Kyle Coppedge  
Dr. Bigelow  
12Apr2018

ENGR 1242, Engineering Fundamentals, Project Report 9

1. **Functional Objectives:**
   1. **The robot should be programmable**
      1. **Constraints**
         1. **Layout**

The PIC must be mounted on a small breadboard.

* + - 1. **Components**

The PIC must receive its power from an 8 V Power Supply.

* + - 1. **Time**

The project must be completed by 5:20PM on Jan 11th.

* + 1. **Test Plan 1 and results**
       1. **Setup**

Connect the PIC Kit 2 and program the PIC.

* + - 1. **Test**

Open the UART Tool and press the reset button.

* + - 1. **Results**

“ECE Rules!” was displayed on the UART Tool. Completed at 3:15PM on Jan 11th.

* + 1. **Statement of success**

The project is successful after the objectives have been met within the constraints.

* 1. **The robot should be programmable and communicate with user.**
     1. **Constraints**
        1. **Layout**

An LED on left and right on their own pins.

* + - 1. **Components**

The PIC must receive its power from a 8 V Power Supply.

* + - 1. **Time**

The project must be demo’d by 5:20PM on Jan 18th.

* + - 1. **Operation**

LEDs should turn on and off independently and the UART should indicate light on or off.

* + 1. **Test Plan 2 and results**
       1. **Setup**

Hook up power (8V to bread board)

Hook up PICKIT

Open UART

* + - 1. **Test**

Press the reset button.

5x-The robot should turn on the left LED then off.

5x-The robot should turn on the right LED then off.

* + - 1. **Results**

The LEDs blinked 5 times. Completed at 3:22PM Jan 18th.

* + 1. **Statement of success**

The project is successful after the objectives have been met within the constraints.

* 1. **The robot should be mobile.**
     1. **Constraints**
        1. **Power**

7.2V Battery with 5A Fuse, 7.2V to drivers has a switch, 7.2V to PIC Regulator. PIC has a reset Switch

* + - 1. **Movement:**

Must use the sequence: FFF RRR FFFF L RRRRR LL.

* + - 1. **Time**

The project must be demo’d by 5:20PM on Feb 22nd.

* + - 1. **Operation**

Four Functions: Forward 9 Inches, backwards 9 Inches, turn right 30 Degrees, turn left 30 Degrees.

* + 1. **Test Plan 5 and results**
       1. **Setup**

Hook up Battery and turn on motors

Download program

Set in course

Hit reset

* + - 1. **Test**

The robot will follow the sequence: FFF RRR FFFF L RRRRR LL and end up within 9 inches of where it started.

* + - 1. **Results**

The sequence was completed at 3:55PM on Feb 15th.

* + 1. **Statement of success**

The project is successful after the objectives have been met within the constraints.

* 1. **The robot should be responsive to obstacles.**
     1. **Requirements**
        1. **Components:**

Use a 7402 IC as a latch to capture bumper events.

* + - 1. **Operation:**

Navigate from one side of the course to the other.

* + 1. **Test Plan 7 and results**
       1. **Setup**

Download program

Hook up Battery and turn on motors

Set in course

Hit reset

* + - 1. **Test**

The robot will start in one half of the course and make it to the other in 1 minute.

* + - 1. **Results**

The sequence was completed at 5:15PM on Mar 6th.

* + 1. **Statement of success**

The project is successful after the objectives have been met within the constraints.

* 1. **The robot should be responsive to light.**
     1. **Requirements**
        1. **Components:**

Use two photo resistors to capture light levels

No cutting of photo resistors

* + - 1. **Operation:**

Move to light target without hitting the wall.

* + 1. **Test Plan 8 and results**
       1. **Setup**

Download program

Hook up Battery and turn on motors

Set in course

Hit reset

* + - 1. **Test**

The robot will make it to under the half circle of light

* + - 1. **Results**

The sequence was completed at 3:35PM on Mar 15th.

* + 1. **Statement of success**

The project is successful after the objectives have been met within the constraints.

* 1. **The robot will be able to complete the mission. Integrating all five major functions.**
     1. **Requirements**
        1. **Course:**

Begin at doorway of PEC105 within the masking tape, directly facing the light.

Traverse the obstacle course.

Stop within the masked area surrounding the beacon at least 2 out of 3 times at demonstration time.

* + - 1. **Operation:**

Move to light target without hitting the wall.

