Lab 4: Distance Measurement using Ultrasonic on TivaC

Name: Shahmir Mustafa Chaudhry ID: 08400 Section: T1

Objective

To interface HC-SR04 Ultrasonic Sensor with TivaC Launchpad to make Critical Distance Measurement System.

In-Lab Task:

Task 1: Interface HC-SR04 with TivaC and Compute Distance using Energia

Task 2: Integrate LED to Distance Measurement System

Task 3: Create a Critical Distance Measurement System

Task 4: Filtering in Measurement System

1 Introduction to HC-SR04 Ultrasonic Sensor

Ultrasonic sensor HC-SR04 is one the many sensors used to measure distance in the range of 2 cm to 400 cm with an accuracy of 3 mm. The sensor module consists of the control circuit and two ultrasonic sensors — one is the transmitter, and another is the receiver as shown in Fig. 1.

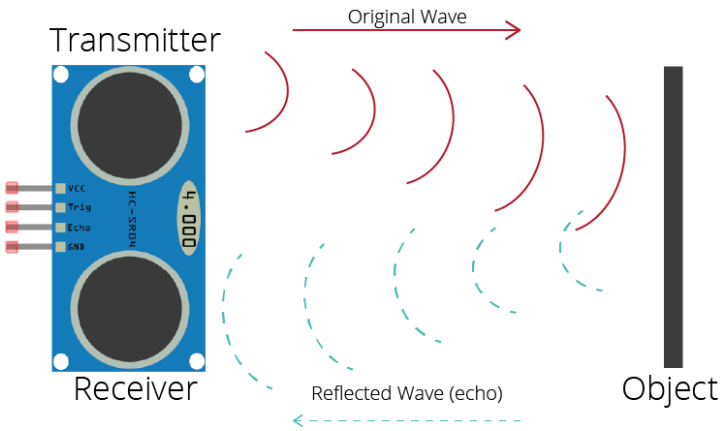


Figure 1: Distance Measurement using HC-SR04

The Fig. 2 given below shows the pinout diagram of the HC-SR04 ultrasonic sensor. It consists of 4 pins. Two of them are power supply pins such as +5V and ground. The other two pins are Trigger and echo pin. The echo output pulse pin is used to get output from the sensor. Trigger pin is used to initiate the sensor to start ranging.

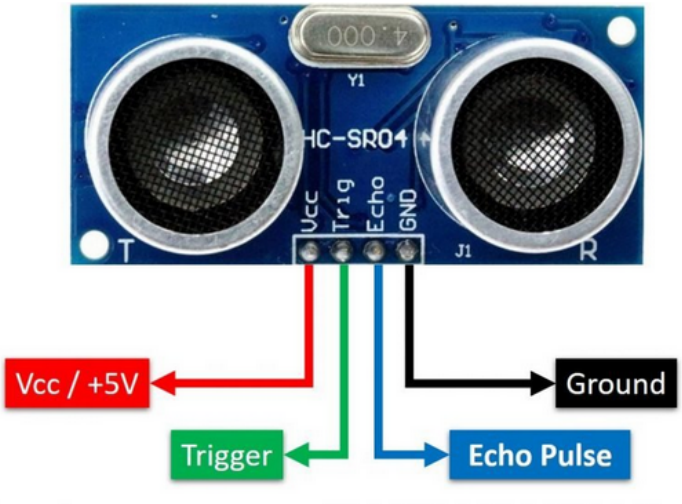


Figure 2: Pin Configuration of HC-SR04

1.1 Distance Measurement Steps using HC-SR04

In order to measure the distance using HC-SR04 using TivaC, we will later on use the following steps in the code in energia environment:

- First of all, we apply 10 microseconds pulse to the Trigger pin of HC-SR04 sensor from general purpose input output pin of TM4C123 microcontroller.
- As a result, the transmitter circuit produces 8 pulses burst of ultrasound waves and each pulse has 40KHz frequency.
- As soon as all eight pulses are transmitted through the air, the echo pin goes high. In other words, the output echo pin makes transition from an active low to active high level as shown in Fig. 3.
- The echo pin remains active high until the ultrasonic sound wave does not reflect back to the receiver circuit after striking the object Under test.
- As soon as the ultrasonic signal received by the receiver circuit after striking with an object, the echo pin goes low.
- By measuring the width of the output pulse, we can measure the distance as shown in Fig. 4.

