EXCHANGING BYTE DATA VIA THE EINSTEIN's USER PORT AND THE PC'S PRINTER PORT

By Chris Coxall

FORWARD

This document shows how the Author established full byte transfer between the Einstein's User Port and the IBM/PC's Printer Port. The objective is to show only how a byte or a series of bytes can be sent and received with handshaking. A pair of program listings are given one for BBCBASIC on the Einstein to send bytes, another for BBCBASIC FOR WINDOWS on a PC to receive the bytes sent. A second pair of programs give listings for BBCBASIC FOR WINDOWS to send and BBCBASIC on the Einstein to receive. Each pair of companion listings are given as separate identities so the BBCBASIC FOR WINDOWS programs are small enough to run in the free demo version of BBCBASIC FOR WINDOWS.

DISCLAIMER: The Author is not a a professional computer programmer or hardware engineer or regards himself to have much experience in these fields. Without any established programming found for connecting the Z80 PIO of the Einstein to the PC's ECP printer port for data transfer, or a hardware connection, the Author was left to his own initiatives to use trial and error to develop the hardware and programming in this document from the technical manuals that were available. The Author has incorporated both the send and receive listings into a single program for his own use. At times both the Einstein and PC were in send mode or both were in receive mode, no hardware, software damage or problems were incurred with Author's computers. It cannot be said this would be true for all computers therefore the Author disclaims any responsibility for damage that is caused by others using this programming or hardware adaptation described.

FURTHER DEVELOPMENT: It is hoped by the Author that other retro Z80 processor computer owners or project builders that have a Z80 PIO installed might find this document helpful in that it gives at least one example of how a parallel link and data exchange can be made. More experience, knowledge and programming could make better use of the ECP parallel port's own built in hardware handshaking. If this known how to be done by any readers of this document it would be helpful if this information, especially program listings, were made available and public in some way. It would certainly be valued by the Author.

Document Created 11 February 2009

EXCHANGING BYTE DATA VIA THE EINSTEIN's USER PORT AND THE PC'S PRINTER PORT

By Chris Coxall

This document shows two programming examples using the built in assembler of BBCBASIC for the Einstein to input and output bytes via its USER PORT. Programming requires configuring the Einstein's internal Z80 PIO IC.

Two programming listing examples are given for BBCBASIC FOR WINDOWS to run in a MS/WINDOWS PC to transmit and receive bytes which run in conjunction with the BBCBASIC programs given for the Einstein.

CONTENTS

PROGRAM LISTINGS

Programming on The PC using BBCBASIC FOR WINDOWS

The programming for BBCBASIC FOR WINDOWS has been kept to basic basic so to run in a window as would be expected to run in a full screen DOS version of basic or as would be expected to be how the Einstein or other vintage computers would display running programs. There has been no attempt to take advantage of BBCBASIC FOR WINDOWS GUI ability to use windows features such as dialogue boxes or drop down menus that can be used with BBCBASIC FOR WINDOWS.

An an exception to simple programming in the listings are system calls to third party WINIO support files to access computer ports directly when using XP and WIN2000. The WINIO files must be present in the same folder that the bbcbasic for windows files are saved to. The intention of simple programming is to show a basic guide which can be followed and understood by others who have a preference to use other programming for writing windows software. The two programs given are just small enough to run in the free Demo version of BBCBASIC FOR WINDOWS. The required third party WINIO support files come with the BBC4W installation package. See BBC4W HELP index-"Input/output using WINIO ".
Download BBCBASIC FOR WINDOWS http://www.rtrussell.co.uk/.

It might be possible for the two BBC4W programs given to exchange data by bytes between two windows PC's via their ECCP printer ports. This has not been tried. Download XP_IN.BBC and XP_OUT.BBC in a zip archive from
The Tatung Einstein Computer Website

Einstein Programming for the Z80 PIO and interrupt handshaking

•Basic principles for mode 0 output and mode 1 input.

Programming the Z80 PIO for mode 0 (eight bit output) or mode 1 (eight bit input) with hardware handshaking requires an interrupt routine. The interrupt is generated by a high or pulse high received on the STB pin of the Einstein's user port set by strobe from the PC's printer port. An output of a byte to the PIO's data port will set the RDY pin on the user port high to be received by Acknowledge on the PC's printer port for mode 0 output. For input mode 1 the RDY signal is active when the input register is empty and is ready to accept data from the peripheral device.

The BBCBASIC programs for the Einstein's user port U_IN_EN.BBC and U_OUT_EN.BBC have been put into an Einstein Disk Image. Download from the *Tatung Einstein Computer Website* . A 3.5 inch Einstein floppy can be made from the image in a PC floppy 3½"drive with CPCDiskXP software DOWNLOAD HERE http://www.cpcmania.com/.

Files in the disk image can be extracted from the image into PC folder using EDIP
Visit the *Tatung Einstein Computer Website*

WRITING TO THE Z80 PIO CONTROL PORT

•FOR MODE 0 and MODE 1
•WITH INTERRUPTS ENABLED

%VVXX1111 . Bits D3-D0 indicate "Set Mode".
Last 4 bits set to 1111 Signifies This Control Word is to set mode.
Bits D5 and D4 are ignored.
Bits D7 and D6 set to 01 sets PIO to mode 1 input
Bits D7 and D6 set to 00 sets PIO to mode 0 output

%VVVVVVV0 . Bit 0 set to 0 sets the low byte interrupt vector set by remaining bits.
D0 is used in this case as a flag bit which when low,
causes V7 through V1 to be loaded into the vector register.
0 in D0 Signifies This Control Word is an Interrupt Vector.
%00001100 interrupt vector low byte set to Hex 12.
HEX 12 is the low byte of a scratch pad address given by the Einstein's Machine Operating System at bootup.
The high byte needs to be in the Z80's I register. This is fixed to HEX FB by the Einstein at bootup.

