OS SERIES VII
GEOFF COX

```
*************************************************************************
*                                                                       *
*       OSBYTE  9   Duration of first colour                            *
*                                                                       *
*************************************************************************
;on entry Y=0, X=new value

                        ;
E6B0    INY             ;Y is incremented to 1
E6B1    CLC             ;clear carry



*************************************************************************
*                                                                       *
*       OSBYTE  10   Duration of second colour                          *
*                                                                       *
*************************************************************************

;on entry Y=0 or 1 if from FX 9 call, X=new value


E6B2    LDA     &0252,Y ;get mark period count
E6B5    PHA             ;push it
E6B6    TXA             ;get new count
E6B7    STA     &0252,Y ;store it
E6BA    PLA             ;get back original value
E6BB    TAY             ;put it in Y
E6BC    LDA     &0251   ;get value of flash counter
E6BF    BNE     &E6D1   ;if not zero E6D1

E6C1    STX     &0251   ;else restore old value
E6C4    LDA     &0248   ;get current video ULA control register setting
E6C7    PHP             ;push flags
E6C8    ROR             ;rotate bit 0 into carry, carry into bit 7
E6C9    PLP             ;get back flags
E6CA    ROL             ;rotate back carry into bit 0
E6CB    STA     &0248   ;store it in RAM copy
E6CE    STA     &FE20   ;and ULA control register

E6D1    BVC     &E6AD   ;then exit via OSBYTE 7/8



*************************************************************************
*                                                                       *
*       OSBYTE  2   select input stream                                 *
*                                                                       *
*************************************************************************

;on input X contains stream number

E6D3    TXA             ;A=X
E6D4    AND     #&01    ;blank out bits 1 - 7
E6D6    PHA             ;push A
E6D7    LDA     &0250   ;and get current ACIA control setting
E6DA    ROL             ;Bit 7 into carry
E6DB    CPX     #&01    ;if X>=1 then
E6DD    ROR             ;bit 7 of A=1
E6DE    CMP     &0250   ;compare this with ACIA control setting
E6E1    PHP             ;push processor
E6E2    STA     &0250   ;put A into ACIA control setting
E6E5    STA     &FE08   ;and write to control register
```

```
E6E8    JSR     &E173   ;set up RS423 buffer
E6EB    PLP             ;get back P
E6EC    BEQ     &E6F1   ;if new setting different from old E6F1 else
E6EE    BIT     &FE09   ;set bit 6 and 7

E6F1    LDX     &0241   ;get current input buffer number
E6F4    PLA             ;get back A
E6F5    STA     &0241   ;store it
E6F8    RTS             ;and return


*************************************************************************
*                                                                       *
*       OSBYTE  13   disable events                                     *
*                                                                       *
*************************************************************************

        ;X contains event number 0-9

E6F9    TYA             ;Y=0 A=0


*************************************************************************
*                                                                       *
*       OSBYTE  14   enable events                                      *
*                                                                       *
*************************************************************************

        ;X contains event number 0-9


E6FA    CPX     #&0A    ;if X>9
E6FC    BCS     &E6AE   ;goto E6AE for exit
E6FE    LDY     &02BF,X ;else get event enable flag
E701    STA     &02BF,X ;store new value in flag
E704    BVC     &E6AD   ;and exit via E6AD


*************************************************************************
*                                                                       *
*       OSBYTE  16   Select A/D channel                                 *
*                                                                       *
*************************************************************************
 ;X contains channel number or 0 if disable conversion

E706    BEQ     &E70B   ;if X=0 then E70B
E708    JSR     &DE8C   ;start conversion

E70B    LDA     &024D   ;get  current maximum ADC channel number
E70E    STX     &024D   ;store new value
E711    TAX             ;put old value in X
E712    RTS             ;and exit
                        ;


*************************************************************************
*                                                                       *
*       OSBYTE 129   Read key within time limit                         *
*                                                                       *
*************************************************************************
 ;X and Y contains either time limit in centi seconds Y=&7F max
 ; or Y=&FF and X=-ve INKEY value
```

```
E313    TYA             ;A=Y
E714    BMI     &E721   ;if Y=&FF the E721
E716    CLI             ;else allow interrupts
E717    JSR     &DEBB   ;and go to timed routine
E71A    BCS     &E71F   ;if carry set then E71F
E71C    TAX             ;then X=A
E71D    LDA     #&00    ;A=0

E71F    TAY             ;Y=A
E720    RTS             ;and return
                        ;
                        ;scan keyboard
E721    TXA             ;A=X
E722    EOR     #&7F    ;convert to keyboard input
E724    TAX             ;X=A
E725    JSR     &F068   ;then scan keyboard
E728    ROL             ;put bit 7 into carry
E729    LDX     #&FF    ;X=&FF
