              BNE XLSFIN
2739 00:EA75 A9 2E          XLSFIN1 LDA #'.'      ;ACK
2740 00:EA77 22 BC F1 00        JSL PUT_CHR
2741 00:EA7B F8                 SED
2742                        .LONGA ON
2743 00:EA7C C2 20              REP #M8
2744 00:EA7E A5 B3              LDA COUNT
2745 00:EA80 18                 CLC
2746 00:EA81 69 01 00           ADC #1
2747 00:EA84 85 B3              STA COUNT
2748 00:EA86 D8                 CLD
2749                        .LONGA OFF
2750 00:EA87 E2 20              SEP #M8
2751
2752 00:EA89 A2 0A 00           LDX #10       ;Show a running count of good records
2753 00:EA8C A9 03              LDA #03       ;on the LCD
2754 00:EA8E 22 FD F1 00        JSL POSITION_TEXT_CURSOR
2755
2756 00:EA92 A5 B4              LDA COUNT+1
2757 00:EA94 20 8C F3           JSR WRT2OUT
2758 00:EA97 A5 B3              LDA COUNT
2759 00:EA99 20 8C F3           JSR WRT2OUT
2760
2761 00:EA9C 22 B2 F1 00    XLSFN   JSL GET_CHR   ;GET END RECORD
2762 00:EAA0 C9 53              CMP #'S'
2763 00:EAA2 D0 F8              BNE XLSFN
2764 00:EAA4 82 59 FF               BRL XLSS      ;REPEAT RECORD LOOP
2765
2766
2767 00:EAA7 22 B2 F1 00    XS28ERR JSL GET_CHR   ;GET END OF RECORD
2768 00:EAAB C9 1B              CMP #ESC
2769 00:EAAD F0 24              BEQ XLS_BAD   ;KICKED OUT!
2770
2771 00:EAAF C9 0A              CMP #L_FEED   ;LF/NEW LINE
2772 00:EAB1 D0 04              BNE XS28ER1
2773 00:EAB3 C9 0D              CMP #C_RETURN     ;CR
2774 00:EAB5 D0 F0              BNE XS28ERR
2775 00:EAB7 A9 3F          XS28ER1 LDA #'?'      ;NAK
2776 00:EAB9 22 BC F1 00        JSL PUT_CHR
```

```
2777  00:EABD  80 DD              BRA XLSFN
2778
2779  00:EABF                  XLSSDONE
2780  00:EABF  22 B2 F1 00        JSL GET_CHR    ;GET CR OR LF
2781  00:EAC3  C9 0D              CMP #C_RETURN      ;CR
2782  00:EAC5  F0 04              BEQ XLSSXIT
2783  00:EAC7  C9 0A              CMP #L_FEED    ;LF
2784  00:EAC9  D0 F4              BNE XLSSDONE
2785
2786  00:EACB  C6 6F           XLSSXIT  DEC ERRORS    ;CORRECT FOR S8 RECORD
2787
2788                     ;       LDA SFLAG0
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
2789                    ;       AND #$FF-ECHOFF
2790                    ;       ORA TMP6+1     ;RESTORE STATE OF ECHO OFF
2791                    ;       STA SFLAG0
2792                    ;       CLI
2793
2794  00:EACD  A5 6F          XLHDONE   LDA ERRORS
2795  00:EACF  D0 02                BNE XLS_BAD
2796  00:EAD1  18                   CLC
2797  00:EAD2  6B                   RTL
2798  00:EAD3
2799  00:EAD3  38             XLS_BAD   SEC
2800  00:EAD4  6B                   RTL
2801
2802
2803  00:EAD5  20 20 20 53 32          S28_Loader .dc '   S28 LOADER '
         38 20 4C 4F 41
         44 45 52 A0
2804  00:EAE3  52 65 63 6F 72          RecordNo  .dc 'Record # '
         64 20 23 A0
2805
2806
2807                    ;* Routine:  BYTE_LONG
2808                    ;*  READ AND STORE BYTE.
2809                    ;*  NO STORE IF SPACE OR TMPC=0.
2810                    ;* Reg Used: ACC,Y,X
2811                    ;* Var Used: TMPC,TMP0
2812                    ;* Routines Called: RDOB,DADD,INCTMP0
2813                    ;* Returned Reg: NONE
2814                    ;*
2815
2816         00:EAEC            BYTE    EQU *
2817  00:EAEC  22 7D F0 00          JSL GET_HEX    ;CHAR IN A, CY=0 IF
2818  00:EAF0  B0 06                BCS BY1        ;BAD DATA
2819  00:EAF2  87 5D                STA [TMP0]     ;STORE BYTE (DIRECT INDIRECT LONG)
2820  00:EAF4  C7 5D                CMP [TMP0]     ;TEST FOR VALID WRITE
2821  00:EAF6  F0 07                BEQ BY2
2822                                  ;NOT A VALID WRITE
2823  00:EAF8  20 54 F1          BY1     JSR INCTMP0    ;increment the address
2824  00:EAFB  C6 57                DEC TMPC
2825  00:EAFD  38                   SEC
2826  00:EAFE  60                   RTS
2827
2828
2829  00:EAFF  20 46 F1          BY2     JSR DADD       ;INCR CKSUM
2830  00:EB02  20 54 F1             JSR INCTMP0    ;GO INCR TMP0 ADR
```

```
2831  00:EB05  C6 57          DEC TMPC
2832  00:EB07  18             CLC
2833
2834
2835  00:EB08  60             RTS
2836
2837                  ;*************************************************
2838                  .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
2839
2840                    ;* Routine:  SET_Breakpoint
2841                    ;*
2842                    ;* SET_Breakpoint replaces the the byte at the
2843                    ;*   given address with a BREAK instruction.
2844                    ;* The BREAK address is requested by this routine.
2845                    ;*
2846                    ;*
2847                    ;* Reg Used: ACC,Y,X
2848                    ;* Var Used: TMP2
2849                    ;* Routines Called:  Get_S_Address
2850                    ;*
2851                    ;* Returned Reg: NONE
2852                    ;*
2853                    ;*
2854
2855 00:EB09                  SET_Breakpoint:
2856
2857 00:EB09 48                    PHA
2858 00:EB0A DA                 PHX
2859 00:EB0B 5A                    PHY
2860 00:EB0C 22 04 F2 00        JSL Get_Address  ;break address
2861 00:EB10 B0 08                 BCS ?9
2862
2863
2864 00:EB12 A9 00          ?2      LDA #0
2865 00:EB14 87 63              STA [TMP2]
2866
2867 00:EB16 9C 8F DF           STZ SB_SENTL
2868
2869
2870 00:EB19 18                    CLC
2871 00:EB1A 7A             ?9      PLY
2872 00:EB1B FA                    PLX
2873 00:EB1C 68                    PLA
2874 00:EB1D 6B                 RTL
2875
2876
2877
2878
2879
2880
2881                    ;*****************************************************
2882                    .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
2883
2884                    ;* Routine:  FILL_Memory
2885                    ;*
2886                    ;* FILL_Memory takes a HEX byte and propogates it
2887                    ;*   through a section of memory. First a starting
2888                    ;*   address is requested then a ending address is
2889                    ;*   requested and finally the HEX byte is requested.
2890                    ;*
2891                    ;*
2892                    ;* Reg Used: ACC,Y,X
2893                    ;* Var Used: TMP0,TMP2,TMP4,TMPC,DIFF
2894                    ;* Routines Called:  Get_S_Address, Get_E_Address
2895                    ;*               HEX2IN, POSITION_TEXT_CURSOR
2896                    ;*               DISP_LCD_STRNG, INCTMP0
2897                    ;*
2898                    ;* Returned Reg: NONE
2899                    ;*
2900                    ;*
2901
2902  00:EB1E                   FILL_Memory:
2903
2904  00:EB1E  48                   PHA
2905  00:EB1F  DA                 PHX
2906  00:EB20  5A                   PHY
2907  00:EB21  22 6F F2 00        JSL Get_S_Address  ;starting address
2908  00:EB25  B0 59                BCS ?9
2909
2910  00:EB27  A6 63              LDX TMP2      ;move start addr to TMP0
2911  00:EB29  86 5D              STX TMP0            ;
2912
2913  00:EB2B  A5 65              LDA TMP2+2          ;
2914  00:EB2D  85 5F              STA TMP0+2          ;
2915
2916  00:EB2F  22 8C F2 00        JSL Get_E_Address  ;ENDING address
2917  00:EB33  B0 4B                BCS ?9
2918  00:EB35  20 89 EE           JSR DCMP      ;IS EA > SA
2919  00:EB38  90 46              BCC ?9        ;NO
2920
2921  00:EB3A  A2 00 00           LDX #0
2922  00:EB3D  A9 06              LDA #06
2923  00:EB3F  22 D6 F1 00        JSL SEND_CR
2924  00:EB43  22 FD F1 00        JSL POSITION_TEXT_CURSOR
2925
2926  00:EB47  A9 00              LDA #0
2927  00:EB49  A2 84 EB           LDX #Enter_HEX
```

```
2928  00:EB4C  22 A1 F3 00      JSL PUT_STR
2929  00:EB50  22 C6 F1 00      JSL GET_PUT_CHR
2930  00:EB54  B0 2A            BCS ?9
2931  00:EB56  20 F0 F0         JSR HEXIN
2932  00:EB59  B0 25            BCS ?9
2933  00:EB5B  0A               ASL A
2934  00:EB5C  0A               ASL A
2935  00:EB5D  0A               ASL A
2936  00:EB5E  0A               ASL A
2937  00:EB5F  85 70            STA TEMP
2938  00:EB61  C8               INY
2939  00:EB62  22 C6 F1 00      JSL GET_PUT_CHR
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
2940  00:EB66  B0 18              BCS ?9
2941  00:EB68  20 F0 F0           JSR HEXIN
2942  00:EB6B  B0 13              BCS ?9
2943  00:EB6D  05 70              ORA TEMP
2944  00:EB6F  85 70              STA TEMP
2945
2946  00:EB71  A5 70        ?2    LDA TEMP
2947  00:EB73  87 5D              STA [TMP0]
2948  00:EB75  20 89 EE           JSR DCMP      ;EA-SA (TMP2-TMP0) DIFF
2949  00:EB78  F0 05                BEQ ?8
2950  00:EB7A  20 54 F1           JSR INCTMP0   ;INC SA
2951  00:EB7D  80 F2                BRA ?2
2952
2953  00:EB7F  18           ?8    CLC
2954  00:EB80  7A           ?9    PLY
2955  00:EB81  FA                 PLX
2956  00:EB82  68                 PLA
2957  00:EB83  6B                 RTL
2958
2959
2960  00:EB84  45 6E 74 65 72     Enter_HEX .dc 'Enter Byte in HEX '
          20 42 79 74 65
          20 69 6E 20 48
          45 58 A0
2961
2962
2963                  ;*****************************************************
2964                  .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
2965
2966                    ;* Routine:  Alter_Memory
2967                    ;*
2968                    ;* Alter_Memory prints 1 line of memory dump from an
2969                    ;*   address thats inputted. The second line prints
2970                    ;*   the same starting address and then allows the
2971                    ;*   programmer to input new data 1 byte at a time
2972                    ;*
2973                    ;* An ENTER key terminates the operation.  A SPACE
2974                    ;*   character will allow the programmer to skip over
2975                    ;*   a memory cell without changing it.
2976                    ;*
2977                    ;* Reg Used: ACC,Y,X
2978                    ;* Var Used: TMP0,TMP1,TMP2,TMP4,SFLAG0,TMPC,DIFF
2979                    ;* Routines Called:  Dump_1_line_to_Screen
2980                    ;*
2981                    ;* Returned Reg: NONE
2982                    ;*
2983                    ;*
2984
2985
2986         00:EB96            Alter_Memory EQU *
2987
2988
2989                          ;First....show what we are changing!
2990  00:EB96  22 73 EC 00          JSL Dump_1_line_to_Output
2991  00:EB9A  90 03                BCC ?AM1
2992  00:EB9C  82 A8 00             BRL ?99
2993
2994  00:EB9F  A5 4E          ?AM1    LDA OUTPUT_XTRL
2995  00:EBA1  89 01                BIT #1       ;IS LCD ON?
2996  00:EBA3  F0 04                BEQ ?1       ;NO
2997  00:EBA5  A9 08                LDA #8       ;8 bytes displayed per line
2998  00:EBA7  80 02                BRA ?2
2999
3000  00:EBA9  A9 10          ?1     LDA #16      ;16 bytes displayed per line
3001  00:EBAB  85 57          ?2     STA TMPC
3002
3003  00:EBAD  A5 62                LDA TMP1+2    ;WRITE LONG ADDRESS
3004  00:EBAF  22 70 F3 00          JSL SEND_HEX_OUT
3005  00:EBB3  A9 3A                LDA #':'
3006  00:EBB5  22 BC F1 00          JSL PUT_CHR
3007  00:EBB9  A5 61                LDA TMP1+1
3008  00:EBBB  22 70 F3 00          JSL SEND_HEX_OUT
3009  00:EBBF  A5 60                LDA TMP1
```

```
3010  00:EBC1  22 70 F3 00          JSL SEND_HEX_OUT
3011  00:EBC5  A9 20           ?AM2     LDA #' '
3012  00:EBC7  22 BC F1 00          JSL PUT_CHR
3013  00:EBCB  22 C6 F1 00    ?AM2.1   JSL GET_PUT_CHR   ;get a hex char
3014  00:EBCF  B0 76                 BCS ?99
3015  00:EBD1  C9 0D                 CMP #C_RETURN
3016  00:EBD3  F0 70                 BEQ ?90     ;we are done
3017  00:EBD5  C9 08                 CMP #BKSP
3018  00:EBD7  F0 62                 BEQ ?70
3019  00:EBD9  C9 20                 CMP #' '
3020  00:EBDB  D0 0E                 BNE ?AM2.6
3021  00:EBDD  A9 08                 LDA #BKSP         ;BACKUP         backup again
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
3022  00:EBDF  22 BC F1 00        JSL PUT_CHR
3023  00:EBE3  A7 60              LDA [TMP1]    ;GET OLD CHAR
3024  00:EBE5  22 70 F3 00        JSL SEND_HEX_OUT
3025  00:EBE9  80 40                 BRA ?AM3
3026
3027  00:EBEB  20 F0 F0      ?AM2.6   JSR HEXIN
3028  00:EBEE  90 05              BCC ?AM2.7      ;its hex
3029  00:EBF0  20 49 EC              JSR ?Q_IT      ;not valid hex
3030  00:EBF3  80 D6                 BRA ?AM2.1    ;TRY AGAIN
3031
3032  00:EBF5  0A            ?AM2.7   ASL A
3033  00:EBF6  0A                 ASL A
3034  00:EBF7  0A                 ASL A
3035  00:EBF8  0A                 ASL A
3036  00:EBF9  85 70              STA TEMP
3037  00:EBFB  22 C6 F1 00   ?AM2.8   JSL GET_PUT_CHR   ;second char of hex byte
3038  00:EBFF  B0 46              BCS ?99
3039  00:EC01  C9 0D              CMP #C_RETURN
3040  00:EC03  F0 40              BEQ ?90     ;we are done
3041  00:EC05  C9 08              CMP #BKSP
3042  00:EC07  F0 F2              BEQ ?AM2.8
3043  00:EC09  20 F0 F0           JSR HEXIN
3044  00:EC0C  90 05              BCC ?AM2.9
3045  00:EC0E  20 49 EC              JSR ?Q_IT      ;Not valid hex
3046  00:EC11  80 E8                 BRA ?AM2.8    ;TRY AGAIN
3047
3048  00:EC13  05 70         ?AM2.9   ORA TEMP
3049  00:EC15  87 60              STA [TMP1]    ;STORE BYTE (DIRECT INDIRECT LONG)
3050  00:EC17  C7 60              CMP [TMP1]    ;TEST FOR VALID WRITE
3051  00:EC19  F0 10              BEQ ?AM3
3052  00:EC1B  22 E4 F1 00        JSL BACKSPACE2 ;BACKUP 2nd char positionS
3053  00:EC1F  A9 3F              LDA #'?'  ;write a "?"
3054  00:EC21  22 BC F1 00        JSL PUT_CHR
3055  00:EC25  A9 3F              LDA #'?'  ;write a "?"
3056  00:EC27  22 BC F1 00        JSL PUT_CHR
3057  00:EC2B
3058  00:EC2B  20 65 F1      ?AM3    JSR INCTMP1    ;GO INCR TMP0 ADR
3059  00:EC2E  C6 57              DEC TMPC
3060  00:EC30  D0 93              BNE ?AM2         ;continue on same line
3061  00:EC32  A9 0D              LDA #C_RETURN
3062  00:EC34  22 BC F1 00        JSL PUT_CHR
3063  00:EC38  82 64 FF           BRL ?AM1         ;start new line
3064
3065
3066  00:EC3B  22 E4 F1 00   ?70     JSL BACKSPACE2
```

```
3067  00:EC3F  20 8C F1          JSR DECTMP1    ;BACKUP POINTER
3068  00:EC42  82 86 FF          BRL ?AM2.1
3069
3070  00:EC45  18            ?90   CLC          ;good exit
3071  00:EC46  6B                  RTL
3072
3073  00:EC47  38            ?99   SEC          ;bad exit
3074  00:EC48  6B                  RTL
3075
3076  00:EC49  A9 08         ?Q_IT  LDA #BKSP   ;BACKUP 1 char position
3077  00:EC4B  22 BC F1 00         JSL PUT_CHR
3078  00:EC4F  A9 3F               LDA #'?'   ;write a "?"
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
3079  00:EC51  22 BC F1 00          JSL PUT_CHR
3080  00:EC55  A9 08                LDA #BKSP       ;BACKUP       backup again
3081  00:EC57  22 BC F1 00          JSL PUT_CHR
3082  00:EC5B  60                      RTS
3083
3084
3085                    ;*****************************************************
3086                    .page
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
3087
3088                    ;* Routine:  DUMP_OUT
3089                    ;*    Variations:
3090                    ;*      DumpS28  ;S28 loader format
3091                    ;*      Dump_to_Printer  formated dump to printer
3092                    ;*      Dump_to_Screen   formated dump to screen
3093                    ;*      Dump_1_line_to_Screen   single line dump
3094                    ;*      Dump_to_Screen_ASCII   ASCII replaces HEX
3095                    ;*         Dump_to__Output  G.P. formated dump
3096                    ;*      Dump_1_line_to_Output:
3097                    ;*
3098                    ;* Reg Used: ACC,Y,X
3099                    ;* Var Used: TMP0,TMP1,TMP2,TMP4,SFLAG0,TMPC,DIFF
3100                    ;* Routines Called: GET_STR,DISP_LCD_STRNG,WRTWO,DCMP,DADD
3101                    ;*                  POSITION_TEXT_CURSOR,CKNOUT,WROB
3102                    ;* Returned Reg: NONE
3103                    ;*
3104                    ;*    DUMP_FLGS (format control byte)
3105                    ;*      Flag1 = output S28+byte-count
3106                    ;*      Flag2 = Format for LCD
3107                    ;*      Flag3 = add spaces between data bytes & HEADER
3108                    ;*      Flag4 = add checksum
3109                    ;*      Flag5 = 8 bytes not 16 bytes
3110                    ;*      Flag6 = ONE LINE ONLY
3111                    ;*      Flag7 = ASCII not Hex data
3112
3113
3114                    ;*************************************************************
3115                    ;*
3116                    ;*  Dump to printer switches the output control to Printer
3117                    ;*    and then requests starting and ending addresses
3118                    ;*    from the keyboard.  A formated output with header line
3119                    ;*    is sent to the printer.  Every 60 lines a new page with
3120                    ;*    a new header is started until the ending address is
3121                    ;*    reached.  The origional output control is re-established
3122                    ;*    before the routine returns to the caller.
3123                    ;*  The routine is aborted and a Cy=1 is returned if the
3124                    ;*    address inputted has a hex conversion error.
3125
3126
3127  00:EC5C                  Dump_to_Printer:
3128
3129  00:EC5C  A5 4E                LDA OUTPUT_XTRL
3130  00:EC5E  48                  PHA
3131  00:EC5F  09 02                ORA #Bit1 ;SET Printer on
```

```
3132  00:EC61  85 4E          STA OUTPUT_XTRL
3133  00:EC63  22 6B EC 00    JSL Dump_to_Output
3134  00:EC67  68             PLA
3135  00:EC68  85 4E             STA OUTPUT_XTRL  ;RESTORE OUTPUT MODES
3136  00:EC6A  6B             RTL
3137
3138
3139                  ;************************************************************
3140                  ;*
3141                  ;*  Dump to output requests starting and ending addresses
3142                  ;*    from any input.  A formated output with header line
3143                  ;*    is sent to the printer.  Every 60 lines a new page with
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
3144                    ;*    a new header is started until the ending address is
3145                    ;*    reached. Then the routine returns to the caller.
3146                    ;*  The routine is aborted and a Cy=1 is returned if the
3147                    ;*    address inputted has a hex conversion error.
3148
3149
3150  00:EC6B                   Dump_to_Output:
3151  00:EC6B  A2 3C 00                 LDX #60
3152  00:EC6E  A9 04                    LDA #Flag3
3153  00:EC70  82 2F 00                 BRL G_XS28OUT
3154
3155
3156                    ;************************************************************
3157                    ;*
3158                    ;*  Dump 1 line to output requests a starting addresses
3159                    ;*    from any input.  A formated output with header line
3160                    ;*    is sent to the general output. Only one line (16 HEX chrs)
3161                    ;*    is sent. Then the routine returns to the caller.
3162                    ;*  The routine is aborted and a Cy=1 is returned if the
3163                    ;*    address inputted has a hex conversion error.
3164
3165
3166  00:EC73                   Dump_1_line_to_Output:
3167  00:EC73  A2 3C 00                 LDX #60
3168  00:EC76  A5 4E                    LDA OUTPUT_XTRL
3169  00:EC78  89 01                    BIT #1        ;IS LCD ON?
3170  00:EC7A  F0 04                    BEQ ?1        ;NO
3171  00:EC7C  A9 36                    LDA #Flag2+Flag3+Flag5+Flag6
3172  00:EC7E  80 02                    BRA ?2
3173
3174  00:EC80  A9 24            ?1          LDA #Flag3+Flag6
3175
3176  00:EC82  4C A2 EC         ?2          JMP G_XS28OUT
3177
3178
3179
3180                    ;************************************************************
3181                    ;*
3182                    ;*  Dump to Screen requests starting and ending addresses
3183                    ;*    from any input.  A formated output with header line
3184                    ;*    is sent to the LCD Screen.  Eight HEX characters per
3185                    ;*    line are displayed.  Every 14 lines a new page with
3186                    ;*    a new header is started until the ending address is
3187                    ;*    reached. Then the routine returns to the caller.
3188                    ;*  The routine is aborted and a Cy=1 is returned if the
```

```
3189                    ;*    address inputted has a hex conversion error.
3190
3191
3192  00:EC85                   Dump_to_Screen:
3193  00:EC85  A2 0C 00                 LDX #12
3194  00:EC88  A9 16                    LDA #Flag2+Flag3+Flag5
3195  00:EC8A  80 31                    BRA L_XS28OUT
3196
3197
3198
3199                    ;*************************************************************
3200                    ;*
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
3201                    ;*  Dump to Screen requests starting and ending addresses
3202                    ;*    from any input.  A formated output with header line
3203                    ;*    is sent to the LCD Screen.  Sixteen ASCII characters
3204                    ;*    per line are displayed.  Every 14 lines a new page with
3205                    ;*    a new header is started until the ending address is
3206                    ;*    reached. Then the routine returns to the caller.
3207                    ;*    A non-printable ASCII is displayed as a ".".
3208                    ;*  The routine is aborted and a Cy=1 is returned if the
3209                    ;*    address inputted has a hex conversion error.
3210
3211
3212  00:EC8C                   Dump_to_Screen_ASCII:
3213  00:EC8C  A2 0C 00                 LDX #12
3214  00:EC8F  A9 42                    LDA #Flag2+Flag7
3215  00:EC91  80 2A                    BRA L_XS28OUT
3216
3217
3218                    ;*************************************************************
3219                    ;*
3220                    ;*  Dump 1 line to Screen requests a starting addresses
3221                    ;*    from any input.  A formated output with header line
3222                    ;*    is sent to the printer. Only one line (16 HEX chrs)
3223                    ;*    is displayed. Then the routine returns to the caller.
3224                    ;*  The routine is aborted and a Cy=1 is returned if the
3225                    ;*    address inputted has a hex conversion error.
3226
3227
3228  00:EC93                   Dump_1_line_to_Screen:
3229  00:EC93  A2 0C 00                 LDX #12
3230  00:EC96  A9 36                    LDA #Flag2+Flag3+Flag5+Flag6
3231  00:EC98  80 23                    BRA L_XS28OUT
3232
3233
3234
3235
3236                    ;*************************************************************
3237                    ;*
3238                    ;*  DumpS28 requests starting and ending addresses from
3239                    ;*    any input.  A Motorola S28 formated output is sent
3240                    ;*    to the general output control. Each line contains
3241                    ;*    a check sum. When the ending address is reached
3242                    ;*    a S8 record is sent before this routine returns.
3243                    ;*  The routine is aborted and a Cy=1 is returned if the
3244                    ;*    address inputted has a hex conversion error.
3245
```

```
3246
3247  00:EC9A                   DumpS28:
3248  00:EC9A  A2 00 00                 LDX #0
3249  00:EC9D  A9 09                    LDA #Flag1+Flag4
3250  00:EC9F  82 00 00                 BRL G_XS28OUT
3251
3252
3253                      ;*************************************************************
3254
3255                      G_XS28OUT:   ;Common input for some dump routines
3256  00:ECA2  86 B1                    STX LINE_MAX
3257  00:ECA4  85 86                    STA DUMP_FLGS
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
3258  00:ECA6  82 27 00                BRL XS28OUT
3259
3260
3261  00:ECA9  41 64 64 72 65          Print_Head  .dc 'Address '
          73 73 A0
3262
3263  00:ECB1  53 38 30 34 30          XSLSTLINE .dc 'S804000000FB'
          30 30 30 30 30
          46 C2
3264
3265        00:0014             XS28BN   EQU 20        ;16 + 3 FOR ADDR
3266                                    ; + 1 FOR CKSUM
3267
3268
3269  00:ECBD                   L_XS28OUT:
3270
3271  00:ECBD  86 B1                    STX LINE_MAX
3272  00:ECBF  85 86                    STA DUMP_FLGS
3273  00:ECC1  A5 4E                    LDA OUTPUT_XTRL
3274  00:ECC3  48                          PHA
3275  00:ECC4  29 01                    AND #Bit0          ;Leave only LCD on
3276  00:ECC6  85 4E                       STA OUTPUT_XTRL
3277  00:ECC8  22 D0 EC 00                 JSL XS28OUT
3278  00:ECCC  68                       PLA
3279  00:ECCD  85 4E                       STA OUTPUT_XTRL  ;RESTORE OUTPUT MODES
3280  00:ECCF  6B                       RTL
3281
3282
3283
3284  00:ECD0                   XS28OUT:
3285
3286  00:ECD0  A5 86                    LDA DUMP_FLGS
3287  00:ECD2  89 20                    BIT #Flag6      ;single line?
3288  00:ECD4  F0 08                    BEQ ?S|E        ;no
3289
3290  00:ECD6  22 04 F2 00              JSL Get_Address  ;starting address
3291  00:ECDA  B0 08                    BCS ?xx
3292  00:ECDC  80 08                       BRA ?S0
3293
3294  00:ECDE  22 6F F2 00   ?S|E     JSL Get_S_Address  ;starting address
3295  00:ECE2  90 02                    BCC ?S0
3296
3297  00:ECE4  38            ?xx      SEC           ;CANCELLING TO EXIT
3298  00:ECE5  6B                       RTL
3299
```

```
3300  00:ECE6  A6 63          ?S0     LDX TMP2      ;move start addr to TMP0
3301  00:ECE8  86 5D                  STX TMP0          ;
3302  00:ECEA  86 60                  STX TMP1      ;TMP1 IS USED IN ALTER MEM
3303
3304  00:ECEC  A5 65                  LDA TMP2+2        ;
3305  00:ECEE  85 5F                  STA TMP0+2        ;
3306  00:ECF0  85 62                  STA TMP1+2
3307
3308                    ;    SETUP THE ENDING ADDRESS      in TMP2
3309  00:ECF2  A5 86                  LDA DUMP_FLGS
3310  00:ECF4  89 20                  BIT #Flag6     ;single line?
3311  00:ECF6  F0 1D                  BEQ ?E0        ;no
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
3312  00:ECF8  89 05              BIT #5
3313  00:ECFA  D0 04              BNE ?S1
3314  00:ECFC  A9 10              LDA #16
3315  00:ECFE  80 02              BRA ?S3
3316
3317  00:ED00  A9 08        ?S1    LDA #8
3318  00:ED02  18           ?S3    CLC
3319  00:ED03  65 5D              ADC TMP0
3320  00:ED05  85 63              STA TMP2
3321
3322  00:ED07  A5 5E              LDA TMP0+1
3323  00:ED09  69 00              ADC #0
3324  00:ED0B  85 64              STA TMP2+1
3325
3326  00:ED0D  A5 5F              LDA TMP0+2
3327  00:ED0F  69 00              ADC #0
3328  00:ED11  85 65              STA TMP2+2
3329  00:ED13  80 0C              BRA ?E2
3330
3331
3332  00:ED15                 ?E0
3333  00:ED15  A9 05              LDA #05
3334  00:ED17  22 FD F1 00        JSL POSITION_TEXT_CURSOR
3335
3336  00:ED1B  22 8C F2 00        JSL Get_E_Address
3337  00:ED1F  B0 C3              BCS ?xx
3338
3339  00:ED21  A9 03        ?E2    LDA #3
3340  00:ED23  22 F1 F1 00        JSL CLEAR_LCD_DISPLAY
3341
3342  00:ED27  A2 00 00           LDX #0
3343  00:ED2A  86 59              STX WRAP
3344  00:ED2C  86 AF              STX LINE_CNT
3345
3346  00:ED2E  22 57 ED 00  ?0     JSL P_HEADER
3347  00:ED32  B0 19              BCS ?xxx
3348
3349
3350  00:ED34  A5 86              LDA DUMP_FLGS
3351  00:ED36  89 20                 BIT #Flag6      ;single line
3352  00:ED38  D0 11                 BNE ?3.1
3353
3354  00:ED3A  89 01              BIT #Flag1
3355  00:ED3C  F0 09              BEQ ?3        ;NOT S28 FORMAT
3356
```

```
3357                            ;WRITE LAST LINE
3358  00:ED3E  A9 00            ?1      LDA #0
3359  00:ED40  A2 B1 EC                LDX #XSLSTLINE
3360  00:ED43  22 A1 F3 00              JSL PUT_STR
3361
3362  00:ED47  22 B2 F1 00      ?3      JSL GET_CHR
3363  00:ED4B  18               ?3.1    CLC
3364  00:ED4C  6B                       RTL
3365
3366  00:ED4D  22 B2 F1 00      ?xxx    JSL GET_CHR
3367  00:ED51  38                       SEC
3368  00:ED52  6B                       RTL
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
3369
3370                    ;*******************************************************
3371                    .page
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
3372
3373                    ;*   ******  Dump_It  *******
3374                    ;*
3375                    ;*   Enter here for custom dump routines. Have DUMP_FLGS
3376                    ;*     in an 8 bit Areg and the number of lines per page
3377                    ;*     to print/display in a 16 bit Xreg.
3378                    ;*   The starting address must be in TMP0 (3bytes) and
3379                    ;*     the ending address must be in TMP2 (3bytes).
3380                    ;*
3381
3382  00:ED53                   Dump_It:
3383  00:ED53  86 B1                  STX LINE_MAX
3384  00:ED55  85 86                  STA DUMP_FLGS
3385
3386                    ;The following subroutine makes each line of output until
3387                    ;  all requested bytes are sent.
3388
3389  00:ED57  A5 86            P_HEADER LDA DUMP_FLGS
3390  00:ED59  89 04                  BIT #Flag3      ;formated oputput
3391  00:ED5B  F0 3C                  BEQ XS28OUTA    ;NO Page HEADER
3392  00:ED5D
3393  00:ED5D  A2 A9 EC               LDX #Print_Head  ;"Address"
3394  00:ED60  A9 00                  LDA #0
3395  00:ED62  22 A1 F3 00            JSL PUT_STR
3396                          ;now print address for columns
3397  00:ED66  A2 10 00               LDX #16
3398  00:ED69  A5 86                  LDA DUMP_FLGS
3399  00:ED6B  89 02                  BIT #Flag2      ;LCD ?
3400  00:ED6D  F0 03                  BEQ ?H0
3401  00:ED6F  A2 08 00               LDX #8       ;# of lines for LCD only
3402  00:ED72  A5 5D            ?H0     LDA TMP0     ;LOW START ADDR
3403  00:ED74  48               ?H1     PHA
3404  00:ED75  DA                     PHX
3405  00:ED76  20 D7 F0               JSR BINASC
3406  00:ED79  22 BC F1 00            JSL PUT_CHR
3407  00:ED7D  A9 20                  LDA #' '
3408  00:ED7F  22 BC F1 00            JSL PUT_CHR
3409  00:ED83  22 BC F1 00            JSL PUT_CHR
3410  00:ED87  FA                     PLX
3411  00:ED88  68                     PLA
3412  00:ED89  CA                     DEX
3413  00:ED8A  F0 03                  BEQ ?H2
3414  00:ED8C  1A                     INC A
3415  00:ED8D  80 E5                  BRA ?H1
3416
```

```
3417  00:ED8F  A9 0D          ?H2     LDA #C_RETURN
3418  00:ED91  22 BC F1 00            JSL PUT_CHR
3419  00:ED95  22 BC F1 00            JSL PUT_CHR
3420
3421  00:ED99  A5 59          XS28OUTA LDA WRAP
3422  00:ED9B  F0 02                  BEQ XWH00
3423  00:ED9D  38                     SEC
3424  00:ED9E  6B                     RTL
3425
3426  00:ED9F                 XWH00
3427  00:ED9F  64 66                  STZ TMP4
3428  00:EDA1  64 67                  STZ TMP4+1     ;CLEAR CKSUM
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
3429  00:EDA3  A9 14          LDA #XS28BN
3430  00:EDA5  85 57          STA TMPC      ;TMPC = 16+4 FOR SHORT
3431
3432  00:EDA7  A5 86          LDA DUMP_FLGS
3433  00:EDA9  89 10          BIT #Flag5    ;LCD ?
3434  00:EDAB  F0 04          BEQ ?XH0
3435  00:EDAD  A9 0C          LDA #12       ;LCD = 8+4
3436  00:EDAF  85 57          STA TMPC
3437
3438  00:EDB1  A5 86    ?XH0  LDA DUMP_FLGS
3439  00:EDB3  89 01          BIT #Flag1
3440  00:EDB5  F0 27          BEQ XWH1A     ;not S28 format
3441
3442  00:EDB7  A9 53          LDA #'S'      ;
3443  00:EDB9  22 BC F1 00    JSL PUT_CHR
3444  00:EDBD  A9 32          LDA #'2'      ;OUTPUT S2
3445  00:EDBF  22 BC F1 00    JSL PUT_CHR
3446
3447  00:EDC3  20 89 EE       JSR DCMP      ;EA-SA (TMP2-TMP0) DIFF
3448  00:EDC6  A5 5C          LDA DIFF+2    ;IN LOC DIFF+2 (IE BANK) OF
3449  00:EDC8  D0 0F          BNE XWH10     ;DIFF  GT 65536
3450  00:EDCA  A5 5B          LDA DIFF+1
3451  00:EDCC  D0 0B          BNE XWH10     ;DIFF > 256
3452  00:EDCE  A5 5A          LDA DIFF
3453  00:EDD0  C9 0F          CMP #15
3454  00:EDD2  B0 05          BCS XWH10     ;DIFF > 16
3455  00:EDD4  18             CLC           ;ADD 3 FOR ADDR
3456  00:EDD5  69 05          ADC #$05      ;ADD 1 FOR CKSUM
3457  00:EDD7  85 57          STA TMPC      ;ADD 1 FOR BYTE CNT
3458
3459  00:EDD9  A5 57    XWH10 LDA TMPC      ;OUTPUT BYTE COUNT
3460  00:EDDB  20 84 F3       JSR CKNOUT    ;RCC CNT IN A
3461
3462  00:EDDE  C6 57    XWH1A DEC TMPC      ;BACK OUT FOR ADDRESS
3463  00:EDE0  C6 57          DEC TMPC      ;AND BYTE COUNT
3464  00:EDE2  C6 57          DEC TMPC
3465  00:EDE4  C6 57          DEC TMPC
3466
3467  00:EDE6  A5 5F          LDA TMP0+2    ;output the address
3468  00:EDE8  20 84 F3       JSR CKNOUT    ;ADD BANK TO CKSM
3469  00:EDEB  A5 86          LDA DUMP_FLGS
3470  00:EDED  89 01          BIT #Flag1
3471  00:EDEF  D0 06          BNE XWH1b     ;no colon after bank addr
3472  00:EDF1  A9 3A          LDA #':'
3473  00:EDF3  22 BC F1 00    JSL PUT_CHR
```

