



Expertise
and insight
for the future

Hoang Nguyen

Gas and Fire Detection System With Xbee

Metropolia University of Applied Sciences

Bachelor of Engineering

Degree Programme in Electronics

Bachelor's Thesis

20 April 2021

Author Title	Hoang Nguyen Gas and Fire Detection System with Xbee
Number of Pages Date	47 pages + 2 appendices 20 April 2021
Degree	Bachelor of Engineering
Degree Program	Electronics
Professional Major	
Instructors	Matti Fischer, Principal Lecturer
<p>Nowadays, with the rapid development of science and technology, electronics have been applied in many fields in practice to serve the needs of health care, protection of lives, and property for people. The main goal of this project was targeted in sensing gas leakage and temperature variation to alert user.</p> <p>Measurements of gas leaking from sensor MQ-2 as well as temperature from 10k thermistor temperature sensor are focused on this project. Several communication standards were used, including serial connection, wireless communication (Xbee and GSM module), LCD display, and bootloader Arduino. The microcontroller used in this project is Arduino Uno R3 with ATmega328P-AU chip. Program for Arduino is written in C programming language. A GSM module with a sim card is used to send message to user smartphone when detecting gas leakage and fire.</p> <p>Two small printed circuit board were successfully created in this project. The sensors unit was capable of sensing and demonstrating gas leakage and temperature variation. The microcontrollers unit performed as expected and data was transferred effectively.</p>	
Keywords	Arduino, Atmega328P, MQ-2, wireless connection, LCD, Xbee, GSM module, printed circuit board

Contents

List of Abbreviations

1	Introduction	1
2	Theoretical Background	2
2.1	What is Microcontroller	2
2.2	Sensors in Human Life	3
2.2.1	What is Temperature Sensor	4
2.2.2	What is Gas Sensor	6
2.3	Communication Protocols for Embedded System	9
2.3.1	Parallel Communication Protocol in Embedded System	10
2.3.2	Serial Communication Protocol in Embedded System	11
2.4	GSM Communication	13
2.5	Printed Circuit Board	14
2.5.1	Substrate (FR4)	15
2.5.2	Copper	15
2.5.3	Solder Mask	15
2.5.4	Silkscreen	16
3	Components	17
3.1	XBee	17
3.2	Rectifier	18
3.3	Voltage Regulator	20
3.3.1	Linear Voltage Regulators	20
3.4	GSM Module	21
3.5	Arduino Uno R3	22
3.6	16x2 LCD Display	24
3.7	MQ2 Gas Sensor	25
4	Removing Arduino Uno Board from The System and Programing Atmega328 Using Arduino IDE	27
5	What is Gas and Fire Detection System	33
6	Creating PCBs for Gas and Fire Detection System	35
6.1	Receiver Part	35

6.2	Transmitter Part	38
7	How GFDS detects fire and gas leakage	41
8	Advantages and Disadvantages	44
9	Conclusion	46
	References	47

Appendices

Appendix 1. Code for Receiver Part

Appendix 2. Code for Transmitter Part

List of Abbreviations

GFDS	Gas and Fire Detection System.
PCB	Printed circuit board.
GSM	Global System for Mobile Communications.
NTC	Negative temperature coefficient.
PTC	Positive temperature coefficient.
GND	Common ground.
VCC	Collector supply voltage
LED	Light emitting diode.
RF	Radio frequency.
U.FL	A miniature coaxial RF connector for high-frequency signals manufactured by Hirose Electric Group.
AC	Alternating current.
DC	Direct current.
FET	Fiel-effect transistor.
MOSFET	Metal-oxide-semiconductor field-effect transistor.
BJT	Bipolar Junction Transistor.
LCD	Liquid-crystal Display.
USB	Universal serial bus.
ICSP	In-circuit serial programming.
PWM	Pulse-width modulation.
TTL	Transistor-transistor logic.
IC	Integrated circuit.

ISP In-system programming.

IDE Integrated Development Environment.

1 Introduction

As people's lives are improved, gas stoves or gas products as fuel for cooking are gaining popularity. Besides the convenience of gas, another issue when using gas is safety. When people are in direct contact with gas (exceeding a certain permissible limit of amount) for a long time, it is possible to get gas poisoning and die. Not only that, gas leaking into the air can easily catch fire and cause fire and explosion, seriously affecting the safety of the user as well as those around. Therefore, detecting and handling gas leakage and fire incidents is an essential issue for people who regularly use gas. This is the reason why this project was born.

The original goal of this project is covered only sensing gas leakage, temperature variation and alert danger to user. Two PCBs were also created. The Gas and Fire Detection System device works as a physical interacting device which senses gas and temperature variation. It includes three essential divisions: receiver, transmitter, and GSM module. Data collected from sensors will be transferred wirelessly to smartphone via GSM module.

Arduino was chosen to be used on this project to set up ATmega328P-AU chips playing an important role on receiver and transmitter circuit with two Xbees to collect and transfer the data from sensors. Bootloader Arduino trick was used to simplify the whole device.

The opening chapter will discuss theoretical background, where information related to the microcontroller, sensors, chips, PCB can be found. The next chapter is introduction of components that are used on doing this project. Chapter four discusses how the Arduino Uno board is removed from GFDS with a simple trick. Chapter five shows the process of making PCB for the transmitter and the receiver. The next chapter is about how GFDS operates, and chapter seven shows the advantages and disadvantages of GFDS. The final chapter is the conclusion after a long process.

2 Theoretical Background

2.1 What is Microcontroller

A microcontroller is a low-cost, low-power computer which is installed in an integrated circuit to control the system's operation. Its purpose is to read, store information, process information, measure time, conduct a reading, and open a particular structure. Many languages can be used for microcontrollers. However, the commonly used languages are C and Assembly. [1.] A block diagram of a microcontroller is shown in figure 1.

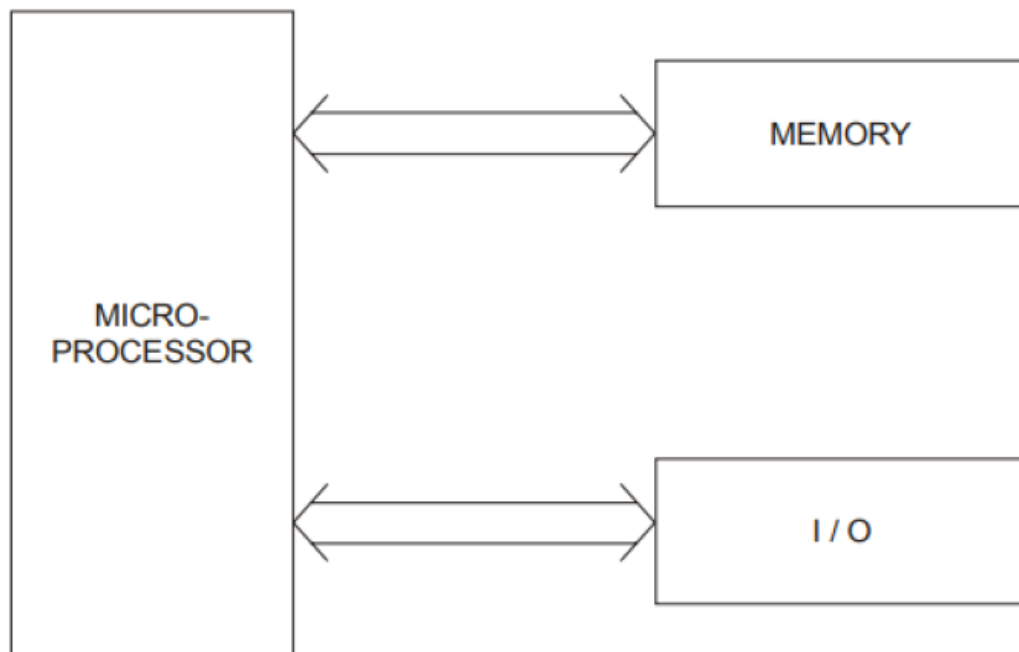


Figure 1. Block diagram of a microcontroller [2]

2.2 Sensors in Human Life

Sensors play an important role in every electronics design since they were created. They are used in homes, offices, vehicles, medicine, and robotics etc. in order to make things simpler and easier. Lights in home or office can be turned on automatically when people present, temperature in a room is automatically modified when it grabs the essential conditions, a warning displayed on screen when car's engine exceeds the typical temperature. These applications are completed due to sensor's functions. The figure 2 below shows some instances of sensor in car.

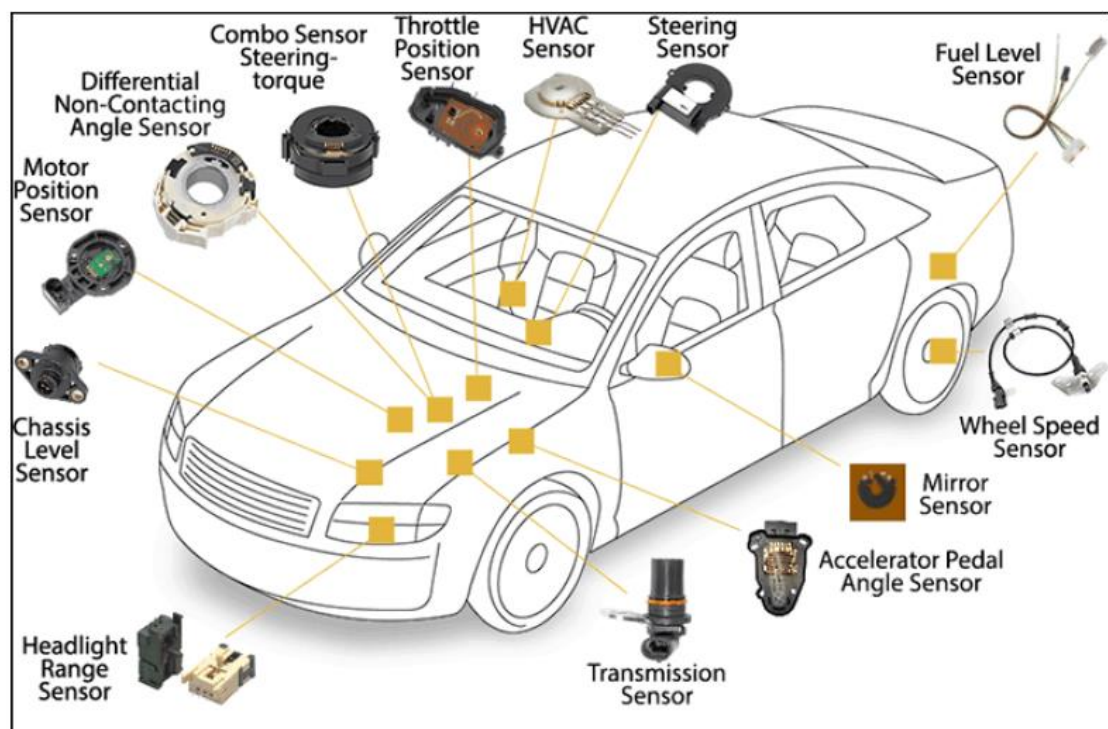


Figure 2. Sensors in an individual car [3]

A transducer is a device that converts energy from one form to a different form. Typically, a transducer converts signals in one form of energy into signals in a different energy form. A sensor is a transducer type in which the physical environment is considered input and produces output in a digital, analog, or optical signal. The output is generated by the device's sensitivity activities concerning light, temperature, humidity, motion, gas, etc. There are several sensor classifications, the first of which includes active and passive sensors. Active sensors require an external power supply (source signal) during operation. On the other hand, passive sensors do not need to be supplied with any

external source signals and directly generate an output response. The second type of classification depends on the sensor's intended use, such as electrical, biological, chemical, radioactive, etc. The third type of classification depends on conversion phenomena with the input and output such as photoelectricity, thermoelectricity, electromagnetic, electromagnetic, etc. Analog and digital sensors are the last categories in the sensor field. Analog sensors produce the analog output. By contrast, digital sensors that work with either digital or discrete data. [4.]

2.2.1 What is Temperature Sensor

The most common sensor in smart system is temperature sensor which estimates the thermal variation of its surrounding environment as input data and generates output data in electronic form to record, monitor, or signal thermal variations from input data. There are different kinds of temperature sensors such as thermistor, resistor temperature detector, infrared, thermocouple. Each kind has its advantages and disadvantages; therefore, choosing a right sensor with reasonable purpose for usage is very necessary.[5.]

Thermistor is a precise and a cost-effective sensor for measuring surrounding temperature variations. Its resistance changes according to surrounding temperature. NTC and PTC are two genres of thermistors. NTC thermistor is popular to measure temperature. Resistance of NTC thermistor decreases as its surrounding temperature grows up. On the other hand, PTC thermistor's resistance increases as its surrounding temperature grows up [6]. NTC thermistor is vulnerable because the material creating thermistors is manganese and oxide nickel.

The relativity between resistance and temperature of a 10kΩ NTC thermistor is shown in figure 3. Due to negative coefficient, its resistance decreases as temperature increases.

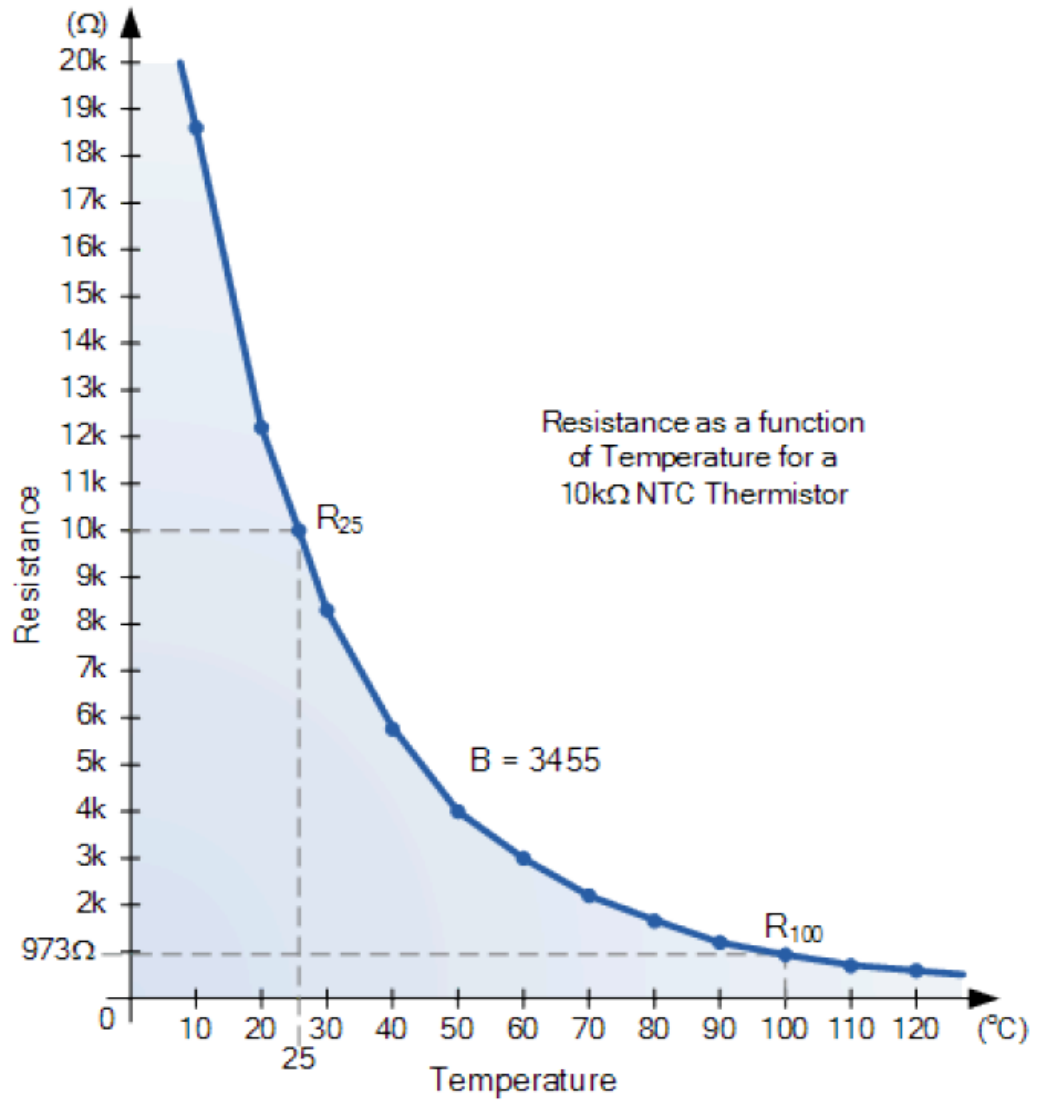


Figure 3. The relation between resistance and temperature [7]

The following equation is thermistor equation:

$$B = \frac{T_2 * T_1}{T_2 - T_1} * \ln \left(\frac{R_1}{R_2} \right) \quad (1)$$

Where:

- B is value of a particular thermistor (given in datasheet of component).
- T1 is the first temperature point in Kelvin (K).
- T2 is the second temperature point in Kelvin (K).
- R1 is the thermistor resistance at temperature T1 in Ohms (Ω)
- R2 is the thermistor resistance at temperature T2 in Ohms (Ω). [7]

2.2.2 What is Gas Sensor

MQ2 is a standard gas sensor in the MQ family of sensors. It is made of metal oxide semiconductor, also known as chemiresistor, as it contains the sensing material; when the gas approaches the material, its resistance will change. The sensor is encased in two layers of fine stainless-steel mesh known as an explosion-proof network. It protects the sensor from exploding caused by the heating element inside the sensor when it senses flammable gas. [8.] The appearance of MQ-2 is demonstrated in figure 4 below.



