OS SERIES 8
GEOFF COX

```
*************************************************************************
*                                                                       *
*        OSBYTE &F7 (247) INTERCEPT BREAK                               *
*                                                                       *
*************************************************************************


EAD9    LDA     &0287   ;get BREAK vector code
EADC    EOR     #&4C    ;produces 0 if JMP not in &287
EADE    BNE     &EAF3   ;if not goto EAF3
EAE0    JMP     &0287   ;else jump to user BREAK code



*************************************************************************
*                                                                       *
*        OSBYTE &90 (144)   *TV                                         *
*                                                                       *
*************************************************************************

                ;X=display delay
                ;Y=interlace flag

EAE3    LDA     &0290   ;VDU vertical adjustment
EAE6    STX     &0290   ;store new value
EAE9    TAX             ;put old value in X
EAEA    TYA             ;put interlace flag in A
EAEB    AND     #&01    ;maximum value =1
EAED    LDY     &0291   ;get old value into Y
EAF0    STA     &0291   ;put new value into A
EAF3    RTS             ;and Exit
                        ;



*************************************************************************
*                                                                       *
*        OSBYTE &93 (147)  WRITE TO FRED                                *
*                                                                       *
*************************************************************************

                        ;X is offset within page
                        ;Y is byte to write
EAF4    TYA             ;
EAF5    STA     &FC00,X ;
EAF8    RTS             ;



*************************************************************************
*                                                                       *
*        OSBYTE &95 (149)  WRITE TO JIM                                 *
*                                                                       *
*************************************************************************

                        ;X is offset within page
                        ;Y is byte to write
                        ;
EAF9    TYA             ;
EAFA    STA     &FD00,X ;
```

```
EAFD    RTS             ;
                        ;



*************************************************************************
*                                                                       *
*        OSBYTE &97 (151)  WRITE TO SHEILA                              *
*                                                                       *
*************************************************************************

                        ;X is offset within page
                        ;Y is byte to write

EAFE    TYA             ;
EAFF    STA     &FE00,X ;
EB02    RTS             ;
                        ;


****************** Silence a sound channel ******************************
                        ;X=channel number

EB03    LDA     #&04    ;mark end of release phase
EB05    STA     &0808,X ;to channel X
EB08    LDA     #&C0    ;load code for zero volume

****** if sound not disabled set sound generator volume ******************

EB0A    STA     &0804,X ;store A to give basic sound level of Zero
EB0D    LDY     &0262   ;get sound output/enable flag
EB10    BEQ     &EB14   ;if sound enabled goto EB14
EB12    LDA     #&C0    ;else load zero sound code
EB14    SEC             ;set carry
EB15    SBC     #&40    ;subtract &40
EB17    LSR             ;divide by 8
EB18    LSR             ;to get into bits 0 - 3
EB19    LSR             ;
EB1A    EOR     #&0F    ;invert bits 0-3
EB1C    ORA     &EB3C,X ;get channel number into top nybble
EB1F    ORA     #&10    ;


EB21    PHP             ;

EB22    SEI             ;disable interrupts
EB23    LDY     #&FF    ;System VIA port A all outputs
EB25    STY     &FE43   ;set
EB28    STA     &FE4F   ;output A on port A
EB2B    INY             ;Y=0
EB2C    STY     &FE40   ;enable sound chip
EB2F    LDY     #&02    ;set and
EB31    DEY             ;execute short delay
EB32    BNE     &EB31   ;
EB34    LDY     #&08    ;then disable sound chip again
EB36    STY     &FE40   ;
EB39    LDY     #&04    ;set delay
EB3B    DEY             ;and loop delay
EB3C    BNE     &EB3B   ;
EB3E    PLP             ;get back flags
EB3F    RTS             ;and exit

*******: Sound parameters look up table **********************************

EB40    DB      &E0,&C0,&A0,&80
```

```
EB44    JMP     &EC59   ;just to allow relative branches in early part
                        ;of sound interrupt routine


*************************************************************************
*                                                                       *
*       PROCESS SOUND INTERRUPT                                         *
*************************************************************************

EB47    LDA     #&00    ;
EB49    STA     &083B   ;zero number of channels on hold for sync
EB4C    LDA     &0838   ;get number of channels required for sync
EB4F    BNE     &EB57   ;if this <>0 then EB57
EB51    INC     &083B   ;else number of chanels on hold for sync =1
EB54    DEC     &0838   ;number of channels required for sync =255

EB57    LDX     #&08    ;set loop counter
EB59    DEX             ;loop
EB5A    LDA     &0800,X ;get value of &800 +offset (sound queue occupancy)
EB5D    BEQ     &EB44   ;if 0 goto EC59 no sound this channel
EB5F    LDA     &02CF,X ;else get buffer busy flag
EB62    BMI     &EB69   ;if negative (buffer empty) goto EB69
EB64    LDA     &0818,X ;else if duration count not zer0
EB67    BNE     &EB6C   ;goto EB6C
EB69    JSR     &EC6B   ;check and pick up new sound if required
EB6C    LDA     &0818,X ;if duration count 0
EB6F    BEQ     &EB84   ;goto EB84
EB71    CMP     #&FF    ;else if it is &FF (infinite duration)
EB73    BEQ     &EB87   ;go onto EB87
EB75    DEC     &081C,X ;decrement 10 mS count
EB78    BNE     &EB87   ;and if 0
EB7A    LDA     #&05    ;reset to 5
EB7C    STA     &081C,X ;to give 50 mSec delay
EB7F    DEC     &0818,X ;and decrement main counter
EB82    BNE     &EB87   ;if not zero then EB87
EB84    JSR     &EC6B   ;else check and get nw sound
EB87    LDA     &0824,X ;if step progress counter is 0 no envelope involved
EB8A    BEQ     &EB91   ;so jump to EB91
EB8C    DEC     &0824,X ;else decrement it
EB8F    BNE     &EB44   ;and if not zero go on to EC59
