* om instabiliteit te vermijden is het verstandig om 8 van IC 4 buiten de voet te buigen.

* * * * * * * * * * * * * CRTC.DOC * * * * * * * * * * * * * * *

Enschede, 15 Jan 1988.

Het aansluiten van een monitor met een 9 pens "IBM" of "IBM cloon" stekker is in principe mogelijk op de VDU kaart van Electuur, alleen moeten er enkele wijzigingen aangebracht worden op de VDU kaart.

Als eerste kan men het beste profesorisch de volgende verbindingen aanbrengen op de VDU kaart.
Als eerste het VIDIO signaal afnemen van pen ~~12~~ 6 IC3 (N22) en deze aansluiten op pen 7 van de 9 pens stekker. (1) geel
Hor. sync. kan men direct afnemen van R5 (of van pen 39 IC11 (6545)) en aansluiten op pen 8 van de 9 pens stekker. (2) groen
Vert. sync. moet eerst geinverteerd worden, afnemen van R6 (of van pen 40 IC11) en via een inverter (b.v pen 9 - 11 IC2) aansluiten op pen 9 van de 9 pens stekker. 11 → 9 * (3) blauw
Intensity, pen 6 van de 9 pens stekker, dient men aan te sluiten op V+ (+5V) (4) paars
Hierna moet men de stekker alleen nog voorzien van een GND die men aansluit op pen 1 en 2 van de 9 pens stekker. (5) grijs

Het is niet nodig dat men enig component van de print verwijderd, zolang men alleen de nieuwe monitor aansluit.
Het is door deze tijdelijke oplossing nog steeds mogelijk Uw "oude" monitor op Uw systeem aan te sluiten.

Als U deze aansluitingen heeft aangebracht kunt U het systeem opstarten.
Als het beeld dat nu op Uw scherm verschijnt niet meer is dan een fractie van Uw totale beeldscherm oppervlak dient men als eerste een register van de CRTC aan te passen.
Dit is register 3 - H/V sync width -.
Vanuit I/O65 wordt hier de waarde $88 ingezet, deze waarde moest ik verhogen tot $8F om een redelijk beeld te krijgen.

Dit kan men b.v. op de volgende manier doen :
Met het nieuwe DOS commando CRTC, waarmee men de verschillende CRTC registers kan aanpassen.
De oude gebruikers van DOS65 zullen dit commando wel kennen, het was er vroeger namelijk ook alleen in een ander jasje, en wel van HAVIsoft.

Met het commando HELP CRTC krijgt een kleine uitleg van het commando CRTC zoals hieronder afgebeeld :

```
Function : change CRTC registers.
Syntax   : CRTC
Options  : No options.
Abbreviation : No abbreviation.
Use + and - to change and E or X to select register
n.b. : The values printed right on the screen are an copy from I/O65.
```

Als U het commando CRTC opstart dient U het volgende op Uw scherm te krijgen. :

```
 Change CRTC registers, select with E or X and change with + or -, quit with Q
```

Register nr.	actual value		I/O65 value
Register 0	: $	- Hor tot (N-1)	$7E
Register 1	: $	- Hor Disp	$50
Register 2	: $	- Hor sync pos	$5F
Register 3	: $	- H/V sync width	$88
Register 4	: $	- Vert tot (N-1)	$1E
Register 5	: $	- Vert tot adj	$05
Register 6	: $	- Vert disp	$19
Register 7	: $	- Vert sync pos	$1C
Register 8	: $	- Mode control	$00
Register 9	: $	- Scanlines (N-1)	$09
Register 10	: $	- Cur start	$00
Register 11	: $	- Cur end	$09
Register 12	: $	- Start HI	$00
Register 13	: $	- Start LO	$00
Register 14	: $	- Cur HI	$00
Register 15	: $	- Cur LO	$00

