ELECTRICAL AND ELECTRONICS ENGINEERING DEPARTMENT

Laboratory Project: Battleship Game Console

Laboratory Project

Objectives

In the EE447 laboratory work, you were expected to familiarize yourself with the operation of TM4C123G and its utility modules. Now, in this final project you are expected to gather the previous experience on the microcontroller with novel information to achieve a multi-functional task. The objectives of the project are as follows:

- Interpretation of the necessities of complex task and encapsulation into sub-task
- Fulfillment of co-operation of utility modules
- Understanding a given complex hardware and compatibility of its components
- Writing a multi-task software for a given complex set up
- Introduction of the serial communication on TM4C123G and utilization of the facility on SPI protocol.

1 Project Definition

In this project you are expected to build a platform to play a different version of the famous "Battleship" game. This game does not rely on your luck, it rather puts your visual memory under test. It is a 2-Player turn-based strategy game controlled with two joysticks and two push buttons and played on a mobile phone screen... Yes, this is a low-budget game console.

The game is played in two stages: The ships are deployed in the first stage and they are tried to be destroyed with mines in the second stage.

1.1 Deployment

The admiral positions the ships

The first player (P1) takes the turn. Moves the cursor on the empty screen using two joysticks, one for up-down and the other for left-right movement. When he decides on a good spot, he places the first ship by pressing Button-1. Then he places three other ships on the screen one after the other, in a similar way. He is just about to give the turn to his enemy. However, P1 played a little trick: The ships that he placed are not all battleships, some of them carry civilians indeed. He did this just by pressing the other button (Button-2) while placing the ships. He overlooks at the screen for the last time to see how he placed the 4 ships and presses a button to clear the screen. The ships are cleared from the screen and the user is prompted that it is P2's turn now.

1.2 Attack

The enemy commander starts bombardment

The second player (P2) takes the turn. The screen is blank. P2 concentrates himself on the screen and as soon as he presses a button the screen is all revealed. He is able to see the location of all 4 enemy ships now! Not only that, he can also see which are battleships and which are civilian ships, since they are different in shape.

However, this sneak peek only lasts for about half a second. After 0.5 seconds, the screen is cleared back again. Now P2 has the control and he has to place the mines on the screen to hit the enemy battleships. Similar to what P1 did, his task is to move the cursor using the joysticks and place the mines using a button. However, there are the following challenges:

- P2 could see the positions of the ships for only a short period, and now he has to rely on his memory to place the mines.
- It is not acceptable to hit civilian ships.
- The placement of the mines has to be done in 20 seconds. P2 can see how much time is left from the clock that counts down on the upper right corner of the screen.

After 20 seconds is passed, the game engine checks the positions of ships with the position of mines. If the mines could successfully hit all the battleships without hurting any civilian ships then P2 wins the game, celebrated with a victory announcement on the screen. Otherwise game is over for P2, which is declared with a humiliating message from the enemy.

After the game ends, it starts back from the deployment phase for a new game.

2 Assumptions, Restrictions and Requirements

The battleship game console and the gameplay are described in Section 1. For the sake of simplicity there will be some assumptions about gameplay. Additionally, there will be restrictions in both the implementation of the gameplay and the hardware to be used in the console in order to encourage you to implement a multi-task software.

2.1 Restrictions

2.1.1 Hardware Restrictions

The game console is composed of the following materials:

1. NOKIA 5110 LCD Screen
2. 2 Potentiometers (As joysticks)
3. 2 Buttons Placed on the TM4C123G Board

The overall hardware to be used in the project is provided in Figure 1.

