

Chapter 1

INTRODUCTION

Safety is the major concern of today's life. Usually many industries are thermo industries and high prone to fire accidents. Even many shopping malls are subjected to fire accidents. Many employees are risking their lives for daily wages. Many industries are striving hard to make their employees' lives more safer. This fire alarm system keeps on running 24*7 and keeps on scanning for fire arise and uses GSM module to communicate with the fire stations. It also displays message about fire status which helps employees in industries or customers in malls to evacuate the place.

1.1 AIM OF THE PROJECT

The aim of our project is to build a **Sensor system** which can transmit information provided by active signals about temperature. By having information about temperature, we check for fire occurrence in industries or shopping malls using **Arduino** and if any fire occurs which will be sensed by sensor system, a message will be sent to fire station through **GSM** module and message will be displayed on **LCD** board thus saving many lives.

1.2 OBJECTIVES

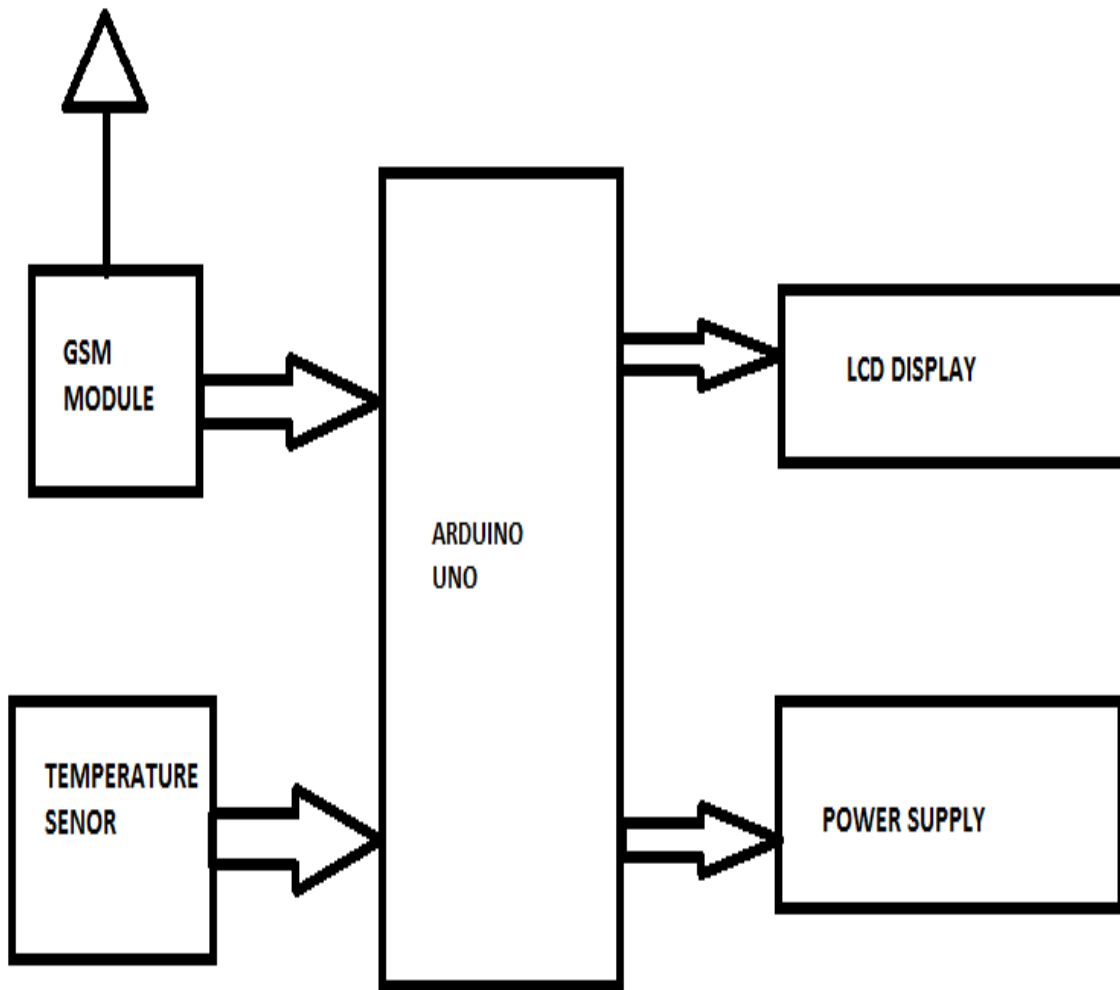
The objective of our project are designing and implementing GSM based fire alarm system using arduino

- 1.to analyse and test the GSM module.
- 2.to analyse and test the arduino.

