CHAPTER ONE

1.1Introduction

Safety against fire is becoming more and more importantruletoday . Monitoring the commercial and residential areas allround is an effective method to reduce personal and property losses due to fire disasters .Firefighting systems used to prevent, extinguishor put out fires in enclosedspaces.The traditional fireprotection units are widely deployedbut that are actuated manually by an operator which may cost time and property .Automatic firefighting systems are installed in building and rooms where the fire hazard relatively is high .An automatic firefightingsystem includes an sensor capable of detecting combustion , alarmsignaling devices, fireextinguishing equipment, starting and stopping devices , and feeders the fire extinguishing substances . Automatic fire protection systems are classified according to the time into ultrahighspeed (to 0.1 second), highspeed(to 3 second),and standard (to 180 second).The fireextinguishing substance can be applied for periods ranging from 30 second to 3600 second[1].A simple automatic fire alarm system for industrial plants buildings based on an Atmega-8A microcontroller using sensors is designed and implemented in this project.

1.2Problem Definition

Design very fast automatic fire alarm system to detects fire and gas leakage in hazard areas such as industrial plant to reduce personal and property losses due to fire disasters.

1.3Objectives

- Design very fast system to monitor and observe the current status of the environment of the industrial plant and send voice and visual alarm when fire occurs.

- Reducing the risks which facing the workers in the industrial plant.

- The automatic firefighting system must be continuously communication with civilian defense emergency number.

- Decreasing the physical effort and thecost for long period of time .

1.4 Methodology

In order to meet the above stated objectives, the system should be simulated in Proteus VSM,Amicrocontroller (Atmega-8A) is used to process the various sensor signals and control the system actuators accordingly.A firefighting pump will be interfaced to the microcontroller through a relay.A software code is developed to controlthe overall system functions.The code is written in C++ language using Basic Compiler for AVR(BASCOM-AVR).

1.5 Project layout

This is project include five chapters:

Chapter two describes overview of firefighting system includes old fire fighter method and fire classification.

Chapter three describes system components as sensors,relay,pump,buzzer.

Chapter four describes simulation and results.

Chapter five conclusion and recommendations.

CHAPTER TWO

OVERVIEW OF FIREFIGHTING SYSTEM

2.1 Overview

Fire is the rapid oxidation of a material in the exothermic chemical process of combustion, releasing heat , light, and varies reaction products. The flame is the visible portion of the fire.Ifthehotenoughgases may becomeionized to produce plasma.Fire is an important process that affects ecological systems around the globe. The positiveeffects of fire include stimulatinggrowthand maintaining various ecologicalsystems. Fire has been used by humans for cooking , generating heat , light, signaling , and propulsion purposes.The negative effects of fire include hazard to life and property ,atmospheric pollution. A firefighter suppresses and extinguishes fires to protect lives and prevent the destruction of property and the environment .

2.2 Old TacticsandTools

The old fire-fighting in the middle ages to volunteer on their own population to extinguish the fire with the aid baldla that fill the water and passed between the hands from water source until the fire place and then come back in the last row to water source again.Fire-fighting units have evolved to the appearance of the water pumps.

2.3 Classification of Fires

Fires are classified by the types of fuel they burn as follows:

Class A:fires involving ordinary combustible materials such as wood , cloth, paper and plastics.

Class B:fires involving flammable liquids , petroleum oils , greases.

Class C:fires involving energized electrical equipment (transformers, motors) in which the use of an extinguishing media that is electrically non conductive is important.

Class D:fires involving combustible metals such as magnesium, titanium, sodium.

2.4 FireAlarmSystems

All firealarmsystemsessentially operateon thesameprinciple. If a detector detects smoke or heat or someone operates a break glass unit (manual break point) , then alarm sounders operate to warn others in the building that there may be a fire and to evacuate. It may also incorporate remote signaling equipment which would alert the fire brigade via a central station [3] . Fire Alarm Systems can be broken down into categories :

2.4.1 ConventionalFire AlarmSystem

In a Conventional Fire Alarm System a number of call points and detectors are wired to the Fire Alarm Control Panel in Zones. A zone is a circuit and typically one would wire a circuit per floor orfire compartment. The firealarm Controlpanel has a number of zones lamps. The reason for having zones is to give a rough idea as to where a fire has occurred. This is important for the fire brigade and of course for the building management. The accuracy of knowing where a fire has started is controlled by the number of zones .A control panel has a number of circuits that have been wired within the building. The Control Panel is wired to a minimum of two sounder circuits which could contain bells, electronic sounders or other audible devices. Each circuit has an end of line device which is used for monitoring purposes[3].

2.4.2 Addressable Systems

The detection principle of anaddressable system is similar to a conventional system except that the control panel can determine exactly which detector or call point has initiated the alarm. The detection circuit is wired as a loop and up to 99 devices may be connected to each loop. The detectors areessentially conventional detectors with an address built in. The address in each detector is set by dil switches and the control panel is programmed to display the information required when that particular detector is operated. Additional fielddevices are available which may be wired to the loop for detection only i.e. it is possible to detect a normally open contact closing such as sprinkler flow switch, or

a normally closed contact opening. Sounders are wired in a minimum of two sounder circuits exactly as a conventional system. Loop isolation modules are available for fitting on to the detection loops such that the loop is sectioned in order to ensure that a short circuit or one fault will only cause the losses of a minimal part of the system [3].

2.4.3Wireless Fire Alarm System

Wireless fire alarm systems are an effective alternative to traditional wired fire alarm systems for all applications. They utilize secure licensefree radio communications to interconnect the sensors and devices (smoke detectors , call-points , etc.) with the controllers . It is a simple concept , which provides many unique benefits and is a full analogue addressable fire detection system without the need for cable[3].

2.5 Key Components of a Fire Alarm System

The key components of the fire alarm system are the controlpanel detector.

2.5.1Control panel

The control panel is the brain of the fire alarm system. The control panel constantly monitors the detectors installed throughout the protected facility for signs of fire .When a fire condition is detected the control panel can carry out a range of activities including Sound the evacuationalarms for the facility,Notify the Fire Service, Notify thebuildingmanagement,Closefireand smoke doors.

2.5.2 Detectors

The fire protection is able to offer a full range of detectors to suit arrange of different environments and hazards.The most fire detectors used in fire and gas detection system is heat and smoke detectors.