* + 1. **Test Plan 9 and results**
       1. **Setup**

Download program

Hook up Battery and turn on motors

Set in course

Hit reset

* + - 1. **Test**

The robot will make it to under the half circle of light at least 2 out of 3 times.

* + - 1. **Results**

The sequence was completed at 4:00PM on April 4th, 2018.

* + 1. **Statement of success**

The project is successful after the objectives have been met within the constraints.

1. **Hardware Design:**
   1. **Hardware System Overview**
      1. **System Block Diagram**

****

* + 1. **Subsystem Descriptions**
       1. PIC: main microcontroller with OS.
       2. Left LED: led on the left which the pic controls.
       3. Right LED: led on the right which the pic controls.
       4. Test Pin 3: puts the robot in test motor mode
       5. Test Pin 4: puts the robot in test bumper mode
    2. **Signal Descriptions** 
       1. redLED: turns it on or off.

|  |  |
| --- | --- |
| **redLED** | **result** |
| **0** | **Off** |
| **1** | **On** |

* + - 1. greenLED: turns it on or off.

|  |  |
| --- | --- |
| **greenLED** | **Result** |
| **0** | **Off** |
| **1** | **On** |

* + - 1. leftMotorCW: turns it clockwise or off.

|  |  |
| --- | --- |
| **leftMotorCW** | **Result** |
| **0** | **Off** |
| **1** | **On (clockwise)** |

* + - 1. leftMotorCCW: turns it counter clockwise or off.

|  |  |
| --- | --- |
| **leftMotorCCW** | **Result** |
| **0** | **Off** |
| **1** | **On (counter clockwise)** |

* + - 1. rightMotorCW: turns it clockwise or off.

|  |  |
| --- | --- |
| **rightMotorCW** | **Result** |
| **0** | **Off** |
| **1** | **On (clockwise)** |

* + - 1. rightMotorCCW: turns it counter clockwise or off.

|  |  |
| --- | --- |
| **rightMotorCCW** | **Result** |
| **0** | **Off** |
| **1** | **On (counter clockwise)** |

* + - 1. testPin3: turns motorTest on or off.

|  |  |
| --- | --- |
| **testPin3** | **Result** |
| **0** | **Off (runOS)** |
| **1** | **On (testSuite)** |

* + - 1. testPin4: turns bumperTest

|  |  |
| --- | --- |
| **testPin4** | **Result** |
| **0** | **Off** |
| **1** | **On** |

* + - 1. resetLatch: turns it on or off.

|  |  |
| --- | --- |
| **resetLatch** | **Result** |
| **0** | **Off** |
| **1** | **On (reset)** |

* + - 1. leftBumper: turns it on or off.

|  |  |
| --- | --- |
| **leftBumper** | **Result** |
| **0** | **Off** |
| **1** | **On (collision)** |

* + - 1. rightBumper: turns it on or off.

|  |  |
| --- | --- |
| **leftBumper** | **Result** |
| **0** | **Off** |
| **1** | **On (collision)** |