1.3 APPLICATIONS

1. SMS based Fire Alarm system are very useful in remote locations where human interaction is limited. Such systems are useful in mines, industrial areas, factories etc.
2. SMS based Fire Alarm system helps to monitor locations and alert during fire that occurs in night time.
3. Quick Actions to shut down Fire – 90% of fire damages occur due to lack of early fire detection. A fire attack is usually silent and people will know about fire only when it has spread across a large area. SMS based Fire Alert system gives warning immediately to multiple mobile numbers and hence remedy actions can be taken quickly. This helps to prevent major damages and losses created by a fire accident.

1.4 BLOCK DIAGRAM



It consist of RDL GSM Module , arduino uno ,temperature sensor LM35 ,12v dc adapter, 16x2 lcd display.

Chapter 2

HARDWARE REQUIREMENT

Hardware components used

- Arduino uno
- Gsm module 900A
- Temperature sensor LM35
- LCD display
- 12v dc adapter
- 10k Rpot
- Patch chords
- Bread board

2.1 Arduino Uno



The Uno is a microcontroller board based on the [ATmega328P](#). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

The Uno can be programmed with the [Arduino Software](#) (IDE). Select "Arduino/Genuino Uno" from the Tools > Board menu (according to the microcontroller on your board). The ATmega328 on the Uno comes preprogrammed with a [bootloader](#) that allows you to upload new code to it

without the use of an external hardware programmer. It communicates using the original STK500 protocol ([reference](#), [C header files](#)).

2.11 TECHNICAL SPECIFICATION

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
Length	68.6 mm
Width	53.4 mm
Weight	25 g

2.12 PIN DISCRIPTION

The power pins are as follows:

- **Vin.** The input voltage to the Uno board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.** This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins.
- **IOREF.** This pin on the Uno board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs to work with the 5V or 3.3V.

Each of the 14 digital pins on the Uno can be used as an input or output, using `pinMode()`, `digitalWrite()`, and `digitalRead()` functions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller.

In addition, some pins have specialized functions:

- **Serial:** 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- **External Interrupts:** 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the `attachInterrupt()` function for details.
- **PWM:** 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the `analogWrite()` function.
- **SPI:** 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
- **LED:** 13. There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

- TWI: A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.

The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the `analogReference()` function. There are a couple of other pins on the board:

- AREF. Reference voltage for the analog inputs. Used with `analogReference()`.
- Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

2.13 COMMUNICATION

- The Uno has a number of facilities for communicating with a computer, another Uno board, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The 16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. The Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

2.2 GSM Module



GSM/GPRS Modem - RS232 is built with Dual Band GSM/GPRS engine - SIM900A, works on frequencies 900/1800 Mhz. The Modem comes with RS232 interface, which allows the user to connect PC as well as microcontroller with RS232 Chip (MAX232). The baud rate is configurable from 9600 - 115200 through AT command. The GSM/GPRS Modem is having internal TCP/IP stack to enable the user to connect with internet via GPRS. It is suitable for sending SMS, Voice as well as DATA transfer application in M2M interface. The onboard Regulated Power supply allows the user to connect wide range of unregulated power supply. Using this modem, user can make audio calls, sending SMS, Reading SMS, attend the incoming calls and internet etc., through simple AT commands.

A **GSM modem** is a specialized type of modem which accepts a SIM card, and operates over a subscription to a mobile operator, just like a mobile phone. From the mobile operator perspective, a GSM modem looks just like a mobile phone

GSM GPRS SIM 900A module with Stub Antenna and SMA connector- USB Features:

- Dual-Band GSM/GPRS 900/ 1800 MHz
- RS232 interface for direct communication with computer or MCU kit.
- FT232 USB driver.
- Power controlled using 29302WU IC.
- ESD Compliance.
- Enable with MIC and SPEAKER socket.
- With slid in SIM card tray.

- With Stub antenna and SMA connector.
- It can directly interface to Raspberry Pi and Beagle Bone
- Input Voltage: 12V DC.
- High quality PCB FR4 Grade with FPT Certified.

2.3 Temperature sensor LM35

LM35 is a precision IC **temperature sensor** with its output proportional to the temperature (in °C). The sensor circuitry is sealed and therefore it is not subjected to oxidation and other processes. With **LM35**, temperature can be measured more accurately than with a thermistor. It also possess low self heating and does not cause more than 0.1 °C temperature rise in still air.

The operating temperature range is from -55°C to 150°C. The output voltage varies by 10mV in response to every °C rise/fall in ambient temperature, *i.e.*, its scale factor is 0.01V/°C.