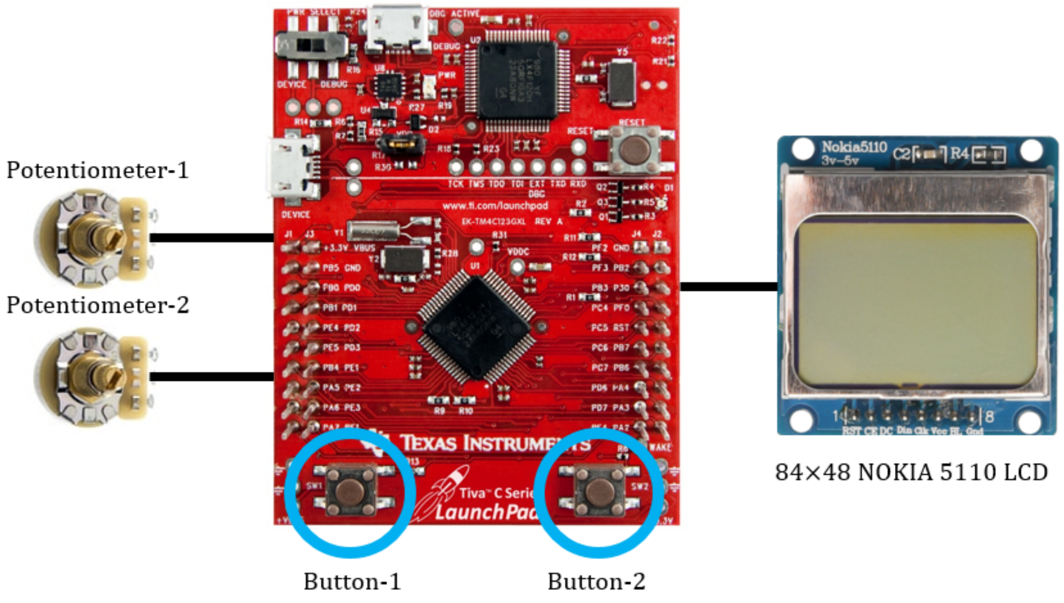


Figure 1: Hardware to be used in the project

2.1.2 Gameplay Restrictions

You have to display different icons for the cursor, battleships and civilian ships. Additionally, you should print digits for the countdown timer. The icons are to be placed within the play area and the countdown timer is to be placed in the top right corner of the screen.

The play area (bounding box with solid black boundary in Figure 2) has width of 64 pixels and height of 32 pixels and its top-left corner is at 7^{th} pixel from the left and 9^{th} pixel from the top. The cursor has to move within the play area and all the icons have to be printed within the play area completely. Namely, exceeding the play area for printing the icons is not allowed. The digits for the countdown timer is to be placed within the empty box of size 14×8 in the top-right corner. You may use the empty spots freely for your own printings if you wish.

For the icons and the countdown timer digits, you can freely choose your own style. However, there are restrictions in the maximum size of the printings. The countdown timer digits is to be placed within the 14×8 pixel box, boundaries included. That is to say, the height of a digit can be 8 bits. The maximum size of the bounding box for the cursor icon can be 5×5 pixels. You can draw whatever you want within that box as your cursor. Similarly, the maximum size of the bounding box for ships can be 8×8 pixels and you can draw whatever you want within that box as your ships. It is important to note that a battleship should be distinguishable from a civilian ship.

The summary of the gameplay restrictions is depicted in Figure 2.

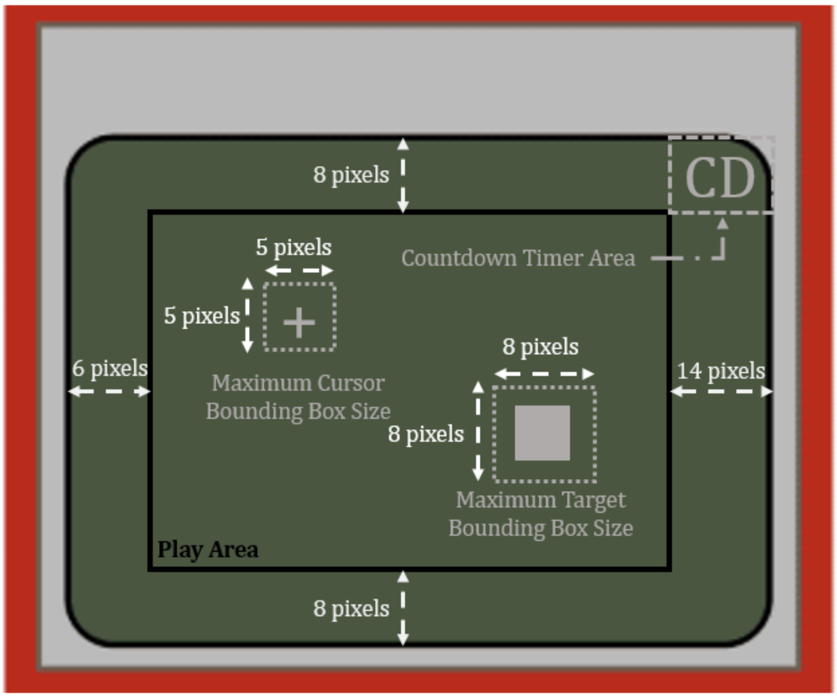


Figure 2: Gameplay restrictions

2.2 Requirements

First of all, you should specify the *pointing pixel* of your cursor icon. The term *pointing pixel* of the cursor is to be used repeatedly in the following explanations. That pixel is considered to point the target location. For example, if you have a + shaped cursor, your pointing pixel can be the one in the middle or if you have a △ shaped cursor, your pointing pixel can be the one on the top.

