OS SERIES IV
GEOFF COX

```
 ************************************************************************
 *                                                                      *
 *  LATERAL FILL ROUTINE                                                *
 *                                                                      *
 ************************************************************************

 D4BF    JSR     &D4AA   ;check current screen state
 D4C2    AND     &D1     ;if A and &D1 <> 0 a plotted point has been found
 D4C4    BNE     &D4B9   ;so D4B9
 D4C6    LDX     #&00    ;X=0
 D4C8    JSR     &D592   ;update pointers
 D4CB    BEQ     &D4FA   ;if 0 then D4FA
 D4CD    LDY     &031A   ;else Y=graphics scan line
 D4D0    ASL     &D1     ;
 D4D2    BCS     &D4D9   ;if carry set D4D9
 D4D4    JSR     &D574   ;else D574
 D4D7    BCC     &D4FA   ;if carry clear D4FA
 D4D9    JSR     &D3FD   ;else D3FD to pick up colour multiplier
 D4DC    LDA     (&D6),Y ;get graphics cell line
 D4DE    EOR     &035A   ;EOR with background colour
 D4E1    STA     &DA     ;and store
 D4E3    BNE     &D4F7   ;if not 0 D4F7
 D4E5    SEC             ;else set carry
 D4E6    TXA             ;A=X
 D4E7    ADC     &0361   ;add pixels/byte
 D4EA    BCC     &D4F0   ;and if carry clear D4F0
 D4EC    INC     &DB     ;else increment &DB
 D4EE    BPL     &D4F7   ;and if +ve D4F7

 D4F0    TAX             ;else X=A
 D4F1    JSR     &D104   ;display a pixel
 D4F4    SEC             ;set carry
 D4F5    BCS     &D4D9   ;goto D4D9


 D4F7    JSR     &D574   ;
 D4FA    LDY     #&00    ;Y=0
 D4FC    JSR     &D5AC   ;
 D4FF    LDY     #&20    ;
 D501    LDX     #&24    ;
 D503    JSR     &CDE6   ;exchange 300/3 +Y with 300/3+X
 D506    JSR     &D4AA   ;check screen pixel
 D509    LDX     #&04    ;Y=5
 D50B    JSR     &D592   ;
 D50E    TXA             ;A=x
 D50F    BNE     &D513   ;if A<>0 d513
 D511    DEC     &DB     ;else  &DB=&dB-1

 D513    DEX             ;X=X-1
 D514    JSR     &D54B   ;
 D517    BCC     &D540   ;
 D519    JSR     &D3ED   ;update pointers
 D51C    LDA     (&D6),Y ;get byte from graphics line
 D51E    EOR     &035A   ;EOR with background colour
 D521    STA     &DA     ;and store it
 D523    LDA     &DC     ;
 D525    BNE     &D514   ;If A-0 back to D514
 D527    LDA     &DA     ;else A=&DA
 D529    BNE     &D53D   ;if A<>d53D
 D52B    SEC             ;else set carry
 D52C    TXA             ;A=x
 D52D    ADC     &0361   ;Add number of pixels/byte
 D530    BCC     &D536   ;and if carry clear D536
```

```
D532    INC     &DB     ;else inc DB
D534    BPL     &D53D   ;and if +ve D53D
D536    TAX             ;get back X
D537    JSR     &D104   ;display a point
D53A    SEC             ;set carry
D53B    BCS     &D519   ;goto D519

D53D    JSR     &D54B   ;
D540    LDY     #&04    ;
D542    JSR     &D5AC   ;


D545    JSR     &D0D9   ;
D548    JMP     &D1B8   ;scale pointers

D54B    LDA     &D1     ;get byte mask
D54D    PHA             ;save it
D54E    CLC             ;clear carry
D54F    BCC     &D560   ;

D551    PLA             ;get back A
D552    INX             ;X=X+1
D553    BNE     &D559   ;if not 0 D559
D555    INC     &DB     ;else inc &DB
D557    BPL     &D56F   ;if +ve D56F
D559    LSR     &D1     ;
D55B    BCS     &D56F   ;if Bit 7 D1 set D56F
D55D    ORA     &D1     ;else or withA
D55F    PHA             ;save result
D560    LDA     &D1     ;A=&D1
D562    BIT     &DA     ;test bits 6 and 7 of &DA
D564    PHP             ;save flags
D565    PLA             ;get into A
D566    EOR     &DC     ;EOR and DC
D568    PHA             ;save A
D569    PLP             ;
D56A    BEQ     &D551   ;

D56C    PLA             ;A=A EOR &D1 (byte mask)
D56D    EOR     &D1     ;
D56F    STA     &D1     ;store it
D571    JMP     &D0F0   ;and display a pixel

D574    LDA     #&00    ;A=0
D576    CLC             ;Clear carry

D577    BCC     &D583   ;goto D583 if carry clear

D579    INX             ;X=X+1
D57A    BNE     &D580   ;If <>0 D580
D57C    INC     &DB     ;else inc &DB
D57E    BPL     &D56F   ;and if +ve d56F

D580    ASL             ;A=A*2
D581    BCS     &D58E   ;if C set D58E
D583    ORA     &D1     ;else A=A OR (&D1)
D585    BIT     &DA     ;set V and M from &DA b6 b7
D587    BEQ     &D579   ;
D589    EOR     &D1     ;A=AEOR &D1
D58B    LSR             ;/2
D58C    BCC     &D56F   ;if carry clear D56F
D58E    ROR             ;*2
D58F    SEC             ;set carry
D590    BCS     &D56F   ;to D56F
```

```
D592    LDA     &0300,X ;Y/A=(&300/1 +X)-(&320/1)
D595    SEC             ;
D596    SBC     &0320   ;
D599    TAY             ;
D59A    LDA     &0301,X ;
D59D    SBC     &0321   ;
D5A0    BMI     &D5A5   ;if result -ve D5A5
D5A2    JSR     &D49B   ;or negate Y/A
D5A5    STA     &DB     ;store A
D5A7    TYA             ;A=Y
D5A8    TAX             ;X=A
D5A9    ORA     &DB     ;
D5AB    RTS             ;exit


                        ;
D5AC    STY     &DA     ;Y=&DA
D5AE    TXA             ;A=X
D5AF    TAY             ;Y=A
D5B0    LDA     &DB     ;A=&DB
D5B2    BMI     &D5B6   ;if -ve D5B6