Figure 4. Appearance of MQ-2 [8]

It also prohibits suspended outside particles from entering the sensor so that only gas particles can pass through the chamber. The mesh is linked to the rest of the body through a copper-plated clamp. [8.] The inside structure of this sensor is illustrated in figure 5 below.

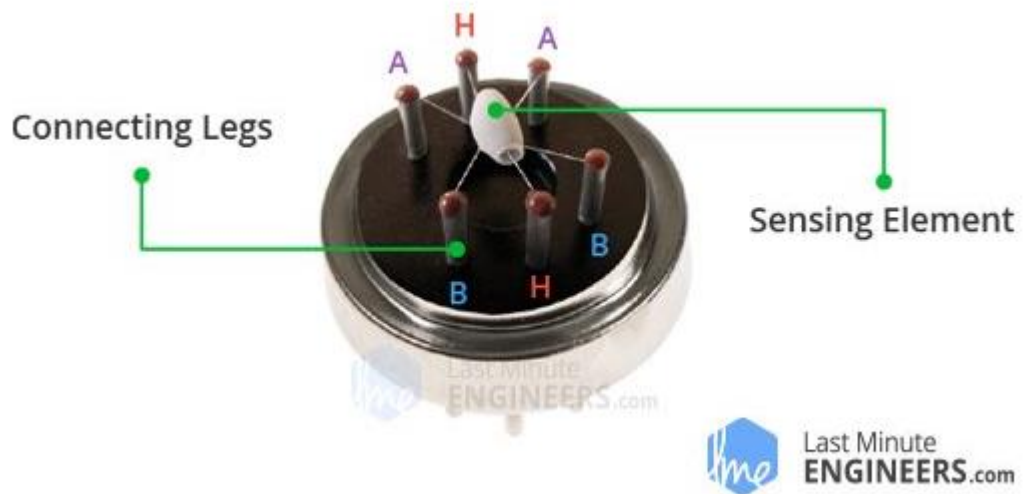


Figure 5. Sensor without outer mesh [8]

The internal structure created by the sensing element and the six connecting legs. Two conductors (H), called electrically conductive alloys, are responsible to heat the sensor element and are connected via a nickel-chromium coil. Four conductors (A and B) are connected via the platinum wire in response to the output signal. These wires are connected to the sensing element's body and transmit minor variations in the current passing through the sensing element. [8.]

Aluminum oxide (Al_2O_3) combines with ceramic and tin dioxide (SnO_2) to make the tubular sensing element. Tin dioxide plays an essential role in the sensitivity to flammable gases. The mission of ceramic substrate is increasing the heating efficiency and guaranteeing that a working temperature is created when the sensor area is heated. Therefore, a heating system includes aluminum-oxide and nickel-chromium coil, while a sensing system consists of coating of tin dioxide and platinum wire. [8.] Sensing element is shown in figure 6 below.

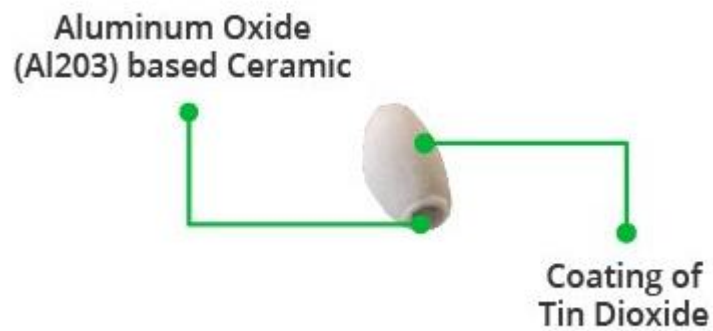


Figure 6. Sensing element [8]

Oxygen is absorbed on the sensor material's surface when an element heats the sensor in the air at a high temperature. The operating theory of MQ gas sensor based on reaction of the donor electrons with oxygen to block the current. Then a reaction of the oxygen atoms with the reducing gases to lessen the surface density of absorbed oxygen; therefore, allowing the current flow through the sensor to generate the analog voltage values. [9.] Working principle of gas sensor is shown in figure 7 below.

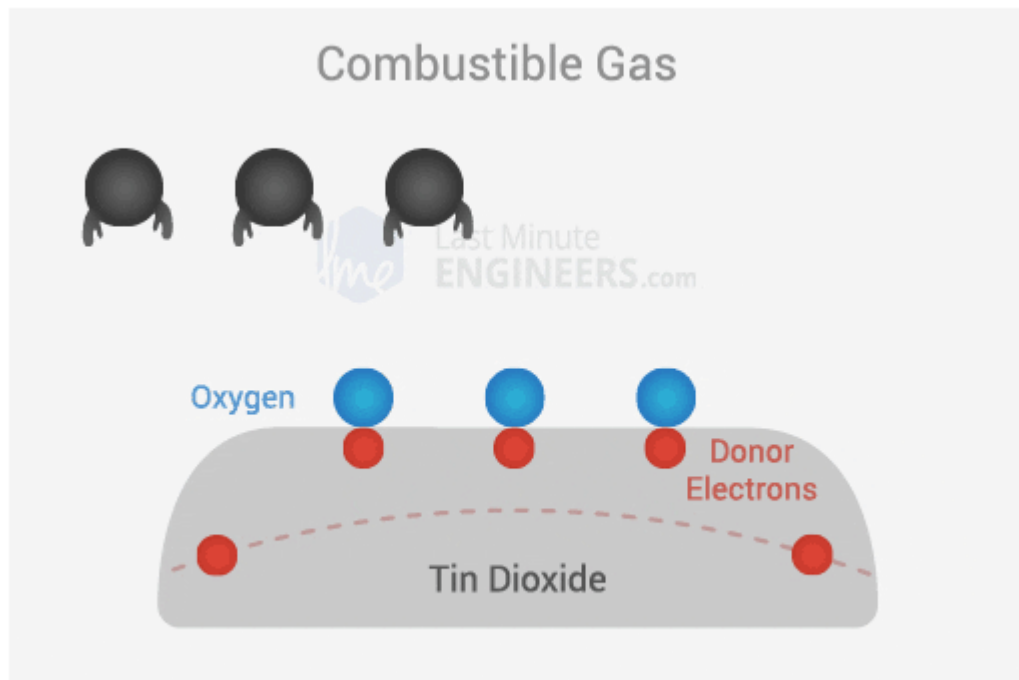


Figure 7. Working principle of gas sensor [8]

2.3 Communication Protocols for Embedded System

The embedded electronics system can operate functionally due to communication protocols. A standard protocol plays a role in circuits or systems to exchange information. Generally, parallel communication and serial communication are two main genres of communication protocols. Deploying parallel or serial communication require various resources as well as they are installed in multiple ways. Parallel communication and serial communication has their own pros and cons.

2.3.1 Parallel Communication Protocol in Embedded System

A parallel communication interface is easily understandable. It transfers all the bits (typically 8 bits) in data simultaneously. Technically, the number of buses and ports is equal to the number of bits transferred. Although it is easy to deploy, its cost is more expensive than the serial communication protocol because it requires much more wires and pins. The implementation of parallel communication is shown in figure 8 below.

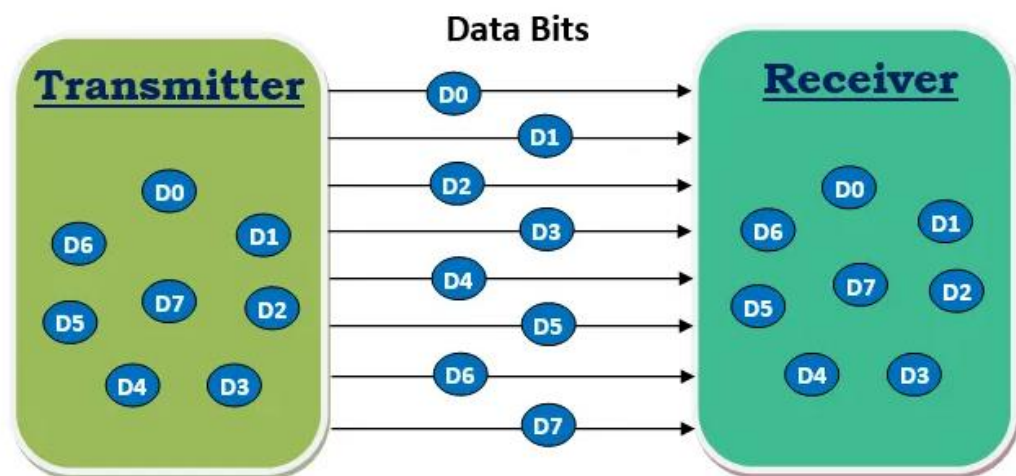


Figure 8. An implementation of parallel communication [12]

Since all bit is transferred simultaneously, the data in parallel communication can be transferred quickly and read easily . This method works in which large amounts of data need to be transferred quickly in real-time. In parallel communication protocol, the wires are very close. However, any signal's strength decreases the further it travels down the line (due to the conductor's resistance); therefore, interference may have occurred when the signal became weak enough. It indicates the possibility of crosstalk occurring. That is why parallel communication is often limited over short distances unless using boosters. [13.]

2.3.2 Serial Communication Protocol in Embedded System

Regarding serial communication, it is more complicated than parallel communication; however, less resources are needed than with parallel communication. It transmits a single bit at a time on a wire (usually never more than four). Serial communication includes two main types: synchronous and asynchronous. A standard clock is mandatory for synchronous communication that controls the speed of data transmission. In contrast, An external clock is not necessary for asynchronous communication for exchanging data. The speed is pre-determined by the baud rate.

There are essential rules or mechanisms for the smooth and perfect transmission of data. Typically, a serial communication interface needs data bit, synchronization bit, parity bit, and baud rate. Baud rate indicates the rate at which data is sent over time. Its units are usually bits per second (bps). Commonly used baud rates are 1200, 2400, 4800, 9600, 19200, 38400, 57600 and 115200. [14.]

A structure of a data packet transferred by serial communication is demonstrated in figure 9. Start bit and stop bits play a role as synchronization bits.



Figure 9. A serial frame [14]

A most standard and simple serial communication system includes two lines: transmitter (TX) and receiver (RX). The transmission is described in figure 10.

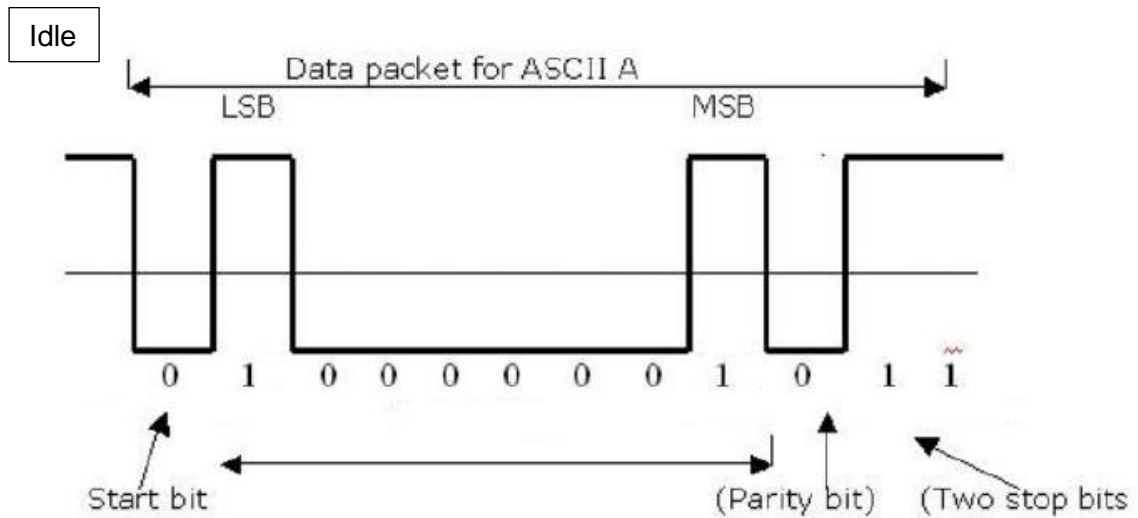


Figure 10. Transmission of a letter "A" by using serial communication [14]

Explanation for a transmission using serial communication:

- Transmit idles high when there is no communication.
- Start bit goes low for 1 bit.
- Data bits are transmitted, least-significant bit goes first.
- Parity bits (optional) using for error checking are sent.
- Stop bits are sent.

2.4 GSM Communication

GSM (Global System for Mobile Communications) is the most common standard for mobile phone systems globally. GSM network is selected by more than 2 billion people in 212 countries and territories. Its popularity allows international roaming arrangements among mobile network operators, allowing subscribers the right to use their phones in many places in the world. GSM differs from its predecessors in terms of signaling, speed, and call quality. Therefore, GSM is a second-generation (2G) mobile phone system. A structure of GSM network is shown in figure 11 below.

