ESE381 Embedded Microprocessor Systems Design II

Spring 2024, K. Short
revised March 24, 2024 4:11 pm

Laboratory 08: AVR128DB48 C Driver for DOGM163W-A LCD

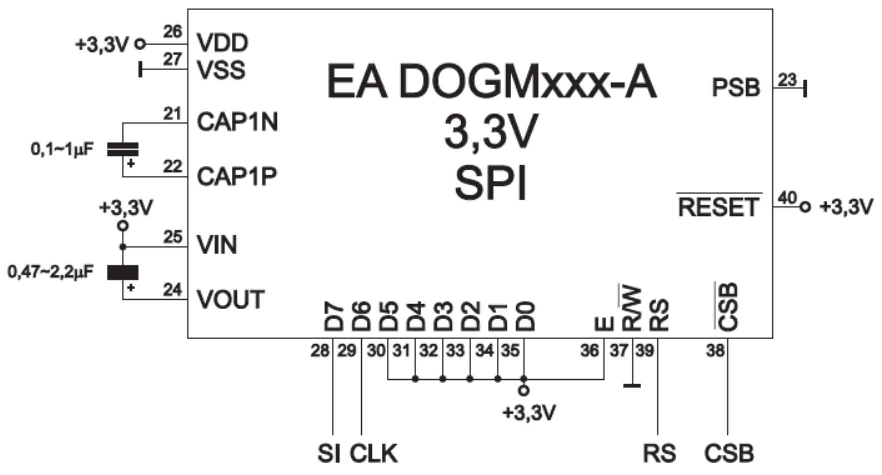
To be performed during the week beginning March 24th.

Prerequisite Reading
1. EA DOG- M Data Sheet.
2.Sitronix ST7036 Dot Matrix Controller Driver Data Sheet.
3. `C_driver_s20_SAML21J18B_main.c`
4. `lcd_dog_asm_driver_m324a.inc`.
5. `dog_module_test.asm`.

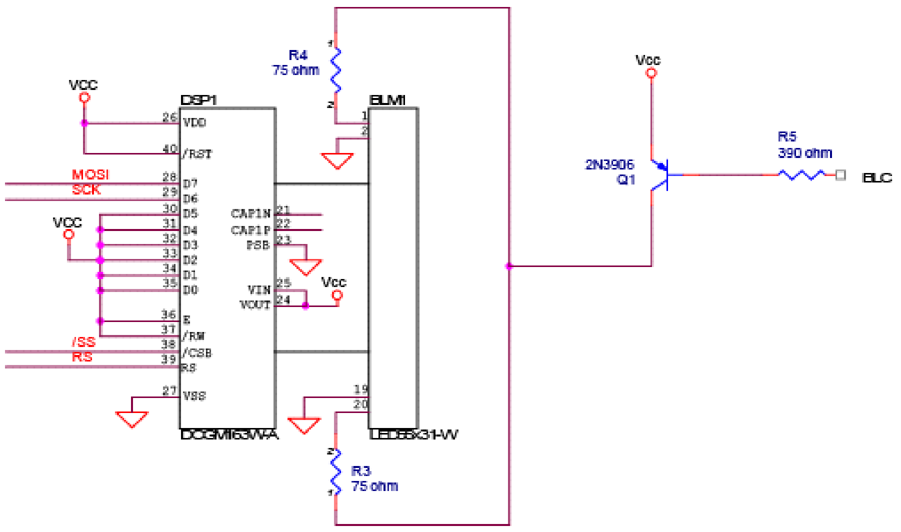
Overview

Liquid Crystal Displays (LCDs) provide a low cost alphanumeric display capability. LCDs are available as complete modules that accept ASCII representations of the data to be displayed. Many of these modules use similar display controller ICs that support a 4/8-bit parallel interface (e.g. HD44780). More recent modules use controllers (e.g. ST7036) that support your choice of a 4/8-bit parallel interface or a SPI serial interface. The result is that the interface and operation of many different LCD modules are similar because of the similarity of their hardware controllers.

The display module you will use is an Electronic Assembly (EA) DOGM163W-A. This is a 3-line, 16-characters per line display. This display module uses a Sitronix ST7036 Dot Matrix Controller Driver IC. The DOGM163W-A module can be operated at a supply voltage of either 3.3 V or 5.0 V. We will be using 3.3V from the Curiosity board. Connections to the module when operated at 3.3V and using its SPI are shown in the following block diagram from the DOG Series data sheet.



