DEVICE CONTROL THROUGH PC'S PARALLEL PORT USING VISUAL BASIC

■ **ADEEB RAZA**

Here is a Windows-based program developed in Microsoft Visual Basic programming language for controlling eight devices through the PC's parallel port or Line Printer Port (LPT). The program accepts the input in decimal number and outputs in binary form across the data pins of the PC's parallel port for controlling the connected devices/appliances.

PC's parallel port

The standard parallel port comprises four control lines, five status lines and eight data lines (refer to the table). It is found on the back of the PC as a D-type 25-pin female connector. Here, we are concerned only with data lines D0 through D7 terminated at pins 2 through 9. These data lines are the primary means of sending information out of the port. Pins 18 through 25 of the connector are grounded.

Control lines of the parallel port are used to provide control signals such as 'form feed' and 'initialise' to the printer.

The five status lines are the only input lines of the standard parallel port. These allow the printer to send signals such as 'error,' 'paper out' and 'busy' to the PC.

Circuit description

Fig. 1 shows the block diagram for device control through the PC's parallel port using Visual Basic. The data output port of the PC's parallel port is used for controlling the devices or appliances. The interface circuit requires regulated 6V DC to drive the loads. Eight MCT2E opto-osolator ICs are used to prevent damage to the parallel port from short-circuit that may occur across the interface circuit. Darlington array IC ULN2803 is used to drive the relays for controlling the devices.

Fig. 2 shows the circuit for device control using the PC's parallel port programmed in Visual Basic. To get the power supply for the circuit, 230V AC mains is stepped down by transformer X1, rectified by bridge rectifier R3151 and filtered by capacitor C1 (1000µF, 25V). The filtered output is fed to input pin 1 of regulator IC 7806. The regulated 6V DC is used to power the interface circuit comprising ICs MCT2E (IC2 through IC9) and ULN2803 (IC1). Optocoupler MCT2E can be replaced with 4N35.

LED1 through LED8 connected across data output pins 2 through 9, respectively, are used to indicate the

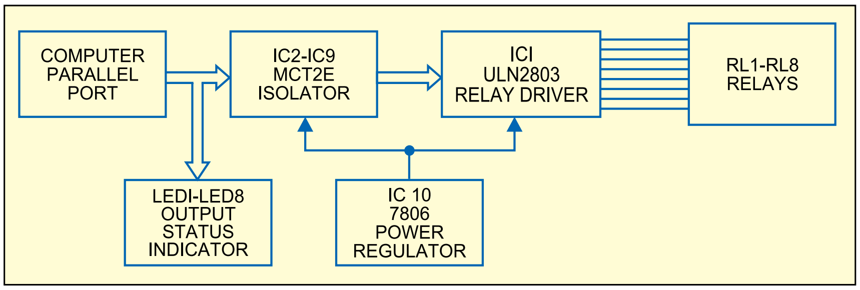


Fig. 1: Block diagram of device control through PC's parallel port using Visual Basic

Parallel-Port Pin Details

Pin number	Traditional use	Port name	Read/Write	Port address	Port bit
2-4	Data out	Data port	W	Base	D0-D2
5-9	Data out	—	W	Base	D3-D7
1	Strobe	Control port	R/W	Base+2	C0
14	Auto feed	—	R/W	Base+2	C1
16	Initialise	—	R/W	Base+2	C2
17	Select input	—	R/W	Base+2	C3
15	Error	Status port	R	Base+1	S3
13	Select	—	R	Base+1	S4
12	Paper end	—	R	Base+1	S5
10	ACK	—	R	Base+1	S6
11	Busy	—	R	Base+1	S7

PARTS LIST

Semiconductors:	
IC1	- ULN2803 relay driver
IC2-IC9	- MCT2E optocoupler
IC10	- 7806 voltage regulator
BR1	- 1A bridge rectifier
Resistors (all ¼-watt, ±5% carbon):	
R1-R16	- 220-ohm resistor
Capacitors:	
C1	- 1000µF, 25V electrolytic capacitor
C2	- 0.1µF ceramic type capacitor
Miscellaneous:	
X1	- 230V AC primary to 0-9V, 250mA secondary transformer
S1	- On/Off switch
RL1-RL8	- 6V, 100-ohm, 1C/O relay
	- 25-pin, D-type parallel-port male connector

status of the loads. Glowing of any of these LEDs indicates that the device connected to that specific output line is 'on.'

IC ULN2803 (Fig. 3) is a Darlington array relay driver that can drive eight relays. Since IC ULN2803 has an internal freewheeling diode to quench the inductive kick, no external freewheeling diodes are required across the relay coils. The devices are connected through the relay contacts to mains.

The relays are used to switch on or off the appliances.

