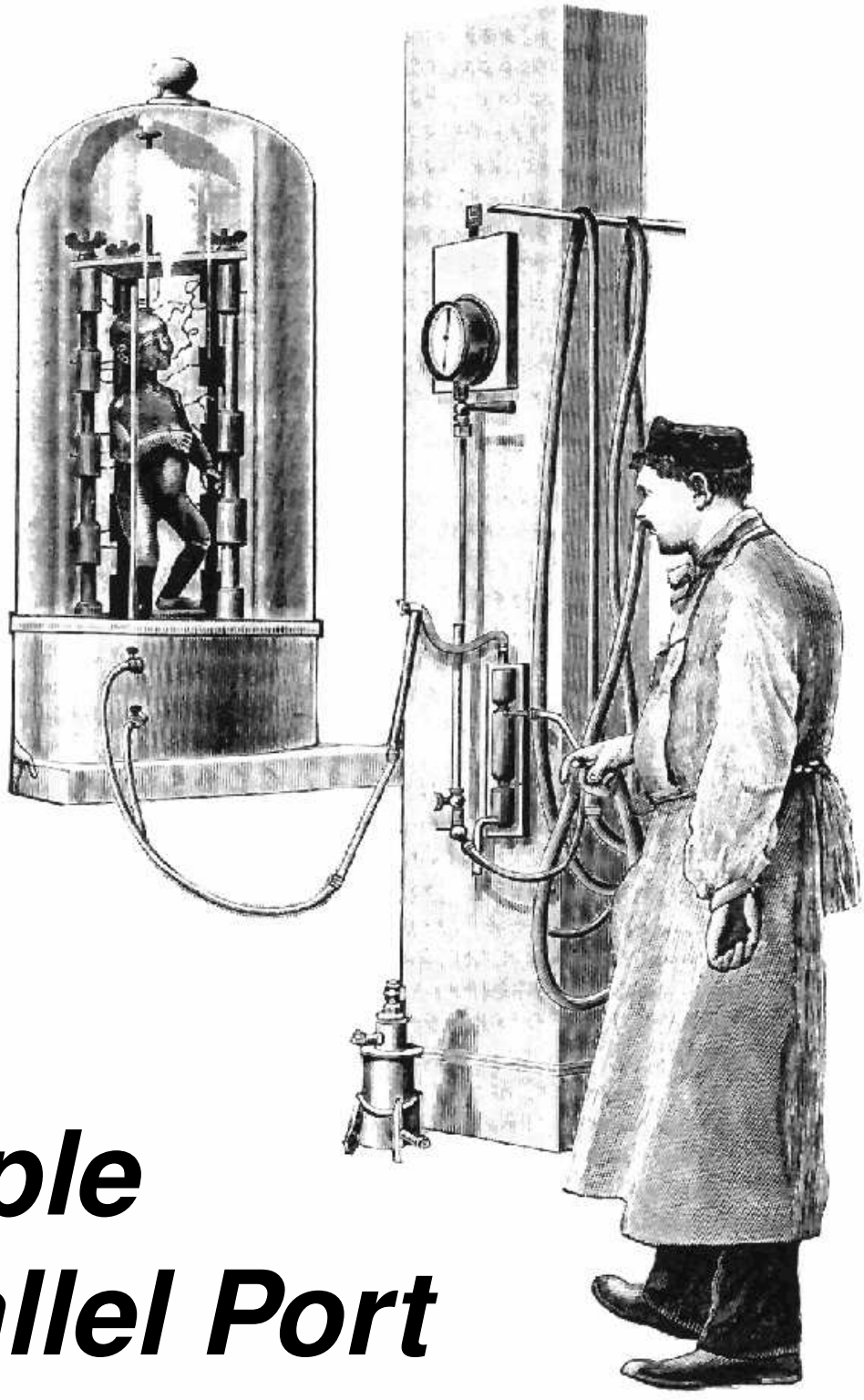
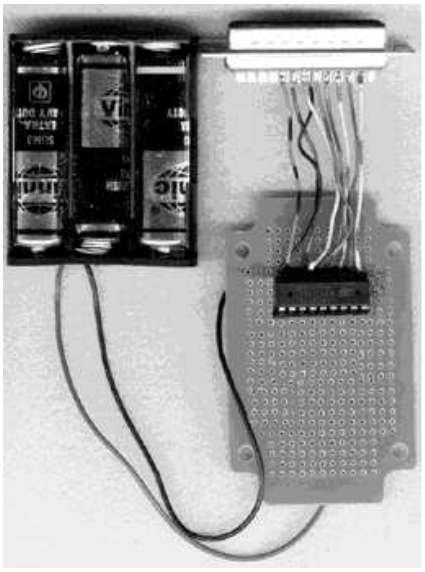


