CHAPTER 1
INTRODUCTION

1.1 INTRODUCTION TO THE PROJECT

In today's world, women safety has become a major issue as they can't step out of their house at any given time due to physical/sexual abuse and a fear of violence.

Atrocities towards (and against) women are forms of oppression hindering the development of women and thereby resulting in gender injustice, this being ideologically supported by a value system, which is androcentric and gender insensitive. Deepening inequalities and struggles by the oppressed section to assert their rights (granted under democracy) have unleashed retaliations by the more privileged and, women situated as they are in the social matrix as non-free, dependent subjectivities, become specially affected ones.

Even in the 21st century where the technology is rapidly growing and new gadgets were developed but still women's and girls are facing problems. Even today in India, women can't move at night in secluded places and even at day time crowded places hundreds and thousands of incidents of physical/sexual abuse happening to every day women in this country. Among other crimes, rape is the fastest growing crime in the country today.

The status of women in India has gone through many great changes over the past few millennia. From equal status with men in ancient times through the low points of the medieval period to the promotion of equal rights by many reformers, the history of women in India has been eventful. In modern India, women have adorned high offices in India including that of the President, Prime Minister, Leader of the Opposition and Speaker of the LokSabha. However, women in India continue to face social challenges and are often victims of abuse and violent crimes and, according to a global poll conducted by Thomson Reuters, India is the "fourth most dangerous country" in the world for women, and the worst country for women among the G20 countries.

This project focuses on a security system that is designed solely to serve the purpose of providing security to women so that they never feel helpless while facing such social

challenges. The Delhi Nirbhaya case that triggered the whole nation was the greatest motivation for this system. It was high time we women needed a change.

1.2 ORGANIZATION OF THE PROJECT REPORT

The organization of the project is as follows:

Chapter 1: It gives the brief introduction of Multi-Function monitoring system in vehicle.

Chapter 2: The literature survey of the project is explained.

Chapter 3: The problem definition, aim and objective of the project are explained.

Chapter 4: Introduction to software tools of the project are explained.

Chapter 5: Advantages and Disadvantages of the proposed system.

Chapter 6: Implementation, result and conclusion of the project are explained.

The Conclusion and future scope of the project are explained.

The Reference Books of the Project.

The Source Code of the Implemented Project.

CHAPTER 2

LITERATURE SURVEY

LITERATURE SURVEY

2.1 EXISTING SYSTEM:

Keeping the same concern in mind many developers have come up with innovative applications. Few of such applications are as follows-

1. VithU app: This is an emergency app initiated by a popular Indian crime television series "Gumrah" aired on Channel in this app when the power button of the Smartphone is pressed twice consecutively, it will begin sending out alert messages with a link to the location of the user every two minutes to the contacts fed into the app.

2. SHE (Society Harnessing Equipment): It is a garment designed by three engineers from Chennai. This garment has an electric circuit that can generate 3800kv of current which can help the victim to escape. In case of multiple attacks, it can send up to 82 electric shocks. Since the fabric is bilayer, the user is not affected. It can also send emergency messages.
3. ILA security: The co-founders of this system, McGivern, James Phillips, and Neil Munn, have designed three personal alarms that can shock and disorient potential attackers and draw attention to dangerous situations.

4. USING Button: proposed system is to design a portable device which resembles a normal belt. It consists of Arduino Board and Button when it Pressed, the device will get activated automatically. Immediately the location of the victim will be tracked with the help of GPS and emergency messages will be sent to three contacts and one to police control room every two minutes with updated location. The screaming alarm unit will be activated and will send out sirens to call out for help.

