User Manual on

Communication Practical – 2

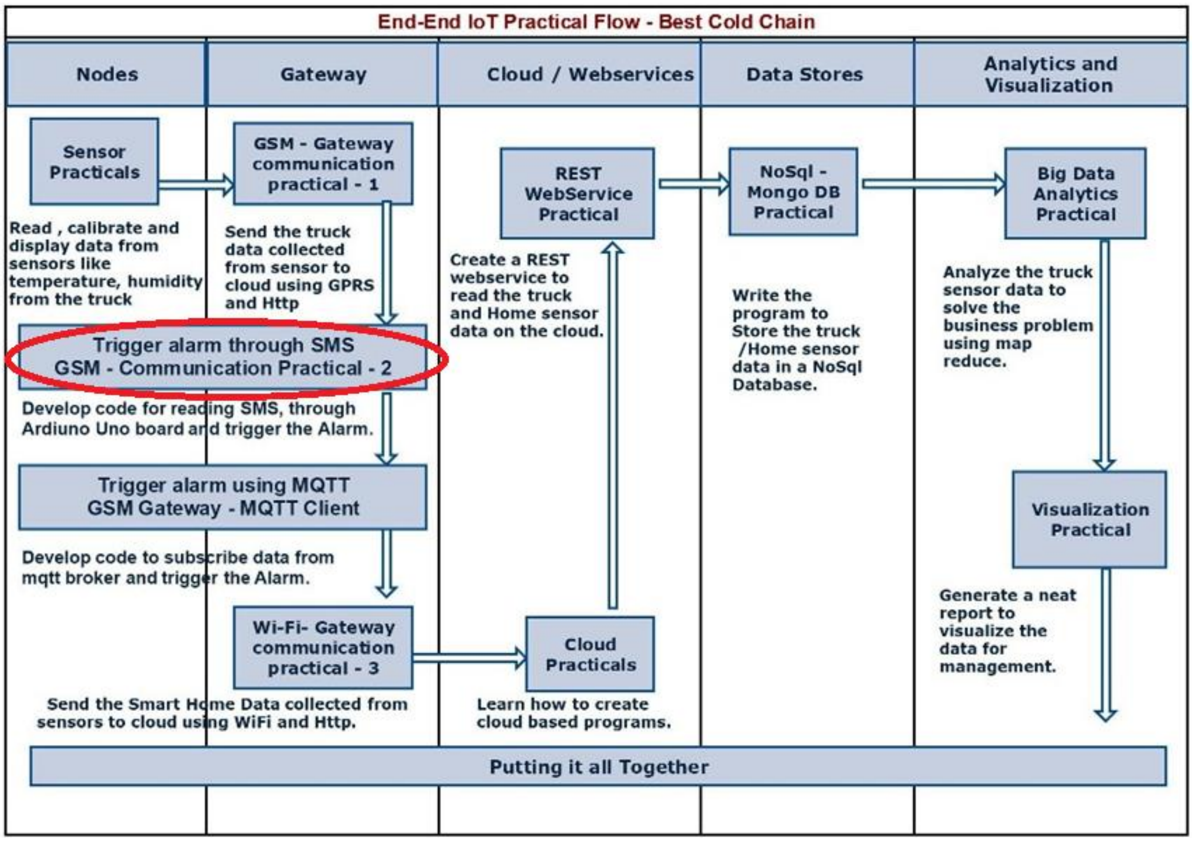
Trigger alarm using SMS

Practical's Objective:

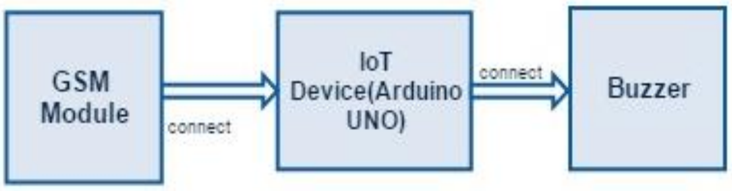
To trigger alarm using SMS.

The user has to send a SMS from the mobile. Then, the gateway will read the SMS, and based on the SMS it sounds a buzzer accordingly.

