'W65C134 Internal ROM Monitor ($F000)'
'MON7.ASM - GENERAL LOOKUP TABLES'

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

Input Filename : IMON.ASM
Output Filename : IMON.OBJ

```
 1                      TTL 'W65C134 Internal ROM Monitor ($F000)'
 2
 3                      ;06-23-1993
 4
 5                      .PW 132
 6
 7  00:0000                     CHIP 65C02
 8                      SPACES ON
 9                  .LONGA OFF
10                  .LONGI OFF
11
12  00:0000                     INCLUDE COPYRITE.ASM
13                  ;*****************************************************************************
14                  ;*                                                           *
15                  ;*          (C) Copyright The Western Design Center           *
16                  ;*             1988,1989,1990,1991,1992,1993,1994             *
17                  ;*                                                           *
18                  ;*  Program written by:                                      *
19                  ;*      J. Paul Hittel, L. A. Hittel and Ralph Greenthal.    *
                    *
20                  ;*                                                           *
21                  ;*****************************************************************************
22
23  00:0000                     INCLUDE MONFLOW.ASM
24                      .STTL 'Monitor / Program Flow'
25                      .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'Monitor / Program Flow'

```
                ;                       PROGRAM FLOW
                ;
                ;     The 134 monitor (called the F-thousand monitor) is screened into the
                ; ROM of the 134.  This section of text describes the steps taken by the
                ; monitor from power up to a command prompt.  It is intended as an overview
                ; only; not a specific, line by line analysis of the code.
                ;
                ;
                ;                       GOALS
                ;
                ;     The monitor must be able to be 'shut off' that is it must exit to
                ; another program immediately after reset if necessary.
                ;
                ;     The monitor must handle the serial port on the 134, and must provide
                ; routines such that another program can easily use the serial port via
                ; the monitor.
                ;
                ;     The monitor must be able to load other programs into RAM, and
                ; provide some debugging capabilities.
                ;
                ;     The monitor must maintain a time of day clock, and be capable of
                ; maintaining that clock on minimum power.
                ;
                ;     The monitor must fit in the $F000 to $FFFF memory space.
                ;
                ;
                ;     These are the main goals of the monitor.
                ;
                ;
                ;
                ;                       RESET SEQUENCE
                ;
                ;     This monitor is intended for the internal ROM of the 134.  As such,
                ; the assumption is that the reset vector is entered from an internal ROM
                ; reset.  In reality, this means that the code is started with the BCR=00.
                ;
                ;     Reset can be either a pushbutton reset or a power up reset.  There
                ; is no simple way to differentiate.  However, we will set some semiphores
                ; (flags) in memory to tell us that certain aspects of the machine are
                ; already initialized and should not be changed by the reset routine.
                ;
                ;     There is a checksum to tell us that the time of day clock is running
                ; and is correct.  If this is the case, the clock value is not reset on
                ; reset.  The same semiphore is used for baud rate.  Therefore, the only
                ; way to get the ToD (Time of Day) clock or the baud rate to completely
                ; reset is to completely remove power from the processor long enough for
                ; the memory to scramble (one minute usually does it).
                ;
                ;     The second reset semiphore used is a three byte sequence in RAM to
                ; indicate that the system is in 'power down'.  If  this is set the system
                ; will jump to the low power vector (NE46) immediately after reset.
                    .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'Monitor / Program Flow'

```
78                  ; RESET OCCURS
79                  ;
80                  ;       Interrupts disabled, stack reset, decimal mode cleared.
81                  ;
82                  ;       Check for low power semiphore; JMP (UNE46) if semiphore set.
83                  ;
84                  ;       Set BCR to 01 - this turns on external address and data lines.
85                  ;
86                  ;       Set PCS3 to $C0 - turns on RAM and ROM chip selects from the 134.
87                  ;
88                  ;       Check location $8000-$8002 for the string 'WDC'.
89                  ;       If it was there, JMP $8004.
90                  ;
91                  ;       Check location $0200-$0202 for the string 'WDC'         .
92                  ;       If it was there, JMP $0204.
93                  ;
94                  ;       Start the fast clock. (Don't use it, just start it.)
95                  ;
96                  ;       Copy our interrupt vectors into the RAM interrupt vectors.
97                  ;
98                  ;       Delay  256 * 5 cycles for fast clock to be stable.
99                  ;
100                 ;       Switch to fast clock.
101                 ;
102                 ;       Enable the NE46 interrupt (but not the overall I bit).
103                 ;       This interrupt is used to detect power going down.
104                 ;
105                 ;       Set Timer 2 for a 1 second interrupt (ToD timer).
106                 ;
107                 ;       Enable T2 interrupt (but not the I bit yet).
108                 ;
109                 ;       Set up pointers to the serial buffers. (tiny, in uP RAM)
110                 ;
111                 ;       Figure out the fast crystal frequency by counting it against
112                 ;       the 32 KHz ToD clock crystal.
113                 ;
114                 ;       Check the ToD clock checksum to see if the clock is valid.  If
115                 ;       the clock is not valid, reset it and reset the baud rate counters
116                 ;       for the serial port to the default values.  If the clock checksum
117                 ;       is OK, leave the clock and the serial baud rate alone.
118                 ;
119                 ;       Set up control port of serial port for Xmit and RX.
120                 ;
121                 ;       Check location $E000 for $4C.  If a 4C is present, JSR $E000.
122                 ;
123                 ;       Read the serial port to clear any inital trash data.
124                 ;
125                 ;       ENABLE INTERRUPTS VIA CLEARING THE I BIT.
126                 ;
127                 ;       Output the initial message to the serial port.
128                 ;       (Small initial buffers and a handshake line held in the FALSE
129                 ;        state may cause the monitor to hang here waiting for buffer space
130                 ;        that will not be available until the handshake line goes TRUE)
131                 ;
132                 ;       Execute a BRK instruction (which takes us to the command interpreter)
```

```
133  00:0000                    .END
134
```

'W65C134 Internal ROM Monitor ($F000)'
'Monitor / Program Flow'

```
135  00:0000                    INCLUDE MONTXT.ASM
136                        .STTL 'MONTXT.ASM - Monitor Commands Defined'
137                        .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MONTXT.ASM - Monitor Commands Defined'

```
138                     ;01-13-1995
139
140
141
142
143             ; WRITTEN BY L. A. HITTEL and RALPH GREENTHAL
144             ;
145             ; ORIGIONAL  DATE:  JUNE 02,1988
146             ;  REVISION 1.01 DATE:  DEC  05,1989
147             ;  REVISION 1.02 DATE:  OCT 08,1991
148             ;      7 BIT SERIAL UART CODE FIXED
149             ;      ^C RETURNS 03h NOW - C still set!
150             ;
151             ; REV:  01.03   never issued
152             ;
153             ; REV:  01.04   June 21, 1993
154             ;    MODIFIED:
155             ;      1) SOME BRANCHES REMOVED TO SAVE SPACE
156             ;      2) UPPER CASE NOT WORKING FOR 'z'
157             ;      3) 4.9152 MHZ ADDED TO TABLES
158             ;      4) ASCI INIT STACK PROBLEM
159             ;      5) RESTORED Acc FOR ROUTINE 'SPAC'
160             ;      6) CHECKSUM fixed
161             ;
162             ; REV:  01.05   JUNE 13, 1994
163             ;      FIXED START-UP SEQUENCE
164             ;
165             ; REV:  01.06   DEC 09, 1994
166             ;      REMOVED ICE BIT IN BCR FOR ROM USE
167             ;
168             ; REV:  01.07   JAN 13, 1995
169             ;      ADDED TEST FOR SRAM AT $200
170             ;
171             ; COM LOG CO. INC. W65C134 Controller Interface Monitor
172             ;
173             ;
174             ;                       MONITOR PREREQUISITES:
175             ;
176             ;
177             ;  This monitor expects a terminal to be connected to the serial port
178             ;  The terminal must be configured as follows:
179             ;
180             ;  Hardware handshaking.
181             ;  8 bit data.
182             ;  No parity.
183             ;  9600 Baud (Unless otherwise noted).
184             ;
185             ;
186             ;
187             ;  On reset, the monitor sends copyright and version notices, as well as
188             ;  a register display to the terminal.  (There are semiphores which can
189             ;  be placed in EPROM to prevent this)
190             ;
191             ;  When the monitor is ready for a command, a '.' (period) is sent.
192             ;
```

```
193                 ;  Commands are entered after the period.  No backsapces are allowed;
194                 ;  this is due to the small initial buffer space.  If a character is
```

'W65C134 Internal ROM Monitor ($F000)'
'MONTXT.ASM - Monitor Commands Defined'

```
195                     ;  entered incorrectly, usually a return (CR) will cancel the command.
196                     ;
197                     ;  In most cases, a control C  (^C) will cancel a command in progress.
198
199                             .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MONTXT.ASM - Monitor Commands Defined'

```
200                     ;
201                     ;                       MONITOR COMMANDS
202                     ;
203                     ; ? or H    Lists the commands available
204                     ;
205                     ;
206                     ; R         Display processor registers (PC,F,A,X,Y,SP)
207                     
208                     ; M         Alter Memory address and locations
209                     ;           The first 4 characters entered after the 'M' set the
210                     ;           current address pointer.  Each pair of characters entered
211                     ;           after the address changes the byte at the current address.
212                     ;           A space entered after a byte change increments the current
213                     ;           address pointer.  A CR will end the command, and
214                     ;           can be entered after the initial address or any number of
215                     ;           byte changes.
216                     
217                     ; >         Increment the current address and display the contents.
218                     ;           (The M command, above, can set the current address pointer.)
219                     
220                     ; <         Decrement the current address and display the contents.
221                     ;           (The M command, above, can set the current address pointer.)
222                     
223                     ; SPACE     Uses the current addres spointer and displays the contents.
224                     
225                     ; C         Displays a checksum of memory from start address to end address.
226                     ;           Format is C SSSS EEEE, where SSSS is start address and EEEE
227                     ;           is the end address.
228                     
229                     ; D         Displays memory from start address to end address.
230                     ;           Format is D SSSS EEEE, where SSSS is the start address...
231                     
232                     ; T         Reads current time of day clock and displays the results.
233                     
234                     ; X         Switches the system from hardware handshake to XON-XOFF.  This
235                     ;           is a toggle command, each time X is entered, the system
236                     ;           switches.  The flag is displayed after each toggle.  The
237                     ;           flag is zero for hardware handshake.
238                     ;
239                     ;
240                     ; A         Alter registers in the order PC F A X Y SP.  A space skips
241                     ;           to the next register.  A CR ends the command.
242                     ;
243                     ; F         Fill memory from start address to end address with value.
244                     ;           The format is F SSSS EEEE VV.
245                     
246                     ; V         Move a block of memory from start address to end address.
247                     ;           The format is V SSSS EEEE XX.  SSSS is the start address,
248                     ;           EEEE is the end address, and XX is the number of bytes.
249                     ;           if XX = 0, 256 bytes are moved.
250                         .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MONTXT.ASM - Monitor Commands Defined'

```
                ;                       Monitor Commands, cont.
                ;
                ; G (CR)    Begin execution from the current address in the PC.
                ;
                ; G ADDR    Set PC to ADDR and begin execution.
                ;
                ; J (CR)    Do a JSR to the current PC address
                ;
                ; J ADDR    Do a JSR to ADDR.
                ;
                ; U         Jump through the USR command vector.  (This is a hook to allow
                ;           additional commands to be added to the monitor.
                ;
                ; B         Jumps to BASIC cold start.
                ;
                ; K         Jumps to BASIC warm start.
                ;
                ; I/O COMMANDS
                ; ------------------------
                ;
                ; S         Start of a data record in Motorola S28 or S19 format.  When
                ;           this command is received, data is not echoed until a CR is
                ;           received.  This command is used to load programs, etc and
                ;           the Error (E) command should be used after loading a number
                ;           of S28 records to check for data errors.
                ;
                ; W         Output data in Motorola S28 format.  This command outputs 16
                ;           byte records (the last record may be less) from start address
                ;           to end address.  The format is W SSSS EEEE
                ;
                ; E         Display the number of S28 receive errors noticed.  While 'S'
                ;           records are received, an accumulation of checksum errors is
                ;           kept.  This command displays that accumulation.  Once
                ;           displayed, the error number is cleared to 0.
                ;
                        .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MONTXT.ASM - Monitor Commands Defined'

```
288                     ;                       Monitor Operation:
289                     ;
290                     ;               The monitor is entered on power up and on a BRK instruction.
291                     ;                Generally, if code goes wild, it will eventually hit a BRK'
292                     ;                and return to the monitor.  At that time, the registers are
293                     ;                displayed and the monitor commands are available to the user.
294                     ;
295                     ;               Interrupts are still running, though, and a bad interrupt can
296                     ;                disable the monitor functions.
297                     ;
298                     ;               Code debugging can be accomplished by placing BRK (00)
299                     ;                instructions in the code, entering the monitor, and examining
300                     ;                memory and registers.  Be careful with the BRK instructions;
301                     ;                if you replace a normal instruction with a BRK you cannot
302                     ;                continue until you return the original instruction to that
303                     ;                location and fix the PC accordingly.
304                     ;
305                     ;               When a BRK instruction has occurred, the PC in the monitor will
306                     ;                be pointing at the location AFTER the BRK instruction.  This
307                     ;                means that if the BRK is placed in the code by an assembler,
308                     ;                and the next instruction follows the BRK, then you can
309                     ;                continue execution simply by hitting 'G' followed be a CR.
310
311                     ;
312                     ;
313                     ;                       INTERRUPTS
314                     ;
315                     ;               The monitor handles all interrupts.  In many cases, this is
316                     ;                done by having a second vector in RAM.  When the monitor
317                     ;                is started, it writes the RAM locations with pointers to its
318                     ;                own interrupt routines.  Unused interrupts simply jump to
319                     ;                RESET.  The user can revector his interrupts by changing
320                     ;                the RAM vector.
321                     ;
322                     ;               To save memory, most of the edge interrupts jump through the
323                     ;                same vector.  This defeats the purpose of having all those
324                     ;                individual vectors in the first place, but then we only have
325                     ;                a limited amount of on chip RAM.
326                     ;
327                     ;               The result is that for the interrupts sharing the common vector,
328                     ;                the interrupt routine must be able to identify which interrupt
329                     ;                was received. See the code for more details.
330
331
332
333  00:0000                    INCLUDE MONZPEQU.ASM
334                         .STTL 'MONZPEQU.ASM - ZP Equates for the $F000 Monitor'
335                         .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MONZPEQU.ASM - ZP Equates for the $F000 Monitor'

```
336                     ;06-21-1993
337
338
339     00:000F         LOWNIB   EQU $0F
340     00:00F0         HINIB    EQU $F0
341     00:0013         S28BN    EQU 19       ;16 + 2 FOR ADDR
342                         ; + 1 FOR CKSUM
343     00:0011         XON      EQU $11      ;DC1/^Q
344     00:0013         XOFF     EQU $13      ;DC3/^S
345     00:0003         CNTRLC   EQU $03
346     00:0018         CNTRLX   EQU $18
347
348
349                 ;65C134 INPUT/OUTPUT ADDRESSES
350
351     00:0030         PD0      EQU $0030
352     00:0031         PD1      EQU $0031
353     00:0032         PD2      EQU $0032
354     00:0034         PDD0     EQU $0034
355     00:0035         PDD1     EQU $0035
356     00:0036         PDD2     EQU $0036
357
358
359     00:0003         PD3      EQU $0003
360
361
362     00:0007         PCS3     EQU $0007     ;Output pin or Memory SEL
363
364
365     00:0008         IFR2     EQU $0008
366     00:0009         IER2     EQU $0009     ;IFR2 INTERRUPT ENABLE
367     00:0010         T1FLG    EQU $10
368     00:0020         T2FLG    EQU $20
369     00:0040         IRQ1FLG  EQU $40
370     00:0080         IRQ2FLG  EQU $80
371
372     00:000A         TCR1     EQU $000A
373     00:000B         TCR2     EQU $000B
374
375     00:000C         T1LL     EQU $000C     ;TIMER 1 LATCH LOW
376     00:000D         T1LH     EQU $000D     ;TIMER 1 LATCH HIGH
377     00:000E         T2LL     EQU $000E     ;TIMER 2 LATCH LOW
378     00:000F         T2LH     EQU $000F     ;TIMER 2 LATCH HIGH
379
380     00:0010         T1CL     EQU $0010     ;TIMER 1 COUNTER LOW
381     00:0011         T1CH     EQU $0011     ;TIMER 1 COUNTER HIGH
382     00:0012         T2CL     EQU $0012     ;TIMER 2 COUNTER LOW
383     00:0013         T2CH     EQU $0013     ;TIMER 2 COUNTER HIGH
384
385                 ;RESERVED
386
387     00:001B         BCR      EQU $001B     ;BUS CNTRL REG
388                         ;BIT 0-EXTERNAL MEM BUS ENABLE
389                         ;BIT 1-PORT 44-47 EDGE SENS IRQ
390                         ;BIT 2-ALWAYS 0
```

```
391                     ;BIT 3-ICE ENABLE=1
392                     ;BIT 4-PORT 50-53 EDGE SENS IRQ
```

'W65C134 Internal ROM Monitor ($F000)'
'MONZPEQU.ASM - ZP Equates for the $F000 Monitor'

```
393                             ;BIT 5-PORT 54-57 EDGE SENS IRQ
394                             ;BIT 6-NMI,IRQ1,IRQ2 ENABLE = 1
395                             ;BIT 7-EXTERNAL $F000-$FFFF = 1
396
397     00:001C             PD4     EQU $001C
398                             ;BIT 7-DATA SET READY NE47 IRQ (INPUT)
399
400     00:001D             PD5     EQU $001D
401
402     00:0020             PD6     EQU $0020
403                             ;BIT 0-RXD (INPUT)
404                             ;BIT 1-TXD (OUTPUT)
405                             ;BIT 2-DATA TERMINAL READY (OUTPUT)
406     00:0004             DTR     EQU $04
407
408     00:001E             PDD4    EQU $001E
409     00:001F             PDD5    EQU $001F
410     00:0021             PDD6    EQU $0021
411
412     00:0022             ACSR    EQU $0022     ;ACI CONTROL
413                             ;BIT 0-XMIT PORT ENABLE
414                             ;BIT 1-XMIT IRQ SOURCE
415                             ;BIT 2-7/8 BIT DATA
416                             ;BIT 3-PARITY ENABLE
417                             ;BIT 4-ODD/EVEN PARITY
418                             ;BIT 5-RECV ENABLE
419                             ;BIT 6-SOFTWARE SEMIPHORE
420                             ;BIT 7-RECV ERROR FLG
421     00:0001             SON     EQU $01
422
423
424     00:0023             ARTD    EQU $0023     ;ACI XMIT/RECV DATA REG
425
426
427     00:0024             TALL    EQU $0024     ;TIMER A LATCH LOW
428     00:0025             TALH    EQU $0025     ;
429     00:0026             TACL    EQU $0026     ;TIMER A COUNTER LOW
430     00:0027             TACH    EQU $0027     ;
431
432     00:0028             TMLL    EQU $0028     ;TIMER M LATCH LOW
433     00:0029             TMLH    EQU $0029     ;TIMER M LATCH HIGH
434     00:002A             TMCL    EQU $002A     ;TIMER M COUNTER LOW
435     00:002B             TMCH    EQU $002B     ;TIMER M COUNTER HIGH
436
437     00:002C             IFR1    EQU $002C     ;INTERRUPT FLG REG 1
438     00:002D             IER1    EQU $002D     ;INTERRUPT ENABLE REG 1
439
440
441                     .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MONZPEQU.ASM - ZP Equates for the $F000 Monitor'

```
442  00:0000                    .PAGE0
443  00:0040                    .ORG $40
444
445  00:0040              UBRK     .ds 2        ;USER BREAK
446  00:0042              UNMI     .ds 2        ;USER NMI VECTOR
447  00:0044              UIRQ2    .ds 2        ;USER IRQ VECTOR
448  00:0046              UIRQ1    .ds 2        ;USER IRQ VECTOR
449  00:0048              UIRQT2   .ds 2        ;USER IRQ TIMER VECTOR
450  00:004A              UIRQT1   .ds 2        ;USER IRQ TIMER VECTOR
451  00:004C              UNE46    .ds 2        ;USER NE46 VECTOR
452  00:004E              UGENIRQ  .ds 2        ;USER GENERAL IRQ VECTORS
453       00:0050         UIRQEND  EQU *
454
455  00:0050              UCMDPTR  .ds 2        ;USER COMMAND PROCESSOR
456                           ;IE 'U'X COMMANDS DEFINED
457                           ;BY THE USER AND HIS CODE
458
459  00:0052              SINPTR   .ds 2        ;SERIAL INPUT BUFFER START ADDR PTR
460  00:0054              SOUTPTR  .ds 2        ;SERIAL OUTPUT BUFFER START ADDR PTR
461  00:0056              SINCNT   .ds 1        ;SERIAL INPUT BUFFER SIZE COUNT 3-255
462  00:0057              SOUTCNT  .ds 1        ;SERIAL OUTPUT BUFFER SIZE COUNT 3-255
463
464  00:0058              PCL      .ds 1        ;PROGRAM COUNTER LOW
465  00:0059              PCH      .ds 1        ;PROGRAM COUNTER HIGH
466  00:005A              FLGS     .ds 1        ;CONDITIONAL CODE REG
467                           ;BIT 0--CARRY BIT
468                           ;BIT 1--ZERO BIT
469                           ;BIT 2--INTERRUPT ENABLE BIT
470                           ;BIT 3--BINARY CODED DECIMAL
471                           ;BIT 4--BREAK
472                           ;BIT 5--NU
473                           ;BIT 6--OVERFLOW
474                           ;BIT 7--MINUS
475  00:005B              ACC      .ds 1        ;TEMP ACC REG
476  00:005C              XR       .ds 1        ;TEMP X REG
477  00:005D              YREG     .ds 1        ;TEMP Y REG
478  00:005E              TMPSP    .ds 1        ;TEMP STACK POINTER
479  00:005F              TMPC     .ds 1        ;COUNT DOWN CTR FOR S28
480  00:0060              TODCKS   .ds 1        ;IRQ TEMP REG FOR ACC
481                           ;BEFORE CKING BRK CMD
482
483
484
485  00:0061              H100HZ   .ds 1        ;1000HZ NMI
486
487  00:0062              TENTHSEC .ds 1        ;0.1 SEC
488  00:0063              SEC      .ds 1        ;SECONDS   0 TO 59
489  00:0064              MIN      .ds 1        ;MINUTES   0 TO 59
490  00:0065              HR       .ds 1        ;HOURS     0 TO 23
491  00:0066              DAY      .ds 1        ;DAY       1 TO 28,29,30,31
492  00:0067              MONTH    .ds 1        ;MONTH 1= JAN 12= DEC
493  00:0068              YR       .ds 1        ;88
494  00:0069              DAYWK    .ds 1        ;DAY OF WEEK 1 = SUNDAY
495                           ;7 = SATURDAY
496  00:006A              DAYLIT   .ds 1        ;DAY LIGHT SAVINGS TIME
```

```
497                             ;BIT 0--ENABLED =1
498                             ;BIT 1-6 NU
```

'W65C134 Internal ROM Monitor ($F000)'
'MONZPEQU.ASM - ZP Equates for the $F000 Monitor'

```
499                             ;BIT 7--IN PROCESS OF
500                             ;       MODIFYING TOD
501     00:0001         DAYLITFLG EQU $01
502     00:0080         DAYLPROG EQU $80
503
504                             ;ALARM VARIABLES
505 00:006B             ASEC    .ds 1       ;SECONDS   0 TO 59
506 00:006C             AMIN    .ds 1       ;MINUTES   0 TO 59
507 00:006D             AHR     .ds 1       ;HOURS     0 TO 23
508 00:006E             ADAY    .ds 1       ;DAY       1 TO 28,29,30,31
509 00:006F             AMONTH  .ds 1       ;MONTH 1= JAN 12= DEC
510 00:0070             AYR     .ds 1       ;88
511 00:0071             ADAYWK  .ds 1       ;DAY OF WEEK 1 = SUNDAY
512                             ;7 = SATURDAY
513 00:0072             SFLAG   .ds 1       ;SERIAL RS-232 FLAGS
514                             ;BIT 0-SERIAL INPUT QUEUE DATA
515                             ;BIT 1-CONTROL 'C' RECEIVED, FLUSH QUEUE
516                             ;BIT 2-XON/XOFF CONTROL USED=1
517                             ;BIT 3-XON/XOFF OR HDW HS SEND OVERFLOW
518                             ;BIT 4-LAST CNTRL CHAR WAS XON=1 XOFF=0
519                             ;BIT 5-ECHO ON/OFF FLAG OFF=1
520                             ;BIT 6-OUTPUT XOFF
521                             ;BIT 7-OUTPUT XON
522     00:0001         SFLG    EQU $01
523     00:0002         CFLG    EQU $02
524     00:0004         XONOFLG EQU $04
525     00:0008         SNDOVF  EQU $08
526     00:0010         LASTXONOF EQU $10
527     00:0020         ECHOFF  EQU $20
528     00:0040         SXOFFLG EQU $40
529     00:0080         SXONFLG EQU $80
530
531 00:0073             SOUTINDX .ds 1      ;INDEX TO OUTPUT SERIAL RS-232C QUEUE
532 00:0074             SOUTEND  .ds 1      ;end of output queue
533 00:0075             SININDX  .ds 1      ;INDEX TO INPUT SERIAL RS-232C QUEUE
534 00:0076             SINEND   .ds 1      ;end of input queue
535
536 00:0077             DISPTYP  .ds 1      ;DISPLAY TYPE AND IF
537                             ;TOD DISPLAY IS ON
538                             ;BIT 0-2 TYPE OF DISPLAY
539                             ;BIT 3-ALARM ENABLE BIT
540                             ;BIT 4-ALARM INTERRUPT HAPPENED
541                             ;BIT 5-POWER UP IN PROGRESS FLG
542                             ;BIT 6-DISPLAY NOT WORKING
543                             ;BIT 7-TOD ON DISPLAY FLG
544
545     00:0008         ALRMENAB EQU $08    ;Enable the ALARM function
546     00:0010         ALRMIRQ EQU $10     ;ALARM INTERRUPT
547     00:0020         PUFLG   EQU $20     ;POWER UP
548
549
550 00:0078             WRAP    .ds 1       ;$FFFF WRAP AROUND
551
552 00:0079             DIFF    .ds 2       ;EA-SA = DIFF & Y REG HAS HIGH MSB
553 00:007B             TMP0    .ds 3       ;START ADDR
```

```
554  00:007E                  TMP2    .ds 3
555  00:0081                  TMP4    .ds 2
```

'W65C134 Internal ROM Monitor ($F000)'
'MONZPEQU.ASM - ZP Equates for the $F000 Monitor'

```
556  00:0083                    TMP6    .ds 2
557  00:0085                    ERRORS  .ds 1        ;S28 DOWNLOAD ERROR COUNT
558  00:0086                    SPEED   .ds 1        ;MAIN XTAL SPEED
559                                 ;0 = 2.000000MHZ
560                                 ;1 = 4.000000MHZ
561                                 ;2 = 2.457600MHZ
562                                 ;3 = 3.686400MHZ
563                                 ;4 = 1.843200MHZ
564                                 ;5 = 4.914   MHZ
565  00:0087
566       00:007E               DEST    EQU TMP2     ;DESTINATION ADDR
567       00:007B               SRCE    EQU TMP0     ;SOURCE ADDR
568       00:0083               TEMP    EQU TMP6
569
570                        .ENDS        ;Ends page 0 declarations
571
572  00:0000                        .END
573
574
575
576  00:8000                        ORG $8000
577       00:0000               IROM    EQU 0        ;defines IROM for conditional assembly to Internal ROM
578  00:8000  57 44 43 00           BYTE 'WDC',00
579  00:8004  A9 81                 LDA #$81
580  00:8006  85 1B                 STA BCR
581  00:8008  A9 C1                 LDA #$C1
582  00:800A  85 07                 STA PCS3
583  00:800C  4C F5 F0              JMP NOEXTROM
584
585
586  00:F000                        ORG $F000
587  00:F000                        INCLUDE MONJMP.ASM
588                        .STTL 'MONJMP.ASM - Monitor JMP Table'
589                        .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MONJMP.ASM - Monitor JMP Table'

```
590                         ;06-21-1993
591
592
593 00:F000                         .ORG $F000
594
595
596 00:F000  4C B0 F6               JMP VERSION    ;GET MONITOR VERSION
597                         ; Returns monitor version in A, month in X, day in Y.
598
599 00:F003  4C 52 FD               JMP ACI_INIT   ;INITIALIZES ACIA
600                         ; Initializes ACI to baud rate in A, data length in X,
601                         ; and parity in Y.
602                         ; Anything in the current buffers is flushed.
603                         ; Baud rate values are:
604
605                         ; A   Baud rate      A       Baud rate
606                         ;---  ---------     ---      ---------
607                         ; 0       75        6         1800
608                         ; 1      110        7         2400
609                         ; 2      150        8         4800
610                         ; 3      300        9         9600
611                         ; 4      600        A         19200
612                         ; 5     1200        B         38400
613
614                         ; X has data length 7 = 7 bits, 8 = 8 bits
615                         ; Y has parity.  B0 = 1 enables, B1 = 1 sets even parity.
616                         ; B1=0 sets odd parity.
617
618 00:F006  4C 8F FC               JMP RD_CHAR    ;RETURNS A CHARACTER
619                         ; from ACIA if one was present.  Otherwise it will return
620                         ; a 00 (null). CY = 1 if ^C is encountered and control C
621                         ; flag reset (see CK_CONTC)
622
623 00:F009  4C D8 FC               JMP CK_CONTC   ;RETURNS WITH C BIT SET if a control "C" has been
624                         ;detected on input else -C = 0
625
626 00:F00C  4C 71 FC               JMP GETCH      ;READ ACIA
627                         ; Wait until we get a character CY = 1 if ^C and
628                         ; acc = null ($00).
629                         ; C flag reset (see CK_CONTC) This routine also echos the
630                         ; character, this feature can be switched off. Echo is bit 5
631                         ; of SFLAG (addr 0072)
632
633 00:F00F  4C 11 FC               JMP OUTCH      ;WRITE ACIA
634                         ;ACC has the character (7-bit ASCII) to send CY = 1 if ^C
635
636 00:F012  4C F4 FB               JMP CRLF       ;PRINT CARRIAGE RETURN.
637                         ; ACC used
638
639 00:F015  4C 09 FC               JMP SPAC       ;PRINTS A SPACE
640                         ; ALL REG preserved
641
642 00:F018  4C 39 FD               JMP ASCBIN     ;ASCII TO BINARY
643
644 00:F01B  4C 59 FC               JMP BINASC     ;BINARY TO ASCII HEX
```

```
645                     ; ACC has lower nibble in ASCII, X has upper nibble in ASCII
646
```

'W65C134 Internal ROM Monitor ($F000)'
'MONJMP.ASM - Monitor JMP Table'

```
00:F01E  4C 06 F5            JMP PRTSTR     ;PRINTS A STRING
                      ; Useful for concatenating strings, and can embed CRLF
                      ; Y = # of bytes to send X = LO & ACC = HI point to text
                      ; CY = 1 if ^C
00:F021
00:F021  4C E0 FC            JMP RDOA       ;READS AN ADDRESS $XXXX
                      ; TMP0 =   starting address low TMP0+1 = starting address high
                      ; This routine also echos the character, this feature can be
                      ; switched off. Echo is bit 5 of SFLAG (addr 0072) and CY = 1
                      ; CY = 0 if SPACE. X reg not used, Y reg restored.
00:F024
00:F024  4C FD FC            JMP RDOB       ;READS A BYTE $ZZ
                      ; hex byte = ACC, and CY = 1 CY = 0 if SPACE
                      ; X reg not used, Y reg restored.  This routine also echos
                      ; the character; this feature can be switched off.
                      ; Echo is bit 5 of SFLAG (addr 0072)
00:F027
00:F027  4C D2 FB            JMP WR_ADDR    ;WRITE AN ADDRESS $XXXX
                      ; CY = 1 if ^C
00:F02A
00:F02A  4C EB FB            JMP WROB       ;WRITE A BYTE $XX
                      ; CY = 1 if ^C
00:F02D
00:F02D  4C 15 FE            JMP ISDECIMAL  ;CHECKS IF ASCII DECIMAL DIGIT
                      ;CY = 1 if not decimal digit
00:F030
00:F030  4C 1E FE            JMP IFASC      ;CHECKS IF ITS ASCII
                      ;CY = 1 if not ASCII
00:F033
00:F033  4C 25 FE            JMP UPPER_CASE ;CONVERT LOWER CASE ASCII TO UPPER CASE
                      ;ASCII character in ACC is converted to upper case ASCII.

