بسم الله الرحمن الرحيم

Sudan University of Science and Technology

College of Graduate Studies

Design and Implementation of a Fire Fighting System Using Microcontroller

تصميم وتنفيذ نظام مكافحة حرائق باستخدام المتحكم الدقيق

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of M.Sc. in Mechatronics Engineering

Prepared by:

Elrasheed Abozaid Elrhaima Elamin

Supervised by:

Dr. Awadalla Taifour Ali Ismail

July 2019

الآيـــة

قال تعالى:

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

لَّيْسَ الْبِرَّ أَن تُوَلُّوا وُجُوهَكُمْ قِبَلَ الْمَشْرِقِ وَالْمَغْرِبِ وَلَٰكِنَّ الْبِرَّ مَنْ آمَنَ بِاللَّهِ وَالْيَوْمِ الْآخِرِ وَالْمَلَائِكَةِ وَالْكِتَابِ وَالنَّبِيِّينَ وَآتَى الْمَالَ عَلَىٰ حُبِّهِ ذَوِي الْقُرْبَىٰ وَالْيَتَامَىٰ وَالْمَسَاكِينَ وَابْنَ السَّبِيلِ وَالسَّائِلِينَ وَفِي الرِّقَابِ وَأَقَامَ الصَّلَاةَ وَآتَى الزَّكَاةَ وَالْمُوفُونَ بِعَهْدِهِمْ إِذَا عَاهَدُوا وَالصَّابِرِينَ فِي الْبَأْسَاءِ وَالضَّرَّاءِ وَحِينَ الْبَأْسِ ۗ أُولَٰئِكَ الَّذِينَ صَدَقُوا وَأُولَٰئِكَ هُمُ الْمُتَّقُونَ (177)

صدق الله العظيم

سورة البقرة – الاية 177

Dedication

I would like to dedicate this work to my mother, father and the rest of my family also to all who helped me..

Thank you all...

Acknowledgment

Great thankful to Allah who lightens our ways, our life and our heart.

Great thankful and appreciation to my supervisor Dr. Awadalla Taifour For his support, guidance throughout this research. Finally, I would like to thank all who support me in this research.

Abstract

This study aims to design and implementation a fire fighting system using Atmega32 microcontroller. This system accomplished by connecting Atmega32 microcontroller to temperature sensors, push buttons, Liquid Crystal Display (LCD) and other components. The system monitors the sensors values continuously to warn or inform the user in abnormal situations or at fire danger. This warning takes different ways such as the appearance of the threat type on the system screen or the operation of different alarm sirens or by sending a Short Message Service (SMS), at some cases the system sends command to fire fighting unit to extinguish fire. The system also has many options that achieve more functions.

مستخلص

يهدف هذا البحث الى دراسة وتصميم نظام مكافحة حرائق باستخدام المتحكم الدقيق. تم تصميم هذا النظام بتوصيل المتحكم الدقيق Atmega32 الى مستشعرات حرارة وازرار تحكم وشاشة عرض بالاضافة الى مكونات اخرى، يقوم النظام بمراقبة مستشعرات الحرارة باستمرار بهدف تحذير المستخدم في الحالات غير الطبيعية أو عند خطر الحرائق حيث يأخذ هذا التحذير طرق مختلفة كظهور نوع التهديد على شاشة النظام او تشغيل سارينات الانذار المختلفة او عبر ارسال رسالة نصية قصيرة الى هاتف المستخدم وفي بعض الحالات يقوم النظام باصدار اوامر مناسبة الى وحدة مكافحة حرائق لاطفاء الحريق . النظام ايضاً مزود بقائمة خيارات لتحقيق وظائف أكثر.

Table of Contents

List of Figures

Chapter One

Introduction

1.1 General Concept

Fire accident is common feature in factories, houses, markets etc. in every country. Due to poor fire protection, lack of adequate fire alarm and emergency exit, fire increases death. With the advancement of human civilization, fire-safety has been a prime concern. Fire hazards can be fatal and denigrating for industrial and household security, also minatory for human life.

The best way to reduce these losses is to respond to the emergency situation as quick as possible. So, there comes the necessity of standalone autonomous fire detection systems. These systems render the works of quick detection, alarm notification, and sometimes initiation of fire extinguishing. A fire or smoke alarm system can be monitored locally in the premises, or remotely at a distant place as prerequirement. Remote alarm system provides the owner of the premise with the advantage of monitoring from distant location and taking immediate actions when an emergency message is received, unlike a manual system. Remote monitoring systems can be designed in various ways-using wireless sensor networks, Ethernet, image processing and other digital communication technologies.

Fire, smoke or the leak of gases threat people's life and their properties. Most building and facilities in Sudan free from fire alarm systems. Firefighting system is used to prevent, extinguish, localize, or block fires in enclosed spaces. Automatic fire-fighting system, which is very important to protect people and properties, is installed in buildings and rooms where the fire hazard is comparatively high.

1.2 Statement of Problem

Although The Existing Fire fighting Systems are reliable and have a wide range of pros, they are accompanied by concerns about being complex, incompact, non-standalone, expensive and having redundant appurtenances.

Therefore, there is necessity for a system which would be reliable and swift responsive as well as simple, easy implementable and cost effective.

1.3 Objectives

The main objectives of this study are to:

1. Design a control circuit for automatic fire fighting system.
2. Simulate the proposed automatic fire fighting system control circuit.
3. Implement the proposed automatic fire fighting system control circuit.

1.4 Methodology

This study develops a microcontroller based system to detect smoke and heat at a house or facilities, by analyzing this signal and take specific action. The system can operate at auto or manual mode in which the system acts as detector only and the owner can activate the actuators remotely as needed.

In this study the system is divided into parts; hardware and software. The hardware a sensor circuitry is designed to develop system awareness and capability to detect over-temperature, smoke and flame. Microcontroller Atmega16 is used to process the various sensor signals and control the system actuators accordingly. A software code is developed to control the overall system functions. The code is written in BASIC language using basic compiler for AVR (BASCOM-AVR). A Global System Mobile (GSM) module is interconnected to the system and used to send SMS indicating the system and environment status. System software simulation is done using Proteus package.

1.5 Layout

This research consists of five chapters, Chapter one contains general introduction, defines problem, objectives and methodology. Chapter two contains a general background and general information to the main parts used in the study. Chapter three handles the hardware and software considerations. Chapter four deals with the system implementation and testing. Finally, chapter five presents the conclusion and recommendations.

Chapter Two

Theoretical Background

2.1 Introduction

All fire alarm systems essentially operate on the same principle. 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. When choosing a fire alarm system many factors need to be considered such as the purpose, building structure and current legislation.

Fire safety in new, extended or altered buildings is the responsibility of the local authority who delegates the responsibility to building control department who enforces and administers the building regulations. In existing commercial buildings, premises are subject to the regulatory reform (fire safety) order 2005. The order states that the 'responsible person' within the organization has a legal duty to undertake a suitable and sufficient fire risk assessment at their premises. This assessment is a detailed report of the fire risk in your building, the adequacy of existing fire precautions and the need for any additional fire precautions. The fire risk Assessment will determine the level of fire alarm system required for business premises. The guidance usually indicates appropriate British standards. The main standard for fire alarm systems is BS5839 pt1:2013[1].

2.2 Sensors

A sensor is generally defined as an input device that provides a usable output in response to a specific physical quantity input. The physical quantity input that is to be measured, called the measurand, affects the sensor in a way that causes a response represented in the output. The output of many modern sensors is an electrical signal, but alternatively, could be a motion, pressure, flow, or other usable type of output. Some examples of sensors include a thermocouple pair, which converts a temperature difference into an electrical

output; a pressure sensing diaphragm, which converts a fluid pressure into a force or position change; and a Linear Variable Differential Transformer (LVDT), which converts a position into an electrical output [2].

The devices that inform the control system about what is actually occurring are called sensors (also known as transducers). As an example, the human body has an amazing sensor system that continually presents our brain with a reasonably complete picture of the environment whether we need it all or not. For a control system, the designer must ascertain exactly what parameters need to be monitored. For example, position, temperature, and pressure and then specify the sensors and data interface circuitry to do the job[3].

Classification of sensors is conventionally by the convention principle, the quantity being measured, the technology used, or the application [4]. Most sensors work by converting some physical parameter such as temperature or position into an electrical signal. This is why sensors are also called transducers, which are devices that convert energy from one form to another. The various types of sensors are the position sensors, angular velocity sensors, proximity sensors, load sensors, pressure sensors, flow Sensors, liquid-level sensors, vision sensors, temperature sensors, etc [3].

2.2.1 Smart and intelligent sensors

Modern definition of smart or intelligent sensors can be formulated now by the following way: Smart sensor is an electronic device, including sensing element, interfacing, signal processing and one or several intelligence functions as self-testing, self-identification, self-validation or self-adaptation. The key word in this definition is intelligence. The self-adaptation is relatively new function of smart sensors. Novel designed self-adaptation smart sensor systems are based on so-called adaptive algorithms, which were used at the first time in various digital measuring systems [5].

Strong growth expected for sensors based on Microelectromechanical system technologies (MEMS), smart sensors and sensors with bus capabilities.

Smart sensors' capability to have more intelligence built into them continues to drive their application in automotive, aerospace and defense, industrial, medical, and most recently – homeland security applications. Proprietary algorithms customized for specific applications analyze sensor data on key parameters to optimize machining, processing, and other component product or process quality [5]. Intelligent wireless sensor-based controls have drawn industry attention on account of reduced costs, better power management, ease in maintenance, and effortless deployment in remote and hard-to-reach areas. They have been successfully deployed in many industrial applications such as maintenance, monitoring, control, security, etc [6].

Because of semiconductor technologies and its Moore's law along with the MEMS technology, sensors are becoming more cost effective, smaller, less power hungry, with more embedded features. The sensors not only provide digital data but can also provide the expected information. For instance, functions like portrait/landscape, specific tap and free fall detections can be left to the sensor due to its capability to detect such simple information with values preset by the device manufacturer [7].

2.2.2 Temperature sensors

Temperature is defined as a specific degree of hotness or coldness as referenced to a specific scale. It can also be defined as the amount of heat energy in an object or system. Heat energy is directly related to molecular energy (vibration, friction and oscillation of particles within a molecule): the higher the heat energy, the greater the molecular energy. Temperature sensors detect a change in a physical parameter such as resistance or output voltage that corresponds to a temperature change [8].

Temperature sensors give an output proportional to temperature. Most temperature sensors have a positive temperature coefficient (desirable), which means that the sensor output goes up as the temperature goes up, but some sensors have a negative temperature coefficient, which means that the output goes down as the temperature goes up. Many control systems require

temperature sensors, if only to know how much to compensate other sensors that are temperature-dependent. Some common types are discussed next.

- Bimetallic temperature sensors: The bimetallic temperature sensor consists of a bimetallic strip wound into a spiral. The bimetallic strip is a laminate of two metals with different coefficients of thermal expansion. As the temperature rises, the metal on the inside expands more than the metal on the outside, and the spiral tends to straighten out. These sensors are typically used for on-off control as in a household thermostat where a mercury switch is rocked from on to off.

- Thermocouples: The thermocouple was developed over 100 years ago and still enjoys wide use, particularly in high-temperature situations. The thermocouple is based on the See beck effect, a phenomenon whereby a voltage that is proportional to temperature can be produced from a circuit consisting of two dissimilar metal wires.

- Resistance temperature detectors: The Resistance Temperature Detector (RTD) is a temperature sensor based on the fact that metals increase in resistance as temperature rises. RTDs have the advantage of being very accurate and stable (characteristics do not change over time). The disadvantages are low sensitivity (small change in resistance per degree), relatively slow response time to temperature changes, and high cost.

- Thermistors: A thermistor is a two-terminal device that changes resistance with temperature. Thermistors are made of oxide-based semiconductor materials and come in a variety of sizes and shapes. Thermistors are nonlinear; therefore, they are not usually used to get an accurate temperature reading but to indicate temperature changes, for example, overheating. Also, most thermistors have a negative temperature coefficient, which means the resistance decreases as temperature increases.

- Integrated-circuit temperature sensors: Integrated circuit (IC) temperature sensors come in various configurations. A common example is the LM34 and LM35 series. The LM34 produces an output voltage that is

proportional to Fahrenheit temperature, and the LM35 produces an output that is proportional to Celsius temperature. Notice that it has three active terminals: supply voltage (Vs), ground, and Vout. The output voltage of the LM35 is directly proportional to °C, that is, Vout = 10mV/°C This equation states that for each 1° increase in temperature, the output voltage increases by 10mV. If only positive temperatures need to be measured, then the simple circuit shown in the spec sheet (bottom middle of datasheet) can be used. If positive and negative temperatures must be measured, then the circuit on the bottom right can be used, which requires a positive and negative supply voltage. The LM35 is a convenient IC to work with because the output voltage is in degrees Celsius. Some ICs, such as the LM135, provide an output that is in degrees kelvin. One degree of kelvin or Celsius represents the same interval of temperature, but the Kelvin scale starts at absolute zero temperature, which is 273°C below freezing. There is also an absolute zero temperature scale for Fahrenheit degrees, called the Rankine scale.

2.2.3 Smoke detectors

Smoke detectors operate on two main principles: ionization detectors which will detect the ionized air from a fire even when there is little or no visible smoke, and the optical type, which is design to detect smoke which is present even when there is little or no rise in temperature. In general, the ionization types are used industrially and the optical types in domestic uses. The ionization type of detector uses a radioactive source, usually americium241, with a low activity level, typically 0.8μCi [4].

The gap between the source plate and the electrode is normally conducting due to the emission of alpha particles (helium ions) by the americium. The ion current is very small, of the order of 10pA, so that any additional insulation leakage would be substantial in comparison. The acceptable level of insulation leakage is of the order of 0.5pA. This implies that the insulators must not be touched, and if the ionization chamber has to be replaced, utmost care must be taken to avoid any contamination of the

insulation by, for example, solder flux. In the presence of smoke from a fire, particles entering the ionization chamber will be struck by the alpha particles, and the alpha particles will cling to the much larger particles of smoke. Because the charged units are now much larger, they cannot travel so quickly in air, and the current is reduced. By detecting the reduction in current, the detector can be made to activate an alarm.

- Ionization smoke sensor: Ionization smoke sensor contains a small amount of radioactive material, americium embedded in a gold foil matrix within an ionization chamber. The matrix is made by rolling gold and americium oxide ingots together to form a foil approximately one micrometer thick. This thin gold-americium foil is then sandwiched between a thicker (~0.25 millimeter) silver backing and a 2 micron thick palladium laminate. This is thick enough to completely retain the radioactive material, but thin enough to allow the alpha particles to pass.

The ionization chamber as shown in Figure 2.1 is basically two metal plates a small distance apart. One of the plates carries a positive charge, the other a negative charge. Between the two plates, air molecules-made up mostly of oxygen and nitrogen atoms-are ionized when electrons are kicked out of the molecules by alpha particles from the radioactive material (alpha particles are big and heavy compared to electrons). The result is oxygen and nitrogen atoms that are positively charged because they are short one electron; the free electrons are negatively charged.

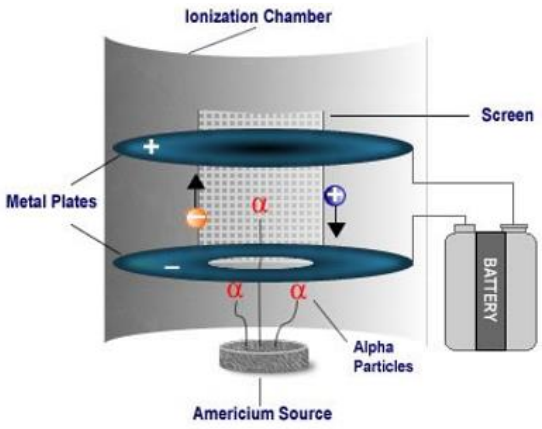

Figure 2.1: The ionization chamber

The Figure 2.2 illustrates how ionization technology works. The positive atoms flow toward the negative plate, as the negative electrons flow toward the positive plate. The movement of the electrons registers as a small but steady flow of current. When smoke enters the ionization chamber, the current is disrupted as the smoke particles attach to the charged ions and restore them to a neutral electrical state. This reduces the flow of electricity between the two plates in the ionization chamber. When the electric current drops below a certain threshold, the sensor sends a signal telling there is a smoke.

In the smoke-free chamber, positive and negative ions create a small current as they migrate to charged plates. Smoke particles and combustion gases interact with the ions generated by the alpha particles, restoring them to their neutral electronic state and decreasing the electrical current passing through the cell Figure 2.2. As fewer ions are available to migrate to the plates, the disrupted current causes the smoke sensor triggers the microcontroller [9].

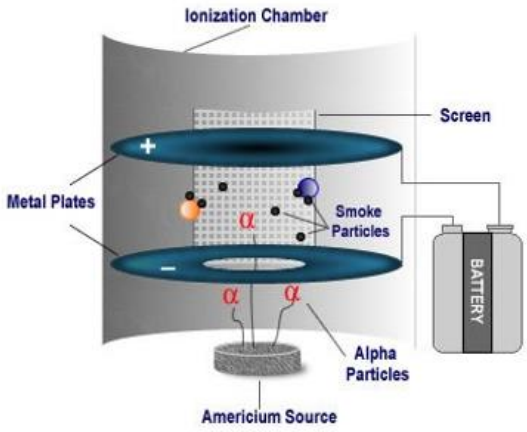

Figure 2.2: Smoke detector inside ionization chamber

2.2.4 Infrared Sensor

All objects emit infrared energy provided their temperature is above absolute zero (0 Kelvin). There is a direct correlation between the infrared energy an object emits and its temperature. Infrared Sensor (IR) measure the infrared energy emitted from an object in the 4–20 micron wavelength and convert the reading to a voltage. Typical IR technology uses a lens to concentrate radiated energy onto a thermopile. The resulting voltage output is amplified and conditioned to provide a temperature reading. Factors that affect

the accuracy of IR sensing are the reflectivity (the measure of a material's ability to reflect infrared energy), transitivity (the measure of a material's ability to transmit or pass infrared energy), and emissivity (the ratio of the energy radiated by an object to the energy radiated by a perfect radiator of the surface being measured). An object that has an emissivity of 0.0 is a perfect reflector, while an object with an emissivity of 1.0 emits (or absorbs) 100% of the infrared energy applied to it [8].