2.31 PIN DIAGRAM:



2.32 PIN DISCRIPTION:

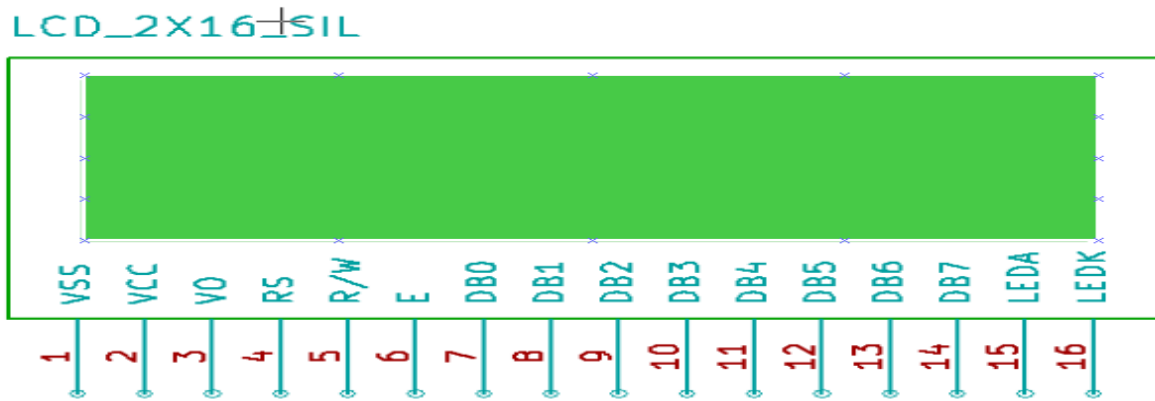
Pin No	Function	Name
1	Supply voltage; 5V (+35V to -2V)	Vcc
2	Output voltage (+6V to -1V)	Output
3	Ground (0V)	Ground

2.4 LCD Display

LCD-Liquid crystal display screen is an electronic display module and find a wide range of applications. A 16x2 lcd display is a very basic module and is very commonly used in various devices and circuits. These modules preferred over seven segment displays and other multisegment LED's. The reason being LCDs are economical; easily programmable ; have no limitations of displaying special and even custom characters (unlike in seven segments),animations and so on.

A 16x2 lcd means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix. This LCD has two registers , namely, command and data. The command register stores the command instructions given to the LCD. A command is an instruction given to LCD to do a predefined task like initializing it, clearing its screen , setting the cursor position, controlling display etc. The data is the ASCII value of the character to be displayed on the LCD.

2.41 PIN DIAGRAM



2.42 PIN DISCRIPTION

ITEM	SYMBOL	LEVEL	FUNCTIONS
1	VSS	0V	Power Ground
2	VDD	+5V	Power Supply For Logic
3	V0	—	Contrast adjust
4	RS	H/L	H:data L:command
5	R/W	H/L	H:read L:write
6	E	H, H→L	Enable singnal
7-14	DB0-DB7	H/L	Data Bus
15	LEDA	+5V	Power supply For LED Backlight
16	LEDK	0V	

2.5 12V DC ADAPTER

This is used to power the gsm module.it convert 230v mains into 12v 2A dc.

Chapter 3

SOFTWARE REQUIREMENTS

Software requirements are

- Arduino 1.6.0 IDE software
- RDL GSM/GPRS Utility software

3.1 Arduino 1.6.0 IDE software

Arduino is an [open-source](#) computer hardware and software company, project and user community that designs and manufactures [microcontroller](#)-based kits for building digital devices and interactive objects that can sense and control objects in the physical world.

Arduino programs may be written in any programming language with a compiler that produces binary machine code. Atmel provides a development environment for their microcontrollers, AVR Studio and the newer Atmel Studio.^{[18][19]}

