#### 07/14/2022, Morning

#### Biomedical and Electrical Engineering – Robotic Control in Medicine

#### Drs. Curtis Wang and Deborah Won, Maria Gonzalez and Eyleen Sanchez Varela

**Terms:**

* Biomedical engineering
* Medical robotics
* Sensors
* Actuators
* Feedback
* Programmable control
* Embedded systems
* Assist devices

### Tools:

* Arduino microcontroller
* Oscilloscope
* DC power supply
* Multimeter
* FSR
* Breadboards
* Jumper wires
* Graduated cylinder – 10mL
* Suave
* Sponges

### Science Questions:

* What are applications of medical robotics?
* What are microcontrollers and why are they useful?
* What are examples of sensors that could be used in medicine?
* How do we make robots and devices carry out desired actions?
  + How do we make a pump run without manual human labor?
  + How do we move an object (arm) where we want it to go?
  + How do we make a robot know when to stop (or start) on its own?

**Introduction**

Robotics, in some form, is around us almost all the time in our modern society. Look around you and notice all the “smart” gadget and electronics we have around us. Your smart phone, cruise control in the car, and even your air conditioner’s thermostat, are all designed with the same basis for operation as robots: programmable automated control. Some other examples that you may not be as familiar with but which are used by millions of people around the world, and may be used by more people you know than you realize, include a heart pacemaker, an insulin pump for diabetes treatment, and surgical robots.



Image credit: Chris Newmarker, The Robot Report

Medical robotics contain little computers called microcontrollers, which sense information from the patient’s body or the patient’s environment and then use that information to make decisions about how to move a surgical tool, or how to deliver an electrical therapy, or how much medicine to deliver, or perhaps, how to move a device designed to assist people who cannot move their legs or arms.

In the U.S. alone, 13.7% of adults have a disability that impairs their ability to move or walk, according to the CDC; many others also cannot use their arms. Many of them are therefore, unable to shower on their own. In this workshop, you will get some experience with microcontrollers and how to use them to control a robotic arm and shampoo dispenser to help someone who has upper limb motor impairment. You will learn how we can get these tiny computers to read in information from sensors and how to get them to move the robotic arm and dispense the right amount of shampoo. We hope you will see very clearly that microcontrollers are very useful tools to helping people with a wide variety of problems!

**Learning Objectives**

* Gain awareness of how robots could be useful in medical health care, and to be able to define “medical robotics”.
* Learn about different types of motors and the types of actuation they do.
* Learn about different sensors and how they are used in robotics.
* Program microcontrollers to carry out functions needed in an automated showering device (namely, holding sponge, dispensing shampoo, moving arm, and scrubbing)
* Instrument circuits which communicate with the microcontroller
* Build working modules of a robotic shower assist device.