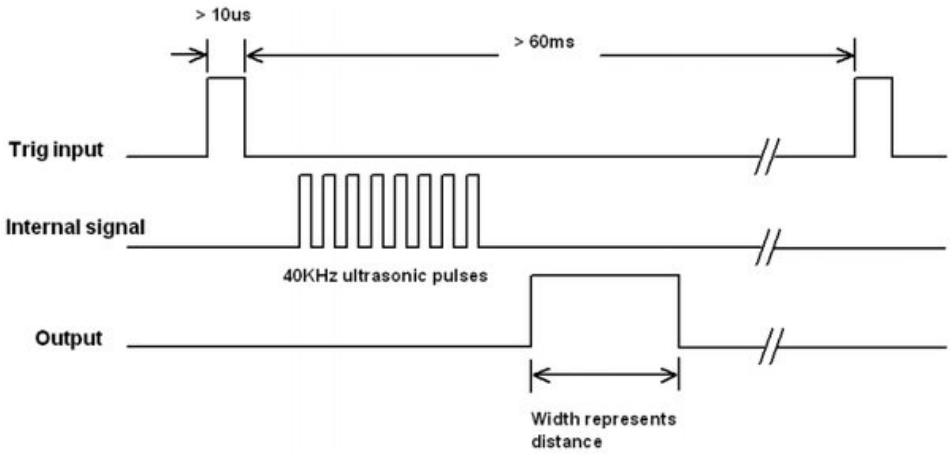


Figure 3: Timing Diagram for HC-SR04

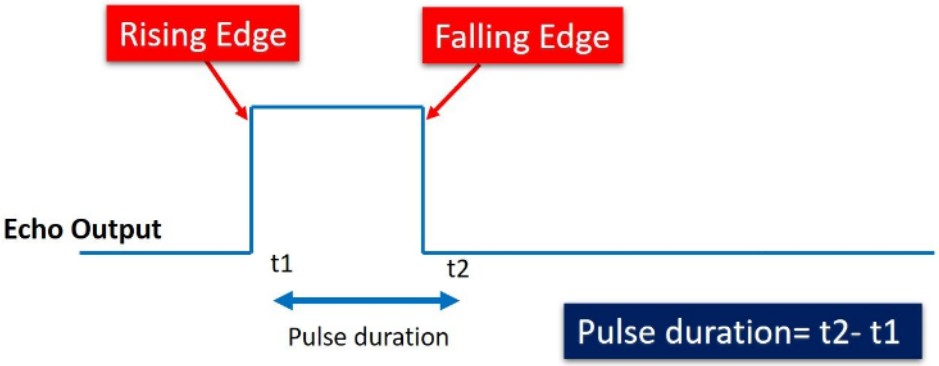


Figure 4: Pulse Duration for HC-SR04 Echo Wave

The pulse width for Echo can vary between 150 µS to 25 mS depending on the distance between the sensor and the obstacle. More the distance, the more time the sonar sound wave takes to receive after reflecting back. If there is no obstacle or out of the range of the maximum range of the sonar sensor (400cm), the transmitted signal will not reflect back. In this case, the echo output signal will go high for 38ms. Hence, if the width of the output pulse is 38ms, that means there is no object in the range of the obstacle sensor. Hence, the measured distance will be zero. Please make sure the surface of object to be detect should have at least area of 0.5 m^2 for better performance.

2 TivaC LaunchPad with Energia

In Energia IDE, unlike Kiel uVision, we can use TivaC LaunchPad pins for various peripherals without the need to activate Ports using registers or specifying function of the pin. But it also

comes with limitations of usability and programmable scope of the board. In Energia, we can refer to Pins of TivaC directly using numeric digit like 1,2,3.. and so on. Pin map for the EK-TM4C123GXL LaunchPad is given in Fig. 5 and Fig. 6 with Black Columns under J1, J2, J3 and J4.

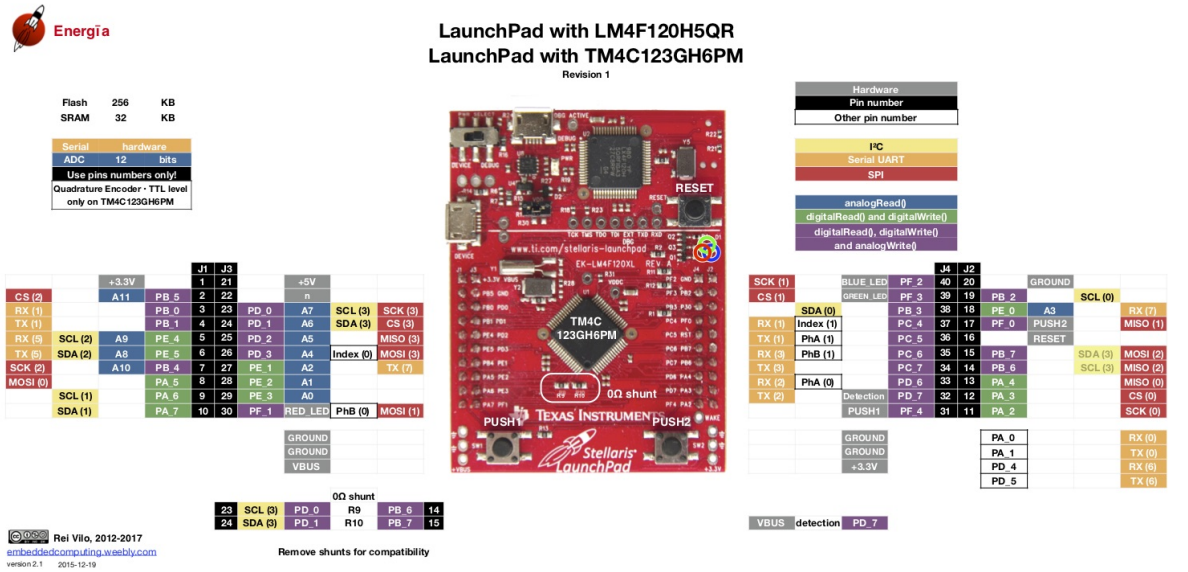


Figure 5: Pin Configuration of TivaC for Energia (Front)

