```
                        ttl     "tp MON65 monitor routines      %s                      Page %d"
                ;********** MONIT11 **********

                        pag     80

                ;               6502 MONITOR
                ;       Written by E.J.M. Visschedijk
                ;Copyright (c) 1986 Kim gebruikersclub Nederland

        9000    monbeg  equ     $9000           ;Begin monitor

                ;Temporary variables on zeropage

        0000    pta     equ     $0000
        0002    ptb     equ     $0002
        0004    tmp0    equ     $0004           ;Current address disassembler

                ;DOS entries

        C020    input   equ     $c020
        C023    output  equ     $c023
        C02F    crlf    equ     $c02f
        C032    spa     equ     $c032
        C035    hnout   equ     $c035
        C038    prbyte  equ     $c038
        C03B    prtext  equ     $c03b
        C041    loupch  equ     $c041

                ;IO65 definitions

        E770    svaint  equ     $e770
        E7B3    brkvector equ   $e7b3
        E7B1    nmivector equ   $e7b1
        E76E    swivector equ   $e76e

                        cont    monit12
                ;********** MONIT12 **********

        9000            org     monbeg

9000 08         mon65   php                     Processorstatus on stack
9001 78                 sei
9002 BA                 tsx
9003 8E 619F            stx     qcomvr          Save stackpointer

9006 AD B3E7            lda     brkvector       Save breakvector
9009 8D 629F            sta     brkvs
900C AD B4E7            lda     brkvector+1
900F 8D 639F            sta     brkvs+1

9012 A9 88              lda     #break&255      Set break vector
9014 8D B3E7            sta     brkvector
9017 A9 90              lda     #break>>8
9019 8D B4E7            sta     brkvector+1

901C AD 6EE7            lda     swivector       Save software interrupt vector
901F 8D 669F            sta     swivs
9022 AD 6FE7            lda     swivector+1
9025 8D 679F            sta     swivs+1

9028 A9 A3              lda     #(swint-1)&255  Set softw. int. vector
902A 8D 6EE7            sta     swivector
902D A9 90              lda     #(swint-1)>>8
902F 8D 6FE7            sta     swivector+1

9032 AD B1E7            lda     nmivector       Save NMIvector
9035 8D 649F            sta     nmivs
9038 AD B2E7            lda     nmivector+1
903B 8D 659F            sta     nmivs+1

903E A9 CB              lda     #nmi&255        Set NMI vector
9040 8D B1E7            sta     nmivector
9043 A9 90              lda     #nmi>>8
9045 8D B2E7            sta     nmivector+1

9048 28                 plp
9049 A9 00              lda     #$00            Reset all variables to $00
904B A2 46              ldx     #(ndvrs-v-1)&255 X is the loopcounter
```

```
904D 9D FF9E    rsta    sta     v-1,x           Reset variable
9050 CA                 dex
9051 D0 FA              bne     rsta            Ready?

9053 A2 7F              ldx     #$7f            Put all CR's in inputbuffer
9055 A9 0D              lda     #$0d
9057 9D 6E9F    rstb    sta     inpbuffer,x
905A CA                 dex
905B 10 FA              bpl     rstb

905D A0 00      setvar  ldy     #$00
905F B9 7190    setvr1  lda     vart1,y         Get offset
9062 F0 0A              beq     setvr2          Branch if end
9064 AA                 tax
9065 B9 7D90            lda     vart2,y         Get data
9068 9D 009F            sta     v,x             Set variable to initial value
906B C8                 iny
906C D0 F1              bne     setvr1

906E 4C CF90    setvr2  jmp     hoofdprog       Jump to the main program

                ;Offsettabel for variables

9071 29         vart1   fcc     spuser
9072 46                 fcc     dntcr
9073 68                 fcc     tabvec
9074 69                 fcc     tabvec+1
9075 2E                 fcc     c02typ          Select 65C02 disassembler
9076 2F                 fcc     dislct          Number of lines to disassemble
9077 10                 fcc     delim           Kind of delimiter between parameters
9078 6A                 fcc     userx           Command X vector to-
9079 6B                 fcc     userx+1         User defined routine
907A 6C                 fcc     usery           Command Y vector to-
907B 6D                 fcc     usery+1         User defined routine
907C 00                 fcc     0               End of tabel

                ;Datatabel for variables

907D FF         vart2   fcc     $ff             SPUSER
907E 25                 fcc     $25             DNTCR
907F F991               fdb     illcoma
9081 01                 fcc     $01             01 is Rockwell, 00 is Synertek/GTE
9082 14                 fcc     20              DISLCT
9083 2C                 fcc     $2c             DELIM is default ,
9084 F691               fdb     illcom          Command X to illegal command
9086 F691               fdb     illcom          Command Y to illegal command

                ;*** Perform break ***

9088 A9 00      break   lda     #$00            Reset flags
908A 2C 419F            bit     disflg          Was disassembler active?
908D 10 03              bpl     btt
908F 20 0E9C            jsr     reszer          Then restore zeropage
9092 20 3BC0    btt     jsr     prtext          Print string
9095 0D                 fcc     $0d
9096 1B6E               fcc     $1b,'n'         Disable invers
9098 1B47               fcc     $1b,'G'         Disable grafics
909A 427265616B         fcc     'Break'
909F 0D00               fcc     $0d,0
90A1 4C F390            jmp     hfderr          Jump to monitor entry


                ;*** Software interrupt routine ***

90A4 AD 70E7    swint   lda     svaint
90A7 8D 249F    i       sta     acc             Save A, X, Y registers
90AA 8E 259F            stx     xreg
90AD 8C 269F            sty     yreg
90B0 68                 pla
90B1 8D 2A9F            sta     preg            Save processor status
90B4 68                 pla
90B5 8D 279F            sta     pcl             Save programcounter low
90B8 68                 pla
90B9 8D 289F            sta     pch             Save programcounter high
90BC BA                 tsx
90BD 8E 299F            stx     spuser          Save stackpointer
90C0 20 3A97            jsr     cmla            Show all registers
90C3 AD 2A9F            lda     preg            Restore old processor status
```

```
90C6 48              pha
90C7 28              plp
90C8 4C F390         jmp    hfderr          Continue elsewhere

                     ;*** Entry after dpressing NMI key ***
                     ;
90CB D8              nmi    cld
90CC 4C A790         jmp    1.b
                     ;
                     cont   monit13
                     ;********** MONIT13 **********

                     ;*** Main program ***

90CF 20 3BC0         hoofdprog jsr  prtext          Print message
90D2 0D4D4F4E36      fcc    '\rMON65 2.01\r'
90D7 3520322E30
90DC 310D
90DE 1B69204861      fcc    $1b,'i HaViSoft ',$1b,'n\r\n',0
90E3 5669536F66
90E8 74201B6E0D
90ED 0A00
90EF BA              tsx                    Get stackpointer
90F0 8E 139F         stx    svstck          And save it
90F3 AE 139F         hfderr ldx    svstck   Error entry, so:
90F6 9A              txs                    Restore the stackpointer
90F7 A2 00           mainpgm ldx   #0
90F9 BD 509F         1      lda    prompt,x
90FC F0 06           beq    2.f
90FE 20 23C0         jsr    output
9101 E8              inx
9102 D0 F5           bne    1.b
9104 8E 479F         2      stx    prcnt
9107 20 2691         jsr    vulbuf          Fill inputbuffer
910A A9 80           lda    #$80            Set flag for printing whole inputbuffer
910C 8D 1E9F         sta    prbffg          After error in second command
910F 20 8191         main   jsr    uitbuffer       Execute command

                     ;This is the returnaddress after execution of a command

9112 20 2FC0         jsr    crlf            Print a CRLF
9115 0E 1E9F         asl    prbffg          Reset the flag
9118 AE 029F         ldx    bufpoint        Get the bufferpointer
911B CA              dex                    Minus 1
911C BD 6E9F         lda    inpbuffer,x     Get the last character of buffer
911F C9 3A           cmp    #$3a
9121 F0 EC           beq    main            After : execute next command
9123 4C F790         jmp    mainpgm         Else restart

                     ;*** Fill the input buffer ***

9126 A2 00           vulbuf ldx    #$00            Reset the counter
9128 8E 029F         stx    bufpoint        Only for UITBUFFER
912B 20 20C0         bufloop jsr   input           Get a character
912E 29 7F           and    #$7f            Don't accept negative characters

9130 C9 0D           cr     cmp    #$0d
9132 D0 04           bne    1.f             Branch if no CR
9134 9D 6E9F         cruit  sta    inpbuffer,x     Put also in buffer
9137 60              rts                    Do not echo

9138 C9 19           1      cmp    #$19            Control Y
913A F0 34           beq    cony            Branch if ^Y
913C C9 20           cmp    #$20            Control characters not allowed
913E 90 EB           bcc    bufloop

9140 C9 7F           cmp    #$7f
9142 D0 0F           bne    stabuf          Check for a delete
9144 E0 00           cpx    #$00            Don't accept delete when on first pos.
9146 F0 E3           beq    bufloop

9148 20 3BC0         delet  jsr    prtext
914B 08200800        fcc    $08,$20,$08,$00 Del. character
914F CA              dex                    Decrease the counter
9150 4C 2B91         jmp    bufloop         Back in loop

9153 9D 6E9F         stabuf sta    inpbuffer,x     Put character in buffer
9156 E8              inx                    Increase counter
```

```
9157 20 23C0          jsr    output          Also echo the character
915A E0 4F            cpx    #$4f            Last character?
915C D0 CD            bne    bufloop         If not, get next
915E A9 07     2      lda    #$07
9160 20 23C0          jsr    output          Ring the bell
9163 20 20C0          jsr    input           Get character
9166 C9 7F            cmp    #$7f
9168 F0 DE            beq    delet           Accept delete
916A C9 0D            cmp    #$0d
916C F0 C6            beq    cruit           Accept also CR
916E D0 EE            bne    2.b

9170 BD 6E9F   cony   lda    inpbuffer,x     Recall the old inputbuffer
9173 C9 0D            cmp    #$0d            End of buffer?
9175 F0 B4            beq    bufloop         Go in loop
9177 20 23C0          jsr    output          Print on output
917A E8               inx                    Next character
917B E0 4F            cpx    #$4f            This is the maximum
917D D0 F1            bne    cony
917F F0 AA            beq    bufloop         Go in loop

                ;*** Read command from buffer ***

9181 AE 029F   uitbuffer ldx  bufpoint        Get the bufferpointer
9184 D0 07            bne    1.f
9186 BD 6E9F          lda    inpbuffer,x
9189 C9 0D            cmp    #$0d
918B F0 23            beq    3.f             Do nothing if only CR was typed
918D BD 6E9F   1      lda    inpbuffer,x     Get character
9190 20 41C0          jsr    loupch
9193 A0 00            ldy    #$00
9195 D9 B191   looktab cmp   comtab,y        Compare with valid commands
9198 F0 08            beq    2.f             Branch if valid
919A C8               iny                    Try next command
919B C0 17            cpy    #23             Only 23 commands
919D D0 F6            bne    looktab         Branch if not on end
919F 6C 689F   error1 jmp    [tabvec]        End of table, jump to vector
91A2 EE 029F   2      inc    bufpoint        Point to next character

91A5 98               tya                    Transfer table offset in A
91A6 0A               asla                   Calculate jumptable offset
91A7 AA               tax                    Offset in X
91A8 BD C791          lda    commtabel+$01,x Get highbyte
91AB 48               pha                    On stack
91AC BD C691          lda    commtabel,x     Get lowbyte
91AF 48               pha                    On stack
91B0 60        3      rts                    Execute command
                ;Returnaddress is in mainprogram

                ;*** Valid commands ***

91B1 40        comtab fcc     '@'            Get address and change contents
91B2 48               fcc     'H'            Hex and ASCII dump
91B3 44               fcc     'D'            Disassembler
91B4 4D               fcc     'M'            Move bytes
91B5 56               fcc     'V'            Verify bytes
91B6 46               fcc     'F'            Fill
91B7 43               fcc     'C'            Check
91B8 50               fcc     'P'            Clear screen
91B9 57               fcc     'W'            Search
91BA 45               fcc     'E'            Execute
91BB 4C               fcc     'L'            List cpu registers
91BC 42               fcc     'B'            Breakpoints
91BD 3C               fcc     '<'            Repeat inputbuffer
91BE 51               fcc     'Q'            Quit monitor as a subroutine
91BF 24               fcc     '$'            Calculate from dec. to hex.
91C0 23               fcc     '#'            Calculate from hex. to dec.
