

LCD Driver for the HC08/HCS08 Family

By **Eduardo Estrada**
Diego Garay
Jaime Herrero
RTAC Americas, Mexico

Overview

This document is a quick reference for an embedded engineer to get a LCD driver for any HC08 or HCS08 MCU up and running. Basic knowledge about the functional description will give the user a better understanding of how the LCD driver works. This application note provides an example that demonstrates the use of the LCD driver. The example may be modified to suit the specific needs of any application.

The code was written for a LCD module that already had a hardware LCD driver compatible with the HD44780 LCD driver, providing the possibility to directly drive the module with a parallel interface.

This application note includes a step-by-step guide to configure the LCD driver features. The example code described in this document is provided in its entirety as zip file, AN2940SW.zip; which can be downloaded from www.freescale.com.

LCD General Operation

Most LCDs have the interface shown in [Figure 1](#).

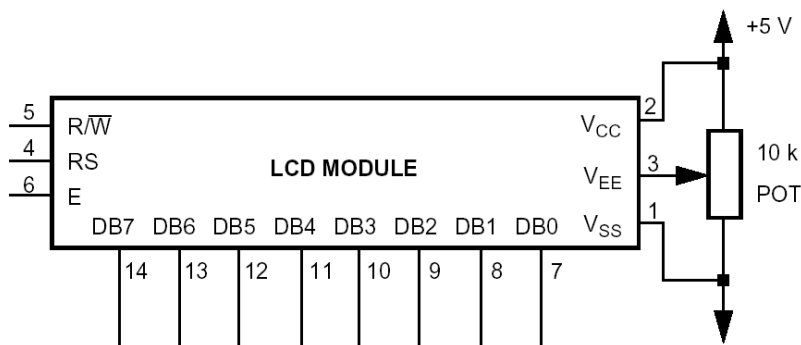


Figure 1. LCD Module Interface

Hardware Interface Description

Table 1. LCD Pin Descriptions

Pin	Description
$\overline{R/W}$	Select read or write: 1 = Read 0 = Write
RS	Select data or instruction: 1 = Data 0 = Instruction
E	Starts data read/write on falling edge
DB0—DB3 ⁽¹⁾	Four high order data bus pins (not used in 4-bit mode)
DB4—DB7 ⁽¹⁾	Four low order data bus pins
V_{SS}	Ground
V_{EE}	Contrast voltage
V_{CC}	4.5 V to 5.0 V

Note 1: Used to send and receive information between the LCD and the MCU.

Software Configuration:

The LCD module has some instructions to customize the display:

Table 2. LCD Module Instructions

Instruction	Description																				
<div>Clear Display</div> <table><tr><td>RS</td><td>R/W</td><td>DB7</td><td>DB6</td><td>DB5</td><td>DB4</td><td>DB3</td><td>DB2</td><td>DB1</td><td>DB0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr></table> <div>(1) Execution Time: 1.64 ms</div>	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	0	0	0	0	0	0	0	0	0	1	<div>The display disappears and the cursor goes to the left edge of the first line of the display.</div>
RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0												
0	0	0	0	0	0	0	0	0	1												
<div>Entry Mode Set</div> <table><tr><td>RS</td><td>R/W</td><td>DB7</td><td>DB6</td><td>DB5</td><td>DB4</td><td>DB3</td><td>DB2</td><td>DB1</td><td>DB0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>I/D</td><td>SH</td></tr></table> <div>(1) Execution Time: 40 μs</div>	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	0	0	0	0	0	0	0	1	I/D	SH	<div>I/D</div> <div>1 = The cursor moves to the right</div> <div>0 = The cursor moves to the left</div> <div>SH</div> <div>1 = Shifts the entire display to the left</div> <div>0 = Shifts the entire display to the right</div>
RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0												
0	0	0	0	0	0	0	1	I/D	SH												
<div>Display On/Off Control</div> <table><tr><td>RS</td><td>R/W</td><td>DB7</td><td>DB6</td><td>DB5</td><td>DB4</td><td>DB3</td><td>DB2</td><td>DB1</td><td>DB0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>D</td><td>C</td><td>B</td></tr></table> <div>(1) Execution Time: 40 μs</div>	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	0	0	0	0	0	0	1	D	C	B	<div>D</div> <div>1 = Display on</div> <div>0 = Display off</div> <div>C</div> <div>1 = The cursor is displayed</div> <div>0 = The cursor is not displayed</div> <div>B</div> <div>1 = The cursor blinks</div> <div>0 = The cursor doesn't blink</div>
RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0												
0	0	0	0	0	0	1	D	C	B												
<div>Function Set</div> <table><tr><td>RS</td><td>R/W</td><td>DB7</td><td>DB6</td><td>DB5</td><td>DB4</td><td>DB3</td><td>DB2</td><td>DB1</td><td>DB0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>DL</td><td>N</td><td>F</td><td>X</td><td>X</td></tr></table> <div>(1) Execution Time: 40 μs</div>	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	0	0	0	0	1	DL	N	F	X	X	<div>DL</div> <div>1 = Data send or receive in 8-bit length</div> <div>0 = Data send or receive in 4-bit length</div> <div>(DB4–DB7 used, the data must be sent twice, first the most significant nibble, next the least significant nibble)</div> <div>N</div> <div>1 = 2 lines</div> <div>0 = 1 line</div> <div>F</div> <div>1 = 5x10 dots character font</div> <div>0 = 5x11 dots character font</div>
RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0												
0	0	0	0	1	DL	N	F	X	X												

NOTES:

1. The execution time is the time delay needed to configure each instruction.

Example Description

The following example describes how to configure the LCD driver, using:

- MC68HC908AP64 MCU
- External clock source with a frequency equal to 9.8304 MHz
- Delay base time of 100 μ s
- LCD settings are: display on, cursor off, blinking off, display 5x10 [character size], 2 lines, data 4-bit length mode
- Pin connection configured to control the LCD (see [Figure 2](#))

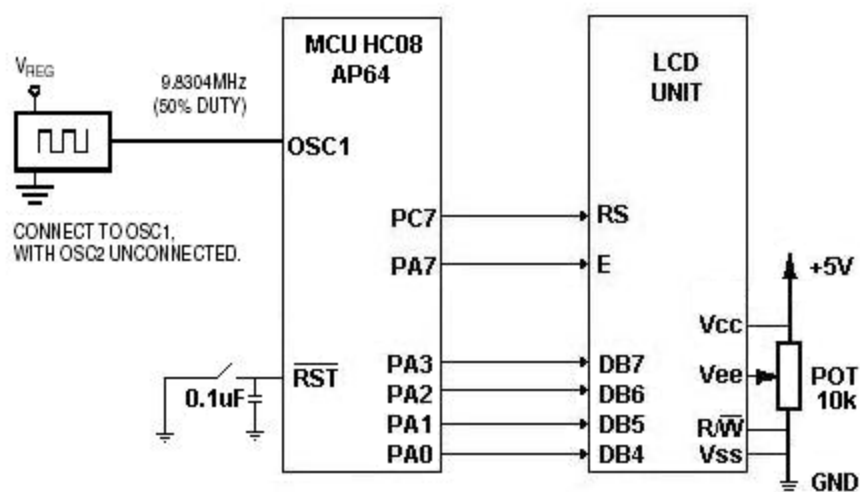


Figure 2. Pin-Out Control Connection

LCD Driver Description

It is necessary to understand the main content of each file from the LCD driver project. This section describes each file.

mcu_driver_select.h

This file allows the user to select the target MCU and contains the following information:

- Specific MCU header, which contains the peripheral declarations
- Definition of two constants:
 - "gTimeBaseInterrupteachus" is the number of microseconds required for each timer interrupt
 - "gTimeBaseInterruptperms" is the number of timer interrupts necessary to equal 1 ms
- Declaration to select the MCU family to be used
- Type definitions for the variables used in the LCD driver

Lcd.h

This file allows the user to select the pinout connection and contains the following information:

- Definition of the data direction and pinout ports, which control the LCD module as long as the constant “IcdExists” is declared
- Definition of flags needed for the correct functionality of the driver; this section must not be modified
- Function declaration (prototypes) of all the subroutines needed in the driver; this section also must not be modified

Lcd.c

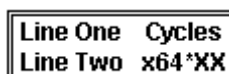
This file contains the functions needed for the LCD driver, as described in [Table 3](#).

Table 3. LCD Driver Functions

Subroutine Name	Type	Parameters	Functionality
LCD_Init	void	void	Initializes the LCD
LCD_Clear	void	void	Clears the LCD
LCD_2L	void	void	Sets the cursor in the second line of LCD
LCD_Print	void	uint8 *where, uint8 length	Prints a string of specified length
LCD_TimeBase	void	void	Controls the delays needed for internal control and configuration
LCD_Status	uint8 status	void	Indicates whether the LCD drive is busy or available
LCD_Cursor	void	uint8 ddramAddress	Sets the cursor in a specific place on the LCD

Main.c

This file contains the main application of the project. The included example will display the message shown in [Figure 3](#) on the LCD.



```

Line One Cycles
Line Two x64*XX

```

Figure 3. LCD Default Message

- The message shows the line number of each line and the number of cycles that can be used while the LCD driver works. “x64” is the time base, which takes each timer overflow interrupt as hexadecimal number and ‘XX’ is the number of times (cycles) the main function was executed since the last time the LCD was refreshed. This number of times is calculated with the following formula: **Cycles = 0x64*0xXX**
- The configuration of the source clock and the timer mode must not be modified.

Setting LCD Driver — Step-by-Step

This section describes how to configure the LCD driver. If you need to download the LCD_driver firmware, please see the [References](#) section for a link to the Lumex web site.

A) Setting the MCU

1. Open the project named LCD_driver.mcp, which is in the folder LCD_driver.
2. Search in the project window for the folder sources and open the mcu_driver_select.h (see [Figure 4](#)).

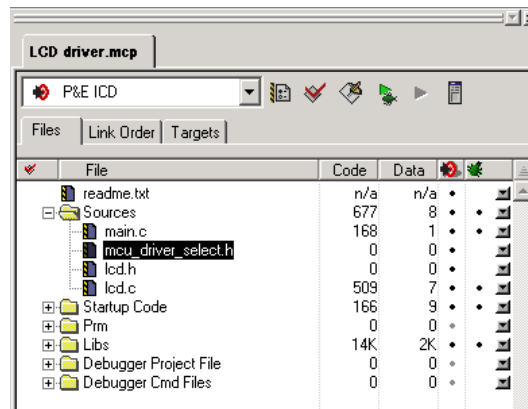


Figure 4. Sources Folder Location in the CodeWarrior Window for mcu_driver_select.h

In the first lines of this file, add the appropriate header of the MCU to be used in order to include the correct peripheral declarations.

```
#ifndef MC68HC908AP64_h
    #define MC68HC908AP64_h
    #include <MC68HC908AP64.h> /* Include peripheral declarations */
#endif
```

3. In the line 11 of the same file, select the family, commenting-out the appropriate line according the MCU family used.

```
/* Define the kind of the MCU */
#define MC908 /* In case of HC08 family */
// #define MCS08 /* In case of HCS08 family */
```

B) Setting the MCU Pin-Out to Control the LCD

Each MCU has a specific number of ports and GPIO pins, so it is necessary to choose which of them will control the LCD.