Simple Parallel Port Interfacing

by xxxtoytech

Simple Parallel Port Interface, 8 output

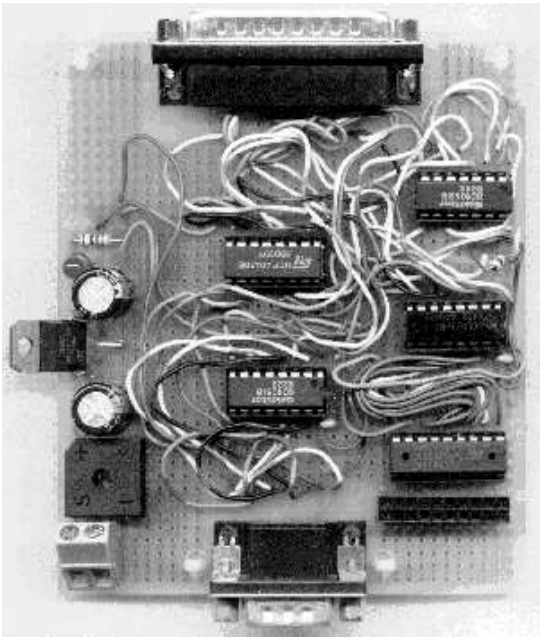


The first project took about 30 minutes to build on a piece of perf board and cost less than $5; see photo at right (shown with batteries). It is from an article by James M. Conrad and Jonathan W. Mills in Circut Cellar #108, July 1999 *"A PC-Based Controller for the Stiquito Robot"*.That article suggests you try Radio Shack perf board (Cat. # 276-150) The ULN2803 is a darlington transistor array with built-in resistors for 5 volt CMOS operation. To turn on an output transistor write a 1 to that bit (0 to 7). To get an idea of what is going on, hook up an LED (Light Emitting Diode) with a 220 ohm resistor between the output of the ULN2803 and a 4.5volt supply (3 AA batteries in series). This circuit is shown as the LED Output Tester on page 4. When transistor is on, the LED will glow. By adding opto-isolators (devices that electrically isolate parts of a circuit by using an LED and photodetector such as a transistor or triac) you can power devices of different voltages.

Use the SIMTEST.BAS program to test this 8 output parallel port interface. The program is designed to run in the MS-DOS QBASIC interpreter. Program steps through all control codes from 0 to 255 (1 byte or 8 bits).

Parallel Port Interface, 8 inputs 8 outputs

Construction of the board took an afternoon, including checking the board. In the photo below right is the finished board with the DB-25 connector at the top and the power supply along the left side. The connector at the bottom center is for connecting the inputs, power supply connection is at the bottom left. For testing I used an old 80386 computer and the QBASIC interpreter that comes with MS-DOS. Using an old computer makes life a bit easier since you're not as concerned with damaging the motherboard etc. The MS-DOS OS gives you access at a low level to the ports, the QBASIC language, although somewhat ugly, is simple to use for testing. Connect to the computer printer port with a standard DB-25 male/female cable from your computer store (I paid about $5 for a 6 foot cable).



The outputs/outputs are set or read independent of each other (unlike the simple interface above). To use this interface Data Bits 0 to 2 are set with the address of the input/output you want to read/set Data Bit 3 selects input or output (1 for output, 0 for input). Data Bit 4 is only used for the outputs and sets the output (0 is off, 1 is on). You then use the Printer Control Register to send an nStrobe signal to the interface to either set the output or read an input. In the case of an input, the value (either 0 or 1) is stored in a latch and can be read by reading the PError (Bit 5)in the Printer Status Register. Reading is simple, just AND the value of the status register with 20 hex (32 decimal). If 0 then the input read is OFF, otherwise the input read is ON. See the TEST.BAS program on page 6-7 and the parallel port info on page 3 to get a better grasp of how this works.