EB91    LDY     &0820,X ;get  envelope data offset from (8C0)
EB94    CPY     #&FF    ;if 255 no envelope set so
EB96    BEQ     &EB44   ;goto EC59
EB98    LDA     &08C0,Y ;else get get step length
EB9B    AND     #&7F    ;zero repeat bit
EB9D    STA     &0824,X ;and store it
EBA0    LDA     &0808,X ;get phase counter
EBA3    CMP     #&04    ;if release phase completed
EBA5    BEQ     &EC07   ;goto EC07
EBA7    LDA     &0808,X ;else start new step by getting phase
EBAA    CLC             ;
EBAB    ADC     &0820,X ;add it to interval multiplier
EBAE    TAY             ;transfer to Y
EBAF    LDA     &08CB,Y ;and get target value base for envelope
EBB2    SEC             ;
EBB3    SBC     #&3F    ;
EBB5    STA     &083A   ;store modified number as current target amplitude
EBB8    LDA     &08C7,Y ;get byte from envelope store
EBBB    STA     &0839   ;store as current amplitude step
EBBE    LDA     &0804,X ;get base volumelevel
EBC1    PHA             ;save it
EBC2    CLC             ;clear carry
EBC3    ADC     &0839   ;add to current amplitude step
EBC6    BVC     &EBCF   ;if no overflow
```

```
EBC8    ROL             ;double it Carry = bit 7
EBC9    LDA     #&3F    ;if bit =1 A=&3F
EBCB    BCS     &EBCF   ;into &EBCF
EBCD    EOR     #&FF    ;else toggle bits (A=&C0)

                        ;at this point the BASIC volume commands are converted
                        ; &C0 (0) to &38 (-15) 3 times, In fact last 3 bits
                        ;are ignored so &3F represents -15

EBCF    STA     &0804,X ;store in current volume
EBD2    ROL             ;multiply by 2
EBD3    EOR     &0804,X ;if bits 6 and 7 are equal
EBD6    BPL     &EBE1   ;goto &EBE1
EBD8    LDA     #&3F    ;if carry clear A=&3F (maximum)
EBDA    BCC     &EBDE   ;or
EBDC    EOR     #&FF    ;&C0 minimum

EBDE    STA     &0804,X ;and this is stored in current volume

EBE1    DEC     &0839   ;decrement amplitude change per step
EBE4    LDA     &0804,X ;get volume again
EBE7    SEC             ;set carry
EBE8    SBC     &083A   ;subtract target value
EBEB    EOR     &0839   ;negative value undicates correct trend
EBEE    BMI     &EBF9   ;so jump to next part
EBF0    LDA     &083A   ;else enter new phase
EBF3    STA     &0804,X ;
EBF6    INC     &0808,X ;

EBF9    PLA             ;get the old volume level
EBFA    EOR     &0804,X ;and compare with the old
EBFD    AND     #&F8    ;
EBFF    BEQ     &EC07   ;if they are the same goto EC07
EC01    LDA     &0804,X ;else set new level
EC04    JSR     &EB0A   ;via EB0A
EC07    LDA     &0810,X ;get absolute pitch value
EC0A    CMP     #&03    ;if it =3
EC0C    BEQ     &EC59   ;skip rest of loop as all sections are finished
EC0E    LDA     &0814,X ;else if 814,X is not 0 current section is not
                        ;complete
EC11    BNE     &EC3D   ;so EC3D
EC13    INC     &0810,X ;else implement a section change
EC16    LDA     &0810,X ;check if its complete
EC19    CMP     #&03    ;if not
EC1B    BNE     &EC2D   ;goto EC2D
EC1D    LDY     &0820,X ;else set A from
EC20    LDA     &08C0,Y ;&820 and &8C0 (first envelope byte)
EC23    BMI     &EC59   ;if negative there is no repeat
EC25    LDA     #&00    ;else restart section sequence
EC27    STA     &0830,X ;
EC2A    STA     &0810,X ;

EC2D    LDA     &0810,X ;get number of steps in new section
EC30    CLC             ;
EC31    ADC     &0820,X ;
EC34    TAY             ;
EC35    LDA     &08C4,Y ;
EC38    STA     &0814,X ;set in 814+X
EC3B    BEQ     &EC59   ;and if 0 then EC59

EC3D    DEC     &0814,X ;decrement
EC40    LDA     &0820,X ;and pick up rate of pitch change
EC43    CLC             ;
EC44    ADC     &0810,X ;
EC47    TAY             ;
EC48    LDA     &08C1,Y ;
```

```
EC4B    CLC             ;
EC4C    ADC     &0830,X ;add to rate of differential pitch change
EC4F    STA     &0830,X ;and save it
EC52    CLC             ;
EC53    ADC     &080C,X ;ad to base pitch
EC56    JSR     &ED01   ;and set new pitch

EC59    CPX     #&04    ;if X=4 (last channel)
EC5B    BEQ     &EC6A   ;goto EC6A (RTS)
EC5D    JMP     &EB59   ;else do loop again

EC60    LDX     #&08    ;X=7 again
EC62    DEX             ;loop
EC63    JSR     &ECA2   ;clear channel
EC66    CPX     #&04    ;if not 4
EC68    BNE     &EC62   ;do it again
EC6A    RTS             ;and return
                        ;
EC6B    LDA     &0808,X ;check for last channel
EC6E    CMP     #&04    ;is it 4 (release complete)
EC70    BEQ     &EC77   ;if so EC77
EC72    LDA     #&03    ;else mark release in progress
EC74    STA     &0808,X ;and store it
EC77    LDA     &02CF,X ;is buffer not empty
EC7A    BEQ     &EC90   ;if so EC90
EC7C    LDA     #&00    ;else mark buffer not empty
EC7E    STA     &02CF,X ;an store it

EC81    LDY     #&04    ;loop counter
EC83    STA     &082B,Y ;zero sync bytes
EC86    DEY             ;
EC87    BNE     &EC83   ;until Y=0

EC89    STA     &0818,X ;zero duration count
EC8C    DEY             ;and set sync count to
EC8D    STY     &0838   ;&FF
EC90    LDA     &0828,X ;get synchronising flag
EC93    BEQ     &ECDB   ;if its 0 then ECDB
EC95    LDA     &083B   ;else get number of channels on hold
EC98    BEQ     &ECD0   ;if 0 then ECD0
EC9A    LDA     #&00    ;else
EC9C    STA     &0828,X ;zero note length interval
EC9F    JMP     &ED98   ;and goto ED98

ECA2    JSR     &EB03   ;silence the channel
ECA5    TYA             ;Y=0 A=Y
ECA6    STA     &0818,X ;zero main count
ECA9    STA     &02CF,X ;mark buffer not empty
ECAC    STA     &0800,X ;mark channel dormant
ECAF    LDY     #&03    ;loop counter
ECB1    STA     &082C,Y ;zero sync flags
ECB4    DEY             ;
ECB5    BPL     &ECB1   ;

ECB7    STY     &0838   ;number of channels to &FF
ECBA    BMI     &ED06   ;jump to ED06 ALWAYS

ECBC    PHP             ;save flags
ECBD    SEI             ;and disable interrupts
ECBE    LDA     &0808,X ;check for end of release
ECC1    CMP     #&04    ;
ECC3    BNE     &ECCF   ;and if not found ECCF
ECC5    JSR     &E45B   ;elseexamine buffer
ECC8    BCC     &ECCF   ;if not empty ECCF
ECCA    LDA     #&00    ;else mark channel dormant
ECCC    STA     &0800,X ;
```

```
ECCF    PLP             ;get back flags

ECD0    LDY     &0820,X ;if no envelope 820=&FF
ECD3    CPY     #&FF    ;
ECD5    BNE     &ECDA   ;then terminate sound
ECD7    JSR     &EB03   ;via EB03
ECDA    RTS             ;else return
                        ;


************ Synchronise sound routines *********************************

ECDB    JSR     &E45B   ;examine buffer if empty carry set
ECDE    BCS     &ECBC   ;
ECE0    AND     #&03    ;else examine next word if>3 or 0
ECE2    BEQ     &EC9F   ;goto ED98 (via EC9F)
ECE4    LDA     &0838   ;else get synchronising count