1. Open the file `lcd.h`, which is located in the sources folder. See [Figure 4](#).
2. Go to the line 6 and make sure that define “`lcdExists`” isn’t disabled. This feature allows us to use this driver with other applications and disable it when it is necessary.

```
#define lcdExists 1          /* If LCD does not exist, do not declare this define */
```

3. Go to the line 8 and specify which configuration must be used for the LCD driver.

NOTE

The MCU pins that control the data pins of the LCD must be consecutive. Configure the data length as 8 or 4 bits for the LCD.

```
#define lcd4bit 1           /* 4 bit interface; comment this line if is 8 bit
                             interface */
#define lcdE    PTD_PTD6    /* Enabled pin of LCD */
#define lcdEDD  DDRD_DDRD6  /* Data Direction of Enabled Pin */
#define lcdRS   PTD_PTD7    /* RS pin of LCD (Data/Instruction Select) */
#define lcdRSDD DDRD_DDRD7  /* Data Direction of RS Pin */
#define lcdPort PTA         /* Port of 4 data bits to lcd connection */
#define lcdPortDD DDRA      /* Data direction for 4 data pins */
#ifdef lcd4bit
#define lcdDataPins 0       /* Number of pin of the port where begin the data
                             pins (4 pins). These pins must be
                             consecutive. Only in case of 4 bit interface */
#endif
#endif
```

References

C) Define the Delay Time Base

In case you need to use a different crystal frequency, f_{OSC} , follow these steps to recalculate the time base.

1. Open main.c, which is in the sources folder (see [Figure 4](#)).
2. Set how much time each timer overflow, TOF, requires (in microseconds and less than to 1000). That value must be placed in the variable "gTimeBaseInterrupteatus", which is in the mcu_driver_select.h file. (See TOF, [Table 4](#))
3. Use the value from the T1MOD equation (see [Table 4](#)) in line 126 of main.c.

Table 4. Defining the Delay Time Base Example

Desired external clock source frequency	(f_{OSC})	=	9.8304 MHz
TIM clock prescaler	($TIM_{Prescaler}$)	=	1
Timer count equation ⁽¹⁾	(t_{Count})	=	$\frac{4 \times TIM_{Prescaler}}{f_{OSC}} = 0.4069 \mu s$
Time required for each overflow interrupt ⁽²⁾	(TOF)	=	100 μs
T1MOD equation:	$\frac{TOF_{Delay}}{t_{Count}} = \frac{100 \mu s}{0.4069 \mu s} = 245.76 \approx 246 = F6 \text{ hexadecimal}$		

NOTES:

1. This equation is correct for the HC08 Family; in case of a HCS08 application, the timer count equation must be multiplied by 2 instead of 4.
2. Define "gTimeBaseInterrupteatus" as the number of microseconds selected in the mcu_driver_select.h file.

References

- Download the LCD_driver project, AN2940SW.zip, from www.freescale.com
- Download the LCD data sheet from Lumex provider at <http://www.lumex.com>
- Download the last CodeWarrior version from <http://www.metrowerks.com>

Considerations

CodeWarrior

The user should get or download the newest version of the compiler CoderWarrior from Metrowerks (the LCD driver was made with CW08 V3.1). HC908 and HCS08 devices can be compiled with this software.

Timer Interface Module (HC08) and Timer PWM Module (HCS08)

The interrupt logic from the timer is already used in this implementation; so the user must consider that functions such as input capture, output compare and PWM generation, are limited or unavailable.

The user must consider that the timer base source in the example is specific to the MC68HC908AP64. Changes to the timer configuration may be required for use with a different MCU.

This page is intentionally blank.

This page is intentionally blank.

This page is intentionally blank.

How to Reach Us:

Home Page:

www.freescale.com

E-mail:

support@freescale.com

USA/Europe or Locations Not Listed:

Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:

Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:

Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:

Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:

Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals", must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.

© Freescale Semiconductor, Inc. 2005. All rights reserved.