Wire the switches to the inputs of the CD4051 so that when the switch is closed it brings the output to ground. You can also interface other circuits to the inputs by using a general purpose NPN transistor (i.e. 2N3904) to ground the input. To approximate analog inputs, sample the switches over time.

Voltage on the outputs can be up to 500mA at 50 volts, however, it's a good idea to use opto-isolators along with the interface board's built-in 5 volt power supply, to isolate the board from high voltages and noise. When using opto-isolators with the board's 5 volt supply use a resistor to limit the current to isolator's ILED (about 270 ohms). You can control whatever you want with the outputs depending upon what type of opto-isolator you use. You can use the LED Output Tester circuit shown on page 4 along with the TEST.BAS program on pages 6-7 for testing the outputs. Use switches between ground and the inputs on CD4051 for testing the inputs.

While it is far from state of the art, this interface is cheap with available components and simple to build (even for the beginner), easy to use and trouble-shoot and versatile.

Using the Parallel Port to send data to the control unit.

To send data to the device, the data byte is first placed in the printer data register for the parallel port that the device is connected to. This is at the base address of the parallel port. The computer pulses strobe line to inform the device that the data is available on pins 2-9. After being read, the ack(nowledge) line (pin 10 on D shell connector) goes low (negative pulse). This pulse is about 10 microseconds in length. The three registers we are concerned with are:

Printer Data Register - holds byte going to computer, bits correspond to the pin 2-9 on the 25 pin D-shell connector. Pin 2 corresponds to the least significant bit (LSB), pin 9 corresponds to the most significant bit (MSB). The printer data register is located at the base address of the port to which the device is connected.

Printer Status Register - holds the return values from the printer. These correspond to the signals on the various pins (see information on pin signal assignment that follows). The printer status register is offset by one port from the base port (base port address + 1). The byte assignments are below.

not used	Bit 0 (LSB)
not used	Bit 1
not used	Bit 2
nFault	Bit 3
Select	Bit 4
PError	Bit 5
nAck	Bit 6
Busy	Bit 7 (MSB)

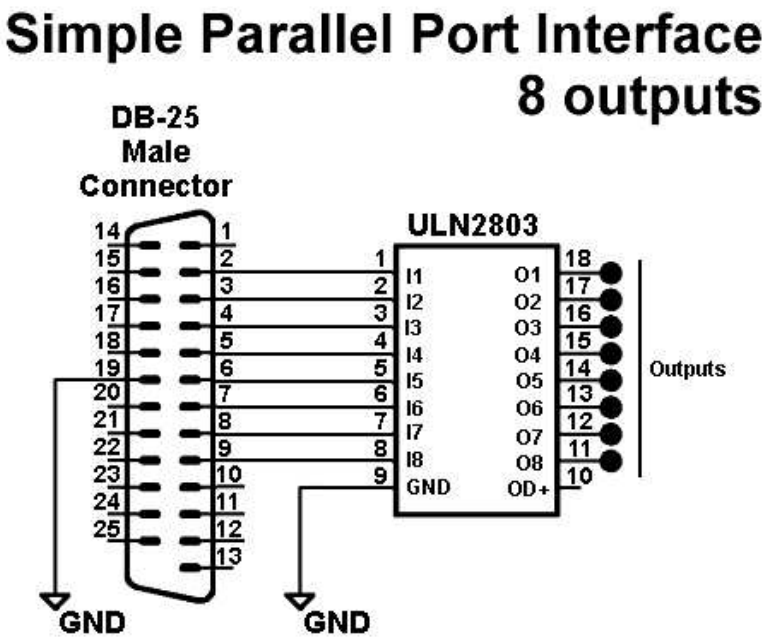
Printer Control Register - sends commands to the printer. This register is offset by two ports from the base address (base port address + 2). Although the programmer can handle these directly, for slow devices at low rates of data transfer it is easier to let low level BIOS or DOS functions handle the commands. For those who are interested the commands follow.