```
3474
3475  00:EDF7  A5 5E          XWH1b    LDA TMP0+1
3476  00:EDF9  20 84 F3               JSR CKNOUT     ;ADD HIGH ADDRESS BYTE TO CKSM
3477  00:EDFC  A5 5D                  LDA TMP0
3478  00:EDFE  20 84 F3               JSR CKNOUT     ;ADD LOW ADDRESS BYTE TO CKSM
3479  00:EE01  A5 86                  LDA DUMP_FLGS
3480  00:EE03  89 01                  BIT #Flag1
3481  00:EE05  D0 06                  BNE XWH2       ;no space after ADDR
3482  00:EE07  A9 20                  LDA #' '
3483  00:EE09  22 BC F1 00            JSL PUT_CHR
3484
3485  00:EE0D  A5 86          XWH2     LDA DUMP_FLGS
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
3486  00:EE0F  89 40          BIT #Flag7
3487  00:EE11  F0 12          BEQ XWH2C      ;not ASCII
3488  00:EE13  A7 5D          LDA [TMP0]     ;WRITE OUT DATA BYTES
3489  00:EE15  C9 7F          CMP #$7F       ;as ASCII characters
3490  00:EE17  B0 04          BCS XWH2A      ;non ASCII
3491  00:EE19  C9 20          CMP #$20
3492  00:EE1B  B0 02          BCS XWH2B
3493  00:EE1D  A9 60    XWH2A LDA #'`'       ;non ASCII
3494  00:EE1F  22 BC F1 00 XWH2B JSL PUT_CHR
3495  00:EE23  80 05          BRA XWH3
3496
3497  00:EE25  A7 5D    XWH2C LDA [TMP0]     ;WRITE OUT DATA BYTES
3498  00:EE27  20 84 F3       JSR CKNOUT     ;INC CKSUM, PRESERVES Areg
3499
3500  00:EE2A  A5 86    XWH3  LDA DUMP_FLGS
3501  00:EE2C  89 04          BIT #Flag3
3502  00:EE2E  F0 06          BEQ XWH3a      ;no formating spaces
3503  00:EE30  A9 20          LDA #' '
3504  00:EE32  22 BC F1 00    JSL PUT_CHR
3505
3506  00:EE36  20 54 F1 XWH3a JSR INCTMP0    ;INC SA
3507  00:EE39  C6 57          DEC TMPC       ;REMAINING BYTE COUNT
3508  00:EE3B  D0 D0          BNE XWH2       ;LOOP FOR 8 OR 16 BYTES
3509  00:EE3D  A5 86          LDA DUMP_FLGS
3510  00:EE3F  89 08          BIT #Flag4
3511  00:EE41  F0 08          BEQ XWH3b      ;no checksum out
3512
3513  00:EE43  A5 66          LDA TMP4
3514  00:EE45  49 FF          EOR #$FF       ;we want 1's complement
3515  00:EE47  22 70 F3 00    JSL SEND_HEX_OUT       ;WRITE CKSUM
3516
3517  00:EE4B  22 D6 F1 00 XWH3b JSL SEND_CR
3518  00:EE4F  20 89 EE       JSR DCMP
3519  00:EE52  90 24          BCC XWH7       ;safety play  SA is > EA
3520
3521  00:EE54  A6 AF    XWH5  LDX LINE_CNT
3522  00:EE56  E8             INX
3523  00:EE57  86 AF          STX LINE_CNT
3524
3525  00:EE59  E4 B1          CPX LINE_MAX
3526  00:EE5B  F0 03          BEQ XWH6
3527  00:EE5D  82 39 FF       BRL XS28OUTA   ;LOOP WHILE EA >= SA
3528
3529  00:EE60  A2 00 00 XWH6  LDX #0         ;end of page
3530  00:EE63  86 AF          STX LINE_CNT
```

```
3531  00:EE65  A5 86              LDA DUMP_FLGS
3532  00:EE67  89 02              BIT #Flag2      ;LCD ?
3533  00:EE69  F0 0F              BEQ XWH8        ;no
3534  00:EE6B  22 B2 F1 00        JSL GET_CHR     ;yes
3535  00:EE6F  A9 03              LDA #3
3536  00:EE71  22 F1 F1 00        JSL CLEAR_LCD_DISPLAY
3537  00:EE75  82 DF FE           BRL P_HEADER
3538
3539  00:EE78  18           XWH7  CLC         ;ALL DONE
3540  00:EE79  6B                 RTL         ;RETURN TO XS28OUT
3541
3542  00:EE7A  A5 86        XWH8  LDA DUMP_FLGS
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
3543  00:EE7C  89 04              BIT #Flag3    ;Header
3544  00:EE7E  F0 06              BEQ XWH9
3545  00:EE80  A9 0C              LDA #$0C      ;FORM FEED
3546  00:EE82  22 BC F1 00        JSL PUT_CHR
3547  00:EE86  82 CE FE      XWH9  BRL P_HEADER
3548
3549
3550
3551                  ;*******************************************************
3552                  ; This routine calculates the difference between where we
3553                  ;  are and the end address. Athree byte value is stored
3554                  ;  in DIFF(0,1,2).  The Z flag will show zero upon return.
3555
3556  00:EE89  38           DCMP  SEC         ;TMP2-TMP0 DBL SUBTRACT
3557  00:EE8A  A5 63              LDA TMP2    ;SUBTRACT LOW ADDRESS
3558  00:EE8C  E5 5D              SBC TMP0
3559  00:EE8E  85 5A              STA DIFF
3560
3561  00:EE90  A5 64              LDA TMP2+1
3562  00:EE92  E5 5E              SBC TMP0+1
3563  00:EE94  85 5B              STA DIFF+1    ;OR LO FOR EQU TEST
3564
3565  00:EE96  A5 65              LDA TMP2+2    ;NOW DO BANK REGISTER
3566  00:EE98  E5 5F              SBC TMP0+2
3567  00:EE9A  85 5C              STA DIFF+2    ;SAVE DIFFERENCE IN BANK SIZES
3568  00:EE9C  05 5A              ORA DIFF      ;OR LO FOR EQU TEST
3569  00:EE9E  05 5B              ORA DIFF+1
3570  00:EEA0  60                   RTS
3571
3572  00:EEA1
3573                  .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
3574                    ;***********************************************
3575                    ;*
3576                    ;*      The Slash (/) command is to allow host computers quick access
3577                    ;*      to memory locations.  It has many forms:
3578                    ;*
3579                    ;*
3580                    ;*      /<c_return> returns the current value of the address
3581                    ;*              pointer.
3582                    ;*
3583                    ;*      /<SPACE> returns DATA at current memory location and
3584                    ;*              increments address pointer.
3585                    ;*
3586                    ;*      /YY<SPACE> writes YY at current memory location pointer,
3587                    ;*              re-reads the location and returns the DATA
3588                    ;*              at that location (as a check for writeable mem)
3589                    ;*              then increments the memory location pointer.
3590                    ;*
3591                    ;*      /XXXX<SPACE> changes the address pointer to 00:XXXX and returns
3592                    ;*              the DATA at that location (as a check for writeable
3593                    ;*              mem) then increments the memory location pointer.
3594                    ;*
3595                    ;*      /bb:XXXX<SPACE> changes the address pointer to bb:XXXX and
returns
3596                    ;*              the DATA at that location (as a check for writeable
3597                    ;*              mem) then increments the memory location pointer.
3598                    ;*
3599                    ;*      /XXXXYY<SPACE> changes the address pointer to 00:XXXX and
3600                    ;*              writes YY at current memory location pointer,
3601                    ;*              re-reads the location and returns the DATA
3602                    ;*              at that location (as a check for writeable mem)
3603                    ;*              then increments the memory location pointer.
3604                    ;*
3605                    ;*      /bb:XXXXYY<SPACE> changes the address pointer to bb:XXXX and
3606                    ;*              writes YY at current memory location pointer,
3607                    ;*              re-reads the location and returns the DATA
3608                    ;*              at that location (as a check for writeable mem)
3609                    ;*              then increments the memory location pointer.
3610                    ;*
3611                    ;*      Any error in input format will result in a NAK return
3612                    ;*
3613
3614  00:EEA1             SLASH:
3615  00:EEA1  78               SEI         ;First, kill the echo!
3616  00:EEA2  A5 43            LDA SFLAG3
3617  00:EEA4  29 20            AND #ECHOFF
```

```
3618  00:EEA6  85 57          STA TMPC      ;SAVE CURRENT STATE OF ECHO OFF
3619  00:EEA8  A9 20          LDA #ECHOFF   ;SET ECHO OFF
3620  00:EEAA  14 43          TRB SFLAG3
3621  00:EEAC  58             CLI
3622
3623  00:EEAD  A9 00          LDA #0
3624  00:EEAF  85 6B          STA TMP6+2
3625
3626  00:EEB1  22 B2 F1 00    JSL GET_CHR
3627
3628  00:EEB5  C9 1B          CMP #ESC
3629  00:EEB7  D0 03             BNE ?2
3630  00:EEB9  82 7E 00       BRL SLASH|OUT
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
3631
3632  00:EEBC  C9 0D          ?2     CMP #C_RETURN  ;CR...RETURN CURRENT ADDRESS
POINTER
3633  00:EEBE  D0 06                   BNE ?3
3634  00:EEC0  20 46 EF              JSR RETURN_ADDR ;return 24 bit address pointer
3635  00:EEC3  82 6E 00              BRL SLASH|END
3636
3637  00:EEC6  C9 20          ?3     CMP #' '       ;SPACE...RETURN A BYTE!
3638  00:EEC8  D0 03                   BNE ?4
3639  00:EECA  82 5F 00              BRL RETURN_BYTE  ;just send byte @ pointer location
3640
3641  00:EECD  20 5C EF       ?4     JSR DO_HEX
3642  00:EED0  90 03                   BCC ?5
3643  00:EED2  82 65 00              BRL SLASH|OUT          ;NOT HEX
3644
3645  00:EED5  A5 58          ?5     LDA TMPC+1    ;might be hi byte of 16 bits
3646  00:EED7  85 6A                 STA TMP6+1
3647
3648  00:EED9  22 B2 F1 00           JSL GET_CHR
3649  00:EEDD  C9 20                 CMP #' '
3650  00:EEDF  F0 42                 BEQ WRITE_BYTE         ;first byte was a DATA byte
3651  00:EEE1  C9 3A                 CMP #':'
3652  00:EEE3  D0 18                 BNE NO_BANK  ;just a 16 bit address
3653
3654  00:EEE5  A5 58                 LDA TMPC+1     ;ITS A BANK ADDRESS
3655  00:EEE7  85 6B                 STA TMP6+2     ;save Bank address
3656  00:EEE9  22 B2 F1 00           JSL GET_CHR     ;get  hi byte of addr
3657  00:EEED  20 5C EF              JSR DO_HEX
3658  00:EEF0  90 03                   BCC ?8
3659  00:EEF2  82 45 00              BRL SLASH|OUT          ;NOT HEX
3660
3661  00:EEF5  A5 58          ?8     LDA TMPC+1    ;save hi byte of 24 bit address
3662  00:EEF7  85 6A                 STA TMP6+1
3663
3664  00:EEF9  22 B2 F1 00    ?9     JSL GET_CHR   ;now get low byte of address
3665
3666  00:EEFD  C9 1B          NO_BANK CMP #ESC
3667  00:EEFF  F0 39                 BEQ SLASH|OUT
3668  00:EF01  20 5C EF              JSR DO_HEX      ;2 ASCII = low byte of address
3669  00:EF04  B0 34                 BCS SLASH|OUT          ;NOT HEX
3670  00:EF06  A5 58                 LDA TMPC+1
3671  00:EF08  85 69                 STA TMP6          ;save low byte for pointer
3672
3673  00:EF0A  A6 69                 LDX TMP6         ;SET ADDRESS POINTER
3674  00:EF0C  86 5D                 STX TMP0         ;TO NEW VALUE
```

```
3675  00:EF0E  A5 6B              LDA TMP6+2
3676  00:EF10  85 5F              STA TMP0+2
3677
3678  00:EF12  22 B2 F1 00   ?gc   JSL GET_CHR  ;do we have more data
3679  00:EF16  C9 20              CMP #' '
3680  00:EF18  F0 12              BEQ RETURN_BYTE  ;send back byte at new address
3681
3682  00:EF1A  C9 1B              CMP #ESC
3683  00:EF1C  F0 1C              BEQ SLASH|OUT
3684  00:EF1E  20 5C EF           JSR DO_HEX
3685  00:EF21  B0 17              BCS SLASH|OUT        ;NOT HEX
3686                                   ;now write this byte at the new address
3687
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
3688                    WRITE_BYTE:         ;WRITE data byte
3689  00:EF23  A5 58               LDA TMPC+1
3690  00:EF25  87 5D               STA [TMP0]
3691  00:EF27  A9 00               LDA #0      ;DUMMY WRITE to clear data buss and
3692  00:EF29  8D 05 E0            STA $E005   ;prevent a read-back echo from no-select.
3693
3694                    RETURN_BYTE:        ;READ data byte
3695  00:EF2C  A7 5D               LDA [TMP0]
3696  00:EF2E  20 8C F3            JSR WRT2OUT  ;send byte back as 2 ASCII
3697  00:EF31  20 54 F1            JSR INCTMP0  ;Increment TMP0 by 1
3698
3699  00:EF34                 SLASH|END:
3700  00:EF34  A5 57                  LDA TMPC    ;good return
3701  00:EF36  04 43               TSB SFLAG3   ;restore echo mode
3702  00:EF38  18                  CLC
3703  00:EF39  6B                  RTL
3704
3705                    SLASH|OUT:  ;error exit
3706  00:EF3A  A9 15                  LDA #NAK    ;SEND NAK
3707  00:EF3C  22 BC F1 00            JSL PUT_CHR
3708  00:EF40  A5 57                  LDA TMPC    ;restore echo mode
3709  00:EF42  04 43               TSB SFLAG3
3710  00:EF44  38                  SEC
3711  00:EF45  6B                  RTL
3712
3713
3714  00:EF46                 RETURN_ADDR:
3715  00:EF46  A5 5F               LDA TMP0+2
3716  00:EF48  20 8C F3            JSR WRT2OUT
3717  00:EF4B  A9 3A                  LDA #':'
3718  00:EF4D  22 BC F1 00         JSL PUT_CHR
3719  00:EF51  A5 5E                  LDA TMP0+1
3720  00:EF53  20 8C F3            JSR WRT2OUT
3721  00:EF56  A5 5D               LDA TMP0
3722  00:EF58  20 8C F3               JSR WRT2OUT
3723  00:EF5B  60                  RTS
3724
3725  00:EF5C                 DO_HEX:
3726  00:EF5C  20 F0 F0            JSR HEXIN
3727  00:EF5F  B0 14               BCS ?ERR            ;NOT HEX
3728  00:EF61  0A                  ASL A
3729  00:EF62  0A                  ASL A
3730  00:EF63  0A                  ASL A
3731  00:EF64  0A                  ASL A
3732  00:EF65  85 58               STA TMPC+1
```

```
3733  00:EF67  22 B2 F1 00     ?GP     JSL GET_CHR
3734  00:EF6B  20 F0 F0                JSR HEXIN
3735  00:EF6E  B0 05                   BCS ?ERR              ;NOT HEX
3736  00:EF70  05 58                   ORA TMPC+1
3737  00:EF72  85 58                   STA TMPC+1
3738  00:EF74  18                      CLC
3739  00:EF75                       ?ERR:
3740  00:EF75  60                      RTS
3741
3742
3743                        .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
3744
3745                ;***********************************************
3746                ;*
3747                ;*      The Pipe (|) command is to allow host computers quick access
3748                ;*      to REGISTER locations.  It has many forms:
3749                ;*
3750                ;*
3751                ;*      |<SPACE>  returns the current value of ALL the
3752                ;*              REGISTERS. Each register sent is separated
3753                ;*              by a space. The order is:
3754                ;*
3755                ;*                  1) Program Counter
3756                ;*                  2) A reg (16 bits)
3757                ;*                  3) X reg (16 bits)
3758                ;*                  4) Y reg (16 bits)
3759                ;*                  5) Stack Pointer (16 bits)
3760                ;*                  6) Direct Page (16 bits)
3761                ;*                  7) Flag reg (8 bits)
3762                ;*                  8) Bank reg (8 bits)
3763                ;*
3764                ;*
3765                ;*
3766                ;*      |Pbb:xxxx   Replaces the program counter with bb:xxxx.
3767                ;*
3768                ;*      |AXXXX      Replaces the contents of the Areg with xxxx.
3769                ;*
3770                ;*      |Xxxxx      Replaces the contents of the Xreg with xxxx.
3771                ;*
3772                ;*      |Yxxxx      Replaces the contents of the Yreg with xxxx.
3773                ;*
3774                ;*      |Sxxxx     Replaces the contents of the Stack Ptr with xxxx.
3775                ;*
3776                ;*      |Dxxxx      Replaces the contents of the Dircet Page register
3777                ;*                with xxxx.
3778                ;*
3779                ;*      |Fxx       Replaces the contents of the Flag reg with xx.
3780                ;*
3781                ;*      |Bxx       Replaces the contents of the Data Bank register
3782                ;*                with xx.
3783                ;*   NOTES:
3784                ;*      If the Areg in an eight bit mode all 16 bits will be changed
3785                ;*        the mode will remain 8 bits.
3786                ;*      If the Xreg and Yreg are in 8 bit modes only the low order
3787                ;*        8 bits will be changed.
3788                ;*      These values are written to memory locations and are entered
```

```
3789                    ;*      into the registers only on the return from software break.
3790                    ;*    Echo mode (if on) will be turned off for the duration of
3791                    ;*      this command. It will be restored upon completion.
3792                    ;*
3793                    ;*    The monitor will return a ACK upon satisfactory completion
3794                    ;*
3795                    ;*    Any error in input format will result in a NAK return
3796                    ;*
3797
3798  00:EF76             PIPE:
3799  00:EF76  78                 SEI         ;First, kill the echo!
3800  00:EF77  A5 43              LDA SFLAG3
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
3801  00:EF79  29 20              AND #ECHOFF
3802  00:EF7B  85 57              STA TMPC      ;SAVE CURRENT STATE OF ECHO OFF
3803  00:EF7D  A9 20              LDA #ECHOFF   ;SET ECHO OFF
3804  00:EF7F  14 43              TRB SFLAG3
3805  00:EF81  58                 CLI
3806
3807  00:EF82  22 B2 F1 00  ?1    JSL GET_CHR   ;get the register ID
3808  00:EF86  A0 00 00           LDY #0
3809  00:EF89  D9 45 F0     ?2    CMP Reg_ID,Y
3810  00:EF8C  F0 09              BEQ ?3
3811  00:EF8E  C8                 INY
3812  00:EF8F  C0 09 00           CPY #9
3813  00:EF92  D0 F5              BNE ?2
3814  00:EF94  82 6D 00           BRL PIPE|ERR
3815
3816  00:EF97  A2 00 00     ?3    LDX #0
3817  00:EF9A  86 69              STX TMP6
3818
3819  00:EF9C  B9 4E F0           LDA Reg_Size,Y
3820  00:EF9F  0A                 ASL A  ;X2
3821  00:EFA0  AA                 TAX
3822  00:EFA1  7C 60 F0           JMP (Reg_Strt,X)
3823
3824
3825  00:EFA4               THREE|BY:
3826  00:EFA4  22 B2 F1 00  ?1    JSL GET_CHR   ;get BANK byte
3827  00:EFA8  20 5C EF           JSR DO_HEX
3828  00:EFAB  90 03                 BCC ?2
3829  00:EFAD  82 54 00           BRL PIPE|ERR  ;NOT HEX
3830
3831  00:EFB0  A5 58        ?2    LDA TMPC+1    ;save hi byte of 24 bit address
3832  00:EFB2  85 6B              STA TMP6+2
3833
3834  00:EFB4  22 B2 F1 00  ?3    JSL GET_CHR
3835  00:EFB8  C9 3A              CMP #':'      ;MUST HAVE BANK SEPARATOR
3836  00:EFBA  F0 03              BEQ TWO|BY
3837  00:EFBC  82 45 00           BRL PIPE|ERR
3838
3839  00:EFBF               TWO|BY:
3840  00:EFBF  22 B2 F1 00  ?1    JSL GET_CHR   ;get HI byte
3841  00:EFC3  20 5C EF           JSR DO_HEX
3842  00:EFC6  90 03                 BCC ?2
3843  00:EFC8  82 39 00           BRL PIPE|ERR  ;NOT HEX
3844
3845  00:EFCB  A5 58        ?2    LDA TMPC+1    ;save hi byte
```

```
3846  00:EFCD  85 6A              STA TMP6+1
3847
3848  00:EFCF                  ONE|BY:
3849  00:EFCF  22 B2 F1 00     ?1   JSL GET_CHR   ;get  LOW byte
3850  00:EFD3  20 5C EF            JSR DO_HEX
3851  00:EFD6  90 03                  BCC ?2
3852  00:EFD8  82 29 00            BRL PIPE|ERR   ;NOT HEX
3853
3854  00:EFDB  A5 58           ?2   LDA TMPC+1    ;save hi byte of 24 bit address
3855  00:EFDD  85 69               STA TMP6
3856
3857  00:EFDF  B9 57 F0            LDA Reg_Addr,Y
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
3858  00:EFE2  85 60              STA TMP1      ;LOW ORDER REG ADDR
3859  00:EFE4  A9 DF              LDA #$DF
3860  00:EFE6  85 61              STA TMP1+1    ;HI ORDER REG ADDR
3861
3862  00:EFE8  B9 4E F0           LDA Reg_Size,Y
3863  00:EFEB  A8                 TAY
3864  00:EFEC  88                 DEY
3865
3866  00:EFED  B9 69 00      ?3    LDA TMP6,Y
3867  00:EFF0  91 60              STA (TMP1),Y
3868  00:EFF2  88                 DEY
3869  00:EFF3  C0 FF FF           CPY #$FFFF
3870  00:EFF6  D0 F5              BNE ?3
3871  00:EFF8  A9 06              LDA #ACK
3872  00:EFFA  22 BC F1 00        JSL PUT_CHR
3873
3874  00:EFFE                PIPE|END:
3875  00:EFFE  A5 57                LDA TMPC      ;good return
3876  00:F000  04 43              TSB SFLAG3    ;restore echo mode
3877  00:F002  18                 CLC
3878  00:F003  6B                 RTL
3879
3880                    PIPE|ERR   ;error exit
3881  00:F004  A9 15                LDA #NAK      ;SEND NAK
3882  00:F006  22 BC F1 00          JSL PUT_CHR
3883  00:F00A  A5 57                LDA TMPC      ;restore echo mode
3884  00:F00C  04 43              TSB SFLAG3
3885  00:F00E  38                 SEC
3886  00:F00F  6B                 RTL
3887
3888
3889  00:F010                RET|REGS:
3890  00:F010  A9 DF              LDA #$DF
3891  00:F012  85 61              STA TMP1+1
3892  00:F014  A2 01 00           LDX #1
3893
3894  00:F017  BF 57 F0 00   ?2    LDA Reg_Addr,X
3895  00:F01B  85 60              STA TMP1
3896
3897  00:F01D  BF 4E F0 00        LDA Reg_Size,X
3898  00:F021  A8                 TAY
3899  00:F022  88                 DEY
3900  00:F023  B1 60         ?3    LDA (TMP1),Y
3901  00:F025  20 8C F3             JSR WRT2OUT
3902  00:F028  88                 DEY
```

```
3903  00:F029  C0 02 00          CPY #2
3904  00:F02C  D0 06             BNE ?4
3905  00:F02E  A9 3A             LDA #':'
3906  00:F030  22 BC F1 00       JSL PUT_CHR
3907  00:F034  C0 FF FF       ?4  CPY #$FFFF
3908  00:F037  D0 EA             BNE ?3
3909  00:F039  22 DF F1 00       JSL SEND_SPACE
3910  00:F03D  E8                INX
3911  00:F03E  E0 09 00          CPX #9
3912  00:F041  D0 D4             BNE ?2
3913  00:F043  80 B9             BRA PIPE|END
3914
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
3915
3916
3917  00:F045  20 50 41 58 59  Reg_ID   .BYTE ' PAXYSDFB'
           53 44 46 42
3918  00:F04E  00 03 02 02 02  Reg_Size .BYTE 0,3,2,2,2,2,2,1,1
           02 02 01 01
3919  00:F057  00 8C 80 82 84 Reg_Addr .BYTE
0,<PCL,<ACC,<XREG,<YREG,<STK|PTR,<DIRREG,<FLGS,<DBREG
           86 88 8B 8A
3920  00:F060  10F0 CFEF BFEF         Reg_Strt .WORD RET|REGS,ONE|BY,TWO|BY,THREE|BY
           A4EF
3921
3922                        .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'Debug routines, Loads & Dumps'

```
3923                    ;****************************************
3924
3925  00:F068                   DSPLYDEC:
3926  00:F068  20 76 F1              JSR DECTMP0
3927  00:F06B  80 03                  BRA DSPLYOLD
3928
3929  00:F06D                   DSPLYINC:
3930  00:F06D  20 54 F1              JSR INCTMP0
3931  00:F070                   DSPLYOLD:
3932  00:F070  20 46 EF              JSR RETURN_ADDR
3933  00:F073  22 DF F1 00           JSL SEND_SPACE
3934  00:F077  A7 5D                 LDA [TMP0]
3935  00:F079  20 8C F3              JSR WRT2OUT
3936  00:F07C  6B                    RTL
3937
3938
3939                       .END
3940
3941
3942                          .STTL 'UTILITY ROUTINES'
3943                          .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'UTILITY ROUTINES'

```
3944  00:F07C                       include r_utils.asm
3945                      ;FILE = R_UTILS.ASM
3946                      ;DATE = 12-29-94
3947
3948
3949
3950
3951                      ; READ HEX BYTE AND RETURN IN A, AND CY=0
3952                      ; IF SPACE OR NON-HEX kCOMMA CY=1
3953
3954                      ;* Routine:  GET_HEX   {no echo}
3955                      ;*
3956                      ;* Reg Used: ACC & X
3957                      ;* Var Used: TEMP
3958                      ;* Routines Called: GET_CHR,ASCBIN
3959                      ;* Returned Reg: Acc  X & Y REGS are PRESERVED
3960                      ;*
3961
3962  00:F07D  DA               GET_HEX  PHX          ;SAVE X
3963  00:F07E  22 B2 F1 00            JSL GET_CHR
3964  00:F082  C9 20                  CMP #' '      ;SPACE?
3965  00:F084  D0 03                  BNE ?3
3966  00:F086  FA                        PLX        ;IF SPACE IN FIRST ASCII, then
3967  00:F087  38                        SEC        ;Cy = 1 and Areg = space
3968  00:F088  6B                     RTL
3969
3970  00:F089  85 70            ?3     STA TEMP      ;SAVE 1ST CHAR
3971  00:F08B  22 B2 F1 00            JSL GET_CHR   ;READ NEXT CHAR
3972
3973  00:F08F  20 BF F0               JSR ASCBIN    ;CY = 1  IF BAD DATA
3974  00:F092  B0 03                  BCS RDOERR
3975  00:F094  FA                     PLX         ;RESTORE X
3976  00:F095  18                     CLC
3977  00:F096  6B                     RTL
3978
3979  00:F097  FA               RDOERR   PLX          ;ON RETURN..CY =1 IF BAD DATA
3980  00:F098  38                     SEC         ;AND Areg = 0
3981  00:F099  A9 00                     LDA #0
3982  00:F09B  6B                     RTL
3983
3984
3985
3986
3987                      ;***********************************************
3988                      ;*   This routine inputs an ASCII chr, looks foa a
```

```
3989                    ;* space. If its a space Cy is set true and the
3990                    ;* space char is returned in the Areg.  If not
3991                    ;* a space char, an attempt is made to this char
3992                    ;* and the next char into a single HEX byte. If
3993                    ;* no non-hex chars are received, the resulting
3994                    ;* hex char is returned in the Areg and Cy is false.
3995                    ;* If a non-hex   char is encountered, the Cy is
3996                    ;* is set and a null is returned in the Areg.
3997
3998
3999                    ;echos input chr
4000
```

'MENSCH COMPUTER ROM SOFTWARE'
'UTILITY ROUTINES'

```
4001
4002                    GET|BYTE:    ;INPUT ASCII - LOOK FOR SPACE
4003
4004 00:F09C DA            ?1     PHX          ;SAVE X
4005 00:F09D 22 B2 F1 00          JSL GET_CHR
4006 00:F0A1 C9 20                CMP #' '     ;SPACE?
4007 00:F0A3 F0 08                BEQ ?2
4008 00:F0A5 C9 1B                CMP #ESC
4009 00:F0A7 F0 04                BEQ ?2
4010 00:F0A9 C9 0D                CMP #C_RETURN
4011 00:F0AB D0 03                BNE ?3
4012
4013 00:F0AD FA            ?2       PLX
4014 00:F0AE 38                     SEC
4015 00:F0AF 6B                   RTL
4016
4017 00:F0B0 85 70         ?3     STA TEMP      ;SAVE 1ST CHAR
4018 00:F0B2 22 BC F1 00          JSL PUT_CHR
4019 00:F0B6 22 C6 F1 00          JSL GET_PUT_CHR   ;READ NEXT CHAR
4020
4021 00:F0BA 20 BF F0             JSR ASCBIN    ;CY = 1  IF BAD DATA
4022 00:F0BD FA                   PLX          ;RESTORE X
4023 00:F0BE 6B                   RTL          ;ON RETURN..CY =1 IF BAD DATA
4024
4025
4026
4027                    ;* Routine:  ASCBIN  ASCII TO BINARY
4028                    ;*
4029                    ;* Reg Used: ACC,X, and Y
4030                    ;* Var Used: TMP6
4031                    ;* Routines Called: HEXIN
4032                    ;* Returned Reg: Acc
4033                    ;*
4034 00:F0BF 20 F0 F0      ASCBIN   JSR HEXIN     ;ACC & X REG HAVE DATA
4035 00:F0C2 B0 11                BCS ASCERR    ;1ST CHAR IN TEMP (HIGH ORDER CHAR)
4036 00:F0C4 85 69                STA TMP6      ;USE VAR TMP6
4037 00:F0C6 A5 70                LDA TEMP
4038 00:F0C8 20 F0 F0             JSR HEXIN
4039 00:F0CB B0 08                BCS ASCERR
4040 00:F0CD 0A                   ASL A
4041 00:F0CE 0A                   ASL A
4042 00:F0CF 0A                   ASL A
4043 00:F0D0 0A                   ASL A
4044 00:F0D1 05 69                ORA TMP6
4045 00:F0D3 18                   CLC
```

```
4046  00:F0D4  60                RTS
4047
4048  00:F0D5  38          ASCERR   SEC
4049  00:F0D6  60                RTS
4050
4051
4052
4053
4054                ;* Routine:  BINASC
4055                ;*
4056                ;* Reg Used: Acc & X
4057                ;* Var Used: NONE
```

'MENSCH COMPUTER ROM SOFTWARE'
'UTILITY ROUTINES'

```
4058                    ;* Routines Called: ASCII
4059                    ;* Returned Reg: Acc & TEMP+1        Y REG is PRESERVED
4060                    ;*
4061
4062  00:F0D7  48           BINASC   PHA          ;CONVERT BYTE TO 2
4063  00:F0D8  4A                    LSR A        ;ASCII CHAR
4064
4065  00:F0D9  4A                    LSR A
4066  00:F0DA  4A                    LSR A
4067  00:F0DB  4A                    LSR A
4068  00:F0DC  20 E4 F0              JSR ASCII    ;CONVERT TO ASCII
4069  00:F0DF  85 71                 STA TEMP+1
4070  00:F0E1  68                    PLA
4071  00:F0E2  29 0F                 AND #LOWNIB
4072                                  ;FALL THRU TO ASCII
4073  00:F0E4  18           ASCII    CLC
4074  00:F0E5  69 06                 ADC #6
4075  00:F0E7  69 F0                 ADC #HINIB
4076  00:F0E9  90 02                 BCC ASC1
4077  00:F0EB  69 06                 ADC #$06
4078  00:F0ED  69 3A         ASC1    ADC #'9'+1   ;GT '9'
4079  00:F0EF  60                    RTS
4080
4081
4082                    .page
```

'MENSCH COMPUTER ROM SOFTWARE'
'UTILITY ROUTINES'

```
4083
4084
4085                    ;************************************************************
4086
4087
4088                    ;
4089                    ;   CONVERTS ASCII HEX TO HEX
4090                    ;
4091
4092
4093                    ;* Routine:  HEXIN
4094                    ;*
4095                    ;*  Enter with ASCII char in the Areg.
4096                    ;*  On Exit: Hex digit in Areg
4097                    ;*   IF CARRY SET On RETURN, THEN NOT ASCII HEX
4098                    ;* Reg Used: Acc
4099                    ;* Var Used: NONE
4100                    ;* Routines Called: ISHEX
4101                    ;* Returned Reg: Acc
4102                    ;
4103
4104                       .global HEXIN
4105
4106  00:F0F0  20 03 F1        HEXIN    JSR ISHEX      ;IS IT HEX
4107  00:F0F3  B0 0C                    BCS HEXNG      ;NO
4108  00:F0F5  C9 3A                    CMP #$3A
4109  00:F0F7  08                       PHP            ;SAVE STATUS
4110  00:F0F8  29 0F                    AND #$0F       ;STRIP OFF LO NIBBLE
4111  00:F0FA  28                       PLP            ;GET STAT
4112  00:F0FB  90 02                    BCC HEXXX      ;WAS NUMBER
4113  00:F0FD  69 08                    ADC #$08       ;WAS ALPHA ADD 8+CY=9
4114  00:F0FF  18               HEXXX    CLC
4115  00:F100  60                       RTS
4116  00:F101
4117  00:F101  38               HEXNG    SEC
4118  00:F102  60                       RTS
4119
4120                    ;* Routine:  ISHEX
4121                    ;*   TESTS FOR VALID ASCII HEX DIGIT
4122                    ;*   C=SET IF NOT HEX
4123                    ;* Reg Used: Acc
4124                    ;* Var Used: NONE
4125                    ;* Routines Called: UPPER_CASE
4126                    ;* Returned Reg: Acc
4127
```

```
4128                            .global ISHEX
4129
4130  00:F103  20 1D F1         ISHEX    JSR UPPER_CASE ;IF NOT MAKE UPPER CASE
4131  00:F106  C9 41                     CMP #'A'       ;LESS THAN 'A'
4132  00:F108  90 03                     BCC ISDECIMAL  ;YES,TRY NUMBER CHECK
4133  00:F10A  C9 47                     CMP #'G'       ;F+1
4134                                      ;IF CY SET THEN GREATER THAN F
4135  00:F10C  60                        RTS            ;IF CY CLR THEN OK
4136
4137
4138                          ;* Routine:  ISDECIMAL
4139                          ;*   CHECKs FOR VALID ASCII Digit.
```

'MENSCH COMPUTER ROM SOFTWARE'
'UTILITY ROUTINES'

```
4140                    ;*   If C set upon return..NOT Valid!
4141                    ;* Reg Used: Acc
4142                    ;* Var Used: NONE
4143                    ;* Routines Called: NONE
4144                    ;* Returned Reg: Acc
4145
4146
4147                        .global ISDECIMAL
4148
4149  00:F10D  C9 30        ISDECIMAL CMP #'0'      ;IS LESS THAN '0'
4150  00:F10F  90 03              BCC ISN1      ;YES,NG
4151  00:F111  C9 3A              CMP #'9'+1    ;IE >9
4152                               ;IF CY SET THEN NG
4153  00:F113  60                 RTS          ;IF CY CLR THEN OK
4154
4155  00:F114  38           ISN1    SEC          ;BAD GUYS EXIT
4156  00:F115  60                 RTS
4157
4158
4159                    ;* Routine:  IFASC
4160                    ;*   CHECK FOR VALID ASCII
4161                    ;* Reg Used: Acc
4162                    ;* Var Used: NONE
4163                    ;* Routines Called: ISHEX
4164                    ;* Returned Reg: Acc
4165
4166
4167                       .global IFASC
4168
4169  00:F116  C9 20        IFASC   CMP #' '      ;IS LESS THAN SPACE
4170  00:F118  90 FA              BCC ISN1      ;YES SO NOT ASCII
4171  00:F11A  C9 7F              CMP #$7F      ;GT TILDA
4172                               ;IF CY SET THEN SO NOT ASCII
4173  00:F11C  60                 RTS          ;IF CY CLR THEN OK
4174
4175
4176
4177                    ;* Routine:  UPPER_CASE
4178                    ;* Reg Used: Acc
4179                    ;* Var Used: NONE
4180                    ;* Routines Called: NONE
4181                    ;* Returned Reg: Acc
4182
4183
4184                        .global UPPER_CASE
```

```
4185
4186  00:F11D  C9 61          UPPER_CASE CMP #'a'     ;CONVERT TO UPPER CASE
4187  00:F11F  90 07                BCC NIBBIN1    ;NOT an upper case char
4188  00:F121  C9 7B                CMP #'z'+1     ;IS IT GT A 'z'
4189  00:F123  B0 03                BCS NIBBIN1    ;NOT an upper case char
4190  00:F125  38                   SEC
4191  00:F126  E9 20                SBC #$20       ;MAKE IT UPPER CASE
4192  00:F128  60             NIBBIN1  RTS
4193
4194
4195
4196                    ;* Routine:  BIN2DEC
```

'MENSCH COMPUTER ROM SOFTWARE'
'UTILITY ROUTINES'