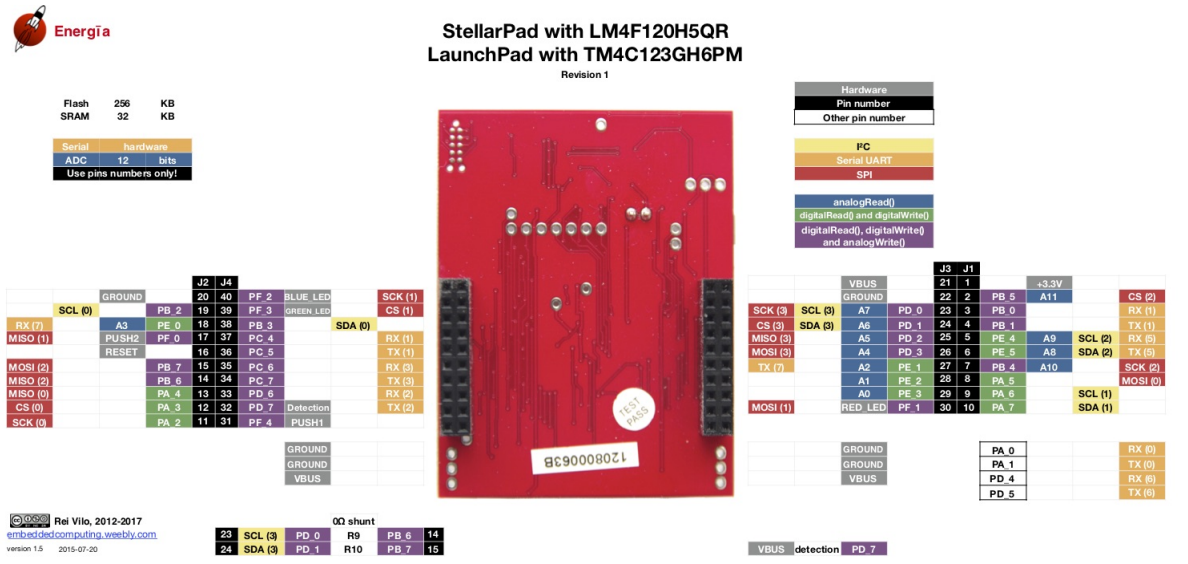


Figure 6: Pin Configuration of TivaC for Energia (Back)

3 In-Lab Tasks

Task 1: Interface HC-SR04 with TivaC and Compute Distance using Energia

In order to use the HC-SR04 with TivaC, follow the steps given below:

1 Make the pin connections between TivaC and HC-SR04 using Table. 1 and Fig 7.

2 Download the file "Ultrasonic.ino" from LMS and load the code in Energia IDE, go through each line carefully to understand each operation as you will be writing your own code for later tasks and later labs.

3 Write your logic in code where you are asked to i.e. "microsecondsToCentimeters()" function as per explanation given.

3 Connect tivaC, compile/build and download code on tivaC. Once done, open the "Serial Monitor" from "Tools" to observe the distance. Use any object to vary the distance in front of Ultrasonic Sensor.

Table 1: Task01 Connections

HC-SR04	TivaC	Energia
Vcc	VBUS	-
Trig	PA7	10
Echo	PA6	9
GND	GND	-

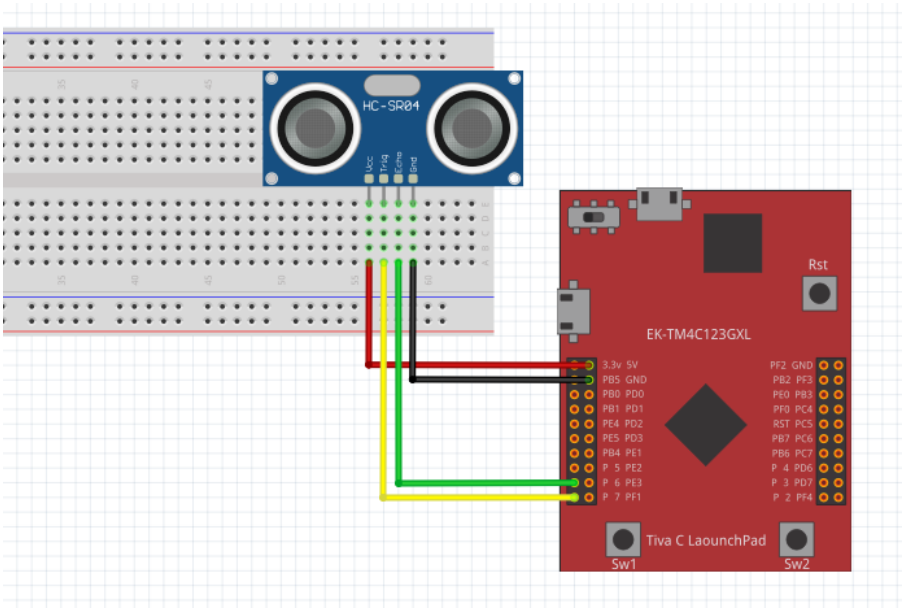


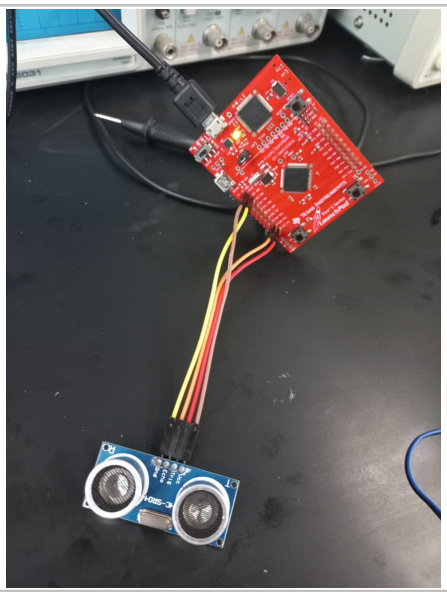
Figure 7: Pin Connection of TivaC with Ultrasonic Sensor