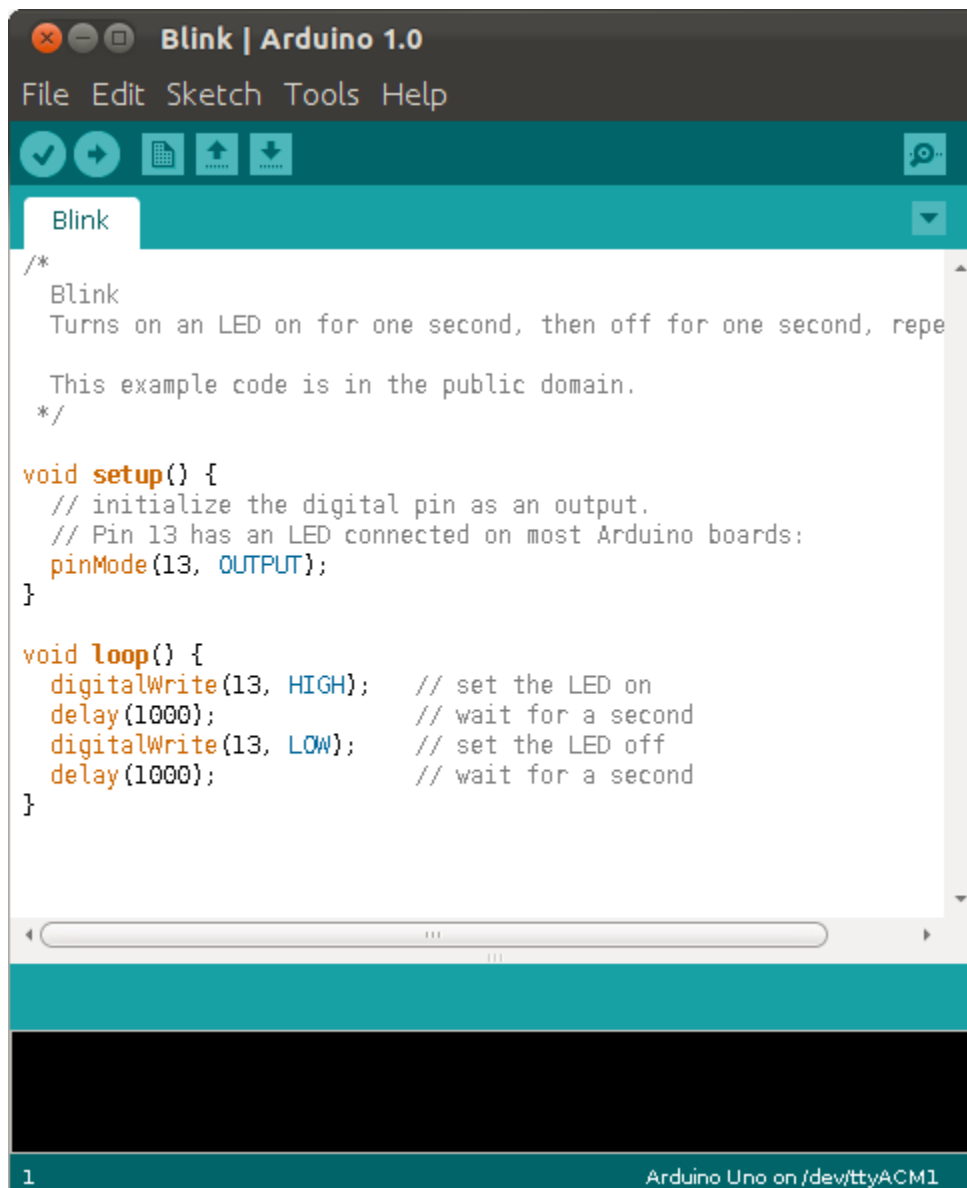
The Arduino project provides the Arduino integrated development environment (IDE), which is a cross-platform application written in Java. It originated from the IDE for the Processing programming language project and the Wiring project. It is designed to introduce programming to artists and other newcomers unfamiliar with software development. It includes a code editor with features such as syntax highlighting, brace matching, and automatic indentation, and provides simple one-click mechanism for compiling and loading programs to an Arduino board. A program written with the IDE for Arduino is called a "sketch".^[20]

The Arduino IDE supports the C and C++ programming languages using special rules of code organization. The Arduino IDE supplies a software library called "Wiring" from the Wiring project, which provides many common input and output procedures. A typical Arduino C/C++ sketch consists of two functions that are compiled and linked with a program stub *main()* into an executable cyclic executive program:

- *setup()*: a function that runs once at the start of a program and that can initialize settings.
- *loop()*: a function called repeatedly until the board powers off.

After compilation and linking with the GNU toolchain, also including with the IDE distribution, the Arduino IDE employs the program *avrdude* to convert the executable code into a text file in

hexadecimal coding that is loaded into the Arduino board by a loader program in the board's firmware.

