M.K.S.S.S's Cummins College of Engineering for Women

Department of Electronics & Telecommunication Engineering

Course: Advanced Processor Lab (20EC503L)
A.Y. (2022-23)
Open Ended Assignment

Fuel Level Monitoring System

Guided by:
Dr. Nitin Palan
E&TC Department

Team Members

1. C22020111245 [3243]
   Piyusha Shendage
2. C22020111246 [3244]
   Vaishnavi Shirsalkar
3. C22020111248 [3246]
   Shivani N Patil
4. C22020111256 [3254]
   Tanvi Tambe
5. C22020111260 [3258]
   Shivani Thorve

Abstract

Problem Statement :

In most cases, when we fill a tank with fuel, we are unsure if we are getting exactly 1L, 2L, or other amounts that we have requested.

Therefore, we must design a system to determine the quantity of fuel we are receiving.

Introduction

Automobiles are no longer considered a luxury in today's fast-paced society; instead, they are merely a necessity for the majority of people. This change coincides with the increased demand for fuel, garages, service stations, etc., which creates new opportunities for fraud. The scam at the gas pump is the most prevalent in India.

This problem is accompanied by modern automobile systems' digital fuel gauges having the drawback of showing a bar or deflecting needle as opposed to the actual amount of gasoline in the rider's fuel tank. Fuel thefts stem from this. Digital fuel meters and anti-theft systems should be installed in vehicles to prevent such issues.

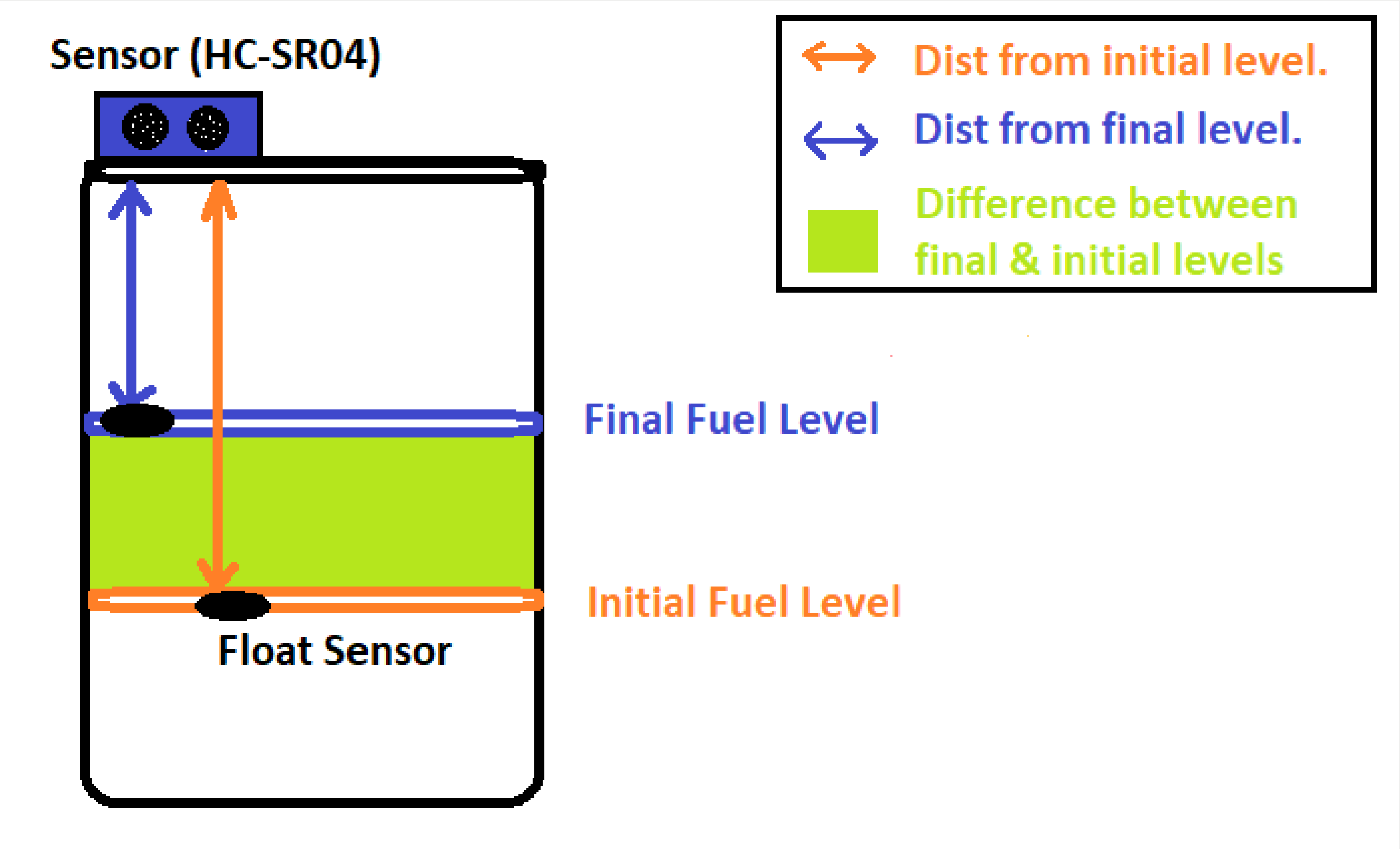
We want to improve user experience by addressing such important issues in our project that simplifies the process of allowing users to determine which stations are scamming them and which are not.

