Issue Two ****************************** October 1996

The International ZX81 Magazine

****

Hello Dear ZX81 Friends,

Welcome to this second issue of The International ZX81 Magazine.
Isn't it great, this feeling that people all over the world are using the same computer you use, even in this age of modern technology with systems which work a 1000 times faster than this little black beauty you have.
The question is... what do you all need it for, this absurd power, with layers of many megabytes which takes a livetime to develop and understand, I'm all for simplicity.
Well, those systems can be nice for a tool, like making a article or communications, but the real work for us hobbyists is those little machines.
Being in touch with the Internet for almost a year now I find that much interest is found on the ZX81/TS1000 hardware side, where modifications are done to the hardware, and things are attached, things which I have absolutely no knowledge off, but the need is there, there is a wealth of information and schematics all over the world and many of you have it, so my motto is, 'let others also enjoy this wealth, so share it with others' There are those on the Internet and out there around the world who do make things, attach things, to do more with their ZX81.
Some of those already send in articles which we are really grateful for, we need people like Carlo Delhez, Jack Raats, Simeon Dwyer, Dan Erickson, Wilf Rigter, who do know how things work so we can follow in their footsteps and be able to build and program from the examples they share.
In this issue you can find some of these articles, a IO card for the ZX81 with schematic by Jack Raats, a machinecode course by Carlo Delhez, using a mouse with the ZX81, and the rest... I won't tell you, see for yourself.
I'm very happy you stick with us, even though there where some little difficulties, and I'm very happy to stick with you, the International ZX81 Magazine does continue to support all ZX81/TS1000 users all over the world, with Peter Liebert_Adelt to put this E-Mail Magazine in paperform for those not connected to the Internet (or like to hold it in their hands), and me taking care of the E-Mail version and helping out Peter with the paper version.
Thanks to all those who replied, and all the encouragement you give us.

The author of The International ZX81 Magazine

Martin van der Zwan
Moerweg 508
2531 BL Den Haag
Netherlands
mvdzwan@syncnet.xs4all.nl

****
The ZX81/TS1000 PIO
****

Martin asked me some weeks ago to write something for his international magazine. So here is my first (maybe my last :-)) article.

Interface 3.

The ultimate interface for the ZX Spectrum and ZX81.
Yes, You've read it good. This interface can be used on the ZX Spectrum and the ZX-81!

The heart of the interface is the Z80 PIO.

=> Connections between the Z80 PIO and the ZX

D0-D7 PIO are connected with D0-D7 on the ZX
+5V PIO (pin 26) is connected to the 5 Volt line of the ZX.
GND PIO (pin 11) is connected with the ground of the ZX
IEI PIO (pin 24) is connected via 10K with the 5 Volt line.
M1 PIO (pin 37) is connected with the M1 of the ZX
RD PIO (pin 35) is connected with the RD of the ZX
CLK PIO (pin 25) is connected with the CLK of the ZX

=> Connections between the Z80 PIO and the outside world.

PA0-PA7 PIO are connected with the data line of the centronics port
through an resistor array of 8 x 10K
PA0-PA4 PIO are also connected to the kempston joystick connector
PB0 PIO is used as RS232 TX output
PB1 PIO is used as STROBE for the centronics interface
PB2 PIO is used as RS232 RTS/DTR output
PB3 PIO not used
PB4 PIO not used
PB5 PIO is used as RS232 CTS/DSR input
PB6 PIO is used as BUSY for the centronics interface
PB7 PIO is used as RS232 RX input

PB5 PIO and PB7 PIO are also connected to ground by a 10K R and parallel to it a zenerdiode from 4.7V

=> Connections between the Z80 PIO and the ZX which makes the addresses

IORQ PIO (pin 36) is connected with the IORQ of the ZX
CE PIO (pin 4) is connected with the A7 of the ZX
C/D PIO (pin 5) is connected with the A6 of the ZX
B/A PIO (pin 6) is connected with the A5 of the ZX