ECE7    BEQ     &ECFE   ;in 0 (complete) goto ECFE
ECE9    INC     &0828,X ;else set sync flag
ECEC    BIT     &0838   ;if 0838 is +ve S has already been set so
ECEF    BPL     &ECFB   ;jump to ECFB
ECF1    JSR     &E45B   ;else get first byte
ECF4    AND     #&03    ;mask bits 0,1
ECF6    STA     &0838   ;and store result
ECF9    BPL     &ECFE   ;Jump to ECFE (ALWAYS!!)

ECFB    DEC     &0838   ;decrement 0838
ECFE    JMP     &ECD0   ;and silence the channel if envelope not in use


************ Pitch setting ***********************************************

ED01    CMP     &082C,X ;If A=&82C,X then pitch is unchanged
ED04    BEQ     &ECDA   ;then exit via ECDA
ED06    STA     &082C,X ;store new pitch
ED09    CPX     #&04    ;if X<>4 then not noise so
ED0B    BNE     &ED16   ;jump to ED16


*********** Noise setting ************************************************

ED0D    AND     #&0F    ;convert to chip format
ED0F    ORA     &EB3C,X ;
ED12    PHP             ;save flags
ED13    JMP     &ED95   ;and pass to chip control routine at EB22 via ED95


ED16    PHA             ;
ED17    AND     #&03    ;
ED19    STA     &083C   ;lose eigth tone surplus
ED1C    LDA     #&00    ;
ED1E    STA     &083D   ;
ED21    PLA             ;get back A
ED22    LSR             ;divide by 12
ED23    LSR             ;
ED24    CMP     #&0C    ;
ED26    BCC     &ED2F   ;
ED28    INC     &083D   ;store result
ED2B    SBC     #&0C    ;with remainder in A
ED2D    BNE     &ED24   ;

                        ;at this point 83D defines the Octave
                        ;A the semitone within the octave
ED2F    TAY             ;Y=A
ED30    LDA     &083D   ;get octave number into A
ED33    PHA             ;push it
```

```
ED34    LDA     &EDFB,Y ;get byte from look up table
ED37    STA     &083D   ;store it
ED3A    LDA     &EE07,Y ;get byte from second table
ED3D    PHA             ;push it
ED3E    AND     #&03    ;keep two LS bits only
ED40    STA     &083E   ;save them
ED43    PLA             ;pull second table byte
ED44    LSR             ;push hi nybble into lo nybble
ED45    LSR             ;
ED46    LSR             ;
ED47    LSR             ;
ED48    STA     &083F   ;store it
ED4B    LDA     &083D   ;get back octave number
ED4E    LDY     &083C   ;adjust for surplus eighth tones
ED51    BEQ     &ED5F   ;
ED53    SEC             ;
ED54    SBC     &083F   ;
ED57    BCS     &ED5C   ;
ED59    DEC     &083E   ;
ED5C    DEY             ;
ED5D    BNE     &ED53   ;
ED5F    STA     &083D   ;
ED62    PLA             ;
ED63    TAY             ;
ED64    BEQ     &ED6F   ;
ED66    LSR     &083E   ;
ED69    ROR     &083D   ;
ED6C    DEY             ;
ED6D    BNE     &ED66   ;
ED6F    LDA     &083D   ;
ED72    CLC             ;
ED73    ADC     &C43D,X ;
ED76    STA     &083D   ;
ED79    BCC     &ED7E   ;
ED7B    INC     &083E   ;
ED7E    AND     #&0F    ;
ED80    ORA     &EB3C,X ;
ED83    PHP             ;push P
ED84    SEI             ;bar interrupts
ED85    JSR     &EB21   ;set up chip access 1
ED88    LDA     &083D   ;
ED8B    LSR     &083E   ;
ED8E    ROR             ;
ED8F    LSR     &083E   ;
ED92    ROR             ;
ED93    LSR             ;
ED94    LSR             ;
ED95    JMP     &EB22   ;set up chip access 2 and return


**************** Pick up and interpret sound buffer data *****************

ED98    PHP             ;push flags
ED99    SEI             ;disable interrupts
ED9A    JSR     &E460   ;read a byte from buffer
ED9D    PHA             ;push A
ED9E    AND     #&04    ;isolate H bit
EDA0    BEQ     &EDB7   ;if 0 then EDB7
EDA2    PLA             ;get back A
EDA3    LDY     &0820,X ;if &820,X=&FF
EDA6    CPY     #&FF    ;envelope is not in use
EDA8    BNE     &EDAD   ;
EDAA    JSR     &EB03   ;so call EB03 to silence channel

EDAD    JSR     &E460   ;clear buffer of redundant data
```

```
EDB0    JSR     &E460   ;and again
EDB3    PLP             ;get back flags
EDB4    JMP     &EDF7   ;set main duration count using last byte from buffer

EDB7    PLA             ;get back A
EDB8    AND     #&F8    ;zero bits 0-2
EDBA    ASL             ;put bit 7 into carry
EDBB    BCC     &EDC8   ;if zero (envelope) jump to EDC8
EDBD    EOR     #&FF    ;invert A
EDBF    LSR             ;shift right
EDC0    SEC             ;
EDC1    SBC     #&40    ;subtract &40
EDC3    JSR     &EB0A   ;and set volume
EDC6    LDA     #&FF    ;A=&FF

EDC8    STA     &0820,X ;get envelope no.-1 *16 into A
EDCB    LDA     #&05    ;set duration sub-counter
EDCD    STA     &081C,X ;
EDD0    LDA     #&01    ;set phase counter
EDD2    STA     &0824,X ;
EDD5    LDA     #&00    ;set step counter
EDD7    STA     &0814,X ;
EDDA    STA     &0808,X ;and envelope phase
EDDD    STA     &0830,X ;and pitch differential
EDE0    LDA     #&FF    ;
EDE2    STA     &0810,X ;set step count
EDE5    JSR     &E460   ;read pitch
EDE8    STA     &080C,X ;set it
EDEB    JSR     &E460   ;read buffer
EDEE    PLP             ;
EDEF    PHA             ;save duration
EDF0    LDA     &080C,X ;get back pitch value
EDF3    JSR     &ED01   ;and set it
EDF6    PLA             ;get back duration
EDF7    STA     &0818,X ;set it
EDFA    RTS             ;and return


********************** Pitch look up table 1*****************************
EDFB        DB          &F0
EDFC        DB          &B7
EDFD        DB          &82
EDFE        DB          &4F
EDFF        DB          &20
EE00        DB          &F3
EE01        DB          &C8
EE02        DB          &A0
EE03        DB          &7B
EE04        DB          &57
EE05        DB          &35
EE06        DB          &16


********************** Pitch look up table 2 *****************************

EE07        DB          &E7
EE08        DB          &D7
EE09        DB          &CB
EE0A        DB          &C3
EE0B        DB          &B7
EE0C        DB          &AA
EE0D        DB          &A2
EE0E        DB          &9A
EE0F        DB          &92
EE10        DB          &8A
EE11        DB          &82
```

```
EE12        DB          &7A

*********: set current filing system ROM/PHROM *************************
EE13    LDA     #&EF    ;get ROM
EE15    STA     &F5     ;store it
EE17    RTS             ;return


********** Get byte from data ROM ***************************************

EE18    LDX     #&0D    ;X=13
EE1A    INC     &F5     ;
EE1C    LDY     &F5     ;get Rom
EE1E    BPL     &EE59   ;if +ve its a sideways ROM else its a PHROM
EE20    LDX     #&00    ;PHROM
EE22    STX     &F7     ;set address pointer in PHROM
EE24    INX             ;
EE25    STX     &F6     ;to 0001
EE27    JSR     &EEBB   ;pass info to speech processor
EE2A    LDX     #&03    ;X=3

EE2C    JSR     &EE62   ;check for speech processor and output until