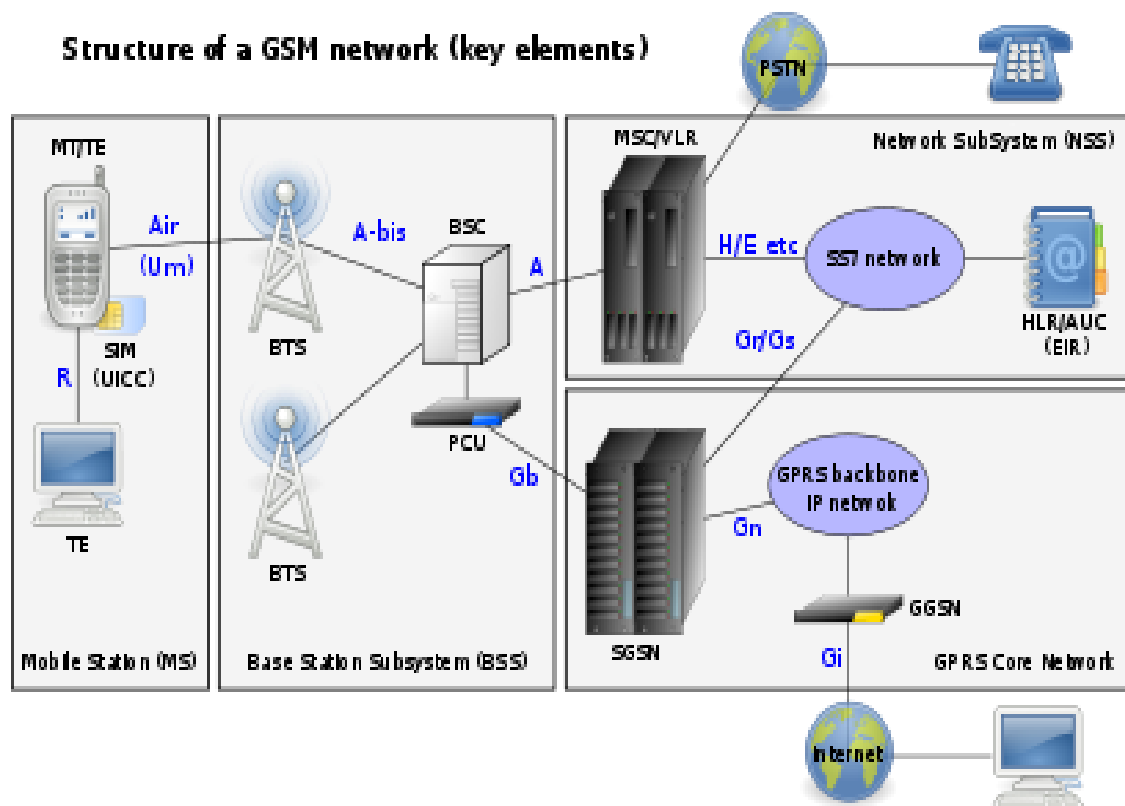


Figure 11. Structure of a GSM network [15]

The network is structured into several discrete sections:

- The Base Station Subsystem.
- The Network and Switching Subsystem.
- The GPRS Core Network.
- The Operations support system (OSS) for maintenance of the network.

GSM modem application with PC or embedded system is described in figure 12 below.

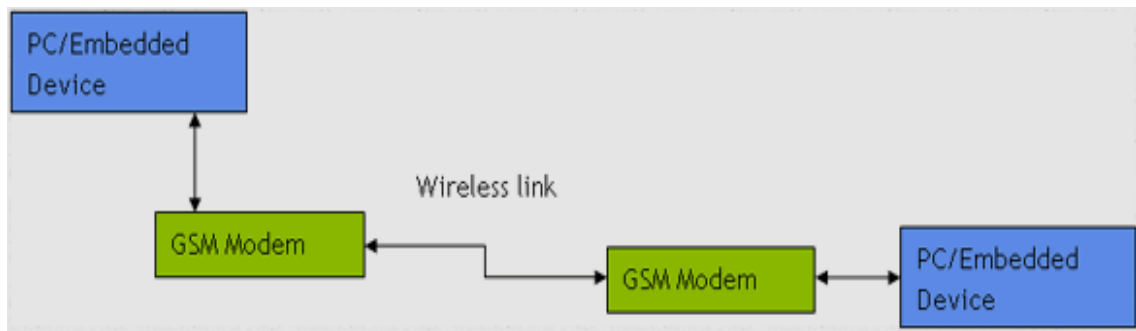


Figure 12. GSM modem application

GSM modem is the intermediate between two embedded devices or wireless devices. For instance, there is an embedded device and a user would like to create a connection from embedded device to a smartphone such as configuring embedded device to send messages to the user smartphone. In this case, GSM modem is responsible for this function.

2.5 Printed Circuit Board

PCB (printed circuit board) is created to mechanically support and electrically connect electronic components such as resistors, capacitors, sensors, and so on. Electronic components are then attached to the board, and etchings are made on its surface, which allows the current to flow through the copper from component to component [16]. A standard PCB should be like a layer cake with alternating layers of various materials laminated together by heat and glue to create a completely single object. The structure of a PCB is shown in figure 13 below.

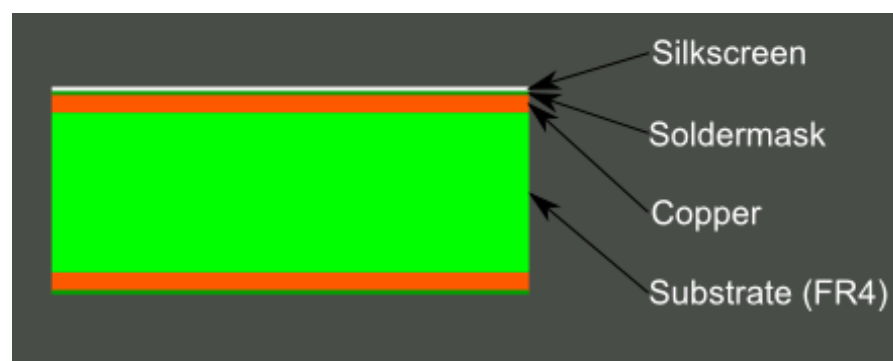


Figure 13. Structure of a PCB [17]

2.5.1 Substrate (FR4)

Basically, the substrate layer is base material and fiberglass. Historically, "FR4" is used most to design fiberglass, which is low-cost material. This solid core gives the PCB its rigidity and thickness. [17.]

2.5.2 Copper

Another layer of PCB is thin copper foil. Technically, copper is attached to both sides of the substrate. In reality, PCB can be few as one layer or as many as sixteen layers or more. Figure 14 below shows the appearance of PCB copper.

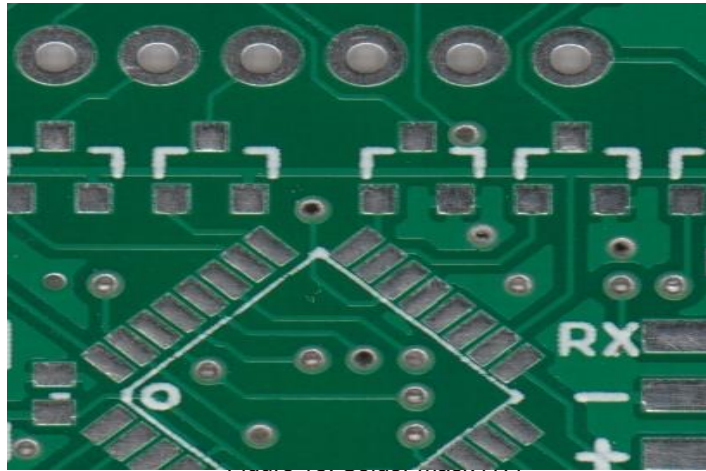


Figure 14. PCB with copper exposed [17]

2.5.3 Solder Mask

The solder mask is the top layer of copper foil. Its color is green and responsibility of solder mask is to prevent oxidation process. This layer also prevents the user from misplaced solder and solder jumper. [17.]

In the figure 15 below, the PCB applies the green solder mask which covers up the small traces but still displaying the silver rings and SMD pads for soldering.



2.5.4 Silkscreen

Above the solder mask layer is the white silkscreen layer. The solder can add letters, numbers, and symbols by silkscreen on PCB, allowing for easier assembly and instructions to better understand the board. Silkscreen labels is often used to point out the function of each pin or LED. [17.] Figure 16 below shows the appearance of silkscreen

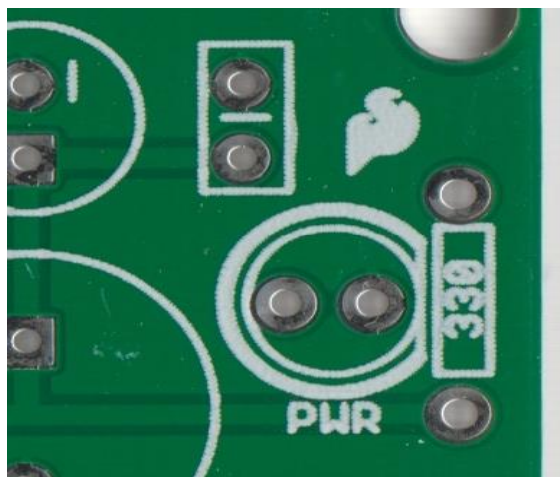


Figure 16. Silkscreen [17]

3 Components

3.1 XBee

The XBee is a low-cost module manufactured by Digi International that is primarily used as a radio communications receiver and transmitter. It uses mesh communication protocols following the ZigBee IEEE 802.15.4 standard. XBee supports peer-to-peer as well as point to multi-point network communications wirelessly with the speed of 250 kbits/s. [11.] An XBee is demonstrated in figure 17 below.

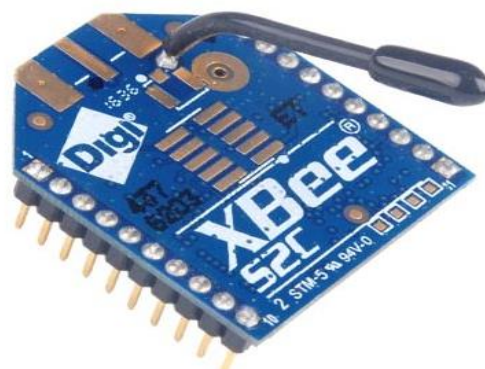


Figure 17. XBee S2C

Most of the XBee modules on the market get 2.4 GHz bandwidth with a variety of antennas. XBee modules usually come with several antenna options, including U.FL connector, onboard chip, RF pad, and integrated PCB. The two standard types of XBee are XBee and XBee PRO. For this project, XBee was selected. [11.] Figure 18 demonstrates 20 pins of a XBee.

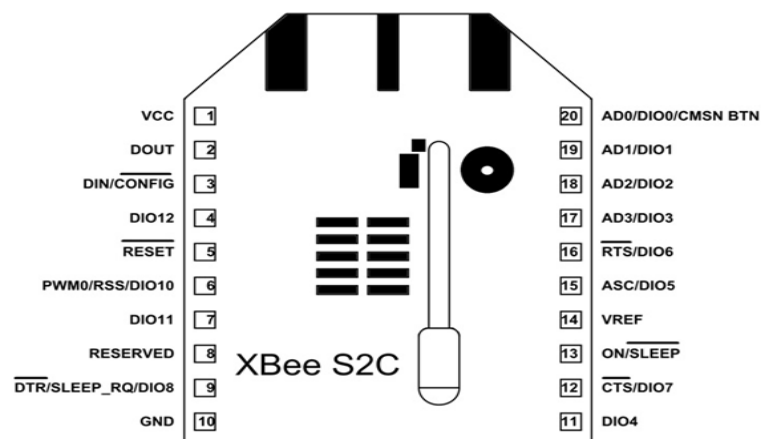


Figure 18. XBee S2C pins [11]

3.2 Rectifier

A rectifier is an electrical device made of one or more than one diode that transforms the alternating current (AC) into direct current (DC). AC current has a sine wave, including positive and negative sides. In case electronic devices receive AC current directly, AC current will destroy them. That is why a rectifier needed to convert AC to DC (the current is only on the positive side) to protect electronic devices. There are many types of rectifiers in the figure below.

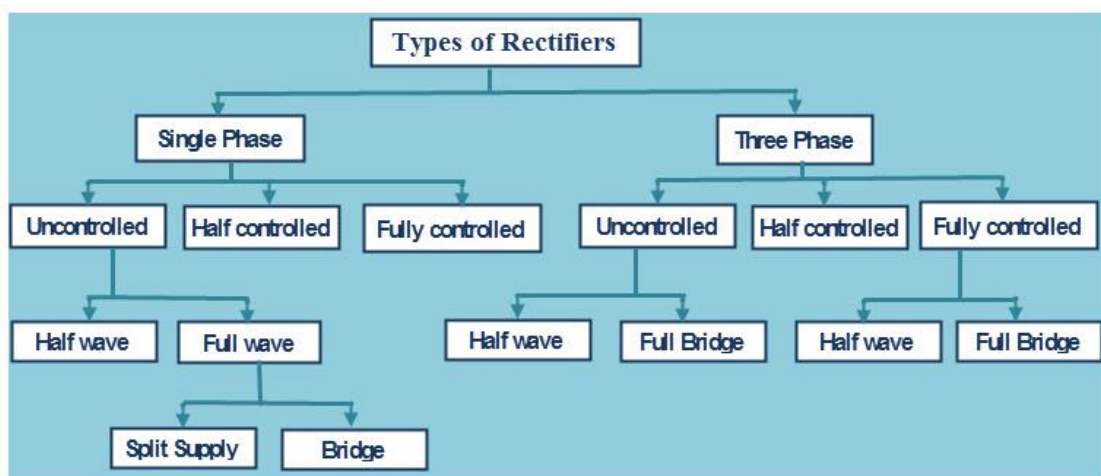


Figure 19. Types of rectifiers.[18]

In this project, the glass passive single-phase bridge rectifier was chosen, its name 2W04G. A full-wave bridge rectifier theory is used to make 2W04G. A closed-loop bridge connection with four single rectifier diodes is formed as a single-phase rectifier to produce the desired output wave. Bridge full wave rectifier is made of four rectifier diodes. It does not use any center tapping; therefore, it reduces its size and cost. As its name implies, the circuit consists of a bridge circuit. The connection of the four diodes in

the circuit is completed in the model of a closed-loop bridge. [19.] Figure 20 shows a full wave bridge rectifier.

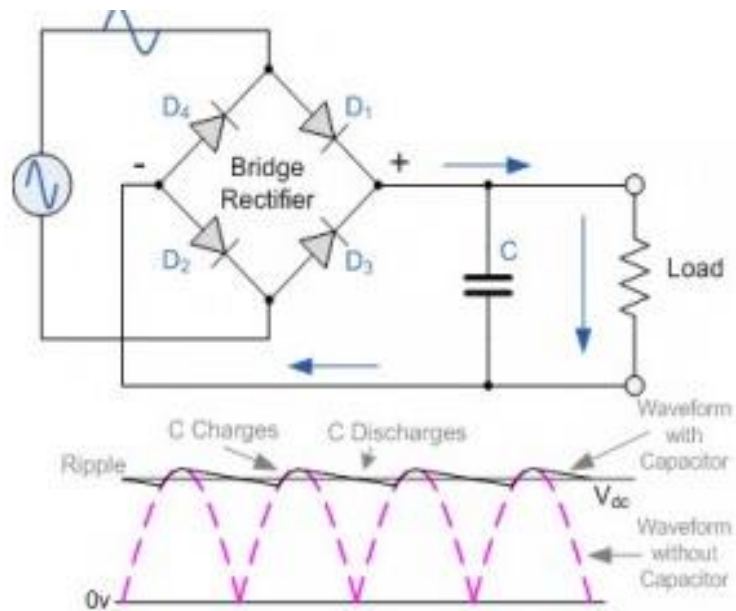


Figure 20. Full wave bridge rectifier [19]

Diodes D1, D2, D3 & D4 are used in this circuit, where two diodes will be conducted simultaneously instead of four, the upper half cycle or lower half cycle is formed by D1 & D3 or D2 & D4 fed to the circuit [19]. The figure 21 below is 2W04G rectifier.



Figure 21. 2W04G rectifier

There are some characteristics of full wave rectifier should be noticed:

- Ripple factor
- Form factor
- DC output current
- Peak inverse voltage
- Root mean square value of load current I_{RMS}
- Rectifier efficiency

3.3 Voltage Regulator

An integrated circuit that generates a fixed output voltage despite the transformations from the input voltage or load conditions is called a voltage regulator. The voltage regulators include two main genres: linear voltage regulators and switching voltage regulators; they are deployed in various applications. Linear voltage regulator is used the most in electronics field.

3.3.1 Linear Voltage Regulators

Series and shunt are two genres of linear voltage regulators. A voltage divider is a role of linear voltage regulator. It uses FET in the Ohmic region. The resistance of the voltage regulator varies with load resulting in constant output voltage. In terms of this regulator, the active pass element's variable conductivity as a MOSFET or a BJT is responsible for changing the output voltage. However, the efficiency of linear voltage regulators is very low [20]. The regulators chosen in this project are LM7805 and LM1117, they are both linear voltage regulator. LM7805 decreases the input voltage and generates a fixed output voltage (5V). On the other hand, LM1117 decreases the input voltage and generates a fixed output voltage (3.3V). A linear voltage regulator is demonstrated in figure 22.