91C1 2B               fcc     '+'            Add two words of 2 bytes
91C2 2D               fcc     '-'            Subtract two words of two bytes
91C3 53               fcc     'S'            Compute the checksum
91C4 58               fcc     'X'            User defined routine
91C5 59               fcc     'Y'            User defined routine

                ;*** Command addresses table ***

91C6 C794      commtabel fdb   cmaap-$01
91C8 0394             fdb     cmh-$01
91CA 1199             fdb     cmd-$01
```

```
91CC 7F95              fdb     cmm-$01
91CE 1496              fdb     cmv-$01
91D0 7196              fdb     cmf-$01
91D2 D696              fdb     cmc-$01
91D4 9698              fdb     cmp-$01
91D6 579B              fdb     cmw-$01
91D8 9B97              fdb     cme-$01
91DA 2E97              fdb     cml-$01
91DC E197              fdb     cmb-$01
91DE 2497              fdb     cm.-$01
91E0 A698              fdb     cmq-$01
91E2 9399              fdb     cm24-$01
91E4 C499              fdb     cm23-$01
91E6 EA99              fdb     cm2b-$01
91E8 039A              fdb     cm2d-$01
91EA 289A              fdb     cms-$01
91EC EF91              fdb     cmx-$01
91EE F291              fdb     cmy-$01

91F0 6C 6A9F   cmx     jmp     [userx]         Jump to user defined command X

91F3 6C 6C9F   cmy     jmp     [usery]         Jump to user defined command Y

               ;*** Illegal command ***

91F6 CE 029F   illcom  dec     bufpoint        Pointer to command
91F9 20 3B92   illcoma jsr     prbufpijl       Print the buffer
91FC 20 3BC0           jsr     prtext          Print text
91FF 070D496C6C        fcc     $07,'\rIllegal command\r\n',0
9204 6567616C20
9209 636F6D6D61
920E 6E640D0A00
9213 4C F390           jmp     hfderr          Jump to error entry

               ;*** Illegal parameters ***

9216 CE 029F   illpar  dec     bufpoint        Pointer to parameter
9219 20 3B92   illpara jsr     prbufpijl       Print the buffer
921C 20 3BC0   illparb jsr     prtext          Print text
921F 070D496C6C        fcc     $07,'\rIllegal parameter(s)\r\n',0
9224 6567616C20
9229 706172616D
922E 6574657228
9233 73290D0A00
9238 4C F390           jmp     hfderr          Jump to error entry

               ;*** Print inputbuffer and arrow to error ***

923B 20 2FC0   prbufpijl jsr   crlf
923E 2C 1E9F           bit     prbffg          Was it the first command
9241 30 12             bmi     illcm2          If yes, then don't print buffer

9243 A2 00             ldx     #$00            Print the buffer
9245 BD 6E9F   1       lda     inpbuffer,x     Get character
9248 C9 0D             cmp     #$0d
924A F0 06             beq     illcom2         On end yet
924C 20 23C0           jsr     output          Print the character
924F E8                inx                     Update pointer
9250 D0 F3             bne     1.b             Branch always

9252 20 2FC0   illcom2 jsr     crlf            Print CRLF
9255 AE 029F   illcm2  ldx     bufpoint        Get pointer
9258 F0 08             beq     pijl            Branch if on first position
925A A9 20     2       lda     #'              Get a space
925C 20 23C0           jsr     output          Print it
925F CA                dex                     Update pointer
9260 D0 F8             bne     2.b             End reached yet?
9262 2C 1E9F   pijl    bit     prbffg          Was it the first command
9265 10 06             bpl     1.f             If not, then don't print prompt spaces
9267 AE 479F           ldx     prcnt
926A 20 5893           jsr     xspace
926D A9 5E     1       lda     #'^             Get the arrow character
926F 4C 23C0           jmp     output          Print the arrow

               ;*** Place arrow on previous space and errormessage ***

9272 20 7892   zpijlsp jsr     zkvsp           Look for previous space
9275 4C 1992           jmp     illpara         Error message
```

```
                ;*** Look for previous space ***

9278 AE 029F    zkvsp   ldx     bufpoint        Get pointer
927B CA         zoek    dex                     One place backwards
927C BD 6E9F            lda     inpbuffer,x     Get character
927F CD 109F            cmp     delim           Check for delimiter
9282 D0 F7              bne     zoek            Continue if not a delimiter
9284 8E 029F            stx     bufpoint        This is the spaceplace
9287 60                 rts

                ;*** Get character from inputbuffer, convert lower to upper ***

9288 AE 029F    gchb    ldx     bufpoint        Get pointer
928B BD 6E9F            lda     inpbuffer,x     Get character
928E EE 029F            inc     bufpoint        Update pointer
9291 4C 41C0            jmp     loupch

                ;*** Get character from inputbuffer ***

9294 AE 029F    gchb1   ldx     bufpoint        Get pointer
9297 BD 6E9F            lda     inpbuffer,x     Get character
929A EE 029F            inc     bufpoint        Update pointer
929D 60                 rts

                        cont    monit14
                ;********** MONIT14 **********
                ;
                ;
                ;*** Get one parameter from inputbuffer ***
                ;
929E 20 0693    eenpar  jsr     insch           ;Get parameter
92A1 D0 3D              bne     parer           ;Branch if error occured
92A3 20 EF92            jsr     copya           ;Copy parameter to PARA
92A6 A0 00              ldy     #$00            ;N=0 for normal exit
92A8 60                 rts
                ;
                ;*** Get two parameters from inputbuffer ***
                ;
92A9 20 E392    tweepar jsr     in              ;Get parameter
92AC CD 109F            cmp     delim           ;Check for delimiter
92AF D0 2F              bne     parer           ;Then incorrect
92B1 20 EF92            jsr     copya           ;Copy first parameter to PARA
92B4 20 0693            jsr     insch           ;Get next parameter
92B7 D0 27              bne     parer           ;Branch if error occured
92B9 20 F392            jsr     copyb           ;Copy this parameter two PARB
92BC A0 00              ldy     #$00            ;N=0 for normal exit
92BE 60                 rts
                ;
                ;*** Get three parameters from inputbuffer ***
                ;
92BF 20 E392    driepar jsr     in              ;Get parameter
92C2 CD 109F            cmp     delim           ;Check for delimiter
92C5 D0 19              bne     parer           ;Then incorrect
92C7 20 EF92            jsr     copya           ;Copy first parameter to PARA
92CA 20 E392            jsr     in              ;Get next parameter
92CD CD 109F            cmp     delim           ;Check for delimiter
92D0 D0 0E              bne     parer           ;Then incorrect
92D2 20 F392            jsr     copyb           ;Copy second parameter in PARB
92D5 20 0693            jsr     insch           ;Get next parameter
92D8 D0 06              bne     parer           ;Branch if error occured
92DA 20 F792            jsr     copyc           ;Copy third parameter to PARC
92DD A0 00              ldy     #$00            ;N=0 for normal exit
92DF 60                 rts
                ;
92E0 A0 FF      parer   ldy     #$ff            ;N=1
92E2 60                 rts
                ;
                ;*** Get parameter ***
                ;
92E3 20 1093    in      jsr     resin           ;Reset inputbytes
92E6 20 8892    ina     jsr     gchb            ;Get character
92E9 20 1993            jsr     hexnum          ;Convert from ASCII to HEX
92EC 10 F8              bpl     ina             ;Last character was HEX
92EE 60                 rts                     ;In A the character following the parameter!!
                ;
                ;*** Copy parameters ***
                ;
```

```
92EF A2 00      copya   ldx     #$00            ;X=0, point to PARA
92F1 F0 06              beq     copy
92F3 A2 02      copyb   ldx     #$02            ;X=2, point to PARB
92F5 D0 02              bne     copy
92F7 A2 04      copyc   ldx     #$04            ;X=4, point to PARC
92F9 AD 119F    copy    lda     inl             ;Get bufferbyte low
92FC 9D 039F            sta     para1,x         ;Copy to parameter pointed by X
92FF AD 129F            lda     inh             ;Get bufferbyte high
9302 9D 049F            sta     parah,x         ;Copy to parameter pointed by X
9305 60                 rts
                ;
9306 20 E392    insch   jsr     in              ;Get parameter and check del.
                ;
                ;*** Check for delimiter ***
                ;
9309 C9 3A      schtt   cmp     #':             ;This is a valid delimiter
930B F0 02              beq     schttend
930D C9 0D              cmp     #$0d            ;CR also allowed
930F 60         schttend rts                    ;Valid then Z=1
                ;
                ;*** Reset bufferbytes ***
                ;
9310 A0 00      resin   ldy     #$00
9312 8C 119F            sty     inl
9315 8C 129F            sty     inh
9318 60                 rts
                ;
                ;*** Convert from ASCII to HEX ***
                ;
9319 20 3593    hexnum  jsr     ashett          ;ASCII to HEX
931C 30 14              bmi     hnub            ;Branch if error
931E A2 04              ldx     #$04            ;Bit counter
9320 0E 119F    hnua    asl     inl             ;Shift INL in INH via carry
9323 2E 129F            rol     inh             ;Into INH
9326 CA                 dex                     ;4 bits in one nibble
9327 D0 F7              bne     hnua            ;Ready?
9329 0D 119F            ora     inl             ;4 LSB's from accu in INL
932C 8D 119F            sta     inl             ;Save it
932F A0 00              ldy     #$00            ;Normal exit, Z=1
9331 60                 rts
9332 A0 FF      hnub    ldy     #$ff            ;N=1 then error
9334 60                 rts
                ;
9335 C9 30      ashett  cmp     #$30            ;Invalid is 00....2F
9337 30 0C              bmi     notvat
9339 C9 3A              cmp     #$3a
933B 30 0B              bmi     valit
933D C9 41              cmp     #$41            ;Invalid is 3A....40
933F 30 04              bmi     notvat
9341 C9 47              cmp     #$47            ;Invalid is 47....FF
9343 30 03              bmi     valit
9345 A0 FF      notvat  ldy     #$ff            ;N=1 then error
9347 60                 rts
                ;
9348 C9 40      valit   cmp     #$40            ;Is it A...F
934A 30 03              bmi     valt            ;Branch if number
934C 18                 clc
934D 69 09              adc     #$09            ;Add 09 for A...F
934F 29 0F      valt    and     #$0f            ;Only 4 LSB's signif.
9351 60                 rts
                ;
                ;*** Print a byte and a space ***
                ;
9352 20 38C0    prbyts  jsr     prbyte          ;Print a byte
9355 4C 32C0            jmp     spa             ;Print a space
                ;
                ;*** Print X spaces ***
                ;
9358 20 32C0    xspace  jsr     spa             ;Print a space
935B CA                 dex                     ;Decrease pointer
935C D0 FA              bne     xspace          ;Ready?
935E 60                 rts
                ;
                ;*** Increase X pointers ***
                ;
935F EE 179F    pntxinc inc     pntxl           ;Increase lowbyte
9362 D0 03              bne     pntxinca
9364 EE 189F            inc     pntxh           ;Increase highbyte
```

```
9367 60        pntxinca rts
               ;
               ;*** Increase Y pointers ***
               ;
9368 EE 1B9F   pntyinc inc    pntyl          ;Increase lowbyte
936B D0 03             bne    pntyinca
936D EE 1C9F           inc    pntyh          ;Increase Highbyte
9370 60        pntyinca rts
               ;
               ;*** Decrease X or Y pointers ***
               ;
9371 A2 00     pointdec ldx   #$00           ;Pointer to X
9373 F0 02             beq    1.f            ;Skip next instruction
9375 A2 04     pntydec ldx    #$04           ;Pointer to Y
9377 38        1       sec
9378 BD 179F           lda    pntxl,x        ;First lowbyte
937B E9 01             sbc    #$01           ;Minus 1
937D 9D 179F           sta    pntxl,x        ;Save result in variable
9380 B0 03             bcs    pntdce         ;Then don't decrease highbyte
9382 DE 189F           dec    pntxh,x        ;Decrease highbyte
9385 60        pntdce  rts
               ;
               ;*** Check parameters ***
               ;
9386 AD 059F   chck    lda    parbl          ;Substract B from A
9389 CD 039F           cmp    paral
938C AD 069F           lda    parbh
938F ED 049F           sbc    parah          ;Result is in carry
9392 60                rts                   ;Error if C=0
               ;
               ;*** Check for end ***
               ;
9393 A0 01     checkend ldy   #$01           ;Loopcounter & pointer
9395 B9 059F   check   lda    parbl,y        ;Get byte
9398 D9 179F           cmp    pntxl,y        ;Check if equal
939B D0 05             bne    1.f            ;Branch if not equal
939D 88                dey                   ;Update pointer
939E 10 F5             bpl    check          ;Next please
93A0 18                clc                   ;C=0 then end reached!!!
93A1 60                rts
93A2 38        1       sec                   ;C=1 then not ready yet
93A3 60                rts
               ;
               ;*** Copy parameters ***
               ;
93A4 AD 039F   copypar lda    paral
93A7 AE 049F           ldx    parah          ;PARA to X
93AA 8D 179F           sta    pntxl
93AD 8E 189F           stx    pntxh
93B0 AD 079F           lda    parcl          ;PARC to Y
93B3 AE 089F           ldx    parch
93B6 8D 1B9F           sta    pntyl
93B9 8E 1C9F           stx    pntyh
93BC 60                rts
               ;
               ;*** Common start for several commands ***
               ;
93BD 20 D393   begin   jsr    testsp         ;Check for space after command
93C0 20 A992           jsr    tweepar        ;Get two parameters
93C3 10 03             bpl    beginb         ;Then no error
93C5 4C 1692           jmp    illpar         ;Error in parameter
93C8 20 B693   beginb  jsr    chck           ;PARA<PARB?
93CB B0 03             bcs    beginc         ;Branch if no error
93CD 4C 7292           jmp    zpijlsp        ;Place arrow and errormessage
93D0 4C A493   beginc  jmp    copypar        ;Copy the parameters
               ;
               ;*** Check for a space after a command ***
               ;
93D3 20 8892   testsp  jsr    qchb           ;Get character
93D6 C9 20             cmp    #$20
93D8 F0 03             beq    testspa        ;Was it a space
93DA 4C F691           jmp    illcom         ;Error
93DD 60        testspa rts
               ;
               ;*** Make selfmodifying routines ***
               ;
93DE A9 AD     adpntx  lda    #$ad           ;LDA ABS
93E0 D0 0A             bne    stpntx
```

```
93E2 A9 BD      bdpntx  lda     #$bd            ;LDA ABS,X
93E4 D0 06              bne     stpntx
93E6 A9 CD      cdpntx  lda     #$cd            ;CMP ABS
93E8 D0 02              bne     stpntx
93EA A9 8D      d8pntx  lda     #$8d            ;STA ABS
93EC 8D 169F    stpntx  sta     pntx            ;In program with pointer X
93EF A9 60              lda     #$60            ;RTS
93F1 8D 199F            sta     pntx+$03
93F4 60                 rts
                ;
93F5 A9 AD      adpnty  lda     #$ad            ;LDA ABS
93F7 D0 02              bne     stpnty
93F9 A9 8D      d8pnty  lda     #$8d            ;STA ABS
93FB 8D 1A9F    stpnty  sta     pnty            ;In program with pointer Y
93FE A9 60              lda     #$60            ;RTS
9400 8D 1D9F            sta     pnty+$03
9403 60                 rts
                ;
                        cont    monit15
                ;********** MONIT15 **********
                ;
                ;*** Hex and ascii dump ***
                ;
9404 20 BD93    cmh     jsr     begin           ;Get parameters and check them
9407 20 E293            jsr     bdpntx          ;Make a selfmodifying program
940A A9 FF              lda     #$ff            ;Reset endflag
940C 8D 229F            sta     eindvlag
940F 20 2FC0            jsr     crlf            ;Print a CRLF
                ;
9412 A2 06              ldx     #$06
9414 20 5893            jsr     xspace          ;Print 6 spaces
9417 A0 00              ldy     #$00            ;Reset columncounter
9419 98         hexdd   tya
941A 20 35C0            jsr     hnout           ;Print a nibble
941D 20 32C0            jsr     spa             ;Two spaces
9420 20 32C0            jsr     spa
9423 C8                 iny                     ;Next columnnumber
9424 C0 10              cpy     #$10            ;0F is the maximum
9426 D0 F1              bne     hexdd           ;Ready?
                ;
9428 20 2FC0    hexdeq  jsr     crlf
942B 2C 229F            bit     eindvlag        ;Stop now?