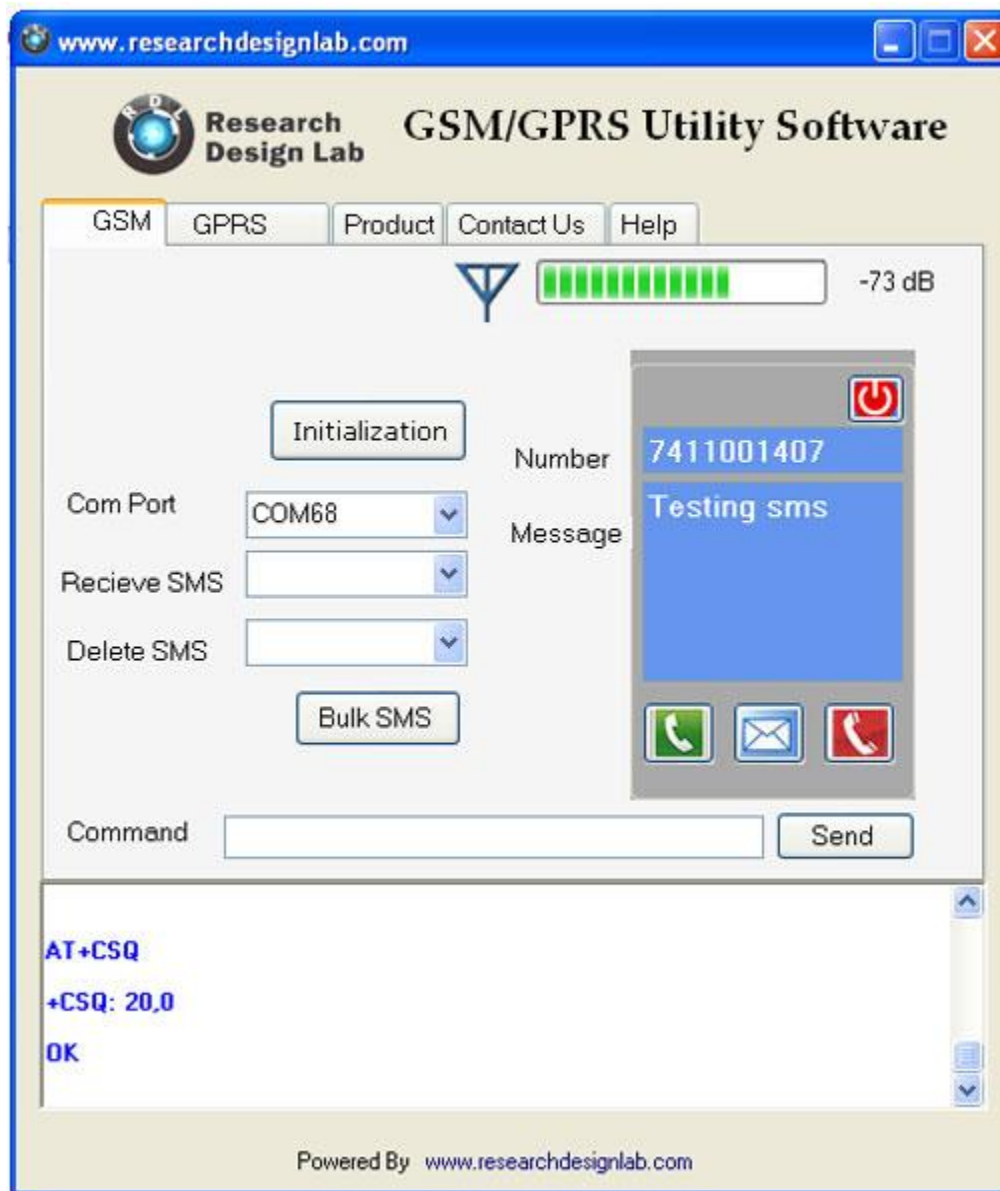
A screenshot of the Arduino IDE interface. The title bar reads "Blink | Arduino 1.0". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for opening, saving, and uploading. The main text area shows the "Blink" sketch, which is a standard Arduino program that turns an LED on and off in a repeating cycle. The code is as follows:

```
/*  
  Blink  
  Turns on an LED on for one second, then off for one second, repeatedly.  
  
  This example code is in the public domain.  
  */  
  
void setup() {  
  // initialize the digital pin as an output.  
  // Pin 13 has an LED connected on most Arduino boards:  
  pinMode(13, OUTPUT);  
}  
  
void loop() {  
  digitalWrite(13, HIGH);   // set the LED on  
  delay(1000);              // wait for a second  
  digitalWrite(13, LOW);    // set the LED off  
  delay(1000);              // wait for a second  
}
```

The status bar at the bottom shows "1" on the left and "Arduino Uno on /dev/ttyACM1" on the right.