Op de lege plaatsen --- zullen natuurlijk de eerste keer dat U dit commando gebruikt de waarden uit I/O65 staan omdat U (als het goed is) nog niet Uw CRTC waarden heeft aangepast, en er nog geen file CRTC.DAT bestaat op Uw systeem schijf.
Als U enige registers van de CRTC heeft veranderd heeft U de mogelijkheid deze nieuwe waarden op te slaan in een file genaamd CRTC.DAT.
Iedere keer dat U het systeem nu opnieuw opstart kunt U met het commando SETCRTC deze nieuwe CRTC waarden in de registers copieren.
Het is dan natuurlijk het logische gevolg dat U het commando SETCRTC in Uw LOGIN.COM zet zodat deze waarden automatisch in de CRTC registers gecopieerd worden zodra U het systeem opstart.

Het is ntuurlijk ook mogelijk om, als U de juiste waarden heeft bepaald een nieuwe I/O65 eprom aanmaakt (laat maken).

Indien deze beschrijving niet geheel correct is verzoek ik een ieder mij hierop te wijzen zodat ik dit kan aanpassen.
Ook is iedere andere oplossing natuurlijk van harte welkom.

Ik hoop dat deze beschrijving duidelijk genoeg is, en dat er geen fouten in verwerkt zijn.....

