Wieder eine tolle Nachricht aus den Niederlanden: Das Projekt Harddisk am Spectrum steht ... und funktioniert. Unser Dank geht an Roelof Koning, der uns die nachfolgenden Informationen zusandte. Leider teilt er mit, daß es keine Serienproduktion geben wird. somit muß also jeder, den dieses Interface interessiert, es sich selber nachbauen (bzw. bauen lassen). Damit es keine Übersetzungsfehler gibt, bringen wir den Artikel im Original (also in englisch). Wir sind aber ganz sicher, daß Roelof bei Nachfragen den Interessierten gerne Auskunft geben wird.

The IDE Harddisk Connector

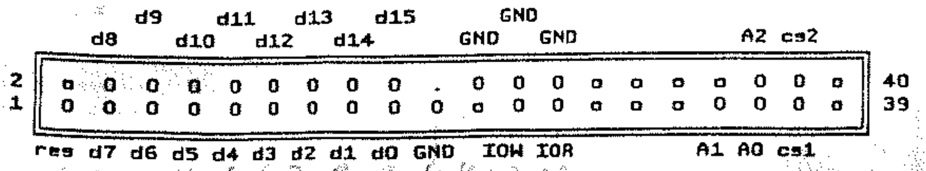
Looking at the circuitdiagram, you could see that besides 8 datalines, only READ, WRITE, and 2 'select'-lines are used to drive the system. These 'select'-lines comes from the adress decoders LS688, Pin 19. This means, that this system also could be driven otherwise. If only 4 signal wires for OUT are avalable. For example a fully wired printerport could do this job (alas, not on the Spectrum)? And maybe it is possible to use even fewer signals.

At last in the way I use this HD-System on Opus now, where the possible maximum speed is not used at all, one could think of returning to the 'PIO' idea. Replacing the 4 small 'datachips' by one PIO or PIA (like 6821?), doing the 8/16 bits conversion under software. So there is also room left for other inventors!

About compability (connectivity) with Plus D I can't tell you much at this moment, but of course this can change in one 'brainwave'.

It's a pity I don't know full details of harddisk references. As I added a R/C filter (470/330) in order to shorten the read-pulse given to HD, this solved some loading errors. So I suppose it had to do with timings. I can't measure these 'high frequencies'.

IDE Harddisk Connector (male)


d9 d11 d13 d15 GND
d8 d10 d12 d14 GND GND A2 cs2
2
1
40
39
res d7 d6 d5 d4 d3 d2 d1 d0 GND IOW IOR A1 A0 cs1

TASK adresses | COMMANDS (some) | STATUS bits | Error Register
res d7 d6 d5 d4 d3 d2 d1 d0 GND IOW IOR A1 A0 cs1

TASK adresses
240 = datareg.
241 = error
242 = sect cnt
243 = sectornr
244 = cylnr lo
245 = cylnr hi
246 = drv/head
247 = command
248 = status
223 = reset

COMMANDS (some)
236 = identify
18 = track 0
114 = seek trk
33 = read sect
49 = writ sect
228 = read buf
232 = writ buf

STATUS bits
0 comm error
1 index hole
2 ECC restored
3 waiting/rdy
4 seek ready
5 write fault
6 drive ready
7 busy
80 & 82 = O.K.

Error Register bits
0 No data addr. mark.
1 Track 0 error
2 Command aborted
4 sector ID error
6 ECC error
7 Bad block

Brief description:

Connecting a standard 16 bits IDE Harddisk (as used on PC) to the (8 bits) Spectrum.

The main purpose of this interface is 8/16 bits data conversion, when reading/writing from/to the DATA-register of the Harddisk. Although the other internal HD-registers are only 8 bits wide, these are not treated apart. Writing a 'dummy' byte is needed here to complete 16 bits. When reading, the 'first' byte is valid, and a dummy-read can be skipped when any OUT instruction on the 'registerport' follows. This 'registerport' is an IO portaddress through which access is gained to the many registers of the Harddisk. A register can be selected by OUT registerport, TASKaddress. Then the 'odd/even' (8/16) counter is reset, and data can be read from or written to this register, using the 'dataport'. Notice that the 'registerport' is OUT only.