                        ;it reports, read byte from ROM
EE2F    CMP     &DF0C,X ;if A<> DF0C+X then EE18 (DF0C = (C))
EE32    BNE     &EE18   ;
EE34    DEX             ;else decrement X
EE35    BPL     &EE2C   ;and do it again
EE37    LDA     #&3E    ;
EE39    STA     &F6     ;get noe lo byte address
EE3B    JSR     &EEBB   ;pass info to speech processor
EE3E    LDX     #&FF    ;
EE40    JSR     &EE62   ;check for speech proc. etc.
EE43    LDY     #&08    ;
EE45    ASL             ;
EE46    ROR     &F7,X   ;
EE48    DEY             ;
EE49    BNE     &EE45   ;
EE4B    INX             ;
EE4C    BEQ     &EE40   ;
EE4E    CLC             ;
EE4F    BCC     &EEBB   ;


************ ROM SERVICE ************************************************

EE51    LDX     #&0E    ;
EE53    LDY     &F5     ;if Y is negative (PHROM)
EE55    BMI     &EE62   ;GOTO EE62
EE57    LDY     #&FF    ;else Y=255
EE59    PHP             ;push flags
EE5A    JSR     &F168   ;offer paged rom service
EE5D    PLP             ;pull processor flags
EE5E    CMP     #&01    ;if A>0  set carry
EE60    TYA             ;A=Y
EE61    RTS             ;return

********* PHROM SERVICE *************************************************
                        ;
EE62    PHP             ;push processor flags
EE63    SEI             ;disable interrupts
EE64    LDY     #&10    ;Y=16
EE66    JSR     &EE7F   ;call EE7F (osbyte 159 write to speech processor
EE69    LDY     #&00    ;Y=0
EE6B    BEQ     &EE84   ;Jump to EE84 (ALWAYS!!)
```

```
*************************************************************************
*                                                                       *
*       OSBYTE 158 read from speech processor *
*                                                                       *
*************************************************************************


EE6D    LDY     #&00    ;Y=0 to set speech proc to read
EE6F    BEQ     &EE82   ;jump to EE82 always

                        ;write A to speech processor as two nybbles

EE71    PHA             ;push A
EE72    JSR     &EE7A   ;to write to speech processor
EE75    PLA             ;get back A
EE76    ROR             ;bring upper nybble to lower nybble
EE77    ROR             ;by rotate right
EE78    ROR             ;4 times
EE79    ROR             ;

EE7A    AND     #&0F    ;Y=lo nybble A +&40
EE7C    ORA     #&40    ;
EE7E    TAY             ;forming command for speech processor


*************************************************************************
*                                                                       *
*       OSBYTE 159 Write to speech processor                            *
*                                                                       *
*************************************************************************


;       on entry data or command in Y

EE7F    TYA             ;transfer command to A
EE80    LDY     #&01    ;to set speech proc to write

                        ;if Y=0 read speech processor
                        ;if Y=1 write speech processor

EE82    PHP             ;push flags
EE83    SEI             ;disable interrupts
EE84    BIT     &027B   ;test for prescence of speech processor
EE87    BPL     &EEAA   ;if not there goto EEAA
EE89    PHA             ;else push A
EE8A    LDA     &F075,Y ;
EE8D    STA     &FE43   ;set DDRA of system VIA to give 8 bit input (Y=0)
                        ;or 8 bit output (Y=1)
EE90    PLA             ;get back A
EE91    STA     &FE4F   ;and send to speech chip
EE94    LDA     &F077,Y ;output Prt B of system VIA
EE97    STA     &FE40   ;to select read or write (dependent on Y)
EE9A    BIT     &FE40   ;loop until
EE9D    BMI     &EE9A   ;speech proceessor reports ready (bit 7 Prt B=0)
EE9F    LDA     &FE4F   ;read speech processor data if  input selected
EEA2    PHA             ;push A
EEA3    LDA     &F079,Y ;reset speech
EEA6    STA     &FE40   ;processor
EEA9    PLA             ;get back A

EEAA    PLP             ;get back flags
EEAB    TAY             ;transfer A to Y
```

```
EEAC    RTS             ;and exit routine
                        ;
EEAD    LDA     &03CB   ;set rom displacement pointer
EEB0    STA     &F6     ;in &F6
EEB2    LDA     &03CC   ;
EEB5    STA     &F7     ;And &F7
EEB7    LDA     &F5     ;if F5 is +ve ROM is selected so
EEB9    BPL     &EED9   ;goto EED9

EEBB    PHP             ;else push processor
EEBC    SEI             ;disable interrupts
EEBD    LDA     &F6     ;get lo displacement
EEBF    JSR     &EE71   ;pass two nyblles to speech proc.
EEC2    LDA     &F5     ;&FA=&F5
EEC4    STA     &FA     ;
EEC6    LDA     &F7     ;get hi displacement value
EEC8    ROL             ;replace two most significant bits of A
EEC9    ROL             ;by 2 LSBs of &FA
EECA    LSR     &FA     ;
EECC    ROR             ;
EECD    LSR     &FA     ;
EECF    ROR             ;
EED0    JSR     &EE71   ;pass two nybbles to speech processor
EED3    LDA     &FA     ;FA has now been divided by 4 so pass
EED5    JSR     &EE7A   ;lower nybble to speech proc.
EED8    PLP             ;get back flags
EED9    RTS             ;and Return
                        ;


************ Keyboard Input and housekeeping ************************
        ;entered from &F00C

EEDA    LDX     #&FF    ;
EEDC    LDA     &EC     ;get value of most recently pressed key
EEDE    ORA     &ED     ;Or it with previous key to check for presses
EEE0    BNE     &EEE8   ;if A=0 no keys pressed so off you go
EEE2    LDA     #&81    ;else enable keybd interupt only by writing bit 7
EEE4    STA     &FE4E   ;and bit 0 of system VIA interupt register
EEE7    INX             ;set X=0
EEE8    STX     &0242   ;reset keyboard semaphore


**********: Turn on Keyboard indicators ******************************
EEEB    PHP             ;save flags
EEEC    LDA     &025A   ;read keyboard status;
                        ;Bit 7  =1 shift enabled
                        ;Bit 6  =1 control pressed
                        ;bit 5  =0 shift lock
                        ;Bit 4  =0 Caps lock
                        ;Bit 3  =1 shift pressed
EEEF    LSR             ;shift Caps bit into bit 3
EEF0    AND     #&18    ;mask out all but 4 and 3
EEF2    ORA     #&06    ;returns 6 if caps lock OFF &E if on
EEF4    STA     &FE40   ;turn on or off caps light if required
EEF7    LSR             ;bring shift bit into bit 3
EEF8    ORA     #&07    ;
EEFA    STA     &FE40   ;turn on or off shift  lock light
EEFD    JSR     &F12E   ;set keyboard counter
EF00    PLA             ;get back flags
EF01    RTS             ;return
                        ;

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