00:F036
00:F036  4C EA FD            JMP MVDATA     ;MOVE DATA
                      ;TMP0,+1 =   Source address (Low, Hi)
                      ;TMP2,+1 =   Destination address low,hi
                      ;Y = # of bytes to be moved.
                      ;Y = 0 moves 256 bytes.
                      ;WARNING: TWO MEMORY AREAS MUST NOT OVER LAP
                      ; CY = 1 IF NO RAM AT DEST LOCATION
00:F039
00:F039  4C 18 F2            JMP START      ;This is used to get back to the command prompt
                      ;for the monitor.  It resets the stack from the memory location
                      ;that gets updated on the BRK command.
00:F03C
00:F03C  4C FB FD            JMP HEXIN      ;CONVERTS ASCII HEX TO
                      ;HEX/BINARY
                      ;returns hex/nibble binary in ACC
                      ;IF CARRY SET THEN NOT ASCII HEX
00:F03F
00:F03F  4C 31 FE            JMP BIN2DEC    ;Converts value in A from hex to packed decimal.
                      ;Do not exceed 99.
00:F042
00:F042  4C 8B F7            JMP MS19OUT    ;MOTOROLA S19 OUTPUT
                      ;TMP0 =   starting address low
```

```
702                     ;TMP0+1 = starting address high
703                     ;TMP2 =   ending address low
```

'W65C134 Internal ROM Monitor ($F000)'
'MONJMP.ASM - Monitor JMP Table'

```
                        ;TMP2+1 = ending address high
                        ;TMP6 =   Offset address low       Address field of S-Record
                        ;TMP6+1 = Offset address high      is data address + offset)
 00:F045
 00:F045  4C BA F6              JMP MS28IN     ;MOTOROLA S28 INPUT (Also S19)
                        ;
 00:F048
 00:F048  4C 55 F8              JMP CHK_SUM    ;CALCULATE CHECK SUM
                        ;TMP0 & TMP0+1 contain starting address
                        ;TMP2 & TMP2+1 contain ending address
                        ;TMP4 & TMP4+1 contains the sum of the digits to
                        ;facilitate either TWO's compliment or ONE's compliment checksum
                        ;Difference between TMP0 & TMP2 should not be GT 255
 00:F04B
 00:F04B  4C FC F9              JMP RD_CLOCK   ;You give this routine a pointer in A,X (A is low).
                        ; This routine puts the real time data in that location in the
                        ; following format:

                        ; Byte       Contents

                        ;  0        Seconds, in 2's complement.
                        ;  1        Minutes,       ""
                        ;  2        Hours (0-23)   ""
                        ;  3        Day of Month   ""
                        ;  4        Month (1-12)   ""
                        ;  5        Year           ""
                        ;  6        Weekday (1-7)  ""
 00:F04E
 00:F04E  4C 12 FA              JMP WR_CLOCK   ;You write a buffer with the time, in the format shown
                        ; above.  Put a pointer to the buffer in A,X and call this routine
                        ; to change the TOD clock.
 00:F051
 00:F051  4C E6 F9              JMP RTC_MODE   ;DAYLIGHT SAVINGS FLAG
                        ; Call this routine with ACC=1 and the RTC will make daylight
                        ; savings shifts when required. If called with ACC=0, no shifts will
                        ; be made. CY = 1 on error
 00:F054
 00:F054  4C 28 FA              JMP WR_ACLOCK  ;You write a buffer with the time, in the format
                        ; shown above.  Put a pointer to the buffer in A,X and call this
                        ; routine to change the alarm clock.
                        ; There is a WILD CARD of $FF for a match.

 00:F057  4C F5 F0              JMP NOEXTROM   ;Entry point written at $8000

 00:F05A  CB EA 60              .BYTE $CB,$EA,$60   ;WAIT RTS INTERNAL ROM


 00:F05D  4C 0B FE              JMP ISHEX      ;TESTS FOR VALID ASCII HEX DIGIT
                        ;CY = 1 if not ASCII HEX and returns char in UPPER case.


 00:F060  4C B4 FD              JMP FLUSH_SERIAL_BUFF ;SETUP QUEUE COUNTERS TO ZERO
                        ;IE FLUSH ALL SERIAL QUEUES

 00:F063                        .END
```

```
759
760  00:F063                INCLUDE MON1.ASM
```

'W65C134 Internal ROM Monitor ($F000)'
'MONJMP.ASM - Monitor JMP Table'

```
761                     .STTL 'MON1.ASM - Initial Code'
762                     .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON1.ASM - Initial Code'

```
763                         ;01-13-1995
764
765
766
767       00:F063           MONVER#  EQU *
768 00:F063  17 01 13 95          .BYTE $17,$01,$13,$95 ;Must also be altered in MON2.ASM
769                                 ;The log-on message has ASCII version
770       00:F067           WDC      EQU *
771 00:F067  57 44 43             .BYTE 'WDC'
772
773       00:F06A           MONIRQTBL EQU *       ;USER INTERRUPTS
774 00:F06A  FCF1                 .WORD MONBRK   ;UBRK
775 00:F06C  8FF0                 .WORD RESET    ;UNMI
776 00:F06E  8FF0                 .WORD RESET    ;UIRQ2
777 00:F070  88F0                 .WORD RESCN    ;UIRQ1
778 00:F072  F8F8                 .WORD TODIRQ   ;UIRQT2
779 00:F074  8FF0                 .WORD RESET    ;UIRQT1
780 00:F076  4BFE                 .WORD PDOWN    ;UNE46
781 00:F078  8FF0                 .WORD RESET    ;UGENIRQ
782 00:F07A  18F2                 .WORD START    ;UCMDPTR
783       00:F07C           MONIRQEND EQU *
784
785       00:F07C           MXTALCALC EQU *       ;MAX
786 00:F07C  C1                   .BYTE $C1     ;2MHZ
787 00:F07D  E1                   .BYTE $E1     ;4MHZ
788 00:F07E  CD                   .BYTE $CD     ;2.4576MHZ
789 00:F07F  DF                   .BYTE $DF     ;3.6864MHZ
790 00:F080  BD                   .BYTE $BD     ;1.8432MHZ
791 00:F081  E7                   .BYTE $E7     ;4.9125 MHZ
792       00:F082           MXTLEND  EQU *
793
794       00:F082           MNTALCALC EQU *       ;MIN
795 00:F082  C2                   .BYTE $C2     ;2MHZ
796 00:F083  E2                   .BYTE $E2     ;4MHZ
797 00:F084  CE                   .BYTE $CE     ;2.4576MHZ
798 00:F085  E0                   .BYTE $E0     ;3.6864MHZ
799 00:F086  BE                   .BYTE $BE     ;1.8432MHZ
800 00:F087  E8                   .BYTE $E8     ;4.9125 MHZ
801       00:F088           MNTLEND  EQU *
802
803
804 00:F088  48            RESCN    PHA
805 00:F089  A9 40                 LDA #$40     ;RESET IRQ1
806 00:F08B  04 08                 TSB IFR2
807 00:F08D  68                    PLA
808 00:F08E  40                    RTI
809
810
811
812       00:F08F           RESET    EQU *
813 00:F08F  78                    SEI
814 00:F090  D8                    CLD          ;CLEAR DECIMAL MODE
815 00:F091  A2 FF                 LDX #$FF
816 00:F093  9A                    TXS
817       [01]                     .IFDEF IROM
```

```
818  00:F094  64 07              STZ PCS3
819  00:F096  A9 01              LDA #$01      ;EXT BUSS BUT NO ICE MODE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON1.ASM - Initial Code'

```
820         [01]                        .ELSE
821                         LDA #$80      ;CS7  EPROM
822                         STA PCS3
823                         LDA #$89      ;EXT BUSS, ICE & EXT ROM
824         [00]                        .ENDIF
825  00:F098  85 1B                     STA BCR
826
827
828                     ;
829                     ;       Reset all regs to reset values (in case we had a JMP reset rather
830                     ;       than a hard reset.  Exception is TCR1 because the chip will die if
831                     ;       we switch to slow clock and shut off fast clock simultaniously,
832
833  00:F09A  64 08                     STZ IFR2
834  00:F09C  64 09                     STZ IER2
835  00:F09E  64 0B                     STZ TCR2
836  00:F0A0  64 1C                     STZ PD4
837  00:F0A2  64 1D                     STZ PD5
838  00:F0A4  64 1E                     STZ PDD4
839  00:F0A6  64 1F                     STZ PDD5
840  00:F0A8  64 20                     STZ PD6
841  00:F0AA  64 21                     STZ PDD6
842  00:F0AC  64 2C                     STZ IFR1
843  00:F0AE  64 2D                     STZ IER1
844  00:F0B0  A9 FF                     LDA #$FF
845  00:F0B2  85 03                     STA PD3
846  00:F0B4  A9 F9                     LDA #$F9
847  00:F0B6  14 0A                     TRB TCR1
848
849         [01]                        .IFDEF IROM
850  00:F0B8  A5 7B                     LDA $7B       ;see if we are in low power mode.
851  00:F0BA  C9 55                     CMP #$55
852  00:F0BC  D0 0F                     BNE NOLPWR
853  00:F0BE  A5 7C                     LDA $7C
854  00:F0C0  C9 AA                     CMP #$AA
855  00:F0C2  D0 09                     BNE NOLPWR
856  00:F0C4  A5 7D                     LDA $7D
857  00:F0C6  C9 88                     CMP #$88
858  00:F0C8  D0 03                     BNE NOLPWR
859
860  00:F0CA  6C 4C 00                  JMP (UNE46)   ;if we were, the vector is there.
861  00:F0CD
862         00:F0CD               NOLPWR   EQU *
863         [00]                        .ENDIF
864
865  00:F0CD  A9 C0                     LDA #$C0      ;ENABLE $8000 & $0200
866  00:F0CF  04 07                     TSB PCS3      ;SO WE CAN CK THE 'WDC'
867
868  00:F0D1  A2 00                     LDX #$00
869  00:F0D3  BD 00 80          CKHIROM  LDA $8000,X     ;CHK FOR EXTERNAL ROM
870  00:F0D6  DD 67 F0                  CMP WDC,X
871  00:F0D9  D0 08                     BNE CKLOWRAM
872  00:F0DB  E8                        INX
873  00:F0DC  E0 03                     CPX #3
874  00:F0DE  D0 F3                     BNE CKHIROM
```

```
875  00:F0E0  4C 04 80                JMP $8004      ;JMP TO EXTERNAL ROM
876
```

'W65C134 Internal ROM Monitor ($F000)'
'MON1.ASM - Initial Code'

```
877         00:F0E3             CKLOWRAM EQU *         ;CHK FOR LOW MEM ROM
878  00:F0E3  A2 00                    LDX #$00
879  00:F0E5  BD 00 02          LORAMLP  LDA $200,X
880  00:F0E8  DD 67 F0                  CMP WDC,X
881  00:F0EB  D0 08                     BNE NOEXTROM
882  00:F0ED  E8                        INX
883  00:F0EE  E0 03                     CPX #3
884  00:F0F0  D0 F3                     BNE LORAMLP
885  00:F0F2  4C 04 02                  JMP $204      ;JMP TO EXTERNAL ROM
886  00:F0F5
887
888  00:F0F5                    NOEXTROM:
889
890  00:F0F5  AD 00 02                  LDA $200      ;CHECK FOR RAM AT $200
891  00:F0F8  49 FF                     EOR #$FF
892  00:F0FA  8D 00 02                  STA $200
893  00:F0FD  CD 00 02                  CMP $200
894  00:F100  D0 0A                     BNE NOEXTRAM
895  00:F102  49 FF                     EOR #$FF
896  00:F104  8D 00 02                  STA $200
897  00:F107  CD 00 02                  CMP $200
898  00:F10A  F0 04                     BEQ EXTRAM
899
900  00:F10C                    NOEXTRAM:
901  00:F10C  A9 40                     LDA #$40      ;TURN OFF SRAM to keep stack
902  00:F10E  14 07                     TRB PCS3      ;INSIDE 65C134
903
904  00:F110                    EXTRAM:
905         [01]                        .IFNDEF IROM
906                              LDA $7B       ;see if we are in low power mode.
907                              CMP #$55
908                              BNE NOLPWR
909
910                              LDA $7C
911                              CMP #$AA
912                              BNE NOLPWR
913
914                              LDA $7D
915                              CMP #$88
916                              BNE NOLPWR
917
918                              JMP (UNE46)    ;if we were, the vector is there.
919
920                       NOLPWR   EQU *
921         [00]                        .ENDIF
922
923
924                       ;  START FAST CLOCK BUT NOT USING YET
925  00:F110  A9 04                     LDA #$04
926  00:F112  04 0A                     TSB TCR1
927
928  00:F114  A2 12                     LDX #MONIRQEND-MONIRQTBL ;SETUP ALL USER INTERRUPTS
929  00:F116  BD 69 F0          FUIRQS   LDA MONIRQTBL-1,X
930  00:F119  95 3F                     STA UBRK-1,X
931  00:F11B  CA                        DEX
```

```
932  00:F11C  D0 F8                BNE FUIRQS
933
```

'W65C134 Internal ROM Monitor ($F000)'
'MON1.ASM - Initial Code'

```
934  00:F11E  A2 00              LDX #$00
935  00:F120  CA            DLY0  DEX
936  00:F121  D0 FD              BNE DLY0
937  00:F123  A9 02              LDA #$02     ;ENABLE FAST CLOCK
938  00:F125  04 0A              TSB TCR1
939  00:F127  A9 04              LDA #$04     ;ENABLE NE46
940  00:F129  04 2D              TSB IER1     ;FOR POWER DOWN SENSE
941  00:F12B
942
943                              ;these are placed here to facilitate
944                              ;quickly changing for other projects
945                              ;near where they are used.
946       00:0004           XMITSIZ  EQU 4
947       00:0010           RECSIZ   EQU $10
948       00:0090           RECLOC   EQU $90
949
950  00:F12B  A9 00              LDA #<32768   ;SET TIMER 2 FOR
951  00:F12D  85 0E              STA T2LL      ;1 SECOND IRQ
952  00:F12F  A9 80              LDA #>32768
953  00:F131  85 0F              STA T2LH
954  00:F133  A9 20              LDA #$20      ;ENABLE TIMER 2 IRQS
955  00:F135  85 09              STA IER2
956  00:F137  A9 18              LDA #$18      ;TIMER 2 ENABLED
957  00:F139  85 0B              STA TCR2
958
959  00:F13B  A9 90              LDA #<RECLOC   ;SETUP SERIAL BUFFERS
960  00:F13D  85 52              STA SINPTR     ;IN PAGE ZERO
961  00:F13F  64 53              STZ SINPTR+1
962  00:F141  A9 A0              LDA #<RECLOC+RECSIZ
963  00:F143  85 54              STA SOUTPTR
964  00:F145  64 55              STZ SOUTPTR+1
965  00:F147  A9 10              LDA #RECSIZ    ;SIZE OF BUFFERS
966  00:F149  85 56              STA SINCNT
967  00:F14B  A9 04              LDA #XMITSIZ   ;SIZE OF XMIT
968  00:F14D  85 57              STA SOUTCNT
969
970                   ;
971                   ;
972                   ;  PATCH THIS BY STARTING HERE AND ENTERING
973                   ;
974                   ;  LDX #3
975                   ;  JMP MXTALFND
976                   ;
977                   ;   This deletes auto clock selection
978                   ;
979
980
981
982
983
984
985                        ;CK MAIN XTAL Frequency
986  00:F14F
987  00:F14F  A5 13         T2ZERO   LDA T2CH
988  00:F151  05 12              ORA T2CL      ;WAIT UNTIL TOD CLOCK
```

```
989  00:F153  D0 FA                BNE T2ZERO     ;READY TO LOAD
990  00:F155
```

'W65C134 Internal ROM Monitor ($F000)'
'MON1.ASM - Initial Code'

```
991  00:F155  A2 03              LDX #3
992  00:F157  3A            T2DELAY  DEC A        ;NOW WAIT A PREDETERMINED
993  00:F158  D0 FD              BNE T2DELAY   ;AMT OF TIME TO CALC XTAL
994  00:F15A  CA                 DEX
995  00:F15B  D0 FA              BNE T2DELAY
996
997  00:F15D  A5 12              LDA T2CL      ;CK A RANGE OF #'S
998
999  00:F15F  A2 06              LDX #MXTLEND-MXTALCALC
1000 00:F161  DD 7B F0      TRYMXTAL CMP MXTALCALC-1,X
1001 00:F164  F0 0F              BEQ MXTALFND
1002 00:F166  CA                 DEX
1003 00:F167  D0 F8              BNE TRYMXTAL
1004
1005 00:F169  A2 06              LDX #MNTLEND-MNTALCALC
1006 00:F16B  DD 81 F0      TRYMNTAL CMP MNTALCALC-1,X
1007 00:F16E  F0 05              BEQ MXTALFND
1008 00:F170  CA                 DEX
1009 00:F171  D0 F8              BNE TRYMNTAL
1010
1011
1012 00:F173  A2 03              LDX #3        ;DEFAULT 2.4576MHZ
1013 00:F175
1014
1015
1016         00:F175         MXTALFND EQU *
1017 00:F175  CA                 DEX
1018 00:F176  86 86              STX SPEED     ;SAVE MAIN XTAL SPEED
1019
1020 00:F178  A9 09              LDA #$09      ;9600 BAUD
1021 00:F17A  A2 08              LDX #8        ;8 BITS
1022 00:F17C  A0 02              LDY #%00000010 ;EVEN PARITY / BUT NO PARITY
1023 00:F17E  20 52 FD           JSR ACI_INIT   ;INITIALIZE ACIA.
1024                            ;hardware handshake is ON
1025
1026 00:F181  A2 07              LDX #DFLTSEND-DFLTS-1
1027 00:F183  A9 00              LDA #00
1028 00:F185  18                 CLC
1029 00:F186  75 62         CKTODLP  ADC SEC-1,X    ;CK IF VALID TOD CLOCK
1030 00:F188  CA                 DEX
1031 00:F189  D0 FB              BNE CKTODLP
1032 00:F18B
1033
1034 00:F18B  49 FF              EOR #$FF
1035 00:F18D  C5 60              CMP TODCKS
1036 00:F18F  F0 05              BEQ GDTOD
1037 00:F191  20 D8 F8      TODERR   JSR INITCLK    ;NOT VALID TOD CLOCK SO FILL
1038
1039 00:F194  80 0C              BRA GDTOD1
1040
1041         00:F196         GDTOD    EQU *
1042 00:F196  20 B4 FD           JSR FLUSH_SERIAL_BUFF
1043 00:F199  64 85              STZ ERRORS
1044 00:F19B  A9 FB              LDA #$FF-XONOFLG
1045 00:F19D  14 72              TRB SFLAG     ;reset all of SFLAG except XON/XOFF,
```

```
1046                            ;leave it in old state.
1047
```

'W65C134 Internal ROM Monitor ($F000)'
'MON1.ASM - Initial Code'

```
1048  00:F19F  20 91 FD          JSR SIOPORTS
1049
1050  00:F1A2  AD 00 E0    GDTOD1   LDA $E000     ;CHK IF WE HAVE E000 ROM
1051  00:F1A5  C9 4C              CMP #$4C
1052  00:F1A7  D0 03              BNE SKPE000
1053  00:F1A9  20 00 E0           JSR $E000
1054        00:F1AC           SKPE000  EQU *
1055  00:F1AC  A5 23              LDA ARTD      ;CLEAR SIO RECEIVE IRQS
1056  00:F1AE
1057  00:F1AE
1058
1059  00:F1AE                     .END
1060
1061  00:F1AE                     INCLUDE MON2.ASM
1062                      .STTL 'MON2.ASM - As we enable interrupts after RESET'
1063                      .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON2.ASM - As we enable interrupts after RESET'

```
1064                        ;01-13-1995
1065
1066  00:F1AE  58                   CLI         ;ENABLE INTERUPTS
1067                               ;WRITE MONITOR VERSION
1068
1069  00:F1AF  A9 F1                LDA #>MONVER
1070  00:F1B1  A2 BD                LDX #<MONVER
1071  00:F1B3  A0 3F                LDY #MONVEND-MONVER
1072  00:F1B5  20 06 F5             JSR PRTSTR
1073  00:F1B8  20 3E FA             JSR SNDTOD    ;DISPLAY TOD
1074  00:F1BB  00 00                BRK         ;ENTER COM BY BREAK
1075
1076
1077        00:F1BD             MONVER   EQU *
1078  00:F1BD  0D 31 33 34 20               .BYTE $0D,'134 ROM Version 1.07'  ;SEE MON1.ASM & MONTXT.ASM
    00F1C2  52 4F 4D 20 56
    00F1C7  65 72 73 69 6F
    00F1CC  6E 20 31 2E 30
    00F1D1  37
1079  00:F1D2  0D 28 43 29 20               .BYTE $0D,'(C) Copyright 1995'
    00F1D7  43 6F 70 79 72
    00F1DC  69 67 68 74 20
    00F1E1  31 39 39 35
1080  00:F1E5  0D 57 65 73 74               .BYTE $0D,'Western Design Center',$0D
    00F1EA  65 72 6E 20 44
    00F1EF  65 73 69 67 6E
    00F1F4  20 43 65 6E 74
    00F1F9  65 72 0D
1081        00:F1FC             MONVEND  EQU *
1082
1083                        .STTL 'MON2.ASM - BRK handler'
1084                        .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON2.ASM - BRK handler'

```
1085
1086
1087          00:F1FC               MONBRK   EQU *         ;MONITOR BREAK
1088  00:F1FC  85 5B                      STA ACC
1089  00:F1FE  86 5C                      STX XR        ;SAVE X
1090  00:F200  84 5D                      STY YREG      ;Y
1091  00:F202  68                         PLA
1092  00:F203  85 5A                      STA FLGS      ;FLAGS
1093  00:F205  18                         CLC           ;for the add below
1094  00:F206  68                         PLA
1095  00:F207  69 FF                      ADC #$FF      ;CY SET TO PC-1 FOR BR
1096  00:F209  85 58                      STA PCL
1097  00:F20B  68                         PLA
1098  00:F20C  69 FF                      ADC #$FF
1099  00:F20E  85 59                      STA PCH
1100  00:F210  BA                         TSX
1101  00:F211  86 5E                      STX TMPSP     ;SAVE ORIG SP
1102  00:F213  58                         CLI           ;ENABLE SERIAL OUTPUT
1103  00:F214  A9 52                      LDA #'R'      ;SET FOR R DISPLAY TO
1104                                    ;PERMIT IMMEDIATE ALTER
1105  00:F216  80 1B                      BRA S0        ;FOLLOWING BREAKPOINT.
1106  00:F218                             .END
1107
1108  00:F218                             INCLUDE MON3.ASM
1109                              .STTL 'MON3.ASM - Monitor BRK/Command Handler'
1110                              .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Monitor BRK/Command Handler'

```
1111                        ;06-23-1993
1112
1113  00:F218  A6 5E        START   LDX TMPSP     ;reset the SP to what it was when we hit the BRK
1114  00:F21A  9A                   TXS
1115  00:F21B  A9 20                LDA #ECHOFF   ;TURN ON ECHO MODE
1116  00:F21D  14 72                TRB SFLAG
1117  00:F21F  20 D8 FC             JSR CK_CONTC  ;CHECK & CLR CONTROL C
1118  00:F222  64 78                STZ WRAP      ;indicates we are past $FFFF
1119  00:F224  20 F4 FB     S000    JSR CRLF
1120  00:F227  A9 2E                LDA #'.'      ;TYPE PROMPTING '.'
1121  00:F229  20 11 FC             JSR OUTCH
1122  00:F22C  B0 EA                BCS START
1123  00:F22E  20 71 FC     S00     JSR GETCH
1124  00:F231  B0 E5                BCS START
1125                    ;
1126                    ;     Raw character is in A.  May be wrong case, etc.  We will JSR to
1127                    ;     alt. parsing if it exists at this point.  The alternate parser
1128                    ;     will jump through the vector table to START if it completes the
1129                    ;     command, and will do an RTS to the regular parser if it does not
1130                    ;     have the command in its table.
1131
1132  00:F233  AE FD EF     S0      LDX $EFFD     ;This location has a JMP to an external parser
1133  00:F236  E0 4C                CPX #$4C      ;if an alternate parser exists
1134  00:F238  D0 03                BNE DFLTPRSR
1135
1136  00:F23A  20 FD EF             JSR $EFFD     ;do the JSR to alternate parser
1137                               ;and then continue to do our standard parsing
1138
1139  00:F23D  C9 0A        DFLTPRSR CMP #$0A
1140  00:F23F  F0 ED                BEQ S00       ;IGNORE LF
1141  00:F241  20 25 FE             JSR UPPER_CASE ;IN ACC/ MAKE SURE
1142                              ;UPPERCASE
1143  00:F244  A2 16                LDX #ADRS-CMDS-1    ;LENGTH OF CMD TABLE
1144  00:F246  DD 67 F2     S1      CMP CMDS,X
1145  00:F249  D0 0D                BNE S2
1146  00:F24B  8A                   TXA
1147  00:F24C  0A                   ASL A         ;X2
1148  00:F24D  AA                   TAX
1149  00:F24E  E0 0A                CPX #TWOSCMD-ADRS   ;IF :, M,<, >,SPACE
1150  00:F250  B0 03                BCS IJMP      ; SPACE 2
1151  00:F252  20 04 FC             JSR SPAC2
1152  00:F255  7C 7E F2     IJMP    JMP (ADRS,X)
1153
1154  00:F258  CA           S2      DEX
1155  00:F259  10 EB                BPL S1        ;LOOP FOR ALL CMDS
1156
1157  00:F25B  A9 3F        ERROPR  LDA #'?'      ;OPERATOR ERR, TYPE'?'
1158  00:F25D  20 11 FC             JSR OUTCH     ; &
1159  00:F260  80 C2                BRA S000      ;send out a "." and try again
1160
1161  00:F262  20 AC F2     UCMD    JSR UCMD1
1162  00:F265  80 B1                BRA START
1163
1164                        .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Monitor BRK/Command Handler'