D5B4    LDA     #&00    ;A=0
D5B6    LDX     &DA     ;X=&DA
D5B8    BNE     &D5BD   ;if <>0 D5BD
D5BA    JSR     &D49B   ;negate
D5BD    PHA             ;
D5BE    CLC             ;
D5BF    TYA             ;
D5C0    ADC     &0300,X ;Y/A+(&300/1 +X)=(&320/1)
D5C3    STA     &0320   ;
D5C6    PLA             ;
D5C7    ADC     &0301,X ;
D5CA    STA     &0321   ;
D5CD    RTS             ;return

*************************************************************************
*                                                                       *
*       OSWORD 13 read last two graphic cursor positions                *
*                                                                       *
*************************************************************************
                        ;
D5CE    LDA     #&03    ;A=3
D5D0    JSR     &D5D5   ;
D5D3    LDA     #&07    ;A=7
D5D5    PHA             ;Save A
D5D6    JSR     &CDE2   ;exchange last 2 graphics cursor coordinates with
                        ;current coordinates
D5D9    JSR     &D1B8   ;convert to external coordinates
D5DC    LDX     #&03    ;X=3
D5DE    PLA             ;save A
D5DF    TAY             ;Y=A
D5E0    LDA     &0310,X ;get graphics coordinate
D5E3    STA     (&F0),Y ;store it in OS buffer
D5E5    DEY             ;decrement Y and X
D5E6    DEX             ;
D5E7    BPL     &D5E0   ;if +ve do it again
D5E9    RTS             ;then Exit
                        ;


*************************************************************************
*                                                                       *
*       PLOT Fill triangle routine                                      *
*                                                                       *
*************************************************************************
```

```
D5EA    LDX     #&20    ;X=&20
D5EC    LDY     #&3E    ;Y=&3E
D5EE    JSR     &D47C   ;copy 300/7+X to 300/7+Y
                        ;this gets XY data parameters and current graphics
                        ;cursor position
D5F1    JSR     &D632   ;exchange 320/3 with 324/7 if 316/7=<322/3
D5F4    LDX     #&14    ;X=&14
D5F6    LDY     #&24    ;Y=&24
D5F8    JSR     &D636   ;
D5FB    JSR     &D632   ;

D5FE    LDX     #&20    ;
D600    LDY     #&2A    ;
D602    JSR     &D411   ;calculate 032A/B-(324/5-320/1)
D605    LDA     &032B   ;and store
D608    STA     &0332   ;result

D60B    LDX     #&28    ;set pointers
D60D    JSR     &D459   ;
D610    LDY     #&2E    ;

D612    JSR     &D0DE   ;copy 320/3 32/31
D615    JSR     &CDE2   ;exchange 314/7 with 324/7
D618    CLC             ;
D619    JSR     &D658   ;execute fill routine

D61C    JSR     &CDE2   ;
D61F    LDX     #&20    ;
D621    JSR     &CDE4   ;
D624    SEC             ;
D625    JSR     &D658   ;

D628    LDX     #&3E    ;;X=&3E
D62A    LDY     #&20    ;;Y=&20
D62C    JSR     &D47C   ;;copy 300/7+X to 300/7+Y
D62F    JMP     &D0D9   ;;this gets XY data parameters and current graphics
                         ;cursor position


D632    LDX     #&20    ;X=&20
D634    LDY     #&14    ;Y=&14
D636    LDA     &0302,X ;
D639    CMP     &0302,Y ;
D63C    LDA     &0303,X ;
D63F    SBC     &0303,Y ;
D642    BMI     &D657   ;if 302/3+Y>302/3+X return
D644    JMP     &CDE6   ;else swap 302/3+X with 302/3+Y

                        ;


*************************************************************************
*                                                                       *
*       OSBYTE 134  Read cursor position                                *
*                                                                       *
*************************************************************************


D647    LDA     &0318   ;read current text cursor (X)
D64A    SEC             ;set carry
D64B    SBC     &0308   ;subtract left hand column of current text window
D64E    TAX             ;X=A
D64F    LDA     &0319   ;get current text cursor (Y)
D652    SEC             ;
```

```
D653    SBC     &030B   ;suptract top row of current window
D656    TAY             ;Y=A
D657    RTS             ;and exit


                        ;PLOT routines continue
                        ;many of the following routines are just manipulations
                        ;only points of interest will be explained
D658    PHP             ;store flags
D659    LDX     #&20    ;X=&20
D65B    LDY     #&35    ;Y=&35
D65D    JSR     &D411   ;335/6=(324/5+X-320/1)
D660    LDA     &0336   ;
D663    STA     &033D   ;
D666    LDX     #&33    ;
D668    JSR     &D459   ;set pointers

D66B    LDY     #&39    ;set 339/C=320/3
D66D    JSR     &D0DE   ;
D670    SEC             ;
D671    LDA     &0322   ;
D674    SBC     &0326   ;
D677    STA     &031B   ;
D67A    LDA     &0323   ;
D67D    SBC     &0327   ;
D680    STA     &031C   ;
D683    ORA     &031B   ;check VDU queque
D686    BEQ     &D69F   ;

D688    JSR     &D6A2   ;display a line
D68B    LDX     #&33    ;
D68D    JSR     &D774   ;update pointers
D690    LDX     #&28    ;
D692    JSR     &D774   ;and again!