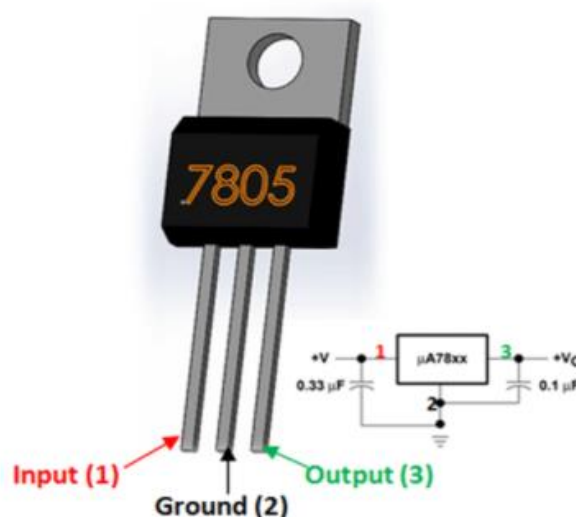


Figure 22. Linear voltage regulator [22]

3.4 GSM Module

A GSM module is similar to a GSM modem. However, there is a difference between them: a GSM modem is an external equipment. On the other hand, a GSM module can be integrated or soldered within the equipment. In this project, a GSM module with SIM800A was chosen. Figure 26 below demonstrates a GSM module with SIM800A.



Figure 26. GSM module with SIM800A

Features and specifications of SIM800A module are shown in figure 27 below:

- SIM800A Quad Band GSM Module
- Bands: GSM 850MHz, EGSM 900MHz, DCS 1800MHz, PCS 1900MHz
- Coding schemes: CS-1, CS-2, CS-3, CS-4 Tx power: Class 4 (2W), Class 1 (1W)
- GPRS class 2/10.
- Control via AT commands (3GPP TS 27.007, 27.005 and SIMCOM enhanced AT command set).
- Voltage Supply Required- 9VDC to 12VDC with atleast 2A Peak Current Capability
- High-Quality Product (Not hobby grade).
- 5V interface for direct communication with MCU kit.
- TTL Rx and TTL Tx and DB9 Connector Based RS232 Outputs
- Configurable baud rate.
- Built-in SIM Card holder.
- Built-in Network Status LED.
- Inbuilt Powerful TCP/IP protocol stack for internet data transfer over GPRS.
- Low power.
- Operating temperature: -40C to +85C
- External Finger type antenna
- Weight - 40gm

Figure 27. Specifications of SIM800A [15]

3.5 Arduino Uno R3

Arduino Uno is a microcontroller board compatible with ATmega328p chip developed by Arduino.cc. It is commonly applied in embedded systems and commands devices work according to the user's demands. Arduino Uno is favorable because it is a cheap microcontroller with features such as cross-platform, open-source software and hardware, and a simple programming environment with C or C++.

There are many Arduino types: Arduino Uno, Arduino Due, Arduino Mega, Arduino Leonardo, and so on. Depending on the user's purpose, types of Arduino will be chosen precisely. Arduino Uno is chosen in this project because it is useful and simple to set up with serial communication techniques. It operates as both a data reader from sensors and a controller to display information on LCD and give commands to other components such as a buzzer.

Fourteen digital input/output pins as well as PWM outputs are architected in Arduino Uno. [24.] Figure 28 shows Arduino Uno R3's pins.

Arduino Uno R3 Pinout

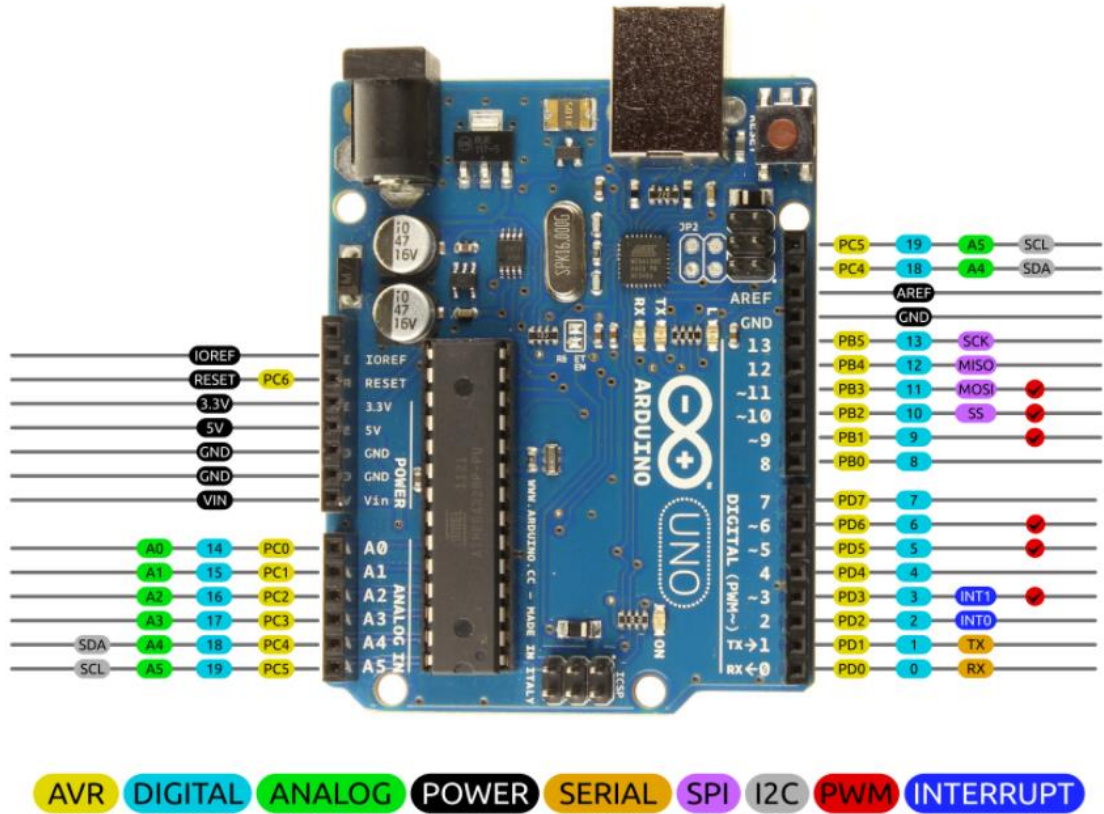


Figure 28. Arduino Uno Pinout. [25]

Technical specification of Arduino Uno is demonstrated in figure 29 below.

Microcontroller	ATmega328p 8-bit AVR family microcontroller
Working voltage	5V
Voltage input (recommended)	7V to 12V (limited to 20V)
Digital pins	14 pins (D0 to D13), out of which 6 pins provide PWM
Analogue pin	6 pins (A0 to A5)
DC current on I/O pins	40mA
DC current on 3.3V pins	50mA
Flash memory	32kB of with 0.5kB used for boot loader
Clock speed	16MHz
Dimension (length x width)	68.6 mm x 53.4 mm

Figure 29. Technical specification of Arduino Uno [25]

The environment in Arduino device is integrated development environment, which is non-commercial to use with some basic skills. IDE is compatible with some operating systems such as Windows, Linux, and Mac. The programmers can use C and C++ in Arduino Uno.

3.6 16x2 LCD Display

LCD Display is a small monitor to display the information in text or characters form, it integrates an LED backlight. With its function, 16 characters is demonstrated on a row and total 32 ASCII characters in two rows. [10.] Figure 30 demonstrates a LCD display.



Figure 30. LCD Display [10]

3.7 MQ2 Gas Sensor

MQ2 gas sensor is a standard gas sensor in the MQ series. It is very cheap and easy to use in an embedded system such as a gas and fire detection system. The figure below demonstrates the structure of MQ2 gas sensor.

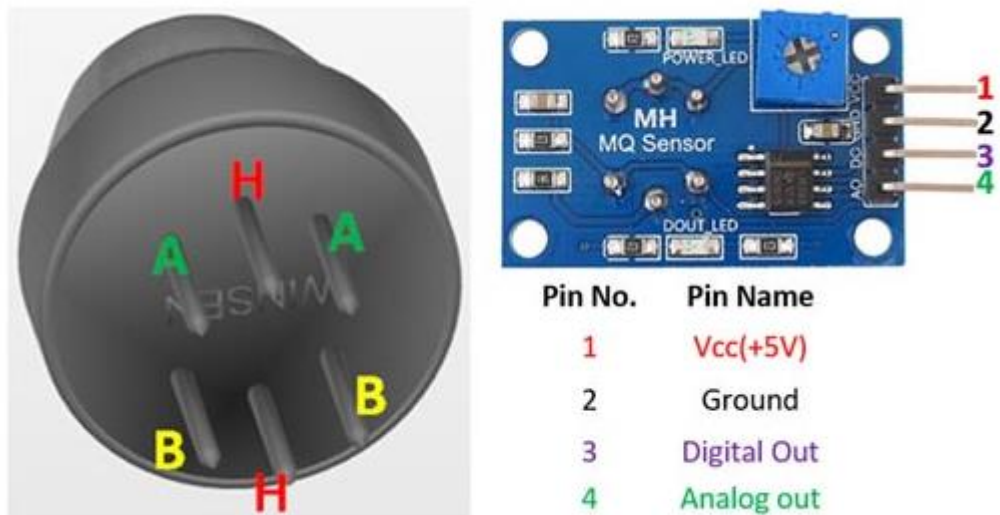


Figure 31. MQ2 Gas Sensor pinout. [27]

Explanation for pin configuration of MQ2 Gas Sensor can be found in the table below.

Table 1. Pin description of MQ2 [27]

Pin No	Pin Name	Description
For Module		
1	Vcc	This pin receives the input voltage to powers the module, typically the operating voltage is +5V
2	Ground	Used to connect the module to the ground
3	Digital Output	This pin is used to get digital output, by setting a threshold value using the potentiometer
4	Analog Output	This pin results to 0-5V analog voltage output based on the intensity of the gas
For Sensor		
1	H-Pins	Out of the two H pins, one pin is connected to supply and the other to ground
2	A-Pins	The A pins and B pins are interchangeable. These pins will be tied to the Supply voltage.
3	B-Pins	The A pins and B pins are interchangeable. One pin will act as output while the other will be pulled to ground.

4 Removing Arduino Uno Board from The System and Programming Atmega328 Using Arduino IDE

The Bootloader is a small piece of code (executable code in hex format) that resides in the microcontroller's memory. It could be understood as a trick to remove an Arduino Uno board allowing an ATmega328P microcontroller chip to stand alone or to be integrated on a PCB. Due to this trick, Arduino accepts the code from the computer and places it in the microcontroller's memory. [28.]

There are five steps to burn the Bootloader into Atmega328 IC using Arduino Uno board.

Step 1: Opening Arduino IDE. Clicking File → Example → ArduinoISP. Then choose ArduinoISP as figure 32 shown below.

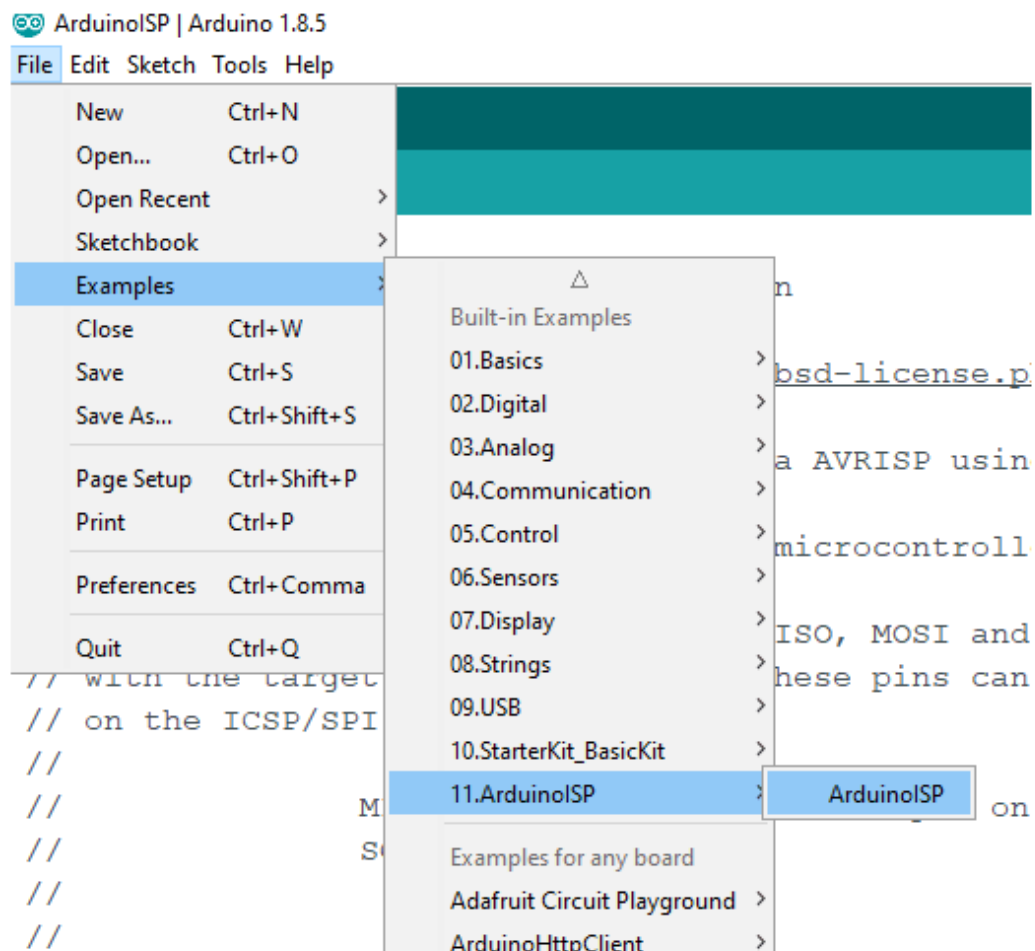
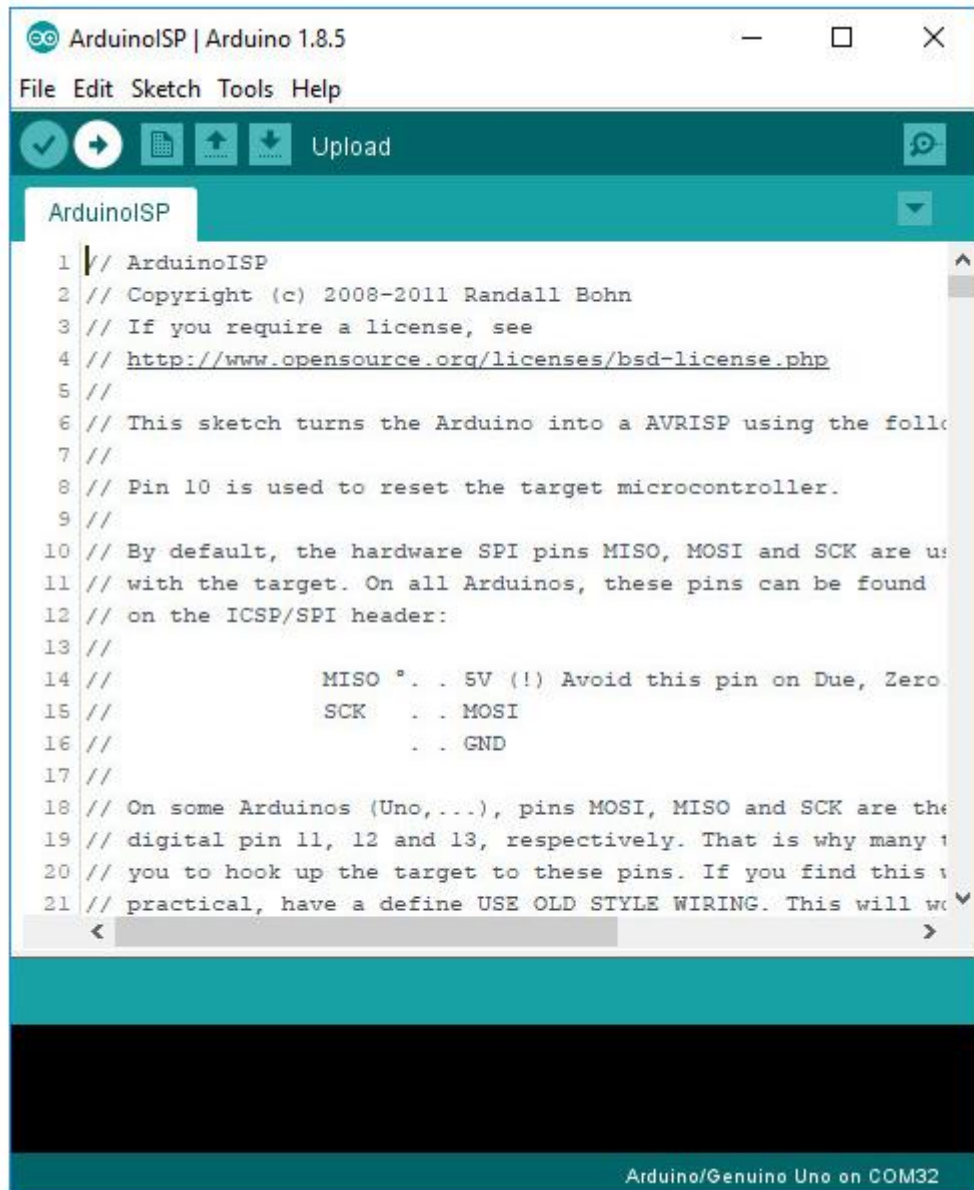


