



APPLICATION NOTE 3913

Using the LCD Simulator with MAX-IDE and IAR Embedded Workbench Development Environments

Abstract: This application note describes how to use the LCD simulator on the MAX-IDE and IAR Embedded Workbench, and explains how to create the LCD memory map and LCD display panel. This article assumes that the reader knows the MAXQ20 core, and is aware of the MAX-IDE and IAR Embedded Workbench. By the end of this application note the user will understand the workings of the LCD simulator in both development environments.

Introduction

The LCD simulator was developed on a Windows® platform with a GUI that simulates the digital behavior, but not the analog behavior, of the LCD controller. Readers can review application notes 3905, "[MAX-IDE Simulator User's Guide for the MAXQ Microcontrollers](#)," and 3378, "[Getting started with the IAR Compiler and the MAXQ2000 Evaluation Kit](#)" for information on working with MAX-IDE and IAR.

Overview

The LCD simulator simulates the digital properties of the LCD controller, including: static, 1/2 mux, 1/3 mux, and 1/4 mux display modes; enabling/disabling LCD operation; and updating the LCD display with the display memory patterns. The LCD simulator ignores analog property changes, including: LCD drive voltage changes, display contrast adjustment, frame frequency effect.

The LCD simulator needs two inputs:

1. LCD segment configuration for static, 1/2, 1/3 and 1/4 display modes
2. LCD pin configuration for static, 1/2, 1/3 and 1/4 display memory

Development Environment Settings for the LCD Simulator

The LCD simulator can be used on two environments, the MAX-IDE and the IAR Embedded Workbench.

1. MAX-IDE Settings

Some MAXQ® devices are configured and installed during the MAX-IDE installation. You can get the list of those devices from the **Device**→**Options** menu. Among these devices, the MAXQ2000 and MAXQ3210 have an LCD controller peripheral. Follow the steps below to create the project for simulation.

- a. Create a Project. This procedure is explained in application note 3905 (see above) on the MAX-IDE simulator.
- b. Add a file that tests the functionality of LCD controller. The file code is shown in **Table 1**.
- c. Select **Device**→**MAXQ2000**→**OK**.
- d. Under the Device menu, click on **LCD Simulator** and the LCD GUI will be displayed, as seen in **Figure 1**.

We are now ready to explore the LCD simulator.

2. IAR IDE Settings

Some MAXQ devices' DDF (Device Description File), SFR, and ROM files are installed under the \$TOOLKIT_DIR\$\config directory during IAR Embedded Workbench installation. Follow the steps below to create the project used for simulation.

- a. Create a project. This procedure is explained in the application note 3378 (see above) on the IAR compiler.
- b. Add a file that tests the functionality of the LCD controller. The code is shown in **Table 2**.
- c. Open **Project**→**Options** and go to the **C Spy Debugger** panel.
- d. Select **Device Simulator** from the options shown.
- e. Select the ddf file of the device to be simulated. In this example the file is maxq200x.ddf under \$TOOLKIT_DIR\$\config.
- f. If the program is assembly, uncheck the "Run to main" option and Check **XLINK**→**Include**→**Ignore C STARTUP** in the library.
- g. Select the utility ROM routine (.hex) of the device to be simulated, i.e., maxq200x.hex
- h. Press **OK**.
- i. Press **Debug** to debug the program.
- j. Peripherals simulated can be seen under the **View Menu** option.
- k. Select the LCD peripheral and the LCD GUI will be displayed as in Figure 1.

We are now ready to explore the LCD simulator.

The LCD GUI snapshot, along with the assembly files to be included with MAX-IDE and IAR, are listed below:

