UNIVERSITY OF TECHNOLOGY SYDNEY

FACULTY OF ENGINEERING

Subject: ***48623 – Mechatronics 2***

Assessment #: ***2***

Assessment Title: ***Actuation and Sensing***

Tutorial Group:	
Student Number:	
Family Name:	
First Name:	

Declaration of Originality

The work contained in this assignment, other than that specifically attributed to another source, is that of the author(s). It is recognised that, will this declaration be found to be false, disciplinary action could be taken and the assignment of the student involved will be given zero marks. In the statement below, I have indicated the extent to which I have collaborated with other students, whom I have named.

Statement of Collaboration

Signature(s)

Marks

Demonstration	/16
Quiz	/6
TOTAL	/22
Assessment I Mark	/10

Tutor use only

✂

Mechatronics 2 - Assignment Receipt

Assignment Title:	***Actuation and Sensing***
Student's Name:	
Assessment Date:	
Tutor Signature:	

Aim

This assessment aims to evaluate the students' skills in interfacing and integration of exteroceptive sensors and actuators. Students are to create functions to control a stepper motor with different operating modes, including continuous rotation and discrete step counting. Students will need to acquire calibrated distance readings from an IR sensor, display them on an LCD and use them as input in the control of a stepper motor.

Background

For this assignment, you will be programming the drive system of an autonomous wheel-based robot. As you are programming this system as a component of a more extensive system, you must ensure that the system has a modular design that can handle different user settings and must have both a normal mode and a debug mode. The specific details are outlined in the Requirements section.

The drive system will consist of the Arduino UNO board, an LCD shield, an IR sensor, and a stepper motor + motor driver board. There is a hypothetical wheel attached to the stepper motor. The size of the wheel is not specified. The programmed size of the wheel can be changed within the debug mode settings, however the default wheel circumference 20 cm. The primary function of the system is to take a distance reading from the IR sensor, and then travel to that distance by rotating the wheel/stepper motor the correct amount to reach that distance. E.g. If the wheel circumference is 20cm and distance reading is 70cm, then the stepper motor would have to rotate 3.5 rotations to reach that distance.

Requirements

Students are required to use the work from previous exercises with the Arduino UNO and create the necessary functions to perform the tasks specified below.

Start Up Mode

- **LCD**
  - **Clock:** When the Arduino is powered on. The LCD should display the time that has passed since the Arduino has been powered on. The display needs to be in the format of minutes: seconds. The screen should need to update every second passed with the seconds resetting and the minutes increment after 60 seconds. This clock should be displayed on the left-most corner of the first line. This clock should be continuously running during the start-up mode and continue to count when in other modes.
  - **Student ID:** The second line should display your student ID number on the left side.
- **User Input**
  - **Button Press:** In this mode, the system should only respond to two user inputs.
    - SELECT Button: This should prompt the system to go into **Drive mode**.
    - Debug Combination: When the following sequence of buttons (Not at the same time) is pressed in order LEFT -> LEFT -> UP -> RIGHT -> SELECT, the system should go into the **Debug mode**. The system should ignore any incorrect sequence attempt.

Debug Mode

- **LCD**
  - **Mode Name:** The name of the mode 'DEBUG Mode' should be displayed on the left side of the first line.

  - **Debug Mode Selection:** The second line should display the test modes and settings that the user can select. The exact text on display: IR CM PM SET E.
    - The user should be able to use the buttons to select and enter the highlighted mode. The current highlighted mode should blink at a frequency of 1Hz.

- **User Input**
  - **Button Press:** The user should be able to use the LEFT and RIGHT buttons to move between the five options inline 2. When the buttons are pressed, the previous or next option should be highlighted (blinking). When the SELECT button is pressed, the system should transition to the current highlighted mode, the modes are:
    - IR: Infrared sensor test mode or **IR Mode**
    - CM: Continuous motor test mode or **CM Mode**
    - PM: Precise motor test mode or **PM Mode**
    - SET: Settings mode or **SET Mode**
    - E: Exit debug and return to **Start Up Mode**
      - The clock in **Start Up** Mode should be continuous running in **Debug Mode** even if it is not visible.

