A tutorial on Parallel port interfacing and programming for beginners. This tutorial covers both hardware and software aspects about programming the parallel port. It also includes a sample schematic and test program in Visual C++. You can make your first parallel port application running straight away.

Introduction

Parallel port is a simple and inexpensive tool for building computer controlled devices and projects. The simplicity and ease of programming makes parallel port popular in electronics hobbyist world. The parallel port is often used in Computer controlled robots, Atmel/PIC programmers, home automation, ...etc... Here a simple tutorial on parallel port interfacing and programming with some examples.

The primary use of parallel port is to connect printers to computer and is specifically designed for this purpose. Thus it is often called as printer Port or Centronics port (this name came from a popular printer manufacturing company 'Centronics' who devised some standards for parallel port). You can see the parallel port connector in the rear panel of your PC. It is a 25 pin female (DB25) connector (to which printer is connected). On almost all the PCs only one parallel port is present, but you can add more by buying and inserting ISA/PCI parallel port cards.

I don't have parallel port on My machine! What do I do ?

Unfortunately, as of today parallel port is a completely deprecated interfacing standard. It is impossible finding a new laptop with parallel port these days. Even almost all branded PCs already shead Parallel Ports from their back panels. Other alternative available in market such as PCI parallel port expansion cards and USB to Parallel Port converters aren't good for anything other than connecting your old printer because of their architectural difference.

As of now the only possible alternative is to use serial ports along with serial to parallel converters (Unfortunately serial ports aren't available on laptops these days) or use USB which is the most popular interface standard in the market as of today. But inherent complexity of implementing USB protocol and hardware requirements keep the hobbyists away for

experimenting with USB.

But there are great alternatives in market which uses USB but as easy as using parallel port. One such device is Numato Lab's 8 Channel USB GPIO Module . Though this is a USB based module, the user doesn't have to worry about the intricacies and complexities of USB protocol.

Parallel port modes

The IEEE 1284 Standard which has been published in 1994 defines five modes of data

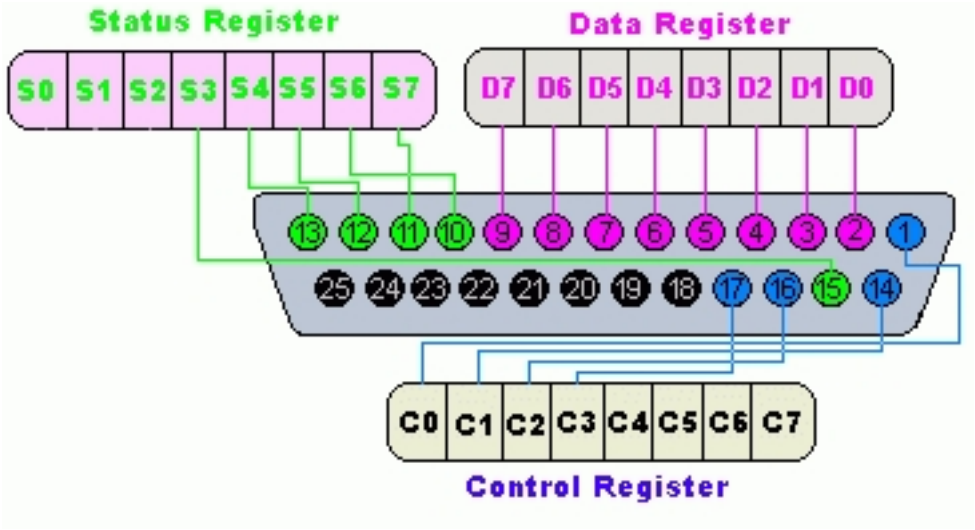
transfer for parallel port. They are,

1) Compatibility Mode
2) Nibble Mode
3) Byte Mode
4) EPP
5) ECP

The programs, circuits and other information found in this tutorial are compatible to almost all types of parallel ports and can be used without any problems (Not tested, just because of confidence !). More information on parallel port operating modes can be found here .