2.3 Microprocessor

The digital IC called a microprocessor, has ushered in a whole new era for control systems electronics. This revolution has occurred because the microprocessor brings the flexibility of program control and the computational power of a computer to bear on any problem. Automatic control applications are particularly well suited to take advantage of this technology, and microprocessor-based control systems are rapidly replacing many older control systems based on analog circuits or electromechanical relays. One of the first microprocessor-based controllers made specifically for control applications was the Programmable Logic Controller (PLC). A microprocessor by itself is not a computer; additional components such as memory and input/output circuits are required to make it operational. However, the microcontroller, which is a close relative of the microprocessor, does contain all the computer functions on a single IC. Microcontrollers lack some of the power and speed of the newer microprocessors, but their compactness is ideal for many control applications; most so-called microprocessor-controlled devices, such as vending machines, are really using microcontrollers. Some specific reasons for using a digital, microprocessor design in control systems are the following:

- Low-level signals from sensors, once converted to digital, can be transmitted long distances virtually error-free.
- A microprocessor can easily handle complex calculations and control strategies.

- Long-term memory is available to keep track of parameters in slow-moving systems.

Changing the control strategy is easy by loading in a new program; no hardware changes are required. Microprocessor-based controllers are more easily connected to the computer network within an organization. This allows designers to enter program changes and read current system status from their desk terminals. To perform all microprocessor functions, the microprocessor incorporates various functional units in an appropriate manner. Such an internal structure or organizational structure of microprocessor, which determines how it operates, is known as its architecture. Typical microprocessor architecture is shown in Figure 2.3.

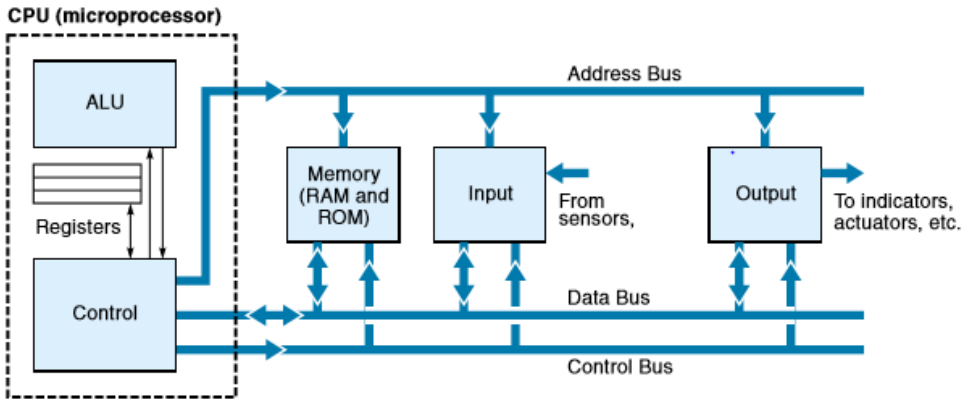

Figure 2.3: Architecture of Microprocessor

2.3.1 Busses

Microcomputer, like all computers, manipulates binary information. The binary information is represented by binary digits, called bits. Microprocessor operates on a group of bits which are referred to as a word. The number of bits making- microprocessor a word varies with the microprocessor. Common word sizes are 4, 8, 12 and 16 bits (microprocessors with 32 bit-word have also of late entered the market). Other binary terms that will be of interest in subsequent discussions are the byte and the nibble, which represent a set of 8 bits and 4 bits, respectively. Figure 2.3 shows busses interconnecting various blocks. These busses allow exchange of words between the blocks. A bus has a wire or line for each bit and thus allows exchange of all bits of a word in

parallel. The processing of bits in the microprocessor is also in parallel. The busses can thus be viewed as data highways. The width of a bus is the number of signal lines that constitute the bus.

The figure shows for simplicity three busses for distinct functions. Over the address bus, the microprocessor transmits the address of that Input Output (I/O) device or memory locations which it desires to access. This address is received by all the devices connected to the processor, but only the device which has been addressed responds. The data bus is used by the microprocessor to send and receive data to and from different devices (I/O and memory) including instructions stored in memory. Obviously the address bus is unidirectional and the data bus is bi-directional. The control bus is used for transmitting and receiving control signals between the microprocessor and various devices in the system.

2.3.2 Arithmetic Logic Unit

The Arithmetic Logic Unit (ALU) is a combinational network that performs arithmetic and logical operations on the data.

2.3.3 Internal registers

A number of registers are normally included in the microprocessor. These are used for temporary storage of data, instructions and addresses during execution of a program. Those in the Intel microprocessor 8085 microprocessor are typical and are described below:

(i) Accumulator (Acc) or result register: This is an 8-bit register used in various arithmetic and logical operations. Out of the two operands to be operated upon, one comes from accumulator (Acc), whilst the other one may be in another internal register or may be brought in by the data bus from the main memory. Upon completion of the arithmetic/logical operation, the result is placed in the accumulator (replacing the earlier operand). Because of the later function, this register is also called as result register.

(ii) General purpose registers or scratch pad memory: There are six general purpose 8-bit registers that can be used by the programmer for a variety of

purposes. These registers, labelled as B, C, D, E, H and L, can be used individually (e.g.,when operation on 8-bit data is desired) or in pairs (e.g., when a 16-bit address is to be stored). Only B-C, D-E and H-L pairs are allowed.

(iii) Instruction Register (IR): This 8-bit register stores the next instruction to be executed. At the proper time this stored word (instruction) is fed to an instruction decoder which decodes it and supplied appropriate signals to the control unit. When the execution has been accomplished the new word in the instruction register is processed.

(iv) Program Counter (PC): This is a 16-bit register which holds the address of the next instruction that has to be fetched from the main memory and loaded into the instruction register. The program controlling the operation is stored in the main memory and instructions are retrieved from this memory normally in order. Therefore, normally the address contained in the PC is incremented after each instruction is fetched. However, certain classes of instruction can modify the PC so that the programmer can provide for branching away from the normal program flow. Examples are instructions in the "jump" and 'call subroutine' groups.

(v) Stack Pointer (SP): This is also a 16-bit register and is used by the programmer to maintain a stack in the memory while using subroutines.

(vi) Status register or condition flags: A status register consisting of a few flip-flops, called as condition flags (in 8085 the number of flags is five) is used to provide indication of certain conditions that arise during arithmetic and logical operations. These are:

'zero' Flag is set if result of instruction is 0.

'sign' Set if Most Significant Bit (MSB) of result is 1.

'parity' Set if result has even parity.

'carry' Set if carry or borrow resulted.

'auxiliary carry' Set if instruction caused a carry out of bit 3 and into bit 4 of the resulting value.

(vii) Dedicated registers: Several other registers are incorporated in the microprocessor for its internal operation. They cannot be accessed by the programmer and hence do not concern much a microprocessor user.

2.3.4 Instruction decoder and control unit

It decodes each instruction and under the supervision of a clock controls the external and internal units ensuring correct logical operation of the system[10].

2.4 Microcontroller

A microcontroller is a single-chip computer. Micro suggests that the device is small, and controller suggests that it is used in control applications. Another term for microcontroller is embedded controller, since most of the microcontrollers are built into or embedded in the devices they control [11].
A microcontroller is a computer with most of the necessary support chips onboard. All computers have several things in common, namely:

1- A Central Processing Unit (CPU) that 'executes' programs.
2- Some Random Access Memory (RAM) where it can store data that is variable.
3- Some Read Only Memory (ROM) where programs to be executed can be stored.
4- I/O devices that enable communication to be established with the outside world i.e. connection to devices such as keyboard, mouse, monitors and other peripherals.

There are a number of other common characteristics that define microcontrollers. If a computer matches a majority of these characteristics, then it can be classified as a 'microcontroller'.

2.4.1 Microcontroller features

Microcontrollers from different manufacturers have different architectures and different capabilities. Some may suit to a particular application while some others may be totally unsuitable. The hardware features of microcontrollers in general are described below.

- Supply voltage: Most microcontrollers operate with the standard 5V supply. Some microcontrollers can operate at as low as 2.7V and some will tolerate 6V without any problems. the allowed limits of the supply voltage in the datasheet.

- The clock: All microcontrollers require an oscillator (known as a clock) to operate. Most microcontrollers will operate with a crystal and two capacitors. Some will operate with resonators or with external resistor-capacitor pair. Some microcontrollers have built-in resistor-capacitor type oscillators and they do not require any external timing components (e.g. PIC12C672). Resonators are not as stable as the crystals but they are more stable than the resistor-capacitor networks. Crystal oscillators should be chosen for applications which require very accurate timing. For applications where the timing stability requirements are very modest, resonators should be chosen. If the application is not time sensitive you should consider using external or internal (if available) resistor-capacitor timing components for simplicity and low cost.

- Timers: Timers are an important part of any microcontroller. A timer is basically a counter which is driven from an accurate clock (or a division of this clock). Timers can be 8-bits or 16-bits long. Data can be loaded into the timers and they can be started and stopped under software control. Most timers can be configured to generate an interrupt when they reach a certain count (usually when they overflow).

- Watchdog: Many microcontrollers have at least one watchdog facility, also known as the Watch Dog Timer (or WDT). A WDT is usually an 8-bit timer with a prescaler option and is clocked from a free running on-chip oscillator. The watchdog is usually refreshed by the user program at regular intervals and a reset occurs ifthe program fails to refresh the watchdog. Watchdog facilities are commonly used in real-time systems where it is required to check the proper termination of one or more activities. All PIC microcontrollers are equipped with a WDT.

- Interrupts: Interrupts are a very important concept in microcontrollers. An interrupt causes a microcontroller to respond to external and internal (e.g. timer) events very quickly. When an interrupt occurs the microcontroller leaves its normal flow of execution and jumps directly to the Interrupt Service Routine (ISR). The source of an interrupt can either be internal or external. Internal interrupts are usually generated by the built-in timer circuits when the timer count reaches a certain value. External interrupts are generated by the devices connected external to the microcontroller and these interrupts are asynchronous, i.e. it is not known when an external interrupt will be generated. An example is the Analogue-to-Digital (A/D) conversion complete interrupt, which is generated when a conversion is completed.

- Analogue-to-digital converter: Some microcontrollers are equipped with A/D converter circuits. Usually these converters are 8-bits, but some microcontrollers have 10- or even 12-bit converters. Some microcontrollers have multiple A/D channels (e.g. PIC16F877 is equipped with eight A/D channels). A/D converters usually generate interrupts when a conversion is complete so that the user program can read the converted data very quickly. A/D converters are very useful in control and monitoring applications since most sensors produce analogue output voltages.

- Serial input-output: Some microcontrollers contain hardware to implement serial asynchronous communications interface. The baud rate and the data format can usually be selected in software by the programmer. The built-in timer circuits are usually used to generate an accurate baud rate. If serial I/O hardware is not provided, it is easy to develop software to implement serial data transfer using any I/O pin of a microcontroller. Some microcontrollers incorporate Serial Peripheral Interface (SPI), Integrated Interconnect (I_2C), or Controller Area Network (CAN) bus interfaces. These enable a microcontroller to interface to other compatible devices easily, usually over a bus structure.

- In-circuit programming: In microcontroller development cycle, a microcontroller is normally removed from its socket and then programmed

using a programmer device. The programmed chip is then re-inserted into its socket, ready for testing. This is usually very tedious work, especially during the development of complex software projects. In-circuit programming enables a microcontroller to be programmed while the chip is in the applications circuit, i.e. there is no need to remove the chip for programming. This feature speeds up the program development cycle considerably.

➢Electrically Erasable Programmable ROM (EEPROM): Data memory EEPROM type memory is also very common in many microcontrollers. The programmer can store non-volatile data in such memory and can also change this data whenever required. For example, if the microcontroller is used to measure the temperature, the maximum and minimum values during a period can be stored in an EEPROM type memory. Some microcontroller types provide between 64 and 256 bytes of EEPROM data memories, while some others do not have any such memories.

➢ Pulse Width Modulation (PWM) output: Some microcontrollers provide PWM outputs which can be used in some electronic applications. One such application is to provide an effective analog output from a microcontroller by varying the duty cycle of the PWM output. It is possible to modify the period or the duty cycle of a PWM output by loading the appropriate registers [12].

2.4.2 Microcontroller architectures

Basically, two types of architectures are used in microcontrollers: Von Neumann architecture and Harvard architecture. Von Neumann architecture is used by a very large percentage of microcontrollers and here all memory space is on the same bus and instruction and data are treated identically. In the Harvard architecture (used by the PIC microcontrollers), code and data storage are on separate buses and this allows code and data to be fetched simultaneously, resulting in a more efficient implementation.

2.4.3 Microcontroller types

The predominant families of microcontrollers are 8-bit types since this word size has proved popular for the vast majority of tasks the devices have

been required to perform. The single byte word is regarded as sufficient for most purposes and has the advantage of easily interfacing with the variety of IC memories and logic circuitry currently available. The serial acronym for the American Standard Code for Information Interchange (ASCII) data is also byte sized making data communications easily compatible with the microcontroller devices. Because the type of application for the microcontroller may vary enormously most manufacturers provide a family of devices, each member of the family capable of fitting neatly into the manufacturer's requirements. This avoids the use of a common device for all applications where some elements of the device would not be used; such a device would be complex and hence expensive.

The microcontroller family would have a common instruction subset but family members differ in the amount, and type, of memory, timer facility, port options, etc. possessed, thus producing cost-effective devices suitable for particular manufacturing requirements. Memory expansion is possible with offchip RAM and/or ROM; for some family members there is no on-chip ROM, or the ROM is either Electrically Programmable ROM (EPROM) or EEPROM known as flash EEPROM which allows for the program to be erased and rewritten many times. Additional on-chip facilities could include Analogue to Digital Conversion (ADC), Digital-to-Analogue Conversion (DAC) and analogue comparators. Some family members include versions with lower pin count for more basic applications to minimize costs [12].

2.4.4 Comparison between microcontroller and microprocessor

Microcontrollers are general purpose microprocessors which have additional parts that allow them to control external devices. Basically, a microcontroller executes a user program which is loaded in its program memory. Under the control of this program, data is received from external devices (inputs), manipulated and then data is sent to external output devices. A microcontroller is a very powerful tool that allows a designer to create sophisticated I/O data manipulation algorithms.

Microcontrollers are classified by the number of bits in a data word. 8-bit microcontrollers are the most popular ones and are used in many applications. 16-bit and 32-bit microcontrollers are much more powerful, but usually more expensive and not required in many small to medium general purpose applications where microcontrollers are used. The simplest microcontroller architecture consists of a microprocessor, memory, and I/O. The microprocessor consists of a central CPU and the Control Unit (CU). The CPU is the brain of a microprocessor and is where all of the arithmetic and logical operations are performed. The control unit controls the internal operations of the microprocessor and sends out control signals to other parts of the microprocessor to carry out the required instructions[13].

A microcontroller may take an input from the device it is controlling and controls the device by sending signals to different components in the device. A microcontroller is often small and low cost. The components may be chosen to minimize size and to be as inexpensive as possible. The actual processor used to implement a microcontroller can vary widely. In many products, such as microwave ovens, the demand on the CPU is fairly low and price is an important consideration. In these cases, manufacturers turn to dedicated microcontroller chips – devices that were originally designed to be low-cost, small, low-power, embedded CPUs [13]. A microprocessor differs from a microcontroller in a number of ways. The main distinction is that a microprocessor requires several other components for its operation, such as program memory and data memory, input-output devices, and an external clock circuit. A microcontroller, on the other hand, has all the support chips incorporated inside its single chip. All microcontrollers operate on a set of instructions (or the user program) 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. Although the assembly language is fast, it has several disadvantages.

An assembly program consists of mnemonics, which makes learning and maintaining a program written using the assembly language difficult. Also, microcontrollers manufactured by different firms have different assembly languages, so the user must learn a new language with every new microcontroller he or she uses. Microcontrollers can also be programmed using a high-level language, such as BASIC, PASCAL, or C. High-level languages are much easier to learn than assembly languages.

In theory, a single chip is sufficient to have a running microcontroller system. In practical applications, however, additional components may be required so the microcomputer can interface with its environment. With the advent of the PIC family of microcontrollers the development time of an electronic project has been reduced to several hours. Basically, a microcomputer executes a user program which is loaded in its program memory. Under the control of this program, data is received from external devices (inputs), manipulated, and then sent to external devices (outputs). A microcontroller is a very powerful tool that allows a designer to create sophisticated input-output data manipulation under program control. Microcontrollers are classified by the number of bits they process. Microcontrollers with 8-bits are the most popular and are used in most microcontroller-based applications. Microcontrollers with 16 and 32 bits are much more powerful, but are usually more expensive and not required in most small- or medium-size general purpose applications that call for microcontrollers. Memory, an important part of a microcontroller system, can be classified into two types: program memory and data memory. Program memory stores the program written by the programmer and is usually nonvolatile (i.e., data is not lost after the power is turned off). Data memory stores the temporary data used in a program and is usually volatile [11].

2.5 Buzzer

For alarm purposes a lot of electric bells, alarms and buzzers are available in the markets that have got different prices and uses. The buzzer that

will use in this thesis is a 5-12V buzzer and has got enough alarm sound to be used in a fire alarm system. Louder buzzer would have been even better but then their operating voltages are high as we had a supply of maximum up to 12V available with us on the board.

2.6 GSM System

A GSM modem is a wireless modem that works with a GSM wireless network. A wireless modem behaves like a dial-up modem. The main difference between them is that a dial-up modem sends and receives data through a fixed telephone line while a wireless modem sends and receives data through radio waves. Like a GSM mobile phone, a GSM modem requires a SIM card from a wireless carrier in order to operate. GSM modem provides full functional capability to serial devices to send SMS and data over GSM network. The product is available as board level or enclosed in metal box. The board level product can be integrated in to various serial devices in providing them SMS and data capability and the unit housed in a metal enclosure can be kept outside to provide serial port connection. The GSM modem supports popular "AT" command set so that users can develop applications quickly. The product has SIM card holder to which activated SIM card is inserted for normal use. The power to this unit can be given from UPS to provide uninterrupted operation. This product provides great feasibility for devices in remote location to stay connected which otherwise would not have been possible where telephone lines do not exist. Application areas are:

- Mobile transport vehicles.
- LAN based SMS servers
- Alarm notification of critical events including servers
- Network monitoring and SMS reporting
- Data transfer applications from remote locations
- Monitor and control of serial services through GSM Network
- Integration to custom software for warehouse, stock, production, dispatch notification through SMS, automatic meter reading, ect.…

2.6.1 SMS communications

Short message service is a type of communications process that enables the transmission of short text messages and data transfers to and from mobile devices such as cell phones. Messages are usually limited from 140 to 160 characters in length and are stored and forwarded at SMS centers. This allows messages and data transfers to be retrieved immediately or at a later time via an SMS center.

2.6.2 Benefits of using SMS

SMS communications provide an affordable and convenient means to send and receive data using mobile devices such as cell phones. Businesses and industry often require 24-hour coverage of their operations and have personnel who are on-call after normal work hours to handle work-related issues and emergencies. There are employees who are responsible for the proper functioning of equipment and processes at remote sites. Managers need to be notified of significant events.