E72B    LDY     #&FF    ;Y=&FF
E72D    BCS     &E731   ;if bit 7 of A was set goto E731 (RTS)
E72F    INX             ;else X=0
E730    INY             ;and Y=0
E731    RTS             ;and exit


********** check occupancy of input or free space of output buffer *******
                        ;X=buffer number
        ;Buffer number  Address         Flag    Out pointer     In pointer
        ;0=Keyboard     3E0-3FF         2CF     2D8             2E1
        ;1=RS423 Input  A00-AFF         2D0     2D9             2E2
        ;2=RS423 output 900-9BF         2D1     2DA             2E3
        ;3=printer      880-8BF         2D2     2DB             2E4
        ;4=sound0       840-84F         2D3     2DC             2E5
        ;5=sound1       850-85F         2D4     2DD             2E6
        ;6=sound2       860-86F         2D5     2DE             2E7
        ;7=sound3       870-87F         2D6     2DF             2E8
        ;8=speech       8C0-8FF         2D7     2E0             2E9

E732    TXA             ;buffer number in A
E733    EOR     #&FF    ;invert it
E735    TAX             ;X=A
E736    CPX     #&02    ;is X>1
E738    CLV             ;clear V flag
E739    BVC     &E73E   ;and goto E73E count buffer

E73B    BIT     &D9B7   ;set V
E73E    JMP     (&022E) ;CNPV defaults to E1D1


************* check RS423 input buffer ***********************************

E741    SEC
E742    LDX     #&01    ;X=1 to point to buffer
E744    JSR     &E738   ;and count it
E747    CPY     #&01    ;if the hi byte of the answer is 1 or more
E749    BCS     &E74E   ;then Return
E74B    CPX     &025B   ;else compare with minimum buffer space
E74E    RTS             ;and exit



*************************************************************************
*                                                                       *
*       OSBYTE 128  READ ADC CHANNEL                                    *
*                                                                       *
```

```
*********************************************************************

;ON Entry: X=0               Exit Y contains number of last channel converted
;          X=channel number       X,Y contain 16 bit value read from channe
;          X<0 Y=&FF              X returns information about various buffers
;          X=&FF (keyboard )      X=number of characters in buffer
;          X=&FE (RS423 Input)    X=number of characters in buffer
;          X=&FD (RS423 output)   X=number of empty spaces in buffer
;          X=&FC (Printer)        X=number of empty spaces in buffer
;          X=&FB (sound 0)        X=number of empty spaces in buffer
;          X=&FA (sound 1)        X=number of empty spaces in buffer
;          X=&F9 (sound 2)        X=number of empty spaces in buffer
;          X=&F8 (sound 3)        X=number of empty spaces in buffer
;          X=&F7 (Speech )        X=number of empty spaces in buffer


E74F    BMI     &E732   ;if X is -ve then E732 count spaces
E751    BEQ     &E75F   ;if X=0 then E75F
E753    CPX     #&05    ;else check for Valid channel
E755    BCS     &E729   ;if not E729 set X & Y to 0 and exit
E757    LDY     &02B9,X ;get conversion values for channel of interest Hi &
E75A    LDA     &02B5,X ;lo byte
E75D    TAX             ;X=lo byte
E75E    RTS             ;and exit


E75F    LDA     &FE40   ;read system VIA port B
E762    ROR             ;move high nybble to low
E763    ROR             ;
E764    ROR             ;
E765    ROR             ;
E766    EOR     #&FF    ;and invert it
E768    AND     #&03    ;isolate the FIRE buttons
E76A    LDY     &02BE   ;get analogue system flag byte
E76D    STX     &02BE   ;store X here
E770    TAX             ;A=X bits 0 and 1 indicate fire buttons
E771    RTS             ;and return




*********************************************************************
*********************************************************************
**                                                                 **
**      OSBYTE  DEFAULT ENTRY POINT                                **
**                                                                 **
**      pointed to by default BYTEV                                **
**                                                                 **
*********************************************************************
*********************************************************************


E772    PHA             ;save A
E773    PHP             ;save Processor flags
E774    SEI             ;disable interrupts
E775    STA     &EF     ;store A,X,Y in zero page
E777    STX     &F0     ;
E779    STY     &F1     ;
```

```
E77B    LDX     #&07    ;X=7 to signal osbyte is being attempted
E77D    CMP     #&75    ;if A=0-116
E77F    BCC     &E7C2   ;then E7C2
E781    CMP     #&A1    ;if A<161
E783    BCC     &E78E   ;then E78E
E785    CMP     #&A6    ;if A=161-165
E787    BCC     &E7C8   ;then EC78
E789    CLC             ;clear carry

E78A    LDA     #&A1    ;A=&A1
E78C    ADC     #&00    ;

********* process osbyte calls 117 - 160 ****************************

E78E    SEC             ;set carry
E78F    SBC     #&5F    ;convert to &16 to &41 (22-65)

E791    ASL             ;double it (44-130)
E792    SEC             ;set carry


E793    STY     &F1     ;store Y
E795    TAY             ;Y=A
E796    BIT     &025E   ;read econet intercept flag
E799    BPL     &E7A2   ;if no econet intercept required E7A2

E79B    TXA             ;else A=X
E79C    CLV             ;V=0
E79D    JSR     &E57E   ; to JMP via ECONET vector
E7A0    BVS     &E7BC   ;if return with V set E7BC


E7A2    LDA     &E5B4,Y ;get address from table
E7A5    STA     &FB     ;store it as hi byte
E7A7    LDA     &E5B3,Y ;repeat for lo byte
E7AA    STA     &FA     ;
E7AC    LDA     &EF     ;restore A
E7AE    LDY     &F1     ;Y
E7B0    BCS     &E7B6   ;if carry is set E7B6

E7B2    LDY     #&00    ;else
E7B4    LDA     (&F0),Y ;get value from address pointed to by &F0/1 (Y,X)