%VVVVV111
Bits 0, 1 and 2 set to 111 This Signifies Control Word is to set interrupt.
%10000111 Bit 7 set to 1 enables interrupts.
A pdf user manual for the Z80 PIO can be downloaded here. http://www.z80.info/zip/z80piomn.pdf

More About The Einstein's Z80 PIO

The Z80 PIO has two ports for 8Bit input and output: Port A and B. On the Einstein port A is configured to be compatible as a centronic's printer port at power up. Electronically this is set as mode 0 output but has a hardware adaptor of a monostable which sets ARDY high as a 1us pulse. With the User Port the BRDY is held high until a high is received by PIO's BSTB pin.

Physically the Einstein's printer port pin out in the manual describes STROBE which is connected to the PIO's ASTB and ACKNOWLEDGE which connects to the PIO's ARDY. The printer port's BUSY, PE and ERROR connect to the Einstein's command/status register.

If the Einstein's printer port is used instead of the user port any of the Einstein's software applications print commands will transfer output data but Additional Programming would be needed for input.

THE IBM/PC ECP PRINTER PORT

Most IBM/PCs today have ECP (Enhanced Capabilities Port) installed. It has number of modes it can be programmed for. For any mode chosen the ECP's Extended Control Register (ECR) has to be configured for the mode selected before other printer port registers can be programmed. From information placed on the internet the modes are listed below.

Modes of Operation

•Standard Mode
Selecting this mode will cause the ECP port to behave as a Standard Parallel Port, without Bi-directional functionality.

•Byte Mode / PS/2 Mode
Behaves as a SPP in Bi-directional (Reverse) mode.

•Parallel Port FIFO Mode
In this mode, any data written to the Data FIFO will be sent to the peripheral using the SPP Handshake. The hardware will generate the handshaking required. Useful with non-ECP devices such as Printers. You can have some of the features of ECP like FIFO buffers and hardware generation of handshaking but? with the existing SPP? handshake instead of the ECP? Handshake.

•ECP FIFO Mode
Standard Mode for ECP Use. This mode uses the ECP Handshake, already described.

•EPP Mode/Reserved
On some chipsets, this mode will enable EPP to be used. While on others, this mode is still reserved.
Reserved Currently Reserved

•FIFO Test Mode
While in this mode, any data written to the Test FIFO Register will be placed into the FIFO and any data read from the Test FIFO register will be read from the FIFO? buffer. The

•FIFO Full/Empty
Status Bits will reflect their true value, thus FIFO depth, among other things can be determined in this mode.

•Configuration Mode
In this mode, the two configuration registers, cnfgA & cnfgB become available at their designated Register Addresses.

The Mode Used For This Programming
Except for the first two modes the remaining are above the knowledge, grasp and programming information found by the Author at the present time for writing. Of the two Byte / PS/2 mode and using polling loops for handshakes seems the simpler and most manageable for bidirectional transfer at the present time.

Using Byte/PS/2 Mode to Input and Output Data

The way the Printer Ports are Read and Written To In This Programming.

ONCE BEFORE BYTE TRANSFER .
To set ECP port to byte mode %001XXXXX needs to be written to the ECR (Enhanced Capabilities port Register) this is located at hexadecimal &402 1026 decimal bytes above the base address. The first XXXXX (0 to 4) bit values will not effect mode selection. The last 3 bits (5 to 7) set the mode and 001 sets byte / PS/2 mode. The ECR byte is read an AND %00111111 then an OR %00100000 is implemented then the byte written back to the ECR.

DATA REGISTER (Continually used for the bytes being transferred**).**
The base address usually at hexadecimal &378 is also the address of the data register where imported and exported bytes are read from and written to.

CONTINUALLY DURING BYTE TRANSFER
The status register at hexadecimal &378+1. With this programming only bit 6 Acknowledge is read from this register by reading the status register byte then performing AND %01000000 function. . In input mode it is polled in a loop until Acknowledge has been put high by a handshake from the peripheral to say its ready to send a data byte. In output mode Acknowledge is polled until set by the peripheral to say it is ready to receive a data byte. To comply with a handshake from a peripheral that uses a pulse to Acknowledge, instead of holding high until it has received a handshake back from the PC, a pulse is used to set a latch ("polebyte" a byte in memory in this programming) to 1 which has to be reset to 0 by the PC programming loop when it is ready to continue to receive or send data.

The Control Register at base hexadecimal &378+2 has only three bits used with this programming.

ONCE BEFORE BYTE TRANSFER
BIT 5 SETS LPT1 PRINTER PORT FOR INPUT or OUTPUT (0 for output 1 for input)
BIT 4 set to 0 DISABLES INTERRUPTS
BIT 0 SETS STROBE LINE LOW or to HIGH (strobe hardware inverted) 0 for high 1 for low. Bits 4 has to be kept to 0 so not to have interrupts enabled and 5 will have to remain at 1 for the input program and 0 for the output program.

To initial the control port for the receiving the port is read, an OR %00100001 then an AND %11101111 is implemented before writing the byte back to the control port. To initial the control port for sending an OR %00000001 then an AND %11001111 is implemented.

CONTINUALLY DURING BYTE TRANSFERS
Bit 0 is set (1) to switch the handshake off to the peripheral while programming prepares a byte on the data port or is reading a byte from the data port then toggles to reset (0) a handshake on to the peripheral to say it is ready to receive a byte in input mode or has a byte ready to send in output mode. While doing so Bit 4 has to be to 0 to keep interrupts disabled and Bit 5 needs to be 0 for the send program or 1 for the receive program.