```
1165
1166
1167                    ;SINGLE LETTER COMMANDS USED ARE:
1168                    ;A,B,C,D,E,F,G,H,J,K
1169                    ;M,R,S,T,U,V,W,X
1170                    ;<,>, ,?,/
1171
1172  00:F267  41               CMDS    .BYTE 'A'     ;ALTER REGISTERS
1173  00:F268  4D                       .BYTE 'M'     ;CHANGE A MEMORY LOC
1174  00:F269  3C                       .BYTE '<'     ;DEC TO NXT MEMORY LOC
1175  00:F26A  3E                       .BYTE '>'     ;INC TO NXT MEMORY LOC
1176  00:F26B  20                       .BYTE ' '     ;REDISPLAY OLD LOCATION
1177  00:F26C  52                       .BYTE 'R'     ;DISPLAY REGISTERS
1178  00:F26D  47                       .BYTE 'G'     ;GO/JMP
1179  00:F26E  4A                       .BYTE 'J'     ;JSR
1180  00:F26F  44                       .BYTE 'D'     ;DUMP MEMORY IN HEX
1181  00:F270  46                       .BYTE 'F'     ;FILL MEMORY
1182  00:F271  43                       .BYTE 'C'     ;CHECK SUM
1183  00:F272  56                       .BYTE 'V'     ;MOVE A BLOCK
1184  00:F273  3F                       .BYTE '?'     ;HELP MENU
1185  00:F274  48                       .BYTE 'H'     ;HELP MENU
1186  00:F275  54                       .BYTE 'T'     ;DISPLAY TIME
1187  00:F276  58                       .BYTE 'X'     ;TOGGLE XON/XOFF MODE
1188  00:F277  53                       .BYTE 'S'     ;S28 LOADER FROM MONITOR
1189  00:F278  57                       .BYTE 'W'     ;S28 DUMPER
1190  00:F279  45                       .BYTE 'E'     ;PRINT ERRORS
1191  00:F27A  55                       .BYTE 'U'     ;USER DEFINED COMMANDS
1192  00:F27B  42                       .BYTE 'B'     ;BASIC COLD START
1193  00:F27C  4B                       .BYTE 'K'     ;BASIC WARM START & CONTINUE
1194  00:F27D  2F                       .BYTE '/'     ;QUICK ACCESS TO MEM FOR HOSTS
1195
1196  00:F27E  C0F3             ADRS    .WORD ALTER    ;CHANGE CURRENT REGS
1197  00:F280  11F3                     .WORD ALTERM   ;ALTER A MEMORY LOC
1198  00:F282  5AF3                     .WORD DSPLYDEC ;DEC ADDR & DISPLAY
1199  00:F284  55F3                     .WORD DSPLYINC ;INC ADDR & DISPLAY
1200  00:F286  64F3                     .WORD DSPLYOLD ;DISPLAY CURRENT ADDR
1201
1202         00:F288              TWOSCMD  EQU *         ;END OF AREA NEEDING
1203                                  ;TWO (2) SPACES
1204  00:F288  D6F2                     .WORD DSPLYR   ;DISPLAY REGS
1205  00:F28A  DBF3                     .WORD GO       ;GOTO/JMP
1206  00:F28C  F4F3                     .WORD GOJSR    ;JSR/JUMP TO SUBROUTINE
1207  00:F28E  11F4                     .WORD WM       ;DUMP MEMORY IN HEX
1208  00:F290  BBF4                     .WORD FILL     ;FILL MEMORY WITH A CONSTANT
1209  00:F292  46F8                     .WORD CHKSUM   ;CALC CHECK SUM
1210  00:F294  6CF8                     .WORD MOVE     ;MOVE A BLOCK OF UP TO 256 BYTES
1211  00:F296  EFF4                     .WORD HELP     ;HELP MENU
1212  00:F298  EFF4                     .WORD HELP     ;HELP MENU
1213  00:F29A  97F8                     .WORD DTIME    ;DISPLAY TIME OF DAY
1214  00:F29C  80F8                     .WORD TGLXONXOFF ;TOGGLE SERIAL XON/XOFF MODEF
1215  00:F29E  C1F6                     .WORD LSS      ;MOTOROLA S28 LOADER FROM MONITOR
1216  00:F2A0  76F7                     .WORD WO       ;MOTOROLA S28 DUMP
1217  00:F2A2  4AF7                     .WORD PERR     ;PRINT ERRORS
1218  00:F2A4  62F2                     .WORD UCMD     ;USER DEFINED CMDS
1219  00:F2A6  AFF2                     .WORD CBASIC   ;COLD START FOR BASIC
```

```
1220  00:F2A8  C1F2                .WORD KBASIC   ;WARM START BASIC & CONTINUE
1221  00:F2AA  9FF8                .WORD SLASH    ;HOST MEMORY ACCESS
```

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Monitor BRK/Command Handler'

```
1222
1223
1224
1225  00:F2AC  6C 50 00       UCMD1   JMP (UCMDPTR)  ;GOTO USER COMMANDS
1226
1227
1228  00:F2AF  AD 00 A0       CBASIC  LDA $A000     ;CK TO SEE IF BASIC THERE
1229  00:F2B2  C9 4C                  CMP #$4C
1230  00:F2B4  F0 03                  BEQ CBAS1
1231  00:F2B6  4C 18 F2               JMP START
1232
1233  00:F2B9  A9 A0          CBAS1   LDA #>$A000
1234  00:F2BB  85 59                  STA PCH
1235  00:F2BD  A9 00                  LDA #<$A000
1236  00:F2BF  80 10                  BRA GBAS
1237
1238
1239  00:F2C1  AD 03 A0       KBASIC  LDA $A003     ;CK TO SEE IF BASIC THERE
1240  00:F2C4  C9 4C                  CMP #$4C
1241  00:F2C6  F0 03                  BEQ KBAS1
1242  00:F2C8  4C 18 F2               JMP START
1243
1244  00:F2CB  A9 A0          KBAS1   LDA #>$A003
1245  00:F2CD  85 59                  STA PCH
1246  00:F2CF  A9 03                  LDA #<$A003
1247  00:F2D1  85 58          GBAS    STA PCL
1248  00:F2D3  4C DE F3               JMP GO1
1249
1250                        .STTL 'MON3.ASM - Commands, Display Registers'
1251                        .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Commands, Display Registers'

```
1252
1253
1254                    ;* Routine:  DSPLYR  DISPLAY REG CMD -PC,F,A,X,Y, and SP
1255                    ;*
1256                    ;* Reg Used: ACC,Y,X
1257                    ;* Var Used: TMPC,TMP0
1258                    ;* Routines Called: REGTTL,WRPC,SETR,SPAC,WROB
1259                    ;* Returned Reg: NONE
1260                    ;*
1261
1262  00:F2D6  20 F3 F2         DSPLYR   JSR REGTTL
1263  00:F2D9  20 DE FB                  JSR WRPC      ;WRITE Program Counter
1264  00:F2DC  20 B7 F3                  JSR SETR
1265  00:F2DF  85 5F                     STA TMPC
1266  00:F2E1  A0 00                     LDY #0
1267  00:F2E3  20 09 FC         R1       JSR SPAC
1268  00:F2E6  B1 7B                     LDA (TMP0),Y   ;DISPLAY 5 REGS
1269  00:F2E8  20 EB FB                  JSR WROB
1270  00:F2EB  C8                        INY
1271  00:F2EC  C6 5F                     DEC TMPC
1272  00:F2EE  D0 F3                     BNE R1
1273  00:F2F0  4C 18 F2         M1       JMP START
1274
1275
1276  00:F2F3  A9 F2             REGTTL   LDA #>REGSTR
1277  00:F2F5  A2 FC                     LDX #<REGSTR
1278  00:F2F7  A0 15                     LDY #REGSEND-REGSTR
1279  00:F2F9  4C 06 F5                  JMP PRTSTR
1280
1281
1282  00:F2FC  0D 41 44 44 52               REGSTR   BYTE $0D,'ADDR F  A  X  Y  SP',$0D
    00F301  20 46 20 20 41
    00F306  20 20 58 20 20
    00F30B  59 20 20 53 50
    00F310  0D
1283  00:F311                    REGSEND
1284
1285                    ;* Routine:  ALTERM
1286                    ;*
1287                    ;* Reg Used: ACC,Y,X
1288                    ;* Var Used: TMP0
1289                    ;* Routines Called: RDOA,SPAC,BYTE
1290                    ;* Returned Reg: NONE
1291                    ;*
1292
1293  00:F311  20 E0 FC         ALTERM   JSR RDOA      ;READ MEM ADDR INTO TMP0
1294  00:F314  B0 DA                     BCS M1        ;ERR IF NO ADDR
1295
1296  00:F316  A5 7C                     LDA TMP0+1
1297  00:F318  48                        PHA           ;push the starting address; we'll need it later
1298  00:F319  A5 7B                     LDA TMP0      ;Display 16 bytes starting at given address
1299  00:F31B  48                        PHA
1300  00:F31C  18                        CLC
1301  00:F31D  69 0F                     ADC #15
1302  00:F31F  85 7E                     STA TMP2
```

```
1303  00:F321  A9 00        LDA #0
1304  00:F323  65 7C        ADC TMP0+1
```

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Commands, Display Registers'

```
1305  00:F325  85 7F          STA TMP2+1
1306  00:F327  20 1A F4       JSR WRROUT     ;this routine messes up the starting address
1307
1308  00:F32A  68             PLA
1309  00:F32B  85 7B          STA TMP0       ;bring back the starting address
1310  00:F32D  68             PLA
1311  00:F32E  85 7C          STA TMP0+1
1312  00:F330  20 F4 FB    KKLLPP1  JSR CRLF
1313  00:F333  20 D6 FB       JSR WROA
1314  00:F336  20 04 FC       JSR SPAC2      ;2 spaces
1315  00:F339  20 09 FC       JSR SPAC       ;and one more
1316  00:F33C  A9 10          LDA #16
1317  00:F33E  85 5F          STA TMPC
1318
1319       00:F340       M0      EQU *
1320  00:F340  20 9B F3       JSR BYTE
1321  00:F343  90 07          BCC M001       ;branch if we read back what we wrote
1322  00:F345  08             PHP            ;save EQU flag status
1323                         ;this is used because there are some periphals
1324                         ;that we CANT read back, so we want to continue
1325  00:F346  A9 3F          LDA #'?'       ;indicate an error
1326  00:F348  20 11 FC       JSR OUTCH      ;by replacing the space with a '?'
1327  00:F34B  28             PLP            ;replace EQU flag from BYTE sub.
1328  00:F34C  F0 E2       M001    BEQ KKLLPP1    ;branch if we have done 16
1329  00:F34E  B0 F0          BCS M0         ;we had a write error - no space needed
1330  00:F350  20 09 FC       JSR SPAC       ;output a space if we are on same line and no error
1331  00:F353  80 EB          BRA M0
1332  00:F355
1333
1334
1335       00:F355       DSPLYINC EQU *         ;INC DISPLAY MEM
1336  00:F355  20 3B F8       JSR INCTMP
1337  00:F358  80 0A          BRA DSPLYOLD
1338
1339
1340       00:F35A       DSPLYDEC EQU *         ;DEC DISPLAY MEM
1341  00:F35A  A5 7B          LDA TMP0
1342  00:F35C  08             PHP
1343  00:F35D  C6 7B          DEC TMP0
1344  00:F35F  28             PLP
1345  00:F360  D0 02          BNE DSPLYOLD   ;REDISPLAY MEM
1346  00:F362  C6 7C          DEC TMP0+1
1347  00:F364  20 D6 FB    DSPLYOLD JSR WROA
1348  00:F367  20 09 FC       JSR SPAC
1349  00:F36A  B2 7B          LDA (TMP0)
1350  00:F36C  20 EB FB       JSR WROB
1351  00:F36F  20 04 FC       JSR SPAC2
1352  00:F372  B2 7B          LDA (TMP0)
1353  00:F374  29 7F          AND #$7F
1354  00:F376  20 1E FE       JSR IFASC      ;CK IF ASCII DATA
1355  00:F379  90 02          BCC OLD1       ;YES ,ASCII
1356  00:F37B  A9 2E          LDA #'.'       ;NOT ASCII SO '.'
1357  00:F37D  20 11 FC    OLD1    JSR OUTCH      ;OUTPUT ASCII
1358  00:F380  4C 18 F2       JMP START
1359
```

```
1360
1361        00:F383             DCMP    EQU *
```

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Commands, Display Registers'

```
1362  00:F383  38                   SEC          ;TMP2-TMP0 DBL SUBTRACT
1363  00:F384  A5 7E                LDA TMP2
1364  00:F386  E5 7B                SBC TMP0
1365  00:F388  85 79                STA DIFF
1366  00:F38A  A5 7F                LDA TMP2+1
1367  00:F38C  E5 7C                SBC TMP0+1
1368  00:F38E  A8                   TAY
1369  00:F38F  05 79                ORA DIFF      ;OR LO FOR EQU TEST
1370  00:F391  60                   RTS
1371
1372
1373         00:F392            PUTP    EQU *
1374  00:F392  A5 7B                LDA TMP0      ;MOVE TMP0 TO PCH,PCL
1375  00:F394  85 58                STA PCL
1376  00:F396  A5 7C                LDA TMP0+1
1377  00:F398  85 59                STA PCH
1378  00:F39A  60                   RTS
1379
1380
1381
1382                  ;* Routine:  BYTE
1383                  ;*  READ AND STORE BYTE.
1384                  ;*  NO STORE IF SPACE OR TMPC=0.
1385                  ;* Reg Used: ACC,Y,X
1386                  ;* Var Used: TMPC,TMP0
1387                  ;* Routines Called: RDOB,DADD,INCTMP
1388                  ;* Returned Reg: NONE
1389                  ;*
1390
1391         00:F39B            BYTE    EQU *
1392  00:F39B  20 FD FC             JSR RDOB      ;CHAR IN A, CY=0 IF
1393  00:F39E  90 10                BCC BY3       ;SPACE,CR, OR COMMA
1394  00:F3A0  92 7B                STA (TMP0)    ;STORE BYTE
1395  00:F3A2  D2 7B                CMP (TMP0)    ;TEST FOR VALID WRITE
1396  00:F3A4  F0 07                BEQ BY2
1397  00:F3A6  20 3B F8             JSR INCTMP    ;increment the address
1398  00:F3A9  C6 5F                DEC TMPC
1399  00:F3AB  38                   SEC
1400  00:F3AC  60                   RTS
1401
1402
1403  00:F3AD  20 2D F8        BY2     JSR DADD      ;INCR CKSUM
1404  00:F3B0  20 3B F8        BY3     JSR INCTMP    ;GO INCR TMP0 ADR
1405  00:F3B3  C6 5F                DEC TMPC
1406  00:F3B5  18                   CLC
1407  00:F3B6  60                   RTS
1408
1409
1410         00:F3B7            SETR    EQU *
1411  00:F3B7  A9 5A                LDA #<FLGS    ;SET TO ACCESS REGS
1412  00:F3B9  85 7B                STA TMP0
1413  00:F3BB  64 7C                STZ TMP0+1    ;WE KNOW ITS IN PAGE ZERO
1414  00:F3BD  A9 05                LDA #5
1415  00:F3BF  60                   RTS
1416                        .STTL 'MON3.ASM - Commands, Alter registers/memory'
```

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Commands, Alter registers/memory'

```
1418
1419
1420                    ;* Routine:  ALTER
1421                    ;*   ALTER LAST DISPLAYED ITEM (ADDR IN TMP0)
1422                    ;*   TO END ENTERING DATA PRESS CR
1423                    ;* Reg Used: ACC,Y,X
1424                    ;* Var Used: TMPC,TMP0
1425                    ;* Routines Called: RDOB,DADD,INCTMP
1426                    ;* Returned Reg: NONE
1427                    ;*
1428
1429         00:F3C0              ALTER    EQU *
1430  00:F3C0  20 F3 F2                JSR REGTTL     ;DISPLAY REGISTER HEADER
1431  00:F3C3  20 E0 FC                JSR RDOA
1432  00:F3C6  B0 03                   BCS A2        ;GOT SPACE, CR OR COMMA SO SKIP MODIFYING PC
1433  00:F3C8  20 92 F3                JSR PUTP      ;ALTER PC
1434         00:F3CB              A2       EQU *
1435                    ;       CMP #CR       ;CHK IF CR IF SO END
1436                    ;       BEQ A6
1437  00:F3CB  20 B7 F3                JSR SETR      ;POINT TO ALTER REGS &
1438  00:F3CE  85 5F                   STA TMPC      ;SAVE BYTE COUNT
1439         00:F3D0              A5       EQU *
1440  00:F3D0  20 09 FC                JSR SPAC      ;PRESERVES Y
1441  00:F3D3  20 9B F3                JSR BYTE
1442  00:F3D6  D0 F8                   BNE A5        ;NOT A CR YET, SO DO NEXT REGISTER
1443  00:F3D8  4C 18 F2         A6       JMP START
1444
1445
1446                         .STTL 'MON3.ASM - Commands, JMP ---GO TO address'
1447                         .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Commands, JMP ---GO TO address'

```
1448
1449                    ;* Routine:  GO
1450                    ;*
1451                    ;* Reg Used: ACC,Y,X
1452                    ;* Var Used: TMP0,SININDX,SINEND,SOUTINDX,SOUTEND
1453                    ;* Routines Called: GOADDR,FLUSH_SERIAL_BUFF,SPAC,PUTP,RDOB
1454                    ;* Returned Reg: NONE
1455                    ;*
1456
1457         00:F3DB              GO      EQU *
1458  00:F3DB  20 05 F4                  JSR GOADDR    ;GET ADDRESS
1459                                    ;if we got a BAD ADDR OR CR,so use OLD address
1460  00:F3DE  20 B4 FD       GO1     JSR FLUSH_SERIAL_BUFF  ;FLUSH SERIAL QUEUE
1461  00:F3E1  A6 5E                  LDX TMPSP
1462  00:F3E3  9A                     TXS          ;ORIG OR NEW SP VALUE TO SP
1463  00:F3E4  A5 59          GO2     LDA PCH
1464  00:F3E6  48                     PHA          ;SAVE ON STACK
1465  00:F3E7  A5 58                  LDA PCL
1466  00:F3E9  48                     PHA
1467  00:F3EA  A5 5A                  LDA FLGS
1468  00:F3EC  48                     PHA
1469  00:F3ED  A5 5B                  LDA ACC
1470  00:F3EF  A6 5C                  LDX XR
1471  00:F3F1  A4 5D                  LDY YREG
1472  00:F3F3  40                     RTI
1473
1474
1475                    ;* Routine:  GOJSR
1476                    ;*
1477                    ;* Reg Used: ACC,Y,X
1478                    ;* Var Used: TMP0,SININDX,SINEND,SOUTINDX,SOUTEND
1479                    ;* Routines Called: GOADDR,FLUSH_SERIAL_BUFF,SPAC,PUTP,RDOB
1480                    ;* Returned Reg: NONE
1481                    ;*
1482
1483         00:F3F4              GOJSR    EQU *        ;GO TO A JSR
1484  00:F3F4  20 05 F4                  JSR GOADDR    ;GET ADDRESS
1485                                    ;if we got a BAD ADDR OR CR,so use OLD address
1486  00:F3F7  20 B4 FD                  JSR FLUSH_SERIAL_BUFF ;FLUSH SERIAL QUEUE
1487  00:F3FA  A6 5E                  LDX TMPSP
1488  00:F3FC  9A                     TXS          ;ORIG OR NEW SP VALUE TO SP
1489  00:F3FD  A9 F2                  LDA #>START-1  ;PLACE RETURN ADDRESS ON STACK
1490  00:F3FF  48                     PHA          ;IE START OF MONITOR
1491  00:F400  A9 17                  LDA #<START-1
1492  00:F402  48                     PHA
1493  00:F403  80 DF                  BRA GO2
1494
1495         00:F405              GOADDR   EQU *        ;GET GO/JMP/JSR ADDRESS
1496  00:F405  20 09 FC                  JSR SPAC
1497  00:F408  20 E0 FC                  JSR RDOA      ;GET ADDRESS
1498  00:F40B  B0 03                  BCS GOAD      ;USE OLD ADDR, because probably a CR
1499                                   ;( cy is SET on CR, SPACE, OR COMMA)
1500  00:F40D  20 92 F3                  JSR PUTP      ;SET UP NEW ADDR
1501  00:F410  60              GOAD    RTS
1502
```

```
1503
1504                    .STTL 'MON3.ASM - Commands, Display Memory'
```

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Commands, Display Memory'

1505 .PAGE

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Commands, Display Memory'

```
1506
1507                    ;HEX MEMORY DUMP ROUTINE
1508                    ;* Routine:  WM
1509                    ;*
1510                    ;* Reg Used: ACC,Y,X
1511                    ;* Var Used: TMPC,TMP0,TMP2,TMP4,TMP6,DIFF,
1512                    ;* Routines Called: RD_SAEA,PRTSTR,WROB,SPAC,
1513                    ;*               SPAC2,BY3,IFASC,OUTCH,DCMP
1514                    ;* Returned Reg: NONE
1515                    ;*
1516
1517  00:F411  20 01 F8         WM      JSR RD_SAEA
1518  00:F414  20 1A F4               JSR WRROUT
1519  00:F417  4C 18 F2               JMP START
1520
1521  00:F41A  A5 7B            WRROUT   LDA TMP0       ;This subroutine displays memory from
1522  00:F41C  48                     PHA             ;TMP0 to TMP2
1523  00:F41D  A4 7C                  LDY TMP0+1
1524  00:F41F  5A                     PHY
1525  00:F420  48                     PHA
1526  00:F421  A9 F4                  LDA #>DUMPSTR
1527  00:F423  A2 B3                  LDX #<DUMPSTR
1528  00:F425  A0 08                  LDY #DUMPEND-DUMPSTR
1529  00:F427  20 06 F5               JSR PRTSTR
1530  00:F42A  A0 10                  LDY #16
1531  00:F42C  68                     PLA
1532  00:F42D  29 0F                  AND #LOWNIB
1533  00:F42F  48               WRX1    PHA
1534  00:F430  20 EB FB               JSR WROB
1535  00:F433  20 09 FC               JSR SPAC
1536  00:F436  68                     PLA
1537  00:F437  1A                     INC A
1538  00:F438  88                     DEY
1539  00:F439  D0 F4                  BNE WRX1
1540  00:F43B  68                     PLA
1541  00:F43C  85 7C                  STA TMP0+1
1542  00:F43E  68                     PLA
1543  00:F43F  85 7B                  STA TMP0
1544
1545  00:F441  A6 78            WM0     LDX WRAP
1546  00:F443  D0 6D                  BNE WMX
1547  00:F445  20 F4 FB               JSR CRLF      ;NEW LINE
1548  00:F448  B0 1D                  BCS MD1
1549  00:F44A  20 D6 FB               JSR WROA      ;PRINT ADDR
1550  00:F44D  B0 18                  BCS MD1
1551  00:F44F  A2 10                  LDX #16       ;BYTES PER LINE
1552  00:F451  86 5F                  STX TMPC
1553  00:F453  20 83 F3               JSR DCMP      ;EA-SA  DIFF IN DIFF & Y
1554
1555  00:F456  98                     TYA           ;Y REG = MSD
1556  00:F457  D0 0A                  BNE WM1
1557  00:F459  A5 79                  LDA DIFF      ;LSB of difference
1558  00:F45B  C9 0F                  CMP #15       ;full line of display
1559  00:F45D  B0 04                  BCS WM1
1560  00:F45F  85 5F                  STA TMPC      ;SHORT LINE
```

```
1561  00:F461  E6 5F                 INC TMPC
1562  00:F463  20 04 FC       WM1     JSR SPAC2
```

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Commands, Display Memory'

```
1563  00:F466  18                   CLC
1564  00:F467  A5 7B          MD1    LDA TMP0      ;SAVE POINTERS TO
1565  00:F469  85 83                STA TMP6      ;DO ASCII
1566  00:F46B  A5 7C                LDA TMP0+1
1567  00:F46D  85 84                STA TMP6+1
1568  00:F46F  A5 5F                LDA TMPC
1569  00:F471  85 81                STA TMP4
1570
1571  00:F473  20 09 FC       WM2    JSR SPAC
1572  00:F476  B2 7B                LDA (TMP0)    ;DATA
1573  00:F478  20 EB FB             JSR WROB      ;TWO HEX BYTES
1574  00:F47B  B0 35                BCS WMX
1575  00:F47D  20 B0 F3             JSR BY3       ;UP ADDR
1576  00:F480  D0 F1                BNE WM2
1577
1578  00:F482  20 04 FC             JSR SPAC2
1579  00:F485  B0 2B                BCS WMX
1580  00:F487  20 04 FC             JSR SPAC2
1581  00:F48A  B0 26                BCS WMX
1582  00:F48C  A5 83                LDA TMP6      ;RESTORE POINTERS
1583  00:F48E  85 7B                STA TMP0
1584  00:F490  A5 84                LDA TMP6+1
1585  00:F492  85 7C                STA TMP0+1
1586  00:F494  A5 81                LDA TMP4
1587  00:F496  85 5F                STA TMPC
1588  00:F498  B2 7B          LISTS2 LDA (TMP0)    ;DATA
1589  00:F49A  29 7F                AND #$7F      ;MASK
1590  00:F49C  20 1E FE             JSR IFASC     ;CK IF ASCII DATA
1591  00:F49F  90 02                BCC LISTS3    ;YES ,ASCII
1592  00:F4A1  A9 2E                LDA #'.'      ;NOT ASCII SO '.'
1593  00:F4A3  20 11 FC       LISTS3 JSR OUTCH     ;OUTPUT ASCII
1594  00:F4A6  B0 0A                BCS WMX       ;GOT A CNTRL C
1595  00:F4A8  20 B0 F3             JSR BY3       ;UP ADDR
1596  00:F4AB  D0 EB                BNE LISTS2
1597  00:F4AD  20 83 F3             JSR DCMP
1598  00:F4B0  B0 8F                BCS WM0       ;AGAIN
1599  00:F4B2  60             WMX    RTS
1600
1601
1602       00:F4B3          DUMPSTR  EQU *
1603  00:F4B3  0D 41 44 44 52                .BYTE $0D,'ADDR   '
    00F4B8  20 20 20
1604       00:F4BB          DUMPEND  EQU *
1605
1606                      .STTL 'MON3.ASM - Commands, Fill memory'
1607                      .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Commands, Fill memory'

```
1608
1609                    ;* Routine:  FILL
1610                    ;*
1611                    ;* Reg Used: ACC,Y,X
1612                    ;* Var Used: TMPC,TMP0,TMP6,DIFF
1613                    ;* Routines Called: RD_SAEA,SPAC,RDOB,BY3,DCMP
1614                    ;* Returned Reg: NONE
1615                    ;*
1616
1617 00:F4BB 20 01 F8       FILL    JSR RD_SAEA   ;READ SA & EA
1618 00:F4BE B0 29                  BCS FILLSX
1619 00:F4C0 20 09 FC               JSR SPAC
1620 00:F4C3 B0 24                  BCS FILLSX
1621 00:F4C5 20 FD FC               JSR RDOB      ;READ FILL CHAR
1622 00:F4C8 90 1F                  BCC FILLSX
1623 00:F4CA 85 83                  STA TMP6
1624 00:F4CC 20 83 F3       FILLS0  JSR DCMP      ;EA - SA
1625 00:F4CF A5 79                  LDA DIFF      ;LSB of difference
1626 00:F4D1 85 5F                  STA TMPC
1627 00:F4D3 E6 5F                  INC TMPC
1628 00:F4D5 A5 83          FILLS1  LDA TMP6
1629 00:F4D7 92 7B                  STA (TMP0)    ;DATA
1630 00:F4D9 D2 7B                  CMP (TMP0)    ;CK IF WRITTEN
1631 00:F4DB D0 0F                  BNE FILLERR
1632 00:F4DD 20 B0 F3               JSR BY3       ;UP ADDR
1633 00:F4E0 D0 F3                  BNE FILLS1
1634 00:F4E2 20 83 F3               JSR DCMP
1635 00:F4E5 F0 02                  BEQ FILLSX
1636 00:F4E7 B0 E3                  BCS FILLS0    ;AGAIN
1637 00:F4E9 4C 18 F2       FILLSX  JMP START
1638
1639 00:F4EC 4C 5B F2       FILLERR JMP ERROPR    ;BAD FILL, MEMORY MISSING
1640
1641                        .STTL 'MON3.ASM - Commands, Help'
1642                        .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Commands, Help'