E7B6    SEC             ;set carry
E7B7    LDX     &F0     ;restore X
E7B9    JSR     &F058   ;call &FA/B

E7BC    ROR             ;C=bit 0
E7BD    PLP             ;get back flags
E7BE    ROL             ;bit 0=Carry
E7BF    PLA             ;get back A
E7C0    CLV             ;clear V
E7C1    RTS             ;and exit


*************** Process OSBYTE CALLS BELOW &75 *************************

E7C2    LDY     #&00    ;Y=0
E7C4    CMP     #&16    ;if A<&16
E7C6    BCC     &E791   ;goto E791

E7C8    PHP             ;push flags
E7C9    PHP             ;push flags

E7CA    PLA             ;pull flags
E7CB    PLA             ;pull flags
```

```
E7CC    JSR     &F168   ;offer paged ROMS service 7/8 unrecognised osbyte/word
E7CF    BNE     &E7D6   ;if roms don't recognise it then E7D6
E7D1    LDX     &F0     ;else restore X
E7D3    JMP     &E7BC   ;and exit


E7D6    PLP             ;else pull flags
E7D7    PLA             ;and A
E7D8    BIT     &D9B7   ;set V and C
E7DB    RTS             ;and return

E7DC    LDA     &EB     ;read cassette critical flag bit 7 = busy
E7DE    BMI     &E812   ;if busy then EB12


E7E0    LDA     #&08    ;else A=8 to check current Catalogue status
E7E2    AND     &E2     ;by anding with CFS status flag
E7E4    BNE     &E7EA   ;if not set (not in use) then E7EA RTS
E7E6    LDA     #&88    ;A=%10001000
E7E8    AND     &BB     ;AND with FS options (short msg bits)
E7EA    RTS             ;RETURN




*************************************************************************
*************************************************************************
**                                                                     **
**      OSWORD  DEFAULT ENTRY POINT                                    **
**                                                                     **
**      pointed to by default WORDV                                    **
**                                                                     **
*************************************************************************
*************************************************************************


E7EB    PHA             ;Push A
E7EC    PHP             ;Push flags
E7ED    SEI             ;disable interrupts
E7EE    STA     &EF     ;store A,X,Y
E7F0    STX     &F0     ;
E7F2    STY     &F1     ;
E7F4    LDX     #&08    ;X=8
E7F6    CMP     #&E0    ;if A=>224
E7F8    BCS     &E78A   ;then E78A with carry set

E7FA    CMP     #&0E    ;else if A=>14
E7FC    BCS     &E7C8   ;else E7C8 with carry set pass to ROMS & exit

E7FE    ADC     #&44    ;add to form pointer to table
E800    ASL             ;double it
E801    BCC     &E793   ;goto E793 ALWAYS!! (carry clear E7F8)
                        ;this reads bytes from table and enters routine



*************************************************************************
*                                                                       *
*       OSWORD  05   ENTRY POINT                                        *
*                                                                       *
*       read a byte from I/O memory                                     *
*                                                                       *
*************************************************************************
;block of 4 bytes set at address pointed to by 00F0/1 (Y,X)
```

```
;XY +0  ADDRESS of byte
;   +4  on exit byte read


E803    JSR     &E815   ;set up address of data block
E806    LDA     (&F9,X) ;get byte
E808    STA     (&F0),Y ;store it
E80A    RTS             ;exit




*************************************************************************
*                                                                       *
*       OSWORD  06   ENTRY POINT                                        *
*                                                                       *
*       write a byte to I/O memory                                      *
*                                                                       *
*************************************************************************
;block of 5 bytes set at address pointed to by 00F0/1 (Y,X)
;XY +0  ADDRESS of byte
;   +4  byte to be written


E80B    JSR     &E815   ;set up address
E80E    LDA     (&F0),Y ;get byte
E810    STA     (&F9,X) ;store it
E812    LDA     #&00    ;a=0
E814    RTS             ;exit


********************: set up data block *********************************

E815    STA     &FA     ;&FA=A
E817    INY             ;Y=1
E818    LDA     (&F0),Y ;get byte from block
E81A    STA     &FB     ;store it
E81C    LDY     #&04    ;Y=4
E81E    LDX     #&01    ;X=1
E820    RTS             ;and exit



*************************************************************************
*                                                                       *
*       OSBYTE  00   ENTRY POINT                                        *
*                                                                       *
*       read OS version number                                          *
*                                                                       *
*************************************************************************


E821    BNE     &E81E   ;if A <> 0 then exit else print error
E823    BRK             ;
E824    DB      &F7             ;error number
E825    DB      'OS 1.20'       ;error message
E82C    BRK




*************************************************************************
*                                                                       *
```

```
*       OSWORD  07   ENTRY POINT                                          *
*                                                                         *
*       make a sound                                                      *
*                                                                         *
***************************************************************************
;block of 8 bytes set at address pointed to by 00F0/1