CHAPTER 3

PROJECT DISCRIPTION

3.1 BLOCK DIAGRAM

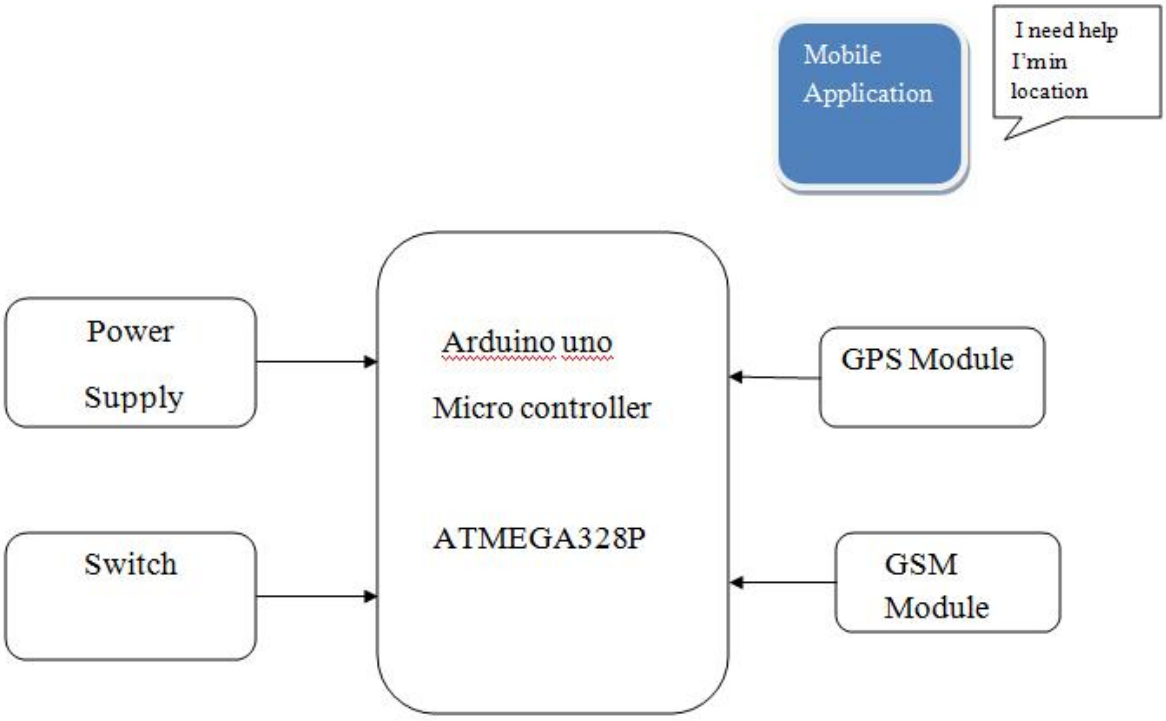

Fig3.1: Block Diagram

3.2 POWER SUPPLY:

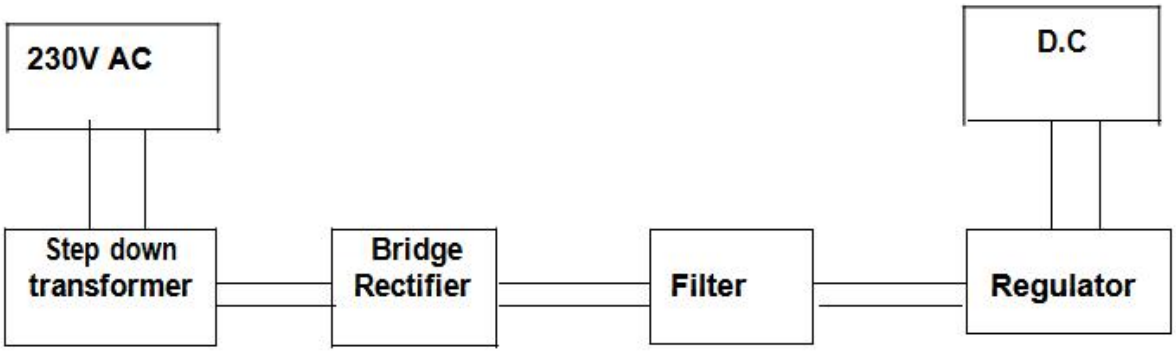

Fig3.2: Power Supply

3.2.1 Transformer:

Usually, D.C voltages are required to operate various electronic equipment and these voltages are 5V, 9V or 12V. But these voltages cannot be obtained directly. Thus the A.C input available at the mains supply i.e., 230V is to be brought down to the required voltage level. This is done by a transformer. Thus, a step down transformer is employed to decrease the voltage to a required level.

3.2.2 Rectifier:

The output from the transformer is fed to the rectifier. It converts A.C. into pulsating D.C. The rectifier may be a half wave or a full wave rectifier. In this project, a bridge rectifier is used because of its merits like good stability and full wave rectification.

3.2.3 Filter:

Capacitive filter is used in this project. It removes the ripples from the output of rectifier and smoothens the D.C. Output received from this filter is constant until the mains voltage and load is maintained constant. However, if either of the two is varied, D.C. voltage received at this point changes. Therefore, a regulator is applied at the output stage.

3.2.4 Voltage Regulator:

As the name itself implies, it regulates the input applied to it. A voltage regulator is an electrical regulator designed to automatically maintain a constant voltage level. In this project, power supply of 5V and 12V are required. In order to obtain these voltage levels, 7805 and 7812 voltage regulators are to be used. The first number 78 represents positive supply and the numbers 05, 12 represent the required output voltage levels.

3.3 INTRODUCTION TO IOT:

Internet of Things (IoT) comprises things that have unique identities and are connected to the Internet. While many existing devices, such as networked computers or 4G-enabled mobile phones, already have some form of unique identities and are also connected to the Internet, the focus on IoT is in the configuration, control and networking via the Internet of devices or "things" that are traditionally not associated with the internet. These include devices such as thermostats, utility meters, a Bluetooth connected headset, irrigation pumps and sensors, or control-circuits for an electric car's engine. Internet of Things is anew revolution in the capabilities of the end points that are connected to the Internet, and is driven by the advancement in capabilities (in combination with lower costs) in sensor networks, mobile devices, wireless communications, networking and cloud technologies. Experts forecast that by the year 2020 there will be a total of 50 billion devices/ things connected to the Internet. Therefore, the major industry players are excited by the prospects of new markets for their products. The products include hardware and software components for IoT endpoints, hubs, or control centers of the IoT universe.