* 1. **Circuit Diagrams**
     1. **Power System**

****

* + 1. **PIC Configuration**

****

* + 1. **Pinouts**
       1. **Voltage Regulator**



* + - 1. **MOSFET**

****

* + - 1. **BJT**

****

* + - 1. **NOR Gate**



* + 1. **LED Circuits**
       1. **redLED**



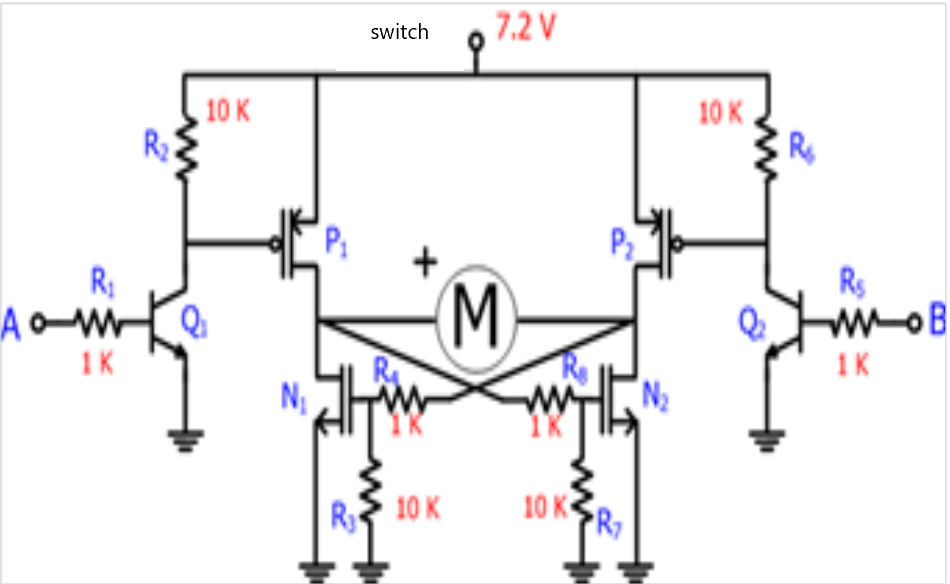
* + - 1. **greenLED**



* + 1. **Test Pins**



* + 1. **Motor Drivers**

****

* + 1. **Bumper System**

****

* + 1. **Light Detection System**

****

1. **Software Design**
   1. **Software System Overview:**
      1. **High-Level Description**

The Main is going to check which mode you would like to run.

The Operating System is interested in redirecting itself around obstacles forever.

The Test Suite is there to test the motors in both directions, and to test both bumpers.

* + 1. **Pin Definitions**

|  |  |  |
| --- | --- | --- |
| **PIC Pin** | **Variable name** | **Description** |
| 3 | testPin3 | Toggle’s test systems and main OS |
| 4 | testPin4 | Toggle’s test systems and main OS |
| 5 | redLED | This is a red LED |
| 6 | greenLED | This is a green LED |
| 7 | rightBumper | This is the right bumper |
| 10 | resetLatch | This is the bumper memory reset |
| 11 | Pin11Timer | This is the right Photo Resistor result |
| 12 | Pin12Timer | This is the left Photo Resistor result |
| 14 | leftBumper | This is the left bumper |
| 23 | leftMotorCW | This is the left motor |
| 24 | leftMotorCCW | This is the left motor |
| 25 | rightMotorCW | This is the right motor |
| 26 | rightMotorCCW | This is the right motor |

* + 1. **Code Listing of definitions.h**

|  |
| --- |
| #define ON 1  #define OFF 0  #define redLED digOutput5  #define greenLED digOutput6  #define testPin3 digInput3  #define testPin4 digInput4  #define leftMotorCW digOutput23  #define leftMotorCCW digOutput24  #define rightMotorCW digOutput25  #define rightMotorCCW digOutput26  #define leftBumper digInput14  #define rightBumper digInput7  #define resetLatch digOutput10  void resetSRLatch();  void setupPins();  void turngreenLEDOn();  void turngreenLEDOff();  void turnRedLEDOn();  void turnRedLEDOFF();  void runRobotOS();  void runTestSuite();  void runMotorTest();  void runBumperTest();  void runLightTest();  void moveForward();  void moveBackward();  void turnRight();  void turnLeft();  void theSequence();  void turnLEDsOff();  void checkBumpers(); |

* + 1. **Detailed Function Descriptions**

|  |  |
| --- | --- |
| Name: | Main() |
| Purpose: | Start program and decide whether to run the test suite or the operating system. |
| Calls: | initializeUART()  setupPins()  pause()  runRobotOS()  runTestSuite()  halt() |
| Code: | initializeUART();  pause(1000);  setupPins();  printf("I'm in the main!\n");  if(testPin3)  printf("test pin 3 \n");  if(testPin4)  printf("test pin 4 \n");  if(testPin3||testPin4)  runTestSuite();  else  runRobotOS();    halt();  return(0); |
|  |  |

|  |  |
| --- | --- |
| Name: | setupPins() |
| Purpose: | Define the pins to a signal type and direction. |
| Calls: | None. |
| Code: | pin3Direction = INPUT; //dip switch  pin4Direction = INPUT; //dip switch  pin5Direction = OUTPUT; //the red LED  pin6Direction = OUTPUT; //the green LED  pin7Direction = INPUT; // RIGHT BUMPER  pin10Direction = OUTPUT; // RESET BUMPER  //pin11Direction right P.R. toggled in checkLight  //pin12Direction left P.R. toggled in checkLight  pin14Direction = INPUT; // LEFT BUMPER  pin23Direction = OUTPUT; //motor  pin24Direction = OUTPUT; //motor  pin25Direction = OUTPUT; //motor  pin26Direction = OUTPUT; //motor  pin3Type = DIGITAL; //dip switch  pin4Type = DIGITAL; //dip switch  pin5Type = DIGITAL; //the red LED  pin6Type = DIGITAL; //the green LED  pin7Type = DIGITAL; // RIGHT BUMPER  //pin10Type = DIGITAL; // RIGHT BUMPER  //pin14Type = DIGITAL; // LEFT BUMPER  pin23Type = DIGITAL; //motor  pin24Type = DIGITAL; //motor  pin25Type = DIGITAL; //motor  pin26Type = DIGITAL; //motor |
|  |  |