```
4197                    ;*    ENTER with binary number in an 8 bit Areg.
4198                    ;*
4199                    ;* Reg Used: Acc
4200                    ;* Var Used: NONE
4201                    ;* Routines Called: NONE
4202                    ;* Returned Reg: Acc
4203
4204
4205                        .global BIN2DEC
4206
4207 00:F129 DA             BIN2DEC  PHX          ;convert Acc to packed decimal (MAX 99)
4208 00:F12A 5A                     PHY
4209 00:F12B 08                     PHP
4210 00:F12C 48                     PHA     ;input byte
4211 00:F12D E2 10                  SEP #X8
4212                       .LONGI OFF
4213 00:F12F 4A                     LSR A
4214 00:F130 4A                     LSR A
4215 00:F131 4A                     LSR A
4216 00:F132 4A                     LSR A
4217 00:F133 A8                     TAY    ;Hi nibble is in Yreg
4218 00:F134 68                     PLA
4219 00:F135 29 0F                  AND #LOWNIB
4220 00:F137 AA                     TAX    ;Low nibble is in Xreg
4221 00:F138 F8                     SED    ;DECIMAL ADD NEEDED!
4222 00:F139 B9 6B FF               LDA BINDECH,Y
4223 00:F13C 18                     CLC
4224 00:F13D 7F 5B FF 00            ADC BINDECL,X
4225 00:F141 D8                     CLD
4226                       .LONGI ON
4227 00:F142 28                     PLP
4228 00:F143 7A                     PLY
4229 00:F144 FA                     PLX
4230 00:F145 60                     RTS
4231
4232
4233
4234
4235
4236
4237                    ;* Routine:  DADD
4238                    ;*          CALCULATE CHECKSUM
4239                    ;* Reg Used: NONE
4240                    ;* Var Used: TMP4
4241                    ;* Routines Called: NONE
```

```
                        ;* Returned Reg: NONE
                        ;*
                                    ;CALCULATE CHECKSUM
00:F146  48                 DADD    PHA         ;SAVE A
00:F147  18                         CLC
00:F148  65 66                      ADC TMP4
00:F14A  85 66                      STA TMP4
00:F14C  A5 67                      LDA TMP4+1
00:F14E  69 00                      ADC #0
00:F150  85 67                      STA TMP4+1
00:F152  68                         PLA         ;RESTORE A
00:F153  60                         RTS
```

'MENSCH COMPUTER ROM SOFTWARE'
'UTILITY ROUTINES'

```
4254
4255
4256
4257                    ;**********************
4258                    ;MODIFIED FOR 816 CODE
4259                    ;**********************
4260
4261                    ;* Routine:  INCTMP0
4262                    ;*
4263                    ;* Reg Used: NONE
4264                    ;* Var Used: TMP0
4265                    ;* Routines Called: NONE
4266                    ;* Returned Reg: NONE
4267                    ;*
4268
4269  00:F154  48               INCTMP0  PHA          ;Increment TMP0 by 1
4270  00:F155  E6 5D                 INC TMP0      ;LO BYTE
4271  00:F157  D0 0A                 BNE INCT0
4272  00:F159  E6 5E                 INC TMP0+1
4273  00:F15B  D0 06                 BNE INCT0
4274  00:F15D  E6 5F                 INC TMP0+2    ;BANK
4275  00:F15F  D0 02                 BNE INCT0
4276  00:F161  E6 59                 INC WRAP
4277  00:F163  68               INCT0    PLA
4278  00:F164  60                    RTS
4279
4280
4281
4282
4283
4284                    ;* Routine:  INCTMP1 Increment TMP1 by 1
4285                    ;*
4286                    ;* Reg Used: NONE
4287                    ;* Var Used: TMP1
4288                    ;* Routines Called: NONE
4289                    ;* Returned Reg: NONE
4290                    ;*
4291
4292  00:F165  48               INCTMP1  PHA          ;INC STRING POINTER
4293  00:F166  E6 60                 INC TMP1      ;LO BYTE
4294  00:F168  D0 0A                 BNE INCT1
4295  00:F16A  E6 61                 INC TMP1+1
4296  00:F16C  D0 06                 BNE INCT1
4297  00:F16E  E6 62                 INC TMP1+2    ;BANK
4298  00:F170  D0 02                 BNE INCT1
```

```
4299  00:F172  E6 59                INC WRAP
4300  00:F174  68             INCT1    PLA
4301  00:F175  60                   RTS
4302
4303                  ;* Routine:  DECTMP0 Decrement TMP1 by 1
4304                  ;*
4305                  ;* Reg Used: NONE
4306                  ;* Var Used: TMP0
4307                  ;* Routines Called: NONE
4308                  ;* Returned Reg: NONE
4309                  ;*
4310
```

'MENSCH COMPUTER ROM SOFTWARE'
'UTILITY ROUTINES'

```
4311
4312  00:F176  48              DECTMP0  PHA
4313  00:F177  A5 5D                LDA TMP0
4314  00:F179  38                   SEC
4315  00:F17A  E9 01                SBC #1
4316  00:F17C  85 5D                STA TMP0
4317  00:F17E  A5 5E                LDA TMP0+1
4318  00:F180  E9 00                SBC #0
4319  00:F182  85 5E                STA TMP0+1
4320  00:F184  A5 5F                LDA TMP0+2
4321  00:F186  E9 00                SBC #0
4322  00:F188  85 5F                STA TMP0+2
4323  00:F18A  68                   PLA
4324  00:F18B  60                   RTS
4325
4326
4327
4328                  ;* Routine:  DECTMP1 Decrement TMP1 by 1
4329                  ;*
4330                  ;* Reg Used: NONE
4331                  ;* Var Used: TMP1
4332                  ;* Routines Called: NONE
4333                  ;* Returned Reg: NONE
4334                  ;*
4335
4336
4337  00:F18C  48              DECTMP1  PHA
4338  00:F18D  A5 60                LDA TMP1
4339  00:F18F  38                   SEC
4340  00:F190  E9 01                SBC #1
4341  00:F192  85 60                STA TMP1
4342  00:F194  A5 61                LDA TMP1+1
4343  00:F196  E9 00                SBC #0
4344  00:F198  85 61                STA TMP1+1
4345  00:F19A  A5 62                LDA TMP1+2
4346  00:F19C  E9 00                SBC #0
4347  00:F19E  85 62                STA TMP1+2
4348  00:F1A0  68                   PLA
4349  00:F1A1  60                   RTS
4350
4351
4352
4353
4354                        .END
4355
```

```
4356
4357                    .STTL 'ROM_I/O ROUTINES'
4358                    .PAGE
```

```
    'MENSCH COMPUTER ROM SOFTWARE'
    'ROM_I/O ROUTINES'

4359 00:F1A1                    include rom_io.asm
4360
4361                    ; FILE: ROM_IO.ASM
4362                    ; DATE: 12-17-94
4363
4364
4365
4366                    SET_GET|PUT_CHR:      ;ROUTINE FOR ROM SETUP
4367 00:F1A2
4368 00:F1A2  A2 B5 F1          LDX #ROM_GET_CHR
4369 00:F1A5  86 73             STX GET_CHR_JMP
4370 00:F1A7  A2 C9 F1          LDX #GET|PUT_PC_CHR
4371 00:F1AA  86 77             STX GET|PUT_CHR_JMP
4372 00:F1AC  A2 BF F1          LDX #ROM_PUT_CHR
4373 00:F1AF  86 75             STX PUT_CHR_JMP
4374 00:F1B1  60                RTS
4375
4376
4377                    GET_CHR:      ;ROM ROUTINE
4378
4379 00:F1B2  6C 73 00          JMP (GET_CHR_JMP)
4380
4381 00:F1B5                ROM_GET_CHR:
4382
4383 00:F1B5  22 F9 FB 00   ?1   JSL GET_BYTE_FROM_PC
4384 00:F1B9  B0 FA             BCS ?1
4385 00:F1BB  6B                RTL
4386
4387                    ;*****************************
4388
4389 00:F1BC                PUT_CHR:
4390
4391 00:F1BC  6C 75 00          JMP (PUT_CHR_JMP)
4392
4393
4394 00:F1BF                ROM_PUT_CHR:
4395
4396 00:F1BF  22 BC FD 00   ?1   JSL SEND_BYTE_TO_PC  ;WAIT FOR BUFFER READY
4397 00:F1C3  B0 FA             BCS ?1
4398 00:F1C5  6B                RTL
4399
4400
4401                    ;*****************************
4402
4403 00:F1C6                GET_PUT_CHR:
```

```
4404  00:F1C6
4405  00:F1C6  6C 77 00            JMP (GET|PUT_CHR_JMP)
4406
4407
4408  00:F1C9                   GET|PUT_PC_CHR:
4409
4410  00:F1C9  22 F9 FB 00   ?1    JSL GET_BYTE_FROM_PC
4411  00:F1CD  B0 FA               BCS ?1
4412  00:F1CF  22 BC FD 00   ?2    JSL SEND_BYTE_TO_PC
4413  00:F1D3  B0 FA               BCS ?2
4414  00:F1D5  6B                  RTL
4415
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM_I/O ROUTINES'

```
4416                    ;*****************************
4417
4418
4419  00:F1D6           SEND_CR:
4420
4421  00:F1D6  A9 0D          LDA #C_RETURN
4422  00:F1D8  4C BC F1          JMP PUT_CHR
4423
4424  00:F1DB           SEND_SPACE2:
4425  00:F1DB  22 DF F1 00    JSL SEND_SPACE
4426  00:F1DF           SEND_SPACE:
4427
4428  00:F1DF  A9 20          LDA #' '
4429  00:F1E1  4C BC F1       JMP PUT_CHR
4430
4431
4432
4433  00:F1E4           BACKSPACE2:
4434  00:F1E4  A9 08          LDA #BKSP       ;BACKUP 1 char position
4435  00:F1E6  22 BC F1 00    JSL PUT_CHR
4436  00:F1EA           BACKSPACE:
4437  00:F1EA  A9 08          LDA #BKSP       ;BACKUP 2nd char position
4438  00:F1EC  22 BC F1 00    JSL PUT_CHR
4439  00:F1F0  6B             RTL
4440
4441
4442
4443
4444  00:F1F1           CLEAR_LCD_DISPLAY:
4445  00:F1F1  22 D6 F1 00    JSL SEND_CR
4446  00:F1F5  A9 03          LDA #3
4447  00:F1F7  6C 79 00          JMP (CLR_LCD_JMP)
4448
4449
4450  00:F1FA           DISP_LCD_STRNG:
4451  00:F1FA  6C 7B 00       JMP (DISP_LCD_JMP)
4452
4453
4454  00:F1FD           POSITION_TEXT_CURSOR:
4455  00:F1FD  6C 7D 00       JMP (TXT_CUR_JMP)
4456
4457
4458  00:F200           SEND_BEEP:
4459  00:F200  6C 7F 00       JMP (SND_BEEP_JMP)
4460
```

```
4461
4462
4463
4464  00:F203                   RTL_EXIT:
4465  00:F203  6B                   RTL
4466
4467                      ;***************************************************
4468                      .page
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM_I/O ROUTINES'

```
4469
4470                    ;*
4471                    ;*   This routine asks for a 3 byte address,
4472                    ;*     checks it and leaves it in TMP2.
4473                    ;*
4474                    ;*   If an error is encounterd, a Cy=1 is returned.
4475                    ;*
4476                    ;*
4477
4478
4479
4480
4481  00:F204                   Get_Address:
4482
4483  00:F204  A9 03                LDA #3
4484  00:F206  22 F1 F1 00          JSL CLEAR_LCD_DISPLAY
4485
4486                    ;    SETUP THE STARTING ADDRESS    in TMP0
4487  00:F20A  A2 00 00             LDX #0
4488  00:F20D  A9 04                LDA #04
4489  00:F20F  22 FD F1 00          JSL POSITION_TEXT_CURSOR
4490
4491  00:F213  A9 00                LDA #0
4492  00:F215  A2 1A F2             LDX #Enter_ADDR
4493  00:F218  80 69                    BRA GET_A_OUT
4494
4495
4496  00:F21A  45 6E 74 65 72          Enter_ADDR .dc 'Enter Address  BB:AAAA '
          20 41 64 64 72
          65 73 73 20 20
          42 42 3A 41 41
          41 41 A0
4497  00:F231  45 6E 74 65 72 Enter_SA  .dc 'Enter Lowest Address   BB:AAAA '
          20 4C 6F 77 65
          73 74 20 41 64
          64 72 65 73 73
          20 20 20 42 42
          3A 41 41 41 41
          A0
4498  00:F250  45 6E 74 65 72 Enter_EA  .dc 'Enter Highest Address  BB:AAAA '
          20 48 69 67 68
          65 73 74 20 41
          64 64 72 65 73
          73 20 20 42 42
          3A 41 41 41 41
```

```
        A0
4499
4500
4501                    ;***********************************************
4502                    .page
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM_I/O ROUTINES'

```
4503
4504                    ;*
4505                    ;*   This routine asks for a 3 byte starting address,
4506                    ;*    checks it and leaves it in TMP2.
4507                    ;*
4508                    ;*   If an error is encounterd, a Cy=1 is returned.
4509                    ;*
4510                    ;*
4511
4512
4513
4514
4515  00:F26F                   Get_S_Address:
4516
4517  00:F26F  A9 03                LDA #3
4518  00:F271  22 F1 F1 00          JSL CLEAR_LCD_DISPLAY
4519
4520                    ;    SETUP THE STARTING ADDRESS    in TMP0
4521  00:F275  A2 00 00             LDX #0
4522  00:F278  A9 04                LDA #04
4523  00:F27A  22 FD F1 00          JSL POSITION_TEXT_CURSOR
4524
4525  00:F27E  A9 00                LDA #0
4526  00:F280  A2 31 F2             LDX #Enter_SA
4527  00:F283                   GET_A_OUT:
4528  00:F283  22 A1 F3 00          JSL PUT_STR
4529  00:F287  22 A0 F2 00          JSL GET_3BYTE_ADDR
4530  00:F28B  6B                       RTL
4531
4532
4533                    ;*************************************************
4534
4535                    ;*
4536                    ;*   This routine asks for a 3 byte ending address,
4537                    ;*    checks it and leaves it in TMP2.
4538                    ;*
4539                    ;*   If an error is encounterd, a Cy=1 is returned.
4540                    ;*
4541                    ;*
4542
4543
4544
4545
4546  00:F28C                   Get_E_Address:
4547
```

```
4548
4549                        ;    SETUP THE ENDING ADDRESS    in TMP0
4550  00:F28C  22 D6 F1 00          JSL SEND_CR
4551  00:F290  A2 00 00             LDX #0
4552  00:F293  A9 05                LDA #05
4553  00:F295  22 FD F1 00          JSL POSITION_TEXT_CURSOR
4554
4555  00:F299  A9 00                LDA #0
4556  00:F29B  A2 50 F2             LDX #Enter_EA
4557  00:F29E  80 E3                    BRA GET_A_OUT
4558
4559
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM_I/O ROUTINES'

```
4560
4561                    ;*******************************************************
4562                    ;*
4563                    ;*  GET_3BYTE_ADDR    Asks for input bytes to form a
4564                    ;*                  24 bit address. Characters are
4565                    ;*                    received from any input device
4566                    ;*                    selected by the CONTROL INPUT
4567                    ;*                    routine.
4568                    ;*
4569                    ;*   Must enter with a JSL command!  Areg = 8 bits.
4570                    ;*
4571                    ;*  This Routine inserts a ':' after the bank address.
4572                    ;*
4573                    ;*   INPUT FORMAT =  BB:AAAA
4574                    ;*
4575                    ;*   3 Byte result returned in TMP2, +1, +2 (address order)
4576                    ;*
4577                    ;*   A Cy = 1 will be returned for the following reasons:
4578                    ;*
4579                    ;*          1) No input device selected.
4580                    ;*          2) An ESC or ENTER received before 6 chars.
4581                    ;*          3) One of the 6 chars inputted is non-hex.
4582                    ;*
4583                    ;*
4584
4585
4586        00:F2A0             GET_3BYTE_ADDR  EQU *
4587
4588  00:F2A0  5A                   PHY
4589  00:F2A1  DA                   PHX
4590  00:F2A2  A0 00 00             LDY #0
4591  00:F2A5  22 C6 F1 00    ?2        JSL GET_PUT_CHR
4592  00:F2A9  B0 37                BCS ?5
4593  00:F2AB  C9 1B                   CMP #ESC
4594  00:F2AD  F0 32                   BEQ ?4
4595  00:F2AF  C9 0D                CMP #C_RETURN
4596  00:F2B1  F0 2E                BEQ ?4
4597  00:F2B3  C9 08                CMP #BKSP
4598  00:F2B5  F0 34                BEQ ?88
4599  00:F2B7  20 03 F1             JSR ISHEX      ;IS IT HEX
4600  00:F2BA  B0 29                   BCS ?8
4601  00:F2BC  99 8E 01             STA >STR_BUF,Y
4602  00:F2BF  C8                   INY
4603  00:F2C0  C0 07 00             CPY #7
4604  00:F2C3  F0 11                BEQ ?3       ;DONE
```

```
4605  00:F2C5  C0 02 00            CPY #2
4606  00:F2C8  D0 DB               BNE ?2
4607  00:F2CA  A9 3A               LDA #':'     ;ADD COLON AFTER BANK ADR
4608  00:F2CC  99 8E 01            STA >STR_BUF,Y
4609  00:F2CF  22 BC F1 00         JSL PUT_CHR
4610  00:F2D3  C8                  INY
4611  00:F2D4  80 CF               BRA ?2
4612
4613  00:F2D6  A9 00            ?3   LDA #0
4614  00:F2D8  A2 8E 01            LDX #STR_BUF
4615  00:F2DB  22 FF F2 00         JSL ADDR_IN    ;IF Cy IS SET OK FOR ERROR
4616  00:F2DF  80 01               BRA ?5
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM_I/O ROUTINES'

```
4617
4618  00:F2E1  38              ?4     SEC
4619  00:F2E2  FA              ?5     PLX
4620  00:F2E3  7A                     PLY
4621  00:F2E4  6B                     RTL
4622
4623
4624  00:F2E5  A9 08           ?8     LDA #BKSP
4625  00:F2E7  22 BC F1 00            JSL PUT_CHR
4626
4627
4628  00:F2EB  C0 00 00        ?88    CPY #0
4629  00:F2EE  F0 B5                  BEQ ?2
4630  00:F2F0  88                     DEY
4631  00:F2F1  C0 02 00               CPY #2
4632  00:F2F4  D0 AF                  BNE ?2
4633  00:F2F6  A9 08                  LDA #BKSP
4634  00:F2F8  22 BC F1 00            JSL PUT_CHR
4635  00:F2FC  88                     DEY
4636  00:F2FD  80 A6                  BRA ?2
4637
4638
4639
4640                    .page
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM_I/O ROUTINES'

```
4641                    ;*******************************************************
4642
4643                    ;*   ADDR_IN    CONVERTS AN ASCII STRING TO AN ADDR
4644                    ;*
4645                    ;*   Must enter with a JSL command!  Areg = 8 bits.
4646                    ;*
4647                    ;*   INPUT FORMAT =  BB:AAAA
4648                    ;*
4649                    ;*   Address of the string:  Areg = Bank  Xreg = 16 Bit addr
4650                    ;*
4651                    ;*   3 Byte result returned in TMP2, +1, +2
4652                    ;*
4653
4654  00:F2FF  5A               ADDR_IN       PHY
4655  00:F300  0B                     PHD          ;SAVE DIRECT REG
4656  00:F301  8B                     PHB
4657  00:F302  F4 00 00               PEA #$0000
4658  00:F305  2B                     PLD          ;SET DIRECT PAGE REG TO PAGE 0
4659  00:F306  F4 00 00               PEA #$0000    ;SET DATA BANK TO 0
4660  00:F309  AB                     PLB
4661  00:F30A  AB                     PLB
4662
4663  00:F30B  86 54                  STX TMPRY_PTR
4664  00:F30D  A0 00 00               LDY #0
4665  00:F310  20 36 F3                  JSR HEX2IN
4666  00:F313  B0 1C                     BCS ?xx
4667  00:F315  85 65                  STA TMP2+2    ;BANK FINISHED
4668  00:F317  C8                     INY
4669  00:F318  B1 54                  LDA (TMPRY_PTR),Y
4670  00:F31A  C9 3A                  CMP #':'
4671  00:F31C  D0 13                  BNE ?xx
4672  00:F31E  C8                     INY
4673  00:F31F  20 36 F3                  JSR HEX2IN
4674  00:F322  B0 0D                     BCS ?xx
4675  00:F324  85 64                  STA TMP2+1
4676  00:F326  C8                     INY
4677  00:F327  20 36 F3               JSR HEX2IN
4678  00:F32A  B0 05                  BCS ?xx
4679  00:F32C  85 63                  STA TMP2      ;ADDR LOW FINISHED
4680  00:F32E  18                        CLC
4681  00:F32F  80 01                     BRA ?XXE
4682
4683  00:F331  38               ?xx     SEC
4684  00:F332  AB               ?XXE    PLB
4685  00:F333  2B                     PLD
```

```
4686  00:F334  7A                PLY
4687  00:F335  6B                RTL
4688
4689
4690  00:F336  B1 54        HEX2IN   LDA (TMPRY_PTR),Y  ;TWO ASCII TO 1 HEX
4691  00:F338  20 F0 F0           JSR HEXIN
4692  00:F33B  B0 12              BCS ?xxx
4693  00:F33D  0A                 ASL A
4694  00:F33E  0A                 ASL A
4695  00:F33F  0A                 ASL A
4696  00:F340  0A                 ASL A
4697  00:F341  85 70              STA TEMP
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM_I/O ROUTINES'

```
4698  00:F343  C8                INY
4699  00:F344  B1 54             LDA (TMPRY_PTR),Y
4700  00:F346  20 F0 F0          JSR HEXIN
4701  00:F349  B0 04             BCS ?xxx
4702  00:F34B  05 70             ORA TEMP
4703  00:F34D  18                   CLC
4704  00:F34E  60                   RTS
4705
4706  00:F34F  38          ?xxx  SEC
4707  00:F350  60                RTS
4708
4709                  ;****************************************
4710                  .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM_I/O ROUTINES'

```
4711
4712                    ;*  This routine is used to write a 3 byte
4713                    ;*   address to the selected outputs in
4714                    ;*   ASCII-Hex.  The address to be sent
4715                    ;*   out must be loaded into TMP0.
4716                    ;*
4717
4718                    WR_3_ADDRESS:  ;WRITE OUT A 3 BYTE ADDRESS
4719 00:F351 AF 5F 00 00          LDA >TMP0+2
4720 00:F355 22 70 F3 00          JSL SEND_HEX_OUT
4721 00:F359 A9 3A                LDA #':'
4722 00:F35B 22 BC F1 00          JSL PUT_CHR
4723 00:F35F AF 5E 00 00          LDA >TMP0+1
4724 00:F363 22 70 F3 00          JSL SEND_HEX_OUT
4725 00:F367 AF 5D 00 00          LDA >TMP0
4726 00:F36B 22 70 F3 00          JSL SEND_HEX_OUT
4727 00:F36F 6B                   RTL
4728
4729
4730                    ;****************************************
4731                    ;*
4732                    ;*   SEND_HEX_OUT
4733                    ;*
4734                    ;*   This routine takes a HEX value from an
4735                    ;*  eight bit Areg, converts it to two ASCII
4736                    ;*  characters and outputs them to the selected
4737                    ;*  outputs.
4738                    ;*
4739                    ;*   Must call with a JSL command
4740                    ;*
4741                    ;*  Routine calls BINASC, PUT_CHR
4742
4743
4744
4745 00:F370                  SEND_HEX_OUT:
4746
4747 00:F370 DA                    PHX         ;WRITE BYTE AS 2 HEX CHAR
4748 00:F371 20 D7 F0             JSR BINASC    ;UNPACK BYTE DATA INTO
4749                                ;TWO ASCII CHARS.
4750                                ;LOW in Areg, HI in TEMP+1
4751
4752 00:F374 48                    PHA         ;WRITE 2 CHARS
4753 00:F375 AF 71 00 00          LDA >TEMP+1    ;WRITE HI BYTE FIRST
4754 00:F379 22 BC F1 00          JSL PUT_CHR
4755 00:F37D 68                    PLA         ;LOW BYTE
```

```
4756  00:F37E  22 BC F1 00          JSL PUT_CHR
4757  00:F382  FA                   PLX
4758  00:F383  6B                   RTL
4759
4760
4761                    ;*************************************************
4762
4763  00:F384  20 46 F1         CKNOUT   JSR DADD      ;CALCULATE CHECKSUM
4764  00:F387  22 70 F3 00               JSL SEND_HEX_OUT
4765  00:F38B  60                        RTS
4766
4767  00:F38C  DA               WRT2OUT  PHX          ;WRITE BYTE AS 2 HEX CHAR
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM_I/O ROUTINES'

```
4768  00:F38D  20 D7 F0          JSR BINASC    ;UNPACK BYTE DATA INTO
4769
4770  00:F390  48                PHA           ;WRITE 2 CHARS-X,A=CHARS
4771  00:F391  AF 71 00 00       LDA >TEMP+1   ;WRITE HI BYTE FIRST
4772  00:F395  22 BC F1 00       JSL PUT_CHR
4773  00:F399  68                PLA
4774  00:F39A  22 BC F1 00       JSL PUT_CHR
4775  00:F39E  FA                PLX
4776  00:F39F  18                CLC
4777  00:F3A0  60                RTS
4778
4779
4780
4781
4782
4783                    ;*****************************************
4784                    .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM_I/O ROUTINES'

```
4785
4786
4787                    ;*  PUT_STR  =  SUBROUTINE TO OUTPUT AN ASCII STRING.
4788                    ;*
4789                    ;*  This routine must called with a JSL command!
4790                    ;*
4791                    ;*   This routine OUTPUTS a string to each of the output
4792                    ;*     PORTS selected by the CONTROL_OUTPUT routine.
4793                    ;*
4794                    ;*   C flag = 1 is returned if no output ports have been
4795                    ;*     enabled via the CONTROL_OUTPUT routine!
4796                    ;*
4797                    ;*  Enter with a 8 bit Areg containing bank address and
4798                    ;*    a 16 bit Xreg containing 16 bit address pointing to
4799                    ;*    the string buffer to be used. The maximum string
4800                    ;*    size is limited to 640 characters.
4801                    ;*
4802                    ;*   The string must be terminated with a NULL or have
4803                    ;*    Bit 7 of the last character equal to a 1.
4804                    ;*
4805                    ;*
4806                    ;*   All registers are saved!
4807                    ;*
4808
4809                    PUT_STR:    ;OUTPUT A STRING
4810
4811  00:F3A1  48                       PHA
4812  00:F3A2  DA                       PHX
4813  00:F3A3  5A                     PHY
4814  00:F3A4  0B                     PHD          ;SAVE DIRECT REG
4815  00:F3A5  8B                     PHB
4816  00:F3A6  F4 00 00               PEA #$0000
4817  00:F3A9  2B                     PLD          ;SET DIRECT PAGE REG TO PAGE 0
4818  00:F3AA  F4 00 00               PEA #$0000     ;SET DATA BANK TO 0
4819  00:F3AD  AB                     PLB
4820  00:F3AE  AB                     PLB
4821
4822  00:F3AF  85 68                  STA TMP4+2
4823  00:F3B1  86 66                  STX TMP4
4824  00:F3B3  A0 00 00                  LDY #0
4825  00:F3B6  B7 66            1?        LDA [TMP4],Y
4826  00:F3B8  F0 12                  BEQ 2?
4827  00:F3BA  08                     PHP
4828  00:F3BB  29 7F                  AND #$7F
4829  00:F3BD  22 BC F1 00    3?       JSL PUT_CHR
```

```
4830  00:F3C1  B0 FA          BCS 3?
4831  00:F3C3  28                PLP
4832  00:F3C4  30 06          BMI 2?
4833  00:F3C6  C8             INY
4834  00:F3C7  C0 80 02       CPY #640
4835  00:F3CA  D0 EA          BNE 1?
4836  00:F3CC  18          2?  CLC
4837  00:F3CD  80 02             BRA 5?
4838
4839  00:F3CF  28          4?  PLP
4840  00:F3D0  38             SEC
4841  00:F3D1  AB          5?    PLB
```

```
    'MENSCH COMPUTER ROM SOFTWARE'
    'ROM_I/O ROUTINES'

4842  00:F3D2  2B                PLD
4843  00:F3D3  7A                   PLY
4844  00:F3D4  FA                PLX
4845  00:F3D5  68                   PLA
4846  00:F3D6  6B                   RTL
4847
4848                  ********************************************
4849
4850                  .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM_I/O ROUTINES'

```
4851
4852                    ;*  GET_STR
4853                    ;*
4854                    ;*  GET_STR  USES GET_PUT_CHR TO RECEIVE CHARACTERS AND
4855                    ;*    PUTS THEM INTO STR_BUF. THE STRING IS TERMINATED
4856                    ;*    WHEN AN ENTER OR AN ESC IS RECEIVED.  THE COMPLETED
4857                    ;*    STRING IS TERMINATED WITH A NULL CHR.
4858                    ;*
4859                    ;*  Enter with a 8 bit Areg containing bank address and
4860                    ;*    a 16 bit Xreg containing 16 bit address pointing to
4861                    ;*    the string buffer to be used.
4862                    ;*
4863                    ;*  IF TERMINATED WITH AN ESC, Cy WILL BE SET UPON RETURN.
4864                    ;*
4865                    ;*  A NULL STRING is returned if no input sources have
4866                    ;*    been enabled using the CONTROL_INPUT routine!
4867                    ;*
4868                    ;*
4869                    ;* WARNING: MUST use a 'JSL' to call this routine!
4870                    ;*
4871
4872
4873  00:F3D7                   GET_STR:
4874  00:F3D7  48                       PHA
4875  00:F3D8  DA                       PHX
4876  00:F3D9  5A                       PHY
4877  00:F3DA  0B                       PHD         ;SAVE DIRECT REG
4878  00:F3DB  8B                       PHB
4879  00:F3DC  F4 00 00                 PEA #$0000
4880  00:F3DF  2B                       PLD         ;SET DIRECT PAGE REG TO PAGE 0
4881  00:F3E0  F4 00 00                 PEA #$0000    ;SET DATA BANK TO 0
4882  00:F3E3  AB                       PLB
4883  00:F3E4  AB                       PLB
4884
4885  00:F3E5  86 54                    STX TMPRY_PTR
4886  00:F3E7  85 56                    STA TMPRY_PTR+2
4887  00:F3E9  A0 00 00                 LDY #0
4888  00:F3EC  8C 89 01         1?      STY STR_BUF_PTR
4889  00:F3EF  22 C6 F1 00      2?      JSL GET_PUT_CHR
4890  00:F3F3  B0 20                    BCS 4?
4891  00:F3F5  C9 08                    CMP #BKSP
4892  00:F3F7  D0 0B                    BNE 3?
4893  00:F3F9  AC 89 01                 LDY STR_BUF_PTR
4894  00:F3FC  F0 F1                    BEQ 2?
4895  00:F3FE  88                       DEY
```

```
4896  00:F3FF  8C 89 01          STY STR_BUF_PTR
4897  00:F402  80 EB             BRA 2?
4898  00:F404
4899  00:F404  C9 1B        3?    CMP #ESC
4900  00:F406  F0 14             BEQ 5?
4901  00:F408  AC 89 01          LDY STR_BUF_PTR
4902  00:F40B  97 54                STA [TMPRY_PTR],Y
4903  00:F40D  C8                INY
4904  00:F40E  8C 89 01          STY STR_BUF_PTR
4905  00:F411  C9 0D             CMP #C_RETURN
4906  00:F413  D0 D7             BNE 1?
4907  00:F415  A9 00        4?    LDA #0        ;ADD END OF STRNG
```

```
    'MENSCH COMPUTER ROM SOFTWARE'
    'ROM_I/O ROUTINES'

4908  00:F417  97 54                STA [TMPRY_PTR],Y
4909  00:F419  18                 CLC
4910  00:F41A  80 01                BRA 6?
4911
4912  00:F41C  38           5?      SEC
4913  00:F41D  AB           6?    PLB
4914  00:F41E  2B                 PLD
4915  00:F41F  7A                 PLY
4916  00:F420  FA                 PLX
4917  00:F421  68                 PLA
4918  00:F422  6B                 RTL
4919
4920                ;*****************************************
4921
4922
4923                     .END
4924
4925
4926                     .STTL 'TONE GEN ROUTINES'
4927                     .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'TONE GEN ROUTINES'

```
4928  00:F422                       include R_TONES.asm
4929                        ;
4930                        ; FILE: R_TONES.ASM
4931                        ; DATE: 12-17-94
4932
4933  00:F423
4934
4935
4936
4937                        CONTROL_TONES:   ;TURN TONES ON & OFF
4938
4939
4940                        ;* WARNING: MUST use a 'JSL' to call this routine!
4941                        ;
4942                        ;  Enter with 8 Bit Areg containing the control info.
4943                        ;    If Areg = 0  the tone generators will be turned off.
4944                        ;    If Areg = 1  tone generator TG0 is turned on
4945                        ;    If Areg = 2  tone generator TG1 is turned on.
4946                        ;    If Areg = 3  both TG0 & TG1 are turned on.
4947                        ;
4948                        ;  Enter with 16 bit Xreg containing the TIMER 5 VALUE
4949                        ;  Enter with 16 bit Yreg containing the TIMER 6 VALUE
4950                        ;
4951                        ;  IF Xreg or Yreg = 0, then use previously set timer value.
4952                        ;
4953  00:F423  48                      PHA
4954  00:F424  0B                      PHD          ;SAVE DIRECT REG
4955  00:F425  8B                      PHB
4956  00:F426  F4 00 00                PEA #$0000
4957  00:F429  2B                      PLD          ;SET DIRECT PAGE REG TO PAGE 0
4958  00:F42A  F4 00 00                PEA #$0000     ;SET DATA BANK TO 0
4959  00:F42D  AB                      PLB
4960  00:F42E  AB                      PLB
4961
4962  00:F42F  09 00                   ORA #0
4963  00:F431  F0 0F                   BEQ ?0
4964  00:F433  C9 01                   CMP #1
4965  00:F435  F0 17                   BEQ ?1
4966  00:F437  C9 02                   CMP #2
4967  00:F439  F0 27                   BEQ ?2
4968  00:F43B  C9 03                   CMP #3
4969  00:F43D  F0 37                   BEQ ?3
4970  00:F43F  68                      PLA              ;NOT PROPER CALL PARAMETER
4971  00:F440  38                      SEC
4972  00:F441  6B                      RTL
```

```
4973
4974
4975  00:F442  A9 06          ?0     LDA #Bit1+Bit2      ;DISABLE TG0 & TG1
4976  00:F444  1C 40 DF              TRB BCR
4977  00:F447  A9 60                 LDA #T5FLG+T6FLG    ;DISABLE TIMERS #5 & 6 ON 265
4978  00:F449  1C 43 DF              TRB !TER
4979  00:F44C  80 42                 BRA ?4
4980  00:F44E
4981  00:F44E  E0 00 00       ?1     CPX #0
4982  00:F451  F0 03                 BEQ ?1.5
4983  00:F453  8E 6A DF              STX !T5CL
4984  00:F456  A9 20          ?1.5   LDA #T5FLG         ;ENABLE TIMER #5
```

'MENSCH COMPUTER ROM SOFTWARE'
'TONE GEN ROUTINES'