Provide your code here with appropriate comments below (Get the circuit demonstration checked with RA within Lab)

```
////////////////////////////////////////////////
//Ultrasonic Pins Definition, here we specify pin usage for our tivaC Board

const int echo = 9, Trig = 10;  //Pin connections for HC-SR04
//const int LED1 = 11;          //Specify Pin for LED1


////////////////////////////////////////////////
//Time Update Variables

long duration, cm;                      //To measure pulse width and distance in cm
long distance;                          //To return distance of object
unsigned long LastUpdateMicrosecs = 0;
unsigned long LastUpdateMillisecs = 0;
unsigned long CurrentMicrosecs = 0;
unsigned long MicrosecsSinceLastUpdate = 0;
float SecondsSinceLastUpdate = 0;


////////////////////////////////////////////////
//Initializing Pins Function

void setup()
{
  Serial.begin(115200);     //Initialize Serial COM Port with 115200 buad rate
  pinMode(Trig, OUTPUT);    //Trigger is output since it will send 8 pulses at 40KHz from HC-SR04
  pinMode(echo, INPUT);     //Echo is input since it will detect the reflected wave from HC-SR04
  //pinMode(LED1, OUTPUT);    //Pin for LED1 should output HIGH(on) or LOW(off)
}


////////////////////////////////////////////////
//Main Loop that will keep on running, it is similar to int main()
void loop()
{
    Update_Ultra_Sonic(); //This function operates HC-SR04
    delay(200);           //Delay between prints on Serial Monitor
}

////////////////////////////////////////////////
//Will update ultrasonic sensors through Serial port

void Update_Ultra_Sonic()
{
  //Setting Trig Signal HIGH for 10us to produce burst of 8 pulses at 40KHz
  digitalWrite(Trig, LOW);
  delayMicroseconds(2);
  digitalWrite(Trig, HIGH);
  delayMicroseconds(10);
  digitalWrite(Trig, LOW);

  //digitalRead(anypin/switch) //Use this function to read the state of any Pin/Switch i.e. SW1 and SW2

  duration = pulseIn(echo, HIGH);   // Reads duration (microseconds) for which Echo pin reads HIGH till wave is reflected
//  if (duration > 0){
//     cm = microsecondsToCentimeters(duration);  // convert the time into a distance, refer to function below
//  }
//  else{
//    cm = -1;
//  }
  cm = microsecondsToCentimeters(duration); // convert the time into a distance, refer to function below

  //Sending through serial port to print on Monitor
  Serial.print("Distance=");
  Serial.print("\t");
  Serial.print(cm);
  Serial.print("\n");
}

////////////////////////////////////////////////
//Conversion of microsecond to centimeter
long microsecondsToCentimeters(long microseconds)
{
// The speed of sound is 340 m/s or 29 microseconds per centimeter.

// convert speed of sound from m/s to cm/us
// Use the formula:     distance(cm) = speed(cm/us) * time(us)

// Write distance formula here
   distance = 0.0343 *microseconds;
// We take half of the distance travelled since its a reflected wave
   return distance / 2;
}
```

Provide snippet for Serial Monitor display

Provide your clear circuit image below



Task 2: Integrate LED to Distance Measurement System

For this task, you will integrate an LED with the Ultrasonic Sensor. You are required to modify the code to **turn the LED on if the distance falls below 10cm**. For this purpose, modify your connections as per Table. 2 and Fig 8. And then uncomment the code required for LED purpose in the provided "Ultrasonic.ino".
You can use **220ohms** resistor for this purpose. Also be careful with LED connections, make sure to connect LED Anode(+ve) to PA2 and LED Cathode(-ve) to GND through resistor.

Table 2: Task02 Connections

HC-SR04/LED	TivaC	Energia
Vcc	VBUS	-
Trig	PA7	10
Echo	PA6	9
GND	GND	-
LED 1	PA2	11