;XY +0  Channel or +0=Flush, Channel:+1=Hold,Sync
;    2  Amplitude
;    4  Pitch
;    6  Duration
;Y=0 on entry

E82D    INY             ;increment Y
E82E    LDA     (&F0),Y ;get channel byte from table
E830    CMP     #&FF    ;if its &FF goto speech
E832    BEQ     &E88D   ;at E8DD
E834    CMP     #&20    ;else if>=&20 set carry
E836    LDX     #&08    ;X=8 for unrecognised OSWORD call
E838    BCS     &E7CA   ;and if carry set off to E7CA to offer to ROMS
E83A    DEY             ;Y=0
E83B    JSR     &E8C9   ;return with Carry set if Flush=1:A=C
E83E    ORA     #&04    ;convert to buffer number
E840    TAX             ;A=X
E841    BCC     &E848   ;and if carry clear (ex E8C9) then E848

E843    JSR     &E1AE   ;else flush buffer
E846    LDY     #&01    ;Y=1

E848    JSR     &E8C9   ;returns with carry set if H=1;A=S
E84B    STA     &FA     ;Sync number =&FA
E84D    PHP             ;ave flags
E84E    LDY     #&06    ;Y=6
E850    LDA     (&F0),Y ;Get Duration byte
E852    PHA             ;push it

E853    LDY     #&04    ;Y=4
E855    LDA     (&F0),Y ;get pitch byte
E857    PHA             ;push it
E858    LDY     #&02    ;get amplitude byte
E85A    LDA     (&F0),Y ;
E85C    ROL             ;H now =bit 0 (carry shifted)
E85D    SEC             ;set carry
E85E    SBC     #&02    ;subract 2
E860    ASL             ;multiply by 4
E861    ASL             ;
E862    ORA     &FA     ;add S byte (0-3)

                        ;at this point bit 7=0 for an envelope
                        ;bits 3-6 give envelope -1, or volume +15
                        ;(i.e.complemented)
                        ;bit 2 gives H
                        ;bits 0-1 =Sync
E864    JSR     &E1F8   ;transfer to buffer
E867    BCC     &E887   ;if C set on exit succesful transfer so E887

E869    PLA             ;else get back
E86A    PLA             ;stored values
E86B    PLP             ;

***************************************************************************
*                                                                         *
*       OSBYTE  117   ENTRY POINT                                         *
```

```
*                                                                       *
*       read VDU status                                                 *
*                                                                       *
*************************************************************************


E86C    LDX     &D0     ;get VDU status byte in X
E86E    RTS             ;and return


************* set up sound data for Bell *********************************


E86F    PHP             ;push P
E870    SEI             ;bar interrupts
E871    LDA     &0263   ;get bell channel number in A
E874    AND     #&07    ; (bits 0-3 only set)
E876    ORA     #&04    ;set bit 2
E878    TAX             ;X=A = bell channel number +4=buffer number
E879    LDA     &0264   ;get bell amplitude/envelope number
E87C    JSR     &E4B0   ;store it in buffer pointed to by X
E87F    LDA     &0266   ;get bell duration
E882    PHA             ;save it
E883    LDA     &0265   ;get bell frequency
E886    PHA             ;save it
E887    SEC             ;set carry

E888    ROR     &0800,X ;and pass into bit 7 to warn that channel is active
E88B    BMI     &E8A4   ;goto E8A4 ALWAYS!!


**********:speech handling routine ************************************

E88D    PHP             ;push flags
E88E    INY             ;Y=2
E88F    LDA     (&F0),Y ;get byte
E891    PHA             ;store it
E892    INY             ;Y=4
E893    LDA     (&F0),Y ;get byte
E895    PHA             ;store it
E896    LDY     #&00    ;Y=0
E898    LDA     (&F0),Y ;get byte
E89A    LDX     #&08    ;X=8
E89C    JSR     &E1F8   ;select speech buffer and pass A
E89F    BCS     &E869   ;if carry set on return restore stack
                        ;and exit
E8A1    ROR     &02D7   ;else clear bit 7 of buffer busy flag to show
                        ;buffer has been set
E8A4    PLA             ;get back byte
E8A5    JSR     &E4B0   ;enter it in buffer X
E8A8    PLA             ;get back next
E8A9    JSR     &E4B0   ;and enter it again
E8AC    PLP             ;get back flags
E8AD    RTS             ;and exit



*************************************************************************
*                                                                       *
*       OSWORD  08   ENTRY POINT                                        *
*                                                                       *
*       define envelope                                                 *
*                                                                       *
*************************************************************************
;block of 14 bytes set at address pointed to by 00F0/1
;XY +0  Envelope number, also in A
```

```
;    1  bits 0-6 length of each step in centi-secsonds bit 7=0 auto repeat
;    2  change of Pitch per step (-128-+127) in section 1
;    3  change of Pitch per step (-128-+127) in section 2
;    4  change of Pitch per step (-128-+127) in section 3
;    5  number of steps in section 1 (0-255)
;    6  number of steps in section 2 (0-255)
;    7  number of steps in section 3 (0-255)
;    8  change of amplitude per step during attack  phase (-127 to +127)
;    9  change of amplitude per step during decay   phase (-127 to +127)
;   10  change of amplitude per step during sustain phase (-127 to +127)
;   11  change of amplitude per step during release phase (-127 to +127)
;   12  target level at end of attack phase   (0-126)
;   13  target level at end of decay  phase   (0-126)

;Y=0 on entry


E8AE    SBC     #&01    ;set up appropriate displacement to storage area
E8B0    ASL             ;A=(A-1)*16 or 15
E8B1    ASL             ;
E8B2    ASL             ;
E8B3    ASL             ;
E8B4    ORA     #&0F    ;
E8B6    TAX             ;X=A
E8B7    LDA     #&00    ;A=0

E8B9    LDY     #&10    ;Y=10

E8BB    CPY     #&0E    ;is Y>=14??