|  |  |
| --- | --- |
| Name: | runRobotOS() |
| Purpose: | Run the robot operating system, instead of the test suite. |
| Calls: | //theSequence()  moveForward()  checkBumpers()  checkLight()  turnLeft()  turnRight()  halt() |
| Code: | printf("The robot is up and running!!\n");  //theSequence(); // FFF RRR FFFF L RRRRR LL  while(1)  {  int i = 0;  for(i = 0; i < 3; i++)  {  moveForward();  }  checkBumpers();  if(checkLight() == 1)  turnLeft();  else  turnRight();    pause(transitionPause);  }  printf("Robot will shut down... \n");  halt(); |
|  |  |

|  |  |
| --- | --- |
| Name: | turnRedLEDOn() |
| Purpose: | Turns the red led on. |
| Calls: | None. |
| Code: | printf("The red LED is on.\n");  redLED = ON; |
|  |  |

|  |  |
| --- | --- |
| Name: | turnRedLEDOff() |
| Purpose: | Turns the red led off. |
| Calls: | None. |
| Code: | printf("The red LED is off.\n");  redLED = OFF; |
|  |  |

|  |  |
| --- | --- |
| Name: | turnGreenLEDOn() |
| Purpose: | Turns the green led on. |
| Calls: | None. |
| Code: | printf("The green LED is on.\n");  greenLED = ON; |
|  |  |

|  |  |
| --- | --- |
| Name: | turnGreenLEDOff() |
| Purpose: | Turns the green led off. |
| Calls: | None. |
| Code: | printf("The green LED is off.\n");  greenLED = OFF; |
|  |  |

|  |  |
| --- | --- |
| Name: | runTestSuite() |
| Purpose: | Decide which test to run |
| Calls: | None. |
| Code: | printf("I am in the test suite\n");  if(testPin3&&!testPin4)  runMotorTest();  else if(!testPin3&&testPin4)  runBumperTest();  else  runLightTest(); |
|  |  |

|  |  |
| --- | --- |
| Name: | runMotorTest() |
| Purpose: | To make sure each motor is functioning. |
| Calls: | turnLeftMotorForward()  turnRightMotorForward()  turnLeftMotorBackward()  turnRightMotorBackward()  turnLEDsOnIfMoving()  motorsOff()  pause() |
| Code: | printf("I'm in the motor test.\n");  motorsOff();    turnLeftMotorForward();  turnLEDsOnIfMoving();  pause(100);  motorsOff();  pause(500);    turnRightMotorForward();  turnLEDsOnIfMoving();  pause(100);  motorsOff();  pause(500);    turnLeftMotorBackward();  turnLEDsOnIfMoving();  pause(100);  motorsOff();  pause(500);    turnRightMotorBackward();  turnLEDsOnIfMoving();  pause(100);  motorsOff(); |
|  |  |

|  |  |
| --- | --- |
| Name: | runBumperTest() |
| Purpose: | To check if the bumpers work. |
| Calls: | None. |
| Code: | printf("I'm in the bumper test.\n");  while(1)  {  if(leftBumper)  {  printf("the left switch is closed\n");  turnRedLEDOn();  }    if(rightBumper)  {    printf("the right switch is closed\n");  turnGreenLEDOn();  }  if(leftBumper || rightBumper)  {  resetSRLatch();  turnLEDsOff();  }  //pause(1000);  } |
| Flowchart: |  |

|  |  |
| --- | --- |
| Name: | runLightTest() |
| Purpose: | To check if the light sensors work. |
| Calls: | None. |
| Code: | printf("I'm in the light test.\n"); |
| Flowchart: |  |