To keep all other Bits in the required condition to toggle Bit 0 the control port is read for an OR function %00000001for hand shake off or an AND %11111110 for handshake on before writing the byte back to the control port.

U_OUT_EN.BBC for the Einstein

```
 10 REM U_OUT_EN.BBC EINSTEIN BBCBASIC PROGRAM
 20 REM TEST PROGRAM TO OUTPUT DATA BYTES
 30 REM TO THE USER PORT USING INTERRUPT HANDSHAKING
 40 REM OUTPUT CAN BE RECEIVED BY A PC ECCP PRINTER PORT
 50 REM USING BYTE MODE SET FOR IMPORT.
 60 REM RECIEVED ASCII DATA BYTES CAN BE READ TO SCREEN BY
 70 REM XP_IN_WIN.BBC PROGRAM RUN BY BBCBASIC FOR WINDOWS.
 80 REM
 90 REM *| WIRING EINSTEIN USER PORT/PC PRINTER PORT
100 REM *|PC data lines D0 to D7 to Einstein D0 to D7
110 REM *|PC strobe to Einstein STB
120 REM *|PC acknowledge to Eistein RDY.
130 REM *| PLUS ONE OR MORE GROUND
140 REM *
150 ON ERROR GOTO 1030
160 HIMEM=HIMEM-80       :REM CREATES SAFE MEMORY AREA ABOVE BBCBASIC
180 DIM code 80
190 code=HIMEM+1
200 REM configpio =start_code+&11 :REM THESE VALUES ARE TO GIVE
210 REM outbyte =start_code+&3F   :REM REFERENCE FOR OTHER ASSEMBLERS
220 REM PAOUT =start_code+&28     :REM IF ASSEMBLE CODE IS ALTERED
230 REM CODE SIZE 64              :REM NEW VALUES WILL BE PRINTED OUT
240 REM BY BASIC LINES 800 to 850
250 PRINT "Assembling code at... &";~code
260 PRINT"PRESS ANY KEY TO CONTINUE ":H=GET
270 FOR pass=0 TO 1
280   P%=code
290   [OPT pass*3
300   .start_code
310   .flag DEFB 0       ;LATCH BYTE FOR INTERRUPT
320    .intrrpt          ;       INTERUPT ROUTINE
330    PUSH HL           ;
340    PUSH DE           ;ENDIF REGISTERS
350    PUSH BC           ;
360    PUSH AF           ;
370    LD HL,flag        ;GET ADDRESS OF flag BYTE INTO HL
380    RES 0,(HL)        ;SET BIT 0 OF flag TO 0
390    POP AF            ;         Note RST ops SHOULD NOT BE
400    POP BC            ;               USED IN AN INTERRUPT ROUTINE
410    POP DE            ;RESTORE REGISTERS
420    POP HL            ;
430    EI               ;ENABLE INTERRUPTS
440    RETI              ;RETURN FROM INTERRUPT
450    .configpio
460    LD  A,&12    ;sets low byte of interrupt address
470    OUT (&33),A  ;writes to USER port reg &12 %00010010
480    LD  A,&0F    ;sets PIO to mode 0 output
490    OUT (&33),A  ;writes to USER port reg &0F %00001111
500    LD  A,&87    ;enables interrupts
510    OUT (&33),A  ;writes to USER port reg &87 %10000111
520    PUSH HL
```

U_OUT_EN.BBC Einstein listing continued

```
530  LD HL,intrrpt  ;LOAD HL INTERRUPT START ADDRESS
540     LD (&FB12),HL  ;LOAD USER PORT SCRATCH PAD INTERRUPT ADDRESS
550        POP HL
560        IM 2          ; ENABLE INTERRUPT 2 MODE
570        RET
580        .PAOUT   ;CODE TO OUTPUT BYTE ENDWHILE IN MEMORY LOCATION outbyte TO
590       PUSH HL
600       PUSH DE          ;ENDIF REGISTERS
610       PUSH BC
620       PUSH AF
630       LD HL,flag    ; FLAG BYTE ADDRESS FOR USER PORT INTERRUPT.
640        .POLL
650        BIT 0,(HL)
660        JR NZ,POLL
670        LD A,(outbyte)
680        OUT (&32),A       ; outport Parallel port data reg.
690        SET 0,(HL)
700        POP AF
710        POP BC
720        POP DE          ;RESTORE REGISTERS
730        POP HL
740        RET
750        .outbyte DEFB 0      ;USER LOADS BYTE FOR OUTPUT
760             .end  RET
770             ]
780           NEXT pass
790           CALL configpio
800           PRINT "CODE BEGINS AT ADDRESS &";~start_code
810           PRINT "END CODE ADDRESS &";~end
820           PRINT "configpio =start_code+&";~configpio-start_code
830           PRINT "outbyte =start_code+&";~outbyte-start_code
840           PRINT "PAOUT =start_code+&";~PAOUT-start_code
850           PRINT "CODE SIZE "; end-start_code
860           PRINT "PRESS ANY KEY TO START"
870           H=GET
880           REM THE EXAMPLE CODE CODE BELOW CAN BE CHANGED
890           REM THE INSTALLED MACHINE CODE CAN BE USED TO
900           REM SEND ANY BYTE OUT TO THE USER PORT
910           REM BY LOADING outbyte ADDRESS WITH THE
920           REM BYTE TO BE SENT AND CALLING PAOUT
930           REPEAT
940             D$=INKEY$(0)
950             FOR I=32 TO 126
960               ?outbyte=I
970               CALL PAOUT
980               PRINT" ";I;
990             NEXT I
1000           UNTIL D$="A"
1010           PRINT"END END END END "
1020           STOP
1030           REM
1040           REPORT:PRINT "AT LINE ";ERL:STOP
```