This makes the following addresses

Port A data register = CE low -> A7 low
C/D low -> A6 low
B/A low -> A5 low
IORQ low
This makes I/O address 31

```
in assembler    PAD equ 31

Port A control register = CE low   -> A7 low
                          C/D high -> A6 high
                          B/A low  -> A5 low
                          IORQ low
                          This makes I/O address 95
in assembler    PAC equ 95

Port B data register = CE low  -> A7 low
                       C/D low -> A6 low
                       B/A low -> A5 high
                       IORQ low
                       This makes I/O address 63
in assembler    PBD equ 63

Port B control register = CE low   -> A7 low
                          C/D high -> A6 high
                          B/A low  -> A5 high
                          IORQ low
                          This makes I/O address 127
in assembler    PBC equ 127
```

THE JOYSTICK INTERFACE OF INTERFACE 3

A. ZX Spectrum
This can be done in Spectrum basic by OUT 95,127 This makes the Z80 PIO port A to act as input lines. Connect the joystick and run the following short program, while playing with the stick.

```
1 OUT 95,127
2 PRINT AT 5,0;IN 31
3 GO TO 2
```

You'll see different numbers between 0 and 31.

B. ZX 81
This only can be done in machine code.
Insert the following in your favorite assembler.

```
PAC     EQU 95
PAD     EQU 31

INIT    LD A,127
        OUT (PAC),A
        RET

READ    IN A,(PAD)
        LD B,0
        LD C,A
        RET
```

assemble it

RAND USR INIT will initialize the Z80 PIO.
PRINT AT 5,0;USR READ will display numbers between 0 and 31 while playing with the stick.

THE CENTRONICS INTERFACE OF INTERFACE 3

A. ZX Spectrum

```
; Here is the assembler source for a very simple driver.
; First we change the printer output address in the CHANNEL INFORMATION
; area (=INIT), then we initialize the PIO (=INIT2)

INIT    LD   HL,(23631)
        LD   DE,15
        ADD  HL,DE
        LD   DE,FILTER
        LD   (HL),E
        INC  HL
        LD   (HL),D
INIT2   LD   A,63
        OUT  (PAC),A        ;MAKES PORT A OUTPUT
        LD   A,255
        OUT  (PBC),A
        LD   A,240
        OUT  (PBC),A        ;MAKES PORT B0-B3 OUTPUT
        LD   A,2                ;MAKES PORT B4-B7 INPUT
        OUT  (PBD),A        ;MAKES STROBE HIGH
        RET

FILTER  CP   #A5
        JR   C,NTOKEN
        SUB  #A5
        JP   #0C10
NTOKEN  RES  0,(IY+1)
        CP   #7F
        JR   C,NGRAPH
        LD   A,"?"
NGRAPH  CP   #0D
        JR   NZ,NOTCR
        CALL OUTPUT
        LD   A,#0A
        CALL OUTPUT
        RET
NOTCR   CP   #20
        RET  C
        JR   NZ,OUTPUT
        SET  0,(IY+1)

OUTPUT  PUSH AF
BUSY    CALL BREAK
        JP   NC,#0D00
        IN   A,(PBD)
        BIT  6,A
        JR   NZ,BUSY
        POP  AF
        OUT  (PAD),A
        XOR  A
        OUT  (PBD),A
        LD   A,2
        OUT  (PBD),A
        RET
```