942E 30 01              bmi     hexdeaq         ;Branch if continue
9430 60                 rts                     ;End of routine
                ;
9431 A2 00      hexdeaq ldx     #$00
9433 AD 189F            lda     pntxh           ;Get higbyte of address
9436 20 38C0            jsr     prbyte          ;Highbyte
9439 AD 179F            lda     pntxl           ;Get lowbyte of address
943C 20 38C0            jsr     prbyte          ;Print it
943F 20 3BC0            jsr     prtext          ;Print text
9442 3A2000             fcc     ': ',0
9445 20 169F    hexdf   jsr     pntx            ;Get data
9448 20 38C0            jsr     prbyte          ;Print it
944B 20 32C0            jsr     spa             ;Print a space
                ;
944E 18                 clc                     ;Calculate endvalue
944F 8A                 txa
9450 6D 179F            adc     pntxl           ;Add the X pointer
9453 8D 149F            sta     pntxlsave       ;Save the result
9456 AD 189F            lda     pntxh           ;Get the highbyte
9459 69 00              adc     #$00            ;Add eventually the carry
945B 8D 159F            sta     pntxhsave       ;Save also this value
                ;
945E A0 01              ldy     #$01            ;Check if on end
9460 B9 059F    hexdg   lda     parbl,y         ;Get endparameter
9463 D9 149F            cmp     pntxlsave,y     ;Compare with current position
9466 D0 08              bne     hexdh           ;Branch if not equal
9468 88                 dey                     ;Compare also lowbytes
9469 10 F5              bpl     hexdg           ;Continue elsewhere
946B A9 00              lda     #$00            ;On end now
946D 8D 229F            sta     eindvlag        ;Set the endflag
9470 E8         hexdh   inx
9471 E0 10              cpx     #$10            ;On end of line?
9473 D0 D0              bne     hexdf           ;Branch if not ready
9475 20 32C0            jsr     spa             ;Else print a space
                ;
9478 A2 00              ldx     #$00
```

```
947A 20 169F    hexdi   jsr     pntx            ;Get again first byte of line
947D 20 9C94            jsr     prgrnr
9480 E8         hexdl   inx                     ;Increase pointer
9481 E0 10              cpx     #$10            ;On end?
9483 D0 F5              bne     hexdi           ;Branch if not ready
9485 20 B694            jsr     incpnt          ;Increase the pointer
9488 4C 2894            jmp     hexdeq          ;Jump back in loop
                ;
948B A9 1B      presc   lda     #$1b
948D 2C                 fcc     $2c             ;Skip next instruction
948E A9 46      prgrf   lda     #'F
9490 2C                 fcc     $2c             ;Skip next instruction
9491 A9 47      prgrg   lda     #'G
9493 2C                 fcc     $2c             ;Skip next instruction
9494 A9 69      prinv   lda     #'i
9496 2C                 fcc     $2c             ;Skip next instruction
9497 A9 6E      prnor   lda     #'n
9499 4C 23C0            jmp     output          ;Print the character
                ;
949C C9 20      prgrnr  cmp     #$20
949E 90 03              bcc     prgrna          ;Branch if controlchar.
94A0 4C 23C0            jmp     output          ;Print the byte as ASCII
94A3 48         prgrna  pha
94A4 20 8B94            jsr     presc           ;Print escape character
94A7 20 8E94            jsr     prgrf           ;Set grafics
94AA 68                 pla
94AB 09 60              ora     #$60            ;Otherwise as control char. printed
94AD 20 23C0            jsr     output          ;Print grafics character
94B0 20 8B94            jsr     presc
94B3 4C 9194            jmp     prgrg           ;Set no grafics and return
                ;
                ;*** Increase X pointers with $10 ***
                ;
94B6 18         incpnt  clc
94B7 AD 179F            lda     pntx1           ;First lowbyte
94BA 69 10              adc     #$10            ;16 bytes for 1 line
94BC 8D 179F            sta     pntx1           ;Save it
94BF AD 189F            lda     pntxh           ;Then highbyte
94C2 69 00              adc     #$00            ;Also the carry
94C4 8D 189F            sta     pntxh           ;Save result
94C7 60                 rts
                ;
                ;*** Command fetch address and change data ***
                ;
94C8 20 1093    cmaap   jsr     resin           ;Reset input buffers
94CB 8C 099F            sty     aapgew          ;Also reset address changed flag
94CE 20 D393            jsr     testsp          ;Next character must be a space
94D1 20 DE93            jsr     adpntx          ;Make a ramprog.
94D4 20 9E92            jsr     eenpar          ;Get one parameter
94D7 10 03              bpl     monb
94D9 4C 1692            jmp     illpar          ;Illegal parameter
94DC CE 029F    monb    dec     bufpoint
94DF 20 8892            jsr     gchb            ;Get last character
94E2 C9 0D              cmp     #$0d
94E4 F0 03              beq     monc            ;Only 'CR' is allowed
94E6 4C F691            jmp     illcom
94E9 20 A493    monc    jsr     copypar         ;Copy address
94EC 20 2FC0    monca   jsr     crlf
94EF AE 189F            ldx     pntxh           ;Get the address
94F2 AC 179F            ldy     pntx1
94F5 20 169F            jsr     pntx            ;Get data from this address
94F8 20 6596            jsr     pradda          ;Print address and data
94FB 20 32C0            jsr     spa
94FE 20 20C0    mond    jsr     input           ;Get character from input device
9501 20 41C0            jsr     loupch
9504 C9 27              cmp     #$27
9506 D0 0F              bne     mond1           ;Was it a '
9508 20 23C0            jsr     output          ;Print it
950B 20 169F            jsr     pntx            ;Get data
950E 20 9C94            jsr     prgrnr          ;Print the data as ascii
9511 20 32C0            jsr     spa
9514 4C FE94            jmp     mond            ;Back in loop
9517 C9 5C      mond1   cmp     #$5c            ;Was it a \
9519 D0 11              bne     mone
951B 20 23C0            jsr     output          ;Echo the \
951E 20 20C0            jsr     input           ;Accept ascii characters
9521 C9 20              cmp     #$20            ;Don't accept control characters
9523 90 D9              bcc     mond
```

```
9525 8D 119F            sta     inl
9528 48                 pha
9529 4C 5D95            jmp     monha
952C C9 0D      mone    cmp     #$0d
952E D0 06              bne     monf            ;Was it a 'CR'
9530 20 6895            jsr     store           ;Then store data if data was changed
9533 4C 2FC0            jmp     crlf            ;End of routine
9536 C9 0A      monf    cmp     #$0a
9538 D0 09              bne     mong            ;Was it a linefeed
953A 20 6895            jsr     store           ;Store data if it was changed
953D 20 5F93            jsr     pntxinc         ;Get next address
9540 4C EC94            jmp     monca           ;Back in loop
9543 C9 0B      mong    cmp     #$0b
9545 D0 09              bne     monh            ;Was it a vert. tab
9547 20 6895            jsr     store           ;Store data if changed
954A 20 7193            jsr     pointdec        ;Get previous address
954D 4C EC94            jmp     monca           ;Back in loop
9550 C9 20      monh    cmp     #$20            ;Don't accept control characters
9552 90 AA              bcc     mond
9554 48                 pha
9555 20 1993            jsr     hexnum
9558 10 03              bpl     monha           ;Character accepted
955A 68                 pla
955B D0 A1              bne     mond            ;Branch always
                ;
955D 68         monha   pla
955E 20 23C0            jsr     output          ;Echo the character
9561 A9 FF              lda     #$ff
9563 8D 099F            sta     aapgew          ;Set flag, data is changed
9566 D0 96              bne     mond
                ;
                ;*** Store the changed data ***
                ;
9568 2C 099F    store   bit     aapgew          ;Was data changed?
956B 10 0C              bpl     strnd           ;Branch if not
956D 20 EA93            jsr     d8pntx          ;Make a ramprog.
9570 AD 119F            lda     inl             ;Get data
9573 20 169F            jsr     pntx            ;Store it
9576 20 DE93            jsr     adpntx          ;Restore old ramprog.
9579 20 1093    strnd   jsr     resin           ;Reset input buffers
957C 8C 099F            sty     aapgew          ;Reset flag
957F 60                 rts
                ;
                        cont    monit16
                ;********** MONIT16 **********
                ;
                ;*** Move databytes ***
                ;
9580 20 FE95    cmm     jsr     beg3            ;Get 3 parameters, check A<B
9583 20 DE93            jsr     adpntx          ;Make a ramprog.
9586 20 F993            jsr     d8pnty          ;Make a ramprog.
9589 AD 039F            lda     paral
958C CD 079F            cmp     parcl           ;Test for move direction
958F AD 049F            lda     parah
9592 ED 089F            sbc     parch
9595 90 1A              bcc     hilo            ;Move downwards
9597 20 A493            jsr     copypar         ;Copy parameters in ramprog.
959A 20 169F    move    jsr     pntx            ;Get byte
959D 20 1A9F            jsr     pnty            ;Move it
95A0 20 9393            jsr     checkend        ;Ready?
95A3 90 09              bcc     movea           ;Branch if yes
95A5 20 5F93            jsr     pntxinc         ;Increase pointers
95A8 20 6893            jsr     pntyinc
95AB 4C 9A95            jmp     move            ;Back in loop
95AE 4C 2FC0    movea   jmp     crlf            ;End of routine
95B1 18         hilo    clc                     ;Calculate last destination address
95B2 AD 079F            lda     parcl
95B5 6D 059F            adc     parbl
95B8 A8                 tay
95B9 AD 089F            lda     parch           ; C':=C+B-A
95BC 6D 069F            adc     parbh
95BF 48                 pha
95C0 38                 sec
95C1 98                 tya
95C2 ED 039F            sbc     paral
95C5 8D 079F            sta     parcl
95C8 68                 pla
95C9 ED 049F            sbc     parah
```

```
95CC 8D 089F            sta     parch
95CF AC 039F            ldy     paral           ;B:=A,A:=B,C:=C'
95D2 AE 049F            ldx     parah
95D5 AD 059F            lda     parbl
95D8 8D 039F            sta     paral
95DB AD 069F            lda     parbh
95DE 8D 049F            sta     parah
95E1 8C 059F            sty     parbl
95E4 8E 069F            stx     parbh
95E7 20 A493            jsr     copypar         ;Copy the parameters in ramprog.
95EA 20 169F    moveb   jsr     pntx            ;Get byte
95ED 20 1A9F            jsr     pnty            ;Move it
95F0 20 9393            jsr     checkend        ;Ready?
95F3 90 B9              bcc     movea           ;Branch if yes
95F5 20 7193            jsr     pointdec        ;Decrease pointers
95F8 20 7593            jsr     pntydec
95FB 4C EA95            jmp     moveb           ;Back in loop
                ;
                ;*** Get 3 parameters and check A<B ***
                ;
95FE 20 D393    beg3    jsr     testsp          ;Was next character a space?
9601 20 BF92            jsr     driepar         ;Get 3 parameters
9604 10 03              bpl     beg3a
9606 4C 1692            jmp     illpar          ;Illegal parameter
9609 20 8693    beg3a   jsr     chck            ;PARA<PARB
960C B0 06              bcs     beg3b           ;Branch if correct
960E 20 7892            jsr     zkvsp           ;Search last space
9611 4C 7292            jmp     zpijlsp         ;Point the arrow and error message
9614 60         beg3b   rts
                ;
                ;*** Verify bytes ***
                ;
9615 20 FE95    cmv     jsr     beg3            ;Get 3 parameters
9618 20 F593            jsr     adpnty          ;Make ramprogs.
961B 20 E693            jsr     cdpntx
961E 20 A493            jsr     copypar         ;Copy parms. into ramprogs.
9621 20 1A9F    veria   jsr     pnty            ;Get byte
9624 20 169F            jsr     pntx            ;Verify
9627 D0 11              bne     vererr          ;Branch if not the same
9629 20 9393    verib   jsr     checkend        ;Ready?
962C B0 03              bcs     veric
962E 4C 2FC0            jmp     crlf            ;End of routine
9631 20 6893    veric   jsr     pntyinc         ;Increase the pointers
9634 20 5F93            jsr     pntxinc
9637 4C 2196            jmp     veria           ;Back in loop
963A 20 4096    vererr  jsr     prerrbyt        ;Print the error byte
963D 4C 2996            jmp     verib           ;Back in loop
                ;
                ;*** Print the differences ***
                ;
9640 48         prerrbyt pha                    ;Save the byte
9641 20 2FC0            jsr     crlf
9644 AE 1C9F            ldx     pntyh           ;Get address
9647 AC 1B9F            ldy     pntyl
964A 68                 pla
964B 20 6596            jsr     pradda          ;Print address and data
964E 20 DE93            jsr     adpntx          ;Make ramprog.
9651 A2 02              ldx     #$02
9653 20 5893            jsr     xspace          ;Print two spaces
9656 20 169F            jsr     pntx            ;Get other byte
9659 AE 189F            ldx     pntxh           ;Get address
965C AC 179F            ldy     pntxl
965F 20 6596            jsr     pradda          ;Print address and data
9662 4C E693            jmp     cdpntx          ;Restore ramprog.