1. End-End IoT Flow diagram:


End-End IoT Practical Flow - Best Cold Chain
Nodes
Gateway
Cloud / Webservices
Data Stores
Analytics and Visualization
Sensor Practicals
Read , calibrate and display data from sensors like temperature, humidity from the truck
GSM - Gateway communication practical - 1
Send the truck data collected from sensor to cloud using GPRS and Http
REST WebService Practical
Create a REST webservice to read the truck and Home sensor data on the cloud.
NoSql - Mongo DB Practical
Write the program to Store the truck /Home sensor data in a NoSql Database.
Big Data Analytics Practical
Analyze the truck sensor data to solve the business problem using map reduce.
Trigger alarm through SMS
GSM - Communication Practical - 2
Develop code for reading SMS, through Ardiuno Uno board and trigger the Alarm.
Trigger alarm using MQTT
GSM Gateway - MQTT Client
Develop code to subscribe data from mqtt broker and trigger the Alarm.
Visualization Practical
Generate a neat report to visualize the data for management.
Wi-Fi- Gateway communication practical - 3
Send the Smart Home Data collected from sensors to cloud using WiFi and Http.
Cloud Practicals
Learn how to create cloud based programs.
Putting it all Together

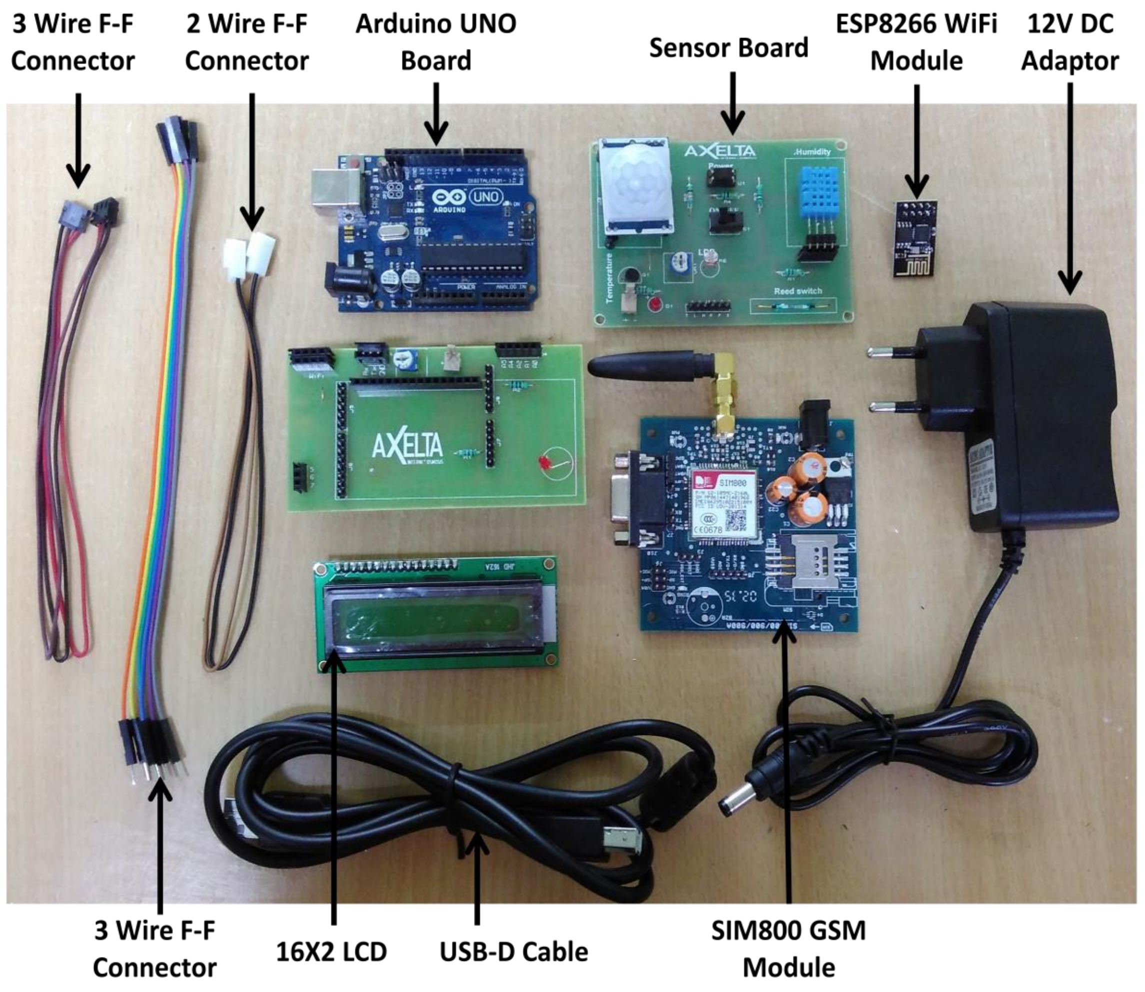
GSM at Node Flow Diagram:



2. GSM Alarm Practical:

GSM Alarm Practical - 1			
Step1	Step2	Step3	Step4
Connect power to Arduino UNO Board	Connecting GSM Module to Arduino UNO Board	Programming to the Arduino UNO Board	Send SMS to the Node

3. Hardware requirements:



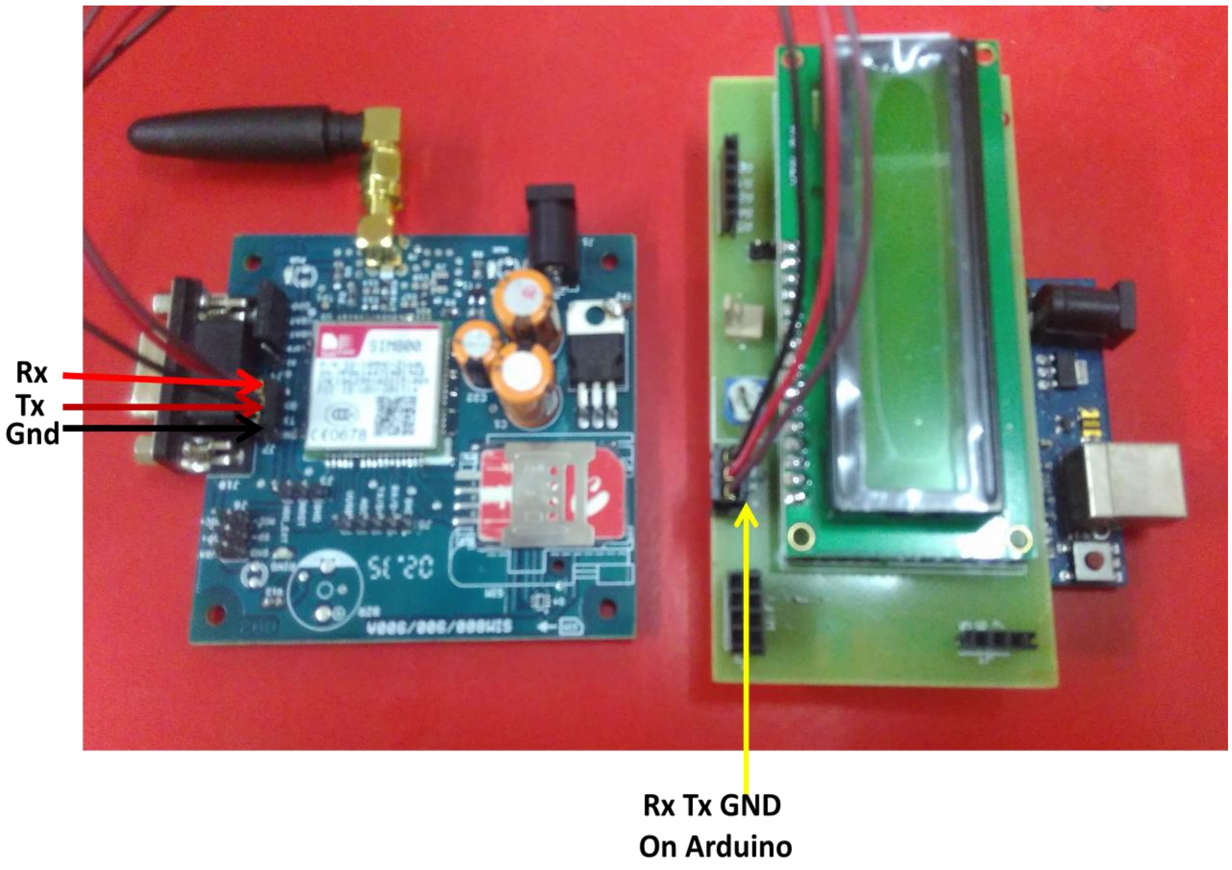
Software Requirement:
same as that of previous Practical