B. ZX-81
Assemble the source below with the artic assembler

```
      ;ASSEMBLY SOURCE
      ;FOR PRINTING WITH
      ;INTERFACE-3 + ZX81

      JR INIT

START CALL CRLF
      CALL BYTE
      LD DE,(+16396)
COPY  CALL NEXT
      LD HL,(+16396)
      LD BC,+726
      AND A
      ADC HL,BC
      AND A
      SBC HL,DE
      RET Z
      LD A,(DE)
      CP 76
      CALL Z CRLF
      CALL BYTE
      JR COPY
NEXT  INC DE
BREAK LD A,7F
      IN A,FE
      RRA
      RET C
      RST 8
      22
CRLF  LD A,5
      CALL BYTE
      LD A,4
      RET
BYTE  CP 40
      RET P
      LD HL,TABLE
      LD C,A
      LD B,0
      ADD HL,BC
      LD A,(HL)
OUTPT PUSH AF
BUSY  CALL BREAK
      IN   A,3F
      BIT  6,A
      JR NZ BUSY
      POP AF
      OUT 1F,A
      XOR A
      OUT 3F,A
      LD A,2
      OUT 3F,A
      RET

INIT  LD A,3F
      OUT 5F,A
```

```
      LD A,FF
      OUT 7F,A
      LD A,F0
      OUT 7F,A
      LD A,2
      OUT 3F,A
      RET

TABLE 20 23 20 20
      0A 0D 07 20
      23 5F 26 22
      40 24 3A 3F
      28 29 3E 3C
      3D 2B 2D 2A
      2F 3B 2C 2E
      30 31 32 33
      34 35 36 37
      38 39 41 42
      43 44 45 46
      47 48 49 4A
      4B 4C 4D 4E
      4F 50 51 52
      53 54 55 56
      57 58 59 5A

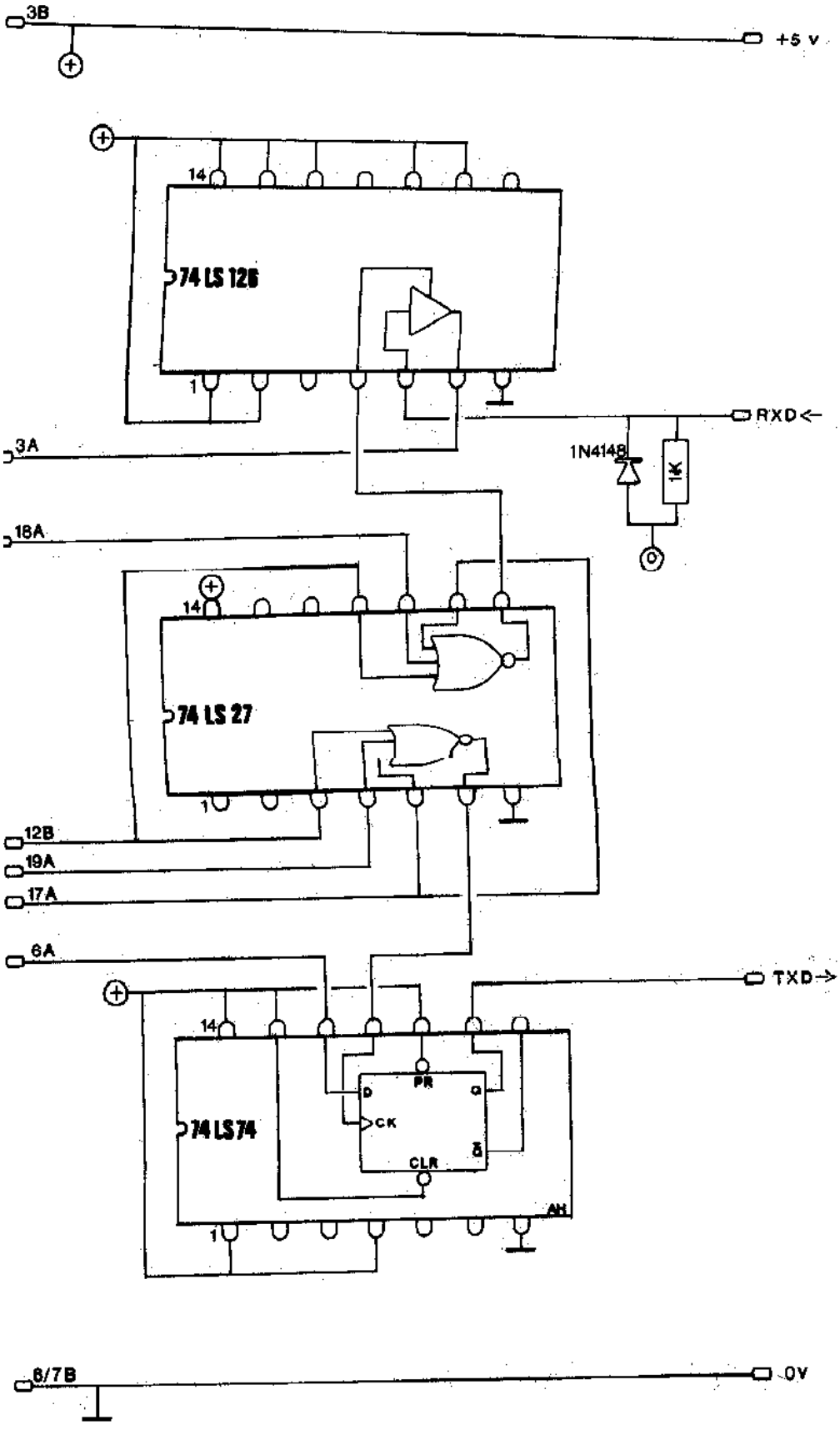
      ;END OF SOURCE
```