Chapter Three

System Hardware and Software Considerations

3.1 System Description

The fire fighting system consists of: Detector group which can be a smoke, flame or heat detector, input keys that allow the user to do different actions, microcontroller Atmega 32, display, a group of LEDs, ULN2003A driver, and the output relays.

3.2 System Hardware Considerations

The main hardware components used to build this system are shown in Figure 3.1.

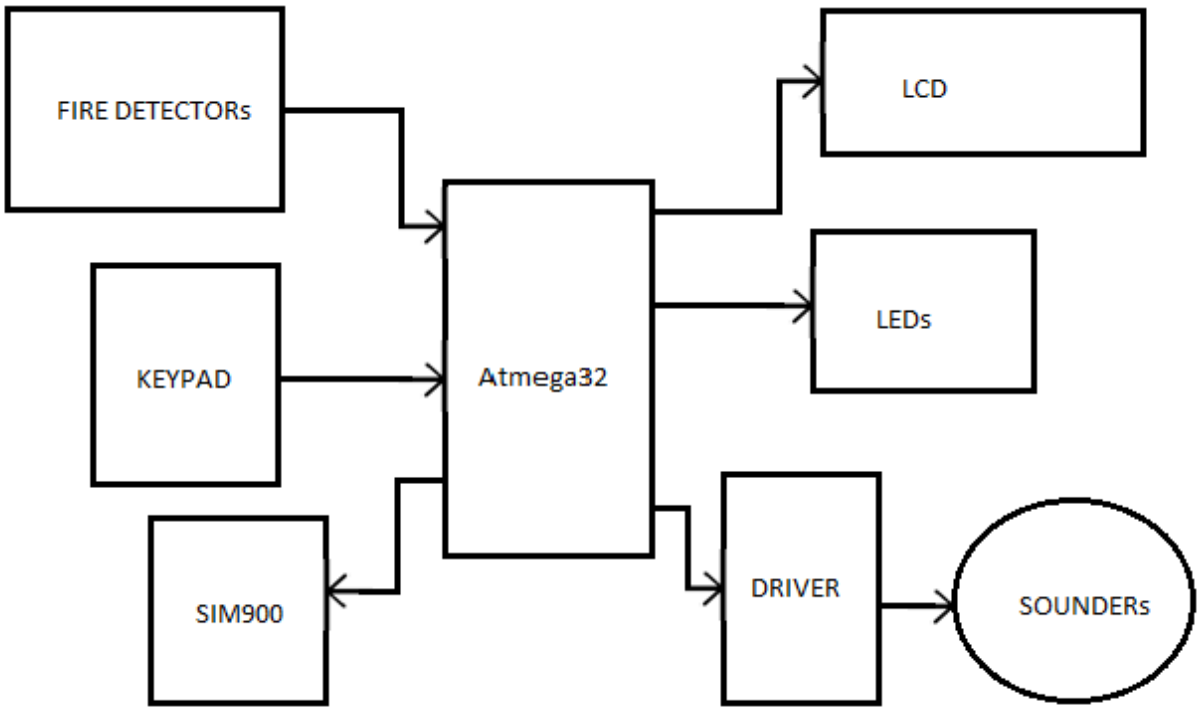

Figure 3.1: Block diagram for the circuit

3.2.1 Atmega32

Atmega32 microcontroller is the main brain of the study and it is hold the program of the operation conditions to be executed to reach the target. Atmga32 used because it is compatible to the study from the side of number of inputs/outputs, the internal memory to download the program and the other features, Figure 3.2 shows the microcontroller's pins.

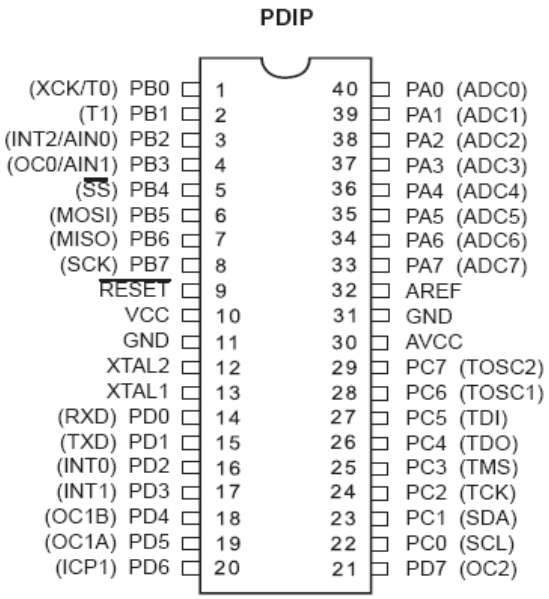

Figure 3.2: Atmega32

- Atmega32 features

• High-performance, low-power Atmel® AVR® 8-bit microcontroller

• Advanced RISC architecture

– 131Powerful instructions – Most single-clock cycle execution

– 32 x 8 general purpose working registers

– Fully static operation

– Up to 16MIPS throughput at 16MHz

– On-chip 2-cycle multiplier

• High endurance non-volatile memory segments

– 32Kbytes of in-system self-programmable flash memory

– 1024bytes EEPROM

– 2Kbyte internal SRAM

– Write/Erase cycles: 10,000 Flash/100,000 EEPROM

– Data retention: 20 years at 85°C/100 years at 25°C (1)

– Optional boot code section with independent lock bits

In-system programming by on-chip boot program

True read-while-write operation

– Programming lock for software security

• JTAG (IEEE std. 1149.1 compliant) interface

– Boundary-scan capabilities according to the JTAG standard

– Extensive on-chip debug support

– Programming of flash, EEPROM, fuses, and lock bits through the JTAG interface.

• Peripheral features

– Two 8-bit timer/counters with separate prescalers and compare modes.

– One 16-bit timer/counter with separate prescaler, compare mode, and capture mode.

– Real time counter with separate oscillator.

– Four PWM channels.

– 8-channel, 10-bit ADC:

8 single-ended channels.

7 differential channels in TQFP package only.

2 differential channels with programmable gain at 1x, 10x, or 200x.

– Byte-oriented two-wire serial interface.

– Programmable serial USART.

– Master/Slave SPI serial interface.

– Programmable watchdog timer, 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.

– Six sleep modes: Idle, ADC noise reduction, power save, power down, standby and extended standby.

• I/O and packages:

– 32 programmable I/O lines.

– 40-pin PDIP, 44-lead TQFP, and 44-pad QFN/MLF.

• Operating voltages:

– 2.7V-5.5V for ATmega32L.

– 4.5V-5.5V for ATmega32.

• Speed grades:

– 0-8MHz for ATmega32L.

– 0-16MHz for ATmega32.

• Power consumption at 1MHz, 3V, 25°C

– Active: 1.1mA

– Idle mode: 0.35mA

– Power-down mode: < 1μA

➢ Pin descriptions:

- VCC**:** Digital supply voltage.
- GND: Ground.
- Port A (PA7..PA0): Port A serves as the analog inputs to the A/D converter. Port A also serves as an 8-bit bi-directional I/O port, if the A/D converter is not used. Port pins can provide internal pull-up resistors (selected for each bit). The port A output buffers have symmetrical drive characteristics with both high sink and source capability. When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated. The port A pins are tri-stated when a reset condition becomes active, even if the clock is not running. In the current system we config a part from port A as analog inputs (PA.0 to PA.3) to receive the signals from the different four zoon's. The internal A/D converts the received signal to binary system and divided with a suitable number to achieve the real value of the signal. PA.4 to PA.7 used as digital output.
- Port B (PB7..PB0): 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 capability. As inputs, port B pins that are externally pulled low will 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. PB.0 to PB.6 used as a digital

input and receives to input keys status, port B.7 configured as digital output to Light Emitting Diodes (LED).

- Port C (PC7..PC0): Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, port C pins that are externally pulled low will source current if the pull-up resistors are activated. The port C pins are tri-stated when a reset condition becomes active, even if the clock is not running. If the JTAG interface is enabled, the pull-up resistors on pins PC5(TDI), PC3(TMS) and PC2(TCK) will be activated even if a reset occurs. The TD0 pin is tri-stated unless TAP states that shift out data are entered. This port is used with the LCD pins PC.4 to PC.7 to transfer data to the LCD, PC.2 and PC.3 as enable and reset signals at LCD. Port C.0 and port C.1 are configured as digital outputs to LEDs.
- Port D (PD7..PD0): Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The port D output buffers have symmetrical drive characteristics with both high sink and source capability. 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. PD.1 is used as a serial transmitter to send data to GSM module, The rest of this port is configured as a digital outputs to LEDs that preview the zoon's status two LEDs for each zoon represent the fault and alarm status.
- RESET: A low level on this pin for longer than the minimum

pulse length will generate a reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a reset.

- XTAL1: Input to the inverting oscillator amplifier and input to the internal clock operating circuit.
- XTAL2: Output from the inverting oscillator amplifier.
- AVCC: AVCC is the supply voltage pin for port A and the A/D converter. It should be externally connected to VCC, even if the A/D converter

is not used. If the A/D converter is used, it should be connected to VCC through a low-pass filter.

- AREF**:** AREF is the analog reference pin for the A/D converter.

3.2.2 SIM900

SIM900 offers the benefit of sending short message to the user inform him with any change at his building, The positive edge of the signal is used to obtained this function that any change from zero to one at zoon's alarm or zoon's faults signals will send automatically in a short massage, Also if two pair of zoon's activate the counter of the actuator and the actuator status.

The SIM900 is a complete quad-band GSM/General Packet Radio Service (GPRS) solution in a Surface-Mount Device (SMT) module which can be embedded in the customer applications. Featuring an industry-standard interface, the SIM900 delivers GSM/GPRS 850/900/1800/1900MHz performance for voice, SMS, data, and fax in a small form factor and with low power consumption. With a tiny configuration of 24mmx24mmx3mm. SIM900 can fit almost all the space requirements in your M2M application, especially for slim and compact demand of design. Figure 3.3 shows the SIM900. The general features of SIM900 are:

- Quad-Band 850/ 900/ 1800/ 1900MHz.
- GPRS multi-slot class 10/8.
- GPRS mobile station class B.
- Compliant to GSM phase 2/2+

  Class 4 (2W @850/ 900MHz).

  Class 1 (1W @ 1800/1900MHz).
- Dimensions: 24*24*3mm.
- Weight: 3.4g.
- Control via AT commands (GSM 07.07, 07.05 and SIMCOM enhanced AT Commands).

- SIM application toolkit.
- Supply voltage range 3.4 ... 4.5V.
- Low power consumption.
- Operation temperature: -30°C to +80°C.

Figure 3.3: SIM900

3.2.3 Fire detectors

There are many type of fire detector according to the detect principle that mentioned in section two, A heat detector is chosen because of the availability, simple, inexpensive and can detect wide range of temperature.

- LM35 Sensor

The LM35 series are precision integrated-circuit temperature sensors, with an output voltage linearly proportional to the centigrade temperature. Thus the LM35 has an advantage over linear temperature sensors calibrated in Kelvin, as the user is not required to subtract a large constant voltage from the output to obtain convenient centigrade scaling. The LM35 does not require any external calibration or trimming to provide typical accuracies of ±¼°C at room temperature and ±¾°C over a full -55°C to +150°C temperature range. The low trimming and calibration at the wafer level. The low output impedance, linear output, and precise inherent calibration of the LM35 make interfacing to readout or control circuitry especially easy. The device is used with single power supplies, or with plus and minus supplies. As the LM35 draws only 60μA from the supply, it has very low self-heating of less than 0.1°C in still air. The LM35 is rated to operate over a −55°C to +150°C temperature range,

while the LM35C is rated for a −40°C to +110°C range (−10° with improved accuracy). The LM35 series is available packaged in hermetic TO transistor packages, while the LM35C, LM35CA, and LM35D are also available in the plastic TO-92 transistor package. The LM35D is also available in an 8-lead surface-mount small outline package and a plastic TO-220 package. The main features of LM35 are:

- Calibrated directly in ° Celsius (Centigrade).
- Linear + 10 mV/°C scale factor .
- 0.5°C ensured accuracy (at +25°C).
- Rated for full −55°C to +150°C range.
- Suitable for remote applications.
- Low cost due to wafer-level trimming.
- Operates from 4 to 30V.
- Less than 60μA current drain.
- Low self-heating, 0.08°C in still air.
- Nonlinearity only ±¼°C typical.
- Low impedance output, 0.1Ω for 1mA Load.

3.2.4 Liquid crystal display

LCD screen is an electronic display module and find a wide range of applications. A 16x2 LCD is a very basic module and is very common in various devices and circuits. They are preferred over seven segment displays. There are many advantages when compared to seven segment displays. They are: LCDs can display characters, numbers and even graphics [12]. A 16x2 LCD means it can display 16 characters per line and there are two such lines. In this LCD each character is displayed in 5x7 pixel matrix. This LCD has two registers, namely, command and data.

The command register stores the command instructions given to the LCD. A command is an instruction given to LCD to do a predefined task like initializing it, clearing its screen, setting the cursor position, controlling display etc. The data register stores the data to be displayed on the LCD. The data is

the ASCII value of the character to be displayed on the LCD. The reasons being: LCDs are economical; easily programmable; have no limitation of displaying special and even custom characters (unlike in seven segments), animations and so on. Figer 3.4 shows 16*2 LCD used.

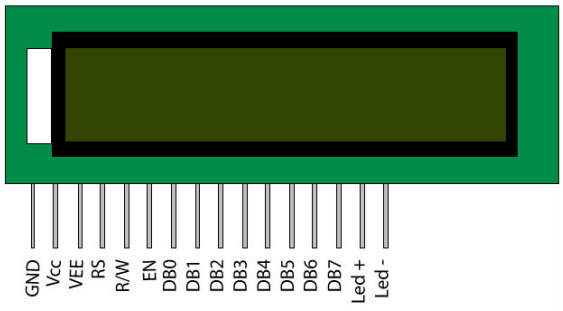

Figure 3.4: Liquid crystal display

- LCD pin discription as follows :

Gnd: Ground pin 0VDC.

VCC: Supply voltage (4.7-5.3) VDC.

VEE: Variable resiste conect to this pin to adjust the contract.

RS: Selects command register when low, and data register when high.

R/W: Low to read from the register and high to write to the register.

EN: Sends data to data pins when a high to low pulse is given.

DB0-DB7 : Data pins used to transtfare data to LCD.uln

LED +: Backlight supply 5vdc

LED -: Backlight ground.

3.2.5 ULN2003 motor driver:

The ULN2003 is an array of seven NPN Darlington transistors capable of 500mA, 50V output. It features common-cathode flyback diodes for switching inductive loads. A Darlington transistor achieves very high current amplification by connecting two bipolar transistors in direct current (DC) coupling so the current amplified by the first transistor is amplified further by the second one. The resultant current gain is the product of those of the two component transistors. The seven Darlington pairs in ULN2003 can operate independently except the common cathode diodes that connect to their respective collectors. The main specifications of ULN2003 motor driver are:

- 500mA rated collector current (single output).
- 0V output (there is a version that supports 100V output).
- Includes output flyback diodes.
- Inputs compatible with TTL and 5-V CMOS logic.

The ULN2003 is known for its high-current, high-voltage capacity, as shown in Figure 3.5. The drivers can be paralleled for even higher current output. Even further, stacking one chip on top of another, both electrically and physically, has been done. Generally it can be used for interfacing with a stepper motor, where the motor requires high ratings which cannot be provided by other interfacing devices.

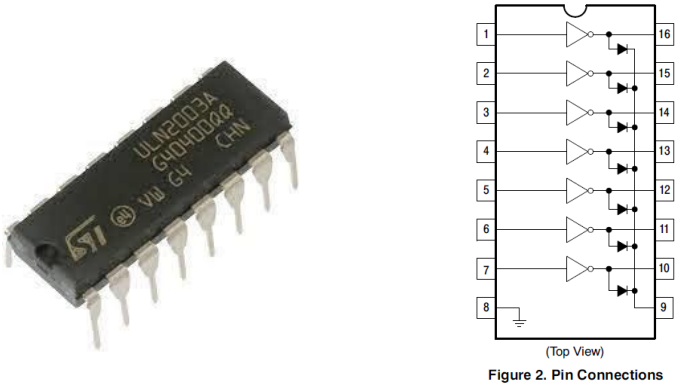

Figure 3.5: ULN2003 motor driver

3.2.6 Push buttons

A push button is a simple switch mechanism which permits user generated changes in the state of a circuit. Push button usually comes with four legs, legs are always connected in groups of two. When the pushbutton is pressed all the four legs are connected. Figure 3.6 shows the push button.

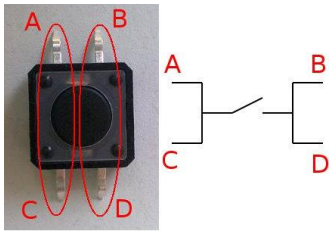

Figure 3.6: Push buttons

3.3 System Software Consideration

BASCOM language is used to write the software program and that language used because it is simple with enough capabilities to achieve the all

condition and cases required. The Proteus program used to simulate the performance.

3.3.1 System code

The system program is written using BASCOM language is shown in Appendix.

3.3.2 System simulation

The system is build up in the Proteus program as shown in Figure 3.7. The overall circuit shown in Figure 3.7 contains the input bush bottoms, sensor inputs LM35 (temperature sensors), LCD, LEDs and the output relays.