```
1643
1644
1645          00:F4EF              HELP     EQU *          ;DISPLAY HELP MENU
1646  00:F4EF  A9 F5                     LDA #>HELPMENU
1647  00:F4F1  A2 1C                     LDX #<HELPMENU
1648  00:F4F3  A0 D8                     LDY #HELPEND1-HELPMENU
1649  00:F4F5  20 06 F5                  JSR PRTSTR
1650  00:F4F8  B0 09                     BCS HELPX      ;BAD RETURN FROM SERIAL
1651  00:F4FA  A9 F5                     LDA #>HELPEND1
1652  00:F4FC  A2 F4                     LDX #<HELPEND1
1653  00:F4FE  A0 BC                     LDY #HELPEND-HELPEND1
1654  00:F500  20 06 F5                  JSR PRTSTR
1655  00:F503  4C 18 F2          HELPX    JMP START
1656
1657
1658                       ;* Routine:  PRTSTR
1659                       ;*
1660                       ;* Reg Used: ACC & Y
1661                       ;* Var Used: TMP0,TMP2,TMP6
1662                       ;* Routines Called: OUTCH
1663                       ;* Returned Reg: NONE
1664                       ;*
1665
1666  00:F506  85 7C              PRTSTR   STA TMP0+1     ;MSB OF ADDRESS
1667  00:F508  86 7B                     STX TMP0       ;LSB OF ADDRESS
1668  00:F50A  84 83                     STY TMP6       ;# OF CHAR TO PRINT/SEND OUT TO SERIAL
1669  00:F50C  A0 00                     LDY #0
1670  00:F50E  B1 7B              HELP1    LDA (TMP0),Y
1671  00:F510  20 11 FC                  JSR OUTCH
1672  00:F513  B0 06                     BCS HELP2      ;GOT A CNTRL C
1673  00:F515  C8                        INY
1674  00:F516  C4 83                     CPY TMP6
1675  00:F518  D0 F4                     BNE HELP1
1676  00:F51A  18                        CLC
1677  00:F51B  60                 HELP2    RTS
1678
1679
1680  00:F51C  0D                 HELPMENU .BYTE $0D
1681  00:F51D  44 20 20 20 20           .BYTE 'D     Display memory',$0D
    00F522  20 20 44 69 73
    00F527  70 6C 61 79 20
    00F52C  6D 65 6D 6F 72
    00F531  79 0D
1682  00:F533  53 50 41 43 45           .BYTE 'SPACE  Display current memory address',$0D
    00F538  20 20 44 69 73
    00F53D  70 6C 61 79 20
    00F542  63 75 72 72 65
    00F547  6E 74 20 6D 65
    00F54C  6D 6F 72 79 20
    00F551  61 64 64 72 65
    00F556  73 73 0D
1683  00:F559  3C 2C 3E 20 20                .BYTE '<,>    Decrement, Increment memory address',$0D
    00F55E  20 20 44 65 63
    00F563  72 65 6D 65 6E
    00F568  74 2C 20 49 6E
```

00F56D 63 72 65 6D 65
00F572 6E 74 20 6D 65

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Commands, Help'

00F577 6D 6F 72 79 20
00F57C 61 64 64 72 65
00F581 73 73 0D
1684 00:F584 4D 20 20 20 20 .BYTE 'M Alter memory',$0D
00F589 20 20 41 6C 74
00F58E 65 72 20 6D 65
00F593 6D 6F 72 79 0D
1685 00:F598 2F 20 20 20 20 .BYTE '/ Host memory access',$0D,$0D
00F59D 20 20 48 6F 73
00F5A2 74 20 6D 65 6D
00F5A7 6F 72 79 20 61
00F5AC 63 63 65 73 73
00F5B1 0D 0D
1686
1687 00:F5B3 52 2C 41 20 20 .BYTE 'R,A Display, Alter registers',$0D
00F5B8 20 20 44 69 73
00F5BD 70 6C 61 79 2C
00F5C2 20 41 6C 74 65
00F5C7 72 20 72 65 67
00F5CC 69 73 74 65 72
00F5D1 73 0D
1688 00:F5D3 47 2C 4A 20 20 .BYTE 'G,J JMP, JSR to PC [location]',$0D
00F5D8 20 20 4A 4D 50
00F5DD 2C 20 4A 53 52
00F5E2 20 74 6F 20 50
00F5E7 43 20 5B 6C 6F
00F5EC 63 61 74 69 6F
00F5F1 6E 5D 0D
1689 00:F5F4 46 2C 56 2C 43 HELPEND1 .BYTE 'F,V,C Block Fill, Move, Checksum',$0D
00F5F9 20 20 42 6C 6F
00F5FE 63 6B 20 46 69
00F603 6C 6C 2C 20 4D
00F608 6F 76 65 2C 20
00F60D 43 68 65 63 6B
00F612 73 75 6D 0D
1690 00:F616 53 2C 57 2C 45 .BYTE 'S,W,E S28 Input, Output, Errors',$0D,$0D
00F61B 20 20 53 32 38
00F620 20 49 6E 70 75
00F625 74 2C 20 4F 75
00F62A 74 70 75 74 2C
00F62F 20 45 72 72 6F
00F634 72 73 0D 0D
1691
1692 00:F638 3F 2C 48 20 20 .BYTE '?,H Help',$0D
00F63D 20 20 48 65 6C
00F642 70 0D
1693 00:F644 42 2C 4B 20 20 .BYTE 'B,K BASIC Start, Continue',$0D
00F649 20 20 42 41 53
00F64E 49 43 20 53 74
00F653 61 72 74 2C 20
00F658 43 6F 6E 74 69
00F65D 6E 75 65 0D
1694 00:F661 54 20 20 20 20 .BYTE 'T Display time',$0D
00F666 20 20 44 69 73
00F66B 70 6C 61 79 20

```
00F670  74 69 6D 65 0D
1695  00:F675  58 20 20 20 20          .BYTE 'X      Toggle handshake mode',$0D
```

```
     'W65C134 Internal ROM Monitor ($F000)'
     'MON3.ASM - Commands, Help'

    00F67A  20 20 54 6F 67
    00F67F  67 6C 65 20 68
    00F684  61 6E 64 73 68
    00F689  61 6B 65 20 6D
    00F68E  6F 64 65 0D
1696  00:F692  55 20 20 20 20          .BYTE 'U      User installed commands'
    00F697  20 20 55 73 65
    00F69C  72 20 69 6E 73
    00F6A1  74 61 6C 6C 65
    00F6A6  64 20 63 6F 6D
    00F6AB  6D 61 6E 64 73
1697          00:F6B0                 HELPEND  EQU *
1698
1699
1700                    ;* Routine:  VERSION
1701                    ;*
1702                    ;* Reg Used: Acc,Y,X
1703                    ;* Var Used: NONE
1704                    ;* Routines Called: NONE
1705                    ;* Returned Reg: Acc,X,Y
1706                    ;*
1707
1708          00:F6B0               VERSION  EQU *         ;RETURN VERSION IN A,X,Y
1709  00:F6B0  AD 63 F0                 LDA MONVER#
1710  00:F6B3  AE 64 F0                 LDX MONVER#+1
1711  00:F6B6  AC 65 F0                 LDY MONVER#+2
1712  00:F6B9  60                       RTS
1713
1714                          .STTL 'MON3.ASM - Commands, S28/S19 HEX loader'
1715                          .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Commands, S28/S19 HEX loader'

```
1716
1717
1718                    ;* Routine:  MS28IN
1719                    ;*
1720                    ;* Reg Used: ACC,Y,X
1721                    ;* Var Used: TMP0,TMP2,TMP4,TMP6,TMPC,ERRORS,SFLAG,
1722                    ;* Routines Called: GETCH,DADD,RDOB,OUTCH,PERR1,CRLF,SPAC,
1723                    ;*                 WROA,SPAC2,BYTE,BY2
1724                    ;* Returned Reg: NONE
1725                    ;*
1726
1727          00:F6BA             MS28IN   EQU *
1728  00:F6BA  20 71 FC                 JSR GETCH
1729  00:F6BD  C9 53                    CMP #'S'      ;FIND NEXT RCD MARK (S)
1730  00:F6BF  D0 F9                    BNE MS28IN
1731
1732
1733          00:F6C1             LSS      EQU *        ;LOAD SINGLE S28 RECORD
1734                                  ;CHECKSUM USED, BUT
1735                                  ;REQUIRED SO THAT IT
1736                                  ;IS COMPATABLE
1737  00:F6C1  E6 85                    INC ERRORS
1738  00:F6C3  78                       SEI
1739  00:F6C4  A5 72                    LDA SFLAG
1740  00:F6C6  29 20                    AND #ECHOFF
1741  00:F6C8  85 84                    STA TMP6+1    ;SAVE CURRENT STATE OF ECHO OFF
1742  00:F6CA  A9 20                    LDA #ECHOFF   ;SET ECHO OFF
1743  00:F6CC  04 72                    TSB SFLAG
1744
1745  00:F6CE  58                       CLI
1746  00:F6CF  20 71 FC                 JSR GETCH     ;GET S RECORD TYPE
1747  00:F6D2  B0 64                    BCS LSS0      ;CHK FOR CNTRL 'C'
1748  00:F6D4  48                       PHA           ;SAVE S RECORD TYPE
1749  00:F6D5  64 81                    STZ TMP4
1750  00:F6D7  64 82                    STZ TMP4+1    ;CLR CKSUM REG
1751  00:F6D9  20 FD FC                 JSR RDOB      ;GET BYTE COUNT
1752  00:F6DC  85 5F                    STA TMPC      ;SAVE BYTE COUNT
1753  00:F6DE  20 2D F8                 JSR DADD
1754  00:F6E1  C6 5F                    DEC TMPC      ;GET S28 ADDR
1755  00:F6E3  C6 5F                    DEC TMPC
1756  00:F6E5  C6 5F                    DEC TMPC
1757  00:F6E7  68                       PLA           ;GET RECORD TYPE
1758  00:F6E8  C9 31                    CMP #'1'
1759  00:F6EA  F0 14                    BEQ SHORTADDR  ;SHORT RECORD
1760  00:F6EC  C9 39                    CMP #'9'
1761  00:F6EE  F0 10                    BEQ SHORTADDR
1762  00:F6F0  C9 32                    CMP #'2'
1763  00:F6F2  F0 04                    BEQ S28LA1
1764  00:F6F4  C9 38                    CMP #'8'
1765  00:F6F6  D0 40                    BNE LSS0
1766  00:F6F8  C6 5F             S28LA1   DEC TMPC
1767  00:F6FA  20 FD FC                 JSR RDOB      ;GET BANK ADDR, But ignore it
1768  00:F6FD  20 2D F8                 JSR DADD      ;ADD TO CKSM
1769  00:F700  20 FD FC          SHORTADDR JSR RDOB     ;SA HO TO TMP0+1
1770  00:F703  85 7C                    STA TMP0+1
```

```
1771  00:F705  20 2D F8          JSR DADD     ;ADD TO CKSM
1772  00:F708  20 FD FC          JSR RDOB     ;SA LO TO TMP0
```

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Commands, S28/S19 HEX loader'

```
1773  00:F70B  85 7B          STA TMP0
1774  00:F70D  20 2D F8       JSR DADD      ;ADD TO CHKSM
1775
1776  00:F710  A5 5F          LDA TMPC      ;CK IF # OF BYTES ZERO
1777  00:F712  F0 0E          BEQ S28G2
1778  00:F714  20 9B F3   S28GD1  JSR BYTE      ;BYTE SUB/R DECRS LCNT
1779  00:F717  90 07          BCC S28G3
1780  00:F719  E6 85          INC ERRORS    ;DEC COUNTER & INC ADDR
1781  00:F71B  E6 81          INC TMP4      ;MESS UP CKSUM SO WILL PRINT ERR
1782  00:F71D  20 AD F3       JSR BY2       ;SO CNT WILL BE RIGHT FROM
1783                          ;BYTE ENDING TOO SOON
1784  00:F720  D0 F2      S28G3   BNE S28GD1    ;ON EXIT
1785  00:F722  20 FD FC   S28G2   JSR RDOB      ;CKSUM FROM HEX RCD>TMP0
1786  00:F725  20 2D F8       JSR DADD
1787  00:F728  A5 81          LDA TMP4      ;GET CHKSUM
1788
1789  00:F72A  C9 FF          CMP #$FF
1790  00:F72C  F0 0A          BEQ LSS0
1791  00:F72E  A9 07          LDA #$07      ;BEEP
1792  00:F730  20 11 FC       JSR OUTCH
1793  00:F733  20 56 F7       JSR PERR1
1794  00:F736  80 02          BRA LSFIN
1795
1796  00:F738  C6 85      LSS0    DEC ERRORS    ;A GOOD LOAD
1797  00:F73A  20 71 FC   LSFIN   JSR GETCH     ;GET CR OR LF
1798  00:F73D  78             SEI
1799  00:F73E  A5 72          LDA SFLAG
1800  00:F740  29 DF          AND #$FF-ECHOFF
1801  00:F742  05 84          ORA TMP6+1    ;RESTORE STATE OF ECHO OFF
1802  00:F744  85 72          STA SFLAG
1803  00:F746  58             CLI
1804  00:F747  4C 18 F2   LHDONE  JMP START     ;GOTO START
1805
1806
1807  00:F74A  20 56 F7   PERR    JSR PERR1
1808  00:F74D  64 85          STZ ERRORS    ;RESET ERROR COUNT
1809  00:F74F  64 7B          STZ TMP0      ;RESET ADDR ALSO
1810  00:F751  64 7C          STZ TMP0+1
1811  00:F753  4C 18 F2       JMP START
1812
1813
1814  00:F756  20 F4 FB   PERR1   JSR CRLF      ;PRINT # OF ERRORS
1815  00:F759  A9 45          LDA #'E'
1816  00:F75B  20 11 FC       JSR OUTCH
1817  00:F75E  20 09 FC       JSR SPAC
1818  00:F761  20 D6 FB       JSR WROA      ;OUTPUT ADDR CLOSE TO
1819  00:F764  20 04 FC       JSR SPAC2     ;ERROR,MAYBE OFF BY 16
1820  00:F767  A5 85          LDA ERRORS
1821  00:F769  4C 87 FA       JMP SNDT1     ;WRITE # OF ERRORS
1822
1823                  .STTL 'MON3.ASM - Commands, S19 HEX output'
1824                  .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Commands, S19 HEX output'

```
1825
1826
1827                    ;* Routine:  MS19OUT
1828                    ;*
1829                    ;* Reg Used: ACC,Y,X
1830                    ;* Var Used: TMP0,TMP2,TMP4,TMP6,SFLAG,TMPC,DIFF
1831                    ;* Routines Called: CRLF,GETCH,RD_SAEA,WRTWO,DCMP,DADD
1832                    ;*                  CKNOUT,WROB
1833                    ;* Returned Reg: NONE
1834                    ;*
1835
1836 00:F76C  53 39 30 33 30 SLASTLINE .BYTE  'S9030000FC'
   00F771  30 30 30 46 43
1837          00:F776               SLASTEND EQU *
1838
1839          00:F776               WO      EQU *
1840 00:F776  64 83                         STZ TMP6      ;TMP6 is the Offset added as the address goes out
1841 00:F778  64 84                         STZ TMP6+1    ;We use TMP6 to 'relocate' as we write the output
1842
1843 00:F77A  20 01 F8                      JSR RD_SAEA   ;RD START ADDR & END ADDR
1844
1845 00:F77D  20 71 FC               WO1    JSR GETCH     ;DELAY FOR FINAL CR
1846 00:F780  20 F4 FB                      JSR CRLF
1847 00:F783  A9 FF                         LDA #$FF      ;indicates need to send end record
1848 00:F785  20 8B F7                      JSR MS19OUT
1849 00:F788  4C 18 F2                      JMP START
1850
1851          00:F78B               MS19OUT  EQU *
1852 00:F78B  64 78                         STZ WRAP
1853 00:F78D  48                            PHA
1854 00:F78E  20 A1 F7                      JSR MS19OUTA
1855 00:F791  68                            PLA
1856 00:F792  F0 0C                         BEQ NOFINAL
1857
1858 00:F794  20 F4 FB                      JSR CRLF      ;WRITE LAST LINE
1859 00:F797  A2 6C                         LDX #<SLASTLINE
1860 00:F799  A9 F7                         LDA #>SLASTLINE
1861 00:F79B  A0 0A                         LDY #SLASTEND-SLASTLINE
1862 00:F79D  20 06 F5                      JSR PRTSTR
1863 00:F7A0  60                     NOFINAL  RTS
1864
1865
1866 00:F7A1  A6 78                  MS19OUTA LDX WRAP
1867 00:F7A3  D0 55                         BNE BCCST2
1868
1869 00:F7A5  20 F4 FB                      JSR CRLF
1870 00:F7A8  64 81                         STZ TMP4
1871 00:F7AA  64 82                         STZ TMP4+1    ;CLEAR CKSUM
1872 00:F7AC  A9 13                         LDA #S28BN
1873 00:F7AE  85 5F                         STA TMPC      ;TMPC = 16+3 FOR SHORT
1874
1875 00:F7B0  A9 31                         LDA #'1'      ;OUTPUT S1
1876 00:F7B2  A2 53                         LDX #'S'      ;
1877 00:F7B4  20 F8 FB                      JSR WRTWO
1878 00:F7B7  20 83 F3                      JSR DCMP      ;EA-SA (TMP0+2-TMP0) DIFFERENCE
```

```
1879  00:F7BA  98             TYA         ;IN LOC DIFF,Y REG HAS MSD
1880  00:F7BB  D0 0B          BNE WH10    ;OF DIFFERENCE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Commands, S19 HEX output'

```
1881  00:F7BD  A5 79          LDA DIFF
1882  00:F7BF  C9 0F          CMP #15
1883  00:F7C1  B0 05          BCS WH10      ;DIFF GT 15
1884  00:F7C3  18             CLC           ;ADD 2 FOR ADDR
1885  00:F7C4  69 04          ADC #$04      ;ADD 1 FOR CKSUM
1886  00:F7C6  85 5F          STA TMPC      ;ADD 1 FOR BYTE CNT
1887  00:F7C8  A5 5F    WH10  LDA TMPC      ;OUTPUT BYTE COUNT
1888  00:F7CA  20 FB F7       JSR CKNOUT    ;RCC CNT IN A
1889
1890  00:F7CD  C6 5F          DEC TMPC
1891  00:F7CF  C6 5F          DEC TMPC
1892  00:F7D1  C6 5F          DEC TMPC
1893
1894  00:F7D3  18       WH1A  CLC           ;output the address plus the offset
1895  00:F7D4  A5 7B          LDA TMP0
1896  00:F7D6  65 83          ADC TMP6
1897  00:F7D8  48             PHA
1898  00:F7D9  A5 7C          LDA TMP0+1
1899  00:F7DB  65 84          ADC TMP6+1
1900  00:F7DD  20 FB F7       JSR CKNOUT    ;ADD TO CKSM
1901  00:F7E0  68             PLA
1902  00:F7E1  20 FB F7       JSR CKNOUT    ;ADD TO CKSM
1903
1904  00:F7E4  B2 7B    WH2   LDA (TMP0)    ;WRITE OUT DATA BYTES
1905  00:F7E6  20 FB F7       JSR CKNOUT    ;INC CKSUM
1906  00:F7E9  20 B0 F3       JSR BY3       ;INC SA
1907  00:F7EC  D0 F6          BNE WH2       ;LOOP FOR 16 BYTES
1908  00:F7EE  A5 81          LDA TMP4
1909  00:F7F0  49 FF          EOR #$FF      ;we want 1's complement
1910  00:F7F2  20 EB FB       JSR WROB      ;WRITE CKSUM
1911  00:F7F5  20 83 F3       JSR DCMP
1912  00:F7F8  B0 A7          BCS MS19OUTA  ;LOOP WHILE EA GT OR = SA
1913
1914  00:F7FA  60       BCCST2  RTS
1915
1916  00:F7FB  20 2D F8 CKNOUT  JSR DADD
1917  00:F7FE  4C EB FB       JMP WROB
1918
1919
1920                    .STTL 'MON3.ASM - Commands, General Purpose Routines'
1921                    .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Commands, General Purpose Routines'

```
1922
1923                    ;* Routine:  RD_SAEA
1924                    ;*
1925                    ;* Reg Used: ACC,Y,X
1926                    ;* Var Used: TMP0,TMP2
1927                    ;* Routines Called: RDOA,SPAC,T2T2,T3T3
1928                    ;* Returned Reg: NONE
1929                    ;*
1930
1931  00:F801  20 09 FC         RD_SAEA  JSR SPAC      ;READ 16 BIT ADDR FORM
1932  00:F804  20 E0 FC                  JSR RDOA
1933
1934  00:F807  A5 7B                     LDA TMP0      ;save SA
1935  00:F809  48                        PHA
1936  00:F80A  18                        CLC
1937  00:F80B  69 0F                     ADC #15
1938  00:F80D  85 7E                     STA TMP2
1939  00:F80F  A5 7C                     LDA TMP0+1
1940  00:F811  48                        PHA
1941  00:F812  69 00                     ADC #0
1942  00:F814  85 7F                     STA TMP2+1    ;EA will be SA+15 if left off
1943
1944  00:F816  20 09 FC                  JSR SPAC      ;put a space between the addresses
1945  00:F819  20 E0 FC                  JSR RDOA      ;get ending address
1946  00:F81C  B0 08                     BCS ENDOK
1947
1948  00:F81E  A5 7B                     LDA TMP0      ;put EA in TMP2
1949  00:F820  85 7E                     STA TMP2
1950  00:F822  A5 7C                     LDA TMP0+1
1951  00:F824  85 7F                     STA TMP2+1
1952
1953  00:F826  68                ENDOK   PLA           ;put SA back in TMP0
1954  00:F827  85 7C                     STA TMP0+1
1955  00:F829  68                        PLA
1956  00:F82A  85 7B                     STA TMP0
1957  00:F82C  60                        RTS
1958
1959
1960
1961
1962                    ;* Routine:  DADD
1963                    ;*
1964                    ;* Reg Used: NONE
1965                    ;* Var Used: TMP4
1966                    ;* Routines Called: NONE
1967                    ;* Returned Reg: NONE
1968                    ;*
1969                              ;CALCULATE CHECKSUM
1970  00:F82D  48                DADD    PHA           ;SAVE A
1971  00:F82E  18                        CLC
1972  00:F82F  65 81                     ADC TMP4
1973  00:F831  85 81                     STA TMP4
1974  00:F833  A5 82                     LDA TMP4+1
1975  00:F835  69 00                     ADC #0
1976  00:F837  85 82                     STA TMP4+1
```

```
1977  00:F839  68                PLA       ;RESTORE A
1978  00:F83A  60                RTS
```

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Commands, General Purpose Routines'

```
1979
1980
1981
1982
1983                    ;* Routine:  INCTMP
1984                    ;*
1985                    ;* Reg Used: NONE
1986                    ;* Var Used: TMP0
1987                    ;* Routines Called: NONE
1988                    ;* Returned Reg: NONE
1989                    ;*
1990
1991                                    ;INC STRING POINTER
1992  00:F83B  E6 7B            INCTMP   INC TMP0      ;LO BYTE
1993  00:F83D  D0 06                     BNE INCT2
1994  00:F83F  E6 7C                     INC TMP0+1
1995  00:F841  D0 02                     BNE INCT2
1996  00:F843  E6 78                     INC WRAP
1997  00:F845  60               INCT2    RTS
1998
1999                          .STTL 'MON3.ASM - Commands, calculate checksum'
2000                          .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Commands, calculate checksum'

```
2001
2002
2003                    ;* Routine:  CHKSUM
2004                    ;*          TMP4 & TMP4+1 contains the sum of the digits to
2005                    ;*          facilitate either TWO's compliment or ONE's compliment checksum
2006                    ;*          Difference between TMP0 & TMP2 should not be GT 255
2007                    ;*
2008                    ;* Reg Used: ACC,Y,X
2009                    ;* Var Used: TMP0,TMP2,TMP4
2010                    ;* Routines Called: SPAC2,WROA4,RD_SAEA,CHK_SUM
2011                    ;* Returned Reg: NONE
2012                    ;*
2013
2014          00:F846               CHKSUM   EQU *         ;CALCULATE CHECK SUM
2015  00:F846  20 01 F8                   JSR RD_SAEA
2016                    ;         BCS CKS_RSTART ;ERROR IN GETTING SA & EA
2017  00:F849  20 55 F8                   JSR CHK_SUM
2018  00:F84C  20 04 FC                   JSR SPAC2
2019  00:F84F  20 DA FB                   JSR WROA4      ;PRINT CHECK SUM
2020          00:F852               CKS_RSTART EQU *
2021  00:F852  4C 18 F2                   JMP START
2022
2023
2024          00:F855               CHK_SUM  EQU *         ;CALCULATE CHECK SUM
2025  00:F855  64 81                      STZ TMP4
2026  00:F857  64 82                      STZ TMP4+1     ;clear checksum register
2027  00:F859  20 83 F3                   JSR DCMP       ;EA-SA in A & Y
2028  00:F85C  F0 0D                      BEQ CKSX       ;A & Y or'd on return
2029  00:F85E  B1 7B              CKS1     LDA (TMP0),Y
2030  00:F860  20 2D F8                   JSR DADD       ;add to TMP4
2031  00:F863  20 3B F8                   JSR INCTMP     ;TMP0+1 to TMP0
2032  00:F866  20 83 F3                   JSR DCMP       ;EA-SA  returns 0 if =
2033  00:F869  D0 F3                      BNE CKS1
2034  00:F86B  60                 CKSX     RTS
2035
2036
2037                          .STTL 'MON3.ASM - Commands, block move'
2038                          .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Commands, block move'

```
2039
2040                    ;* Routine:  MOVE
2041                    ;*
2042                    ;* Reg Used: ACC,Y,X
2043                    ;* Var Used: TMP0,TMP2
2044                    ;* Routines Called: SPAC,RD_SAEA,RDOB,MVDATA
2045                    ;* Returned Reg: NONE
2046                    ;*
2047
2048
2049  00:F86C  20 01 F8         MOVE    JSR RD_SAEA    ;MOVE A BLOCK UPTO 256 BYTES
2050                    ;       BCS CKS_RSTART ;ERROR IN GETTING SA & EA
2051  00:F86F  20 09 FC               JSR SPAC
2052  00:F872  20 FD FC               JSR RDOB      ;GET # OF BYTES
2053  00:F875  A8                     TAY
2054  00:F876  20 EA FD               JSR MVDATA
2055  00:F879  B0 02                  BCS MOVEBAD
2056  00:F87B  80 D5                  BRA CKS_RSTART
2057
2058  00:F87D  4C 5B F2         MOVEBAD  JMP ERROPR    ;DID NOT MOVE MEMORY
2059                                  ;NOT THERE
2060                          .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Commands, block move'

```
2061
2062
2063          00:F880              TGLXONXOFF EQU *       ;TOGGLE SERIAL XON/XOFF MODE
2064  00:F880  20 09 FC                JSR SPAC
2065  00:F883  78                       SEI
2066  00:F884  A5 72                    LDA SFLAG
2067  00:F886  49 04                    EOR #XONOFLG
2068  00:F888  85 72                    STA SFLAG
2069  00:F88A  58                       CLI
2070  00:F88B  29 04                    AND #XONOFLG
2071  00:F88D  F0 02                    BEQ TGLX1
2072  00:F88F  A9 01                    LDA #1
2073  00:F891  20 EB FB          TGLX1    JSR WROB      ;OUTPUT STATUS OF BIT
2074  00:F894  4C 18 F2                 JMP START
2075
2076
2077                          .STTL 'MON3.ASM - Commands, display ToD clock'
2078                          .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON3.ASM - Commands, display ToD clock'

```
2079  00:F897  20 F4 FB       DTIME   JSR CRLF
2080  00:F89A  20 3E FA               JSR SNDTOD
2081  00:F89D  80 B3                  BRA CKS_RSTART
2082                   ;
2083                   ;       The Slash (/) command is to allow host computers quick access
2084                   ;       to memory locations.  It has many forms:
2085                   ;
2086                   ;       /<SPACE> returns curr. mem location & increments pointer
2087                   ;       /XX<SPACE> writes curr. to mem pointer, re-reads and returns
2088                   ;               the data at the pointer (as a check for writable mem)
2089                   ;               then increments pointer.
2090                   ;       /XXXX<SPACE> changes the pointer, returns data, inc's pointer.
2091                   ;       /XXXXYY<SPACE>  changes pointer, writes data, reads & returns data,
2092                   ;                       and increments the pointer.
2093                   ;
2094  00:F89F  78              SLASH   SEI          ;kill the echo
2095  00:F8A0  A5 72                  LDA SFLAG
2096  00:F8A2  29 20                  AND #ECHOFF
2097  00:F8A4  85 84                  STA TMP6+1    ;SAVE CURRENT STATE OF ECHO OFF
2098  00:F8A6  A9 20                  LDA #ECHOFF   ;SET ECHO OFF
2099  00:F8A8  04 72                  TSB SFLAG
2100
2101  00:F8AA  58                     CLI
2102
2103  00:F8AB  20 FD FC               JSR RDOB      ;get the first byte
2104  00:F8AE  90 10                  BCC RETBYTE   ;branch if it was a SPACE
2105  00:F8B0  48                     PHA           ;save the data
2106  00:F8B1  20 FD FC               JSR RDOB      ;get the next byte
2107  00:F8B4  90 07                  BCC WRTBYTE   ;no next byte - write one byte, read, return
2108  00:F8B6  85 7B                  STA TMP0      ;two bytes in succession - an address
2109  00:F8B8  68                     PLA
2110  00:F8B9  85 7C                  STA TMP0+1
2111  00:F8BB  80 E2                  BRA SLASH     ;we had an address and set it - back to beginning.
2112
2113  00:F8BD  68              WRTBYTE  PLA
2114  00:F8BE  92 7B                  STA (TMP0)
2115
2116  00:F8C0  B2 7B           RETBYTE  LDA (TMP0)
2117  00:F8C2  20 EB FB               JSR WROB      ;write the byte to the serial port
2118  00:F8C5  E6 7B                  INC TMP0      ;increment the address
2119  00:F8C7  D0 02                  BNE OUTSLASH
2120  00:F8C9  E6 7C                  INC TMP0+1
2121
2122  00:F8CB  78              OUTSLASH SEI          ;put the echo back
2123  00:F8CC  A5 72                  LDA SFLAG
2124  00:F8CE  29 DF                  AND #$FF-ECHOFF
2125  00:F8D0  05 84                  ORA TMP6+1    ;RESTORE STATE OF ECHO OFF
2126  00:F8D2  85 72                  STA SFLAG
2127  00:F8D4  58                     CLI
2128
2129  00:F8D5  4C 18 F2               JMP START
2130
2131  00:F8D8                         .END
2132
2133  00:F8D8                         INCLUDE MON4.ASM
```