D695    INC     &031B   ;update VDU queque
D698    BNE     &D688   ;and if not empty do it again
D69A    INC     &031C   ;else increment next byte
D69D    BNE     &D688   ;and do it again

D69F    PLP             ;pull flags
D6A0    BCC     &D657   ;if carry clear exit
D6A2    LDX     #&39    ;
D6A4    LDY     #&2E    ;
D6A6    STX     &DE     ;
D6A8    LDA     &0300,X ;is 300/1+x<300/1+Y
D6AB    CMP     &0300,Y ;
D6AE    LDA     &0301,X ;
D6B1    SBC     &0301,Y ;
D6B4    BMI     &D6BC   ;if so D6BC
D6B6    TYA             ;else A=Y
D6B7    LDY     &DE     ;Y=&DE
D6B9    TAX             ;X=A
D6BA    STX     &DE     ;&DE=X
D6BC    STY     &DF     ;&DF=Y
D6BE    LDA     &0300,Y ;
D6C1    PHA             ;
D6C2    LDA     &0301,Y ;
D6C5    PHA             ;
D6C6    LDX     &DF     ;
D6C8    JSR     &D10F   ;check for window violations
D6CB    BEQ     &D6DA   ;
D6CD    CMP     #&02    ;
D6CF    BNE     &D70E   ;
D6D1    LDX     #&04    ;
D6D3    LDY     &DF     ;
D6D5    JSR     &D482   ;
```

```
D6D8    LDX     &DF     ;
D6DA    JSR     &D864   ;set a screen address
D6DD    LDX     &DE     ;X=&DE
D6DF    JSR     &D10F   ;check for window violations
D6E2    LSR             ;A=A/2
D6E3    BNE     &D70E   ;if A<>0 then exit
D6E5    BCC     &D6E9   ;else if C clear D6E9
D6E7    LDX     #&00    ;
D6E9    LDY     &DF     ;
D6EB    SEC             ;
D6EC    LDA     &0300,Y ;
D6EF    SBC     &0300,X ;
D6F2    STA     &DC     ;
D6F4    LDA     &0301,Y ;
D6F7    SBC     &0301,X ;
D6FA    STA     &DD     ;
D6FC    LDA     #&00    ;
D6FE    ASL             ;
D6FF    ORA     &D1     ;
D701    LDY     &DC     ;
D703    BNE     &D719   ;
D705    DEC     &DD     ;
D707    BPL     &D719   ;
D709    STA     &D1     ;
D70B    JSR     &D0F0   ;display a point
D70E    LDX     &DF     ;restore X
D710    PLA             ;and A
D711    STA     &0301,X ;store it
D714    PLA             ;get back A
D715    STA     &0300,X ;and store it
D718    RTS             ;exit
                        ;
D719    DEC     &DC     ;
D71B    TAX             ;
D71C    BPL     &D6FE   ;
D71E    STA     &D1     ;
D720    JSR     &D0F0   ;display a point
D723    LDX     &DC     ;
D725    INX             ;
D726    BNE     &D72A   ;
D728    INC     &DD     ;
D72A    TXA             ;
D72B    PHA             ;
D72C    LSR     &DD     ;
D72E    ROR             ;
D72F    LDY     &0361   ;number of pixels/byte
D732    CPY     #&03    ;if 3 mode = goto D73B
D734    BEQ     &D73B   ;
D736    BCC     &D73E   ;else if <3 mode 2 goto D73E
D738    LSR     &DD     ;else rotate bottom bit of &DD
D73A    ROR             ;into Accumulator

D73B    LSR     &DD     ;rotate bottom bit of &DD
D73D    LSR             ;into Accumulator
D73E    LDY     &031A   ;Y=line in current graphics cell containing current
                        ;point
D741    TAX             ;X=A
D742    BEQ     &D753   ;
D744    TYA             ;Y=Y-8
D745    SEC             ;
D746    SBC     #&08    ;
D748    TAY             ;

D749    BCS     &D74D   ;
D74B    DEC     &D7     ;decrement byte of top line off current graphics cell
D74D    JSR     &D104   ;display a point
```

```
D750    DEX             ;
D751    BNE     &D744   ;
D753    PLA             ;
D754    AND     &0361   ;pixels/byte
D757    BEQ     &D70E   ;
D759    TAX             ;
D75A    LDA     #&00    ;A=0
D75C    ASL             ;
D75D    ORA     &0363   ;or with right colour mask
D760    DEX             ;
D761    BNE     &D75C   ;
D763    STA     &D1     ;store as byte mask
D765    TYA             ;Y=Y-8
D766    SEC             ;
D767    SBC     #&08    ;
D769    TAY             ;
D76A    BCS     &D76E   ;if carry clear
D76C    DEC     &D7     ;decrement byte of top line off current graphics cell
D76E    JSR     &D0F3   ;display a point
D771    JMP     &D70E   ;and exit via D70E

D774    INC     &0308,X ;
D777    BNE     &D77C   ;
D779    INC     &0309,X ;
D77C    SEC             ;
D77D    LDA     &0300,X ;
D780    SBC     &0302,X ;
D783    STA     &0300,X ;
D786    LDA     &0301,X ;
D789    SBC     &0303,X ;
D78C    STA     &0301,X ;
D78F    BPL     &D7C1   ;
D791    LDA     &030A,X ;
D794    BMI     &D7A1   ;
D796    INC     &0306,X ;
D799    BNE     &D7AC   ;
D79B    INC     &0307,X ;
D79E    JMP     &D7AC   ;
D7A1    LDA     &0306,X ;
D7A4    BNE     &D7A9   ;
D7A6    DEC     &0307,X ;
D7A9    DEC     &0306,X ;
D7AC    CLC             ;
D7AD    LDA     &0300,X ;
D7B0    ADC     &0304,X ;
D7B3    STA     &0300,X ;
D7B6    LDA     &0301,X ;
D7B9    ADC     &0305,X ;
D7BC    STA     &0301,X ;
D7BF    BMI     &D791   ;
D7C1    RTS             ;
                        ;
                        ;


*************************************************************************
*                                                                       *
*       OSBYTE 135  Read character at text cursor position              *
*                                                                       *
*************************************************************************

D7C2    LDY     &0360   ;get number of logical colours
D7C5    BNE     &D7DC   ;if Y<>0 mode <>7 so D7DC
D7C7    LDA     (&D8),Y ;get address of top scan line of current text chr
D7C9    LDY     #&02    ;Y=2
D7CB    CMP     &C4B7,Y ;compare with conversion table
```

```
D7CE    BNE     &D7D4   ;if not equal D7d4
D7D0    LDA     &C4B6,Y ;else get next lower byte from table
D7D3    DEY             ;Y=Y-1
D7D4    DEY             ;Y=Y-1
D7D5    BPL     &D7CB   ;and if +ve do it again
D7D7    LDY     &0355   ;Y=current screen mode
D7DA    TAX             ;return with character in X
D7DB    RTS             ;
                        ;
D7DC    JSR     &D808   ;set up copy of the pattern bytes at text cursor
D7DF    LDX     #&20    ;X=&20
D7E1    TXA             ;A=&20
D7E2    PHA             ;Save it
D7E3    JSR     &D03E   ;get pattern address for code in A
D7E6    PLA             ;get back A
D7E7    TAX             ;and X
D7E8    LDY     #&07    ;Y=7
D7EA    LDA     &0328,Y ;get byte in pattern copy
D7ED    CMP     (&DE),Y ;check against pattern source
D7EF    BNE     &D7F9   ;if not the same D7F9
D7F1    DEY             ;else Y=Y-1
D7F2    BPL     &D7EA   ;and if +ve D7EA
D7F4    TXA             ;A=X