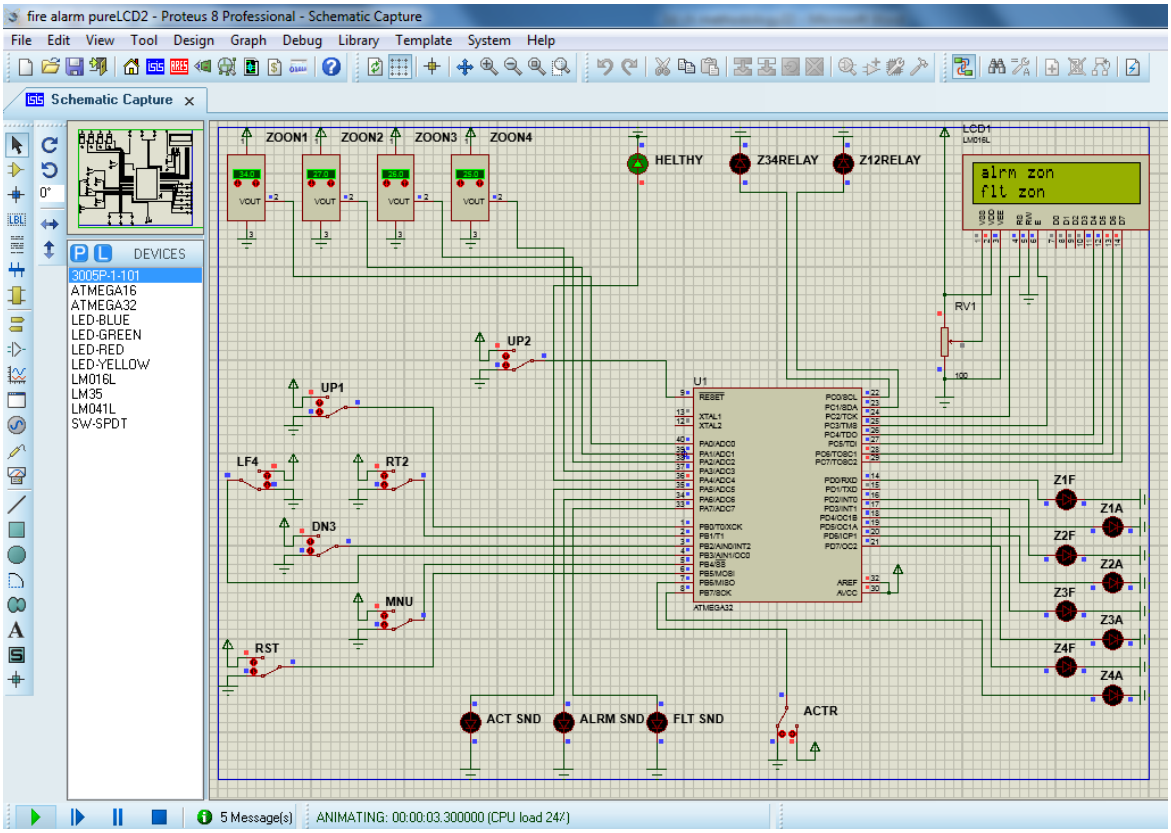

Figure 3.7: The main circuit design using Proteus software

The main circuit consist of four LM35 each one connected to zoon sensor inputs and labels as zoon1 to zoon4. The second componet is the push buttons UP1 for up and the number one, RT2 the right and number two, DN3 the down button and represent the number three, LF4 for the left button and the number four, MNU is the menu button and RST is the reset button. The direction keys used to nevegate or to select an item in the menue options. The healthy LED is a green LED that anly goes on, when all conditions are healthy, Z12RElAY

zoon one and zoon two acuater relay this is the final output to release the fire fighting liqued or gas to this area . Z34RELAY zoon three and zoon four acuater relay.

3.4 System Operation

The system LCD the word ALRM ZON and FLT ZON then check the input analog heats element for any change also check the input keys, if one heat degree reach the maximum setting (60 centegrate) the alarm point the LCD shows the alarm zoon with a number from one to four the alarm sounder operate and the zoons alarm red LED flashing, this situation stay as its as long as the alarm exist, when this tempreture goes down anly the flashing stop and the zoon lamp stay in on postion tell the user press reset key, when we press reset key the system check the alarm status clear LCD and the sounder if the signal is normal. The same response if there is alarm at two not pair zoons like zoon one and three or zoon two and zoon four (the pair zoons is zoon one and two, zoon three and four).

For more safty and accuracy normally two zoons used in pair to protect one area which need a firefithing methods like the CO_2 gas or FOUM, here zoon one and two can used together for that reson or zoon three and four. When alarm detected at two pair of zoons the LCD shows alarm in zoon X/Y gas or liqued release in Z. X/Y indicate zones number, and Z is a count down number starting from ten before the system send a comand to the gas or liqued relay, then the screen LCD shows the messege gas/liqued released. For more flexbality to the system there is many option at menu that offer to the end user many option to the operation and mentainance. Figure 3.8 shows a flow chart to the keys function and the system menue option which allows the user to change the system setting easily as follow:

- Test LED : By presseing UP1 key all LEDs goes on for 10 second then off which allow the user to test all LEDs .
- Read the sensor valus: User can check the value of the zoons sensor by pressing DN3 key, the result apear at the system LCD for ten seconds.

- ACT setting: In the practical application it is recommended to not activate the fire fighting method in auto mode to avoied losing the fire fighting mediom by a wrong reson , this setting provide to mode to the final output (actuater) manual only and the auto mode .
- SMS setting: The system has a cabaility to send a short messege with any alarm or fault at the system and this function can be activated or cancel.
- MNT setting: Farther option allow the user to simulate any of inputs sensor it ofer three option for each zoon force healthy, force alarm and force fault.

Figure 3.8 shows the keys functions of the system .

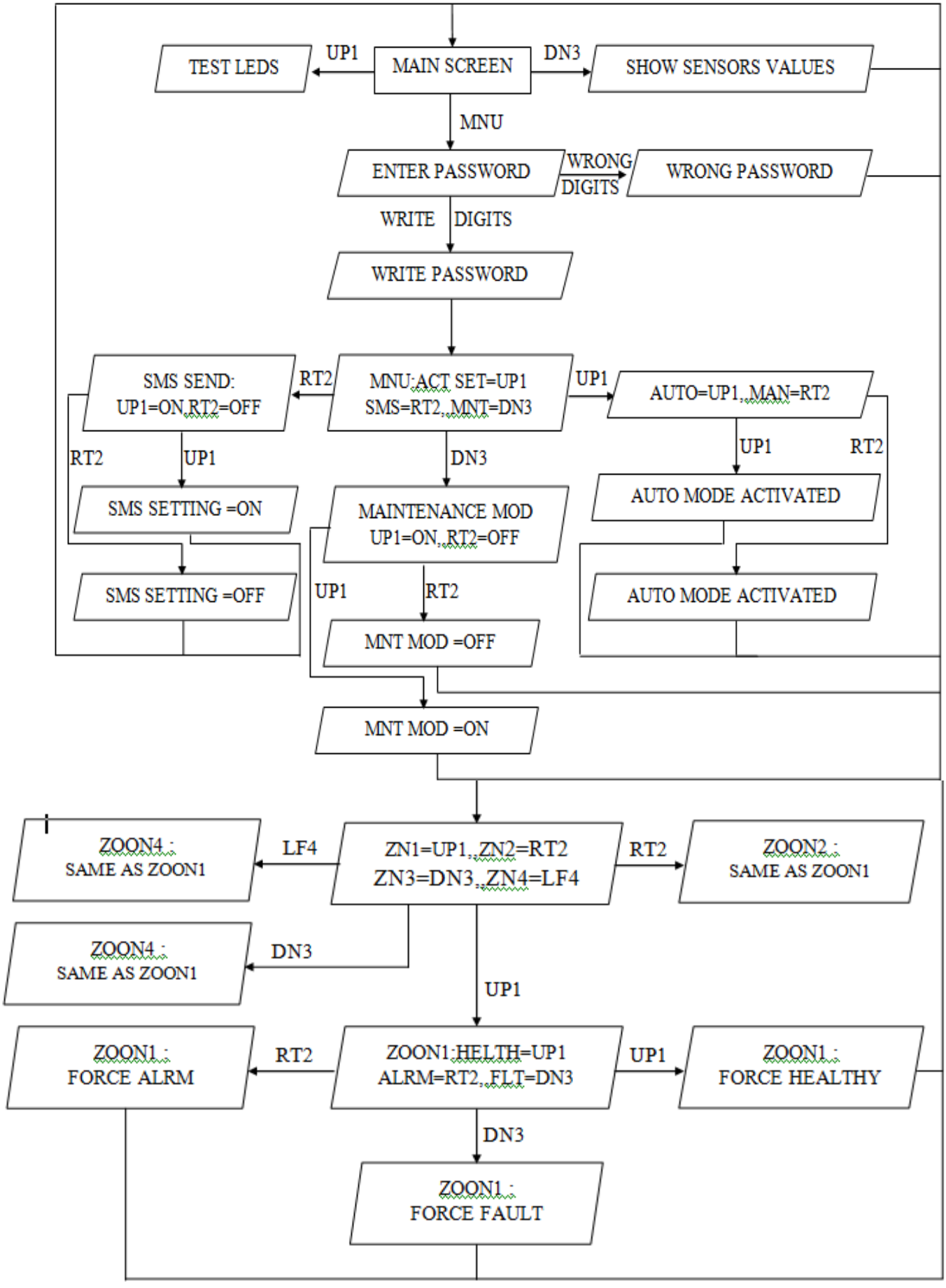

Figure 3.8: Keys function flow chart

Chapter Four

System Implementation and Testing

4.1 System Implementation

In this study the system is divide to six groups should be connected together.

4.1.1 Power supply board

As shown in Figure 4.1, the power supply board offers the required power to the circuits and consists of 5VDC power supply and 5VDC bins and slots, the board should connect to 220VAC source.

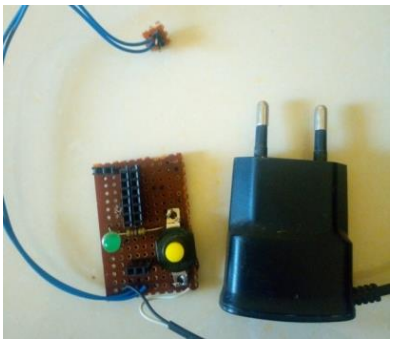

Figure 4.1: Power supply board

4.1.2 Main board

This board contains Atmega32 microcontroller and the output relays. Also there are slots to connect the other boards, slots for power supply and slots for the USBASP programmer can used any time to modify the microcontroller program. Figure 4.2 shows the main board.

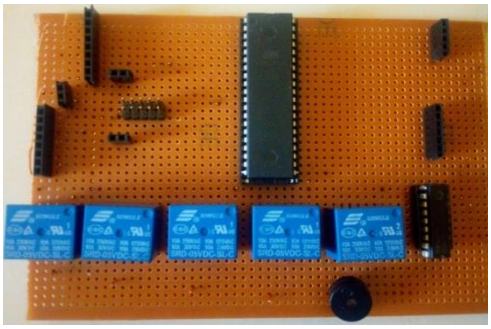

Figure 4.2: Main board

4.1.3 Liquid crystal display

The board carries the LCD, variable resistance to adjust the LCD

brightness, data pins and power supply pins. Figure 4.3 shows the LCD board.

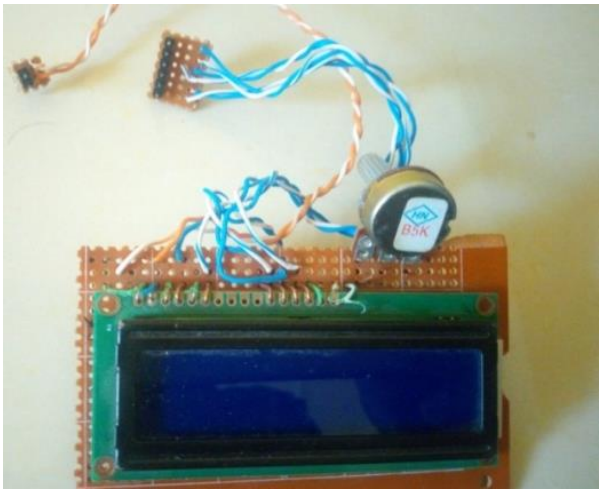

Figure 4.3: LCD board

4.1.4 Keys board

The navigation and selection keys to communicate to the system implemented to this board in addition to a green LED which represent the healthy condition of the system shown in Figure 4.4. The board end with data pins to connect to the main board and power supply pins to connect to the power board.

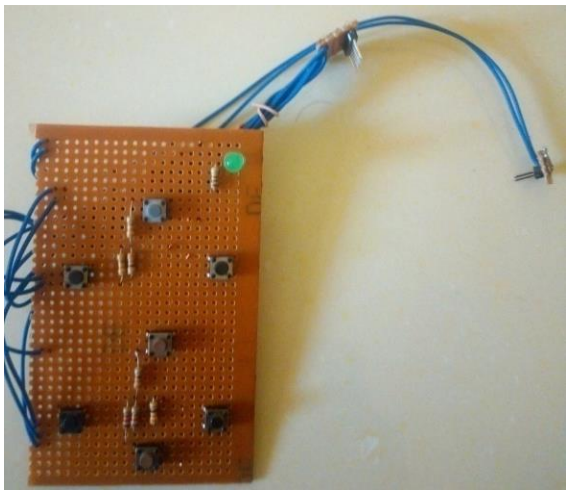

Figure 4.4: keys board

4.1.5 Sensors board

Figure 4.5 shows sensor board which hold the LM35 temperature sensors and connected with a data terminal which carry the analog values of the temperature and a power pins to feed the sensor with the necessary power to operate.

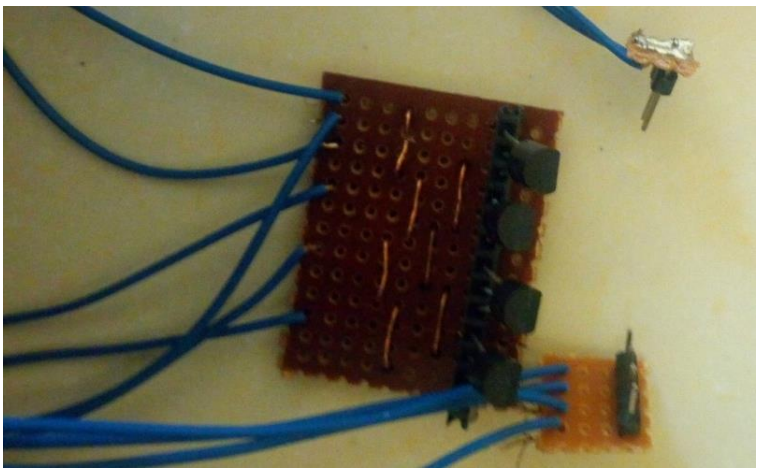

Figure 4.5: Sensors board

4.1.6 LEDs board

There are four yellow LEDs indicate a fault condition of the protected zones and four red LEDs represent the alarm in each zone. Figure 4.6, shows LEDs board.

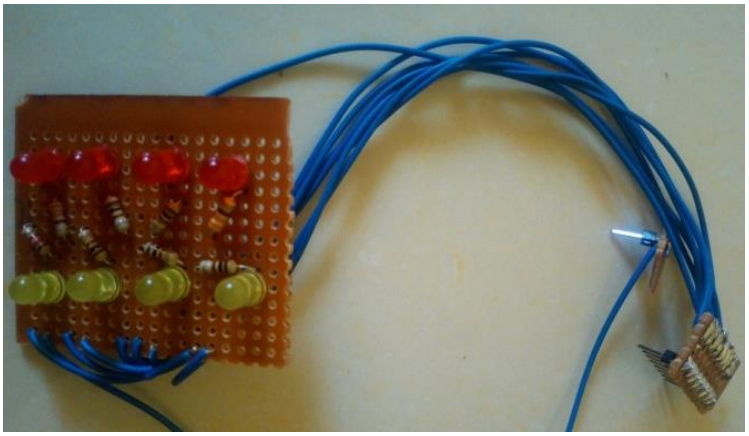

Figure 4.6: LEDs board

4.2 System Testing

In order to test the system all parts should connect with the main board throw the write slots also should powered from the 5VDC power supply and switch on the power. The different cases of operations and conditions are:

4.2.1 Case one: Normal condition

When temperatures sensors values shows normal values beyond both alarm and fault level, the fault and alarm lines in the LCD shows empty lines, all zone's LEDs are off, the healthy LED is on, and all output relays are off. This is shown in Figure 4.7.

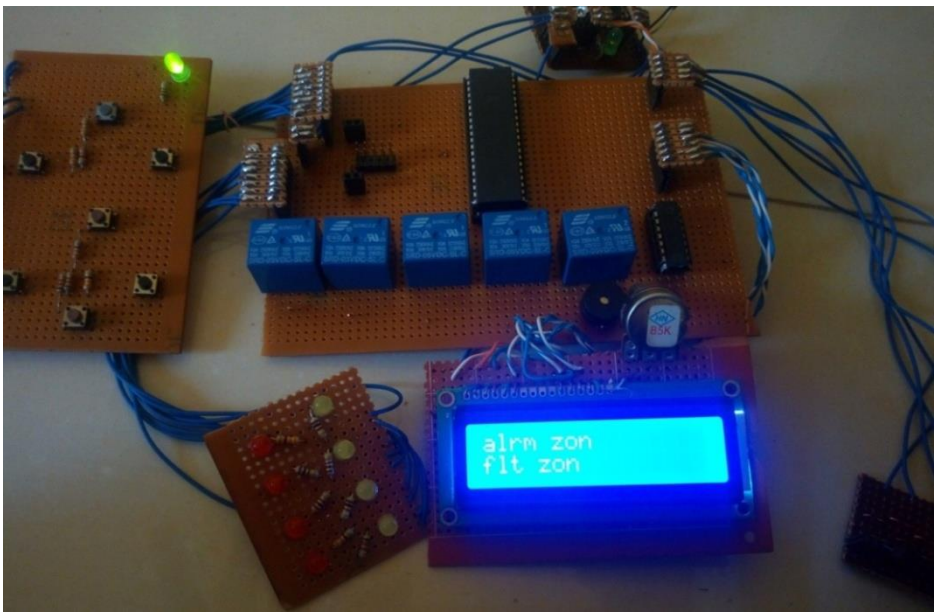

Figure 4.7: Normal condition

4.2.2 Case two: Alarm in one zone

One of the zone's temperature reach alarm values, the LCD in Figure 4.8 shows this zone number, the zone's alarm red LED flashing, the alarm sounder relay operate and system send short message to the user phone.

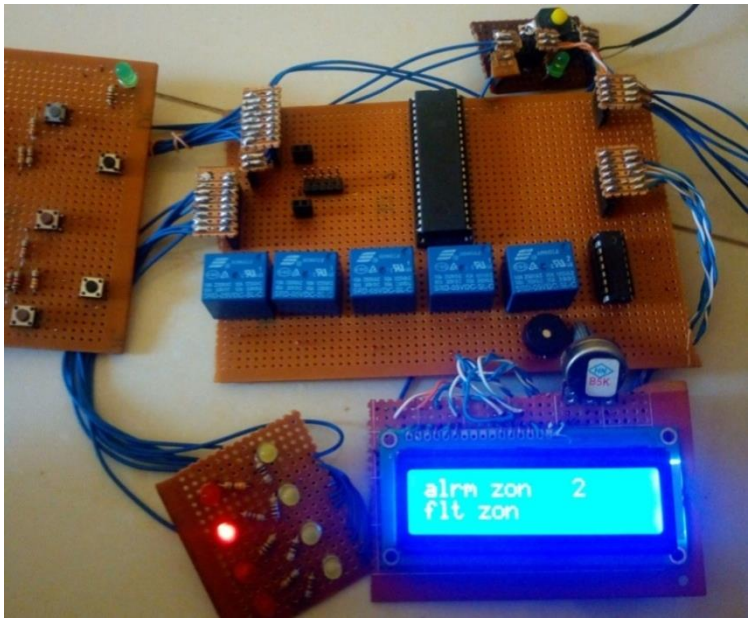

Figure 4.8: Alarm in zone two

4.2.3 Case three: Alarm in two not pair zones

The four zones are divided into two pairs, zone one and two, and the other pair are zone three and four, if two not pair zones temperature reach alarm values zone 1 and 3, 1 and 4, 2 and 3 or 2 and 4, the LCD display the zones

numbers, the zone's alarm red LEDs flashing, the alarm sounder relay operate and short message send to the user phone. This case is shown in Figure 4.9.