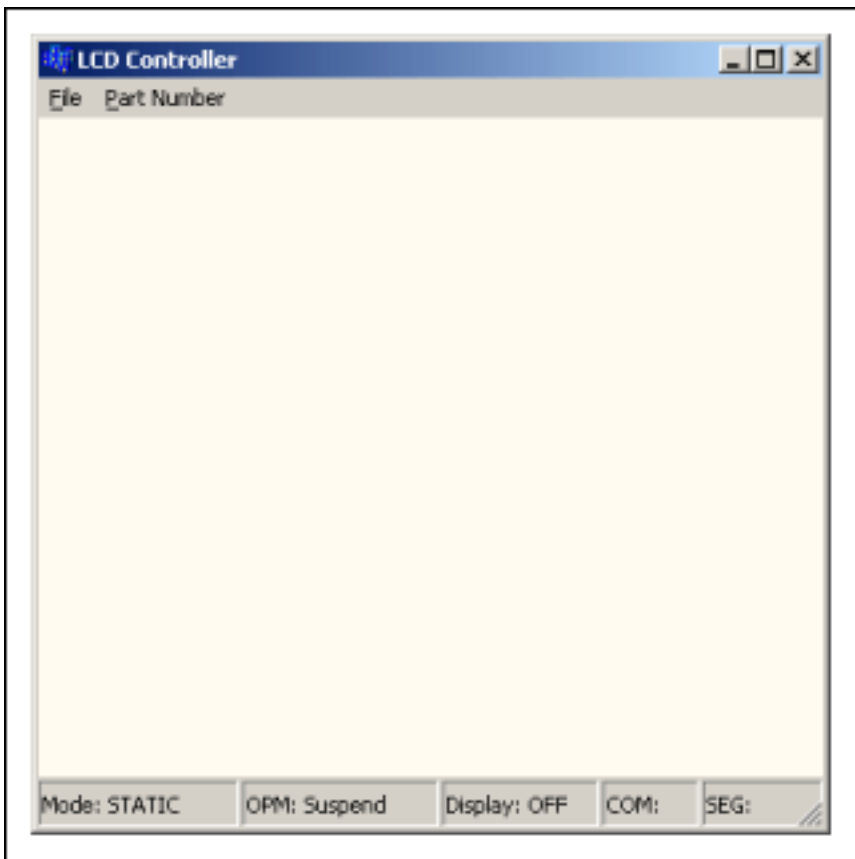


Figure 1. LCD simulator GUI.

Table 1. Max-Ide_Lcd2MuxMode.asm Simulates the LCD Working in 1/2 Mux Mode in the MAX-IDE Environment

```

;=====
; MaxQ20MS10XF (BG)
;=====
; LCD Simulator test

#include "maxq2000.inc"

SEGMENT CODE ; CODE segment starts here
Org 0 ; SET Absolute address of code = 0
jump main

Org 20h ; SET Absolute address of code = 32 HEX

_2MUXMODE:

move LCRA, #08FFh ; Select 1/2 MUX mode of display
move LCFG.0, #0h ; Disable LCD RAM Display
move LCFG.1, #01h ; Enable LCD Operation mode
move LCFG.4, #01h ; PCF0: Configure Pins as Segment pins
move LCFG.5, #01h ; PCF1
move LCFG.6, #01h ; PCF2
move LCFG.7, #01h ; PCF3

move EIE0.7, #01h ; INT4, Change pattern to check for all bits
; Fill the Display memory with data to be displayed
move LCDD0, #00Fh ; LCDD0
move LCDD1, #00Fh ; LCDD1
move LCDD2, #00Fh ; LCDD1
move LCDD3, #00Fh ; LCDD2
move LCDD4, #00Fh ; LCDD4
move LCDD5, #00Fh ; LCDD5
move LCDD6, #00Fh ; LCDD6
move LCDD7, #00Fh ; LCDD7
move LCDD8, #0FFh ; LCDD8, Make it 00 to draw in Black color

move LCFG.0, #1h ; Enable Display
; Display can be changed even when LCD is ON.
move LCDD0, #0FFh ; LCDD0
move LCDD1, #0FFh ; LCDD1

RET

main:
CALL _2MUXMODE ; Call the LCD 1/2 Mux Mode
SJUMP $ ; RUN forever

END

```

The above source code is available for [download](#).

Table 2. IAR_Lcd2MuxMode.asm Simulates the LCD Working in 1/2 Mux Mode in the IAR Embedded Workbench

```

=====
;
; MaxQ20MS10XF (BG)
;
=====
; LCD Simulator test

PROGRAM test_lcd_sim

#include "maxq20ms10xf.inc"

ASEGN W :CODE, 00
ORG 00
jump main

_2MUXMODE:

move LCRA, #08FFh ; Select 1/2 MUX mode of display
move LCFG.0, #0h ; Disable LCD RAM Display
move LCFG.1, #01h ; Enable LCD Operation mode
move LCFG.4, #01h ; PCF0: Configure Pins as Segment pins
move LCFG.5, #01h ; PCF1
move LCFG.6, #01h ; PCF2
move LCFG.7, #01h ; PCF3

move EIE0.7, #01h ; INT4, Change pattern to check for all bits
; Fill the Display memory with data to be displayed
move LCD0, #00Fh ; LCDD0
move LCD1, #00Fh ; LCDD1
move LCD2, #00Fh ; LCDD1
move LCD3, #00Fh ; LCDD2
move LCD4, #00Fh ; LCDD4
move LCD5, #00Fh ; LCDD5
move LCD6, #00Fh ; LCDD6
move LCD7, #00Fh ; LCDD7
move LCD8, #0FFh ; LCDD8, Make it 00 to draw in Black color

move LCFG.0, #1h ; Enable Display
; Display can be changed even when LCD is ON.
move LCD0, #0FFh ; LCDD0
move LCD1, #0FFh ; LCDD1

RET

main:
CALL _2MUXMODE ; Call the LCD 1/2 Mux Mode
Loop:
SJUMP Loop ; RUN forever

END

```

The above source code is available for [download](#).