IR Mode (Infrared Sensing Mode)

- **LCD**
  - **Mode Name:** The name of the mode 'IR Mode' should be displayed on the left side of the first line.
  - **IR Reading:** The second line of the LCD should display the IR sensor readings in cm units (Not ADC values). The distance value displayed must be accurate and stable.
- **IR Sensor**
  - Analog voltage value must be read from the sensor using the microcontroller's ADC and displayed as a distance (not a voltage).
  - Calibration must be performed by taking measurements and finding the relationship between distance and voltage.
  - The sensor should be polled frequently and averaged to increase accuracy. It is up to the student to decide which polling and averaging rates produce the most accurate and reliable measurements via experimentation.
- **User Input**
  - **Button Press**: The SELECT button should cause the system to exit **IR Mode** return to **Debug Mode**

CM Mode (Continuous Mode)

- **LCD**
  - **Mode Name:** The name of the mode 'CM Mode' should be displayed on the left side of the first line.
  - **Menu Options:** The second line of the LCD should display two options commands, start motor and exit mode option. The exact text on display: Start Exit. Similar to **Debug Mode**, there should be a highlighted (blinking) option based on the user's button press.
  - **Start Motor:** If the user selected the 'Start' option, then the second line should be replaced with an information readout. This line should display the current rotation (CW or CCW) and speed (A value to denote the current speed). If the user prompts to return to the **Menu Options**, the second line should redisplay the menu option.
- **Stepper Motor**
  - **Menu Options:** The motor should be idle during menu option selection.

  - **Start Motor:** When the user has selected the 'Start' option in the menu, the motor should begin to rotate continuously. The user can use the buttons to control the speed and rotation of the motor. The user should also be able to stop the motor and return to the menu options.
  - The motor must have **at least** three distinct speed levels.
    - By default, the motor should rotate clockwise and at a midlevel speed. Whenever the motor stops and restarts/re-enter **Start Motor,** the motor should return to the default setting regardless of previous changes.
- **User Input**
  - **Menu Options Button Press**: The user should be able to use the LEFT and RIGHT buttons to move between the two options inline 2. When the SELECT button is pressed, the system should execute the current highlighted command:
    - Start: The system should start the motor movement and exhibit the **Start Motor** behaviours.
  - **Start Motor Button Press:** The user should be able to use the buttons to control the motor behaviour while the motor is rotating. The motor should remain continuous and not stop when a button is pressed (except SELECT).
    - LEFT and RIGHT: This should change the direction of motor rotation (CW or CCW). Each button should be mapped to a specific direction of rotation.
    - UP and DOWN: This should increase the speed of the motor, and the DOWN button should decrease the speed of the motor.
    - SELECT: This should stop the rotation of the motor and return to the **Menu Options**.

PM Mode (Precise-Step Mode)

- **LCD**
  - **Mode Name:** The name of the mode 'PM Mode' should be displayed on the left side of the first line.
  - **Step Count:** The second line should display the stepper motor step counts set and remaining. The default step count set should be 100 steps. The user should be able to modify the step count setting. Once the user has started the motor rotation, the step count should count down with each step the motor takes.
- **Stepper Motor**
  - **Precise Stepping:** Motor should begin to rotate when the user has set the step count and initiated the stepping through the button press. The motor should rotate the precise number of steps indicated by the user.
- **User Input**
  - **Button Press:** The user should be able to use the buttons to set the number of steps rotated by the motor and start the motor rotation.
    - UP and DOWN: This should increment or decrement the current step count; every button press should increase/decrease the count by 100 steps. When the button is held down, the count should continually increase/decrease.
    - LEFT: This should reset the step count back to the default value of 100 steps.
    - Right: This should start the motor rotation; the step count cannot be modified during the rotation. The LCD step count should decrease as the motor rotates.
    - SELECT: This should stop the rotation of the motor, exit this mode, and return to the **Debug Mode**.