```
*                                                                       *
* MAIN KEYBOARD HANDLING ROUTINE   ENTRY FROM KEYV                      *
* ================================================                      *
*                                                                       *
*                        ENTRY CONDITIONS                               *
*                        ================                               *
* C=0, V=0 Test Shift and CTRL keys.. exit with N set if CTRL pressed   *
*                                  ........with V set if Shift pressed  *
*                                                                       *
* C=1, V=0 Scan Keyboard as OSBYTE &79                                  *
*                                                                       *
* C=0, V=1 Key pressed interrupt entry                                  *
*                                                                       *
* C=1, V=1 Timer interrupt entry                                        *
*                                                                       *
*                                                                       *
*************************************************************************

EF02    BVC     &EF0E   ;if V is clear then leave interrupt routine
EF04    LDA     #&01    ;disable keyboard interrupts
EF06    STA     &FE4E   ;by writing to VIA interrupt vector
EF09    BCS     &EF13   ;if timer interrupt then EF13
EF0B    JMP     &F00F   ;else to F00F

EF0E    BCC     &EF16   ;if test SHFT & CTRL goto EF16
EF10    JMP     &F0D1   ;else to F0D1
                        ;to scan keyboard
*************************************************************************
*       Timer interrupt entry                                           *
*************************************************************************