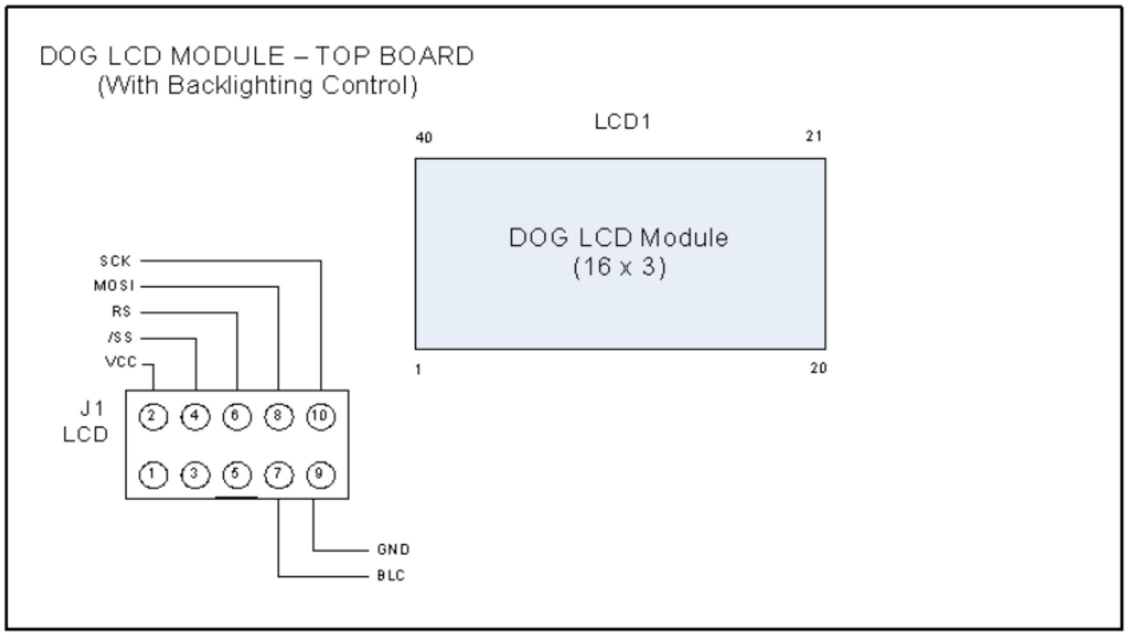
For higher contrast, the DOGM163W-A can be used with a backlight. LED backlights of various colors are available. You will use a white backlight (EA LED55X31-W). You will be provided a prewired display that uses the DOGM163W-A LCD Module and an EA LED55X31-W LED backlight. A schematic of the circuit on the prewired display board is shown below.



In the block diagram you see the SPI bus connections on the left. The PNP bipolar transistor on the right controls the intensity of the backlight. If connection BLC is at 0V, the transistor is fully ON and the backlight will be at its highest intensity. As the voltage at BLC is increased, the back-light output is reduced. This voltage could be controlled in a system to allow the user to adjust the backlight intensity.

The connections to the display board are brought out to a 10 pin header, J1. A flat cable is used to connect the display board to a 10-pin header that you must put on your breadboard and wire to the AVR128DB48. For our initial use of the LCD, you should simply ground BLC at the 10-pin

header on your breadboard.
The layout of the display board is shown below.



The ST7036 Dot Matrix Controller Hardware Driver IC on the DOG LCD Module includes its own microcontroller that controls what is displayed on the LCD display. Commands and data are sent over the SPI interface from the master microcontroller to the DOG LCD Module. The microcontroller on the ST7036 executes these commands as its instructions.

Basic driver software is needed that lets the AVR128DB48 communicate with the DOGM163W-A LCD using a SPI interface. This software must also provide functions to initialize and update the LCD. This software must be written in C for the AVR128DB48 in this laboratory. You could start from scratch and write this software or you could port this software from versions that we currently have in other assembly languages and in C for different microcontrollers. While an advantage of C is that its code is supposed to be portable. This is not a trivial process when the C code is referencing microcontroller peripherals.

You will have two choices of source code to port from if you decide to port existing software rather than to write your own C code from scratch.

In ESE381 for spring 2020 a C source file version of a DOGM163W-A LCD driver was written. However, this was for a SAML21J18B microcontroller which is a 32-bit ARM architecture implemented by Atmel. Since this is already written in C, I suggest you use this as the source code to port.

In ESE280 for fall 2019, the assembly language file `lcd_dog_asm_driver_m324.inc`, provided functions to operate the DOGM163W-A LCD. References to .inc and .asm files refer to the files from Laboratory 5 of fall 2019. Since the DOGM163W-A LCD is not a fast device, there is no particular speed advantage in having its drivers written in assembly language. However, there might be an advantage in terms of having accurate time delays written in assembly language to meet the LCD timing requirements for optimal LCD performance and to have delay times independent of the compiler optimization level.

Operation of the DOG LCD Module requires a power on initialization sequence. This can be done by a software function. Function `init_lcd_dog` is used for this purpose. After initialization, you could directly send commands to the ST7036 to control the display of data along with the data to be displayed. Using this approach, your program would deal directly with the ST7036 on a character by character basis.