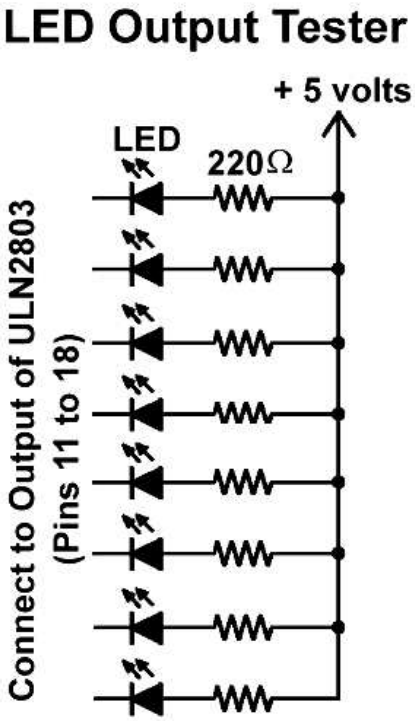
nStrobe	Bit 0 (LSB)
nAutoFd	Bit 1
nInit	Bit 2
nSelectIn	Bit 3
interrupt	Bit 4 if set high causes nAck signal to trigger interrupt 017h
not used	Bit 5
not used	Bit 6
not used	Bit 7 (MSB)

Use IBM compatibility mode (default for parallel port) 25-pin D-Shell connector (female on computer, male on device)