Rubric 1 : **Approach towards the given problem and innovative idea [CO4]**
Selection and placement of sensors, actuators etc, placement of the total system in a physical world

Approach

Above the float sensor are ultrasonic sensors. Before filing, we first determine the original fuel level. The new level after filling the fuel tank is calculated in the second step.

We calculate the difference between the two levels, and we can use this calculation to determine how many litres of fuel are filled in the tank.

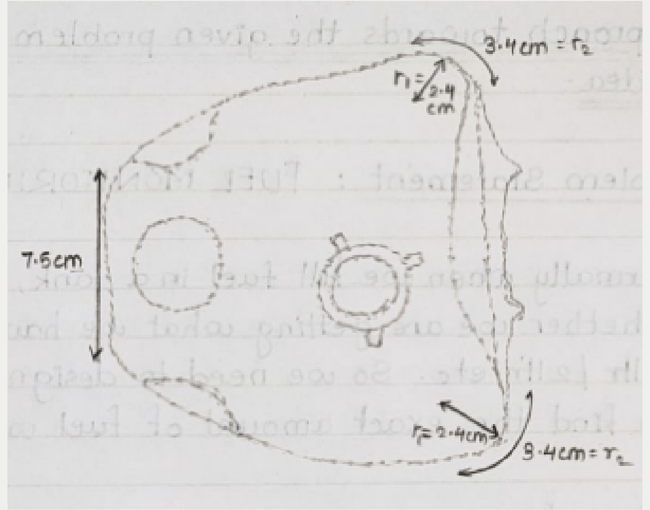


dig 0. Placement of Sensor in a physical world.

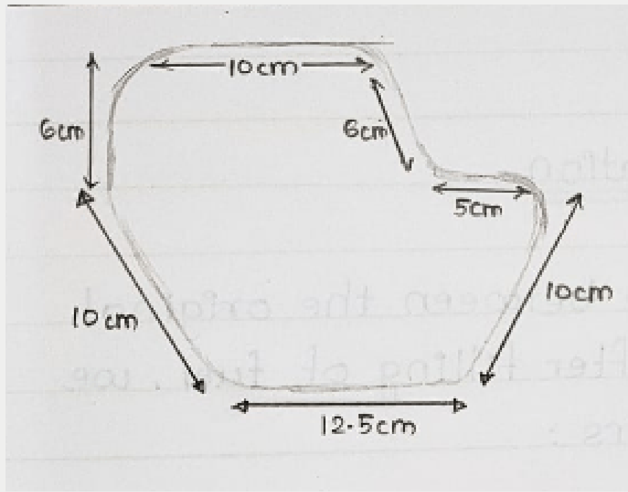
To obtain the relation of distance with respect to liters of fuel filled, we performed a case study of an **ACTIVA** fuel tank.

Img 1 . Activa Fuel Tank (5.3 litres)

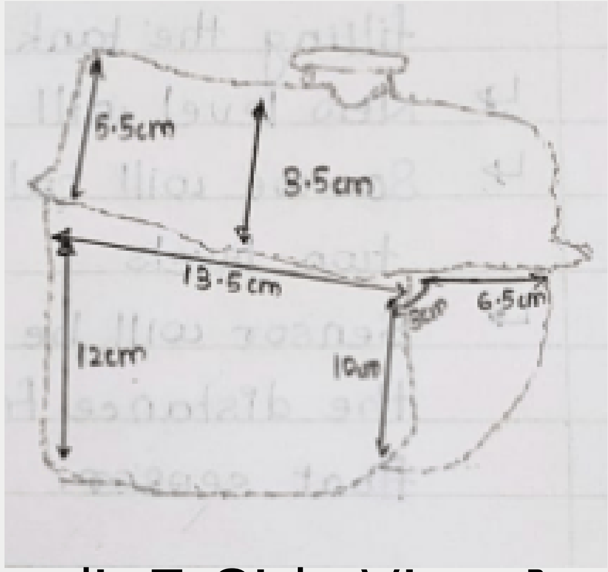
We observed the distance filled for each 100 mL increment at every stage and plotted a graph to see the relation between volume and distance.



