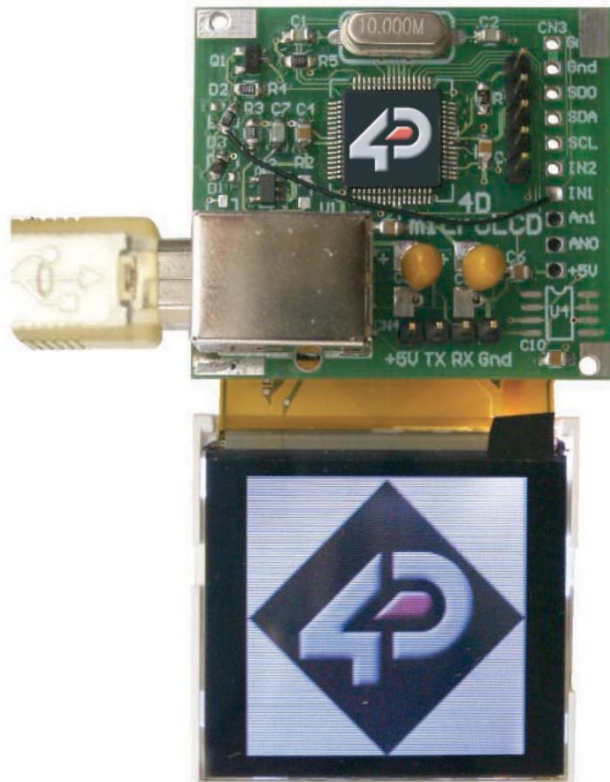


MicroLCD

USERS MANUAL

Revision 1.1



4D Systems



4D Systems

MicroLCD

PROPRIETARY INFORMATION

The information contained in this document is the property of [4D Systems Pty. Ltd.](#), and may be the subject of patents pending or granted, and must not be copied or disclosed without prior written permission. It should not be used for commercial purposes without prior agreement in writing.

[4D Systems Pty. Ltd.](#) Endeavours to ensure that the information in this document is correct and fairly stated but does not accept liability for any error or omission. The development of [4D Systems](#) products and services is continuous and published information may not be up to date. It is important to check the current position with [4D Systems](#).

Contact details are available from the company web site at www.4dsystems.com.au

All trademarks recognised and acknowledged.

Copyright [4D Systems Pty. Ltd.](#) 2000-2005



Table of contents

1. μ LCD Description

- 1.1 Introduction
- 1.2 μ LCD features

2. μ LCD Command set

- 2.1 Command set
 - 2.1.1 Erase Screen
 - 2.1.2 Background Colour
 - 2.1.3 Put Pixel
 - 2.1.4 Read Pixel
 - 2.1.5 Draw Circle
 - 2.1.6 Draw Line
 - 2.1.7 Font Size
 - 2.1.8 Opaque or Transparent Text
 - 2.1.9 Place Text Character (formatted)
 - 2.1.10 LCD Display Control Functions
 - 2.1.11 Add User Bitmapped Character
 - 2.1.12 Display User Bitmapped Character
 - 2.1.13 Paint Area
- 2.2 μ LCD Serial Interface
- 2.3 μ LCD USB Interface

3. Specifications

- 3.1 μ LCD pin-outs
- 3.2 65,536 Colour Bitmap Organisation
- 3.3 Power-Up Reset