                ;
                ;*** Print address in X,Y and data in A ***
                ;
9665 48         pradda  pha
9666 8A                 txa
9667 20 38C0            jsr     prbyte          ;First print address
966A 98                 tya
966B 20 5293            jsr     prbyts
966E 68                 pla
966F 4C 38C0            jmp     prbyte          ;Print the data
                ;
                ;*** Fill command ***
                ;
9672 20 B296    cmf     jsr     twpaby          ;Get two parameters
```

```
9675 20 EA93          jsr     d8pntx          ;Make a ramprog.
9678 20 A493          jsr     copypar         ;Copy parameters into ramprog.
967B 20 8892          jsr     gchb            ;Get next character
967E C9 27            cmp     #$27
9680 F0 1E            beq     fillb           ;Branch if it is ascii
9682 CE 029F          dec     bufpoint
9685 20 E392          jsr     in              ;Check the data
9688 20 0993  fillf   jsr     schtt
968B D0 1F            bne     fillerr         ;Branch if illegal data
968D AD 119F          lda     inl             ;Get the byte
9690 AA       fillaa  tax                     ;Save the fillbyte
9691 8A       filla   txa                     ;Restore the fillbyte
9692 20 169F          jsr     pntx            ;Store the byte
9695 20 9393          jsr     checkend        ;Ready?
9698 90 15            bcc     fillend         ;Branch if yes
969A 20 5F93          jsr     pntxinc         ;Increase pointer
969D 4C 9196          jmp     filla           ;Continue elsewhere
96A0 20 8892  fillb   jsr     gchb            ;Get fillbyte as ascii
96A3 8D 119F          sta     inl             ;Save it
96A6 20 8892          jsr     gchb            ;Get next character
96A9 4C 8896          jmp     fillf           ;Continue elsewhere
96AC 4C 1692  fillerr jmp     illpar          ;Illegal parameter
96AF 4C 2FC0  fillend jmp     crlf            ;End of routine
              ;
96B2 20 D393  twpaby  jsr     testsp          ;Test for a space
96B5 20 E392          jsr     in              ;Get one parameter
96B8 CD 109F          cmp     delim           ;Check for delimiter
96BB D0 EF            bne     fillerr
96BD 20 EF92          jsr     copya           ;Parameter in PARA
96C0 20 E392          jsr     in              ;Get next parameter
96C3 CD 109F          cmp     delim           ;Check for delimiter
96C6 D0 E4            bne     fillerr
96C8 20 F392          jsr     copyb           ;Parameter in PARB
96CB 20 8693          jsr     chck            ;Check A<B
96CE B0 06            bcs     twpend          ;Branch if correct
96D0 20 7892          jsr     zkvsp           ;Search the space
96D3 4C 7292          jmp     zpijlsp         ;Error message
96D6 60       twpend  rts
              ;
              ;*** Check command ***
              ;
96D7 20 B296  cmc     jsr     twpaby          ;Get two parameters
96DA 20 DE93          jsr     adpntx          ;Make a ramprog.
96DD 20 A493          jsr     copypar         ;Copy parameters into ramprog.
96E0 20 8892          jsr     gchb            ;Get next character
96E3 C9 27            cmp     #$27
96E5 F0 21            beq     cmcb            ;Branch if it was ascii
96E7 CE 029F          dec     bufpoint
96EA 20 E392          jsr     in              ;Check the data
96ED 20 0993  cmcf    jsr     schtt           ;Check delimiter
96F0 D0 BA            bne     fillerr
96F2 20 169F  cmcc    jsr     pntx            ;Get byte
96F5 CD 119F          cmp     inl             ;Compare with typed character
96F8 D0 1A            bne     cmcg            ;Branch if different
96FA 20 9393  cmcd    jsr     checkend        ;Ready?
96FD B0 03            bcs     cmce
96FF 4C 2FC0          jmp     crlf            ;End of routine
9702 20 5F93  cmce    jsr     pntxinc         ;Increase the pointer
9705 4C F296          jmp     cmcc            ;Back in loop
9708 20 8892  cmcb    jsr     gchb            ;Get ascii character
970B 8D 119F          sta     inl             ;Save it
970E 20 8892          jsr     gchb            ;Get next character
9711 4C ED96          jmp     cmcf            ;Continue elsewhere
9714 48       cmcg    pha
9715 20 2FC0          jsr     crlf
9718 68               pla
9719 AC 179F          ldy     pntxl
971C AE 189F          ldx     pntxh
971F 20 6596          jsr     pradda          ;Print error byte
9722 4C FA96          jmp     cmcd            ;Back in loop
              ;
              ;*** Repeat the inputbuffer ***
              ;
9725 A9 00    cm.     lda     #$00            ;Reset bufferpointer
9727 8D 029F          sta     bufpoint
972A 68               pla                     ;Forget returnaddress
972B 68               pla
972C 4C 0F91          jmp     main            ;Jump in mainloop
```

```
                        ;
                        ;*** List CPU registers command ***
                        ;
972F 20 8892    cml     jsr     gchb            ;Get next character
9732 20 0993            jsr     schtt           ;Check delimiter
9735 F0 03              beq     cmla
9737 4C F691            jmp     illcom
973A 20 38C0    cmla    jsr     prtext          ;Print text
973D 1B6E               fcc     $1b,'n'         Disable invers
973F 1B47               fcc     $1b,'G'         Disable grafics
9741 0D41202058         fcc     '\rA  X  Y  PC   SP NV BDIZC\r',0
9746 2020592020
974B 5043202020
9750 5350204E56
9755 204244495A
975A 430D00
975D AD 249F            lda     acc             ;Get A
9760 20 5293            jsr     prbyts
9763 AD 259F            lda     xreg            ;Get X
9766 20 5293            jsr     prbyts
9769 AD 269F            lda     yreg            ;Get Y
976C 20 5293            jsr     prbyts
976F AD 289F            lda     pch             ;Get programcounter
9772 20 38C0            jsr     prbyte
9775 AD 279F            lda     pcl
9778 20 5293            jsr     prbyts
977B AD 299F            lda     spuser          ;Get stackpointer
977E 20 5293            jsr     prbyts
9781 AD 2A9F            lda     preg            ;Get processor status
9784 A2 08      showpra ldx     #$08            ;Bitcounter
9786 0A         spra    asla
9787 48                 pha                     ;If carry=1 then print 1
9788 20 9297            jsr     pr01            ;Print 0 or 1
978B 68                 pla
978C CA                 dex
978D D0 F7              bne     spra            ;Ready?
978F 4C 2FC0            jmp     crlf            ;End of routine
                        ;
9792 90 03      pr01    bcc     pr0             ;Print 0 if carry=0
9794 A9 01              lda     #$01
9796 2C                 fcc     $2c             ;Skip next instruction
9797 A9 00      pr0     lda     #$00
9799 4C 35C0    pr01a   jmp     hnout           ;Print 0 or 1
                        ;
                                cont    monit17
                        ;*********** MONIT17 **********
                        ;
                        ;*** Command execute ***
                        ;
979C 20 8892    cme     jsr     gchb            ;Get next character
979F C9 20              cmp     #$20
97A1 F0 1E              beq     cmec            ;Branch if space
97A3 C9 0D              cmp     #$0d
97A5 F0 03              beq     cmea            ;Branch if 'CR'
97A7 4C F691    cmeb    jmp     illcom
97AA 2C 429F    cmea    bit     bkptf1          ;Was there a breakpoint
97AD 10 F8              bpl     cmeb            ;Branch if not
97AF 20 7D98            jsr     rsbkpt          ;Reset the breakpoint
97B2 AD 439F            lda     bkptl           ;Get breakpointaddress
97B5 8D 039F            sta     para1
97B8 AD 449F            lda     bkpth
97BB 8D 049F            sta     parah
97BE 4C CC97            jmp     runa            ;Execute from breakpointaddress
97C1 20 1093    cmec    jsr     resin
97C4 20 9E92    run     jsr     eenpar          ;Get parameter
97C7 10 03              bpl     runa
97C9 4C 1692            jmp     illpar
97CC AD 049F    runa    lda     parah           ;Get startaddress
97CF 48                 pha
97D0 AD 039F            lda     para1
97D3 48                 pha
97D4 AD 2A9F            lda     preg            ;Also restore Proc.status
97D7 48                 pha
97D8 AE 259F            ldx     xreg            ;And X,Y regs.
97DB AC 269F            ldy     yreg
97DE AD 249F            lda     acc             ;Restore accu
97E1 40                 rti                     ;Execute program pointed by PARA
                        ;
```

```
                ;*** Place breakpoint ***
                ;
97E2 20 D393    cmb     jsr     testsp          ;Test for a space
97E5 20 8892            jsr     gchb            ;Get next char.
97E8 C9 52              cmp     #$52
97EA D0 10              bne     cmba            ;Was next character a 'R
97EC 20 8892            jsr     gchb
97EF C9 0D              cmp     #$0d
97F1 F0 03              beq     cmbaa           ;Branch if 'CR'
97F3 4C F691    cmbab   jmp     illcom
97F6 20 7D98    cmbaa   jsr     rsbkpt          ;Reset breakpoint
97F9 4C 2FC0            jmp     crlf            ;End of routine reset breakpoint
                ;
97FC C9 53      cmba    cmp     #$53
97FE D0 42              bne     cmbd            ;Was it a 'S'
9800 20 8892            jsr     gchb
9803 C9 0D              cmp     #$0d            ;Delimiter must be 'CR'
9805 D0 EC              bne     cmbab
9807 2C 429F            bit     bkptfl          ;Check if there is a breakpoint
980A D0 15              bne     cmbac           ;Branch if yes
980C 20 3BC0            jsr     prtext
980F 070D4E6F20         fcc     7,'\rNo breakpoint\r',0
9814 627265616B
9819 706F696E74
981E 0D00
9820 60                 rts
9821 20 3BC0    cmbac   jsr     prtext          ;Show breakpoint position
9824 0D42726561         fcc     '\rBreakpoint: $',0
9829 6B706F696E
982E 743A202400
9833 AE 449F            ldx     bkpth           ;Get address
9836 AC 439F            ldy     bkptl
9839 AD 459F            lda     bkptvr          ;Get data
983C 20 6596            jsr     pradda          ;Print address and data
983F 4C 2FC0            jmp     crlf            ;End routine show breakpoint
                ;
9842 CE 029F    cmbd    dec     bufpoint
9845 20 9E92            jsr     eenpar          ;Get parameter
9848 10 03              bpl     cmbc
984A 4C 1692            jmp     illpar
984D 2C 429F    cmbc    bit     bkptfl          ;Was there already a breakpoint?
9850 10 03              bpl     cmbb            ;Branch if first
9852 20 7D98            jsr     rsbkpt          ;Reset last breakpoint
9855 20 DE93    cmbb    jsr     adpntx          ;Make a ramprog.
9858 20 A493            jsr     copypar         ;Copy parameters into ramprog.
985B AD 039F            lda     para1           ;Save breakpoint address
985E 8D 439F            sta     bkptl
9861 AD 049F            lda     parah
9864 8D 449F            sta     bkpth
9867 20 169F            jsr     pntx            ;Get byte from breakpoint address
986A 8D 459F            sta     bkptvr          ;Save data
986D 20 EA93            jsr     d8pntx          ;Change ramprog.
9870 A9 00              lda     #$00
9872 20 169F            jsr     pntx            ;Place breakpoint
9875 A9 FF              lda     #$ff
9877 8D 429F            sta     bkptfl          ;Set flag for breakpoint active
987A 4C 2FC0            jmp     crlf            ;End of routine place a breakpoint
                ;
987D 20 EA93    rsbkpt  jsr     d8pntx          ;Make a ramprog.
9880 AD 439F            lda     bkptl           ;Get breakpoint address
9883 8D 179F            sta     pntxl           ;Copy it into ramprog.
9886 AD 449F            lda     bkpth
9889 8D 189F            sta     pntxh
988C A9 00              lda     #$00            ;Reset flag
988E 8D 429F            sta     bkptfl
9891 AD 459F            lda     bkptvr          ;Get old value
9894 4C 169F            jmp     pntx            ;And restore it
                ;
                ;*** Command page clear ***
                ;
9897 20 8892    cmp     jsr     gchb            ;Check for correct delimiter
989A 20 0993            jsr     schtt
989D F0 03              beq     pclear
989F 4C F691            jmp     illcom
98A2 A9 0C      pclear  lda     #$0c            ;Formfeed
98A4 4C 23C0            jmp     output          ;Print it
                ;
                ;*** Command Q, exit monitor as a subroutine ***
```

```
                        ;
98A7 20 8892    cmq     jsr     gchb            ;Check for correct delimiter
98AA 20 0993            jsr     schtt
98AD F0 03              beq     cmqa
98AF 4C F691    cmqaa   jmp     illcom
98B2 08         cmqa    php
98B3 78                 sei

98B4 AD 629F            lda     brkvs           Restore BRKVECTOR
98B7 8D B3E7            sta     brkvector
98BA AD 639F            lda     brkvs+1
98BD 8D B4E7            sta     brkvector+1

98C0 AD 649F            lda     nmivs           Restore NMIVECTOR
98C3 8D B1E7            sta     nmivector
98C6 AD 659F            lda     nmivs+1
98C9 8D B2E7            sta     nmivector+1

98CC AD 669F            lda     swivs           Restore software interrupt vector
98CF 8D 6EE7            sta     swivector
98D2 AD 679F            lda     swivs+1
98D5 8D 6FE7            sta     swivector+1

98D8 20 2FC0            jsr     crlf
98DB 28                 plp
98DC 68                 pla
98DD 68                 pla
98DE 60                 rts                     End of routine

                ;*** Routine to get decimal numbers ***
                ;
98DF A9 0D      gtcr    lda     #$0d            ;Delimiter must be 'CR'
98E1 D0 03              bne     gtspa           ;Branch always
98E3 AD 109F    gtsp    lda     delim           ;Get the valid delimiter
98E6 8D 1F9F    gtspa   sta     schei
98E9 8E 209F            stx     onderg          ;In X was the low limit
98EC 8C 219F            sty     boveng          ;In Y was the high limit
98EF 20 8D9A            jsr     dechex          ;Read numbers
98F2 F0 0D              beq     gtfout          ;Error
98F4 B0 0B              bcs     gtfout          ;Error
98F6 CE 029F            dec     bufpoint
98F9 20 8892            jsr     gchb            ;Get the delimiter
98FC CD 1F9F            cmp     schei           ;Check the delimiter
98FF F0 03              beq     gtnotf
9901 4C 1692    gtfout  jmp     illpar
9904 AD 119F    gtnotf  lda     inl             ;Get the value
9907 CD 209F            cmp     onderg          ;Check low limit
990A 90 F5              bcc     gtfout          ;Must be >= low
990C CD 219F            cmp     boveng          ;Check high limit
990F B0 F0              bcs     gtfout          ;Must be < high
9911 60                 rts
                ;
                        cont    monit18
                ;********** MONIT18 **********
                ;
                ;*** Disassembler ***
                ;
9912 20 D393    cmd     jsr     testsp          ;Test for space
9915 F0 03              beq     cmda
9917 4C F691            jmp     illcom
991A 20 E392    cmda    jsr     in              ;Get 1 parameter
991D C9 0D              cmp     #$0d
991F F0 43              beq     cmdpag          ;If only 1 parm. then disasm. per page
9921 CD 109F            cmp     delim           ;If delimiter then get next parameter
9924 F0 03              beq     cmdc
9926 4C 1692    cmdb    jmp     illpar
9929 20 EF92    cmdc    jsr     copya           ;Copy first parameter PARA
992C 20 E392            jsr     in              ;Get next parameter
992F 20 0993            jsr     schtt
9932 D0 F2              bne     cmdb            ;Branch if error
9934 20 F392            jsr     copyb           ;Copy second parameter in PARB
9937 20 8693            jsr     chck            ;PARA<PARB?