Hardware

The pin outs of DB25 connector is shown in the picture below



The lines in DB25 connector are divided in to three groups, they are

1) Data lines (data bus)
2) Control lines
3) Status lines

As the name refers , data is transferred over data lines , Control lines are used to control the peripheral and of course , the peripheral returns status signals back computer through Status lines. These lines are connected to Data, Control And Status registers internally . The details of parallel port signal lines are given below

Pin No (DB25)	Signal name	Direction	Register - bit
1	nStrobe	Out	Control-0
2	Data0	In/Out	Data-0
3	Data1	In/Out	Data-1
4	Data2	In/Out	Data-2
5	Data3	In/Out	Data-3
6	Data4	In/Out	Data-4
7	Data5	In/Out	Data-5
8	Data6	In/Out	Data-6
9	Data7	In/Out	Data-7
10	nAck	In	Status-6
11	Busy	In	Status-7
12	Paper-Out	In	Status-5
13	Select	In	Status-4

14	Linefeed	Out	Control-1
15	nError	In	Status-3
16	nInitialize	Out	Control-2
17	nSelect-Printer	Out	Control-3
18-25	Ground	-	-

Parallel port registers

As you know, the Data, Control and status lines are connected to there corresponding registers inside the computer. So by manipulating these registers in program , one can easily read or write to parallel port with programming languages like 'C' and BASIC.

The registers found in standard parallel port are ,

1) data register
2) Status register
3) Control register

As there names specifies, Data register is connected to Data lines, Control register is connected to control lines and Status register is connected to Status lines. (Here the word connection does not mean that there is some physical connection between data/control/status lines. The registers are virtually connected to the corresponding lines.). So what ever you write to these registers , will appear in corresponding lines as voltages, Of course, you can measure it with a multimeter. And What ever you give to Parallel port as voltages can be read from these registers(with some restrictions). For example , if we write '1' to Data register , the line Data0 will be driven to +5v. Just like this ,we can programmatically turn on and off any of the data lines and Control lines.

Where these registers are ?

In an IBM PC, these registers are IO mapped and will have unique address. We have to find these addresses to work with parallel port. For a typical PC , the base address of LPT1 is 0x378 and of LPT2 is 0x278. The data register resides at this base address , status register at baseaddress + 1 and the control register is at baseaddress + 2. So once we have the base address , we can calculate the address of each registers in this manner. The table below shows the register addresses of LPT1 and LPT2. Click here for tutorial on how to find base address of Parallel/Serial ports with sample source code.

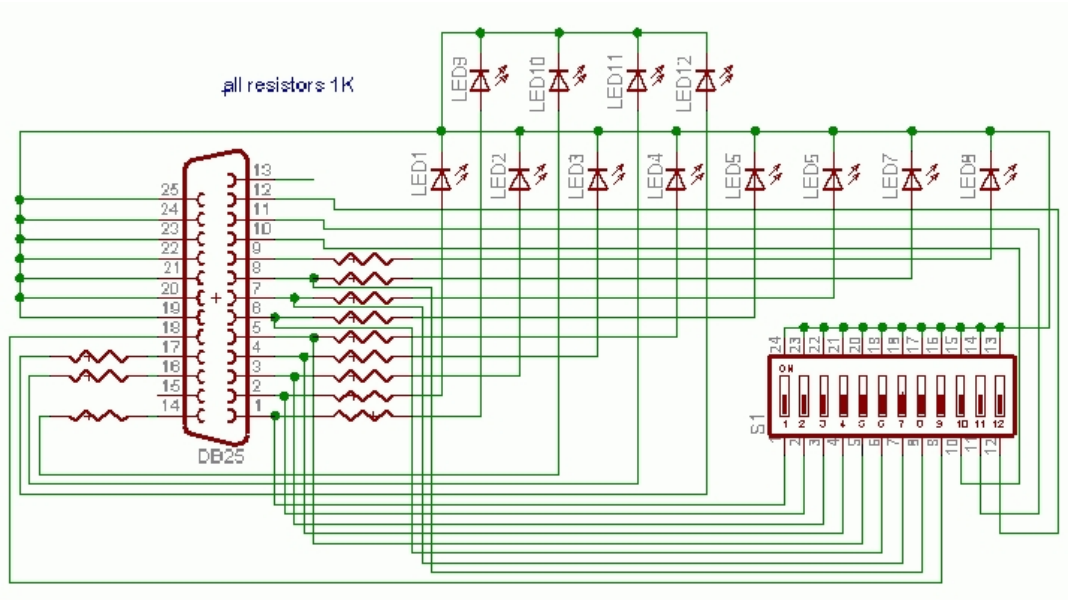
Register	LPT1	LPT2
data registar(baseaddress + 0)	0x378	0x278
status register (baseaddress + 1)	0x379	0x279
control register (baseaddress + 2)	0x37a	0x27a