An example to use this routine

```
10 RAND USR INIT
20 FOR N=1 TO 20
30 PRINT "HELLO WORLD"
40 NEXT N
50 RAND USR START
60 STOP
```

TO BE CONTINUED. (Jack Raats)

Editors note: (with the E-Mail version the ZX81 program ZX81PIO.P is included, the schematic is in ZX81PIO.GIF)


3B
+5 v
14
74 LS 126
1
RXD←
3A
1N4148
1K
18A
14
74 LS 27
1
12B
19A
17A
6A
TXD→
14
PR
D
Q
CK
CLR
74 LS74
1
6/7B
0V

****
Cheap High Re(solution)
****

For many ZX81/TS1000 owners the Spectrum High Resolution is something to be jealous of.

If you calculate what with the ZX81 in reality is the resolution then you start thinking by yourself why Sinclair for this Computer has forgotten (left out) a high resolution mode.

Besides color it's very simple to give the ZX81 the same resolution as the Spectrum, all it needs is one little IC and a little program.

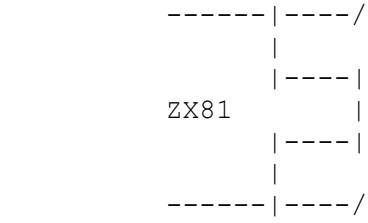
This program called HIRES takes up so little memory space that on a ZX81 with 16k you have the same memory size as on a 16k ZX Spectrum.

HIRES gives you commands like PLOT, DRAW and ELLIPS, and for extra convenience are those commands a direct extension of the basic subset.

To compare it with existing software based highres programs is HIRES much faster and easier to use, also it is very cheap compared with HRG modules.

Here is how to make it:

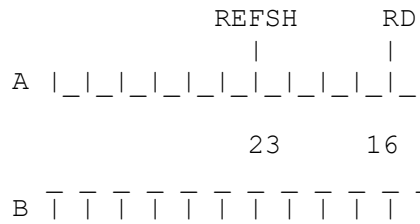
Schematic:



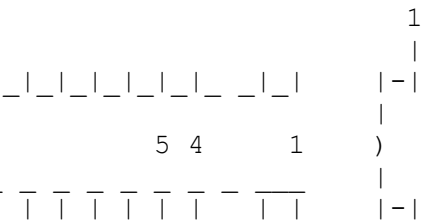
Pin layout:

IC Pin		Connector Pin
1	to	23A of the ZX81
2	to	16A of the ZX81
3	to	16A of the 16K Rampack
7	to	4B or 5B
14	to	1B

Edge Connector:



Chip to use:



This may look to technical for the non technical amongst us (including me) but I assure you, it isn't as difficult as it looks.

You just need to connect the 5 wires (3 for the gate inside the chip and 2 for the electricity to activate the chip) from the chip to the connector.

If you do this inside the ZX81/TS1000 make sure you don't do it all the way to the end of the connector but as far as possible in the back (towards the ZX81) so the 16k Rampack still slides nicely on the ZX81's connector.

You need to carefully cut the 23A track with a little sharp knife (this is the track (line) which is the most left on the component side of the ZX81, when you are opening the ZX81 and hold the connector towards you, you

can see this above, (edge connector) this is the layout on the back of the ZX81.

Be careful not to damage any nearby lines, they are very close to each other.

Now you must also cut (just a small cut to make sure the ZX81 does not connect this line to the Rampack) the 16A line (see above), it's pin 8 counted from 23A to the right.

Now you solder wire 1 from pin1 (see above, chip to use) of the 74LS08 and the other end to line 23A from the ZX81 (that's the line which goes to the ZX81, not to the Rampack, remember... you just had to cut it).