D7F5    CPX     #&7F    ;is X=&7F (delete)
D7F7    BNE     &D7D7   ;if not D7D7
D7F9    INX             ;else X=X+1
D7FA    LDA     &DE     ;get byte lo address
D7FC    CLC             ;clear carry
D7FD    ADC     #&08    ;add 8
D7FF    STA     &DE     ;store it
D801    BNE     &D7E8   ;and go back to check next character if <>0

D803    TXA             ;A=X
D804    BNE     &D7E1   ;if <>0 D7E1
D806    BEQ     &D7D7   ;else D7D7


***************** set up pattern copy **********************************

D808    LDY     #&07    ;Y=7

D80A    STY     &DA     ;&DA=Y
D80C    LDA     #&01    ;A=1
D80E    STA     &DB     ;&DB=A
D810    LDA     &0362   ;A=left colour mask
D813    STA     &DC     ;store an &DC
D815    LDA     (&D8),Y ;get a byte from current text character
D817    EOR     &0358   ;EOR with text background colour
D81A    CLC             ;clear carry
D81B    BIT     &DC     ;and check bits of colour mask
D81D    BEQ     &D820   ;if result =0 then D820
D81F    SEC             ;else set carry
D820    ROL     &DB     ;&DB=&DB+Carry
D822    BCS     &D82E   ;if carry now set (bit 7 DB originally set) D82E
D824    LSR     &DC     ;else  &DC=&DC/2
D826    BCC     &D81B   ;if carry clear D81B
D828    TYA             ;A=Y
D829    ADC     #&07    ;ADD ( (7+carry)
D82B    TAY             ;Y=A
D82C    BCC     &D810   ;
D82E    LDY     &DA     ;read modified values into Y and A
D830    LDA     &DB     ;
D832    STA     &0328,Y ;store copy
D835    DEY             ;and do it again
D836    BPL     &D80A   ;until 8 bytes copied
D838    RTS             ;exit
```

```
                        ;
********* pixel reading ************************************************

D839    PHA             ;store A
D83A    TAX             ;X=A
D83B    JSR     &D149   ;set up positional data
D83E    PLA             ;get back A
D83F    TAX             ;X=A
D840    JSR     &D85F   ;set a screen address after checking for window
                        ;violations
D843    BNE     &D85A   ;if A<>0 D85A to exit with A=&FF
D845    LDA     (&D6),Y ;else get top line of current graphics cell
D847    ASL             ;A=A*2 C=bit 7
D848    ROL     &DA     ;&DA=&DA+2 +C  C=bit 7 &DA
D84A    ASL     &D1     ;byte mask=bM*2 +carry from &DA
D84C    PHP             ;save flags
D84D    BCS     &D851   ;if carry set D851
D84F    LSR     &DA     ;else restore &DA with bit '=0
D851    PLP             ;pull flags
D852    BNE     &D847   ;if Z set D847
D854    LDA     &DA     ;else A=&DA AND number of colours in current mode -1
D856    AND     &0360   ;
D859    RTS             ;then exit
                        ;
D85A    LDA     #&FF    ;A=&FF
D85C    RTS             ;exit
                        ;


**********: check for window violations and set up screen address **********

D85D    LDX     #&20    ;X=&20
D85F    JSR     &D10F   ;
D862    BNE     &D85C   ;if A<>0 there is a window violation so D85C
D864    LDA     &0302,X ;else set up graphics scan line variable
D867    EOR     #&FF    ;
D869    TAY             ;
D86A    AND     #&07    ;
D86C    STA     &031A   ;in 31A
D86F    TYA             ;A=Y
D870    LSR             ;A=A/2
D871    LSR             ;A=A/2
D872    LSR             ;A=A/2
D873    ASL             ;A=A*2 this gives integer value bit 0 =0
D874    TAY             ;Y=A
D875    LDA     (&E0),Y ;get high byte of offset from screen RAM start
D877    STA     &DA     ;store it
D879    INY             ;Y=Y+1
D87A    LDA     (&E0),Y ;get lo byte
D87C    LDY     &0356   ;get screen map type
D87F    BEQ     &D884   ;if 0 (modes 0,1,2) goto D884
D881    LSR     &DA     ;else &DA=&DA/2
D883    ROR             ;and A=A/2 +C if set
                        ;so 2 byte offset =offset/2

D884    ADC     &0350   ;add screen top left hand corner lo
D887    STA     &D6     ;store it
D889    LDA     &DA     ;get high  byte
D88B    ADC     &0351   ;add top left hi
D88E    STA     &D7     ;store it
D890    LDA     &0301,X ;
D893    STA     &DA     ;
D895    LDA     &0300,X ;
D898    PHA             ;
D899    AND     &0361   ;and then Add pixels per byte-1
D89C    ADC     &0361   ;
```

```
D89F    TAY             ;Y=A
D8A0    LDA     &C406,Y ;A=&80 /2^Y using look up table
D8A3    STA     &D1     ;store it
D8A5    PLA             ;get back A
D8A6    LDY     &0361   ;Y=&number of pixels/byte
D8A9    CPY     #&03    ;is Y=3 (modes 1,6)
D8AB    BEQ     &D8B2   ;goto D8B2
D8AD    BCS     &D8B5   ;if mode =1 or 4 D8B5
D8AF    ASL             ;A/&DA =A/&DA *2
D8B0    ROL     &DA     ;

D8B2    ASL             ;
D8B3    ROL     &DA     ;

D8B5    AND     #&F8    ;clear bits 0-2
D8B7    CLC             ;clear carry
D8B8    ADC     &D6     ;add A/&DA to &D6/7
D8BA    STA     &D6     ;
D8BC    LDA     &DA     ;
D8BE    ADC     &D7     ;
D8C0    BPL     &D8C6   ;if result +ve D8C6
D8C2    SEC             ;else set carry
D8C3    SBC     &0354   ;and subtract screen memory size making it wrap round

D8C6    STA     &D7     ;store it in &D7
D8C8    LDY     &031A   ;get line in graphics cell containing current graphics
D8CB    LDA     #&00    ;point  A=0
D8CD    RTS             ;And exit
                        ;
D8CE    PHA             ;Push A
D8CF    LDA     #&A0    ;A=&A0
D8D1    LDX     &026A   ;X=number of items in VDU queque
D8D4    BNE     &D916   ;if not 0 D916
D8D6    BIT     &D0     ;else check VDU status byte
D8D8    BNE     &D916   ;if either VDU is disabled or plot to graphics
                        ;cursor enabled then D916
D8DA    BVS     &D8F5   ;if cursor editing enabled D8F5
D8DC    LDA     &035F   ;else get 6845 register start setting
D8DF    AND     #&9F    ;clear bits 5 and 6
D8E1    ORA     #&40    ;set bit 6 to modify last cursor size setting
D8E3    JSR     &C954   ;change write cursor format
D8E6    LDX     #&18    ;X=&18
D8E8    LDY     #&64    ;Y=&64
D8EA    JSR     &D482   ;set text input cursor from text output cursor
D8ED    JSR     &CD7A   ;modify character at cursor poistion
D8F0    LDA     #&02    ;A=2
D8F2    JSR     &C59D   ;bit 1 of VDU status is set to bar scrolling


D8F5    LDA     #&BF    ;A=&BF
D8F7    JSR     &C5A8   ;bit 6 of VDU status =0
D8FA    PLA             ;Pull A
D8FB    AND     #&7F    ;clear hi bit (7)
D8FD    JSR     &C4C0   ;entire VDU routine !!