During the deployment phase, the pointing pixel of the cursor should point to the top-left corner of the bounding box of the ship icon. For example, if your ship icon is bounded by a 7×5 box, then the top-left corner of that box is to be placed at the location where the pointing pixel of the cursor is.

When a ship is placed, it has to be present in the screen until the end of the deployment phase.

No icon is required to display a mine. A mine is considered to be placed at the location where the pointing pixel of the cursor is, and the size of it is one pixel only. If that exact pixel is within the boundaries of a battleship then it is considered a match. P2 can place up to 4 mines.

It is important to note that the timings should be **exact**. That is to say, P2 will have exactly 0.5 seconds to see the position of the ships and exactly 20 seconds to place the mines. Those timings should not be affected by other operations.

2.3 Assumptions

For the sake of simplicity, you can make the following assumptions:

1. No error exists in the operating frequency of the MCU. Namely, if you are using the MCU at 16 MHz, then you may assume that each cycle is exactly $1/16\,\mu s$
2. P1 places at least one battleship.
3. P1 never places overlapping ships. Namely, the bounding boxes of the ships never intersect.

Anything that you develop that relaxes the aforementioned assumptions is welcome and to be considered as bonus. However, to have bonus points from the project you have to fulfill the requirements described in Section 2.2 by satisfying the restrictions described in Section 2.1.

3 Tips

On board buttons are connected to pins PF0 and PF4 [1]. Note that PF0 is locked [2] and you should unlock that pin to program it. You might want to refer pages 684 and 685 of the TM4C123GH6PM manual [2].

You will need to use interrupts and configure the priority of the interrupts. Recall that to configure an interrupt, you should know the interrupt number of the interrupt source you plan to use so that you can decide which NVIC registers (see page 141 of [2]) to be configured. You can find the interrupt number of an interrupt source from the table in page 104 of [2].

4 Background Information: Serial Peripheral Interface

Serial transmission involves sending one bit at a time, such that the data is spread out over time. Compared to parallel communication, many fewer lines are required to transmit data. This requires fewer pins but adds complexity. Serialized data is not generally sent at a uniform rate through a channel. Instead, there is usually a burst of regularly spaced binary data bits followed by a pause, after which the data flow resumes. Packets of binary data are sent in this manner, possibly with variable-length pauses between packets, until the message has been fully transmitted.

In synchronous systems, separate channels are used to transmit data and timing information. The timing channel transmits clock pulses to the receiver. Upon receipt of a clock pulse, the receiver reads the data channel and latches the bit value found on the channel at that moment.

The Serial Peripheral Interface (SPI) is a synchronous serial communication interface specification used for short distance communication, primarily in embedded systems and in this project it will be used. SPI involves a master and a slave, or multiple slaves. SPI interfacing involves 3 or more wires, consisting of a clock, serial data out, serial data in, and chip select if necessary. The master MCU basically sets the clock rate for the slaves, asks a specific one to listen up using the chip select port, and sends them commands via its serial data out port, and expects to receive the output from the slave through the serial data in port.

4.1 Synchronous Serial Interface in the TM4C

The TM4C has four Synchronous Serial Interface modules (SSI). The SSI is used to send synchronous serial communication to other devices, and can be configured to follow various protocols. We will use SPI in this lab, which requires that we specify which device will be sending data (Master), and which device(s) will be receiving data (Slave). When sending, the data is sent loaded into a FIFO buffer, and sent out according to the configured bit rate. Each FIFO buffer is 16 bits wide, and 8 locations deep. The size of the data can be configured to be from 4 to 16 bits wide depending on your needs.