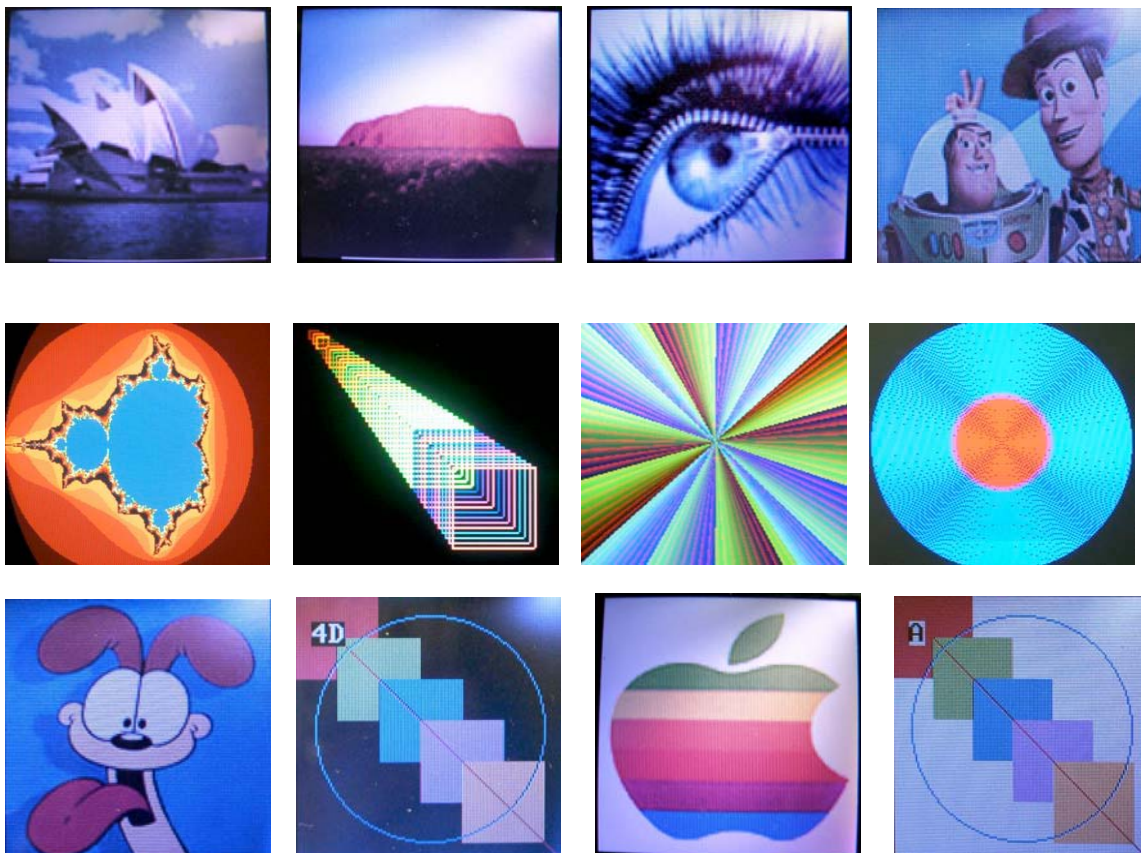


1 μ LCD Description

1.1 Introduction

The μ LCD is a compact & cost effective all in one 'SMART' LCD Display with an embedded graphics controller that will deliver 'stand-alone' functionality to your project. The 'simple to use' embedded commands not only control background colour but can produce text in a variety of sizes as well as draw shapes (which can include user definable bitmapped characters such as logos) in 65,536 colours whilst freeing up the host processor from the 'processor hungry' screen control functions. This means a simple micro-controller with a standard serial or USB interface can drive the μ LCD module with total ease.

Figures below show some of the graphics capability of the μ LCD.





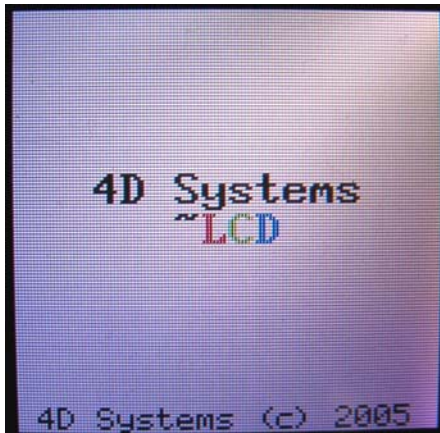
4D Systems

MicroLCD

1.2 μ LCD Features

The μ LCD is aimed at being integrated into a variety of different applications via a wealth of features designed to facilitate any given functionality quickly and cost effectively and thus reduce 'time to market'. These features are as follows:

- 65,536 colours.
- Three selectable font sizes (5x7, 8x8 and 8x12).
- User defined bitmapped characters, 64 by 8x8.
- Vector drawing controls for circles, lines, squares.
- Standard 'built in' ASCII character set (See fig 5).
- Logic level RS232 serial interface to μ LCD from host.
- Auto baud rate detection from 300 baud to 128Kbit/sec.
- USB Interface available (micro-USB).