XP_IN.BBC for the PC

Note. Line numbers are optional with BBC4W

```
10 *|        XP_IN.BBC
20 *|PC WINDOWS PROGRAM TO RUN IN BBCBASIC FOR WINDOWS
30 *|Windows BBC4W program XP_IN.BBC  works with
40 *|U_OUT_EN.BBC running in the Einstein
50 *|
60 *|It works in conjunction with the Einstein
70 *|with PC ECCP printer port linked with the user
80 *|port on the Einstein. It will input data bytes
90 *|from the program U_OUT_EN.BBC running on the Einstein.
100 *|
110 *|        WIRING EINSTEIN USER PORT
120 *|PC data lines D0 to D7 to Einstein D0 to D7
130 *|PC strobe to Einstein STB
140 *|PC acknowledge to Eistein RDY.
150 *|Plus one or more ground lines.
160 *|
170 *|
180 *|NEXT LINES CHECK TO SEE IF WINIO FILES ARE IN THE SAME DIRECTORY
190 SYS "GetFileAttributesA", "winio.vxd" TO ret%
200 IF ret%=-1 PRINT "ERROR winio.vxd not in directory "'@dir$:STOP ELSE PRINT"winio.vxd ok"
210 SYS "GetFileAttributesA", "winiov1.sys" TO ret%
220 IF ret%=-1 PRINT "ERROR winio1.sys not in directory "'@dir$:STOP ELSE PRINT"winiov1.sys ok"
230 SYS "GetFileAttributesA", "winiov1.dll" TO ret%
240 IF ret%=-1 PRINT "ERROR winio1.dll not in directory "'@dir$:STOP  ELSE PRINT"winiov1.dll ok"
250 *|NEXT TWO LINES TO SHUT DOWN WINIO ON PROGRAM ERRORS AND AT CLOSE
260 *|IF WINIO IS NOT CLOSED IT WILL NOT ALWAYS INSTALL AGAIN WITHOUT A REBOOT
270 ON CLOSE  SYS ShutdownWinIo% :PRINT:REPORT:STOP
280 ON ERROR  SYS ShutdownWinIo% :PRINT:REPORT:STOP
290 datap%=&378:             *|PRINTER LPT1 PORTS
300 status%=&378+1:          *|
310 controll%=&378+2:        *|
320 eccport%=&378+&402:      *|
330 DIM  dataport% 1:        *|MEMORY LOCATIONS TO STORE
340 DIM eccp% 1   :          *|PRINTER PORT READ AND WRITE BYTES
350 DIM statusp% 1 :         *|FOR WINIO SYSTEM CALLS
360 DIM controllp% 1:        *|SEE HELP Input/output using WINIO
370
380 *|          ******************************
390 *|       ************************************
400 *|    ******** LOADS THIRD PARTY WINIO LIBRARY *******
410 *|    FILES  WINIO.DLL, WINIO.SYS and WINIO.VXD must be in the
420 *|   same directory (folder) as the executable program using them.
430 *|Note to install WINIO IN 2000 and XP administrative privileges ARE NEEDED
440 *|*************************************************************************
450 SYS "LoadLibrary", "WINIO.DLL" TO winio%
460 IF winio% = 0 ERROR 0, "Could not load WINIO"
470 SYS "GetProcAddress", winio%, "InitializeWinIo" TO InitializeWinIo%
480 SYS "GetProcAddress", winio%, "ShutdownWinIo" TO ShutdownWinIo%
490 SYS "GetProcAddress", winio%, "GetPortVal" TO GetPortVal%
```

XP_IN.BBC PC listing continued

```
500 SYS "GetProcAddress", winio%, "SetPortVal" TO SetPortVal%
510 SYS InitializeWinIo% TO ok%
520 IF (ok% AND 1) = 0 ERROR 0, "Could not initialise WINIO"
530 *|      ********************************************************
540 *|
550 *|*************************************************************************
560 PROC_set_byte_mode:      *|SETS UP ECP PARALLEL FOR BYTE MODE
570 PRINT
580 PROC_controll_input:     *|SETS PRINTER PORT FOR INPUT
590 PRINT"RUN U_OUT_EN.BBC ON THE EINSTEIN THEN "
600 PRINT"PRESS ANY KEY TO RECEIVE INPUT FROM PRINTER PORT"
610 H=GET
620 PRINT""TO CANCEL PRESS ESCAPE ON THE EINSTEIN KEY BOARD FIST  "'\
630 \"THEN ESCAPE ON THE PC KEYBOARD. "
640 *|                  **********************
650 *|                   ********************
660 *|            ****************************************
670 *|       ************************************************************
 680 *|********** MAIN PROGRAM FOR EXPORTING Bytes TO THE PRINTER PORT
******************
690 *|
700 *|
710 pollbyte=0: *| var to latch an acknowledge pulse
720 REPEAT
730   key$=INKEY$(0)
740   PROC_get
750   data%=?^dataport%
760   IF data%=13 PRINT CHR$(data%)
770   IF data% =>32 AND data%=<126 PRINT CHR$(data%);
780 UNTIL key$="A" OR key$="a"
790 SYS ShutdownWinIo%
800 STOP
810 *| ********************** END OF TO RECEIVE DATA  ********************
820 *|                ***********************
830 *|           *********DEF PROC_get************
840 *|WHEN THE LPT1 PORT HAS BEEN INITIATED AND WINIO INSTALLED
850 *|PROC_get IS THE ONLY PROCEDURE NEEDED FOR THE USER TO IMPORT A BYTE
860 *|FROM THE pc PRINTER PORT data%=?^dataport% PUTS IMPORTED BYTE IN data% VAR
870 DEF PROC_get
880 REPEAT:PROC_poll_ack:UNTIL pollbyte<>0
890 PROC_strobe_low
900 SYS GetPortVal%, datap%, ^dataport%, 1
910 pollbyte=0
920 PROC_strobe_high
930 ENDPROC
940 *| ************************************************************************
950
960 *|   ************* PROC_set_byte_mode DOES WHAT IS SAYS ***************
970 DEF  PROC_set_byte_mode
980 SYS GetPortVal%, eccport%, ^eccp%, 1
```