Working with the LCD Simulator

As mentioned above, the LCD simulator depends on two XML input files:

1. Segment Configuration File
2. Pin Configuration File

1. Segment Configuration File

This file defines the type of the LCD display panel as either 7-segment or alphanumeric. The user can design the LCD display type by editing the XML file. The file details each segment as line or dot or any other shape (all multiples of a line) for static, 1/2 mux, 1/3 mux, and 1/4 mux. Without this input file, the LCD simulator will not update the GUI.

2. Pin Configuration File

Each MAXQ controller with a LCD peripheral comes in different pin configuration that assigns a different pin number for static, 1/2 mux, 1/3 mux, and 1/4 mux. Also some segment pins are multiplexed with two different functions:

- a. I/O function
- b. External interrupt, with I/O having the low priority and external interrupt having the higher priority.

The pin-configuration file supplies the controller pin-package details, SFR display memory-map details, and the multiplex functionality with an I/O and interrupt event for each display mode and for each package type. Without these inputs the LCD simulator cannot correctly simulate the data in the LCDDx register that may, or may not be sourcing the segment data on the pin.

On selecting the MAXQ2000 device, these two files will be loaded automatically into the MAX-IDE environment. Alternatively, you can supply these files in the `maxq200x.sfr` file in IAR under **[LCDConfigFiles]** option as follows:

[LCDConfigFiles]

ConfigFile = config\lcd_config.xml

PinConfigFile = config\lcd_pin_config.xml

The user can override these files using the **File**→**Open** option of the GUI.

LCD Registers and Their Simulation

1. LCD Register Adjust (LCRA): Selects one of the display modes: static, 1/2 mux, 1/3 mux, and 1/4 mux. Changes to this register will be reflected on the status bar of the LCD GUI.
2. LCD Configuration Register (LCFG): Each bit setting and the GUI status is described below:

Bit Settings	GUI Changes
DPE = 1	Display: ON
DPE = 0	Display: OFF
OPM = 1	OPM: Normal
OPM = 0	OPM: Suspend

3. LCDDx Registers: LCDDx registers can be updated when DPE = 0/1. Contents of LCDDx registers will be updated on the GUI when DPE = 1 and OPM = 1. Red will represent LCDs that are sourced; black represents LCDs that are present, but currently not sourced. LCDs not sourced can result from several factors:

- a. The PCF_x bit of the LCFG registers is not SET, making the SEG pins work as I/O pins.
- b. The LCD SEG pin that is multiplexed with an interrupt functionality is enabled.
- c. The LCDD_x register contains 0 in that bit position.
- d. Moving the PC mouse on the display updates the COM and SEG.

GUI Changes for the Example Program

MAXQ2000 comes with three different pinout packages: 56, 68, and 100 pins. The test program (see Tables 1 and 2 above) tests the functionality of the MAXQ2000 68-pin package. Select the MAXQ2000 68-pin configuration from the part number list, and execute the test program. Note the GUI changes.

```

move LCRA,          #08FFh      : Mode: 1/2 Mux
move LCFG.0,        #0h         : Display: OFF
move LCFG.1,        #01h        : OPM: Normal
move LCFG.4,        #01h        : PCF0: Configures I/O pins as segment pins
move LCFG.5,        #01h        : PCF1: Configures I/O pins as segment pins
move LCFG.6,        #01h        : PCF2: Configures I/O pins as segment pins
move LCFG.7,        #01h        : PCF3: Configures I/O pins as segment pins

move EIE0.7,       #01h        : Enables Interrupt Function of INT7, Comment
                                : /uncomment to see the MUX behavior of SEG pins

```

The display pattern is moved from the LCDD0 register to the LCDD8 register. A 1 represents the segments to be sourced, and a 0 represents segments that will not be sourced.

```

move LCFG.0,       #1h         : Display: ON

```

With EIE0.7 disabled (comment **move** EIE0.7, #01h) and after executing the "**move** LCFG.0, #1" statement, the GUI display will look like **Figure 3**.