```
2134                    .STTL 'MON4.ASM - Time of Day Clock Routines'
2135                    .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON4.ASM - Time of Day Clock Routines'

```
2136                        ;06-22-1993
2137
2138
2139                ;TIME OF DAY CLOCK
2140                ;* Routine:  INITCLK
2141                ;*
2142                ;* Reg Used: ACC & X
2143                ;* Var Used: SEC,MIN,HR,DAY,MONTH,YR,ASEC, ....
2144                ;* Routines Called: NONE
2145                ;* Returned Reg: NONE
2146                ;*
2147
2148
2149        00:F8D8             INITCLK  EQU *
2150 00:F8D8  A2 07                 LDX #DFLTSEND-DFLTS-1 ;LOAD TOD DEFAULT
2151 00:F8DA  BD A0 FF      ICLK1    LDA DFLTS-1,X
2152 00:F8DD  95 62                 STA SEC-1,X
2153 00:F8DF  CA                    DEX
2154 00:F8E0  D0 F8                 BNE ICLK1
2155 00:F8E2  A2 07                 LDX #DFLTSEND-DFLTS-1 ;RESET ALARM CLOCK ALSO
2156 00:F8E4  74 6A          ICLK2    STZ ASEC-1,X
2157 00:F8E6  CA                    DEX
2158 00:F8E7  D0 FB                 BNE ICLK2
2159 00:F8E9  A2 07                 LDX #DFLTSEND-DFLTS-1 ;LOAD TOD CHECKSUM
2160 00:F8EB  A9 00                 LDA #00
2161 00:F8ED  18                    CLC
2162 00:F8EE  75 62          ICLK3    ADC SEC-1,X
2163 00:F8F0  CA                    DEX
2164 00:F8F1  D0 FB                 BNE ICLK3
2165 00:F8F3  49 FF                 EOR #$FF
2166 00:F8F5  85 60                 STA TODCKS
2167 00:F8F7  60                    RTS
2168
2169                        STTL 'MON4.ASM - Time of day clock IRQ routine'
2170                        PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON4.ASM - Time of day clock IRQ routine'

```
2171
2172
2173                    ;* Routine:  TODIRQ
2174                    ;*
2175                    ;* Reg Used: ACC,Y,X
2176                    ;* Var Used: SEC,MIN,HR,DAY,MONTH,YR,DAYWK,DAYLIT
2177                    ;*          ASEC,AMIN,AHR,ADAY,AMONTH,AYR,ADAYWK
2178                    ;* Routines Called: NONE
2179                    ;* Returned Reg: NONE
2180                    ;*
2181
2182         00:F8F8              TODIRQ   EQU *         ;MONITOR TIME OF DAY IRQ
2183  00:F8F8  48                        PHA
2184  00:F8F9  A9 20                     LDA #T2FLG     ;RESET TIMER 2 IRQ
2185  00:F8FB  04 08                     TSB IFR2
2186  00:F8FD  5A                        PHY
2187  00:F8FE  DA                        PHX
2188  00:F8FF  A9 FF                     LDA #$FF       ;RESET WATCHDOG TIMER 'M'
2189  00:F901  85 28                     STA TMLL
2190  00:F903  85 2A                     STA TMCL
2191  00:F905  85 29                     STA TMLH
2192  00:F907  85 2B                     STA TMCH
2193  00:F909  E6 63                     INC SEC        ;INCREMENT SECONDS
2194  00:F90B  A9 3B                     LDA #59
2195  00:F90D  C5 63                     CMP SEC
2196  00:F90F  B0 34                     BCS EXITOCT
2197  00:F911  64 63                     STZ SEC        ;ROLLED OVER
2198
2199  00:F913  E6 64                     INC MIN        ;INCREMENT MINUTES
2200  00:F915  C5 64                     CMP MIN
2201  00:F917  B0 2C                     BCS EXITOCT
2202  00:F919  64 64                     STZ MIN        ;ROLLED OVER
2203
2204  00:F91B  E6 65                     INC HR         ;INCREMENT HOUR
2205  00:F91D  A5 65                     LDA HR
2206  00:F91F  C9 01                     CMP #1
2207  00:F921  D0 24                     BNE TODINT8
2208  00:F923  A9 01              OCTOBER  LDA #DAYLITFLG ;IS DAYLIGHT SAVINGS ON
2209  00:F925  24 6A                     BIT DAYLIT
2210  00:F927  F0 1C                     BEQ EXITOCT
2211  00:F929  A5 67                     LDA MONTH
2212  00:F92B  C9 0A                     CMP #10        ;IS IT OCTOBER
2213  00:F92D  D0 16                     BNE EXITOCT
2214  00:F92F  A5 69                     LDA DAYWK      ;IS IT SUNDAY
2215  00:F931  C9 01                     CMP #$01
2216  00:F933  D0 10                     BNE EXITOCT    ;NO
2217  00:F935  A5 66                     LDA DAY        ;IS IT LAST SUNDAY
2218  00:F937  C9 19                     CMP #25
2219  00:F939  90 0A                     BCC EXITOCT
2220  00:F93B  A9 80                     LDA #DAYLPROG  ;CK IF ALREADY SET BACK
2221  00:F93D  14 6A                     TRB DAYLIT
2222  00:F93F  D0 04                     BNE EXITOCT
2223  00:F941  04 6A                     TSB DAYLIT
2224  00:F943  64 65                     STZ HR
2225  00:F945  80 4F              EXITOCT  BRA T2EXIT
```

'W65C134 Internal ROM Monitor ($F000)'
'MON4.ASM - Time of day clock IRQ routine'

```
2228          00:F947             TODINT8  EQU *
2229  00:F947  C9 18                    CMP #24
2230  00:F949  90 02                    BCC EXITA6
2231  00:F94B  64 65                    STZ HR        ;ROLLED OVER
2232  00:F94D  A5 65              EXITA6   LDA HR
2233  00:F94F  F0 02                    BEQ TODINT9
2234  00:F951  80 43                    BRA T2EXIT
2235
2236  00:F953  E6 69              TODINT9  INC DAYWK
2237  00:F955  A5 69                    LDA DAYWK
2238  00:F957  C9 07                    CMP #7
2239  00:F959  90 04                    BCC INCDAY
2240  00:F95B  A9 01                    LDA #1
2241  00:F95D  85 69                    STA DAYWK     ;ROLLED OVER
2242
2243  00:F95F  E6 66              INCDAY   INC DAY
2244  00:F961  A9 01                    LDA #DAYLITFLG ;IS DAY LIGHT SAVINGS ON
2245  00:F963  24 6A                    BIT DAYLIT
2246  00:F965  F0 06                    BEQ INCADAY    ;NO
2247  00:F967  A5 67                    LDA MONTH
2248  00:F969  C9 04                    CMP #4        ;IS IT APRIL
2249  00:F96B  F0 64                    BEQ APRIL
2250  00:F96D  A5 66              INCADAY  LDA DAY       ;INCREMENT DAYS
2251  00:F96F  A6 67                    LDX MONTH
2252  00:F971  DD 94 FF                CMP LASTDY-1,X
2253  00:F974  90 20                    BCC T2EXIT
2254
2255  00:F976  E0 02                    CPX #2        ;INCREMENT MONTH
2256  00:F978  D0 0C                    BNE INCMTH    ;NOT FEBRUARY
2257  00:F97A  A5 68                    LDA YR
2258  00:F97C  25 03                    AND %00000011  ;IS IT LEAP YR
2259  00:F97E  D0 06                    BNE INCMTH
2260
2261  00:F980  A5 66                    LDA DAY       ;ITS FEB AND LEAP YR
2262  00:F982  C9 1D                    CMP #29
2263  00:F984  F0 10                    BEQ T2EXIT
2264
2265  00:F986  A0 01              INCMTH   LDY #1        ;ROLLED OVER
2266  00:F988  84 66                    STY DAY
2267  00:F98A  E6 67                    INC MONTH
2268  00:F98C  A5 67                    LDA MONTH
2269  00:F98E  C9 0D                    CMP #13
2270  00:F990  90 04                    BCC T2EXIT
2271  00:F992  84 67                    STY MONTH     ;MONTH 1= JAN
2272
2273  00:F994  E6 68                    INC YR
2274
2275
2276          00:F996             T2EXIT   EQU *
2277  00:F996  A5 77                    LDA DISPTYP   ;CK IF ALARM ENABLED
2278  00:F998  89 08                    BIT #ALRMENAB
2279  00:F99A  F0 23                    BEQ EXITA
2280  00:F99C  A5 6B                    LDA ASEC      ;CHK IF ALARM CLOCK IS
2281  00:F99E  05 6C                    ORA AMIN      ;RUNNING
2282  00:F9A0  05 6D                    ORA AHR
```

```
2283  00:F9A2  05 6E          ORA ADAY
2284  00:F9A4  05 71          ORA ADAYWK
```

'W65C134 Internal ROM Monitor ($F000)'
'MON4.ASM - Time of day clock IRQ routine'

```
2285  00:F9A6  05 6F            ORA AMONTH
2286  00:F9A8  05 70            ORA AYR
2287  00:F9AA  F0 13            BEQ EXITA
2288                           ;CHK IF WE HAVE AN ALARM
2289  00:F9AC  A2 07            LDX #DFLTSEND-DFLTS-1
2290  00:F9AE  B5 6A      CKALARM  LDA ASEC-1,X
2291  00:F9B0  C9 FF            CMP #$FF
2292  00:F9B2  F0 04            BEQ CKAL1
2293  00:F9B4  D5 62            CMP SEC-1,X
2294  00:F9B6  D0 07            BNE EXITA
2295
2296  00:F9B8  CA         CKAL1    DEX
2297  00:F9B9  D0 F3            BNE CKALARM
2298  00:F9BB  A9 10            LDA #ALRMIRQ   ;SET ALARM FLAG
2299  00:F9BD  04 77            TSB DISPTYP
2300        00:F9BF         EXITA    EQU *
2301  00:F9BF  A2 07            LDX #DFLTSEND-DFLTS-1 ;LOAD TOD CHECKSUM
2302  00:F9C1  A9 00            LDA #00
2303  00:F9C3  18               CLC
2304  00:F9C4  75 62      CLKSUM   ADC SEC-1,X
2305  00:F9C6  CA               DEX
2306  00:F9C7  D0 FB            BNE CLKSUM
2307  00:F9C9  49 FF            EOR #$FF
2308  00:F9CB  85 60            STA TODCKS
2309  00:F9CD  FA               PLX
2310  00:F9CE  7A               PLY
2311  00:F9CF  68               PLA
2312  00:F9D0  40               RTI
2313
2314
2315
2316
2317
2318  00:F9D1  A5 69      APRIL    LDA DAYWK     ;IS IT SUNDAY
2319  00:F9D3  C9 01            CMP #$01
2320  00:F9D5  D0 96            BNE INCADAY    ;NO
2321  00:F9D7  A5 66            LDA DAY       ;IS IT 1ST SUNDAY
2322  00:F9D9  C9 08            CMP #8
2323  00:F9DB  90 03            BCC APR1
2324  00:F9DD  4C 6D F9         JMP INCADAY
2325
2326
2327  00:F9E0  A9 01      APR1     LDA #1
2328  00:F9E2  85 65            STA HR
2329  00:F9E4  80 B0            BRA T2EXIT
2330
2331                     STTL 'MON4.ASM - General ToD subroutines'
2332                     PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON4.ASM - General ToD subroutines'

```
2333
2334                    ;* Routine:  RTC_MODE
2335                    ;*
2336                    ;* Reg Used: ACC
2337                    ;* Var Used: DAYLIT
2338                    ;* Routines Called: NONE
2339                    ;* Returned Reg: NONE
2340                    ;*
2341
2342         00:F9E6            RTC_MODE EQU *         ;ENABLE/DISABLE DAYLIGHT
2343  00:F9E6  30 12                BMI RTC_ERR    ;VALUE EITHER 0 OR 1 ONLY
2344  00:F9E8  C9 02                CMP #$02
2345  00:F9EA  B0 0E                BCS RTC_ERR
2346  00:F9EC  F0 06                BEQ RTC0
2347  00:F9EE  A9 01                LDA #DAYLITFLG
2348  00:F9F0  04 6A                TSB DAYLIT     ;SAVINGS TIME
2349  00:F9F2  80 04                BRA RTC1
2350
2351  00:F9F4  A9 01            RTC0 LDA #DAYLITFLG
2352  00:F9F6  14 6A                TRB DAYLIT
2353  00:F9F8  18               RTC1     CLC
2354  00:F9F9  60                   RTS
2355
2356  00:F9FA  38               RTC_ERR  SEC           ;ERROR IN PARAMETERS
2357  00:F9FB  60                   RTS
2358
2359
2360
2361
2362                    ;* Routine:  RD_CLOCK
2363                    ;*
2364                    ;* Reg Used: ACC,Y,X
2365                    ;* Var Used: TMP0
2366                    ;* Routines Called: NONE
2367                    ;* Returned Reg: NONE           Y REG is RESTORED
2368                    ;*
2369
2370         00:F9FC            RD_CLOCK EQU *         ;READS CLOCK HR,MIN,SEC
2371  00:F9FC  5A                   PHY           ;MONTH,DAY,YR
2372  00:F9FD  85 7B                STA TMP0
2373  00:F9FF  86 7C                STX TMP0+1
2374  00:FA01  08                   PHP
2375  00:FA02  78                   SEI
2376  00:FA03  A0 00                LDY #0
2377  00:FA05  B9 63 00         RD_CLP   LDA SEC,Y
2378  00:FA08  91 7B                STA (TMP0),Y
2379  00:FA0A  C8                   INY
2380  00:FA0B  C0 07                CPY #7
2381  00:FA0D  D0 F6                BNE RD_CLP
2382  00:FA0F  28                   PLP
2383  00:FA10  7A                   PLY
2384  00:FA11  60                   RTS
2385
2386
2387
```

```
2388                    ;* Routine:  WR_CLOCK
2389                    ;*
```

'W65C134 Internal ROM Monitor ($F000)'
'MON4.ASM - General ToD subroutines'

```
2390                    ;* Reg Used: ACC,X,Y
2391                    ;* Var Used: TMP0
2392                    ;* Routines Called: NONE
2393                    ;* Returned Reg: NONE          Y REG is RESTORED
2394                    ;*
2395
2396          00:FA12               WR_CLOCK EQU *         ;WRITES CLOCK SEC,MIN,HR
2397  00:FA12  5A                       PHY          ;MONTH,DAY,YR,DofW
2398  00:FA13  85 7B                    STA TMP0
2399  00:FA15  86 7C                    STX TMP0+1
2400  00:FA17  A0 00                    LDY #0
2401  00:FA19  08                       PHP
2402  00:FA1A  78                       SEI
2403  00:FA1B  B1 7B               WR_CLP   LDA (TMP0),Y
2404  00:FA1D  99 63 00                 STA SEC,Y
2405  00:FA20  C8                       INY
2406  00:FA21  C0 07                    CPY #7
2407  00:FA23  D0 F6                    BNE WR_CLP
2408  00:FA25  28                       PLP
2409  00:FA26  7A                       PLY
2410  00:FA27  60                       RTS
2411
2412
2413                    ;
2414                    ;
2415                    ;       Alarm Clock:  To use the alarm clock, you set it using the routine
2416                    ;               below.  You must set all 7 parameters, but if you set
2417                    ;               any of them to $FF, it will be ignored.  For example,
2418                    ;               if you want the alarm clock to go off at 1:00 pm on
2419                    ;               Saturdays, you would set it to:
2420                    ;
2421                    ;               00 00 0D FF FF FF 07
2422                    ;
2423                    ;               This is zero seconds, zero minutes, 13 hours,
2424                    ;               any day, any month, any year, and the 7th day of the
2425                    ;               week (Sun = 1, Sat = 7).
2426                    ;
2427                    ;               Once set, you must enable the alarm clock.  This is
2428                    ;               by setting bit 3 of location $77 to a '1'.  If you
2429                    ;               want to turn it off (which saves a bit of interrupt
2430                    ;               time), set the enable bit to a 0.
2431                    ;
2432                    ;               When the appropriate time hits, the clock interrupt
2433                    ;               will set a flag.  The flag is bit 4 of location $77.
2434                    ;               A '1' in that bit indicates the alarm went off.  The
2435                    ;               flag stays set until you reset it.
2436                    ;
2437                    ;               Remember that 'ANY' is just that; if you set the alarm
2438                    ;               for Jan 3, 1990 by using: FF FF FF 03 01 5A FF,
2439                    ;               you'll get an alarm set every second all day long on
2440                    ;               Jan 3, 1990.  Note also that if you want a specific
2441                    ;               time and date, you should set the day of the week to
2442                    ;               'ANY' to avoid a mis-match.
2443                    ;
2444                    ;               If you want every other day, then you'll have to set
```

```
2445            ;           the alarm for every day and check for every other day
2446            ;           by yourself.
```

'W65C134 Internal ROM Monitor ($F000)'
'MON4.ASM - General ToD subroutines'

```
2447                    ;
2448                    ;
2449                    ;
2450                    ;
2451                    ;
2452                    ;
2453
2454                    ;* Routine:  WR_ACLOCK
2455                    ;*
2456                    ;* Reg Used: ACC,X
2457                    ;* Var Used: TMP0
2458                    ;* Routines Called: NONE
2459                    ;* Returned Reg: NONE          Y REG is RESTORED
2460                    ;*
2461
2462          00:FA28               WR_ACLOCK EQU *         ;WRITES ALARM CLOCK DATA
2463  00:FA28  5A                       PHY           ;MONTH,DAY,YR
2464  00:FA29  85 7B                    STA TMP0      ;A has low address of alarm time
2465  00:FA2B  86 7C                    STX TMP0+1    ;X has high address of alarm time
2466  00:FA2D  08                       PHP
2467  00:FA2E  78                       SEI           ;no alarms in the middle of setting it, please.
2468  00:FA2F  A0 00                    LDY #0
2469  00:FA31  B1 7B               WR_ACLP  LDA (TMP0),Y
2470  00:FA33  99 6B 00                 STA ASEC,Y
2471  00:FA36  C8                       INY
2472  00:FA37  C0 07                    CPY #7
2473  00:FA39  D0 F6                    BNE WR_ACLP
2474  00:FA3B  28                       PLP
2475  00:FA3C  7A                       PLY
2476  00:FA3D  60                       RTS
2477
2478
2479
2480                    ;* Routine:  SNDTOD
2481                    ;*
2482                    ;* Reg Used: ACC,X
2483                    ;* Var Used:
2484                    ;* Routines Called: SNDT1,SPAC,OUTCH,SNDTIME,CRLF
2485                    ;* Returned Reg: NONE
2486                    ;*
2487                    ;* WARNING: if the time rolls in the middle of this, the result is wrong.
2488                    ;*
2489
2490  00:FA3E  A5 69               SNDTOD   LDA DAYWK     ;SEND OUT TOD CLOCK
2491  00:FA40  20 87 FA                 JSR SNDT1
2492  00:FA43  20 09 FC                 JSR SPAC
2493  00:FA46  A5 67                    LDA MONTH
2494  00:FA48  20 87 FA                 JSR SNDT1
2495  00:FA4B  A9 2F                    LDA #'/'
2496  00:FA4D  20 11 FC                 JSR OUTCH
2497  00:FA50  A5 66                    LDA DAY
2498  00:FA52  20 87 FA                 JSR SNDT1
2499  00:FA55  A9 2F                    LDA #'/'
2500  00:FA57  20 11 FC                 JSR OUTCH
2501  00:FA5A  A9 13                    LDA #19
```

```
2502  00:FA5C  20 87 FA     JSR SNDT1
2503  00:FA5F  A5 68        LDA YR
```

'W65C134 Internal ROM Monitor ($F000)'
'MON4.ASM - General ToD subroutines'

```
2504  00:FA61  20 87 FA          JSR SNDT1
2505  00:FA64  20 04 FC          JSR SPAC2
2506  00:FA67  20 6D FA          JSR SNDTIME    ;DISPLAY TIME
2507  00:FA6A  4C F4 FB          JMP CRLF
2508
2509
2510                ;* Routine:  SNDTIME
2511                ;*
2512                ;* Reg Used: ACC,X
2513                ;* Var Used:
2514                ;* Routines Called: SNDT1,OUTCH,BINASC,WRTWO
2515                ;* Returned Reg: NONE
2516                ;*
2517                ;* WARNING: if the time rolls in the middle, output will be wrong
2518                ;*
2519
2520  00:FA6D  A5 65          SNDTIME  LDA HR
2521  00:FA6F  20 87 FA           JSR SNDT1
2522  00:FA72  A9 3A              LDA #':'
2523  00:FA74  20 11 FC           JSR OUTCH
2524  00:FA77  B0 17              BCS SNDTDONE
2525  00:FA79  A5 64              LDA MIN
2526  00:FA7B  20 87 FA           JSR SNDT1
2527  00:FA7E  A9 3A              LDA #':'
2528  00:FA80  20 11 FC           JSR OUTCH
2529  00:FA83  B0 0B              BCS SNDTDONE
2530  00:FA85  A5 63              LDA SEC
2531
2532  00:FA87  20 31 FE       SNDT1    JSR BIN2DEC
2533  00:FA8A  20 59 FC           JSR BINASC
2534  00:FA8D  20 F8 FB           JSR WRTWO
2535  00:FA90  60             SNDTDONE RTS
2536
2537
2538
2539  00:FA91                     END
2540
2541  00:FA91                     INCLUDE MON5.ASM
2542                      .STTL 'MON5.ASM - Serial Routines'
2543                      .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON5.ASM - Serial Routines'

```
2544                        ;06-23-1993
2545
2546                    ;* Routine:  IRQAR
2547                    ;*
2548                    ;* Reg Used: ACC & Y
2549                    ;* Var Used: SFLAG,SINEND
2550                    ;* Routines Called: FLUSH_SERIAL_BUFF
2551                    ;* Returned Reg: NONE
2552                    ;*
2553
2554                                ;CALLED BY IRQ ROUTINE
2555         00:FA91            IRQAR   EQU *       ;QUEUE UP SERIAL BYTE
2556  00:FA91  48                       PHA
2557  00:FA92  5A                       PHY
2558  00:FA93  A5 23                    LDA ARTD     ;GET DATA
2559  00:FA95  29 7F                    AND #$7F     ;MAKE ASCII
2560  00:FA97  D0 03                    BNE RQ5
2561  00:FA99  4C 2B FB                 JMP RECDONE  ;CHK FOR NULL
2562
2563
2564  00:FA9C  C9 19            RQ5     CMP #$19     ;QUICK CHK FOR CNTRL CHAR
2565  00:FA9E  B0 2F                    BCS RQ8
2566  00:FAA0  C9 03                    CMP #CNTRLC  ;CHK IF CONTROL 'C'
2567  00:FAA2  D0 14                    BNE RQ6
2568  00:FAA4  20 B4 FD         RECSC0  JSR FLUSH_SERIAL_BUFF ;GOT A CONTROL 'C'
2569  00:FAA7  A5 72                    LDA SFLAG
2570  00:FAA9  29 FE                    AND #$FF-SFLG  ;CLR CHAR PENDING FLG
2571  00:FAAB  09 02                    ORA #CFLG    ;SO FLUSH QUEUE
2572  00:FAAD  89 04                    BIT #XONOFLG
2573  00:FAAF  F0 02                    BEQ RECSC1
2574  00:FAB1  09 90                    ORA #SXONFLG+LASTXONOF
2575  00:FAB3  85 72            RECSC1  STA SFLAG
2576  00:FAB5  4C 2B FB                 JMP RECDONE
2577
2578
2579
2580  00:FAB8  C9 18            RQ6     CMP #CNTRLX  ;CHK IF CONTROL 'X'
2581  00:FABA  F0 E8                    BEQ RECSC0
2582  00:FABC  48                       PHA
2583  00:FABD  A5 72                    LDA SFLAG
2584  00:FABF  89 04                    BIT #XONOFLG
2585  00:FAC1  F0 0B                    BEQ RQ7A
2586  00:FAC3  68                       PLA
2587  00:FAC4  C9 13                    CMP #XOFF    ;CHK XON/XOFF PROTOC
2588  00:FAC6  F0 53                    BEQ RECSXOFF
2589  00:FAC8  C9 11                    CMP #XON
2590  00:FACA  F0 55                    BEQ RECSXON
2591  00:FACC  80 01                    BRA RQ8
2592
2593  00:FACE  68               RQ7A    PLA
2594  00:FACF  A4 75            RQ8     LDY SININDX  ;GET CURRENT PTR
2595  00:FAD1  C8                       INY
2596  00:FAD2  C4 76                    CPY SINEND   ;IS BUFF FULL
2597  00:FAD4  D0 0D                    BNE RQ9
2598  00:FAD6  48                       PHA
```

```
2599  00:FAD7  A5 76        LDA SINEND
2600  00:FAD9  1A           INC A       ;REMOVE OLDEST CHAR
```

'W65C134 Internal ROM Monitor ($F000)'
'MON5.ASM - Serial Routines'

```
2601  00:FADA  C5 56          CMP SINCNT
2602  00:FADC  D0 02          BNE RQ8A
2603  00:FADE  A9 00          LDA #$00
2604  00:FAE0  85 76      RQ8A    STA SINEND
2605  00:FAE2  68              PLA
2606  00:FAE3  C4 56      RQ9     CPY SINCNT    ;GET SIZE OF QUEUE
2607  00:FAE5  90 02          BCC RECS11    ;CK FOR WRAP AROUND
2608  00:FAE7  A0 00          LDY #0        ;WRAP AROUND
2609  00:FAE9  84 75      RECS11  STY SININDX
2610  00:FAEB  91 52          STA (SINPTR),Y ;SAVE DATA IN QUE
2611  00:FAED  A9 01          LDA #SFLG
2612  00:FAEF  04 72          TSB SFLAG
2613  00:FAF1  C8             INY           ;IS BUFF ALMOST FULL?
2614  00:FAF2  C8             INY
2615  00:FAF3  C8             INY
2616  00:FAF4  C8             INY
2617  00:FAF5  C8             INY
2618
2619  00:FAF6  C4 56          CPY SINCNT    ;GET SIZE OF QUEUE
2620  00:FAF8  90 05          BCC RQ0
2621  00:FAFA  98             TYA
2622  00:FAFB  38             SEC
2623  00:FAFC  E5 56          SBC SINCNT
2624  00:FAFE  A8             TAY
2625  00:FAFF  C4 76      RQ0     CPY SINEND    ;We'll kill buffer control if IN+5 = END
2626  00:FB01  D0 28          BNE RECDONE   ;branch if we haven't hit the stop point
2627  00:FB03
2628        00:FB03       RQ1     EQU *
2629  00:FB03  A5 72          LDA SFLAG     ;XON/XOFF OR DTR?
2630  00:FB05  89 04          BIT #XONOFLG
2631  00:FB07  F0 0C          BEQ SNDXOFFHW
2632  00:FB09  89 10          BIT #LASTXONOF
2633  00:FB0B  F0 1E          BEQ RECDONE   ;ALREADY SENT XOFF
2634  00:FB0D  29 EF          AND #$FF-LASTXONOF   ;SEND XOFF NXT XMIT IRQ
2635  00:FB0F  09 40          ORA #SXOFFLG  ;XON/XOFF HS
2636  00:FB11  85 72          STA SFLAG
2637  00:FB13  80 10          BRA RECDONET
2638
2639        00:FB15       SNDXOFFHW EQU *
2640  00:FB15  A9 04          LDA #DTR      ;SETUP HW HS
2641  00:FB17  04 20          TSB PD6       ;DTR HIGH
2642  00:FB19  80 10          BRA RECDONE   ;HOLD OFF FURTHER XMISSION
2643
2644
2645  00:FB1B  A9 08      RECSXOFF LDA #SNDOVF    ;OVERRUNNING OTHER GUYS
2646  00:FB1D  04 72          TSB SFLAG     ;SERIAL INPUT BUFFER SO
2647  00:FB1F  80 0A          BRA RECDONE
2648
2649
2650  00:FB21  A9 08      RECSXON  LDA #SNDOVF    ;OVERFLOW ON XMIT OVER
2651  00:FB23  14 72          TRB SFLAG
2652
2653  00:FB25  A5 22      RECDONET LDA ACSR      ;TURN ON SERIAL XMIT
2654  00:FB27  09 03          ORA #$03
2655  00:FB29  85 22          STA ACSR
```

```
2656
2657  00:FB2B  7A              RECDONE  PLY          ;RECEIVE IRQ DONE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON5.ASM - Serial Routines'

```
2658  00:FB2C  68                 PLA
2659  00:FB2D  40                 RTI
2660                   .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON5.ASM - Serial Routines'