**Lab/Activity Instructions**

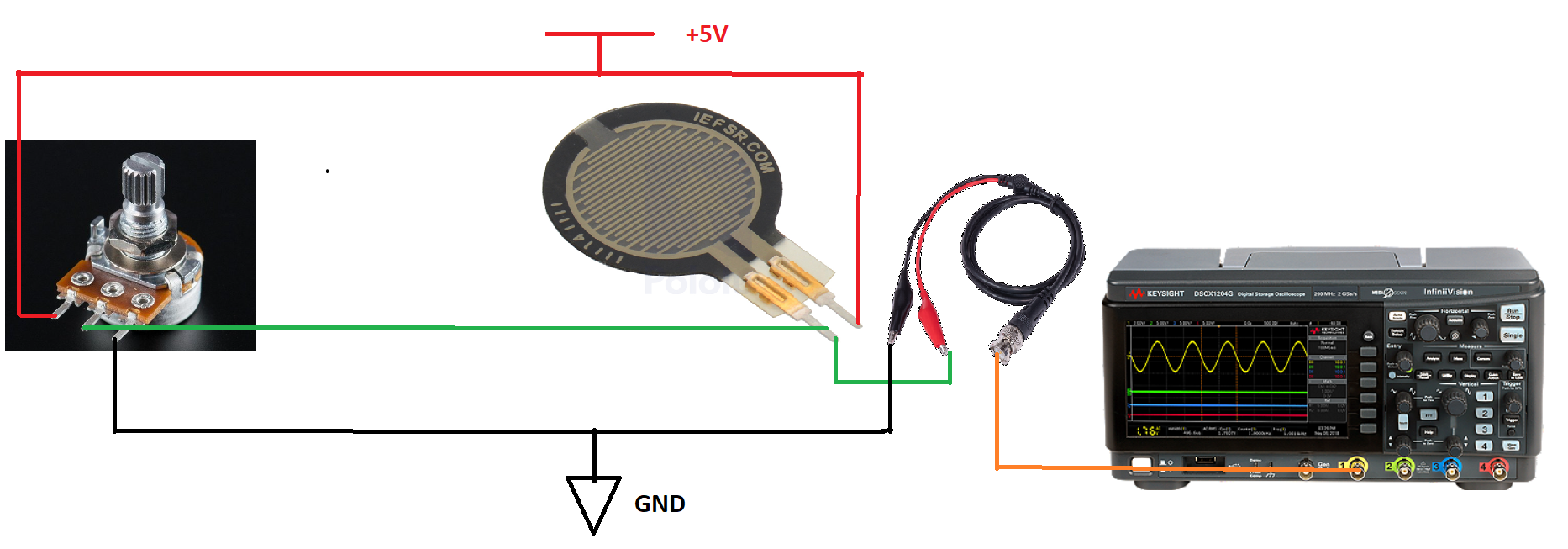
Be as detailed as possible so the individuals or groups can execute the activity on their own as much as they want to. Of course, we will be there to assist/guide, but we want to give them freedom to execute on their own if they enjoy independent thinking.

1. Dispense 8 mL of shampoo –
   1. Control flow rate:
      1. Before connecting anything, turn power supply on and adjust the 0-12V knob to 0V. Then turn off power supply.
      2. Connect DC motor to power supply with power supply off. Red banana cable to red motor lead. Black banana cable to black motor lead.
      3. Turn power supply on, and gradually adjust 0-12V knob clockwise. Log results.
      4. Turn off and attach the pump (plastic housing) and tubing. Dip the inlet tubing into the shampoo and the outlet tubing into the empty graduated cylinder.
      5. Turn power supply on and increase the voltage until the shampoo starts moving through the tubing.
      6. Determine flow rate by timing how long it takes to fill graduated cylinder with 10mL of shampoo. Log results
   2. Control timing:
      1. Empty and wash out graduated cylinder.
      2. Disconnect the positive power supply lead.
      3. Add the switch in series.
      4. Based on flow rate, determine how long to turn pump on in order to get 8mL of shampoo dispensed.
      5. Set timer for partner to turn the switch off to get 8mL of shampoo dispensed.
      6. Have your partner turn her back to the graduated cylinder and turn the switch on, time the designated duration, and then switch the pump off when she thinks 8mL has been dispensed. Log results.
2. Position arm – stepper motor  
   **WARNING: MOTOR SHIELD and STEPPER MOTOR gets hot**

*Do not touch them during this experiment. If you need any assistance with wiring, please let a student facilitator know if you're in need of any help.*

* 1. Control angular position of motor
     1. Connect H-bridge to stepper motor according to wiring diagram in Fig. 1
     2. Connect your laptop to the Pynq board, turn Pynq board on.
     3. Launch Moba xTerm.
     4. Click Session
     5. Click Serial
     6. Select Serial Port that Pynq is connected to (if you’re not sure which one, ask the mentors to help you)
     7. Select 115200 for Speed.
     8. Hit OK
     9. When the terminal (black window) appears, hit enter.
     10. For user: enter “xilinx”
     11. For password: enter “xilinx”
     12. Type ip a
     13. Under eth0, find the address that is a series of 4 numbers separated by “.” – e.g., 192.168.137.176. Copy and paste the address into a web browser
     14. Find and open the Stepper\_motor\_code.ino file.
     15. Run the code and watch the stepper motor. Can you control its angular position?
     16. Log results.
  2. Control linear position of arm
     1. Now, see if you can adjust one number in the code to get the arm jig to be positioned at certain points along the rail: approximately halfway across and ¼ of the way across.
     2. Log results.