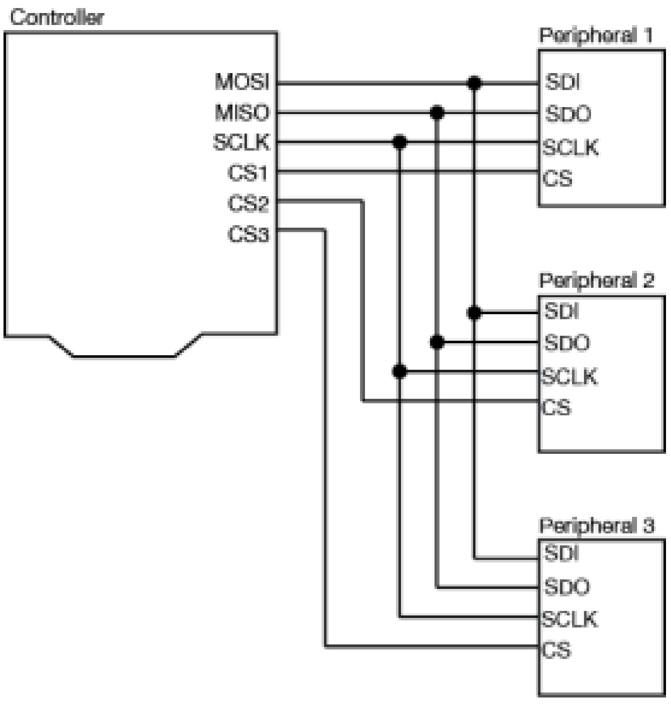


Figure 3: Synchronous Serial Interface in the TM4C

The pin connections for all SSI ports are labeled in Figure 3. When using Synchronous Serial Communication, there are typically 4 pins (lines).

- RX (MOSI) – Receiving data line
- TX (MISO) – Sending (transmitting) data
- Clk (SCLK) – The clock each bit is synched with
- Fss – Used to tell the slave that data is being sent

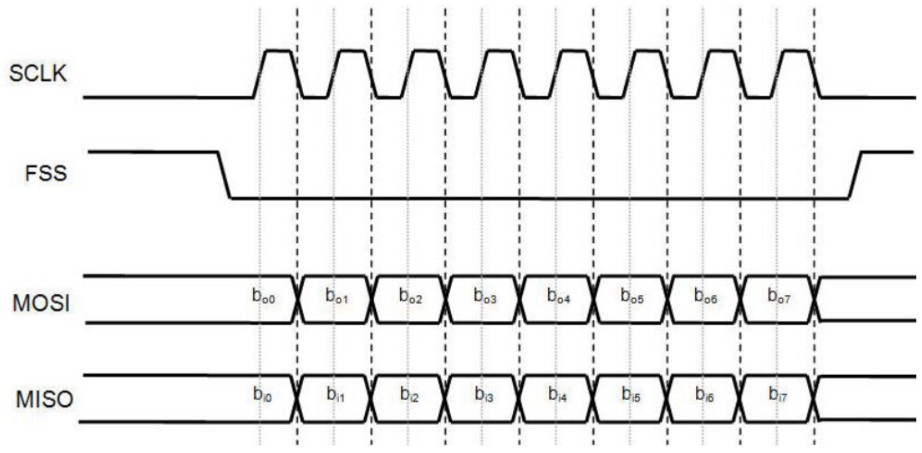


Figure 4: How SPI signals change as data is sent

4.2 SPI Configuration

In order to have Nokia 5110 functioning, 3 separate configurations are required, first of which is GPIO where the corresponding I/O pins are initialized to work as SPI pins. In the second part, the SPI module on the TM4C123 is configured to be compatible with LCD. Finally, in the NOKIA 5110 configuration part, the display is initialized for communication and to receive data to be displayed.

The Nokia 5110 uses SPI signals to receive commands, text, and images to be displayed. As to be noticed in Figure 5 on the SPI signals that there is only one data output signal. The LCD somehow, needs to distinguish whether the data being sent is data meant to be displayed, or if it is a command meant to control the screen. This is done not through the SPI module, but "manually" through a GPIO pin. The Nokia screen has a pin named Data/Command (DC). When the DC signal is low, the Nokia interprets the incoming SPI bits as a command. Similarly, if the DC signal is high, the SPI bits are interpreted as data to be displayed. The DC signal can be set high or low long before the last bit is sent.