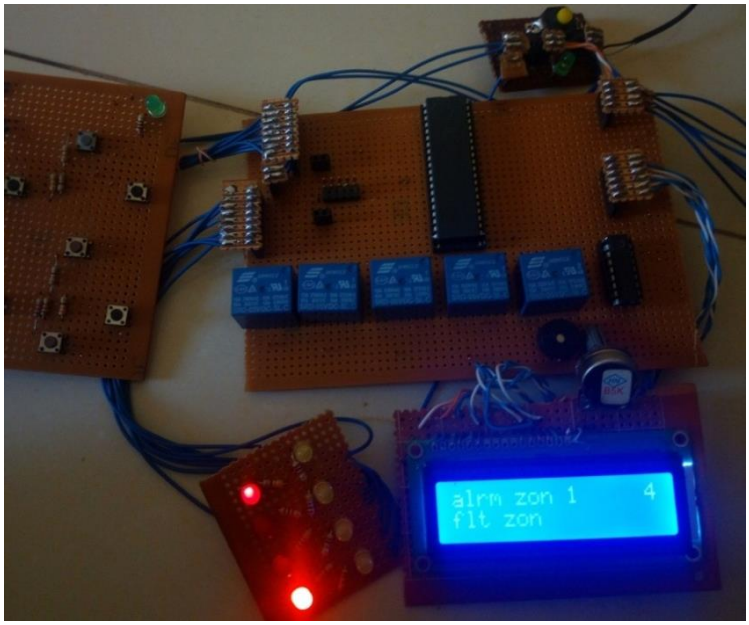

Figure 4.9: Alarm in zone one and four

4.2.4 Case four: Alarm in two pair zones

If two pair zones temperature reach alarm values zone 1 and 2 or 3 and 4, the zone's alarm red LEDs flashing, the LCD display those zones numbers as shown in Figure 4.10.

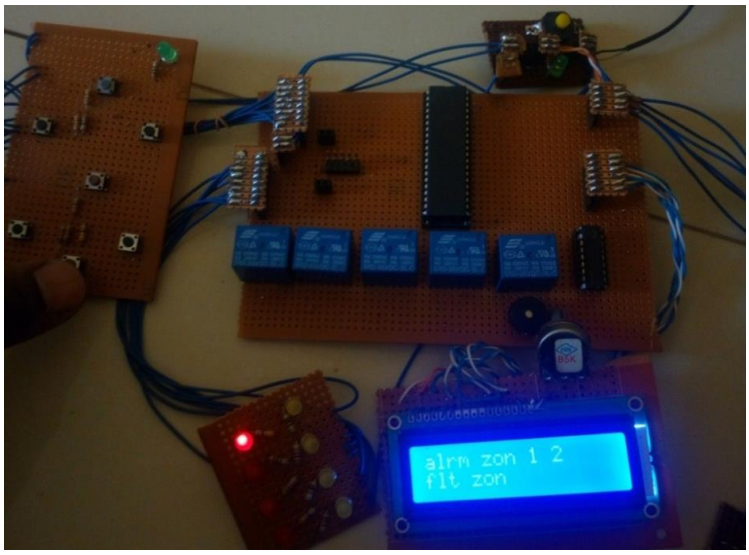

Figure 4.10: Alarm in zone one and two

The system then shows the message: alarm in zone x/y (1 and 2 or 3 and 4), gas/liquid released in z (a count_down started from z=10), the alarm sounder relay operate, This is shone in Figure 4.11.

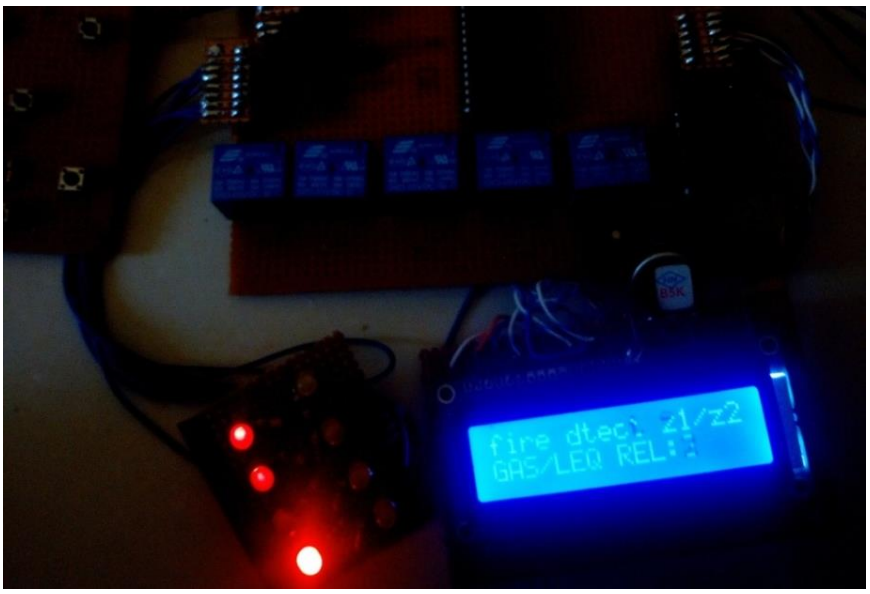

Figure 4.11: Alarm in zone one and two count down started

In the end of the count down to zero the specific gas or liquid output relay goes on and the LCD shows: alarm in zone x/y, gas/liquid released. Also short message sends to the user phone at all previous states. This is shown in Figure 4.12.

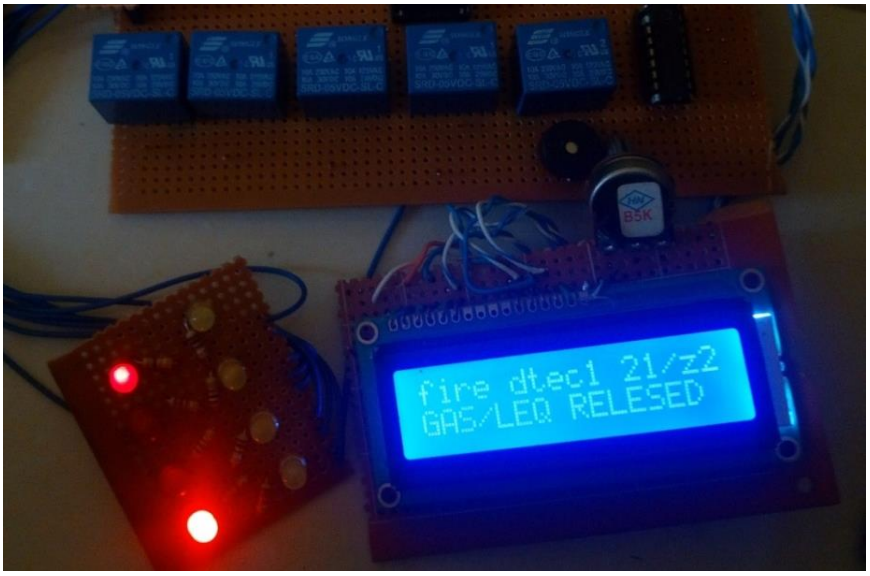

Figure 4.12: Alarm in zone one and two count_down end

4.2.5 Case five: Faults

Un logical temperatures values means some fault in the sensor so the system indicate that with showing the affected zone number in the fault line of the LCD, the fault zone LED goes on and the fault sounder operate. This is shown in Figure 4.13.

Figure 4.13: Fault in zone one, three and four

4.2.5 System options

The system options as follows:

- Upper key/ test LEDs

This key has two function navigation or selection at the menu and a test LEDs function. When this key pressed from the main screen all LEDs in the system goes on for one second as shown in Figure 4.14.

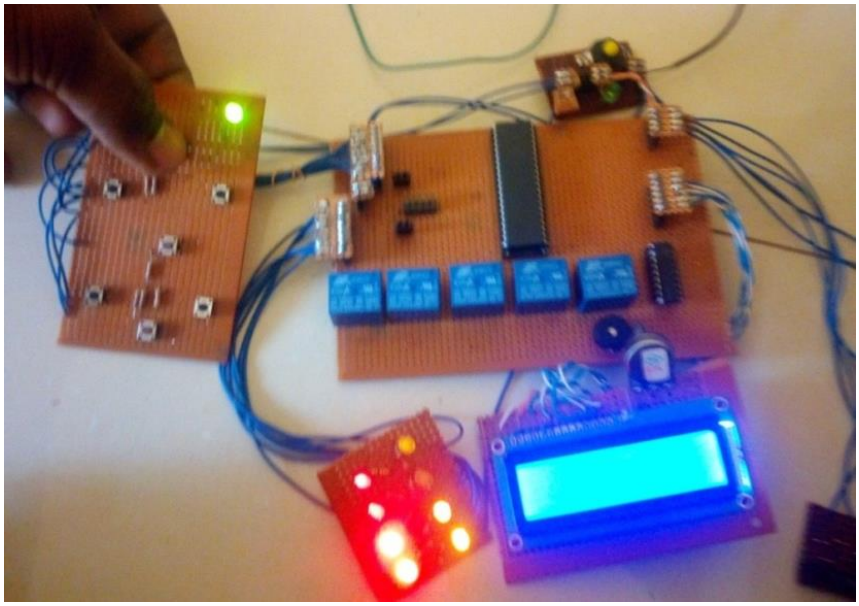

Figure 4.14: Test LEDs

- Down key/ zones temperatures

This key has double function navigation or selection at the menu and shows the zones current temperatures. When this key pressed from the main screen. This is shown in Figure 4.15.

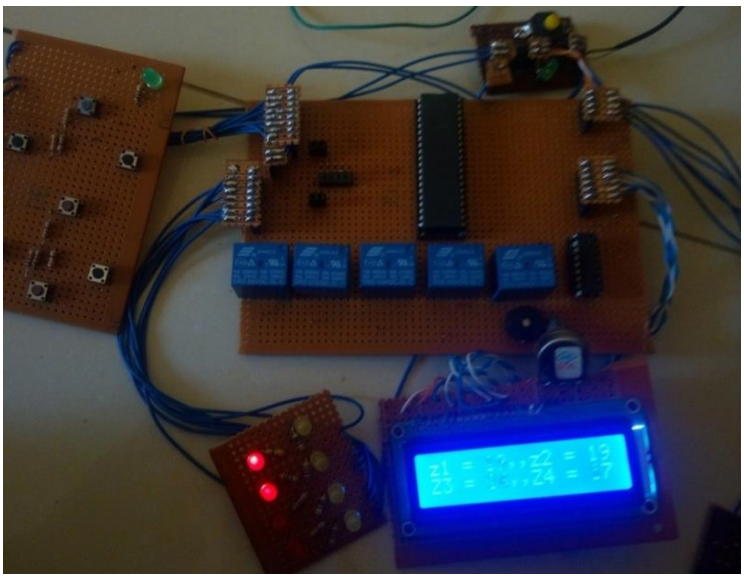

Figure 4.15: Zones temperatures

- Manual actuator key

This key exists in all fire fighting systems normally protected with a cover to operate the system manually, if this key pressed the system LCD shows the massage: “Manual release AC, Z12=UP1,,Z34=DN3”, This is an option to the user to determine which pair of zone should operate, This is shown in Figure 4.16.

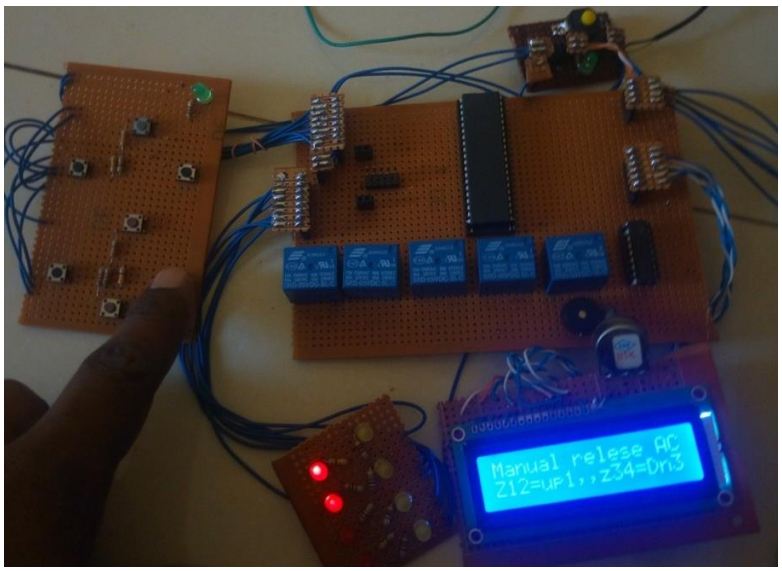

Figure 4.16: Manual actuator key

Pressing upper key lead to operate zone 1/2 actuator and down key operate zone 3/4 actuator. The LCD shows the selected pare of zones before the count down started and both count sounder and alarm sounder operate. This is shown in Figures 4.17 and 4.18.

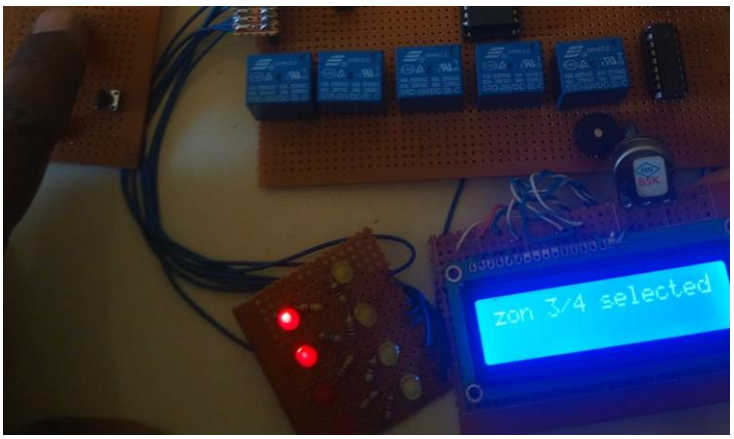

Figure 4.17: Manual actuator key zone selected

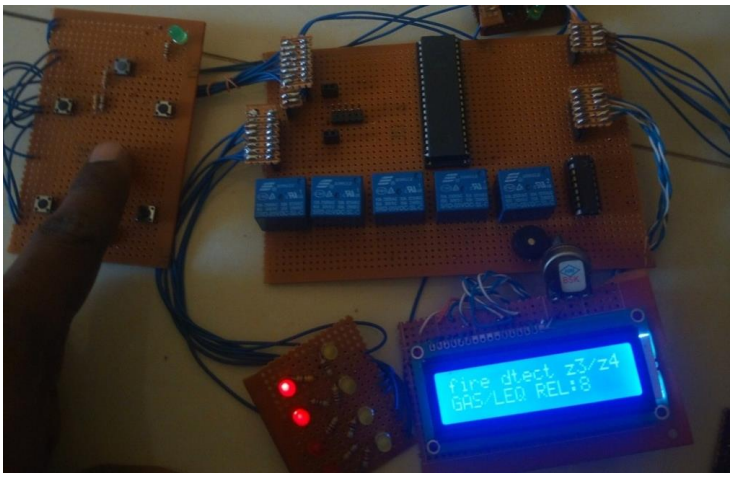

Figure 4.18: Manual actuator key count down

- Menu key

The system menu protected with password, If menu key pressed the LCD show enter password massage if wrong digits hits “wrong password” massage appears and return to main screen otherwise, “correct password” appear at the system LCD and show the main menu selections, this is shown in Figure 4.19.

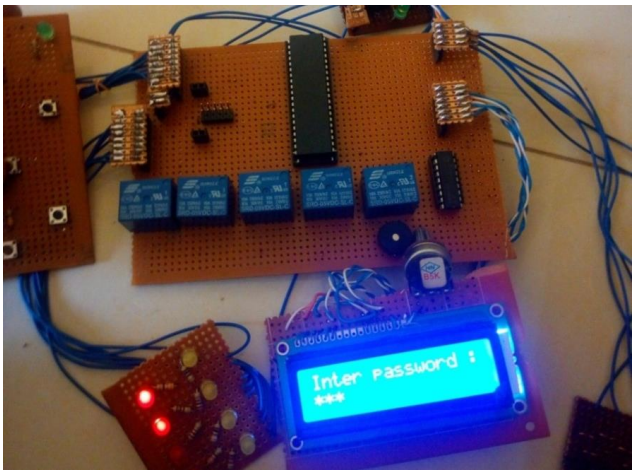

Figure 4.19: Main menu, password

Correct password give the user the permeation to access the main menu and showing three option to select from, actuator setting, SMS setting and maintenance mode selection. Also the required key to enter any sub menu, this is shown in Figure 4.20.

Figure 4.20: Main menu, main items

- actuator setting: this option can be selected by pressing the upper key ”UP1”, There are two setting available:

Auto mode: can be selected by pressing the upper key again, the system shows “Auto mode active”, The actuator then operate regarding to the zones status.

Manual mode: can be selected by pressing the upper key again and the system shows “Manual mode active”, The actuator then operate only with the manual key regardless to the zones status. Figure 4.21-4.22 shows the system screen when each mode selected.