2 μ LCD Command Set

The heart of the μ LCD is the easy to understand command set. This comprises of a handful of easy to learn instructions that can draw lines, circles, squares, etc, to provide a full text and graphical user interface. The commands are sent to the μ LCD via its serial connection (4 pin header). **Please note that the Rx and the Tx signals are at 3.3V levels. If interfacing to a host system running at 5V levels, then 1K series resistors must be inserted between the Host Tx/Rx and the μ LCD Rx/Tx signals...**

2.1 Command set

(E) Erase Screen	1 byte	2.1.1	8
(B) Background Colour	3 bytes	2.1.2	9
(P) Put Pixel	5 bytes	2.1.3	10
(R) Read Pixel	3 bytes	2.1.4	11
(C) Draw Circle	6 bytes	2.1.5	12
(L) Draw Line	7 bytes	2.1.6	13
(F) Font Size	2 bytes	2.1.7	14
(O) Opaque or Transparent Text	2 bytes	2.1.8	15
(T) Place Text Character (formatted)	6 bytes	2.1.9	16
(Y) LCD Display Control functions	3 bytes	2.1.10	17
(A) Add User Bitmapped Character	10 bytes	2.1.11	18
(D) Display User Bitmapped Character	6 bytes	2.1.12	19
(p) paint Area	7 bytes	2.1.13	20



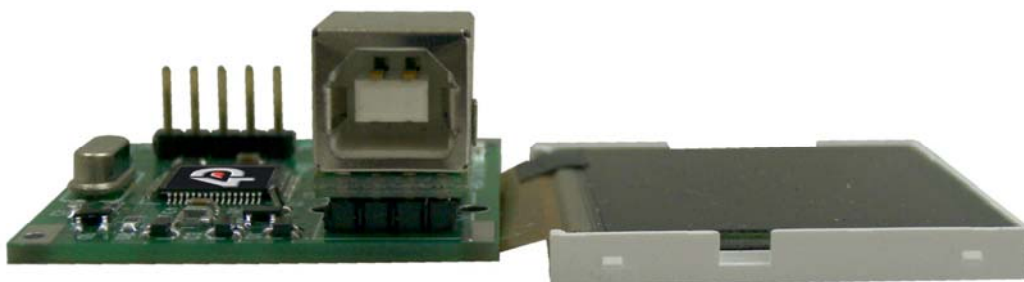
COMMAND PROTOCOL

The following are each of the commands with the correct syntax. Please note that all command examples listed below are in hex (**00hex**). Due to the high colour depth of the μ LCD, a pixel colour value will not fit into a single byte, a byte can only hold a maximum value of 255. Therefore the colour is represented as a 2 byte value, **colour(msb)**, **colour(lsb)**. The most significant byte (msb) is transmitted first followed by the least significant byte (lsb). This format is called the big endian. So for a 2 byte colour value of **013Fhex** the byte order can be shown as (**01hex**),(**3Fhex**).

NOTE: When transmitting the command and data bytes to the μ LCD, do not include any separators such as commas ',' or spaces ' ' or brackets '(' ')' between the bytes. The examples show these separators purely for legibility; these must not be included when transmitting data to the μ LCD.

When a μ LCD command is sent, the μ LCD will reply back with a single acknowledge byte called the **ACK (06hex)**. This tells the host that the command was understood and the operation is completed. It will take the μ LCD anywhere between 1 to several milliseconds to reply back with an **ACK**, depending on the command and the operation the μ LCD has to perform. If the μ LCD receives a command that it does not understand it will reply back with a negative acknowledge called the **NAK (15hex)**.

If a command that has 5 bytes but only 4 bytes are sent, the command will not be executed and the μ LCD will wait until another byte is sent before trying to execute the command. There is no timeout on the μ LCD when incomplete commands are sent. The μ LCD will reply back with a **NAK** for each invalid command it receives. For correct operation make sure the command bytes are sent in the correct sequence.





4D Systems

MicroLCD

2.1.1 Erase Screen (E)

Syntax : cmd

cmd : 45hex, Eascii

Description : This command clears the entire screen using the current background colour.

Example : 45hex

Clear the screen.