```
4985  00:F458  0C 43 DF            TSB !TER
4986  00:F45B  A9 02               LDA #Bit1         ;ENABLE TG0
4987  00:F45D  0C 40 DF            TSB BCR
4988  00:F460  80 2E               BRA ?4
4989
4990  00:F462  C0 00 00       ?2    CPY #0
4991  00:F465  F0 03               BEQ ?2.5
4992  00:F467  8C 6C DF            STY !T6CL
4993  00:F46A  A9 40          ?2.5   LDA #T6FLG         ;ENABLE TIMER #6
4994  00:F46C  0C 43 DF            TSB !TER
4995  00:F46F  A9 04               LDA #Bit2         ;ENABLE TG1
4996  00:F471  0C 40 DF            TSB BCR
4997  00:F474  80 1A               BRA ?4
4998
4999  00:F476  E0 00 00       ?3    CPX #0
5000  00:F479  F0 03               BEQ ?3.3
5001  00:F47B  8E 6A DF            STX !T5CL
5002  00:F47E  C0 00 00       ?3.3   CPY #0
5003  00:F481  F0 03               BEQ ?3.6
5004  00:F483  8C 6C DF            STY !T6CL
5005  00:F486  A9 60          ?3.6   LDA #T5FLG+T6FLG    ;ENABLE TIMERS #5 & 6 ON 265
5006  00:F488  0C 43 DF            TSB !TER
5007  00:F48B  A9 06               LDA #Bit1+Bit2     ;ENABLE TG0 & TG1
5008  00:F48D  0C 40 DF            TSB BCR
5009
5010  00:F490  AB             ?4      PLB
5011  00:F491  2B                     PLD
5012  00:F492  68                   PLA
5013  00:F493  18                   CLC
5014  00:F494  6B                   RTL
5015
5016
5017
5018                        .END
5019
5020
5021
5022                          .STTL 'R_CLOCK.ASM - Time of Day Clock Routines'
5023                          .page
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_CLOCK.ASM - Time of Day Clock Routines'

```
5024  00:F494                       INCLUDE R_CLOCK.ASM
5025                        ;FILE:  'R_CLOCK.ASM - Time of Day Clock Routines'
5026                        ;DATE:   02-04-1995
5027
5028
5029
5030                        ;*  ROUTINE:  SET_DATE
5031                        ;*
5032                        ;*    WARNING  This routine must be entered with a JSL command.
5033                        ;*
5034                        ;*    Enter with a 16 bit Xreg pointing to a nine character
5035                        ;*      buffer that contains an update string.
5036                        ;*
5037                        ;*    The updating format  = MM-DD-YY(null).  The null termination
5038                        ;*      is unnecessary.
5039                        ;*
5040                        ;*    The Carry Bit will be set upon return if a format error
5041                        ;*      is found.
5042                        ;*
5043  00:F495
5044  00:F495                     SET_DATE:
5045
5046  00:F495  48                       PHA
5047  00:F496  5A                       PHY
5048  00:F497  DA                       PHX
5049  00:F498  08                          PHP
5050  00:F499  0B                       PHD          ;SAVE DIRECT REG
5051  00:F49A  8B                       PHB
5052  00:F49B  F4 00 00                 PEA #$0000
5053  00:F49E  2B                       PLD          ;SET DIRECT PAGE REG TO PAGE 0
5054  00:F49F  F4 00 00                 PEA #$0000     ;SET DATA BANK TO 0
5055  00:F4A2  AB                       PLB
5056  00:F4A3  AB                       PLB
5057
5058  00:F4A4  E2 20                    SEP #M8
5059                             .LONGA OFF
5060
5061  00:F4A6  20 F8 F4                 JSR CONVERT2ASCII
5062  00:F4A9  F0 45                       BEQ BAD_EXIT
5063  00:F4AB  B0 43                    BCS BAD_EXIT
5064  00:F4AD  C9 0D                    CMP #13
5065  00:F4AF  B0 3F                    BCS BAD_EXIT
5066  00:F4B1  8D 91 DF                 STA !MONTH
5067  00:F4B4  E8                       INX
5068  00:F4B5  BD 00 00                 LDA !0,X
```

```
5069  00:F4B8  C9 2D              CMP #'-'
5070  00:F4BA  F0 04              BEQ ?2
5071  00:F4BC  C9 2F              CMP #'/'
5072  00:F4BE  D0 30              BNE BAD_EXIT
5073  00:F4C0  E8           ?2    INX
5074  00:F4C1  20 F8 F4           JSR CONVERT2ASCII
5075  00:F4C4  F0 2A                BEQ BAD_EXIT
5076  00:F4C6  B0 28              BCS BAD_EXIT
5077  00:F4C8  C9 20              CMP #32
5078  00:F4CA  B0 24              BCS BAD_EXIT
5079  00:F4CC  8D 92 DF           STA !DAY
5080  00:F4CF  E8                 INX
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_CLOCK.ASM - Time of Day Clock Routines'

```
5081  00:F4D0  BD 00 00          LDA !0,X
5082  00:F4D3  C9 2D             CMP #'-'
5083  00:F4D5  F0 04             BEQ ?3
5084  00:F4D7  C9 2F             CMP #'/'
5085  00:F4D9  D0 15             BNE BAD_EXIT
5086  00:F4DB  E8          ?3     INX
5087  00:F4DC  20 F8 F4          JSR CONVERT2ASCII
5088  00:F4DF  B0 0F             BCS BAD_EXIT
5089  00:F4E1  C9 64             CMP #100
5090  00:F4E3  B0 0B             BCS BAD_EXIT
5091  00:F4E5  8D 93 DF          STA !YR
5092
5093  00:F4E8                    GOOD_EXIT:
5094  00:F4E8  AB                PLB         ;RESTORE BANK
5095  00:F4E9  2B                PLD         ;RESTORE DIRECT REG
5096  00:F4EA  28                   PLP
5097  00:F4EB  FA                PLX
5098  00:F4EC  7A                PLY
5099  00:F4ED  68                PLA
5100  00:F4EE  18                CLC
5101  00:F4EF  6B                RTL
5102
5103  00:F4F0                    BAD_EXIT:
5104  00:F4F0  AB                PLB         ;RESTORE BANK
5105  00:F4F1  2B                PLD         ;RESTORE DIRECT REG
5106  00:F4F2  28                   PLP
5107  00:F4F3  FA                PLX
5108  00:F4F4  7A                PLY
5109  00:F4F5  68                PLA
5110  00:F4F6  38                SEC
5111  00:F4F7  6B                RTL
5112
5113
5114        00:F4F8              CONVERT2ASCII EQU *  ;CONVERT 2 ASCII CHARs TO HEX
5115
5116  00:F4F8  BD 00 00          LDA !0,X
5117  00:F4FB  C9 58             CMP #'X'
5118  00:F4FD  F0 24             BEQ ?2
5119  00:F4FF  C9 78             CMP #'x'
5120  00:F501  F0 20             BEQ ?2
5121  00:F503  20 0D F1          JSR ISDECIMAL
5122  00:F506  B0 1A             BCS ?1
5123  00:F508  29 0F             AND #LOWNIB
5124  00:F50A  0A                ASL A       ;MULTIPLY BY 10
5125  00:F50B  85 6C             STA TMP8
```

```
5126  00:F50D  0A          ASL A
5127  00:F50E  0A          ASL A
5128  00:F50F  18          CLC
5129  00:F510  65 6C       ADC TMP8
5130  00:F512  85 6C       STA TMP8
5131  00:F514  E8          INX
5132  00:F515  BD 00 00    LDA !0,X
5133  00:F518  20 0D F1    JSR ISDECIMAL
5134  00:F51B  B0 05       BCS ?1
5135  00:F51D  29 0F       AND #LOWNIB
5136  00:F51F  18          CLC
5137  00:F520  65 6C       ADC TMP8
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_CLOCK.ASM - Time of Day Clock Routines'

```
5138  00:F522  60              ?1      RTS
5139
5140  00:F523  E8              ?2      INX
5141  00:F524  18                      CLC
5142  00:F525  A9 FF                   LDA #$FF
5143  00:F527  60                      RTS
5144
5145                    ;*  ROUTINE:  SET_TIME
5146                    ;*
5147                    ;*    WARNING  This routine must be entered with a JSL command.
5148                    ;*
5149                    ;*    Enter with a 16 bit Xreg pointing to a nine character
5150                    ;*      buffer that contains an update string.
5151                    ;*
5152                    ;*    The updating format  = HH:MM:SS(null).  The null termination
5153                    ;*      is unnecessary.
5154                    ;*
5155                    ;*    The Carry Bit will be set upon return if a format error
5156                    ;*      is found.
5157                    ;*
5158  00:F528
5159  00:F528                     SET_TIME:
5160
5161  00:F528  48                       PHA
5162  00:F529  5A                       PHY
5163  00:F52A  DA                       PHX
5164  00:F52B  08                          PHP
5165  00:F52C  0B                       PHD          ;SAVE DIRECT REG
5166  00:F52D  8B                       PHB
5167  00:F52E  F4 00 00                 PEA #$0000
5168  00:F531  2B                       PLD          ;SET DIRECT PAGE REG TO PAGE 0
5169  00:F532  F4 00 00                 PEA #$0000     ;SET DATA BANK TO 0
5170  00:F535  AB                       PLB
5171  00:F536  AB                       PLB
5172
5173  00:F537  E2 20                    SEP #M8
5174                                 .LONGA OFF
5175
5176  00:F539  20 F8 F4                 JSR CONVERT2ASCII
5177  00:F53C  B0 34                    BCS ST_EXIT
5178  00:F53E  C9 18                    CMP #24
5179  00:F540  B0 30                    BCS ST_EXIT
5180  00:F542  8D 94 DF                 STA !HR
5181  00:F545  E8                       INX
5182  00:F546  BD 00 00                 LDA !0,X
```

```
5183  00:F549  C9 3A           CMP #':'
5184  00:F54B  D0 25           BNE ST_EXIT
5185  00:F54D  E8              INX
5186  00:F54E  20 F8 F4        JSR CONVERT2ASCII
5187  00:F551  B0 9D           BCS BAD_EXIT
5188  00:F553  C9 3C           CMP #60
5189  00:F555  B0 1B           BCS ST_EXIT
5190  00:F557  8D 95 DF        STA !MIN
5191  00:F55A  E8              INX
5192  00:F55B  BD 00 00        LDA !0,X
5193  00:F55E  C9 3A           CMP #':'
5194  00:F560  D0 10           BNE ST_EXIT
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_CLOCK.ASM - Time of Day Clock Routines'

```
5195  00:F562  E8                INX
5196  00:F563  20 F8 F4          JSR CONVERT2ASCII
5197  00:F566  B0 88             BCS BAD_EXIT
5198  00:F568  C9 3C             CMP #60
5199  00:F56A  B0 06             BCS ST_EXIT
5200  00:F56C  8D 96 DF          STA !SEC
5201
5202  00:F56F  82 76 FF              BRL GOOD_EXIT
5203
5204
5205  00:F572  82 7B FF       ST_EXIT  BRL BAD_EXIT
5206
5207                    .page
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_CLOCK.ASM - Time of Day Clock Routines'

```
5208
5209                    ;*  ROUTINE:  SET_ALARM
5210                    ;*
5211                    ;*    WARNING  This routine must be entered with a JSL command.
5212                    ;*
5213                    ;*    Enter with a 16 bit Xreg pointing to a nine character
5214                    ;*      buffer that contains an update string.
5215                    ;*
5216                    ;*    The updating format  = HH:MM:SS(null).  The null termination
5217                    ;*      is unnecessary.
5218                    ;*
5219                    ;*    The Carry Bit will be set upon return if a format error
5220                    ;*      is found.
5221                    ;*
5222  00:F575
5223  00:F575                     SET_ALARM:
5224
5225  00:F575  48                       PHA
5226  00:F576  5A                       PHY
5227  00:F577  DA                       PHX
5228  00:F578  08                          PHP
5229  00:F579  0B                       PHD          ;SAVE DIRECT REG
5230  00:F57A  8B                       PHB
5231  00:F57B  F4 00 00                 PEA #$0000
5232  00:F57E  2B                       PLD          ;SET DIRECT PAGE REG TO PAGE 0
5233  00:F57F  F4 00 00                 PEA #$0000     ;SET DATA BANK TO 0
5234  00:F582  AB                       PLB
5235  00:F583  AB                       PLB
5236
5237  00:F584  E2 20                    SEP #M8
5238                                .LONGA OFF
5239
5240  00:F586  20 F8 F4                 JSR CONVERT2ASCII
5241  00:F589  B0 39                    BCS SA_EXIT
5242  00:F58B  C9 18                    CMP #24
5243  00:F58D  B0 35                    BCS SA_EXIT
5244  00:F58F  8D 9B DF                 STA !AHR
5245  00:F592  E8                       INX
5246  00:F593  BD 00 00                 LDA !0,X
5247  00:F596  C9 3A                    CMP #':'
5248  00:F598  D0 2A                    BNE SA_EXIT
5249  00:F59A  E8                       INX
5250  00:F59B  20 F8 F4                 JSR CONVERT2ASCII
5251  00:F59E  B0 24                    BCS SA_EXIT
5252  00:F5A0  C9 3C                    CMP #60
```

```
5253  00:F5A2  B0 20          BCS SA_EXIT
5254  00:F5A4  8D 9C DF       STA !AMIN
5255  00:F5A7  E8             INX
5256  00:F5A8  BD 00 00       LDA !0,X
5257  00:F5AB  C9 3A          CMP #':'
5258  00:F5AD  D0 15          BNE SA_EXIT
5259  00:F5AF  E8             INX
5260  00:F5B0  20 F8 F4       JSR CONVERT2ASCII
5261  00:F5B3  B0 0F          BCS SA_EXIT
5262  00:F5B5  C9 3C          CMP #60
5263  00:F5B7  B0 0B          BCS SA_EXIT
5264  00:F5B9  8D 9D DF       STA !ASEC
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_CLOCK.ASM - Time of Day Clock Routines'

```
5265
5266  00:F5BC  A9 01                LDA #ALRMENAB     ;TURN ON ALARM CHECK
5267  00:F5BE  0C B6 DF           TSB FLAGS
5268
5269  00:F5C1  82 24 FF             BRL GOOD_EXIT
5270
5271  00:F5C4                 SA_EXIT:
5272  00:F5C4  82 29 FF             BRL BAD_EXIT
5273
5274
5275
5276
5277                    ;*  ROUTINE:  RESET_ALARM
5278                    ;*
5279                    ;*    WARNING  This routine must be entered with a JSL command.
5280                    ;*
5281                    ;*    Entering this routine causes the ALARM to be disabled.
5282                    ;*
5283                    ;*
5284  00:F5C7
5285  00:F5C7               RESET_ALARM:
5286
5287  00:F5C7  48               PHA
5288  00:F5C8  5A               PHY
5289  00:F5C9  DA               PHX
5290  00:F5CA  08                  PHP
5291  00:F5CB  0B               PHD          ;SAVE DIRECT REG
5292  00:F5CC  8B               PHB
5293  00:F5CD  F4 00 00         PEA #$0000
5294  00:F5D0  2B               PLD          ;SET DIRECT PAGE REG TO PAGE 0
5295  00:F5D1  F4 00 00         PEA #$0000     ;SET DATA BANK TO 0
5296  00:F5D4  AB               PLB
5297  00:F5D5  AB               PLB
5298
5299  00:F5D6  E2 20            SEP #M8
5300                         .LONGA OFF
5301
5302  00:F5D8
5303  00:F5D8  A9 03               LDA #ALRMENAB+ALRMIRQ          ;TURN OFF ALARM
CHECK
5304  00:F5DA  1C B6 DF         TRB FLAGS
5305
5306  00:F5DD  A9 FF               LDA #$FF
5307  00:F5DF  A2 00 00         LDX #0
5308  00:F5E2  9D 97 DF      ?1    STA !ADAYWK,X
```

```
5309  00:F5E5  E8                   INX
5310  00:F5E6  E0 07 00             CPX #7
5311  00:F5E9  D0 F7                BNE ?1
5312
5313  00:F5EB  82 FA FE                BRL GOOD_EXIT
5314
5315
5316                    ;*  ROUTINE:  GET_ALARM_STATUS
5317                    ;*
5318                    ;*    WARNING  This routine must be entered with a JSL command.
5319                    ;*
5320                    ;*
5321                    ;*    This routine returns with the results in the Areg and the
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_CLOCK.ASM - Time of Day Clock Routines'

```
5322                    ;*          Carry bit.  If Cy = 1 then the ALARM has been triggered.
5323                    ;*        If the Areg = 0  the ALARM hasn not been set.
5324                    ;*
5325                    ;*    This routine will reset any ALARM that has been triggered.
5326                    ;*
5327                    ;*
5328  00:F5EE
5329  00:F5EE                     GET_ALARM_STATUS:
5330
5331
5332  00:F5EE  5A                       PHY
5333  00:F5EF  DA                       PHX
5334  00:F5F0  08                          PHP
5335  00:F5F1  0B                       PHD          ;SAVE DIRECT REG
5336  00:F5F2  8B                       PHB
5337  00:F5F3  F4 00 00                 PEA #$0000
5338  00:F5F6  2B                       PLD          ;SET DIRECT PAGE REG TO PAGE 0
5339  00:F5F7  F4 00 00                 PEA #$0000     ;SET DATA BANK TO 0
5340  00:F5FA  AB                       PLB
5341  00:F5FB  AB                       PLB
5342
5343  00:F5FC  E2 20                    SEP #M8
5344                              .LONGA OFF
5345
5346  00:F5FE  A9 02                       LDA #ALRMIRQ
5347  00:F600  1C B6 DF                 TRB FLAGS
5348  00:F603  D0 12                    BNE ALARM_SET
5349
5350  00:F605  A9 01                       LDA #ALRMENAB     ;TURN ON ALARM CHECK
5351  00:F607  2C B6 DF                 BIT FLAGS
5352  00:F60A  D0 02                       BNE ?1
5353  00:F60C  A9 00                       LDA #0
5354
5355  00:F60E  AB                 ?1      PLB          ;RESTORE BANK
5356  00:F60F  2B                       PLD          ;RESTORE DIRECT REG
5357  00:F610  28                          PLP
5358  00:F611  FA                       PLX
5359  00:F612  7A                       PLY
5360  00:F613  18                       CLC
5361  00:F614  09 00                    ORA #0        ;RESTORE Areg FLAGS
5362  00:F616  6B                       RTL
5363
5364
5365  00:F617  AB                 ALARM_SET PLB          ;RESTORE BANK
5366  00:F618  2B                       PLD          ;RESTORE DIRECT REG
```

```
5367  00:F619  28                PLP
5368  00:F61A  FA               PLX
5369  00:F61B  7A               PLY
5370  00:F61C  38               SEC
5371  00:F61D  6B               RTL
5372
5373
5374
5375
5376                ;*  ROUTINE:  READ_ALARM
5377                ;*
5378                ;*     WARNING  This routine must be entered with a JSL command.
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_CLOCK.ASM - Time of Day Clock Routines'

```
5379                    ;*
5380                    ;*    Enter with a 16 bit Xreg pointing to a nine character buffer.
5381                    ;*
5382                    ;*    The returned format  = HH:MM:SS(null).  The null termination
5383                    ;*      is for the C programmers.
5384                    ;*
5385                    ;*
5386                    ;*
5387  00:F61E
5388  00:F61E                   READ_ALARM:
5389
5390
5391  00:F61E  48                       PHA
5392  00:F61F  5A                       PHY
5393  00:F620  DA                       PHX
5394  00:F621  08                         PHP
5395  00:F622  0B                       PHD         ;SAVE DIRECT REG
5396  00:F623  8B                       PHB
5397  00:F624  F4 00 00                 PEA #$0000
5398  00:F627  2B                       PLD         ;SET DIRECT PAGE REG TO PAGE 0
5399  00:F628  F4 00 00                 PEA #$0000    ;SET DATA BANK TO 0
5400  00:F62B  AB                       PLB
5401  00:F62C  AB                       PLB
5402
5403  00:F62D  AD 9B DF                 LDA !AHR
5404  00:F630  C9 FF                    CMP #$FF
5405  00:F632  F0 0D                    BEQ ?1        ;ALARM DONT CARE CODE
5406
5407  00:F634  20 55 F8                 JSR B_DCONV
5408
5409  00:F637  A9 3A                    LDA #':'
5410  00:F639  9D 02 00                 STA !2,X
5411  00:F63C  9D 05 00                 STA !5,X
5412  00:F63F  80 08                    BRA ?2
5413
5414  00:F641  A9 78            ?1      LDA #'x'
5415  00:F643  9D 00 00                 STA !0,X
5416  00:F646  9D 01 00                 STA !1,X
5417
5418  00:F649  E8               ?2      INX
5419  00:F64A  E8                       INX
5420  00:F64B  E8                       INX
5421  00:F64C  AD 9C DF                 LDA !AMIN
5422  00:F64F  C9 FF                    CMP #$FF
5423  00:F651  F0 05                    BEQ ?3        ;ALARM DONT CARE CODE
```

```
5424
5425  00:F653  20 55 F8          JSR B_DCONV
5426  00:F656  80 08             BRA ?4
5427
5428  00:F658  A9 78        ?3    LDA #'x'
5429  00:F65A  9D 00 00          STA !0,X
5430  00:F65D  9D 01 00          STA !1,X
5431
5432  00:F660  E8           ?4    INX
5433  00:F661  E8                 INX
5434  00:F662  E8                 INX
5435  00:F663  AD 9D DF           LDA !ASEC
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_CLOCK.ASM - Time of Day Clock Routines'

```
5436  00:F666  C9 FF              CMP #$FF
5437  00:F668  F0 05              BEQ ?5        ;ALARM DONT CARE CODE
5438
5439  00:F66A  20 55 F8           JSR B_DCONV
5440  00:F66D  80 08              BRA ?6
5441
5442  00:F66F  A9 78        ?5     LDA #'x'
5443  00:F671  9D 00 00           STA !0,X
5444  00:F674  9D 01 00           STA !1,X
5445
5446
5447  00:F677  A9 00        ?6     LDA #0
5448  00:F679  9D 02 00           STA !2,X
5449
5450  00:F67C  82 69 FE              BRL GOOD_EXIT
5451                    .page
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_CLOCK.ASM - Time of Day Clock Routines'

```
5452
5453          00:F67F                CLOCK_CK_SUM EQU *
5454
5455                            .GLOBAL CLOCK_CK_SUM
5456
5457  00:F67F  48                       PHA
5458  00:F680  DA                       PHX
5459  00:F681  08                          PHP
5460  00:F682  C2 10                    REP #X8
5461                              .LONGI ON
5462
5463  00:F684  A2 07 00                 LDX #DFLTSEND-DFLTS-1 ;LOAD TOD CHECKSUM
5464  00:F687  A9 00                    LDA #00
5465  00:F689  18                       CLC
5466  00:F68A  7D 8F DF          ?1      ADC !DAYWK-1,X
5467  00:F68D  CA                       DEX
5468  00:F68E  D0 FA                    BNE ?1
5469  00:F690  6D 58 DF                 ADC T4LL      ;USE TIMER #4 CLK
5470  00:F693  6D 59 DF                 ADC T4LH      ;FOR CALCULATING TODCKS
5471  00:F696  49 FF                    EOR #$FF      ;USED IN LOW POWER MODE
5472  00:F698  8D A3 DF                 STA !TODCKS
5473
5474  00:F69B  28                          PLP
5475  00:F69C  FA                       PLX
5476  00:F69D  68                       PLA
5477  00:F69E  60                       RTS
5478
5479
5480                              STTL 'R_CLOCK.ASM - Time of day clock IRQ routine'
5481                              .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_CLOCK.ASM - Time of day clock IRQ routine'

```
5482
5483
5484                    ;* Routine:  TODIRQ
5485                    ;*
5486                    ;* Reg Used: ACC,Y,X
5487                    ;* Var Used: SEC,MIN,HR,DAY,MONTH,YR,DAYWK,DAYLIT
5488                    ;*          ASEC,AMIN,AHR,ADAY,AMONTH,AYR,ADAYWK
5489                    ;* Routines Called: NONE
5490                    ;* Returned Reg: NONE
5491                    ;*
5492
5493                        .global TODIRQ
5494
5495        00:F69F             TODIRQ   EQU *         ;MONITOR TIME OF DAY IRQ
5496  00:F69F  48                   PHA
5497  00:F6A0  5A                   PHY
5498  00:F6A1  DA                   PHX
5499  00:F6A2  08                     PHP
5500  00:F6A3  0B                   PHD          ;SAVE DIRECT REG
5501  00:F6A4  8B                   PHB
5502  00:F6A5  F4 00 00             PEA #$0000
5503  00:F6A8  2B                   PLD          ;SET DATA BANK REG TO PAGE 0
5504  00:F6A9  F4 00 00             PEA #$0000
5505  00:F6AC  AB                   PLB          ;SET BANK TO ZERO
5506  00:F6AD  AB                   PLB
5507  00:F6AE  E2 30                SEP #M8+X8     ;SET X & Acc SHORT
5508                    .LONGA OFF
5509                    .LONGI OFF
5510
5511  00:F6B0  A9 02                LDA #T1FLG     ;RESET TIMER 1 IRQ
5512  00:F6B2  8D 44 DF             STA  TIFR
5513
5514  00:F6B5                   TOD_AGIN:
5515  00:F6B5  EE 96 DF             INC !SEC       ;INCREMENT SECONDS
5516  00:F6B8  AD 96 DF             LDA !SEC
5517  00:F6BB  C9 3C                CMP #60
5518  00:F6BD  F0 16                BEQ ?CL1
5519  00:F6BF  C9 37                CMP #55
5520  00:F6C1  B0 5A                BCS EXITOCT
5521  00:F6C3  A9 08                LDA #RES_COMP
5522  00:F6C5  1C B6 DF             TRB FLAGS
5523  00:F6C8  F0 53                BEQ EXITOCT
5524  00:F6CA  EE 96 DF             INC !SEC       ;INCREMENT SECONDS
5525  00:F6CD  EE 96 DF             INC !SEC       ;INCREMENT SECONDS
5526  00:F6D0  EE 96 DF             INC !SEC       ;3 SEC'S TOTAL!
```

```
5527  00:F6D3  80 48              BRA EXITOCT
5528  00:F6D5
5529  00:F6D5  9C 96 DF     ?CL1  STZ !SEC      ;ROLLED OVER
5530
5531  00:F6D8  EE 95 DF           INC !MIN      ;INCREMENT MINUTES
5532  00:F6DB  A9 3C              LDA #60
5533  00:F6DD  CD 95 DF           CMP !MIN
5534  00:F6E0  D0 3B              BNE EXITOCT
5535  00:F6E2  9C 95 DF           STZ !MIN      ;ROLLED OVER
5536
5537  00:F6E5  EE 94 DF           INC !HR       ;INCREMENT HOUR
5538  00:F6E8  A9 02              LDA #DATE_CHK
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_CLOCK.ASM - Time of day clock IRQ routine'

```
5539  00:F6EA  0C 38 01          TSB FORMAT_FLAGS ;FOR DISPLAY UPDATE
5540  00:F6ED  AD 94 DF          LDA !HR
5541  00:F6F0  C9 01             CMP #1
5542  00:F6F2  D0 2B             BNE TODINT8
5543  00:F6F4  A9 01        OCTOBER  LDA #DAYLITFLG ;IS DAYLIGHT SAVINGS ON
5544  00:F6F6  2C A2 DF          BIT !DAYLIT
5545  00:F6F9  F0 22             BEQ EXITOCT
5546  00:F6FB  AD 91 DF          LDA !MONTH
5547  00:F6FE  C9 0A             CMP #10       ;IS IT OCTOBER
5548  00:F700  D0 1B             BNE EXITOCT
5549  00:F702  AD 90 DF          LDA !DAYWK     ;IS IT SUNDAY
5550  00:F705  C9 01             CMP #$01
5551  00:F707  D0 14             BNE EXITOCT    ;NO
5552  00:F709  AD 92 DF          LDA !DAY       ;IS IT LAST SUNDAY
5553  00:F70C  C9 19             CMP #25
5554  00:F70E  90 0D             BCC EXITOCT
5555  00:F710  A9 80             LDA #DAYLPROG  ;CK IF ALREADY SET BACK
5556  00:F712  1C A2 DF          TRB !DAYLIT
5557  00:F715  D0 06             BNE EXITOCT
5558  00:F717  0C A2 DF          TSB !DAYLIT
5559  00:F71A  9C 94 DF          STZ !HR
5560  00:F71D  80 67        EXITOCT  BRA T1EXIT
5561
5562
5563       00:F71F          TODINT8  EQU *
5564  00:F71F  C9 18             CMP #24
5565  00:F721  90 03             BCC EXITA6
5566  00:F723  9C 94 DF          STZ !HR       ;ROLLED OVER
5567  00:F726  AD 94 DF     EXITA6   LDA !HR
5568  00:F729  F0 02             BEQ TODINT9
5569  00:F72B  80 59             BRA T1EXIT
5570
5571  00:F72D  EE 90 DF     TODINT9  INC !DAYWK
5572  00:F730  AD 90 DF          LDA !DAYWK
5573  00:F733  C9 07             CMP #7
5574  00:F735  90 05             BCC INCDAY
5575  00:F737  A9 01             LDA #1
5576  00:F739  8D 90 DF          STA !DAYWK     ;ROLLED OVER
5577
5578  00:F73C  EE 92 DF     INCDAY   INC !DAY
5579  00:F73F  A9 01             LDA #DAYLITFLG ;IS DAY LIGHT SAVINGS ON
5580  00:F741  2C A2 DF          BIT !DAYLIT
5581  00:F744  F0 0A             BEQ INCADAY    ;NO
5582  00:F746  AD 91 DF          LDA !MONTH
5583  00:F749  C9 04             CMP #4        ;IS IT APRIL
```

```
5584  00:F74B  D0 03              BNE INCADAY
5585  00:F74D  82 7F 00           BRL APRIL
5586
5587  00:F750  AD 92 DF     INCADAY  LDA !DAY       ;INCREMENT DAYS
5588  00:F753  AE 91 DF           LDX !MONTH
5589  00:F756  DD 3E FF           CMP !LASTDY-1,X
5590  00:F759  90 2B              BCC T1EXIT
5591
5592  00:F75B  E0 02              CPX #2        ;INCREMENT MONTH
5593  00:F75D  D0 12              BNE INCMTH    ;NOT FEBRUARY
5594  00:F75F  AD 93 DF           LDA !YR
5595  00:F762  25 03              AND %00000011  ;IS IT LEAP YR
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_CLOCK.ASM - Time of day clock IRQ routine'

```
5596  00:F764  D0 0B              BNE INCMTH
5597
5598  00:F766  AD 92 DF           LDA !DAY       ;ITS FEB AND LEAP YR
5599  00:F769  C9 1D              CMP #29
5600  00:F76B  F0 19              BEQ T1EXIT
5601  00:F76D  22 00 F2 00           JSL SEND_BEEP
5602  00:F771  A0 01          INCMTH   LDY #1        ;ROLLED OVER
5603  00:F773  8C 92 DF           STY !DAY
5604  00:F776  EE 91 DF           INC !MONTH
5605  00:F779  AD 91 DF           LDA !MONTH
5606  00:F77C  C9 0D              CMP #13
5607  00:F77E  90 06              BCC T1EXIT
5608  00:F780  8C 91 DF           STY !MONTH     ;MONTH 1= JAN
5609
5610  00:F783  EE 93 DF           INC !YR
5611
5612
5613        00:F786           T1EXIT   EQU *
5614  00:F786  A9 01              LDA #TIME_CHK  ;FOR DISPLAY UPDATE
5615  00:F788  0C 38 01           TSB FORMAT_FLAGS
5616  00:F78B
5617  00:F78B  AD B6 DF           LDA !FLAGS     ;CK IF ALARM ENABLED
5618  00:F78E  89 01              BIT #ALRMENAB
5619  00:F790  F0 1C              BEQ EXITA
5620                               ;CHK IF WE HAVE AN ALARM
5621  00:F792  A2 00              LDX #0
5622  00:F794  BD 9B DF       CKALARM  LDA !AHR,X
5623  00:F797  C9 FF              CMP #$FF
5624  00:F799  F0 05              BEQ CKAL1
5625  00:F79B  DD 94 DF           CMP !HR,X
5626  00:F79E  D0 0E              BNE EXITA
5627
5628  00:F7A0  E8              CKAL1    INX
5629  00:F7A1  E0 03              CPX #3
5630  00:F7A3  D0 EF              BNE CKALARM
5631  00:F7A5  A9 02              LDA #ALRMIRQ   ;SET ALARM FLAG
5632  00:F7A7  0C B6 DF           TSB FLAGS
5633  00:F7AA  64 83              STZ T_TIME
5634  00:F7AC  64 84              STZ T_TIME+1
5635
5636  00:F7AE                  EXITA:
5637  00:F7AE  20 7F F6           JSR CLOCK_CK_SUM   ;DO CLOCK CHKSUM
5638
5639  00:F7B1  C2 10              REP #X8
5640                         LONGI ON
```

```
5641  00:F7B3
5642  00:F7B3  AE B8 DF          LDX PD_TIMER
5643  00:F7B6  F0 10             BEQ ?66
5644  00:F7B8  CA                DEX
5645  00:F7B9  8E B8 DF          STX PD_TIMER
5646  00:F7BC  D0 0A             BNE ?66
5647
5648
5649  00:F7BE  A2 20 4E              LDX #20000
5650  00:F7C1  CA            ?XA     DEX
5651  00:F7C2  D0 FD             BNE ?XA
5652
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_CLOCK.ASM - Time of day clock IRQ routine'

```
5653  00:F7C4  22 5E E6 00          JSL ENTER_LOW_POWER_MODE
5654
5655  00:F7C8  AB             ?66    PLB          ;RESTORE SIZE OF Acc & X/Y REGS
5656  00:F7C9  2B                    PLD
5657  00:F7CA  28                      PLP
5658  00:F7CB  FA                    PLX
5659  00:F7CC  7A                    PLY
5660  00:F7CD  68                    PLA
5661  00:F7CE  40                    RTI
5662
5663
5664
5665  00:F7CF  AD 90 DF       APRIL   LDA !DAYWK     ;IS IT SUNDAY
5666  00:F7D2  C9 01                 CMP #$01
5667  00:F7D4  D0 07                 BNE APR0      ;NO
5668  00:F7D6  AD 92 DF              LDA !DAY       ;IS IT 1ST SUNDAY
5669  00:F7D9  C9 08                 CMP #8
5670  00:F7DB  90 03                 BCC APR1
5671  00:F7DD  82 70 FF       APR0    BRL INCADAY
5672
5673
5674  00:F7E0  A9 01          APR1    LDA #1
5675  00:F7E2  8D 94 DF              STA !HR
5676  00:F7E5  80 9F                 BRA T1EXIT
5677
5678
5679
5680                    .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_CLOCK.ASM - Time of day clock IRQ routine'

```
5681
5682                    ;*  ROUTINE:  READ_DATE
5683                    ;*
5684                    ;*    WARNING  This routine must be entered with a JSL command.
5685                    ;*
5686                    ;*    Enter with a 16 bit Xreg pointing to a nine character buffer.
5687                    ;*
5688                    ;*    The returned format  = MM-DD-YY(null).  The null termination
5689                    ;*      is for the C programmers.
5690                    ;*
5691                    ;*
5692                    ;*
5693  00:F7E7
5694  00:F7E7                     READ_DATE:
5695
5696
5697  00:F7E7  48                       PHA
5698  00:F7E8  5A                       PHY
5699  00:F7E9  DA                       PHX
5700  00:F7EA  08                          PHP
5701  00:F7EB  0B                       PHD          ;SAVE DIRECT REG
5702  00:F7EC  8B                       PHB
5703  00:F7ED  F4 00 00                 PEA #$0000
5704  00:F7F0  2B                       PLD          ;SET DIRECT PAGE REG TO PAGE 0
5705  00:F7F1  F4 00 00                 PEA #$0000     ;SET DATA BANK TO 0
5706  00:F7F4  AB                       PLB
5707  00:F7F5  AB                       PLB
5708
5709  00:F7F6  AD 91 DF                 LDA !MONTH
5710  00:F7F9  20 55 F8                 JSR B_DCONV
5711
5712  00:F7FC  A9 2D                    LDA #'-'
5713  00:F7FE  9D 02 00                 STA !2,X
5714  00:F801  9D 05 00                 STA !5,X
5715
5716  00:F804  E8                       INX
5717  00:F805  E8                       INX
5718  00:F806  E8                       INX
5719  00:F807  AD 92 DF                 LDA !DAY
5720  00:F80A  20 55 F8                 JSR B_DCONV
5721
5722  00:F80D  E8                       INX
5723  00:F80E  E8                       INX
5724  00:F80F  E8                       INX
5725  00:F810  AD 93 DF                 LDA !YR
```

```
5726  00:F813  20 55 F8              JSR B_DCONV
5727
5728  00:F816  A9 00                 LDA #0
5729  00:F818  9D 02 00              STA !2,X
5730
5731  00:F81B  82 CA FC                 BRL GOOD_EXIT
5732
5733
5734                     ;*  ROUTINE:  READ_TIME
5735                     ;*
5736                     ;*    WARNING  This routine must be entered with a JSL command.
5737                     ;*
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_CLOCK.ASM - Time of day clock IRQ routine'

```
5738                    ;*    Enter with a 16 bit Xreg pointing to a nine character buffer.
5739                    ;*
5740                    ;*    The returned format  = HH:MM:SS(null).  The null termination
5741                    ;*      is for the C programmers.
5742                    ;*
5743                    ;*
5744                    ;*
5745  00:F81E
5746  00:F81E                 READ_TIME:
5747
5748  00:F81E  48                   PHA
5749  00:F81F  5A                   PHY
5750  00:F820  DA                   PHX
5751  00:F821  08                     PHP
5752  00:F822  0B                   PHD         ;SAVE DIRECT REG
5753  00:F823  8B                   PHB
5754  00:F824  F4 00 00             PEA #$0000
5755  00:F827  2B                   PLD         ;SET DIRECT PAGE REG TO PAGE 0
5756  00:F828  F4 00 00             PEA #$0000    ;SET DATA BANK TO 0
5757  00:F82B  AB                   PLB
5758  00:F82C  AB                   PLB
5759
5760  00:F82D  AD 94 DF             LDA !HR
5761  00:F830  20 55 F8             JSR B_DCONV
5762
5763  00:F833  A9 3A                LDA #':'
5764  00:F835  9D 02 00             STA !2,X
5765  00:F838  9D 05 00             STA !5,X
5766
5767  00:F83B  E8                   INX
5768  00:F83C  E8                   INX
5769  00:F83D  E8                   INX
5770  00:F83E  AD 95 DF             LDA !MIN
5771  00:F841  20 55 F8             JSR B_DCONV
5772
5773  00:F844  E8                   INX
5774  00:F845  E8                   INX
5775  00:F846  E8                   INX
5776  00:F847  AD 96 DF             LDA !SEC
5777  00:F84A  20 55 F8             JSR B_DCONV
5778
5779  00:F84D  A9 00                LDA #0
5780  00:F84F  9D 02 00             STA !2,X
5781
5782  00:F852  82 93 FC               BRL GOOD_EXIT
```

```
5783
5784         00:F855              B_DCONV EQU *   ;CONVERT BINARY NUMBER TO
DUODECIMAL
5785
5786  00:F855  DA                        PHX
5787  00:F856  48                        PHA
5788  00:F857  29 0F                     AND #LOWNIB
5789  00:F859  EB                        XBA
5790  00:F85A  A9 00                     LDA #0
5791  00:F85C  EB                        XBA
5792  00:F85D  AA                        TAX
5793  00:F85E  BD 5B FF                  LDA !BINDECL,X
5794  00:F861  85 6C                     STA TMP8
```

'MENSCH COMPUTER ROM SOFTWARE'
'R_CLOCK.ASM - Time of day clock IRQ routine'