Software program

Before going into details of the program, let us figure out some limitations of Visual Basic programming for interfacing the circuit. Visual Basic cannot directly access the computer hardware to control the external world. All the hardware requests must go through the supported file format of Windows operating system.

So the best way to manipulate the parallel port is the printer object. The printer object allows text and graphics to be printed on the printer through the parallel port of the PC. While all is well with this option, it is useless when you want a direct control of the hardware. In order to control the port directly, we must use something external to our program. A dynamic link library (DLL) file called 'WIN95IO.DLL' is used for that purpose.

The WIN95IO.DLL file is meant for a 32-bit machine, supported by Visual Basic Versions 4, 5 and 6. No matter which version you are using, the DLL file must be in the Windows\system directory of your machine. The interface control software program can be developed

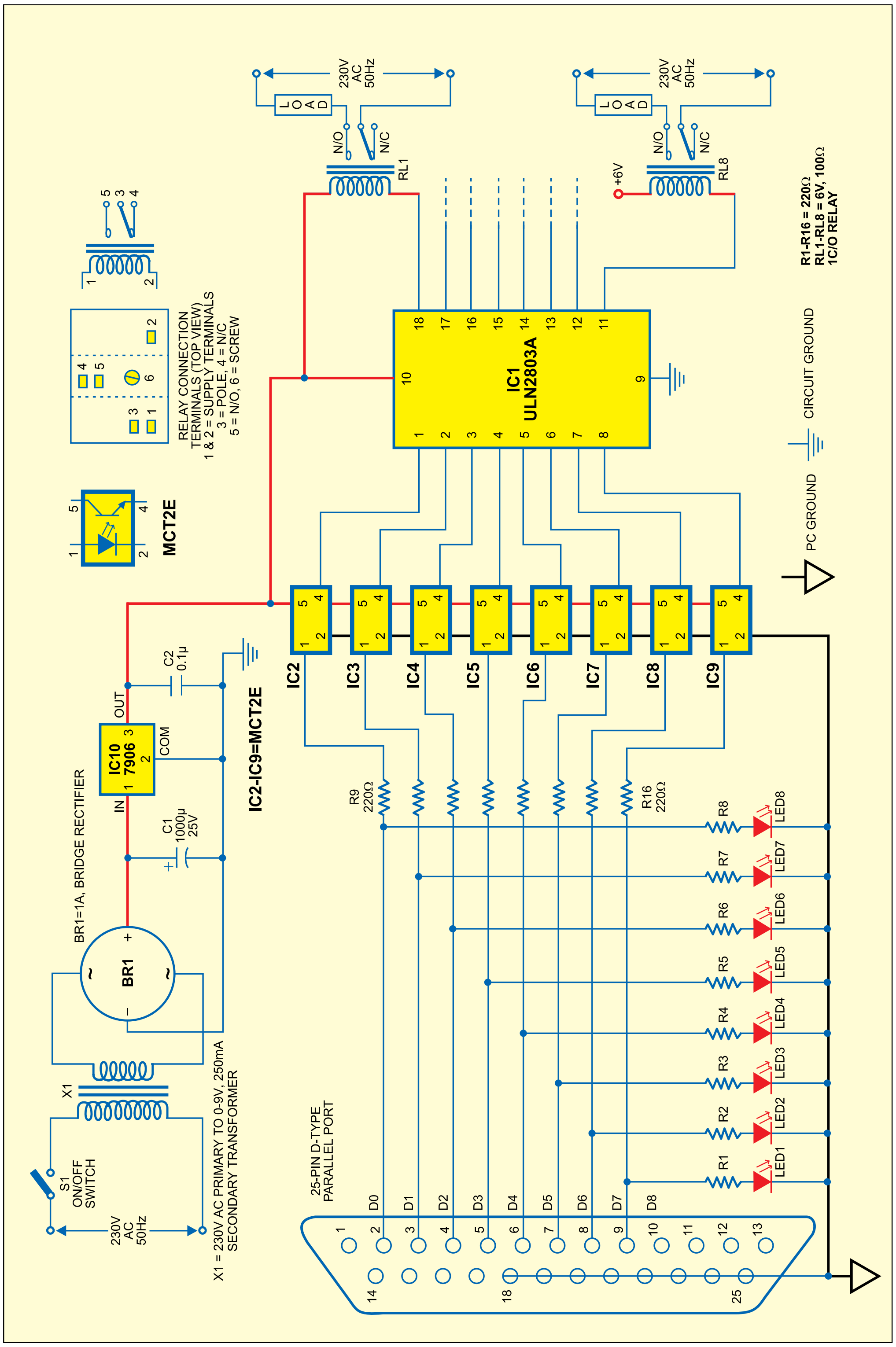


Fig. 2: Circuit for device control through PC's parallel port using Visual Basic

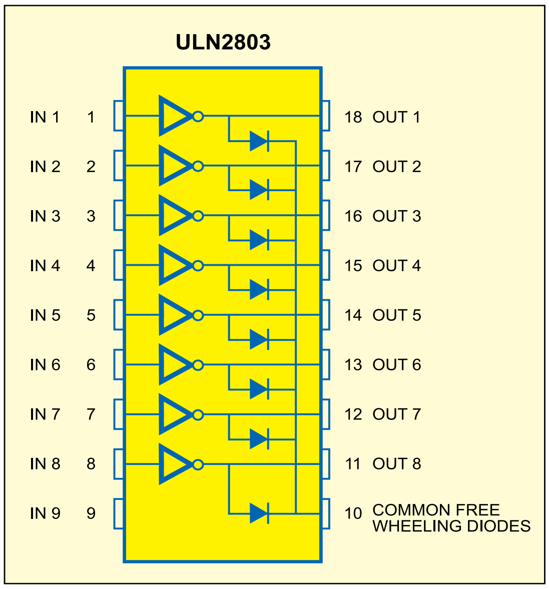


Fig. 3: Pin details of ULN2803

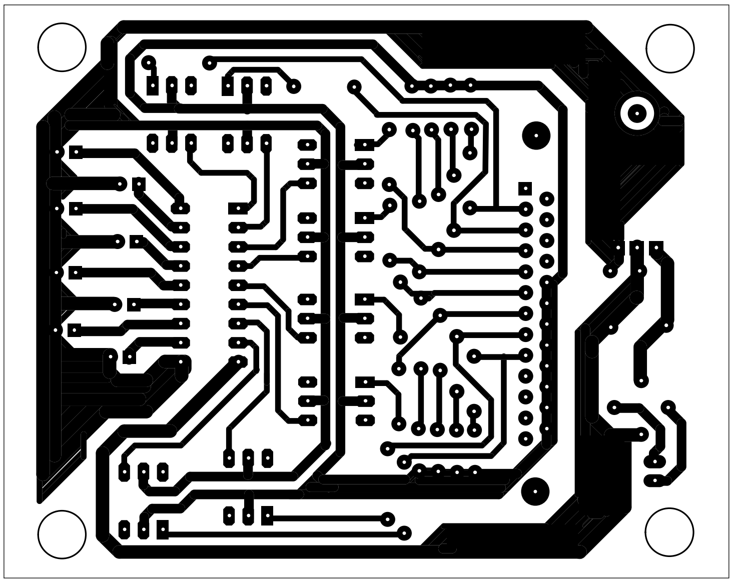
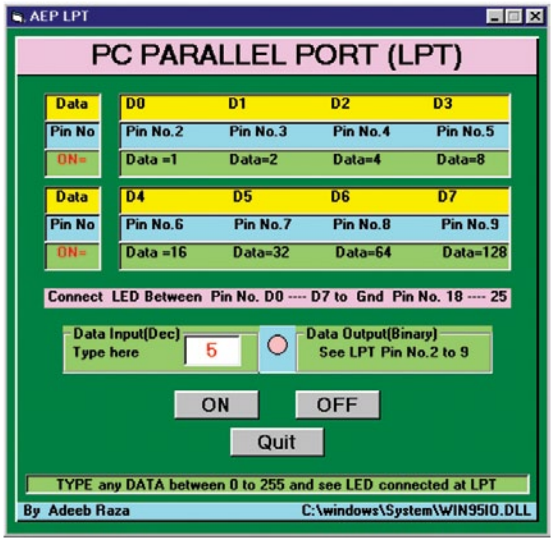


Fig. 5: Actual-size, single-side PCB layout for device control through PC's parallel port using Visual Basic


AEP LPT

PC PARALLEL PORT (LPT)

Data	D0	D1	D2	D3
Pin No	Pin No.2	Pin No.3	Pin No.4	Pin No.5
ON=	Data =1	Data=2	Data=4	Data=8

Data	D4	D5	D6	D7
Pin No	Pin No.6	Pin No.7	Pin No.8	Pin No.9
ON=	Data =16	Data=32	Data=64	Data=128