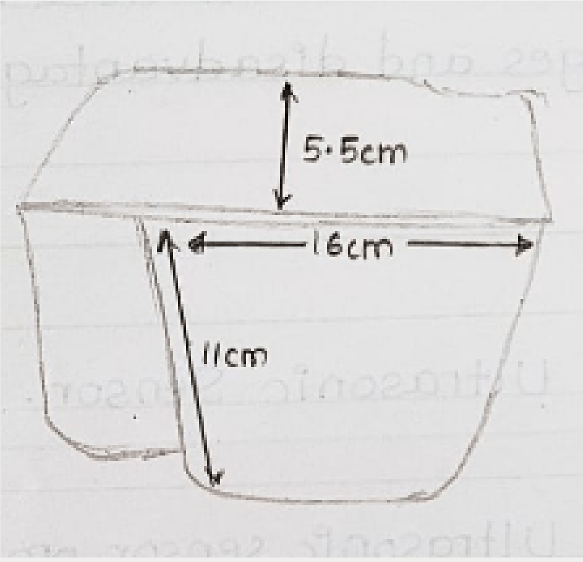
dig1.. Top View



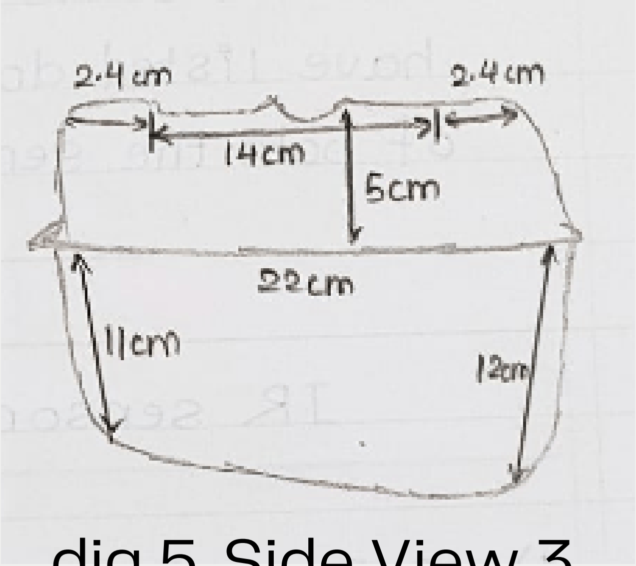
dig2. Bottom View



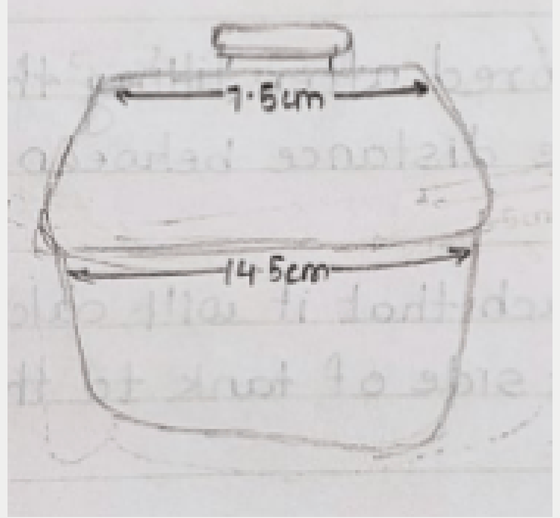
dig3. Side View 1



dig4. Side View 2



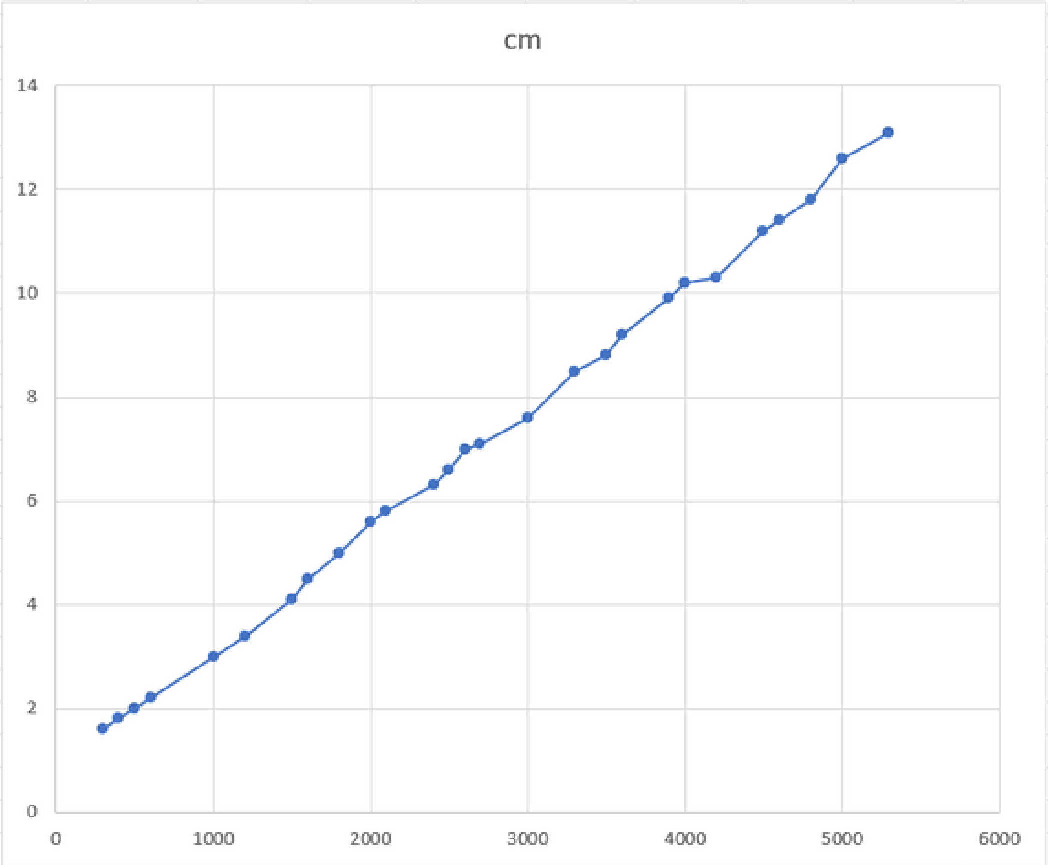
dig 5. Side View 3



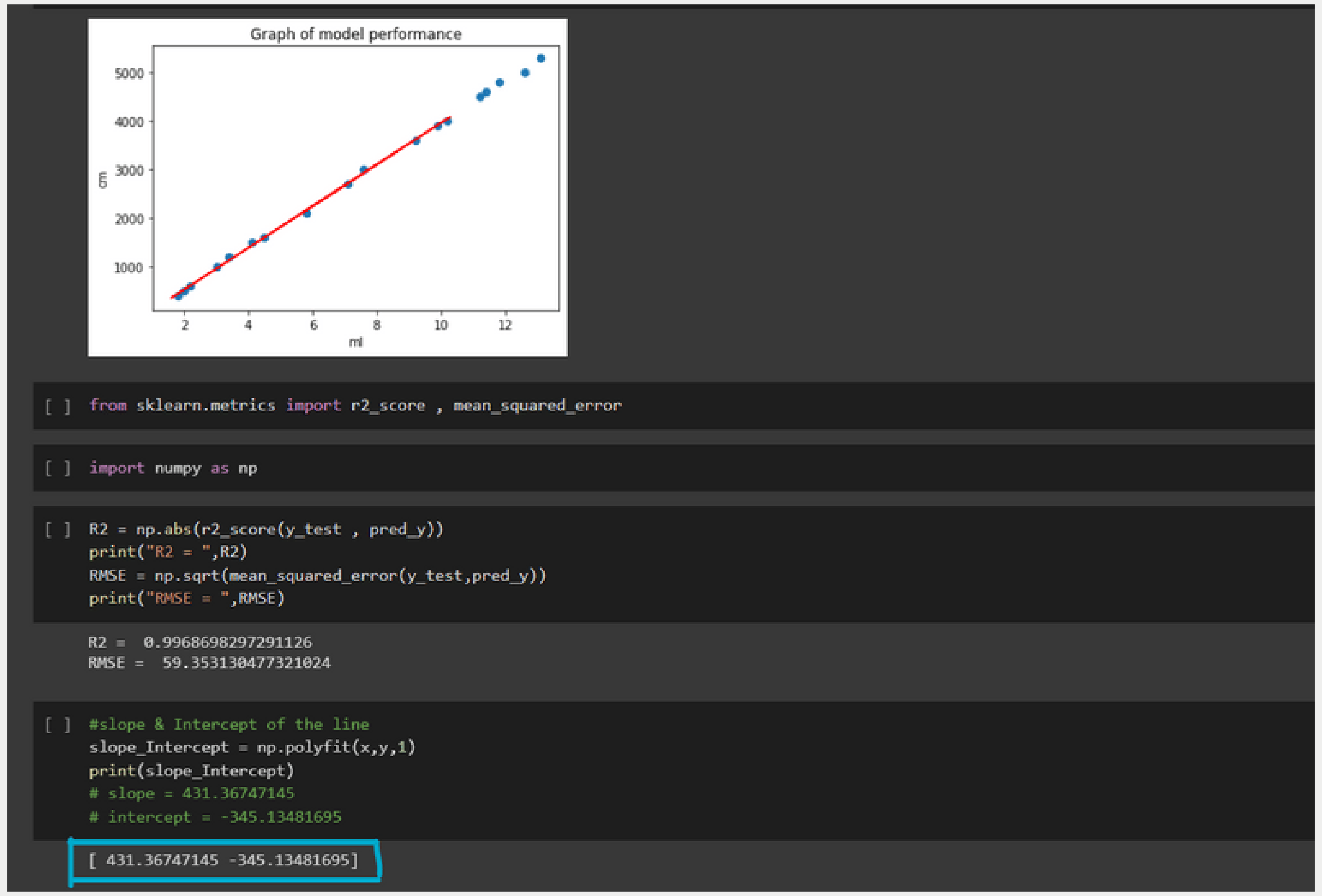
dig6. Side View 4

Approach

ml	cm
300	1.6
400	1.8
500	2
600	2.2
1000	3
1200	3.4
1500	4.1
1600	4.5
1800	5
2000	5.6
2100	5.8
2400	6.3
2500	6.6
2600	7
2700	7.1
3000	7.6
3300	8.5
3500	8.8
3600	9.2
3900	9.9
4000	10.2
4200	10.3
4500	11.2
4600	11.4
4800	11.8
5000	12.6
5300	13.1



Img 2. Distance vs Litres



Img 3. Linear Regression model fitting

From the graph it can be seen that the relation between volume and distance is linear and we fitted a linear regression model on this dataset to obtain the function describing the relationship between distance and the volume of fuel filled in the tank.

After fitting the model we find our data can be efficiently represented by the Y = MX + C equation where M (slope) is 431.36747145 and C (intercept) is -345.13481695.