Figure 32. Arduino ISP [29]

Step 2: Uploading this code to the Arduino board. Choosing the com port and board from the tool menu and press the upload button as figure 33 demonstrated below.



```

1 // ArduinoISP
2 // Copyright (c) 2008-2011 Randall Bohn
3 // If you require a license, see
4 // http://www.opensource.org/licenses/bsd-license.php
5 //
6 // This sketch turns the Arduino into a AVRISP using the follow
7 //
8 // Pin 10 is used to reset the target microcontroller.
9 //
10 // By default, the hardware SPI pins MISO, MOSI and SCK are us
11 // with the target. On all Arduinos, these pins can be found
12 // on the ICSP/SPI header:
13 //
14 //             MISO °. . 5V (!) Avoid this pin on Due, Zero
15 //             SCK   . . MOSI
16 //             . . . . GND
17 //
18 // On some Arduinos (Uno, ...), pins MOSI, MISO and SCK are the
19 // digital pin 11, 12 and 13, respectively. That is why many t
20 // you to hook up the target to these pins. If you find this v
21 // practical, have a define USE_OLD_STYLE_WIRING. This will wo

```

Arduino/Genuino Uno on COM32

Figure 33. Code for Bootloader [29]

Step 3: After done uploading, disconnecting the Arduino board from the computer and make the connections of Arduino board with Atmega328 as demonstrated in below diagram.

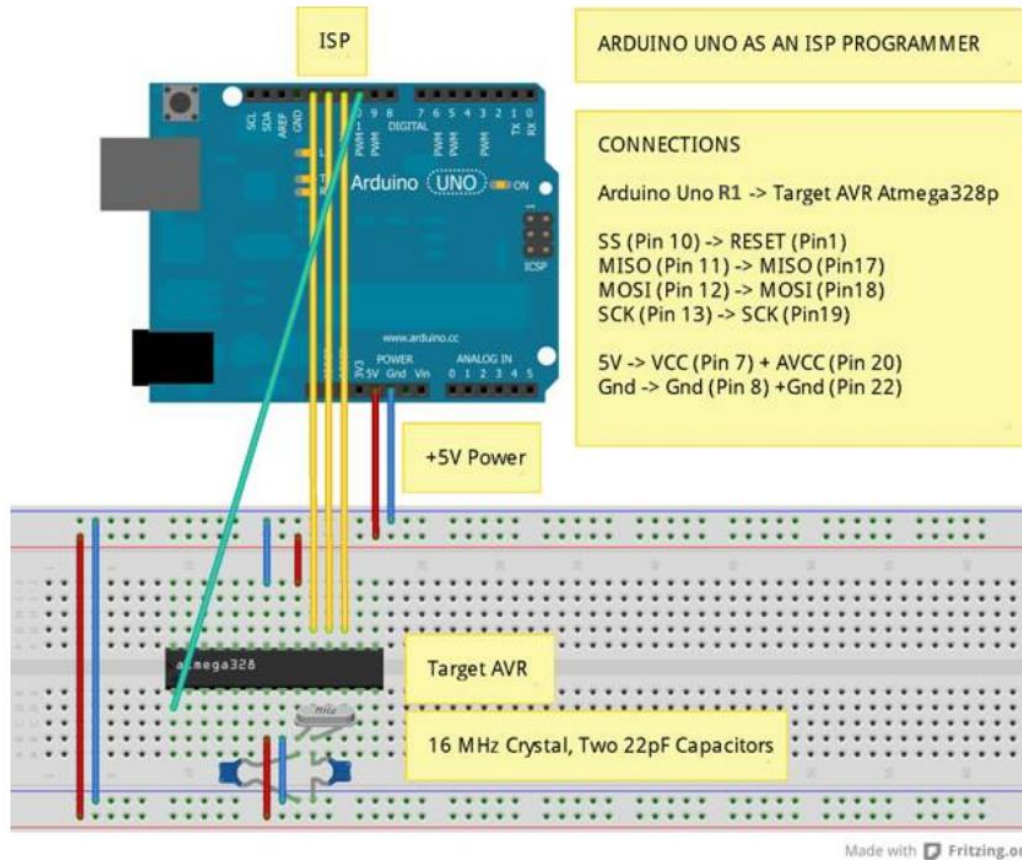


Figure 34. Circuit diagram for Bootloader [30]

Step 4: Connecting the Arduino board with the computer and then opening Arduino IDE. Clicking **Tools**, choosing **Board** as **Arduino/Genuine Uno**, then choosing the correct **Port** where the Arduino board is connected to the computer. Choosing **Programmer** as **“Arduino as ISP”** as figure 35 shown below.

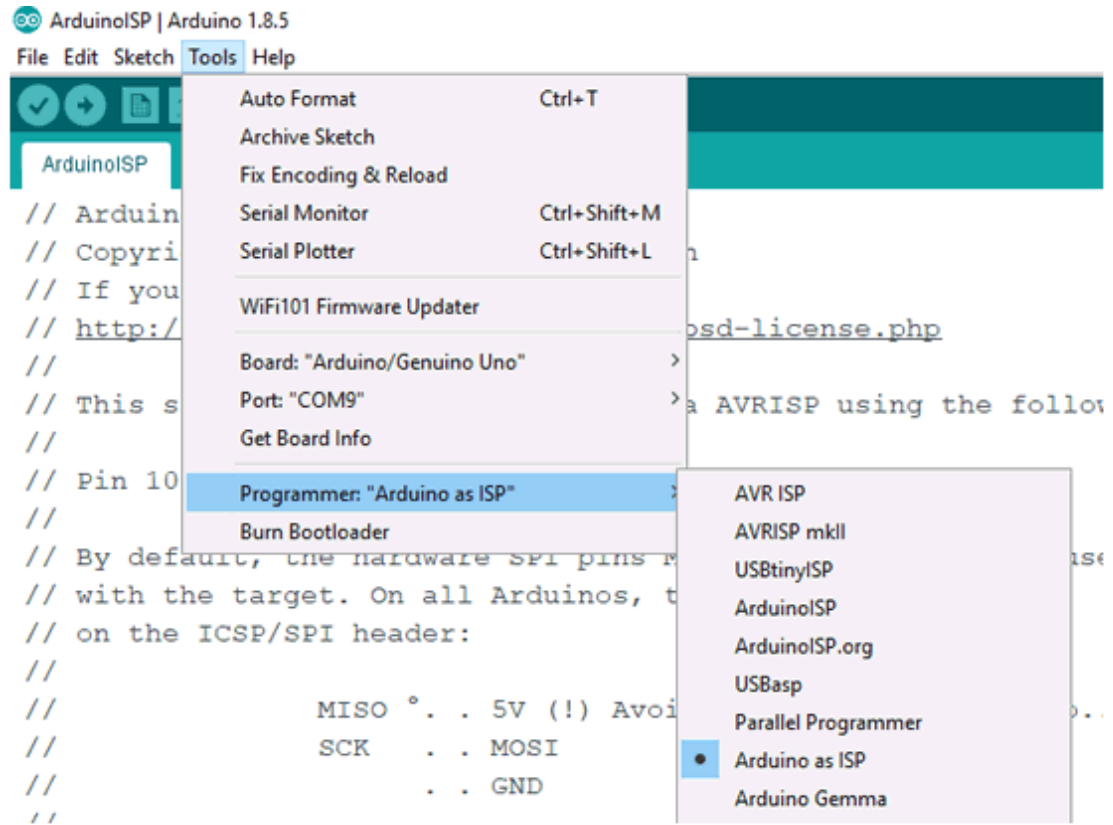
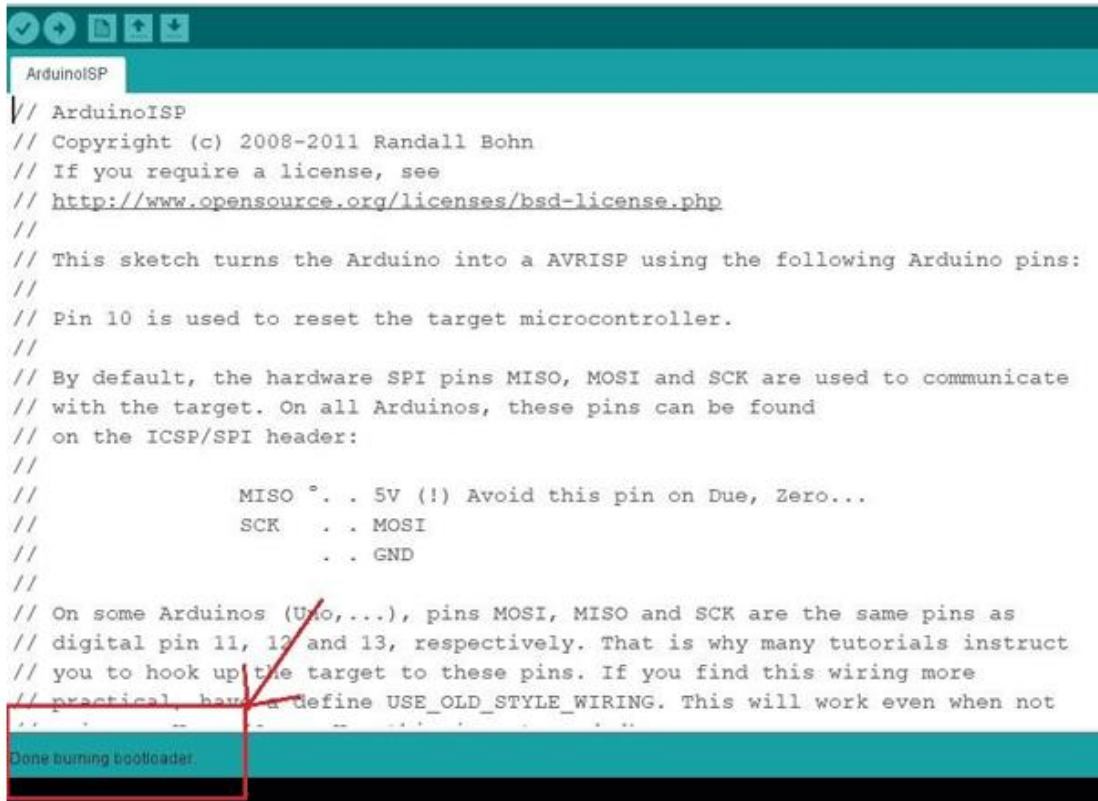


Figure 35. Configurations for Bootloader [29]

Step 5: Clicking **Tools** again and choosing **Burn Bootloader**. Few seconds later, bootloader is uploaded successfully as figure 36 illustrated below.



```

ArduinoISP
// ArduinoISP
// Copyright (c) 2008-2011 Randall Bohn
// If you require a license, see
// http://www.opensource.org/licenses/bsd-license.php
//
// This sketch turns the Arduino into a AVRISP using the following Arduino pins:
//
// Pin 10 is used to reset the target microcontroller.
//
// By default, the hardware SPI pins MISO, MOSI and SCK are used to communicate
// with the target. On all Arduinos, these pins can be found
// on the ICSP/SPI header:
//
//
//           MISO  . . 5V (!) Avoid this pin on Due, Zero...
//           SCK   . . MOSI
//           .     . . GND
//
// On some Arduinos (Uno,...), pins MOSI, MISO and SCK are the same pins as
// digital pin 11, 12 and 13, respectively. That is why many tutorials instruct
// you to hook up the target to these pins. If you find this wiring more
// practical, have a define USE_OLD_STYLE_WIRING. This will work even when not

```

Done burning bootloader.

Figure 36. Successful uploading for Bootloader [29]

The next move is to program ATmega328 microcontroller chip using Arduino board.

Step 1: Using an Arduino board without ATmega328. Creating the connections of Arduino board with a breadboard as illustrated below.

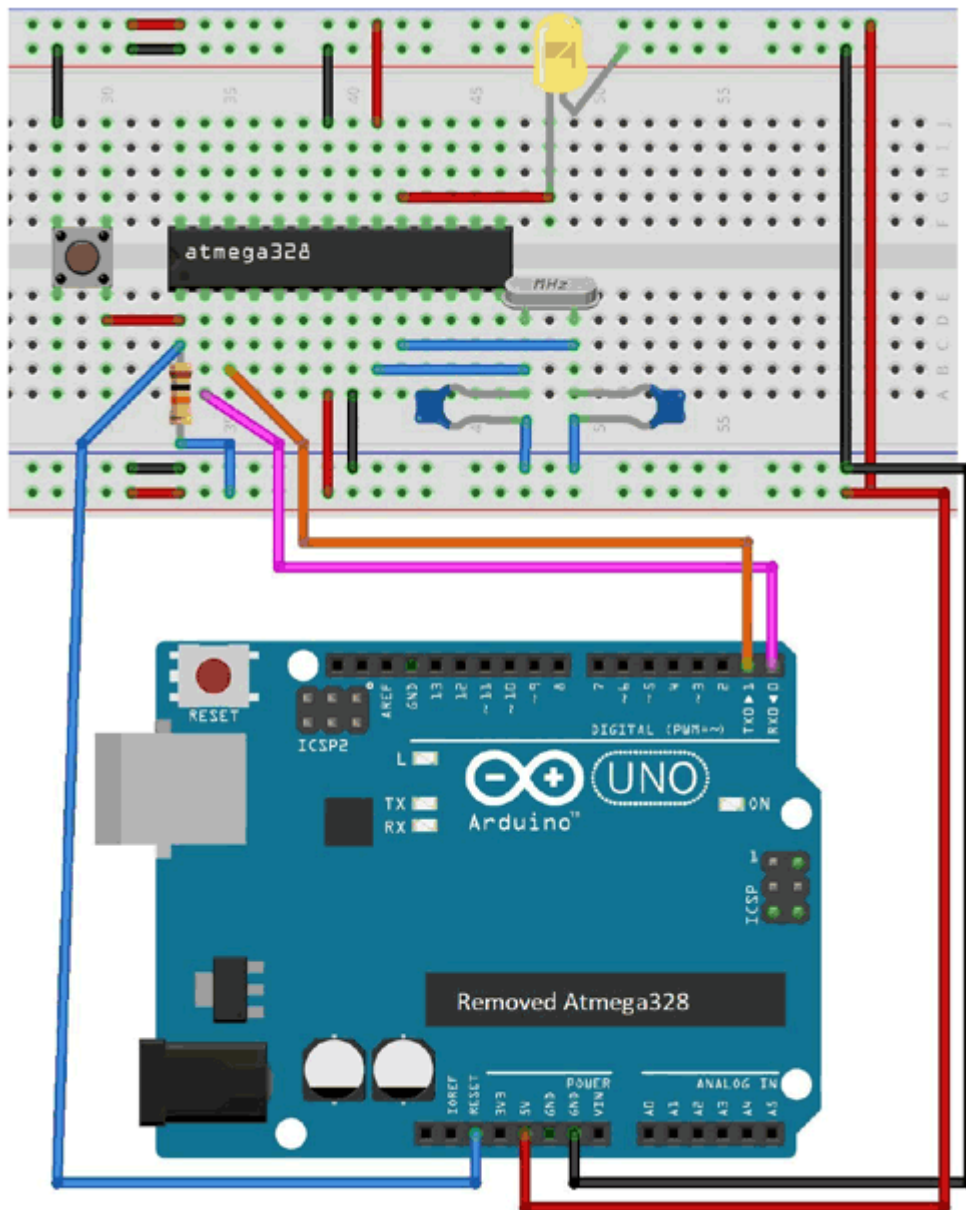


Figure 37. Circuit diagram for chip programming [29]

Step 2: Connecting the Arduino board to the computer via a USB port and opening Arduino IDE. Clicking Tools → Board → Arduino Uno, then Programmer → USBasp and correcting com port of the board.