3.2 RDL GSM/GPRS UTILITY SOFTWARE

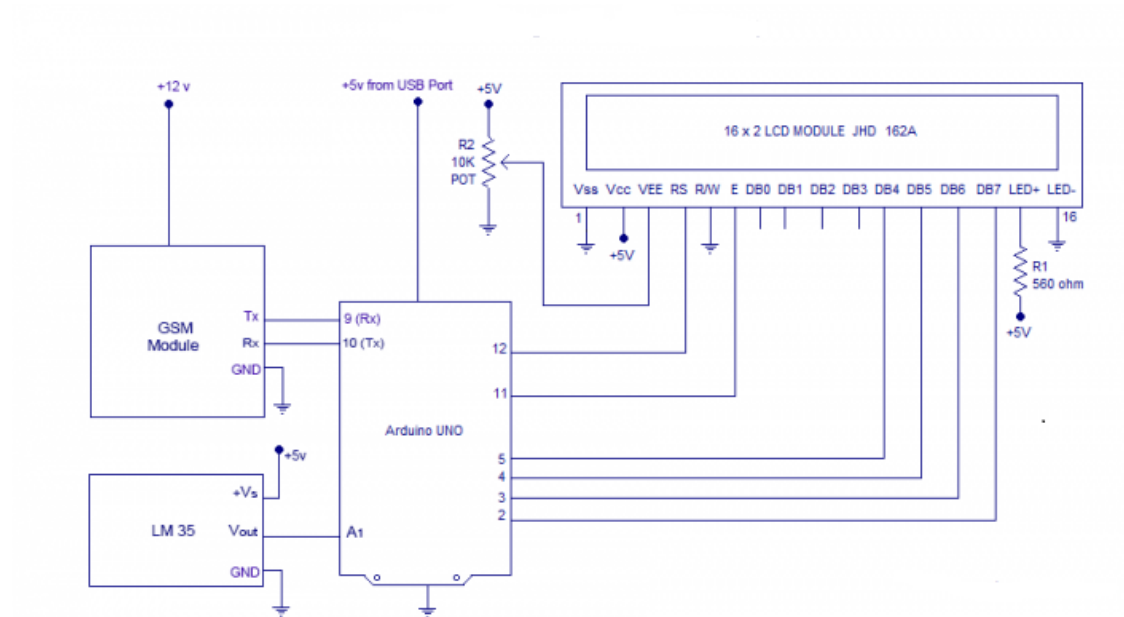
This software is used to interface GSM with the computer and to check the working of the gsm.



Chapter 4

WORKING

4.1 CIRCUIT



4.2 C CODE

```
#include <SoftwareSerial.h>
#include<LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
SoftwareSerial mySerial(6, 7);

int sensor=A1;
float temp_read,Temp_alert_val,Temp_shut_val;
int sms_count=0,Fire_Set;

void setup()
{
    pinMode(sensor,INPUT);
    mySerial.begin(9600);
    Serial.begin(9600);
    lcd.begin(16,2);
    delay(500);
}

void loop()
{
    CheckFire();
    CheckShutDown();
}
```

```
void CheckFire()
{
  lcd.setCursor(0,0);
  lcd.print("Fire Scan - ON");
  Temp_alert_val=CheckTemp();
  if(Temp_alert_val>45)
  {
    SetAlert(); // Function to send SMS Alerts
  }
}

float CheckTemp()
{
  temp_read=analogRead(sensor); // reads the sensor output (Vout of LM35)
  temp_read=temp_read*5; // converts the sensor reading to temperature
  temp_read=temp_read/10; // adds the decimal point
  return temp_read; // returns temperature value in degree celsius
}

void SetAlert()
{
  while(sms_count<3) //Number of SMS Alerts to be sent
  {
    SendTextMessage(); // Function to send AT Commands to GSM module
  }
  Fire_Set=1;
  lcd.setCursor(0,1);
  lcd.print("Fire Alert! SMS Sent!");
```



```
}
```

```
void CheckShutDown()
```

```
{
```

```
if(Fire_Set==1)
```

```
{
```

```
Temp_shut_val=CheckTemp();
```

```
if(Temp_shut_val<28)
```

```
{
```

```
lcd.setCursor(0,1);
```

```
lcd.print("Fire Shut! SAFE NOW");
```

```
sms_count=0;
```

```
Fire_Set=0;
```

```
}}}
```

```
void SendTextMessage()
```

```
{
```

```
mySerial.println("AT+CMGF=1"); //To send SMS in Text Mode
```

```
delay(2000);
```

```
mySerial.println("AT+CMGS=\"+917815017418\"\\r"); // change to the phone number you  
using
```

```
delay(2000);
```

```
mySerial.println("Fire in NEW ROOM!");//the content of the message
```

```
delay(200);
```

```
mySerial.println((char)26);//the stopping character
```

```
delay(5000);
```

```
mySerial.println("AT+CMGS=\"+919448303436\"\\r"); // change to the phone number you  
using
```

```
delay(2000);  
mySerial.println("Fire in NEW ROOM!");//the content of the message  
delay(200);  
mySerial.println((char)26);//the message stopping character  
delay(5000);  
sms_count++;  
}
```

4.3 WORKING

When we develop Fire Alarm Systems or such critical systems, the one important aspect we have to keep in mind is the real world scenario. A **“fire”** can occur any time (24×7). This means our system must constantly monitor fire 24×7 all the month and year. If you look into the program, you will see it has only 2 function calls inside void loop() – that is CheckFire() **and** CheckShutDown()

CheckFire() – is the function which monitors occurrence of a fire 24×7. This function fetches the temperature measured by LM35 and stores it to the variable **Temp_alert_val** for comparison. This temperature value is compared against a set value of **45 degree Celsius**. Usually room temperature is between 25 degree Celsius and 30 degree Celsius in tropical areas. This will vary with continents and locations. You have to change this comparison value by measuring the average room temperature of the installation site!