XP_IN.BBC PC listing continued

```
 990 byte=?^eccp%
1000 byte=byte AND %00111111
1010 byte=byte OR  %00100000
1020 SYS SetPortVal%, eccport%, byte, 1
1030 SYS GetPortVal%, eccport%, ^eccp%, 1
1040 byte=?^eccp%
1050 D$=FN_BIN(byte)
1060 PRINT '"ENHANCED PARALLEL PORT"
1070 PRINT"      BIT PATTEN                    ";D$
1080 PRINT" Needed to set ECP port to byte mode.       %001XXXXX"
1090 IF (byte AND %11100000)=%00100000 PRINT " BYTE MODE SET " ELSE PRINT"BYTE MODE
NOT SET "
1100 ENDPROC
1110
1120 *|   ************  PROC_controll_input  ********************
1130 *|   ************  SETS LPT1 PRINTER PORT FOR INPUT
1140 *|   ************  DISABLES INTERRUPTS
1150 *|   ************  SETS STROBE LINE LOW (strobe hardware inverted)
1160 DEF PROC_controll_input
1170 SYS GetPortVal%, controll%, ^controllp%, 1
1180 ctrl=ctrl OR %00100001       :*| ; set bits 0 to 1 inverted 1 sets STROBE line to low.
1190 ctrl=ctrl AND %11101111      :*| ; bit 5 set to 1 for input
1200 :                            *| ; bit 4 set to 0 no interrupts set
1210 SYS SetPortVal%, controll%, ctrl, 1
1220 SYS GetPortVal%, controll%, ^controllp%, 1
1230 ctrl=?^controllp%
1240 D$=FN_BIN(ctrl)
1250 PRINT "CONTROLL PORT BIT PATTEN                  ";D$
1260 PRINT " Needed bit 5 set to 1 for data input     %XX1XXXXX"
1270 PRINT " To set bit 0 to 1. Sets STROBE line to low. %XXXXXXX1"
1280 PRINT " STROBE line hardware inverted"
1290 PRINT " Bit 4 set to 0 for no interrupts set.     %XXX0XXXX"
1300 IF (ctrl AND %00100000)=32 PRINT "         SET UP FOR IMPORT INSTALLED " ELSE PRINT
"ERROR"
1310 PRINT
1320 ENDPROC
1330
1340
1350 DEF PROC_poll_ack
1360 SYS GetPortVal%, status%, ^statusp%, 1
1370 stat=?^statusp%
1380 ack%=stat AND %01000000
1390 IF ack%<>0 pollbyte=1
1400 ENDPROC
1410
1420
1430 DEF PROC_strobe_high
1440 SYS GetPortVal%, controll%, ^controllp%, 1
1450 ctrl=?^controllp%
1460 ctrl=ctrl AND %11111110       :*| ; set bit 0 to 0. Inverted-0 sets STROBE line to high.
```

XP_IN.BBC PC listing continued

```
1470 SYS SetPortVal%, controll%, ctrl, 1
1480 ENDPROC
1490
1500
1510 DEF PROC_strobe_low
1520 SYS GetPortVal%, controll%, ^controllp%, 1
1530 ctrl=?^controllp%
1540 ctrl=ctrl OR %00000001        :*| ; set bit 0 to 1. Inverted-1 sets STROBE line to low.
1550 SYS SetPortVal%, controll%, ctrl, 1
1560 ENDPROC
1570
1580
1590 *| *************** FUNCTION PRINTS OUT BINARY STRING OF VAR *************
1600 *| E,G VAR NUMERIC VALUE 7 WOULD RETURN STRING VAR %00000111
1610 DEF FN_BIN(A%):LOCAL A$
1620 REPEAT A$=STR$(A% AND 1)+A$:A%=A% >>> 1:UNTIL LEN A$=8:A$="%"+A$
1630 =A$
1640 *| ************************************************************************
1650
1660 *| FINISH
```

U_IN_EN.BBC for the Einstein

```
 10 REM U_IN_EN.BBC EINSTEIN BBCBASIC PROGRAM
 20 REM TEST PROGRAM TO IMPORT DATA BYTES
 30 REM FROM THE USER PORT USING INTERRUPT HANDSHAKING
 40 REM IMPORT CAN BE RECEIVED FROM A PC ECCP PRINTER PORT
 50 REM RUNNING OUT_XP_large IN BBCBASIC FOR WINDOWS
 60 REM ECCP SET TO BYTE MODE AND FOR IMPORT.
 70 REM
 80 REM *| WIRING EINSTEIN USER PORT/PC PRINTER PORT
 90 REM *|PC data lines D0 to D7 to Einstein D0 to D7
100 REM *|PC strobe to Einstein STB
110 REM *|PC acknowledge to Eistein RDY.
120 REM *| PLUS ONE OR MORE GROUND
130 REM *
140 ON ERROR GOTO 1110
150 HIMEM=HIMEM-80
170 usrcon=&33
180 usrdata=&32
190 DIM code 80
200 code=HIMEM+1
210 PRINT "Assembling...code at &";~HIMEM+1
220 PRINT"PRESS ANY KEY TO CONTINUE ":H=GET
230 FOR pass=0 TO 1
240   P%=code
250   [OPT pass*3
260   .start_code
270   .flag DEFB 0
```