CHAPTER THREE

DESCRIPTIONOFCOMPONENTS SYSTEMS

3.1 System Layout

The block diagram of the hardware of implementation of the entire system is as shown in Figure (3.1) .The aim of the project is to illustrate the usage of the fire fighter and its applications and the minimum equipment required to construct the firefighting system is a microcontroller , pump water , smoke sensor , temperature sensor , flame sensor ,led , buzzer and relay.

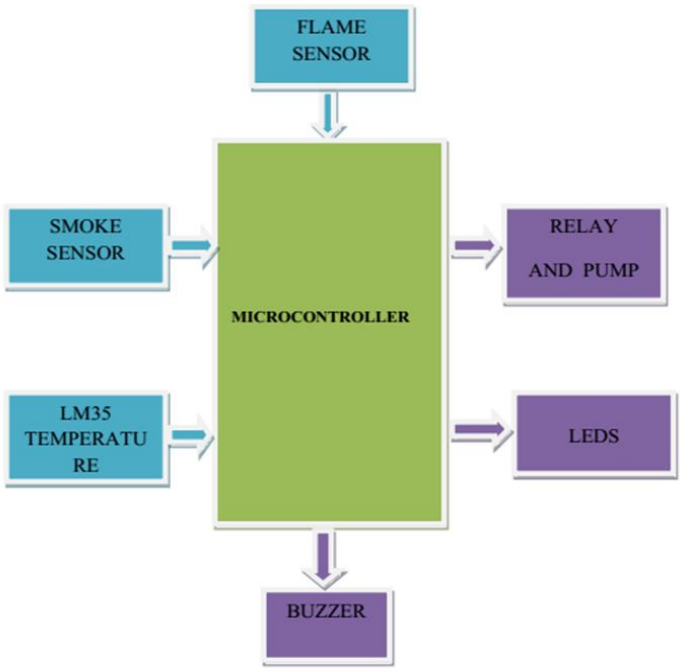


Figure 3.1 Hardware Implementation of Automatic Firefighting system.

3.2 Input Unit

Inputsystemconsists oftowsensors. The mechanism of the input system is described as follow.

3.2.1 LM35 Temperature Sensor

The LM35 is an integrated circuit sensor that can be used to measure temperature with an electrical output proportional to the temperature.

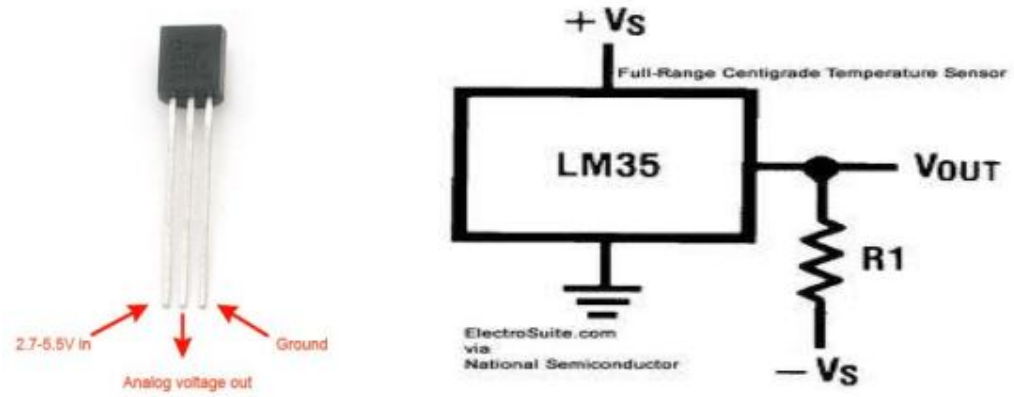


Figure 3.2: LM35 Temperature Sensor

3.2.2Smoke Sensor

A smoke detector is a device that detects smoke typically as an indicator of fire. Commercial , industrial , and mass residential devices issue a signal to a fire alarm system while household detectors , known as smoke alarms , generally issue a local audible or visual alarm from the detector itself. Smoke detectors are typically housed in a disk-shaped plastic enclosure about 150 millimeters (6 in) in diameter and 25 millimeters (1 in) thick, but the shape can vary by manufacturer or

product line. Most smoke detectors work either by optical detection (photoelectric) or by physical process (ionization) , while others use both detection methods to increase sensitivity to smoke. Sensitive alarms can be used to detect , and thus deter , smoking in areas where it is banned such as toilets and schools . Smoke detectors in large commercial , industrial , and residential building are usually powered by a central fire alarm system, which is powered by the building power with a battery backup. However , in many single family detached and smaller multiple family housings, a smoke alarm is often powered only by a single disposable battery [6].

Figure 3.3: Smoke Sensor

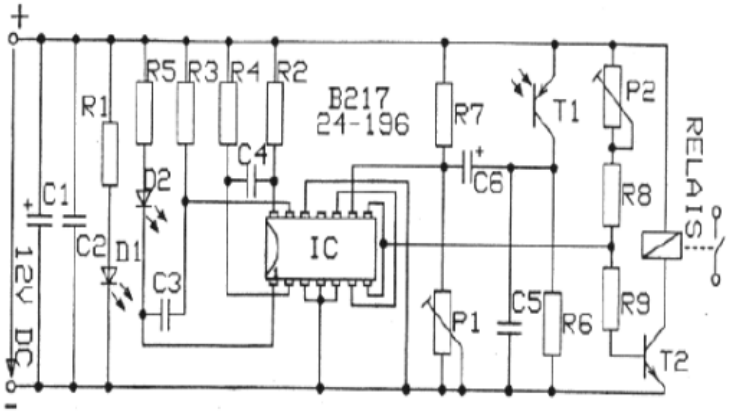


Figure 3.4: A Smoke sensor circuit

3.2.3Flame Sensor

Flame type detectors are sophisticated equipment to detect the flame phenomena of a fire. These detectors have various types depending on the light wavelength they use .Such as , ultraviolet , near infrared , and combination of UV/IR type detectors.

UV detectors generally work with wavelengths shorter than 300 nm. This type of detectors can detect fires and explosions situations within 3-4 milliseconds from the UV radiation emitted from the incident. However , to reduce false alarm triggered by UV sources such as lighting , arc welding etc. a time delay is often included in the UV flame detector. The near Infrared sensor or visual flame detectors work with wavelengths between 0.7 to 1.1 micrometer. One of the most reliable technologies available for fire detection, namely multiple channel or pixel array sensors , monitors flames in the near IR band. The Infrared (IR) flame detectors work within the infrared spectral band (700 nm – 1 mm) .