If fire occurs, room temperature will cross 45 degrees (within seconds) and an inner subroutine **SetAlert()** will be invoked. SetAlert() is the function that controls number of SMS alerts sent to each mobile number loaded in the program. The number of SMS alerts sent can be altered by changing the stopping condition of while loop. The stopping condition **sms_count<3** – means **3 SMS alerts** will be sent to each mobile number. If you want to send 5 alerts, just change the stopping condition to **sms_count<5** – you got it ? The function to send SMS (using AT Commands) – **SendTextMessage()** will be called **3 times** if SMS alert **count is 3**. This function SendTextMessage() will be invoked as many times as the number SMS alerts set in the program.

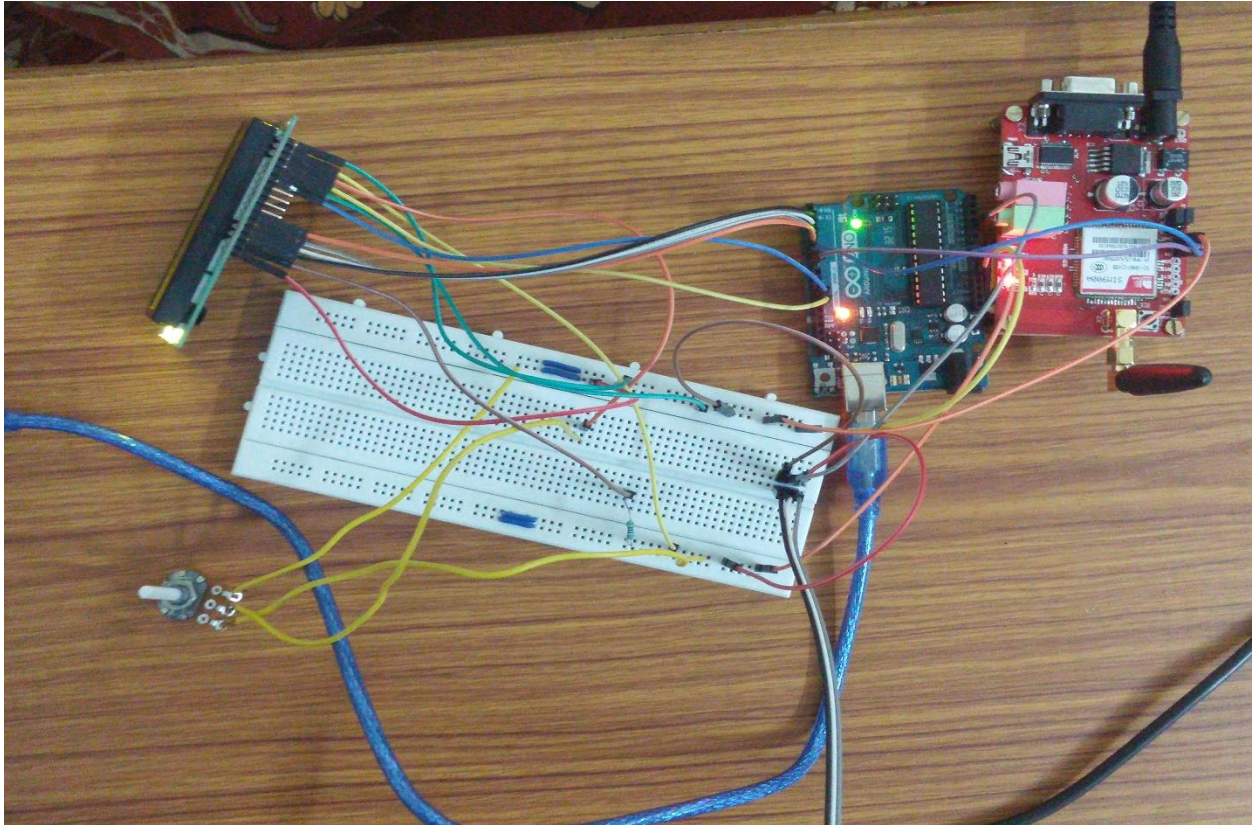
Note:- We have limited the number of SMS alerts using the stopping condition. Once a fire accident occurs and the set number of SMS alerts has been sent, the system will not send any more SMS! The system assumes that its job is over by sending SMS. Humans has to come and shut down the fire. After sending alerts, the system will start monitoring Shut Down process. Once the Fire has been shut down, system will reactivate its SMS alert settings by **resetting the sms_count** variable back to zero.