|  |  |
| --- | --- |
| Name: | theSequence() |
| Purpose: | Perform the required sequence in project 5 |
| Calls: | motorsOff()  moveForward()  turnRight()  moveBackward()  turnLeft() |
| Code: | motorsOff();  int i = 0;  for(i = 0; i<3; i++) // F F F  moveForward();    motorsOff();  pause(transitionPause);    for(i = 0; i<3; i++) // R R R  turnRight();    motorsOff();  pause(transitionPause);    for(i = 0; i<4; i++) // F F F F  moveForward();    motorsOff();  pause(transitionPause);    turnLeft(); // L  motorsOff();  pause(transitionPause);    for(i = 0; i<5; i++) // R R R R R  moveBackward();    motorsOff();  pause(transitionPause);    for(i = 0; i<2; i++) // L L  turnLeft();    motorsOff(); |
|  |  |

|  |  |
| --- | --- |
| Name: | moveForward() |
| Purpose: | Move the robot forward 9 inches |
| Calls: | turnLeftMotorForward()  turnRightMotorForward()  turnLEDsOnIfMoving()  turnLEDsOff() |
| Code: | printf("I am moving forward\n");  turnLeftMotorForward();  pause(leftOnMs);  turnRightMotorForward();  turnLEDsOnIfMoving();  pause(forwardDelay);  turnLEDsOff();  motorsOff();  pause(movePause); |
|  |  |
| Name: | moveBackward() |
| Purpose: | Move the robot backward 9 inches |
| Calls: | turnLeftMotorBackward()  turnRightMotorBackward()  turnLEDsOnIfMoving()  turnLEDsOff() |
| Code: | printf("I am moving backward\n");  turnRightMotorBackward();  pause(rightOnMs);  turnLeftMotorBackward();  turnLEDsOnIfMoving();  pause(backwardDelay);  turnLEDsOff();  motorsOff();  pause(movePause); |
|  |  |

|  |  |
| --- | --- |
| Name: | turnRight() |
| Purpose: | Turn right 30 Degrees |
| Calls: | turnLeftMotorForward()  turnLEDsOnIfMoving()  turnLEDsOff() |
| Code: | printf("I am pivoting right\n");  turnLeftMotorForward();  turnLEDsOnIfMoving();  pause(turnRightDelay);  turnLEDsOff();  motorsOff();  pause(movePause);); |
|  |  |

|  |  |
| --- | --- |
| Name: | turnLeft() |
| Purpose: | Move the robot forward 9 inches |
| Calls: | turnRightMotorForward()  turnLEDsOnIfMoving()  turnLEDsOff() |
| Code: | printf("I am pivoting left\n");  turnRightMotorForward();  turnLEDsOnIfMoving();  pause(turnLeftDelay);  turnLEDsOff();  motorsOff();  pause(movePause); |
|  |  |

|  |  |
| --- | --- |
| Name: | motorsOff() |
| Purpose: | To turn all the motors off with one call |
| Calls: | None. |
| Code: | printf("motorsOff\n");  leftMotorCW = OFF;  leftMotorCCW = OFF;  rightMotorCW = OFF;  rightMotorCCW = OFF;  turnLEDsOff(); |
| Flowchart: |  |

|  |  |
| --- | --- |
| Name: | turnLEDsOnIfMoving() |
| Purpose: | To turn on the LEDs if the motor is on |
| Calls: | turnGreenLEDOn()  turnRedLEDOn() |
| Code: | if(leftMotorCCW == ON || leftMotorCW == ON)  turnGreenLEDOn();//left    if(rightMotorCW == ON || rightMotorCCW == ON)  turnRedLEDOn();//right |
| Flowchart: |  |

|  |  |
| --- | --- |
| Name: | turnLEDsOff() |
| Purpose: | Turn all the LEDs off with one call |
| Calls: | None. |
| Code: | turnRedLEDOff();//left  turnGreenLEDOff();//right |
| Flowchart: |  |

|  |  |
| --- | --- |
| Name: | turnLeftMotorForward() |
| Purpose: | Turn the left motor forward until turned off |
| Calls: | None. |
| Code: | printf("leftMotorForward\n");  leftMotorCW = ON; |
| Flowchart: |  |

|  |  |
| --- | --- |
| Name: | turnLeftMotorBackward() |
| Purpose: | Turn the left motor backward until turned off |
| Calls: | None. |
| Code: | printf("leftMotorBackward\n");  leftMotorCCW = ON; |
| Flowchart: |  |