Usual response time of these detectors is 3 – 5 seconds. Also , there is UV and IR combined flame detectors , which compare the threshold signal in two ranges to detect fire and minimize false alarms.

Flame detectors are expensive and complex , though they provide very reliable and accurate response . They can operate in highly sensitive environment where other detectors can't be used. Aircraft maintenance facilities, fuel loading platforms, mines, high-tech industries etc. Use these flame detectors for safety [6].

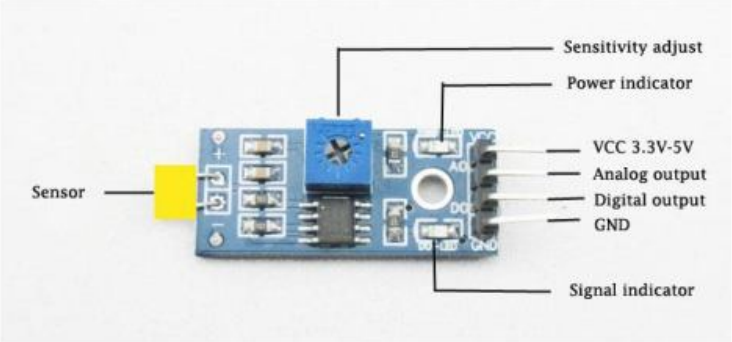


Figure 3.5: Flame sensor

3.3System Process

Processing system acts as the brain of robot, which generates desired output for corresponding inputs .For that purpose, microcontrollers are used. In present days there are several companies that manufacture microcontrollers , for example A TMEL ,Microchip , Intel,MOTROLAetc. ATmega-8A microcontroller is used in this robot. It is an ATMEL product. It is also called AVR.

3.4 Output Unit

An output device is any piece of hardware equipment used to communicate the results of data processing carried out by an information processing system(such as a microcontroller) which converts the electronically generated information into sensors.

3.4.1 Transistor –NPN(BC 548)

A transistor is a semiconductor device used to amplify and switch electronic signals and electrical power. It is composed of semiconductor material with at least three terminals for connection to an external circuit. A voltage or current applied to one pair of the transistor's terminal changes the current through another pair of terminals. Because the controlled (output) power can be higher than the controlling (input) power, a transistor can amplify a signal. This is the BC548, NPN silicon BJT (Bipolar Junction Transistor). This little transistor can help drive large loads or amplifying or switching applications. The BC548 is specifically rated at 50V and 800mA max [9].

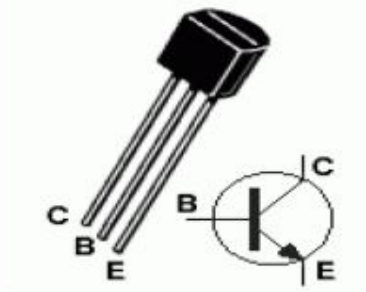


Figure 3.7: BC548 Transistor

3.4.2Relay

Relay are operated switches many relays use an electromagnet to operate a switching mechanism mechanically but other operating principles are also used. Relays are used where it is necessary to control a circuit by

a low power signal with complete electrical isolation between control and controlled circuits or where several circuits must be controlled by one signal. Relays and switches come in different configurations. The most common are single pole single Throw SPST is the simplest with only two contacts. Single pole Double Throw SPDT has three contacts. The contacts are usually labeled common COM , normally open NO and normally closed NC. The normally closed contact will be connected to the common contact when no power is applied to the coil. The normally open contact will be open i.e. not connected when no power is applied to the coil. When the coil is energized the common is connected to the normally open contact and the normally closed contact is left floating. The Double pole versions are the same as the single pole version except there are two switches that open and close together . Figure 3.6 illustrates the relay [10].

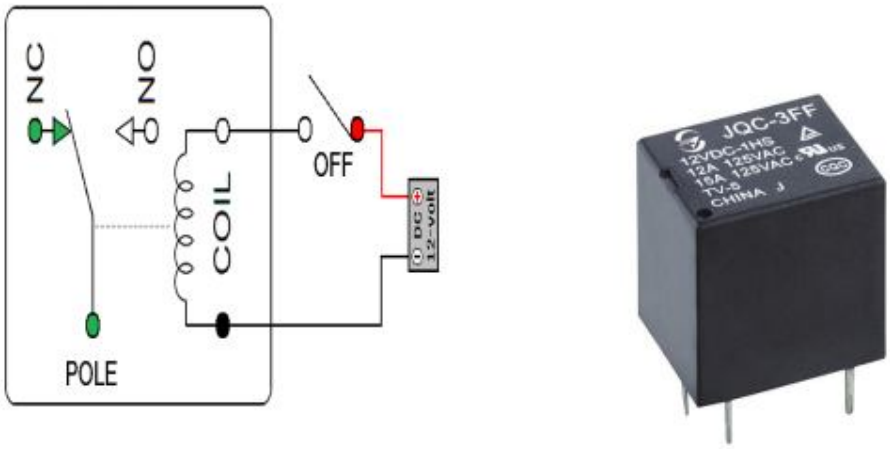


Figure 3.8: Relay

3.4.3 Pump

A pump is a device that moves fluids (liquids or gases) or sometimes slurries by mechanical action. Pumps can be classified into three major groups according to the method they use to move the fluid: direct lift , displacement , and gravity pumps .[1]

Pumps operate by some mechanism (typically reciprocating or rotary) and consume energy to perform mechanical work by moving the fluid. Pumps operate via many energy sources , including manual operation , electricity , engines ,or wind power come in many sizes from microscopic for use in medical applications to large industrial pumps.

Mechanical pumps serve in a wide range of applications such as pumping water from wells , aquarium filtering , pond filtering and aeration , in the car industry for pumping oil and natural gas or for operating cooling towers. In the medical industry , pumps are used for biochemical processes in developing and manufacturing medicine , and as artificial replacements for body parts , in particular the artificial heart and penile prosthesis [11].

Figure 3.9:Pump.