```
2661
2662                    ;* Routine:  IRQAT
2663                    ;*
2664                    ;* Reg Used: ACC & Y
2665                    ;* Var Used: SFLAG,SOUTINDX
2666                    ;* Routines Called: NONE
2667                    ;* Returned Reg: NONE
2668                    ;*
2669                                ;CALLED BY IRQ ROUTINE
2670          00:FB2E           IRQAT    EQU *        ;DEQUEUE SERIAL BYTE
2671  00:FB2E  48                    PHA          ;FROM OUTPUT BUFFER
2672  00:FB2F  5A                    PHY          ;SEE OUTCH ROUTINE
2673  00:FB30  A5 72                 LDA SFLAG     ;CK IF WE ARE OVERFLOWING
2674  00:FB32  29 C0                 AND #SXOFFLG+SXONFLG
2675  00:FB34  F0 1C                 BEQ SQ0A      ;QUICK CK FOR CNTRL FLGS
2676  00:FB36  C9 C0                 CMP #SXOFFLG+SXONFLG
2677  00:FB38  F0 14                 BEQ IRQATERR
2678  00:FB3A  89 40                 BIT #SXOFFLG
2679  00:FB3C  D0 08                 BNE SNDXOFF    ;SEND XOFF
2680  00:FB3E  A9 80           SNDXON   LDA #SXONFLG
2681  00:FB40  14 72                 TRB SFLAG
2682  00:FB42  A9 11                 LDA #XON
2683  00:FB44  80 3F                 BRA SQ1
2684
2685
2686          00:FB46           SNDXOFF  EQU *
2687  00:FB46  A9 50                 LDA #SXOFFLG+LASTXONOF
2688  00:FB48  14 72                 TRB SFLAG
2689  00:FB4A  A9 13                 LDA #XOFF
2690  00:FB4C  80 37                 BRA SQ1
2691
2692  00:FB4E  A9 C0           IRQATERR LDA #SXONFLG+SXOFFLG
2693  00:FB50  14 72                 TRB SFLAG
2694  00:FB52  A5 22           SQ0A     LDA ACSR
2695  00:FB54  29 02                 AND #$02          ;XMIT IRQ ON
2696  00:FB56  D0 10                 BNE XIRQ1
2697  00:FB58  A4 73           XIRQ2    LDY SOUTINDX   ;IS BUFFER NOW EMPTY
2698  00:FB5A  C4 74                 CPY SOUTEND
2699  00:FB5C  D0 1C                 BNE XIRQ3
2700  00:FB5E  A5 22                 LDA ACSR      ;DISABLE SERIAL XMIT
2701  00:FB60  29 02                 AND #$02      ;EVERYTHING IS OUT OF
2702  00:FB62  D0 43                 BNE SQ3A
2703  00:FB64  A9 00                 LDA #00
2704  00:FB66  80 1D                 BRA SQ1
2705
2706  00:FB68  A5 72           XIRQ1    LDA SFLAG
2707  00:FB6A  89 04                 BIT #XONOFLG
2708  00:FB6C  D0 06                 BNE XIRQ1A
2709  00:FB6E  A5 1C                 LDA PD4       ;DSR = P47
2710  00:FB70  30 35                 BMI SQ3A
2711  00:FB72  80 E4                 BRA XIRQ2
2712
2713  00:FB74  89 08           XIRQ1A   BIT #SNDOVF
2714  00:FB76  F0 E0                 BEQ XIRQ2
2715  00:FB78  80 2D                 BRA SQ3A
```

```
2716
2717  00:FB7A  C8              XIRQ3    INY
```

'W65C134 Internal ROM Monitor ($F000)'
'MON5.ASM - Serial Routines'

```
2718  00:FB7B  C4 57          CPY SOUTCNT    ;GET SIZE OF QUEUE
2719  00:FB7D  90 02          BCC SQ0
2720  00:FB7F  A0 00          LDY #0
2721  00:FB81  84 73     SQ0  STY SOUTINDX
2722  00:FB83  B1 54          LDA (SOUTPTR),Y ;GET DATA FROM QUEUE
2723  00:FB85  85 23     SQ1  STA ARTD       ;SEND DATA
2724  00:FB87  A5 72          LDA SFLAG
2725  00:FB89  89 04          BIT #XONOFLG
2726  00:FB8B  D0 04          BNE SQ2
2727  00:FB8D  A5 1C          LDA PD4
2728  00:FB8F  30 21          BMI SQ3        ;P47 HI     DSR TEST
2729  00:FB91  A5 72     SQ2  LDA SFLAG
2730  00:FB93  89 08          BIT #SNDOVF
2731  00:FB95  D0 1B          BNE SQ3
2732  00:FB97  A4 73          LDY SOUTINDX
2733  00:FB99  C4 74          CPY SOUTEND    ;IS BUFF EMPTY
2734  00:FB9B  F0 15          BEQ SQ3
2735  00:FB9D  A5 22          LDA ACSR
2736  00:FB9F  29 FC          AND #$FC
2737  00:FBA1  09 01          ORA #SON
2738  00:FBA3  85 22          STA ACSR
2739  00:FBA5  80 84          BRA RECDONE
2740
2741  00:FBA7  A9 02     SQ3A LDA #$02       ;set the output high to prevent falsing
2742  00:FBA9  04 20          TSB PD6        ;when we shut it off.
2743  00:FBAB  A9 03          LDA #$03       ;NO MORE CHAR
2744  00:FBAD  14 22          TRB ACSR       ;turn ACSR off
2745  00:FBAF  4C 2B FB       JMP RECDONE
2746
2747  00:FBB2  A9 03     SQ3  LDA #$03       ;TURN ON XMIT & RECV
2748  00:FBB4  04 22          TSB ACSR
2749  00:FBB6  4C 2B FB       JMP RECDONE
2750
2751                    .PAGE 'HARDWARE HANDSHAKE'
```

'W65C134 Internal ROM Monitor ($F000)'
'MON5.ASM - Serial Routines'

```
2752
2753
2754  00:FBB9  48          NE47    PHA          ;HARDWARE HS INPUT
2755  00:FBBA  A9 08               LDA #$08     ;DSR
2756  00:FBBC  04 2C               TSB IFR1     ;RESET IRQ
2757  00:FBBE  A5 72               LDA SFLAG
2758  00:FBC0  89 04               BIT #XONOFLG
2759  00:FBC2  D0 08               BNE NE47A    ;XON/XOFF MODE
2760  00:FBC4  A5 22               LDA ACSR
2761  00:FBC6  09 03               ORA #SON+$02
2762  00:FBC8  85 22               STA ACSR
2763  00:FBCA  80 04               BRA NE47B
2764
2765  00:FBCC  A9 08          NE47A   LDA #$08     ;DISABLE NE47 IRQS
2766  00:FBCE  14 2D               TRB IER1     ;SHOULD BE IN XON/XOFF MODE
2767  00:FBD0  68          NE47B   PLA
2768  00:FBD1  40               RTI
2769                    ;
2770                    ;
2771                    ;
2772                    ; SERIAL WRITES
2773                    ;
2774
2775                    ;* Routine:  WR_ADDR
2776                    ;*
2777                    ;* Reg Used: ACC & X
2778                    ;* Var Used: TMPC,TMP0
2779                    ;* Routines Called: WROB,BINASC,WRTWO
2780                    ;* Returned Reg: NONE          X & Y REG are PRESERVED
2781                    ;*
2782
2783  00:FBD2  85 7B          WR_ADDR  STA TMP0     ;WRITE ADDRESS IN TMP0
2784  00:FBD4  86 7C               STX TMP0+1
2785                              ;WRITE ADDR FROM TMP0
2786  00:FBD6  A2 01          WROA    LDX #1       ;DISPLAY TMP0
2787  00:FBD8  80 06               BRA WROA1
2788
2789  00:FBDA  A2 07          WROA4   LDX #TMP4-TMP0+1
2790  00:FBDC  80 02               BRA WROA1    ;USUALLY CHKSUM
2791
2792  00:FBDE  A2 DE          WRPC    LDX #PCL-TMP0+1 ;DISPLAY PROGRAM CTR
2793  00:FBE0  B5 7A          WROA1   LDA TMP0-1,X
2794  00:FBE2  48                  PHA
2795  00:FBE3  B5 7B               LDA TMP0,X
2796  00:FBE5  20 EB FB            JSR WROB
2797  00:FBE8  B0 18               BCS WRTWORTS   ;GOT A CNTRL 'C'
2798  00:FBEA  68                  PLA
2799
2800  00:FBEB  DA             WROB    PHX          ;WRITE BYTE AS 2 HEX CHAR
2801  00:FBEC  20 59 FC            JSR BINASC   ;UNPACK BYTE DATA INTO
2802                             ;TWO ASCII CHARS.
2803  00:FBEF  20 F8 FB            JSR WRTWO    ;X,A=CHARS
2804  00:FBF2  FA                  PLX
2805  00:FBF3  60                  RTS
2806
```

```
2807
2808  00:FBF4  A9 0D          CRLF    LDA #$0D      ;OUTPUT CR WITH AUTO LF
```

'W65C134 Internal ROM Monitor ($F000)'
'MON5.ASM - Serial Routines'

```
2809  00:FBF6  80 19          BRA OUTCH
2810
2811  00:FBF8  48          WRTWO   PHA        ;WRITE 2 CHARS-X,A=CHARS
2812  00:FBF9  8A              TXA        ;WRITE X FIRST
2813  00:FBFA  20 11 FC        JSR OUTCH
2814  00:FBFD  B0 03           BCS WRTWORTS
2815  00:FBFF  68              PLA
2816  00:FC00  80 0F           BRA OUTCH
2817
2818  00:FC02  68          WRTWORTS PLA        ;GOT A CNTRL 'C'
2819  00:FC03  60          SPACRTS  RTS
2820
2821
2822                ;* Routine:  SPAC2
2823                ;*
2824                ;* Reg Used: ACC
2825                ;* Var Used: NONE
2826                ;* Routines Called: SPAC
2827                ;* Returned Reg: Acc X & Y REG are PRESERVED
2828                ;*
2829
2830
2831  00:FC04  20 09 FC    SPAC2   JSR SPAC
2832  00:FC07  B0 FA           BCS SPACRTS   ;GOT ^C
2833  00:FC09  48          SPAC    PHA
2834  00:FC0A  A9 20           LDA #' '
2835  00:FC0C  20 11 FC        JSR OUTCH
2836  00:FC0F  68              PLA
2837  00:FC10  60              RTS
2838
2839
2840                ;* Routine:  OUTCH
2841                ;*
2842                ;* Reg Used: Acc & Y
2843                ;* Var Used: SOUTEND
2844                ;* Routines Called: NONE
2845                ;* Returned Reg: Acc          X & Y REG are PRESERVED
2846                ;*
2847
2848
2849       00:FC11       OUTCH   EQU *      ;PLACE CHAR IN OUTPUT
2850  00:FC11  48              PHA        ;QUEUE
2851  00:FC12  5A              PHY        ;SAVE YREG
2852  00:FC13  48              PHA
2853  00:FC14  A4 74        OUTCH1  LDY SOUTEND   ;CK IF CURRENT QUEUE POS
2854  00:FC16  C8              INY        ;POINT TO NXT DATA
2855  00:FC17  C4 57           CPY SOUTCNT   ;DO WE WRAP
2856  00:FC19  90 02           BCC SNDSD2    ;NO
2857  00:FC1B  A0 00           LDY #0        ;WE WRAPPED
2858  00:FC1D  C4 73        SNDSD2  CPY SOUTINDX  ;DID WE OVERRUN QUEUE
2859  00:FC1F  F0 F3           BEQ OUTCH1    ;YES, SO WAIT
2860
2861  00:FC21  A5 72           LDA SFLAG     ;CK FOR CNTRL 'C'
2862  00:FC23  89 02           BIT #CFLG
2863  00:FC25  D0 2D           BNE OUTCH3    ;GOT CNTRL 'C'
```

2864 00:FC27 78 SEI
2865

'W65C134 Internal ROM Monitor ($F000)'
'MON5.ASM - Serial Routines'

```
2866  00:FC28  84 74          STY SOUTEND
2867  00:FC2A  68             PLA         ;GET DATA
2868  00:FC2B  91 54          STA (SOUTPTR),Y ;PUT DATA IN QUEUE
2869  00:FC2D  A5 72          LDA SFLAG     ;CK IF HWHS OR SOFTWARE HS
2870  00:FC2F  89 04          BIT #XONOFLG
2871  00:FC31  D0 06          BNE SNDOUT1   ;SW HS ON
2872  00:FC33  A5 1C          LDA PD4       ;CK IF WE ARE TO HOLD OFF
2873  00:FC35  30 18          BMI OUTCH2    ;DONT TURN ON SERIAL IRQS
2874  00:FC37  80 04          BRA SNDOUT2
2875
2876  00:FC39  89 08     SNDOUT1  BIT #SNDOVF    ;CHK FOR SW HS
2877  00:FC3B  D0 12          BNE OUTCH2    ;HAVE A XOFF SO DONT XMIT
2878
2879  00:FC3D  A5 22     SNDOUT2  LDA ACSR
2880  00:FC3F  89 01          BIT #SON      ;IS SERIAL IRQ ON
2881  00:FC41  F0 08          BEQ OUTCH1A   ;NO
2882  00:FC43  89 02          BIT #$02      ;ARE WE IN FAST SERIAL
2883  00:FC45  F0 08          BEQ OUTCH2    ;MODE--YES
2884  00:FC47  29 FD          AND #$FF-$02  ;GOTO FAST SERIAL MODE
2885  00:FC49  80 02          BRA OUTCH1B
2886
2887  00:FC4B  09 03     OUTCH1A  ORA #SON+$02   ;SERIAL IRQ SINGLE CHR MODE
2888  00:FC4D  85 22     OUTCH1B  STA ACSR
2889  00:FC4F  58        OUTCH2   CLI
2890  00:FC50  7A             PLY
2891  00:FC51  68             PLA
2892  00:FC52  18             CLC
2893  00:FC53  60             RTS
2894
2895  00:FC54  68        OUTCH3   PLA         ;RESTORE STK ON CNTRL 'C'
2896  00:FC55  7A             PLY
2897  00:FC56  68             PLA
2898  00:FC57  38             SEC
2899  00:FC58  60             RTS
2900
2901
2902                ;* Routine:  BINASC
2903                ;*
2904                ;* Reg Used: Acc & X
2905                ;* Var Used: NONE
2906                ;* Routines Called: ASCII
2907                ;* Returned Reg: Acc & X         Y REG is PRESERVED
2908                ;*
2909
2910  00:FC59  48        BINASC   PHA         ;CONVERT BYTE TO 2
2911  00:FC5A  4A             LSR A        ;ASCII CHAR
2912  00:FC5B  4A             LSR A
2913  00:FC5C  4A             LSR A
2914  00:FC5D  4A             LSR A
2915  00:FC5E  20 65 FC       JSR ASCII     ;CONVERT TO ASCII
2916  00:FC61  AA             TAX
2917  00:FC62  68             PLA
2918  00:FC63  29 0F          AND #LOWNIB
2919                         ;FALL THRU TO ASCII
2920  00:FC65  18        ASCII    CLC
```

```
2921  00:FC66  69 06        ADC #6
2922  00:FC68  69 F0        ADC #HINIB
```

'W65C134 Internal ROM Monitor ($F000)'
'MON5.ASM - Serial Routines'

```
2923  00:FC6A  90 02              BCC ASC1
2924  00:FC6C  69 06              ADC #$06
2925  00:FC6E  69 3A        ASC1  ADC #'9'+1   ;GT '9'
2926  00:FC70  60                 RTS
2927
2928                      .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON5.ASM - Serial Routines'

```
2929
2930                    ;* Routine:  GETCH
2931                    ;*
2932                    ;* Reg Used: Acc
2933                    ;* Var Used: NONE
2934                    ;* Routines Called: RD_CHAR,OUTCH
2935                    ;* Returned Reg: Acc          X & Y REG are PRESERVED
2936                    ;*
2937
2938 00:FC71 20 8F FC      GETCH   JSR RD_CHAR   ;WAIT UNTIL WE GET A CHAR
2939 00:FC74 B0 16              BCS GETC4     ;HAD ^C,SO ERROR
2940 00:FC76 C9 00              CMP #$00
2941 00:FC78 F0 F7              BEQ GETCH     ;WAIT FOR INPUT
2942 00:FC7A 48                 PHA           ;SAVE DATA
2943 00:FC7B A5 72              LDA SFLAG     ;CHK IF ECHO
2944 00:FC7D 29 20              AND #ECHOFF
2945 00:FC7F D0 08              BNE GETC3
2946 00:FC81 68                 PLA
2947 00:FC82 C9 0A              CMP #$0A      ;LINE FEED
2948 00:FC84 F0 04              BEQ GETC3A
2949 00:FC86 4C 11 FC           JMP OUTCH     ;ECHO BACK INPUT
2950                           ;CY SET SO ERROR
2951
2952 00:FC89 68              GETC3   PLA
2953 00:FC8A 18              GETC3A  CLC
2954 00:FC8B 60                 RTS
2955
2956 00:FC8C A9 03           GETC4   LDA #3        ;%rev  SEND ^C TO CALLER
2957 00:FC8E 60                 RTS
2958
2959
2960
2961
2962                    ;* Routine:  RD_CHAR
2963                    ;*
2964                    ;* Reg Used: ACC & Y
2965                    ;* Var Used: SINEND,SFLAG
2966                    ;* Routines Called: CK_CONTC
2967                    ;* Returned Reg: Acc          X & Y REG are PRESERVED
2968                    ;*
2969
2970
2971         00:FC8F    RD_CHAR  EQU *         ;CHK FOR CHARACTER
2972 00:FC8F A5 72              LDA SFLAG     ;GET SERIAL BYTE
2973 00:FC91 29 03              AND #SFLG+CFLG ;FROM INPUT QUEUE
2974 00:FC93 F0 3E              BEQ RD_CH0
2975 00:FC95 5A                 PHY
2976 00:FC96 78                 SEI           ;PUT THERE BY RECSBYTE
2977 00:FC97 A4 76              LDY SINEND    ;CK IF CURRENT QUEUE POS
2978 00:FC99 C8                 INY           ;POINT TO NXT DATA
2979 00:FC9A C4 56              CPY SINCNT    ;DO WE WRAP
2980 00:FC9C 90 02              BCC GETSD4
2981 00:FC9E A0 00              LDY #0        ;WE WRAPPED
2982 00:FCA0 20 D8 FC        GETSD4  JSR CK_CONTC  ;RESET ^C FLAG
2983 00:FCA3 B0 30              BCS RD_CH2    ;HAD ^C
```

```
2984  00:FCA5  84 76        STY SINEND
2985  00:FCA7  B1 52        LDA (SINPTR),Y ;GET DATA
```

'W65C134 Internal ROM Monitor ($F000)'
'MON5.ASM - Serial Routines'

```
2986  00:FCA9  48              PHA
2987  00:FCAA  C4 75           CPY SININDX    ;IS SAME AS END OF QUEUE
2988  00:FCAC  D0 22           BNE GETSD3
2989
2990  00:FCAE  A5 72           LDA SFLAG      ;CK IF XON/XOFF
2991  00:FCB0  89 04           BIT #XONOFLG   ;OR HARDWARE HS
2992  00:FCB2  F0 14           BEQ GETSD1
2993  00:FCB4  89 10           BIT #LASTXONOF ;HAS XON ALREADY BEEN SENT?
2994  00:FCB6  D0 14           BNE GETSD2
2995  00:FCB8  A9 90           LDA #SXONFLG+LASTXONOF
2996  00:FCBA  04 72           TSB SFLAG
2997  00:FCBC  A5 22           LDA ACSR
2998  00:FCBE  89 01           BIT #SON
2999  00:FCC0  D0 0A           BNE GETSD2
3000  00:FCC2  09 03           ORA #SON+$02
3001  00:FCC4  85 22           STA ACSR
3002  00:FCC6  80 04           BRA GETSD2
3003
3004        00:FCC8        GETSD1   EQU *         ;HANDLE HARDWARE HS
3005  00:FCC8  A9 04           LDA #DTR       ;DTR LOW, OK FOR
3006  00:FCCA  14 20           TRB PD6        ;OTHER GUY TO SEND
3007  00:FCCC  A9 01         GETSD2   LDA #SFLG      ;NO MORE SERIAL CHARS
3008  00:FCCE  14 72           TRB SFLAG
3009  00:FCD0  68            GETSD3   PLA           ;GET DATA
3010  00:FCD1  58              CLI
3011  00:FCD2  7A              PLY
3012  00:FCD3  18            RD_CH0   CLC           ;NO DATA RETURN NULL
3013  00:FCD4  60              RTS
3014
3015  00:FCD5  58            RD_CH2   CLI           ;NG ^C, RETURN NULL
3016  00:FCD6  7A              PLY           ;AND CY = 1
3017  00:FCD7  60              RTS
3018
3019
3020                ;* Routine:  CK_CONTC
3021                ;*
3022                ;* Reg Used: Acc
3023                ;* Var Used: SFLAG
3024                ;* Routines Called: NONE
3025                ;* Returned Reg: Acc
3026                ;*
3027
3028        00:FCD8        CK_CONTC EQU *         ;CHK  FOR CONTROL "C"
3029  00:FCD8  A9 02           LDA #CFLG
3030  00:FCDA  14 72           TRB SFLAG
3031  00:FCDC  F0 F5           BEQ RD_CH0     ;CLR CY NOT ^C
3032  00:FCDE  38              SEC           ;FOUND CNTRL 'C'
3033  00:FCDF  60              RTS
3034
3035
3036                ; READ HEX ADR, RETURN HO IN TMP0,
3037                ; LO IN TMP0+1 AND CY=1
3038                ; IF SPACE,CR, OR COMMA CY=0
3039
3040
```

```
3041                    ;* Routine:  RDOA
3042                    ;*
```

'W65C134 Internal ROM Monitor ($F000)'
'MON5.ASM - Serial Routines'

```
3043                    ;* Reg Used: Acc
3044                    ;* Var Used: TMP0
3045                    ;* Routines Called: RDOB
3046                    ;* Returned Reg: Acc
3047                    ;*
3048
3049 00:FCE0 64 7B      RDOA    STZ TMP0      ;CLR ADDRESS SO ALL ZEROS
3050 00:FCE2 64 7C              STZ TMP0+1
3051 00:FCE4 20 FD FC           JSR RDOB      ;READ 2 CHAR BYTE
3052 00:FCE7 90 0B              BCC RDOA2A    ;SPACE,CR OR COMMA
3053
3054 00:FCE9 85 7C              STA TMP0+1
3055 00:FCEB 20 FD FC           JSR RDOB      ;NO NEED TO CK FOR
3056 00:FCEE 90 07              BCC RDOA2
3057 00:FCF0 85 7B              STA TMP0      ;DELIMETER JUST SAVE
3058 00:FCF2 18                 CLC           ;C is clear if addr is OK
3059 00:FCF3 60                 RTS
3060
3061 00:FCF4 20 04 FC   RDOA2A  JSR SPAC2
3062 00:FCF7 85 7B      RDOA2   STA TMP0
3063 00:FCF9 64 7C              STZ TMP0+1
3064 00:FCFB 38                 SEC           ;carry set if Addr = 0
3065 00:FCFC 60                 RTS
3066
3067
3068                    ; READ HEX BYTE AND RETURN IN A, AND CY=1
3069                    ; IF SPACE,CR OR COMMA CY=0
3070
3071                    ;* Routine:  RDOB
3072                    ;*
3073                    ;* Reg Used: ACC & X
3074                    ;* Var Used: TMP0
3075                    ;* Routines Called: GETCH,ASCBIN
3076                    ;* Returned Reg: Acc          X & Y REGS are PRESERVED
3077                    ;*
3078
3079 00:FCFD DA          RDOB    PHX           ;SAVE X
3080 00:FCFE 20 71 FC           JSR GETCH
3081 00:FD01 B0 34              BCS RDOB4     ;CNTRL 'C'
3082 00:FD03 C9 0D              CMP #$0D      ;CR?
3083 00:FD05 D0 03              BNE RDOB1
3084
3085 00:FD07 4C 18 F2   RDCR    JMP START     ;start resets stack to last BRK position
3086
3087 00:FD0A C9 20      RDOB1   CMP #' '      ;CHK FOR SPACE
3088 00:FD0C F0 04              BEQ RDOB1A
3089 00:FD0E C9 2C              CMP #','      ;CHK FOR COMMA
3090 00:FD10 D0 07              BNE RDOB2
3091 00:FD12 20 09 FC   RDOB1A  JSR SPAC      ;keep command line spacing constant
3092 00:FD15 A2 30              LDX #'0'      ;FILL WITH ZEREOS
3093                           ;BOTH Acc & X REG NEED DATA IE '00'
3094 00:FD17 80 12              BRA RDOB2A    ;PROCESS ALL ZEROS
3095
3096
3097 00:FD19 AA          RDOB2  TAX           ;SAVE 1ST CHAR IN X REG
```

```
3098  00:FD1A  20 71 FC        JSR GETCH     ;READ NEXT CHAR
3099  00:FD1D  B0 18           BCS RDOB4     ;^C OR ^X
```

'W65C134 Internal ROM Monitor ($F000)'
'MON5.ASM - Serial Routines'

```
3100  00:FD1F  C9 0D          CMP #$0D      ;CR?
3101  00:FD21  F0 E4          BEQ RDCR
3102  00:FD23  C9 2C          CMP #','      ;CK FOR COMMA DELIMITER
3103  00:FD25  F0 04          BEQ RDOB2A
3104  00:FD27  C9 20          CMP #' '      ;SPACE DELIMETER
3105  00:FD29  D0 08          BNE RDOB3
3106  00:FD2B  A9 30      RDOB2A   LDA #'0'
3107  00:FD2D  20 39 FD       JSR ASCBIN    ;PROCESS ALL ZEROS
3108  00:FD30  18             CLC
3109  00:FD31  80 04          BRA RDOB4
3110
3111  00:FD33  20 39 FD   RDOB3    JSR ASCBIN
3112  00:FD36  38             SEC           ;CY=1
3113  00:FD37  FA         RDOB4    PLX          ;RESTORE X
3114  00:FD38  60             RTS
3115
3116
3117                ;* Routine:  ASCBIN  ASCII TO BINARY
3118                ;*
3119                ;* Reg Used: ACC,X, and Y
3120                ;* Var Used: TMP6
3121                ;* Routines Called: HEXIN
3122                ;* Returned Reg: Acc & X
3123                ;*
3124  00:FD39  20 FB FD   ASCBIN   JSR HEXIN     ;ACC & X REG HAVE DATA
3125                          ;1ST CHAR IN X REG
3126  00:FD3C  08               PHP
3127  00:FD3D  5A               PHY
3128  00:FD3E  78               SEI           ;DISABLE INTR SO WE CAN
3129  00:FD3F  A4 83            LDY TMP6      ;save TMP6 in Y
3130  00:FD41  85 83            STA TMP6      ;USE VAR TMP6 AS TMP STORAGE
3131  00:FD43  8A               TXA
3132  00:FD44  20 FB FD         JSR HEXIN
3133  00:FD47  0A               ASL A
3134  00:FD48  0A               ASL A
3135  00:FD49  0A               ASL A
3136  00:FD4A  0A               ASL A
3137  00:FD4B  05 83            ORA TMP6
3138  00:FD4D  84 83            STY TMP6      ;restore TMP6
3139  00:FD4F  7A               PLY
3140  00:FD50  28               PLP
3141  00:FD51  60               RTS
3142
3143
3144                    .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON5.ASM - Serial Routines'

```
3145
3146                    ;* Routine:  ACI_INIT
3147                    ;*
3148                    ;* Reg Used: ACC,Y,X
3149                    ;* Var Used: SOUTINDX,SOUTEND,SININDX,SINEND,ERRORS,SFLAG,TMP6,
3150                    ;* Routines Called: FLUSH_SERIAL_BUFF
3151                    ;* Returned Reg: NONE
3152                    ;*
3153                    ;     Baud Rates         | For even parity, Y = 03
3154                    ;                        | For Odd parity,  Y = 01
3155                    ;0    75 BAUD            | No parity, Y = 00 or 02
3156                    ;1   110 BAUD            |
3157                    ;2   150 BAUD            |
3158                    ;3   300 BAUD            | Data length in X is either
3159                    ;4   600 BAUD            | 8 for 8 bits, or anything
3160                    ;5  1200 BAUD            | else for 7 bits.
3161                    ;6  1800 BAUD            |
3162                    ;7  2400 BAUD            |
3163                    ;8  4800 BAUD            |
3164                    ;9  9600 BAUD            |
3165                    ;A 19200 BAUD            |
3166                    ;B 38400 BAUD            |
3167
3168
3169
3170         00:FD52              ACI_INIT EQU *         ;A=BAUD,X=DATA LENGTH,
3171                                  ;SETUP POINTERS
3172  00:FD52  08                     PHP          ;SAVE INTERUPT STATUS
3173  00:FD53  78                     SEI          ;DISABLE ANY IRQ'S
3174  00:FD54  20 B4 FD               JSR FLUSH_SERIAL_BUFF
3175  00:FD57  64 72                  STZ SFLAG
3176  00:FD59  64 85                  STZ ERRORS    ;CLR SERIAL ERROR COUNT
3177  00:FD5B  5A                     PHY          ;Y=PARITY
3178  00:FD5C  DA                     PHX          ;DATA LENGTH
3179  00:FD5D  C9 0C                  CMP #$0C      ;IS ACC VALID 75-38400
3180  00:FD5F  B0 5C                  BCS ACI_ERR
3181  00:FD61  0A                     ASL A         ;X2
3182  00:FD62  85 83                  STA TMP6
3183  00:FD64  A4 86                  LDY SPEED     ;MULT BY 11 FOR MAIN XTAL
3184  00:FD66  B9 F3 FE               LDA BAUDOFFSET,Y
3185  00:FD69  18                     CLC
3186  00:FD6A  65 83                  ADC TMP6
3187  00:FD6C  AA                     TAX
3188  00:FD6D  BD F9 FE               LDA ACIBAUD,X  ;SETUP COUNTERS
3189  00:FD70  85 24                  STA TALL
3190  00:FD72  BD FA FE               LDA ACIBAUD+1,X
3191  00:FD75  85 25                  STA TALH
3192  00:FD77  A9 20                  LDA #$20      ;rev 2  Enable Receive - 7 Bit
3193  00:FD79  FA                     PLX
3194  00:FD7A  E0 08                  CPX #$08      ;8 BIT?
3195  00:FD7C  D0 02                  BNE ACI_I1
3196  00:FD7E  09 04                  ORA #$04      ;SWITCH TO 8 BIT
3197  00:FD80  85 83              ACI_I1   STA TMP6
3198  00:FD82  68                     PLA          ;GET PARITY WAS IN Y REG
3199  00:FD83  2A                     ROL A
```