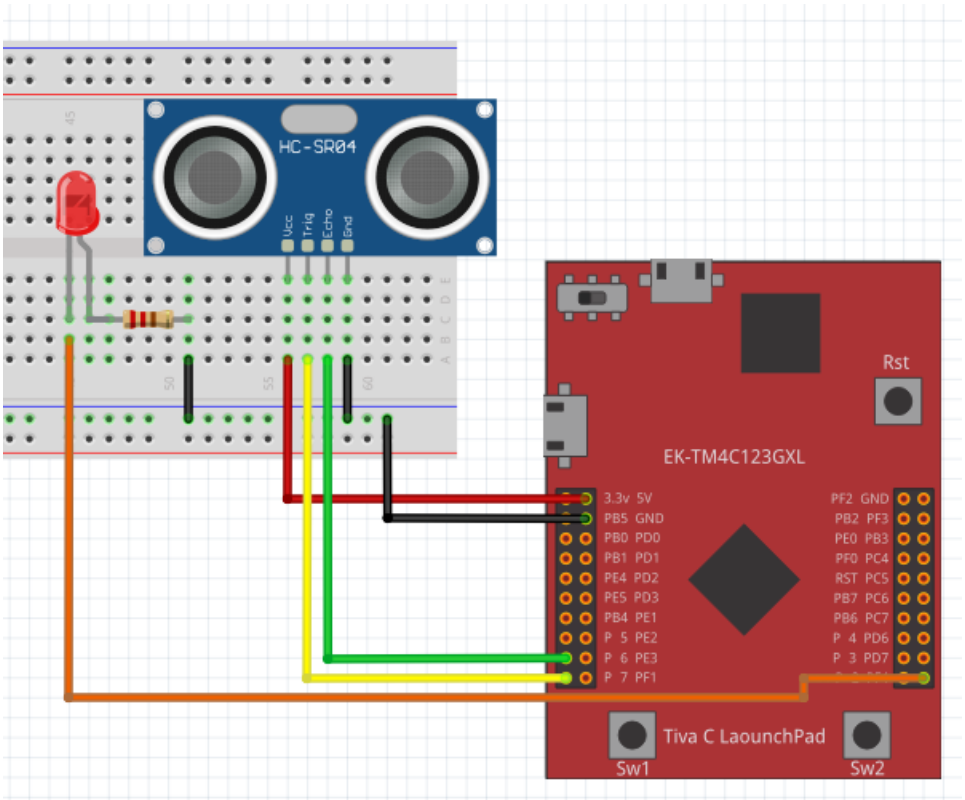
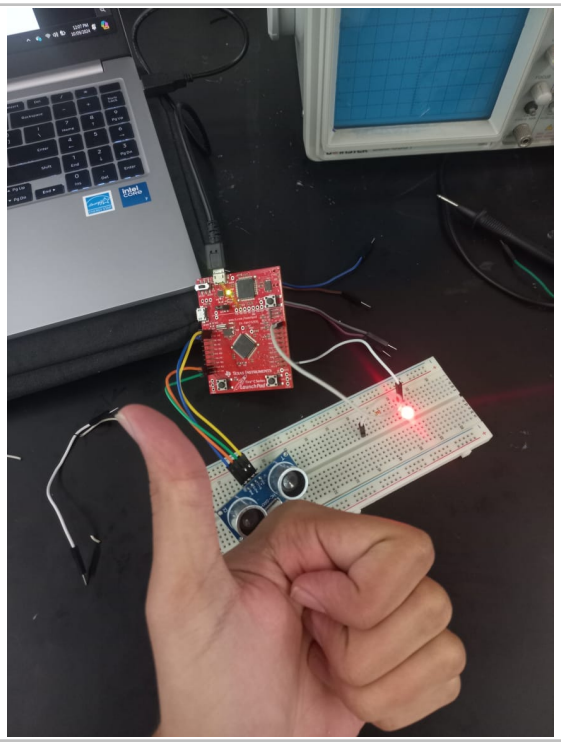


Figure 8: Pin Connection of TivaC with Ultrasonic Sensor and LED

Provide your code here with appropriate comments below (Get the circuit demonstration checked with RA within Lab)

```
////////////////////////////////////////////////
//Ultrasonic Pins Definition, here we specify pin usage for our tivaC Board

const int echo = 9, Trig = 10;  //Pin connections for HC-SR04
const int LED1 = 11;          //Specify Pin for LED1


////////////////////////////////////////////////
//Time Update Variables

long duration, cm;                        //To measure pulse width and distance in cm
long distance;                        //To return distance of object
unsigned long LastUpdateMicrosecs = 0;
unsigned long LastUpdateMillisecs = 0;
unsigned long CurrentMicrosecs = 0;
unsigned long MicrosecsSinceLastUpdate = 0;
float SecondsSinceLastUpdate = 0;


////////////////////////////////////////////////
//Initializing Pins Function

void setup()
{
  Serial.begin(115200);     //Initialize Serial COM Port with 115200 buad rate
  pinMode(Trig, OUTPUT);    //Trigger is output since it will send 8 pulses at 40KHz from HC-SR04
  pinMode(echo, INPUT);     //Echo is input since it will detect the reflected wave from HC-SR04
  pinMode(LED1, OUTPUT);    //Pin for LED1 should output HIGH(on) or LOW(off)
}


////////////////////////////////////////////////
//Main Loop that will keep on running, it is similar to int main()
void loop()
{
    Update_Ultra_Sonic(); //This function operates HC-SR04
    delay(200);           //Delay between prints on Serial Monitor
}

////////////////////////////////////////////////
//Will update ultrasonic sensors through Serial port

void Update_Ultra_Sonic()
{
  //Setting Trig Signal HIGH for 10us to produce burst of 8 pulses at 40KHz
  digitalWrite(Trig, LOW);
  delayMicroseconds(2);
  digitalWrite(Trig, HIGH);
  delayMicroseconds(10);
  digitalWrite(Trig, LOW);

  //digitalRead(anypin/switch) //Use this function to read the state of any Pin/Switch i.e. SW1 and SW2

  duration = pulseIn(echo, HIGH);   // Reads duration (microseconds) for which Echo pin reads HIGH till wave is reflected
//  if (duration > 0){
//     cm = microsecondsToCentimeters(duration);  // convert the time into a distance, refer to function below
//  }
//  else{
//    cm = -1;
//  }
  cm = microsecondsToCentimeters(duration); // convert the time into a distance, refer to function below

  //Sending through serial port to print on Monitor
  Serial.print("Distance=");
  Serial.print("\t");
  Serial.print(cm);
  Serial.print("\n");
  if (cm<10){
    digitalWrite(LED1, HIGH);
  }
  else{
    digitalWrite(LED1, LOW);
  }
}

////////////////////////////////////////////////
//Conversion of microsecond to centimeter
long microsecondsToCentimeters(long microseconds)
{
// The speed of sound is 340 m/s or 29 microseconds per centimeter.

// convert speed of sound from m/s to cm/us
// Use the formula:     distance(cm) = speed(cm/us) * time(us)

// Write distance formula here
   distance = (0.0343) *microseconds;
// We take half of the distance travelled since its a reflected wave
   return distance / 2;
}
```