E8BD    BCS     &E8C1   ;yes then E8C1
E8BF    LDA     (&F0),Y ;else get byte from parameter block
E8C1    STA     &08C0,X ;and store it in appropriate area
E8C4    DEX             ;decrement X
E8C5    DEY             ;Decrement Y
E8C6    BNE     &E8BB   ;if not 0 then do it again
E8C8    RTS             ;else exit
                        ;note that envelope number is NOT transferred

E8C9    LDA     (&F0),Y ;get byte
E8CB    CMP     #&10    ;is it greater than 15, if so set carry
E8CD    AND     #&03    ;and 3 to clear bits 2-7
E8CF    INY             ;increment Y
E8D0    RTS             ;and exit



*************************************************************************
*                                                                       *
*       OSWORD  03   ENTRY POINT                                        *
*                                                                       *
*       read interval timer                                             *
*                                                                       *
*************************************************************************
F0/1 points to block to store data

E8D1    LDX     #&0F    ;X=&F displacement from clock to timer
E8D3    BNE     &E8D8   ;jump to E8D8



*************************************************************************
*                                                                       *
*       OSWORD  01   ENTRY POINT                                        *
*                                                                       *
```

```
*       read system clock                                                      *
*                                                                             *
*******************************************************************************
F0/1 points to block to store data


E8D5    LDX     &0283   ;X=current system clock store pointer

E8D8    LDY     #&04    ;Y=4
E8DA    LDA     &028D,X ;read byte
E8DD    STA     (&F0),Y ;store it in parameter block
E8DF    INX             ;X=x+1
E8E0    DEY             ;Y=Y-1
E8E1    BPL     &E8DA   ;if Y>0 then do it again
E8E3    RTS             ;else exit



*******************************************************************************
*                                                                             *
*       OSWORD  04   ENTRY POINT                                              *
*                                                                             *
*       write interval timer                                                  *
*                                                                             *
*******************************************************************************
F0/1 points to block to store data


E8E4    LDA     #&0F    ;offset between clock and timer
E8E6    BNE     &E8EE   ;jump to E8EE ALWAYS!!



*******************************************************************************
*                                                                             *
*       OSWORD  02   ENTRY POINT                                              *
*                                                                             *
*       write system clock                                                    *
*                                                                             *
*******************************************************************************
F0/1 points to block to store data


E8E8    LDA     &0283   ;get current clock store pointer
E8EB    EOR     #&0F    ;and invert to get inactive timer
E8ED    CLC             ;clear carry

E8EE    PHA             ;store A
E8EF    TAX             ;X=A
E8F0    LDY     #&04    ;Y=4
E8F2    LDA     (&F0),Y ;and transfer all 5 bytes
E8F4    STA     &028D,X ;to the clock or timer
E8F7    INX             ;
E8F8    DEY             ;
E8F9    BPL     &E8F2   ;if Y>0 then E8F2
E8FB    PLA             ;get back stack
E8FC    BCS     &E8E3   ;if set (write to timer) E8E3 exit
E8FE    STA     &0283   ;write back current clock store
E901    RTS             ;and exit
```

```
*************************************************************************
*                                                                       *
*       OSWORD  00   ENTRY POINT                                        *
*                                                                       *
*       read line from current input to memory                          *
*                                                                       *
*************************************************************************
;F0/1 points to parameter block
;       +0/1    buffer address for input
;       +2      Maximum line length
;       +3      minimum acceptable ASCII value
;       +4      maximum acceptable ASCII value

E902    LDY     #&04    ;Y=4

E904    LDA     (&F0),Y ;transfer bytes 4,3,2 to 2B3-2B5
E906    STA     &02B1,Y ;
E909    DEY             ;decrement Y
E90A    CPY     #&02    ;until Y=1
E90C    BCS     &E904   ;

E90E    LDA     (&F0),Y ;get address of input buffer
E910    STA     &E9     ;store it in &E9 as temporary buffer
E912    DEY             ;decrement Y
E913    STY     &0269   ;Y=0 store in print line counter for paged mode
E916    LDA     (&F0),Y ;get lo byte of address
E918    STA     &E8     ;and store in &E8
E91A    CLI             ;allow interrupts
E91B    BCC     &E924   ;Jump to E924

E91D    LDA     #&07    ;A=7
E91F    DEY             ;decrement Y
E920    INY             ;increment Y
E921    JSR     OSWRCH  ;and call OSWRCH

E924    JSR     OSRDCH  ;else read character  from input stream
E927    BCS     &E972   ;if carry set then illegal character or other error
                        ;exit via E972
E929    TAX             ;X=A
E92A    LDA     &027C   ;A=&27C get character destination status
E92D    ROR             ;put VDU driver bit in carry
E92E    ROR             ;if this is 1 VDU driver is disabled
E92F    TXA             ;X=A
E930    BCS     &E937   ;if Carry set E937
E932    LDX     &026A   ;get number of items in VDU queque
E935    BNE     &E921   ;if not 0 output character and loop round again

E937    CMP     #&7F    ;if character is not delete
E939    BNE     &E942   ;goto E942
E93B    CPY     #&00    ;else is Y=0
E93D    BEQ     &E924   ;and goto E924
E93F    DEY             ;decrement Y
E940    BCS     &E921   ;and if carry set E921 to output it
E942    CMP     #&15    ;is it delete line &21
E944    BNE     &E953   ;if not E953
E946    TYA             ;else Y=A, if its 0 we are still reading first
                        ;character
E947    BEQ     &E924   ;so E924
E949    LDA     #&7F    ;else output DELETES

E94B    JSR     OSWRCH  ;until Y=0
E94E    DEY             ;
E94F    BNE     &E94B   ;

E951    BEQ     &E924   ;then read character again
```

```
E953    STA     (&E8),Y ;store character in designated buffer
E955    CMP     #&0D    ;is it CR?