EF13    INC     &0242   ;increment keyboard semaphore (to 0)


*************************************************************************
*       Test Shift and Control Keys entry                               *
*************************************************************************

EF16    LDA     &025A   ;read keyboard status;
                        ;Bit 7  =1 shift enabled
                        ;Bit 6  =1 control pressed
                        ;bit 5  =0 shift lock
                        ;Bit 4  =0 Caps lock
                        ;Bit 3  =1 shift pressed
EF19    AND     #&B7    ;zero bits 3 and 6
EF1B    LDX     #&00    ;zero X to test for shift key press
EF1D    JSR     &F02A   ;interrogate keyboard X=&80 if key determined by
                        ;X on entry is pressed
EF20    STX     &FA     ;save X
EF22    CLV             ;clear V
EF23    BPL     &EF2A   ;if no key press (X=0) then EF2A else
EF25    BIT     &D9B7   ;set M and V
EF28    ORA     #&08    ;set bit 3 to indicate Shift was pressed
EF2A    INX             ;check the control key
EF2B    JSR     &F02A   ;via keyboard interrogate
EF2E    BCC     &EEEB   ;if carry clear (entry via EF16) then off to EEEB
                        ;to turn on keyboard lights as required
EF30    BPL     &EF34   ;if key not pressed goto EF30
EF32    ORA     #&40    ;or set CTRL pressed bit in keyboard status byte in A
EF34    STA     &025A   ;save status byte
EF37    LDX     &EC     ;if no key previously pressed
EF39    BEQ     &EF4D   ;then EF4D
EF3B    JSR     &F02A   ;else check to see if key still pressed
EF3E    BMI     &EF50   ;if so enter repeat routine at EF50
EF40    CPX     &EC     ;else compare X with last key pressed (set flags)
EF42    STX     &EC     ;store X in last key pressed
```

```
EF44    BNE     &EF4D   ;if different from previous (Z clear) then EF4D
EF46    LDX     #&00    ;else zero
EF48    STX     &EC     ;last key pressed
EF4A    JSR     &F01F   ;and reset repeat system
EF4D    JMP     &EFE9   ;


********** REPEAT ACTION ************************************************
EF50    CPX     &EC     ;if X<>than last key pressed
EF52    BNE     &EF42   ;then back to EF42
EF54    LDA     &E7     ;else get value of AUTO REPEAT COUNTDOWN TIMER
EF56    BEQ     &EF7B   ;if 0 goto EF7B
EF58    DEC     &E7     ;else decrement
EF5A    BNE     &EF7B   ;and if not 0 goto EF7B
                        ;this means that either the repeat system is dormant
                        ;or it is not at the end of its count
EF5C    LDA     &02CA   ;next value for countdown timer
EF5F    STA     &E7     ;store it
EF61    LDA     &0255   ;get auto repeat rate from 0255
EF64    STA     &02CA   ;store it as next value for Countdown timer
EF67    LDA     &025A   ;get keyboard status
EF6A    LDX     &EC     ;get last key pressed
EF6C    CPX     #&D0    ;if not SHIFT LOCK key (&D0) goto
EF6E    BNE     &EF7E   ;EF7E
EF70    ORA     #&90    ;sets shift enabled, & no caps lock all else preserved
EF72    EOR     #&A0    ;reverses shift lock disables Caps lock and Shift enab
EF74    STA     &025A   ;reset keyboard status
EF77    LDA     #&00    ;and set timer
EF79    STA     &E7     ;to 0
EF7B    JMP     &EFE9   ;

EF7E    CPX     #&C0    ;if not CAPS LOCK
EF80    BNE     &EF91   ;goto EF91
EF82    ORA     #&A0    ;sets shift enabled and disables SHIFT LOCK
EF84    BIT     &FA     ;if bit 7 not set by (EF20) shift NOT pressed
EF86    BPL     &EF8C   ;goto EF8C
EF88    ORA     #&10    ;else set CAPS LOCK not enabled
EF8A    EOR     #&80    ;reverse SHIFT enabled

EF8C    EOR     #&90    ;reverse both SHIFT enabled and CAPs Lock
EF8E    JMP     &EF74   ;reset keyboard status and set timer


*********** get ASCII code ***********************************************
        ;on entry X=key pressed internal number

EF91    LDA     &EFAB,X ;get code from look up table
EF94    BNE     &EF99   ;if not zero goto EF99 else TAB pressed
EF96    LDA     &026B   ;get TAB character

EF99    LDX     &025A   ;get keyboard status
EF9C    STX     &FA     ;store it in &FA
EF9E    ROL     &FA     ;rotate to get CTRL pressed into bit 7
EFA0    BPL     &EFA9   ;if CTRL NOT pressed EFA9

EFA2    LDX     &ED     ;get no. of previously pressed key
EFA4    BNE     &EF4A   ;if not 0 goto EF4A to reset repeat system etc.
EFA6    JSR     &EABF   ;else perform code changes for CTRL

EFA9    ROL     &FA     ;move shift lock into bit 7
EFAB    BMI     &EFB5   ;if not effective goto EFB5 else
EFAD    JSR     &EA9C   ;make code changes for SHIFT

EFB0    ROL     &FA     ;move CAPS LOCK into bit 7
EFB2    JMP     &EFC1   ;and Jump to EFC1
```