Provide your clear circuit image below



Task 3: Create a Critical Distance Measurement System

Program you TivaC board to design a "Critical Distance Measurement System" by using the Table in Table. 3 to decide circuit connections. The system should contain Ultrasonic Sensor interfaced with 3 Red LEDs using TivaC. The system should fulfill the following requirement:

1. LED 1 ON for range 15-30cm
2. LED 1 and LED 2 ON for range 5-15cm
3. LED 1, LED 2 and LED 3 ON for range 0-5cm
4. All LEDs should **Blink** simultaneously and periodically if there is no object in sight i.e. range $>$ 100cm

5 The system should also contain a main ON/OFF function using the built-in SW1 and SW2 switches of TivaC using following command:
- Read pin PF0 for SW2 state
- Read pin PF4 for SW1 state

Table 3: Task03 Connections

HC-SR04/LED	**TivaC**	**Energia**
Vcc	VBUS	-
Trig	PA7	10
Echo	PA6	9
GND	GND	-
LED 1	PA2	11
LED 2	PA3	12
LED 3	PA4	13
-	SW1/PF4	31
-	SW2/PF0	17

6. Serial Monitor should display the appropriate current state of all LEDs and distance for measurement system
7. Choose your own appropriate pins for LED 1,2 and 3 from pin-mapping chart and fill in Table 3. Make sure the selected pins have digitalWrite() functionality as shown in pin-mapping chart.

Attach your code and circuit image in the sections given next.

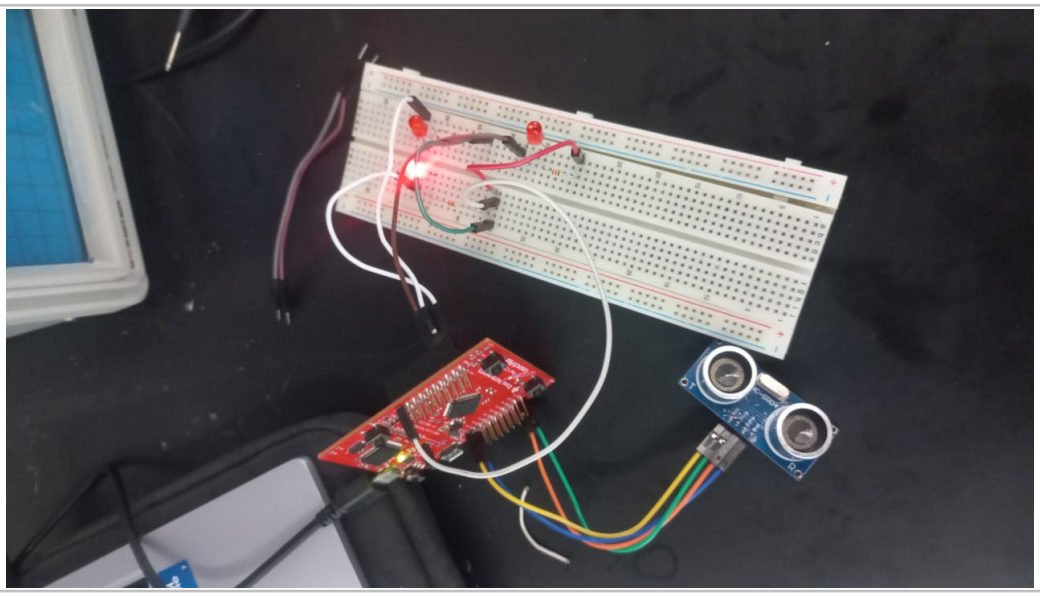
Provide your code here with appropriate comments below (Get the circuit demonstration checked with RA within Lab)

```
////////////////////////////////////////////////
//Ultrasonic Pins Definition, here we specify pin usage for our tivaC Board

const int echo = 9, Trig = 10;  //Pin connections for HC-SR04
const int LED1 = 11;          //Specify Pin for LED1
const int LED2 = 12;
const int LED3 = 13;
const int SW1 = 31;
const int SW2 = 17;

////////////////////////////////////////////////
//Time Update Variables

long duration, cm;                      //To measure pulse width and distance in cm
long distance;                          //To return distance of object
unsigned long LastUpdateMicrosecs = 0;
unsigned long LastUpdateMillisecs = 0;
unsigned long CurrentMicrosecs = 0;
unsigned long MicrosecsSinceLastUpdate = 0;
float SecondsSinceLastUpdate = 0;
bool systemOn = false;

////////////////////////////////////////////////
//Initializing Pins Function

void setup()
{
  Serial.begin(115200);     //Initialize Serial COM Port with 115200 buad rate
  pinMode(Trig, OUTPUT);    //Trigger is output since it will send 8 pulses at 40KHz from HC-SR04
  pinMode(echo, INPUT);     //Echo is input since it will detect the reflected wave from HC-SR04
  pinMode(LED1, OUTPUT);    //Pin for LED1 should output HIGH(on) or LOW(off)
  pinMode(LED2, OUTPUT);
  pinMode(LED3, OUTPUT);
  pinMode(SW1, INPUT_PULLUP);
  pinMode(SW2, INPUT_PULLUP);
}


////////////////////////////////////////////////
//Main Loop that will keep on running, it is similar to int main()
void loop()
{
{
    // Check if SW1 is pressed to turn the system ON
    if (digitalRead(SW1) == LOW) {
        systemOn = true;  // Set flag to true when SW1 is pressed
    }

    // Check if SW2 is pressed to turn the system OFF
    if (digitalRead(SW2) == LOW) {
        systemOn = false;  // Set flag to false when SW2 is pressed
        TurnOffLEDs();     // Turn off all LEDs
    }

    // If the system is ON, update the ultrasonic sensor
    if (systemOn) {
        Update_Ultra_Sonic(); // This function operates HC-SR04 and controls LEDs
        delay(200);           // Delay between prints on Serial Monitor
    }
}
}

////////////////////////////////////////////////
//Will update ultrasonic sensors through Serial port

void Update_Ultra_Sonic()
{
  //Setting Trig Signal HIGH for 10us to produce burst of 8 pulses at 40KHz
  digitalWrite(Trig, LOW);
  delayMicroseconds(2);
  digitalWrite(Trig, HIGH);
  delayMicroseconds(10);
  digitalWrite(Trig, LOW);

  //digitalRead(anypin/switch) //Use this function to read the state of any Pin/Switch i.e. SW1 and SW2

  duration = pulseIn(echo, HIGH);   // Reads duration (microseconds) for which Echo pin reads HIGH till wave is reflected
//  if (duration > 0){
//     cm = microsecondsToCentimeters(duration);  // convert the time into a distance, refer to function below
//  }
//  else{
//     cm = -1;
//  }
  cm = microsecondsToCentimeters(duration); // convert the time into a distance, refer to function below

  //Sending through serial port to print on Monitor
  Serial.print("Distance=");
  Serial.print("\t");
  Serial.print(cm);
  Serial.print("\n");

  if (cm>15 and cm<30){
    digitalWrite(LED1, HIGH);
    digitalWrite(LED2, LOW);
    digitalWrite(LED3, LOW);
    Serial.print("One light");
    Serial.print("\n");
  }
  else if(cm>5 and cm <15){
    digitalWrite(LED1, HIGH);
    digitalWrite(LED2, HIGH);
    digitalWrite(LED3, LOW);
    Serial.print("Two light");
    Serial.print("\n");
  }
  else if(cm< 5){
    digitalWrite(LED1, HIGH);
    digitalWrite(LED2, HIGH);
    digitalWrite(LED3, HIGH);
    Serial.print("Three light");
    Serial.print("\n");
  }
  else if(cm>30){
  blink();
     }
}

void blink(){
    digitalWrite(LED1, HIGH);
    digitalWrite(LED2, HIGH);
    digitalWrite(LED3, HIGH);
    delay(500);
```