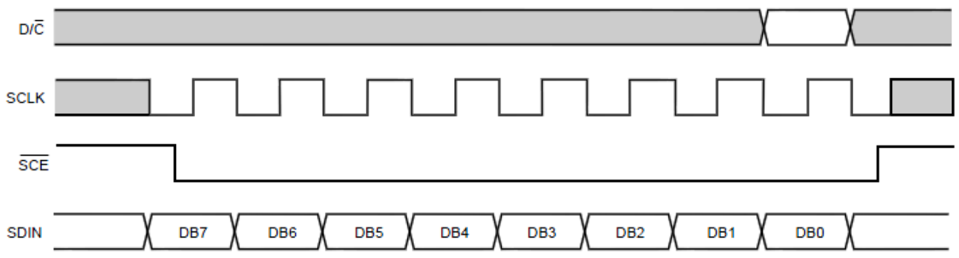


Figure 5: How the DC signal is timed with the SPI signals

For GPIO and SPI configuration parts, you may refer to page 965 of the TM4C123 Datasheet for details and 967 for register map.

GPIO Configuration:

1. Enable the clock for GPIOx **(RCGCGPIO)**
2. Wait until GPIOx is ready **(PRGPIO)**
3. Configure the CLK, CS (FSS), MOSI (Tx), and MISO(Rx) pins as a digital pin **(DEN)**
4. Set directions for the pins **(DIR)**
5. Configure the CLK, CS (FSS), MOSI (Tx), and MISO(Rx) pins for their alternate function **(AFSEL)**
6. Configure the CLK, CS (FSS), MOSI (Tx), and MISO(Rx) port control pins to route the SSI interface to the pins **(PCTL)**.

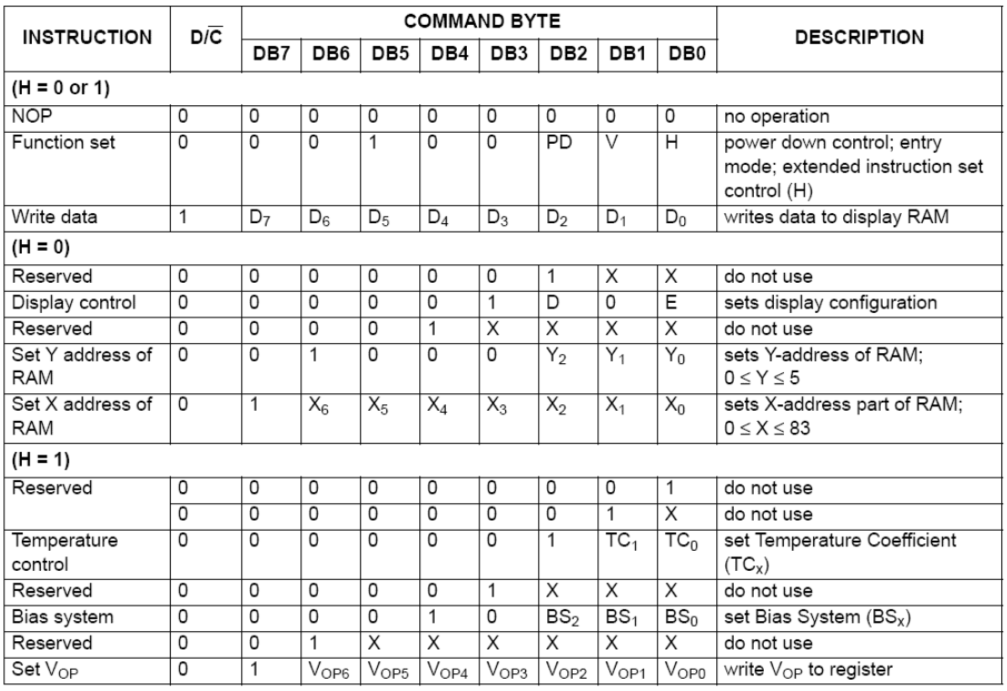
SPI Configuration:

1. Enable the clock for SSIx **(RCGCSSI)**
2. Wait for the SSI peripheral to be ready **(PRSSI)**
3. Disable the SPI interface **(CR1)**
4. Set the clock rate of the SPI Clock **(CPSR, CR0)** accordingly. Please mind the maximum data rate that the LCD is capable of working with.
5. Set the data size to be 8-bits and Freescale mode **(CR0)**
6. Set the SPI mode **(CR0)** accordingly.
7. Re-enable the SPI interface **(CR1)**

For Nokia 5110 LCD configuration you may refer to Nokia5110Datasheet.pdf provided along with the project manual, where a detailed version of the instruction set and data transmission graphs are provided for clarification.