993A B0 03              bcs     cmdd            ;Branch if correct
993C 4C 7292            jmp     zpijlsp
993F 20 FE9B    cmdd    jsr     sazer           ;Save used zeropage addresses
9942 AD 039F            lda     para1           ;Copy parameters to zeropage
9945 85 04              sta     tmp0
9947 AD 049F            lda     parah
```

```
994A 85 05              sta     tmp0+$01
994C AD 059F    cmde    lda     parbl           ;On end yet?
994F C5 04              cmp     tmp0
9951 AD 069F            lda     parbh
9954 E5 05              sbc     tmp0+$01
9956 90 06              bcc     cmdf
9958 20 1E9C            jsr     disasm          ;Disassemble one instruction
995B 4C 4C99            jmp     cmde            ;Back in loop
995E 20 2FC0    cmdf    jsr     crlf
9961 4C 0E9C            jmp     reszer          ;Restore zeropage and end of routine
                ;
9964 20 FE9B    cmdpag  jsr     sazer           ;Save zeropage
9967 AD 119F            lda     inl             ;Copy parameter
996A 85 04              sta     tmp0
996C AD 129F            lda     inh
996F 85 05              sta     tmp0+$01
9971 AD 2F9F    cmdg    lda     dislct          ;Disassemble DISLCT lines
9974 8D 239F    cmdga   sta     count
9977 20 1E9C    cmdh    jsr     disasm          ;Disassemble one instruction
997A CE 239F            dec     count
997D D0 F8              bne     cmdh            ;Ready?
997F 20 20C0            jsr     input           ;Get character from input
9982 C9 0D              cmp     #$0d
9984 F0 D8              beq     cmdf            ;Branch to end in case of 'CR'
9986 C9 20              cmp     #$20
9988 D0 04              bne     cmdi            ;Branch if not a space
998A A9 01              lda     #1
998C D0 E6              bne     cmdga           ;Then continue 1 line
998E 20 2FC0    cmdi    jsr     crlf
9991 4C 7199            jmp     cmdg            ;Continue on all other keys
                ;
                ;*** Command $, convert to hex ***
                ;
9994 20 D393    cm24    jsr     testsp          ;Test for a space
9997 20 8D9A            jsr     dechex          ;Decimal to hex conversion
999A B0 26              bcs     ilparm
999C F0 16              beq     overfl          ;Branch if overflow
999E C9 20              cmp     #$20
99A0 F0 20              beq     ilparm
99A2 20 2FC0    prlhb   jsr     crlf
99A5 AD 129F            lda     inh             ;Print to hex bytes
99A8 20 38C0            jsr     prbyte
99AB AD 119F            lda     inl
99AE 20 38C0            jsr     prbyte
99B1 4C 2FC0            jmp     crlf            ;End of routine
99B4 20 3BC0    overfl  jsr     prtext
99B7 20204F7665         fcc     '  Overflow',0
99BC 72666C6F77
99C1 00
99C2 4C 1692    ilparm  jmp     illpar
                ;
                ;*** Command #, convert to decimal ***
                ;
99C5 20 D393    cm23    jsr     testsp          ;Test for a space
99C8 20 529A            jsr     gthex           ;Ascii to two hex bytes
99CB B0 F5              bcs     ilparm
99CD F0 E5              beq     overfl          ;Then overflow
99CF CE 029F            dec     bufpoint
99D2 20 8892            jsr     gchb            ;Get last typed character
99D5 C9 20              cmp     #$20
99D7 F0 E9              beq     ilparm          ;Space was incorrect
99D9 AD 129F            lda     inh             ;Get highbyte
99DC AE 119F            ldx     inl             ;Get lowbyte
99DF 20 F49A            jsr     conver          ;Convert from hex two bytes to decimal 3 bytes
99E2 20 2FC0            jsr     crlf
99E5 20 339B            jsr     pridec          ;Print the decimal numbers
99E8 4C 2FC0            jmp     crlf            ;End of routine
                ;
                ;*** Add two hex numbers ***
                ;
99EB 20 1D9A    cm2b    jsr     bgncm2          ;Get two parameters
99EE 18                 clc                     ;Add them
99EF AD 039F            lda     paral
99F2 6D 059F            adc     parbl
99F5 8D 119F            sta     inl
99F8 AD 049F            lda     parah
99FB 6D 069F            adc     parbh
99FE 8D 129F            sta     inh
```

```
9A01 4C A299           jmp     prlhb           ;Print the result
                       ;
                       ;*** Subtract two hex numbers ***
                       ;
9A04 20 1D9A   cm2d    jsr     bgncm2          ;Get two hex numbers
9A07 38                sec                     ;Subtract them
9A08 AD 039F           lda     para1
9A0B ED 059F           sbc     parb1
9A0E 8D 119F           sta     in1
9A11 AD 049F           lda     parah
9A14 ED 069F           sbc     parbh
9A17 8D 129F           sta     inh
9A1A 4C A299           jmp     prlhb           ;Print the result
                       ;
                       ;Get two parameters
                       ;
9A1D 20 D393   bgncm2  jsr     testsp          ;Test for a space
9A20 20 A992           jsr     tweepar         ;Get the parameters
9A23 10 03             bpl     bgnacm
9A25 4C 1692           jmp     illpar
9A28 60        bgnacm  rts                     ;Ready
                       ;
                       ;*** Calculate a checksum ***
                       ;
9A29 20 BD93   cms     jsr     begin           ;Get parameters
9A2C 20 DE93           jsr     adpntx          ;Make a ramprog.
9A2F 20 1093           jsr     resin           ;Reset INL and INH
9A32 18        cmsa    clc
9A33 20 169F           jsr     pntx            ;Get byte
9A36 6D 119F           adc     in1             ;Sum in INL and INH
9A39 8D 119F           sta     in1
9A3C AD 129F           lda     inh
9A3F 69 00             adc     #$00
9A41 8D 129F           sta     inh
9A44 20 9393           jsr     checkend        ;Check for ready
9A47 90 06             bcc     cmsb            ;Branch if ready
9A49 20 5F93           jsr     pntxinc         ;Increase pointer
9A4C 4C 329A           jmp     cmsa
9A4F 4C A299   cmsb    jmp     prlhb           ;Print result
                       ;
                       ;*** Ascii to hex conversion ***
                       ;
9A52 AE 029F   gthex   ldx     bufpoint        ;Get inputbufferpointer
9A55 E8                inx
9A56 8E 019F           stx     res             ;Count number of characters
9A59 20 E392           jsr     in              ;Get a character from inputbuffer
9A5C C9 0D             cmp     #$0d
9A5E F0 0B             beq     lshxa           ;Branch if delimiter
9A60 CD 109F           cmp     delim           ;Check delimiter
9A63 F0 06             beq     lshxa
9A65 C9 3A             cmp     #$3a
9A67 F0 02             beq     lshxa
9A69 38                sec                     ;Illegal character
9A6A 60                rts
9A6B 38        lshxa   sec
9A6C AD 029F           lda     bufpoint
9A6F ED 019F           sbc     res             ;4 characters is the maximum
9A72 F0 07             beq     lshxb           ;0 characters is not allowed
9A74 C9 05             cmp     #$05
9A76 B0 05             bcs     lshxc
9A78 A9 FF             lda     #$ff            ;Set flag for
9A7A 60                rts                     ;Correct exit
9A7B 38        lshxb   sec                     ;Error exit (illegal parameter)
9A7C 60                rts
9A7D A9 00     lshxc   lda     #$00            ;Error exit (overflow)
9A7F 60                rts
                       ;
                       ;*** Convert ascii byte to decimal nibble ***
                       ;
9A80 38        ckget   sec
9A81 E9 30             sbc     #$30
9A83 90 06             bcc     ckgt1           ;Branch if smaller then '0'
9A85 C9 0A             cmp     #$0a
9A87 B0 02             bcs     ckgt1           ;Branch if greater then '9'
9A89 18                clc
9A8A 60                rts                     ;Normal exit
9A8B 38        ckgt1   sec                     ;Error exit (illegal parameter)
9A8C 60                rts
```

```
                        ;
                        ;*** Convert from decimal to hex ***
                        ;
9A8D A0 00      dechex  ldy     #$00            ;Y is digit counter
9A8F 20 8892    dbt1    jsr     gchb            ;Get character
9A92 C9 0D              cmp     #$0d
9A94 F0 1A              beq     dbt3            ;Branch if delimiter
9A96 CD 109F            cmp     delim           ;Check delimiter
9A99 F0 15              beq     dbt3
9A9B C9 3A              cmp     #$3a
9A9D F0 11              beq     dbt3
9A9F 20 809A            jsr     ckget           ;Convert ascii byte to decimal nibble
9AA2 B0 07              bcs     stcker          ;Restore the stack after error
9AA4 48                 pha                     ;Decimal nibble on stack
9AA5 C8                 iny
9AA6 C0 06              cpy     #$06            ;Maximum is 5 digits
9AA8 D0 E5              bne     dbt1            ;Branch if no error
9AAA 18                 clc
9AAB 68         stcker  pla                     ;Restore the stack
9AAC 88                 dey                     ;In Y number of digits on stack
9AAD D0 FC              bne     stcker
9AAF 60         dbt2    rts                     ;Error exit
9AB0 98         dbt3    tya                     ;Delimiter found
9AB1 F0 FC              beq     dbt2            ;0 digits not allowed
9AB3 18                 clc
9AB4 8C 019F            sty     res             ;Save number of digits
9AB7 20 1093            jsr     resin           ;Reset INL and INH
9ABA A2 FF              ldx     #$ff
9ABC 8E 229F            stx     eindvlag        ;Set flag for correct exit
9ABF 68         sdh1    pla                     ;Get nibbles from stack
9AC0 AA                 tax                     ;Decimal nibble is loopcounter now
9AC1 F0 15              beq     sdh3            ;Do not loop if digit is zero
9AC3 B9 EA9A    sdh2    lda     dhtba,y         ;Get lowbyte value from table
9AC6 6D 119F            adc     inl             ;Result is in INL and INH
9AC9 8D 119F            sta     inl
9ACC B9 EF9A            lda     dhtbb,y         ;Get highbyte value from table
9ACF 6D 129F            adc     inh
9AD2 8D 129F            sta     inh
9AD5 CA                 dex
9AD6 D0 EB              bne     sdh2            ;Loop until loopcounter is 0
9AD8 C8         sdh3    iny                     ;Get next digit
9AD9 90 05              bcc     sdh4            ;Loop until no digits anymore
9ADB A9 00              lda     #0
9ADD 8D 229F            sta     eindvlag        ;More digits, then set error flag
9AE0 CC 019F    sdh4    cpy     res             ;All digits done?