E957    BEQ     &E96C   ;if so E96C
E959    CPY     &02B3   ;else check the line length
E95C    BCS     &E91D   ;if = or greater loop to ring bell
E95E    CMP     &02B4   ;check minimum character
E961    BCC     &E91F   ;if less than minimum backspace
E963    CMP     &02B5   ;check maximum character
E966    BEQ     &E920   ;if equal E920
E968    BCC     &E920   ;or less E920
E96A    BCS     &E91F   ;then JUMP E91F

E96C    JSR     OSNEWL  ;output CR/LF
E96F    JSR     &E57E   ;call Econet vector

E972    LDA     &FF     ;A=ESCAPE FLAG
E974    ROL             ;put bit 7 into carry
E975    RTS             ;and exit routine



*************************************************************************
*                                                                       *
*       OSBYTE  05   ENTRY POINT                                        *
*                                                                       *
*       SELECT PRINTER TYPE                                             *
*                                                                       *
*************************************************************************



E976    CLI             ;allow interrupts briefly
E977    SEI             ;bar interrupts
E978    BIT     &FF     ;check if ESCAPE is pending
E97A    BMI     &E9AC   ;if it is E9AC
E97C    BIT     &02D2   ;else check bit 7 buffer 3 (printer)
E97F    BPL     &E976   ;if not empty bit 7=0 E976

E981    JSR     &E1A4   ;check for user defined routine
E984    LDY     #&00    ;Y=0
E986    STY     &F1     ;F1=0




*************************************************************************
*                                                                       *
*       OSBYTE  01   ENTRY POINT                                        *
*                                                                       *
*       READ/WRITE USER FLAG (&281)                                     *
*                                                                       *
*          AND                                                          *
*                                                                       *
*       OSBYTE  06   ENTRY POINT                                        *
*                                                                       *
*       SET PRINTER IGNORE CHARACTER                                    *
*                                                                       *
*************************************************************************
; A contains osbyte number


E988    ORA     #&F0    ;A=osbyte +&F0
E98A    BNE     &E99A   ;JUMP to E99A
```

```
*************************************************************************
*                                                                       *
*       OSBYTE  0C   ENTRY POINT                                        *
*                                                                       *
*       SET  KEYBOARD AUTOREPEAT RATE                                   *
*                                                                       *
*************************************************************************


E98C    BNE     &E995   ;if &F0<>0 goto E995
E98E    LDX     #&32    ;if X=0 in original call then X=32
E990    STX     &0254   ;to set keyboard autorepeat delay ram copy
E993    LDX     #&08    ;X=8



*************************************************************************
*                                                                       *
*       OSBYTE  0B   ENTRY POINT                                        *
*                                                                       *
*       SET  KEYBOARD AUTOREPEAT DELAY                                  *
*                                                                       *
*************************************************************************


E995    ADC     #&CF    ;A=A+&D0 (carry set)



*************************************************************************
*                                                                       *
*       OSBYTE  03   ENTRY POINT                                        *
*                                                                       *
*       SELECT OUTPUT STREAM                                            *
*                                                                       *
*               AND                                                     *
*                                                                       *
*                                                                       *
*       OSBYTE  04   ENTRY POINT                                        *
*                                                                       *
*       ENABLE/DISABLE CURSOR EDITING                                   *
*                                                                       *
*************************************************************************


E997    CLC             ;c,ear carry
E998    ADC     #&E9    ;A=A+&E9

E99A    STX     &F0     ;sto2e X



*************************************************************************
*                                                                       *
*       OSBYTE  A6-FF   ENTRY POINT                                     *
*                                                                       *
*       READ/ WRITE SYSTEM VARIABLE OSBYTE NO. +&190                    *
```

```
*                                                                      *
************************************************************************


E99C    TAY             ;Y=A
E99D    LDA     &0190,Y ;i.e. A=&190 +osbyte call!
E9A0    TAX             ;preserve this
E9A1    AND     &F1     ;new value = OLD value AND Y EOR X!