Step 3: Inserting the code needed to be uploaded into ATmega328 in Arduino IDE and pressing upload button.

To see full code for gas and fire detection system, check:

- [Appendix 1](#) which contains code lines for receiver part.
- [Appendix 2](#) which contains code lines for transmitter part.

5 What is Gas and Fire Detection System

The figure 38 below demonstrates the whole Gas and Fire Detection System.

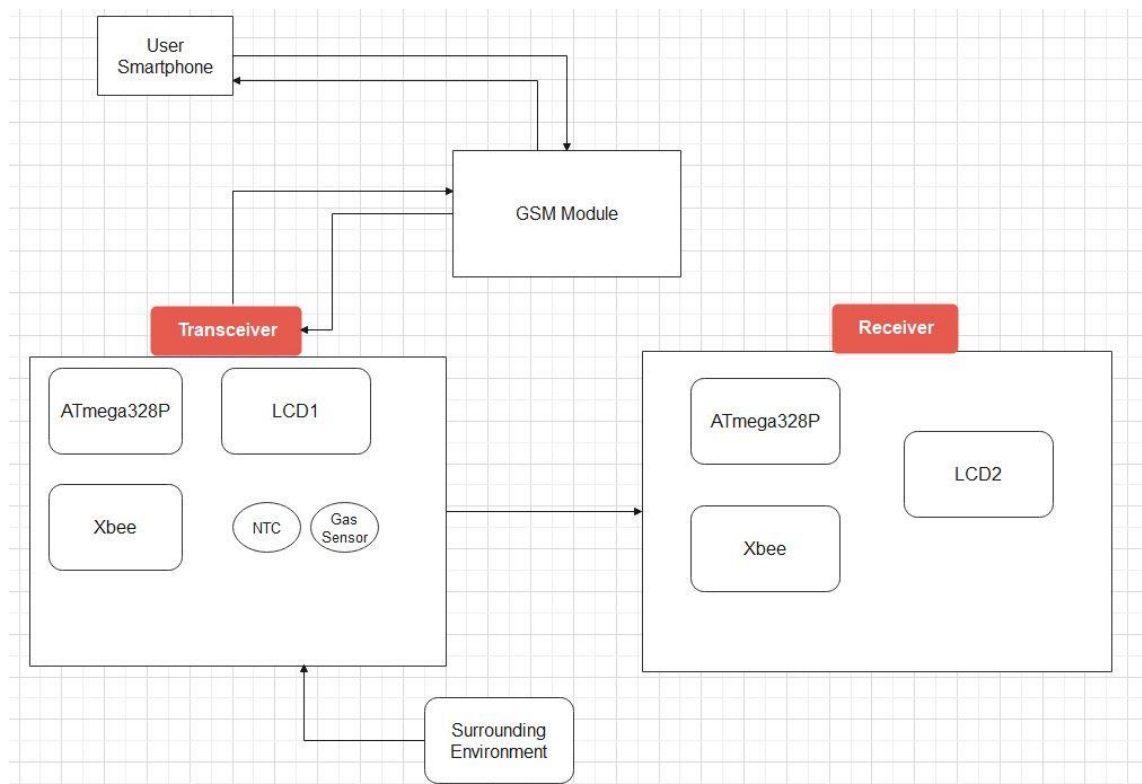


Figure 38. Graph describes Gas and Fire Detection System

Gas and Fire Detection System includes three parts: transmitter, receiver, and GSM module. GSM module is connected to the transceiver and it plays an essential role as an intermediate between GFDS and the user smartphone. The transceiver consists of few main components such as ATmega328P, Xbee, LCD1, NTC, and Gas Sensor. The ATmega328P chip plays an important role as a brain of the transceiver, the operating code is loaded to ATmega328P chip to decide what it has to do. While Xbee is a connector between the transceiver and the receiver, it is going to send the signal from the transceiver to the receiver. LCD1 is a LCD Display showing the information what the transceiver is holding and understanding. NTC and Gas Sensor is two types of sensor

to measure the surrounding temperature. The receiver consists of few main components such as ATmega328P, Xbee, and LCD1. The ATmega328P chip still plays an essential role as a brain of the receiver to operate the receiver. While Xbee is used to connect the receiver to the transceiver. LCD2 is a LCD Display showing the information passed from the transceiver. The principle of this system is that the user needs to activate this system by calling the number of the simcard on GSM Module. If activation is successful, the information about the system and surrounding environment are displayed on the LCD Display. In case the temperature of the surrounding environment exceeds 45°C or gas leakage detection, it will indicate a warning message sent from GSM Module to the user smartphone.

The figure 39 below illustrates the Gas and Fire Detection System with PCB.



Figure 39. Gas and Fire Detection System with PCB

GSM module is the blue PCB connected to the transmitter and it plays an essential role as a connector between GFDS and the user smartphone. The transmitter is the biggest

A 12V DC will go through 2W04G rectifier and is converted to 5V DC by 7805 Voltage Regulator. 7805 Voltage Regulator is needed because the supply voltage for ATmega328 should be from 5V to 9V. After that, a LM1117 is used to convert 5V DC to 3.3V DC because the supply voltage for Xbee is 3.3V.

To make LCD 16*2 operate, VSS is connected to ground, VDD is connected to 5V supply, VEE is connected to potentiometer, pin 5 is connected to ground, and pins 4,6,11,12,13,14 of LCD are connected to pins 17,18,23,24,25,26 of ATmega328 respectively.

About ATmega328, pins 13,14 are connected to pins 3,2 of Xbee respectively, pins 9,10 are connected to Crystal Oscillator 16MHz, pin 8 is connected to ground, and pin 1 is connected to reset button.

After completing the schematic of receiver, a PCB layout is created. The figure 40 below shows a PCB layout of receiver part.

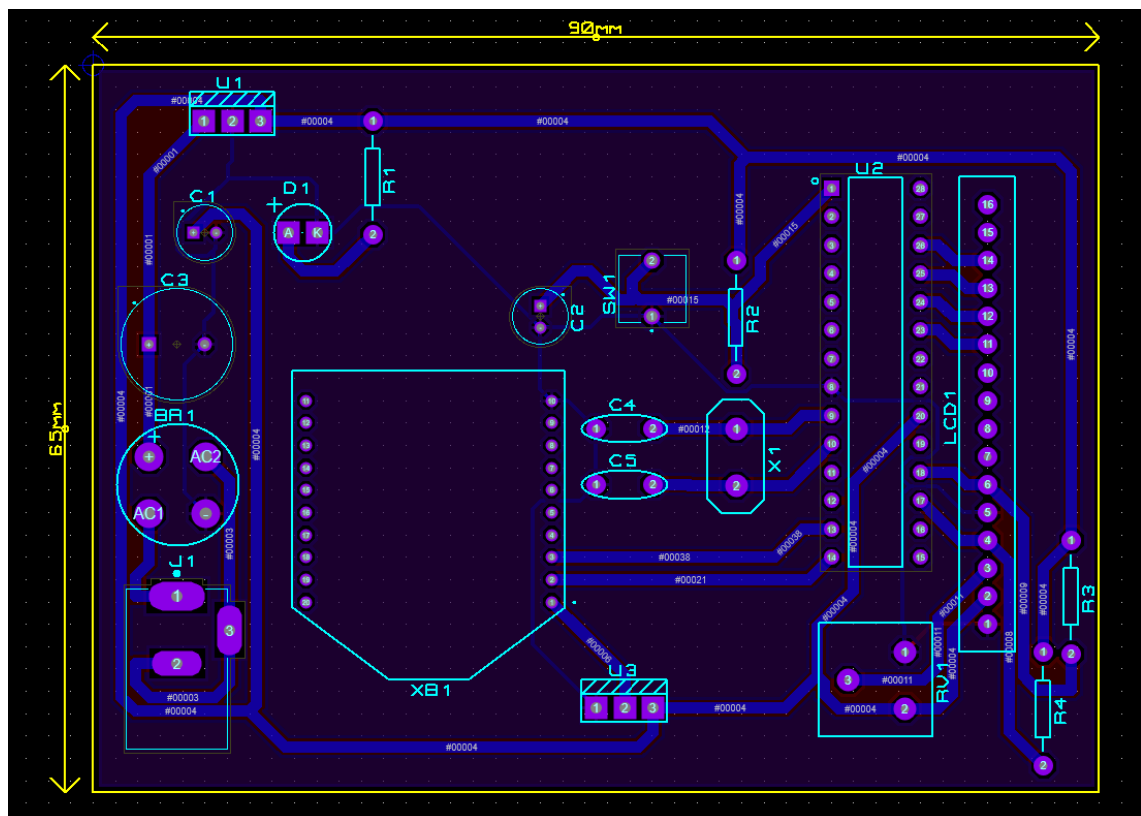
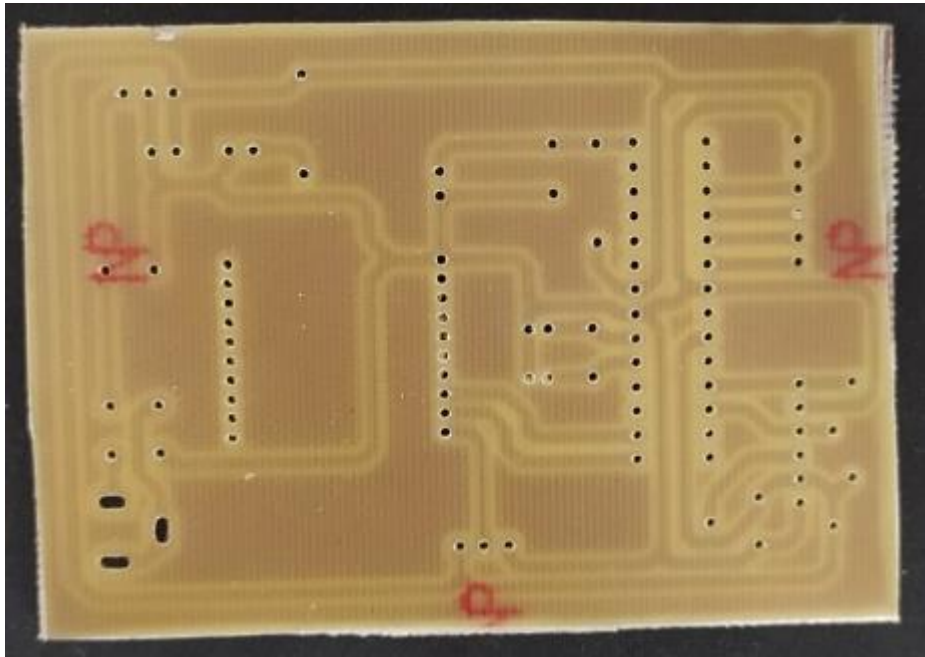


Figure 41. PCB layout for receiver part

The dimension of the PCB board is 90mm width and 65mm height. Components will be placed on top layer and routed on bottom layer; therefore, the bottom layer is solder side. A ground plane is created on both top layer and bottom layer to reduce noise and



interference. When completing PCB layout, a milling machine or hand-made method can be used to make a real PCB. The figure 41 below demonstrate a PCB for receiver part.

The next step is placing components on top side of PCB and solder the bottom side to connect components together. The completed PCB is shown in figure 42 below.



Figure 42. Top side of PCB for receiver part



Figure 43. Top side of completed PCB for receiver part

6.2 Transmitter Part

The figure 43 below is a schematic of transmitter in GFDS is drawn by using Proteus Professional 8.11.

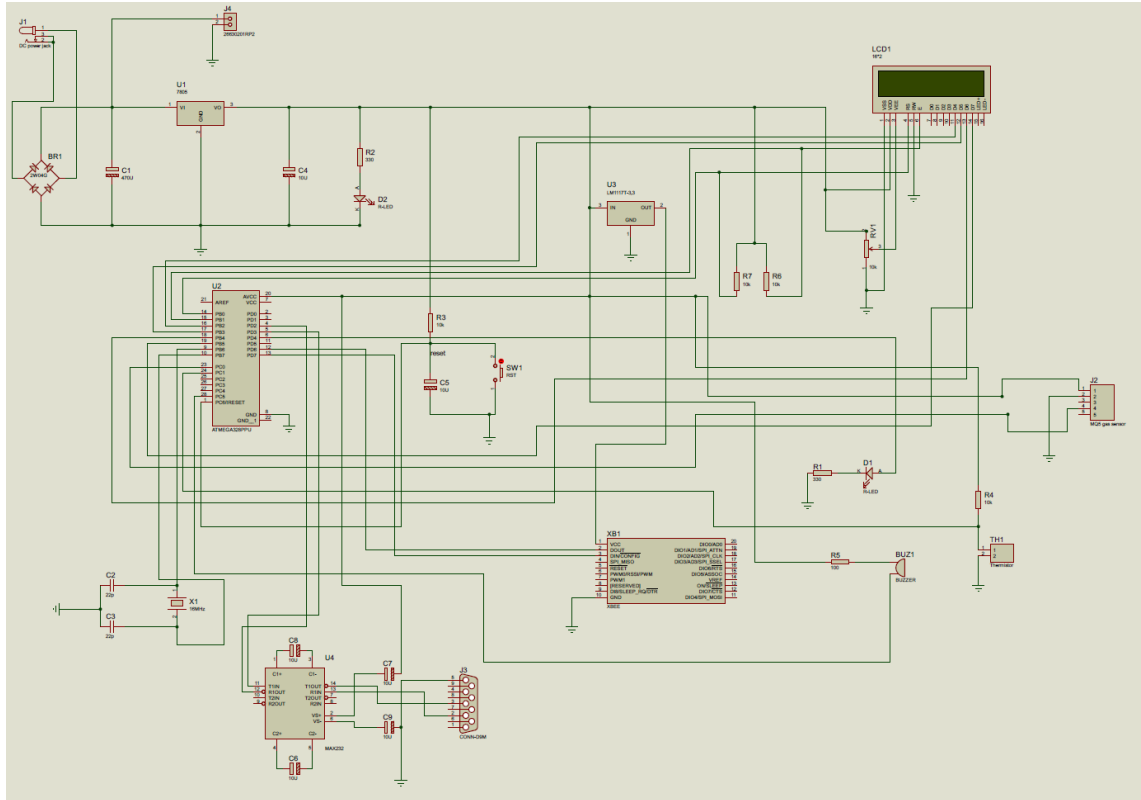


Figure 44. Schematic for transmitter

The initial setup of transmitter part is similar to receiver part with a DC power jack, 2W04G rectifier, 7805 Voltage Regulator, LM1117 Voltage Regulator, LCD 16*2, ATmega328, and Xbee. There are some components added: a reliment, a MQ-5 gas sensor, a 10K thermistor, a buzzer, a connector, and a MAX232 (dual transmitter / dual receiver) to convert RX, TX, CTS, RTS signal.