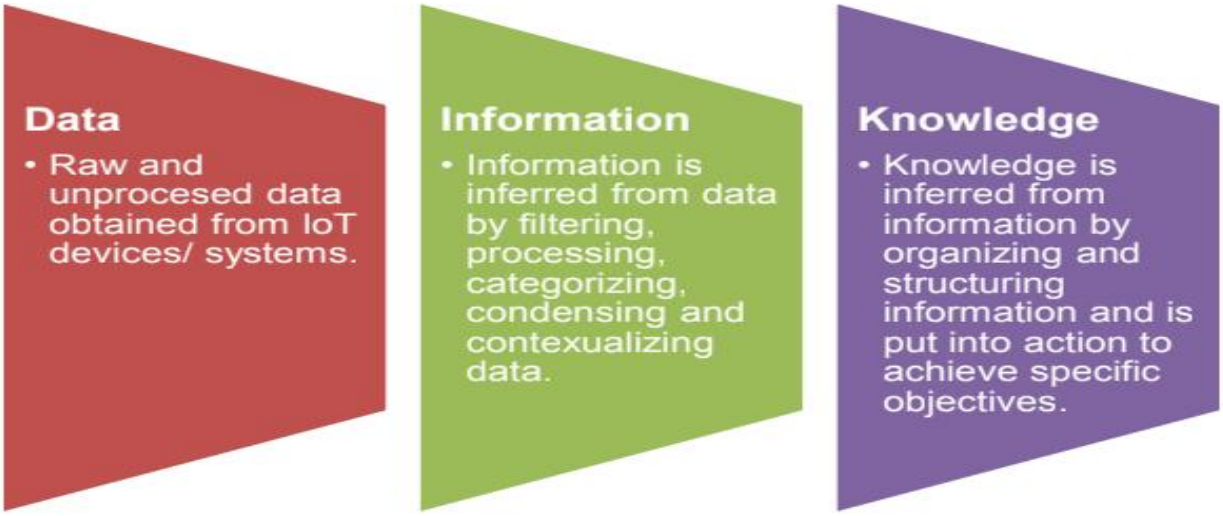

Fig3.3: Inferring information and knowledge from data

The scope of IoT is not limited to just connecting things (devices, appliances, machines) to the Internet. IoT allows these things to communicate and exchange data (Control& information, that could include data associated with users) while executing meaningful application towards a common user or machine goal. Data itself does not have a meaning until it is contextualized processed into useful information. Applications on IoT networks extract and create information from lower level data by filtering, processing, categorizing, condensing and contextualizing the data. This information obtained is then organized and structured to infer knowledge about the system and/or its users, its environment, and its operations and progress towards its objectives, allowing a smarter performance, as shown in Figure 1.1.For example, consider a series of raw sensor measurements ((72,45):(84,56)) generated by a weather monitoring station, which by themselves do not have any meaning or context. To give meaning to the data .a context is added. Which in this example can be that each tuple in data represents the temperature and humidity measured every minute .With this context added we know the meaning (or information) of the measured data tuples. Further information is obtained by categorizing .considering or processing this data .For example the average temperature and humidity readings for last five minutes is obtained by averaging the last five data tuples .The next step is to organize the information and understand the relationships between pieces of information to infer knowledge which can be put into action .For example, an alert is raised if the average temperature in last five minutes exceeds 120F,and this alert may be conditioned on the user's geographical position as well.

The applications of Internet of Things span a wide range of domains including (but not limited to) homes, cities, environment, energy, systems, retail, logistics, industry, agriculture and health as listed .For homes, IOT has several applications such as smart lighting that adapt the lighting to suit the ambient conditions .smart appliances that can be remotely monitored and controlled, intrusion detection systems, smart smoke detectors, etc. For cities, IOT has applications such as smart parking systems that provide status updates on available slots ,smart lighting that helps in saving energy ,smart roads that provide information on driving conditions and structural health monitoring systems .For environment ,IOT has applications such as weather monitoring , air and noise pollution , forest fire detection and river flood detection systems . For energy systems, IOT has applications such as including smart grids, grid integration of renewable energy sources and prognostic health management systems. For retail domain, IOT has applications such as inventory management, smart payments and smart vending machines. For agriculture domain, IOT has applications such as smart irrigation systems that help in saving water while enhancing productivity and green house control systems. Industrial applications such as smart irrigation systems. Industrial applications of IOT include machine diagnostics and prognosis systems. For health and lifestyle, IOT has applications such as health and fitness monitoring systems and wearable electronics.

3.3.1 Definition & Characteristics of IoT:

Definition: A dynamic global network infrastructure with self-configuring capabilities based on standard and interoperable communication protocols where physical and virtual “thing" have identities, physical attributes, and virtual personalities and use intelligent interfaces ,and are seamlessly integrated into the information network , often communicate data associated with users and their environments.

Let us examine this definition of IOT further to put some of the terms into perspective. Dynamic & Self-Adapting: IOT devices and systems may have the capability to dynamically adapt with the changing contexts and take actions based on their operating conditions. User’s context or sensed environment. For example, consider surveillance system compromising of a number of surveillance cameras. The surveillance cameras day or night. Cameras could switch from lower resolution modes when any motion is detected and alert nearby cameras to do the same. In this

example, the surveillance system is adapting itself based on the context and changing (e.g., dynamic) conditions. Self-Configuring: IOT devices may have self-configuring capability, allowing a large number of devices to work together to provide certain functionality (such as weather monitoring). These devices have the ability configure themselves (in association with the IOT infrastructure), setup the networking, and fetch latest software upgrades with minimal manual or user intervention. Interoperable Communication Protocols: IOT devices may support a number of interoperable communication protocols and communicate with other devices and also with the infrastructure. We describe some of the commonly used communication protocols and models in later sections.

3.3.2 Physical Design of IoT:

Things in IoT:

The "Things" in IOT usually refers to IOT devices which have unique identities and can perform remote sensing, actuating and monitoring capabilities. IOT devices can exchange data with other connected devices and applications (directly or indirectly), or collect data from other devices and process the data either locally or send the data to centralized servers or cloud-based. Application back-ends for processing the data, or perform some tasks locally and other tasks within the IOT infrastructure, based on temporal and space constraints (i.e., memory, processing capabilities, communication latencies and speeds, and deadlines). Figure1.3 shows a block diagram of a typical IOT device. An IOT device may consist of several interfaces for connections to other devices, both wired and wireless. These include (i) I/O interfaces for sensors, (ii) interfaces for internet connectivity, (iii) memory and storage interfaces and (iv) audio/video interfaces. An IOT device can collect various types of data from the on-board or attached sensors, such as temperature, humidity, light intensity. The sensed data can be connected actuators that allow them to interact with other physical entities (including non-IOT devices and systems) in the vicinity of the device. For example, a relay switch connected to an IOT device can turn an appliance on/off based on the commands sent to the IOT device over the Internet. IOT device can also be varied types, for instance, wearable sensors, smart watches, LED lights, auto mobiles and industrial machines, Almost all IOT devices generate data in some form or the other which when processed by data analytics systems leads to useful information to guide further actions locally or remotely. For instance, sensor data generated by a soil moisture

monitoring device in a garden, when processed can help in determining the optimum watering schedules.

3.3.3 IoT Protocols:

Link Layer Link Layer protocols determine how the data is physically sent over the network's physical layer or medium (e. g., copper wire, coaxial a cable, or a radio wave). The scope of the link layer is the network connection to which host is attached. Hosts on the same link exchange data packets over the link layer using link layer protocols. Link layer determines how the packets are coded and signaled by the hardware device over the medium to which the host is attached (such as a coaxial cable). Let us now look at some link layer protocols which are relevant in the context of IOT.

• 802.3-ETHERNET: IEEE 802.3 is a collection of wired Ethernet standards for the link layer. For example, 802.3 is the standard for 10BASIES Ethernet that uses coaxial cable as a shared medium, 802.3.i is the standard for 10BASE-T Ethernet over copper twisted-pair connections, 802.3ae is the standard for 10Gbit/Ethernet over fiber, and so on. optic connections, These standards provide data rates from 10Mbs to40Gbs and higher. The shared medium in Ethernet can be a coaxial cable, twisted-pair wire or an optical fiber. The shared medium (i. e., broadcast medium)carries the communication for all the devices propagation conditions and transceiver capabilities. The specifications of the 802.3 standards are available on the IEEE802.3 working group website.

