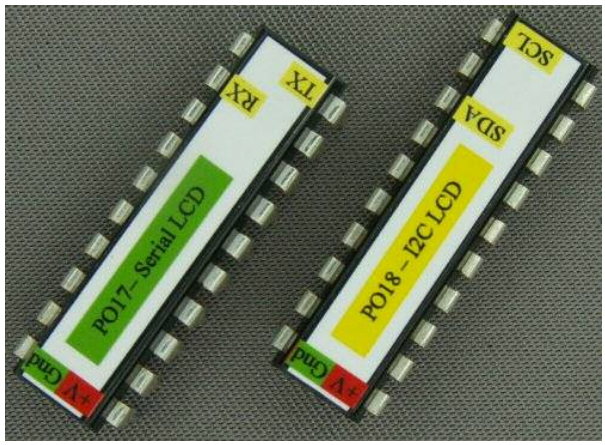


P017

I2C or Serial LCD

Product specification Sep. 2013 V0.a

Contents

I2C or Serial LCD P017/8

Rev	Change
Sep 2013	Preliminary

1. Introduction

The P017 is a serial LCD controller and the P018 is an I2C LCD controller. This is supplied and an IC but there is also a PCB.

A special feature is that it has three PWM back light outputs and so can control the new tri-colour backlight LCD.

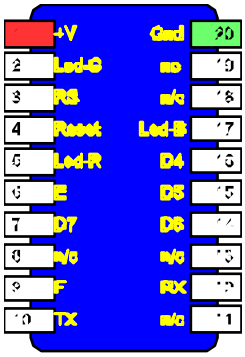
More data and examples with free software can be found at www.pichips.co.uk

2. Features

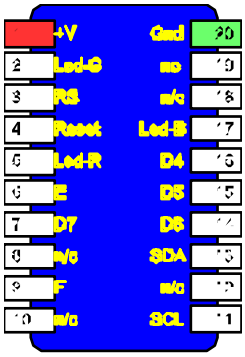
- Wide voltage range 2.5V to 5.5V
- 3 PWM back light outputs
- Three 10 bit ADC channels
- Simple serial or I2C protocol
- User configurable EEPROM

3. Electrical interface

The IC is a 20 pin DIL for ease of use and has two versions:



PO17 Serial IC

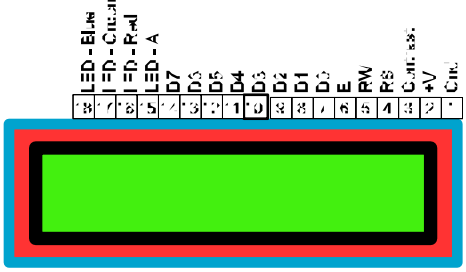


PO18 I2C IC

There are two versions of the IC, the only electrical difference is that one has TX,RX pins and the other has SCL, SDA pins.

3.1. Connecting to LCD

Nearly all character based LCD's now have the same electrical interface and shown is the most common one.



TRI-Colour back light LCD shown

Normally this type of display has 16 pins with pin 15 and 16 taking care of the back light. On a tri-colour led backlight display there are two extra pins for the other two LED's, these can be ignored for 'normal' back lit displays.

3.1.1.Standard LCD Wiring

LCD PIN	IC PIN	Function
1	20	Ground
2	1	+V, this should match the LCD normally 5V but can be 3.3V
3		Contrast, connect to trimmer see circuit diagram
4	3	RS - lCD control
5		Ground
6	6	E LCD control
7		Leave unconnected
8		Leave unconnected
9		Leave unconnected
10		Leave unconnected
11	16	D4 data line
12	15	D5 data line
13	14	D6 data line
14	7	D7 data line
15		+V
16	5	LED red or if there is one back light then use this pin
17	2	LED green or leave disconnected
18	17	LED blue or leave disconnected

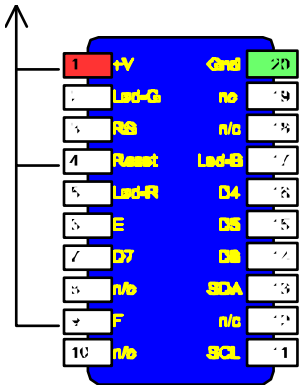
IC to LCD table

The above table will apply to the majority of character LCD types, these are usually specified as so many characters x so many lines. e.g. 20x4

