Writing a sector to disk

For writing a sector to disk, the Commodore DOS offers the block write command. Due to heavy bugs in the B-W command, Commodore has sacrificed one of the user commands as a bugfix replacement. So instead of B-W you simply use U2.

The format of this DOS command is: “U2 <channel> <drive> <track> <sector>”

The drive parameter is only used for dual disk drives, so for all common C64/C128/C16 drives this parameter will always be 0.

Parameters track and sector explain themselves. They are sent in PETSCII format, so in assembler often a binary to PETSCII conversion is needed.

A speciality of this command is the channel parameter. Actually you can't simply send this command to the drive and then start to send sector bytes. For the sending of the bytes you have to open another file which is adressed by this parameter.

Before you send bytes into the channel buffer, it is neccessary to set the buffer pointer to 0 via the B-P command.

BASIC code:

```
10 SA=8192
20 OPEN 2,8,2,"#"
30 OPEN 15,8,15,"B-P 2 0"
40 FOR I=0 TO 255
50 A=PEEK(SA):SA=SA+1
60 PRINT#2,CHR$(A);
70 NEXT I
80 PRINT#15,"U2 2 0 18 0"
90 CLOSE 15:CLOSE 2
```

Assembler code:

```
sector_address = $2000  ; just an example

        ; open the channel file

        LDA #cname_end-cname
        LDX #<cname
        LDY #>cname
        JSR $FFBD     ; call SETNAM

        LDA #$02      ; file number 2
        LDX $BA       ; last used device number
        BNE .skip
        LDX #$08      ; default to device 8
.skip   LDY #$02      ; secondary address 2
        JSR $FFBA     ; call SETLFS
```

```
        JSR $FFC0     ; call OPEN
        BCS .error    ; if carry set, the file could not be opened

        ; open the command channel

        LDA #bpcmd_end-bpcmd
        LDX #<bpcmd
        LDY #>bpcmd
        JSR $FFBD     ; call SETNAM
        LDA #$0F      ; file number 15
        LDX $BA       ; last used device number
        LDY #$0F      ; secondary address 15
        JSR $FFBA     ; call SETLFS

        JSR $FFC0     ; call OPEN (open command channel and send B-P
command)
        BCS .error    ; if carry set, the file could not be opened

        ; check drive error channel here to test for
        ; FILE NOT FOUND error etc.

        LDX #$02      ; filenumber 2
        JSR $FFC9     ; call CHKOUT (file 2 now used as output)

        LDA #<sector_address
        STA $AE
        LDA #>sector_address
        STA $AF

        LDY #$00
.loop   LDA ($AE),Y   ; read byte from memory
        JSR $FFD2     ; call CHROUT (write byte to channel buffer)
        INY
        BNE .loop     ; next byte, end when 256 bytes are read

        LDX #$0F      ; filenumber 15
        JSR $FFC9     ; call CHKOUT (file 15 now used as output)

        LDY #$00
.loop2  LDA bwcmd,Y   ; read byte from command string
        JSR $FFD2     ; call CHROUT (write byte to command channel)
        INY
        CPY #bwcmd_end-bwcmd
        BNE .loop2    ; next byte, end when 256 bytes are read
.close

        JSR $FFCC     ; call CLRCHN

        LDA #$0F      ; filenumber 15
        JSR $FFC3     ; call CLOSE
```

```
        LDA #$02      ; filenumber 2
        JSR $FFC3     ; call CLOSE

        JSR $FFCC     ; call CLRCHN
        RTS
.error
        ; Akkumulator contains BASIC error code

        ; most likely errors:
        ; A = $05 (DEVICE NOT PRESENT)

        ... error handling for open errors ...
        JMP .close    ; even if OPEN failed, the file has to be closed

cname:  .TEXT "#"
cname_end:

bpcmd:  .TEXT "B-P 2 0"
bpcmd_end:

bwcmd:  .TEXT "U2 2 0 18 0"
        .BYTE $0D     ; carriage return, required to start command
bwcmd_end:
```