```
5795  00:F863  68          PLA
5796  00:F864  4A          LSR A
5797  00:F865  4A          LSR A
5798  00:F866  4A          LSR A
5799  00:F867  4A          LSR A
5800  00:F868  AA          TAX
5801  00:F869  BD 6B FF    LDA !BINDECH,X
5802  00:F86C  18          CLC
5803  00:F86D  F8          SED
5804  00:F86E  65 6C       ADC TMP8
5805  00:F870  D8          CLD
5806  00:F871  FA          PLX
5807
5808  00:F872  48          PHA         ;OUTPUT DATA TO BUFFER VIA Xreg
5809  00:F873  29 0F       AND #LOWNIB
5810  00:F875  09 30       ORA #$30
5811  00:F877  9D 01 00    STA !1,X
5812  00:F87A  68          PLA
5813  00:F87B  4A          LSR A
5814  00:F87C  4A          LSR A
5815  00:F87D  4A          LSR A
5816  00:F87E  4A          LSR A
5817  00:F87F  09 30       ORA #$30
5818  00:F881  9D 00 00    STA !0,X
5819
5820  00:F884  60          RTS
5821
5822
5823
5824                       .END
5825
5826
5827
5828                       .sttl 'ROM Serial I/O Routines   UARTs'
5829                       .page
```

```
    'MENSCH COMPUTER ROM SOFTWARE'
    'ROM Serial I/O Routines   UARTs'

5830  00:F884                       INCLUDE R_SERIAL.ASM
5831                      ;FILE  R_Serial.asm
5832                      ;DATE: 12-17-1994
5833
5834
5835                      ;* Routine:  IRQ_SERIAL_RCV0
5836                      ;*
5837                      ;* This is the interrupt routine that buffers characters
5838                      ;*   received from the Keyboard.
5839                      ;*
5840                      ;* All REGISTERS are preserved!
5841                      ;*
5842                      ;* Returned Reg: NONE
5843                      ;*
5844                      ;*
5845                      ;* Important Variables:
5846                      ;*   SINEND0  index for last character removed from buffer
5847                      ;*   SININDX0  index for last character placed in buffer
5848                      ;*   SIN_BUF0  base location of keyboard input buffer
5849                      ;*   SINCNT0   buffer size
5850
5851
5852
5853                                ;CALLED BY IRQ ROUTINE
5854        00:F885               IRQAR0 EQU *   ;QUEUE UP SERIAL BYTE
5855  00:F885  48                       PHA
5856  00:F886  5A                       PHY
5857  00:F887  DA                       PHX
5858  00:F888  08                          PHP
5859  00:F889  0B                       PHD          ;SAVE DIRECT REG
5860  00:F88A  8B                       PHB
5861  00:F88B  F4 00 00                 PEA #$0000
5862  00:F88E  2B                       PLD          ;SET DIRECT PAGE REG TO PAGE 0
5863  00:F88F  F4 00 00                 PEA #$0000     ;SET DATA BANK TO 0
5864  00:F892  AB                       PLB
5865  00:F893  AB                       PLB
5866  00:F894  E2 20                    SEP #M8       ;SET Acc SHORT
5867  00:F896  C2 10                       REP #X8         ;SET X & Y LONG
5868                      .LONGA OFF
5869                      .LONGI ON
5870
5871  00:F898  A9 01                       LDA #01         ;CLEAR INTERRUPT
5872  00:F89A  8D 48 DF                 STA UIFR      ;Flag Reg
5873
5874  00:F89D  AD 71 DF                    LDA ARTD0      ;GET DATA CHAR
```

```
 5875  00:F8A0  48                PHA
 5876  00:F8A1  C9 20             CMP #' '      ;USE SPACE TO CLEAR ALARM
 5877  00:F8A3  D0 0C             BNE RECV0_R8
 5878  00:F8A5
 5879  00:F8A5  A9 02             LDA #ALRMIRQ   ;HAS ALARM BEEN SET?
 5880  00:F8A7  2C B6 DF          BIT FLAGS
 5881  00:F8AA  F0 05             BEQ RECV0_R8   ;NO
 5882
 5883  00:F8AC  A9 04             LDA #ALRMRST   ;WILL CAUSE TIMERS TO RESET
ALARM
 5884  00:F8AE  0C B6 DF          TSB FLAGS
 5885
 5886  00:F8B1  A4 00          RECV0_R8 LDY SININDX0    ;CHECK FOR BUFFER FULL
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
5887  00:F8B3  C8                 INY
5888  00:F8B4  C4 06              CPY SINCNT0    ;BUFFER SIZE
5889  00:F8B6  D0 03                 BNE RECV0_R9
5890  00:F8B8  A0 00 00              LDY #0              ;OVER RAN END OF BUFFER
5891  00:F8BB  C4 02           RECV0_R9 CPY SINEND0
5892  00:F8BD  D0 10              BNE RECV0_R11
5893
5894  00:F8BF  A9 40                 LDA #$40       ;Just overran the input buffer
5895  00:F8C1  04 48                 TSB STATUS_S0       ;SET the STATUS FLAG
5896  00:F8C3  A6 02              LDX SINEND0    ;THROW AWAY 1 CHAR..THE
5897  00:F8C5  E8                 INX            ;OUTPUT IS NOT KEEPING UP!
5898  00:F8C6  E4 06              CPX SINCNT0
5899  00:F8C8  D0 03              BNE RECV0_R10
5900  00:F8CA  A2 00 00           LDX #0         ;OVER-RAN END OF BUFFER
5901  00:F8CD  86 02           RECV0_R10 STX SINEND0   ;SAVE OUTPUT POINTER
5902  00:F8CF  84 00           RECV0_R11 STY SININDX0   ;SAVE INPUT PTR
5903  00:F8D1  68                    PLA
5904  00:F8D2  91 04                 STA (SIN_BUF0),Y ;STORE DATA
5905
5906  00:F8D4  A9 01                 LDA #SFLG
5907  00:F8D6  04 40                 TSB SFLAG0  ;SET CHAR READY FLAG
5908
5909  00:F8D8  C8                    INY            ;SEE IF BUFFER NEARING FULL
5910  00:F8D9  C8                 INY
5911  00:F8DA  C8                    INY
5912  00:F8DB  C8                 INY
5913  00:F8DC  C8                    INY
5914  00:F8DD  C4 06                 CPY SINCNT0         ;OUTPUT PTR
5915  00:F8DF  90 03                 BCC RECV0_R12
5916  00:F8E1  38                    SEC
5917  00:F8E2  E5 06              SBC SINCNT0
5918  00:F8E4  C4 02           RECV0_R12 CPY SINEND0
5919  00:F8E6  F0 03                 BEQ RECV0_R14       ;OK  NOTE: It is not posible
5920  00:F8E8  82 CA 04              BRL REC_DONE   ;to jump past the output ptr
5921                                       ;because we check after each
5922                                       ;char inputted.
5923
5924
5925  00:F8EB  A9 40           RECV0_R14 LDA #BEEP     ;flag to computer for input overflow
5926  00:F8ED  04 40              TSB SFLAG0
5927  00:F8EF  82 C3 04              BRL REC_DONE
5928  00:F8F2
5929                        .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
5930                    ;* Routine:  IRQAT0
5931                    ;*
5932                    ;* Var Used: SFLAG0,SOUTINDX0,SOUTEND0,SOUTCNT0
5933                    ;*
5934                    ;* Returned Reg: NONE
5935                    ;*
5936                    ;* Important Variables:
5937                    ;*   SOUTENDO   index for last character STORED in buffer
5938                    ;*   SOUTINDX0  index for last character REMOVED from buffer
5939                    ;*   SOUT_BUF0  base location of keyboard output buffer
5940                    ;*   SOUTCNT0   keyboard output buffer size
5941
5942                                    ;CALLED BY IRQ ROUTINE
5943          00:F8F2             IRQAT0 EQU *            ;DEQUEUE SERIAL BYTE
5944  00:F8F2  48                       PHA           ;FROM OUTPUT BUFFER
5945  00:F8F3  5A                       PHY           ;SEE OUTCH_PORTX ROUTINE
5946  00:F8F4  DA                       PHX
5947  00:F8F5  08                          PHP
5948  00:F8F6  0B                       PHD           ;SAVE DIRECT REG
5949  00:F8F7  8B                       PHB
5950  00:F8F8  F4 00 00                 PEA #$0000
5951  00:F8FB  2B                       PLD           ;SET DIRECT PAGE REG TO PAGE 0
5952  00:F8FC  F4 00 00                 PEA #$0000       ;SET DATA BANK TO 0
5953  00:F8FF  AB                       PLB
5954  00:F900  AB                       PLB
5955  00:F901  E2 20                    SEP #M8         ;SET Acc SHORT
5956  00:F903  C2 10                       REP #X8          ;SET X & Y LONG
5957                      .LONGA OFF
5958                      .LONGI ON
5959
5960  00:F905  A9 02                       LDA #02          ;CLEAR INTERRUPT
5961  00:F907  8D 48 DF                 STA UIFR
5962
5963  00:F90A  AD 70 DF                 LDA ACSR0          ;are we waiting for the serial reg
5964  00:F90D  29 02                    AND #DISCH         ;to clear?
5965  00:F90F  F0 07                    BEQ TRAN0_2B      ;NO
5966
5967  00:F911  AD 20 DF                 LDA PD4          ;DSR0 = P45
5968  00:F914  29 20                    AND #$20
5969  00:F916  D0 3F                    BNE TRAN0_3A    ;DSR0 IS FALSE!
5970
5971  00:F918  A4 20             TRAN0_2B LDY SOUTINDX0     ;IS BUFFER NOW EMPTY
5972  00:F91A  C4 22                    CPY SOUTEND0
5973  00:F91C  D0 11                    BNE TRAN0_3       ;KEEP GOING
5974
```

```
 5975  00:F91E  AD 70 DF       TRAN0_2C LDA ACSR0       ;DISABLE SERIAL XMIT
 5976  00:F921  29 02                  AND #DISCH        ;EVERYTHING IS OUT OF
 5977  00:F923  D0 32                  BNE TRAN0_3A    ;SERIAL XMIT REG
 5978  00:F925  A9 00                  LDA #00    ;SEND NULL TO CLEAR INTERRUPT
 5979  00:F927  80 12                  BRA TRAN0_1
 5980
 5981
 5982  00:F929  89 08           TRAN0_1A BIT #SNDOVF       ;DID WE HAVE AN OVERFLOW
CONDITION?
 5983  00:F92B  F0 EB                  BEQ TRAN0_2B    ;NO
 5984  00:F92D  80 28                  BRA TRAN0_3A    ;YES
 5985
 5986          00:F92F           TRAN0_3  EQU *
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
5987  00:F92F  C8                 INY
5988  00:F930  C4 26              CPY SOUTCNT0     ;DID WE ROLL OVER IN THE BUFFER
5989
5990  00:F932  90 03              BCC TRAN0_0      ;NO
5991  00:F934  A0 00 00           LDY #0           ;YEP
5992  00:F937  84 20        TRAN0_0  STY SOUTINDX0
5993  00:F939  B1 24              LDA (SOUTBUF0),Y    ;GET DATA FROM QUEUE
5994        00:F93B         TRAN0_1 EQU *
5995  00:F93B  8D 71 DF           STA ARTD0        ;SEND DATA
5996
5997  00:F93E  A5 40        TRAN0_2  LDA SFLAG0
5998  00:F940  89 08              BIT #SNDOVF
5999  00:F942  D0 13              BNE TRAN0_3A    ;WE'RE THROUGH
6000
6001  00:F944  A4 20              LDY SOUTINDX0
6002  00:F946  C4 22              CPY SOUTEND0     ;IS BUFF EMPTY
6003  00:F948  F0 1A              BEQ TRAN0_3B    ;YES  START SHUTDOWN
6004  00:F94A  A9 02              LDA #DISCH
6005  00:F94C  1C 70 DF           TRB ACSR0
6006  00:F94F  A9 01              LDA #SON         ;TURN OFF DISCHARE MODE
6007  00:F951  0C 70 DF           TSB ACSR0        ;AND SET TRANS IRQ'S
6008  00:F954  82 5E 04           BRL TRANS_DONE
6009
6010        00:F957         TRAN0_3A EQU *              ;SHUT DOWN UART OUTPUT
6011  00:F957  A9 02              LDA #Bit1        ;set the output high to prevent
6012  00:F959  0C 22 DF           TSB PD6          ;FALSING when we shut it off.
6013  00:F95C  A9 03              LDA #SON+DISCH
6014  00:F95E  1C 70 DF           TRB ACSR0
6015  00:F961  82 51 04           BRL TRANS_DONE
6016
6017        00:F964         TRAN0_3B EQU *              ;START SHUTDOWN
6018  00:F964  A9 03              LDA #SON+DISCH     ;TURN ON XMIT & DISCHARGE
6019  00:F966  0C 70 DF           TSB ACSR0
6020  00:F969  82 49 04           BRL TRANS_DONE
6021
6022                        .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
6023
6024
6025                    ;REAL TIME INPUT PORT 1   --- PRINTER
6026
6027
6028                    ;* Routine:  IRQAR1
6029                    ;*
6030                    ;* This routine is called by the interrrupt vector
6031                    ;*   and puts characters from the printer into the
6032                    ;*   printer input buffer.  XON/XOFF protocal is used
6033                    ;*   to control data flow to the printer.
6034                    ;*
6035                    ;* Reg Used: ACC
6036                    ;*
6037                    ;* Var Used: SFLAG1,SININDX1,SINEND1,SINCNT1
6038                    ;*
6039                    ;* Returned Reg: NONE
6040                    ;*
6041                    ;* Important Variables:
6042                    ;*   SINEND1   index for last character STORED in buffer
6043                    ;*   SININDX1  index for last character REMOVED from buffer
6044                    ;*   SIN_BUF1  base location of printer output buffer
6045                    ;*   SINCNT1   printer output buffer size
6046
6047
6048        00:F96C             IRQAR1 EQU *   ;QUEUE UP SERIAL BYTE
6049  00:F96C  48                       PHA
6050  00:F96D  5A                       PHY
6051  00:F96E  DA                       PHX
6052  00:F96F  08                          PHP
6053  00:F970  0B                       PHD          ;SAVE DIRECT REG
6054  00:F971  8B                       PHB
6055  00:F972  F4 00 00                 PEA #$0000
6056  00:F975  2B                       PLD          ;SET DIRECT PAGE REG TO PAGE 0
6057  00:F976  F4 00 00                 PEA #$0000    ;SET DATA BANK TO 0
6058  00:F979  AB                       PLB
6059  00:F97A  AB                       PLB
6060  00:F97B  E2 20                    SEP #M8       ;SET Acc SHORT
6061  00:F97D  C2 10                       REP #X8         ;SET X & Y LONG
6062                    .LONGA OFF
6063                    .LONGI ON
6064
6065  00:F97F  A9 04                       LDA #04         ;CLEAR INTERRUPT
6066  00:F981  8D 48 DF                 STA UIFR
6067
```

```
6068  00:F984  18                CLC             ;Clear C
6069  00:F985  AD 73 DF          LDA ARTD1       ;GET DATA CHAR
6070  00:F988  48              PHA
6071  00:F989  A5 41           LDA SFLAG1      ;CHECK FOR XON/XOFF OPERATION
6072  00:F98B  89 04           BIT #XONOFLG
6073  00:F98D  F0 0A              BEQ RECV1_R8        ;NOT XON/XOFF PROTOCALL
6074  00:F98F  68                 PLA
6075  00:F990  48                 PHA
6076  00:F991  C9 13              CMP #XOFF
6077  00:F993  F0 53           BEQ RECV1_XOFF
6078  00:F995  C9 11           CMP #XON
6079  00:F997  F0 56           BEQ RECV1_XON
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
6080  00:F999  A4 08          RECV1_R8 LDY SININDX1    ;CHECK FOR BUFFER FULL
6081  00:F99B  C8                    INY
6082  00:F99C  C4 0E                 CPY SINCNT1     ;BUFFER SIZE
6083  00:F99E  D0 03                    BNE RECV1_R9
6084  00:F9A0  A0 00 00                 LDY #0                ;OVER RAN END OF BUFFER
6085  00:F9A3  C4 0A          RECV1_R9 CPY SINEND1
6086  00:F9A5  D0 10                 BNE RECV1_R11
6087
6088  00:F9A7  A9 40                    LDA #$40       ;Just overran the input buffer
6089  00:F9A9  04 49                    TSB STATUS_S1       ;SET the STATUS FLAG
6090  00:F9AB  A6 0A                 LDX SINEND1     ;THROW AWAY 1 CHAR..THE
6091  00:F9AD  E8                    INX                   ;OUTPUT IS NOT KEEPING UP!
6092  00:F9AE  E4 0E                 CPX SINCNT1
6093  00:F9B0  D0 03                 BNE RECV1_R10
6094  00:F9B2  A2 00 00              LDX #0              ;OVER-RAN END OF BUFFER
6095  00:F9B5  86 0A          RECV1_R10 STX SINEND1   ;SAVE OUTPUT POINTER
6096  00:F9B7  84 08          RECV1_R11 STY SININDX1   ;SAVE INPUT PTR
6097  00:F9B9  68                    PLA
6098  00:F9BA  91 0C                 STA (SIN_BUF1),Y ;STORE DATA
6099
6100  00:F9BC  A9 01                 LDA #SFLG
6101  00:F9BE  04 41                 TSB SFLAG1  ;SET CHAR READY FLAG
6102
6103  00:F9C0  C8                    INY              ;SEE IF BUFFER NEARING FULL
6104  00:F9C1  C8                 INY
6105  00:F9C2  C8                    INY
6106  00:F9C3  C8                 INY
6107  00:F9C4  C8                    INY
6108  00:F9C5  C4 0E                 CPY SINCNT1           ;OUTPUT PTR
6109  00:F9C7  90 03                 BCC RECV1_R12
6110  00:F9C9  38                    SEC
6111  00:F9CA  E5 0E                 SBC SINCNT1
6112  00:F9CC  C4 0A          RECV1_R12 CPY SINEND1
6113  00:F9CE  F0 03                 BEQ RECV1_R14       ;OK  NOTE: It is not posible
6114  00:F9D0  82 E2 03              BRL REC_DONE   ;to jump past the output ptr
6115                                       ;because we check after each
6116                                       ;char inputted.
6117
6118
6119  00:F9D3  A5 41          RECV1_R14 LDA SFLAG1    ;CHECK PROTOCAL
6120  00:F9D5  89 04                 BIT #XONOFLG
6121  00:F9D7  F0 23                 BEQ RECV1_DTR
6122  00:F9D9  A9 10                 LDA #LASTXONOF
6123  00:F9DB  14 41                 TRB SFLAG1
6124  00:F9DD  D0 03                 BNE RECV1_R15
```

```
6125  00:F9DF  82 D3 03              BRL REC_DONE       ;XOFF ALREADY SENT
6126
6127  00:F9E2  A9 40          RECV1_R15 LDA #SXOFFLG
6128  00:F9E4  04 41                TSB SFLAG1
6129  00:F9E6  80 0C                BRA REC1_DONET
6130
6131  00:F9E8  68             RECV1_XOFF PLA             ;ADJUST STACK PTR
6132  00:F9E9  A9 08                LDA #SNDOVF
6133  00:F9EB  04 41                TSB SFLAG1  ;MAKE INPUT STOP
6134  00:F9ED  80 05                BRA REC1_DONET
6135
6136
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
6137  00:F9EF  68             RECV1_XON PLA
6138  00:F9F0  A9 08                 LDA #SNDOVF
6139  00:F9F2  14 41                 TRB SFLAG1  ;START UP INPUT AGAIN
6140  00:F9F4  A9 03           REC1_DONET LDA #3
6141  00:F9F6  0C 72 DF              TSB ACSR1    ;START UP OUTPUT
6142  00:F9F9  82 B9 03             BRL REC_DONE
6143
6144
6145  00:F9FC  A9 04           RECV1_DTR LDA #4
6146  00:F9FE  0C 21 DF              TSB PD5        ;SET DTR HI
6147  00:FA01  82 B1 03              BRL REC_DONE
6148
6149
6150                         .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
6151
6152                    ;* Routine:  IRQAT1    REAL-TIME PRINTER OUTPUT
6153                    ;*
6154                    ;*  This routine takes data from the printer output
6155                    ;*    buffer and sends it out the UART port. It is
6156                    ;*    called by a hardware interrupt vector.
6157                    ;*
6158                    ;* Var Used: SFLAG1,SOUTINDX1,SOUTEND1,SOUTCNT1
6159                    ;*
6160                    ;* Returned Reg: NONE
6161                    ;*
6162                    ;* Important Variables:
6163                    ;*   SOUTEND1   index for last character STORED in buffer
6164                    ;*   SOUTINDX1  index for last character REMOVED from buffer
6165                    ;*   SOUT_BUF1  base location of printer output buffer
6166                    ;*   SOUTCNT1   PRINTER output buffer size
6167                    ;*
6168                    ;*
6169                                    ;CALLED BY IRQ ROUTINE
6170        00:FA04             IRQAT1 EQU *   ;DEQUEUE SERIAL BYTE
6171  00:FA04  48                       PHA          ;FROM OUTPUT BUFFER
6172  00:FA05  5A                       PHY          ;SEE OUTCH_PORTX ROUTINE
6173  00:FA06  DA                       PHX
6174  00:FA07  08                          PHP
6175  00:FA08  0B                       PHD          ;SAVE DIRECT REG
6176  00:FA09  8B                       PHB
6177  00:FA0A  F4 00 00                 PEA #$0000
6178  00:FA0D  2B                       PLD          ;SET DIRECT PAGE REG TO PAGE 0
6179  00:FA0E  F4 00 00                 PEA #$0000     ;SET DATA BANK TO 0
6180  00:FA11  AB                       PLB
6181  00:FA12  AB                       PLB
6182  00:FA13  E2 20                    SEP #M8       ;SET Acc SHORT
6183  00:FA15  C2 10                       REP #X8         ;SET X & Y LONG
6184                          .LONGA OFF
6185                          .LONGI ON
6186
6187  00:FA17  A9 08                       LDA #08         ;CLEAR INTERRUPT
6188  00:FA19  8D 48 DF                 STA UIFR
6189
6190  00:FA1C  A5 41                    LDA SFLAG1     ;CK IF WE ARE OVERFLOWING
6191  00:FA1E  29 C0                    AND #SXOFFLG+SXONFLG
6192  00:FA20  F0 1C                    BEQ TRAN1_0A       ;QUICK CK FOR CNTRL FLGS
6193  00:FA22  C9 C0                    CMP #SXOFFLG+SXONFLG
6194  00:FA24  F0 14                    BEQ TRAN1_ERR
6195  00:FA26  89 40                    BIT #SXOFFLG
```

```
 6196  00:FA28  D0 08                BNE TRAN1_XOFF    ;SEND XOFF
 6197
 6198         00:FA2A             TRAN1_XON EQU *
 6199  00:FA2A  A9 80                LDA #SXONFLG      ;Turn off XON FLG & send XON
 6200  00:FA2C  14 41                TRB SFLAG1
 6201  00:FA2E  A9 11                LDA #XON
 6202  00:FA30  80 43                BRA TRAN1_1
 6203
 6204
 6205         00:FA32             TRAN1_XOFF  EQU *              ;Turn off XOFF FLG + LASTXON
FLG
 6206                                          ;and send XON
 6207  00:FA32  A9 50                LDA #SXOFFLG+LASTXONOF
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
6208  00:FA34  14 41          TRB SFLAG1
6209  00:FA36  A9 13          LDA #XOFF
6210  00:FA38  80 3B          BRA TRAN1_1
6211
6212        00:FA3A       TRAN1_ERR EQU *          ;both XON & XOFF FLGs are set..they
6213                            ;cancel one another!
6214  00:FA3A  A9 C0          LDA #SXONFLG+SXOFFLG
6215  00:FA3C  14 41          TRB SFLAG1
6216
6217        00:FA3E       TRAN1_0A EQU *
6218  00:FA3E  AD 72 DF       LDA ACSR1        ;Are we waiting for the serial reg
6219  00:FA41  89 02          BIT #DISCH       ;to clear?.
6220  00:FA43  F0 0D          BEQ TRAN1_2B     ;NO
6221
6222  00:FA45  A5 41       TRAN1_1B LDA SFLAG1       ;XON/ XOFF PROTOCALL?
6223  00:FA47  89 04          BIT #XONOFLG
6224  00:FA49  D0 18          BNE TRAN1_1A     ;YES
6225                             ;CHECK IF WE HAVE HRDW HS & DSR1 IS LOW
6226  00:FA4B  AD 21 DF       LDA PD5          ;DSR1 = P53
6227  00:FA4E  29 08          AND #Bit3
6228  00:FA50  D0 4C          BNE TRAN1_3A     ;DSR1 IS FALSE
6229
6230  00:FA52  A4 28       TRAN1_2B LDY SOUTINDX1    ;IS BUFFER NOW EMPTY
6231  00:FA54  C4 2A          CPY SOUTEND1
6232  00:FA56  D0 11          BNE TRAN1_3      ;KEEP GOING
6233  00:FA58  AD 72 DF       LDA ACSR1        ;DISABLE SERIAL XMIT
6234  00:FA5B  29 02          AND #DISCH       ;EVERYTHING IS OUT OF
6235  00:FA5D  D0 3F          BNE TRAN1_3A     ;THE SERIAL XMIT REG
6236  00:FA5F  A9 00          LDA #00          ;SEND NULL TO CLEAR INTERRUPT
6237  00:FA61  80 12          BRA TRAN1_1
6238
6239  00:FA63  89 08       TRAN1_1A BIT #SNDOVF      ;DID WE HAVE AN OVERFLOW
CONDITION?
6240  00:FA65  F0 EB          BEQ TRAN1_2B     ;NO
6241  00:FA67  80 35          BRA TRAN1_3A     ;YES
6242
6243        00:FA69       TRAN1_3  EQU *
6244  00:FA69  C8             INY
6245  00:FA6A  C4 2E          CPY SOUTCNT1     ;DID WE ROLL OVER IN THE BUFFER
6246
6247  00:FA6C  90 03          BCC TRAN1_0      ;NO
6248  00:FA6E  A0 00 00       LDY #0           ;YEP
6249  00:FA71  84 28       TRAN1_0  STY SOUTINDX1
6250
6251  00:FA73  B1 2C          LDA (SOUTBUF1),Y    ;GET DATA FROM QUEUE
```

```
6252          00:FA75              TRAN1_1  EQU *
6253  00:FA75  8D 73 DF                STA ARTD1        ;SEND DATA
6254
6255  00:FA78  A5 41                   LDA SFLAG1       ;CHECK PROTOCALL
6256  00:FA7A  89 04                   BIT #XONOFLG
6257  00:FA7C  D0 07                   BNE TRAN1_2      ;XON/XOFF
6258
6259  00:FA7E  AD 21 DF                LDA PD5          ;DTR PROTOCALL
6260  00:FA81  29 08                   AND #Bit3        ;CHECK DSR LEAD FOR DTR IN
6261  00:FA83  D0 19                   BNE TRAN1_3A     ;P53 HI   --STOP!
6262
6263  00:FA85  A5 41              TRAN1_2  LDA SFLAG1
6264  00:FA87  89 08                   BIT #SNDOVF
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
6265  00:FA89  D0 13          BNE TRAN1_3A     ;WE'RE THROUGH  DISCHARGE IS
DONE
6266
6267  00:FA8B  A4 28          LDY SOUTINDX1
6268  00:FA8D  C4 2A          CPY SOUTEND1     ;IS BUFF EMPTY
6269  00:FA8F  F0 1A          BEQ TRAN1_3B     ;YES - START SHUTDOWN
6270  00:FA91  A9 02          LDA #DISCH
6271  00:FA93  1C 72 DF       TRB ACSR1
6272  00:FA96  A9 01          LDA #SON
6273  00:FA98  0C 72 DF       TSB ACSR1
6274  00:FA9B  82 17 03       BRL TRANS_DONE
6275
6276        00:FA9E       TRAN1_3A EQU *          ;SHUT DOWN UART OUTPUT
6277  00:FA9E  A9 08          LDA #Bit3        ;set the output high to prevent falsing
6278  00:FAA0  0C 22 DF       TSB PD6          ;when we shut it off.
6279  00:FAA3  A9 03          LDA #SON+DISCH
6280  00:FAA5  1C 72 DF       TRB ACSR1
6281  00:FAA8  82 0A 03       BRL TRANS_DONE
6282
6283        00:FAAB       TRAN1_3B EQU *          ;START SHUTDOWN
6284  00:FAAB  A9 03          LDA #SON+DISCH     ;TURN ON XMIT & DISCHARGE
6285  00:FAAD  0C 72 DF       TSB ACSR1
6286  00:FAB0  82 02 03       BRL TRANS_DONE
6287
6288
6289                          .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
6290
6291                    ;REAL TIME INPUT PORT 2   --- MODEM
6292
6293
6294                    ;* Routine:  IRQAR2
6295                    ;*
6296                    ;* This routine is called by the interrrupt vector
6297                    ;*   and places characters from the MODEM into the
6298                    ;*   MODEM input buffer.  XON/XOFF protocal is used
6299                    ;*   to control data flow to & from the MODEM.
6300                    ;*
6301                    ;* Reg Used: ACC
6302                    ;*
6303                    ;* Var Used: SFLAG2,SININDX2,SINEND2,SINCNT2
6304                    ;*
6305                    ;* Returned Reg: NONE
6306                    ;*
6307                    ;* Important Variables:
6308                    ;*   SINEND2;REAL TIME INPUT PORT 1      --- MODEM
6309                    ;*   SININDX2  index for last character REMOVED from buffer
6310                    ;*   SIN_BUF2  base location of MODEM input buffer
6311                    ;*   SINCNT2   equate of MODEM input buffer size
6312
6313
6314                    ;REAL TIME INPUT PORT 2      MODEM computer
6315
6316        00:FAB3             IRQAR2 EQU *   ;QUEUE UP SERIAL BYTE
6317  00:FAB3  48                       PHA
6318  00:FAB4  5A                       PHY
6319  00:FAB5  DA                       PHX
6320  00:FAB6  08                          PHP
6321  00:FAB7  0B                       PHD          ;SAVE DIRECT REG
6322  00:FAB8  8B                       PHB
6323  00:FAB9  F4 00 00                 PEA #$0000
6324  00:FABC  2B                       PLD          ;SET DIRECT PAGE REG TO PAGE 0
6325  00:FABD  F4 00 00                 PEA #$0000     ;SET DATA BANK TO 0
6326  00:FAC0  AB                       PLB
6327  00:FAC1  AB                       PLB
6328  00:FAC2  E2 20                    SEP #M8       ;SET Acc SHORT
6329  00:FAC4  C2 10                       REP #X8         ;SET X & Y LONG
6330                    .LONGA OFF
6331                    .LONGI ON
6332
6333  00:FAC6  A9 10                       LDA #$10       ;CLEAR INTERRUPT
6334  00:FAC8  8D 48 DF                 STA UIFR
```

```
6335
6336  00:FACB  AD 75 DF          LDA ARTD2     ;GET DATA CHAR
6337  00:FACE  48               PHA
6338  00:FACF  A5 42            LDA SFLAG2    ;CHECK FOR XON/XOFF OPERATION
6339  00:FAD1  89 04            BIT #XONOFLG
6340  00:FAD3  F0 0A               BEQ RECV2_R8       ;NOT XON/XOFF PROTOCALL
6341  00:FAD5  68                  PLA
6342  00:FAD6  48                  PHA
6343  00:FAD7  C9 13               CMP #XOFF
6344  00:FAD9  F0 53            BEQ RECV2_XOFF
6345  00:FADB  C9 11            CMP #XON
6346  00:FADD  F0 56            BEQ RECV2_XON
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
6347  00:FADF  A4 10        RECV2_R8 LDY SININDX2    ;CHECK FOR BUFFER FULL
6348  00:FAE1  C8                 INY
6349  00:FAE2  C4 16               CPY SINCNT2     ;BUFFER SIZE
6350  00:FAE4  D0 03                  BNE RECV2_R9
6351  00:FAE6  A0 00 00               LDY #0                  ;OVER RAN END OF BUFFER
6352  00:FAE9  C4 12        RECV2_R9 CPY SINEND2
6353  00:FAEB  D0 10               BNE RECV2_R11
6354
6355  00:FAED  A9 40                  LDA #$40        ;Just overran the input buffer
6356  00:FAEF  04 4A                  TSB STATUS_S2       ;SET the STATUS FLAG
6357  00:FAF1  A6 12               LDX SINEND2     ;THROW AWAY 1 CHAR..THE
6358  00:FAF3  E8                  INX                     ;OUTPUT IS NOT KEEPING UP!
6359  00:FAF4  E4 16               CPX SINCNT2
6360  00:FAF6  D0 03               BNE RECV2_R10
6361  00:FAF8  A0 00 00            LDY #0              ;OVER-RAN END OF BUFFER
6362  00:FAFB  86 12        RECV2_R10 STX SINEND2   ;SAVE OUTPUT POINTER
6363  00:FAFD  84 10        RECV2_R11 STY SININDX2   ;SAVE INPUT PTR
6364  00:FAFF  68                     PLA
6365  00:FB00  91 14                  STA (SIN_BUF2),Y ;STORE DATA
6366
6367  00:FB02  A9 01                  LDA #SFLG
6368  00:FB04  04 42                  TSB SFLAG2  ;SET CHAR READY FLAG
6369
6370  00:FB06  C8                     INY             ;SEE IF BUFFER NEARING FULL
6371  00:FB07  C8                  INY
6372  00:FB08  C8                     INY
6373  00:FB09  C8                  INY
6374  00:FB0A  C8                     INY
6375  00:FB0B  C4 16                  CPY SINCNT2         ;OUTPUT PTR
6376  00:FB0D  90 03                  BCC RECV2_R12
6377  00:FB0F  38                     SEC
6378  00:FB10  E5 16               SBC SINCNT2
6379  00:FB12  C4 12        RECV2_R12 CPY SINEND2
6380  00:FB14  F0 03                  BEQ RECV2_R14       ;OK  NOTE: It is not posible
6381  00:FB16  82 9C 02               BRL REC_DONE   ;to jump past the output ptr
6382                                        ;because we check after each
6383                                        ;char inputted.
6384
6385
6386  00:FB19  A5 42        RECV2_R14 LDA SFLAG2    ;CHECK PROTOCAL
6387  00:FB1B  89 04                  BIT #XONOFLG
6388  00:FB1D  F0 23                  BEQ RECV2_DTR
6389  00:FB1F  A9 10                  LDA #LASTXONOF
6390  00:FB21  14 42                  TRB SFLAG2
6391  00:FB23  D0 03                  BNE RECV2_R15
```

```
6392  00:FB25  82 8D 02              BRL REC_DONE       ;XOFF ALREADY SENT
6393
6394  00:FB28  A9 40          RECV2_R15 LDA #SXOFFLG
6395  00:FB2A  04 42                TSB SFLAG2
6396  00:FB2C  80 0C                BRA REC2_DONET
6397
6398  00:FB2E  68             RECV2_XOFF PLA             ;ADJUST STACK PTR
6399  00:FB2F  A9 08                LDA #SNDOVF
6400  00:FB31  04 42                TSB SFLAG2  ;MAKE INPUT STOP
6401  00:FB33  80 05                BRA REC2_DONET
6402
6403
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
6404  00:FB35  68             RECV2_XON PLA
6405  00:FB36  A9 08                 LDA #SNDOVF
6406  00:FB38  14 42                 TRB SFLAG2  ;START UP INPUT AGAIN
6407  00:FB3A  A9 03           REC2_DONET LDA #3
6408  00:FB3C  0C 74 DF              TSB ACSR2    ;START UP OUTPUT
6409  00:FB3F  82 73 02             BRL REC_DONE
6410
6411
6412  00:FB42  A9 10           RECV2_DTR LDA #$10
6413  00:FB44  0C 21 DF              TSB PD5        ;SET DTR HI
6414  00:FB47  82 6B 02              BRL REC_DONE
6415
6416
6417                          .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
6418
6419
6420                    ;* Routine:  IRQAT2    REAL-TIME MODEM OUTPUT
6421                    ;*
6422                    ;* This routine removes charaters from the MODEM
6423                    ;*   output buffer and sends them out PORT 2.
6424                    ;*   XON/XOFF protocal is used to control data flow.
6425                    ;*
6426                    ;* Var Used: SFLAG2,SOUTINDX2,SOUTEND2,SOUTCNT2
6427                    ;*
6428                    ;* Returned Reg: NONE
6429                    ;*
6430                    ;* Important Variables:
6431                    ;*   SOUTEND2   index for last character STORED in buffer
6432                    ;*   SOUTINDX2  index for last character REMOVED from buffer
6433                    ;*   SOUT_BUF2  base location of MODEM output buffer
6434                    ;*   SOUTCNT2   equate of MODEM output buffer size
6435                    ;*
6436                    ;*
6437                    ;* Returned Reg: NONE
6438                    ;*
6439                                    ;CALLED BY IRQ ROUTINE
6440        00:FB4A             IRQAT2 EQU *   ;DEQUEUE SERIAL BYTE
6441  00:FB4A  48                       PHA          ;FROM OUTPUT BUFFER
6442  00:FB4B  5A                       PHY          ;SEE OUTCH_PORTX ROUTINE
6443  00:FB4C  DA                       PHX
6444  00:FB4D  08                          PHP
6445  00:FB4E  0B                       PHD          ;SAVE DIRECT REG
6446  00:FB4F  8B                       PHB
6447  00:FB50  F4 00 00                 PEA #$0000
6448  00:FB53  2B                       PLD          ;SET DIRECT PAGE REG TO PAGE 0
6449  00:FB54  F4 00 00                 PEA #$0000     ;SET DATA BANK TO 0
6450  00:FB57  AB                       PLB
6451  00:FB58  AB                       PLB
6452  00:FB59  E2 20                    SEP #M8       ;SET Acc SHORT
6453  00:FB5B  C2 10                       REP #X8         ;SET X & Y LONG
6454                    .LONGA OFF
6455                    .LONGI ON
6456
6457  00:FB5D  A9 20                       LDA #$20       ;CLEAR INTERRUPT
6458  00:FB5F  8D 48 DF                 STA UIFR
6459
6460  00:FB62  A5 42                    LDA SFLAG2     ;CK IF WE ARE OVERFLOWING
6461  00:FB64  29 C0                    AND #SXOFFLG+SXONFLG
6462  00:FB66  F0 1C                    BEQ TRAN2_0A       ;QUICK CK FOR CNTRL FLGS
```