D900    LDA     #&40    ;A=&40
D902    JMP     &C59D   ;exit


D905    LDA     #&20    ;A=&20
D907    BIT     &D0     ;if bit 6 cursor editing is set
D909    BVC     &D8CB   ;
D90B    BNE     &D8CB   ;or bit 5 is set exit &D8CB
D90D    JSR     &D7C2   ;read a character from the screen
D910    BEQ     &D917   ;if A=0 on return exit via D917
D912    PHA             ;else store A
D913    JSR     &C664   ;perform cursor right
```

```
D916    PLA             ;restore A
D917    RTS             ;and exit
                        ;
D918    LDA     #&BD    ;zero bits 2 and 6 of VDU status
D91A    JSR     &C5A8   ;
D91D    JSR     &C951   ;set normal cursor
D920    LDA     #&0D    ;A=&0D
D922    RTS             ;and return
                        ;this is response of CR as end of edit line


*************************************************************************
*                                                                       *
*       OSBYTE 132  Read bottom of display RAM                          *
*                                                                       *
*************************************************************************
                        ;
D923    LDX     &0355   ;get current screen mode


*************************************************************************
*                                                                       *
*       OSBYTE 133  Read lowest address for given mode                  *
*                                                                       *
*************************************************************************

D926    TXA             ;A=X
D927    AND     #&07    ;MOD 7!
D929    TAY             ;Y=A
D92A    LDX     &C440,Y ;X=get RAM size key
D92D    LDA     &C45E,X ;A=high byte of start address
D930    LDX     #&00    ;X=0
D932    BIT     &028E   ;get available RAM
D935    BMI     &D93E   ;if bit 7 set then 32k so D93E
D937    AND     #&3F    ;AND A with &3F
D939    CPY     #&04    ;if Y<4
D93B    BCS     &D93E   ;then D93E
D93D    TXA             ;else A=0 to return null value
D93E    TAY             ;Y=A
D93F    RTS             ;and return


*************************************************************************
*                                                                       *
*       DEFAULT VECTOR TABLE                                            *
*                                                                       *
*************************************************************************

D940    DB      10,E3   ;&E310 = USERV                                  &200
D942    DB      54,DC   ;&DC54 = BRKV                                   &202
D944    DB      93,DC   ;&DC93 = IRQ1V                                  &204
D946    DB      89,DE   ;&DE89 = IRQ2V                                  &206
D948    DB      89,DF   ;&DF89 = CLIV                                   &208
D94A    DB      72,E7   ;&E772 = BYTEV                                  &20A
D94C    DB      EB,E7   ;&E7EB = WORDV                                  &20C
D94E    DB      A4,E0   ;&E0A4 = WRCHV                                  &20E
D950    DB      C5,DE   ;&DEC5 = RDCHV                                  &210
D952    DB      7D,F2   ;&F27D = FILEV                                  &212
D954    DB      8E,F1   ;&F18E = ARGSV                                  &214
D956    DB      C9,F4   ;&F4C9 = BGETV                                  &216
D958    DB      29,F5   ;&F529 = BPUTV                                  &218
D95A    DB      A6,FF   ;&FFA6 = GBPBV                                  &21A
D95C    DB      CA,F3   ;&F3CA = FINDV                                  &21C
D95E    DB      B1,F1   ;&F1B1 = FSCV                                   &21E
```

```
D960    DB      A6,FF   ;&FFA6 = EVNTV                                  &220
D962    DB      A6,FF   ;&FFA6 = UPTV                                  &222
D964    DB      A6,FF   ;&FFA6 = NETV                                  &224
D966    DB      A6,FF   ;&FFA6 = VDUV                                  &226
D968    DB      02,EF   ;&EF02 = KEYV                                  &228
D96A    DB      B3,E4   ;&E4B3 = INSBV                                 &22A
D96C    DB      64,E4   ;&E464 = REMVB                                 &22C
D96E    DB      D1,E1   ;&E1D1 = CNPV                                  &22E
D970    DB      A6,FF   ;&FFA6 = IND1V                                 &230
D972    DB      A6,FF   ;&FFA6 = IND2V                                 &232
D974    DB      A6,FF   ;&FFA6 = IND3V                                 &234


*************************************************************************
*                                                                       *
*       MOS VARIABLES DEFAULT SETTINGS                                  *
*************************************************************************

*read/written by Osbytes &A6 to &FC
*addresses &236 to &28C =address -&D740

D976    DB      90,01   ;Mos variables address  (address to Add to osbyte
                        ;number) in lo hi format (&190)                 &236

D978    DB      9F,0D   ;Rom pointer address for indirecting into ROMS
                        ;=09DF (lo hi format)                           &238

D97A    DB      A1,02   ;ROM information table address (&2A1)           &23A

D97C    DB      2B,F0   ;Key translation table address (&F02B)          &23C

D97E    DB      00,03   ;VDU variables start 0300                      &23E

D980    DB      00      ;CFS/Vertical sync Timeout counter             &240
D981    DB      00      ;current input buffer number                   &241
D982    DB      FF      ;keyboard interrupt processing flag            &242
D983    DB      00      ;primary OSHWM  (default page)                 &243
D984    DB      00      ;current OSHWM  (PAGE)                         &244
D985    DB      01      ;RS423 Mode                                    &245
D986    DB      00      ;character defininition explosion switch       &246
D987    DB      00      ;Filing system flag ROM=2 CFS=0                &247
D988    DB      00      ;current Video ULA control register            &248
D989    DB      00      ;current pallette setting                      &249
D98A    DB      00      ;number of ROM enabled at last BRK             &24A
D98B    DB      FF      ;number of BASIC ROM                           &24B
D98C    DB      04      ;current ADC channel number                    &24C
D98D    DB      04      ;maximum ADC channel number                    &24D
D98E    DB      00      ;ADC conversion type 0 or 0C=12 bit 8=8 Bit    &24E
D98F    DB      FF      ;RS423 busy flag (bit 7 = 0 =busy)             &24F


D990    DB      56      ;curent ACIA control register setting          &250
D991    DB      19      ;flash counter                                 &251