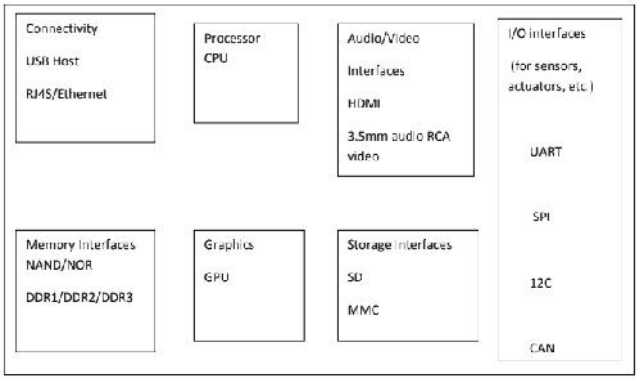

Fig3.4: Generic block diagram of IOT Device

3.3.4 Logical Design of IoT:

Logical design of an IoT system refers to an abstract representation of the entities and processes without going into the low-leve; specifies of the implementation. In this section we describe the functional blocks of an IoT system and the communication APIs that are used for the examples in this book. The steps in logical design are described in additional detail in Chapter-5.

3.3.5 IoT Functional Blocks:

An IoT system comprises of a number of functional blocks that provide the system the capabilities for identification, sensing, actuation, and management as shown in Figure 1.6.
These functional blocks are described as follows:

• Device: An IoT system comprises of devices that provide sensing, actuation, monitoring and control functions. You learned about IoT devices in section 1.2.

• Communication: The communication block handles the communication for the Iot system. You learned about various protocols used for communication by IoT systems in section 1.2.

• Services: An IoT system uses various types of IoT services such as services for device monitoring, device control services, data publishing services and services for device discovery.

• Management: Management functional block provides various functions to govern the IoT system.

• Security: Security functional block secures the IoT system and by providing functions such as authentication, authorization, message and content integrity, and data security.

• Application: IoT applications provide an interface that the users can use to control and monitor various aspects of the IoT system. Applications also allow users to view the system status and view or analyze the processed data.

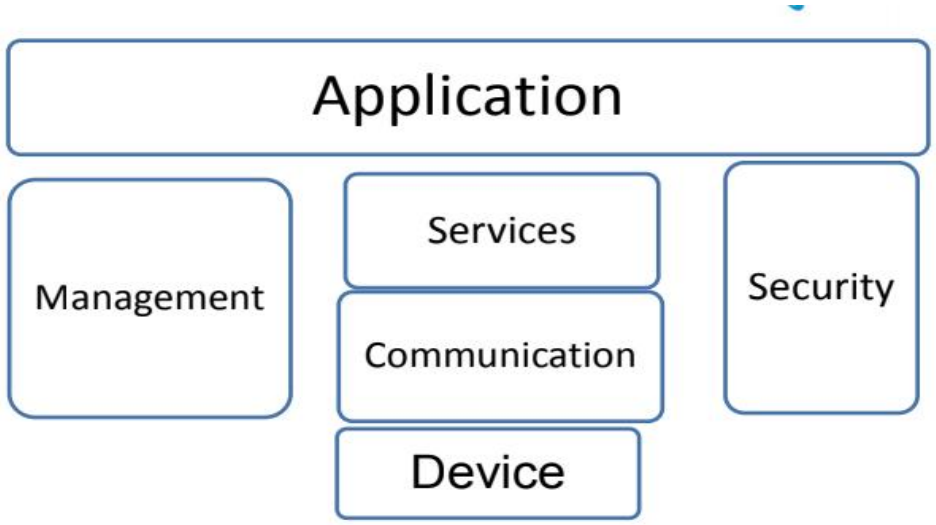

Fig3.5: Functional Blocks of IoT

Request-Response: Request-Response is a communication model in which the client sends request to the server and the server responds to the requests. When the server receives a request, it decided how to respond, fetches the data, retrieves resource representations, prepares the response, and then sends the response to the client. Request- Response model is a stateless communication model and each request-response pair is independent of others. Figure1.7 shows the client-server interactions in the request-response model.

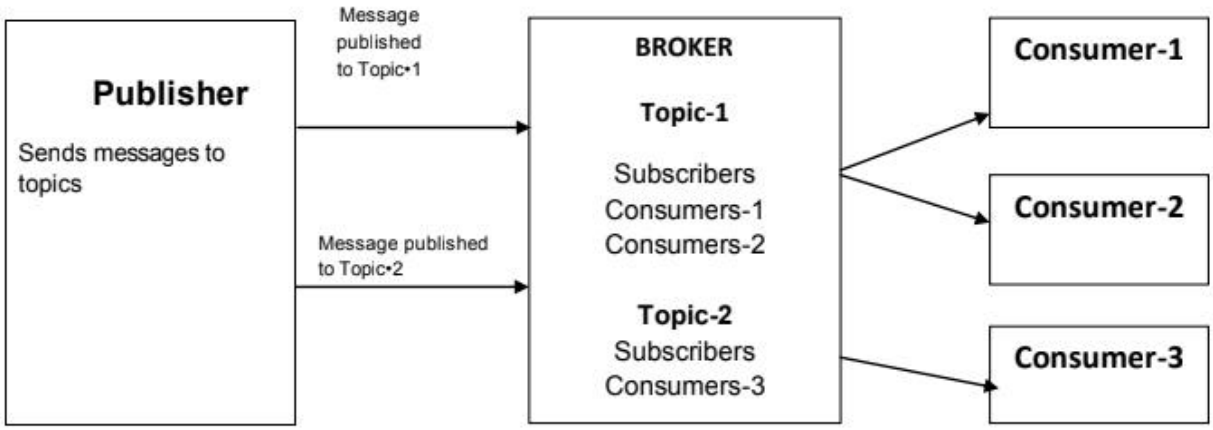

Fig3.6: Publish-Subscribe communication model

Publish-Subscribe: Publish-Subscribe is a communication model that involves publishers, brokers and consumers. Publishers are the source of data. Publishers send the data to the topics which are managed by the broker. Publishers are not aware of the consumers. Consumers subscribe to the topics which are managed by the broker. When the broker receives data for a topic from the publisher, it sends the data to all the subscribed consumers. Figure 1.8 shows the publisher-broker-consumer interactions in the publish-subscribe model.

