Lab 6 Prelab

**Name**: Chimzim Ogbondah

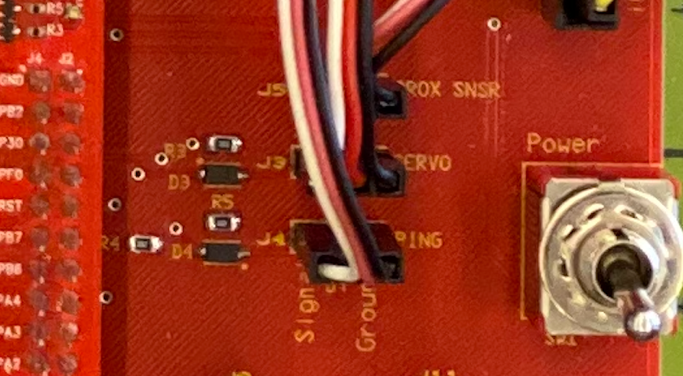
**Lab Partner Name (if you worked together and are submitting the same document or mostly the same answers):**

**Lab Section**: 1

**Submit your prelab document as a PDF file in Canvas under the corresponding prelab assignment. Every student submits their own prelab. Lab partners are allowed to work on the prelab together and submit the same document (if there is actual collaboration on the document). For full credit, the prelab must be submitted prior to the start of lab. Text responses should be typed or printed neatly.**

Complete the following preparation before doing Prelab 6 and before the Tuesday class meeting: see preparation guide, [reading-video-prep-ADC.pdf](https://canvas.iastate.edu/courses/69323/files/10329308/download?wrap=1), under [Readings - Week 6 - ADC](https://canvas.iastate.edu/courses/69323/pages/readings-week-6-adc). Prep time is estimated as 30-40 minutes via short videos.

1. System sketch

Similar to system sketches in previous labs, sketch a diagram that shows how the infrared (IR) sensor connects to the microcontroller. Refer to the file Cybot-Baseboard-LCD-Schematic.pdf, as needed. In the photo of the CyBot baseboard, shown at right, the jumper wires are connected to the sensors and servo motor. PROX SNSR refers to the IR sensor.

Your diagram should show the port and the pin (or bit) number of the port that the IR sensor is connected to. Give the port name and pin number used in the microcontroller (e.g, Port A pin 0, PA0).

Port B pin 4 PB4

1. The voltage from the IR sensor is sampled using a specific channel of the microcontroller’s Analog to Digital Converter (ADC). The channel used in the lab is Analog INput (AIN) channel 10. This is fixed for the CyBot based on the wiring of the sensor to the microcontroller. How do you know that the wiring corresponds to AIN 10? Hint: see Table 23-5 in the Tiva datasheet.
   1. It gives the analog function and under the function it has AIN10 which lines up with pin PB4 (they are matched up with specific GPIO ports)
2. Refer to the list of ADC registers at the end of the prelab. The list is itemized using letters A, B, C, etc. (in no particular order). Note that some registers are used for initialization (e.g., Program 14.1 in VYES Chapter 14), and some registers are used for sampling the input (e.g., Program 14.2 in VYES Chapter 14). Even though the registers look complicated, and the ADC has lots of options, we will focus on a relatively small subset of functionality that has been highlighted for you.  
     
   Assuming the GPIO port has already been initialized, the default 16 MHz clock source will be used for the ADC, and polling will be used to read a conversion result, which 6 registers from the list must be configured in initialization code before using the ADC for a conversion? You can identify registers by their letter (A, B, C, etc.) in the list.
   1. **ADCEMUX (G)**
   2. **ADCACTSS ©**
   3. **ADCSSCTL0-3 (J)**
   4. **ADCSSMUX0-3 (I)**
   5. **ADCPSSI (H)**
   6. **RCGCADC**
3. Select one of the 6 ADC registers you identified in question 3. **ADCEMUX**  
   Answer parts a-c for this register.
   1. What is the C macro name for this register from the system header file?

* **ADC\_EMUX\_R**  
  1. What is the memory address of this memory-mapped I/O register?
* 1. Sketch the register as shown in the datasheet description. Circle the bit or bit fields that must be set up for this lab.

1. **Optional** (Bonus): Suppose the digital conversion result (digital encoding or quantization value) obtained from a TM4C ADC sample of the IR sensor is 2050 (decimal). Corresponding to this result, estimate – without actually measuring – the distance in centimeters to the object being detected. Use available information including VYES book Chapter 14, the IR sensor datasheet (e.g., page 8), and ADC datasheet (e.g., Figure 13-8). Note that actual measurements may differ for various reasons, but a rough estimate is possible. (2 points)

ADC Register Usage List

1. ADC Run Mode Clock Gating Control (**RCGCADC**): Control the clocking to enable the ADC module
2. ADC Peripheral Ready (**PRADC**): Check whether the ADC module is ready for access
3. ADC Active Sample Sequencer (**ADCACTSS**): Enable or disable a sample sequencer (SS) unit
4. ADC Raw Interrupt Status (**ADCRIS**): Flag to indicate whether ADC conversion is done in SS (may be polled by software without sending an interrupt to the NVIC), status bit for each SS
5. ADC Interrupt Mask (**ADCIM**): Enable or disable sending interrupts to NVIC, mask bit for each SS
6. ADC Interrupt Status and Clear (**ADCISC**): Clear SS “interrupt” status by writing 1
7. ADC Event Multiplexer Select (**ADCEMUX**): Select the event that initiates a sample for each SS, use default of 0 (processor, which means software via PSSI)
8. ADC Processor Sample Sequence Initiate (**ADCPSSI**): Initiate a sample in a SS (start ADC conversion)
9. ADC Sample Sequence Input Multiplexer Select (0-3) (**ADCSSMUX0-3**): Select the analog input channel for each sample taken by the SS (one register for each SS, 4-bit field for each sample, sequence starts with sample 0)
10. ADC Sample Sequence Control (0-3) (**ADCSSCTL0-3**): For each sample, 4 bits configure the sample (e.g., 0b0110 for stopping with this sample and raising an “interrupt” when done with the sample)
11. ADC Sample Sequence Result FIFO (0-3) (**ADCSSFIFO0-3**): Contains conversion result(s) for samples taken by an SS (one register for each SS), consecutive reads return sample 0, sample 1, etc.