D992    DB      19      ;mark period count                             &252
D993    DB      19      ;space period count                            &253
D994    DB      32      ;keyboard Auto-repeat delay                    &254
D995    DB      08      ;keyboard Auto-repeat rate                     &255
D996    DB      00      ;*EXEC file handle  (0 -not allocated)         &256
D997    DB      00      ;*SPOOL file handle (0 -not allocated)         &257
D998    DB      00      ;bit 0 Escape enable/disable                   &258
                        ;bit 1 BREAK normal/clear memory
D999    DB      00      ;Econet disable keyboard flag                  &259
D99A    DB      20      ;keyboard status        bit 3=1 shift pressed  &25A
                                                bit 4=0 caps  lock
                                                bit 5=0 shift lock
                                                bit 6=1 control bit
```

```
                                                    bit 7=1 shift enabled
D99B    DB      09      ;buffer space left at buffer full signal        &25B
D99C    DB      00      ;RS423 input suppression flag                   &25C
D99D    DB      00      ;cassette/RS423 flag  (0=CFS, &40=RS423)        &25D
D99E    DB      00      ;Econet OS call interception flag (bit 7)       &25E
D99F    DB      00      ;Econet OSRDCH interception flag (bit 7)        &25F

D9A0    DB      00      ;Econet OSWRCH interception flag (bit 7)        &260
D9A1    DB      50      ;speech enable/disable flac (50/20)             &261
D9A2    DB      00      ;sound output enable flag                       &262
D9A3    DB      03      ;BELL channel number                            &263
D9A4    DB      90      ;BELL amplitude/Envelope number                 &264
D9A5    DB      64      ;BELL frequency                                 &265
D9A6    DB      06      ;BELL duration                                  &266
D9A7    DB      81      ;bit 7=1 ignore start up message                &267
                        ;bit 0=1 ignore RFS !BOOT error

D9A8    DB      00      ;length of KEY string                           &268
D9A9    DB      00      ;PRINT line counter                             &269
D9AA    DB      00      ;number of items in VDU queque (2s complement   &26A
                        ;of number required)
D9AB    DB      09      ;TAB key value                                  &26B
D9AC    DB      1B      ;ESCAPE Character                               &26C

        ;The following are input buffer code interpretation bytes for
        ;keys returning the following keys C0-FF are available via
        ;keypads only!
        ;0=ignore key
        ;1=expand as normal key
        ;2-FF add to base for ASCII code

D9AD    DB      01      ;C0-CF                                          &26D
D9AE    DB      D0      ;D0-DF                                          &26E
D9AF    DB      E0      ;E0-EF                                          &26F
D9B0    DB      F0      ;F0-FF                                          &270
D9B1    DB      01      ;80-8F                                          &271
D9B2    DB      80      ;90-9F                                          &272
D9B3    DB      90      ;A0-AF                                          &273
D9B4    DB      00      ;B0-BF                                          &274


D9B5    DB      00      ;ESCAPE key status (0=ESC, 1,=ASCII)            &275
D9B6    DB      00      ;ESCAPE action                                  &276
D9B7    DB      FF      ;USER 6522 Bit IRQ mask                         &277
D9B8    DB      FF      ;6850 ACIA Bit IRQ bit mask                     &278
D9B9    DB      FF      ;System 6522 IRQ bit mask                       &279
D9BA    DB      00      ;Tube prescence flag                            &27A
D9BB    DB      00      ;speech processor prescence flag                &27B
D9BC    DB      00      ;character destination status                   &27C
D9BD    DB      00      ;cursor editing status                          &27D

****************** Warm reset high water mark ***************************

D9BE    DB      00      ;unused                                         &27E
D9BF    DB      00      ;unused                                         &27F
D9C0    DB      00      ;country code                                   &280
D9C1    DB      00      ;user flag                                      &281
D9C2    DB      64      ;serial ULA control register setting            &282
D9C3    DB      05      ;current system clock store pointer             &283
D9C4    DB      FF      ;soft key status (unstable)                     &284
D9C5    DB      01      ;printer destination                            &285
D9C6    DB      0A      ;printer ignore character                       &286

********* COLD RESET High water mark ************************************

D9C7    DB      00      ;user BREAK routine address JMP                 &288
```

```
D9C8    DB      00      ;user BREAK routine address lo                  &288
D9C9    DB      00      ;user BREAK routine address hi                  &289
D9CA    DB      00      ;unused                                         &28A
D9CB    DB      00      ;unused                                         &28B
D9CC    DB      FF      ;current language rom no.                       &28C

********** RESET High Water mark for Power up ****************************
;later flags dealt with in routines




*******************************************************************************
*******************************************************************************
**                                                                           **
**                                                                           **
**      RESET (BREAK) ENTRY POINT                                            **
**                                                                           **
**      Power up Enter with nothing set, 6522 System VIA IER bits            **
**      0 to 6 will be clear                                                 **
**                                                                           **
**      BREAK IER bits 0 to 6 one or more will be set 6522 IER               **
**      not reset by BREAK                                                   **
**                                                                           **
*******************************************************************************
*******************************************************************************

D9CD    LDA     #&40    ;set NMI first instruction to RTI
D9CF    STA     &0D00   ;NMI ram start

D9D2    SEI             ;disable interrupts just in case
D9D3    CLD             ;clear decimal flag