9AE3 D0 DA              bne     sdh1            ;Back in loop
9AE5 AD 229F    sdh5    lda     eindvlag        ;If 0 then error, $FF is correct
9AE8 18                 clc
9AE9 60                 rts
                        ;
                        ;Table for decimal to hex conversion
                        ;
9AEA 01         dhtba   fcc     $01             ;*1
9AEB 0A                 fcc     $0a             ;*10
9AEC 64                 fcc     $64             ;*100
9AED E8                 fcc     $e8             ;*1000
9AEE 10                 fcc     $10             ;*10000
9AEF 00         dhtbb   fcc     $00             ;*1
9AF0 00                 fcc     $00             ;*10
9AF1 00                 fcc     $00             ;*100
9AF2 03                 fcc     $03             ;*1000
9AF3 27                 fcc     $27             ;*10000

                        ;*** Convert from 2 hex bytes to 3 decimal bytes ***

9AF4 8E 2B9F    conver  stx     deci1           Lowbyte
9AF7 8D 2C9F            sta     deci2           Highbyte
9AFA A9 00              lda     #0
9AFC 8D 2D9F            sta     deci3
9AFF AA                 tax
9B00 F8                 sed
9B01 F0 1B              beq     hdf             Branch always
9B03 CE 2C9F    hda     dec     deci2
9B06 8A         .       txa
9B07 69 56              adc     #$56            Add 256
9B09 AA                 tax
9B0A 68                 pla
9B0B 69 02              adc     #$02
```

```
9B0D 90 0F              bcc     hdf
9B0F EE 2D9F            inc     deci3
9B12 D0 0A              bne     hdf
9B14 CE 2B9F    hdb     dec     deci1
9B17 8A                 txa
9B18 69 01              adc     #1
9B1A AA                 tax
9B1B 68                 pla
9B1C 69 00              adc     #0
9B1E 48         hdf     pha
9B1F 18                 clc
9B20 AD 2B9F            lda     deci1
9B23 D0 EF              bne     hdb
9B25 AD 2C9F            lda     deci2           End test
9B28 D0 D9              bne     hda
9B2A D8                 cld
9B2B 68                 pla
9B2C 8D 2C9F            sta     deci2
9B2F 8E 2B9F            stx     deci1
9B32 60                 rts

                ;*** Print decimal numbers ***

9B33 AD 2D9F    pridec  lda     deci3
9B36 D0 0B              bne     conoe           Branch if no leading zero
9B38 AD 2C9F            lda     deci2
9B3B D0 15              bne     conof           Branch if no leading zero
9B3D AD 2B9F            lda     deci1
9B40 4C 35C0            jmp     hnout           Print least sign. byte
9B43 20 35C0    conoe   jsr     hnout           Print byte 3
9B46 AD 2C9F            lda     deci2
9B49 20 38C0            jsr     prbyte          Print byte 2
9B4C AD 2B9F    conog   lda     deci1
9B4F 4C 38C0            jmp     prbyte          Print byte 1
9B52 20 35C0    conof   jsr     hnout
9B55 4C 4C9B            jmp     conog

                ;*** Command W, search a string ***

9B58 20 B296    cmw     jsr     twpaby          ;Get two parameters
9B5B 20 8892            jsr     gchb            ;Get next character
9B5E C9 27              cmp     #$27
9B60 D0 5B              bne     cmwh            ;Branch if no ascii string
9B62 A0 00              ldy     #$00            ;Reset counter for stringlength
9B64 20 9492    cmwd    jsr     gchbl           ;Get next character also lower case
9B67 20 0993            jsr     schtt           ;Check for delimiter
9B6A F0 0B              beq     cmwe            ;Branch if delimiter
9B6C 99 319F            sta     whbuf,y         ;Put characters in searchbuffer
9B6F C8                 iny                     ;Update stringlength
9B70 C0 11              cpy     #17             ;Maximum is 16
9B72 D0 F0              bne     cmwd
9B74 4C 1692    cmwaa   jmp     illpar
9B77 8C 309F    cmwe    sty     nmb             ;Save stringlength
9B7A 98                 tya
9B7B F0 F7              beq     cmwaa           ;0 characters not allowed
9B7D 20 2FC0            jsr     crlf
9B80 20 E293            jsr     bdpntx          ;Make ramprog.
9B83 20 A493            jsr     copypar         ;Copy parameters into ramprog.
9B86 A2 00              ldx     #$00
9B88 20 9393    cmwf    jsr     checkend        ;Ready?
9B8B B0 01              bcs     cmwfaa
9B8D 60                 rts                     ;End of routine
9B8E 20 169F    cmwfaa  jsr     pntx            ;Get character from selected area
9B91 DD 319F            cmp     whbuf,x         ;Compare with buffer
9B94 F0 0F              beq     cmwg
9B96 AD 469F            lda     dntcr           ;Don't care character
9B99 DD 319F            cmp     whbuf,x         ;Compare with buffer
9B9C F0 07              beq     cmwg
9B9E 20 5F93    cmwfa   jsr     pntxinc         ;Increase pointer
9BA1 A2 00              ldx     #$00
9BA3 F0 E3              beq     cmwf            ;Branch always back in loop
9BA5 E8         cmwg    inx                     ;Then character the same
9BA6 EC 309F            cpx     nmb             ;Do all characters match?
9BA9 D0 DD              bne     cmwf            ;Branch if not
9BAB AD 189F            lda     pntxh           ;String found!!, Print address
9BAE 20 38C0            jsr     prbyte          ;Print highbyte
9BB1 AD 179F            lda     pntxl
9BB4 20 38C0            jsr     prbyte          ;Print lowbyte
```

```
9BB7 20 2FC0          jsr     crlf
9BBA 4C 9E9B          jmp     cmwfa
                      ;
9BBD A0 00      cmwh  ldy     #$00            ;Input is hex
9BBF CE 029F          dec     bufpoint
9BC2 98         cmwi  tya                     ;Y is byte counter
9BC3 48               pha
9BC4 20 8B92          jsr     gchb            ;Get next character
9BC7 20 0993          jsr     schtt           ;Check for delimiter
9BCA F0 23            beq     cmwl            ;Branch if delimiter
9BCC 20 1093          jsr     resin           ;Reset INL and INH
9BCF 20 1993          jsr     hexnum          ;Convert input ascii to hex nibble
9BD2 F0 03            beq     cmwk
9BD4 4C 1692    cmwj  jmp     illpar
9BD7 20 8B92    cmwk  jsr     gchb            ;Get next character
9BDA 20 1993          jsr     hexnum          ;Convert input ascii to hex nibble
9BDD D0 F5            bne     cmwj
9BDF 68               pla                     ;Get bytenumber
9BE0 A8               tay
9BE1 AD 119F          lda     inl             ;Get hex byte
9BE4 99 319F          sta     whbuf,y         ;Put in searchbuffer
9BE7 C8               iny
9BE8 C0 11            cpy     #17             ;Maximum is 16 bytes
9BEA D0 D6            bne     cmwi            ;Back in loop if not on maximum
9BEC 4C 1692          jmp     illpar
9BEF 68         cmwl  pla                     ;Get number of bytes
9BF0 A8               tay
9BF1 4C 779B          jmp     cmwe            ;Continue elsewhere
                      ;
                      cont    monit19
                      ;*********** MONIT19 ***********
                      ;
                      ;*** Disassembler ***
                      ;
9BF4 A5 05      wroa  lda     tmp0+$01        ;Print current address
9BF6 20 38C0          jsr     prbyte
9BF9 A5 04            lda     tmp0
9BFB 4C 38C0          jmp     prbyte
                      ;
9BFE A5 04      sazer lda     tmp0            ;Save used zeropage addresses
9C00 8D 0A9F          sta     stmp0
9C03 A5 05            lda     tmp0+$01
9C05 8D 0B9F          sta     stmp0+$01
9C08 A9 FF            lda     #$ff            ;Set flag disassembler active
9C0A 8D 419F          sta     disflg
9C0D 60               rts
                      ;
9C0E AD 0A9F    reszer lda    stmp0           ;Restore zeropage addresses
9C11 85 04            sta     tmp0
9C13 AD 0B9F          lda     stmp0+$01
9C16 85 05            sta     tmp0+$01
9C18 A9 00            lda     #$00
9C1A 8D 419F          sta     disflg          ;Reset disassembler active flag
9C1D 60               rts
                      ;
                      ;*** Disassemble one instruction ***
                      ;
9C1E 20 309C    disasm jsr    dis             ;Disassemble
9C21 38               sec                     ;Calculate next opcode address
9C22 A5 04            lda     tmp0
9C24 6D 0C9F          adc     inslen
9C27 85 04            sta     tmp0
9C29 A5 05            lda     tmp0+$01
9C2B 69 00            adc     #$00
9C2D 85 05            sta     tmp0+$01
9C2F 60               rts
                      ;
                      ;*** Disassembler  65C02, Wozniak, Baum, De Vries ***
                      ;
9C30 20 2FC0    dis   jsr     crlf
9C33 20 F49B          jsr     wroa            ;Print address of instruction
9C36 A2 02            ldx     #2
9C38 20 5893          jsr     xspace          ;Print two spaces
9C3B A2 00            ldx     #$00            ;Get offset
9C3D A1 04            lda     [tmp0,x]        ;Get opcode
9C3F 20 B39C          jsr     indexlen        ;Determine index and instruction length
9C42 48               pha                     ;Save index
9C43 20 279D          jsr     prthex          ;Print instruction in hex
```

```
9C46 68              pla                    ;Get index back
9C47 48              pha                    ;Save it again
9C48 20 539D         jsr     prtmnem        ;Print mnemonic
9C4B 68              pla                    ;Get index back
9C4C 10 17           bpl     operpr         ;If positive, print operand
9C4E 29 02           and     #$02           ;If one operand
9C50 F0 13           beq     operpr         ;Print that
9C52 20 659C         jsr     operpr         ;Else print first operand
9C55 A9 2C           lda     #',            ;Followed by a comma
9C57 20 23C0         jsr     output
9C5A E6 04           inc     tmp0           ;Adapt 'PC'
9C5C D0 02           bne     npc1           ;If page crossed
9C5E E6 05           inc     tmp0+1         ;Adapt high also
9C60 A9 9D     npc1  lda     #$9d           ;Get opcode for relative
9C62 8D 0F9F         sta     tmp6           ;In addressmode code
                     ;
                     ;*** Print the operand of an instruction ***
                     ;
9C65 A2 06     operpr ldx    #$06           ;Get offset
9C67 E0 03     operlp cpx    #$03           ;If three now
9C69 D0 14           bne     chrtst         ;Print address
9C6B AC 0C9F         ldy     inslen         ;Get length
9C6E F0 0F           beq     chrtst         ;If zero, print no hex byte
9C70 AD 0F9F   adrlp  lda    tmp6           ;Else get mode
9C73 C9 E8           cmp     #$e8           ;Check for relative
9C75 B1 04           lda     [tmp0],y       ;Get operand byte
9C77 B0 1D           bcs     reladout       ;If relative, print address and exit
9C79 20 38C0         jsr     prbyte         ;Else print high
9C7C 88              dey                    ;Point to low
9C7D D0 F1           bne     adrlp          ;And print that also
9C7F 0E 0F9F   chrtst asl    tmp6           ;Get next bit
9C82 90 0E           bcc     invchr         ;If set,
9C84 BD AF9E         lda     chrtab1-1,x    ;Get next character
9C87 20 23C0         jsr     output         ;Print it
9C8A BD B59E         lda     chrtab2-1,x    ;Get second character
9C8D F0 03           beq     invchr         ;If zero, loop
9C8F 20 23C0         jsr     output         ;Else print it
9C92 CA        invchr dex                   ;Adapt counter
9C93 D0 D2           bne     operlp         ;If not all done, loop
9C95 60              rts                    ;Else exit
                     ;
                     ;*** Print relative offset as an absolute address ***
                     ;
9C96 20 A79C   reladout jsr  addoffs        ;Calculate PC+offset+carry
9C99 AA              tax                    ;Save the lowbyte in X
9C9A E8              inx                    ;Adapt it (now PC+offset+1+carry)
9C9B D0 01           bne     npc2           ;If page crossed
9C9D C8              iny                    ;Adapt high
9C9E 98        npc2  tya                    ;Move highbyte to accu
9C9F 20 38C0         jsr     prbyte         ;Print it in hex
9CA2 8A              txa                    ;Get lowbyte in accu
9CA3 4C 38C0         jmp     prbyte         ;Print it in hex and exit
                     ;
                     ;*** Calculate offset+current address in YA ***
                     ;
9CA6 38        addoffc sec                  ;Entry to use if carry clear
9CA7 A4 05     addoffs ldy   tmp0+$01       ;Get highbyte
9CA9 AA              tax                    ;Set flags according to A
9CAA 10 01           bpl     r11            ;If negative
9CAC 88              dey                    ;Adapt high
9CAD 65 04     r11   adc     tmp0           ;Add current low
9CAF 90 01           bcc     r12            ;If page crossed
9CB1 C8              iny                    ;Adapt high
9CB2 60        r12   rts                    ;And exit
                     ;
                     ;*** Calculate index, instructionlength and addressmodecode ***
                     ;
9CB3 A8        indexlen tay                 ;Save opcode in Y
9CB4 4A              lsra                   ;Get LSB in carry
9CB5 90 0F           bcc     evenop         ;Branch if opcode even
9CB7 4A              lsra                   ;Else get bit 1 in carry
9CB8 90 08           bcc     oddop          ;If opcode ends on 11
9CBA 4A              lsra                   ;Get next bit
9CBB 90 18           bcc     illop          ;If clear, illegal instruction
9CBD AE 2E9F         ldx     c02typ         ;1 is Rockwell, 0 is Synertek/GTE
9CC0 B0 06           bcs     rockwl         ;Get opcode
9CC2 29 07     oddop and     #$07           ;Else opcode ends on 01
9CC4 09 80           ora     #$80           ;Add tableoffset
```

```
9CC6 4A        evenop  lsra                    ;Get next bit in carry
9CC7 AA                tax                     ;Move offset to X
9CC8 BD 919D   rockw1  lda     adrmod,x        ;Get addressmodecode
9CCB B0 04             bcs     bit2on          ;If bit 2 was
9CCD 4A                lsra                    ;Off, move
9CCE 4A                lsra                    ;High nibble
9CCF 4A                lsra                    ;to
9CD0 4A                lsra                    ;Low nibble
9CD1 29 0F     bit2on  and     #$0f            ;Get low nibble
9CD3 D0 04             bne     validop         ;If zero
9CD5 A0 33     illop   ldy     #$33            ;Get dummy opcode
9CD7 A9 00             lda     #$00            ;And dummy addressmodecode
9CD9 AA        validop tax                     ;Get index in lengthtable
9CDA BD D89D           lda     lengthtab,x     ;Get length and operandcode
9CDD 8D 0F9F           sta     tmp6            ;In TMP6
9CE0 29 03             and     #$03            ;Get instructionlength minus 1
9CE2 8D 0C9F           sta     inslen          ;Save it
9CE5 98                tya                     ;Restore opcode
9CE6 C9 9E             cmp     #$9e            ;If STZ ABS,X
9CE8 D0 02             bne     decode
9CEA A9 9C             lda     #$9c            ;Get code for STZ ABS
9CEC 4A        decode  lsra                    ;Get LSB in carry
9CED 90 1C             bcc     evenop2         ;If clear do even opcode
9CEF 4A        odd     lsra                    ;Else opcode is odd
9CF0 90 0D             bcc     oddop2          ;If opcode ends with 11
9CF2 4A                lsra                    ;Get next bit
9CF3 90 2D             bcc     illop2          ;If clear, illegal
9CF5 4A                lsra                    ;Get bit 2 in carry
9CF6 98                tya                     ;Get opcode back
9CF7 2A                rola                    ;Move MSB to carry, C into LSB
9CF8 2A                rola                    ;Convert to
9CF9 29 03             and     #$03            ;Opcode index
9CFB 09 E0             ora     #$e0            ;Add table offset
9CFD D0 25             bne     exit            ;And exit
9CFF C9 22     oddop2  cmp     #$22            ;If opcode is BIT#
9D01 D0 04             bne     notbit
9D03 A9 09             lda     #$09            ;Get index to BIT
9D05 D0 1D             bne     exit            ;And exit
9D07 4A        notbit  lsra                    ;Else get 3 MSB's
9D08 38                sec                     ;Add tableoffset
9D09 B0 0C             bcs     oddop3          ;And calculate offset
9D0B 4A        evenop2 lsra                    ;If opcode ended with 00
9D0C 90 16             bcc     exit            ;Exit (index ready now)
9D0E 4A                lsra                    ;Else end was 10
9D0F B0 04             bcs     x6xe            ;If end was 110, do X6XE
9D11 C9 14             cmp     #$14            ;Else opcode is LDX #?