3.2. IC Pin out

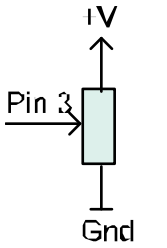
Pin	Description
1	+V, this can range from 2.5V to 5.5V but it would normally match the LCD voltage requirements
2	PWM output for driving a back light and can be set from 0 to +V
3	RS control for the LCD
4	Reset, This **must** be tied to +V, taking this low will reset the IC. It can be tied to a resistor if a reset needs to be implemented. The IC will reset on power up.
5	PWM output for driving a back light and can be set from 0 to +V. This is the default and can be used for displays with monochrome back lights
6	E control for the LCD
7	D7 connected to data input 7 of the LCD
8	no connection
9	F this **must** be tied to +V for normal operation
10	TX (output) line for serial devices no connection for I2C devices
11	SCL (clock) for I2C devices, no connection for serial devices
12	RX (input) for serial devices no connection for I2C devices
13	SDA (data) line for I2C devices no connection for serial devices
14	D6 connected to data input 6 of the LCD
15	D5 connected to data input 5 of the LCD
16	D4 connected to data input 4 of the LCD
17	PWM output for driving a back light and can be set from 0 to +V
18	no connection
19	no connection
20	Ground

IC pin out Table for P017/8



For correct operation Pins 4 and 9 must be tied to +V.

3.2.1. Contrast



This is the normal contrast arrangements using a trimmer or potentiometer. The resistance should be between 1 and 5K.

3.3. I2C

The I2C interface is the standard arrangement, somewhere along the bus a pull up resistor is required for the SCL and SDA lines.

The default device address is 0x70

3.4. Serial

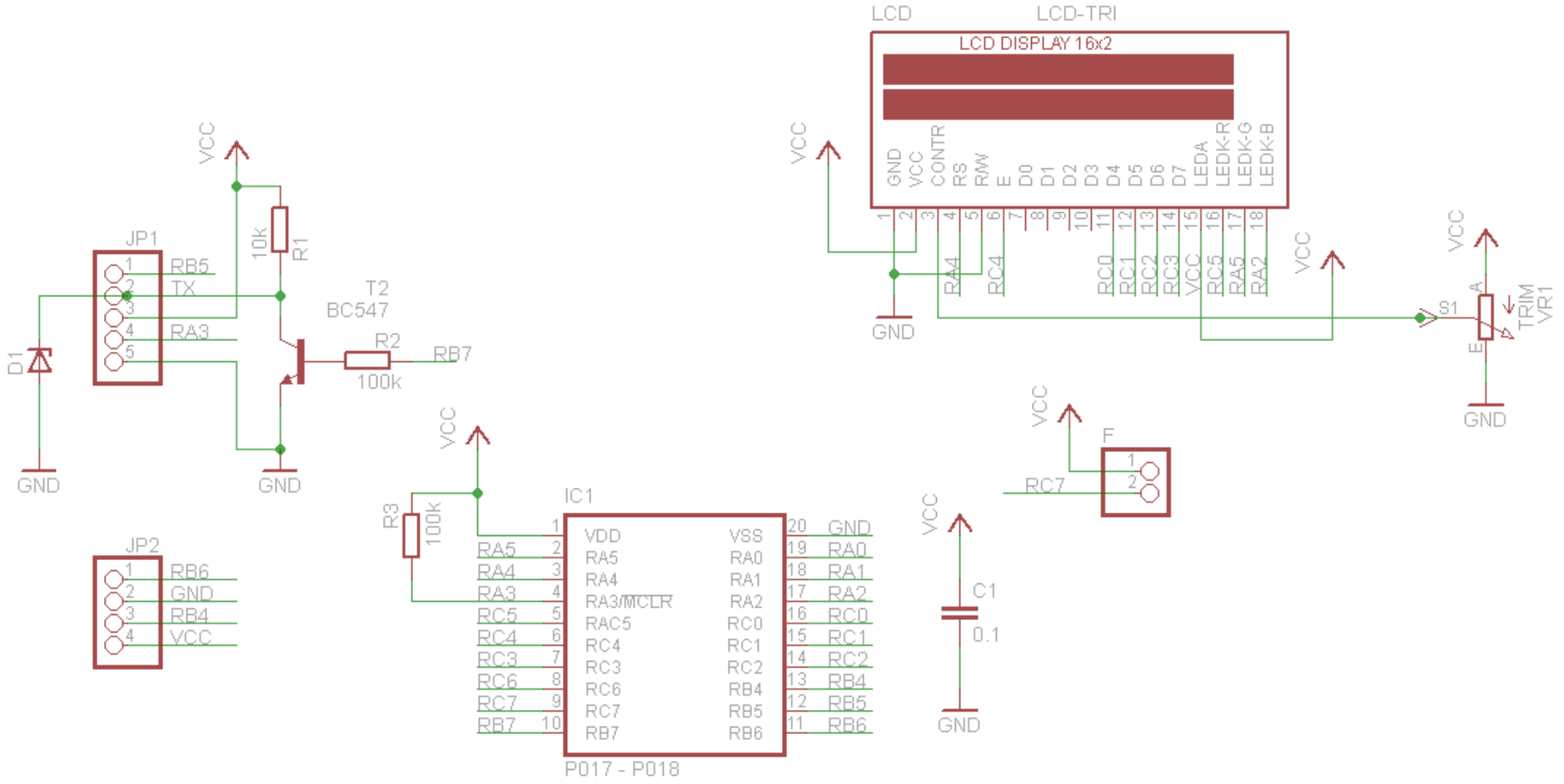
The serial interface is a standard 1 start bit 8 data bits and 1 stop bit and is initially set to 9600 Baud. This is user changeable from 2400 to 115200 in 8 steps.