2.1.2 Background Colour (B)

Syntax : cmd, colour(msb), colour(lsb)

cmd : 42hex, Bascii

colour : pixel colour value: 2 bytes (16 bits) msb, lsb
65,536 colours to choose from

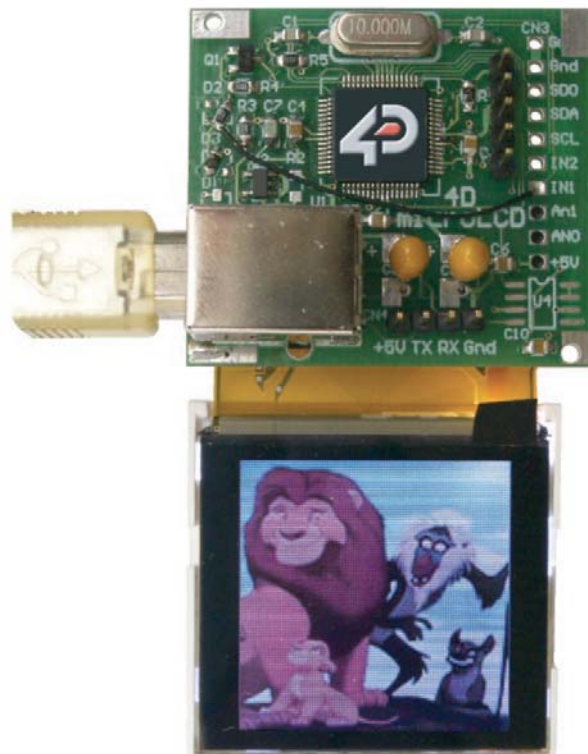
Black = 0000hex, 0dec

White = FFFFhex, 65,535dec, 1111111111111111bin

Description : This command sets the current background colour. Once this command is sent, only the background colour will change. Any other object on the screen with a different colour value will not be affected.

Example : 42hex, FFFFhex

Set the background colour to value 65,535 (white).





2.1.3 Put Pixel (P)

Syntax : cmd, x, y, colour(msb), colour(lsb)

cmd : 50hex, Pascii

x : horizontal pixel position. 0dec to 127dec (00hex to 7Fhex).

y : vertical pixel position. 0dec to 127dec (00hex to 7Fhex).

colour : pixel colour value: 2 bytes (16 bits) msb, lsb

65,536 colours to choose from

Black = 0000hex, 0dec

White = FFFFhex, 65,535dec, 1111111111111111bin

Description : This command will put a coloured pixel at location (x, y) on the screen.

Example : 50hex, 01hex, 0Ahex, 00hex, 00hex

Puts a black (0000hex) pixel at location x = 01dec (01hex) and y = 10dec (0Ahex).





2.1.4 Read Pixel (R)

Syntax : cmd, x, y

cmd : 52hex, Rascii

x : horizontal pixel position. 0dec to 127dec (00hex to 7Fhex).

y : vertical pixel position. 0dec to 127dec (00hex to 7Fhex).

Description : This command will read the colour value of pixel at location (x, y) on the screen and return it to the host. This is a useful command when for example a white pointer is moved across the screen and the host can read the colour on the screen and switch the colour of the pointer when it's on top of a light coloured area.

Example : 52hex, 01hex, 01hex

μLCD reply : 00hex, 1Fhex

Reads a blue (001Fhex) pixel at location x = 1dec (01hex) and y = 1dec (01hex).



2.1.5 Draw Circle (C)

Syntax : cmd, x, y, rad, colour(msb), colour(lsb)

cmd : 43hex, Cascii

x : horizontal circle centre position. 0dec to 127dec (00hex to 7Fhex).

y : vertical circle centre position. 0dec to 127dec (00hex to 7Fhex).

rad : radius size of the circle. 0dec to 127dec (00hex to 7Fhex).

colour : circle colour value: 2 bytes (16 bits) msb, lsb
65,536 colours to choose from

Black = 0000hex, 0dec

White = FFFFhex, 65,535dec, 1111111111111111bin

Description : This command will draw a coloured circle centred at (x, y) with a radius determined by the value of rad.

Example : 43hex, 3Fhex, 3Fhex, 22hex, FFhex, FFhex

Draws a white circle (FFFFhex) centred at x = 63dec (3Fhex) and y = 63dec (3Fhex) with a radius of 34dec (22hex).