U_IN_EN.BBC Einstein listing continued

```
 280  .intrrpt    ;interupt routine
 290  PUSH HL
 300  PUSH DE     ;save registers
 310  PUSH BC
 320  PUSH AF
 330  LD HL,flag
 340  RES 1,(HL)  ;at an interrupt byte at address flag has bit 1 set to zero.
 350  POP AF
 360  POP BC      ;retrieve registers
 370  POP DE
 380  POP HL
 390  EI          ;enables interuppts
 400  RETI        ;returns from interrupt routine.
 420   .configpio         ;ROUTINE TO CONFIGER USER PORT FOR MODE 1 and INTERRUPS
ENABLED
 430   LD  A,&12       ;sets low byte of interrupt address
 440   OUT (usrcon),A  ;writes to user port CONTROL reg &12 %00001100
 450   ;sets low byte of interupt vector address
 460   ;high byte of interupt vector address in Z80 I reg
 470   ;set by Einstein at boot up to &FB.
 480   LD  A,&4F       ;sets PIO to mode 1
 490   OUT (usrcon),A  ;writes to Einstein user port reg &4F %01001111
 500   LD  A,&87       ;enables interrupts
 510   OUT (usrcon),A  ;writes to parallel user port control reg &87 %10000111
 520   PUSH HL
 530   LD HL,intrrpt   ; LOAD reg HL address of interupt routine.
 540   LD (&FB12),HL   ; LOAD interupt address into interupt vector at &FB12.
 550   POP HL
 560   IM 2            ; ENABLE INTERRUPT 2 MODE
 570   RET
 580   .getbyte        ;ROUTINE TO IMPORT A SINGLE BYTE FROM THE USER PORT
 590   PUSH HL
 600   PUSH DE         ;save registers on the stack
 610   PUSH BC
 620   PUSH AF
 630   LD HL,flag
 640   .POLL           ;LOOP to poll bit 1 of flag set by interrupt routine.
 650   LD HL,flag
 660   BIT 1,(HL)
 670   JR NZ,POLL      ;IF flag bit 1 not zero loop back.
 680   IN A,(usrdata)  ;IF flag bit 1 zero get byte from user port into "A" register.
 690   LD (imput_byte),A    ;Put byte value of imported byte now in "A" register
 700   ;into memory location "imput_byte"
 710   SET 1,(HL)      ;resets bit 1 of byte at memory location "flag" to 1.
 720   POP AF
 730   POP BC          ;retrieve registers from stack.
 740   POP DE
 750   POP HL
 760   RET
 770   .imput_byte DEFB 0
```

U_IN_EN.BBC Einstein listing continued

```
780    .end
790    ]
800  NEXT pass
810  CALL configpio
820  PRINT "CODE BEGINS AT ADDRESS &";~start_code
830  PRINT "END CODE ADDRESS &";~end
840  PRINT "flag=start_code+&";~flag-start_code
850  PRINT "configpio =start_code+&";~configpio-start_code
860  PRINT "getbyte =start_code+&";~getbyte-start_code
870  PRINT "imput_byte=start_code+&";~imput_byte-start_code
880  PRINT "CODE SIZE "; end-start_code
890  PRINT "PRESS ANY KEY TO START"
900  REM USER PROGRAMING CAN BE ADDED HERE
910  REM RECEIVED IMPORT BYTES FROM USER PORT
920  REM ARE FOUND BY CALLING getbyte THEN
930  REM PEEKING THE ADDRESS imput_byte
940  REM "n=?imput_byte" IN BBCBASIC
950  REM
960  REM IN THE EXAMPLE PROGRAM BELOW
970  REM BYTE VALUES 32 to 126 TO BE PRINTED
980  REM TO SCREEN AS ASCII CHARACTERS IN THE PC
990  REM WITH BBC4W RUNNING P_IN_WIN.BBC
1000  REM
1010  PRINT "PRESS ANY KEY TO RUN EXAMPLE PROGRAM"
1020  H=GET
1030  REPEAT
1040    D$=INKEY$(0)
1050    n=?imput_byte
1060    CALL getbyte
1070    IF n=13 PRINT
1080    IF n=>32 AND n=<126 PRINT CHR$(n);
1090  UNTIL D$="A" OR n=27
1100  STOP
```

XP_OUT.BBC for the PC

Note. Line numbers are optional with BBC4W

```
 10 *|XP_OUT.BBC is just small enough to run in
 20 *|the trial BBC4W demo version.
 30 *|If additional basic lines are added delete
 40 *|comments before running in demo version
 50 *|a NO ROOM error could mean rebooting
 60 *|before WINIO will install again.
 70 *|
 80 *|         XP_OUT.BBC
 90 *|PC WINDOWS PROGRAM TO RUN IN BBCBASIC FOR WINDOWS
100 *|Transfers bytes from the PC printer port to
110 *|Einstein's USER PORT.
120 *|Requires WINIO support files WINIO.VXD, WINIOV1.DLL
130 *|and WINIOV1SYS in the same folder as this program.
140 *|
```

