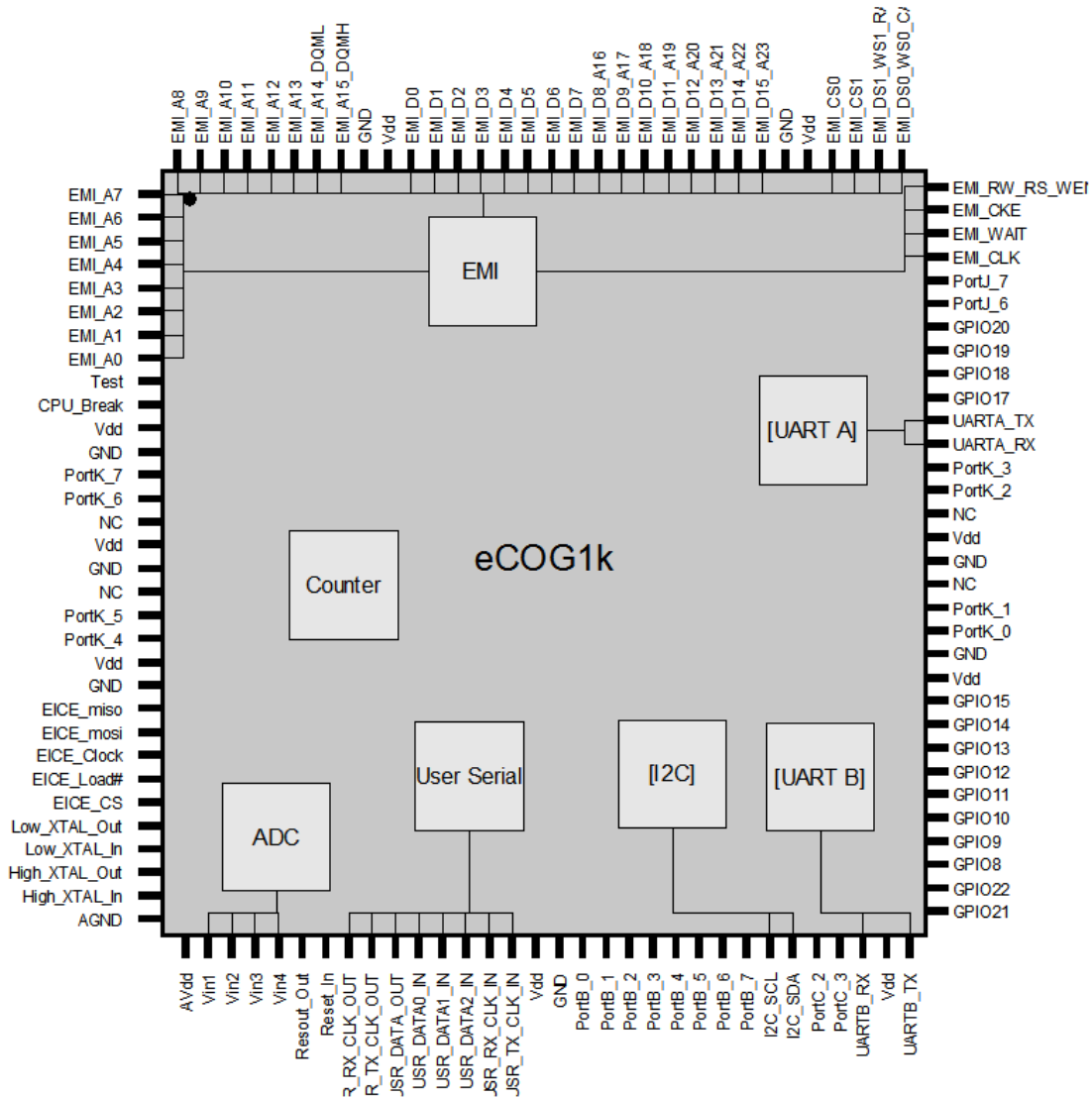




AN069 – Alphanumeric LCD Library

Version 1.0

This application note describes a simple library for driving alphanumeric LCDs that use an HD44780 compatible parallel interface.