• **Push-Pull:** Push-Pull is a communication model in which the data producers push the data to queues and the consumers pull the data from the queues. Producers do not need to be aware of the consumers. Queues help in decoupling the messaging between the producers and consumers. Queues also act as a buffer which helps in situations when there is a mismatch between the rate at which the producers push data and the rate at which the consumers pull data. Figure 1.9 shows the publisher- queue-consumer interactions in the push-pull model.

• **Exclusive Pair:** Exclusive Pair is a bi-directional, fully duplex communication model that uses a persistent connection between the client and server. Once the connection is setup it remains open until the client sends a request to close the connection. Client and server can send messages to each other after connection setup. Exclusive pair is a stateful communication model and the server is aware of all the open connections. Figure 1.10 shows the client -server interactions in the exclusive pair model.

3.4 IOT ENABLING TECHNOLOGIES:

IOT is enabled by several technologies including wireless sensor networks, cloud computing, Big Data analytics, embedded systems, security protocols and architectures, communication protocols, web services, mobile Internet and semantic search engines. This section provides an overview of some of these technologies which play a key role in IoT Wireless sensor networks: A wireless Sensor Network (WSN) comprises of distributed devices with Sensors which are used to monitor the environmental and physical conditions. a WSN consist of a number of end-nodes and routers and a coordinator. The coordinator collects the data from all the nodes.

Coordinator also acts as a gateway that connects the WSN to the Internet. Some examples are WSNs used in IoT systems are described as follows:

• Weather monitoring systems use WSNs in which the nodes collect temperature, humidity and other data, which is aggregated and analyzed
• Indoor air quality monitoring systems use WSNs to collect data on the indoor air quality and concentration of various gaseous
• Soil moisture monitoring systems use WSNs to monitor soil moisture at various locations
• Surveillance systems use WSNs for collecting surveillance data (such as motion detection data)
• Smart grids use WSNs for monitoring the grid at various points
• Structural health monitoring systems use WSNs to monitor the health of structures (buildings, bridges)by collecting vibration data from sensor nodes deployed at various points at the structure
WSNs are enabled by wireless communication protocols such as IEEE 802.15.4.ZigBee is one of the most popular wireless technologies used by WSNs. ZigBee specifications are based on IEE 802.15.4.ZigBee operates at 2.4GHZ frequency and offer data rates up to 250KB/s and range from 10 to 100 meters depending on the power output and environmental conditions. The power of WSNs lies in their ability to deploy large number of low cost and low power- sensing nodes for continuous monitoring of environmental and physical conditions. WSNs are self organizing networks. Since WSNs have large number of nodes, manual configuration of each node is not possible. The self organizing capability of WSN makes the network robust. In the event of failure of some node or addition of new nodes to the network, the network can reconfigure itself

3.4.1 Communication Protocols:

Communication protocols form the backbone of IoT systems and enable network connectivity and coupling to applications. Communication protocols allow devices to exchange data over the network. These protocols define the data exchange formats, data encoding, addressing schemes for devices and routing of packets from source to destination. Other functions of the protocols include sequence control, flow control and retransmission of lost packets.

3.4.2 Embedded Systems:

An embedded system is a computer system that has computer hardware and software embedded to perform specific tasks. In contrast to general purpose computers or personal computers which can perform various tasks, embedded systems are designed to perform a specific set of tasks. Key component of an embedded system include microprocessor, micro controller , memory, networking unit, input/output unit and storage. Some embedded systems have specialized processors such as digital signal processors, graphics processors and application specific processors. Embedded systems rum embedded running operating systems such as real time operating systems(RTOS). Embedded systems range from low cost miniaturized devices such as digital watches to devices such as digital cameras, point of sale terminals, vending machines, appliances, etc.