Programming Concepts

Almost all programming languages allow programmers to access parallel port using some library functions. For example , Borland C is providing "Inportb" and "Outportb" functions to read or write IO mapped peripherals. But the examples provided here in this tutorial is written VC++ and can be easily ported to other compilers like Borland C and Turbo C. Visual Basic does not have any functions or support to access parallel port directly, but it is possible to add such capabilities to your VB application by writing a dll in VC++ and calling its exported functions from VB. VC++ provides two functions to access IO mapped peripherals, '_inp' for reading and '_outp' for writing. These functions are declared in "conio.h".

Hardware for testing sample programs

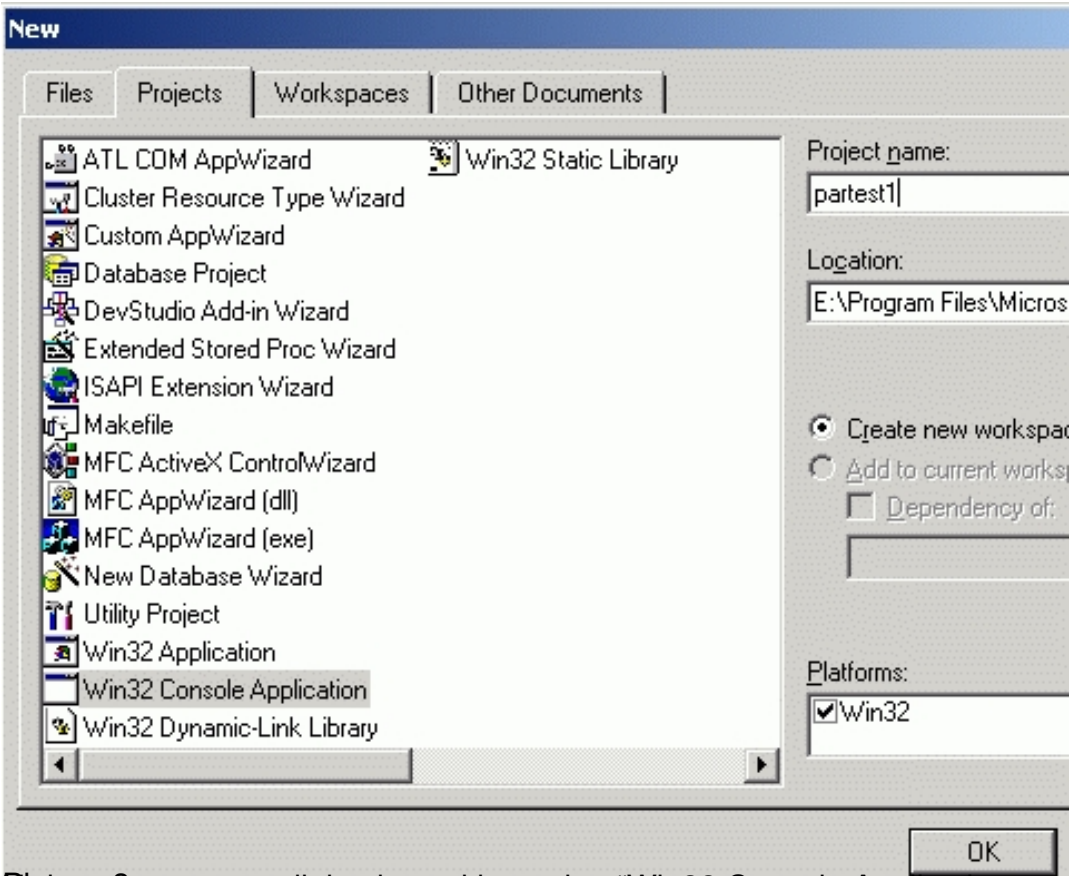
The schematic diagram of the test circuit is shown below. It is recommended to build this circuit before testing the sample programs