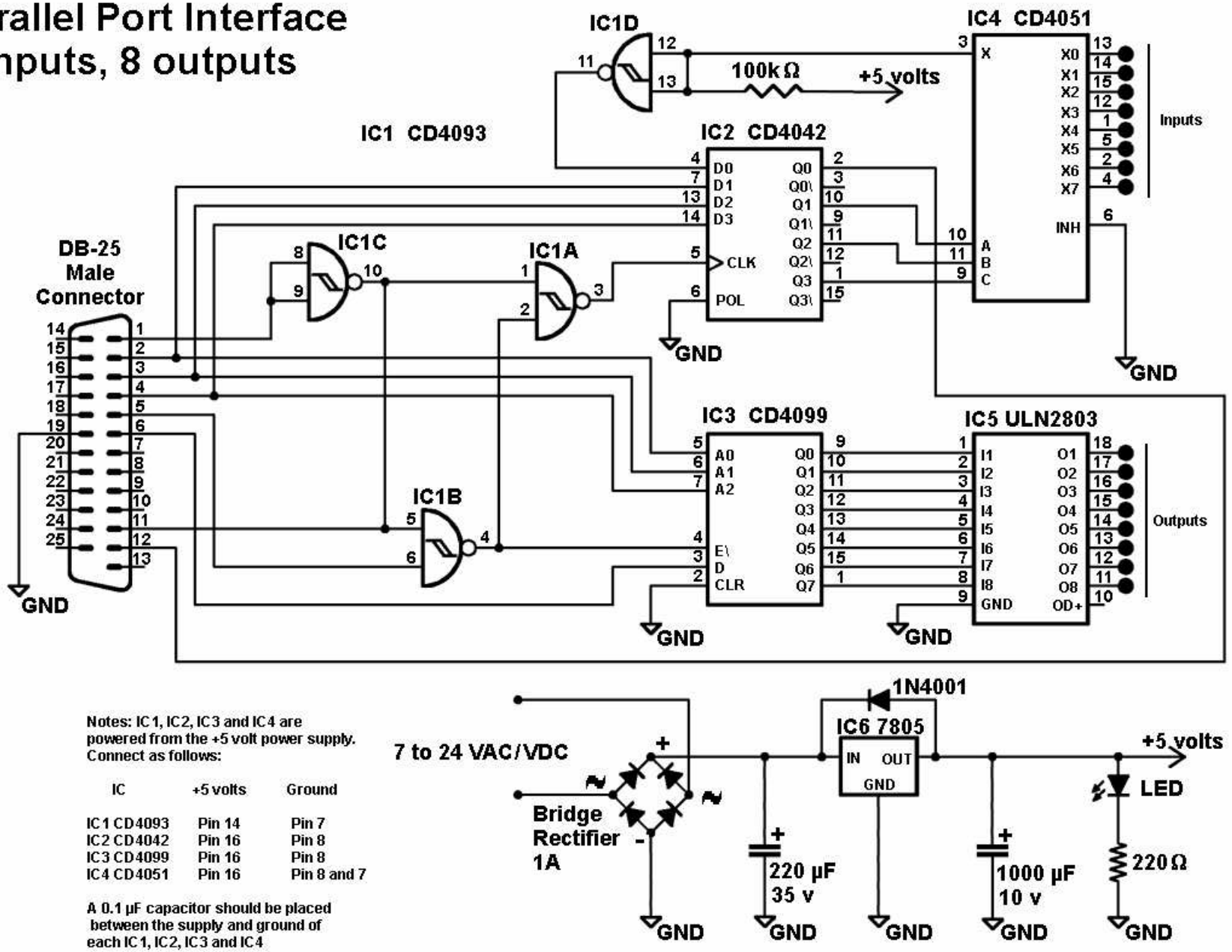
Pin	Signal	Comments
1	nStrobe	from computer, negative pulse tells device data is ready
2	Data 0	data from computer (least significant bit)
3	Data 1	data from computer
4	Data 2	data from computer
5	Data 3	data from computer
6	Data 4	data from computer
7	Data 5	data from computer
8	Data 6	data from computer
9	Data 7	data from computer (most significant bit)
10	nAck	acknowledge, device sends negative pulse to computer
11	Busy	device sets this high to tell computer it is busy
12	PError	from device to computer
13	Select	from device to computer
14	nAutoFd	from computer to device
15	nFault	from device to computer
16	nInit	from computer to device, negative pulse resets device
17	nSelectIn	from computer to device
18-25	Ground	signal ground

```
REM - SIMTEST.BAS - This program is a simple test program for an
REM - 8 output parallel port interface designed to run in the
REM - MS-DOS QBASIC interpreter. Program steps through all control
REM - codes from 0 to 255 (1 byte or 8 bits)

DIM pport%(3)
pport%(0) = &H278
pport%(1) = &H378
pport%(2) = &H3BC
cport% = pport%(0)
togg% = 0
CLS
PRINT : PRINT
PRINT "Simple Parallel Port InterfaceTest"
PRINT "Press ESC to exit program"
PRINT "Press P to change Port"
PRINT "Press T to sTep through outputs"
OUT cport%, togg%
PRINT "All outputs are OFF"
PRINT "Current port is "; HEX$(cport%)

REM - main loop of program -

DO
ks$ = INKEY$
SELECT CASE ks$
CASE CHR$(27)
        EXIT DO
CASE "P", "p"
        PRINT "Change Port"
        PRINT "Enter 2 for port 0278h"
        PRINT "Press 3 for port 0378h (default port)"
        PRINT "Press B for port 03BCh"
        INPUT PortSelect$
        SELECT CASE PortSelect$:
        CASE "2"
                cport% = pport%(0)
        CASE "3"
                cport% = pport%(1)
        CASE "B": CASE "b"
                cport% = pport%(2)
        CASE ELSE
                PRINT "Not a valid choice, Press P to try again"
        END SELECT
        PRINT "Port Selected", "0"; HEX$(cport%); "h"
        togg% = 0
CASE "T", "t"
        REM - The following OUT writes a data byte to Printer Data Register
        OUT cport%, togg%
        PRINT "Output "; togg%
        togg% = togg% + 1
        IF togg% = 256 THEN togg% = 0
END SELECT
LOOP
REM - End of program
```





Parallel Port Interface
8 inputs, 8 outputs



Notes: IC1, IC2, IC3 and IC4 are powered from the +5 volt power supply. Connect as follows:

IC	+5 volts	Ground
IC1 CD4093	Pin 14	Pin 7
IC2 CD4042	Pin 16	Pin 8
IC3 CD4099	Pin 16	Pin 8
IC4 CD4051	Pin 16	Pin 8 and 7