Wire 2 you solder on pin 2 of the chip and the other end you solder to 16A (the ZX81's side) of the ZX81.

Wire 3 you solder on pin 3 of the chip and the other end you solder to 16A of the ZX81 (this you should solder on the line going to the Rampack).

Now with these 3 wires we have just connected 1 gate of the 74LS08 to the ZX81 (the 74LS08 has four of these).

Let's give the chip some current with wire 4 and 5.

Wire 4 you solder on pin 7 of the chip (this is ground) and the other end you solder to pin 4B or 5B (look above, this is the bottom of the connector, the solder side of the ZX81's print).

Wire 4 you solder on pin 14 (this is plus) of the chip and solder the other end to 1B (that's the most right pin on the bottom of the connector, solder side of the ZX81).

For those of you who don't want to open the ZX81 it is also possible to place this chip inside the 16k Rampack, now it works fine with the normal 16k Sinclair or TS1000 Rampacks I haven't tried other Rampacks yet.

You must then cut the same lines and connect the wires, you could also, if you don't want to open anything use a flatcable or print with male and female connector on which you can place the chip in between the ZX81 and the Rampack.

Note:
It's better to use the 74LS08 instead of the 7408, because the 7408 works slower and uses more electricity.
The program is included with the magazine and called HR-TOOL.P

Here are the commands to use with the program.

RAND USR 16514 Starts interpreter
RAND USR 16519 Restarts interpreter, this is needed after a INPUT command because this command turns off the interpreter.
RAND USR 16522 Switches to Hires screen.

Note: I have translated this article and adapted it for the magazine, which was originally by Enno Borgsteede.

Good luck,
Martin van der Zwan

Z X 8 1 - W E A R E R E A D Y N O W
****

****
D.H.Z. Programming in Machinecode
****

Many computerusers think that programming in machinecode (MC) is much more difficult than programming in Basic.

My idea is, it isn't.

In this series I want to show you how easy it is to program the ZX81 in MC with some practical examples.

I will also compare it with Basic by the length of the MC and Basic routine.

The easier way to program in machinecode is to use a assembler, it is also possible to develop a MC program on paper, and then manually translate it to hexcodes and then to type them in, but this way you spend more time on uninteresting searchwork instead of programming.

Also, if you forget a instruction somewhere halfway you're code you must translate you're whole program again manually, with a assembler this works automatically and many times faster.

I use a assembler I completely rewrote myself, the Artic ZX-Assembler, this assembler has a excellent editor build inside, which you can use to type in you're program (typing in you're source, and then also assemble, this means, turning it into hexcode) I will present the example program in this article as MC instructions, and not the translation in hexcodes, so the reader of this article can type it in a assembler to experiment with.

The artic Assembler uses all numbers as hexcodes, unless a value starts with a "+" or "-" sign, in that case they are decimal.

In this series I will do likewise, so: 20 (hex)=+32(dec) and +20(dec)= 14(hex).

I want to close this first introduction part with a small routine to display the first 64 characters of the characterset, this could look like this (DHZMC01):

```
        XOR A     ; begin with A=0
LOOP    RST 10    ; PRINT CHR$(A)
        INC A     ; A=A+1
        CP +64    ; at 64 yet?
        RET Z     ; yes: then stop
        JR LOOP   ; no: then continue
```

'XOR' is a handy instruction to set register A to zero, 'RST 10' calls the ROM address +16, there is at that location a routine to print CHR$(A) to the display, 'INC A' increases the register A with 1, 'CP +64' looks if A has reached +64 yet, 'RET Z' returns to Basic when A contains value +64, 'JR LOOP' jumps back to the 'RST 10' command that prints now CHR$(1) etc.

This machinecode program is about 45 times faster than the fastest possibility in Basic!, it also only spends 8 bytes memory. This must be convincing enough.

Carlo Delhez.

****
Using a geos mouse with the ZX81
********** from ZX Team Magazine

With a Geos mouse lightwave pulses are used internally which continue when the mouse is not moved.

In this situation 4 direction-bits have a high value.

To have this result also for the mousebuttons the keybit through a 10k resistor is connected with plus.

Now we can start with the with the software, you must call the right bit and move the cursor on the screen in the desired direction, and use for example the mousebutton to leave the program.