3.5 IOT ARCHITECTURE:

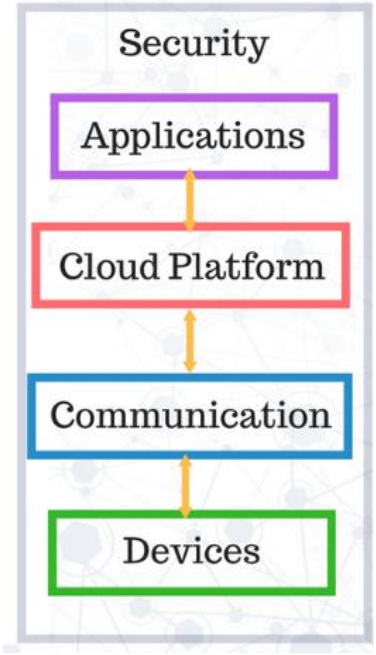

Fig3.7: IOT Architecture

3.6 INTRODUCTION TO ATMEGA328P MICROCONTROLLER:

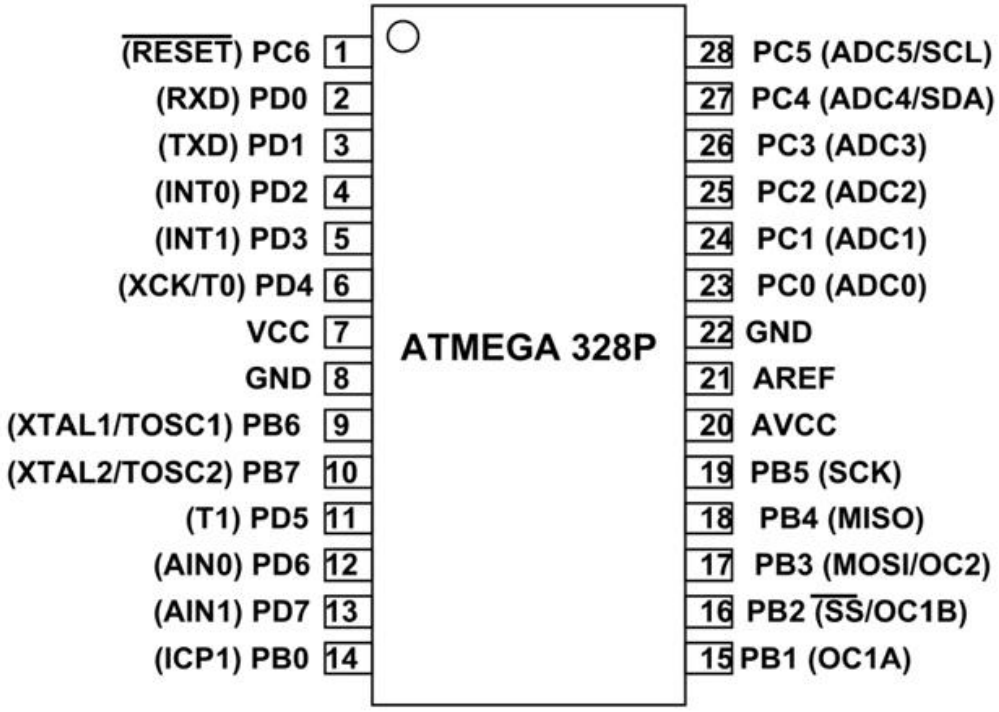

Fig3.8 ATMEGA328P Pin out

3.6.1 Pin Descriptions:

Pin No 7: VCC

Digital supply voltage.

Pin No 8: GND

Ground.

Pin No 9:10: Port B (PB7:0) XTAL1/XTAL2/TOSC1/TOSC2

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source 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.

Depending on the clock selection fuse settings, PB6 can be used as input to the inverting Oscillator amplifier and input to the internal clock operating circuit.

Depending on the clock selection fuse settings, PB7 can be used as output from the inverting Oscillator amplifier.

If the Internal Calibrated RC Oscillator is used as chip clock source, PB7..6 is used as TOSC2..1 input for the Asynchronous Timer/Counter2 if the AS2 bit in ASSR is set.

Pin No 28: Port C (PC5:0)

Port C is a 7-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The PC5..0 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.

Pin No 1: PC6/RESET

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

If the RSTDISBL Fuse is unprogrammed, PC6 is used as a Reset input. A low level on this pin for longer than the minimum pulse length will generate a Reset, even if the clock is not running. The minimum pulse length is given in Shorter pulses are not guaran-teed to generate a Reset.

Pin No 13: Port D (PD7:0)

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.

Pin No 20: AV_{CC}

AV_{CC} is the supply voltage pin for the A/D Converter, PC3:0, and ADC7:6.

It should be externally connected to V_{CC}, even if the ADC is not used. If the ADC is used, it should be connected to V_{CC} through a low-pass filter. Note that PC6..4 use digital supply voltage, V_{CC}.

Pin No 21: AREF

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

Pin No 23:28: ADC7:6 (TQFP and QFN/MLF Package Only)

In the TQFP and QFN/MLF package, ADC7:6 serve as analog inputs to the A/D converter.

These pins are powered from the analog supply and serve as 10-bit ADC channels.

Overview

The ATmega48PA/88PA/168PA/328P is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega48PA/88PA/168PA/328P achieves throughputs approaching 1 MIPS per MHz allowing the system designed to optimize power consumption versus processing speed.

3.6.2 Block Diagram:

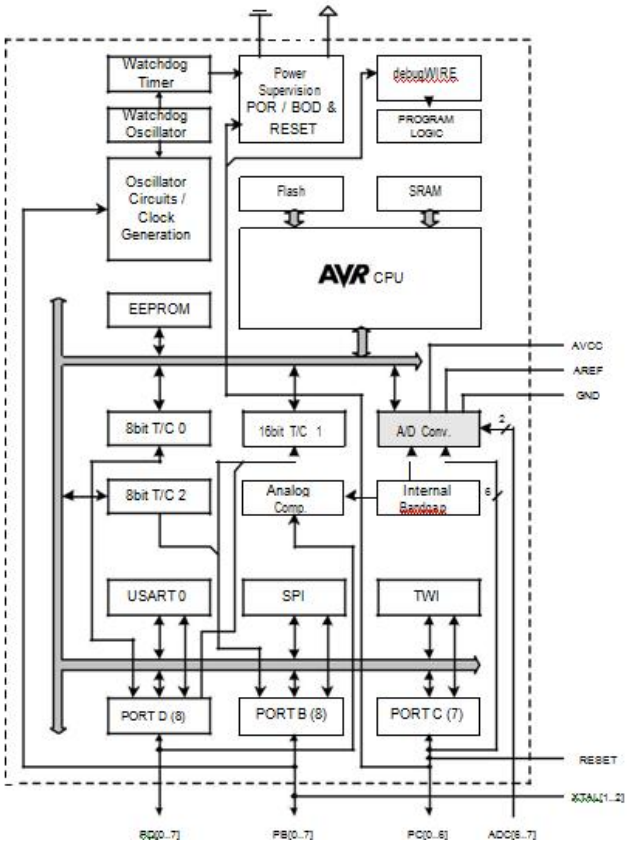

Fig3.9: Block Diagram

The AVR core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than con-ventional CISC microcontrollers.

The ATmega48PA/88PA/168PA/328P provides the following features: 4/8/16/32K bytes of In-System Programmable Flash with Read-While-Write capabilities, 256/512/512/1K bytes EEPROM, 512/1K/1K/2K bytes SRAM, 23 general purpose I/O lines, 32 general purpose work-ing registers, three flexible Timer/Counters with compare modes, internal and external interrupts, a serial programmable USART, a byte-oriented 2-wire Serial Interface, an SPI serial port, a 6-channel 10-bit ADC (8 channels in TQFP and QFN/MLF packages), a programmable Watchdog Timer with internal Oscillator, and five software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, USART, 2-wire Serial Inter-face, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next inter-rupt or hardware reset. In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduc-tion mode stops the CPU and all I/O modules except asynchronous timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low power consumption.

The device is manufactured using Atmel’s high density non-volatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed In-System through an SPI serial interface, by a conventional non-volatile memory programmer, or by an On-chip Boot pro-gram running on the AVR core. The Boot program can use any interface to download the application program in the Application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a

monolithic chip, the Atmel ATmega48PA/88PA/168PA/328P is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega48PA/88PA/168PA/328P AVR is supported with a full suite of program and system development tools including: C Compilers, Macro Assemblers, Program Debugger/Simulators, In-Circuit Emulators, and Evaluation kits.

3.6.3 Resources:

A comprehensive set of development tools, application notes and datasheets are available for download on http://www.atmel.com/avr.