Y = **(431.36747145 * DISTANCE) - 345.13481695**

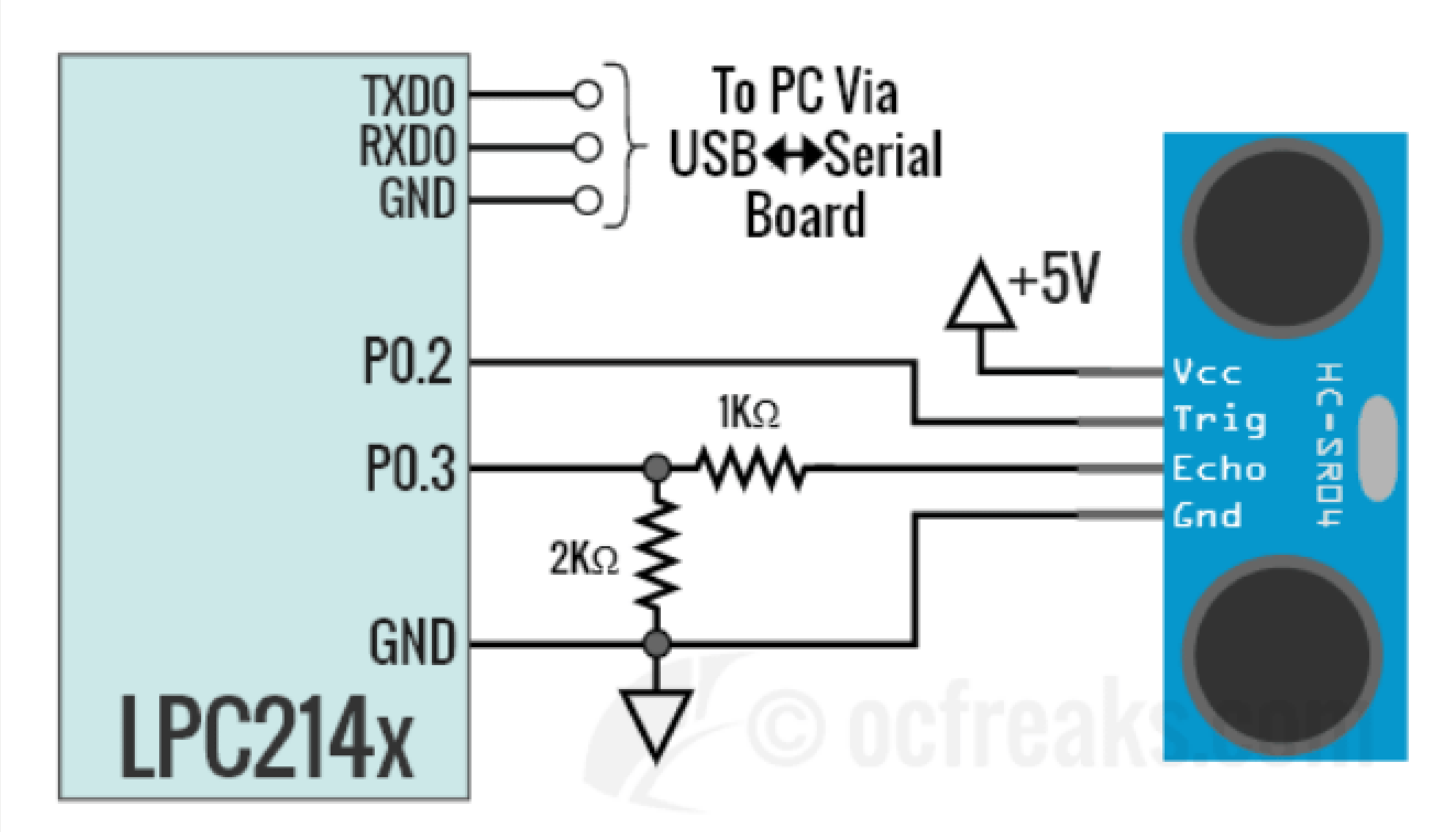
Rubric 2 : **Block diagram and explanation [CO4]**

Logical explanation of each block with the comparison and selection of components

To obtain the distance, we have considered two sensors : Ultrasonic and IR sensor.
To choose the appropriate sensor for our need, we have compared both the sensors.

	ULTRASONIC SENSOR	IR SENSOR
RANGE	2 cm – 4 m	10 cm – 80 cm
USAGE	Provides us accurate distance measurements	Useful to detect the objects.
BEAM WIDTH	30 degrees	75 degrees
UNIT COST	130 INR	750 INR

As Ultrasonic provides us an accurate distance measurement than IR, we chose **Ultrasonic ranging module HC-SR04** that provides 2cm – 400cm non-contact measurement function, the ranging accuracy can reach to 3mm.