SET Mode (Settings Mode)

- **LCD**
  - **Mode Name:** The name of the mode 'SETTINGS Mode' should be displayed on the left side of the first line.
  - **Settings Options:** The second line of this mode should display "Wheel: xx cm", the default xx value is 20cm. The xx value is the size of the wheel that should be fitted onto the motor. This value should affect the number of rotations the motor should take when in **Drive Mode**. If the value has been previously changed, then the value should be the previously set value. The user should use the buttons to change the value of xx.
- **User Input**
  - **Button Press:** The buttons should be used to select menu options and adjust the wheel size and exit this mode.
    - UP and DOWN: This should adjust the wheel size value in increments of 10 cm.
    - SELECT: This should exit the current mode, and the system should return to **DEBUG Mode**.

Drive Mode

- **LCD**
  - **Mode Name:** The name of the mode 'Drive Mode' should be displayed on the left side of the first line.
  - **Distance, Revolutions, Steps:** The second line should have three values displayed on the LCD. These values should be continuously updated based on the sensor readings. Once the user locks in the distance and initiates driving, the values should represent the distance, revolutions, and steps remaining from the target.
    - **Distance**: This should be the distance value read by the IR sensor in cm. While driving this should be the distance remaining from the target.
    - **Revolution:** This should be a scalar value (1 decimal place) representing how many revolutions should be needed to reach the distance values. This value should be dependent on the wheel size value that is set in **SET Mode**. While driving this should be the revolutions remaining to reach the target.
    - **Steps:** This should be an integer value that represents how many stepper motor steps are required to reach the target. While driving this should be the steps remaining to reach the target.
- **IR Sensor**
  - The IR sensor should provide an accurate and stable reading that should be displayed on the LCD.
  - Once the user initiates the driving, the system should lock the distance value read at that time, and the IR sensor should stop reading values.
- **Stepper Motor**
  - When driving has begun, the stepper motor should rotate the precise amount of steps/revolutions required to reach the locked in distance hypothetically.
  - The stepper motor should be able to rotate CW, and CCW based on the user input.
  - The motor should rotate at a speed of 1 step/millisecond.
- **User Input**
  - **Button Press**: The user should be able to use the buttons to initiate the driving function and exit this mode.
    - UP and DOWN: This should lock in the distance value and initiate driving. UP should have the motor drive to the target by rotating CW while DOWN should be performing the same operation in a CCW rotation.

- SELECT: This button should return the system to **Start Up Mode**. The clock should show the time elapsed since the system initial start-up.

Quiz

- Students will have to complete an online quiz that aims to test their knowledge on the components of this assignment.
- Following aspects may be tested in the quiz :
  - Overall system design including the hardware and software components.
  - An understanding of each task and related software components and how they were implemented.
  - All calibrations and any mathematical formulae used when programming the system.

Notes

- Start this assignment **early** to complete all tasks by the deadline.
- Write down what your program needs to do mathematically AND logically (pseudocode) before you begin programming. It is easier to translate this into C code .
- **YOU ARE NOT PERMITTED TO USE THE ARDUINO LIBRARIES AND OTHER NON-STANDARD LIBRARIES**; however, you are permitted to use the LiquidCrystal.h library for the LCD. You are allowed to use the C standard libraries.
- Make sure you attend class each week as some of the significant components of this assignment may be looked at.

Assessment Criteria Linkages

Criteria	Weight (%)	SLOs
Meeting the specification	60	1
Appropriateness of the software design	20	2
Neatness of the software (modular)	20	2

SLOs: Subject learning objectives

Rules for Submission

Due Date:

7 September 2020, 11:59pm

QuizStudents will be required to answer knowledge-based questions about the system design; this will be done through UTS Online. This quiz will be available on the due date of the assignment (00:01am to 11:59pm).

Code

The code must be submitted in a **PDF Document** and in a **single .ino file.** Students must submit **all their code** into UTS Online (Two submission section for PDF and .ino code). The code must be in text form and must be able to be built in a single file. **This code will be used to assess your demonstration; you need to ensure that your program works when uploaded to our generic setup (Uno, LCD Shield, Stepper Motor, IR Sensor).** It should be in **PDF format**.