3.6.4 Data Retention:

Reliability Qualification results show that the projected data retention failure rate is much less than 1 PPM over 20 years at 85°C or 100 years at 25°C.

3.6.5 Features:

ATMEGA328P – Simplified Features

CPU	8-Bit AVR
Number of pins	28
Operating voltage	+1.8V to +5v
Number of programmable I/O Pins	23
Communication Interface	Master/Slave SPI Serial Interface(17,18,19 PINS) [Can be used for programming this controller] Programmable Serial USART(2,3 PINS) [Can be used for programming this controller] Two-wire Serial Interface(27,28 PINS)[Can be used to connect peripheral devices like Servos, sensors and memory devices]
JTAG Interface	Not Available
ADC Module	6 Channels, 10 bit resolution ADC
Timer Module	Two 8-bit counters with Separate Prescaler and compare mode, One

	16-bit counter with Separate Prescaler,compare mode and capture mode.
Analog Comparators	1(12,13 PINS)
DAC Module	Nil
PWM Channels	6
External Oscillator	0-4MHz @ 1.8V to 5.5V 0-10MHz @ 2.7V to 5.5V 0-20MHz @ 4.5V to 5.5V
Internal Oscillator	8MHz Calibrated Internal Oscillator
Program Memory /Flash Memory	32 Bytes
CPU Speed	1 MIPS for 1 MHz
RAM	2 KB Internal SRAM
EEPROM	1 KB EEPROM
Program Lock	Yes
Operating Temperature	-40c to +105c

3.7 ARDUINO UNO BOARD:

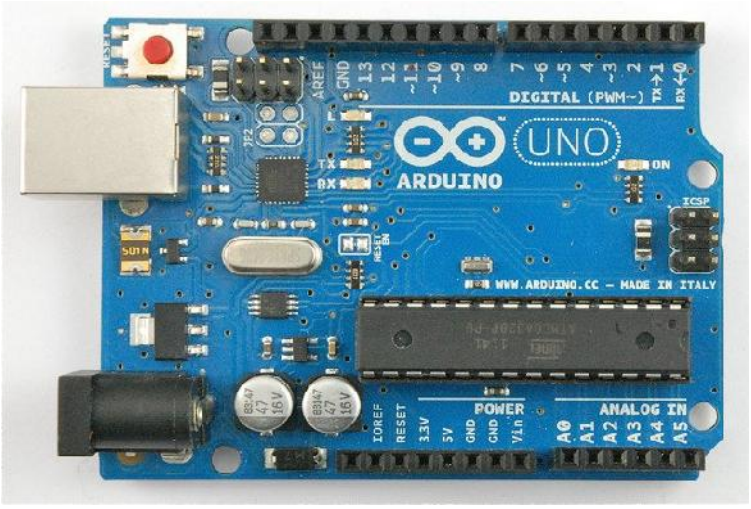

Fig3.10: Arduino uno Board

The Arduino UNO is an open-source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino.cc. The board is equipped with sets of

digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits. The board has 14 Digital pins, 6 Analog pins, and programmable with the Arduino IDE (Integrated Development Environment) via a type B USB cable. It can be powered by a USB cable or by an external 9 volt battery, though it accepts voltages between 7 and 20 volts. It is also similar to the Arduino Nano and Leonardo. The hardware reference design is distributed under a Creative Commons Attribution Share-Alike 2.5 license and is available on the Arduino website. Layout and production files for some versions of the hardware are also available. "Uno" means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Uno board and version 1.0 of Arduino Software (IDE) were the reference versions of Arduino, now evolved to newer releases. The Uno board is the first in a series of USB Arduino boards, and the reference model for the Arduino platform. The ATmega328 on the Arduino Uno comes preprogrammed with a bootloader that allows uploading new code to it without the use of an external hardware programmer.It communicates using the original STK500 protocol.The Uno also differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it uses the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

3.7.1 Communication Pins:

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

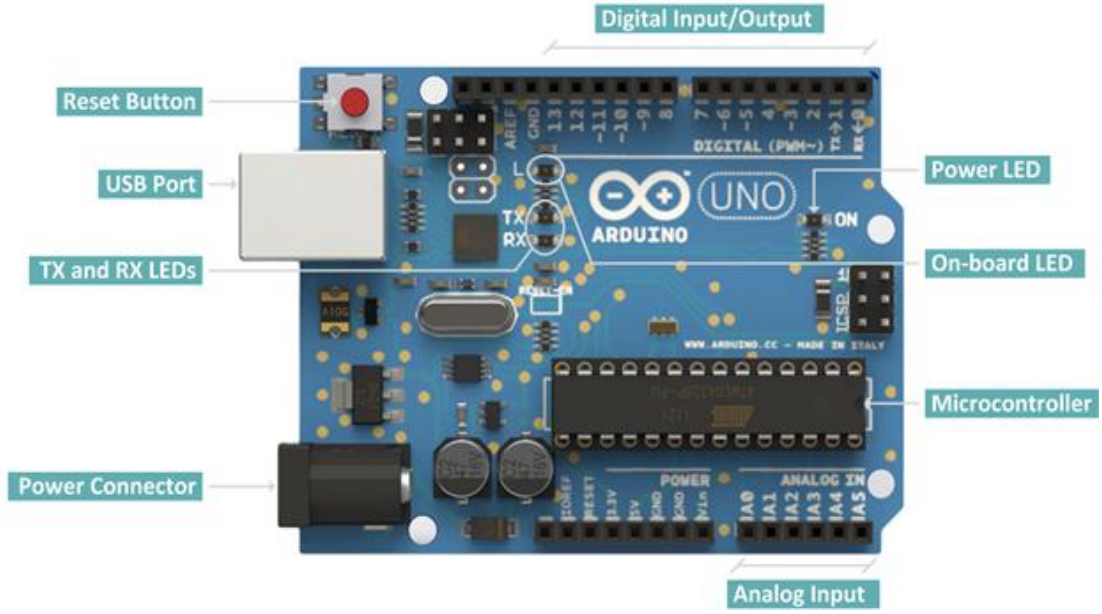

Fig3.11: Arduino uno Pin Out

3.7.2 Reset Button:

Rather than requiring a physical press of the reset button before an upload, the Arduino/Genuino Uno board is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip.