9D13 D0 06             bne     x2xa            ;If not, do other X2XA
9D15 09 20     x6xe    ora     #$20            ;Else add table offset
9D17 6A        oddop3  ror a                   ;Opcode is X6 or XE
9D18 4A        x2      lsra                    ;Calculate table offset
9D19 D0 09             bne     exit            ;And exit
9D1B 38        x2xa    sec                     ;Opcode is X2 or XA
9D1C 6A                ror a                   ;Shift in table offset
9D1D 90 F9             bcc     x2              ;If X2, get offset and exit
9D1F 49 D0             eor     #$d0            ;Opcode was XA, convert to offset
9D21 2C                fcc     $2c             ;Skip next instruction
9D22 A9 11     illop2  lda     #$11            ;
9D24 A0 00     exit    ldy     #$00            ;Get offset zero (point to opcode)
9D26 60                rts                     ;and exit
                       ;
                       ;*** Subroutine PRTHEX ***
                       ;
9D27 A2 14     prthex  ldx     #20             ;20 columns object field
9D29 48                pha                     ;Save index
9D2A 29 FF             and     #$ff            ;Set flags according to A
9D2C 10 07             bpl     norock1         ;If positive proceed
9D2E 29 02             and     #$02            ;If XF opcode
9D30 F0 03             beq     norock1         ;
9D32 EE 0C9F           inc     inslen          ;Adapt length
9D35 B1 04     norock1 lda     [tmp0],y        ;Get instruction byte
9D37 20 38C0           jsr     prbyte          ;Print it
9D3A 20 32C0           jsr     spa             ;Print a space
9D3D CC 0C9F           cpy     inslen          ;Set carry if last byte
9D40 CA                dex
9D41 CA                dex                     ;Minus 1 byte and 1 space
9D42 CA                dex
9D43 C8                iny                     ;Point to next byte
9D44 90 EF             bcc     norock1         ;If not all done, print that
```

```
9D46 68              pla                ;Get index back
9D47 10 07           bpl    norock2     ;If positive, exit
9D49 29 02           and    #$02        ;If XF code
9D4B F0 03           beq    norock2
9D4D CE 0C9F         dec    inslen      ;Adapt instruction length
9D50 4C 5893 norock2 jmp    xspace      ;Print X spaces
                     ;
                     ;**** Subroutine PRTMNEM ***
                     ;
9D53 48      prtmnem pha                ;Save index
9D54 29 7F           and    #$7f        ;Remove MSB
9D56 A8              tay                ;Get real index in Y
9D57 B9 E89D         lda    frstlttr,y  ;Get encode first and half second letter
9D5A 8D 0D9F         sta    tmp4
9D5D B9 4C9E         lda    thrdlttr,y  ;Get half second and third letter
9D60 8D 0E9F         sta    tmp4+$01
9D63 A2 03           ldx    #$03        ;Get letter counter
9D65 A9 00   lp2     lda    #$00        ;Clear result area
9D67 A0 05           ldy    #$05        ;5 bits to do
9D69 0E 0E9F lp1     asl    tmp4+$01    ;Get -
9D6C 2E 0D9F         rol    tmp4        ;Next -
9D6F 2A              rola               ;Bit in accu
9D70 88              dey                ;Until 5 bits -
9D71 D0 F6           bne    lp1         ;Done
9D73 69 3F           adc    #$3f        ;Then convert to ascii
9D75 20 23C0         jsr    output      ;Print it
9D78 CA              dex                ;All letters done?
9D79 D0 EA           bne    lp2         ;If not, repeat
9D7B 68              pla                ;Else get index back
9D7C 10 0E           bpl    norock3     ;If MSB set -
9D7E B1 04           lda    [tmp0],y    ;Get opcode
9D80 4A              lsra               ;Move bitnumber into place
9D81 4A              lsra
9D82 4A              lsra
9D83 4A              lsra
9D84 29 07           and    #$07        ;Get lowest 3 bits
9D86 09 30           ora    #$30        ;Convert to ascii
9D88 20 23C0         jsr    output      ;Print bitnumber
9D8B 24              fcc    $24         ;Skip next instruction
9D8C E8      norock3 inx                ;Get counter
9D8D E8              inx
9D8E 4C 5893 lp3     jmp    xspace      ;Print spaces
                     ;
                     cont   monit20
                     ;*********** MONIT20 ***********
                     ;
                     ;*** Addressingmodetable ***
                     ;
                     ;Each nibble represents an addressmode
                     ;
                     ; 0 = Illegal opcode
                     ; 1 = Immediate
                     ; 2 = Zero page
                     ; 3 = Absolute
                     ; 4 = Illegal opcode
                     ; 5 = Accumulator or implied
                     ; 6 = Indexed indirect
                     ; 7 = Indirect indexed
                     ; 8 = Zero page
                     ; 9 = Absolute,X
                     ; A = Absolute,Y
                     ; B = Indirect
                     ; C = Zero page,Y
                     ; D = Relative
                     ; E = Absolute Indexed Indirect
                     ; F = Indirect zero page
                     ;
9D91 40224533DF adrmod  fcc    $40,$22,$45,$33,$df,$28,$45,$39
9D96 284539
9D99 30224533DF         fcc    $30,$22,$45,$33,$df,$88,$45,$99
9D9E 884599
9DA1 40024533DF         fcc    $40,$02,$45,$33,$df,$08,$45,$09
9DA6 084509
9DA9 402245B3DF         fcc    $40,$22,$45,$b3,$df,$88,$45,$e9
9DAE 8845E9
9DB1 D0224433DF         fcc    $d0,$22,$44,$33,$df,$8c,$44,$39
9DB6 8C4439
9DB9 11224433DF         fcc    $11,$22,$44,$33,$df,$8c,$44,$9a
```

```
9DBE 8C449A
9DC1 1022443 3DF        fcc     $10,$22,$44,$33,$df,$08,$45,$09
9DC6 084509
9DC9 1022443 3DF        fcc     $10,$22,$44,$33,$df,$08,$45,$09
9DCE 084509
9DD1 621378A900         fcc     $62,$13,$78,$a9,$00,$01,$00
9DD6 0100
                        ;
                        ;*** Opcode length table ***
                        ;
                        ;Lowest to bits represents instructionlength minus 1
                        ;
9DD8 00         lengthtab fcc   $00             ;Illegal opcode
9DD9 21                 fcc     $21             ;Zero page
9DDA 81                 fcc     $81             ;Immediate
9DDB 82                 fcc     $82             ;Absolute
9DDC 00                 fcc     $00             ;Illegal opcode
9DDD 00                 fcc     $00             ;Accumulator of implied
9DDE 59                 fcc     $59             ;Indexed indirect ($..,X)
9DDF 4D                 fcc     $4d             ;Indirect indexed ($..),Y
9DE0 91                 fcc     $91             ;Zero page,X
9DE1 92                 fcc     $92             ;Absolute,X
9DE2 86                 fcc     $86             ;Absolute,Y
9DE3 4A                 fcc     $4a             ;Indirect ($....)
9DE4 85                 fcc     $85             ;Zeropage,Y
9DE5 9D                 fcc     $9d             ;Relative
9DE6 5A                 fcc     $5a             ;Absolute indexed indirect ($....,X)
9DE7 49                 fcc     $49             ;Indirect zeropage ($..)
                        ;
                        ;*** Mnemonic table ***
                        ;
                        ;Each mnemonic is represented by two bytes.
                        ;Which contain three offsets to ascii.
                        ;Each offset is 5 bits long.
                        ;When the 5 bit offset is added to 3F, a letter
                        ;or questionmark is obtained.
                        ;
9DE8            frstlttr
                        ;               BRK TSB PHP TSB BPL TRB CLC TRB
9DE8 1CAD8AAD1C         fcc     $1c,$ad,$8a,$ad,$1c,$ac,$23,$ac
9DED AC23AC
                        ;               JSR BIT PLP BIT BMI BIT SEC BIT
9DF0 5D1A8B1A1B         fcc     $5d,$1a,$8b,$1a,$1b,$1a,$a1,$1a
9DF5 1AA11A
                        ;               RTI ??? PHA JMP BVC ??? CLI ???
9DF8 9D008A5B1D         fcc     $9d,$00,$8a,$5b,$1d,$00,$23,$00
9DFD 002300
                        ;               RTS STZ PLA JMP BVS STZ SEI JMP
9E00 9DA58B5B1D         fcc     $9d,$a5,$8b,$5b,$1d,$a5,$a1,$5b
9E05 A5A15B
                        ;               BRA STY DEY STY BCC STY TYA STZ
9E08 1CA529A519         fcc     $1c,$a5,$29,$a5,$19,$a5,$ae,$a5
9E0D A5AEA5
                        ;               LDY LDY TAY LDY BCS LDY CLV LDY
9E10 6969A86919         fcc     $69,$69,$a8,$69,$19,$69,$23,$69
9E15 692369
                        ;               CPY CPY INY CPY BNE ??? CLD ???
9E18 242453241B         fcc     $24,$24,$53,$24,$1b,$00,$23,$00
9E1D 002300
                        ;               CPX CPX INX CPX BEQ ??? SED ???
9E20 2424532419         fcc     $24,$24,$53,$24,$19,$00,$a1,$00
9E25 00A100
                        ;               ORA AND EOR ADC STA LDA CMP SBC
9E28 84133411A5         fcc     $84,$13,$34,$11,$a5,$69,$23,$a0
9E2D 6923A0
                        ;               ASL ROL LSR ROR STX LDX DEC INC
9E30 159C6D9CA5         fcc     $15,$9c,$6d,$9c,$a5,$69,$29,$53
9E35 692953
                        ;               ASL INA ROL DEA LSR PHY ROR PLY
9E38 15539C296D         fcc     $15,$53,$9c,$29,$6d,$8a,$9c,$8b
9E3D 8A9C8B
                        ;               TXA TXS TAX TSX DEX PHX NOP PLX
9E40 AEAEA8AD29         fcc     $ae,$ae,$a8,$ad,$29,$8a,$7c,$8b
9E45 8A7C8B
                        ;               RMB SMB BBR BBS
9E48 9BA31818           fcc     $9b,$a3,$18,$18
9E4C            thrdlttr
                        ;               BRK TSB PHP TSB BPL TRB CLC TRB
```

```
9E4C D80662065A         fcc     $d8,$06,$62,$06,$5a,$c6,$48,$c6
9E51 C648C6
                        ;       JSR BIT PLP BIT BMI BIT SEC BIT
9E54 26AA62AA94         fcc     $26,$aa,$62,$aa,$94,$aa,$88,$aa
9E59 AA88AA
                        ;       RTI ??? PHA JMP BVC ??? CLI ???
9E5C 540044A2C8         fcc     $54,$00,$44,$a2,$c8,$00,$54,$00
9E61 005400
                        ;       RTS STZ PLA JMP BVS STZ SEI JMP
9E64 687644A2E8         fcc     $68,$76,$44,$a2,$e8,$76,$94,$a2
9E69 7694A2
                        ;       BRA STY DEY STY BCC STY TYA STZ
9E6C C474B47408         fcc     $c4,$74,$b4,$74,$08,$74,$84,$76
9E71 748476
                        ;       LDY LDY TAY LDY BCS LDY CLV LDY
9E74 7474B47428         fcc     $74,$74,$b4,$74,$28,$74,$6e,$74
9E79 746E74
                        ;       CPY CPY INY CPY BNE ??? CLD ???
9E7C 7474F474CC         fcc     $74,$74,$f4,$74,$cc,$00,$4a,$00
9E81 004A00
                        ;       CPX CPX INX CPX BEQ ??? SED ???
9E84 7272F272A4         fcc     $72,$72,$f2,$72,$a4,$00,$8a,$00
9E89 008A00
                        ;       ORA AND EOR ADC STA LDA CMP SBC
9E8C C4CA264844         fcc     $c4,$ca,$26,$48,$44,$44,$a2,$c8
9E91 44A2C8
                        ;       ASL ROL LSR ROR STX LDX DEC INC
9E94 1A1A262672         fcc     $1a,$1a,$26,$26,$72,$72,$88,$c8
9E99 7288C8
                        ;       ASL INA ROL DEA LSR PHY ROR PLY
9E9C 1AC41A8426         fcc     $1a,$c4,$1a,$84,$26,$74,$26,$74
9EA1 742674
                        ;       TXA TXS TAX TSX DEX PHX NOP PLX
9EA4 4468B232B2         fcc     $44,$68,$b2,$32,$b2,$72,$22,$72
9EA9 722272
                        ;       RMB SMB BBR BBS
9EAC 8686E6E8           fcc     $86,$86,$e6,$e8

                ;*** Character table for operand printout ***
                ;
                ;Table is used backwards, starting with offset X=6
                ;
                ;X=6: $     (before hex oper.)