CheckShutDown() – is the function which monitors if fire was shut down. We need to entertain this function only if a fire accident has occurred. To limit the entry to the statements inside this routine, we have introduced a variable **Fire_Set**. This variable status will be set to value **1** when a fire accident occurs (check the statement inside **SetAlert()**). The statements inside **CheckShutDown()** will be executed only if the value of **Fire_Set==1**. (If not there was no fire accident and we don't need to waste time executing **ShutDown** checking statements). We consider the fire has been shut down once room temperature is back to normal. So if our variable **Temp_shut_val** falls less than 28 degrees, we consider fire has been shut and things are safe. We start our **FireAlarm** monitoring again with **SMS Alerts** active! (We reset the **Fire_Set** variable and **sms_count** variable back to zero – which are conditions of normal room status)

Chapter 5

RESULT

5.1 Output in LCD display

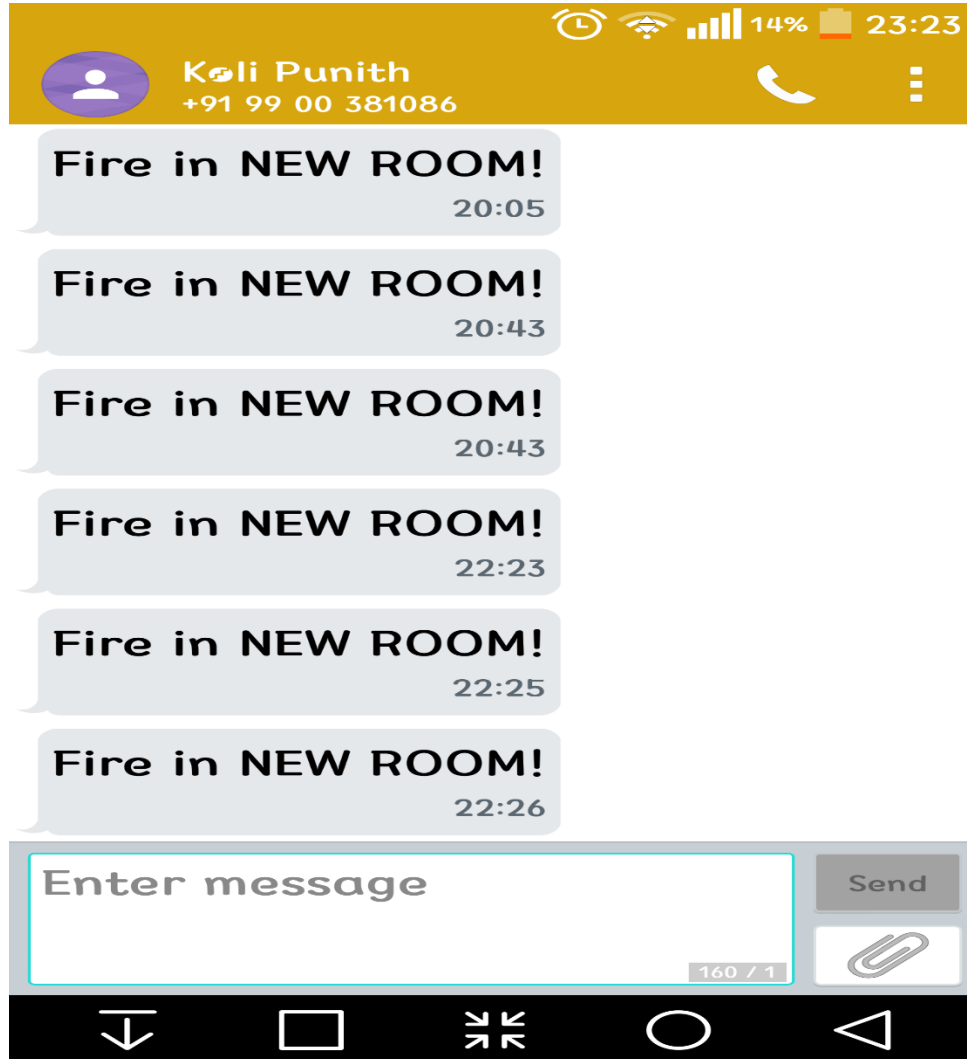


Circuit was successfully implemented as shown in figure above



Continuous fire scanning resulting in detection of fire which result in a message display on lcd.

5.2 screenshot of text message



Screenshot of message received in the entered number.

Chapter 6

CONCLUSION

Temperature monitoring , fire detection and message transmission is demonstrated using implemented fire alarm system.

6.1 Future scope

In future we can add GPS module to this system so that exact location of fire is easily found and exact location information can be sent through text message to fire stations.

Bibliography

<https://www.arduino.cc/>

<http://researchdesignlab.com/wireless/gsm-modem.html>

<http://www.circuitstoday.com/>

LCD display -16x2 LCD display datasheet.

Temperature sensor –LM35 datasheet.