```
EFB5    ROL     &FA     ;move CAPS LOCK into bit 7
EFB7    BMI     &EFC6   ;if not effective goto EFC6
EFB9    JSR     &E4E3   ;else make changes for CAPS LOCK on, return with
                        ;C clear for Alphabetic codes
EFBC    BCS     &EFC6   ;if carry set goto EFC6 else make changes for
EFBE    JSR     &EA9C   ;SHIFT as above

EFC1    LDX     &025A   ;if shift enabled bit is clear
EFC4    BPL     &EFD1   ;goto EFD1
EFC6    ROL     &FA     ;else get shift bit into 7
EFC8    BPL     &EFD1   ;if not set goto EFD1
EFC8    BPL     &EFD1   ;if not set goto EFD1
EFCA    LDX     &ED     ;get previous key press
EFCC    BNE     &EFA4   ;if not 0 reset repeat system etc. via EFA4
EFCE    JSR     &EA9C   ;else make code changes for SHIFT
EFD1    CMP     &026C   ;if A<> ESCAPE code
EFD4    BNE     &EFDD   ;goto EFDD
EFD6    LDX     &0275   ;get Escape key status
EFD9    BNE     &EFDD   ;if ESCAPE returns ASCII code goto EFDD
EFDB    STX     &E7     ;store in Auto repeat countdown timer

EFDD    TAY             ;
EFDE    JSR     &F129   ;disable keyboard
EFE1    LDA     &0259   ;read Keyboard disable flag used by Econet
EFE4    BNE     &EFE9   ;if keyboard locked goto EFE9
EFE6    JSR     &E4F1   ;put character in input buffer
EFE9    LDX     &ED     ;get previous keypress
EFEB    BEQ     &EFF8   ;if none  EFF8
EFED    JSR     &F02A   ;examine to see if key still pressed
EFF0    STX     &ED     ;store result
EFF2    BMI     &EFF8   ;if pressed goto EFF8
EFF4    LDX     #&00    ;else zero X
EFF6    STX     &ED     ;and &ED

EFF8    LDX     &ED     ;get &ED
EFFA    BNE     &F012   ;if not 0 goto F012
EFFC    LDY     #&EC    ;get first keypress into Y
EFFE    JSR     &F0CC   ;scan keyboard from &10 (osbyte 122)

F001    BMI     &F00C   ;if exit is negative goto F00C
F003    LDA     &EC     ;else make last key the
F005    STA     &ED     ;first key pressed i.e. rollover

F007    STX     &EC     ;save X into &EC
F009    JSR     &F01F   ;set keyboard repeat delay
F00C    JMP     &EEDA   ;go back to EEDA


*************************************************************************
*   Key pressed interrupt entry point                                   *
*************************************************************************
                        ;enters with X=key
F00F    JSR     &F02A   ;check if key pressed

F012    LDA     &EC     ;get previous key press
F014    BNE     &F00C   ;if none back to housekeeping routine
F016    LDY     #&ED    ;get last keypress into Y
F018    JSR     &F0CC   ;and scan keyboard
F01B    BMI     &F00C   ;if negative on exit back to housekeeping
F01D    BPL     &F007   ;else back to store X and reset keyboard delay etc.


*************** Set Autorepeat countdown timer *************************

F01F    LDX     #&01    ;set timer to 1
F021    STX     &E7     ;
```

```
F023    LDX     &0254   ;get next timer value
F026    STX     &02CA   ;and store it
F029    RTS             ;
```

```
*************** Interrogate Keyboard routine ***********************
                        ;
F02A    LDY     #&03    ;stop Auto scan
F02C    STY     &FE40   ;by writing to system VIA
F02F    LDY     #&7F    ;set bits 0 to 6 of port A to input on bit 7
                        ;output on bits 0 to 6
F031    STY     &FE43   ;
F034    STX     &FE4F   ;write X to Port A system VIA
F037    LDX     &FE4F   ;read back &80 if key pressed (M set)
F03A    RTS             ;and return
```

```
*********************************************************************
*                                                                   *
*       KEY TRANSLATION TABLES                                      *
*                                                                   *
*       7 BLOCKS interspersed with unrelated code                   *
*********************************************************************
```

```
*key data block 1

F03B    DB      71,33,34,35,84,38,87,2D,5E,8C
        ;       q ,3 ,4 ,5 ,f4,8 ,f7,- ,^ ,rt
```

```
*********************************************************************
*                                                                   *
*       OSBYTE 120  Write KEY pressed Data                          *
*                                                                   *
*                                                                   *
*********************************************************************
```

```
F045    STY     &EC     ;store Y as latest key pressed
F047    STX     &ED     ;store X as previous key pressed
F049    RTS             ;and exit

*key data block 2

F04A    DB      80,77,65,74,37,69,39,30,5F,8E
        ;       f0,w ,e ,t ,7 ,i ,9 ,0 ,_ ,lft

F055    JMP     (&FDFE) ;Jim paged entry vector
F058    JMP     (&FA)   ;

*key data block 3

F05A    DB      31,32,64,72,36,75,6F,70,5B,8F
        ;       1 ,2 ,d ,r ,6 ,u ,o ,p ,[ ,dn
```