```
6463  00:FB68  C9 C0              CMP #SXOFFLG+SXONFLG
6464  00:FB6A  F0 14              BEQ TRAN2_ERR
6465  00:FB6C  89 40              BIT #SXOFFLG
6466  00:FB6E  D0 08              BNE TRAN2_XOFF    ;SEND XOFF
6467
6468        00:FB70           TRAN2_XON EQU *
6469  00:FB70  A9 80              LDA #SXONFLG
6470  00:FB72  14 42              TRB SFLAG2
6471  00:FB74  A9 11              LDA #XON
6472  00:FB76  80 43              BRA TRAN2_1
6473
6474
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
6475        00:FB78             TRAN2_XOFF  EQU *
6476
6477  00:FB78  A9 50                LDA #SXOFFLG+LASTXONOF
6478  00:FB7A  14 42                TRB SFLAG2
6479  00:FB7C  A9 13                LDA #XOFF
6480  00:FB7E  80 3B                BRA TRAN2_1
6481
6482        00:FB80             TRAN2_ERR EQU *
6483
6484  00:FB80  A9 C0                LDA #SXONFLG+SXOFFLG
6485  00:FB82  14 42                TRB SFLAG2
6486
6487        00:FB84             TRAN2_0A EQU *
6488  00:FB84  AD 74 DF             LDA ACSR2
6489  00:FB87  29 02                AND #DISCH        ;XMIT IRQ ON
6490  00:FB89  F0 0D                BEQ TRAN2_2B
6491
6492  00:FB8B  A5 42          TRAN2_1B LDA SFLAG2
6493  00:FB8D  89 04                BIT #XONOFLG
6494  00:FB8F  D0 18                BNE TRAN2_1A      ;XON/OFF PROTOCALL
6495                                   ;DO WE HAVE HRDW HS & DSR2 IS LOW
6496  00:FB91  AD 21 DF             LDA PD5          ;DSR2 = NOT AVAILABLE
6497  00:FB94  29 20                AND #Bit5        ;DSR* ON PD5-5
6498  00:FB96  D0 4C                BNE TRAN2_3A
6499
6500  00:FB98  A4 30          TRAN2_2B LDY SOUTINDX2    ;IS BUFFER NOW EMPTY
6501  00:FB9A  C4 32                CPY SOUTEND2
6502  00:FB9C  D0 11                BNE TRAN2_3       ;DATA BUFFER NOT EMPTY
6503  00:FB9E  AD 74 DF             LDA ACSR2        ;INITIATE SERIAL XMIT SHUTDOWN
6504  00:FBA1  29 02                AND #DISCH        ;HAVE WE INITIATED SHUTDOWN?
6505  00:FBA3  D0 3F                BNE TRAN2_3A      ;YES
6506  00:FBA5  A9 00                LDA #00          ;NO..LETS START
6507  00:FBA7  80 12                BRA TRAN2_1       ;FILL WITH NULL CHAR
6508
6509
6510  00:FBA9  89 08          TRAN2_1A BIT #SNDOVF      ;ARE WE IN XOFF STOP MODE?
6511  00:FBAB  F0 EB                BEQ TRAN2_2B      ;NO
6512  00:FBAD  80 35                BRA TRAN2_3A      ;YES SHUTDOWN AND EXIT
6513
6514        00:FBAF             TRAN2_3 EQU *            ;SEND CHAR FROM BUFFER
6515  00:FBAF  C8                   INY
6516  00:FBB0  C4 36                CPY SOUTCNT2     ;CHECK FOR POINTER ROLL-OVER
6517
6518  00:FBB2  90 03                BCC TRAN2_0       ;OK
6519  00:FBB4  A0 00 00             LDY #0           ;ROLL AROUND TO BUFFER START
```

```
6520  00:FBB7  84 30          TRAN2_0  STY SOUTINDX2
6521
6522  00:FBB9  B1 34               LDA (SOUTBUF2),Y    ;GET DATA FROM QUEUE
6523        00:FBBB          TRAN2_1 EQU *
6524  00:FBBB  8D 75 DF            STA ARTD2        ;SEND DATA
6525
6526  00:FBBE  A5 42               LDA SFLAG2       ;CHECK PROTOCALL
6527  00:FBC0  89 04               BIT #XONOFLG
6528  00:FBC2  D0 07               BNE TRAN2_2      ;XON/XOFF
6529
6530  00:FBC4  AD 21 DF            LDA PD5          ;DTR PROTOCALL
6531  00:FBC7  29 20               AND #Bit5        ;CHECK DSR LEAD FOR DTR IN
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
6532  00:FBC9  D0 19          BNE TRAN2_3A     ;P55 HI   --STOP!
6533
6534  00:FBCB  A5 42      TRAN2_2  LDA SFLAG2
6535  00:FBCD  89 08          BIT #SNDOVF
6536  00:FBCF  D0 13          BNE TRAN2_3A     ;WE'RE THROUGH  DISCHARGE IS
DONE
6537
6538  00:FBD1  A4 30          LDY SOUTINDX2    ;IS BUFFER NOW EMPTY
6539  00:FBD3  C4 32          CPY SOUTEND2
6540  00:FBD5  F0 1A          BEQ TRAN2_3B     ;START SHUTDOWN
6541  00:FBD7  A9 02          LDA #DISCH       ;TURN OFF SHUTDOWN MODE
6542  00:FBD9  1C 74 DF       TRB ACSR2        ;IN CASE IT WAS ON
6543  00:FBDC  A9 01          LDA #SON
6544  00:FBDE  0C 74 DF       TSB ACSR2
6545  00:FBE1  82 D1 01       BRL TRANS_DONE   ;EXIT
6546
6547
6548       00:FBE4       TRAN2_3A EQU *          ;*** TURN OFF UART ****
6549  00:FBE4  A9 20          LDA #Bit5        ;set the output high to prevent falsing
6550  00:FBE6  0C 22 DF       TSB PD6          ;when we shut it off.
6551  00:FBE9  A9 03          LDA #SON+DISCH
6552  00:FBEB  1C 74 DF       TRB ACSR2
6553  00:FBEE  82 C4 01       BRL TRANS_DONE
6554
6555       00:FBF1       TRAN2_3B EQU *          ;** START SHUTDOWN MODE **
6556  00:FBF1  A9 03          LDA #SON+DISCH     ;TURN ON SERIAL CLR INTR.
6557  00:FBF3  0C 74 DF       TSB ACSR2
6558  00:FBF6  82 BC 01       BRL TRANS_DONE
6559
6560                      .PAGE
```

```
      'MENSCH COMPUTER ROM SOFTWARE'
      'ROM Serial I/O Routines   UARTs'

6561                          .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
6562
6563                    ;* Routine:  GET_BYTE_FROM_PC
6564                    ;*
6565                    ;* This routine returns a character in the "A" reg
6566                    ;*  from the HOST input buffer. If the buffer
6567                    ;*  is empty, a null char is returned and the "C"
6568                    ;*  flag is set.
6569                    ;*
6570                    ;* This routine MUST be called using a JSL command!
6571                    ;*
6572                    ;* Routines Called: OUTCH3
6573                    ;*
6574                    ;* Returned Reg: Acc
6575                    ;* Returned Flags   If c is set no DATA is returned.
6576                    ;*   Instead an error code is returned in the A Reg.
6577                    ;*   ERROR CODES:
6578                    ;*   $00 on Return indicates NO DATA available.
6579                    ;*   $80 on Return indicates ^C or ESCape received.
6580                    ;*
6581                    ;* Important Variables:
6582                    ;*   SINEND2  index for last character removed from buffer
6583                    ;*   SININDX2  index for last character placed in buffer
6584                    ;*   SIN_BUF2  base location of HOST input buffer
6585                    ;*   SINCNT2   HOST input buffer size
6586
6587
6588
6589        00:FBF9             GET_BYTE_FROM_PC   EQU *
6590
6591
6592  00:FBF9  5A                   PHY
6593  00:FBFA  DA                   PHX
6594  00:FBFB  08                      PHP
6595  00:FBFC  0B                   PHD          ;SAVE DIRECT REG
6596  00:FBFD  8B                   PHB
6597  00:FBFE  F4 00 00             PEA #$0000
6598  00:FC01  2B                   PLD          ;SET DIRECT PAGE REG TO PAGE 0
6599  00:FC02  F4 00 00             PEA #$0000     ;SET DATA BANK TO 0
6600  00:FC05  AB                   PLB
6601  00:FC06  AB                   PLB
6602  00:FC07  E2 20                SEP #M8       ;SET Acc SHORT
6603  00:FC09  C2 10                   REP #X8         ;SET X & Y LONG
6604                     .LONGA OFF
6605                     .LONGI ON
6606
```

```
6607  00:FC0B  18                CLC             ;CLEAR C
6608  00:FC0C  A5 43           LDA SFLAG3      ;GET SERIAL BYTE
6609  00:FC0E  29 01           AND #SFLG       ;FROM INPUT QUEUE
6610  00:FC10  D0 04           BNE P3_GETSD5
6611
6612  00:FC12  64 47           STZ SDATA_SI3   ;No data RETURN A NULL
6613  00:FC14  80 56           BRA P3_RD_CH1
6614
6615  00:FC16  78           P3_GETSD5 SEI          ;PUT THERE BY RECSBYTE
6616  00:FC17  A4 1A           LDY SINEND3     ;CK IF CURRENT QUEUE POS
6617  00:FC19  C8              INY             ;POINT TO NXT DATA
6618  00:FC1A  C4 1E           CPY SINCNT3     ;DO WE WRAP
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
6619  00:FC1C  90 03              BCC P3_GETSD4
6620  00:FC1E  A0 00 00           LDY #0          ;WE WRAPPED
6621
6622        00:FC21           P3_GETSD4 EQU *
6623  00:FC21  84 1A              STY SINEND3
6624  00:FC23  B1 1C              LDA (SIN_BUF3),Y  ;GET DATA
6625  00:FC25  85 47              STA SDATA_SI3
6626  00:FC27  58                 CLI
6627  00:FC28  C4 18              CPY SININDX3     ;IS SAME AS END OF QUEUE
6628  00:FC2A  D0 25              BNE P3_GETSD3
6629
6630  00:FC2C  A5 43              LDA SFLAG3       ;CK IF XON/XOFF
6631  00:FC2E  89 04              BIT #XONOFLG     ;OR HARDWARE HS
6632  00:FC30  F0 16              BEQ P3_GETSD1
6633
6634  00:FC32  89 10              BIT #LASTXONOF   ;HAS XON ALREADY BEEN SENT?
6635  00:FC34  D0 17              BNE P3_GETSD2
6636
6637  00:FC36  A9 90              LDA #SXONFLG+LASTXONOF
6638  00:FC38  04 43              TSB SFLAG3
6639
6640  00:FC3A  AD 76 DF           LDA ACSR3
6641  00:FC3D  89 01              BIT #SON
6642  00:FC3F  D0 0C              BNE P3_GETSD2
6643  00:FC41  A9 03              LDA #SON+DISCH
6644  00:FC43  0C 76 DF           TSB ACSR3
6645  00:FC46  80 05              BRA P3_GETSD2
6646
6647        00:FC48           P3_GETSD1 EQU *          ;HANDLE HARDWARE HS
6648  00:FC48  A9 80              LDA #$80        ;DTR LOW, OK FOR
6649  00:FC4A  1C 21 DF           TRB PD5         ;OTHER GUY TO SEND
6650        00:FC4D           P3_GETSD2 EQU *
6651  00:FC4D  A9 01              LDA #SFLG       ;NO MORE SERIAL CHARS
6652  00:FC4F  14 43              TRB SFLAG3
6653
6654  00:FC51  A5 43           P3_GETSD3 LDA SFLAG3      ;CHK IF ECHO
6655  00:FC53  29 20              AND #ECHOFF
6656  00:FC55  F0 06              BEQ P3_RD_CH0
6657  00:FC57  A5 47              LDA SDATA_SI3
6658  00:FC59  22 BC FD 00        JSL SEND_BYTE_TO_PC  ;ECHO BACK INPUT
6659
6660        00:FC5D           P3_RD_CH0 EQU *
6661
6662  00:FC5D  A2 08 07           LDX #POWER_DOWN_COUNT
6663  00:FC60  8E B8 DF           STX PD_TIMER
```

```
6664
6665  00:FC63  AB                PLB         ;RESTORE BANK
6666  00:FC64  2B                PLD         ;RESTORE DIRECT REG
6667  00:FC65  28                   PLP
6668  00:FC66  FA                PLX
6669  00:FC67  7A                PLY
6670  00:FC68  18                CLC
6671  00:FC69  A5 47             LDA SDATA_SI3
6672  00:FC6B  6B                RTL
6673
6674        00:FC6C           P3_RD_CH1 EQU *
6675  00:FC6C  AB                PLB         ;RESTORE BANK
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
6676  00:FC6D  2B             PLD         ;RESTORE DIRECT REG
6677  00:FC6E  28               PLP
6678  00:FC6F  FA             PLX
6679  00:FC70  7A             PLY
6680  00:FC71  38             SEC         ;NO DATA - return a NULL
6681  00:FC72  A9 00             LDA #0       ;with C flag set
6682  00:FC74  6B             RTL
6683
6684
6685
6686
6687                        .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
6688
6689
6690                    ;REAL TIME INPUT PORT 3   --- PC
6691
6692
6693                    ;* Routine:  IRQAR3
6694                    ;*
6695                    ;* This routine is called by the interrrupt vector
6696                    ;*   and puts characters from the printer into the
6697                    ;*   PC input buffer.  XON/XOFF protocal is used
6698                    ;*   to control data flow through the PC.
6699                    ;*
6700                    ;*
6701                    ;* Var Used: SFLAG3,SOUTINDX3,SOUTEND3,SOUTCNT3
6702                    ;*
6703                    ;* Returned Reg: NONE
6704                    ;*
6705                    ;* Important Variables:
6706                    ;*   SINEND3 index for last character stored in buffer
6707                    ;*   SININDX3  index for last character REMOVED from buffer
6708                    ;*   SIN_BUF3  base location of P.C. output buffer
6709                    ;*   SINCNT3   P.C. output buffer size
6710
6711
6712                    ;REAL TIME INPUT PORT 3     PC
6713
6714        00:FC75             IRQAR3 EQU *   ;QUEUE UP SERIAL BYTE
6715  00:FC75  48                       PHA
6716  00:FC76  5A                       PHY
6717  00:FC77  DA                       PHX
6718  00:FC78  08                          PHP
6719  00:FC79  0B                       PHD          ;SAVE DIRECT REG
6720  00:FC7A  8B                       PHB
6721  00:FC7B  F4 00 00                 PEA #$0000
6722  00:FC7E  2B                       PLD          ;SET DIRECT PAGE REG TO PAGE 0
6723  00:FC7F  F4 00 00                 PEA #$0000     ;SET DATA BANK TO 0
6724  00:FC82  AB                       PLB
6725  00:FC83  AB                       PLB
6726  00:FC84  E2 20                    SEP #M8       ;SET Acc SHORT
6727  00:FC86  C2 10                       REP #X8         ;SET X & Y LONG
6728                    .LONGA OFF
6729                    .LONGI ON
6730
6731  00:FC88  A9 40                       LDA #$40       ;CLEAR INTERRUPT
6732  00:FC8A  8D 48 DF                 STA UIFR
```

```
6733
6734  00:FC8D  AD 77 DF            LDA ARTD3      ;GET DATA CHAR
6735  00:FC90  48                  PHA
6736  00:FC91  A5 43               LDA SFLAG3     ;CHECK FOR XON/XOFF OPERATION
6737  00:FC93  89 04               BIT #XONOFLG
6738  00:FC95  F0 0A               BEQ RECV3_R8         ;NOT XON/XOFF PROTOCALL
6739  00:FC97  68                  PLA
6740  00:FC98  48                  PHA
6741  00:FC99  C9 13               CMP #XOFF
6742  00:FC9B  F0 52               BEQ RECV3_XOFF
6743  00:FC9D  C9 11               CMP #XON
6744  00:FC9F  F0 55               BEQ RECV3_XON
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
6745  00:FCA1  A4 18        RECV3_R8 LDY SININDX3    ;CHECK FOR BUFFER FULL
6746  00:FCA3  C8                   INY
6747  00:FCA4  C4 1E                CPY SINCNT3     ;BUFFER SIZE
6748  00:FCA6  D0 03                   BNE RECV3_R9
6749  00:FCA8  A0 00 00                LDY #0                ;OVER RAN END OF BUFFER
6750  00:FCAB  C4 1A        RECV3_R9 CPY SINEND3
6751  00:FCAD  D0 10                BNE RECV3_R11
6752
6753  00:FCAF  A9 40                   LDA #$40       ;Just overran the input buffer
6754  00:FCB1  04 4B                   TSB STATUS_S3       ;SET the STATUS FLAG
6755  00:FCB3  A6 1A                LDX SINEND3    ;THROW AWAY 1 CHAR..THE
6756  00:FCB5  E8                   INX                 ;OUTPUT IS NOT KEEPING UP!
6757  00:FCB6  E4 1E                CPX SINCNT3
6758  00:FCB8  D0 03                BNE RECV3_R10
6759  00:FCBA  A2 00 00             LDX #0              ;OVER-RAN END OF BUFFER
6760  00:FCBD  86 1A        RECV3_R10 STX SINEND3   ;SAVE OUTPUT POINTER
6761  00:FCBF  84 18        RECV3_R11 STY SININDX3   ;SAVE INPUT PTR
6762  00:FCC1  68                   PLA
6763  00:FCC2  91 1C                STA (SIN_BUF3),Y ;STORE DATA
6764
6765  00:FCC4  A9 01                LDA #SFLG
6766  00:FCC6  04 43                TSB SFLAG3  ;SET CHAR READY FLAG
6767
6768  00:FCC8  C8                   INY              ;SEE IF BUFFER NEARING FULL
6769  00:FCC9  C8                INY
6770  00:FCCA  C8                   INY
6771  00:FCCB  C8                INY
6772  00:FCCC  C8                   INY
6773  00:FCCD  C4 1E                CPY SINCNT3           ;OUTPUT PTR
6774  00:FCCF  90 03                BCC RECV3_R12
6775  00:FCD1  38                   SEC
6776  00:FCD2  E5 1E                SBC SINCNT3
6777  00:FCD4  CC DA FC     RECV3_R12 CPY RECV3_R14           ;OK  NOTE: It is not posible
6778  00:FCD7  82 DB 00             BRL REC_DONE   ;to jump past the output ptr
6779                                      ;because we check after each
6780                                      ;char inputted.
6781
6782
6783  00:FCDA  A5 43        RECV3_R14 LDA SFLAG3    ;CHECK PROTOCAL
6784  00:FCDC  89 04                BIT #XONOFLG
6785  00:FCDE  F0 23                BEQ RECV3_DTR
6786  00:FCE0  A9 10                LDA #LASTXONOF
6787  00:FCE2  14 43                TRB SFLAG3
6788  00:FCE4  D0 03                BNE RECV3_R15
6789  00:FCE6  82 CC 00             BRL REC_DONE       ;XOFF ALREADY SENT
```

```
6790
6791  00:FCE9  A9 40          RECV3_R15 LDA #SXOFFLG
6792  00:FCEB  04 43                TSB SFLAG3
6793  00:FCED  80 0C                   BRA RECV3_DONET
6794
6795  00:FCEF  68             RECV3_XOFF PLA              ;ADJUST STACK PTR
6796  00:FCF0  A9 08                LDA #SNDOVF
6797  00:FCF2  04 43                   TSB SFLAG3  ;MAKE INPUT STOP
6798  00:FCF4  80 05                   BRA RECV3_DONET
6799
6800
6801  00:FCF6  68             RECV3_XON PLA
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
6802  00:FCF7  A9 08                LDA #SNDOVF
6803  00:FCF9  14 43                TRB SFLAG3  ;START UP INPUT AGAIN
6804  00:FCFB  A9 03          RECV3_DONET LDA #3
6805  00:FCFD  0C 76 DF              TSB ACSR3    ;START UP OUTPUT
6806  00:FD00  82 B2 00            BRL REC_DONE
6807
6808
6809  00:FD03  A9 80          RECV3_DTR LDA #$80
6810  00:FD05  0C 21 DF              TSB PD5        ;SET DTR HI
6811
6812                             ;CALLED BY IRQ ROUTINE
6813
6814
6815                        .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
6816
6817                    ;* Routine:  IRQAT3    REAL-TIME PC OUTPUT
6818                    ;*
6819                    ;* This routine removes charaters from the PC
6820                    ;*   output buffer and sends them out PORT 3.
6821                    ;*   XON/XOFF protocal is used to control data flow.
6822                    ;*
6823                    ;* Var Used: SFLAG3,SOUTINDX3,SOUTEND3,SOUTCNT3
6824                    ;*
6825                    ;* Returned Reg: NONE
6826                    ;*
6827                    ;* Important Variables:
6828                    ;*   SOUTEND2   index for last character STORED in buffer
6829                    ;*   SOUTINDX2  index for last character REMOVED from buffer
6830                    ;*   SOUT_BUF2  base location of HOST output buffer
6831                    ;*   SOUTCNT2   HOST output buffer size
6832                    ;*
6833                    ;*
6834                    ;* Returned Reg: NONE
6835                    ;*
6836 00:FD08
6837
6838                    ;REAL TIME OUTPUT PORT 3  PC
6839
6840                                ;CALLED BY IRQ ROUTINE
6841
6842                                ;CALLED BY IRQ ROUTINE
6843       00:FD08              IRQAT3 EQU *   ;DEQUEUE SERIAL BYTE
6844 00:FD08  48                       PHA          ;FROM OUTPUT BUFFER
6845 00:FD09  5A                       PHY          ;SEE OUTCH_PORTX ROUTINE
6846 00:FD0A  DA                       PHX
6847 00:FD0B  08                          PHP
6848 00:FD0C  0B                       PHD          ;SAVE DIRECT REG
6849 00:FD0D  8B                       PHB
6850 00:FD0E  F4 00 00                 PEA #$0000
6851 00:FD11  2B                       PLD          ;SET DIRECT PAGE REG TO PAGE 0
6852 00:FD12  F4 00 00                 PEA #$0000     ;SET DATA BANK TO 0
6853 00:FD15  AB                       PLB
6854 00:FD16  AB                       PLB
6855 00:FD17  E2 20                    SEP #M8       ;SET Acc SHORT
6856 00:FD19  C2 10                       REP #X8         ;SET X & Y LONG
6857                    .LONGA OFF
6858                    .LONGI ON
6859
6860 00:FD1B  A9 80                    LDA #$80       ;CLEAR INTERRUPT
```

```
6861  00:FD1D  8D 48 DF          STA UIFR
6862
6863  00:FD20  A5 43             LDA  SFLAG3     ;CK IF WE ARE OVERFLOWING
6864  00:FD22  29 C0             AND #SXOFFLG+SXONFLG
6865  00:FD24  F0 1C             BEQ TRAN3_0A       ;QUICK CK FOR CNTRL FLGS
6866  00:FD26  C9 C0             CMP #SXOFFLG+SXONFLG
6867  00:FD28  F0 14             BEQ TRAN3_ERR
6868  00:FD2A  89 40             BIT #SXOFFLG
6869  00:FD2C  D0 08             BNE TRAN3_XOFF    ;SEND XOFF
6870
6871        00:FD2E          TRAN3_XON EQU *
6872  00:FD2E  A9 80             LDA #SXONFLG
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
6873  00:FD30  14 43            TRB SFLAG3
6874  00:FD32  A9 11            LDA #XON
6875  00:FD34  80 46            BRA TRAN3_1
6876
6877
6878        00:FD36         TRAN3_XOFF  EQU *
6879
6880  00:FD36  A9 50            LDA #SXOFFLG+LASTXONOF
6881  00:FD38  14 43            TRB SFLAG3
6882  00:FD3A  A9 13            LDA #XOFF
6883  00:FD3C  80 3E            BRA TRAN3_1
6884
6885        00:FD3E         TRAN3_ERR EQU *
6886
6887  00:FD3E  A9 C0            LDA #SXONFLG+SXOFFLG
6888  00:FD40  14 43            TRB SFLAG3
6889
6890        00:FD42         TRAN3_0A EQU *
6891  00:FD42  AD 76 DF         LDA ACSR3
6892  00:FD45  29 02            AND #DISCH      ;XMIT IRQ ON
6893  00:FD47  F0 0D            BEQ TRAN3_2B
6894
6895  00:FD49  A5 43         TRAN3_1B LDA SFLAG3
6896  00:FD4B  89 04            BIT #XONOFLG
6897  00:FD4D  D0 1B            BNE TRAN3_1A    ;XON/OFF PROTOCALL
6898                               ;DO WE HAVE HRDW HS & DSR2 IS LOW
6899  00:FD4F  AD 21 DF         LDA PD5       ;DSR2 = PD5-7
6900  00:FD52  29 80            AND #Bit7
6901  00:FD54  D0 26            BNE TRAN3_1
6902
6903  00:FD56  A4 38         TRAN3_2B LDY SOUTINDX3   ;IS BUFFER NOW EMPTY
6904  00:FD58  C4 3A            CPY SOUTEND3
6905  00:FD5A  D0 14            BNE TRAN3_3     ;DATA BUFFER NOT EMPTY
6906  00:FD5C  AD 76 DF         LDA ACSR3      ;INITIATE SERIAL XMIT SHUTDOWN
6907  00:FD5F  29 02            AND #DISCH      ;HAVE WE INITIATED SHUTDOWN?
6908  00:FD61  D0 41            BNE TRAN3_3A    ;YES
6909  00:FD63  A9 00            LDA #00       ;NO..LETS START
6910  00:FD65  8D 77 DF         STA ARTD3      ;FILL WITH NULL CHAR
6911  00:FD68  80 46            BRA TRAN3_3B    ;START SHUTDOWN & EXIT
6912
6913
6914  00:FD6A  89 08         TRAN3_1A BIT #SNDOVF     ;ARE WE IN XOFF STOP MODE?
6915  00:FD6C  F0 E8            BEQ TRAN3_2B    ;NO
6916  00:FD6E  80 34            BRA TRAN3_3A    ;YES SHUTDOWN AND EXIT
6917
```

```
6918          00:FD70             TRAN3_3 EQU *           ;SEND CHAR FROM BUFFER
6919  00:FD70  C8                        INY
6920  00:FD71  C4 3E                     CPY SOUTCNT3    ;CHECK FOR POINTER ROLL-OVER
6921
6922  00:FD73  90 03                     BCC TRAN3_0     ;OK
6923  00:FD75  A0 00 00                  LDY #0          ;ROLL AROUND TO BUFFER START
6924  00:FD78  84 38              TRAN3_0  STY SOUTINDX3
6925
6926  00:FD7A  B1 3C                     LDA (SOUTBUF3),Y    ;GET DATA FROM QUEUE
6927          00:FD7C             TRAN3_1 EQU *
6928  00:FD7C  8D 77 DF                  STA ARTD3       ;SEND DATA
6929
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
6930  00:FD7F  A5 43          LDA SFLAG3      ;CHECK PROTOCALL
6931  00:FD81  89 04          BIT #XONOFLG
6932  00:FD83  D0 07          BNE TRAN3_2     ;XON/XOFF
6933
6934  00:FD85  AD 21 DF       LDA PD5         ;DTR PROTOCALL
6935  00:FD88  29 80          AND #Bit7       ;CHECK DSR LEAD FOR DTR IN
6936  00:FD8A  D0 18          BNE TRAN3_3A    ;PD5-7 HI   --STOP!
6937
6938  00:FD8C  A5 43      TRAN3_2  LDA SFLAG3
6939  00:FD8E  89 08          BIT #SNDOVF
6940  00:FD90  D0 12          BNE TRAN3_3A    ;WE'RE THROUGH  DISCHARGE IS
DONE
6941
6942  00:FD92  A4 38          LDY SOUTINDX3   ;IS BUFFER NOW EMPTY
6943  00:FD94  C4 3A          CPY SOUTEND3
6944  00:FD96  F0 18          BEQ TRAN3_3B    ;START SHUTDOWN
6945  00:FD98  A9 02          LDA #DISCH      ;TURN OFF SHUTDOWN MODE
6946  00:FD9A  1C 76 DF       TRB ACSR3       ;IN CASE IT WAS ON
6947  00:FD9D  A9 01          LDA #SON
6948  00:FD9F  0C 76 DF       TSB ACSR3
6949  00:FDA2  80 11          BRA TRANS_DONE  ;EXIT
6950
6951
6952        00:FDA4       TRAN3_3A EQU *          ;*** TURN OFF UART ****
6953  00:FDA4  A9 80          LDA #Bit7       ;set the output high to prevent falsing
6954  00:FDA6  0C 22 DF       TSB PD6         ;when we shut it off.
6955  00:FDA9  A9 03          LDA #SON+DISCH
6956  00:FDAB  1C 76 DF       TRB ACSR3
6957  00:FDAE  80 05          BRA TRANS_DONE
6958
6959        00:FDB0       TRAN3_3B EQU *          ;** START SHUTDOWN MODE **
6960  00:FDB0  A9 03          LDA #SON+DISCH  ;TURN ON SERIAL CLR INTR.
6961  00:FDB2  0C 76 DF       TSB ACSR3
6962
6963
6964        00:FDB5       REC_DONE EQU *
6965
6966        00:FDB5       TRANS_DONE EQU *
6967  00:FDB5  AB             PLB         ;RESTORE BANK
6968  00:FDB6  2B             PLD         ;RESTORE DIRECT REG
6969  00:FDB7  28               PLP
6970  00:FDB8  FA             PLX
6971  00:FDB9  7A             PLY         ;IRQ IS DONE!
6972  00:FDBA  68             PLA
6973  00:FDBB  40             RTI
```

6974
6975 .PAGE

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
6976
6977
6978                    ;* Routine:  SEND_BYTE_TO_PC
6979                    ;*
6980                    ;* This routine buffers a character to be sent through
6981                    ;*   the PC.  The ACC must contain the character to
6982                    ;*   be transmitted.
6983                    ;*
6984                    ;* This routine MUST be called using a JSL command!
6985                    ;*
6986                    ;* Reg Used: ACC
6987                    ;* Var Used: SFLAG3,SOUTINDX3,SOUTEND3,SOUTCNT3
6988                    ;*
6989                    ;* Returned Reg: NONE
6990                    ;*
6991                    ;* Important Variables:
6992                    ;*   SOUTEND3   index for last character STORED in buffer
6993                    ;*   SOUTINDX3  index for last character REMOVED from buffer
6994                    ;*   SOUT_BUF3  base location of HOST output buffer
6995                    ;*   SOUTCNT3   HOST of keyboard output buffer size
6996
6997
6998
6999
7000        00:FDBC             SEND_BYTE_TO_PC   EQU *
7001
7002  00:FDBC  48                   PHA          ;FROM OUTPUT BUFFER
7003  00:FDBD  5A                   PHY
7004  00:FDBE  DA                   PHX
7005  00:FDBF  08                      PHP
7006  00:FDC0  0B                   PHD          ;SAVE DIRECT REG
7007  00:FDC1  8B                   PHB
7008  00:FDC2  F4 00 00             PEA #$0000
7009  00:FDC5  2B                   PLD          ;SET DIRECT PAGE REG TO PAGE 0
7010  00:FDC6  F4 00 00             PEA #$0000    ;SET DATA BANK TO 0
7011  00:FDC9  AB                   PLB
7012  00:FDCA  AB                   PLB
7013  00:FDCB  E2 20                SEP #M8       ;SET Acc SHORT
7014  00:FDCD  C2 10                   REP #X8         ;SET X & Y LONG
7015  00:FDCF  48                   PHA          ;DATA BYTE
7016
7017                    .LONGA OFF
7018                    .LONGI ON
7019
7020  00:FDD0  A2 08 07             LDX #POWER_DOWN_COUNT
```

```
7021  00:FDD3  8E B8 DF          STX PD_TIMER
7022
7023  00:FDD6  A4 3A        OUTCH31  LDY SOUTEND3   ;CK IF CURRENT QUEUE POS
7024  00:FDD8  C8                INY          ;POINT TO NXT DATA
7025  00:FDD9  C4 3E             CPY SOUTCNT3  ;DO WE WRAP
7026  00:FDDB  90 03             BCC OUTCH3D2   ;NO
7027  00:FDDD  A0 00 00          LDY #0        ;WE WRAPPED
7028        00:FDE0          OUTCH3D2 EQU *
7029  00:FDE0  C4 38             CPY SOUTINDX3  ;DID WE OVERRUN QUEUE
7030  00:FDE2  F0 30             BEQ OUTCH33    ;YES, So set C and return
7031
7032  00:FDE4  78                SEI
```

'MENSCH COMPUTER ROM SOFTWARE'
'ROM Serial I/O Routines UARTs'

```
7033
7034  00:FDE5  84 3A          STY SOUTEND3
7035  00:FDE7  68             PLA          ;GET DATA
7036  00:FDE8  91 3C          STA (SOUTBUF3),Y ;PUT DATA IN QUEUE
7037  00:FDEA  A5 43          LDA SFLAG3     ;CK IF HWHS OR SOFTWARE HS
7038  00:FDEC  89 04          BIT #XONOFLG
7039  00:FDEE  D0 12          BNE OUTCH3_A1  ;XON/OFF SW IS ON
7040  00:FDF0  AD 76 DF       LDA ACSR3      ;HARDWARE HANDSHAKE PROTOCALL
7041  00:FDF3  89 01          BIT #SON       ;IS SERIAL IRQ ON
7042  00:FDF5  F0 0F          BEQ OUTCH31A   ;NO
7043  00:FDF7  89 02          BIT #DISCH     ;ARE WE IN NORMAL SERIAL
7044  00:FDF9  F0 10          BEQ OUTCH32    ;MODE--YES
7045  00:FDFB  A9 02          LDA #DISCH     ;GOTO NORMAL SERIAL MODE
7046  00:FDFD  1C 76 DF       TRB ACSR3
7047  00:FE00  80 09          BRA OUTCH32
7048
7049       00:FE02       OUTCH3_A1 EQU *
7050  00:FE02  89 08          BIT #SNDOVF    ;CHK FOR SW HS
7051  00:FE04  D0 05          BNE OUTCH32    ;HAVE A XOFF SO DONT XMIT
7052
7053  00:FE06  A9 01       OUTCH31A LDA #SON      ;SERIAL IRQ SINGLE CHR MODE
7054  00:FE08  0C 76 DF       TSB ACSR3
7055  00:FE0B  58          OUTCH32  CLI
7056  00:FE0C  AB             PLB          ;RESTORE BANK
7057  00:FE0D  2B             PLD          ;RESTORE DIRECT REG
7058  00:FE0E  28                PLP
7059  00:FE0F  FA             PLX
7060  00:FE10  7A             PLY          ;RECEIVE IRQ DONE
7061  00:FE11  68             PLA
7062  00:FE12  18             CLC
7063  00:FE13  6B             RTL
7064
7065
7066  00:FE14  A9 80       OUTCH33       LDA #$80       ;Overran output buffer
7067  00:FE16  04 4B          TSB STATUS_S3 ;SET STATUS FLAG
7068  00:FE18  68             PLA          ;RESTORE STK ON CNTRL 'C'
7069  00:FE19  AB             PLB          ;RESTORE BANK
7070  00:FE1A  2B             PLD          ;RESTORE DIRECT REG
7071  00:FE1B  28                PLP
7072  00:FE1C  FA             PLX
7073  00:FE1D  7A             PLY          ;RECEIVE IRQ DONE
7074  00:FE1E  68             PLA
7075  00:FE1F  38             SEC
7076  00:FE20  6B             RTL
7077
```

```
7078
7079                    .STTL 'SERIAL PORT INITIALIZATION'
7080                    .PAGE
```