On the Spectrum only two free IO addresses have to be found, and the dipswitches on the interface must be set accordingly. On a Spectrum + OPUS-combination the addresses 251 and 191 are O.K.

A special case is 'Reset'. Feeding this 'taskaddress' (223) to the 'registerport', does not reset the Harddisk immediately. A read from the dataport must follow in order to activate this command.

Example in BASIC:

```
 20 LET rp=191: LET dp=251              : REM registerport,dataport
100 OUT rp,BIN 11011111                 : REM a 'low' on the resetline
101 LET a=IN dp                         : REM make registerport active
102 OUT rp,BIN 11111111                 : REM all controls 'off'
103 LET a=IN dp: LET a=IN dp: PAUSE 30  : REM reset finished now
```

Harddisk Programming on Spectrum
The following example in BASIC will explain the programming.

```
REM --- read sector 14, head 2, on track 354, to adress 16384

 10 LET dp=251                          : REM 'Data-port'
 20 LET rp=191                          : REM 'Register-port'
 80 LET sc=14
 81 LET hd=2+160                        : REM BIN 10100xxx = drive 1, head xxx
 82 LET trhi=1
 83 LET trlo=96                         : REM 1*256 + 96 = 354
 84 LET adrhi=64
 85 LET adrlo=0                         : REM 64*256 + 0 = 16384
200 OUT rp,243                          : REM point to 'sectornumber'
201 OUT dp,sc                           : REM move byte into register
202 OUT dp,0                            : REM >> dummy data, not used
210 OUT rp,244                          : REM point to 'cylindernr.lo' register
211 OUT dp,trlo: OUT dp,0               : REM move low byte into register +
                                          dummy
212 OUT dp,0                            : REM dummy, just complete 16 bits data
220 OUT rp,245                          : REM point to 'cylindernr.hi' register
221 OUT dp,trhi: OUT dp,0               : REM move high byte into register +
                                          dummy
230 OUT rp,246                          : REM point to 'drive/head' register
231 OUT dp,hd: OUT dp,0                 : REM move byte into register + dummy
240 OUT rp,242                          : REM point to 'nr. of sectors' register
```

```
220 OUT rp,245                          : REM point to 'cylindernr.hi' register
221 OUT dp,trhi: OUT dp,0               : REM move high byte into register +
                                          dummy
230 OUT rp,246                          : REM point to 'drive/head' register
231 OUT dp,hd: OUT dp,0                 : REM move byte into register + dummy
240 OUT rp,242                          : REM point to 'nr. of sectors' register
```

```
241 OUT dp,1: OUT dp,0                  : REM signal '1 sector only' + dummy
250 OUT rp,247                          : REM point to command register
251 OUT dp,33: OUT dp,33                : REM give read sector command + dummy
260 OUT rp,247                          : REM point to the status register
261 IF IN dp >127 THEN GO TO 260        : REM readStatus + testBusy!
REM Leave the dummy, as any OUT-rp resets the 8/16 counter!
270 OUT rp,240                          : REM point to the dataregister now!
280 RANDOMIZE USR readblock             : REM fetch 512 databytes from HD
290 OUT rp,247: LET status=IN dp: LET dummy=IN dp
291 IF stat=80 OR stat=82 THEN PRINT "O.K.": STOP
292 PRINT "Error: ";stat: STOP
```

At readblock: POKE the machinecode to read 512 bytes from Harddisk.