J.H.G.M. Banser
BANSOFT
Haaksbergerstraat 199
7513 EM Enschede
tel : 053-324137

```
17-Jan-88 14:06  AS-CRTC  Page 1


;
;* * * * * * * * AS-CRTC * * * * * * * *
;
;description :  assembleer CRTC.MAC
;

ASN B
> -TF CRTC.LST AS -L CRTC
COPY CRTC.BIN CRTC
SETM -RWDC CRTC



17-Jan-88 14:07  CRTC.LST  Page 1


Last assembled address: A6FF
Errors detected: 0



17-Jan-88 14:08  AS-SET  Page 1


;
;* * * * * * * * AS-SET * * * * * * * *
;
;description :  assembleer SETCRTC.MAC
;

ASN B
> -TF SETCRTC.LST AS -L SETCRTC
COPY SETCRTC.BIN SETCRTC
SETM -RWDC SETCRTC
```

```
;* * * * * * * * * * * * SETCRTC.MAC * * * * * * * * * * * * *
;
;Discription : Get crtc register values from disk and fill crtc registers
;
;Last upgrating 30-12-87
;

        opt     nolis
        lib     0:library
        opt     lis

yi      macro   src
        ldy     #src&255
        lda     #src>>8
        endm

        org     $200

SET     yi      loadbuf         get new CRTC values
        jsr     getval
        ldx     #15
1       txa
        sta     crtcar
        lda     buffer,x        copy buffer into crtc registers
        sta     crtcrf
        dex
        bpl     1.b
        rts

getval  tsx
        stx     stpo
        jsr     command
        ldx     stpo
        txs
        rts

stpo    res     1,0
loadbuf fcc     'LOAD 0:CRTC.DAT 300,30F',0

        org     $300
buffer
        end     SET
```

```
;* * * * * * * * * * * * CRTC.MAC * * * * * * * * * * * * * *
;
;Discription : change crtc register values
;
;Last upgrating 15-01-1988
;

ctb     equ     $f141           pointer to crtc tabel

;       macro's

pos     macro   xpos,ypos
        ldx     #xpos
        ldy     ypos
        jsr     posit
        endm

pos2    macro   xpos,ypos
        ldx     #xpos
        ldy     #ypos
        jsr     posit
        endm

yi      macro   src
        ldy     #src&255
        lda     #src>>8
        endm

        opt     nolis
        lib     0:library
        opt     lis

        org     $a000

CRTC    jmp     crtc
        fcc     $c8,$c5,$cc,$d0
help    jsr     prstr
        fcc     '\rFunction : change CRTC registers.'
        fcc     '\rSyntax   : CRTC '
        fcc     '\rOptions  : No options.'
        fcc     '\rAbbreviation : No abbreviation.'
        fcc     '\rUse \'+\' and \'-\' to change and E or X'
        fcc     'to select register'
        fcc     '\rn.b. : The first values printed on the screen'
        fcc     ' are an copy from I/O65.'
        fcc     '\r\r',0

crtc    tsx
        stx     stpo
        yi      loadbuf         get crtc registersfrom disk
        jsr     command
        ldx     stpo
        txs

        jsr     cls
        jsr     prstr
        fcc     $1d,2,2,'Change CRTC registers, select with E or X'
        fcc     ' and change with + or -, quit with Q '
        fcc     $1d,8,3,'Register nr.  actual value              I/O65 value'
        fcc     $1d,8,4,'Register  0 : $     - Hor tot (N-1)       $7E'
```

```
        fcc     $14,8,5,'Register  1 : $      - Hor Disp              $50'
        fcc     $14,8,6,'Register  2 : $      - Hor sync pos          $5F'
        fcc     $14,8,7,'Register  3 : $      - H/V sync width        $88'
        fcc     $14,8,8,'Register  4 : $      - Vert tot (N-1)        $1E'
        fcc     $14,8,9,'Register  5 : $      - Vert tot adj          $05'
        fcc     $14,8,10,'Register  6 : $       - Vert disp            $19'
        fcc     $14,8,11,'Register  7 : $       - Vert sync pos        $1C'
        fcc     $14,8,12,'Register  8 : $       - Mode control         $00'
        fcc     '\r       Register  9 : $       - Scanlines (N-1)      $09'
        fcc     $14,8,14,'Register 10 : $       - Cur start            $00'
        fcc     $14,8,15,'Register 11 : $       - Cur end              $09'
        fcc     $14,8,16,'Register 12 : $       - Start HI             $00'
        fcc     $14,8,17,'Register 13 : $       - Start LO             $00'
        fcc     $14,8,18,'Register 14 : $       - Cur HI               $00'
        fcc     $14,8,19,'Register 15 : $       - Cur LO               $00'
        fcc     0

        lda     #$1d                set register values on screen
        sta     ycur                (from CRTC.DAT)
        ldx     #$0f
        stx     xreg
1       dec     ycur
        pos     24,ycur
        ldx     xreg
        lda     buffer,x
        jsr     prbyt               print accu as byte
        dec     xreg
        bpl     1.b

        ldx     #$00