Arduino UNO Board Connections :

Don't change the sensor connections. They will also remain same .

Connecting GSM Modem to Arduino UNO Board

- The Arduino UNO board has a 3-wire connector on the left side.
- Connect this **3-wire connector to the GSM board**. It is a one to one placement connector, which can be connected only in one way.
- Insert a **SIM card in the GSM modem** and make sure it has GPRS activated and sufficient recharge/money in it.
- Connect **12V Adapter to GSM Modem** & power the board then wait for signal (We can find on board signal by checking the Blinking LED on GSM board which shifts its frequency from fast pace to slow pace) .
- Now connect USB cable from Arduino UNO board to PC.



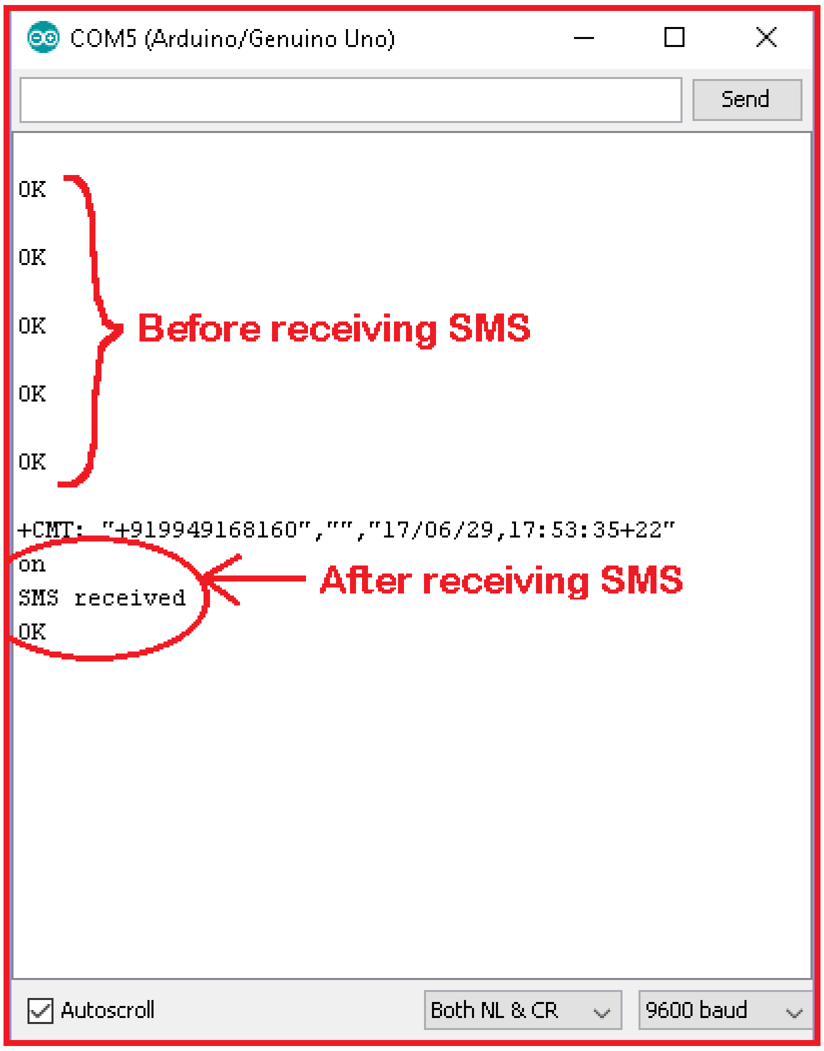
Programming:

Create a new sketch [program]. You can find the "**Communication practical2_code.txt**" by the link: https://drive.google.com/file/d/0Bz7GE98wyjOxbXlRb0NESDhOLTQ/view

Type "**on**" and Send SMS from the mobile phone to the SIM used in the Node .