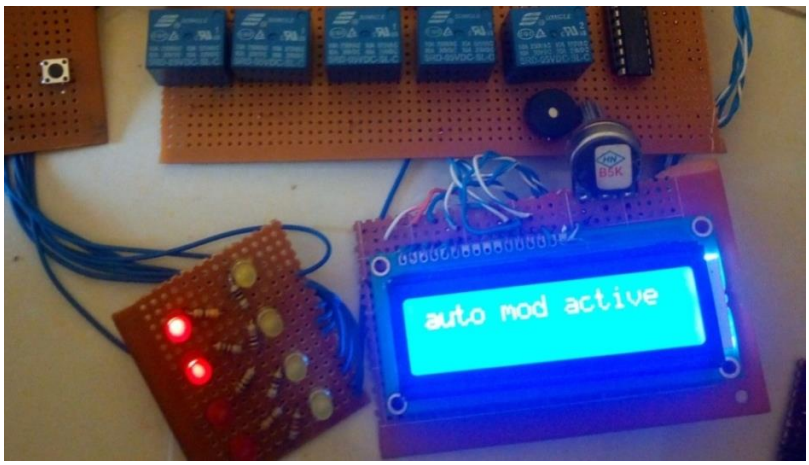

Figure 4.21: Main menu, actuator sub menu, auto mood

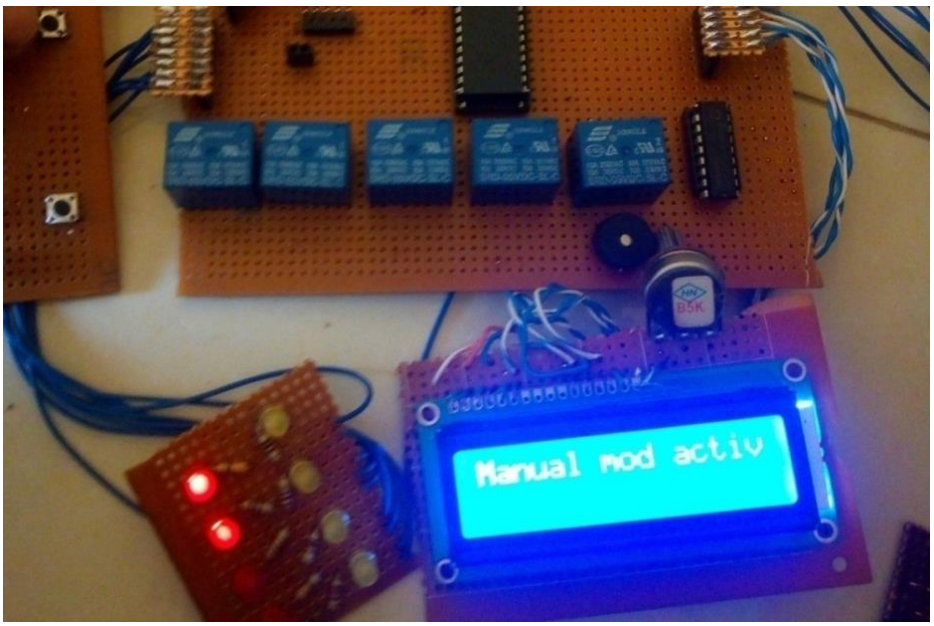

Figure 4.22: Main menu, actuator sub menu, Manual mod

- SMS setting: select this option by pressing the Wright key from the main menu items, The screen shows Two setting available to this option and the required key to select a setting as shown in Figure 4.23.

Figure 4.23: Main menu, SMS sub menu

SMS on: Press the upper key activate SMS at the system, the screen shows "SMS setting =on", The system send messages at any status change, This is shown in Figure 4.24.

SMS off: Press the wright key cancel SMS at the system, the screen shows "SMS setting =off" and the system stop sending messages.

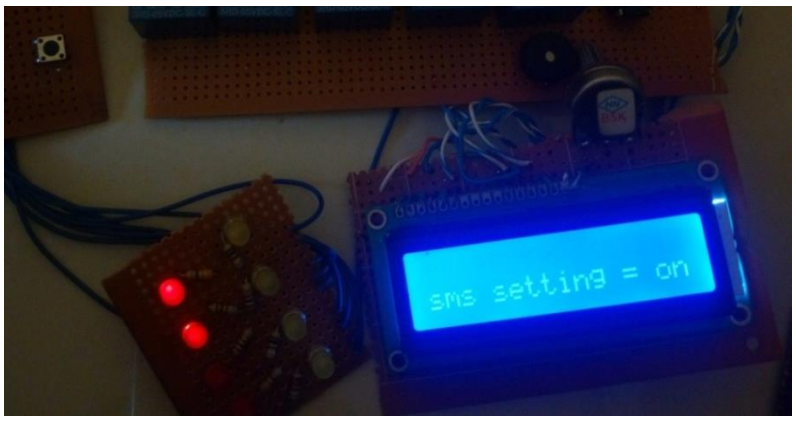

Figure 4.24: Main menu, SMS sub menu, on value selected

- Maintenance mode: this option can be selected by pressing the Down key from the main menu items, the screen shows two setting available to this option and the required key to select a setting as shown in Figure 4.25.

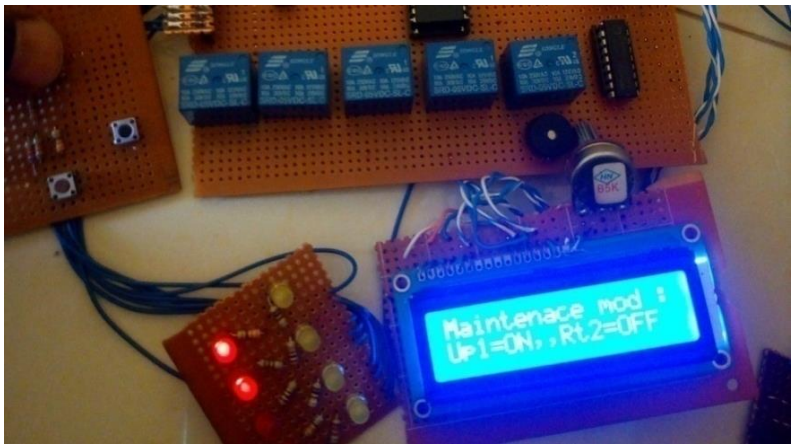

Figure 4.25: Main menu, Maintenance sub menu

Maintenance mode on: this option can be selected by pressing the up key from the maintenance menu, the screen shows “Maintenance mode on”, as shown in Figure 4.26.

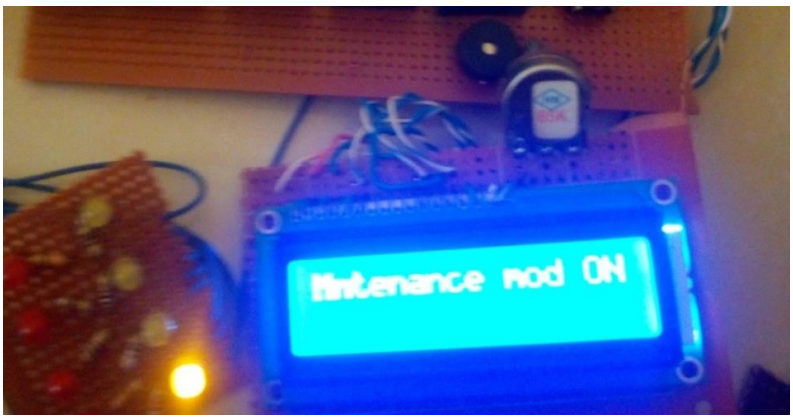

Figure 4.26: Main menu, Maintenance sub menu, on mode

Set this mode lead to other screen to select a specific zone to modify and the required key for selection, this is shown in Figure 4.27.

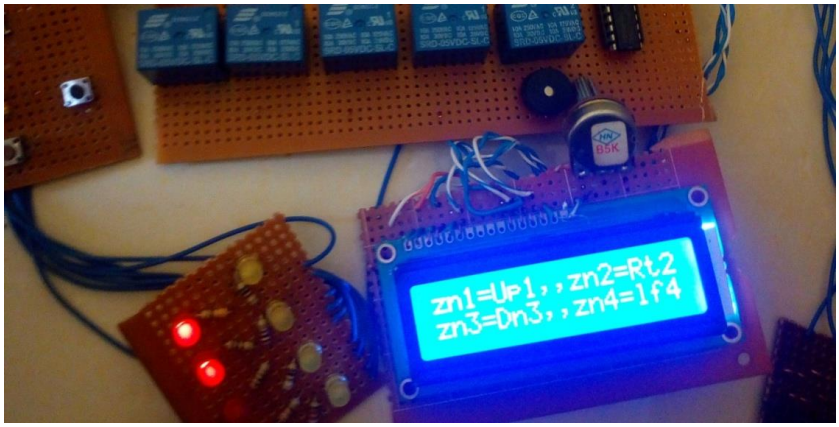

Figure 4.27: Main menu, Maintenance sub menu, zone selection

Three options available for the selected zone, force healthy the zoon forced to be in a healthy condition regardless to real value of the temperature, force alarm the zone forced to alarm and force fault the system force the selected zone into fault situation. This is shown in Figure 4.28-4.29.

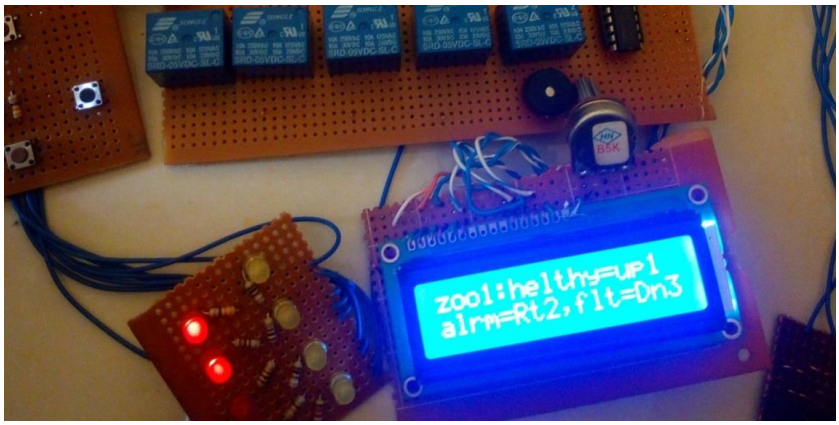

Figure 4.28: Main menu, Maintenance sub menu, zone options

Figure 4.29: Main menu, Maintenance sub menu, value entered to the zone

Maintenance mode off: This option cancel all forced values from all zones and return to a normal condition, this is shown in Figure 4.29

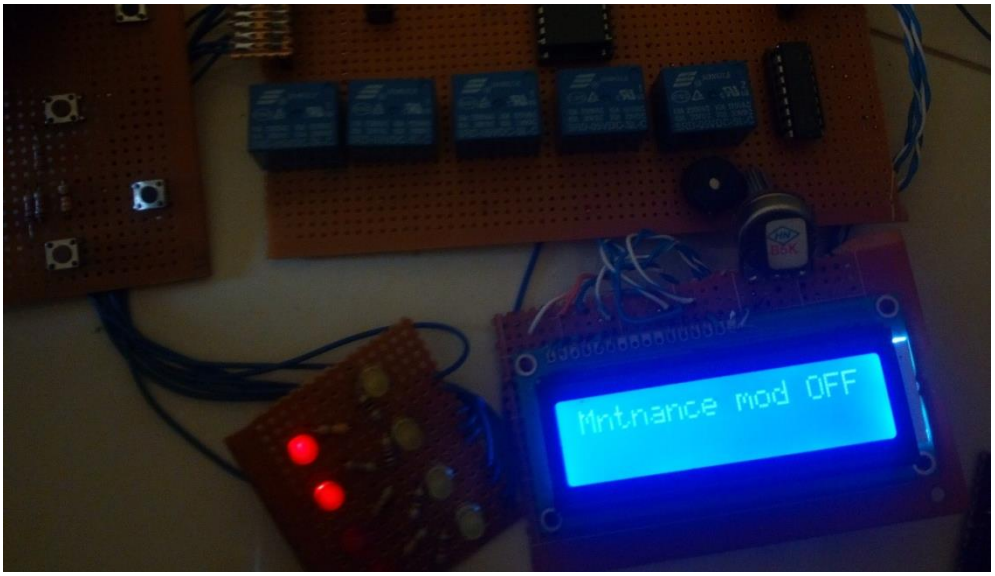

Figure 4.30: Main menu, Maintenance mod deactivated

4.2.7 Short messages sending

The system is programmed to send a notification massages to the user phone at any change at the zones situation as going from healthy to alarm or the opposite, and at the fighting medium evacuation process from the detecting a fire at tow couple zones, the countdown and the release command. Figure 4.31 and Figure 4.32 show examples to short messages send via the system.

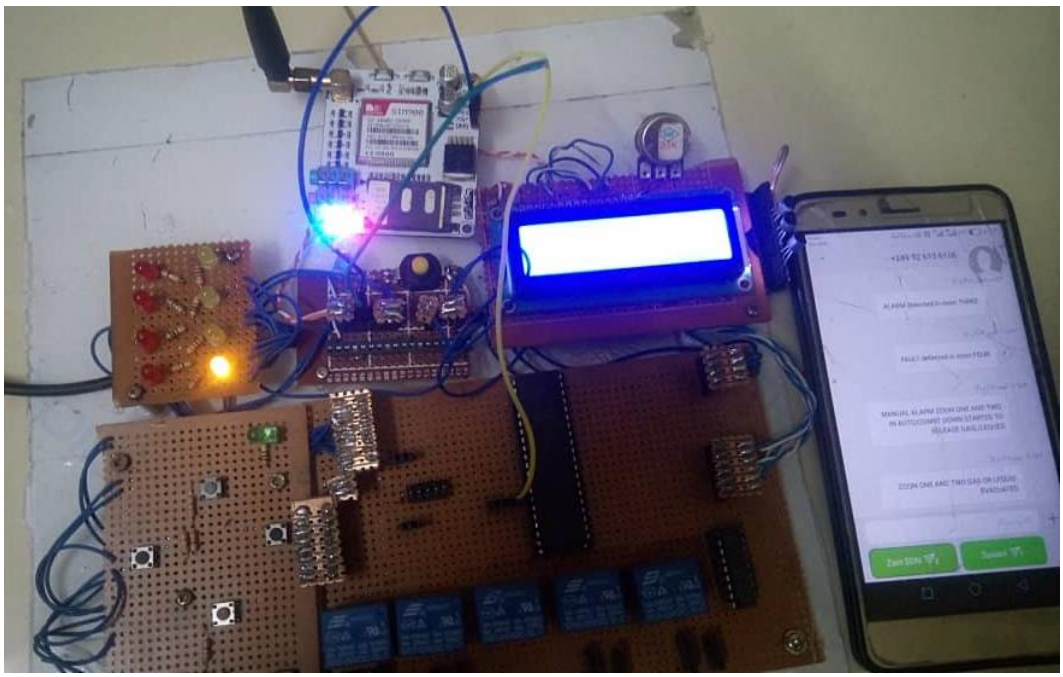

Figure 4.31: The system with SIM900 connected

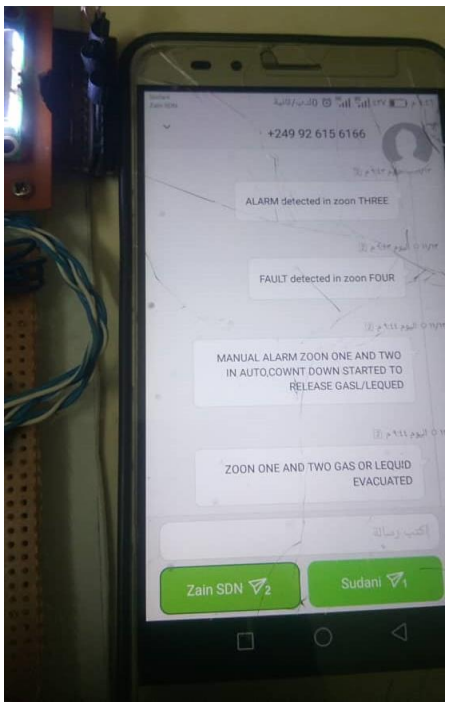

Figure 4.32: Messages sends by the system

Chapter Five

Conclusion and Recommendations

5.1 Conclusion

In this study a fire fighting system is build up using Atmega32 microcontroller. The system is tested and performs successfully and achieved all required operation conditions. In addition to that there is interactive with the system through SMS notification.

5.2 Recommendations

- Upgrade the system to be remote controlled.
- Design farther types of fire detectors.
- System development and adaptation for use as integrated systems.

References

[1] Zhang, L. and Wang, "Design and Implementation of Automatic Fire Alarm System Based on Wireless Sensor, Networks". Proceedings of the International Symposium on Information Processing (ISIP'09), Huangshan, P.410-413, August 2009.

[2] David S. Nyce, "Linear Position Sensors: Theory and Application", John Wiley & Sons, Inc., 2004.

[3] Kilian, "Modern Control Technology: Components and Systems", 2nd Edition, 2000.

[4] Ian R. Sinclair, "Sensors and transducers, third edition", 2001.

[5] Sergey Y. Yurish, "Sensors: Smart vs. Intelligent", International Frequency Sensor Association, Sensors & Transducers Journal, 2007.

[6] Harish Ramamurthy, Asad M. Madni, "Smart Sensor Platform for Industrial Monitoring and Control", 2005.

[7] Stéphane Gervais-Ducouret, "Next Smart Sensors Generation", Freescale Semiconductor 134 Av. Eisenhower, 31023 Toulouse, France 2011.

[8] Jon S. Wilson, "Sensor Technology Handbook", Elsevier Inc., 2005.

[9] Richard Wotiz, "Ionization Detectors", Circuit Cellar Inc., November 2011

[10] A.P. Godse, "Microprocessor and Microcontroller Systems", Technical Publications Pune, 2007.

[11] Dogan Ibrahim, "Advanced PIC Microcontroller Projects in C – From USB to RTOS with the PIC18F Series", 2008.

[12] Dogan Ibrahim, "Microcontroller based temperature monitoring and control", Elsevier Science & Technology Books, September 2002.

[13] Kenneth J. Ayala, "The 8051 Microcontroller Architecture, Programming and Application", Western Carolina University, 1991.

