CSC 460

Design and Analysis of Real-time Systems

Project 1

February 4, 2020

Joel Kerfoot – V00855134

Braiden Cutforth – V00853754

Introduction / Problem

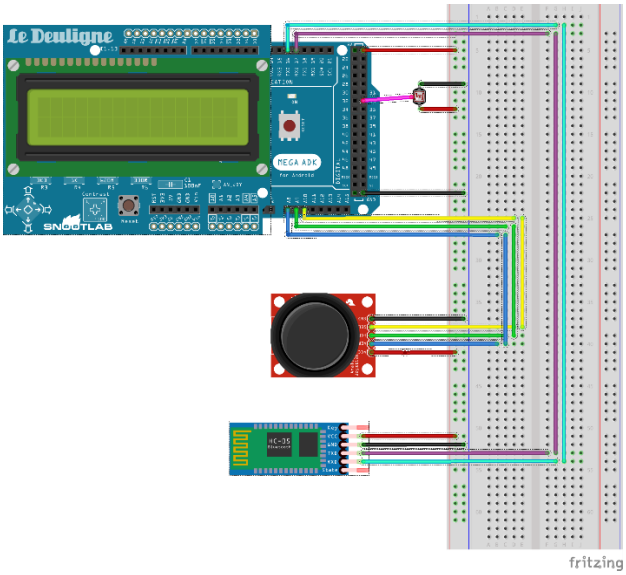
The following report describes in detail the system that was built for Project 1. The goal of the project is to have a base station with a joystick that controls servo motors and a laser over Bluetooth on the remote station. With the controls at hand, the goal is to get the laser on the remote station to point at the photo sensor on the base station. An additional goal is to have the systems use as little CPU time as possible, while maintaining good responsiveness of the controls.

System Overview

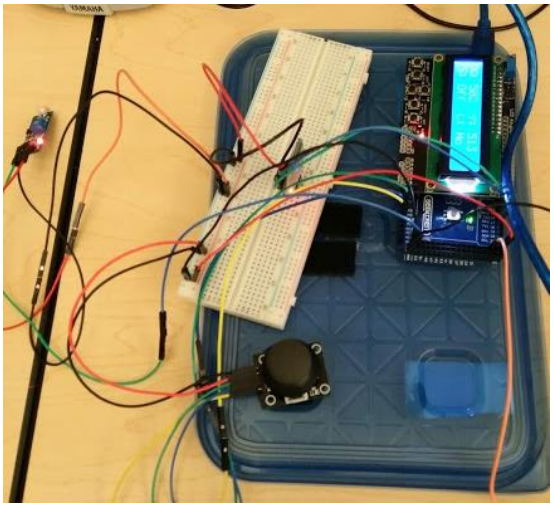
Project 1 uses two Arduino MEGA 2560 boards, one designated as the base station and the other as the remote station. The base station consists of a joystick, light sensor, and an LCD Keypad Shield. The remote station consists of a two servo motors in a pan and tilt kit, and a laser. The base and remote station communicate using a HC-05 Bluetooth chip on each board. The base station sends instructions to the remote station to control the servos and turn the laser on and off. The LCD on the base station displays the state of the joystick and whether the light sensor is being hit by the laser.