The Arduino UNO board will check the received data and makes an LED to glow (or sounds a buzzer) as an indication of alarm.

If you set correct **baud rate as 9600 for GSM** You will get the output as given below :



```
OK

OK

OK

OK

OK

+CMT: "+919949168160","","17/06/29,17:53:35+22"
on
SMS received
OK
```

Step by Step Explanation for ‘C’ program in Arduino.

Step1: Create a new sketch

Step2: Include the header file “**LiquidCrystal.h**”, to use the functions related to LCD

```
#include<LiquidCrystal.h>
```

Step3: Include the header file “**SoftwareSerial.h**”, to use the functions related to UART.

```
#include <SoftwareSerial.h>
```

Step4: Make the global declaration of the UART pins with the microcontroller pins of the Arduino board, i.e., RX, TX with pins 3, 4 using “SoftwareSerial”.

```
SoftwareSerial GSM_Serial(3,4);
```

Step5: Make the global declaration of the LCD pins with the microcontroller pins of the Arduino board, i.e., RS, E, D4, D5, D6, D7, with pins 8, 9, 10, 11, 12, 13 using “LiquidCrystal”.

```
LiquidCrystal lcd(8,9,10,11,12,13);
```

Step6: Initialize a variable as integers with the pin numbers to which the buzzer is connected.

```
int BUZ=A3;
```

Step7:

Initialize a string with the predefined message.

```
String response_SMS = "on";
```

Step8:

8.2 Write a **“setup”** function and initialize the UART communication with 9600 baud rate and the LCD as 16 columns with 2 rows with the help of **“lcd.begin”** function.

8.1 Sets the data rate in bits per second (baud) for serial data transmission. Both sides of the serial connection (i.e. the Arduino and your computer)

8.3 Define the mode of the buzzer pin as “OUTPUT” using **“pinMode”** function.

8.4 Then, make the buzzer low.

```
void setup()
```

```
  {
    GSM_Serial.begin(9600);
    lcd.begin(16,2);
   Serial.begin(9600);
    pinMode(BUZ,OUTPUT);
    digitalWrite(BUZ,LOW);
  }
```

Step9:

9.1 Start a **“gsminit”** function.

9.2 Using **“GSM_Serial.println”** function, send the command “AT”

9.3 Write 2000 mille seconds delay for each GSM AT command.

9.3 Using “**GSM_Serial.println**” function, send the command “ATE0” for making the ECHO mode off.

9.4 Write 2 seconds delay.

9.5 Using **“GSM_Serial.println”** function, send the command “AT+CMGF=1” for making the Text mode.

```
void gsminit()
  {
    GSM_Serial.println("AT");
    delay(2000);
    GSM_Serial.println("ATE0");
    delay(2000);
    GSM_Serial.println("AT+CMGF=1");
    delay(2000);
  }
```

step10:

10.1 Start a **"gsm_read()"** function.

10.2 using **"GSM_Serial.available"** we can check Serial port whether SMS data is there or not.

10.3 using **"Serial.println"** function, we can print the String "gsm_input" on User system using debugging port.

10.3 using **"gsm_input.trim"** function Trim all the leading / trailing white spaces from the input

10.4 using **"find_string"** function, we can Find the starting location of the message location number. the message location number is preceeded by a ','

10.6 using **" gsm_input.toLowerCase"** function, we can change the String gsm_input into lower case.