Dig 7. Block Diagram

- **LPC2148**
  Microcontroller
- **Ultrasonic sensor**

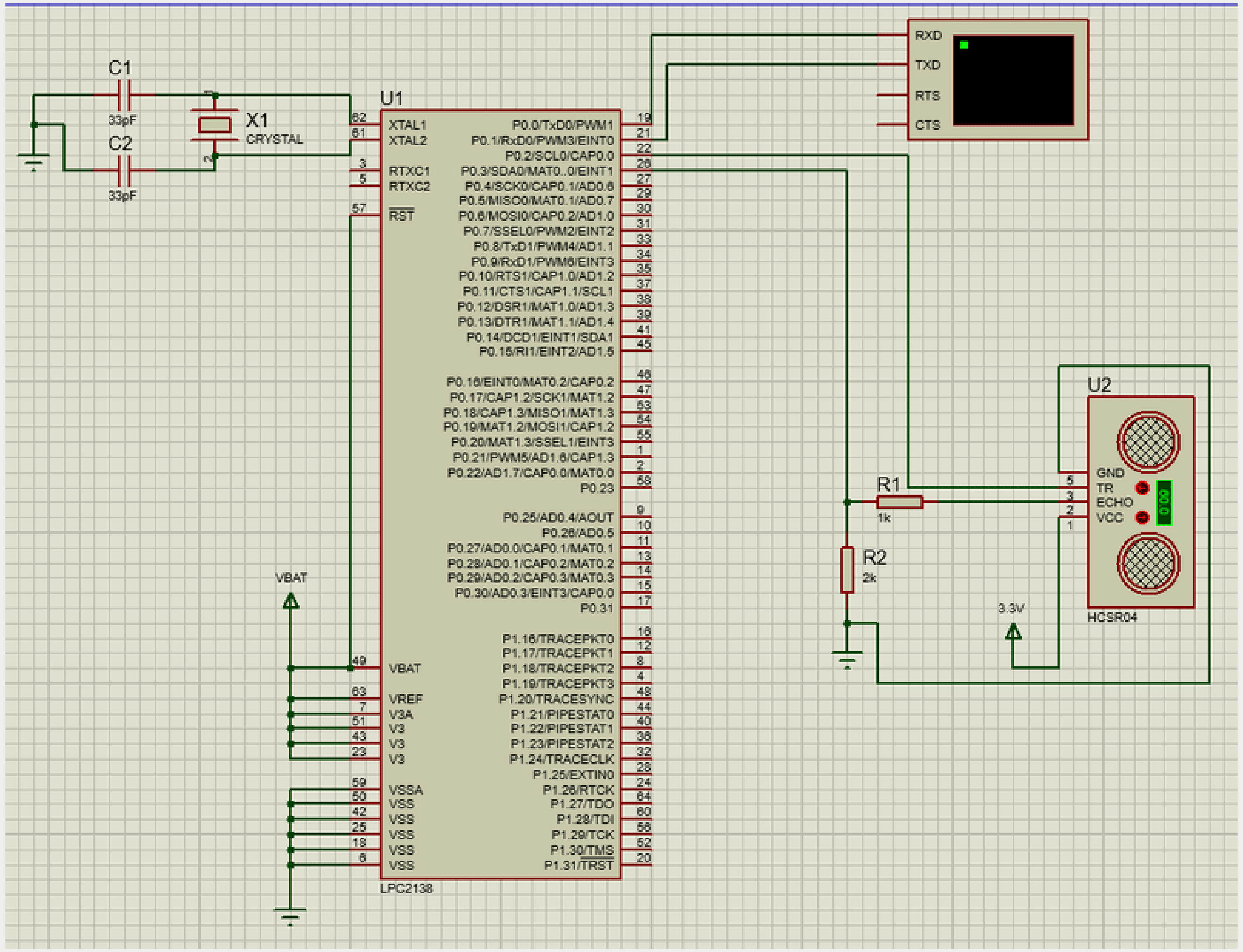
It is an electronic device that measures the distance of a target object by emitting ultrasonic sound waves, and converts the reflected sound into an electrical signal. Ultrasonic sensors have two main components: the transmitter (which emits the sound using piezoelectric crystals) and the receiver (which encounters the sound after it has traveled to and from the target).

- **Serial Communication Port**

A serial port connection can be used for inter-processor communication within a system or for communication with different parts of a system. We use Serial Communication Monitor (UART1 Terminal in Keil) to observe the simulation and output of our code.

Rubric 3 : **Hardware design [CO2,CO3]**

Design a foolproof system using sensors and I/O devices for particular application