```
3200  00:FD84  2A          ROL A
3201  00:FD85  2A          ROL A
```

'W65C134 Internal ROM Monitor ($F000)'
'MON5.ASM - Serial Routines'

```
3202  00:FD86  05 83          ORA TMP6
3203  00:FD88  85 22          STA ACSR
3204  00:FD8A  20 91 FD       JSR SIOPORTS
3205  00:FD8D  28             PLP
3206  00:FD8E  A9 00          LDA #$00
3207  00:FD90  60             RTS
3208
3209
3210
3211  00:FD91  A9 06       SIOPORTS LDA #$06      ;SET P62 RXD AS OUTPUT
3212  00:FD93  04 21          TSB PDD6     ;AND TXD
3213  00:FD95  A9 01          LDA #$01
3214  00:FD97  14 21          TRB PDD6     ;SET RXD AS INPUT
3215  00:FD99  A9 02          LDA #$02     ;SET TXD TO DEFAULT MARK
3216  00:FD9B  04 20          TSB PD6      ;TELL OTHER GUY TO XMIT
3217  00:FD9D  A9 04          LDA #$04     ;SERIAL DATA TO ME
3218  00:FD9F  14 20          TRB PD6
3219  00:FDA1  A9 80          LDA #$80     ;P47 AS INPUT
3220  00:FDA3  14 1E          TRB PDD4     ;SETUP PORTS
3221  00:FDA5  A9 08          LDA #$08     ;ENABLE NE47
3222  00:FDA7  04 2D          TSB IER1
3223  00:FDA9  A9 02          LDA #$02     ;SET PORT 44-47
3224  00:FDAB  04 1B          TSB BCR      ;ENABLED
3225  00:FDAD  A9 0E          LDA #$0E     ;SETUP SERIAL IRQS
3226  00:FDAF  85 0A          STA TCR1
3227  00:FDB1  64 23          STZ ARTD     ;SET DATA TO NULL
3228  00:FDB3  60             RTS
3229
3230
3231       00:FDB4       FLUSH_SERIAL_BUFF EQU * ;ARTD ERROR
3232  00:FDB4  64 73          STZ SOUTINDX   ;SETUP QUEUE COUNTERS TO ZERO
3233  00:FDB6  64 74          STZ SOUTEND    ;IE FLUSH ALL SERIAL QUEUES
3234  00:FDB8  64 75          STZ SININDX
3235  00:FDBA  64 76          STZ SINEND
3236  00:FDBC  60             RTS
3237
3238
3239
3240       00:FDBD       ACI_ERR  EQU *
3241  00:FDBD  FA             PLX
3242  00:FDBE  7A             PLY
3243  00:FDBF  A9 04          LDA #DTR     ;TELL OTHER GUY TO XMIT
3244  00:FDC1  14 20          TRB PD6      ;SERIAL DATA TO ME
3245  00:FDC3  28             PLP
3246  00:FDC4  A9 FF          LDA #$FF
3247  00:FDC6  60             RTS
3248
3249
3250                    .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON5.ASM - Serial Routines'

```
3251
3252                    ;* Routine:  TGLXONMODE
3253                    ;*
3254                    ;* Reg Used: ACC,Y,X
3255                    ;* Var Used: TMPC,TMP0
3256                    ;* Routines Called: REGTTL,WRPC,SETR,SPAC,WROB
3257                    ;* Returned Reg: NONE
3258                    ;*
3259
3260
3261        00:FDC7             TGLXONMODE EQU *       ;TOGGLE SERIAL XON/XOFF MODE
3262  00:FDC7  78                       SEI
3263  00:FDC8  A5 72                    LDA SFLAG
3264  00:FDCA  49 04                    EOR #XONOFLG
3265  00:FDCC  89 04                    BIT #XONOFLG
3266  00:FDCE  F0 0C                    BEQ TGLXONA
3267  00:FDD0  29 B7                    AND #$FF-SXOFFLG-SNDOVF
3268  00:FDD2  09 90                    ORA #SXONFLG+LASTXONOF
3269  00:FDD4  85 72                    STA SFLAG
3270  00:FDD6  A9 08                    LDA #$08      ;DISABLE NE47
3271  00:FDD8  14 2D                    TRB IER1
3272  00:FDDA  80 08                    BRA TGLXONB
3273
3274  00:FDDC  29 27             TGLXONA  AND #$FF-SXONFLG-SXOFFLG-SNDOVF-LASTXONOF
3275  00:FDDE  85 72                    STA SFLAG
3276  00:FDE0  A9 08                    LDA #$08      ;ENABLE NE47
3277  00:FDE2  04 2D                    TSB IER1
3278  00:FDE4  A9 04             TGLXONB  LDA #$04      ;DTR LOW
3279  00:FDE6  14 20                    TRB PD6
3280  00:FDE8  58                       CLI
3281  00:FDE9  60                       RTS
3282
3283  00:FDEA                           .END
3284
3285  00:FDEA                           INCLUDE MON6.ASM
3286                          .STTL 'MON6.ASM - Library Subroutines'
3287                          .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON6.ASM - Library Subroutines'

```
3288                        ;06-21-1993
3289
3290        00:007B             DVDN    EQU TMP0
3291        00:007E             DVSR    EQU TMP2
3292        00:0081             RMNDR   EQU TMP4
3293                ;
3294                ;MOVE A BLOCK OF DATA'
3295                ;
3296                ;08-21-1988
3297
3298
3299
3300                ;SRCE--ADDRESS OF SOURCE BLOCK
3301                ;DEST--ADDRESS OF DESTINATION BLOCK
3302                ;Y--# OF BYTES TO BE MOVED
3303                ; Y=0 IS TO MOVE 256 BYTES
3304                ;WARNING: TWO MEMORY AREAS MUST NOT OVER LAP
3305                ; CY = 1 IF MEMORY NOT THERE
3306
3307 00:FDEA  88             MVDATA  DEY         ;DEC INDEX
3308 00:FDEB  B1 7B                LDA (SRCE),Y  ;GET SOURCE BYTE
3309 00:FDED  91 7E                STA (DEST),Y  ;STORE DATA
3310 00:FDEF  D1 7E                CMP (DEST),Y  ;CK IF WRITTEN
3311 00:FDF1  D0 06                BNE MVBAD
3312 00:FDF3  C0 00                CPY #$00     ;ARE WE DONE
3313 00:FDF5  D0 F3                BNE MVDATA   ;NOT DONE
3314 00:FDF7  18                   CLC          ;GOOD MOVE
3315 00:FDF8  60                   RTS          ;RETURN
3316
3317 00:FDF9  38             MVBAD   SEC
3318 00:FDFA  60                   RTS
3319
3320
3321
3322                ;
3323                ;   CONVERTS ASCII HEX TO HEX
3324                ;
3325
3326                ;01-19-1989
3327
3328
3329                ;* Routine:  HEXIN
3330                ;*   IF CARRY SET THEN NOT ASCII HEX
3331                ;*   ACC IN AND OUTPUT
3332                ;* Reg Used: Acc
3333                ;* Var Used: NONE
3334                ;* Routines Called: ISHEX
3335                ;* Returned Reg: Acc
3336                ;
3337
3338
3339 00:FDFB  20 0B FE       HEXIN   JSR ISHEX    ;IS IT HEX
3340 00:FDFE  B0 0A                BCS HEXXX
3341 00:FE00  C9 3A                CMP #$3A
3342 00:FE02  08                   PHP          ;SAVE STATUS
```

```
3343  00:FE03  29 0F          AND #$0F      ;STRIP OF LS NIBBLE
3344  00:FE05  28             PLP           ;GET STAT
```

'W65C134 Internal ROM Monitor ($F000)'
'MON6.ASM - Library Subroutines'

```
3345  00:FE06  90 02              BCC HEXXX     ;WAS NUMBER
3346  00:FE08  69 08              ADC #$08      ;WAS ALPHA ADD 8+CY=9
3347  00:FE0A  60           HEXXX  RTS
3348
3349
3350              ;* Routine:  ISHEX
3351              ;*   TESTS FOR VALID ASCII HEX DIGIT
3352              ;*   C=SET IF NOT HEX
3353              ;* Reg Used: Acc
3354              ;* Var Used: NONE
3355              ;* Routines Called: UPPER_CASE
3356              ;* Returned Reg: Acc
3357
3358  00:FE0B  20 25 FE     ISHEX  JSR UPPER_CASE ;IF NOT MAKE UPPER CASE
3359  00:FE0E  C9 41              CMP #'A'      ;LESS THAN 'A'
3360  00:FE10  90 03              BCC ISDECIMAL  ;YES,TRY NUMBER CHECK
3361  00:FE12  C9 47              CMP #'G'      ;F+1
3362                             ;IF CY SET THEN GREATER THAN F
3363  00:FE14  60                 RTS           ;IF CY CLR THEN OK
3364
3365
3366              ;* Routine:  ISDECIMAL
3367              ;*   CHECK FOR VALID ASCII #
3368              ;* Reg Used: Acc
3369              ;* Var Used: NONE
3370              ;* Routines Called: NONE
3371              ;* Returned Reg: Acc
3372
3373  00:FE15  C9 30        ISDECIMAL CMP #'0'      ;IS LESS THAN '0'
3374  00:FE17  90 03              BCC ISN1      ;YES,NG
3375  00:FE19  C9 3A              CMP #'9'+1    ;IE >9
3376                             ;IF CY SET THEN NG
3377  00:FE1B  60                 RTS           ;IF CY CLR THEN OK
3378
3379  00:FE1C  38           ISN1  SEC           ;BAD GUYS EXIT
3380  00:FE1D  60                 RTS
3381
3382
3383              ;* Routine:  IFASC
3384              ;*   CHECK FOR VALID ASCII
3385              ;* Reg Used: Acc
3386              ;* Var Used: NONE
3387              ;* Routines Called: ISHEX
3388              ;* Returned Reg: Acc
3389
3390  00:FE1E  C9 20        IFASC  CMP #' '      ;IS LESS THAN SPACE
3391  00:FE20  90 FA              BCC ISN1      ;YES SO NOT ASCII
3392  00:FE22  C9 7F              CMP #$7F      ;GT TILDA
3393                             ;IF CY SET THEN SO NOT ASCII
3394  00:FE24  60                 RTS           ;IF CY CLR THEN OK
3395
3396
3397
3398              ;* Routine:  UPPER_CASE
3399              ;* Reg Used: Acc
```

```
3400            ;* Var Used: NONE
3401            ;* Routines Called: NONE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON6.ASM - Library Subroutines'

```
3402                    ;* Returned Reg: Acc
3403
3404
3405  00:FE25  C9 61            UPPER_CASE CMP #'a'    ;CONVERT TO UPPER CASE
3406  00:FE27  90 07                BCC NIBBIN1    ;NOT an upper case char
3407  00:FE29  C9 7B                CMP #'z'+1     ;IS IT GT A 'z'
3408  00:FE2B  B0 03                BCS NIBBIN1    ;NOT an upper case char
3409  00:FE2D  38                   SEC
3410  00:FE2E  E9 20                SBC #$20       ;MAKE IT UPPER CASE
3411  00:FE30  60               NIBBIN1  RTS
3412
3413
3414
3415                    ;* Routine:  BIN2DEC
3416                    ;* Reg Used: Acc
3417                    ;* Var Used: NONE
3418                    ;* Routines Called: NONE
3419                    ;* Returned Reg: Acc
3420
3421
3422  00:FE31  DA               BIN2DEC  PHX           ;convert Acc to packed decimal (MAX 99)
3423  00:FE32  48                   PHA
3424  00:FE33  4A                   LSR A
3425  00:FE34  4A                   LSR A
3426  00:FE35  4A                   LSR A
3427  00:FE36  4A                   LSR A
3428  00:FE37  AA                   TAX
3429  00:FE38  F8                   SED
3430  00:FE39  68                   PLA
3431  00:FE3A  29 0F                AND #$0F
3432  00:FE3C  48                   PHA
3433  00:FE3D  BD C1 FF             LDA BINDECH,X
3434  00:FE40  18                   CLC
3435  00:FE41  FA                   PLX
3436  00:FE42  7D B1 FF             ADC BINDECL,X
3437  00:FE45  D8                   CLD
3438  00:FE46  FA                   PLX
3439  00:FE47  60                   RTS
3440
3441  00:FE48                       INCLUDE MON7.ASM
3442                        .STTL 'MON7.ASM - Special interrupts - Power Down'
3443                        .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON7.ASM - Special interrupts - Power Down'

```
3444                        ;06-23-1993
3445
3446
3447        00:FE48             NE46 EQU *          ;POWER FAIL PENDING
3448  00:FE48  6C 4C 00            JMP (UNE46)
3449                    ;
3450                    ;
3451                    ; POWER DOWN:
3452                    ;
3453                    ; When we get a power down interrupt, we come here.  We shut down all
3454                    ; interrupts (except TOD clock), and clear any that were pending.
3455                    ; we reset the stack to FF, we write a semiphore into RAM to indicate that
3456                    ; we are powered down, and we run the power down routine. We slow down then
3457                    ; shut off the fast clock.  We turn all i/o ports to inputs.
3458                    ;
3459                    ; The power down routine executes the time of day interrupt, then checks RAM
3460                    ; for a semiphore indicating that there is a low power routine in RAM.
3461                    ; If there is we JSR to $0088.  The RAM routine can do anything it likes and
3462                    ; returns with a RTS.
3463                    ;
3464                    ; After the RTS (or after deciding there is no RAM routine) we check for
3465                    ; power back up.  If power has returned to the system, we bring up a few
3466                    ; key registers then jump to RESET.
3467                    ;
3468                    ; If a physical reset occurs while we are powered down, we must return to
3469                    ; the power down code w/out restarting anything on the bus.  We use the first
3470                    ; semiphore above to flag this condition.
3471  00:FE4B
3472                    ; IMPORTANT SEMIPHORES:
3473                    ;
3474                    ; If we are in low power mode, locations $7B = $55, $7C = $AA, $7D = $88.
3475                    ; This is checked by RESET.
3476                    ;
3477                    ; If there is a valid routine in RAM, it starts at $0088, and locations
3478                    ; $7E = $55, $7F = $AA. If that is the case, the low power
3479                    ; routine will jsr to $0088 once per second.
3480                    ;
3481                    ; RAM LOCATIONS:
3482                    ;
3483                    ; RAM locations F8-FF are used for the stack to do the TOD interrupt.
3484                    ;
3485                    ; $40 - $87  remain through power down; don't mess with them.
3486                    ;
3487
3488
3489        00:FE4B             PDOWN    EQU *        ;GOTO LOW POWER MODE
3490
3491  00:FE4B  A9 55                LDA #$55      ;set the LP semiphore
3492  00:FE4D  85 7B                STA $7B
3493  00:FE4F  A9 AA                LDA #$AA
3494  00:FE51  85 7C                STA $7C
3495  00:FE53  A9 88                LDA #$88
3496  00:FE55  85 7D                STA $7D
3497
3498  00:FE57  A9 7A                LDA #%01111010 ;DISABLE EDGE IRQ'S
```

```
3499  00:FE59  14 1B                TRB BCR       ;DISABLE NMI & IRQ'S
3500
```

'W65C134 Internal ROM Monitor ($F000)'
'MON7.ASM - Special interrupts - Power Down'

```
3501  00:FE5B  64 2D                STZ IER1      ;DISABLE INTERRUPTS
3502  00:FE5D  A9 FF                LDA #$FF
3503  00:FE5F  85 2C                STA IFR1      ;RESET ANY INTERRUPTS
3504
3505  00:FE61  A9 20                LDA #$20      ;ENABLE ONLY T2
3506  00:FE63  85 09                STA IER2
3507
3508  00:FE65  A9 DF                LDA #$DF
3509  00:FE67  04 08                TSB IFR2      ;RESET ANY EXCEPT T2
3510
3511  00:FE69  A9 F8                LDA #<TODIRQ   ;MAKE TOD CLK INTERNAL
3512  00:FE6B  85 48                STA UIRQT2
3513  00:FE6D  A9 F8                LDA #>TODIRQ
3514  00:FE6F  85 49                STA UIRQT2+1
3515
3516  00:FE71  64 22                STZ ACSR      ;DISABLE REC & XMIT SERIAL
3517
3518  00:FE73  A5 23                LDA ARTD      ;RESET ANY SERIAL READS
3519  00:FE75  64 23                STZ ARTD      ;RESET ANY SERIAL WRITES
3520
3521  00:FE77  64 1C                STZ PD4       ;PUT ALL PORTS LOW
3522  00:FE79  64 1D                STZ PD5
3523  00:FE7B  64 20                STZ PD6
3524  00:FE7D  64 1E                STZ PDD4      ;RESET PORTS TO INPUTS
3525  00:FE7F  64 1F                STZ PDD5
3526  00:FE81  64 21                STZ PDD6
3527
3528  00:FE83  A9 18                LDA #$18      ;T1 DISABLE, LEAVE T2 (TOD)
3529  00:FE85  85 0B                STA TCR2
3530
3531  00:FE87  A9 04                LDA #%00000100 ;shift to slow clock
3532  00:FE89  85 0A                STA TCR1
3533
3534  00:FE8B  A9 00                LDA #$00      ;WAIT UNTIL SLOW CLOCK
3535  00:FE8D  3A             PDLP1    DEC A        ;CAN TAKE EFFECT
3536  00:FE8E  D0 FD                BNE PDLP1
3537
3538  00:FE90  64 0A                STZ TCR1      ;fast clock is now OFF
3539
3540        [01]                    .IFDEF IROM
3541  00:FE92  64 1B                STZ BCR       ;GO TO ALL INTERNAL
3542        [01]                    .ELSE
3543                      LDA #$81
3544                      STA BCR       ;for test
3545        [00]                    .ENDIF
3546
3547  00:FE94  64 07                STZ PCS3      ;kill CS outputs
3548        [01]                    .IFDEF IROM
3549  00:FE96  64 03                STZ PD3       ;internal ROM, set all output to 0
3550        [01]                    .ELSE
3551                      LDA #$FF
3552                      STA PD3       ;external ROM, set all high CS outs
3553        [00]                    .ENDIF
3554
3555  00:FE98  A2 FF                LDX #$FF      ;RESET STACK TO INTERNAL
```

```
3556  00:FE9A  9A                    TXS
3557
```

'W65C134 Internal ROM Monitor ($F000)'
'MON7.ASM - Special interrupts - Power Down'

```
3558  00:FE9B  64 30          STZ PD0       ;PUT ALL PORTS LOW
3559  00:FE9D  64 31          STZ PD1
3560  00:FE9F  64 32          STZ PD2
3561  00:FEA1  64 34          STZ PDD0      ;RESET PORTS TO INPUTS
3562  00:FEA3  64 35          STZ PDD1
3563  00:FEA5  64 36          STZ PDD2
3564
3565  00:FEA7  A9 20     NE46LP1  LDA #PUFLG     ;set flag to indicate
3566  00:FEA9  14 77          TRB DISPTYP    ;we haven't seen any
3567                          ;power up yet.
3568
3569  00:FEAB  58             CLI
3570
3571  00:FEAC  DB        NE46LP   .BYTE $DB     ;STOP until Interrupt
3572  00:FEAD  EA             NOP
3573                  ;
3574                  ; We get here after doing a TOD Interrupt
3575                  ;
3576                  ;
3577
3578  00:FEAE  A5 7E          LDA $7E
3579  00:FEB0  C9 55          CMP #$55
3580  00:FEB2  D0 09          BNE NORAMRT
3581  00:FEB4  A5 7F          LDA $7F
3582  00:FEB6  C9 AA          CMP #$AA
3583  00:FEB8  D0 03          BNE NORAMRT
3584  00:FEBA  20 88 00       JSR $0088     ;JSR to RAM routine
3585
3586  00:FEBD  A5 1C     NORAMRT  LDA PD4       ;CK PD46 FOR POWER UP
3587  00:FEBF  89 40          BIT #$40
3588  00:FEC1  F0 E4          BEQ NE46LP1    ;reset PUP flag if Necss.
3589
3590  00:FEC3  A9 20          LDA #PUFLG     ;SET 1ST TIME FLG
3591  00:FEC5  04 77          TSB DISPTYP    ;NOTICED PU FLG
3592  00:FEC7  F0 E3          BEQ NE46LP     ;a debounce of 2
3593  00:FEC9
3594                  ;
3595                  ; POWER BACK UP - Restart system
3596                  ;
3597                  ;
3598
3599  00:FEC9  64 7B          STZ $7B       ;CLEAR ALL SEMIPHORES
3600  00:FECB  64 7C          STZ $7C
3601  00:FECD  64 7D          STZ $7D
3602  00:FECF  64 7E          STZ $7E
3603  00:FED1  64 7F          STZ $7F
3604
3605  00:FED3  6C FC FF       JMP ($FFFC)
3606
3607
3608
3609       00:FED6       PE44    EQU *
3610       00:FED6       PE45    EQU *
3611       00:FED6       PE50    EQU *
3612       00:FED6       PE51    EQU *
```

```
3613        00:FED6             NE52    EQU *
3614        00:FED6             NE53    EQU *
```

'W65C134 Internal ROM Monitor ($F000)'
'MON7.ASM - Special interrupts - Power Down'

```
3615         00:FED6              IRQRESERVED EQU *
3616         00:FED6              PE54    EQU *
3617         00:FED6              PE55    EQU *
3618         00:FED6              PE56    EQU *
3619         00:FED6              NE57    EQU *
3620         00:FED6              GENIRQ  EQU *
3621  00:FED6  6C 4E 00                JMP (UGENIRQ)
3622
3623
3624
3625  00:FED9  6C 4A 00       IRQT1   JMP (UIRQT1)
3626
3627
3628  00:FEDC  6C 48 00       IRQT2   JMP (UIRQT2)
3629
3630
3631  00:FEDF  6C 46 00       IRQ1    JMP (UIRQ1)
3632
3633
3634  00:FEE2  6C 44 00       IRQ2    JMP (UIRQ2)
3635
3636
3637  00:FEE5  6C 42 00       NMIRQ   JMP (UNMI)
3638
3639
3640
3641  00:FEE8  6C 40 00       IRQBRK  JMP (UBRK)
3642
3643
3644
3645         00:FEF3              TABLE_START EQU $FEF3
3646
3647
3648         00:0008              ZZZSPACE EQU TABLE_START-*    ;gives space left in the ROM
3649
3650         00:FEEC              LASTBYTE EQU *+1
3651
3652                         .STTL 'MON7.ASM - GENERAL LOOKUP TABLES'
3653                         .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON7.ASM - GENERAL LOOKUP TABLES'

```
3654
3655  00:FEF3                   .ORG TABLE_START
3656
3657        00:FEF3           BAUDOFFSET EQU *
3658  00:FEF3  00               .BYTE 00      ;2.0000MHZ
3659  00:FEF4  18               .BYTE 24      ;4.0000MHZ
3660  00:FEF5  30               .BYTE 48      ;2.4576MHZ
3661  00:FEF6  48               .BYTE 72      ;3.6864MHZ
3662  00:FEF7  60               .BYTE 96      ;1.8432MHZ
3663  00:FEF8  78               .BYTE 120     ;4.9125 MHZ
3664
3665
3666
3667
3668        00:FEF9           ACIBAUD  EQU *
3669                            ;2MHZ
3670  00:FEF9  8206             .WORD $0682   ;   75 BAUD
3671  00:FEFB  6F04             .WORD $046F   ;  110 BAUD
3672  00:FEFD  4003             .WORD $0340   ;  150 BAUD
3673  00:FEFF  A001             .WORD $01A0   ;  300 BAUD
3674  00:FF01  CF00             .WORD $00CF   ;  600 BAUD
3675  00:FF03  6700             .WORD $0067   ; 1200 BAUD
3676  00:FF05  4400             .WORD $0044   ; 1800 BAUD
3677  00:FF07  3300             .WORD $0033   ; 2400 BAUD
3678  00:FF09  1900             .WORD $0019   ; 4800 BAUD
3679  00:FF0B  0C00             .WORD $000C   ; 9600 BAUD DOES NOT WORK RELIABLY
3680  00:FF0D  0600             .WORD $0006   ;19200 BAUD BAD WONT WORK AT 2MHZ
3681  00:FF0F  0200             .WORD $0002   ;38400 BAUD BAD WONT WORK AT 2MHZ
3682
3683                            ;4MHZ
3684  00:FF11  040D             .WORD $0D04   ;   75 BAUD
3685  00:FF13  E008             .WORD $08E0   ;  110 BAUD
3686  00:FF15  8206             .WORD $0682   ;  150 BAUD
3687  00:FF17  4003             .WORD $0340   ;  300 BAUD
3688  00:FF19  A001             .WORD $01A0   ;  600 BAUD
3689  00:FF1B  CF00             .WORD $00CF   ; 1200 BAUD
3690  00:FF1D  8A00             .WORD $008A   ; 1800 BAUD
3691  00:FF1F  6700             .WORD $0067   ; 2400 BAUD
3692  00:FF21  3300             .WORD $0033   ; 4800 BAUD
3693  00:FF23  1900             .WORD $0019   ; 9600 BAUD
3694  00:FF25  0C00             .WORD $000C   ;19200 BAUD
3695  00:FF27  0600             .WORD $0006   ;38400 BAUD BAD WONT WORK AT 2MHZ
3696
3697                            ;2.4576MHZ
3698  00:FF29  FF07             .WORD $07FF   ;   75 BAUD
3699  00:FF2B  7305             .WORD $0573   ;  110 BAUD
3700  00:FF2D  FF03             .WORD $03FF   ;  150 BAUD
3701  00:FF2F  FF01             .WORD $01FF   ;  300 BAUD
3702  00:FF31  FF00             .WORD $00FF   ;  600 BAUD
3703  00:FF33  7F00             .WORD $007F   ; 1200 BAUD
3704  00:FF35  5400             .WORD $0054   ; 1800 BAUD
3705  00:FF37  3F00             .WORD $003F   ; 2400 BAUD
3706  00:FF39  1F00             .WORD $001F   ; 4800 BAUD
3707  00:FF3B  0F00             .WORD $000F   ; 9600 BAUD
3708  00:FF3D  0700             .WORD $0007   ;19200 BAUD
```

```
3709  00:FF3F  0300                 .WORD $0003    ;38400 BAUD
3710
```

'W65C134 Internal ROM Monitor ($F000)'
'MON7.ASM - GENERAL LOOKUP TABLES'

```
3711                                    ;3.6864MHZ
3712  00:FF41  FF0B                      .WORD $0BFF    ;   75 BAUD
3713  00:FF43  2E08                      .WORD $082E    ;  110 BAUD
3714  00:FF45  FF05                      .WORD $05FF    ;  150 BAUD
3715  00:FF47  FF02                      .WORD $02FF    ;  300 BAUD
3716  00:FF49  7F01                      .WORD $017F    ;  600 BAUD
3717  00:FF4B  BF00                      .WORD $00BF    ; 1200 BAUD
3718  00:FF4D  7F00                      .WORD $007F    ; 1800 BAUD
3719  00:FF4F  5F00                      .WORD $005F    ; 2400 BAUD
3720  00:FF51  2F00                      .WORD $002F    ; 4800 BAUD
3721  00:FF53  1700                      .WORD $0017    ; 9600 BAUD
3722  00:FF55  0B00                      .WORD $000B    ;19200 BAUD
3723  00:FF57  0500                      .WORD $0005    ;38400 BAUD
3724
3725                                    ;1.8432MHZ
3726  00:FF59  FF05                      .WORD $05FF    ;   75 BAUD
3727  00:FF5B  1604                      .WORD $0416    ;  110 BAUD
3728  00:FF5D  FF02                      .WORD $02FF    ;  150 BAUD
3729  00:FF5F  7F01                      .WORD $017F    ;  300 BAUD
3730  00:FF61  BF00                      .WORD $00BF    ;  600 BAUD
3731  00:FF63  5F00                      .WORD $005F    ; 1200 BAUD
3732  00:FF65  3F00                      .WORD $003F    ; 1800 BAUD
3733  00:FF67  2F00                      .WORD $002F    ; 2400 BAUD
3734  00:FF69  1700                      .WORD $0017    ; 4800 BAUD
3735  00:FF6B  0B00                      .WORD $000B    ; 9600 BAUD
3736  00:FF6D  0500                      .WORD $0005    ;19200 BAUD
3737  00:FF6F  0200                      .WORD $0002    ;38400 BAUD
3738
3739
3740                                    ;4.9152MHZ
3741  00:FF71  FF0F                      .WORD $0FFF    ;   75 BAUD
3742  00:FF73  E80A                      .WORD $0AE8    ;  110 BAUD
3743  00:FF75  FF07                      .WORD $07FF    ;  150 BAUD
3744  00:FF77  FF03                      .WORD $03FF    ;  300 BAUD
3745  00:FF79  FF01                      .WORD $01FF    ;  600 BAUD
3746  00:FF7B  FF00                      .WORD $00FF    ; 1200 BAUD
3747  00:FF7D  AA00                      .WORD $00AA    ; 1800 BAUD
3748  00:FF7F  7F00                      .WORD $007F    ; 2400 BAUD
3749  00:FF81  3F00                      .WORD $003F    ; 4800 BAUD
3750  00:FF83  1F00                      .WORD $001F    ; 9600 BAUD
3751  00:FF85  0F00                      .WORD $000F    ;19200 BAUD
3752  00:FF87  0700                      .WORD $0007    ;38400 BAUD
3753
3754
3755
3756                    ;
3757                    ;
3758                    ;TIME OF DAY MAX MIN TABLES
3759                    ;
3760                    ;
3761          00:FF89               MAXTTBL  EQU *
3762  00:FF89  3C                       .BYTE 60      ;MIN
3763  00:FF8A  18                       .BYTE 24      ;HR
3764  00:FF8B  20                       .BYTE 32      ;DAY
3765  00:FF8C  0D                       .BYTE 13      ;MONTH
```

```
3766  00:FF8D  64                   .BYTE 100     ;YR
3767  00:FF8E  08                   .BYTE 8       ;DAY OF WEEK
```