Provide snippet for Serial Monitor display

```
Distance=       21
One light
Distance=       13
Two light
Distance=       8
Two light
Distance=       7
Two light
Distance=       7
Two light
Distance=       10
Two light
Distance=       13
Two light
Distance=       11
Two light
Distance=       10
Two light
Distance=       5
Distance=       4
Three light
Distance=       4
Three light
Distance=       5
Distance=       7
Two light
Distance=       3
Three light
Distance=       8
Two light
Distance=       1204
Distance=       1204
Distance=       1204
Distance=       1204
Distance=       1204
Distance=       1204
Distance=       1204
Distance=       1204
```

Provide your clear circuit image below



Task 4: Filtering in Measurement System

As you have noticed in your previously designed measurement system that there are unwanted erroneous readings introduced due to loose wire connections, voltage fluctuations and sensor inaccuracies. We will address this issue by designing a simple filtering averaging algorithm. In order to implement this filter, you have to perform the following steps:

1. Collect multiple readings (10-15) from ultrasonic sensor by adding value into same variable.
2. Use delay of 100-200ms between above collected measurements
3. Divide the cumulative variable by total number of measurements taken to average it.
4. Display the averaged value on Serial Monitor.

Do you observe any improvement in the filtered measured output? What are the drawbacks of using such an averaging system?