Base Station

Wiring Diagram

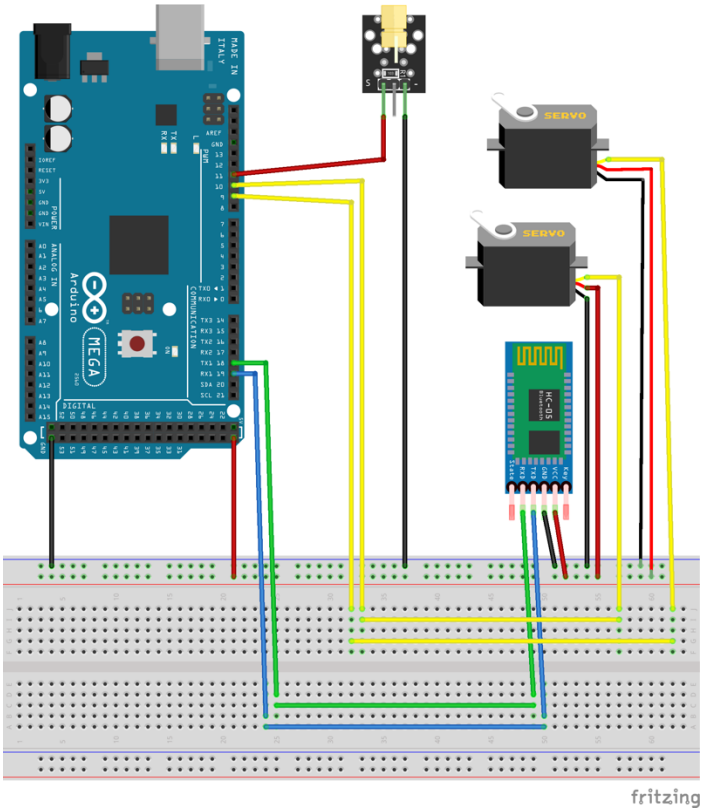


Photo

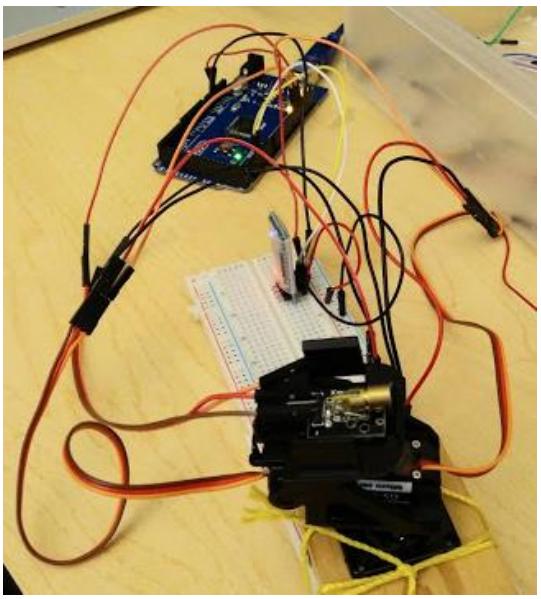


Remote Station

Wiring Diagram



Photo



Joystick

The joystick has three pins VRx, VRy, and SW. The x and y are analog pins that give values ranging from 0 to 1023 with neutral being around (512, 512). The readings are scaled down by 512 so the values range from -512 at the minimum and 512 at the maximum position. The values are exponential averaged by multiplying the current reading by 80% and adding the previous reading multiplied by 20%. The values are then normalized to range from -62 to 62 this combined with exponential averaging provides smoother control over the servos. A dead zone of plus or minus 10 around natural to make it easier for the user to stop the movement of the servos. The switch was plugged into an analog port on the Arduino. When not pressed the Arduino read values over 500 and close to 0 when pressed. The Switch was set up to toggle the laser to make it easier to control with the laser on.

Bluetooth

The base station sends the joystick reading the remote station to control the servo and laser. The data is sent in packets represented using the following syntax:

(<x_control>,<y_corntol>,<laser_on>)

The open parenthesis represents the start of the packet and the close parenthesis the end of the packet. The values are comma delimited where the x control is the first value, followed by the y control, and

finally the laser control. The x and y control are values ranging from -62 to 62 where 0 is not moving and the laser on is a pseudo Boolean value where 1 is on and -1 is off.

Servos

The servos are attached to a pan and tilt kit, with the laser attached to the top. One servo is responsible for the panning motion (horizontal movement) and another is responsible for the tilt motion (vertical movement). The servos are connected by three wires; 5v, ground and pulse control. To control the position of the servo, different PWM values are sent to the pulse control line. These values range from a minimum of 500 microseconds (0.5ms) to 2500 microseconds (2.5ms) with 1500 microseconds (1.5ms) being the neutral positions. It is important to note that sending a pulse width outside of these ranges will cause the servo to try to go beyond its range of motion. If this happens damage may occur to the servo, so it is important to maintain values within the range mentioned above.

Laser

The laser is a simple diode. It is connected to ground and a digital pin on the Arduino. When HIGH is written to the digital pin, the laser will turn on. When LOW is written to the pin, the laser will turn off.

Light Sensor

The light sensor is used to detect if the laser at on the remote station is pointing at it. A digital light sensor was used, connected by 3 pins; ground, 5v and signal. The signal pin was attached to the Arduino. The sensor was configured using the onboard potentiometer, so the sensor returns 0 when there is light and 1 otherwise on the signal pin.

LCD

The LCD displays the readings taken from the joystick and light sensor. The topline displays the x and y values output from the joystick, i.e. values ranging from 0 to 1023. The bottom line displays whether the switch has been toggled on or off and the whether the light sensor is detecting the laser or not. Each time the display is updated to screen is cleared so the new values can be displayed with minimal cursor manipulation.

Schedulers

The time triggered scheduler on the base station has three tasks getting the joystick controls, detecting the laser, and updating the display. Getting the joystick readings and detecting the laser is scheduled every 100ms. 10 measurements per seconds provided a responsive enough system for human operators without wasting unnecessary amounts of CPU time. If the system was computer controlled more readings it would be beneficial to take more readings to have more precise control over the servos. The LCD was scheduled to be updated every 200ms because it wasn't critical to display the exact measurement at that time. It also makes it easier to read when the values aren't updated as frequently.

The time triggered schedule on the base station had one scheduled task. It would read from the Bluetooth device every 100ms, parse the values and immediately update the values for the servo. The 100ms timing was decided based on the frequency of data being sent from the base station. It made sense to schedule everything together as the continuous task of the remote station was just to receive data and update the servos/laser.