'W65C134 Internal ROM Monitor ($F000)'
'MON7.ASM - GENERAL LOOKUP TABLES'

```
3768
3769        00:FF8F             MINTTBL  EQU *
3770  00:FF8F  00                       .BYTE 0       ;MIN
3771  00:FF90  00                       .BYTE 0       ;HR
3772  00:FF91  01                       .BYTE 1       ;DAY
3773  00:FF92  01                       .BYTE 1       ;MONTH
3774  00:FF93  00                       .BYTE 0       ;YR
3775  00:FF94  01                       .BYTE 1       ;DAY OF WEEK
3776
3777
3778  00:FF95  1F               LASTDY   .BYTE 31      ;JANUARY
3779  00:FF96  1C                       .BYTE 28      ;FEBRUARY-EXCEPT LEAP YR
3780  00:FF97  1F                       .BYTE 31      ;MARCH
3781  00:FF98  1E                       .BYTE 30      ;APRIL
3782  00:FF99  1F                       .BYTE 31      ;MAY
3783  00:FF9A  1E                       .BYTE 30      ;JUNE
3784  00:FF9B  1F                       .BYTE 31      ;JULY
3785  00:FF9C  1F                       .BYTE 31      ;AUGUST
3786  00:FF9D  1E                       .BYTE 30      ;SEPTEMBER
3787  00:FF9E  1F                       .BYTE 31      ;OCTOBER
3788  00:FF9F  1E                       .BYTE 30      ;NOVEMBER
3789  00:FFA0  1F                       .BYTE 31      ;DECEMBER
3790
3791  00:FFA1  00               DFLTS    .BYTE 0       ;SEC
3792  00:FFA2  00                       .BYTE 00      ;MINUTES
3793  00:FFA3  0C                       .BYTE 12      ;HOUR
3794  00:FFA4  04                       .BYTE 04      ;DAY
3795  00:FFA5  07                       .BYTE 07      ;MONTH
3796  00:FFA6  5D                       .BYTE 93      ;YEAR
3797  00:FFA7  01                       .BYTE 1       ;DAY OF WEEK
3798  00:FFA8  00                       .BYTE 0       ;DAYLIGHT SAVING OFF
3799        00:FFA9             DFLTSEND EQU *
3800
3801                    ;
3802                    ;  STTL 'CONVERSION TABLES
3803                    ;
3804        00:FFA9             HEXTOPOS EQU *
3805  00:FFA9  01 02 04 08              .BYTE $01,$02,$04,$08
3806  00:FFAD  10 20 40 80              .BYTE $10,$20,$40,$80
3807
3808
3809        00:FFB1             BINDECL  EQU *
3810  00:FFB1  00 01 02 03 04           .BYTE $00,$01,$02,$03,$04,$05,$06,$07,$08,$09
   00FFB6  05 06 07 08 09
3811  00:FFBB  10 11 12 13 14                 .BYTE $10,$11,$12,$13,$14,$15
   00FFC0  15
3812
3813  00:FFC1  00 16 32 48 64 BINDECH  .BYTE $00,$16,$32,$48,$64,$80,$96
   00FFC6  80 96
3814                         .PAGE
```

'W65C134 Internal ROM Monitor ($F000)'
'MON7.ASM - GENERAL LOOKUP TABLES'

```
3815
3816  00:FFC8  F8F8                .WORD TODIRQ   ;MONITOR TOD IRQ PTR
3817  00:FFCA  FCF1                .WORD MONBRK   ;MONITOR BREAK IRQ PTR
3818  00:FFCC  4BFE                .WORD PDOWN    ;POWER DOWN IRQ PTR
3819  00:FFCE  33F2                .WORD S0       ;JMP INTO MONITOR CMD PARSER
3820  00:FFD0  D6FE                .WORD PE44
3821
3822  00:FFD2  D6FE                .WORD PE45
3823  00:FFD4  48FE                .WORD NE46
3824  00:FFD6  B9FB                .WORD NE47
3825  00:FFD8  D6FE                .WORD PE50
3826  00:FFDA  D6FE                .WORD PE51
3827  00:FFDC  D6FE                .WORD NE52
3828  00:FFDE  D6FE                .WORD NE53
3829  00:FFE0  D6FE                .WORD IRQRESERVED
3830  00:FFE2  D6FE                .WORD IRQRESERVED
3831  00:FFE4  2EFB                .WORD IRQAT
3832  00:FFE6  91FA                .WORD IRQAR
3833  00:FFE8  D6FE                .WORD IRQRESERVED
3834  00:FFEA  D6FE                .WORD PE54
3835  00:FFEC  D6FE                .WORD PE55
3836  00:FFEE  D6FE                .WORD PE56
3837  00:FFF0  D6FE                .WORD NE57
3838
3839  00:FFF2  D9FE                .WORD IRQT1
3840  00:FFF4  DCFE                .WORD IRQT2
3841  00:FFF6  DFFE                .WORD IRQ1
3842  00:FFF8  E2FE                .WORD IRQ2
3843  00:FFFA  E5FE                .WORD NMIRQ
3844  00:FFFC  8FF0                .WORD RESET
3845  00:FFFE  E8FE                .WORD IRQBRK
3846
3847
3848
3849
3850         [01]                  .IFTRUE   LASTBYTE.UGT.BAUDOFFSET
3851                         .EXIT  "It won't fit in the ROM!!!!"
3852         [00]                  .ENDIF
3853
3854  00:0000                      .END
```

Defined Symbol Name Value References

1434 A2 = 00:F3CB 1432
1439 A5 = 00:F3D0 1442
1443 A6 00:F3D8
475 ACC 00:005B 1088 1469
3668 ACIBAUD = 00:FEF9 3188 3190
3240 ACI_ERR = 00:FDBD 3180
3197 ACI_I1 00:FD80 3195
3170 ACI_INIT = 00:FD52 599 1023
412 ACSR = 00:0022 2653 2655 2694 2700 2735 2738 2744 2748 2760 2762
2879 2888 2997 3001 3203 3516
508 ADAY 00:006E 2283
511 ADAYWK 00:0071 2284
1196 ADRS 00:F27E 1143 1149 1152
507 AHR 00:006D 2282
545 ALRMENAB = 00:0008 2278
546 ALRMIRQ = 00:0010 2298
1429 ALTER = 00:F3C0 1196
1293 ALTERM 00:F311 1197
506 AMIN 00:006C 2281
509 AMONTH 00:006F 2285
2327 APR1 00:F9E0 2323
2318 APRIL 00:F9D1 2249
424 ARTD = 00:0023 1055 2558 2723 3227 3518 3519
2925 ASC1 00:FC6E 2923
3124 ASCBIN 00:FD39 642 3107 3111
2920 ASCII 00:FC65 2915
505 ASEC 00:006B 2156 2280 2290 2470
510 AYR 00:0070 2286
3657 BAUDOFFSET = 00:FEF3 3184 3850
1914 BCCST2 00:F7FA 1867
387 BCR = 00:001B 580 825 3224 3499 3541
3422 BIN2DEC 00:FE31 697 2532
2910 BINASC 00:FC59 644 2533 2801
3813 BINDECH 00:FFC1 3433
3809 BINDECL = 00:FFB1 3436
1403 BY2 00:F3AD 1396 1782
1404 BY3 00:F3B0 1393 1575 1595 1632 1906
1391 BYTE = 00:F39B 1320 1441 1778
1233 CBAS1 00:F2B9 1230
1228 CBASIC 00:F2AF 1219
523 CFLG = 00:0002 2571 2862 2973 3029
2014 CHKSUM = 00:F846 1209
2024 CHK_SUM = 00:F855 711 2017
2296 CKAL1 00:F9B8 2292
2290 CKALARM 00:F9AE 2297
869 CKHIROM 00:F0D3 874
877 CKLOWRAM = 00:F0E3 871
1916 CKNOUT 00:F7FB 1888 1900 1902 1905
2029 CKS1 00:F85E 2033
2034 CKSX 00:F86B 2028
2020 CKS_RSTART = 00:F852 2056 2081
1029 CKTODLP 00:F186 1031
3028 CK_CONTC = 00:FCD8 623 1117 2982
2304 CLKSUM 00:F9C4 2306
1172 CMDS 00:F267 1143 1144

345 CNTRLC = 00:0003 2566
346 CNTRLX = 00:0018 2580

Defined	Symbol Name	Value	References
Pre	CODE	00:0000	570
2808	CRLF	00:FBF4	636 1119 1312 1547 1814 1846 1858 1869 2079 2507
1970	DADD	00:F82D	1403 1753 1768 1771 1774 1786 1916 2030
Pre	DATA	00:0000	
491	DAY	00:0066	2217 2243 2250 2261 2266 2321 2497
496	DAYLIT	00:006A	2209 2221 2223 2245 2348 2352
501	DAYLITFLG	= 00:0001	2208 2244 2347 2351
502	DAYLPROG	= 00:0080	2220
494	DAYWK	00:0069	2214 2236 2237 2241 2318 2490
1361	DCMP	= 00:F383	1553 1597 1624 1634 1878 1911 2027 2032
566	DEST	= 00:007E	3309 3310
1139	DFLTPRSR	00:F23D	1134
3791	DFLTS	00:FFA1	1026 2150 2151 2155 2159 2289 2301
3799	DFLTSEND	= 00:FFA9	1026 2150 2155 2159 2289 2301
552	DIFF	00:0079	1365 1369 1557 1625 1881
536	DISPTYP	00:0077	2277 2299 3566 3591
935	DLY0	00:F120	936
1340	DSPLYDEC	= 00:F35A	1198
1335	DSPLYINC	= 00:F355	1199
1347	DSPLYOLD	00:F364	1200 1337 1345
1262	DSPLYR	00:F2D6	1204
2079	DTIME	00:F897	1213
406	DTR	= 00:0004	2640 3005 3243
1604	DUMPEND	= 00:F4BB	1528
1602	DUMPSTR	= 00:F4B3	1526 1527 1528
3290	DVDN	= 00:007B	
3291	DVSR	= 00:007E	
527	ECHOFF	= 00:0020	1115 1740 1742 1800 2096 2098 2124 2944
1953	ENDOK	00:F826	1946
1157	ERROPR	00:F25B	1639 2058
557	ERRORS	00:0085	1043 1737 1780 1796 1808 1820 3176
2300	EXITA	= 00:F9BF	2279 2287 2294
2232	EXITA6	00:F94D	2230
2225	EXITOCT	00:F945	2196 2201 2210 2213 2216 2219 2222
904	EXTRAM	00:F110	898
1617	FILL	00:F4BB	1208
1639	FILLERR	00:F4EC	1631
1624	FILLS0	00:F4CC	1636
1628	FILLS1	00:F4D5	1633
1637	FILLSX	00:F4E9	1618 1620 1622 1635
466	FLGS	00:005A	1092 1411 1467
3231	FLUSH_SERIAL_BUFF	= 00:FDB4	755 1042 1460 1486 2568 3174
929	FUIRQS	00:F116	932
1247	GBAS	00:F2D1	1236
1041	GDTOD	= 00:F196	1036
1050	GDTOD1	00:F1A2	1039
3620	GENIRQ	= 00:FED6	
2952	GETC3	00:FC89	2945
2953	GETC3A	00:FC8A	2948
2956	GETC4	00:FC8C	2939
2938	GETCH	00:FC71	626 1123 1728 1746 1797 1845 2941 3080 3098
3004	GETSD1	= 00:FCC8	2992
3007	GETSD2	00:FCCC	2994 2999 3002
3009	GETSD3	00:FCD0	2988
2982	GETSD4	00:FCA0	2980

```
1457  GO                = 00:F3DB     1205
1460  GO1                 00:F3DE     1248
```

Defined Symbol Name Value References

1463 GO2 00:F3E4 1493
1501 GOAD 00:F410 1498
1495 GOADDR = 00:F405 1458 1484
1483 GOJSR = 00:F3F4 1206
485 H100HZ 00:0061
1645 HELP = 00:F4EF 1211 1212
1670 HELP1 00:F50E 1675
1677 HELP2 00:F51B 1672
1697 HELPEND = 00:F6B0 1653
1689 HELPEND1 00:F5F4 1648 1651 1652 1653
1680 HELPMENU 00:F51C 1646 1647 1648
1655 HELPX 00:F503 1650
3339 HEXIN 00:FDFB 692 3124 3132
3804 HEXTOPOS = 00:FFA9
3347 HEXXX 00:FE0A 3340 3345
340 HINIB = 00:00F0 2922
490 HR 00:0065 2204 2205 2224 2231 2232 2328 2520
2151 ICLK1 00:F8DA 2154
2156 ICLK2 00:F8E4 2158
2162 ICLK3 00:F8EE 2164
438 IER1 = 00:002D 843 940 2766 3222 3271 3277 3501
366 IER2 = 00:0009 834 955 3506
3390 IFASC 00:FE1E 673 1354 1590
437 IFR1 = 00:002C 842 2756 3503
365 IFR2 = 00:0008 806 833 2185 3509
1152 IJMP 00:F255 1150
2250 INCADAY 00:F96D 2246 2320 2324
2243 INCDAY 00:F95F 2239
2265 INCMTH 00:F986 2256 2259
1997 INCT2 00:F845 1993 1995
1992 INCTMP 00:F83B 1336 1397 1404 2031
2149 INITCLK = 00:F8D8 1037
577 IROM = 00:0000 817 849 905 3540 3548
3631 IRQ1 00:FEDF 3841
369 IRQ1FLG = 00:0040
3634 IRQ2 00:FEE2 3842
370 IRQ2FLG = 00:0080
2555 IRQAR = 00:FA91 3832
2670 IRQAT = 00:FB2E 3831
2692 IRQATERR 00:FB4E 2677
3641 IRQBRK 00:FEE8 3845
3615 IRQRESERVED = 00:FED6 3829 3830 3833
3625 IRQT1 00:FED9 3839
3628 IRQT2 00:FEDC 3840
3373 ISDECIMAL 00:FE15 670 3360
3358 ISHEX 00:FE0B 751 3339
3379 ISN1 00:FE1C 3374 3391
1244 KBAS1 00:F2CB 1241
1239 KBASIC 00:F2C1 1220
1312 KKLLPP1 00:F330 1328
3650 LASTBYTE = 00:FEEC 3850
3778 LASTDY 00:FF95 2252
526 LASTXONOF = 00:0010 2574 2632 2634 2687 2993 2995 3268 3274
1804 LHDONE 00:F747
1588 LISTS2 00:F498 1596

1593 LISTS3 00:F4A3 1591
879 LORAMLP 00:F0E5 884

Defined Symbol Name Value References

339 LOWNIB = 00:000F 1532 2918
1797 LSFIN 00:F73A 1794
1733 LSS = 00:F6C1 1215
1796 LSS0 00:F738 1747 1765 1790
1319 M0 = 00:F340 1329 1331
1328 M001 00:F34C 1321
1273 M1 00:F2F0 1294
3761 MAXTTBL = 00:FF89
1564 MD1 00:F467 1548 1550
489 MIN 00:0064 2199 2200 2202 2525
3769 MINTTBL = 00:FF8F
794 MNTALCALC = 00:F082 1005 1006
801 MNTLEND = 00:F088 1005
1087 MONBRK = 00:F1FC 774 3817
783 MONIRQEND = 00:F07C 928
773 MONIRQTBL = 00:F06A 928 929
492 MONTH 00:0067 2211 2247 2251 2267 2268 2271 2493
1081 MONVEND = 00:F1FC 1071
1077 MONVER = 00:F1BD 1069 1070 1071
767 MONVER# = 00:F063 1709 1710 1711
2049 MOVE 00:F86C 1210
2058 MOVEBAD 00:F87D 2055
1851 MS19OUT = 00:F78B 700 1848
1866 MS19OUTA 00:F7A1 1854 1912
1727 MS28IN = 00:F6BA 708 1730
3317 MVBAD 00:FDF9 3311
3307 MVDATA 00:FDEA 680 2054 3313
785 MXTALCALC = 00:F07C 999 1000
1016 MXTALFND = 00:F175 1001 1007
792 MXTLEND = 00:F082 999
3447 NE46 = 00:FE48 3823
3571 NE46LP 00:FEAC 3592
3565 NE46LP1 00:FEA7 3588
2754 NE47 00:FBB9 3824
2765 NE47A 00:FBCC 2759
2767 NE47B 00:FBD0 2763
3613 NE52 = 00:FED6 3827
3614 NE53 = 00:FED6 3828
3619 NE57 = 00:FED6 3837
3411 NIBBIN1 00:FE30 3406 3408
3637 NMIRQ 00:FEE5 3843
900 NOEXTRAM 00:F10C 894
888 NOEXTROM 00:F0F5 583 746 881
1863 NOFINAL 00:F7A0 1856
862 NOLPWR = 00:F0CD 852 855 858
3586 NORAMRT 00:FEBD 3580 3583
2208 OCTOBER 00:F923
1357 OLD1 00:F37D 1355
2849 OUTCH = 00:FC11 633 1121 1158 1326 1357 1593 1671 1792 1816 2496
2500 2523 2528 2809 2813 2816 2835 2949
2853 OUTCH1 00:FC14 2859
2887 OUTCH1A 00:FC4B 2881
2888 OUTCH1B 00:FC4D 2885
2889 OUTCH2 00:FC4F 2873 2877 2883
2895 OUTCH3 00:FC54 2863

2122 OUTSLASH 00:F8CB 2119
Pre PAGE0 00:0040 442 443

```
Defined         Symbol Name          Value            References

 465   PCH                    00:0059     1099  1234  1245  1377  1463
 464   PCL                    00:0058     1096  1247  1375  1465  2792
 362   PCS3                  = 00:0007     582   818   866   902  3547
 351   PD0                   = 00:0030    3558
 352   PD1                   = 00:0031    3559
 353   PD2                   = 00:0032    3560
 359   PD3                   = 00:0003     845  3549
 397   PD4                   = 00:001C     836  2709  2727  2872  3521  3586
 400   PD5                   = 00:001D     837  3522
 402   PD6                   = 00:0020     840  2641  2742  3006  3216  3218  3244  3279  3523
 354   PDD0                   = 00:0034    3561
 355   PDD1                   = 00:0035    3562
 356   PDD2                   = 00:0036    3563
 408   PDD4                   = 00:001E     838  3220  3524
 409   PDD5                   = 00:001F     839  3525
 410   PDD6                   = 00:0021     841  3212  3214  3526
3535   PDLP1                    00:FE8D    3536
3489   PDOWN                     = 00:FE4B     780  3818
3609   PE44                  = 00:FED6    3820
3610   PE45                  = 00:FED6    3822
3611   PE50                  = 00:FED6    3825
3612   PE51                  = 00:FED6    3826
3616   PE54                  = 00:FED6    3834
3617   PE55                  = 00:FED6    3835
3618   PE56                  = 00:FED6    3836
1807   PERR                     00:F74A    1217
1814   PERR1                    00:F756    1793  1807
1666   PRTSTR                    00:F506     647  1072  1279  1529  1649  1654  1862
 547   PUFLG                  = 00:0020    3565  3590
1373   PUTP                   = 00:F392    1433  1500
1267   R1                    00:F2E3     1272
3085   RDCR                     00:FD07    3101
3049   RDOA                     00:FCE0     652  1293  1431  1497  1932  1945
3062   RDOA2                    00:FCF7    3056
3061   RDOA2A                    00:FCF4    3052
3079   RDOB                     00:FCFD     658  1392  1621  1751  1767  1769  1772  1785  2052  2103
                                 2106  3051  3055
3087   RDOB1                    00:FD0A    3083
3091   RDOB1A                    00:FD12    3088
3097   RDOB2                    00:FD19    3090
3106   RDOB2A                    00:FD2B    3094  3103
3111   RDOB3                    00:FD33    3105
3113   RDOB4                    00:FD37    3081  3099  3109
3012   RD_CH0                   00:FCD3    2974  3031
3015   RD_CH2                   00:FCD5    2983
2971   RD_CHAR                   = 00:FC8F     618  2938
2370   RD_CLOCK                  = 00:F9FC     718
2377   RD_CLP                   00:FA05    2381
1931   RD_SAEA                   00:F801    1517  1617  1843  2015  2049
2657   RECDONE                   00:FB2B    2561  2576  2626  2633  2642  2647  2739  2745  2749
2653   RECDONET                  00:FB25    2637
 948   RECLOC                 = 00:0090     959   962
2609   RECS11                   00:FAE9    2607
2568   RECSC0                   00:FAA4    2581
2575   RECSC1                   00:FAB3    2573
```

```
 947    RECSIZ                  = 00:0010       962    965
2645    RECSXOFF                  00:FB1B       2588
```

Defined Symbol Name Value References

2650 RECSXON 00:FB21 2590
1283 REGSEND 00:F311 1278
1282 REGSTR 00:F2FC 1276 1277 1278
1276 REGTTL 00:F2F3 1262 1430
804 RESCN 00:F088 777
812 RESET = 00:F08F 775 776 779 781 3844
2116 RETBYTE 00:F8C0 2104
3292 RMNDR = 00:0081
2625 RQ0 00:FAFF 2620
2628 RQ1 = 00:FB03
2564 RQ5 00:FA9C 2560
2580 RQ6 00:FAB8 2567
2593 RQ7A 00:FACE 2585
2594 RQ8 00:FACF 2565 2591
2604 RQ8A 00:FAE0 2602
2606 RQ9 00:FAE3 2597
2351 RTC0 00:F9F4 2346
2353 RTC1 00:F9F8 2349
2356 RTC_ERR 00:F9FA 2343 2345
2342 RTC_MODE = 00:F9E6 736
1132 S0 00:F233 1105 3819
1123 S00 00:F22E 1140
1119 S000 00:F224 1159
1144 S1 00:F246 1155
1154 S2 00:F258 1145
341 S28BN = 00:0013 1872
1785 S28G2 00:F722 1777
1784 S28G3 00:F720 1779
1778 S28GD1 00:F714 1784
1766 S28LA1 00:F6F8 1763
488 SEC 00:0063 1029 2152 2162 2193 2195 2197 2293 2304 2377 2404
2530
1410 SETR = 00:F3B7 1264 1437
513 SFLAG 00:0072 1045 1116 1739 1743 1799 1802 2066 2068 2095 2099
2123 2126 2569 2575 2583 2612 2629 2636 2646 2651
2673 2681 2688 2693 2706 2724 2729 2757 2861 2869
2943 2972 2990 2996 3008 3030 3175 3263 3269 3275
522 SFLG = 00:0001 2570 2611 2973 3007
1769 SHORTADDR 00:F700 1759 1761
461 SINCNT 00:0056 966 2601 2606 2619 2623 2979
534 SINEND 00:0076 2596 2599 2604 2625 2977 2984 3235
533 SININDX 00:0075 2594 2609 2987 3234
459 SINPTR 00:0052 960 961 2610 2985
3211 SIOPORTS 00:FD91 1048 3204
1054 SKPE000 = 00:F1AC 1052
2094 SLASH 00:F89F 1221 2111
1837 SLASTEND = 00:F776 1861
1836 SLASTLINE 00:F76C 1859 1860 1861
2876 SNDOUT1 00:FC39 2871
2879 SNDOUT2 00:FC3D 2874
525 SNDOVF = 00:0008 2645 2650 2713 2730 2876 3267 3274
2858 SNDSD2 00:FC1D 2856
2532 SNDT1 00:FA87 1821 2491 2494 2498 2502 2504 2521 2526
2535 SNDTDONE 00:FA90 2524 2529
2520 SNDTIME 00:FA6D 2506

2490 SNDTOD 00:FA3E 1073 2080
2686 SNDXOFF = 00:FB46 2679

```
Defined         Symbol Name          Value            References

2639   SNDXOFFHW               = 00:FB15      2631
2680   SNDXON                   00:FB3E
 421   SON                   = 00:0001      2737  2761  2880  2887  2998  3000
 462   SOUTCNT                  00:0057       968  2718  2855
 532   SOUTEND                  00:0074      2698  2733  2853  2866  3233
 531   SOUTINDX                 00:0073      2697  2721  2732  2858  3232
 460   SOUTPTR                  00:0054       963   964  2722  2868
2833   SPAC                    00:FC09       639  1267  1315  1330  1348  1440  1496  1535  1571  1619
                                1817  1931  1944  2051  2064  2492  2831  3091
2831   SPAC2                    00:FC04      1151  1314  1351  1562  1578  1580  1819  2018  2505  3061
2819   SPACRTS                  00:FC03      2832
 558   SPEED                    00:0086      1018  3183
2721   SQ0                     00:FB81      2719
2694   SQ0A                     00:FB52      2675
2723   SQ1                     00:FB85      2683  2690  2704
2729   SQ2                     00:FB91      2726
2747   SQ3                     00:FBB2      2728  2731  2734
2741   SQ3A                     00:FBA7      2702  2710  2715
 567   SRCE                  = 00:007B      3308
1113   START                    00:F218       688   782  1122  1124  1162  1231  1242  1273  1358  1443
                                1489  1491  1519  1637  1655  1804  1811  1849  2021  2074
                                2129  3085
 528   SXOFFLG                  = 00:0040      2635  2674  2676  2678  2687  2692  3267  3274
 529   SXONFLG                  = 00:0080      2574  2674  2676  2680  2692  2995  3268  3274
 381   T1CH                  = 00:0011
 380   T1CL                  = 00:0010
 367   T1FLG                  = 00:0010
 376   T1LH                  = 00:000D
 375   T1LL                  = 00:000C
 383   T2CH                   = 00:0013       987
 382   T2CL                   = 00:0012       988   997
 992   T2DELAY                  00:F157       993   995
2276   T2EXIT                  = 00:F996      2225  2234  2253  2263  2270  2329
 368   T2FLG                   = 00:0020      2184
 378   T2LH                   = 00:000F       953
 377   T2LL                   = 00:000E       951
 987   T2ZERO                   00:F14F       989
3645   TABLE_START              = 00:FEF3      3648  3655
 430   TACH                    = 00:0027
 429   TACL                    = 00:0026
 428   TALH                    = 00:0025      3191
 427   TALL                    = 00:0024      3189
 372   TCR1                   = 00:000A       847   926   938  3226  3532  3538
 373   TCR2                   = 00:000B       835   957  3529
 568   TEMP                    = 00:0083
 487   TENTHSEC                 00:0062
2073   TGLX1                   00:F891      2071
3274   TGLXONA                  00:FDDC      3266
3278   TGLXONB                  00:FDE4      3272
3261   TGLXONMODE               = 00:FDC7
2063   TGLXONXOFF               = 00:F880      1214
 435   TMCH                    = 00:002B      2192
 434   TMCL                    = 00:002A      2190
 433   TMLH                    = 00:0029      2191
 432   TMLL                    = 00:0028      2189
```

553 TMP0 00:007B 567 1268 1296 1298 1304 1309 1311 1341 1343 1346
1349 1352 1364 1367 1374 1376 1394 1395 1412 1413

1521 WRROUT 00:F41A 1306 1518
2113 WRTBYTE 00:F8BD 2107

Defined	Symbol Name	Value	References
2811	WRTWO	00:FBF8	1877 2534 2803
2818	WRTWORTS	00:FC02	2797 2814
1533	WRX1	00:F42F	1539
2462	WR_ACLOCK	= 00:FA28	741
2469	WR_ACLP	00:FA31	2473
2783	WR_ADDR	00:FBD2	664
2396	WR_CLOCK	= 00:FA12	732
2403	WR_CLP	00:FA1B	2407
2706	XIRQ1	00:FB68	2696
2713	XIRQ1A	00:FB74	2708
2697	XIRQ2	00:FB58	2711 2714
2717	XIRQ3	00:FB7A	2699
946	XMITSIZ	= 00:0004	967
344	XOFF	= 00:0013	2587 2689
343	XON	= 00:0011	2589 2682
524	XONOFLG	= 00:0004	1044 2067 2070 2572 2584 2630 2707 2725 2758 2870 2991 3264 3265
476	XR	00:005C	1089 1470
493	YR	00:0068	2257 2273 2503
477	YREG	00:005D	1090 1471
3648	ZZZSPACE	= 00:0008	

Lines Assembled : 3854 Errors : 0