The interface can be connected to a UART or USB to serial device and expected the voltage to be TTL levels (0 and 5V or 0 and 3.3V). By default and when connecting directly to one of the above devices the output (TX) is correct for a single device.

There is an opportunity to have more than one device share the TX line and this is achieved with an open collector circuit. The circuit diagram of the PO17 PCB shows how this can be achieved. When using this circuit the output (TX) requires inverting and this is done via a setting in the EEPROM.

By default the serial interface is set at 9600 Baud and the serial device address is '**p**'.

4. Circuit Diagram of the PO17/8 PCB



5. Serial Command Set

Default address ('p')

The commands are sent to the serial interface byte by byte, however for convenience the byte values chosen coincide with ASCII characters. This makes debugging on a terminal very easy.

All commands will be referred to by their ASCII value but remember on a microcontroller host system, sending 'a' on a terminal is just the same as sending the value 97.

Where appropriate all of the serial commands listed in the summary table below have an I2C equivalent.

The full command details are listed later on in the text.

Command Set Summary	
b	Back light
c	LCD command
d	LCD data
W	Write to EEPROM
R	Read from EEPROM
I	Invert TX
C	Reset
D	Device ID number
V	Firmware Version
H	Hello

Table 1 Command Set summary

All of the above commands require a device address to be specified before sending the command and also **every command sequence must be terminated with CR** ("\r") (13) (0xd).

The device will return ACK (6) on all successful commands and NACK (21) on unsuccessful commands.

Any command beginning with an address that does not match the devices address is ignored.

6. EEPROM values

The EEPROM contains important values that control the way the device behaves. All of the values can be changed by the user using the 'W' command.

The EEPROM consists of 255 bytes and in general the first 16 bytes are used by the system, the second 16 byte are used by the device and the rest of the bytes can normally be used by the user.

Adr	Default Value	Description
0	0	System Use
1	112	Device address
2	6	ACK value
3	21	NACK value
4	3	Baud rate
5	1	Error reporting 1 = on
6	13	End of line
7	1	Invert TX 1=invert
14	112	Device address copy
16	10	Back light red value
17	10	Back light green value
18	10	back light blue value
32		Sign on message start
250	112	Device address copy

Table 2 EEPROM use

The user is free to use any locations that are not occupied by the system but for future use it is best to avoid locations below 32.

EEPROM values are only read on start up so when changing values they will not normally take effect until the device is reset.

6.1. Address

These EEPROM locations contains the device address. By convention the address is set to values between the values 97 to 122, no checking is made by the device so setting values outside this range may or may not work.

For security the address is stored in three places and to change the address of the device at least two of the locations need to be set otherwise the device will detect the anomaly at start up and revert to the majority value.

Normally to change the address of a device locations 1 and 14 are both changed. The device will detect this at start up and change the address in location 250 to match.

6.2. ACK character

By default this is 6 but can be changed using the EERPOM Write command. The effect will not be implemented until the device is reset.

6.3. NACK character

By default this is 21 but can be changed using the EERPOM Write command. The effect will not be implemented until the device is reset.

6.4. Baud Rate

The Baud rate has the following values:

0. no valid

1. Baud rate is fixed at 2400
2. Baud rate is fixed at 4800
3. Baud rate is fixed at 9600 (default*)
4. Baud rate is fixed at 14400
5. Baud rate is fixed at 19200
6. Baud rate is fixed at 38400
7. Baud rate is fixed at 57600
8. Baud rate is fixed at 115200

6.5. Turn off Error reporting

By default error reporting is enabled and this will be reported and an output prefixed by Error, for example '**Error 2**'. This may get in the way of the program trying to control the device and so it can be disabled with this command. The effect will not be implemented until the device is reset. This does not apply to I2C if available.

6.6. CR Character

By default this is 13 which is the standard ASCII CR and the whole protocol relies on this being at the end of every command. It may be that this is unsuitable in some systems and so this can be changed.