        ldy     #$04
        sty     ycur
3       lda     #$ff
        sta     ok
        stx     xreg
        jsr     flash               flash byte and get input
        ldx     xreg
        lda     areg
        cmp     #'+
        bne     1.f
        inc     buffer,x
        jmp     2.f
1       cmp     #'-
        bne     1.f
        dec     buffer,x
2       txa
        sta     crtcar              select reg
        lda     buffer,x            get value from buffer
        sta     crtcrf              put value in reg
        jmp     3.b

1       and     #$5f
        cmp     #'E                 E ?
        bne     1.f                 no
        cpx     #$00
        beq     2.f
        dex                         verlaag xreg voor buffer pointer
        dec     ycur
        jmp     3.b
2       ldx     #$0f
```

```
        ldy     #$13
        sty     ycur            set new x,y
        jmp     3.b

1       cmp     #'X             X ?
        bne     1.f             no, it must been Q(uit)
        cpx     #$0f
        beq     2.f
        inx
        inc     ycur
        jmp     3.b
2       ldx     #$00
        ldy     #$04
        sty     ycur            set new x,y
        jmp     3.b

1       pos2    1,21            char was Q(uit) - END -
        jsr     prstr
        fcc     'Save new defauld (on drive 0) ? (Y/*) : ',0
1       jsr     prtime          update time
        jsr     getev           get char
        beq     1.b             no char
        and     #$5f            Upper case
        cmp     #'Y             Y(es) ?
        bne     1.f
        jsr     prstr
        fcc     $1d,1,22,'Save new CRTC values',0
        tsx
        stx     stpo
        yi      savebuf         set command pointer
        jsr     command         execute command
        ldx     stpo
        txs
1       pos2    1,23            char was Q(uit) - END -
        rts

;                       clear screen
cls     lda     #$0c
        jsr     out
        rts

;                       flash byte check for input
flash   pos     24,ycur
        jsr     inv
        ldx     xreg
        lda     buffer,x        get byte
        jsr     prbyt           print it
        jsr     wait            wait 1 sec
6       pos     24,ycur         posit cursor
        jsr     norm            print normal
        ldx     xreg
        lda     buffer,x        get byte
        jsr     prbyt           print
        lda     ok              input char ?
        beq     1.f             yes
        jsr     wait            no, wait 1 sec
        lda     ok              check input
        beq     1.f             ok !
        jmp     flash           flash byte
1       rts
```

```
;                          wait 1 second to turn collor
wait    ldx     sec
        cpx     clsec
        bne     1.f                 yes
        ldy     #$00
        jsr     getev               get char
        beq     wait                no char
4       cmp     inputb,y            compare with +-eExXqQ
        beq     3.f
        iny
        cpy     #$09
        bne     4.b
        jmp     wait
3       sta     areg                save input char
        lda     #$00
        sta     ok
1       jsr     prtime              update time
        rts


inv     lda     #$1b
        jsr     out                 print byte normal
        lda     #'i
        jsr     out
        rts

norm    lda     #$1b
        jsr     out                 print byte normal
        lda     #'n
        jsr     out
        rts

prtime  ldx     sec
        cpx     clsec               update time on status row
        beq     2.f
        stx     clsec
        jsr     prtim
2       rts

;       Variables

xreg    res     1,0     temporary buffer for x register
yreg    res     1,0     temporary buffer for y register
areg    res     1,0     temporary buffer for accu
xcur    res     1,0
ycur    res     1,0
stpo    res     1,0
ok      res     1,ff    input char ok
sec1    res     1,00    copy of sec
inputb  fcc     '+-eExXqQ',$00
loadbuf fcc     'LOAD 0:CRTC.DAT A6F0,A6FF',0
savebuf fcc     'SAVE 0:CRTC.DAT A6F0,A6FF',0

        org     $a6f0
buffer  fcc     $7e
        fcc     $50     temporary buffer for CRTC registers
        fcc     $5f
        fcc     $88
        fcc     $1e
```

```
        fcc     $05
        fcc     $19
        fcc     $1c
        fcc     $00
        fcc     $09
        fcc     $00
        fcc     $09
        fcc     $00
        fcc     $00
        fcc     $00
        fcc     $00

        end     CRTC