'MENSCH COMPUTER ROM SOFTWARE'
'SERIAL PORT INITIALIZATION'

```
7081
7082                    ;* Routine:  SELECT_COMMON_BAUD
7083                    ;*
7084                    ;*   ENTER using a JSL Command
7085                    ;*
7086                    ;* This routine sets the Baud Rate for the ALL UARTS
7087                    ;*   except UART 2 (MODEM port).
7088                    ;*   Enter the routine with a table index value
7089                    ;*   (0-D) for the Baud Rate. Use an 8-Bit Areg.
7090                    ;*
7091                    ;* Reg Used: ACC,Y,X  All registers are saved!
7092                    ;*
7093                    ;* Returned Reg: NONE  a C flag indicats error.
7094                    ;*
7095                    ;     Baud Rates
7096                    ;
7097                    ;0    110 BAUD
7098                    ;1    150 BAUD
7099                    ;2    300 BAUD
7100                    ;3    600 BAUD
7101                    ;4   1200 BAUD
7102                    ;5   1800 BAUD
7103                    ;6   2400 BAUD
7104                    ;7   4800 BAUD
7105                    ;8   9600 BAUD
7106                    ;9  14400 BAUD
7107                    ;A  19200 BAUD
7108                    ;B  38400 BAUD
7109                    ;C  57600 BAUD
7110                    ;D 115000 BAUD
7111
7112
7113         00:FE21             SELECT_COMMON_BAUD_RATE  EQU *   ;A=BAUD
7114
7115  00:FE21
7116  00:FE21  48                    PHA
7117  00:FE22  5A                    PHY
7118  00:FE23  DA                    PHX
7119  00:FE24  08                       PHP
7120  00:FE25  0B                    PHD          ;SAVE DIRECT REG
7121  00:FE26  8B                    PHB
7122  00:FE27  F4 00 00              PEA #$0000
7123  00:FE2A  2B                    PLD          ;SET DIRECT PAGE REG TO PAGE 0
7124  00:FE2B  F4 00 00              PEA #$0000     ;SET DATA BANK TO 0
7125  00:FE2E  AB                    PLB
```

```
7126  00:FE2F  AB                   PLB
7127  00:FE30  E2 10                  SEP #X8         ;SET X & Y SHORT
7128                          .LONGA OFF
7129                          .LONGI OFF
7130
7131  00:FE32  78                   SEI          ;DISABLE ANY IRQ'S
7132
7133                    ;     BAUD RATE is in Areg
7134  00:FE33  C9 0C                CMP #$0C      ;IS ACC VALID 75-38400
7135  00:FE35  B0 34                BCS ACI_ERR
7136  00:FE37  0A                   ASL A         ;X2
7137  00:FE38  85 4C                STA STEMP_Sx
```

'MENSCH COMPUTER ROM SOFTWARE'
'SERIAL PORT INITIALIZATION'

```
7138  00:FE3A  AC B5 DF          LDY SPEED     ;MULT BY 11 FOR MAIN XTAL
7139  00:FE3D  B9 AA FE          LDA !BAUDOFFSET,Y
7140  00:FE40  18                CLC
7141  00:FE41  65 4C             ADC STEMP_Sx
7142  00:FE43  AA                TAX
7143  00:FE44  BD AF FE          LDA !ACIBAUD,X  ;SETUP BAUD RATE COUNTER
7144  00:FE47  8D 58 DF          STA T4LL
7145  00:FE4A  BD B0 FE          LDA !ACIBAUD+1,X
7146  00:FE4D  8D 69 DF          STA T4CH      ;LOADS THE LATCH & COUNTER
7147  00:FE50  A9 04             LDA #Bit2     ;8 BIT DATA
7148  00:FE52  0C 70 DF          TSB ACSR0
7149  00:FE55  0C 72 DF          TSB ACSR1
7150  00:FE58  0C 76 DF          TSB ACSR3
7151  00:FE5B  A9 10             LDA #Bit4
7152  00:FE5D  1C 46 DF          TRB TIER      ;NO TIMER IRQ
7153  00:FE60  0C 43 DF          TSB TER       ;ENABLE COUNTER
7154  00:FE63  AB                PLB          ;RESTORE BANK
7155  00:FE64  2B                PLD          ;RESTORE DIRECT REG
7156  00:FE65  28                   PLP
7157  00:FE66  18                CLC
7158
7159  00:FE67  FA             ACI_OUT  PLX
7160  00:FE68  7A                PLY          ;RECEIVE IRQ DONE
7161  00:FE69  68                PLA
7162  00:FE6A  6B                RTL
7163
7164
7165  00:FE6B  AB             ACI_ERR  PLB          ;RESTORE BANK
7166  00:FE6C  2B                PLD          ;RESTORE DIRECT REG
7167  00:FE6D  28                   PLP
7168  00:FE6E  38                SEC          ;Show ERROR!
7169  00:FE6F  80 F6             BRA ACI_OUT
7170
7171
7172
7173        00:FE71             SIOPORTS EQU *     ;GENERAL SETUP OF ALL UART PORTS
7174
7175
7176                     ;           ACSRx DEFINITIONS
7177                           ;BIT 0-XMIT PORT ENABLE
7178                           ;BIT 1-XMIT DISCHARGE IRQ
7179                           ;BIT 2-7/8 BIT DATA
7180                           ;BIT 3-PARITY ENABLE
7181                           ;BIT 4-ODD/EVEN PARITY
7182                           ;BIT 5-RECV ENABLE
```

```
 7183                           ;BIT 6-SOFTWARE SEMIPHORE
 7184                           ;BIT 7-RECV ERROR FLG
 7185
 7186  00:FE71  A9 FF                LDA #$FF      ;START WITH ALL OUTPUTS
 7187  00:FE73  8D 26 DF             STA PDD6
 7188
 7189                            ;SET TXD0,TXD1,TXD2,TXD3 TO DEFAULT MARK
(P61,P63,P65,P67)
 7190  00:FE76  0C 22 DF             TSB PD6       ;FORCE RXD INPUTS TO MARK (BUS
HOLDING DEVICES)
 7191  00:FE79
 7192                            ;SETUP DATA DIRECTION REG INPUT = 0, OUTPUT = 1
 7193  00:FE79  A9 55                LDA #$55
 7194  00:FE7B  1C 26 DF             TRB PDD6      ;SET RXD0,RXD1,RXD2,RXD3 AS INPUTs
(P60,P62,P64,P66)
```

'MENSCH COMPUTER ROM SOFTWARE'
'SERIAL PORT INITIALIZATION'

```
7195
7196                                    ;PD5 - B1,B3,B5 & B7  ARE INPUTS  [DSRs]
7197  00:FE7E  8D 25 DF                STA PDD5      ;PD5 - B0,B2,B4 & B6  ARE OUTPUTS [DTRs]
7198                                    ;MAKE  DTR0-3 LOW
7199  00:FE81  1C 21 DF                TRB PD5       ;TO ENABLE SERIAL DATA
7200
7201  00:FE84  A9 B0                   LDA #$B0      ;UARTs 0,1,3  will use TIMER #4
7202  00:FE86  0C 42 DF                TSB TCR
7203
7204  00:FE89  AD 77 DF                LDA ARTD3     ;CLEAR INPUT REGISTER
7205
7206                        ;    Start Timers
7207
7208  00:FE8C  A9 18                   LDA #T3FLG+T4FLG ;ENABLE TIMER 3 & 4
7209  00:FE8E  0C 43 DF                TSB TER
7210  00:FE91  60                      RTS
7211
7212
7213
7214                            .END
7215
7216
7217          00:FE92               DELASTBYTE EQU *
7218
7219          [01]                  .IFTRUE   DELASTBYTE.UGT.$00FEA0
7220                                .EXIT  "It won't fit in the ROM!!!!"
7221          [00]                  .ENDIF
7222
7223
7224                                .STTL 'TABLES AND CONSTANTS'
7225                                .page
```

'MENSCH COMPUTER ROM SOFTWARE'
'TABLES AND CONSTANTS'

```
7226  00:FE91                       INCLUDE TABLES.ASM
7227                      ; FILE;  TABLES.ASM--TIMER CONSTANTS
7228                      ; DATE:  11-22-1994
7229
7230  00:FEA0                       .org $00:FEA0
7231
7232        00:FEA0             T2_1MSEC_TBL EQU *
7233  00:FEA0  7300                 .WORD 1843200/16000   ;1.8432 MHZ
7234  00:FEA2  9900                 .WORD 2457600/16000   ;2.4576 MHZ
7235  00:FEA4  E600                 .WORD 3686400/16000   ;3.6864 MHZ
7236  00:FEA6  3301                 .WORD 4915200/16000   ;4.9152 MHZ
7237  00:FEA8  8001                 .WORD 6144000/16000   ;6.1440 MHZ
7238
7239
7240        00:FEAA             BAUDOFFSET EQU *
7241  00:FEAA  00                   .BYTE 00      ;1.8432 MHZ
7242  00:FEAB  1A                   .BYTE 26      ;2.4576 MHZ
7243  00:FEAC  34                   .BYTE 52      ;3.6864 MHZ
7244  00:FEAD  4E                   .BYTE 78      ;4.9125 MHZ
7245  00:FEAE  68                   .BYTE 104     ;6.1440 MHZ
7246
7247
7248
7249
7250        00:FEAF             ACIBAUD  EQU *
7251                              ;1.8432MHZ
7252  00:FEAF  1604               .WORD $0416   ;  110 BAUD
7253  00:FEB1  FF02               .WORD $02FF   ;  150 BAUD
7254  00:FEB3  7F01               .WORD $017F   ;  300 BAUD
7255  00:FEB5  BF00               .WORD $00BF   ;  600 BAUD
7256  00:FEB7  5F00               .WORD $005F   ; 1200 BAUD
7257  00:FEB9  3F00               .WORD $003F   ; 1800 BAUD
7258  00:FEBB  2F00               .WORD $002F   ; 2400 BAUD
7259  00:FEBD  1700               .WORD $0017   ; 4800 BAUD
7260  00:FEBF  0B00               .WORD $000B   ; 9600 BAUD
7261  00:FEC1  0800               .WORD $0008   ;14400 BAUD
7262  00:FEC3  0500               .WORD $0005   ;19200 BAUD
7263  00:FEC5  0200               .WORD $0002   ;38400 BAUD
7264  00:FEC7  0100               .WORD $0001   ;57600 BAUD
7265
7266                              ;2.4576MHZ
7267  00:FEC9  7305               .WORD $0573   ;  110 BAUD
7268  00:FECB  FF03               .WORD $03FF   ;  150 BAUD
7269  00:FECD  FF01               .WORD $01FF   ;  300 BAUD
7270  00:FECF  FF00               .WORD $00FF   ;  600 BAUD
```

```
7271  00:FED1  7F00                .WORD $007F    ; 1200 BAUD
7272  00:FED3  5400                .WORD $0054    ; 1800 BAUD
7273  00:FED5  3F00                .WORD $003F    ; 2400 BAUD
7274  00:FED7  1F00                .WORD $001F    ; 4800 BAUD
7275  00:FED9  0F00                .WORD $000F    ; 9600 BAUD
7276  00:FEDB  0B00                .WORD $000B    ;14400 BAUD
7277  00:FEDD  0700                .WORD $0007    ;19200 BAUD
7278  00:FEDF  0300                .WORD $0003    ;38400 BAUD
7279  00:FEE1  0200                .WORD $0002    ;57600 BAUD BAD WONT WORK AT
2.4576MHZ
7280
7281                               ;3.6864MHZ
7282  00:FEE3  2E08                .WORD $082E    ;  110 BAUD
```

'MENSCH COMPUTER ROM SOFTWARE'
'TABLES AND CONSTANTS'

```
7283  00:FEE5  FF05              .WORD $05FF    ;  150 BAUD
7284  00:FEE7  FF02              .WORD $02FF    ;  300 BAUD
7285  00:FEE9  7F01              .WORD $017F    ;  600 BAUD
7286  00:FEEB  BF00              .WORD $00BF    ; 1200 BAUD
7287  00:FEED  7F00              .WORD $007F    ; 1800 BAUD
7288  00:FEEF  5F00              .WORD $005F    ; 2400 BAUD
7289  00:FEF1  2F00              .WORD $002F    ; 4800 BAUD
7290  00:FEF3  1700              .WORD $0017    ; 9600 BAUD
7291  00:FEF5  1100              .WORD $0011    ;14400 BAUD
7292  00:FEF7  0B00              .WORD $000B    ;19200 BAUD
7293  00:FEF9  0500              .WORD $0005    ;38400 BAUD
7294  00:FEFB  0300              .WORD $0003    ;57600 BAUD
7295
7296                             ;4.9152MHZ
7297  00:FEFD  E80A              .WORD $0AE8    ;  110 BAUD
7298  00:FEFF  FF07              .WORD $07FF    ;  150 BAUD
7299  00:FF01  FF03              .WORD $03FF    ;  300 BAUD
7300  00:FF03  FF01              .WORD $01FF    ;  600 BAUD
7301  00:FF05  FF00              .WORD $00FF    ; 1200 BAUD
7302  00:FF07  AA00              .WORD $00AA    ; 1800 BAUD
7303  00:FF09  7F00              .WORD $007F    ; 2400 BAUD
7304  00:FF0B  3F00              .WORD $003F    ; 4800 BAUD
7305  00:FF0D  1F00              .WORD $001F    ; 9600 BAUD
7306  00:FF0F  1700              .WORD $0017    ;14400 BAUD  ; 14,629 ACTUAL
7307  00:FF11  0F00              .WORD $000F    ;19200 BAUD
7308  00:FF13  0700              .WORD $0007    ;38400 BAUD
7309  00:FF15  0400              .WORD $0004    ;57600 BAUD
7310
7311                             ;6.1440 MHZ
7312  00:FF17  A20D              .WORD $0DA2    ;  110 BAUD
7313  00:FF19  FF09              .WORD $09FF    ;  150 BAUD
7314  00:FF1B  FF04              .WORD $04FF    ;  300 BAUD
7315  00:FF1D  7F02              .WORD $027F    ;  600 BAUD
7316  00:FF1F  3F01              .WORD $013F    ; 1200 BAUD
7317  00:FF21  DF00              .WORD $00DF    ; 1800 BAUD
7318  00:FF23  9F00              .WORD $009F    ; 2400 BAUD
7319  00:FF25  4F00              .WORD $004F    ; 4800 BAUD
7320  00:FF27  2700              .WORD $0027    ; 9600 BAUD
7321  00:FF29  1D00              .WORD $001D    ;14400 BAUD  ; 14,222 ACTUAL
7322  00:FF2B  1300              .WORD $0013    ;19200 BAUD
7323  00:FF2D  0900              .WORD $0009    ;38400 BAUD
7324  00:FF2F  0600              .WORD $0006    ;57600 BAUD
7325
7326
7327                       ;
```

```
7328                    ;
7329                    ;TIME OF DAY MAX MIN TABLES
7330                    ;
7331                    ;
7332        00:FF31             MAXTTBL  EQU *
7333  00:FF31  08                   .BYTE 8       ;DAY OF WEEK
7334  00:FF32  0D                   .BYTE 13      ;MONTH
7335  00:FF33  20                   .BYTE 32      ;DAY
7336  00:FF34  64                   .BYTE 100     ;YR
7337  00:FF35  18                   .BYTE 24      ;HR
7338  00:FF36  3C                   .BYTE 60      ;MIN
7339  00:FF37  3C                   .BYTE 60      ;SEC
```

'MENSCH COMPUTER ROM SOFTWARE'
'TABLES AND CONSTANTS'

```
7340
7341        00:FF38             MINTTBL  EQU *
7342  00:FF38  01                       .BYTE 1       ;DAY OF WEEK
7343  00:FF39  01                       .BYTE 1       ;MONTH
7344  00:FF3A  01                       .BYTE 1       ;DAY
7345  00:FF3B  00                       .BYTE 0       ;YR
7346  00:FF3C  00                       .BYTE 0       ;HR
7347  00:FF3D  00                       .BYTE 0       ;MIN
7348  00:FF3E  00                       .BYTE 0       ;SEC
7349
7350
7351  00:FF3F  1F             LASTDY   .BYTE 31      ;JANUARY
7352  00:FF40  1C                       .BYTE 28      ;FEBRUARY-EXCEPT LEAP YR
7353  00:FF41  1F                       .BYTE 31      ;MARCH
7354  00:FF42  1E                       .BYTE 30      ;APRIL
7355  00:FF43  1F                       .BYTE 31      ;MAY
7356  00:FF44  1E                       .BYTE 30      ;JUNE
7357  00:FF45  1F                       .BYTE 31      ;JULY
7358  00:FF46  1F                       .BYTE 31      ;AUGUST
7359  00:FF47  1E                       .BYTE 30      ;SEPTEMBER
7360  00:FF48  1F                       .BYTE 31      ;OCTOBER
7361  00:FF49  1E                       .BYTE 30      ;NOVEMBER
7362  00:FF4A  1F                       .BYTE 31      ;DECEMBER
7363
7364        00:FF4B             DFLTS    EQU *
7365  00:FF4B  05                       .BYTE 5       ;DAY OF WEEK
7366
7367  00:FF4C  07                       .BYTE 07      ;MONTH
7368  00:FF4D  01                       .BYTE 01      ;DAY
7369  00:FF4E  5D                       .BYTE 93      ;YEAR
7370
7371  00:FF4F  0C                       .BYTE 12      ;HOUR
7372  00:FF50  00                       .BYTE 00      ;MINUTES
7373  00:FF51  00                       .BYTE 0       ;SEC
7374  00:FF52  00                       .BYTE 0       ;DAYLIGHT SAVING OFF
7375        00:FF53             DFLTSEND EQU *
7376
7377                      ;
7378                      ;  STTL 'CONVERSION TABLES
7379                      ;
7380        00:FF53             HEXTOPOS EQU *
7381  00:FF53  01 02 04 08              .BYTE $01,$02,$04,$08
7382  00:FF57  10 20 40 80              .BYTE $10,$20,$40,$80
7383
7384
```

```
7385          00:FF5B                 BINDECL  EQU *
7386  00:FF5B  00 01 02 03 04                    .BYTE $00,$01,$02,$03,$04,$05,$06,$07,$08,$09
          05 06 07 08 09
7387  00:FF65  10 11 12 13 14          .BYTE $10,$11,$12,$13,$14,$15
          15
7388
7389  00:FF6B  00 16 32 48 64            BINDECH  .BYTE $00,$16,$32,$48,$64,$80,$96
          80 96
7390
7391  00:FF72  00 0A 14 1E 28            DECBIN   .BYTE 00,10,20,30,40,50,60,70,80,90
          32 3C 46 50 5A
7392
```

```
    'MENSCH COMPUTER ROM SOFTWARE'
    'TABLES AND CONSTANTS'

7393
7394
7395        00:0004             ROMSPACE EQU $00:FF80-*  ;gives space left in the ROM BEFORE
TABLES
7396
7397                          .END
7398
7399
7400
7401                          .STTL 'IRQVCTRS.ASM--IRQ VECTOR EQUATES FOR WDC65C265'
7402                          .page
```

'MENSCH COMPUTER ROM SOFTWARE'
'IRQVCTRS.ASM--IRQ VECTOR EQUATES FOR WDC65C265'

```
7403  00:FF72                       INCLUDE IRQVTRS.ASM
7404
7405                    ; FILE: IRQVCTRS.ASM
7406                    ; DATE: 10-30-94
7407
7408
7409                    INT_VECTORS     SECTION
7410  00:FF80                           .ORG    00:FF80H
7411
7412                    ;    NATIVE MODE VECTORS
7413
7414  00:FF80  2801                 .WORD UNIRQT0  ; TIMER 0 INTERRUPT
7415  00:FF82  2401                 .WORD UNIRQT1  ; TIMER 1 INTERRUPT   TIME OF DAY
CLOCK
7416  00:FF84  2001                 .WORD UNIRQT2  ; TIMER 2 INTERRUPT   DOWN TIMERS
& UPTIMER (STOPWATCH)
7417  00:FF86  B0E0                 .WORD RESET    ; TIMER 3 INTERRUPT
7418  00:FF88  B0E0                 .WORD RESET    ; TIMER 4 INTERRUPT
7419  00:FF8A  B0E0                 .WORD RESET    ; TIMER 5 INTERRUPT
7420  00:FF8C  B0E0                 .WORD RESET    ; TIMER 6 INTERRUPT
7421  00:FF8E  1C01                 .WORD UNIRQT7  ; TIMER 7 INTERRUPT
7422  00:FF90  1801                 .WORD EDGEIRQS ; POSITIVE EDGE INTERRUPT ON P56
7423  00:FF92  1801                 .WORD EDGEIRQS ; NEGATIVE EDGE INTERRUPT ON P57
7424  00:FF94  1801                 .WORD EDGEIRQS ; POSITIVE EDGE INTERRUPT ON P60
7425  00:FF96  1801                 .WORD EDGEIRQS ; POSITIVE EDGE INTERRUPT ON P62
FOR PWM
7426  00:FF98  1801                 .WORD EDGEIRQS ; NEGATIVE EDGE INTERRUPT ON P64
7427  00:FF9A  1801                 .WORD EDGEIRQS ; NEGATIVE EDGE INTERRUPT ON P66
7428  00:FF9C  1401                 .WORD PIBIRQ   ; PARALLEL INTERFACE BUS (PIB)
INTERRUPT
7429  00:FF9E  0801                 .WORD UNIRQ    ; IRQ LEVEL INTERRUPT
7430  00:FFA0  85F8                 .WORD IRQAR0   ; UART0 RECEIVER INTERRUPT
7431  00:FFA2  F2F8                 .WORD IRQAT0   ; UART0 TRANSMITTER INTERRUPT
7432  00:FFA4  6CF9                 .WORD IRQAR1   ; UART1 RECEIVER INTERRUPT
7433  00:FFA6  04FA                 .WORD IRQAT1   ; UART1 TRANSMITTER INTERRUPT
7434  00:FFA8  B3FA                 .WORD IRQAR2   ; UART2 RECEIVER INTERRUPT
7435  00:FFAA  4AFB                 .WORD IRQAT2   ; UART2 TRANSMITTER INTERRUPT
7436  00:FFAC  75FC                 .WORD IRQAR3   ; UART3 RECEIVER INTERRUPT
7437  00:FFAE  08FD                 .WORD IRQAT3   ; UART3 TRANSMITTER INTERRUPT
7438  00:FFB0  B0E0                 .WORD RESET    ; RESERVED INTERRUPT
7439  00:FFB2  B0E0                 .WORD RESET    ; RESERVED
7440  00:FFB4  0C01                 .WORD COPIRQ   ; COPROCESSOR INTERRUPT
7441  00:FFB6  0001                 .WORD UBRK     ; BRK - NATIVE SOFTWARE INTERRUPT
7442  00:FFB8  1001                 .WORD IABORT   ; ABORT INTERRUPT
7443  00:FFBA  0401                 .WORD UNMI     ; NMI - NONMASKABLE INTERRUPT JMP
```

```
INTO MONITOR CMD PARSER
 7444  00:FFBC  B0E0                .WORD RESET    ; RESERVED INTERRUPT
 7445  00:FFBE  B0E0                .WORD RESET    ; RESERVED INTERRUPT
 7446
 7447
 7448
 7449                      ;EMULATION MODE VECTOR TABLE
 7450
 7451  00:FFC0  B0E0                .WORD RESET    ; TIMER 0 INTERRUPT
 7452  00:FFC2  B0E0                .WORD RESET    ; TIMER 1 INTERRUPT  TIME OF DAY
CLOCK
 7453  00:FFC4  B0E0                .WORD RESET    ; TIMER 2 INTERRUPT  DOWN TIMERS &
UPTIMER (STOPWATCH)
 7454  00:FFC6  B0E0                .WORD RESET    ; TIMER 3 INTERRUPT
 7455  00:FFC8  B0E0                .WORD RESET    ; TIMER 4 INTERRUPT
 7456  00:FFCA  B0E0                .WORD RESET    ; TIMER 5 INTERRUPT
 7457  00:FFCC  B0E0                .WORD RESET    ; TIMER 6 INTERRUPT
 7458  00:FFCE  B0E0                .WORD RESET    ; TIMER 7 INTERRUPT
 7459  00:FFD0  B0E0                .WORD RESET    ; POSITIVE EDGE INTERRUPT ON P56
```

'MENSCH COMPUTER ROM SOFTWARE'
'IRQVCTRS.ASM--IRQ VECTOR EQUATES FOR WDC65C265'

```
 7460  00:FFD2  B0E0                .WORD RESET   ; NEGATIVE EDGE INTERRUPT ON P57
 7461  00:FFD4  B0E0                .WORD RESET   ; POSITIVE EDGE INTERRUPT ON P60
 7462  00:FFD6  B0E0                .WORD RESET   ; POSITIVE EDGE INTERRUPT ON P62 FOR
PWM
 7463  00:FFD8  B0E0                .WORD RESET   ; NEGATIVE EDGE INTERRUPT ON P64
 7464  00:FFDA  B0E0                .WORD RESET   ; NEGATIVE EDGE INTERRUPT ON P66
 7465  00:FFDC  B0E0                .WORD RESET   ; PARALLEL INTERFACE BUS (PIB)
INTERRUPT
 7466  00:FFDE  B0E0                .WORD RESET   ; IRQ LEVEL INTERRUPT
 7467  00:FFE0  B0E0                .WORD RESET   ; UART0 RECEIVER INTERRUPT
 7468  00:FFE2  B0E0                .WORD RESET   ; UART0 TRANSMITTER INTERRUPT
 7469  00:FFE4  B0E0                .WORD RESET   ; UART1 RECEIVER INTERRUPT
 7470  00:FFE6  B0E0                .WORD RESET   ; UART1 TRANSMITTER INTERRUPT
 7471  00:FFE8  B0E0                .WORD RESET   ; UART2 RECEIVER INTERRUPT
 7472  00:FFEA  B0E0                .WORD RESET   ; UART2 TRANSMITTER INTERRUPT
 7473  00:FFEC  B0E0                .WORD RESET   ; UART3 RECEIVER INTERRUPT
 7474  00:FFEE  B0E0                .WORD RESET   ; UART3 TRANSMITTER INTERRUPT
 7475  00:FFF0  B0E0                .WORD RESET   ; RESERVED INTERRUPT
 7476  00:FFF2  B0E0                .WORD RESET   ; RESERVED INTERRUPT
 7477  00:FFF4  B0E0                .WORD RESET   ; COPROCESSOR INTERRUPT
 7478  00:FFF6  B0E0                .WORD RESET   ; RESERVED INTERRUPT
 7479  00:FFF8  B0E0                .WORD RESET   ; ABORT INTERRUPT
 7480  00:FFFA  B0E0                .WORD RESET   ; NMI - NONMASKABLE INTERRUPT JMP
INTO MONITOR CMD PARSER
 7481  00:FFFC  B0E0                .WORD RESET   ; RES - RESTART INTERRUPT
 7482  00:FFFE  B0E0                .WORD RESET   ; BRK - SOFTWARE INTERRUPT
 7483
 7484                          .ENDS
 7485  00:FF7C
 7486                          .END
 7487
 7488
 7489                         .ENDS
 7490
 7491                         .END
```

Defined Symbol Name Value References

858 ACC 00:DF80 2196 2204 2213 2221 2274 2276 2331 2333
2466 2468 3919
2428 ACCCCEND 00:E871
2423 ACCCCTBL 00:E84D 2370
7250 ACIBAUD = 00:FEAF 7143 7145
7165 ACI_ERR 00:FE6B 7135
7159 ACI_OUT 00:FE67 7169
1032 ACK = 00:0006 3871
367 ACSR0 = 00:DF70 5963 5975 6005 6007 6014 6019 7148
380 ACSR1 = 00:DF72 6141 6218 6233 6271 6273 6280 6285 7149
382 ACSR2 = 00:DF74 6408 6488 6503 6542 6544 6552 6557
384 ACSR3 = 00:DF76 1637 6640 6644 6805 6891 6906 6946 6948
6956 6961 7040 7046 7054 7150
932 ADAY 00:DF99
929 ADAYWK 00:DF97 1522 5308
4654 ADDR_IN 00:F2FF 4615
1780 ADRS 00:E3CA 1702 1733
935 AHR 00:DF9B 5244 5403 5622
5365 ALARM_SET 00:F617 5348
1000 ALRMENAB = 00:0001 2104 5266 5303 5350 5618
999 ALRMIRQ = 00:0002 1641 2106 5303 5346 5631 5879
998 ALRMRST = 00:0004 5883
2986 Alter_Memory = 00:EB96 1097 1781
2460 ALTER_REGS 00:E8AC 1780
2607 ALTER|ERR 00:E9C1 2480 2494 2506 2526 2540 2576 2590
936 AMIN 00:DF9C 5254 5421
931 AMONTH 00:DF98
5671 APR0 00:F7DD 5667
5674 APR1 00:F7E0 5670
5665 APRIL 00:F7CF 5585
379 ARTD0 = 00:DF71 5874 5995
381 ARTD1 = 00:DF73 6069 6253
383 ARTD2 = 00:DF75 6336 6524
385 ARTD3 = 00:DF77 6734 6910 6928 7204
4078 ASC1 00:F0ED 4076
4034 ASCBIN 00:F0BF 1189 3973 4021
4048 ASCERR 00:F0D5 4035 4039
4073 ASCII 00:F0E4 4068
937 ASEC 00:DF9D 5264 5435
723 ATG = 00:0000
933 AYR 00:DF9A
664 A_MENSCH = 00:0010 1626 2055 2112
4436 BACKSPACE 00:F1EA 1099
4433 BACKSPACE2 00:F1E4 3052 3066
5103 BAD_EXIT 00:F4F0 5062 5063 5065 5072 5075 5076 5078 5085

```
                                5088   5090   5187   5197   5205   5272
7240    BAUDOFFSET                = 00:FEAA       7139
 183    BCR                     = 00:DF40      1236   2086   4976   4987   4996   5008
 625    BEEP                    = 00:0040      5925
1033    BELL                    = 00:0007
4207    BIN2DEC                   00:F129      1192   4205
4062    BINASC                    00:F0D7      1195   3405   4748   4768
7389    BINDECH                   00:FF6B      4222   5801
7385    BINDECL                 = 00:FF5B      4224   5793
  99    Bit0                    = 00:0001       737    804   1360   2085   2092   2116   2650   3275
 100    Bit1                    = 00:0002       738    805   1364   2090   3131   4975   4986   5007
                                6011
```