CONNECTIONS

TxD	→	P0.0
RxD	→	P0.1
Trigger Pin	→	P0.2
Echo Pin	→	P0.3
Vcc	→	5V

Rubric 4 : **System Software development [CO1,CO2,CO3]**

Modular software programming with proper comments

In main.c file :

```
#include <lpc214x.h>
#include <stdio.h>
#include "lib_funcs.h"
#include "StringUART0Trans.c"
#include "FloatToString.c"
#define TRIG (1<<2) //P0.2
#define ECHO (1<<3) //P0.3

int main(void)
{
 initUART0();  //Initialize UART0 for retargeted printf()
 initClocks(); //Set PCLK = CCLK = 60Mhz - used by: UART, Timer and ADC
 initTimer0(); //Init Timer for delay functions
 int echoTime=0;
 float distance1;
 float distance2;

 IO0DIR |= TRIG;    //Set P0.2(TRIG) as output
 IO0DIR &= ~(ECHO); //Set P0.3(ECHO) as input (explicitly)
 IO0CLR |= TRIG;    //Set P0.2 LOW initially

  UART0_String_Tx("\n\t\t\t FUEL LEVEL MONITOR ");
  //Output 10us HIGH on TRIG pin
  IO0SET |= TRIG;
  delayUS(10);
  IO0CLR |= TRIG;

  while(!(IO0PIN & ECHO)); //Wait for a HIGH on ECHO pin
  startTimer0(); //Start counting
  while(IO0PIN & ECHO); //Wait for a LOW on ECHO pin
  echoTime = stopTimer0()/10000; //Stop counting and save value(us) in echoTime

  distance2 = 13.1-(echoTime/58.2); //Find the distance
UART0_String_Tx("\n Distance after filling the tank : ");
delayMS(100);
float_to_string(distance2);
 for( int i=0;i<7;i++)
 {
  uartWrite(r[i]);
 }
UART0_String_Tx(" cm ");
float finalliters = (431.3674714 * distance2) - 345.134817;
float_to_string(finalliters);
UART0_String_Tx("\t Litres after filling the tank : ");
 for( int i=0;i<7;i++)
 {
  uartWrite(r[i]);
 }
UART0_String_Tx(" ml ");

delayMS(1000); //wait 1 second for next update

//Output 10us HIGH on TRIG pin
IO0SET |= TRIG;
delayUS(10);
IO0CLR |= TRIG;

while(!(IO0PIN & ECHO)); //Wait for a HIGH on ECHO pin
startTimer0(); //Start counting
while(IO0PIN & ECHO); //Wait for a LOW on ECHO pin
echoTime = stopTimer0()/10000; //Stop counting and save value(us) in echoTime

distance1 = 13.1-(echoTime/58.2); //Find the distance
UART0_String_Tx("\n Distance before filling the tank : ");
delayMS(100);
float_to_string(distance1);
  for( int i=0;i<7;i++)
 {
  uartWrite(r[i]);
 }
UART0_String_Tx(" cm ");
float initliters = (431.3674714 * distance1) - 345.134817;
float_to_string(initliters);
UART0_String_Tx("\t Litres before filling the tank : ");
for( int i=0;i<7;i++)
{
 uartWrite(r[i]);
}
UART0_String_Tx(" ml ");

// Difference between the distances of both initial and final levels.
float diff_dist = distance2 - distance1;
float_to_string(diff_dist);
UART0_String_Tx("\n Diff : ");
for( int i=0;i<7;i++)
{
 uartWrite(r[i]);
}
UART0_String_Tx(" cm ");
float diff_litres = (431.3674714 * diff_dist) - 345.134817;
float_to_string(diff_litres);
UART0_String_Tx("\n You got ");
for( int i=0;i<7;i++)
{
 uartWrite(r[i]);
}
UART0_String_Tx(" ml of fuel ");
}
```

In lib_funcs.c file :

```
#include <lpc214x.h>
#include <stdio.h>
#include "lib_funcs.h"
struct __FILE
{
 int dummyVar; //Just for the sake of redefining __FILE, we won't we using it anyways ;)
};
FILE __stdout; //STDOUT
FILE __stdin;  //STDIN
int fputc(int c, FILE * stream)
{
 uartWrite(c); //Transmit Character
 return c; //return the character written to denote a successful write
}

int fgetc(FILE * stream)
{
 char c = uartRead(); //Receive Character
 uartWrite(c); //To echo Received characters back to serial Terminal
 return c;
}

void uartWrite(char c)
{
 while (!(U0LSR & THRE)); //wait till the THR is empty
 if( c == '\n' ) //Send <CR+LF>
 {
  U0THR = CARRIAGE_RETURN;
  while(!(U0LSR & THRE));
  U0THR = LINE_FEED;
 }
 else
 {
  U0THR = c;
 }
}
char uartRead(void)
{
 while(!(U0LSR & RDR)); // wait till any data arrives
 return U0RBR;
}
```

```
void initTimer0(void)
{
 //Assuming PCLK = 60Mhz
 T0CTCR = 0x0;
 T0PR = 60-1; //60 clock cycles @60Mhz = 1 us
 T0TCR = 0x02; //Reset Timer
}

void delayUS(unsigned long microseconds) //Using Timer0
{
 T0TCR = 0x02; //Reset Timer
 T0TCR = 0x01; //Enable timer
 while(T0TC < microseconds); //wait until timer counter reaches the desired delay
 T0TCR = 0x00; //Disable timer
}

void delayMS(unsigned int milliseconds)
{
 delayUS(milliseconds * 1000);
}

void startTimer0(void)
{
 T0TCR = 0x02; //Reset Timer
 T0TCR = 0x01; //Enable timer
}

unsigned int stopTimer0(void)
{
 T0TCR = 0x00; //Disable timer
 return T0TC;
}

void initUART0(void)
{
 //Assuming PCLK = 60Mhz
 PINSEL0 = 0x5;  /* Select TxD for P0.0 and RxD for P0.1 */
 U0LCR = 0x80;
 U0DLM = 0x00;
 U0DLL = 0x62;
```

```
U0FDR = (MULVAL<<4) | DIVADDVAL; /* MULVAL=15(bits - 7:4) , DIVADDVAL=0(bits - 3:0)  */
 //U0LCR &= 0x0F; // Set DLAB=0 to lock MULVAL and DIVADDVAL
 U0LCR = 0x03;
 //BaudRate is now ~9600 and we are ready for UART communication!
}


void setupPLL0(void)
{
 //Note : Assuming 12Mhz Xtal is connected to LPC2148.

 PLL0CON = 0x01;
 PLL0CFG = 0x24;
}

void feedSeq(void)
{
 PLL0FEED = 0xAA;
 PLL0FEED = 0x55;
}

void connectPLL0(void)
{
 while( !( PLL0STAT & PLOCK ));
 PLL0CON = 0x03;
}

void initClocks(void)
{
 setupPLL0();
 feedSeq(); //sequence for locking PLL to desired freq.
 connectPLL0();
 feedSeq(); //sequence for connecting the PLL as system clock

 //SysClock is now ticking @ 60Mhz!

 VPBDIV = 0x01; // PCLK is same as CCLK i.e 60Mhz

 //PLL0 Now configured!
}
```