XP_OUT.BBC PC listing continued

```
 150 *|Windows BBC4W program OUT_XP_large.BBC works with
 160 *|U_IN_EN.BBC running in the Einstein to send bytes
 170 *|from the PC printer port to the Einstein's user
 180 *|port.
 190 *|
 200 *|        WIRING EINSTEIN USER PORT
 210 *|PC data lines D0 to D7 to Einstein D0 to D7
 220 *|PC strobe to Einstein STB
 230 *|PC acknowledge to Eistein RDY.
 240 *|Plus one or more ground lines.
 250 *|
 260 *|
 270 *|This BBC4W Prog will work for win95/98 AND XP
 280 *|The program is intended to poll acknowledge for
 290 *|a pulse as output by RDY pin 5 of the
 300 *|Einstein's Z80 PIO user port.
 310 *|
 320 *|
 330 *|     WHEN Linked To The Einstein User Port
 340 *|All handshaking lines work as expected and data
 350 *|can be received by U_IN_EN.BBC when run on the Einstein.
 360
 370 datap%=&378:             *|PRINTER LPT1 PORTS
 380 status%=&378+1:          *|
 390 controll%=&378+2:        *|
 400 eccport%=&378+&402:      *|
 410 DIM  dataport% 1:        *|MEMORY LOCATIONS TO STORE
 420 DIM eccp% 1   :          *|PRINTER PORT READ AND WRITE BYTES
 430 DIM statusp% 1 :         *|FOR WINIO SYSTEM CALLS
 440 DIM controllp% 1:        *|SEE HELP INDEX "Input/output using WINIO"
 450 *|
 460 *|NEXT LINES CHECK TO SEE IF WINIO FILES ARE IN THE SAME DIRECTORY
 470 SYS "GetFileAttributesA", "winio.vxd" TO ret%
 480 IF ret%=-1 PRINT "ERROR winio.vxd not in directory "'@dir$:STOP ELSE PRINT"winio.vxd ok"
 490 SYS "GetFileAttributesA", "winiov1.sys" TO ret%
 500 IF ret%=-1 PRINT "ERROR winio1.sys not in directory "'@dir$:STOP ELSE PRINT"winiov1.sys ok"
 510 SYS "GetFileAttributesA", "winiov1.dll" TO ret%
 520 IF ret%=-1 PRINT "ERROR winio1.dll not in directory "'@dir$:STOP  ELSE PRINT"winiov1.dll ok"
 530 *|NEXT TWO LINES TO SHUT DOWN WINIO ON PROGRAM ERRORS AND AT CLOSE
  540 *|IF WINIO IS NOT CLOSED IT WILL NOT ALLWAYS INSTALL AGAIN WITHOUT A
REBOOT
 550 ON CLOSE  SYS ShutdownWinIo% :STOP
 560 ON ERROR  SYS ShutdownWinIo% :STOP
 570 *|          ******************************
 580 *|       ************************************
 590 *|    ******** LOADS THIRD PARTY WINIO LIBRARY *******
 600 *|    FILES  WINIO.DLL, WINIO.SYS and WINIO.VXD must be in the
 610 *|   same directory (folder) as the executable program using them.
 620 *|Note to install WINIO IN 2000 and XP administrative privileges ARE NEEDED
```

XP_OUT.BBC PC listing continued

```
 630 *|*************************************************************************
 640 SYS "LoadLibrary", "WINIO.DLL" TO winio%
 650 IF winio% = 0 ERROR 0, "Could not load WINIO"
 660 SYS "GetProcAddress", winio%, "InitializeWinIo" TO InitializeWinIo%
 670 SYS "GetProcAddress", winio%, "ShutdownWinIo" TO ShutdownWinIo%
 680 SYS "GetProcAddress", winio%, "GetPortVal" TO GetPortVal%
 690 SYS "GetProcAddress", winio%, "SetPortVal" TO SetPortVal%
 700 SYS InitializeWinIo% TO ok%
 710 IF (ok% AND 1) = 0 ERROR 0, "Could not initialise WINIO"
 720 *|*************************************************************************
 730 *|      ************************************************************
 740 *|
 750 PROC_set_byte_mode:      *|SETS UP ECCP PARALLEL FOR BYTE MODE
 760 PRINT
 770 PROC_controll_output:    *|SETS PRINTER PORT FOR OUT PUT
 780 *| ******************************************
 790 *|                    ********************
 800 *|            ***************************************
 810 *|       ********************************************************************
  820 *|********** MAIN PROGRAM FOR EXPORTING Bytes TO THE PRINTER PORT
******************
 830 *|
 840 *|
 850 pollbyte=0: REM var to latch an acknowledge pulse
 860 PRINT "      MAIN PROGRAM FOR EXPORTING BYTES TO THE PRINTER PORT "
 870 PRINT "PRESS ANY KEY TO CONTINUE MAIN PROGRAM "'
 880 H=GET
  890 PRINT "RUN U_IN_EN.BBC on the Einstein. """TO CANCEL PRESS ESCAPE ON THE
EINSTEIN KEY BOARD FIST  "'\
 900 \" THEN ESCAPE ON THE PC KEYBOARD. "
 910 *|REPEAT and FOR NEXT loop create an example program.
 920 REPEAT
 930   FOR i=32 TO 126
 940     D$=INKEY$(0)
 950     *|The next four lines get a single byte from
 960     *|the printer port and set handshaking.
 970     PROC_strobe_low
 980     REPEAT:PROC_poll_ack:UNTIL pollbyte=1
 990     SYS SetPortVal%, datap%, i, 1
1000      pollbyte=0
1010     PROC_strobe_high
1020   NEXT i
1030 UNTIL D$="A"
1040 SYS ShutdownWinIo%
1050 STOP
1060 *| ************************************************************************
1070
1080 *|
1090 *|   ************* PROC_set_byte_mode DOES WHAT IS SAYS **************
1100 DEF PROC_set_byte_mode
1110 SYS GetPortVal%, eccport%, ^eccp%, 1
```