It isn't as easy as it looks, because with a mousemovement for example, upwards the mouse is also moved left and right (the human hand isn't that accurate), the software must find the preferred direction and ignore all other, unwanted movements.

That sounds complicated, but it isn't, here a example: someone moves the mouse upwards to move the cursor upwards, bit zero also goes low, because faulty movements go left and right, also bit 2 or 3 also go sometimes low.

Now we can write in software... the bit which goes the first a 100 times low is the right direction, eq the direction the cursor must move.

The value 100 we can also make variable and this has the next meaning:

if the value gets smaller the cursor moves faster on the screen but also more inaccurate (value is 5 times lower: bit 2 of 3 goes almost exactly 5 times on low, just like bit 0).

If the value get higher the cursor moves slower, but very accurate (value is 100, bit 2 of 3 isn't that accurate 100 times on low, as bit 0, the desired direction) When established the desired direction, for example a bit went 100 times low then the cursor can be directed in that position.

Also the counter must count this continuously, how many times a bit goes low be set to zero, and then it starts all over again.

Here is the program:

```
4082 ;----------------------
4082 ;-                     -
4082 ;-     GEOS-MAUS       -
4082 ;-  SCHITEC SOFTWARE   -
4082 ;-   TEL:09931/5181    -
4082 ;-                     -
4082 ;----------------------
4082 ;;
4082 ; DIMENSIONEN
4082 ;
4082 ZEILU00                        EINSTELLUNG DES BEREICHS, IN DEM SICH DER
4083 ZEILD15                        KURSOR BEWEGEN SOLL.
4084 SPALL00
4085 SPALR1F
4086 ;;
4086 UP   00                        ZAEHLER FUER DIE VIER RICHTUNGS-BITS, WIE
4087 DOWN 00                        OFT DIESE BITS AUF LOW GEHEN.
4088 LEFT 00
4089 RIGHT00
408A ;;
408A ; INITIALISIERUNG
408A ;;
408A      LD A,$02       3E02     INITIALISIERUNG DES PORTS (8255)
408C      LD ($0008),A   320800
408F      LD A,$F0       3EF0
4091      LD ($C021),A   3221C0
4094      LD A,$9B       3E9B
4096      LD (STRG),A    320330
4099 ;;
4099      LD HL,(DFILE)  2A0C40   KURSOR AUSGEBEN, HIER AM OBEREN LINKEN
409C      INC HL         23       BILDSCHIRMRAND.WIRD DER BEREICH GEAENDERT,
409D      CALL CRSR      CDF240   MUSS HIER ENTSPRECHEND DIE KURSORPOSITION
40A0 ;;                           GEANDERT WERDEN.
```