Defined Symbol Name Value References

101 Bit2 = 00:0004 739 1360 1612 2090 2650 4975 4995 5007
7147
102 Bit3 = 00:0008 740 1308 1364 1615 1621 2058 2085 2090
2650 6227 6260 6277
103 Bit4 = 00:0010 741 1297 1360 7151
104 Bit5 = 00:0020 742 1364 1636 6497 6531 6549
105 Bit6 = 00:0040 743 1357 1360 1633 1638
106 Bit7 = 00:0080 1353 1364 1629 1638 6900 6935 6953
1035 BKSP = 00:0008 3017 3021 3041 3076 3080 4434 4437 4597
4624 4633 4891
2823 BY1 00:EAF8 2818
2829 BY2 00:EAFF 2821
2816 BYTE = 00:EAEC 2716
5784 B_DCONV = 00:F855 5407 5425 5439 5710 5720 5726 5761 5771
5777
1052 CAN = 00:0018
758 CARD 00:00B5
617 CFLG = 00:0002
1284 CHKAGAIN 00:E117 1299
1296 CHKLROM 00:E12B 1288
1286 CHKPCMLP 00:E11A 1291
1318 CHKRAM 00:E137
1321 CHKRAM1 00:E13A 1326
1553 CHK_MROM 00:E298 1558
5628 CKAL1 00:F7A0 5624
5622 CKALARM 00:F794 5630
4763 CKNOUT 00:F384 3460 3468 3476 3478 3498
1503 CKTODLP 00:E247 1505
4444 CLEAR_LCD_DISPLAY 00:F1F1 2308 2654 3340 3536 4484 4518
5453 CLOCK_CK_SUM = 00:F67F 1526 5455 5637
703 CLR_LCD_JMP 00:0079 1564 4447
1752 CMDS 00:E3B4 1702 1703
Pre CODE 00:FF7C 773 1014 1072 7484 7489
4937 CONTROL_TONES 00:F423 1103
5114 CONVERT2ASCII = 00:F4F8 5061 5074 5087 5176 5186 5196 5240
5250
5260
788 COPIRQ 00:010C 7440
757 COUNT 00:00B3 2679 2744 2747 2756 2758
1040 C_RETURN = 00:000D 1951 2424 2425 2425 2427 2433 2435 2438
2440 2735 2773 2781 3015 3039 3061 3417
3632 4010 4421 4595 4905
4245 DADD 00:F146 2692 2706 2709 2712 2720 2726 2829 4763
Pre DATA 00:0100 781 782 848
805 DATE_CHK = 00:0002 5538

```
 907  DAY                 00:DF92     5079  5552  5578  5587  5598  5603  5668  5719
 951  DAYLIT                00:DFA2     5544  5556  5558  5580
 956  DAYLITFLG              = 00:0001     5543  5579
 957  DAYLPROG               = 00:0080     5555
 904  DAYWK                 00:DF90     1503  1518  5466  5549  5571  5572  5576  5665
 864  DBREG                 00:DF8A     2243  2254  3919
1044  DC1               = 00:0011     1062
1045  DC2               = 00:0012
1046  DC3               = 00:0013     1063
1047  DC4               = 00:0014
3556  DCMP                00:EE89     2918  2948  3447  3518
1991  DDATE                00:E628     1795
7391  DECBIN               00:FF72
```

```
Defined         Symbol Name          Value           References

 4312   DECTMP0                    00:F176      3926
 4337   DECTMP1                    00:F18C      3067
 7217   DELASTBYTE                   = 00:FE92      7219
  764   dest                     00:00BC
 1010   DFLASTBYTE                   = 00:DFBA      1012
 1699   DFLTPRSR                   00:E378
 7364   DFLTS                    = 00:FF4B      1500   1515   1516   1521   5463
 7375   DFLTSEND                   = 00:FF53      1500   1515   1521   5463
 1012   DFSPACE                  = 00:0005
  996   DIALDELY                 = 00:0010
  684   DIFF                     00:005A      3448   3450   3452   3559   3563   3567   3568   3569
  863   DIRREG                     00:DF88      2240   2257   2333   2340   2342   2468   2561   2563
                                   3919
  377   DISCH                    = 00:0002      5964   5976   6004   6013   6018   6219   6234   6270
                                   6279   6284   6489   6504   6541   6551   6556   6643
                                   6892   6907   6945   6955   6960   7043   7045
  655   DISPTYP                    00:0053      1627   1642   2056   2113
  490   DISP_CNTL_DIR              = 00:DF13
  481   DISP_CNTL_REG              = 00:DF11
  479   DISP_DATA_DIR              = 00:DF12
  470   DISP_DATA_DREG             = 00:DF10
 2367   DISP_FLAGS                 00:E7F3      2312
  704   DISP_LCD_JMP               00:007B      1565   4451
 4450   DISP_LCD_STRNG               00:F1FA      2658   2666
  652   DISP_PTR                   00:0050
 2325   DISP_REGS                  00:E7A1      2310
  667   DISP_TOD_FLG               = 00:0080
 1433   DLY0                     00:E1E7      1434
 1439   DLY1                     00:E1F2      1440
 2383   DMP_FLG1                   00:E80F      2379
 2376   DMP_FLGS                   00:E806      2387
 2384   DMP_FLGX                   00:E811      2381
  962   DOWNT0                   00:DFA5
  963   DOWNT1                   00:DFA7
  964   DOWNT2                   00:DFA9
  965   DOWNT3                   00:DFAB
  966   DOWNT4                   00:DFAD
 3725   DO_HEX                     00:EF5C      3641   3657   3668   3684   3827   3841   3850
 2048   DO_LOW_PWR_PGM               00:E661      1105   1568
 1561   DO_STD                   00:E2A9      1555
 3925   DSPLYDEC                   00:F068      1782
 3929   DSPLYINC                   00:F06D      1783
 3931   DSPLYOLD                   00:F070      1784   3927
  395   DSR0                     = 00:0002
  396   DSR1                     = 00:0008
```

```
 397   DSR2                = 00:0020
 398   DSR3                = 00:0080
1972   DTIME                 00:E5FB      1794
 720   DTMF                 = 00:0006
1003   DTMFTMR                 00:DFB7
 390   DTR0                = 00:0001
 391   DTR1                = 00:0004
 392   DTR2                = 00:0010
 393   DTR3                = 00:0040
 663   DTYPMSK               = 00:000F
2297   DUMPREGS                00:E784      1107   1786   2247
3247   DumpS28                 00:EC9A      1109   1798
```

```
Defined         Symbol Name          Value             References

3166   Dump_1_line_to_Output          00:EC73      1111   2990
3228   Dump_1_line_to_Screen          00:EC93      1113
 736   DUMP_FLGS                    00:0086      3257   3272   3286   3309   3350   3384   3389   3398
                                    3432   3438   3469   3479   3485   3500   3509   3531
                                    3542
3382   Dump_It                      00:ED53      1123
3150   Dump_to_Output                 00:EC6B      1115   1789   3133
3127   Dump_to_Printer                00:EC5C      1117
3192   Dump_to_Screen                 00:EC85      1119
3212   Dump_to_Screen_ASCII           00:EC8C      1121
 824   D|TBUF                       00:01B6
 876   EBIT                       = 00:DF8C      2342   2563
 621   ECHOFF                       = 00:0020      3617   3619   3801   3803   6655
 791   EDGEIRQS                       00:0118      7422   7423   7424   7425   7426   7427
 285   EIER                       = 00:DF47      1384
 265   EIFR                       = 00:DF45      1381
1053   EM                         = 00:0019
1031   ENQ                         = 00:0005
4496   Enter_ADDR                     00:F21A      4492
4498   Enter_EA                     00:F250      4556
2960   Enter_HEX                      00:EB84      2927
2041   ENTER_LOW_POWER_MODE           00:E65E      1799   5653
4497   Enter_SA                     00:F231      4526
1969   ENTR_DATE                      00:E5E8      1999
1968   ENTR_TIME                      00:E5D5      1980
1030   EOT                         = 00:0004
 692   ERRORS                         00:006F      1533   2677   2685   2718   2732   2786   2794
1055   ESC                        = 00:001B      2669   2768   3628   3666   3682   4008   4593   4899
1050   ETB                        = 00:0017
1029   ETX                        = 00:0003
5636   EXITA                        00:F7AE      5619   5626
5567   EXITA6                       00:F726      5565
5560   EXITOCT                        00:F71D      5520   5523   5527   5534   5545   5548   5551   5554
                                    5557
2902   FILL_Memory                    00:EB1E      1125   1790
 737   Flag1                      = 00:0001      3249   3354   3439   3470   3480
 738   Flag2                      = 00:0002      3171   3194   3214   3230   3399   3532
 739   Flag3                      = 00:0004      3152   3171   3174   3194   3230   3390   3501   3543
 740   Flag4                      = 00:0008      3249   3510
 741   Flag5                      = 00:0010      3171   3194   3230   3433
 742   Flag6                      = 00:0020      3171   3174   3230   3287   3310   3351
 743   Flag7                      = 00:0040      3214   3486
 984   FLAGS                        00:DFB6      1527   1546   2103   5267   5304   5347   5351   5522
                                    5617   5632   5880   5884
 866   FLGS                         00:DF8B      2207   2224   2269   2373   3919
```

```
 802   FORMAT_FLAGS                00:0138      5539   5615
1039   FORM_FEED                 = 00:000C
1056   FS                        = 00:001C
1493   FUIRQS                      00:E237      1496
1530   GDTOD                       00:E276      1511
4586   GET_3BYTE_ADDR            = 00:F2A0      1127   4529
4481   Get_Address                 00:F204      1141   1960   2860   3290
5329   GET_ALARM_STATUS              00:F5EE      1129
4527   GET_A_OUT                   00:F283      4493   4557
6589   GET_BYTE_FROM_PC          = 00:FBF9      1131   4383   4410
4377   GET_CHR                     00:F1B2      1133   1680   2668   2686   2734   2761   2767   2780
                                   3362   3366   3534   3626   3648   3656   3664   3678
```

```
Defined         Symbol Name          Value            References

                                                      3733  3807  3826  3834  3840  3849  3963  3971
                                                      4005
 700    GET_CHR_JMP                    00:0073       4369  4379
4546    Get_E_Address                  00:F28C       1143  2916  3336
3962    GET_HEX                        00:F07D       1135  2690  2704  2707  2710  2725  2817
4403    GET_PUT_CHR                    00:F1C6       1137  2929  2939  3013  3037  4019  4591  4889
1959    GET_SAVE_PC                    00:E5C3       1933  1948
4873    GET_STR                        00:F3D7       1139  1984  2003
4515    Get_S_Address                  00:F26F       1145  2907  3294
4002    GET|BYTE                       00:F09C       2476  2490  2502  2522  2536  2572  2586
 702    GET|PUT_CHR_JMP                00:0077       4371  4405
4408    GET|PUT_PC_CHR                 00:F1C9       4370
1955    go8                          00:E5C0       1945  1949  1952
5093    GOOD_EXIT                      00:F4E8       5202  5269  5313  5450  5731  5782
2252    GO_AGAIN                       00:E74E       1955
1947    GO_JML                         00:E5B6       1787
1932    GO_JSL                         00:E5A1       1788
1057    GS                           = 00:001D
3255    G_XS28OUT                      00:ECA2       3153  3176  3250
 946    H100HZ                       00:DF9E
 947    H10HZ                        00:DF9F
 948    H1HZ                         00:DFA0
1898    HELP                           00:E3F6       1792  1793
1908    HELPMENU                       00:E404       1901
4690    HEX2IN                         00:F336       4665  4673  4677
4106    HEXIN                          00:F0F0       1198  2931  2941  3027  3043  3726  3734  4034
                                                      4038  4104  4691  4700
4117    HEXNG                          00:F101       4107
7380    HEXTOPOS                     = 00:FF53
4114    HEXXX                          00:F0FF       4112
  38    HINIB                        = 00:00F0       4075
 910    HR                           00:DF94       5180  5537  5540  5559  5566  5567  5625  5675
                                                      5760
1036    HTAB                         = 00:0009
 789    IABORT                         00:0110       7442
1516    ICLK1                          00:E25E       1520
1522    ICLK2                          00:E26A       1524
4169    IFASC                          00:F116       1201  4167
1733    IJMP                           00:E3B1       1723
5587    INCADAY                        00:F750       5581  5584  5671
5578    INCDAY                         00:F73C       5574
5602    INCMTH                         00:F771       5593  5596
4277    INCT0                          00:F163       4271  4273  4275
4300    INCT1                          00:F174       4294  4296  4298
4269    INCTMP0                        00:F154       2721  2823  2830  2950  3506  3697  3930
```

```
4292   INCTMP1                  00:F165      3058
 761   INPUT_SRC                00:00B7
 638   INPUT_XTRL               00:004D      1624   2649
 726   INTKNT1                  00:0085
7409   INT_VECTORS              00:FF80      7410
 640   IOTEMP                   00:004F
  17   IROM                   = 00:0001      1074   1233   1279   2071
5854   IRQAR0                   = 00:F885      7430
6048   IRQAR1                   = 00:F96C      7432
6316   IRQAR2                   = 00:FAB3      7434
6714   IRQAR3                   = 00:FC75      7436
5943   IRQAT0                   = 00:F8F2      7431
```

```
Defined         Symbol Name         Value           References

 6170   IRQAT1                  = 00:FA04      7433
 6440   IRQAT2                  = 00:FB4A      7435
 6843   IRQAT3                  = 00:FD08      7437
 4149   ISDECIMAL                 00:F10D      1204  4132  4147  5121  5133
 4130   ISHEX                   00:F103      1207  4106  4128  4599
 4155   ISN1                  00:F114      4150  4170
 2195   JSL|RTL_IN              00:E6F9      1941
 7351   LASTDY                  00:FF3F      5589
  620   LASTXONOF                = 00:0010      6122  6207  6389  6477  6634  6637  6786  6880
  749   LINE_CNT                00:00AF      3344  3521  3523  3530
  750   LINE_MAX                00:00B1      3256  3271  3383  3525
   37   LOWNIB                  = 00:000F      4071  4219  5123  5135  5788  5809
  707   LO_PWR_JMP                00:0081      1569  2043
 1037   L_FEED                  = 00:000A      2737  2771  2783
 3269   L_XS28OUT                 00:ECBD      3195  3215  3231
  129   M8                    = 00:0020      1671  1695  1712  1719  2302  2743  2750  5058
                                    5173  5237  5299  5343  5507  5866  5955  6060
                                    6182  6328  6452  6602  6726  6855  7013
 7332   MAXTTBL                  = 00:FF31
  911   MIN                   00:DF95      5190  5531  5533  5535  5770
 7341   MINTTBL                 = 00:FF38
 1398   MINXTALCALC               = 00:E1A9      1468  1474
 1255   MONDATE                 00:E0ED      2026
 1422   MONIRQEND                = 00:E1E2      1492
 1408   MONIRQTBL                = 00:E1AE      1492  1493
 1427   MONSUP1                  = 00:E1E2      1388
  906   MONTH                   00:DF91      5066  5546  5582  5588  5604  5605  5608  5709
 1256   MONVEND                   00:E106
 1250   MONVER                  00:E0B7      1652
 1252   MONVRS                  00:E0CA      2025
 1391   MXTALCALC                = 00:E1A4      1468  1469
 1481   MXTALFND                 = 00:E22E      1475
 1404   MXTLEND                 = 00:E1AE
 1048   NAK                   = 00:0015      3706  3881
  295   NE57ENABLE               = 00:0002
  298   NE64ENABLE               = 00:0010
  299   NE66ENABLE               = 00:0020
 4192   NIBBIN1                 00:F128      4187  4189
 2180   NMIBRK                  00:E6E4      1410  2162
 2157   NMIFLG                  = 00:0002      2185
 2211   NMIS                  00:E70E      2172  2190
 2232   NMIS_J                  00:E731      2208
 2191   NMIX                  00:E6F6      2187
 1335   NOEXTMEM                  00:E14B      1323
 3666   NO_BANK                 00:EEFD      3652
```

```
 666   NO_DISPLAY                = 00:0040
1026   NULL                    = 00:0000
5543   OCTOBER                   00:F6F4
3848   ONE|BY                  00:EFCF      3920
 747   OUTBUF                  00:0087      1974   1977   1993   1996
7023   OUTCH31                   00:FDD6
7053   OUTCH31A                   00:FE06      7042
7055   OUTCH32                   00:FE0B      7044   7047   7051
7066   OUTCH33                   00:FE14      7030
7028   OUTCH3D2                  = 00:FDE0      7026
7049   OUTCH3_A1                  = 00:FE02      7039
 760   OUTPUT_TMP                  00:00B6
```

```
Defined         Symbol Name          Value            References

 639   OUTPUT_XTRL                  00:004E     1616  2994  3129  3132  3135  3168  3273  3276
                                  3279
 830   P1LASTBYTE                    = 00:01C0      832
 832   P1SPACE                     = 00:003F
6647   P3_GETSD1                     = 00:FC48     6632
6650   P3_GETSD2                     = 00:FC4D     6635  6642  6645
6654   P3_GETSD3                      00:FC51     6628
6622   P3_GETSD4                     = 00:FC21     6619
6615   P3_GETSD5                      00:FC16     6610
6660   P3_RD_CH0                     = 00:FC5D     6656
6674   P3_RD_CH1                     = 00:FC6C     6613
 Pre   PAGE0                        00:0000      552   553
 887   PCH                         00:DF8D     2228  2264  2412  2496  2500
 886   PCL                         00:DF8C     1963  2226  2266  2414  2508  2512  3919
 171   PCS7                        = 00:DF27     1283  1298  1316  1346  2084
2474   PC_CNTR_IN                      00:E8C8
 131   PD0                         = 00:FD00     2066
 132   PD1                         = 00:FD01     2067
 133   PD2                         = 00:FD02     2068
 134   PD3                         = 00:FD03     2069
 163   PD4                         = 00:DF20     1354  1358  5967
 164   PD5                         = 00:DF21     1362  1634  6146  6226  6259  6413  6496  6530
                                      6649  6810  6899  6934  7199
 165   PD6                         = 00:DF22     1366  1631  2115  6012  6278  6550  6954  7190
 166   PD7                         = 00:DF23     1370  2083
 135   PDD0                         = 00:DF04     2062
 136   PDD1                         = 00:DF05     2063
 137   PDD2                         = 00:DF06     2064
 138   PDD3                         = 00:DF07     2065
 167   PDD4                         = 00:DF24     1352
 168   PDD5                         = 00:DF25     1361  7197
 169   PDD6                         = 00:DF26     1365  1630  7187  7194
1006   PD_TIMER                        00:DFB8     1372  1373  5642  5645  6663  7021
 294   PE56ENABLE                      = 00:0001
 296   PE60ENABLE                      = 00:0004
 790   PIBIRQ                       00:0114     7428
 300   PIBIRQENABLE                    = 00:0040
3798   PIPE                        00:EF76     1801
3874   PIPE|END                        00:EFFE     3913
3880   PIPE|ERR                        00:F004     3814  3829  3837  3843  3852
4454   POSITION_TEXT_CURSOR            00:F1FD     2662  2754  2924  3334  4489  4523
4553
  35   POWER_DOWN_COUNT              = 00:0708     6662  7020
3261   Print_Head                   00:ECA9     3393
 665   PUFLG                       = 00:0020
```

4389	PUT_CHR	00:F1BC	1147 1678 1683 1709 2384 2488 2740 2776 3006 3012 3022 3054 3056 3062 3077 3079 3081 3406 3408 3409 3418 3419 3443 3445 3473 3483 3494 3504 3546 3707 3718 3872 3882 3906 4018 4422 4429 4435 4438 4609 4625 4634 4722 4754 4756 4772 4774 4829
701	PUT_CHR_JMP	00:0075	4373 4391
4809	PUT_STR	00:F3A1	1149 1654 1726 1902 1978 1981 1997 2000 2371 2396 2403 2559 2928 3360 3395 4528
970	PWD_CELLS	00:DFB3	1387 2120
297	PWMENABLE	= 00:0008	
519	PWR_XTRL_DIR	= 00:DFE3	1623
510	PWR_XTRL_REG	= 00:DFE1	1622 2059

Defined	Symbol Name	Value	References
767	PZLASTBYTE	= 00:00C0	769
769	PZSPACE	= 00:003F	
3389	P_HEADER	00:ED57	3346 3537 3547
3979	RDOERR	00:F097	3974
1724	RDY	00:E39D	1710
1730	Ready_Now	00:E3A8	1725
5388	READ_ALARM	00:F61E	1151
5694	READ_DATE	00:F7E7	1153 1994
5746	READ_TIME	00:F81E	1155 1975
6140	REC1_DONET	00:F9F4	6129 6134
6407	REC2_DONET	00:FB3A	6396 6401
2804	RecordNo	00:EAE3	2665
5901	RECV0_R10	00:F8CD	5899
5902	RECV0_R11	00:F8CF	5892
5918	RECV0_R12	00:F8E4	5915
5925	RECV0_R14	00:F8EB	5919
5886	RECV0_R8	00:F8B1	5877 5881
5891	RECV0_R9	00:F8BB	5889
6145	RECV1_DTR	00:F9FC	6121
6095	RECV1_R10	00:F9B5	6093
6096	RECV1_R11	00:F9B7	6086
6112	RECV1_R12	00:F9CC	6109
6119	RECV1_R14	00:F9D3	6113
6127	RECV1_R15	00:F9E2	6124
6080	RECV1_R8	00:F999	6073
6085	RECV1_R9	00:F9A3	6083
6131	RECV1_XOFF	00:F9E8	6077
6137	RECV1_XON	00:F9EF	6079
6412	RECV2_DTR	00:FB42	6388
6362	RECV2_R10	00:FAFB	6360
6363	RECV2_R11	00:FAFD	6353
6379	RECV2_R12	00:FB12	6376
6386	RECV2_R14	00:FB19	6380
6394	RECV2_R15	00:FB28	6391
6347	RECV2_R8	00:FADF	6340
6352	RECV2_R9	00:FAE9	6350
6398	RECV2_XOFF	00:FB2E	6344
6404	RECV2_XON	00:FB35	6346
6804	RECV3_DONET	00:FCFB	6793 6798
6809	RECV3_DTR	00:FD03	6785
6760	RECV3_R10	00:FCBD	6758
6761	RECV3_R11	00:FCBF	6751
6777	RECV3_R12	00:FCD4	6774
6783	RECV3_R14	00:FCDA	6777
6791	RECV3_R15	00:FCE9	6788

6745 RECV3_R8 00:FCA1 6738
6750 RECV3_R9 00:FCAB 6748
6795 RECV3_XOFF 00:FCEF 6742
6801 RECV3_XON 00:FCF6 6744
6964 REC_DONE = 00:FDB5 5920 5927 6114 6125 6142 6147 6381 6392 6409 6414 6778 6789 6806
2432 REGTBL1 00:E871 2395
2437 REGTBL2 00:E898 2402 2558
2392 REGTTL1 00:E81E 2326 2462
2399 REGTTL2 00:E82A 2339
3919 Reg_Addr 00:F057 3857 3894
3917 Reg_ID 00:F045 3809

```
Defined         Symbol Name          Value           References

 2517   REG_IN                    00:E924
 2554   REG_IN_II                 00:E963
 2349   REG_OUT                   00:E7CF      2336   2345   2471   2566
 3918   Reg_Size                  00:F04E      3819   3862   3897
 3920   Reg_Strt                  00:F060      3822
 1231   RESET                   = 00:E0B0      1185   1411   1412   1413   1414   1415   1416   1417
                                   1419   1420   1421   7417   7418   7419   7420   7438
                                   7439   7444   7445   7451   7452   7453   7454   7455
                                   7456   7457   7458   7459   7460   7461   7462   7463
                                   7464   7465   7466   7467   7468   7469   7470   7471
                                   7472   7473   7474   7475   7476   7477   7478   7479
                                   7480   7481   7482
 5285   RESET_ALARM                 00:F5C7      1157
  997   RES_COMP                  = 00:0008      1545   5521
 3714   RETURN_ADDR                 00:EF46      3634   3932
 3694   RETURN_BYTE                 00:EF2C      3639   3680
 3889   RET|REGS                  00:F010      3920
 1550   RET|TO|MENSCH               00:E295      1805
 7395   ROMSPACE                  = 00:0004
 4381   ROM_GET_CHR                 00:F1B5      4368
  811   ROM_IBUF0                 00:0139      1584
  812   ROM_IBUF1                 00:0143      1586
  813   ROM_IBUF2                 00:014D      1588
  814   ROM_IBUF3                 00:0157      1590
  816   ROM_OBUF0                 00:0161      1594
  817   ROM_OBUF1                 00:016B      1596
  818   ROM_OBUF2                 00:0175      1598
  819   ROM_OBUF3                 00:017F      1600
 4394   ROM_PUT_CHR                 00:F1BF      4372
 1669   ROM|START                   00:E358      2249
 1058   RS                      = 00:001E
 4464   RTL_EXIT                    00:F203      1563
  141   RVD08                     = 00:DF08
  181   RVD28                     = 00:DF28
  331   RVD4A                     = 00:DF4A
  695   R_TYPE                      00:0072
 1693   S0                        00:E374
 1680   S00                       00:E366      1682   1727
 1703   S1                        00:E37E      1706
 1712   S2                        00:E38E      1704
 2803   S28_Loader                  00:EAD5      2657
 5271   SA_EXIT                     00:F5C4      5241   5243   5248   5251   5253   5258   5261   5263
 2165   SBREAK                      00:E6D6      1159   1409   2162
 2156   SBRK                      = 00:0001      2170   2188
  889   SB_SENTL                    00:DF8F      1382   2171   2186   2189   2253   2867
```

```
 627  SDATA_SI0                00:0044
 628  SDATA_SI1                00:0045
 629  SDATA_SI2                00:0046
 630  SDATA_SI3                00:0047      6612   6625   6657   6671
 912  SEC                   00:DF96      5200   5515   5516   5524   5525   5526   5529   5776
 506  SEGA_DATA_DIR              = 00:DFE2
 496  SEGA_DATA_REG              = 00:DFE0      1618   1619
7113  SELECT_COMMON_BAUD_RATE       = 00:FE21      1161   1531
4458  SEND_BEEP                 00:F200      5601
7000  SEND_BYTE_TO_PC            = 00:FDBC      1163   4396   4412   6658
4419  SEND_CR                   00:F1D6      1165   1650   1676   1903   1973   1992   2313   2363
                               2923   3517   4445   4550
```

```
Defined         Symbol Name          Value            References

 4745   SEND_HEX_OUT              00:F370     1169  2355  2359  3004  3008  3010  3024
3515
                                  4720  4724  4726  4764
 4426   SEND_SPACE                 00:F1DF     1167  2329  2357  2465  2515  2533  2583  3909
                                  3933  4425
 4424   SEND_SPACE2                00:F1DB     1722  2328  2360  2385  2464  2514  2550
2568
                                  2600
 5223   SET_ALARM                  00:F575     1171
 2855   SET_Breakpoint             00:EB09     1173  1803
 5044   SET_DATE                   00:F495     1175  2005
 4366   SET_GET|PUT_CHR              00:F1A2     1571
 5159   SET_TIME                   00:F528     1177  1986
  603   SFLAG0                    00:0040     1538  1539  1540  1541  5907  5926  5997
  605   SFLAG1                    00:0041     6071  6101  6119  6123  6128  6133  6139  6190
                                  6200  6208  6215  6222  6255  6263
  606   SFLAG2                    00:0042     6338  6368  6386  6390  6395  6400  6406  6460
                                  6470  6478  6485  6492  6526  6534
  607   SFLAG3                    00:0043     1613  3616  3620  3701  3709  3800  3804  3876
                                  3884  6608  6630  6638  6652  6654  6736  6766
                                  6783  6787  6792  6797  6803  6863  6873  6881
                                  6888  6895  6930  6938  7037
  616   SFLG                    = 00:0001     5906  6100  6367  6609  6651  6765
 1042   SI                      = 00:000F
  564   SINCNT0                   00:0006     1575  5888  5898  5914  5917
  569   SINCNT1                   00:000E     1576  6082  6092  6108  6111
  574   SINCNT2                   00:0016     1577  6349  6359  6375  6378
  579   SINCNT3                   00:001E     1578  6618  6747  6757  6773  6776
  562   SINEND0                   00:0002     5891  5896  5901  5918
  567   SINEND1                   00:000A     6085  6090  6095  6112
  572   SINEND2                   00:0012     6352  6357  6362  6379
  577   SINEND3                   00:001A     1608  6616  6623  6750  6755  6760
  561   SININDX0                  00:0000     5886  5902
  566   SININDX1                  00:0008     6080  6096
  571   SININDX2                  00:0010     6347  6363
  576   SININDX3                  00:0018     1609  6627  6745  6761
  563   SIN_BUF0                  00:0004     1585  5904
  568   SIN_BUF1                  00:000C     1587  6098
  573   SIN_BUF2                  00:0014     1589  6365
  578   SIN_BUF3                  00:001C     1591  6624  6763
 7173   SIOPORTS                  = 00:FE71     1543
 3614   SLASH                     00:EEA1     1800
 3699   SLASH|END                   00:EF34     3635
 3705   SLASH|OUT                   00:EF3A     3630  3643  3659  3667  3669  3683  3685
  619   SNDOVF                  = 00:0008     5982  5998  6132  6138  6239  6264  6399  6405
```

```
                                6510   6535   6796   6802   6914   6939   7050
 706   SND_BEEP_JMP                00:007F      1567   4459
 721   SNGL                 = 00:0002
1041   SO                   = 00:000E
1027   SOH                  = 00:0001
 376   SON                  = 00:0001      6006   6013   6018   6272   6279   6284   6543   6551
                                6556   6641   6643   6947   6955   6960   7041   7053
 763   source               00:00B8
 584   SOUTBUF0                00:0024      1595   5993
 589   SOUTBUF1                00:002C      1597   6251
 594   SOUTBUF2                00:0034      1599   6522
 599   SOUTBUF3                00:003C      1601   6926   7036
 585   SOUTCNT0                00:0026      1579   5988
 590   SOUTCNT1                00:002E      1580   6245
```

Defined	Symbol Name	Value	References
595	SOUTCNT2	00:0036	1581 6516
600	SOUTCNT3	00:003E	1582 6920 7025
583	SOUTEND0	00:0022	5972 6002
588	SOUTEND1	00:002A	6231 6268
593	SOUTEND2	00:0032	6501 6539
598	SOUTEND3	00:003A	1611 6904 6943 7023 7034
582	SOUTINDX0	00:0020	5971 5992 6001
587	SOUTINDX1	00:0028	6230 6249 6267
592	SOUTINDX2	00:0030	6500 6520 6538
597	SOUTINDX3	00:0038	1610 6903 6924 6942 7029
995	SPDFLG	= 00:0020	
974	SPEED	00:DFB5	1483 7138
204	SSCR	= 00:DF41	1349 1436 2091 2093
1268	START	= 00:E109	1237
632	STATUS_S0	00:0048	5895
633	STATUS_S1	00:0049	6089
634	STATUS_S2	00:004A	6356
635	STATUS_S3	00:004B	6754 7067
636	STEMP_Sx	00:004C	7137 7141
861	STK|PTR	00:DF86	1944 2237 2260 3919
823	STR_BUF	00:018E	1983 2002 4601 4608 4614
822	STR_BUF_HDR	00:018B	
821	STR_BUF_PTR	00:0189	4888 4893 4896 4901 4904
1028	STX	= 00:0002	
5205	ST_EXIT	00:F572	5177 5179 5184 5189 5194 5199
1054	SUB	= 00:001A	
622	SXOFFLG	= 00:0040	6127 6191 6193 6195 6207 6214 6394 6461 6463 6465 6477 6484 6791 6864 6866 6868 6880 6887
623	SXONFLG	= 00:0080	6191 6193 6199 6214 6461 6463 6469 6484 6637 6864 6866 6872 6887
1049	SYN	= 00:0016	
351	T0CH	= 00:DF61	
350	T0CL	= 00:DF60	
245	T0FLG	= 00:0001	
334	T0LH	= 00:DF51	
333	T0LL	= 00:DF50	
353	T1CH	= 00:DF63	1448 1456
352	T1CL	= 00:DF62	1446 1457 1466
1460	T1DELAY	00:E212	1461
5613	T1EXIT	= 00:F786	5560 5569 5590 5600 5607 5676
246	T1FLG	= 00:0002	1449 5511
336	T1LH	= 00:DF53	
335	T1LL	= 00:DF52	
1456	T1ZERO	00:E20A	1458

```
 355   T2CH                  = 00:DF65
 354   T2CL                  = 00:DF64
 247   T2FLG                  = 00:0004
 338   T2LH                  = 00:DF55
 337   T2LL                  = 00:DF54
7232   T2_1MSEC_TBL                = 00:FEA0
 357   T3CH                  = 00:DF67
 356   T3CL                  = 00:DF66
 248   T3FLG                  = 00:0008      7208
 340   T3LH                  = 00:DF57
 339   T3LL                  = 00:DF56
 359   T4CH                  = 00:DF69      7146
```

```
Defined         Symbol Name         Value           References

 358    T4CL                = 00:DF68
 249    T4FLG                = 00:0010      7208
 342    T4LH                = 00:DF59      1507   5470
 341    T4LL                = 00:DF58      1506   5469   7144
 361    T5CH                = 00:DF6B
 360    T5CL                = 00:DF6A      4983   5001
 250    T5FLG                = 00:0020      4977   4984   5005
 344    T5LH                = 00:DF5B
 343    T5LL                = 00:DF5A
 363    T6CH                = 00:DF6D
 362    T6CL                = 00:DF6C      4992   5004
 251    T6FLG                = 00:0040      4977   4993   5005
 346    T6LH                = 00:DF5D
 345    T6LL                = 00:DF5C
 365    T7CH                = 00:DF6F
 364    T7CL                = 00:DF6E
 252    T7FLG                = 00:0080
 348    T7LH                = 00:DF5F
 347    T7LL                = 00:DF5E
 214    TCR                 = 00:DF42      1378   7202
 693    TEMP                  00:0070      2937   2943   2944   2946   3036   3048   3970   4017
                              4037   4069   4697   4702   4753   4771
 949    TENTHSEC              00:DFA1
 236    TER                 = 00:DF43      1375   1450   4978   4985   4994   5006   7153   7209
3825    THREE|BY              00:EFA4      3920
 275    TIER                = 00:DF46      1377   1451   7152
 255    TIFR                = 00:DF44      1376   5512
 804    TIME_CHK              = 00:0001      5614
 685    TMP0                  00:005D      2368   2393   2400   2411   2413   2415   2556   2705
                              2708   2711   2819   2820   2911   2914   2947   3301
                              3305   3319   3322   3326   3402   3467   3475   3477
                              3488   3497   3558   3562   3566   3674   3676   3690
                              3695   3715   3719   3721   3934   4270   4272   4274
                              4313   4316   4317   4319   4320   4322   4719   4723
                              4725
 686    TMP1                  00:0060      3003   3007   3009   3023   3049   3050   3302   3306
                              3858   3860   3867   3891   3895   3900   4293   4295
                              4297   4338   4341   4342   4344   4345   4347
 687    TMP2                  00:0063      1962   1964   2332   2341   2351   2354   2467   2528
                              2532   2542   2546   2562   2578   2582   2592   2596
                              2865   2910   2913   3300   3304   3320   3324   3328
                              3557   3561   3565   4667   4675   4679
 688    TMP4                  00:0066      2688   2689   2719   2727   3427   3428   3513   4247
                              4248   4249   4251   4822   4823   4825
 689    TMP6                  00:0069      3624   3646   3655   3662   3671   3673   3675   3817
```

```
                          3832  3846  3855  3866  4036  4044
 690   TMP8               00:006C     5125  5129  5130  5137  5794  5804
 679   TMPC               00:0057     2334  2335  2343  2344  2361  2469  2470  2551
                          2564  2565  2601  2691  2693  2694  2695  2703
                          2714  2722  2824  2831  3001  3059  3430  3436
                          3457  3459  3462  3463  3464  3465  3507  3618
                          3645  3654  3661  3670  3689  3700  3708  3732
                          3736  3737  3802  3831  3845  3854  3875  3883
 677   TMPRY_PTR             00:0054     4663  4669  4690  4699  4885  4886  4902  4908
 994   TMRIFLG            = 00:0040
 959   TODCKS             00:DFA3     1510  5472
5563   TODINT8            = 00:F71F     5542
```

```
Defined         Symbol Name         Value           References

 5571   TODINT9                  00:F72D      5568
 5495   TODIRQ                 = 00:F69F      1418   5493
 5514   TOD_AGIN                 00:F6B5
  888   TPBR                   00:DF8E      1965   2230   2262   2410   2482   2486
 5992   TRAN0_0                  00:F937      5990
 5994   TRAN0_1                = 00:F93B      5979
 5982   TRAN0_1A                 00:F929
 5997   TRAN0_2                  00:F93E
 5971   TRAN0_2B                 00:F918      5965   5983
 5975   TRAN0_2C                 00:F91E
 5986   TRAN0_3                = 00:F92F      5973
 6010   TRAN0_3A                 = 00:F957      5969   5977   5984   5999
 6017   TRAN0_3B                 = 00:F964      6003
 6249   TRAN1_0                  00:FA71      6247
 6217   TRAN1_0A                 = 00:FA3E      6192
 6252   TRAN1_1                = 00:FA75      6202   6210   6237
 6239   TRAN1_1A                 00:FA63      6224
 6222   TRAN1_1B                 00:FA45
 6263   TRAN1_2                  00:FA85      6257
 6230   TRAN1_2B                 00:FA52      6220   6240
 6243   TRAN1_3                = 00:FA69      6232
 6276   TRAN1_3A                 = 00:FA9E      6228   6235   6241   6261   6265
 6283   TRAN1_3B                 = 00:FAAB      6269
 6212   TRAN1_ERR                  = 00:FA3A      6194
 6205   TRAN1_XOFF                 = 00:FA32      6196
 6198   TRAN1_XON                  = 00:FA2A
 6520   TRAN2_0                  00:FBB7      6518
 6487   TRAN2_0A                 = 00:FB84      6462
 6523   TRAN2_1                = 00:FBBB      6472   6480   6507
 6510   TRAN2_1A                 00:FBA9      6494
 6492   TRAN2_1B                 00:FB8B
 6534   TRAN2_2                  00:FBCB      6528
 6500   TRAN2_2B                 00:FB98      6490   6511
 6514   TRAN2_3                = 00:FBAF      6502
 6548   TRAN2_3A                 = 00:FBE4      6498   6505   6512   6532   6536
 6555   TRAN2_3B                 = 00:FBF1      6540
 6482   TRAN2_ERR                  = 00:FB80      6464
 6475   TRAN2_XOFF                 = 00:FB78      6466
 6468   TRAN2_XON                  = 00:FB70
 6924   TRAN3_0                  00:FD78      6922
 6890   TRAN3_0A                 = 00:FD42      6865
 6927   TRAN3_1                = 00:FD7C      6875   6883   6901
 6914   TRAN3_1A                 00:FD6A      6897
 6895   TRAN3_1B                 00:FD49
 6938   TRAN3_2                  00:FD8C      6932
```

```
 6903   TRAN3_2B                 00:FD56      6893   6915
 6918   TRAN3_3                = 00:FD70      6905
 6952   TRAN3_3A                 = 00:FDA4      6908   6916   6936   6940
 6959   TRAN3_3B                 = 00:FDB0      6911   6944
 6885   TRAN3_ERR                  = 00:FD3E      6867
 6878   TRAN3_XOFF                 = 00:FD36      6869
 6871   TRAN3_XON                  = 00:FD2E
 6966   TRANS_DONE                   = 00:FDB5      6008   6015   6020   6274   6281   6286   6545
6553
                                  6558   6949   6957
 1474   TRYMINXTAL                   00:E224      1470
 1469   TRYMXTAL                   00:E21C      1472   1477
 2101   TRY_RESTART                  00:E6A8      2122   2124
```

```
Defined         Symbol Name          Value             References

 3839   TWO|BY                    00:EFBF      3836   3920
  705   TXT_CUR_JMP                 00:007D      1566   4455
  724   T_TIME                    00:0083      1535   5633   5634
  798   UALRMIRQ                    00:0134      1644   1647
  311   UART0R                   = 00:0001
  312   UART0T                   = 00:0002
  313   UART1R                   = 00:0004
  314   UART1T                   = 00:0008
  315   UART2R                   = 00:0010
  316   UART2T                   = 00:0020
  317   UART3R                   = 00:0040
  318   UART3T                   = 00:0080
  785   UBRK                      00:0100      1494   7441
  320   UIER                     = 00:DF49      1379   1639
  302   UIFR                     = 00:DF48      1380   5872   5961   6066   6188   6334   6458   6732
                                                6861
  787   UNIRQ                     00:0108      7429
  795   UNIRQT0                     00:0128      7414
  794   UNIRQT1                     00:0124      7415
  793   UNIRQT2                     00:0120      7416
  792   UNIRQT7                     00:011C      7421
  786   UNMI                      00:0104      7443
 4186   UPPER_CASE                  00:F11D      1210   1700   4130   4184
  968   UPT0                    00:DFAF
  797   URESTART                    00:0130      2108   2118
 1059   US                       = 00:001F
  796   USER_CMD                    00:012C      1802
  143   VCS0                     = 00:DF10      145    146    147    148    149    150    151    152
                                                153    154    155    156    157    158    159    160
 2023   VERSION                     00:E655      1179
  402   VIA2_PDA                   = 00:DFE1
  403   VIA2_PDDA                  = 00:DFE3
  156   VIA_ACR                    = 00:DF1B
  160   VIA_P0RA                   = 00:DF1F
  146   VIA_PA                   = 00:DF11
  145   VIA_PB                   = 00:DF10
  157   VIA_PCR                    = 00:DF1C
  148   VIA_PDDA                   = 00:DF13
  147   VIA_PDDB                   = 00:DF12
  159   VIA_PIER                   = 00:DF1E
  158   VIA_PIFR                   = 00:DF1D
  155   VIA_PSR                    = 00:DF1A
 1038   VTAB                     = 00:000B
  150   V_PT1CH                    = 00:DF15
  149   V_PT1CL                    = 00:DF14
```

```
 152   V_PT1LH                = 00:DF17
 151   V_PT1LL                = 00:DF16
 154   V_PT2CH                = 00:DF19
 153   V_PT2CL                = 00:DF18
1258   WDC                    00:E106      1287   1322   1554
 682   WRAP                   00:0059      3343   3421   4276   4299
3688   WRITE_BYTE                00:EF23      3650
4767   WRT2OUT                   00:F38C      2483   2497   2509   2529   2543   2579   2593   2757
                                 2759   3696   3716   3720   3722   3901   3935
2154   WRT_ERR                 = 00:FFB4
4718   WR_3_ADDRESS                 00:F351      1181   2416
2409   WR_PC                   00:E836      2327   2463
```

Defined Symbol Name Value References

128 X8 = 00:0010 1272 1429 1463 1485 1670 1694 2303 4211
5460 5507 5639 5867 5956 6061 6183 6329
6453 6603 6727 6856 7014 7127
2794 XLHDONE 00:EACD
2734 XLSFIN 00:EA69 2738
2739 XLSFIN1 00:EA75 2736
2761 XLSFN 00:EA9C 2763 2777
2681 XLSS = 00:EA00 2764
2732 XLSS0 00:EA67 2700
2779 XLSSDONE 00:EABF 2701 2784
2786 XLSSXIT 00:EACB 2782
2799 XLS_BAD 00:EAD3 2671 2769 2795
1063 XOFF = 00:0013 6076 6209 6343 6479 6741 6882
1062 XON = 00:0011 6078 6201 6345 6471 6743 6874
618 XONOFLG = 00:0004 1537 6072 6120 6223 6256 6339 6387 6493
6527 6631 6737 6784 6896 6931 7038
859 XREG 00:DF82 2233 2272 3919
2673 XS28A 00:E9F5 2670
3265 XS28BN = 00:0014 3429
2671 XS28EEE 00:E9F2 2651
2775 XS28ER1 00:EAB7 2772
2767 XS28ERR 00:EAA7 2730 2774
2725 XS28G2 00:EA5A 2715
2724 XS28G3 00:EA58 2717
2716 XS28GD1 00:EA47 2724
2646 XS28IN = 00:E9C3 1183 2674
2703 XS28LA1 00:EA26 2698
3284 XS28OUT 00:ECD0 3258 3277
3421 XS28OUTA 00:ED99 3391 3527
2676 XS28ROM 00:E9F9 1796
3263 XSLSTLINE 00:ECB1 3359
3426 XWH00 00:ED9F 3422
3459 XWH10 00:EDD9 3449 3451 3454
3462 XWH1A 00:EDDE 3440
3475 XWH1b 00:EDF7 3471
3485 XWH2 00:EE0D 3481 3508
3493 XWH2A 00:EE1D 3490
3494 XWH2B 00:EE1F 3492
3497 XWH2C 00:EE25 3487
3500 XWH3 00:EE2A 3495
3506 XWH3a 00:EE36 3502
3517 XWH3b 00:EE4B 3511
3521 XWH5 00:EE54
3529 XWH6 00:EE60 3526
3539 XWH7 00:EE78 3519

```
3542    XWH8                    00:EE7A       3533
3547    XWH9                    00:EE86       3544
 908    YR                    00:DF93       5091   5594   5610   5725
 860    YREG                    00:DF84       2234   2271   3919


        Lines Assembled : 7491              Errors : 0
```