XP_OUT.BBC PC listing continued

```
1120 byte=?^eccp%
1130 byte=byte AND %00111111
1140 byte=byte OR  %00100000
1150 SYS SetPortVal%, eccport%, byte, 1
1160 SYS GetPortVal%, eccport%, ^eccp%, 1
1170 byte=?^eccp%
1180 D$=FN_BIN(byte)
1190 PRINT "ENHANCED PARALLEL PORT"
1200 PRINT"      BIT PATTEN                    ";D$
1210 PRINT" Needed to set ECP port to byte mode.       %001XXXXX"
1220 IF (byte AND %11100000)=%00100000 PRINT " BYTE MODE SET " ELSE PRINT"BYTE MODE
NOT SET "
1230 ENDPROC
1240
1250 *|   ************  PROC_controll_output  ********************
1260 *|   ************  SETS LPT1 PRINTER PORT FOR OUTPUT
1270 *|   ************  DISABLES INTERRUPTS
1280 *|   ************  SETS STROBE LINE LOW (strobe hardware inverted)
1290 DEF PROC_controll_output
1300 SYS GetPortVal%, controll%, ^controllp%, 1
1310 REM PROC_ctrl
1320 ctrl=ctrl OR %00000001       :*| ; set bits 0 to 1 inverted 1 sets STROBE line to low.
1330 ctrl=ctrl AND %11001111      :*| ; bit 5 set to 0 for output
1340 :                            *| ; bit 4 set to 0 no interrupts set
1350 SYS SetPortVal%, controll%, ctrl, 1
1360 SYS GetPortVal%, controll%, ^controllp%, 1
1370 ctrl=?^controllp%
1380 D$=FN_BIN(ctrl)
1390 PRINT "CONTROLL PORT BIT PATTEN                 ";D$
1400 PRINT " Needed bit 5 set to 0 for data outport     %XX0XXXXX"
1410 PRINT " Needed bit 4 set to 0 for NO interrupts    %XXX0XXXX"
1420 PRINT " To set bit 0 to 1. Sets STROBE line to low. %XXXXXXX1"
1430 PRINT " STROBE line hardware inverted"
1440 ENDPROC
1450
1460 DEF PROC_poll_ack
1470 SYS GetPortVal%, status%, ^statusp%, 1
1480 stat=?^statusp%
1490 ack%=stat AND %01000000
1500 IF ack%<>0 pollbyte=1
1510 ENDPROC
1520
1530
1540 DEF PROC_strobe_high
1550 SYS GetPortVal%, controll%, ^controllp%, 1
1560 ctrl=?^controllp%
1570 ctrl=ctrl AND %11111110       :*| ; set bit 0 to 0. Inverted-0 sets STROBE line to high.
1580 SYS SetPortVal%, controll%, ctrl, 1
1590 ENDPROC
1600
```

XP_OUT.BBC PC listing continued

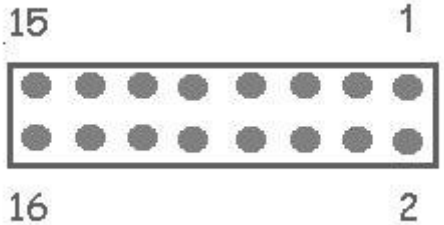
```
1610 DEF PROC_strobe_low
1620 SYS GetPortVal%, controll%, ^controllp%, 1
1630 ctrl=?^controllp%
1640 ctrl=ctrl OR %00000001        :*| ; set bit 0 to 1. Inverted-1 sets STROBE line to low.
1650 SYS SetPortVal%, controll%, ctrl, 1
1660 ENDPROC
1670
1680
1690 *| ************** FUNCTION PRINTS OUT BINARY STRING OF VAR *************
1700 *| E,G VAR NUMERIC VALUE 7 WOULD RETURN STRING VAR %00000111
1710 DEF FN_BIN(A%):LOCAL A$
1720 REPEAT A$=STR$(A% AND 1)+A$:A%=A% >>> 1:UNTIL LEN A$=8:A$="%"+A$
1730 =A$
1740 *| ************************************************************************
1750
1760 *|FINISH
```

WIRING LINK

USER PORT PIN			PC PRINTER PORT PIN	
2	D0	to	2	D0
4	D1	to	3	D1
5	RDY	to	10	ACK
6	D2	to	4	D2
8	D3	to	5	D3
10	D4	to	6	D4
11	STB	to	1	STB
12	D5	to	7	D5
14	D6	to	8	D6
16	D7	to	9	D7
One or more pins			pins	
3 7 9 or 13 GROUND		to	18-25	GROUND

USER PORT PIN OUT

1	5V
2	D0
3	0V
4	D1
5	RDY
6	D2
7	0V
8	D3
9	0V
10	D4
11	STB
12	D5
13	0V
14	D6
15	5V
16	D7



EINSTEIN USER PORT

PC Printer Port

PC PRINTER PORT PIN OUT

1	STB
2	D0
3	D1
4	D2
5	D3
6	D4
7	D5
8	D6
9	D7
10	ACK
11	BSY
12	PAPER
13	SEL
14	AF
15	ERROR
16	INI
17	DSL
18-25	GROUND

END