E9A3    EOR     &F0     ;
E9A5    STA     &0190,Y ;store it
E9A8    LDA     &0191,Y ;get value of next byte into A
E9AB    TAY             ;Y=A
E9AC    RTS             ;and exit


******* SERIAL BAUD RATE LOOK UP TABLE **********************************

E9AD    DB      &64     ; % 01100100      75
E9AE    DB      &7F     ; % 01111111     150
E9AF    DB      &5B     ; % 01011011     300
E9B0    DB      &6D     ; % 01101101    1200
E9B1    DB      &C9     ; % 11001001    2400
E9B2    DB      &F6     ; % 11110110    4800
E9B3    DB      &D2     ; % 11010010    9600
E9B4    DB      &E4     ; % 11100100   19200
E9B5    DB      &40     ; % 01000000


************************************************************************
*                                                                      *
*       OSBYTE  &13   ENTRY POINT                                      *
*                                                                      *
*       Wait for Animation                                             *
*                                                                      *
************************************************************************


E9B6    LDA     &0240   ;read vertical sync counter
E9B9    CLI             ;allow interrupts briefly
E9BA    SEI             ;bar interrupts
E9BB    CMP     &0240   ;is it 0 (Vertical sync pulse)
E9BE    BEQ     &E9B9   ;no then E9B9



************************************************************************
*                                                                      *
*       OSBYTE  160   ENTRY POINT                                      *
*                                                                      *
*       READ VDU VARIABLE                                              *
*                                                                      *
************************************************************************
;X contains the variable number
;0 =lefthand column in pixels current graphics window
;2 =Bottom row in pixels current graphics window
;4 =Right hand column in pixels current graphics window
;6 =Top row in pixels current graphics window
;8 =lefthand column in absolute characters current text window
;9 =Bottom row in absolute characters current text window
;10 =Right hand column i. absolute characters current text window
;11 =Top row in absolute characters current text window
;12-15 current graphics origin in external coordinates
;16-19 current graphics cursor in external coordina4es
;20-23 old graphics cursor in internal coordinates
```

```
;24 current text cursor in X and Y

E9C0    LDY     &0301,X ;get VDU variable hi
E9C3    LDA     &0300,X ;low
E9C6    TAX             ;X=low byte
E9C7    RTS             ;and exit



*************************************************************************
*                                                                       *
*       OSBYTE  18   ENTRY POINT                                        *
*                                                                       *
*       RESET SOFT KEYS                                                 *
*                                                                       *
*************************************************************************

E9C8    LDA     #&10    ;set consistency flag
E9CA    STA     &0284   ;

E9CD    LDX     #&00    ;X=0

E9CF    STA     &0B00,X ;and wipe clean
E9D2    INX             ;soft key buffer
E9D3    BNE     &E9CF   ;until X=0 again

E9D5    STX     &0284   ;zero consistency flag
E9D8    RTS             ;and exit



*************************************************************************
*                                                                       *
*        OSBYTE &76 (118) SET LEDs to Keyboard Status                   *
*                                                                       *
*************************************************************************


                        ;osbyte entry with carry set
                        ;called from &CB0E, &CAE3, &DB8B

E9D9    PHP             ;PUSH P
E9DA    SEI             ;DISABLE INTERUPTS
E9DB    LDA     #&40    ;switch on CAPS and SHIFT lock lights
E9DD    JSR     &E9EA   ;via subroutine
E9E0    BMI     &E9E7   ;if ESCAPE exists (M set) E9E7
E9E2    CLC             ;else clear V and C
E9E3    CLV             ;before calling main keyboard routine to
E9E4    JSR     &F068   ;switch on lights as required
E9E7    PLP             ;get back flags
E9E8    ROL             ;and rotate carry into bit 0
E9E9    RTS             ;Return to calling routine
                        ;
***************** Turn on keyboard lights and Test Escape flag ************
                        ;called from &E1FE, &E9DD
                        ;
E9EA    BCC     &E9F5   ;if carry clear
E9EC    LDY     #&07    ;switch on shift lock light
E9EE    STY     &FE40   ;
E9F1    DEY             ;Y=6
E9F2    STY     &FE40   ;switch on Caps lock light
E9F5    BIT     &FF     ;set minus flag if bit 7 of &00FF is set indicating
E9F7    RTS             ;that ESCAPE condition exists, then return
```

```
                        ;
****************** Write A to SYSTEM VIA register B ************************
                        ;called from &CB6D, &CB73
E9F8    PHP             ;push flags
E9F9    SEI             ;disable interupts
E9FA    STA     &FE40   ;write register B from Accumulator
E9FD    PLP             ;get back flags
E9FE    RTS             ;and exit
                        ;



*************************************************************************
*                                                                       *
*       OSBYTE 154 (&9A) SET VIDEO ULA                                  *

*                                                                       *
*************************************************************************


E9FF    TXA             ;osbyte entry! X transferred to A thence to

*******Set Video ULA control register **entry from VDU routines **************
                        ;called from &CBA6, &DD37

EA00    PHP             ;save flags
EA01    SEI             ;disable interupts
EA02    STA     &0248   ;save RAM copy of new parameter
EA05    STA     &FE20   ;write to control register
EA08    LDA     &0253   ;read  space count
EA0B    STA     &0251   ;set flash counter to this value
EA0E    PLP             ;get back status
EA0F    RTS             ;and return



*************************************************************************
*                                                                       *
*        OSBYTE &9B (155) write to pallette register                    *

*                                                                       *
*************************************************************************

                ;entry X contains value to write
EA10    TXA             ;A=X
EA11    EOR     #&07    ;convert to palette format
EA13    PHP             ;
EA14    SEI             ;prevent interrupts
EA15    STA     &0249   ;store as current palette setting
EA18    STA     &FE21   ;store actual colour in register
EA1B    PLP             ;get back flags
EA1C    RTS             ;and exit
                        ;


*************************************************************************
*       GSINIT  string initialisation                                   *
*       F2/3 points to string offset by Y                               *
*                                                                       *
*       ON EXIT                                                         *
*       Z flag set indicates null string,                               *
*       Y points to first non blank character                           *
*       A contains first non blank character                            *
*************************************************************************
```

```
EA1D    CLC             ;clear carry

EA1E    ROR     &E4     ;Rotate moves carry to &E4
EA20    JSR     &E03A   ;get character from text
EA23    INY             ;increment Y to point at next character
EA24    CMP     #&22    ;check to see if its '"'
EA26    BEQ     &EA2A   ;if so EA2A (carry set)
EA28    DEY             ;decrement Y
EA29    CLC             ;clear carry
EA2A    ROR     &E4     ;move bit 7 to bit 6 and put carry in bit 7
EA2C    CMP     #&0D    ;check to see if its CR to set Z
EA2E    RTS             ;and return


*************************************************************************
*       GSREAD  string read routine                                     *
*       F2/3 points to string offset by Y                               *
*                                                                       *
*************************************************************************
                        ;
EA2F    LDA     #&00    ;A=0
EA31    STA     &E5     ;store A
EA33    LDA     (&F2),Y ;read first character
EA35    CMP     #&0D    ;is it CR
EA37    BNE     &EA3F   ;if not goto EA3F
EA39    BIT     &E4     ;if bit 7=1 no 2nd '"' found
EA3B    BMI     &EA8F   ;goto EA8F
EA3D    BPL     &EA5A   ;if not EA5A

EA3F    CMP     #&20    ;is less than a space?