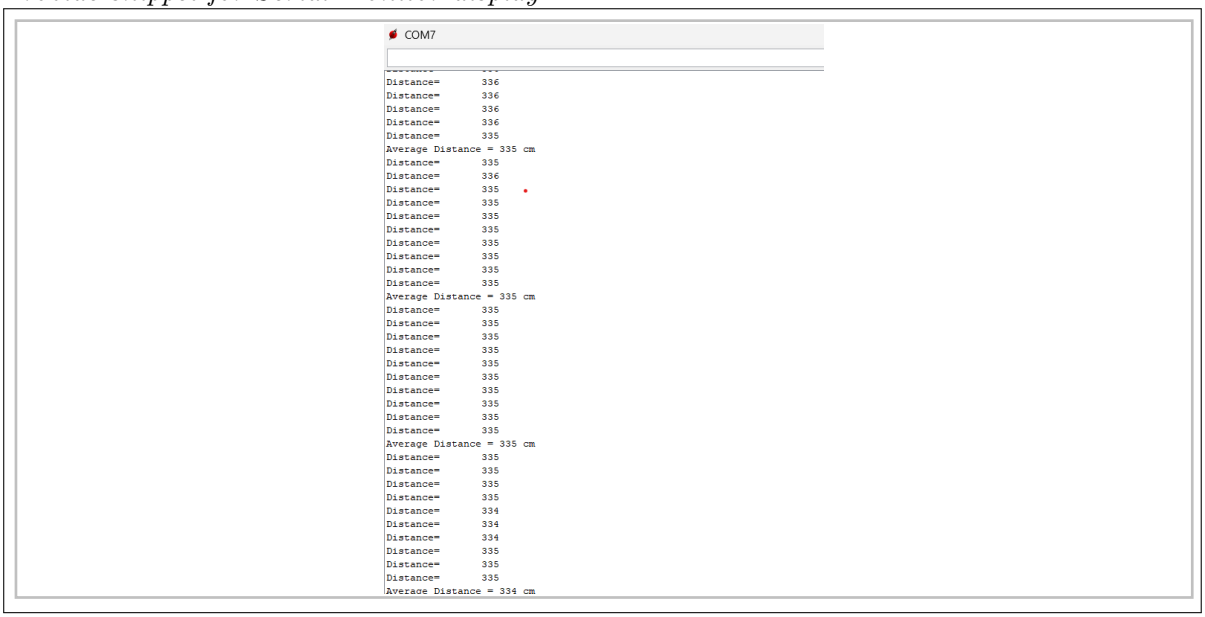
pros: taking multiple readings enables a better estimate of the position of the object from the sensor. An averaged out value has lesser significance of error. The filtered output is therefore more consistent.

cons: if there is changes in the data in real time happening, then the averaged out value would consider it part of the same measurement and therefore slag behind, This is also a loss of data. If the movement of the object is quicker than the averaging filter.

Provide your code here with appropriate comments below (Get the circuit demonstration checked with RA within Lab)

```
////////////////////////////////////////////////
//Ultrasonic Pins Definition, here we specify pin usage for our tivaC Board

const int echo = 9, Trig = 10;  //Pin connections for HC-SR04
//const int LED1 = 11;          //Specify Pin for LED1


////////////////////////////////////////////////
//Time Update Variables

long duration, cm;                      // To measure pulse width and distance in cm
long distance = 0;                      // To return distance of object
int numReadings = 0;                    // Counter for the number of readings
long totalDist = 0;                     // Cumulative sum of distances
long avgDist ;
unsigned long LastUpdateMicrosecs = 0;
unsigned long LastUpdateMillisecs = 0;
unsigned long CurrentMicrosecs = 0;
unsigned long MicrosecsSinceLastUpdate = 0;
float SecondsSinceLastUpdate = 0;


////////////////////////////////////////////////
//Initializing Pins Function

void setup()
{
  Serial.begin(115200);     //Initialize Serial COM Port with 115200 buad rate
  pinMode(Trig, OUTPUT);    //Trigger is output since it will send 8 pulses at 40KHz from HC-SR04
  pinMode(echo, INPUT);     //Echo is input since it will detect the reflected wave from HC-SR04
  //pinMode(LED1, OUTPUT);    //Pin for LED1 should output HIGH(on) or LOW(off)
}


////////////////////////////////////////////////
//Main Loop that will keep on running, it is similar to int main()
void loop() {
    // Reset the cumulative variables before taking new readings
    totalDist = 0;
    numReadings = 0;

    // Collect 10 readings and accumulate the total distance
    for (int i = 0; i < 10; i++) {
        totalDist+=Update_Ultra_Sonic(); // Take a single reading
        numReadings++;
        delay(100);           // Delay between readings (100ms)
    }

    // Calculate the average distance
    avgDist = totalDist / numReadings;

    // Display the averaged distance on Serial Monitor
    Serial.print("Average Distance = ");
    Serial.print(avgDist);
    Serial.println(" cm");

    delay(1000); // Delay before starting the next measurement cycle
}
////////////////////////////////////////////////
//Will update ultrasonic sensors through Serial port

long Update_Ultra_Sonic()
{    //Setting Trig Signal HIGH for 10us to produce burst of 8 pulses at 40KHz
    digitalWrite(Trig, LOW);
    delayMicroseconds(2);
    digitalWrite(Trig, HIGH);
    delayMicroseconds(10);
    digitalWrite(Trig, LOW);
    duration = pulseIn(echo, HIGH);   // Reads duration (microseconds) for which Echo pin reads HIGH till wave is reflected


  //digitalRead(anypin/switch) //Use this function to read the state of any Pin/Switch i.e. SW1 and SW2

    cm = microsecondsToCentimeters(duration); // convert the time into a distance, refer to function below

  //Sending through serial port to print on Monitor
    Serial.print("Distance=");
    Serial.print("\t");
    Serial.print(cm);
    Serial.print("\n");

    return cm;
}

////////////////////////////////////////////////
//Conversion of microsecond to centimeter
long microsecondsToCentimeters(long microseconds)
{
   distance = 0.0343*microseconds;


return distance / 2;
}
```

Provide snippet for Serial Monitor display



4 Assessment Rubrics

Marks distribution

		LR2	LR4	LR5	LR9
In-lab	Task 1	5 points	-	5 points	10 points
	Task 2	10 points	10 points	10 points	
	Task 3	10 points	10 points	10 points	
	Task 4	-	10 points	10 points	
Total Marks 100					

Marks obtained

		LR2	LR4	LR5	LR9
In-lab	Task 1		-		
	Task 2				
	Task 3				
	Task 4	-			
Obt. Marks out of 100					