A 0.1 µF capacitor should be placed between the supply and ground of each IC1, IC2, IC3 and IC4

```
REM - TEST.BAS - This program is a simple test program for an 8 input, 8 output
REM  - parallel port device designed to run in the MS-DOS QBASIC interpreter
DIM pport%(3)
pport%(0) = &H278
pport%(1) = &H378
pport%(2) = &H3BC
cport% = pport%(0)
togg% = 8

REM - Turn off all outputs
FOR i = 0 TO 7 STEP 1
OUT cport%, togg%
OUT (cport% + 2), 1
togg% = togg% + 1
NEXT i
togg% = 24

CLS : PRINT : PRINT
PRINT "Simple Parallel Port Interface Device Test"
PRINT "----------------------------------------------------"
PRINT "Press ESC to exit program"
PRINT "Press H for Help"
PRINT "Press P to change Port"
PRINT "Press S to Scan inputs"
PRINT "Press T to Toggle outputs on and off"
PRINT : PRINT "All outputs are OFF"
PRINT "Current port is "; HEX$(cport%)

DO
ks$ = INKEY$
SELECT CASE ks$
CASE CHR$(27)
        EXIT DO
CASE "H", "h"
        PRINT "Press ESC to exit program"
        PRINT "Press H for Help"
        PRINT "Press P to change Port"
        PRINT "Press S to Scan inputs"
        PRINT "Press T to Toggle outputs on and off"
CASE "P", "p"
        PRINT "Change Port"
        PRINT "Enter 2 for port 0278h"
        PRINT "Press 3 for port 0378h (default port)"
        PRINT "Press B for port 03BCh"
        INPUT PortSelect$
        SELECT CASE PortSelect$
        CASE "2"
                cport% = pport%(0)
        CASE "3"
                cport% = pport%(1)
        CASE "B": CASE "b"
                cport% = pport%(2)
        CASE ELSE
                PRINT "Not a valid choice, Press P to try again"
        END SELECT
        PRINT "Port Selected", "0"; HEX$(cport%); "h"
        togg% = 8
```

```
CASE "S", "s"
        PRINT "Scan Inputs"
        PRINT "————————————"
        FOR i = 0 TO 7 STEP 1
        OUT cport%, i
        OUT (cport% + 2), 1
        x% = INP(cport% + 1) AND &H20
        IF x% > 0 THEN PRINT "HIGH #"; (i + 1) ELSE PRINT "LOW #"; (i + 1)
        NEXT i
CASE "T", "t"
        IF togg% > 23 THEN PRINT "Output "; (togg% - 23); " ON" ELSE PRINT ; "Output "; (togg% - 7); "OFF"
        REM - The following OUT writes a data byte to Printer Data Register
        OUT cport%, togg%
        REM - The  following OUT writes a control byte to the
        REM - Printer Control Register which sends an nStrobe signal
        REM - to Pin 1 of the DB-25 connector
        OUT (cport% + 2), 1
        togg% = togg% + 1
        IF togg% = 16 THEN togg% = 24
        IF togg% = 32 THEN togg% = 8
END SELECT
LOOP
REM - End of program
```

Codes for Addressing the 8 input/output interface

Codes for each input/output numbered 0 through 7 (X0-X7 on the CD4051 inputs, O1-O8 outputs on the ULN2803) first in decimal numbering followed by the hex code in brackets. The read input code places the value of the input (0 or 1) in a latch. To actually get the value look at the PError value (Bit 5) of the Printer Status Register.

I/O	Set Output LOW	Set Output HIGH	Read Input
0	8 (08h)	24 (18h)	0 (00h)
1	9 (09h)	25 (19h)	1 (01h)
2	10 (0Ah)	26 (1Ah)	2 (02h)
3	11 (0Bh)	27 (1Bh)	3 (03h)
4	12 (0Ch)	28 (1Ch)	4 (04h)
5	13 (0Dh)	29 (1Dh)	5 (05h)
6	14 (0Eh)	30 (1Eh)	6 (06h)
7	15 (0Fh)	31 (1Fh)	7 (07h)