3.4.4 Light Emitting Diode (LED)

Light-emitting diode (LED) is a two-lead semiconductor light source that resembles a basic PN-junction diode , except that an LED also emits light .When an LED's anode lead has a voltage drop, current flows. Electrons are able to recombine with holes within the device , releasing energy in the form of the light (corresponding to the energy of the photon) is determined by the energy band gap of the semiconductor , as shown in the Figure (3.8).

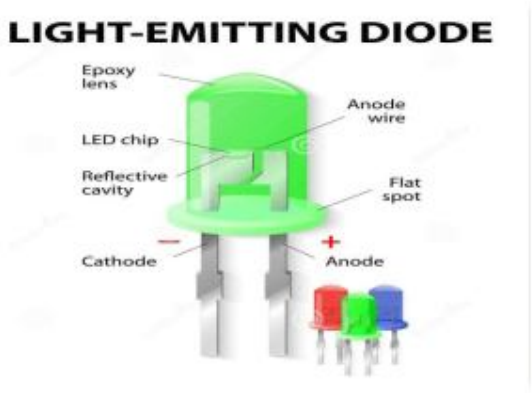


Figure 3.10: Light Emitting Diode

3.4.5 Buzzer

Abuzzer or beeper is an audio signaling devise which may be mechanical , electromechanical or piezoelectric . Typical uses of buzzers and beepers include alarm devices, timers and confirmation of user input such as a mouse click or keystroke.

Figure 3.11: Buzzer

CHAPTER FOUR

SIMULATION AND RESULTS

4.1 Programming Microcontroller

Programs are known as software. A program is set of instructions that the microprocessorcanexecute.The program is stored in the microcontroller's memory in the form of binary numbers called machine instructions . All microcontrollers operate on a set of instructions stored in their memory. A microcontroller fetches the instructions from its program memory one by one , decodes these instructions , and then carries out the required operations. Microcontrollers have traditionally been programmed using the assembly language of the target device. And when you write a source code with the C language the compiler of the C language is necessary .

4.2 Machine Code

A machine instruction is a combination of 0s and 1s that informs the CPU to perform certain operation . The most fundamental program from is machine code is the binary instructions that cause the CPU to perform the desired operations . Machine code or machine language is a system of instructions and data executed directly by CPU.

Machine code may regarded as a primitive programming language or as the lowest level representation of a compiled assembled computer program. Programs in interpreted languages are no represented by machine code however their interpreter which may be seen as a processor executing the higher level program. Machine code is sometimes called native code when referring to platform dependent parts of language features or libraries . Machine code should not be confused with byte code , which is executed by an interpreter.

Every processor or processor family has its own machine code instruction set. Instructions are patterns of bits that by physical design correspond to different commands to the machine . The instruction set is thus specific to a class of processors using the same architecture .

4.3 High Level Language

High level languages such as C , C++ , and Java were invented to avoid the problems of assembly language programming . High level Languages are close to plain English and hence a program written in a high level language becomes easier to understand . Statements in high level language often needs to be implemented by tensor even hundreds of assembly instructions . The programmer can now work on the program logic at a much higher level which makes the programming job much easier . A program written in a high level language is also called a source code and it requires a software program called a compiler to translate it into machine instructions . A compiler compiles a program into object code . Just as there are cross assemblers there are cross

compilers that run on one machine but translate programs into machine instructions to be executed on a computer with a different instruction set. Some high level languages are interpreted that is they use an interpreter to scan the user's source code and perform the operations specified interpreters do not generate object code . Programming languages that use this approach include Basic , Lisp , and Prolog . The Java language is partially compiled and partially interpreted program written in Java language is first compiled into byte code and then interpreted . The design purpose of this language is compiled once run everywhere .

High level languages are not perfect one of the major problems with high level languages is that the machine code compiled from a program written in a high level language is much longer and cannot run as fast as its equivalent in the assembly language . For this reason , many time critical programs are still written in assembly language .

4.4 Code Simulator

Now once you have the object file you can transfer the op-codes into the AVR controller and if you have written a program exactly according to the system design it may work correctly. However there are good chances that your program does not work as expected . In such a case you will need to find out where the errors are and make suitable chances or additions to the source file assembly the source file again and load the machine code into the controller . This may be a time consuming iterative process rather than transferring the op-codes

into the controller and debugging the code it is possible to simulate the working of the AVR controller on the development PC itself without downloading the machine code into the controller . A program that simulates or mimics the AVR controller is called a simulator . A simulator can execute the program code one code at a time displaying the result on the screen contents of registers , ports , status, etc. and this can help ensure that the program works as expected .

4.4.1 AVR simulator

The AVR simulator executes object code files generated for the AVR microcontrollers . It also supports simulation of various I/O functions. The simulator can be controlled through a command line as well as through menus . The AVR simulator is very easy to use .

4.4.2 AVR studio

The AVR Studio is a development tool for AVR controllers . It allows control of execution of programs in circuit emulator or on the built in AVR instruction set simulator . The AVR studio supports source level execution or AVR assembler programs as well as C programs compiled with IAR . At the time of starting the AVR Studio if the AVR in circuit emulator is connected and powered on the AVR Studio detects it and enables execution of programs on the emulator . Otherwise it invokes the built in AVR instruction set simulator for source level simulation .

4.4.3Proteus VSM

It allow to draw a complete circuit for a microcontroller based system and then test it interactively , all from within the same piece of software . Meanwhile ISIS retains a host of features aimed at the Printed Circuit Board PCB designer , so that the same design can be exported for production with ARES or other PCB layout software . Major features of PROTEUS VSM include :

-Support for both interactive and graph based simulation.

-CPU models available for popular microcontrollers such as the PIC and 8051 series.

-Interactive peripheral models include Light Emitted diode LED and liquid crystal display LCD and a whole library of switches , pots, lamps etc.Virtual instruments include voltmeters , ammeters , a dual beam oscilloscope and a 24 channel logic analyzer .