EA41    BCC     &EA8F   ;goto EA8F
EA43    BNE     &EA4B   ;if its not a space EA4B
EA45    BIT     &E4     ;is bit 7 of &E4 =1
EA47    BMI     &EA89   ;if so goto EA89
EA49    BVC     &EA5A   ;if bit 6 = 0 EA5A
EA4B    CMP     #&22    ;is it '"'
EA4D    BNE     &EA5F   ;if not EA5F
EA4F    BIT     &E4     ;if so and Bit 7 of &E4 =0 (no previous ")
EA51    BPL     &EA89   ;then EA89
EA53    INY             ;else point at next character
EA54    LDA     (&F2),Y ;get it
EA56    CMP     #&22    ;is it '"'
EA58    BEQ     &EA89   ;if so then EA89

EA5A    JSR     &E03A   ;read a byte from text
EA5D    SEC             ;and return with
EA5E    RTS             ;carry set
                        ;
EA5F    CMP     #&7C    ;is it '|'
EA61    BNE     &EA89   ;if not EA89
EA63    INY             ;if so increase Y to point to next character
EA64    LDA     (&F2),Y ;get it
EA66    CMP     #&7C    ;and compare it with '|' again
EA68    BEQ     &EA89   ;if its '|' then EA89
EA6A    CMP     #&22    ;else is it '"'
EA6C    BEQ     &EA89   ;if so then EA89
EA6E    CMP     #&21    ;is it !
EA70    BNE     &EA77   ;if not then EA77
EA72    INY             ;increment Y again
EA73    LDA     #&80    ;set bit 7
EA75    BNE     &EA31   ;loop back to EA31 to set bit 7 in next CHR
EA77    CMP     #&20    ;is it a space
EA79    BCC     &EA8F   ;if less than EA8F Bad String Error
EA7B    CMP     #&3F    ;is it '?'
EA7D    BEQ     &EA87   ;if so EA87
EA7F    JSR     &EABF   ;else modify code as if CTRL had been pressed
```

```
EA82    BIT     &D9B7   ;if bit 6 set
EA85    BVS     &EA8A   ;then EA8A
EA87    LDA     #&7F    ;else set bits 0 to 6 in A

EA89    CLV             ;clear V
EA8A    INY             ;increment Y
EA8B    ORA     &E5     ;
EA8D    CLC             ;clear carry
EA8E    RTS             ;Return
                        ;
EA8F    BRK             ;
EA90    DB      &FD     ;error number
EA93    DB 'Bad String' ; message
EA9B    BRK             ;

************ Modify code as if SHIFT pressed ****************************

EA9C    CMP     #&30    ;if A='0' skip routine
EA9E    BEQ     &EABE   ;
EAA0    CMP     #&40    ;if A='@' skip routine
EAA2    BEQ     &EABE   ;
EAA4    BCC     &EAB8   ;if A<'@' then EAB8
EAA6    CMP     #&7F    ;else is it DELETE




EAA8    BEQ     &EABE   ;if so skip routine
EAAA    BCS     &EABC   ;if greater than &7F then toggle bit 4
EAAC    EOR     #&30    ;reverse bits 4 and 5
EAAE    CMP     #&6F    ;is it &6F (previously ' _' (&5F))
EAB0    BEQ     &EAB6   ;goto EAB6
EAB2    CMP     #&50    ;is it &50 (previously '`' (&60))
EAB4    BNE     &EAB8   ;if not EAB8
EAB6    EOR     #&1F    ;else continue to convert ` _
EAB8    CMP     #&21    ;compare &21 '!'
EABA    BCC     &EABE   ;if less than return
EABC    EOR     #&10    ;else finish conversion by toggling bit 4
EABE    RTS             ;exit
                        ;
                        ;ASCII codes &00 &20 no change
                        ;21-3F have bit 4 reverses (31-3F)
                        ;41-5E A-Z have bit 5 reversed a-z
                        ;5F & 60 are reversed
                        ;61-7E bit 5 reversed a-z becomes A-Z
                        ;DELETE unchanged
                        ;&80+ has bit 4 changed

************** Implement CTRL codes ************************************


EABF    CMP     #&7F    ;is it DEL
EAC1    BEQ     &EAD1   ;if so ignore routine
EAC3    BCS     &EAAC   ;if greater than &7F go to EAAC
EAC5    CMP     #&60    ;if A<>'`'
EAC7    BNE     &EACB   ;goto EACB
EAC9    LDA     #&5F    ;if A=&60, A=&5F

EACB    CMP     #&40    ;if A<&40
EACD    BCC     &EAD1   ;goto EAD1  and return unchanged
EACF    AND     #&1F    ;else zero bits 5 to 7
EAD1    RTS             ;return
```

```
                        ;
EAD2    DB      '/!BOOT'
EAD8    DB      &0D
```