Connect LED Between Pin No. D0 ---- D7 to Gnd Pin No. 18 ---- 25

Data Input(Dec) Type here 5

Data Output(Binary) See LPT Pin No.2 to 9

TYPE any DATA between 0 to 255 and see LED connected at LPT

By Adeeb Raza C:\windows\System\WIN95IO.DLL

Fig. 4: Screen that appears when program is run

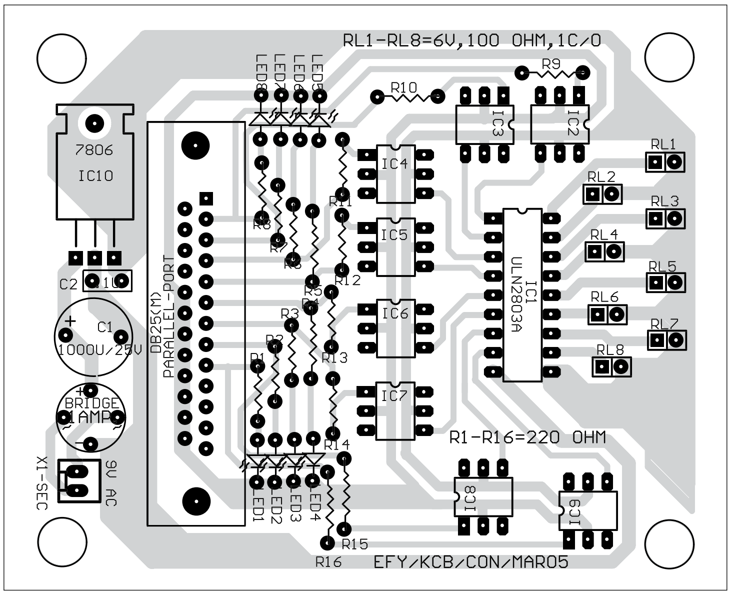


Fig. 6: Component layout for the PCB

thereon. No matter which DLL you use, it won't work under Windows NT due to security reasons.

The program code is given at the end of this article. It is assumed here that Microsoft Visual Basic 6 is installed on your PC and you have the basic programming knowledge.

The program coding is simple and you can write it yourself. Launch Visual Basic from the desktop and open a new project by selecting the 'Standard EXE' option. By default, it will open an empty project window on the screen with 'Form 1' as the file name. The form is one of the supported files of the Visual Basic. Pick the required components as shown in the screenshot (Fig. 4) from the toolbox on the left-hand side of the screen. The properties of each component can be set from the right-hand side of the screen.

The coding starts by declaring 'WIN95IO.DLL' in the first line "Private Declare Sub vbOut Lib 'WIN95IO.DLL' (ByVal AEPPort As Integer, ByVal AEPData as Integer)." The computer port is defined as 'AEPPort.' Its base address is assigned as 378 (in hex) by the program line "AEPPort=&H378." The 'vbOut' statement is used to send a bit to a port, for example, 'vbOut [port],[number]'

When you are done with coding, compile and run the program. You'll get the screen as shown in Fig. 4. Save the project file with '.vbp' extension. Make the executable file from 'File' menu.

EFY note. Form 1 is named as 'Arport' and Project 1 file as 'Arport.vbp.'

Construction

Construct the circuit for device control on any general-purpose PCB. Use eight flexible wires for data bus (D0 through D7) by connecting their one end to the PCB and the other end to the respective data pins of the 25-pin, D-type parallel-port male connector. This male connector connects to the female connector on the PC. An actual-size, single-side PCB for the circuit and its component layout are shown in Figs 5 and 6, respectively.

Testing procedure

1. Install Microsoft Visual Basic 6 on your system.

2. Fabricate or get the PCB shown in Fig. 5.

3. Connect the 8-data line male connector to the female connector on the PC.

4. Launch Visual Basic from the desktop and develop the application as explained in the software program section. Save the project file with extension '.vbp.' Alternatively, you can copy the executable file 'Arport' from the EFY-CD to your system.

5. Open 'Arport' and click 'Input Edit' box. You're prompted to input the data in decimal form. For example, input '5' and click 'On' button using mouse. The indicator on the screen will turn 'red.' Then LED7 and LED5 connected across the parallel port will glow, which corresponds to binary output '00000101.' The appliances connected to the respective output lines will turn on.

6. To turn off the appliances, click 'Off' button on the screen.

7. To exit the application, click 'Quit' button.

Download source code: http://www.efymag.com/admin/issuepdf/Device%20Control.zip

SOURCE CODE (Arport)

```
Private Declare Sub vbOut Lib "WIN95IO.DLL"
(ByVal AEPPort As Integer, ByVal AEPData as
Integer)
Dim AEPPort As Integer
Dim AEPData As Integer

Sub AEPOut(Data As Integer)
vbOut AEPPort, AEPData
End Sub
Private Sub Form_Load()
Shape1.Visible = False
pat = 0
End Sub
Private Sub Command1_Click()
AEPPort = &H378
pat = Text1.Text
AEPData = Val(pat)
AEPOut (Data)
If pat = "" Then GoTo y Else GoTo x
x:
Shape1.Visible = True
Shape2.Visible = False
y:
End Sub
Private Sub Command2_Click()
AEPPort = &H378
pat = 0
AEPData = Val(pat)
AEPOut (Data)
Shape1.Visible = False
Shape2.Visible = True
End Sub

Private Sub Command3_Click()
AEPPort = &H378
pat = 0
AEPData = Val(pat)
AEPOut (Data)
End
End Sub
```