4.5Download a Program into Microcontroller

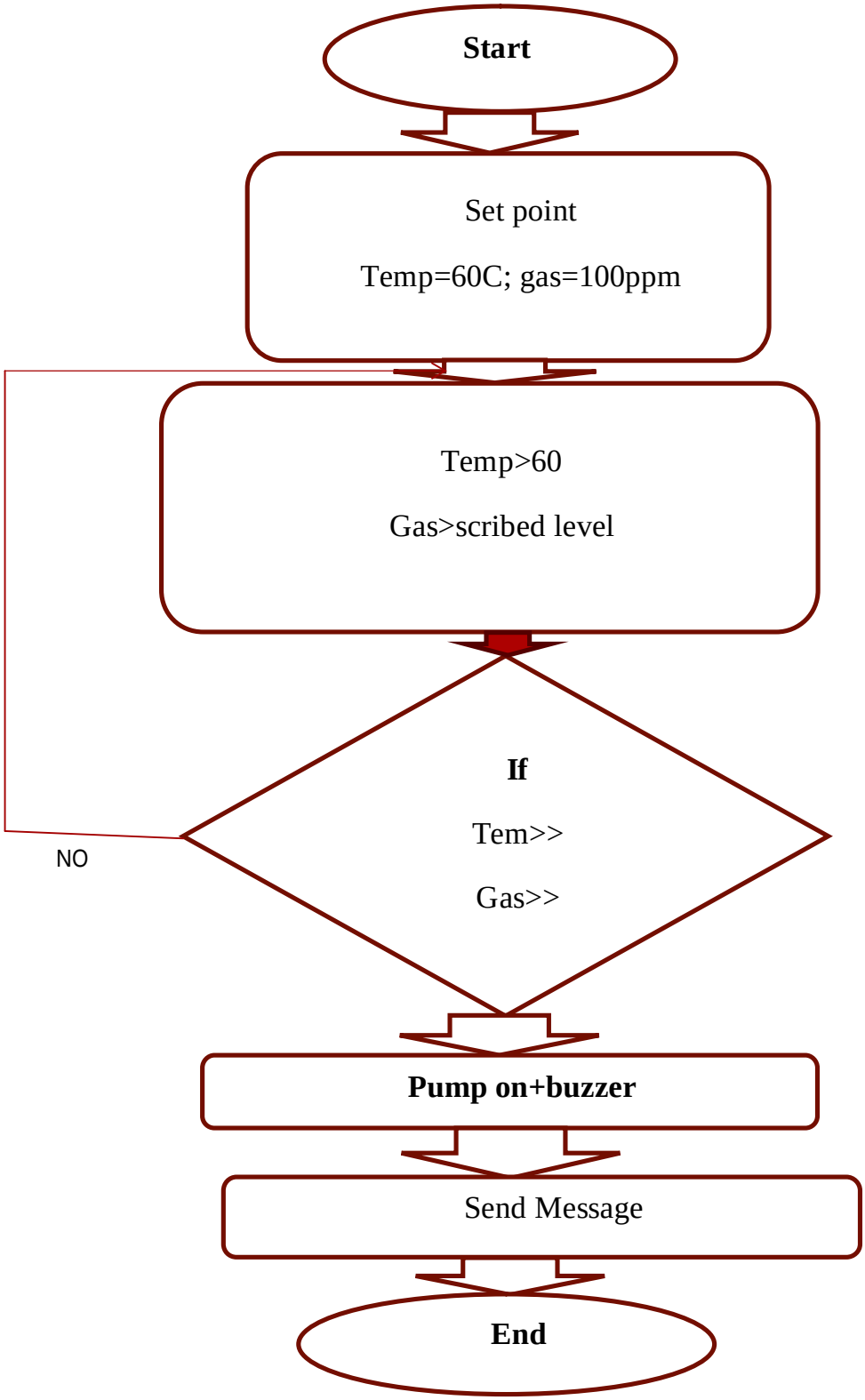
After compiling a microcontroller program in Bascom will be able to generate hex files from the program codes . Hex file is a machine code that corresponds to our language codes. Only microcontroller and microprocessor can interpret it. After building a hex file write it to the memory of microcontroller . This processor is called burning of microcontroller .Now open Proteus and make the design which will

include the required microcontroller and the double click on the microcontroller and then include that hex file which was created . This way you can simulate your design.

4.6 Software Description

The software was written in ATMEL AVR Basic Compiler (BASCOM – AVR) . BASCOM – AVR is an Integrated Development Environment (IDE) that includes a BASIC Compiler for the AtmelAVR microcontrollerfamily EditorAVRSimulator and In System programming support for a rangeof3rdparty hardware .

4.7 Flow Chart



4.8 System Operation

When the system is powered up and switched on the system will run in auto mode , the microcontroller then proceeds examining the two sensors located in each room .First the microcontroller calibrates the overall temperature starting in all rooms and incase it detects that the temperature of any of these rooms has gone outside its normal parameters , which is above 60 C corresponding alarm ledswill light up, then the microcontroller is proceeds with examining the gas sensors in the same manner , and if smoke is detected in any room , the microcontroller will light up its corresponding indicator hazard led , also microcontroller will switch on the firefighting pump . The same procedure is followed with the flaming sensor , if flames are detected the microcontroller will light up its corresponding indicator hazard led , and also will switch on the firefighting pump . The firefighting pump is meant to operate in a single room rather than the all rooms.

The manual mode designed to run at emergency case , it work by pressing the fire alarm call point switch. In this mode led's , buzzer , and firefighting pump in all rooms will switch on , and continuous running until it reset.

4.9 Simulation design

When the system operate on auto mode if one smoke or temperature sensor has gone outside its normal parameters , alarm led light up.

If tow smoke and temperature sensor has gone outside its normal parameters together , hazard led light up , pump and buzzer on.

When flame sensor detected flame hazard led light up , pump and buzzer on. At manual mode , when fire alarm call point switched on , hazard red led light up , pump and buzzer switch on , and continuous running until it rested.