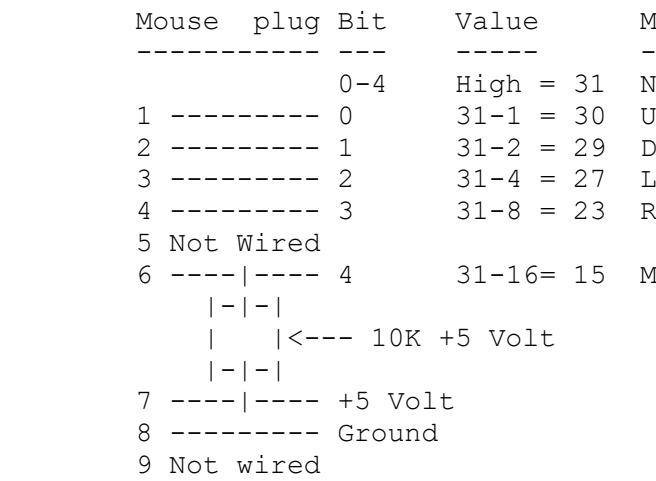
```
40A0 ; B=ZEILE->ZEILU
40A0 ; C=SPALTE->SPALL
40A0 ;;
40A0      LD BC,$0000    010000  BC-REGISTER IST ZEILEN- UND SPALTEN-
40A3 ;;                          ZAEHLER.MUSS BEI AENDERUNG DES BEREICHS
40A3 ; HAUPTSCHLEIFE             ENTSPRECHEND GEAENDERT WERDEN.
40A3 ;;
40A3 MAUS ZOR A          AF      VIER RICHTUNGS-BIT-ZAEHLER MIT NULL LADEN.
40A4      LD (UP),A      328640
40A7      LD (DOWN),A    328740
40AA      LD (LEFT),A    328840
40AD      LD (RIGHT),A   328940
40B0 ;;
40B0 M1   PUSH HL        E5
40B1      PUSH BC        C5
40B2      LD HL,UP       218640  HL MIT ERSTEM RICHTUNGS-BIT-ZAEHLER LADEN.
40B5      LD B,$04       0604
40B7      LH A,(PORT)    3A0030  PORT EINLESEN
40BA      CPL            2F      UND KOMPLEMENTIEREN DE AKKUS.
40BB      LD C,A         4F      KOMPLEMENTIERTEN PORT-WERT NACH C LADEN.
40BC M2   LD A,C         79
40BD      AND $01        E601    WERT VON 0 IN AKKU.
40BF      ADD A,(HL)     86      RICHTUNGS-BIT-ZAEHLER ZU AKKU HINZU-
40C0      LD (HL),A      77      ADDIEREN UND ZEAHLER MIT NEUEM WERT LADEN.
40C1      CP COUNT       FE7D    VERGLEICHEN VON ZAEHLER MIT VARIABLE.
40C3      JR Z,M3        2805    JA, DANN SPRUNG ZU M3
40C5      INC HL         23      NEIN, DANN RICHTUNGS-BIT-ZAEHLER ERHOEHEN
40C6      RRC C          CB09    UND NAECHSTES RICHTUNGS-BIT VORBEREITEN.
40C8      DJNZ M2        10F2    WIEDER BEI M2 BEGINNEN.
40CA M3   POP BC         C1
40CB      POP HL         E1
40CC      LD A,(PORT)    3A0030  MAUS-TASTE ABFRAGEN.
40CF      BIT 4,A        CB67
40D1      JP Z,ENDE      CA4441
40D4 ;;
40D4      LD A,(UP)      3A8640  VORZUGSRICHTUNG BESTIMMEN UND ENTSPRECH-
40D7      CP COUNT       FE7D    ENDE KURSORBEWEGUNG DURCHFUEHREN.
40D9      JR Z,CU        281C    HAT NOCH KEIN RICHTUNGS-BIT-ZAEHLER DEN
40DB      LD A,(DOWN)    3A8740  VERGLEICHSWERT ERREICHT, WIRD ERNEUT DER
40DE      CP COUNT       FE7D    PORT EINGELESEN.
40E0      JR Z,CD        282A
40E2      LD A,(LEFT)    3A8840
40E5      CP COUNT       FE7D
40E7      JR Z,CL        2837
40E9      LD A,(RICHT)   3A8940
40EC      CP COUNT       FE7D
40EE      JR Z,CR        2842
40F0 ;;
40F0      JR M1          18BE    WIEDER VON VORNE BEGINNEN.
40F2 ;;
40F2 ; KURSOR AUSGEBEN
40F2 ;;
40F2 CRSR LD A,(HL)      7E      INVERIERUNG DES ZIECHENS AUF DEM BILD-
40F3      SUB $80        D680    SHIRM.
40F5      LD (HL),A      77
40F6      RET            C9
40F7 ;;
40F7 ; KURSOR AUF
```

```
40F7 ;;
40F7 CU   LD A,(ZEILU)   3A8240  UEBERPRUEFUNG OB KURSOR CHON OBEN IST.
40FA      CP B           B8      JA, DANN RUECKSPRUNG. NEIN, DANN 33 VON HL
40FB      JR Z,MAUS      28A6    ABZEIHEN UND KURSOR WIEDER AUGEBEN.
40FD      CALL CRSR      CDF240
4100      LD DE,$0021    112100
4103      XOR A          AF
4104      SBC HL,DE      ED52
4106      CALL CRSR      CDF240
4109      DEC B          05
410A      JR MAUS        1897
410C ;;
410C ; KURSOR AB
410C ;;
410C CD   LD A,(ZEILD)   3A8340  UEBERPRUEFUNG OB KURSOR SCHON UNTEN IST.
410F      CP B           B8      JA, DANN RUECKSPRUNG.NEIN, DANN 33 ZU HL
4110      JR Z,MAUS      2891    ADDIEREN UND KURSOR WIEDER AUSGEBEN.
4112      CALL CRSR      CDF240
4115      LD DE,$0021    112100
4118      ADD HL,DE      19
4119      CALL CRSR      CDF240
411C      INC B          04
411D      JP MAUS        C3A340
4120 ;;
4120 ; KURSOR LINKS
4120 ;;
4120 CL   LD A,(SPALL)   3A8440  UEBERPRUEFUNG OB KRSOR SCHON LINKS IST.
4123      CP C           B9      JA, DANN RUECKSPRUNG.NEIN, DANN HL ER-
4124      JP Z,MAUS      CAA340  NEIDRIGEN UND KURSOR WIEDER AUSGEBEN.
4127      CALL CRSR      CDF240
412A      DEC HL         2B
412B      CALL CRSR      CDF240
412E      DEC C          0D
412F      JP MAUS        C3A340
4132 ;;
4132 ; KURSOR RECHTS
4132 ;;
4132 CR   LD A,(SPALR)   3A8540  UEBERPRUEFUNG OB KURSOR SCHON RECHTS.
4135      CP C           B9      JA, DANN RUECKSPRUNG.NEIN, DANN HL ER-
4136      JP Z,MAUS      CAA340  HOEHEN UND KURSOR WIEDER AUSGEBEN.
4139      CALL CRSR      CDF240
413C      INC HL         23
413D      CALL CRSR      CDF240
4140      INC C          0C
4141      JP MAUS        C3A340
4144 ;;
4144 ; PROGRAMMENDE
4144 ;;
4144 ENDE CALL CRSR      CDF240
4147      RET            C9
```