An alternative way is to write a driver function for the ST7036. The function written uses three 17-byte memory buffers (arrays) in the microcontroller's SRAM. One buffer is provided for each line of the LCD display. Each location in a buffer corresponds to a character position on the corresponding line of the LCD display. The first location in a buffer corresponds to the leftmost character position in the corresponding line. Characters are represented in these buffers by their corresponding ASCII codes. The last location in the buffer is the end of string character.

The function `update_lcd_dog` handles all of the details required to display data in the SRAM buffers on the lines of the DOG LCD Module. Your application program has only to write the ASCII code for each character you want displayed into the corresponding location in the microcontroller's SRAM buffers and then call the `update_lcd_dog` function.

C functions are provided in the file `C_driver_s20_SAML21J18B_main.c`. Corresponding assembly language subroutines `init_lcd_dog` and `update_lcd_dog` are provided in the include file `lcd_dog_asm_driver_m324a.inc`. and the `dog_module_test` program uses those subroutines to test the DOG Module display. You should study the assembly language and C code carefully in order to port the code.

Design Tasks

Design Task 1: Hardware Interface to DOG Module
The LCD will be used as the primary information display allowing the display of 48 alphanumeric characters. The LCD is to be interfaced to the AVR128DB48 using SPI0. Use /SS to provide the chip-select for the LCD.

Use the following pins for the SPI signals:

PA4 for MOSI
PA6 for SCK
PA7 for /SS
PA5 for MISO

PC0 for RS of LCD

Thus, the LCD and the MAX5402 will share the MOSI, MISO, and SCK interface signals and have separate chip select signals.

Draw a schematic that shows the connections from the AVR128DB48 to the 10-pin header used to connect to the DOG Module, but do not include the circuitry of the prewired display board.

Submit your schematic as part of your prelab.

Design Task 2: LCD DOG Driver in C

Write a C driver named `lcd_dog_AVR128_driver`. This file must contain C functions with the same names as the external subroutines in `lcd_dog_asm_driver_m324`.

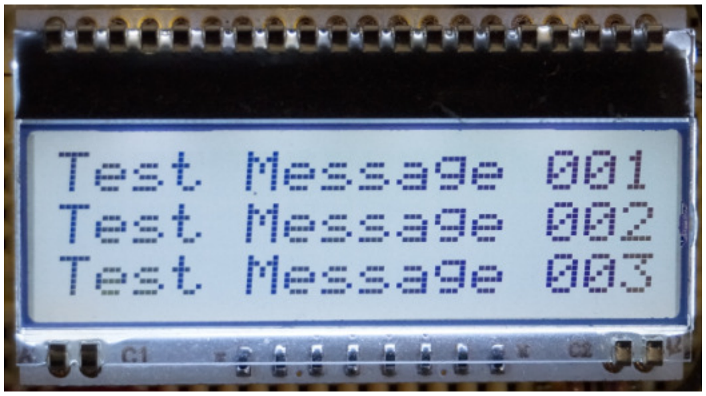
You can base your C version of the driver on the SAML21J18B C version or the assembly language version and information from the Sitronix ST7036 data sheet (see in particular page 43). We will be operating the LCD at 3.3 V, so the timing requirements for operation at that voltage apply.

Write `lcd_dog_AVR128_driver` so that its delays adjust appropriately for different operating frequencies of the AVR128DB48. That is, you want to be able to specify the frequency in a `#define` preprocessor directive and the delays are all automatically adjusted when the program is compiled. Assume that the lowest microcontroller frequency could be 1 MHz, even though we will normally be using 4 MHz. Assume that the highest frequency is 20 MHz.

Submit your C source file for this program as part of your prelab. Make sure that your program includes a program header, a header for each function, and clear comments throughout.

Design Task 3: Basic Verification of DOG Module Using a Test Program

A test program named `DOG_LCD_BasicTest` was used to verify the wiring of the LCD and the assembly language drivers. This program needs to be ported from the assembly language version to C, so that it can be used to test the C drivers that you have written for the LCD and your wiring of the LCD. This program should display the following lines.



Write a C version of the program `DOG_LCD_BasicTest`. Review the C library function `sprintf` that can be very useful for writing strings of characters to the display buffers with the messages you want displayed.

Submit your C source file for this program as part of your prelab. Make sure that your program includes a program header, a header for each function, and clear comments throughout.

Laboratory Activity

Laboratory Task 1: Basic Verification of DOG Module Using Test Program
Add the wiring to interface the DOG Module to the AVR128DB48. Do not remove the wiring to the MAX5402. Load the test program `DOG_LCD_BasicTest.c` that you wrote and run it. The display should display the messages shown in the display image shown in Design Task 3. If there are problems, check your wiring and debug your code.

When your program runs correctly, get a TA to verify its operation and obtain the TA's sign off.

Leave the hardware that you have wired on the breadboard intact.

Questions

1. What SPI mode must be configured in the SPI module to be compatible with DOG LCD. Annotate a copy of the DOG LCD timing diagram to show how you determine the mode.

2. What is the maximum bit rate at which you can send SPI data to the DOG LCD. Explain how you determined this value.