10.7 using **"find_stirng"** function, we can check if the message contains the text stored in the response1 variable.

```
void gsm_read()
      {
```

```
    if(GSM_Serial.available())
    {
     String gsm_input="";
     while(GSM_Serial.available()) // read the data into a variable as long as the buffer is
not empty
     {
       gsm_input+= (char)GSM_Serial.read();
     }
     Serial.print(gsm_input);
     gsm_input.trim();         // Trim all the leading / trailing white spaces from the input
     if(find_string(gsm_input,"+CMTI: "))
     {
       int loc = find_char_loc(gsm_input,','); // Find the starting location of the message
location number. the message location number is preceeded by a ','
      String no = gsm_input.substring(loc+1); // extract the message location number from
the CMTI response
       GSM_Serial.println("AT+CMGR="+no);
      }
     gsm_input.toLowerCase();
     if(find_string(gsm_input,response1)) // Check if the message contains the text stored
in the response1 variable
     {
       lcd.clear();
       lcd.print("SMS received-on");
       lcd.setCursor(0,1);
       lcd.print("ALARM ON ");
       delay(1000);
       digitalWrite(BUZ,HIGH);
       delay(10000);
       lcd.clear();
       lcd.print("waiting for SMS");
       lcd.setCursor(0,1);
       lcd.print("ALARM OFF");
       digitalWrite(BUZ,LOW);
       GSM_Serial.println("AT+CMGD=1,4");
       delay(3000);
     }
    }
  }
```

Step11:

11.1 Start a function with the name **“loop”.** This is the Main loop/function of the whole code/program.

11.2 Call the “lcd.clear” function to erase any old/garbage data on the lcd.

11.3 With the help of "lcd.setCursor" function, set the lcd cursor position to 0th column and 0th row.

11.4 With the help of "lcd.print" function, display some string data – "**Axelta systems**".

11.5 With the help of "lcd.setCursor" function, set the lcd cursor position to 0th column and 1st row.

11.6 With the help of "lcd.print" function, display some string data – "**Waiting for sms**".

11.6 Call the delay function for 1000 milli seconds.

11.7 Start an infinite "while" loop.

11.8 Check if the SMS data is received from Serial port using "GSM_Serial.available" function.

11.9 Continue the loop until the SMS data is available.

11.10 Read the received SMS data from "GSM_Serial.read" function and save to the variable string "response_SMS".

11.11 Using the function **"find_string**", check if the received data is equal to "on".

11.12 If the condition is true, make the buzzer pin HIGH.

11.13 Start a "for" loop for 8 times and display the message "**SMS-Received", "ALARAM ON"** and clear the lcd with some time delay.

11.14 Then, make the buzzer LOW and display the message **"Waiting for a SMS","ALARAM OFF"** on the lcd.

```
void loop()
   {
      lcd.clear();
      lcd.setCursor(0,0);
      lcd.print("Axetla Systems");
      gsminit();
      GSM_Serial.println("AT+CMGD=1,4");
      lcd.clear();
      lcd.print("waiting for SMS");
      lcd.setCursor(0,1);
      lcd.print("ALARM OFF");
      while(1)
      {
       gsm_read();
      }
   }
```

Step12:

12.1 Write a Boolean function as “find_string” with String parameters as “base, search”.
12.2 Initialize a variable as integer and save the length of the base string
12.3 Start a “for” loop and iterate from the beginning of the base string till the end minus length of the substring.
12.4 Check if the extracted Substring Matches the Search String
12.5 If it matches exit the function with a true value
12.6 If the above loop did not find any matches, control would come here and return a false value

```
 boolean find_string(String base, String search)
 {
  int len = search.length();
  for(int m = 0; m<((base.length()-len)+1);m++)
  {
   if(base.substring(m,(m+len))==search)
   {
    return true;
   }
  }
  return false;
 }
boolean find_char_loc(String base, char search)
  {
   for(int m = 0; m < base.length();m++)// Iterate from the beginning of the base string till the end minus length of the substring
   {
    if(base[m]==search) // Check if the character Matches the Search character
    {
     return m; // if it matches exit the function with the current location value
    }
   }
   return 0; // if the above loop did not find any matches, control would come here and return a zero value
  }
```