2.1.6 Draw Line (L)

Syntax : cmd, x1, y1, x2, y2, colour(msb), colour(lsb)

cmd : 4Chex, Lascii

x1 : horizontal position of line start. 0dec to 127dec (00hex to 7Fhex).

y1 : vertical position of line start. 0dec to 127dec (00hex to 7Fhex).

x2 : horizontal position of line end. 0dec to 127dec (00hex to 7Fhex).

y2 : vertical position of line end. 0dec to 127dec (00hex to 7Fhex).

colour : line colour value: 2 bytes (16 bits) msb, lsb
65,536 colours to choose from

Black = 0000hex, 0dec

White = FFFFhex, 65,535dec, 1111111111111111bin

Description : This command will draw a coloured line from point (x1, y1) to point (x2, y2) on the screen.

Example : 4Chex, 00hex, 00hex, 7Fhex, 7Fhex, FFhex, FFhex

Draws a white line from (x1=0, y1=0) to (x2=127, y2=127).





2.1.7 Font Size (F)

Syntax : cmd, size

cmd : 46hex, Fascii

size : = 00hex : 5x7 small size font
= 01hex : 8x8 medium size font
= 02hex : 8x12 large size font

Description : This command will change the size of the font according to the value set by **size**. Changes take place after the command is sent. Any character on the screen with the old font size will remain as it was.

Example1: 46hex, 00hex Select small 5x7 fonts
Example1: 46hex, 01hex Select medium 8x8 fonts
Example1: 46hex, 02hex Select large 8x12 fonts





2.1.8 Opaque / Transparent Text (O)

Syntax : cmd, mode

cmd : 4Fhex, Oascii

mode : = 00hex : Transparent Text, objects behind the text can be seen.
= 01hex: Opaque Text, objects behind text is blocked by background

Description : This command will change the attribute of the text so that an object behind the text can either be blocked or transparent. Changes take place after the command is sent.

This command will change the attribute so that when a character is written, it will either write just the character alone (Transparent Mode) so any original character will be seen as well as the new, or overwrite any existing data with the new character.

Example1: 4Fhex, 00hex Transparent Mode

Example2: 4Fhex, 01hex Opaque Text



2.1.9 Place Text Character (formatted) (T)

Syntax : cmd, char, column, row, colour(msb), colour(lsb)

cmd : 54hex, Tascii

char : inbuilt standard ASCII character, 32dec to 127dec (20hex to 7Fhex)

column : horizontal position of character, see range below:
0 - 20 for 5x7 font, 0 - 15 for 8x8 and 8x12 font.

row : vertical position of character:
0 - 15 for 5x7 and 8x8 font, 0 – 9 for 8x12 font.

colour : character colour value: 2 bytes (16 bits) msb, lsb
65,536 colours to choose from
Black = 0000hex, 0dec
White = FFFFhex, 65,535dec, 1111111111111111bin

Description : This command will place a coloured ASCII character (from the ASCII chart) on the screen at a location specified by (**column, row**). The position of the character on the screen is determined by the predefined horizontal and vertical positions available for a given core.

Example : 54hex, 41hex, 00hex, 00hex, Ffhex, FFhex

Place character 'A' (41hex) at column = 0, row = 0, colour = white (65,535).



2.1.10 LCD Display Control Functions (Y)

Syntax : cmd, mode, value

cmd : 59hex, Yascii

mode : = 00hex : **BACKLIGHT CONTROL.**

- value** = 00hex: Backlight OFF
- = 01hex: Backlight DIM (default). LED current = 50mA
- = 02hex: Backlight BRIGHT. LED current = 80mA

mode : = 01hex : **DISPLAY ON/OFF.**

- value** = 00hex: Display OFF
- = 01hex: Display ON

mode : = 02hex : **LCD CONTRAST.**

- value** = 0dec to 40dec : Contrast range (default = 23dec)

mode : = 03hex : **LCD POWER-UP/POWER-DOWN.**

- value** = 00hex: LCD Power-Down
- = 01hex: LCD Power-Up

Note: It is imperative that the LCD be issued with the Power-Down command before switching off the power. This command switches off the internal voltage boosters and current amplifiers and they need to be turned off before main power is removed. If the power is removed without issuing this command, the LCD maybe damaged. This command also turns off the backlight. This command need not only be issued to shutdown but can be issued to conserve power by turning off the display and the backlight.

The Power-Up command does not need to be executed when applying power. If a Power-Down command has been issued and Power is not switched off, the Power-Up command can be sent to Power the display back up again.



