

Application Note

Driving a character type LCD from a PC printer port

General:

Character (alphanumeric) type LCD modules are most commonly driven from an embedded microcontroller. It is sometimes desirable to be able to drive this type of module directly from a PC. This would allow the module to be tested or a demonstration or simulation to be set up quickly and with a minimum of engineering. This application note describes a method of driving an LCD character module from the printer port of a PC with minimal external hardware.

The printer port of a PC:

There are at least three versions of printer ports available now on PCs.

- SPP (Standard Parallel Port)
- EPP (Enhanced Parallel Port)
- ECP (Extended Capability Port)

To make this design as universal as possible it will be based on the least capable and oldest specification, SPP.

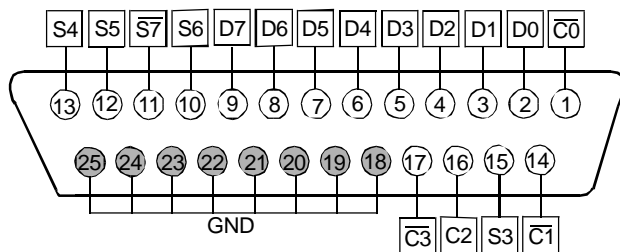


Figure 1

Figure 1 illustrates the pinout of the parallel port and the register that controls each pin. The connector drawing is the female connector on the PC.

To maintain compatibility with all versions of parallel ports now available on PCs only the output capabilities are used in this example. The LCD module provides a busy flag that can be read to control the timing of data and command writes. The timing of all data and command transfers is known and this is used to time the transfer of data and commands to the LCD in software.

The parallel port has 12 buffered TTL output pins which are latched and can be written under program control using the processor's Out instruction. The port also has input pins but they are not used in this example.

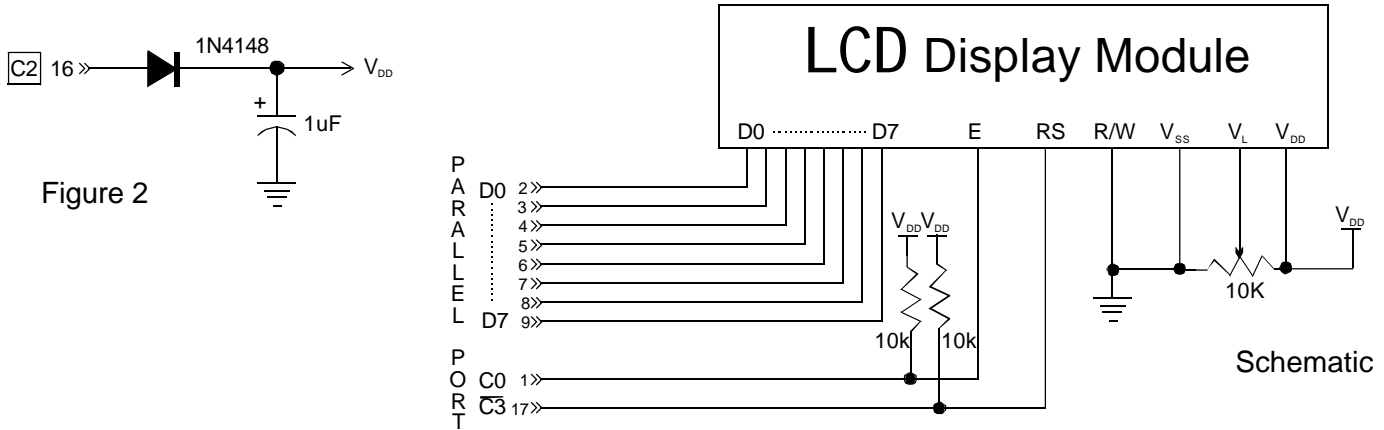
Hardware Description:

The suggested circuit is quite simple. The display, data and control lines are connected to the printer port lines as shown in Table 1 and the schematic. The read/write (RW) line is tied low to fix the display in the write mode. The Enable (E) and the Register Select (RS) lines are tied to two of the parallel port control lines. Most parallel ports have active or passive pull-ups on these lines, but not all. To be universal 10k pull-up resistors are shown on these two lines. Contrast control is provided by a 10k pot. DC power for the LCD and the back light, if installed, must be provided externally.