Appendix

System Code

```
$regfile = "m32def.dat"
$crystal = 4000000
$hwstack = 40
$swstack = 16
$framesize = 32
'*****************************************************************
Config Lcd = 16 * 2
Config Lcdpin = Pin , Db4 = Portc.4 , Db5 = Portc.5 , Db6 = Portc.6 , Db7 =
Portc.7 , E = Portc.3 , Rs = Portc.2
Config Adc = Single , Prescaler = Auto
Config Pina.4 = Output
Config Pina.5 = Output
Config Pina.6 = Output
Config Pina.7 = Output
Config Pind.0 = Output
Config Pind.2 = Output
Config Pind.3 = Output
Config Pind.4 = Output
Config Pind.5 = Output
Config Pind.6 = Output
Config Pind.7 = Output
Config Pinb.0 = Input
Config Pinb.1 = Input
Config Pinb.2 = Input
Config Pinb.3 = Input
Config Pinb.4 = Input
Config Pinb.5 = Input
```

```
Config Pinb.6 = Input
Config Pinb.7 = Output
Config Pinc.0 = Output
Config Pinc.1 = Output
Up1 Alias Pinb.0
Rt2 Alias Pinb.1
Dn3 Alias Pinb.2
Lf4 Alias Pinb.3
Mnu Alias Pinb.4
Rst Alias Pinb.5
Mrr Alias Pinb.6
Dim W As Word
Dim X As Word
Dim Y As Word
Dim Z As Word
Dim F11 As Byte
Dim F12 As Byte
Dim F21 As Byte
Dim F22 As Byte
Dim F31 As Byte
Dim F32 As Byte
Dim F41 As Byte
Dim F42 As Byte
Dim Co As Word
Dim Rs1 As Byte
Dim A1 As Byte
Dim A11 As Byte
Dim Ora As Byte
Dim A2 As Byte
Dim A3 As Byte
```

```
Dim A4 As Byte
Dim A5 As Byte
Dim A6 As Byte
Dim A7 As Byte
Dim A8 As Byte
Dim S1 As Byte
Dim S2 As Byte
Dim X12 As Byte
Dim X34 As Byte
Dim M12 As Byte
Dim M34 As Byte
Dim C0 As Byte
Dim C15 As Byte
Dim C11 As Byte
Dim C12 As Byte
Dim C25 As Byte
Dim C21 As Byte
Dim C22 As Byte
Dim C35 As Byte
Dim C31 As Byte
Dim C32 As Byte
Dim C45 As Byte
Dim C41 As Byte
Dim C42 As Byte
Dim Z1a As Byte
Dim Z2a As Byte
Dim Z3a As Byte
Dim Z4a As Byte
Dim Rl1 As Byte
Dim Rl2 As Byte
```

```
Dim Ev1 As Byte
Dim Ev2 As Byte
Dim Md As Byte
Dim Ad1 As Byte
Dim Fd1 As Byte
Dim Ad2 As Byte
Dim Fd2 As Byte
Dim Ad3 As Byte
Dim Fd3 As Byte
Dim Ad4 As Byte
Dim Fd4 As Byte
Dim S11 As Byte
Dim S12 As Byte
Dim S21 As Byte
Dim S22 As Byte
Dim S31 As Byte
Dim S32 As Byte
Dim S41 As Byte
Dim S42 As Byte
Dim X125 As Byte
Dim X345 As Byte
Dim Gr12 As Byte
Dim G12 As Byte
Dim L12 As Byte
Dim Gr34 As Byte
Dim G34 As Byte
Dim L34 As Byte
Dim Grp12 As Byte
Dim Grp34 As Byte
Declare Sub Reading
```

```
Declare Sub Rset
Declare Sub Res
Declare Sub Menu
Declare Sub Sms
Declare Sub Acst
Declare Sub Smon
Declare Sub Smof
Declare Sub Mntm
Declare Sub Mmzs
Declare Sub Acsa
Declare Sub Acsm
Declare Sub Mmz1
Declare Sub Mmz2
Declare Sub Mmz3
Declare Sub Mmz4
Declare Sub Fhz1
Declare Sub Faz1
Declare Sub Ffz1
Declare Sub Fhz2
Declare Sub Faz2
Declare Sub Ffz2
Declare Sub Fhz3
Declare Sub Faz3
Declare Sub Ffz3
Declare Sub Fhz4
Declare Sub Faz4
Declare Sub Ffz4
Declare Sub Mmon
Declare Sub Mmof
Declare Sub Rlgs
```

```
Declare Sub Pass1
Declare Sub Pass2
Declare Sub Pass3
Declare Sub Pass4
Declare Sub Pass5
Declare Sub Pass6
Declare Sub Pass7
Declare Sub Pass8
Declare Sub Pass9
Declare Sub Test1
Declare Sub Test2
Declare Sub Test3
Declare Sub Gmn
Declare Sub Rturn
Declare Sub Tstled
Declare Sub Mrar
Declare Sub Mrar12
Declare Sub Mrar34
A11 = 0
Rl1 = 10
Rl2 = 10
Ev1 = 0
Ev2 = 0
S1 = 1
'*****************************************************************
Main:
Locate 1 , 1
Lcd "alrm zon"
Locate 2 , 1
Lcd "flt zon"
```

```
'*****************************************************************
Do
Cursor Off Noblink
Start Adc
W = Getadc(0)
W = W / 2
X = Getadc(1)
X = X / 2
Y = Getadc(2)
Y = Y / 2
Z = Getadc(3)
Z = Z / 2
'*****************************************************************
If W > 2 And W < 70 Then
F11 = 0
F12 = 0
Else
If W >= 70 Then
F11 = 1
F12 = 0
Else
If W <= 2 Then
F11 = 0
F12 = 1
End If
End If
End If

If C15 = 1 Then
F11 = 0
```

```
F12 = 0
Else
If C11 = 1 Then
F11 = 1
F12 = 0
Else
If C12 = 1 Then
F11 = 0
F12 = 1
End If
End If
End If

If F11 = 1 Then
Portd.5 = 1
Locate 1 , 10
Lcd "1"
Waitms 10
Portd.5 = 0
Waitms 10
Portd.5 = 1
End If

If F12 = 1 Then
Portd.0 = 1
Locate 2 , 9
Lcd "1"
Waitms 10
Portd.0 = 0
Waitms 10
```

```
Portd.0 = 1
End If

If F11 > S11 Then
Ad1 = 1
Else
Ad1 = 0
End If

If Ad1 = 1 And S1 = 1 Then
Print "ATE0"
Waitms 500
Print "AT+CMGF=1"
Waitms 500
Print "AT+CMGS=" ; Chr(34) ; "0900902402" ; Chr(34) ; Chr(13)
Waitms 500
Print "alarm detected in zoon one" ; Chr(26)
End If

If F12 > S12 Then
Fd1 = 1
Else
Fd1 = 0
End If
If Fd1 = 1 And S1 = 1 Then
Print "ATE0"
Waitms 500
Print "AT+CMGF=1"
Waitms 500
Print "AT+CMGS=" ; Chr(34) ; "0900902402" ; Chr(34) ; Chr(13)
```

```
Waitms 500
Print "Fault detected in zoon one" ; Chr(26)
End If
'*****************************************************************
Debounce Mnu , 1 , Pass1 , Sub
Debounce Rst , 1 , Rset , Sub
Debounce Up1 , 1 , Tstled , Sub
Debounce Dn3 , 1 , Res , Sub
Debounce Mrr , 1 , Mrar , Sub
'*****************************************************************
If X > 2 And X < 70 Then
F21 = 0
F22 = 0
Else
If X >= 70 Then
F21 = 1
F22 = 0
Else
If X <= 2 Then
F21 = 0
F22 = 1
End If
End If
End If

If C25 = 1 Then
F21 = 0
F22 = 0
Else
If C21 = 1 Then
```

```
F21 = 1
F22 = 0
Else
If C22 = 1 Then
F21 = 0
F22 = 1
End If
End If
End If

If F21 = 1 Then
Portd.6 = 1
Locate 1 , 12
Lcd "2"
Waitms 500
Portd.6 = 0
Waitms 500
Portd.6 = 1
End If

If F22 = 1 Then
Portd.2 = 1
Locate 2 , 11
Lcd "2"
Waitms 500
Portd.2 = 0
Waitms 500
Portd.2 = 1
End If
```

```
If F21 > S21 Then
Ad2 = 1
Else
Ad2 = 0
End If

If Ad2 = 1  Then
Print "ATE0"
Waitms 500
Print "AT+CMGF=1"
Waitms 500
Print "AT+CMGS=" ; Chr(34) ; "0900902402" ; Chr(34) ; Chr(13)
Waitms 500
Print "alarm detected in zoon two" ; Chr(26)
End If

If F22 > S22 Then
Fd2 = 1
Else
Fd2 = 0
End If

If Fd2 = 1 And S1 = 1 Then
Print "ATE0"
Waitms 500
Print "AT+CMGF=1"
Waitms 500
Print "AT+CMGS=" ; Chr(34) ; "0900902402" ; Chr(34) ; Chr(13)
Waitms 500
Print "Fault detected in zoon two" ; Chr(26)
```

```
End If
'*****************************************************************
Debounce Mnu , 1 , Pass1 , Sub
Debounce Rst , 1 , Rset , Sub
Debounce Up1 , 1 , Tstled , Sub
Debounce Dn3 , 1 , Res , Sub
Debounce Mrr , 1 , Mrar , Sub
'*****************************************************************
If Y > 2 And Y < 70 Then
F31 = 0
F32 = 0
Else
If Y >= 70 Then
F31 = 1
F32 = 0
Else
If Y <= 2 Then
F31 = 0
F32 = 1
End If
End If
End If

If C35 = 1 Then
F31 = 0
F32 = 0
Else
If C31 = 1 Then
F31 = 1
F32 = 0
```

```
Else
If C32 = 1 Then
F31 = 0
F32 = 1
End If
End If
End If

If F31 = 1 Then
Portd.7 = 1
Locate 1 , 14
Lcd "3"
Waitms 500
Portd.7 = 0
Waitms 500
Portd.7 = 1
End If
If F32 = 1 Then
Portd.3 = 1
Locate 2 , 13
Lcd "3"
Waitms 500
Portd.3 = 0
Waitms 500
Portd.3 = 1
End If

If F31 > S31 Then
Ad3 = 1
Else
```

```
Ad3 = 0
End If
If Ad3 = 1 And S1 = 1 Then
Print "ATE0"
Waitms 500
Print "AT+CMGF=1"
Waitms 500
Print "AT+CMGS=" ; Chr(34) ; "0900902402" ; Chr(34) ; Chr(13)
Waitms 500
Print "ALARM detected in zoon THREE" ; Chr(26)
End If

If F32 > S32 Then
Fd3 = 1
Else
Fd3 = 0
End If

If Fd3 = 1 And S1 = 1 Then
Print "ATE0"
Waitms 500
Print "AT+CMGF=1"
Waitms 500
Print "AT+CMGS=" ; Chr(34) ; "0900902402" ; Chr(34) ; Chr(13)
Waitms 500
Print "FAULT detected in zoon THREE" ; Chr(26)
End If
'************************************************************
Debounce Mnu , 1 , Pass1 , Sub
Debounce Rst , 1 , Rset , Sub
```

```
Debounce Up1 , 1 , Tstled , Sub
Debounce Dn3 , 1 , Res , Sub
Debounce Mrr , 1 , Mrar , Sub
'*************************************************************
If Z > 2 And Z < 70 Then
F41 = 0
F42 = 0
Else
If Z >= 70 Then
F41 = 1
F42 = 0
Else
If Z <= 2 Then
F41 = 0
F42 = 1
End If
End If
End If

If C45 = 1 Then
F41 = 0
F42 = 0
Else
If C41 = 1 Then
F41 = 1
F42 = 0
Else
If C42 = 1 Then
F41 = 0
F42 = 1
```

```
End If
End If
End If

If F41 = 1 Then
Portb.7 = 1
Locate 1 , 16
Lcd "4"
Waitms 500
Portb.7 = 0
Waitms 500
Portb.7 = 1
End If

If F42 = 1 Then
Portd.4 = 1
Locate 2 , 15
Lcd "4"
Waitms 500
Portd.4 = 0
Waitms 10
Portd.4 = 1
End If

If F41 > S41 Then
Ad4 = 1
Else
Ad4 = 0
End If
```

```
If Ad4 = 1 And S1 = 1 Then
Print "ATE0"
Waitms 500
Print "AT+CMGF=1"
Waitms 500
Print "AT+CMGS=" ; Chr(34) ; "0900902402" ; Chr(34) ; Chr(13)
Waitms 500
Print "ALARM detected in zoon FOUR" ; Chr(26)
End If

If F42 > S42 Then
Fd4 = 1
Else
Fd4 = 0
End If
If Fd4 = 1 And S1 = 1 Then
Print "ATE0"
Waitms 500
Print "AT+CMGF=1"
Waitms 500
Print "AT+CMGS=" ; Chr(34) ; "0900902402" ; Chr(34) ; Chr(13)
Waitms 500
Print "FAULT detected in zoon FOUR" ; Chr(26)
End If
'*****************************************************************
Debounce Mnu , 1 , Pass1 , Sub
Debounce Rst , 1 , Rset , Sub
Debounce Up1 , 1 , Tstled , Sub
Debounce Dn3 , 1 , Res , Sub
Debounce Mrr , 1 , Mrar , Sub
```

```
'*****************************************************************
If F11 = 0 And F12 = 0 And F21 = 0 And F22 = 0 And F31 = 0 And F32 = 0
And F41 = 0 And F42 = 0 Then
Porta.4 = 1
Else
Porta.4 = 0
End If

If F11 = 1 And Md = 0 Then
Z1a = 1
Else
Z1a = 0
End If
If F21 = 1 And Md = 0 Then
Z2a = 1
Else
Z2a = 0
End If
If F31 = 1 And Md = 0 Then
Z3a = 1
Else
Z3a = 0
End If
If F41 = 1 And Md = 0 Then
Z4a = 1
Else
Z4a = 0
End If
'*****************************************************************
If F11 = 1 And C11 = 0 Then
```

```
Porta.6 = 1
Else
If F21 = 1 And C21 = 0 Then
Porta.6 = 1
Else
If F31 = 1 And C31 = 0 Then
Porta.6 = 1
Else
If F41 = 1 And C41 = 0 Then
Porta.6 = 1
Else
Porta.6 = 0
End If
End If
End If
End If
'*************************************************************
If F12 = 1 And C12 = 0 Then
Porta.7 = 1
Else
If F22 = 1 And C22 = 0 Then
Porta.7 = 1
Else
If F32 = 1 And C32 = 0 Then
Porta.7 = 1
Else
If F42 = 1 And C42 = 0 Then
Porta.7 = 1
Else
Porta.7 = 0
```

```
End If
End If
End If
End If
'*******************************************************
If Z1a = 1 And Z2a = 1 Then
X12 = 1
Else
X12 = 0
End If
If X12 = 1 Or M12 = 1 Then
X125 = 1
Else
X125 = 0
End If

If X125 = 1 And Ev1 = 0 Then
Porta.5 = 1
Cls
Locate 1 , 1
Lcd "fire dtect z1/z2"
Locate 2 , 1
Lcd "GAS/LEQ REL:"
Lcd Rl1
Waitms 2000
Decr Rl1
End If

If X125 = 1 And Rl1 = 0 Then
Portc.1 = 1
```

```
Gr12 = 1
Ev1 = 1
Locate 2 , 1
Lcd "GAS/LEQ RELESED"
Waitms 500
Rl1 = 10
Else
If Z1a = 0 Or Z2a = 0 And M12 = 0 Then
Rl1 = 10
Porta.5 = 0
End If
End If
'*****************************************************************
Debounce Mnu , 1 , Pass1 , Sub
Debounce Rst , 1 , Rset , Sub
Debounce Up1 , 1 , Tstled , Sub
Debounce Dn3 , 1 , Res , Sub
Debounce Mrr , 1 , Mrar , Sub
'*****************************************************************
If X12 = 1 And G12 = 0 And S1 = 1 Then
Print "ATE0"
Waitms 500
Print "AT+CMGF=1"
Waitms 500
Print "AT+CMGS=" ; Chr(34) ; "0900902402" ; Chr(34) ; Chr(13)
Waitms 500
Print "ALARM IN ZOON ONE AND TWO IN AUTO,COWNT DOWN
STARTED TO RELEASE GASL/LEQUED" ; Chr(26)
End If
If M12 = 1 And L12 = 0 And S1 = 1 Then
```

```
Print "ATE0"
Waitms 500
Print "AT+CMGF=1"
Waitms 500
Print "AT+CMGS=" ; Chr(34) ; "0900902402" ; Chr(34) ; Chr(13)
Waitms 500
Print "MANUAL ALARM ZOON ONE AND TWO IN AUTO,COWNT
DOWN STARTED TO RELEASE GASL/LEQUED" ; Chr(26)
End If
'*****************************************************************
If Z3a = 1 And Z4a = 1 Then
X34 = 1
Else
X34 = 0
End If
If X34 = 1 Or M34 = 1 Then
X345 = 1
Else
X345 = 0
End If
If X345 = 1 And Ev2 = 0 Then
Porta.5 = 1
Cls
Locate 1 , 1
Lcd "fire dtect z3/z4"
Locate 2 , 1
Lcd "GAS/LEQ REL:"
Lcd Rl2
Waitms 2000
Decr Rl2
```

```
End If
If X345 = 1 And Rl2 = 0 Then
Portc.0 = 1
Gr34 = 1
Ev2 = 1
Locate 2 , 1
Lcd "GAS/LEQ,,RELESED"
Waitms 500
Rl2 = 10
Porta.5 = 0
Else
If Z3a = 0 Or Z4a = 0 And M34 = 0 Then
Rl2 = 10
End If
End If

If X34 = 1 And G34 = 0 And S1 = 1 Then
Print "ATE0"
Waitms 500
Print "AT+CMGF=1"
Waitms 500
Print "AT+CMGS=" ; Chr(34) ; "0900902402" ; Chr(34) ; Chr(13)
Waitms 500
Print "ALARM IN ZOON THREE AND FOUR IN AUTO,COWNT DOWN
STARTED TO RELEASE GASL/LEQUED" ; Chr(26)
End If
'*****************************************************************
Debounce Mnu , 1 , Pass1 , Sub
Debounce Rst , 1 , Rset , Sub
Debounce Up1 , 1 , Tstled , Sub
```

```
Debounce Dn3 , 1 , Res , Sub
Debounce Mrr , 1 , Mrar , Sub
'*************************************************************
If M34 = 1 And L34 = 0 And S1 = 1 Then
Print "ATE0"
Waitms 500
Print "AT+CMGF=1"
Waitms 500
Print "AT+CMGS=" ; Chr(34) ; "0900902402" ; Chr(34) ; Chr(13)
Waitms 500
Print "MANUAL ALARM ZOON THREE AND FOUR IN AUTO,COWNT
DOWN STARTED TO RELEASE GASL/LEQUED" ; Chr(26)
End If

If Gr12 = 1 And Grp12 = 0 And S1 = 1 Then
Print "ATE0"
Waitms 500
Print "AT+CMGF=1"
Waitms 500
Print "AT+CMGS=" ; Chr(34) ; "0900902402" ; Chr(34) ; Chr(13)
Waitms 500
Print "ZOON ONE AND TWO GAS OR LEQUID EVACUATED" ; Chr(26)
End If

If Gr34 = 1 And Grp34 = 0 Then
Print "ATE0"
Waitms 500
Print "AT+CMGF=1"
Waitms 500
Print "AT+CMGS=" ; Chr(34) ; "0900902402" ; Chr(34) ; Chr(13)
```

```
Waitms 500
Print "ZOON THREE AND FOUR GAS OR LEQUID EVACUATED" ;
Chr(26)
End If
'*****************************************************************
If Md = 1 Then
Porta.4 = 1
Waitms 10
Porta.4 = 0
Debounce Mnu , 1 , Pass1 , Sub
Debounce Rst , 1 , Rset , Sub
Debounce Up1 , 1 , Tstled , Sub
Debounce Dn3 , 1 , Res , Sub
Debounce Mrr , 1 , Mrar , Sub
End If

S11 = F11
S12 = F12
S21 = F21
S22 = F22
S31 = F31
S32 = F32
S41 = F41
S42 = F42
G12 = X12
L12 = M12
G34 = X34
L34 = M34
Grp12 = Gr12
Grp34 = Gr34
```

```
Loop
'*****************************************************************
Sub Rset
Cls
Portd.0 = 0
Portd.2 = 0
Portd.3 = 0
Portd.4 = 0
Portd.5 = 0
Portd.6 = 0
Portd.7 = 0
Porta.7 = 0
Porta.6 = 0
Porta.5 = 0
Porta.4 = 0
Portb.7 = 0
M12 = 0
M34 = 0
If Z1a = 0 Or Z2a = 0 Then
Portc.1 = 0
Ev1 = 0
Else
If Z3a = 0 Or Z4a = 0 Then
Portc.0 = 0
Ev2 = 0
End If
End If
Goto Main
End Sub
'*****************************************************************
```