NOKIA 5110 Configuration:

1. To initialize the Nokia screen first toggle the Reset pin by holding it low for 100ms then setting it high.
2. Send the following commands to inialize the display
   - Set H=1 for Extended Command Mode, V=0 for Horizontal Addressing
   - Set V_{OP}. You may need to sweep values between 0x[B0-C0] for correct operation.
   - Set temperature control value. You may need to sweep values between 0x[04-07] for correct operation.
   - Set voltage bias value as 0x13.
   - Set H=0 for Basic Command Mode
   - Configure for Normal Display Mode
   - Set Cursor to detemine the start address
3. Send data to be displayed.



INSTRUCTION	D/$\overline{C}$	COMMAND BYTE								DESCRIPTION
		DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
(H = 0 or 1)										
NOP	0	0	0	0	0	0	0	0	0	no operation
Function set	0	0	0	1	0	0	PD	V	H	power down control; entry mode; extended instruction set control (H)
Write data	1	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	writes data to display RAM
(H = 0)										
Reserved	0	0	0	0	0	0	1	X	X	do not use
Display control	0	0	0	0	0	1	D	0	E	sets display configuration
Reserved	0	0	0	0	1	X	X	X	X	do not use
Set Y address of RAM	0	0	1	0	0	0	Y_2	Y_1	Y_0	sets Y-address of RAM; $0 \leq Y \leq 5$
Set X address of RAM	0	1	X_6	X_5	X_4	X_3	X_2	X_1	X_0	sets X-address part of RAM; $0 \leq X \leq 83$
(H = 1)										
Reserved	0	0	0	0	0	0	0	0	1	do not use
	0	0	0	0	0	0	0	1	X	do not use
Temperature control	0	0	0	0	0	0	1	TC_1	TC_0	set Temperature Coefficient (TC_x)
Reserved	0	0	0	0	0	1	X	X	X	do not use
Bias system	0	0	0	0	1	0	BS_2	BS_1	BS_0	set Bias System (BS_x)
Reserved	0	0	1	X	X	X	X	X	X	do not use
Set V_{OP}	0	1	V_{OP6}	V_{OP5}	V_{OP4}	V_{OP3}	V_{OP2}	V_{OP1}	V_{OP0}	write V_{OP} to register

Figure 6: Instruction Format

It is possible to write data into the address of memory (DDRAM) of the Nokia LCD continuously and values of X-Address and Y-Address will be increased automatically. In this case, there are 2 methods to configure the operation format of address; firstly, Vertical Addressing Mode (V=1), 1 value of Y-Address will be increased every time; and secondly, Horizontal Addressing Mode (V=0), 1 value of X-Address will be increased every time. One may use either addressing as pleased; however it is to be noted that when the cursor is set to an arbitrary location, a very short delay, in the order of nsec is required for the cursor to settle.

5 Deliverables

Your are supposed to attempt the project work in pairs, you may send an e-mail, single for each group, to the coordination, until 10th Dec, 2018 to let us know the identities of the partners in collaboration.

- **Pre-Report:** A summary of the framework, at most 3 pages. **Deadline: Jan 4th, 2019**
- **Final Report:** A full description of your work, including photos of your setup and fully executable codes (printed). Please clearly indicate how you fulfill the requirements by satisfying the restrictions. Moreover, your codes should be well-commented. **Deadline: Jan 20th, 2019**
- **Source Code:** A zipped Keil μvision project folder with fully executable codes is to be uploaded to ODTUClass. **Deadline: Jan 20th, 2019**
- **Lab Demo:** Demonstration of the operation of the project. **On Jan 21st-22nd, 2019**

References

[1] TI, “Tiva$^{\text{TM}}$ c series tm4c123g launchpad evaluation board user’s guide.” http://www.ti.com/lit/ug/spmu296/spmu296.pdf.

[2] TI, “Tiva$^{\text{TM}}$ tm4c123gh6pm microcontroller data sheet.” http://www.ti.com/lit/ds/spms376e/spms376e.pdf.