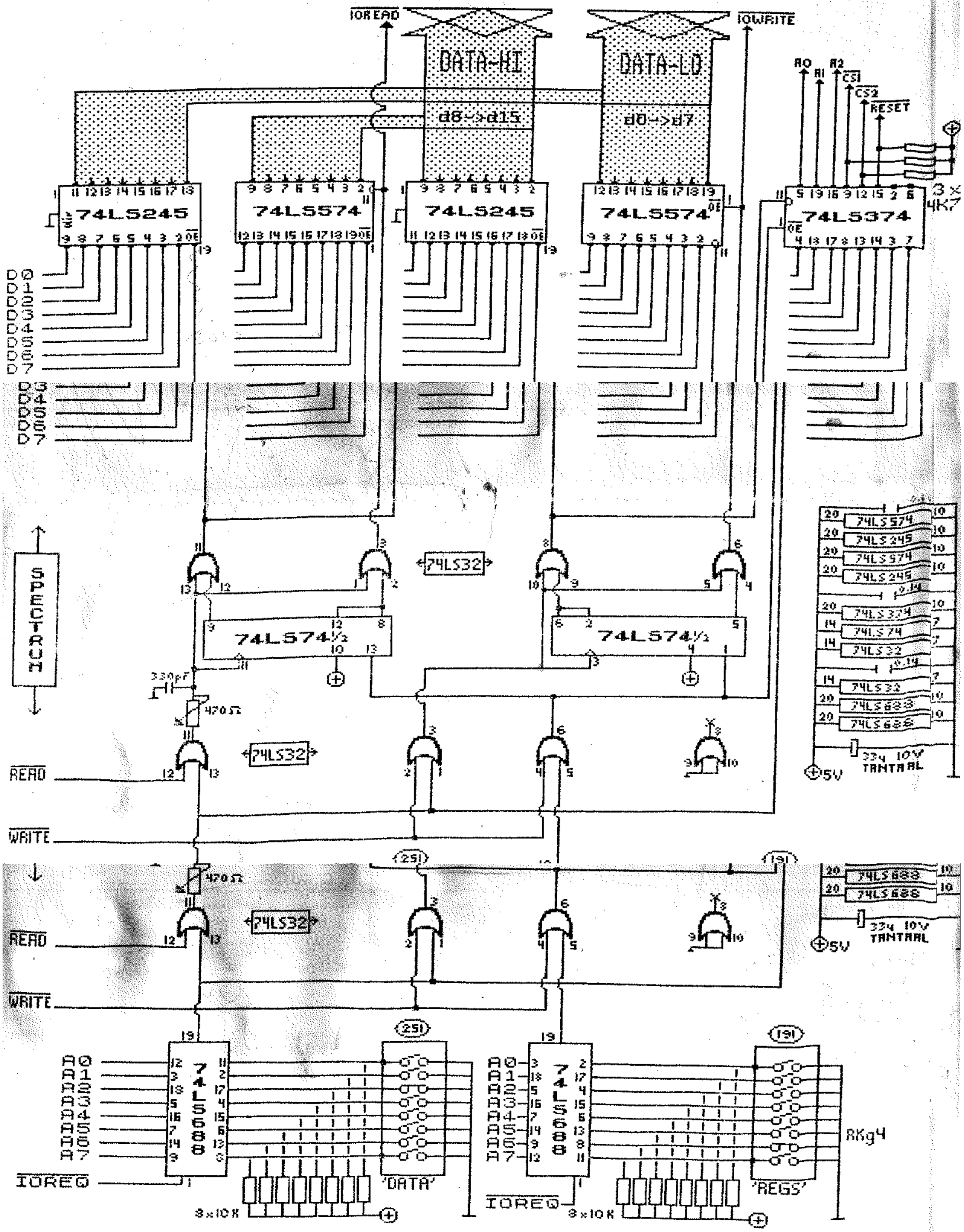
```
243                ; DI
33,adrlo,adrhi     ; LD HL,adr.      ---- memoryaddress to load to
1,dp,0             ; LD BC,dataport ---- C=port, B=0=256!
237,178            ; INIR            ---- move B.bytes to (HL)
237,178            ; INIR            ---- B reached 0,=256 again!
251                ; EI
201                ; RET
```

There should be an identical 'writeblock' routine, where both INIR's (237,178) are replaced by OTIR's (237,179).

Note: Inside a 'full' program the testing of statusbits might be done in a more elaborate way.

Roelof Koning (SGG), Selwerderstr. 26, NL-9717 GK Groningen, Holland

CONNECTOR IDE



Dieser Schaltplan wurde von uns nach dem handgezeichneten Original von Roelof Koning nachgezeichnet. Für Übertragungsfehler oder Fehler im Original übernehmen wir keine Haftung!

This circuit diagram was painted by us under using of the handpainted original from Roelof Koning. For errors by transfering or in the original we will not be liable for.