```

```
;
;* * * * * * * * LIBRARY.MAC * * * * * * * *
;
;Description :  Most used I/O entries from : I/O65 + DOS65
;
dos       equ     $c000
command   equ     dos+$06          execute command
in        equ     dos+$20          get from input device
out       equ     dos+$23          put on output device
inecho    equ     dos+$26          get and put char
bufin     equ     dos+$29          get characters from input device
bufuit    equ     dos+$2c          execute command
crlf      equ     dos+$2f          print a crlf
spa       equ     dos+$32          print spatie
hnout     equ     dos+$35          a nibble or hex out
prbyt     equ     dos+$38          print ACCU as a byte
prstr     equ     dos+$3b          print string ending with $00
ashex     equ     dos+$3e          convert asc. to hex.
loupch    equ     dos+$d1          accu lower to upper;chec eoln =>z=1
hexdec    equ     dos+$d4          convert hex. to dec.
dechex    equ     dos+$d7          accu dec to hex

inputx    equ     dos+$da          get from input device x
outputx   equ     dos+$dd          put on output device x
sopt      equ     dos+$68          scan options; opt => X
spar      equ     dos+$6b          scan parameters; mask => X
msior     equ     dos+$a2a         reset / set mode bit

readsct   equ     dos+$10c0        read sector,  Tr=X, Sec=Y
writsct   equ     dos+$10c3        write sector,    "     "
checsct   equ     dos+$10c6        check sector     "     "
wrchsct   equ     dos+$10c9        write and chec sector
ermes1    equ     dos+$10b7        print error message
create    equ     dos+$1039        create ascii file
fopen     equ     dos+$1042        open file
sread     equ     dos+$1003        single char. read
swrite    equ     dos+$100c        single char. write
fclose    equ     dos+$104b        close ascii file

io65      equ     $f000
inpmon    equ     io65+$09
maksln    equ     io65+$1e         make new statusline
norstat   equ     io65+$21         define normal statusline
posit     equ     io65+$2d         position the cursor to X,Y
unpos     equ     io65+$27         get old cursor position back
prtim     equ     io65+$583        print the time
prtima    equ     io65+$5ab        print a hex byte as two decimal nibbles
shdatb    equ     io65+$5c4        print the date
clckup    equ     io65+$5df        update clock and date on statusline
memout    equ     io65+$8d6        memory as I/O device output routine
meminp    equ     io65+$85f        memory as I/O device input routine
prout     equ     io65+$88c        centronics printer output routine
getaskey  equ     io65+$8ac        get char from keyboard buffer, check funtions
getkey    equ     io65+$8cb        get char from keyboard buffer
getev     equ     io65+$91b        get a key, if no key Accu=$00
conver    equ     io65+$f02        convert 2 hex bytes to 3 decimal bytes
pridec    equ     io65+$fd1        print decimal numbers (from conver)
pribyt    equ     io65+$f6a        print accu as a byte
prnibl    equ     io65+$f76        print hex number per nibble
prhd      equ     io65+$f8e        print a number (dec.) from a (or x) and y
```



```
hexde     equ     io65+$fa2        convert hex in A to dec. in x and y

acia      equ     $e130            acia adresses
recreg    equ     acia             receiver data register
trmreg    equ     acia             transmit data register
sr        equ     acia+1           status register
cmd       equ     acia+2           command register
ctl       equ     acia+3           control register

crtc1     equ     #$e1d0
crtcar    equ     crtc1            address register
crtcrf    equ     crtc1+$01        register file

viaa      equ     $e100            via a adresses
orba      equ     viaa             output/input reg. `b'
orna      equ     viaa+1           output/input reg. `a'
```

```
t1cla    equ     viaa+4          t1 low-order latches/counter
t1cha    equ     viaa+5          t1 high-order counter
t1lla    equ     viaa+6          t1 low-order latches
t1lha    equ     viaa+7          t1 high-order latches
t2cla    equ     viaa+8          t2 low-order latches/counter
t2cha    equ     viaa+9          t2 high-order counter
sra      equ     viaa+10         shift reg.
acra     equ     viaa+11         auxiliary control reg.
pcra     equ     viaa+12         peripheral control reg.
ifra     equ     viaa+13         interrupt flag reg.
iera     equ     viaa+14         interrupt enable reg.
iraa     equ     viaa+15         output/input reg. `a', exept no `handshake'