The connections to ATmega328's pins are adjusted. Pins 4,5 of ATmega328 are connected to pins 1,2 of MAX232, pin 6 is connected to LED D1, pins 12,13 are connected to pins 2,3 of Xbee, pins 14,15,16,17,18,19 are connected to pins 4,6,11,12,13,14 of LCD 16*2, pin 23 is connected to pin 4 of gas sensor, pin 24 is connected to one leg of 10K thermistor, pins 9,10 are connected to Crystal Oscillator,

The dimension of the PCB board is 111mm width and 111mm height. Components will be placed on top layer and routed on bottom layer; therefore, the bottom layer is solder side. A ground plane is created on both top layer and bottom layer to reduce noise and interference. When completing PCB layout, a milling machine or hand-made method can be used to make a real PCB. The figure 45 below demonstrate a PCB for receiver part.

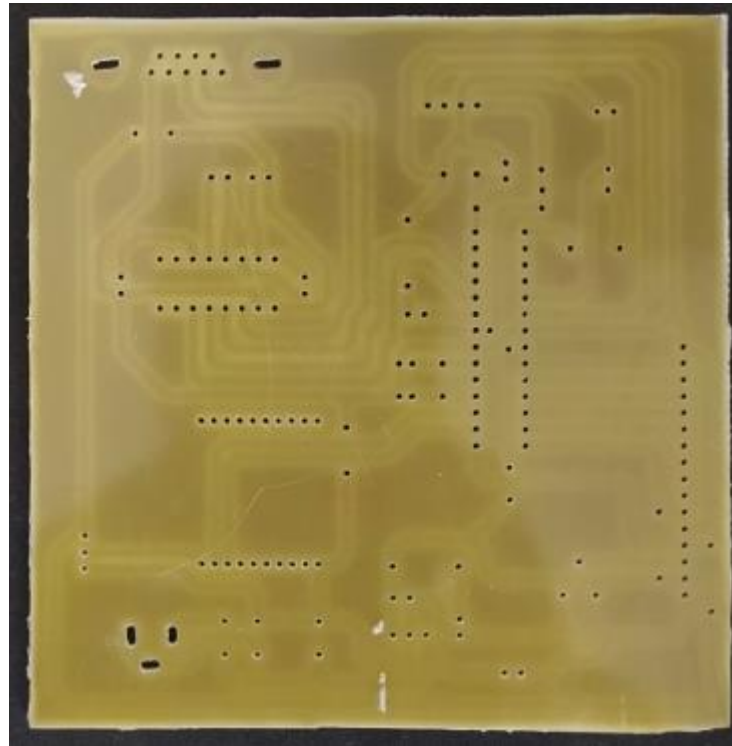


Figure 46. Top side of PCB for transmitter part

The next step is placing components on top side of PCB and solder the bottom side to connect components together. The completed PCB is shown in figure 46 below.

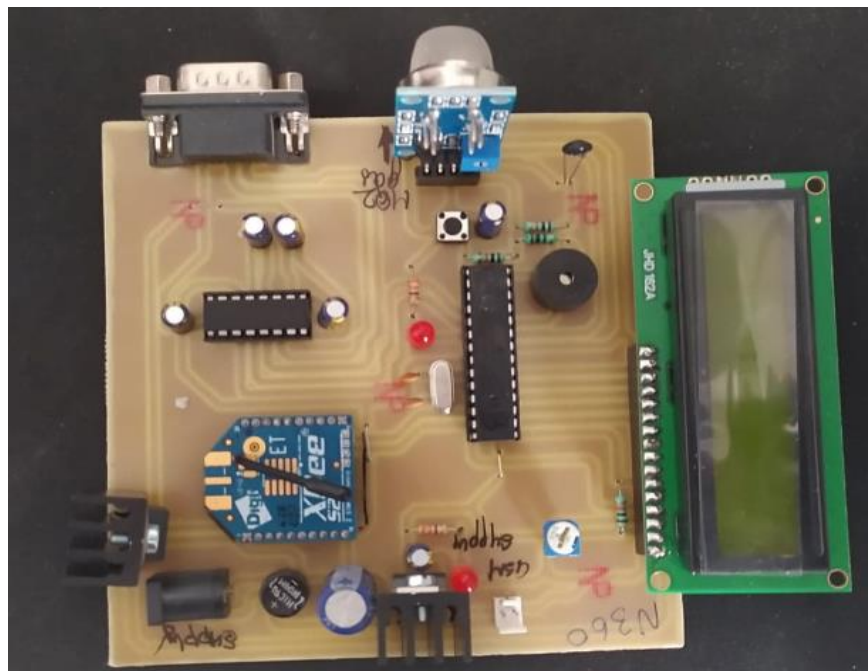


Figure 47. Top side of completed PCB for transmitter part

7 How GFDS detects fire and gas leakage

Firstly, the user needs to call the phone number of the sim card on GSM module to configure the whole system. When the configuration is done, a message is sent to the user smartphone as shown in figure 47 below.

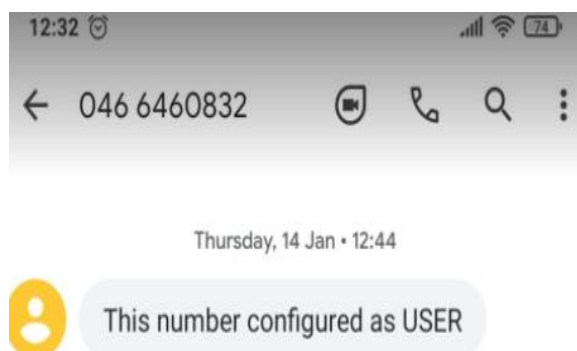


Figure 48. Configuration

After that, the LCD 16*2 on transmitter will show the surrounding temperature via 10K thermistor as figure 48 below.

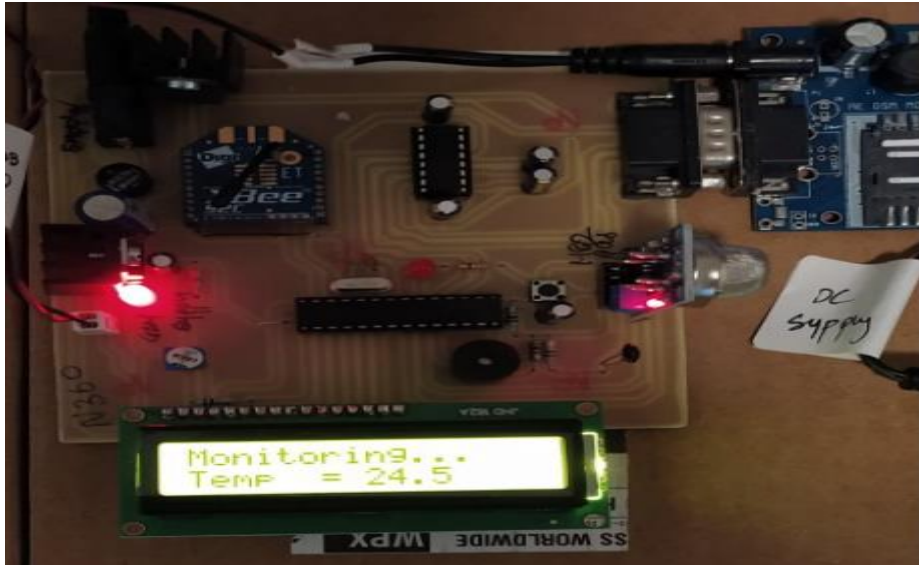


Figure 49. Surrounding temperature

To test fire detection, a lighter is opened close to 10K thermistor. When the temperature on LCD 16*2 is above 42°C, indicating a buzzer on transmitter alerts and a warning message will be displayed on the second LCD 16*2 as figure 49 below.

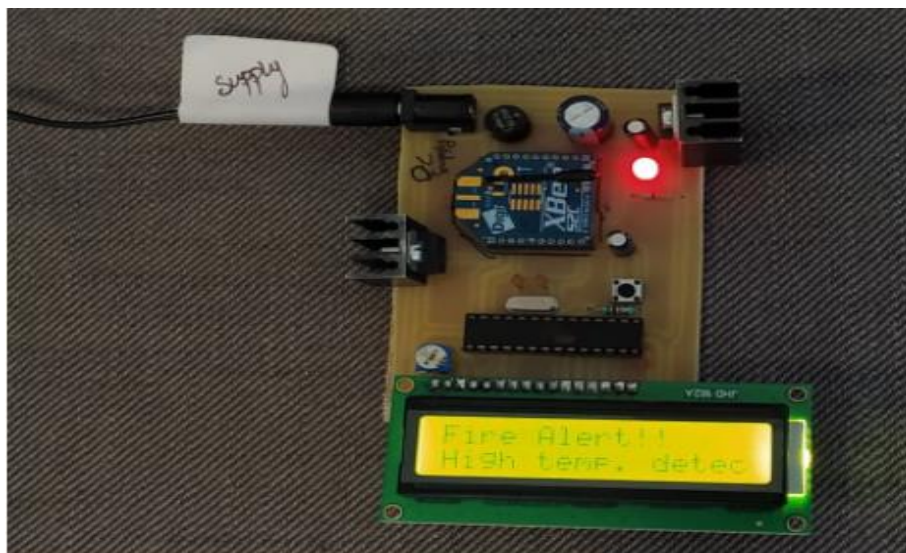


Figure 50. Warning message for fire on LCD

And a message is sent to the user smartphone simultaneously as figure 50 below.

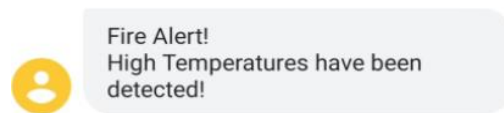


Figure 51. Warning message for fire on smartphone

To test gas detection, a gas lighter is opened close to MQ5 gas sensor in 5 to 7 seconds. This action will indicate a buzzer on transmitter alerts and a warning message will be displayed on the second LCD 16*2 as figure 51 below.

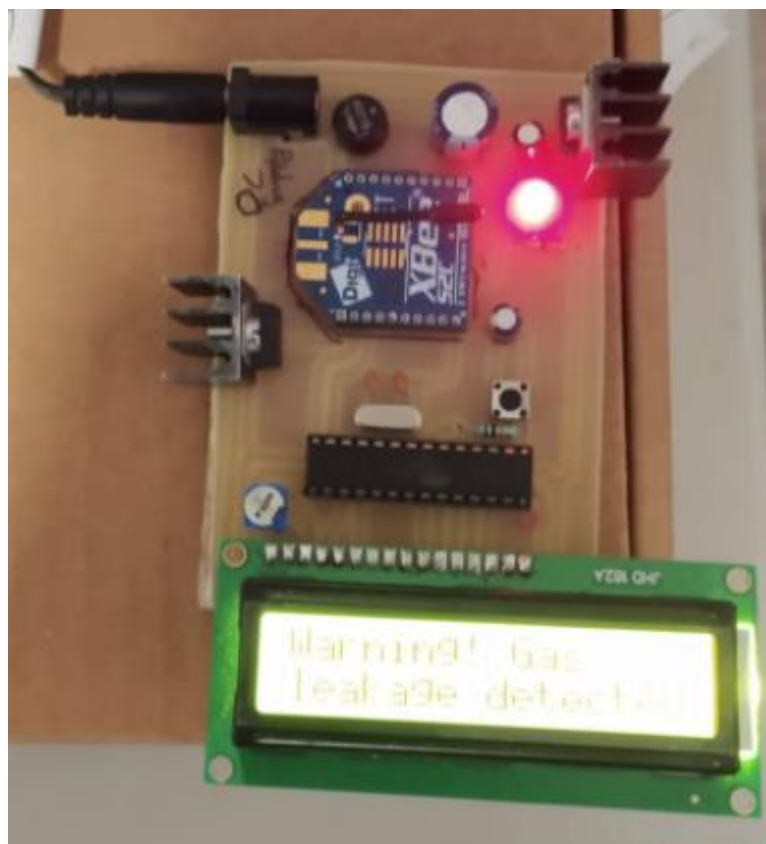


Figure 52. Warning message for gas leakage on LCD

And a message is sent to the user smartphone simultaneously as figure 52 below.

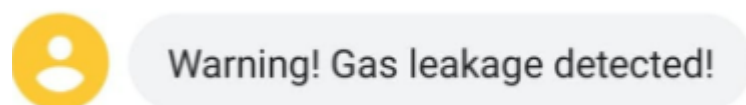


Figure 53. Warning message for gas on smartphone

8 Advantages and Disadvantages

This project was completed at a reasonable price. The amount of expense for hardware was demonstrated. These components are cheap and expected to buy from the internet market. Table 2 demonstrate the expense for hardware.

Table 2. Expense for hardware

TX-Category	Quantity	References	Value	Price (Euro)
Modules	1	M1	SIM5320E	70
Capacitors	1	C1 (polarized capacitor)	470uf	2
	2	C2-C3	22pF	
	2	C4-C5 (polarized capacitor)	10uF	
Resistors	1	R1	330	2
	1	R2 (ceramic resistor)	330R	
	4	R3-R4,R6-R7	10k	
	1	R5	100	
Integrated Circuits	1	U1 (voltage regulator-linear)	LM1117	0.37
	1	U2	7805	0.5
	1	U3	ATMEGA328P	1.76
Diodes	2	D1-D2	R-LED	1
Miscellaneous	1	BR1	2W04G	0.782
	1	BUZ1	BUZZER	2
	1	J1	DC POWER JACK	3.3
	1	J2(FEMALE)	MQ5 gas sensor	1
	1	LCD1	16*2	1
	1	RV1	10k Preset	1.5
	1	SW1	RST	0.45
	1	TH1	Thermister	1.6
	1	X1	16Mhz	3.28
	1	XB1	XBEE	19.42
RX-Category	Quantity	References	Value	Price
Capacitors	2	C1-C2 (polarized capacitor)	10uF	3
	1	C3 (polarized capacitor)	470uf	
	2	C4-C5	22pf	
Resistors	1	R1	330	0.5
	3	R2-R4	10k	
	1	U1	7805	0.5

Integrated Circuits	1	U2	ATMEGA328P	1.76
	1	U3	LM1117	0.37
Diodes	1	D1	R-LED	1
Miscellaneous	1	BR1	2W04G	0.782
	1	J1	DC POWER JACK	3.3
	1	LCD1	16*2	1
	1	RV1	10k	1.5
	1	SW1	RST	0.45
	1	X1	16MHz	3.28
	1	XB1	XBEE	19.42
			Arduino Uno	18.86
			Total	167.184

The most expensive components are Xbee (19.42 EUR) and Arduino Uno (18.86 EUR); however, they could be cheaper from the other retailers on Amazon.

Despite having this advantage, GFDS has a problem with MQ5 gas sensor and GSM module. MQ5 gas sensor is unable to measure smoke concentration, and it can just realize gas leakage and give a warning message. The operating range of GSM module is not too far. If GSM module is placed in a closed area and the user is outside, the GSM module cannot reach the user smartphone.

9 Conclusion

The main goal of this project was to create a device that can alert the user about gas leakage and fire. The project was divided into two parts, transmitter and receiver. In order to complete the transmitter and the receiver, an Arduino Uno, ATmega328 chip, gas sensor, temperature sensor, and other components are used. The most challenging process is creating a PCB for the transmitter and receiver and soldering the components on PCB; however, it can be completed by applying theoretical knowledge and software. The PCB had been operating without internal error. The GFDS operates as expected. Gas and temperature sensor are able to realize the danger to warn the user.

Generally, the transmitter and the receiver of this project successfully operated as intended. The captured temperature and gas leakage by GFDS have low error and high precision. Transferring data between the transmitter and the receiver by wire-less communication worked appropriately. To conclude, everything had been operating as expected.

Although GFDS operates smoothly, there are some ideas that can be used to improve this system:

- Replacing MQ5 gas sensor with a more modern sensor to measure the smoke concentration.
- Connecting GFDS to a water system to release water in case fire detected.
- Replacing a newer GSM module to improve reaching range between GSM module and the user smartphone.