                ;X=5: ($    (before hex oper.)
                ;X=4: #$    (before hex oper.)
                ;X=3: ,X    (after hex oper.)
                ;X=2: )     (after hex oper.)
                ;X=1: ,Y    (after hex oper.)
                ;X=0:       not used
                ;
                ;
9EB0 2C292C2328 chrtab1 fcc     ',),#($'        ;First character of a pair
9EB5 24
9EB6 5900582424 chrtab2 fcc     'Y',0,'X$$',0   ;Second character
9EBB 00
                ;
                        cont    monit21
                ;********** MONIT21 **********

                ;Start variable aria
          9F00          org     monbeg+$f00
9F00                    v
9F00            savstackp res   1               ;Software stackpointer
9F01            res     res     1               ;Temporary arithmetic result
9F02            bufpoint res    1               ;Inputbuffer pointer
9F03            paral   res     1               ;Parameter A
9F04            parah   res     1
9F05            parbl   res     1               ;Parameter B
9F06            parbh   res     1
9F07            parcl   res     1               ;Parameter C
9F08            parch   res     1
9F09            aapgew  res     1               ;Changed something-flag for @ command
9F0A            stmp0   res     2               ;Variables for disassembler
9F0C            inslen  res     1               ;Instruction length
9F0D            tmp4    res     2
9F0F            tmp6    res     1
9F10            delim   res     1               ;Kind of delimiter between parameters
9F11            inl     res     1               ;Lowbyte input parameter
```

```
9F12            inh     res     1               ;Highbyte input parameter
9F13            svstck  res     1               ;Stackpointer saveaddress in mainprogram
9F14            pntxlsave res   1               ;Saveaddress for PNTXL
9F15            pntxhsave res   1               ;Saveaddress for PNTXH
9F16            pntx    res     1               ;Program in ram, self modifying
9F17            pntxl   res     1
9F18            pntxh   res     2               ;Also one byte for RTS
9F1A            pnty    res     1               ;Program in ram, self modifying
9F1B            pntyl   res     1
9F1C            pntyh   res     2               ;Also one byte for RTS
9F1E            prbffg  res     1               ;'Print inputbuffer flag' after error
9F1F            schei   res     1               ;Kind of delimiter
9F20            onderg  res     1               ;Low border for get decimal
9F21            boveng  res     1               ;High border for get decimal
9F22            eindvlag res    1               ;End of hexdump, errorflag in arithmatics
9F23            count   res     1               ;Count number of lines in disasm.
9F24            acc     res     1               ;A register
9F25            xreg    res     1               ;X register
9F26            yreg    res     1               ;Y register
9F27            pcl     res     1               ;Program counter low register
9F28            pch     res     1               ;Program counter high register
9F29            spuser  res     1               ;Stack pointer register
9F2A            preg    res     1               ;Processor status
9F2B            deci1   res     1               ;Variables for calculate decimal
9F2C            deci2   res     1
9F2D            deci3   res     1
9F2E            c02typ  res     1               ;Select kind of 65C02
9F2F            dislct  res     1               ;Number of lines to disassemble
9F30            nmb     res     1               ;Number of characters to look for
9F31            whbuf   res     16              ;'Look for' buffer
9F41            disflg  res     1               ;Flag if BREAK was in disassembler
9F42            bkptfl  res     1               ;Breakpoint set flag
9F43            bkptl   res     1               ;Address breakpoint lowbyte
9F44            bkpth   res     1               ;Address breakpoint highbyte
9F45            bkptvr  res     1               ;Data from breakpointaddress
9F46            dntcr   res     1               ;Wild character
9F47            ndvrs
9F47            prcnt   res     1               ;Number of prompt characters

        9F50            org     v+$50

9F50 4D4F4E3E20 prompt  fcc     'MON> ',0
9F55 00
9F56            reserve res     11
9F61            qcomvr  res     1               ;Variable for monitor as subroutine
9F62            brkvs   res     2               ;Saveaddress BREAK vector
9F64            nmivs   res     2               ;Saveaddress NMI vector
9F66            swivs   res     2               ;Saveaddress software interrupt vector
9F68            tabvec  res     2               ;End of cmd.interpr.-routine vec., 2 bytes

9F6A            userx   res     2               ;Vector for user def. command X
9F6C            usery   res     2               ;Vector for user def. command Y

                ;*** Buffer aria ***

9F6E            inpbuffer res   80              ;Input/command buffer, 80 char.

        9000            end     mon65
```

label table

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| aapgew | 9F09 | acc | 9F24 | addoffc | 9CA6 | addoffs | 9CA7 | adpntx | 93DE |
| adpnty | 93F5 | adrlp | 9C70 | adrmod | 9D91 | ashett | 9335 | bdpntx | 93E2 |
| beg3 | 95FE | beg3a | 9609 | beg3b | 9614 | begin | 93BD | beginb | 93C8 |
| beginc | 93D0 | bgnacm | 9A28 | bgncm2 | 9A1D | bit2on | 9CD1 | bkptfl | 9F42 |
| bkpth | 9F44 | bkptl | 9F43 | bkptvr | 9F45 | boveng | 9F21 | break | 9088 |
| brkvector | E7B3 | brkvs | 9F62 | btt | 9092 | bufloop | 912B | bufpoint | 9F02 |
| c02typ | 9F2E | cdpntx | 93E6 | chck | 9386 | check | 9395 | checkend | 9393 |
| chrtab1 | 9EB0 | chrtab2 | 9EB6 | chrtst | 9C7F | ckget | 9A80 | ckgt1 | 9A8B |
| cm. | 9725 | cm23 | 99C5 | cm24 | 9994 | cm2b | 99EB | cm2d | 9A04 |
| cmaap | 94C8 | cmb | 97E2 | cmba | 97FC | cmbaa | 97F6 | cmbab | 97F3 |
| cmbac | 9821 | cmbb | 9855 | cmbc | 984D | cmbd | 9842 | cmc | 96D7 |
| cmcb | 9708 | cmcc | 96F2 | cmcd | 96FA | cmce | 9702 | cmcf | 96ED |
| cmcg | 9714 | cmd | 9912 | cmda | 991A | cmdb | 9926 | cmdc | 9929 |
| cmdd | 993F | cmde | 994C | cmdf | 995E | cmdg | 9971 | cmdga | 9974 |
| cmdh | 9977 | cmdi | 998E | cmdpag | 9964 | cme | 979C | cmea | 97AA |
| cmeb | 97A7 | cmec | 97C1 | cmf | 9672 | cmh | 9404 | cml | 972F |
| cmla | 973A | cmm | 9580 | cmp | 9897 | cmq | 98A7 | cmqa | 98B2 |
| cmqaa | 98AF | cms | 9A29 | cmsa | 9A32 | cmsb | 9A4F | cmv | 9615 |
| cmw | 9B58 | cmwaa | 9B74 | cmwd | 9B64 | cmwe | 9B77 | cmwf | 9B88 |
| cmwfa | 9B9E | cmwfaa | 9B8E | cmwg | 9BA5 | cmwh | 9BBD | cmwi | 9BC2 |
| cmwj | 9BD4 | cmwk | 9BD7 | cmwl | 9BEF | cmx | 91F0 | cmy | 91F3 |
| commtabel | 91C6 | comtab | 91B1 | conoe | 9B43 | conof | 9B52 | conog | 9B4C |
| conver | 9AF4 | cony | 9170 | copy | 92F9 | copya | 92EF | copyb | 92F3 |
| copyc | 92F7 | copypar | 93A4 | count | 9F23 | cr | 9130 | crlf | C02F |
| cruit | 9134 | d8pntx | 93EA | d8pnty | 93F9 | dbt1 | 9A8F | dbt2 | 9AAF |
| dbt3 | 9AB0 | dechex | 9A8D | deci1 | 9F2B | deci2 | 9F2C | deci3 | 9F2D |
| decode | 9CEC | delet | 9148 | delim | 9F10 | dhtba | 9AEA | dhtbb | 9AEF |
| dis | 9C30 | disasm | 9C1E | disflg | 9F41 | dislct | 9F2F | dntcr | 9F46 |
| driepar | 92BF | eenpar | 929E | eindvlag | 9F22 | error1 | 919F | evenop | 9CC6 |
| evenop2 | 9D0B | exit | 9D24 | filla | 9691 | fillaa | 9690 | fillb | 96A0 |
| fillend | 96AF | fillerr | 96AC | fillf | 9688 | frstlttr | 9DE8 | gchb | 9288 |
| gchb1 | 9294 | gtcr | 98DF | gtfout | 9901 | gthex | 9A52 | gtnotf | 9904 |
| gtsp | 98E3 | gtspa | 98E6 | hda | 9B03 | hdb | 9B14 | hdf | 9B1E |
| hexdd | 9419 | hexdeaq | 9431 | hexdeq | 9428 | hexdf | 9445 | hexdg | 9460 |
| hexdh | 9470 | hexdi | 947A | hexdl | 9480 | hexnum | 9319 | hfderr | 90F3 |
| hilo | 95B1 | hnout | C035 | hnua | 9320 | hnub | 9332 | hoofdprog | 90CF |
| illcm2 | 9255 | illcom | 91F6 | illcom2 | 9252 | illcoma | 91F9 | illop | 9CD5 |
| illop2 | 9D22 | illpar | 9216 | illpara | 9219 | illparb | 921C | ilparm | 99C2 |
| in | 92E3 | ina | 92E6 | incpnt | 94B6 | indexlen | 9CB3 | inh | 9F12 |
| inl | 9F11 | inpbuffer | 9F6E | input | C020 | insch | 9306 | inslen | 9F0C |
| invchr | 9C92 | lengthtab | 9DD8 | looktab | 9195 | loupch | C041 | lp1 | 9D69 |
| lp2 | 9D65 | lp3 | 9D8E | lshxa | 9A6B | lshxb | 9A7B | lshxc | 9A7D |
| main | 910F | mainpgm | 90F7 | mon65 | 9000 | monb | 94DC | monbeg | 9000 |
| monc | 94E9 | monca | 94EC | mond | 94FE | mond1 | 9517 | mone | 952C |
| monf | 9536 | mong | 9543 | monh | 9550 | monha | 955D | move | 959A |
| movea | 95AE | moveb | 95EA | ndvrs | 9F47 | nmb | 9F30 | nmi | 90CB |
| nmivector | E7B1 | nmivs | 9F64 | norock1 | 9D35 | norock2 | 9D50 | norock3 | 9D8C |
| notbit | 9D07 | notvat | 9345 | npc1 | 9C60 | npc2 | 9C9E | odd | 9CEF |
| oddop | 9CC2 | oddop2 | 9CFF | oddop3 | 9D17 | onderg | 9F20 | operlp | 9C67 |
| operpr | 9C65 | output | C023 | overfl | 99B4 | parah | 9F04 | paral | 9F03 |
| parbh | 9F06 | parbl | 9F05 | parch | 9F08 | parcl | 9F07 | parer | 92E0 |
| pch | 9F28 | pcl | 9F27 | pclear | 98A2 | pijl | 9262 | pntdce | 9385 |
| pntx | 9F16 | pntxh | 9F18 | pntxhsave | 9F15 | pntxinc | 935F | pntxinca | 9367 |
| pntxl | 9F17 | pntxlsave | 9F14 | pnty | 9F1A | pntydec | 9375 | pntyh | 9F1C |
| pntyinc | 9368 | pntyinca | 9370 | pntyl | 9F1B | pointdec | 9371 | pr0 | 9797 |
| pr01 | 9792 | pr01a | 9799 | pradda | 9665 | prbffg | 9F1E | prbufpijl | 923B |
| prbyte | C038 | prbyts | 9352 | prcnt | 9F47 | preg | 9F2A | prerrbyt | 9640 |
| presc | 948B | prgrf | 948E | prgrg | 9491 | prgrna | 94A3 | prgrnr | 949C |
| pridec | 9B33 | prinv | 9494 | prlh6 | 99A2 | prnor | 9497 | prompt | 9F50 |
| prtext | C03B | prthex | 9D27 | prtmnem | 9D53 | pta | 0000 | ptb | 0002 |
| qcomvr | 9F61 | reladout | 9C96 | res | 9F01 | reserve | 9F56 | resin | 9310 |
| reszer | 9C0E | rl1 | 9CAD | rl2 | 9CB2 | rockw1 | 9CC8 | rsbkpt | 987D |
| rsta | 904D | rstb | 9057 | run | 97C4 | runa | 97CC | savstackp | 9F00 |
| sazer | 9BFE | schei | 9F1F | schtt | 9309 | schttend | 930F | sdh1 | 9ABF |
| sdh2 | 9AC3 | sdh3 | 9AD8 | sdh4 | 9AE0 | sdh5 | 9AE5 | setvar | 905D |
| setvr1 | 905F | setvr2 | 906E | showpra | 9784 | spa | C032 | spra | 9786 |
| spuser | 9F29 | stabuf | 9153 | stcker | 9AAB | stmp0 | 9F0A | store | 9568 |
| stpntx | 93EC | stpnty | 93FB | strnd | 9579 | svaint | E770 | svstck | 9F13 |
| swint | 90A4 | swivector | E76E | swivs | 9F66 | tabvec | 9F68 | testsp | 93D3 |
| testspa | 93DD | thrdlttr | 9E4C | tmp0 | 0004 | tmp4 | 9F0D | tmp6 | 9F0F |
| tweepar | 92A9 | twpaby | 96B2 | twpend | 96D6 | uitbuffer | 9181 | userx | 9F6A |
| usery | 9F6C | v | 9F00 | validop | 9CD9 | valit | 9348 | valt | 934F |
| vart1 | 9071 | vart2 | 907D | vererr | 963A | veria | 9621 | verib | 9629 |
| veric | 9631 | vulbuf | 9126 | whbuf | 9F31 | wroa | 9BF4 | x2 | 9D18 |
| x2xa | 9D1B | x6xe | 9D15 | xreg | 9F25 | xspace | 9358 | yreg | 9F26 |
| zkvsp | 9278 | zoek | 927B | zpijlsp | 9272 | | | | |

Errors detected: 0