```
*********************************************************************
*                                                                   *
* Main entry to keyboard routines                                   *
*                                                                   *
```

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

F065    BIT     &D9B7   ;set V and M
F068    JMP     (&0228) ;i.e. KEYV

*key data block 4

F06B    DB      01,61,78,66,79,6A,6B,40,3A,0D
        ;       CL,a ,x ,f ,y ,j ,k ,@ ,: ,RETN  N.B CL=CAPS LOCK

*speech routine data
F075    DB      00,FF,01,02,09,0A

*key data block 5

F07B    DB      02,73,63,67,68,6E,6C,3B,5D,7F
        ;       SL,s ,c ,g ,h ,n ,l ,; ,] ,DEL N.B. SL=SHIFT LOCK



*********************************************************************
*                                                                   *
*       OSBYTE 131  READ OSHWM  (PAGE in BASIC)                     *
*                                                                   *
*                                                                   *
*********************************************************************


F085    LDY     (&0244) ;read current OSHWM
F088    LDX     #&00    ;
F08A    RTS             ;

*key data block 6

F08B    DB              00 ,7A,20   ,76,62,6D,2C,2E,2F,8B
        ;               TAB,Z ,SPACE,V ,b ,m ,, ,. ,/ ,copy

***** set input buffer number and flush it ***************************

F095    LDX     &0241   ;get current input buffer
F098    JMP     &E1AD   ;flush it

*key data block 7

F09B    DB      1B,81,82,83,85,86,88,89,5C,8D
        ;      ESC,f1,f2,f3,f5,f6,f8,f9,\ ,


F0A5    JMP     (&0220) ;goto eventV handling routine



*********************************************************************
*                                                                   *
*       OSBYTE 15  FLUSH SELECTED BUFFER CLASS                      *
*                                                                   *
*                                                                   *
*********************************************************************

                        ;flush selected buffer
                        ;X=0 flush all buffers
                        ;X>1 flush input buffer

F0A8    BNE     &F095   ;if X<>1 flush input buffer only
F0AA    LDX     #&08    ;else load highest buffer number (8)
```

```
F0AC    CLI             ;allow interrupts
F0AD    SEI             ;briefly!
F0AE    JSR     &F0B4   ;flush buffer
F0B1    DEX             ;decrement X to point at next buffer
F0B2    BPL     &F0AC   ;if X>=0 flush next buffer
                        ;at this point X=255


*************************************************************************
*                                                                       *
*       OSBYTE 21  FLUSH SPECIFIC BUFFER                                *
*                                                                       *
*                                                                       *
*************************************************************************
;on entry X=buffer number

F0B4    CPX     #&09    ;is X<9?
F0B6    BCC     &F098   ;if so flush buffer or else
F0B8    RTS             ;exit
                        ;
*************************************************************************
*                                                                       *
*               Issue *HELP to ROMS                                     *
*************************************************************************
F0B9    LDX     #&09    ;
F0BB    JSR     &F168   ;
F0BE    JSR     &FA4A   ;print following message routine return after BRK
F0C1    DB      &0D     ;carriage return
F0C2    DS      'OS 1.20' ;help message
F0C9    DB      &0D     ;carriage return
F0CA    BRK             ;
F0CB    RTS             ;



*************************************************************************
*                                                                       *
*       OSBYTE 122  KEYBOARD SCAN FROM &10 (16)                         *
*                                                                       *
*                                                                       *
*************************************************************************

                        ;
F0CC    CLC             ;clear carry
F0CD    LDX     #&10    ;set X to 10
                        ;


*************************************************************************
*                                                                       *
*       OSBYTE 121  KEYBOARD SCAN FROM VALUE IN X                       *
*                                                                       *
*                                                                       *
*************************************************************************

F0CF    BCS     &F068   ;if carry set (by osbyte 121) F068
                        ;Jmps via KEYV and hence back to;


*************************************************************************
*        Scan Keyboard C=1, V=0 entry via KEYV                          *
*************************************************************************

F0D1    TXA             ;if X is +ve goto F0D9
F0D2    BPL     &F0D9   ;
F0D4    JSR     &F02A   ;else interrogate keyboard
```

```
F0D7    BCS     &F12E   ;if carry set F12E to set Auto scan else
F0D9    PHP             ;push flags
F0DA    BCC     &F0DE   ;if carry clear goto F0DE else
F0DC    LDY     #&EE    ;set Y so next operation saves to 2cd
F0DE    STA     &01DF,Y ;can be 2cb,2cc or 2cd
F0E1    LDX     #&09    ;set X to 9
F0E3    JSR     &F129   ;select auto scan
F0E6    LDA     #&7F    ;set port A for input on bit 7 others outputs
F0E8    STA     &FE43   ;
F0EB    LDA     #&03    ;stop auto scan
F0ED    STA     &FE40   ;
F0F0    LDA     #&0F    ;select non-existent keyboard column F (0-9 only!)
F0F2    STA     &FE4F   ;
F0F5    LDA     #&01    ;cancel keyboard interrupt
F0F7    STA     &FE4D   ;
F0FA    STX     &FE4F   ;select column X (9 max)
F0FD    BIT     &FE4D   ;if bit 1 =0 there is no keyboard interrupt so
F100    BEQ     &F123   ;goto F123
F102    TXA             ;else put column address in A

F103    CMP     &01DF,Y ;compare with 1DF+Y
F106    BCC     &F11E   ;if less then F11E
F108    STA     &FE4F   ;else select column again
F10B    BIT     &FE4F   ;and if bit 7 is 0
F10E    BPL     &F11E   ;then F11E
F110    PLP             ;else push and pull flags
F111    PHP             ;
F112    BCS     &F127   ;and if carry set goto F127
F114    PHA             ;else Push A
F115    EOR     &0000,Y ;EOR with EC,ED, or EE depending on Y value
F118    ASL             ;shift left
F119    CMP     #&01    ;set carry if = or greater than number holds EC,ED,EE
F11B    PLA             ;get back A
F11C    BCS     &F127   ;if carry set F127
F11E    CLC             ;else clear carry
F11F    ADC     #&10    ;add 16
F121    BPL     &F103   ;and do it again if 0=<result<128
F123    DEX             ;decrement X
F124    BPL     &F0E3   ;scan again if greater than 0
F126    TXA             ;
F127    TAX             ;
F128    PLP             ;pull flags

F129    JSR     &F12E   ;call autoscan
F12C    CLI             ;allow interrupts
F12D    SEI             ;disable interrupts


**************Enable counter scan of keyboard columns ******************
                        ;called from &EEFD, &F129

F12E    LDA     #&0B    ;select auto scan of keyboard
F130    STA     &FE40   ;tell VIA
F133    TXA             ;Get A into X
F134    RTS             ;and return
```