Sample program in VC++

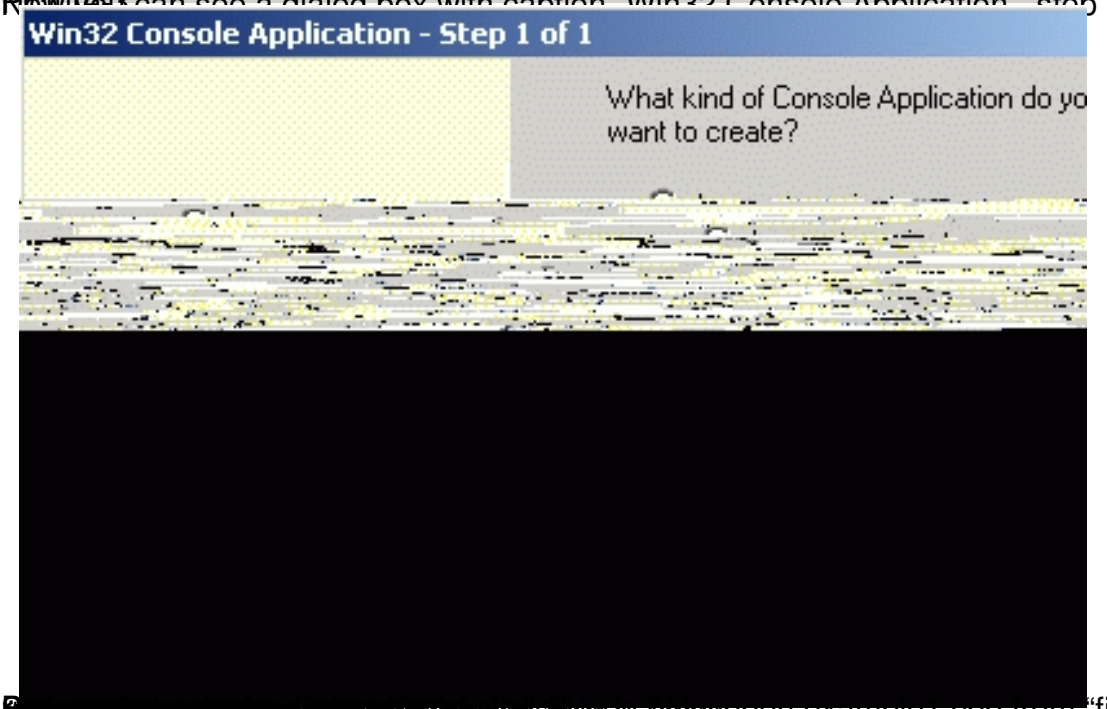
Writing a parallel port interfacing program in VC++ is very easy. Here is the steps to write your first parallel port interfacing application in VC++.

Start VC++ IDE , Select 'New' from File menu.Then select “Win32 Console Application” from “Projects” tab(picture-3). enter project name as “partest1” , then click OK button.



Picture-3

Now you can see a dialog box with caption “Win32 Console Application - step 1 of 1” (picture-4).



[redacted] open example1.cpp from “fileview” and [redacted] inpout32.dll [redacted]