This in-class activity includes (3) problems.

|  |  |
| --- | --- |
| **Purpose of this in-class activity** | As discussed in A09, the Taum Sauk Reservoir is relied upon by many in Southwest Missouri to manage the electric load on the power grid. This activity is to utilize the TI Kits to create a real-world warning system for the Taum Sauk Reservoir and analyze the random error inherit to such a system. |

**Relevant Course Resources**:

|  |  |
| --- | --- |
| Pre-Class Videos | * None yet |
| Course Resources | * Getting Started with Energia and the TI Kits * Block Diagram Basics |
| Lecture Slides | * Class ## Slides * A09 |

|  |
| --- |
| **Communication Errors between the TI Kits and Your Computer:**   1. Make sure that your TI Kit is plugged in correctly using the provided Micro USB to USB A cable. There should be a green power LED lit up when this occurs. 2. The COM port selected in Energia is not the correct port. While it is generally the highest port number available. This is not always the case. All COM Ports should be tried if there are communication issues. 3. Ensure all drivers have been installed. For instructions on how to install, please see Step 2 in [Getting Started](https://engineering.purdue.edu/fye_i2i/ti-kit-activities-getting-started/). 4. Restart Energia IDE 5. Restart your computer (Should not be required, but can help depending on your computer’s settings)   **Submission Instructions:**   1. Re-name your answer sheet as, **ENGR131\_ICA06A\_*Team##*.docx**, where *team##* is your assigned ENGR 131 team. 2. Save your files to your **Purdue Career Account** (This is your Purdue storage space. For more information see <https://www.itap.purdue.edu/connections/careeraccount>) 3. Submit your work through the designated **Brightspace In-Class Activity Drop Box at** [https:/purdue.brightspace.com/](https://mycourses.purdue.edu/) |

|  |  |
| --- | --- |
| **Task 3a** | **Configuring the Ultrasonic Range Sensor** |
| **Goal** | This problem tests your ability to obtain a real time readout on the four-digit digital display from the Grove Ultrasonic Range Sensor. To do this, you will need to complete the following:   1. Open the Sketchbook/ENGR131/Activity 3 – Tom Sauk/Task3a\_ultrasonic\_ranger file      1. Before compiling and uploading the sketch, first you need to add to the board.    1. Connect the Grove Starter Kit Ultrasonic Ranger Sensor to J15 of the Boosterpack using the four-prong connector cable      * 1. Connect the Grove Starter Kit 4 Digit Digital Display to J13 of the Boosterpack using the four-prong connector cable      * 1. Connect the Boosterpack underneath the TI Kit board. |
| **Solution: Block Diagram** |  |
| **Solution: Picture** |  |
| **Reference: CODE** | /\*  Ultrasonic-Ranger  Measure the distance to obstacles in front and display the value on  the Grove 4-Digit Display    The circuit:  \* 4-Digit Display attached to Pin 38 and 39 (J14 plug on Grove Base BoosterPack)  \* Ultrasonic Ranger attached to Pin 24 (J6 plug on Grove Base BoosterPack)    \* Note:      This example code is in the public domain.    http://www.seeedstudio.com/depot/Grove-Ultrasonic-Ranger-p-960.html  \*/    #include "TM1637.h"  #include "Ultrasonic.h"  /\* Macro Define \*/  #define CLK 40 /\* 4-Digit Display clock pin \*/  #define DIO 39  #define BLINK\_LED RED\_LED /\* 4-Digit Display data pin \*/  #define ULTRASONIC\_PIN 38 /\* pin of the Ultrasonic Ranger \*/  /\* Global Variables \*/  TM1637 tm1637(CLK, DIO); /\* 4-Digit Display object \*/  Ultrasonic ultrasonic(ULTRASONIC\_PIN); /\* Ultrasonic Ranger object \*/  int distance = 0;  /\*obstacles in front \*/  int blink\_internal = 0; /\*variable to store the distance to obstacles in front \*/  int8\_t bits[4] = {0}; /\* array to store the single bits of the value \*/  /\* the setup() method runs once, when the sketch starts \*/  void setup()  {  /\* Initialize 4-Digit Display \*/  tm1637.init();  tm1637.set(BRIGHT\_TYPICAL);  pinMode(RED\_LED, OUTPUT);  }  /\* the loop() method runs over and over again \*/  void loop()  {  distance = ultrasonic.MeasureInCentimeters(); /\* read the value from the sensor \*/    memset(bits, 0, 4); /\* reset array before we use it \*/  for(int i = 3; i >= 0; i--)  {  /\* Convert the value to individual decimal digits for display \*/  bits[i] = distance % 10;  distance = distance / 10;  tm1637.display(i, bits[i]); /\* display on 4-Digit Display \*/  }  delay(100);  } |