6.7. Back light values

The back light values can range from 0 (off) to 10 (full on) and the settings in the EERPOM hold the values that are set on reset.

6.8. Sign on Message

The sign on message can be changed form the default by writing to theses EEPROM locations. To access an LCD command use a value of 1 first and always finish the message with 0.

For example to clear the display and then have "fred" written the following bytes would be in EEPROM

EEPROM	Byte	Notes
32	1	Command follows
33	1	This is the command to clear the screen
34	102	'f'
35	114	'r'
36	101	'e'
37	100	'd'
38	0	Terminate message with 0

7. Commands

All serial commands are proceeded by an address and terminate with CR (0xd). In the examples given below the address is 'f' or 0x66

When a command is accepted by the device it always returns ACK which by default is the value 6. If the device rejects the command then it will return NACK, value 21

Serial	I2C	range	Default Value	EEPROM Location	Description
b,r,g,b	1,r,g,b	0-10			**Back light Control** The back light brightness is specified as 3 values to cater for displays that have colour back lights. If there is just one backlight then normally the red values is used but all three still need specifying. **Examples**: For a single colour backlight LCD set the value to 1/2 brightness **p5,0,0** For a multi colour display set it as yellow, full brightness. **p10,10,0 (red+gree=yellow)** **I2C** a single byte is used, for the previous example: **s 1 10 10 0 p**
c	2	0-255			**LCD Command** This will send a command (as opposed to data) to the LCD. Commands can control the cursor or clear the screen etc. Consult the data sheet for the LCD display concerned. **Examples**: Clear the screen **pc1** Turn off the cursor **pc12** Cursor block flashing **pc15** Normal cursor **pc14** **I2C** Example of the above **s 2 14 p**
d	3	Send byte's			**Data** This sends data to the display, it will display what is sent. Multiple bytes can be sent up to the internal buffer size which is 80 bytes. The data will be sent on receiving CR

					Example: **dfred** (prints 'fred' to the screen) **I2C** This will also accept multiple bytes but **MUST** be terminated by 13 Example: Send 'A' **s 3 65 13 p** Send 'fred' **s 3 102 114 101 100 13 p**
Wn,m	0x91	n=0-255 m=0-255			**Write to EEPROM** This will write a single byte to an EEPROM location, the command format is: <adr>W<EEPROM address>,<value> Care should be taken when using this command for two reasons: 1) The EEPROM can only be written to a certain number of times all be it a large number. 2) The EEPROM contains system information that is used at start up a wrong value could mean loss of communication with the device that would require a factory reset. **I2C Example write 23 to location 7** s 0x91 7 23 p
Rn,m	0x90	n=0-255 m=0-255			**Read from EEPROM** The EEPROM values can be read with this command given a starting address and the number of bytes to read. <adr>R<"EEPROM adr><#bytes> This example will output the first 16 bytes of EEPROM **fR0,16** The output from the device will commence after receiving CR and will consist of a string of data terminated with ACK. The sting will be in the form of text delimited by ',' and all of the values will be decimal. An example of output for the first 5 bytes of EERPOM would be: "0,97,6,21,0"<ACK> **I2C** I2C will read only single values at a time, to read from location 3: s 0x90 3 r g-1 p
D	0xa1				**Device ID** Returns a number representing the device product number as a string **fD**

					Output from the above would be: "4601"<ACK> **I2C** returns two bytes representing a 16 bit number, high byte first s 0xa1 r g-2 p
I	n/a		1	7	**Toggle Inverted** This command will invert the output of the TX pin and store the value to EEPROM. A value of 1 is inverted and 0 is not inverted. The command will toggle form one to the other. If the TX pin requires changing, instead of ACK being returned by the device a value of 0x3e ('>') is returned. This is easily detected and this command can be issued to correct it. Example **fI** This is only needed as a **one time** operation as the change is automatically written to EEPROM
C	0x95				**Reset** Resets an individual device. This is a soft reset. A soft reset will normally be the same as a reset at start-up but this may not always be the case. Obviously no ACK will be returned by this command. Example **fC** **I2C** **s 0x95 p**
V	0xa0				**Version** Returns the firmware version as a string in the format "H.L" Example **fV** **I2C** Sends two bytes, major revision first so 2.7 world be 2 and 7
H	n/a				**Hello** This command is used to check what devices are on the bus. It simply returns ACK but where there is more then one device on the bus the following sudo code will list them: for j = 97 to 122 Send(chr$(j)+"H\r") if ack received then print device j found If a device is found then the other attributes such as device ID can be obtained. Also user information could be stored in the devices EEPROM and retrieved.