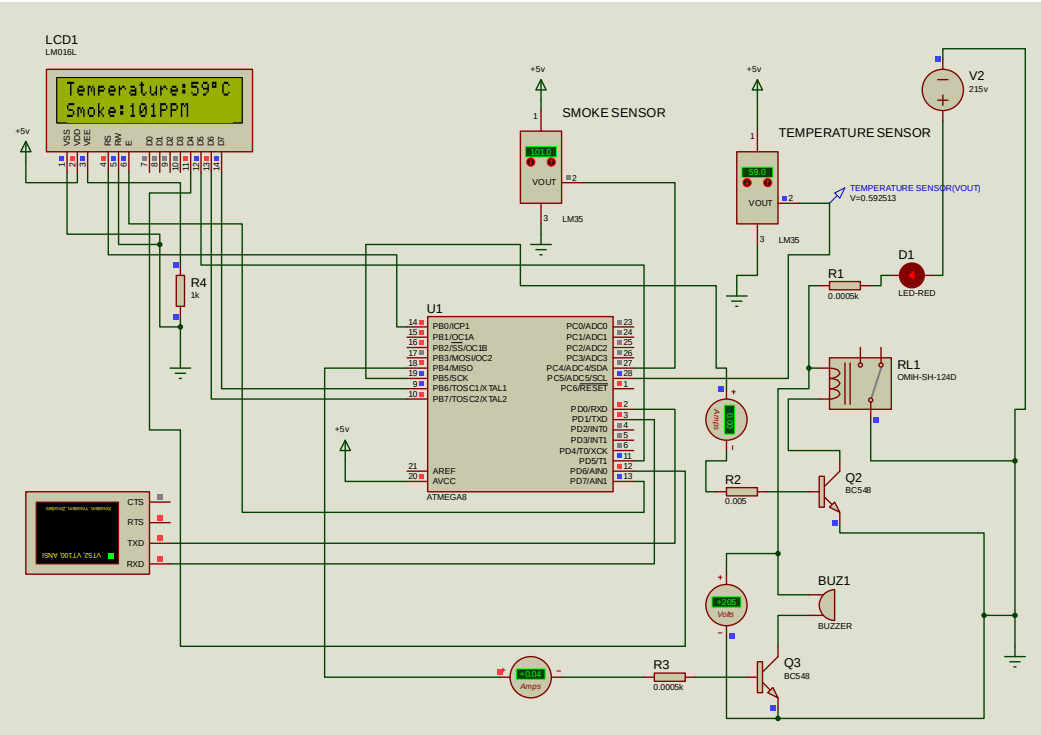


Figure 5.1: Fire and Gas detection simulation system.

4.10 Results

The system performs automatic fire fighting task when the system assures the fire occurrence . Table (5.1) below shows the results

whenoperating the system . The table indicates the action taken for each happening.

Table 5.1: Theresults when operating the system

Temperature sensor	Smoke sensor	Hazard led and buzzer	action
0	0	off	No action
0	1	on	Alarm led on
1	0	on	Pump on+ alarm led on
1	1	on	Pump+ alarm led on

Note: Logic 1=sensor is activated, logic 0= sensor is not activated.

4.11Discussion

As shown in table (5.1) , the alarm led switch on when smoke sensor or temperature sensor activated . While pump , hazard led , and buzzer

are switch on ,when smoke sensor or temperature sensor , or both activated.

CHAPTER FIVE

CONCLUSION AND RECOMMENDATION

5.1Conclusion

The designed fire-fighting system is able to detect and deal with overtemperature and smoke . After develop and testing the software the automatic fire-fighting system has successfully completed the tasks as expected and as outlined in the report . The system is capable of indicating its status on an LED , buzzer , and detecting and putting out fire. The ATmega-8A microcontroller is used to process the sensor circuitry input and control the indicator panel , and also used to control a firefighting pump to put out fire when detected . A simulation of the system is created and it gave a good performance.

5.2Recommendations

Thisproject leads to recommendations concerning the wireless Firefighting system problems in the evaluation of both source code and simulation. The following points summarize some recommendations for the future work and further improvements :

1-Use wireless GSM/GPRS module that enables the system to wirelessly alert certain someone (e.g. the owner) by texting their predefined numbers and inform them with the current status of their controlled environment.

2-Insert a solar cell to the system as a power backup.

References

[1]ABS GUIDANCE NOTES ON FIRE-FIGHTING SYSTEMS , America Bureau of Shipping Incorporated by Act of Legislature of the State of New York 1862 , Copyright 2005.

[2]Int. J. Communications , Network and System Sciences, 2016 , 7 ,386-395, www.scirp.org/journal/ijcns.

[3]TimWilmshurst , " Designing Embeded system with PIC microcontrollers principles and application" , newness , London , 2007.

[4]Ahmed Z.Kulab, FadyR.Alokka and others "Educational Material for PICmicrocontroller" , Islamic University of Gazza.

[5]DhananjayV.Gader "Programming and Customizing the AVR Microcontroller" , McGraw-hill,2001.

[6]Han –Way Hung "PIC microcontroller An Introduction to software and Hardware Interfacing" , Thomson Delmar lerning , 2005.

[7]Martin Bates, " Interfacing PIC Microcontrollers Embedded Design by Interactive Simulation " , Jordan Hill, 2006.

[8]Fire Alarm Systems Training Manual, potter Electronic signal company LLC,www.pottersignal.com.

[9]Jon Hind Bsc ,Fire and Gas Detection and control in the Process Industry, Risk&safety Group, Worley parsons,Kazakhstan.

[10]GSM Based Fire Alarm System ,www.circuit4you.com, Friday ,June 5,**2015**.

[11]Integrated fire and Gas Solution,www.honeywellprocess.com ,2009

[12]Describe Industrial Fire Detection and Alarm Systems, www.hdc.com.

APPENDIX

AppendixA

ATMEGA 8A-DATA SHEET

Features:

- High-performance, low – power Atmel AVR 8-bit Microcontroller.

- Advanced RISC Architecture.

-130 powerful Instruction – Most Single – clock Cycle Execution.

-32×8 General purpose working Registers.

-Fully Static Operation.

-Up to 16 MIPS Throughput at 16MHz.

-On-chip 2-cycle Multiplier.

- High Endurance Non-volatileMemorysegments

-8Kbytes of In-system Self-programmable Flash program memory

-512 Bytes EEPROM.

-1Kbyte Internal SRAM

-Write/Erase Cycles10, 000Flash /100,000 EEPROM.

-Data retention: 20 years at 85□ C/100 years at 25□ C.

-Optional Boot Code section with Independent lockBits

- In-System Programming by on-chip Boot program

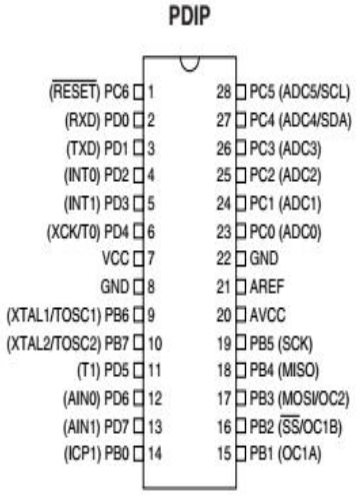
- True Read-While –Write Operation

-Programming lock for software security.