|  |  |
| --- | --- |
| **Task 3b** | **Modifying Code using if-else statement** |
| **Goal** | To introduce the concept of an if-else statement in code, in order to modify the ultrasonic ranger to be used as a warning system for the Taum Sauk Reservoir. If-else statements are a type of conditional statement like “If there is peanut butter in the cupboard, I will have a peanut-butter and jelly sandwich. If not, I will have ham and cheese.” This can be applied to this problem by having the ultrasonic ranger measure the distance from the top of the reservoir to the water surface and if it is within a certain threshold (15cm), then to set off a warning using the buzzer. To do this, you will need to complete the following:   1. Open the Sketchbook/ENGR131/Activity 3 – Tom Sauk/Task3a\_ultrasonic\_ranger file    1. Add the Highlighted Red Text from the example code below to this file. 2. Before compiling and uploading the sketch, first you need to add to the board.    1. Starting with the same set up as before, add the Grove Starter Kit Buzzer to J16 of the Booster Pack using the four-prong connector cable. |
| **Solution: Serial Monitor Plot** |  |
| **Reference: CODE** | Ultrasonic-Ranger Measure the distance to obstacles in front and display the value on the Grove 4-Digit Display The circuit: \* 4-Digit Display attached to Pin 38 and 39 (J14 plug on Grove Base BoosterPack) \* Ultrasonic Ranger attached to Pin 24 (J6 plug on Grove Base BoosterPack) \* Note: This example code is in the public domain. http://www.seeedstudio.com/depot/Grove-Ultrasonic-Ranger-p-960.html \*/  #include "TM1637.h" #include "Ultrasonic.h"  /\* Macro Define \*/ #define CLK 40 /\* 4-Digit Display clock pin \*/ #define DIO 39 #define BLINK\_LED RED\_LED /\* 4-Digit Display data pin \*/ #define ULTRASONIC\_PIN 38 /\* pin of the Ultrasonic Ranger \*/ #define BUZZER\_PIN 37  /\* Global Variables \*/ TM1637 tm1637(CLK, DIO); /\* 4-Digit Display object \*/ Ultrasonic ultrasonic(ULTRASONIC\_PIN); /\* Ultrasonic Ranger object \*/ int distance = 0; /\*obstacles in front \*/ int blink\_internal = 0; /\*variable to store the distance to obstacles in front \*/ int8\_t bits[4] = {0}; /\* array to store the single bits of the value \*/ const int threshold = 15; /\* the threshold level \*/  /\* the setup() method runs once, when the sketch starts \*/ void setup() { /\* Initialize 4-Digit Display \*/ tm1637.init(); tm1637.set(BRIGHT\_TYPICAL); pinMode(RED\_LED, OUTPUT); pinMode(BUZZER\_PIN, OUTPUT);  digitalWrite(BUZZER\_PIN, LOW); } /\* the loop() method runs over and over again \*/ void loop() { distance = ultrasonic.MeasureInCentimeters(); /\* read the value from the sensor \*/  if(distance < threshold) { digitalWrite(BUZZER\_PIN, HIGH); delay(100); digitalWrite(BUZZER\_PIN, LOW);  }  memset(bits, 0, 4); /\* reset array before we use it \*/ for(int i = 3; i >= 0; i--) { /\* Convert the value to individual decimal digits for display \*/ bits[i] = distance % 10; distance = distance / 10; tm1637.display(i, bits[i]); /\* display on 4-Digit Display \*/ } delay(100); } |