Measurements

Base Station

On the base station there are larger spikes when the CPU isn't idle every 200ms. This is when the LCD is being updated. The 100ms increment in between where only the joystick and light sensor readings are being updated contribute negligible CPU time compared to updating the display. For a 200ms period there is 13ms where the CPU isn't idle this works out to the CPU on the base station being idle 94% of the time. See figure 1 and 2 below for the measurements taken from the logic analyzer

Figure 1 – Base Station Fine Measurements

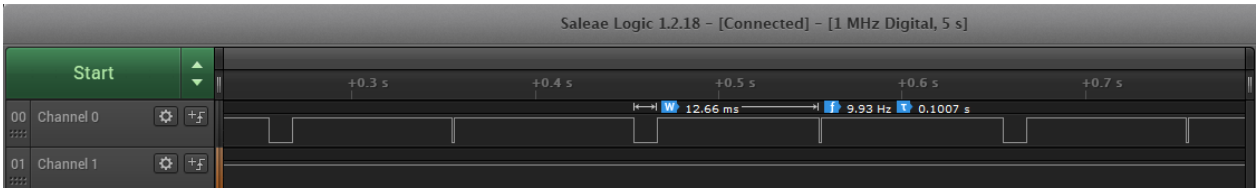


Figure 2 – Base Station Coarse Measurements

Remote Station

On the remote station, the timing diagram is quite simple. Once every 100ms the signal goes low for ~1ms. This is the CPU performing the read from the Bluetooth and updating the servo motors. Based on the timing diagram, we can see the CPU sits idle for approximately 99% of the time.

Figure 3 – Remote Station Fine Measurements

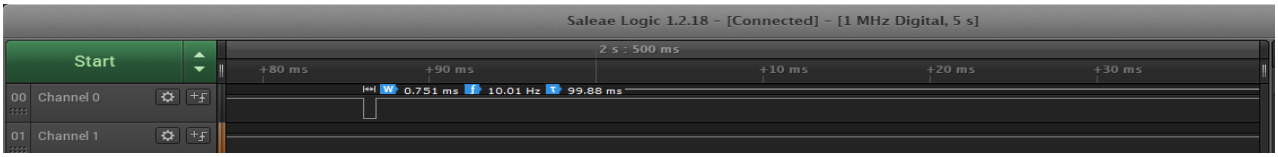
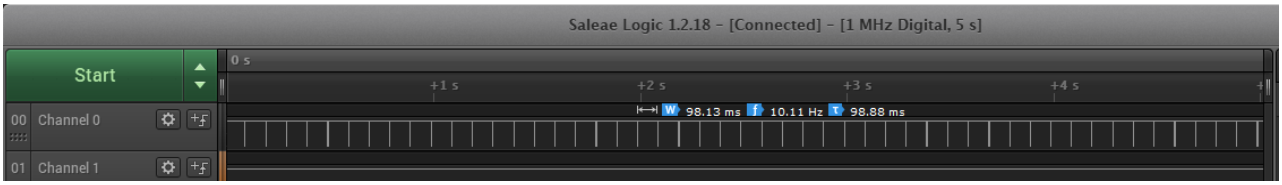


Figure 4 – Remote Station Coarse Measurments



Testing

During the project, all separate components were tested individually to ensure they worked, that we were able to wire them correctly and that we were able to control them as expected.

Bluetooth

The Bluetooth module was tested first by setting up one device to receive data and print to the serial monitor and another device to take input from the serial monitor and send it over Bluetooth. This allowed us to verify that the devices were properly paired and wired. In addition, we were able to test small bits of code for reading and writing from the Bluetooth devices.

Joystick

The joystick was tested by wiring it up, reading the raw analog values and outputting them to the serial monitor. This allowed us to see the ranges of values it produced, see the neutral values and determine a dead zone around the neutral position. The dead zone was an area around neutral where the values read were inconsistent. Overall, the testing lead to more accurate readings from the joystick.

Servos

Before testing the servos, we used the logic analyzer to confirm that the range of pulse widths that were being sent over the control wire were within the appropriate range. To do this, instead of wiring the control wire to the servo, wire it to a channel on the logic analyzer. Use the same servo code and analyze the different values that your code produces. Once it is verified that values outside of the servo limits aren't being sent, then connect the servo. Now the servo should respond to the different pulse widths and move accordingly.

Laser

The laser was tested like any LED. Attaching the ground and 5v pins appropriately, the laser turned on. Laser control was tested by wiring the 5v pin to a digital pin on the Arduino and writing HIGH and LOW (on and off).

Light Sensor

The light sensor was tested in ambient room lighting conditions of the lab. The photoresistor on the breakout board was used. When the sensor is connected to power and ground and it is detecting light, a green LED is illuminated on the board. We adjusted an onboard resistor such that the light was off in ambient lighting conditions. Then we adjusted it slightly so the light would illuminate when the laser was on the photo sensor. Once this was complete, a digital read from the third pin would read 0 when the laser was on the photo sensor and a 1 otherwise.

LCD

The LCD was tested simply by attaching the shield to the Arduino, configuring the pins with the library code and initially writing 'Hello World!' to the display. Then through the use of other library functions different things were able to be accomplished (clearing the display, moving the cursor etc.)

Conclusion

The system in Project 1 worked by having two Arduinos communicating over Bluetooth. The base station took reading from the joystick and sent it as a tuple to the remote station which parsed the values to control the pan and tilt servos and the laser. The base station the detected the presence of the laser and output the state of the joystick and light sensor to the LCD display.