1. Scrub – servo motor
   1. View and predict PWM output
      1. Wire servomotor to Pynq board as shown in Fig. 2.
      2. Connect the PWM output to the oscilloscope. In your workbook, sketch the output you predict to see.
      3. Now run servoGo.py and log results.
   2. Adjust the code to make it scrub for 5s. Log results.
   3. Adjust the code to make the arm scrub more of the back (i.e., control the length along back that it scrubs)
   4. Adjust the code to make arm scrub faster. Log results.
2. Feedback control with gripper
   1. Open and close gripper
      1. On your Yoga tablet, open the Arduino app
      2. In C:\LaunchPadBW\ => find and open gripper.ino.
      3. Run through the code and observe the actions of the gripper. The gripper should open and close.
      4. Find the place in the code where the gripper position (or how wide the gripper is open or closed) is controlled.
      5. In your worksheet, plan how you will adjust the code to try to get the gripper to close on the sponge and be able to hold the sponge without squishing the sponge. Then implement the code changes. In order to test out how well you managed to get your controller to hold onto something without squashing it, you will actually use your finger in place of the sponge. When the gripper is wide open, insert your index finger. Determine what position you need the gripper to stop at in order to hold but not squeeze your finger.
      6. Log results.
   2. Adding the FSR: voltage divider output
      1. Wire the voltage divider circuit with the FSR as shown in Fig. 3
      2. Turn on the 5V supply, and then gradually push harder and softer on the FSR. Log results. What happens to the LED as you push harder on the FSR? As you release the FSR?
   3. Get gripper to hold sponge with right amount of pressure
      1. Open gripperWithFSR.py
      2. Adjust the code so that the gripper will stop closing when the FSR senses pressure against the sponge.
      3. Holding the terminal end (not the sensor end), hold the FSR sensor portion against the sponge. Run the code and test whether the gripper is able to automatically stop closing when it reaches the desired pressure. Log results.



**Figure 3**

**Worksheet**

As you are designing your lab, think about how the students can use this workbook to log their observations, draw their design (and re-design), troubleshoot, etc.

1. Dispense 8 mL of shampoo –
2. Control flow rate:
3. What happens as you turn the knob clockwise?

What happens as you turn the knob counter-clockwise?

Why do you think this happens?  
  
What do you think would happen if you swapped the leads?

Try swapping the leads, and describe your results:

1. How long did it take to fill 10mL?  
     
   Calculate the estimated flow rate (include units):
2. Control timing:
3. Based on flow rate, determine how long to turn pump on in order to get 8mL of shampoo dispensed:
4. How much shampoo was actually dispensed?  
     
     
   How much error did you have? If you know how, calculate percent error.  
     
     
   What are some possible causes of this error?
5. Position arm – stepper motor
   1. Controlling angular position of the stepper motor
6. Can you get the motor to go to 180o? How about 45o?  
     
     
     
     
     
     
     
     
   How can we get this type of the motor control the arm’s position along the rail?  
     
     
     
     
     
   1. Control linear position
7. Suppose you want the arm jig exactly halfway across the rail. What do you need to change in the code in order to get the arm jig positioned correctly?  
     
     
     
   After modifying and testing your code, how close was the actual position to the desired position?  
     
     
     
   If you had any errors, what are some possible causes for these errors?
8. Scrub – servo motor
   1. View and predict PWM output
9. Sketch the PWM output you predict to see.
10. What did the PWM output actually look like? How did they compare to your predictions?  
      
      
      
    1. Were you able to control the duration of the scrubbing?
    2. Were you able to control over what length the back was scrubbed?
    3. What is the maximum speed you were able to get the scrubber to go?   
         
         
       What limits this speed?
11. Feedback control with gripper
    1. Open and close gripper
12. How do you need to modify the code to have the gripper stop closing at a desired position?
13. How well is the gripper stopping at the desired amount of pressure?

What drawbacks do you see to this approach of setting a fixed position to hold the sponge?

* 1. Adding the FSR: voltage divider output

1. What do you see on the oscilloscope as you push harder on the FSR?   
     
     
   As you release the FSR?  
     
     
   1. Get gripper to hold sponge with right amount of pressure
2. Were you able to get the gripper to hold the sponge correctly without too much or too little force?