References

- 1 Techopedia. Definition-Microcontroller. [Online] [Cited on January 18th 2021]. <https://www.techopedia.com/definition/3641/microcontroller>
- 2 Judith Cardell. Overview of Microprocessor. [Online] [Cited on January 18th 2021]. <http://www.science.smith.edu/~jcardell/Courses/EGR328/Readings/uProc%20Overview.pdf>
- 3 RayPCB. The schematic diagram of automotive sensor signal conditioning. [Online] [Cited on January 18th 2021]. <https://www.raypcb.com/the-schematic-diagram-of-automotive-sensor-signal-conditioning/>
- 4 Ravi Teja. What is a sensor? Different type of sensors, application. [Online] [Cited on January 18th 2021]. <https://www.electronicshub.org/different-types-sensors/>
- 5 Fierceelectronics. What is a temperature sensor. [Online] [Cited on January 18th 2021]. <https://www.fierceelectronics.com/sensors/what-a-temperature-sensor>
- 6 Omega. What is a Thermistor and how does it work. [Online] [Cited on January 18th 2021]. <https://www.omega.com/en-us/resources/thermistor>
- 7 Electronics-Tutorials. Thermistor and NTC thermistor. [Online] [Cited on January 18th 2021]. <https://www.electronics-tutorials.ws/io/thermistors.html>
- 8 Lastminuteengineers. How MQ2 Gas/Smoke Sensor Works? & interface it with Arduino. [Online] [Cited on January 20th 2021]. <https://lastminuteengineers.com/mq2-gas-senser-arduino-tutorial/>
- 9 Elprocus. MQ2 Gas Sensor Working and Its Applications. [Online] [Cited on January 20th 2021]. <https://www.elprocus.com/an-introduction-to-mq2-gas-sensor/>
- 10 Lastminuteengineers. Interfacing 16x2 Character LCD Module with Arduino. [Online] [Cited on January 20th 2021]. <https://lastminuteengineers.com/arduino-1602-character-lcd-tutorial/>
- 11 Arduino-Tutorials. What is XBee? [Online] [Cited on January 20th 2021]. <https://www.arduino-tutorials.com/what-is-xbee/>
- 12 ElectricalFundaBlog. Parallel Communication. [Online] [Cited on February 15th 2021]. <https://electricalfundablog.com/parallel-communication-characteristics/>
- 13 Phil Croucher. Parallel Communication Explained. [Online] [Cited on February 15th 2021]. <http://hardwarehell.com/articles/parallel.htm>

- 14 JIMBLOM. Serial Communication. [Online] [Cited on February 15th 2021]. <https://learn.sparkfun.com/tutorials/serial-communication/>
- 15 4G 5G World. Global System for Mobile communication (GSM). [Online] [Cited on February 16th 2021]. <http://4g5gworld.com/wiki/global-system-mobile-communications-gsm#:~:text=There%20are%20five%20different%20cell,according%20to%20the%20implementation%20environment.&text=Umbrella%20cells%20are%20used%20to,in%20coverage%20between%20those%20cells.>
- 16 Erika Granath. What's a printed circuit board (PCB)? [Online] [Cited on February 16th 2021]. <https://www.power-and-beyond.com/whats-a-printed-circuit-board-pcb-a-893758/>
- 17 SFUPTOWNMAKER. PCB Basics. [Online] [Cited on February 16th 2021]. <https://learn.sparkfun.com/tutorials/pcb-basics/all>
- 18 Elprocus. Working on Different Types of Rectifiers. [Online] [Cited on February 18th 2021]. <https://www.elprocus.com/different-types-rectifiers-working/>
- 19 Elprocus. What is a full wave Rectifier: Circuit with Working Theory. [Online] [Cited on February 18th 2021]. <https://www.elprocus.com/full-wave-rectifier-circuit-working-theory/>
- 20 Elprocus. Different Types of Voltage Regulators with Working Principle. [Online] [Cited on February 18th 2021]. <https://www.elprocus.com/types-of-voltage-regulators-and-working-principle/>
- 21 Voltage Regulator-Sasmita. Transistor Shunt Voltage Regulator. [Online] [Cited on February 18th 2021]. <https://electronicspost.com/transistor-shunt-voltage-regulator/>
- 22 Component101. 7805 Voltage Regulator IC. [Online] [Cited on February 19th 2021]. <https://components101.com/ics/7805-voltage-regulator-ic-pinout-datasheet>
- 23 Electronicscomp. SIM800A GSM GPRS Module with RS232 Interface and SMA Antenna. [Online] [Cited on February 19th 2021]. <https://www.electronicscomp.com/sim800a-gsm-gprs-module-with-antenna>
- 24 Arduino. Overview. [Online] [Cited on February 19th 2021]. <https://www.arduino.cc/en/pmwiki.php?n=Main/arduinoBoardUno>
- 25 Bouni-Arduino pinout. [Online] [Cited on February 19th 2021]. <https://github.com/Bouni/Arduino-Pinout>
- 26 Akhil Ch. Hardware Structure of Arduino Uno. [Online] [Cited on February 19th 2021]. <https://www.instructables.com/Hardware-Structure-of-ARDUINO-UNO/>

- 27 Components101. MQ2 Gas Sensor. [Online] [Cited on February 19th 2021]. <https://components101.com/sensors/mq2-gas-sensor>
- 28 Electronicshub. How to Burn Bootloader on ATmega328 using Arduino Uno. [Online] [Cited on March 17th 2021]. <https://www.electronicshub.org/burn-bootloader-on-atmega328/>
- 29 Rishabh Jain. How to Burn Arduino Bootloader in Atmega328IC and Program it using Arduino IDE. [Online] [Cited on March 17th 2021]. <https://circuitdigest.com/microcontroller-projects/how-to-burn-bootloader-in-atmega328p-and-program-using-arduino-ide>
- 30 Manu08. Burning the Bootloader on ATMega328 Using Arduino UNO As ISP. [Online] [Cited on March 17th 2021]. <https://www.instructables.com/Burning-the-Bootloader-on-ATMega328-using-Arduino-/>

Appendix 1: Code for Receiver Part

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

SoftwareSerial myfirstSerial(8, 7);
LiquidCrystal lcd(11, 12, A0, A1, A2, A3);
String temp_value;
float first_temp = 17.00;
float temperature_value;
char p[4];
char l[5];
char u;
int a = 0, b = 0;
char temp[4];
const int value1 = 1;
const int value2 = 2;
void setup()
{
  pinMode(value1, OUTPUT);
  pinMode(value2, OUTPUT);
  digitalWrite(value1, LOW);
  digitalWrite(value2, HIGH);
  lcd.begin(16, 2);
  lcd.clear();
  lcd.print("Gas&Fire Detect");
  lcd.setCursor(0, 1);
  lcd.print("SMS & Xbee Alrt");
  delay(3000);
  myfirstSerial.begin(9600);
}

void loop()
{
  if (myfirstSerial.available())
  {
    while (myfirstSerial.available() > 0)
    {
      u = myfirstSerial.read();
      if (u == '*')
        a = 1;
      // else if (u == '@')
      //   a = 0;
      else if (u == '#')
        a = 2;
    }

    if (a==1)
    {
      lcd.clear();
      lcd.print("Fire Alert!!");
      lcd.setCursor (0, 1);
      lcd.print("High temp. detected");
      while (!(myfirstSerial.available()));
    }
  }
}
```



```
    a = 0;
}

else if (a==2)
{
    lcd.clear();
    lcd.print("Warning! Gas ");
    lcd.setCursor(0, 1);
    lcd.print("leakage detected!");
    while (!(myfirstSerial.available()));
    a = 0;
}
else
{
    lcd.clear();
    lcd.print("CNG&LPG Detect ");
    lcd.setCursor(0, 1);
    lcd.print("With GSM Alert");
    delay(200);
}

}
else {
    delay(500);
    while (!(myfirstSerial.available()))
    {
        lcd.clear ();
        lcd.setCursor (5, 0); //coloumn 0 row 0
        lcd.print ("Zigbee");
        lcd.setCursor (0, 1);
        lcd.print("  Disconnected");
        delay (200); // 2 secs
    }
}
}
```

Appendix 2: Code for Transmitter Part

```
#include <LiquidCrystal.h>
#include <math.h>
#include <SoftwareSerial.h>
SoftwareSerial myfirstSerial(6, 7);
SoftwareSerial mygsm(3, 2);

#define RECEIVE_A_CALL 1
#define IDLE_STATUE 2
#define BUSY_STATUE 3
#define NO_ANSWER_STATUE 4
#define TALKING_STATUE 5
#define ERROR_STATUE 6

#define configforsms "This number is configured."
#define userstopcall "USER cut the call"
#define noanswering "USER not Answering call..."
#define establishacall "call establish.."
#define callisdisconnected "disconnecting call"
#define callisunavailable "can't make CALL"
#define ERROR_str "ERROR"

#define RECEIVE_A_CALL 1
#define IDLE_STATUE 2
#define BUSY_STATUE 3
#define NO_ANSWER_STATUE 4
#define TALKING_STATUE 5
#define ERROR_STATUE 6

#define pins "pin"
#define fine "OK"
#define errorstring "ERROR"
#define conntedstatue "Connected"
#define connectingstatue "connecting"
#define CLCC "+CLCC:"
#define atCLCC "AT+CLCC"
#define set_textmode "AT+CMGF=1"
#define list_messages "+CMGL:"
#define send_messages "AT+CMGS=\"\"
#define delete_message "AT+CMGD="
#define unread_message "AT+CMGL=\"REC UNREAD\"\"
#define wait_for_delete "AT+CMGD=0,4"
#define new_message_indications "AT+CNMI=0,0,0,0"
#define CLIP "+CLIP"

String firsttempt;
int sensor_value = 0;
int pwm_value = 0;
int buzzpwm_value = 0;
int updated_message = 0;
double temp_value = 0;

const int pinforsensor = A0;
const int pinforled = 4;
const int pinforbuzz = A5;
```

```
boolean gsm_connect_statue = false;
boolean connecting_statue = true;
int configuration_statue = 0;
unsigned long current_time ;
String telenumner;
String message = "";
String comingdata = "";
String comingstring = "";
String indexmessage;
int value1;
int value2;
String calling, number1;
String messaging, number2; // = get_sender_number();
double value3 = 0;

LiquidCrystal lcd(8, 9, 10, 11, 12, 13 );
void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);

  myfirstSerial.begin(9600);

  mygsm.begin(38400);
  pinMode(pinformled, OUTPUT);
  digitalWrite(pinformled, LOW);
  pinMode(pinformbuzz, OUTPUT);
  digitalWrite(pinformbuzz, HIGH);
  lcd.clear();
  lcd.print("Gas&Fire Detect");
  lcd.setCursor(0, 1);
  lcd.print("SMS & Xbee Alrt");
  delay(3000);
  lcd.clear();
  lcd.print(F("connecting to "));
  lcd.setCursor(0, 1);
  lcd.print(F("gsm"));
  lcd.clear();
  lcd.print(F("Connecting GSM.."));
  if (gsm_connect_statue())
  {
    lcd.clear();
    lcd.print(F("GSM Connected.."));
  }
  else
  {
    lcd.clear();
    lcd.print(F("not connected"));
    while (1);
  }
  delay(2000);
  lcd.clear();
  lcd.print(F("Waiting for USER"));
  lcd.setCursor(0, 1);
  lcd.print(F("To Call"));
  while (!configuration_statue)
  {
    switch (call_status())
```

```
{
  case IDLE_STATUE: // Nothing is happening
    break;

  case RECEIVE_A_CALL: // Yes! Someone is calling us
    telenumber = get_call_number();
    lcd.clear();
    lcd.print(F("Receiving call"));
    lcd.setCursor(0, 1);
    lcd.print(telenumber);
    hangcall();
    delay(1500);
    telenumber = telenumber;
    lcd.clear();
    lcd.print(F("Sending SMS"));
    sendsms(F("This number configured as USER"), telenumber);
    lcd.clear();
    lcd.print(F("SMS sent.."));
    configuration_statue = 1;
  }
}
delay(1000);
}

void loop()
{
  while (1)
  {
    sensor_value = analogRead(pinforsensor);
    pwm_value = map(sensor_value, 0, 1023, 0, 255);
    pwm_value -= 30;
    if (pwm_value < 0)
    {
      pwm_value = 0;
    }

    firsttempt = String(thermistor_value(analogRead(A1)), 1);
    for (int p = 0; p < 10; p++)
    {
      value3 += thermistor_value(analogRead(A1));
    }
    value3 = value3 / 10;
    if (pwmVal > 50 || thermistor_value(analogRead(A1)) > 40)
    {
      if (!updated_message)
      {
        if (thermistor_value(analogRead(A1)) > 40)
        {
          myfirstSerial.listen();
          firsttempt = firsttempt + '*';
          myfirstSerial.print(templ);
          gsm.listen();
          digitalWrite(pinforbuzz, LOW);
          digitalWrite(pinforled, HIGH);
          lcd.clear();
          lcd.print("HighTemperatures");
          lcd.setCursor(0, 1);
        }
      }
    }
  }
}
```

```
lcd.print("Fire Alert!");
delay(3000);
updated_message = 1;
lcd.clear();
lcd.print(F("Sending SMS"));
sendsms(F("Fire Alert!\nHigh Temperatures have been
detected!"), telenumber);
lcd.clear();
lcd.print(F("SMS sent.."));
delay(1000);
while (thermistor_value(analogRead(A1)) > 40);
{
  lcd.clear();
  lcd.print("HighTemperatures");
  lcd.setCursor(0, 1);
  lcd.print("Temp = ");
  lcd.print(thermistor_value(analogRead(A1)));
  myfirstSerial.listen();
  firsttempt = firsttempt + '*';
  myfirstSerial.print(firsttempt);
  delay(200);
}
}
else
{
  myfirstSerial.listen();
  firsttempt = firsttempt + '#';
  myfirstSerial.print(templ);
  digitalWrite(pinforbuzz, HIGH);
  digitalWrite(pinforled, LOW);
  mygsm.listen();
  lcd.clear();
  lcd.print("AlertCombustible");
  lcd.setCursor(0, 1);
  lcd.print("Gas Detected");
  updated_message = 1;
  delay(3000);
  digitalWrite(pinforbuzz, LOW);
  digitalWrite(pinforled, HIGH);
  lcd.clear();
  lcd.print(F("Sending SMS"));
  sendsms(F("Warning! Gas leakage detected!"), telenumber);
  lcd.clear();
  lcd.print(F("SMS sent.."));
  delay(1000);
  while (pwm_value > 50)
  {
    sensor_value = analogRead(pinforsensor);
    pwm_value = map(sensor_value, 0, 1023, 0, 255);
    pwm_value -= 30;
    if (pwm_value < 0)
    {
      pwm_value = 0;
    }
    lcd.clear();
    lcd.print("AlertCombustible");
    lcd.setCursor(0, 1);
    lcd.print("Gas Present");
```

```

        myfirstSerial.listen();
        firsttempt = firsttempt + '#';
        myfirstSerial.print(firsttempt);
        delay(200);
    }
}
}
else
{
    digitalWrite(pinforbuzz, HIGH);
    digitalWrite(pinforled, LOW);
    updated_message = 0;
    lcd.clear();
    lcd.print("Monitoring...");
    lcd.setCursor(0, 1);
    lcd.print("Temp = ");
    for (int p = 0; p < 10; p++)
    {
        value3 += thermistor_value(analogRead(A1));
    }
    value3 = value3 / 10;
    lcd.print( value3, 1);

    myfirstSerial.listen();
    firsttempt = firsttempt + '@';

myfirstSerial.print(firsttempt);////////////////////////////////////

    delay(600);
}
}
}

double thermistor_value(int ADC_value) {
    double tempt_value;
    tempt_value = log(10000.0 / (1024.0 / ADC_value - 1)); // for
pull-up configuration
    tempt_value = 1 / (0.001129148 + (0.000234125 + (0.0000000876741
* tempt_value * tempt_value )) * tempt_value );
    tempt_value = tempt_value - 273.15;           // Convert Kelvin
to Celcius
    return tempt_value;
}

```