```
Sub Tstled
Cls
Portd.0 = 1
Portd.2 = 1
Portd.3 = 1
Portd.4 = 1
Portd.5 = 1
Portd.6 = 1
Portd.7 = 1
Portb.7 = 1
Porta.4 = 1
Portd.0 = 1
Waitms 5000
Call Rset
Goto Main
End Sub
'************************************************************
Sub Pass1:
Cls
Co = 0
Do
Locate 1 , 1
Lcd "Inter password :"
Debounce Up1 , 1 , Pass2 , Sub
Debounce Rt2 , 1 , Pass3 , Sub
Debounce Dn3 , 1 , Pass3 , Sub
Debounce Lf4 , 1 , Pass3 , Sub
Debounce Mnu , 1 , Gmn , Sub
Debounce Rst , 1 , Gmn , Sub
Incr Co
```

```
Loop Until Co = 500
Cls
Goto Main
End Sub
'*****************************************************************
Sub Gmn:
Goto Main
End Sub
Return
'*****************************************************************
Sub Rturn:
Co = 1
End Sub
Return
'*****************************************************************
Sub Pass2:
Co = 2
Do
Locate 1 , 1
Lcd "Inter password :"
Locate 2 , 1
Lcd "*"
Debounce Up1 , 1 , Pass5 , Sub
Debounce Rt2 , 1 , Pass4 , Sub
Debounce Dn3 , 1 , Pass5 , Sub
Debounce Lf4 , 1 , Pass5 , Sub
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
Incr Co
Loop Until Co = 500
```

```
Cls
Goto Main
End Sub
'*****************************************************************
Sub Pass3:
Co = 0
Do
Locate 1 , 1
Lcd "Inter password :"
Locate 2 , 1
Lcd "*"
Debounce Up1 , 1 , Pass5 , Sub
Debounce Rt2 , 1 , Pass5 , Sub
Debounce Dn3 , 1 , Pass5 , Sub
Debounce Lf4 , 1 , Pass5 , Sub
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
Incr Co
Loop Until Co = 500
Cls
Goto Main
End Sub
'*****************************************************************
Sub Pass4:
Co = 0
Do
Locate 1 , 1
Lcd "Inter password :"
Locate 2 , 2
Lcd "*"
```

```
Debounce Up1 , 1 , Pass7 , Sub
Debounce Rt2 , 1 , Pass7 , Sub
Debounce Dn3 , 1 , Pass6 , Sub
Debounce Lf4 , 1 , Pass7 , Sub
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
Incr Co
Loop Until Co = 500
Cls
Goto Main
End Sub
'************************************************************
Sub Pass5:
Co = 0
Do
Locate 1 , 1
Lcd "Inter password :"
Locate 2 , 2
Lcd "*"
Debounce Up1 , 1 , Pass7 , Sub
Debounce Rt2 , 1 , Pass7 , Sub
Debounce Dn3 , 1 , Pass7 , Sub
Debounce Lf4 , 1 , Pass7 , Sub
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
Incr Co
Loop Until Co = 500
Cls
Goto Main
End Sub
```

```
'*****************************************************************
Sub Pass6:
Co = 0
Do
Locate 1 , 1
Lcd "Inter password :"
Locate 2 , 3
Lcd "*"
Debounce Up1 , 1 , Pass9 , Sub
Debounce Rt2 , 1 , Pass9 , Sub
Debounce Dn3 , 1 , Pass9 , Sub
Debounce Lf4 , 1 , Pass8 , Sub
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
Incr Co
Loop Until Co = 500
'Co = 19
Goto Main
End Sub
'*****************************************************************
Sub Pass7:
Co = 0
Do
Locate 1 , 1
Lcd "Inter password :"
Locate 2 , 3
Lcd "*"
Debounce Up1 , 1 , Pass9 , Sub
Debounce Rt2 , 1 , Pass9 , Sub
Debounce Dn3 , 1 , Pass9 , Sub
```

```
Debounce Lf4 , 1 , Pass9 , Sub
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
Incr Co
Loop Until Co = 500
Cls
Goto Main
End Sub
'****************************************************************
Sub Pass8:
Co = 0
Locate 1 , 1
Lcd "Inter password :"
Locate 2 , 4
Lcd "*"
Waitms 20
Do
Cls
Locate 2 , 1
Lcd "correct password"
Waitms 100
Cls
Waitms 100
Incr Co
Loop Until Co = 3
Goto Menu
Cls
Goto Main
End Sub
'****************************************************************
```

```
Sub Pass9:
Co = 0
Locate 1 , 1
Lcd "Inter password :"
Locate 2 , 4
Lcd "*"
Waitms 20
Do
Cls
Locate 2 , 1
Lcd "wrong password"
Waitms 100
Cls
Waitms 100
Incr Co
Loop Until Co = 3
Cls
Goto Main
End Sub
'*****************************************************************
Sub Menu:
Cls
Co = 0
Do
Locate 1 , 1
Lcd "MENU:"
Locate 1 , 6
Lcd "Actuater set=Up1"
Locate 2 , 1
Lcd "SMS=Rt2,,"
```

```
Locate 2 , 10
Lcd "MNT=Lf3"
Debounce Up1 , 1 , Acst , Sub
Debounce Rt2 , 1 , Sms , Sub
Debounce Dn3 , 1 , Mntm , Sub
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
Incr Co
Loop Until Co = 900
Cls
Goto Main
End Sub
'*****************************************************************
Sub Res:
Cls
Co = 0
Do
 Cursor Off Noblink
Start Adc
W = Getadc(0)
W = W / 2
X = Getadc(1)
X = X / 2
Y = Getadc(2)
Y = Y / 2
Z = Getadc(3)
Z = Z / 2
Locate 1 , 1
Lcd "z1 = "
Lcd W
```

```
Lcd ",,z2 = "
Lcd X
Locate 2 , 1
Lcd "Z3 = "
Lcd Y
Lcd ",,Z4 = "
Lcd Z
Debounce Mnu , 1 , Gmn , Sub
Debounce Rst , 1 , Gmn , Sub
Waitms 500
Incr Co
Loop Until Co = 1000
Co = 0
Cls
Goto Main
End Sub
'*****************************************************************
Sub Sms
Cls
Co = 2
Do
Locate 1 , 1
Lcd "sms sending ="
Locate 2 , 1
Lcd "Up1=on,,"
Lcd "Rt2=off"
Debounce Up1 , 1 , Smon , Sub
Debounce Rt2 , 1 , Smof , Sub
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
```

```
If Co = 1 Then
Goto Menu
End If
Incr Co
Loop Until Co = 500
Co = 0
Cls
Goto Main
End Sub
'*****************************************************************
Sub Smon
Cls
Co = 2
Do
Locate 2 , 1
Lcd "sms setting = on"
S1 = 1
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Sms
End If
Incr Co
Loop Until Co = 500
Cls
Goto Main
End Sub
'*****************************************************************
Sub Smof
Cls
```

```
Co = 2
Do
Locate 2 , 1
Lcd "sms setting = off"
S1 = 0
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Sms
End If
Incr Co
Loop Until Co = 500
Cls
Goto Main
End Sub
'*****************************************************************
Sub Acst
Cls
Co = 2
Do
Locate 1 , 1
Lcd "Automatic = Up1"
Locate 1 , 2
Lcd "Manual  = Rt2"
Debounce Up1 , 1 , Acsa , Sub
Debounce Rt2 , 1 , Acsm , Sub
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Menu
```

```
End If
Incr Co
Loop Until Co = 500
Co = 0
Cls
Goto Main
End Sub
'*************************************************************
Sub Mntm
Cls
Co = 2
Do
Locate 1 , 1
Lcd "Maintenace mod :"
Locate 2 , 1
Lcd "Up1=ON,,"
Lcd "Rt2=OFF"
Debounce Up1 , 1 , Mmon , Sub
Debounce Rt2 , 1 , Mmof , Sub
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Menu
End If
Incr Co
Loop Until Co = 500
Co = 0
Cls
Goto Main
End Sub
```

```
'*****************************************************************
Sub Mmon
Cls
Co = 2
Do
Locate 1 , 1
Lcd "Mntenance mod ON"
C0 = 1
Waitms 1000
Goto Mmzs
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Mntm
End If
Incr Co
Loop Until Co = 500
Cls
Goto Main
End Sub
'*****************************************************************
Sub Mmof
Cls
Co = 2
Do
Locate 1 , 1
Lcd "Mntnance mod OFF"
C11 = 0
C12 = 0
C15 = 0
```

```
C21 = 0
C22 = 0
C25 = 0
C31 = 0
C32 = 0
C35 = 0
C41 = 0
C42 = 0
C45 = 0
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Mntm
End If
Incr Co
Loop Until Co = 500
Cls
Goto Main
End Sub
'****************************************************************
Sub Acsa
Cls
Co = 2
Do
Locate 1 , 1
Lcd "auto mod active"
Md = 0.
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
```

```
Goto Acst
End If
Incr Co
Loop Until Co = 500
Cls
Goto Main
End Sub
'*************************************************************
Sub Acsm
Cls
Co = 2
Do
Locate 1 , 1
Lcd "Manual mod activ"
Md = 1
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Acst
End If
Incr Co
Loop Until Co = 500
Cls
Goto Main
End Sub
'*************************************************************
Sub Mmzs
Cls
Co = 2
Do
```

```
Locate 1 , 1
Lcd "zn1=Up1,,"
Locate 1 , 10
Lcd "zn2=Rt2"
Locate 2 , 1
Lcd "zn3=Dn3,,"
Locate 2 , 10
Lcd "zn4=lf4"
Debounce Up1 , 1 , Mmz1 , Sub
Debounce Rt2 , 1 , Mmz2 , Sub
Debounce Dn3 , 1 , Mmz3 , Sub
Debounce Lf4 , 1 , Mmz4 , Sub
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Mmon
End If
Incr Co
Loop Until Co = 500
Co = 0
Cls
Goto Main
End Sub
'*****************************************************************
Sub Mmz1
Cls
Co = 2
Do
Locate 1 , 1
Lcd "zoo1:helthy=up1"
```

```
Locate 2 , 1
Lcd "alrm=Rt2,,"
Locate 2 , 10
Lcd "flt=Dn3"
Debounce Up1 , 1 , Fhz1 , Sub
Debounce Rt2 , 1 , Faz1 , Sub
Debounce Dn3 , 1 , Ffz1 , Sub
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Mmzs
End If
Incr Co
Loop Until Co = 500
Co = 0
Cls
Goto Main
End Sub
'****************************************************************
Sub Fhz1
Cls
Co = 2
Do
Locate 1 , 1
Lcd "zoon1 setting:"
Locate 2 , 1
Lcd "force helthy"
C15 = 1
C11 = 0
C12 = 0
```

```
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Mmz1
End If
Incr Co
Loop Until Co = 500
Cls
Goto Main
End Sub
'*****************************************************************
Sub Faz1
Cls
Co = 2
Do
Locate 1 , 1
Lcd "zoon1 setting:"
Locate 2 , 1
Lcd "force Alarm"
C11 = 1
C12 = 0
C15 = 0
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Mmz1
End If
Incr Co
Loop Until Co = 500
Cls
```

```
Goto Main
End Sub
'*****************************************************************
Sub Ffz1
Cls
Co = 2
Do
Locate 1 , 1
Lcd "zoon1 setting:"
Locate 2 , 1
Lcd "force fault"
C15 = 0
C11 = 0
C12 = 1
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Mmz1
End If
Incr Co
Loop Until Co = 500
Cls
Goto Main
End Sub
'*****************************************************************
Sub Mmz2
Cls
Co = 2
Do
Locate 1 , 1
```

```
Lcd "zoo2:helthy=up1"
Locate 2 , 1
Lcd "alrm=Rt2,,"
Locate 2 , 10
Lcd "flt=Dn3"
Debounce Up1 , 1 , Fhz2 , Sub
Debounce Rt2 , 1 , Faz2 , Sub
Debounce Dn3 , 1 , Ffz2 , Sub
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Mmzs
End If
Incr Co
Loop Until Co = 500
Co = 0
Cls
Goto Main
End Sub
'*****************************************************************
Sub Fhz2
Cls
Co = 2
Do
Locate 1 , 1
Lcd "zoon2 setting:"
Locate 2 , 1
Lcd "force helthy"
C25 = 1
C21 = 0
```

```
C22 = 0
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Mmz2
End If
Incr Co
Loop Until Co = 500
Cls
Goto Main
End Sub
'*************************************************************
Sub Faz2
Cls
Co = 2
Do
Locate 1 , 1
Lcd "zoon2 setting:"
Locate 2 , 1
Lcd "force Alarm"
C21 = 1
C22 = 0
C25 = 0
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Mmz2
End If
Incr Co
Loop Until Co = 500
```

```
Cls
Goto Main
End Sub
'*************************************************************
Sub Ffz2
Cls
Co = 2
Do
Locate 1 , 1
Lcd "zoon2 setting:"
Locate 2 , 1
Lcd "force fault"
C25 = 0
C21 = 0
C22 = 1
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Mmz2
End If
Incr Co
Loop Until Co = 500
Cls
Goto Main
End Sub
'*************************************************************
Sub Mmz3
Cls
Co = 2
Do
```

```
Locate 1 , 1
Lcd "zoon3:helthy=up1"
Locate 2 , 1
Lcd "alrm=Rt2,,"
Locate 2 , 10
Lcd "flt=Dn3"
Debounce Up1 , 1 , Fhz3 , Sub
Debounce Rt2 , 1 , Faz3 , Sub
Debounce Dn3 , 1 , Ffz3 , Sub
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Mmzs
End If
Incr Co
Loop Until Co = 500
Co = 0
Cls
Goto Main
End Sub
'*****************************************************************
Sub Fhz3
Cls
Co = 2
Do
Locate 1 , 1
Lcd "zoon3 setting:"
Locate 2 , 1
Lcd "force helthy"
C35 = 1
```

```
C31 = 0
C32 = 0
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Mmz3
End If
Incr Co
Loop Until Co = 500
Cls
Goto Main
End Sub
'*************************************************************
Sub Faz3
Cls
Co = 2
Do
Locate 1 , 1
Lcd "zoon3 setting:"
Locate 2 , 1
Lcd "force Alarm"
C31 = 1
C32 = 0
C35 = 0
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Mmz3
End If
Incr Co
```

```
Loop Until Co = 500
Cls
Goto Main
End Sub
'*****************************************************************
Sub Ffz3
Cls
Co = 2
Do
Locate 1 , 1
Lcd "zoon3 setting:"
Locate 2 , 1
Lcd "force fault"
C35 = 0
C31 = 0
C32 = 1
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Mmz3
End If
Incr Co
Loop Until Co = 500
Cls
Goto Main
End Sub
'*****************************************************************
Sub Mmz4
Cls
Co = 2
```

```
Do
Locate 1 , 1
Lcd "zoon4:helthy=up1"
Locate 2 , 1
Lcd "alrm=Rt2,,"
Locate 2 , 10
Lcd "flt=Dn3"
Debounce Up1 , 1 , Fhz4 , Sub
Debounce Rt2 , 1 , Faz4 , Sub
Debounce Dn3 , 1 , Ffz4 , Sub
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Mmzs
End If
Incr Co
Loop Until Co = 500
Co = 0
Cls
Goto Main
End Sub
'*************************************************************
Sub Fhz4
Cls
Co = 2
Do
Locate 1 , 1
Lcd "zoon4 setting:"
Locate 2 , 1
Lcd "force helthy"
```

```
C45 = 1
C41 = 0
C42 = 0
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Mmz4
End If
Incr Co
Loop Until Co = 500
Cls
Goto Main
End Sub
'*****************************************************************
Sub Faz4
Cls
Co = 2
Do
Locate 1 , 1
Lcd "zoon4 setting:"
Locate 2 , 1
Lcd "force Alarm"
C41 = 1
C42 = 0
C45 = 0
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Mmz4
End If
```

```
Incr Co
Loop Until Co = 500
Cls
Goto Main
End Sub
'*****************************************************************
Sub Ffz4
Cls
Co = 2
Do
Locate 1 , 1
Lcd "zoon4 setting:"
Locate 2 , 1
Lcd "force fault"
C45 = 0
C41 = 0
C42 = 1
Debounce Mnu , 1 , Rturn , Sub
Debounce Rst , 1 , Gmn , Sub
If Co = 1 Then
Goto Mmz4
End If
Incr Co
Loop Until Co = 500
Cls
Goto Main
End Sub
'*****************************************************************
Sub Mrar
Cls
```

```
Co = 2
Do
Locate 1 , 1
Lcd "Manual relese AC:"
Locate 2 , 1
Lcd "Z12=up1,,"
Locate 2 , 10
Lcd "z34=Dn3"
Debounce Up1 , 1 , Mrar12 , Sub
Debounce Dn3 , 1 , Mrar34 , Sub
Debounce Mnu , 1 , Gmn , Sub
Debounce Rst , 1 , Gmn , Sub
Incr Co
Loop Until Co = 500
Cls
Goto Main
End Sub
'*************************************************************
Sub Mrar12
Cls
Co = 2
Do
Locate 1 , 1
Lcd "zon 1/2 selected"
M12 = 1
Debounce Mnu , 1 , Gmn , Sub
Debounce Rst , 1 , Gmn , Sub
Incr Co
Loop Until Co = 500
Cls
```

```
Goto Main
End Sub
'*************************************************************
Sub Mrar34
Cls
Co = 2
Do
Locate 1 , 1
Lcd "zon 3/4 selected"
M34 = 1
Debounce Mnu , 1 , Gmn , Sub
Debounce Rst , 1 , Gmn , Sub
Incr Co
Loop Until Co = 500
Cls
Goto Main
End Sub
```