In FloatToString.c file :

```
unsigned char r[6];

int n_tu(int number, int count)
{
   int result = 1;
   while(count-- > 0)
      result *= number;

   return result;
}

unsigned char float_to_string(float f)
{
   long long int length, length2, i, number, position, sign;
   float number2;

   sign = -1;   // -1 == positive number
   if (f < 0)
   {
      sign = '-';
      f *= -1;
   }

   number2 = f;
   number = f;
   length = 0;  // Size of decimal part
   length2 = 0; // Size of tenth

   /* Calculate length2 tenth part */
   while( (number2 - (float)number) != 0.0 && !((number2 - (float)number) < 0.0) )
   {
       number2 = f * (n_tu(10.0, length2 + 1));
       number = number2;

       length2++;
   }
```

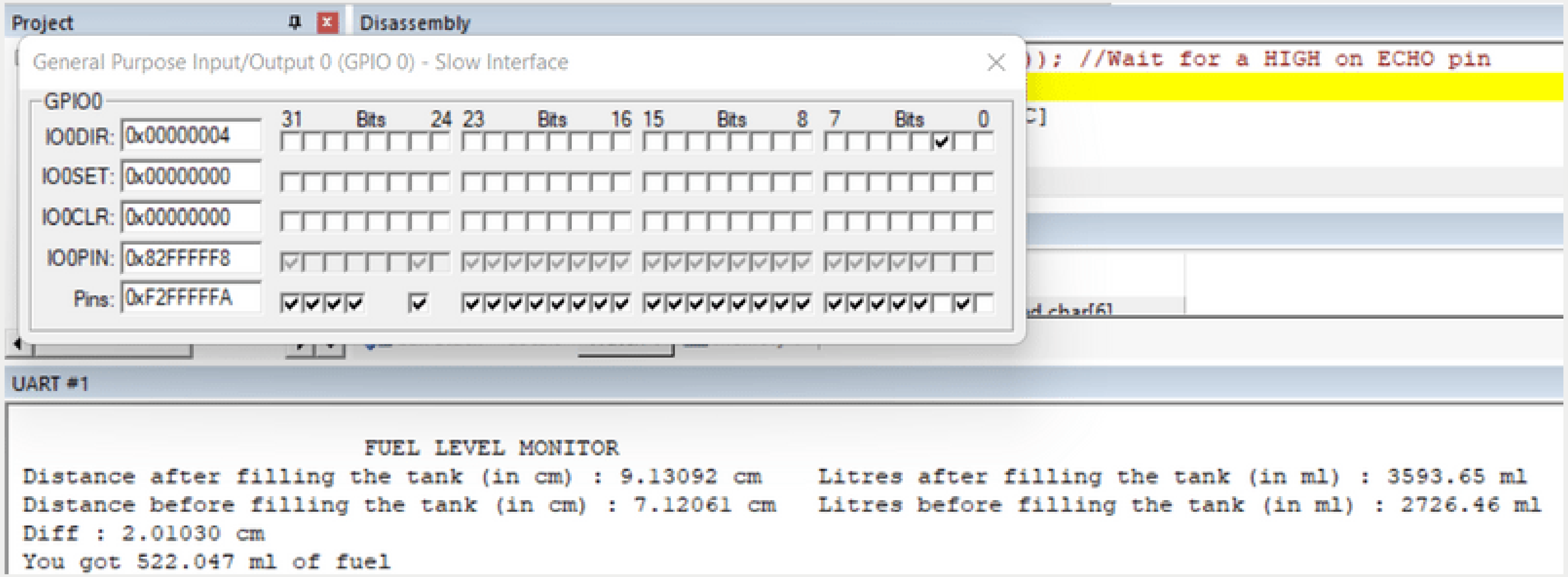
```
   /* Calculate length decimal part */
   for (length = (f > 1) ? 0 : 1; f > 1; length++)
      f /= 10;

   position = length;
   length = length + 1 + length2;
   number = number2;
   if (sign == '-')
   {
      length++;
      position++;
   }

   for (i = length; i >= 0 ; i--)
   {
      if (i == (length))
         r[i] = '\0';
      else if(i == (position))
         r[i] = '.';
      else if(sign == '-' && i == 0)
         r[i] = '-';
      else
      {
         r[i] = (number % 10) + '0';
         number /=10;
      }
   }
}
```

In StringUART0Trans.c file :

```
#include "lib_funcs.h"
void UART0_String_Tx(unsigned char *serial_ptr)
{
 unsigned int i;
 for(i=0; serial_ptr[i] != '\0'; i++)
 {
  uartWrite(serial_ptr[i]);
 }
}
```

Rubric 5 : **Simulate the total system (Firmware) [CO1,CO2,CO3,CO4]**

Integration of hardware and software i.e. total system integration and working software simulation



Thank you!!