D9D4    LDX     #&FF    ;reset stack to where it should be
D9D6    TXS             ;(&1FF)
D9D7    LDA     &FE4E   ;read interupt enable register of the system VIA
D9DA    ASL             ;shift bit 7 into carry
D9DB    PHA             ;save what's left
D9DC    BEQ     &D9E7   ;if Power up A=0 so D9E7
D9DE    LDA     &0258   ;else if BREAK pressed read BREAK Action flags (set by
                        ;*FX200,n)
D9E1    LSR             ;divide by 2
D9E2    CMP     #&01    ;if (bit 1 not set by *FX200)
D9E4    BNE     &DA03   ;then &DA03
D9E6    LSR             ;divide A by 2 again (A=0 if *FX200,2/3 else A=n/4


********** clear store routine ****************************************

D9E7    LDX     #&04    ;get page to start clearance from (4)
D9E9    STX     &01     ;store it in ZP 01
D9EB    STA     &00     ;store A at 00

D9ED    TAY             ;and in Y to set loop counter

D9EE    STA     (&00),Y ;clear store
D9F0    CMP     &01     ;until address &01 =0
D9F2    BEQ     &D9FD   ;
D9F4    INY             ;increment pointer
D9F5    BNE     &D9EE   ;if not zero loop round again
D9F7    INY             ;else increment again (Y=1) this avoids overwriting
                        ;RTI instruction at &D00
D9F8    INX             ;increment X
D9F9    INC     &01     ;increment &01
D9FB    BPL     &D9EE   ;loop until A=&80 then exit
```

```
                        ;note that RAM addressing for 16k loops around so
                        ;&4000=&00 hence checking &01 for 00.  This avoids
                        ;overwriting zero page on BREAK


D9FD    STX     &028E   ;writes marker for available RAM 40 =16k,80=32
DA00    STX     &0284   ;write soft key consistency flag

**+********** set up system VIA ***************************************

DA03    LDX     #&0F    ;set PORT B to output on bits 0-3 Input 4-7
DA05    STX     &FE42   ;



*************************************************************************
*                                                                       *
*        set addressable latch IC 32 for peripherals via PORT B         *
*                                                                       *
*       ;bit 3 set sets addressed latch high adds 8 to VIA address      *
*       ;bit 3 reset sets addressed latch low                           *
*                                                                       *
*       Peripheral              VIA bit 3=0             VIA bit 3=1     *
*                                                                       *
*       Sound chip              Enabled                 Disabled        *
*       speech chip (RS)        Low                     High            *
*       speech chip (WS)        Low                     High            *
*       Keyboard Auto Scan      Disabled                Enabled         *
*       C0 address modifier     Low                     High            *
*       C1 address modifier     Low                     High            *
*       Caps lock  LED          ON                      OFF             *
*       Shift lock LED          ON                      OFF             *
*                                                                       *
*       C0 & C1 are involved with hardware scroll screen address        *
*************************************************************************

        ;X=&F on entry

DA08    DEX             ;loop start
DA09    STX     &FE40   ;write latch IC32
DA0C    CPX     #&09    ;is it 9
DA0E    BCS     &DA08   ;if so go back and do it again
                        ;X=8 at this point
                        ;Caps lock On, SHIFT lock undetermined
                        ;Keyboard Autoscan on
                        ;sound disabled (may still sound)
DA10    INX             ;X=9
DA11    TXA             ;A=X
DA12    JSR     &F02A   ;interrogate keyboard
DA15    CPX     #&80    ;for keyboard links 9-2 and CTRL key (1)
DA17    ROR     &FC     ;rotate MSB into bit 7 of &FC




DA19    TAX             ;get back value of X for loop
DA1A    DEX             ;decrement it
DA1B    BNE     &DA11   ;and if >0 do loop again
                        ; on exit if Carry set link 3 made
                        ;link 2 = bit 0 of &FC and so on
                        ;if CTRL pressed bit 7 of &FC=1
                        ;X=0
DA1D    STX     &028D   ;clear last BREAK flag
DA20    ROL     &FC     ;CTRL is now in carry &FC is keyboard links
```

```
DA22    JSR     &EEEB   ;set LEDs carry on entry  bit 7 of A on exit
DA25    ROR             ;get carry back into carry flag

****** set up page 2 ************************************************

DA26    LDX     #&9C    ;
DA28    LDY     #&8D    ;
DA2A    PLA             ;get back A from &D9DB
DA2B    BEQ     &DA36   ;if A=0 power up reset so DA36 with X=&9C Y=&8D
DA2D    LDY     #&7E    ;else Y=&7E
DA2F    BCC     &DA42   ;and if not CTRL-BREAK DA42 WARM RESET
DA31    LDY     #&87    ;else Y=&87 COLD RESET
DA33    INC     &028D   ;&28D=1

DA36    INC     &028D   ;&28D=&28D+1
DA39    LDA     &FC     ;get keyboard links set
DA3B    EOR     #&FF    ;invert
DA3D    STA     &028F   ;and store at &28F
DA40    LDX     #&90    ;X=&90


**********: set up page 2 ************************************************

        ;on entry        &28D=0 Warm reset, X=&9C, Y=&7E
                         ;&28D=1 Power up  , X=&90, Y=&8D
                         ;&28D=2 Cold reset, X=&9C, Y=&87

DA42    LDA     #&00    ;A=0
DA44    CPX     #&CE    ;zero &200+X to &2CD
DA46    BCC     &DA4A   ;
DA48    LDA     #&FF    ;then set &2CE to &2FF to &FF
DA4A    STA     &0200,X ;
DA4D    INX             ;
DA4E    BNE     &DA44   ;
                         ;A=&FF X=0
DA50    STA     &FE63   ;set port A of user via to all outputs (printer out)

DA53    TXA             ;A=0
DA54    LDX     #&E2    ;X=&E2
DA56    STA     &00,X   ;zero zeropage &E2 to &FF
DA58    INX             ;
DA59    BNE     &DA56   ;X=0

DA5B    LDA     &D93F,Y ;copy data from &D93F+Y
DA5E    STA     &01FF,Y ;to &1FF+Y
DA61    DEY             ;until
DA62    BNE     &DA5B   ;1FF+Y=&200

DA64    LDA     #&62    ;A=&62
DA66    STA     &ED     ;store in &ED
DA68    JSR     &FB0A   ;set up ACIA
                         ;X=0
************** clear interrupt and enable registers of Both VIAs ********

DA6B    LDA     #&7F    ;
DA6D    INX             ;
DA6E    STA     &FE4D,X ;
DA71    STA     &FE6D,X ;
DA74    DEX             ;
DA75    BPL     &DA6E   ;

DA77    CLI             ;briefly allow interrupts to clear anything pending
DA78    SEI             ;disallow again N.B. All VIA IRQs are disabled
DA79    BIT     &FC     ;if bit 6=1 then JSR &F055 (normally 0)
DA7B    BVC     &DA80   ;else DA80
```

```
DA7D    JSR     &F055   ;F055 JMP (&FDFE) probably causes a BRK unless
                        ;hardware there redirects it.