viab     equ     $e110           via b adresses
orbb     equ     viab            output/input reg. `b'
orab     equ     viab+1          output/input reg. `a'
ddrbb    equ     viab+2          data direction reg. `b'
ddrab    equ     viab+3          data direction reg. `a'
t1clb    equ     viab+4          t1 low-order latches/counter
t1chb    equ     viab+5          t1 high-order counter
t1llb    equ     viab+6          t1 low-order latches
t1lhb    equ     viab+7          t1 high-order latches
t2clb    equ     viab+8          t2 low-order latches/counter
t2chb    equ     viab+9          t2 high-order counter
srb      equ     viab+10         shift reg.
acrb     equ     viab+11         auxiliary control reg.
pcrb     equ     viab+12         peripheral control reg.
ifrb     equ     viab+13         interrupt flag reg.
ierb     equ     viab+14         interrupt enable reg.
irab     equ     viab+15         output/input reg. `a', exept no `handshake'

clsec    equ     $e718           var for clock update
acctl    equ     $e734           var for acia control reg
accmd    equ     $e735           var for acia command reg
deci1    equ     $e73a           var for calculate decimal
deci2    equ     $e73b
deci3    equ     $e73c
intv1    equ     $e750           T1  via a, Clock interrupt
intv2    equ     $e752           T2  via a
intv3    equ     $e754           CB1 via a
intv4    equ     $e756           CB2 via a
intv5    equ     $e758           SR  via a
```



```
intv6    equ     $e75a           CA1 via a, Keyboard interrupt
intv7    equ     $e75c           CA2 via a
intv8    equ     $e75e           T1  via b
intv9    equ     $e760           T2  via b
intv10   equ     $e762           CB1 via b
intv11   equ     $e764           CB2 via b
intv12   equ     $e766           SR  via b
intv13   equ     $e768           CA1 via b
intv14   equ     $e76a           CA2 Dvia b
intv15   equ     $e76c           ACIA interrupt
intv16   equ     $e76e           Software interrupt
outv6    equ     $e789           Output device 6
inpv6    equ     $e78b           Input device 6
outv7    equ     $e78d           Output device 7
inpv7    equ     $e78f           Input device 7
outv8    equ     $e791           Output device 8
inpv8    equ     $e793           Input device 8

start    equ     $e7ad           pointer to start of videoram
unrint   equ     $e7af           Unreconized interrupt vector
nmivec   equ     $e7b1           NMI vector
brkvec   equ     $e7b3           BREAK vector
irqvec   equ     $e7b5           IRQ vector

wrbeg    equ     $e776           Begin address mem. as output
wrend    equ     $e778           End address mem. as output
rebeg    equ     $e77a           Begin address mem. as input
reend    equ     $e77c           End address mem. as input
hours    equ     $e782           buffer for hours
min      equ     $e783           buffer for minutes
sec      equ     $e784           buffer for seconds
sec1.20  equ     $e737           count 1/20 seconds
keypnt   equ     $e7b7           pointer from keyboardbuffer
keybuf   equ     $e7b8           keyboard buffer, 80 char.
maxbuf   equ     $28             max. number of characters in buffer
inbuf    equ     $aa00           input-buffer
driv     equ     $abb4           a-drive buffer
utrk     equ     $abb6           x-track buffer
```

```
t1          equ         $f0                 parameters
t2          equ         t1+2
t3          equ         t1+4
t4          equ         t1+6
opt         equ         $fe                 option flag
mode        equ         $ff                 parameter flag
spnt        equ         $50                 two addresses for [spnt],y
sysb        equ         $e2                 dospointer to system sector
rwpoin      equ         $e8                 read/write pointer
esc         equ         $1b                 escape character
sp          equ         $20                 space
ff          equ         $0c                 formfeed
fb          equ         $14                 flag byte for x,y coordinates
SOH         equ         $01                 SOH Start of Heading (CC)
EOT         equ         $04                 EOT End of Transmission (CC)
ACK         equ         $06                 ACK Acknowledge (CC)
NAK         equ         $15                 NAK Negative Acknowledge (CC)
CAN         equ         $18                 CAN Cancel
EOF         equ         $1A                 SUB Substitute
```



```
            end
```