2.1.11 Add User Bitmapped Character (A)

Syntax : cmd, char#, data1, data2,, dataN

cmd : 41hex, Aascii

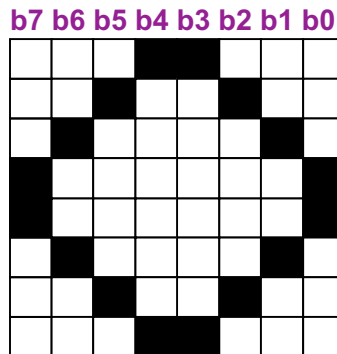
char# : bitmap character number to add to memory:
0 to 63 (00h to 3Fh), 64 characters of 8x8 format.

data1 to dataN : number of data bytes that make up the composition and format of the bitmapped character. The 8x8 bitmap composition is 1 byte wide (8bits) by 8 bytes deep which makes N = 1x8 = 8.

Description : This command will add a user defined bitmapped character into the internal memory.

Example1: 41hex, 01hex, 18hex, 24hex, 42hex, 81hex, 81hex, 42hex, 24hex, 18hex

This adds and saves user defined 8x8 bitmap as character number 1 into memory as seen below.



- data1 (hex = 18h)
- data2 (hex = 24h)
- data3 (hex = 42h)
- data4 (hex = 81h)
- data5 (hex = 81h)
- data6 (hex = 42h)
- data7 (hex = 24h)
- data8 (hex = 18h)

Example of a 8x8 user defined bitmap



2.1.12 Display User Bitmapped Character (D)

Syntax : cmd, char#, x, y, colour(msb), colour(lsb)

cmd : 44hex, Dascii

char# : which user defined character number to display from the selected group. 0dec to 63dec (00hex to 3Fhex), of 8x8 format.

x : horizontal display position of the character. 0dec to 127dec (00hex to 7Fhex).

y : vertical display position of the character. 0dec to 127dec (00hex to 7Fhex).

colour : character colour value: 2 bytes (16 bits) msb, lsb
65,536 colours to choose from

Black = 0000hex, 0dec

White = FFFFhex, 65,535dec, 1111111111111111bin

Description : This command displays the previously defined user bitmapped character at location (x, y) on the screen. User defined bitmaps allow drawing & displaying unlimited graphic patterns quickly & effectively.

Example 1: 44hex, 01hex, 00hex, 00hex, F8hex, 00hex

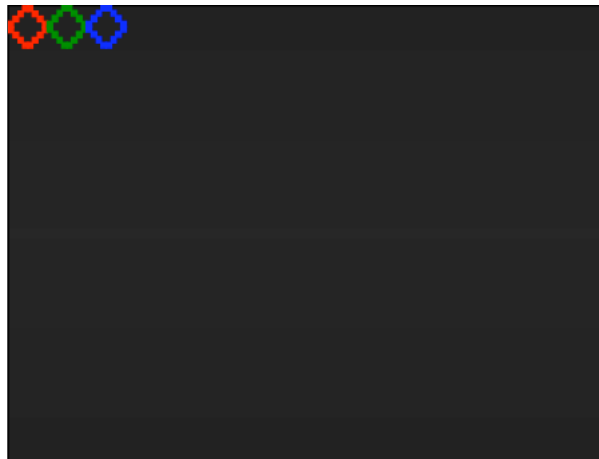
Display 8x8 bitmap character number 1 at x = 0, y = 0, colour = red

Example 2: 44hex, 01hex, 08hex, 00hex, 07hex, E0hex

Display 8x8 bitmap character number 1 at x = 8, y = 0, colour = green

Example 3: 44hex, 01hex, 10hex, 00hex, 00hex, 1Fhex

Display 8x8 bitmap character number 1 at x = 16, y = 0, colour = blue





2.1.13 paint Area (**p**) NOTE: Lower Case “p”

Syntax : cmd, x1, y1, x2, y2, colour(msb), colour(lsb)

cmd : 70hex, pascii

x1 : top left horizontal start position of the Area. Refer to Section 3.1 for ranges for x for different cores.

x1 : top left horizontal start position of the Area. 0 to 127 (00hex to 7Fhex).

y1 : top left vertical start position of the Area. 0dec to 127 (00hex to 7Fhex).

x2 : bottom right horizontal end position of the Area. 0 to 127 (00hex to 7Fhex).

y2 : bottom right vertical end position of the Area. 0 to 127 (00hex to 7Fhex).

colour : Area colour value: 2 bytes (16 bits) msb, lsb
65,536 colours to choose from

Black = 0000hex, 0dec

White = FFFFhex, 65,535dec, 1111111111111111bin

Description : This command will paint a specified area on the screen. **x1, y1** refers to the top left corner of the area and **x2, y2** refers to the bottom right hand corner of the area on the screen. If colour is chosen to be that of the background then the effect will be erasure.