                        ;
DA80    LDX     #&F2    ;enable interrupts 1,4,5,6 of system VIA
DA82    STX     &FE4E   ;
                        ;0      Keyboard enabled as needed
                        ;1      Frame sync pulse
                        ;4      End of A/D conversion
                        ;5      T2 counter (for speech)
                        ;6      T1 counter (10 mSec intervals)
                        ;
DA85    LDX     #&04    ;set system VIA PCR
DA87    STX     &FE4C   ;
                        ;CA1 to interrupt on negative edge (Frame sync)
                        ;CA2 Handshake output for Keyboard
                        ;CB1 interrupt on negative edge (end of conversion)
                        ;CB2 Negative edge (Light pen strobe)
                        ;
DA8A    LDA     #&60    ;set system VIA ACR
DA8C    STA     &FE4B   ;
                        ;disable latching
                        ;disable shift register
                        ;T1 counter continuous interrupts
                        ;T2 counter timed interrupt

DA8F    LDA     #&0E    ;set system VIA T1 counter (Low)
DA91    STA     &FE46   ;
                        ;this becomes effective when T1 hi set

DA94    STA     &FE6C   ;set user VIA PCR
                        ;CA1 interrupt on -ve edge (Printer Acknowledge)
                        ;CA2 High output (printer strobe)
                        ;CB1 Interrupt on -ve edge (user port)
                        ;CB2 Negative edge (user port)

DA97    STA     &FEC0   ;set up A/D converter
                        ;Bits 0 & 1 determine channel selected
                        ;Bit 3=0 8 bit conversion bit 3=1 12 bit

DA9A    CMP     &FE6C   ;read user VIA IER if = &0E then DAA2 chip present
DA9D    BEQ     &DAA2   ;so goto DAA2
DA9F    INC     &0277   ;else increment user VIA mask to 0 to bar all
                        ;user VIA interrupts

DAA2    LDA     #&27    ;set T1 (hi) to &27 this sets T1 to &270E (9998 uS)
DAA4    STA     &FE47   ;or 10msec, interrupts occur every 10msec therefore
DAA7    STA     &FE45   ;

DAAA    JSR     &EC60   ;clear the sound channels

DAAD    LDA     &0282   ;read serial ULA control register
DAB0    AND     #&7F    ;zero bit 7
DAB2    JSR     &E6A7   ;and set up serial ULA

DAB5    LDX     &0284   ;get soft key status flag
DAB8    BEQ     &DABD   ;if 0 (keys OK) then DABD
DABA    JSR     &E9C8   ;else reset function keys


*************************************************************************
*                                                                       *
*       Check sideways ROMS and make catalogue                          *
*                                                                       *
*************************************************************************
```

```
                        ;X=0
DABD    JSR     &DC16   ;set up ROM latch and RAM copy to X
DAC0    LDX     #&03    ;set X to point to offset in table
DAC2    LDY     &8007   ;get copyright offset from ROM

                        ; DF0C = )C( BRK
DAC5    LDA     &8000,Y ;get first byte
DAC8    CMP     &DF0C,X ;compare it with table byte
DACB    BNE     &DAFB   ;if not the same then goto DAFB
DACD    INY             ;point to next byte
DACE    DEX             ;(s)
DACF    BPL     &DAC5   ;and if still +ve go back to check next byte

                        ;this point is reached if 5 bytes indicate valid
                        ;ROM (offset +4 in (C) string)


*************************************************************************
* Check first 1k of each ROM against higher priority Roms to ensure that*
* there are no matches, if a match found ignore lower priority ROM      *
*************************************************************************

DAD1    LDX     &F4     ;get RAM copy of ROM No. in X
DAD3    LDY     &F4     ;and Y

DAD5    INY             ;increment Y to check
DAD6    CPY     #&10    ;if ROM 15 is current ROM
DAD8    BCS     &DAFF   ;if equal or more than 16 goto &DAFF
                        ;to store catalogue byte
DADA    TYA             ;else put Y in A
DADB    EOR     #&FF    ;invert it
DADD    STA     &FA     ;and store at &FA
DADF    LDA     #&7F    ;store &7F at
DAE1    STA     &FB     ;&FB to get address &7FFF-Y

DAE3    STY     &FE30   ;set new ROM
DAE6    LDA     (&FA),Y ;Get byte
DAE8    STX     &FE30   ;switch back to previous ROM
DAEB    CMP     (&FA),Y ;and compare with previous byte called
DAED    BNE     &DAD5   ;if not the same then go back and do it again
                        ;with next rom up
DAEF    INC     &FA     ;else increment &FA to point to new location
DAF1    BNE     &DAE3   ;if &FA<>0 then check next byte
DAF3    INC     &FB     ;else inc &FB
DAF5    LDA     &FB     ;and check that it doesn't exceed
DAF7    CMP     #&84    ;&84 (1k checked)
DAF9    BCC     &DAE3   ;then check next byte(s)

DAFB    LDX     &F4     ;X=(&F4)
DAFD    BPL     &DB0C   ;if +ve then &DB0C

DAFF    LDA     &8006   ;get rom type
DB02    STA     &02A1,X ;store it in catalogue
DB05    AND     #&8F    ;check for BASIC (bit 7 not set)
DB07    BNE     &DB0C   ;if not BASIC the DB0C
DB09    STX     &024B   ;else store X at BASIC pointer


DB0C    INX             ;increment X to point to next ROM
DB0D    CPX     #&10    ;is it 15 or less
DB0F    BCC     &DABD   ;if so goto &DABD for next ROM
```