|  |  |
| --- | --- |
| **Task 3c** | **Real World Data Collection – Error Analysis** |
| **Goal** | Up to this point in the course, we have assumed that the measurements we have taken using the TI Kits and sensors have been absolutely correct. This is unfortunately not the case as any time data is taken, there exists two types of error: Systemic and Random errors. In this problem, you will be investigating the random error inherit to the Ultrasonic Range Sensor as it is important to know what the limits of the equipment are for a warning system used by the Taum Sauk Reservoir. To do this, you will need to complete the following:   1. As in Task 3b, utilize the Grove Starter Kit Ultrasonic Range Sensor and Buzzer. 2. Place a ruler on the table with your Ultrasonic Range Sensor sideways as shown below:      1. Place an index card at the end of the ruler and slowly move it towards Ultrasonic Range Sensor, when the buzzer goes off, stop moving the index card and measure how far from the sensor it is. Repeat this measurement ten times. |
| **Solution: Range Measurements** | |  |  | | --- | --- | | Test | Distance from object when buzzer sounded | | 1 |  | | 2 |  | | 3 |  | | 4 |  | | 5 |  | | 6 |  | | 7 |  | | 8 |  | | 9 |  | | 10 |  | |
| **Solution: Error Analysis** | |  |  |  | | --- | --- | --- | | Test | Percent Error | SSE | | 1 |  |  | | 2 |  |  | | 3 |  |  | | 4 |  |  | | 5 |  |  | | 6 |  |  | | 7 |  |  | | 8 |  |  | | 9 |  |  | | 10 |  |  | | Average |  |  | |
| **Reference: CODE** | Ultrasonic-Ranger Measure the distance to obstacles in front and display the value on the Grove 4-Digit Display The circuit: \* 4-Digit Display attached to Pin 38 and 39 (J14 plug on Grove Base BoosterPack) \* Ultrasonic Ranger attached to Pin 24 (J6 plug on Grove Base BoosterPack) \* Note: This example code is in the public domain. http://www.seeedstudio.com/depot/Grove-Ultrasonic-Ranger-p-960.html \*/  #include "TM1637.h" #include "Ultrasonic.h"  /\* Macro Define \*/ #define CLK 40 /\* 4-Digit Display clock pin \*/ #define DIO 39 #define BLINK\_LED RED\_LED /\* 4-Digit Display data pin \*/ #define ULTRASONIC\_PIN 38 /\* pin of the Ultrasonic Ranger \*/ #define BUZZER\_PIN 37  /\* Global Variables \*/ TM1637 tm1637(CLK, DIO); /\* 4-Digit Display object \*/ Ultrasonic ultrasonic(ULTRASONIC\_PIN); /\* Ultrasonic Ranger object \*/ int distance = 0; /\*obstacles in front \*/ int blink\_internal = 0; /\*variable to store the distance to obstacles in front \*/ int8\_t bits[4] = {0}; /\* array to store the single bits of the value \*/ const int threshold = 15; /\* the threshold level \*/  /\* the setup() method runs once, when the sketch starts \*/ void setup() { /\* Initialize 4-Digit Display \*/ tm1637.init(); tm1637.set(BRIGHT\_TYPICAL); pinMode(RED\_LED, OUTPUT); pinMode(BUZZER\_PIN, OUTPUT);  digitalWrite(BUZZER\_PIN, LOW); } /\* the loop() method runs over and over again \*/ void loop() { distance = ultrasonic.MeasureInCentimeters(); /\* read the value from the sensor \*/  if(distance < threshold) { digitalWrite(BUZZER\_PIN, HIGH); delay(100); digitalWrite(BUZZER\_PIN, LOW);  }  memset(bits, 0, 4); /\* reset array before we use it \*/ for(int i = 3; i >= 0; i--) { /\* Convert the value to individual decimal digits for display \*/ bits[i] = distance % 10; distance = distance / 10; tm1637.display(i, bits[i]); /\* display on 4-Digit Display \*/ } delay(100); } |