Example : 70hex, 00hex, 00hex, 10hex, 10hex, 00hex, 00hex

Paint the area BLACK that has its top left corner at x1=0, y1=0 and its bottom RIGHT corner at x2=16, y2=16.



2.2 μ LCD Serial Interface (TTL)

The μ LCD needs to be connected via a serial link to a host system. The host uses this serial link to send commands to the μ LCD so that characters and graphics can be displayed on the screen. Use the signal pin-outs as well as the application example shown in the following section for correct connection to the host.

Please note that the serial connection (Rx/Tx) is at TTL levels (0 – 3.3V) and the logic levels are “high” = 1 = 3.3V, “low” = 0 = 0V. If interfacing to a host system running at 5V levels, then 1K series resistors must be inserted between the Host Tx/Rx and the μ LCD Rx/Tx signals.

Auto Baud Detect:

As previously mentioned, the μ LCD core has an auto-baud detect function which can operate from **300 baud to 128000 baud**. Prior to any graphical formatting and commands being sent to the core, it must first be initialised by sending the ASCII character ‘U’ (**55h**) after power-up. This will allow the core to determine and lock on to the baud rate of the host automatically without needing any further setup. This must be done every time the core is powered up.

If the host needs to change the baud rate, the μ LCD must be powered down and powered back up again. The “U” command cannot be used to change the baud rate during the middle of normal usage.

Serial Timing:

Each μ LCD command is made up of a sequence of data bytes. Some commands are a single byte and others are multiple bytes. When a command is sent to the μ LCD and the operation is completed, the μ LCD will reply back with a single acknowledge byte called the **ACK** (06h). This tells the host that the command was understood and the operation is completed. It will take the μ LCD anywhere between 1 to several milliseconds to reply back with an **ACK**, depending on the command and the operation the μ LCD has to perform. If the μ LCD receives a command that it does not understand it will reply back with a negative acknowledge called the **NAK** (15h).

For example, if a command has 5 bytes but only 4 bytes are sent, the command will not be executed and when the next following command bytes are sent the μ LCD will reply back with a **NAK** for each and every byte it receives. For correct operation make sure the command bytes are sent in the correct sequence.

Note: No termination character is to be sent at the end of the command sequence. i.e. don’t send any CR, or Null, or any other end of command bytes.



4D Systems

MicroLCD

2.3 μ LCD USB Interface

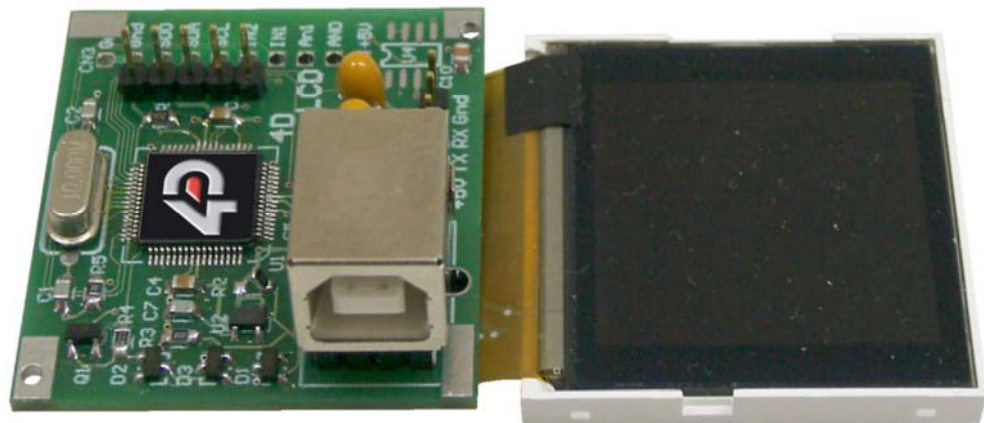
Using a standard USB interface on a computer, you can control the μ LCD-USB device by sending commands to it using a program to send hex data to the μ LCD-USB. The USB device on the μ LCD-USB, simply captures the USB data and converts it into serial TTL data for the μ LCD. USB Drivers are available from:

the 4D website:

www.4dsystems.com.au

or from the Dontronics website:

<http://www.dontronics.com/micro-usb.html>





3. Specifications

The μ LCD has the following electrical specifications which must be adhered to at all times to prevent damage to the device. The μ LCD module footprint is 38mm x 38mm.

Symbol	Characteristic	Min	Typ	Max	Units
Vdd	Supply voltage	4.5V	5V	5.5V	V
I	Current	100mA	120mA	140mA	mA
Deg C	Operating temp	0	30	70	C
Tpu	Power-up delay	800		1000	mS
Vtx	Serial Tx Voltage	2.7V	3.0V	3.3V	V
Vrx	Serial Rx Voltage	2.7V	3.0V	3.6V	V

3.1 μ LCD Host Interface in-outs

Pin outs for μ LCD

Pin 1 =	+5V
Pin 2 =	Tx
Pin 3 =	Rx
Pin 4 =	GND



3.2 65,536 Colour Bitmap Organisation

The μ LCD 65K colour byte is organised as 5 bits for Red(D11, D12, D13, D14, D15), 6 bits for Green(D5, D6, D7, D8, D9, D10) and 5 bits for Blue(D0, D1, D2, D3, D4). This will give a combination of $32 \times 64 \times 32 = 65,536$ colours. Each colour is not limited to 4 shades. For example a lighter shade of Red can be obtained by adding a little bit of the Green and a little bit of the Blue. Full Red and full Green will result in Yellow. Some experimentation will be needed to obtain the desired colour.

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R	R	R	R	R	G	G	G	G	G	G	B	B	B	B	B
E	E	E	E	E	R	R	R	R	R	R	L	L	L	L	L
D	D	D	D	D	E	E	E	E	E	E	U	U	U	U	U
					N	N	N	N	N	N	E	E	E	E	E
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Example: To Obtain the Colour Yellow

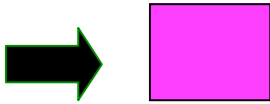
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R	R	R	R	R	G	G	G	G	G	G	B	B	B	B	B
E	E	E	E	E	R	R	R	R	R	R	L	L	L	L	L
D	D	D	D	D	E	E	E	E	E	E	U	U	U	U	U
					N	N	N	N	N	N	E	E	E	E	E
1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0





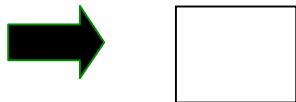
Example: To Obtain the Colour Magenta

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R	R	R	R	R	G	G	G	G	G	G	B	B	B	B	B
E	E	E	E	E	R	R	R	R	R	R	L	L	L	L	L
D	D	D	D	D	E	E	E	E	E	E	U	U	U	U	U
					N	N	N	N	N	N	E	E	E	E	E
					0	0	0	0	0	0	1	1	1	1	1



Example: To Obtain the Colour White

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R	R	R	R	R	G	G	G	G	G	G	B	B	B	B	B
E	E	E	E	E	R	R	R	R	R	R	L	L	L	L	L
D	D	D	D	D	E	E	E	E	E	E	U	U	U	U	U
					N	N	N	N	N	N	E	E	E	E	E
					1	1	1	1	1	1	1	1	1	1	1





3.3 Power-Up Reset

When the μ LCD comes out of a power up reset it initialises the video RAM and the internal Display registers. Allow up to 800ms to 1000ms before attempting to communicate with the μ LCD. The power up sequence of events should be as follows:

- Allow 800ms to 1000ms after power-up for μ LCD to settle. Do not attempt to communicate with the μ LCD during this period. The μ LCD may send garbage on its Tx Data line during this period, the host should disable its Rx Data reception.
- The host transmits the ASCII 'U' (capital U, 55hex) as the first command so the μ LCD can lock onto the hosts serial baud rate. The μ LCD will respond with an 'ACK' (06h). See section 2.3
- The μ LCD is now ready to accept screen function commands from the host.