Demonstration

Due to the current online classes situation, we will not be performing a face to face demonstration. In order to mark your demonstrations, the tutors will set up a similar hardware configuration with an UNO, Stepper Motor, Motor driver, IR Sensor, and LCD Shield (both versions). Your code provided will be uploaded via the Arduino IDE. If you are using other IDEs, you must ensure that the code can be built and uploaded using the Arduino IDE. We will be directly copying and pasting the code into the Arduino IDE. We will not be accepting any custom written header files. **You must provide clear instructions for the details of your hardware set up (pin connection and other hardware set ups).**

Things to Include in the instructions:

- Hardware Set up instructions for the demonstration:
  - Which pins did you use for the hardware components?
  - Which LCD shield did you use? (Red v1.1 or Black v1.0)?
  - Failure to provide these details **will** negatively affect your demonstration marks.
- Additional Hardware:
  - You must confirm use of additional hardware with tutors before you decide to use them.
  - E.g. PSU, batteries, and RC circuits.

Students found using other student's code will receive no marks for this assignment and have disciplinary action taken against them by the university.

Support and Assistance

Support and assistance for this assignment will be available by posting questions on the "Tutorials and Assignments" forum on UTSOnline. This forum is monitored electronically and as such will have the same response time as a direct email. Please use the forum so that other students may benefit from the answers given.

Face to face support is available during the lecture and/or tutorial timeslot. Please email to make an appointment.

Students with difficulty meeting assessment requirements

Students who experience **significant** difficulty, or anticipate that they will experience significant difficulty, in meeting assessment requirements must submit an "Application for Special Consideration form" (available at http://www.sau.uts.edu.au/assessment/consideration/online.html) to the Registrar **before** the due date of the assessment item. Significant difficulty means

i. Serious illness or psychological condition.
ii. Loss or bereavement
iii. Hardship/trauma

Note also that students may apply for special consideration because of illness or other circumstances (**not work related**) beyond their control. The "Application for Special Consideration form" has a section that must be filled in by a doctor, counsellor or other relevant professional authority. A medical certificate alone is not adequate and will not be accepted. Note that it is up to the students to provide adequate information about their circumstances. University staff will not chase additional information and the Subject Coordinator has the right to reject applications that lack sufficient information.

It is the student's responsibility to contact the Subject Coordinator to find out what action has been taken and to obtain details of any additional assessment required or learning and assessment special arrangements.

For further details please refer to section 4.6 of the "Coursework Assessment Policy and Procedures Manual".

Marking Scheme

	Item	Mark Allocation	Marks Awarded
Overall System (3 Marks)	Buttons are correctly debounced	1.5	
	LCD Displays Modes Clearly	0.5	
	LCD Displays the second line clearly (Menu options, numerical data) (-0.5 for each incorrect mode)	1	
Start Up Mode (2 Marks)	Display the correct time elapsed	1	
	System Transitions to the correct mode based on the input sequence. (-0.5 not correctly enter debug mode)	1	
Debug Mode (0.5 Marks)	Correctly Transitions using the menu options	0.5	
IR Mode (1 Mark)	Distance continuously displayed on LCD in cm	0.5	
	Distance is accurate (within 5%) and stable	0.5	
CM Mode (2.5 Marks)	Correctly Transitions using the menu options	0.5	
	Left and Right buttons for CCW and CW rotation	1	
	Up and Down buttons for speed control	1	
PM Mode (3 Marks)	Buttons perform the correctly mapped functions (-0.5 for each incorrect button)	2	
	The motor rotates and displays the correct number of steps.	1	
SET Mode (1 Marks)	Functionally change the wheel size setting (checked in **Drive Mode**)	1	
Drive Mode (3 Marks)	The system allows the user to lock in a stable distance reading and initiate driving.	1	
	The motor rotates the correct amount based on the locked-in distance	1	
	The motor can rotate both CW and CCW based on user input	0.5	
	Display the correct conversion for distance -> revolutions -> Steps	0.5	
Quiz (6 Marks)	System Design Questions	6	
Total (22 Marks)		22	