- **Peripheral Features**
  - **Two 8-bit Timer/Counters with Separate Prescaler, one Compare Mode**
  - **One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode**
  - **Real Time Counter with Separate Oscillator**
  - **Three PWM Channels**
  - **8-channel ADC in TQFP and QFN/MLF package**
    - **Eight Channels 10-bit Accuracy**
  - **6-channel ADC in PDIP package**
    - **Six Channels 10-bit Accuracy**
  - **Byte-oriented Two-wire Serial Interface**
  - **Programmable Serial USART**
  - **Master/Slave SPI Serial Interface**
  - **Programmable Watchdog Timer with Separate On-chip Oscillator**
  - **On-chip Analog Comparator**
- **Special Microcontroller Features**
  - **Power-on Reset and Programmable Brown-out Detection**
  - **Internal Calibrated RC Oscillator**
  - **External and Internal Interrupt Sources**
  - **Five Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, and Standby**
- **I/O and Packages**
  - **23 Programmable I/O Lines**
  - **28-lead PDIP, 32-lead TQFP, and 32-pad QFN/MLF**
- **Operating Voltages**
  - **2.7 - 5.5V**
- **Speed Grades**
  - **0 - 16MHz**
- **Power Consumption at 4Mhz, 3V, 25·C**
  - **Active: 3.6mA**
  - **Idle Mode: 1.0mA**
  - **Power-down Mode: 0.5µA**

Pin Configurations

Figure 1-1. Pinout ATmega8A



Pin Description

-VCC:

- Digital supply voltage.

-GND:

Ground

-Port B (PB7:PB0)-XTAL1/XTAL2/TOSC1/TOSC2:

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The port B output buffers have symmetrical drive characteristics with both high sink and source current if the pull-up resistors are activated. The port B pins are tri-stated when a reset condition becomes active , even if the clock is not running.

Depending on the clock selection fuse settings , PB6 can be used as input to the inverting Oscillator amplifier and input to the internal clock operating circuit. Depending on the clock selection fuse settings , PB7 can be used as output from the inverting oscillator amplifier. If the Internal Calibrated RC oscillator is used as chip clock source , PB7:6 is used as TOSC2:1 input for the Asynchronous Timer/Counter2 if the AS2 bit in ASSR is set.

-Port c (PC5:PC0):

Port C is a 7-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). ThePort C output buffers have symmetrical drive characteristics with both high sink and sourcecapability. As inputs, Port C pins that are externally pulled low will source current if the pull-upresistors are activated. The Port C pins are tri-stated when a reset condition becomes active,even if the clock is not running.

-PC6/RESET:

If the RSTDISBL Fuse is programmed, PC6 is used as an I/O pin. Note that the electrical characteristics of PC6 differ from those of the other pins of Port C.

If the RSTDISBL Fuse is unprogrammed, PC6 is used as a Reset input. A low level on this pinfor longer than the minimum pulse length will generate a Reset, even if the clock is not running.

-Port D (PD7:PD0):

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). ThePort D output buffers have symmetrical drive characteristics with both high sink and sourcecapability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active,even if the clock is not running.

-RESET:

Reset input. A low level on this pin for longer than the minimum pulse length will generate areset, even if the clock is not running.

-AVCC:

AVCC is the supply voltage pin for the A/D Converter, Port C (3:0), and ADC (7:6). It should beexternally connected to VCC, even if the ADC is not used. If the ADC is used, it should be connected to VCC through a low-pass filter. Note that Port C (5:4) use digital supply voltage, VCC.

-AREF:

AREF is the analog reference pin for the A/D Converter.

-ADC7:6 (TQFP and QFN/MLNN Package only):

In the TQFP and QFN/MLF package, ADC7:6 serve as analog inputs to the A/D converter.
These pins are powered from the analog supply and serve as 10-bit ADC channels.