Confidential and Proprietary Information

©Cyan Technology Ltd, 2008

This document contains confidential and proprietary information of Cyan Technology Ltd and is protected by copyright laws. Its receipt or possession does not convey any rights to reproduce, manufacture, use or sell anything based on information contained within this document.

Cyan Technology™, the Cyan Technology logo and Max-eICE™ are trademarks of Cyan Holdings Ltd. CyanIDE® and eCOG® are registered trademarks of Cyan Holdings Ltd. Cyan Technology Ltd recognises other brand and product names as trademarks or registered trademarks of their respective holders.

Any product described in this document is subject to continuous developments and improvements. All particulars of the product and its use contained in this document are given by Cyan Technology Ltd in good faith. However, all warranties implied or expressed, including but not limited to implied warranties of merchantability, or fitness for purpose, are excluded.

This document is intended only to assist the reader in the use of the product. Cyan Technology Ltd shall not be liable for any loss or damage arising from the use of any information in this guide, any error or omission in such information, or any incorrect use of the product.

This product is not designed or intended to be used for on-line control of aircraft, aircraft navigation or communications systems or in air traffic control applications or in the design, construction, operation or maintenance of any nuclear facility, or for any medical use related to either life support equipment or any other life-critical application. Cyan Technology Ltd specifically disclaims any express or implied warranty of fitness for any or all of such uses. Ask your sales representative for details.



Revision History

Version	Date	Notes
V1.0	06/11/2007	First release.

Contents

1	Introduction	5
2	Glossary	5
3	Requirements	5
4	Software Installation	5
5	Using the Library	6
5.1	Call Backs.....	6
5.2	eCOG1k Development Board (Issue 2).....	6
5.3	eCOG1X Development Board (Revision B).....	6
Appendix A	API Functions.....	7
	<i>lcd_rst</i>	7
	<i>lcd_clr</i>	7
	<i>lcd_xy</i>	7
	<i>lcd_putc</i>	7
	<i>lcd_puts</i>	7
	<i>lcd_cursor</i>	8
	<i>lcd_char_def</i>	8

1 Introduction

This application note describes a simple library to control alphanumeric LCDs with controllers based on the HD44780 parallel interface.

The library controls the LCD using seven GPIO lines. Four are required for the data bus and three for the control signals.

The library is suitable for use with either the eCOG1k or eCOG1X microcontrollers, and example code for the use of either is supplied with this application note.

2 Glossary

eCOG1	Cyan Technology target micro controller
LCD	Liquid Crystal Display

3 Requirements

- PC running eCOG1 CyanIDE V1.4.2
- An eCOG1k development board (Issue 2) or an eCOG1X development board (Revision B).

A zip archive file named **AN069LIB.zip** contains the common LCD library code for this application note. A second zip archive file named **AN069SW.zip** contains the source code for use with this application note as a pair of example projects, one for the eCOG1k and one for the eCOG1X. These two zip files are available for download from the Cyan Technology website www.cyantechology.com.

4 Software Installation

Extract the library software file **AN069LIB.zip** to the CyanIDE install directory, usually `<C:\Program Files\Cyan Technology\CyanIDE>`. This creates a new library directory *LCD* alongside the existing eCOG1k and eCOG1X library directories.

Extract the example software file **AN069SW.zip** to any convenient location, such as the CyanIDE projects directory `<C:\Documents and Settings\<username>\My Documents\CyanIDE Projects>`. This creates a new directory AN069SW with two further subdirectories. The two directories contain the example projects for the eCOG1k development board V2.1 and for the eCOG1X development board rev B.

5 Using the Library

To use the LCD library with CyanIDE, add the library path to both the compiler include and linker library search path lists. See the CyanIDE user manual for instructions on adding additional libraries to projects.