|  |  |
| --- | --- |
| Name: | turnRightMotorForward() |
| Purpose: | Turn the right motor forward until turned off |
| Calls: | None. |
| Code: | printf("rightMotorForward\n");  rightMotorCW = ON; |
| Flowchart: |  |

|  |  |
| --- | --- |
| Name: | turnRightMotorBackward() |
| Purpose: | Turn the right motor backward until turned off |
| Calls: | None. |
| Code: | printf("rightMotorBackward\n");  rightMotorCCW = ON; |
| Flowchart: |  |

|  |  |
| --- | --- |
| Name: | checkBumpers() |
| Purpose: | Check if any bumpers have been pushed |
| Calls: | None. |
| Code: | int i = 0;  int leftReaction = 0, rightReaction = 0;  if(leftBumper && rightBumper)  {  leftReaction = 1;  rightReaction = 1;  }  else if(leftBumper)  {  leftReaction = 1;  }  else if (rightBumper)  {  rightReaction = 1;  }  else  {  pause(10);  }  if(leftReaction && !rightReaction)  {  pause(movePause);  moveBackward();  pause(movePause);  turnLeft();  pause(movePause);  moveForward();    }  if(rightReaction && !leftReaction)  {    pause(movePause);  moveBackward();  pause(movePause);  turnRight();  pause(movePause);  moveForward();  }  if(leftReaction && rightReaction)  {    pause(movePause);  moveBackward();  pause(movePause);  turnLeft();  pause(movePause);  turnLeft();  pause(movePause);  turnLeft();  pause(movePause);  moveForward();  }  pause(10);  if(leftReaction || rightReaction)  {  resetSRLatch();    }  pause(movePause); |
| Flowchart: |  |

|  |  |
| --- | --- |
| Name: | resetSRLatch () |
| Purpose: | Turn the right motor forward until turned off |
| Calls: | None. |
| Code: | printf("the latch is reset\n");  resetLatch = ON;  pause(20);  resetLatch = OFF; |
| Flowchart: |  |

|  |  |
| --- | --- |
| Name: | Int checkLight () |
| Purpose: | Determine if light is greater on left or right |
| Calls: | startPinTimer11()  startPinTimer12()  readPinTimer11()  readPinTimer12()  halt() |
| Code: | int result = 0;  int leftCount = 0, rightCount = 0;  int rightLight, leftLight;  int i = 0;  for(i = 0; i < 3; i++)  {  rightLight = 9000;  leftLight = 9000;  pin11Direction = OUTPUT; // make this output  pin12Direction = OUTPUT;  digOutput11 = 1; // ready the RC Circuit!  digOutput12 = 1;  pause(5); // wait for it?.  pin11Direction = INPUT; // Go! Change to input pin type  pin12Direction = INPUT;  startPinTimer11(); // start the timer  startPinTimer12();  pause(5); // let the capacitor charge  /\* Print the values.. Just use them as integers \*/  rightLight = readPinTimer11();  leftLight = readPinTimer12();    if(rightLight > leftLight)  leftCount++;  else if(leftLight > rightLight)  rightCount++;  if(leftLight < 290 || rightLight < 290)  halt();    printf("Right Light: %d\r\n", readPinTimer11());  printf("Left Light: %d\r\n", readPinTimer12());  printf(" \r\n");  //pause(1000); // wait one second to slow down the UART Display  }  if(leftCount > rightCount)  result = 1;  else if(rightCount < leftCount)  result = 2;  return result; |
| Flowchart: |  |

|  |  |
| --- | --- |
| Name: | runLighTest() |
| Purpose: | Report the light levels to the UART |
| Calls: | None. |
| Code: | printf("I'm in the light test.\n");  while(testPin3 && testPin4)  {  printf("i'm in the loop\n");  pin11Direction = OUTPUT; // make this output  pin12Direction = OUTPUT;  digOutput11 = 1; // ready the RC Circuit!  digOutput12 = 1;  pause(5); // wait for it?.  pin11Direction = INPUT; // Go! Change to input pin type  pin12Direction = INPUT;  startPinTimer11(); // start the timer  startPinTimer12();  pause(5); // let the capacitor charge  /\* Print the values.. Just use them as integers \*/  printf("Right Light: %d\r\n", readPinTimer11());  printf("Left Light: %d\r\n", readPinTimer12());  printf(" \r\n");  pause(1000); // wait one second to slow down the UART Display  } |
| Flowchart: |  |