WRITING AN I2C DRIVER
FOR
A NEWHAVEN LCD DISPLAY

ECE 372 Winter 2018

February 10, 2018
Douglas V. Hall
Portland, OR

INTRODUCTION:

I2C is a 2-wire serial interface that is used with many currently available devices and modules such as A/D converters , displays, EEPROMS, etc. For this reason, most current processors have a built-in I2C controller, so you don't have to generate the I2C SCL and SDA signals by bit-banging GPIO pins. The TI Sitara AM 3358 processor you will be using has 3 built-in I2C controllers I2C(0-2). The overall goal of this project is display your name on a New Haven 2x20 LCD display by using the Beagle BoneBlack I2C1 controller to communicate with the display.

You have a choice of ONE of two different ways to build this project as follows:

1. Use Code Composer Studio and implement the program in ARM assembly language as you have done with previous projects. This is very doable. Students have successfully implemented a wide variety of I2C Projects like this in the past with Assembly language.

2. Use Code Composer Studio and implement the program in C using the I2C overview in the text and below with the process described in a handout that Charles Stoll wrote and is refining. This will be available when needed.

Key point is that all the initial research about I2C, the I2C controller, handshaking with I2C, and how you control the display are the same for both implementations, so you have lots of work to do on research and algorithms before needing the C directions.

NOTE: You MUST do all work by yourself with no help from anyone except the Instructor and the TA. Please do NOT contact companies for assistance. You are not a paying customer and therefore not entitled to ask them to solve your problems.

NOTE: YOU MUST KEEP AN AS-YOU-GO LOG OF THE ACTUAL STEPS YOU TOOK, YOUR DATA FINDINGS AT EACH STEP, PROBLEMS ENCOUNTERED, HOW YOU SOLVED THESE PROBLEMS, AND HOW YOU TESTED YOUR PROGRAM.

GRADING

Grading on this project will be a little different from the "all or nothing" grading of previous ECE 371 and ECE 372 projects as follows.

1. Successful communication with display using polled handshaking as in text example, 75% maximum.

2. Successful communication with display using Interrupt-based handshaking, 100% maximum.
3. Successful communication with display using Interrupt-based handshaking, and impressive manipulation of display, or program, 120% maximum.

REFERENCES

Hall Text, Chapter 7 section on I2C.

AM335x Sitara™ Processors Technical Reference Manual,
http://www.ti.com/lit/ug/spruh73p/spruh73p.pdf

NHD-C0220BiZ-FSW-FBW-3V3M New Haven Display Module
http://www.newhavendisplay.com/specs/NHD-C0220BiZ-FSW-FBW-3V3M.pdf

Sitronix Display Controller on New Haven LCD Board
https://www.newhavendisplay.com/app_notes/ST7036.pdf

DELIVERABLES

Your documentation for the project should include:

A. A detailed design log that clearly shows all the development steps you took and the results at each step.
B. Clearly written High-Level algorithms and Low-Level algorithms for the Initialization and communication sections.
C. Fully documented source file for the final program signed by the TA to verify that your program work and meet specifications. At signoff you will be expected to be able to accurately discuss details of your implementation.
D. A signed statement that you developed and wrote this program by yourself with no help from anyone except the instructor and/or the T.A. and that you did not give any help to anyone else. (Any evidence of joint work will result in project grades of zeros for all parties involved.)

SUGGESTED PROCEDURE:

1. Read through the Hall Chapter 7 section on I2C to get and overview of SCL and SDA signals, Start condition, Stop condition, ACK, etc. Note that the Sitara I2C controller is different from the Zeus I2C controller, so registers obviously different but general approach for communicating with controller similar.

2. Next, consider some high level initialization you need to do as follows
A. Take a look at the attached pin diagram for the P9 connector on the BeagleBone Black Board to find the pins identified as I2C1_SCL and I2C1_SDA. Note that the default for Pin 17 is the spi0_cs0 signal, so you will have to find the conf_ spi0_cs0 register in the Control Module and change the mode to connect the I2C_SCL signal to

pin 17. Likewise, you have to find the conf_ spi0_d1 register in the Control Module and change the mode as needed to connect the I2C_SDA signal to pin 18.

B. Think about how you turn on the clock for the I2C1 module with the appropriate register in the Clock Module as you did for the timer and UART.

3. Read through the I2C Chapter in the Sitara manual. Study Figure 21-8 to give you another view of the I2C waveforms for the 7-bit addressing mode you will be using. Then pay particular attention to section 21.3.15. to help you make a high level list of the steps required to initialize the I2C controller. Note how to get the system's 48MHz clock scaled down to 12 MHz and how to get a 100 Kbps SCL for standard mode operation (F/S) stands for Fast/Standard mode and HS stands for High Speed Mode.) Also note that for the first implementation you will be using polled mode transmit as I did in the Text example, instead of interrupt based transmission and you will not be using the DMA capability due to the small number of bytes to transfer. (From my own experience I found it best to get control of the hardware and the data flow path with simply polling, that I could step through, instead of starting with the added complexity of the interrupt system.)

4. Carefully read sections 21.3.15.3, 21.3.15.3, and 21.3.15.4 and 21.3.15.6.

5. Read though the Register list in Table 21-8 and mark any mentioned in the introductory sections. Then skim through the descriptions for the registers you have marked to find the specified bits you will use to initialize the controller, start a transmission, and transfer the next byte when the controller is ready. I find it useful to print out copies of the descriptions for the registers I am going to be using, so I can write notes on them and include them in my log.

6. Construct your high-level algorithm and then your low level algorithm for the program sections up to this point. This much is basically generic for communicating with any I2C device on a polled basis except for the steps to read from a slave as I did with the rangefinder in the text example. (The LCD I2C interface with the LCD Module does not allow reading of LCD Controller registers. You can only write to them.)

7. Carefully study the manual for the LCD module to determine the bytes you have to send to it to get it in the desired mode and determine how you send the characters you want displayed. Make a list of the required initialization words and the words needed to display your name. (You can use single height on two lines or double height. The displays have other capabilities, so you can also get fancy, if you want.)

8. Complete your High-level and Low-level algorithms (note structure of text example).

9a. If you are doing the all assembly language version, translate your Low-Level algorithms to assembly language. Then test and debug the program as needed. When the polled version correctly displays your name, have the TA or Instructor sign it off. Be prepared to briefly explain how your program or sections of it work.

9b. Once you get the polled version working, modify your algorithms to implement the handshaking on an interrupt basis with the Interrupt Controller. Then modify a **copy** of your program to include the parts necessary for it to work on an interrupt basis. When this is working, have the TA or Instructor sign it off. Be prepared to briefly explain how your program or sections of it work.

10a. If you are doing the C version of the program, then AFTER doing steps 1-8 above, **Very carefully** study Charles' I2CLCD C program handouts to help you implement the program in C.

10b. 12. Complete, compile, run, and debug the program. When you are successful in displaying your name, ask TA or Instructor to sign off basic display. Be prepared to briefly explain how program works.

11. Finally, think about how you can make the display rotate around to the right in a loop or blink on and off, then implement your choice and demonstrate to TA or Instructor when working. (Extra credit for this, up to 20 credits, if impressive.)

GRADING KEY FOR ECE372 DESIGN PROJECT D.V. HALL WINTER 2018

	POSSIBLE	SCORE
WORKING PROGRAM With TA questions answered correctly	**50**	**_____**
ALOGRITHMS (CLEAR AND COMPLETE)	**20**	**_____**
LOG (DETAILED AND "AS YOU GO")	**30**	**_____**
TOTAL	**100**	**_____**
Extra Feature	**up to 20**	