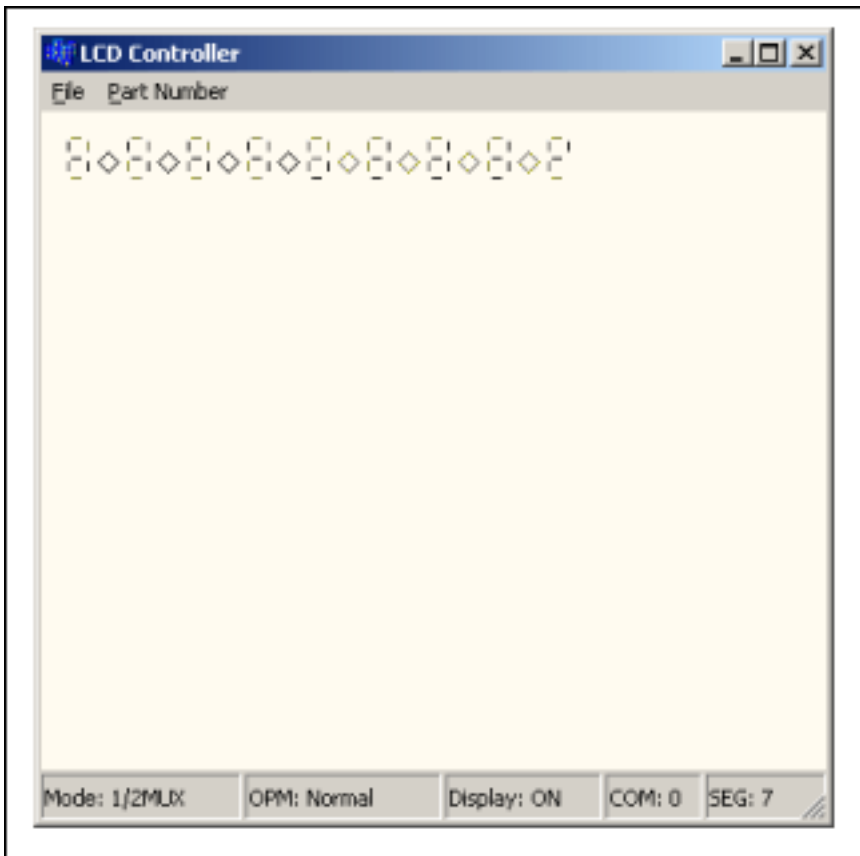


Figure 2. MAXQ2000 64-pin LCD panel.

After executing

```
move LCD0, #0ffh  
move LCD1, #0ffh
```

The GUI will look like:

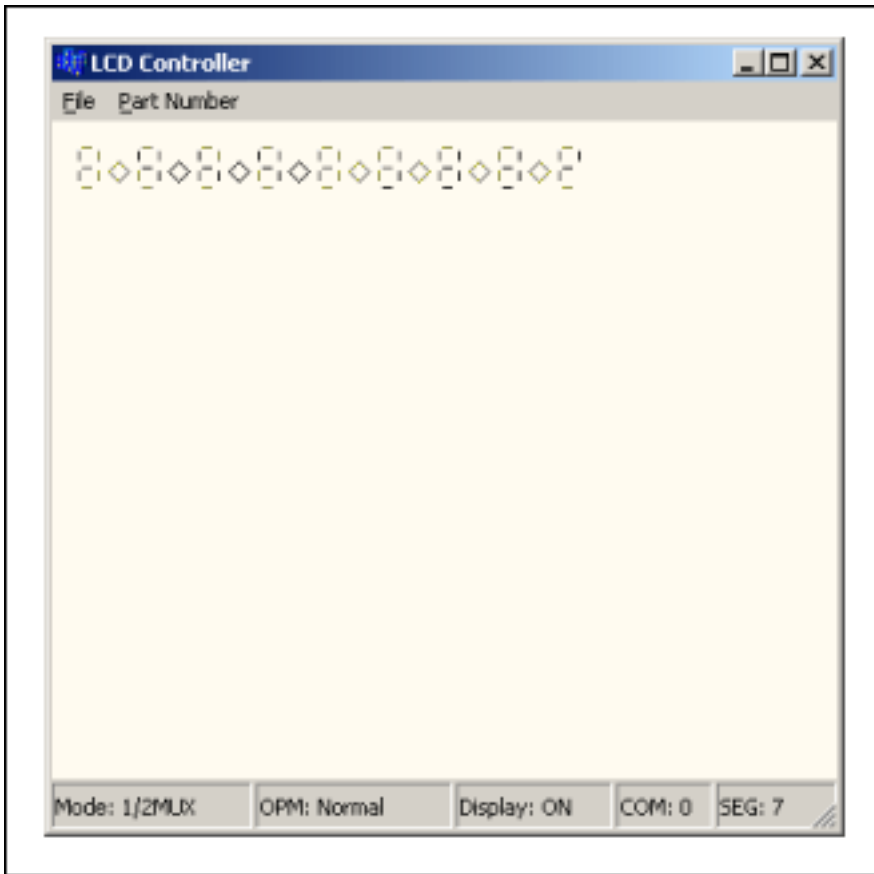


Figure 3. MAXQ2000 64-pin LCD panel display changes with $DPE = 1$.

The difference, you will notice, is that COM0:SEG2, COM0:SEG3, COM0:SEG6, and COM0:SEG7 change color from black to red which indicates that the LCDs are sourced.

With EIE0.7 enabled (SEG31 multiplexed with INT7), the GUI will look like:

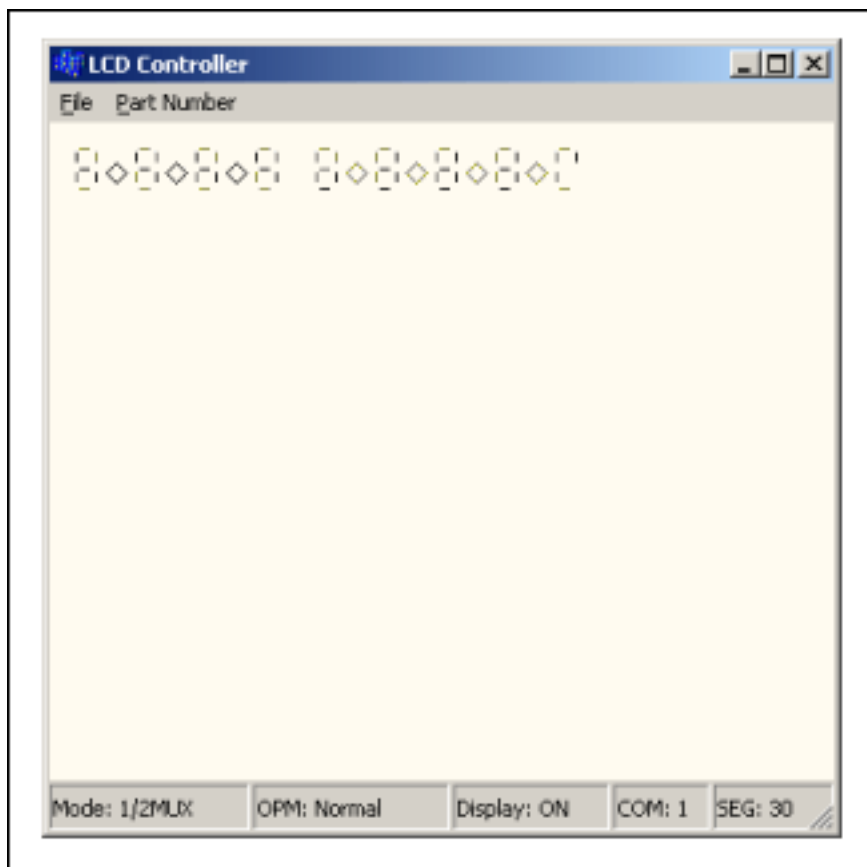


Figure 4. MAXQ2000 64-pin LCD panel with SEG multiplexed with INT.

Compare **Figure 2** with **Figure 4**. COM0:SEG31 and COM1:SEG31 are not sourced as LCD segments.

Note: The User can design/modify the LCD display panel and MAXQ pinout input of the LCD simulator. The XML input file can be opened in any editor and XML tag names used are self-explanatory.

Conclusion

The MAXQ device simulator can be used to develop and debug the LCD simulator for MAXQ10 and MAXQ20 microcontrollers. The application developed is then ready to run on the hardware.

MAXQ is a registered trademark of Maxim Integrated Products, Inc.

Application Note 3913: www.maxim-ic.com/an3913

More Information

For technical support: www.maxim-ic.com/support

For samples: www.maxim-ic.com/samples

Other questions and comments: www.maxim-ic.com/contact

Automatic Updates

Would you like to be automatically notified when new application notes are published in your areas of interest?
[Sign up for EE-Mail.](#)