The hardware:

Hardware connection between Geos mouse and the ZX81 port.

Mouse <----> ZX81 Port



For those who use a ZX81 and want to use the programs on the ZX81
****

Included with this magazine is a utility from Wilf Rigter to be able to use the .P files on you're real ZX81.
The program is in the archive called ZXTAPE.ARJ (ZXTAPE.ZIP)

How to Subscribe (This info is for the paper version only!!)
****

You like this issue of the International ZX81 Magazine and want to get the next one?
Well we can't tell you yet when it will be published, we hope to edit the International ZX81 Magazine three or four time a year, but this will depend on You're support.
It is very cheap and easy to subscribe for the next issue.
The International ZX81 Magazine is free, you only have to pay for return postage, All you have to do is send you're address - please in good readable CAPITAL BLOCKLETTERS - and add:

in the Netherlands or Germany	1 stamp for a normal letter
inside Western Europe	1 US$ or 1 IRC (International reply coupon)
outside Western Europe	2 US$ or 1 IRC
by air mail outside	3 US$ or 2 IRC's

If you want to get it free at all, please read the internet newsgroup comp.sys.sinclair.
Martin will place an information with the server-address where you can get the International ZX81 Magazine.

Copyright - copyleft - no copywrong - copyduty!
****

Everyone has the right to copy this issue of the International ZX81 Magazine or parts of it for non commercial purposes, don't forget the reference. Please give it to other users of "Zeddy and family". but do not ask more than the cost for copying.

If you are a member of a computerclub, please write a short review of the International ZX81 Magazine for the next issue of your club magazine.

Do you have the possibility to place an (free) advertisement in computer-, electronic- or amateurradio-magazines??Do it, to support the International ZX81 Magazine.

All information in the International ZX81 Magazine is for personal use only. We can never ever guarantee for nothing and anything. Please observe you're national laws.

The International ZX81 Magazine Issue 2 May 1996

Editor (E-Mail version)
Articles, Adds, Ideas, Help, Questions, Comments
Martin van der Zwan
Moerweg 508
2531 BL, Den Haag
Netherlands
mvdzwan@syncnet.xs4all.nl

Editor (Paper version)
Subscription, Articles, Adds, Ideas, Help, Questions, Comments
Peter Liebert-Adelt
Luetzowstr. 3
D-38102 Braunschweig
Germany
P. Liebert@t-online.de