The library functions may then be called by including the following statement at the top of any C source file that needs them:

```
#include <lcd_lib.h>
```

5.1 Call Backs

The library uses one call back that must be supplied by the applications. This is the function:

```
void delay_ms(unsigned int count);
```

This should cause a delay of `count` milliseconds before returning, and is used only by the `lcd_rst()` function.

In the example code Counter 1 is used to generate a 1mS delay, that is then repeated for the count number of times.

5.2 eCOG1k Development Board (Issue 2)

The `<eCOG1k LCD>` project drives the LCD on the Development Board. The LCD module is powered from a 5V supply. Note that the LCD data signals are bidirectional and are driven to 5V by the LCD module during read cycles. This means that a buffer or level shift device is required between the eCOG1k and the LCD module. U9 (74LCX245) performs this function on the Development Board.

5.3 eCOG1X Development Board (Revision B)

The eCOG1X Development board does not have an alphanumeric LCD on it, and one must be connected to the board to use this application software. The easiest point to access suitable connections is to remove the multiplexed LCD and use the "Camera Interface" header J11.

Port P is 5V tolerant, so the LCD interface can be directly connected to these pins without the need for a buffer as required by the eCOG1k.

The `<eCOG1X LCD>` project drives an LCD Connected to J11 thus:

J11	eCOG1X Port	LCD Connection
Pin 5	Port P_0	LCD Data D4
Pin 6	Port P_1	LCD Data D5
Pin 7	Port P_2	LCD Data D6
Pin 8	Port P_3	LCD Data D7
Pin 9	Port P_4	LCD Control RS
Pin 10	Port P_5	LCD Control R/W
Pin 11	Port P_6	LCD Control E

Appendix A API Functions

lcd_rst

```
#include <lcd_lib.h>
void lcd_rst(unsigned int gpio_d4, unsigned int gpio_d5,
             unsigned int gpio_d6, unsigned int gpio_d7,
             unsigned int gpio_rs, unsigned int gpio_rnw,
             unsigned int gpio_e);
```

This function initialises the LCD. It must be supplied with the GPIO allocations for the data and control lines. This should be called before all other library functions.

lcd_clr

```
#include <lcd_lib.h>
void lcd_clr( void );
```

This function clears the LCD and returns the cursor to the upper left corner of the LCD.

lcd_xy

```
#include <lcd_lib.h>
void lcd_xy(unsigned int x, unsigned int y);
```

This function moves the cursor to the specified character position in the LCD memory. The upper left corner of the LCD is the origin at location (1,1).

lcd_putc

```
#include <lcd_lib.h>
void lcd_putc(unsigned int c);
```

This function places the specified character at the current cursor location and increments the cursor position.

lcd_puts

```
#include <lcd_lib.h>
int lcd_puts(const char *s);
```

This function displays the specified character string, starting at the current cursor location. The function returns the number of characters displayed.

lcd_cursor

```
#include <lcd_lib.h>
void lcd_cursor(unsigned int mode);
```

This function changes the LCD cursor type. The modes available are:

mode	Description
LCD_CURSOR_OFF	No cursor is displayed.
LCD_CURSOR_ON	An underline cursor is displayed.
LCD_CURSOR_BLINK	A blinking block cursor is displayed.

lcd_char_def

```
#include <lcd_lib.h>
void lcd_char_def(unsigned int id, const char *data);
```

This function defines one of the eight user defined characters available in the LCD module.

The *id* parameter specifies which character is to be defined, in the range 0 to 7.

The data is an array of eight bytes that define the map of the pixels in the character. The first data byte defines the top row, and the data works down from there.

The five least significant bits of each byte define the individual pixels.

In the example code, three characters are defined to generate the jumping man animation used. The first of these is the man in the standing position, and this is defined thus:

0x10	0x08	0x04	0x02	0x01	Data
	■	■	■		0x0E
	■	□	■		0x0A
	■	■	■		0x0E
	□	■	□		0x04
	■	■	■		0x0E
■	□	■	□	■	0x15
□	■	□	■	□	0x0A
■	□	□	□	■	0x11

Once the character is defined, the character can be displayed by using the specified character code between 0x00 and 0x07.