* * * * * * * * * * * * CRTC.DOC * * * * * * * * * * * * *

Enschede, 15 Jan 1988.

Het aansluiten van een monitor met een 9 pens "IBM" of "IBM cloon" stekker is in principe mogelijk op de VDU kaart van Electuur, alleen moeten er enkele wijzigingen aangebracht worden op de VDU kaart.

Als eerste kan men het beste profesorisch de volgende verbindingen aanbrengen op de VDU kaart.
Als eerste het VIDIO signaal afnemen van pen 12 IC3 (N22) en deze aansluiten op pen 7 van de 9 pens stekker.
Hor. sync. kan men direct afnemen van R5 (of van pen 39 IC11 (6545)) en aansluiten op pen 8 van de 9 pens stekker.
Vert. sync. moet eerst geinverteerd worden, afnemen van R6 (of van pen 40 IC11) en via een inverter (b.v pen 9 - 11 IC2) aansluiten op pen 9 van de 9 pens stekker.
Intensity, pen 6 van de 9 pens stekker, dient men aan te sluiten op V+ (+5V)
Hierna moet men de stekker alleen nog voorzien van een GND die men aansluit op pen 1 en 2 van de 9 pens stekker.

Het is niet nodig dat men enig component van de print verwijderd, zolang men alleen de nieuwe monitor aansluit.
Het is door deze tijdelijke oplossing nog steeds mogelijk Uw "oude" monitor op Uw systeem aan te sluiten.

Als U deze aansluitingen heeft aangebracht kunt U het systeem opstarten.
Als het beeld dat nu op Uw scherm verschijnt niet meer is dan een fractie van Uw totale beeldscherm oppervlak dient men als eerste een register van de CRTC aan te passen.
Dit is register 3 - H/V sync width -.
Vanuit I/O65 wordt hier de waarde $88 ingezet, deze waarde moest ik verhogen tot $8F om een redelijk beeld te krijgen.

Dit kan men b.v. op de volgende manier doen :
Met het nieuwe DOS commando CRTC, waarmee men de verschillende CRTC registers kan aanpassen.
De oude gebruikers van DOS65 zullen dit commando wel kennen, het was er vroeger namelijk ook alleen in een ander jasje, en wel van HAVIsoft.

Met het commando HELP CRTC krijgt een kleine uitleg van het commando CRTC zoals hieronder afgebeeld :

```
Function : change CRTC registers.
Syntax   : CRTC
Options  : No options.
Abbreviation : No abbreviation.
Use + and - to change and E or X to select register
n.b. : The values printed right on the screen are an copy from I/O65.
```

Als U het commando CRTC opstart dient U het volgende op Uw scherm te krijgen. :

```
 Change CRTC registers, select with E or X and change with + or -, quit with
       Register nr.  actual value              I/O65 value
       Register  0 : $      - Hor tot (N-1)        $7E
       Register  1 : $      - Hor Disp             $50
```

```
Register  2 : $      - Hor sync pos          $5F
Register  3 : $      - H/V sync width        $88
Register  4 : $      - Vert tot (N-1)        $1E
Register  5 : $      - Vert tot adj          $05
Register  6 : $      - Vert disp             $19
Register  7 : $      - Vert sync pos         $1C
Register  8 : $      - Mode control          $00
Register  9 : $      - Scanlines (N-1)       $09
Register 10 : $      - Cur start             $00
Register 11 : $      - Cur end               $09
Register 12 : $      - Start HI              $00
Register 13 : $      - Start LO              $00
Register 14 : $      - Cur HI                $00
Register 15 : $      - Cur LO                $00
                       ^
                       !
```

Op de lege plaatsen --- zullen natuurlijk de eerste keer dat U dit commando gebruikt de waarden uit I/O65 staan omdat U (als het goed is) nog niet Uw CRTC waarden heeft aangepast, en er nog geen file CRTC.DAT bestaat op Uw systeem schijf.
Als U enige registers van de CRTC heeft veranderd heeft U de mogelijkheid deze nieuwe waarden op te slaan in een file genaamd CRTC.DAT.
Iedere keer dat U het systeem nu opnieuw opstart kunt U met het commando SETCRTC deze nieuwe CRTC waarden in de registers copieren.
Het is dan natuurlijk het logische gevolg dat U het commando SETCRTC in Uw LOGIN.COM zet zodat deze waarden automatisch in de CRTC registers gecopieerd worden zodra U het systeem opstart.

Het is ntuurlijk ook mogelijk om, als U de juiste waarden heeft bepaald een nieuwe I/O65 eprom aanmaakt (laat maken).

Indien deze beschrijving niet geheel correct is verzoek ik een ieder mij hierop te wijzen zodat ik dit kan aanpassen.
Ook is iedere andere oplossing natuurlijk van harte welkom.

Ik hoop dat deze beschrijving duidelijk genoeg is, en dat er geen fouten in verwerkt zijn.....

J.H.G.M. Banser
BANSOFT
Haaksbergerstraat 199
7513 EM Enschede
tel : 053-324137