AppendixB

Programming the controller
AVR Studio C Code

```
#include <avr/io.h>
#include <string.h>

#define E   PD7
#define RS   PB0

void display(char string[16], char LineNo);
voiddisplaybyte(char D);
voiddispinit(void);
voidepulse(void);
voiddelay_ms(unsigned int de);

voidUSART_Transmit(char data );
voidsenddata(char string[16]);
voidUSART_Init();
voidUSART_Receive();
voidsendSMS();

charmystr[6];
intTemperature,setpoint,Smoke,SmokeSet;
unsigned char u8_data;
//=================================================================
//       Main Function
//=================================================================
int main(void)
{
Setpoint=60; //Temperature Limit For detection of Fire
 SmokeSet=100; //Smoke Set point
 Char Flag;

 DDRB = 0xF1; //Set LCD Port Direction
 DDRD = 0xE0;
 PORTB = 0x06; //Pull up for switches

Delay_ms(500);  //Initialize LCD
Dispinit();
delay_ms(200);
```

```
USART_Init();  //9600 Baud rate at internal oscillator Clock 1MHz

display("Temperature:32 C",1);
display("Smoke:55 PPM    ",2);
while(1)
  {
    //Measure Temperature and Display
    ADMUX=0xE5;
    ADCSRA=0xC7;  //Internal Reference 2.56V
while (!(ADCSRA & (1<<ADIF)));
    Temperature=ADCH;
    ADCSRA |= 1<<4;
Sprint (mystr, "%03d", Temperature);
display("Temperature:",1);
displaybyte(mystr[1]);
displaybyte(mystr[2]);
displaybyte(0xDF);
displaybyte('C');
displaybyte(0x20);

    ADMUX=0xE4;   //Smoke Sensor
    ADCSRA=0xC7;  //Internal Reference 2.56V
while (!(ADCSRA & (1<<ADIF)));
    Smoke=ADCH;   //Do some math here for calibration
    ADCSRA |= 1<<4;
sprintf(mystr, "%03d", Smoke);
display("Smoke:",2);
displaybyte(mystr[0]);
displaybyte(mystr[1]);
displaybyte(mystr[2]);
displaybyte('P');
displaybyte('P');
displaybyte('M');
displaybyte(0x20);

  //Compare with Set Points and Send SMS
if(Temperature>setpoint || Smoke>SmokeSet)
  {
   //Over Temperature SMS
if(Flag==0)
    {
SendSMS();
    Flag=1;
    PORTB &=~(1<<PB5);  //Turn of  Electrical Supply
    PORTB |=(1<<PB4);  //Turn on buzzer
    }
```

```
  }
else
  {
   Flag=0;
   PORTB |=(1<<PB5);   //Keep on Electrical Supply
   PORTB &=~(1<<PB4);   //Turn off buzzer
  }
  }
}

//=====================================================================
//       LCD Display Initialization Function
//=====================================================================
voiddispinit(void)
{
int count;
 char init[]={0x43,0x03,0x03,0x02,0x28,0x01,0x0C,0x06,0x02,0x02};

 PORTB &= ~(1<<RS);          // RS=0
for (count = 0; count <= 9; count++)
  {
displaybyte(init[count]);
  }
 PORTB |= 1<<RS;    //RS=1
}


//=====================================================================
//       Enable Pulse Function
//=====================================================================
voidepulse(void)
{
 PORTD |= 1<<E;
delay_ms(1); //Adjust delay if required
 PORTD &= ~(1<<E);
delay_ms(1); //Adjust delay if required
}


//=====================================================================
//       Send Single Byte to LCD Display Function
//=====================================================================
voiddisplaybyte(char D)
{
//D4=PD6
//D5=PD5
```

```
//D6=PB7
//D7=PB6
 //data is in Temp Register
char K1;
  K1=D;
  K1=K1 & 0xF0;
  K1=K1 >> 4;  //Send MSB

  PORTD &= 0x9F;  //Clear data pins
  PORTB &= 0x3F;

if((K1 & 0x01)==0x01){PORTD |= (1<<PD6);}
if((K1 & 0x02)==0x02){PORTD |= (1<<PD5);}
if((K1 & 0x04)==0x04){PORTB |= (1<<PB7);}
if((K1 & 0x08)==0x08){PORTB |= (1<<PB6);}

epulse();

  K1=D;
  K1=K1 & 0x0F;  //Send LSB
  PORTD &= 0x9F;  //Clear data pins
  PORTB &= 0x3F;

if((K1 & 0x01)==0x01){PORTD |= (1<<PD6);}
if((K1 & 0x02)==0x02){PORTD |= (1<<PD5);}
if((K1 & 0x04)==0x04){PORTB |= (1<<PB7);}
if((K1 & 0x08)==0x08){PORTB |= (1<<PB6);}
epulse();
}

//=================================================================
//       Display Line on LCD at desired location Function
//=================================================================
void display(char string[16], char LineNo)
{
intlen,count;

 PORTB &= ~(1<<RS);          // RS=0 Command Mode

if(LineNo==1)
 {
displaybyte(0x80);  //Move Coursor to Line 1
 }
else
 {
displaybyte(0xC0);  //Move Coursor to Line 2
```

```
 }
 PORTB |= (1<<RS);          // RS=1 Data Mode



len = strlen(string);

for (count=0;count<len;count++)
  {
displaybyte(string[count]);
 }
}


//=====================================================================
//       Delay Function
//=====================================================================
voiddelay_ms(unsigned int de)
{
unsignedint rr,rr1;
for (rr=0;rr<de;rr++)
   {

for(rr1=0;rr1<30;rr1++)   //395
  {
asm("nop");
  }

   }
}


voidUSART_Transmit(char data )
{
 UDR = data;
 /* Wait for empty transmit buffer */
while ( !( UCSRA & (1<<UDRE)) )
 ;
 /* Put data into buffer, sends the data */

}


voidsenddata(char string[16])
{
intlen,count;
```

```
len = strlen(string);

for (count=0;count<len;count++)
  {
USART_Transmit(string[count]);
 }
}

voidUSART_Init()
{
/* Set baud rate */
 UBRRH = 0x00;  //12, 9600 Baud At 1MHz
 UBRRL =12;
//Set double speed enabled
  UCSRA |= (1<<U2X);
/* Enable receiver and transmitter */
 UCSRB = (1<<RXEN)|(1<<TXEN);
/* Set frame format: 8data, 2stop bit */
 UCSRC = (1<<URSEL)|(1<<USBS)|(3<<UCSZ0);
//Set interrupt on RX
//  UCSRB |= (1<<RXCIE);
}

voidUSART_Receive()
{
/* Wait for data to be received */
while ( !(UCSRA & (1<<RXC)) )
;
/* Get and return received data from buffer */
u8_data=UDR;
}

voidsendSMS()
{
senddata("AT+CMGD=1");
USART_Transmit(13);
USART_Transmit(10);
delay_ms(1000);

senddata("AT+CMGF=1");
    USART Transmit(13);
USART_Transmit(10);
delay_ms(1000);

senddata("AT+CMGW=");
USART_Transmit(34);
```

```
senddata("+919812345678");  //Enter Your Mobile number
USART_Transmit(34);
USART_Transmit(13);
USART_Transmit(10);
delay_ms(1000);

senddata("Alert: Fire Detected");
USART_Transmit(13);
USART_Transmit(10);
delay_ms(1000);
senddata("Temperature:");

    ADMUX=0xE5;
    ADCSRA=0xC7;  //Internal Referance 2.56V
while (!(ADCSRA & (1<<ADIF)));
    Temperature=ADCH;
    ADCSRA |= 1<<4;

sprintf(mystr, "%03d", Temperature);
USART_Transmit(mystr[1]);
USART_Transmit(mystr[2]);
USART_Transmit('C');
USART_Transmit(13);
USART_Transmit(10);

senddata("Smoke Level:");
    ADMUX=0xE4;   //Smoke Sensor
    ADCSRA=0xC7;  //Internal Referance 2.56V
while (!(ADCSRA & (1<<ADIF)));
    Smoke=ADCH;   //Do some math here for calibration
    ADCSRA |= 1<<4;
sprintf(mystr, "%03d", Smoke);
USART_Transmit(mystr[0]);
USART_Transmit(mystr[1]);
USART_Transmit(mystr[2]);
USART_Transmit('P');
USART_Transmit('P');
USART_Transmit('M');
USART_Transmit(13);
USART_Transmit(10);


delay_ms(1000);
USART_Transmit(26); //Cntrl+Z
delay_ms(1000);
delay_ms(1000);
```

```
senddata("AT+CMSS=1");
USART_Transmit(13);
USART_Transmit(10);
delay_ms(1000);
}
```