LCDs require very little power to operate, typically less than 5mA. The outputs on the parallel port are able to

Application Note

source up to 10mA, so it is possible to power the LCD from one of the output lines. This doesn't apply to the back light which requires from 50mA to 300mA. One of the control lines can be used for this purpose as shown



in Figure2. This also has the advantage of being able to shut off the display via software. A 1 on this line turns the display on and a 0 turns it off.

The type of cable used to connect the parallel port to the LCD will determine the maximum length of the cable. Ribbon cable, for instance, should not be used in lengths over 3'. A shielded, twisted pair cable can be used up to 50'. The quickest and most economical way of building a shielded, twisted pair in small quantities is to use a commercial printer cable of the desired length and cut off the connector that would normally connect to the printer. The wires are then prepared and connected to the LCD module.

Software:

Most contemporary PCs support 3 parallel ports at addresses 278/378/3BC. All values are in hex. Usually only one port is physically installed and in most systems it is at address 378 and is assigned to LPT1. An output instruction to the base address of the port, 278/378/3BC, will latch data to the data port of the LCD as shown below.

BIT	7	6	5	4	3	2	1	0
PIN	9	8	7	6	5	4	3	2
LCD	D7	D6	D5	D4	D3	D2	D1	D0

An output instruction to the base address of the port +2, 27A/37A/3BE, will latch the lower 4 bits of the data bus to the control pins of the LCD. Only two of these signals are needed to control most LCD character modules. The exception is the 40 character by 4 line modules. These modules have two controllers on them and have an extra enable line. The port pin C1 can be used for this 2nd enable on the 40x4 modules. The bits are assigned to the LCD as shown below.

BIT	7	6	5	4	3	2	1	0
PIN	-	-	-	-	17	16	14	1
LCD					RS		E2*	E

* = 40x4 modules only.

Application Note

Program Example:

The following sample code is written in Microsoft C and will display a message on a two line by 16 character LCD module. It can be used as a guide in working with larger or smaller modules as it has all of the elements needed. A total of 80 characters is written to the display which is the maximum any LCD character display with one controller can store. This program will thus work for a 1, 2 or 4 line display although only two lines will display characters on a 4 line module.

A more sophisticated way of writing characters is to issue a SET DD RAM ADDRESS command for each line to be written. Check with the data sheet for the particular module you are using to set the DD RAM address and line lengths.

The program first initializes the display and then sends the two lines of data. The displayed message should read:

```
>411 Technology<
ABC abc 123,!@$?
```

```
/* Sample Software to display a message on a 16 x2 character LCD module
/* from a parallel port of a PC

#include <time.h>
#include <conio.h>
#include <string.h>

#define PORTADDRESS 0x378 /*Enter port address here */

#define DATA PORTADDRESS+0 /*LCD data port */
#define CONTROL PORTADDRESS+2 /*LCD control port */

void main (void)

{
/* Total of 40 characters including spaces in each line of string */
char string[] = {">411 Technology< "
                "ABC abc 123,!@$? "};
char init[10];
int count;
int len;
init[0] = 0x0F; /* Initialize display */
init[1] = 0x01; /* Clear display */
init[2] = 0x38; /* Dual line 8 bits */

_out(CONTROL, _inp(CONTROL) & 0xDF); /* Reset control port */
```

Application Note

```
_out(CONTROL, _inp(CONTROL) | 0x08); /*Set RS */

/* Initialization routine */

for (count = 0; count <= 2; count++)
{
    _out(DATA, init[count]);
    _out(CONTROL, _inp(CONTROL) | 0x01; /*Set Enable */
    delay(20);
    _out(CONTROL, _inp(CONTROL) & 0xFE; /*Reset Enable */
    delay(20);
}

_out(CONTROL, _inp(CONTROL) & 0xF7); /*Reset RS */

/* Output the message */

len = strlen(string);

for (count = 0; count < len; count++)
{
    _out(DATA, string[count]);
    _out(CONTROL, _inp(CONTROL) | 0x01; /*Set Enable */
    delay(2);
    _out(CONTROL, _inp(CONTROL) & 0xFE); /*Reset Enable */
    delay(2);
}
}
```

References:

Specifications for the Samsung KS0066 LCD controller chip. www.usa.samsungsemi.com/search/