This setup has other implications. When the Uno is connected to a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened.

3.8 INTRODUCTION TO GPS (GLOBAL POSITIONING SYSTEM):

The Global Positioning System, usually called GPS, and originally named NAVSTAR, is a satellite navigation system used for determining one's precise location almost anywhere on Earth. A GPS unit receives time signal transmissions from multiple satellites, and calculates its position by triangulating this data. The GPS was designed by and is controlled by the United States Department of Defense and can be used by anybody for free. The cost of maintaining the system is approximately $400 million per year.

Measurement uncertainty of the majority of commercial GPS receivers varies from 10-11 to 10-13 by the frequency scale, and from 100 ns to 50 ns by the time scale, being dependent on the receiver design. The main sources of uncertainty in GPS measurements are the GPS receiver position error, the orbital error, the satellite and receiver clock errors, the ionosphere and the troposphere delays, the receiver internal delay, the satellite antenna and cable delay, the receiver noise, and the multipath error. The frequency uncertainty for a GPS receiver is larger than that for Cs-standard by 2- 3 orders within a short -time interval (1 – 1000 s), and by one order within a long-term interval of about one week.

It may help to think of a GPS receiver as similar to a standard radio. Like the radio in your car, a GPS receiver is collecting radio signals from the “ether” and magically turning these signals into information we can use. In the case of a GPS receiver the “stations” are satellites broadcasting 11,000 miles away in space and the music is a binary code, but the antenna and radio hardware are subject to similar kinds of interference that affect your car radio’s ability to produce a clear sound. In your car speaker we hear this interference as “static”; in your GPS receiver the interference may result in positional “static”, i.e., degradation of accuracy.

A better radio receiver and antenna system, fewer terrain obstructions, a stronger connection to the broadcasting station all result in better sound quality for your car radio and better positional quality for your GPS radio. A GPS receiver derives its location or positional “fix” with distance measurements (called pseudo ranges) from multiple satellites at precisely the same time, a “measurement epoch”. Attributes collected and stored with the position for

each feature can be used in GIS map making and analysis. While there are only so many things you can do to improve your car radio's performance, by contrast there are many more things users can do to influence GPS positional quality.

Fig3.12: GSM Module

3.8.1 Satellites:

The United States Global Positioning System (GPS) is the first fully operational Global Navigation Satellite System (GNSS). Each satellite broadcasts a signal that is used by receivers to determine precise position anywhere in the world. The receiver tracks multiple satellites and determines a pseudo range measurement (a range measurement based on time) that is then used to determine the user location. A minimum of four satellites is necessary to establish an accurate three-dimensional position.

The Department of Defense (DOD) is responsible for operating the GPS satellite constellation and monitors the GPS satellites to ensure proper operation. Every satellite's orbital parameters (ephemeris data) are sent to each satellite for broadcast as part of the data message embedded in the GPS signal. The GPS coordinate system is the Cartesian earth-centered earth -fixed coordinates as specified in the World Geodetic System reference system 1984 (WGS-84).

24 GPS satellites are currently in orbit around the earth. the first was launched in 1972 and the latest satellite was launched in 2012. The maximum available at any time from a point in Oregon is generally between 4 to 11. The satellites send out radio signals that are collected and read by the GPS receiver.

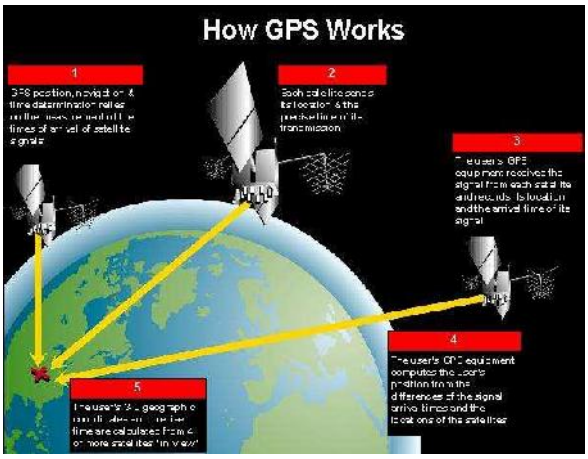

Fig3.13: Working of GPS Modem

The GPS is made up of three parts:

1. Satellites orbiting the Earth
2. Control and monitoring stations on Earth
3. The GPS receivers owned by users.

GPS satellites broadcast signals from space that are picked up and identified by GPS receivers. Each GPS receiver then provides three-dimensional location (latitude, longitude, and altitude) plus the time.

3.8.2 Space Segment:

24+ satellites

20,200 km altitude 55 degrees' inclination 12-hour orbital period

5 ground control stations

Each satellite passes over a ground monitoring station every 12 hours

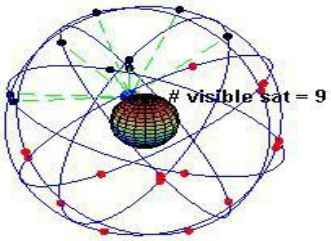

Fig3.14: GPS Satellite System

The space segment is composed of the orbiting GPS satellites or Space Vehicles (SV) in GPS parlance. The GPS design originally called for 24 SVs, this was modified to six planes with four satellites each. The orbital planes are centered on the Earth, not rotating with respect to the distant stars. The six planes have approximately 55° inclination (tilt relative to Earth's equator) and are separated by 60° right ascension of the ascending node (angle along the equator from a reference point to the orbit's intersection.

The full constellation of 24 satellites that make up the GPS space segment are orbiting the earth about 20,200 km above us. They are constantly moving, making two complete orbits in less than 24 hours. These satellites are travelling at speeds of roughly 7,000 miles an hour.

GPS satellites are powered by solar energy. They have backup batteries onboard to keep them running in the event of a solar eclipse, when there's no solar power. Small rocket boosters on each satellite keep them flying in the correct path.

Here are some other interesting facts about the GPS satellites (also called NAVSTAR, the official U.S. Department of Defense name for GPS):
The first GPS satellite was launched in 1978.
A full constellation of 24 satellites was achieved in 1994.
Each satellite is built to last about 10 years. Replacements are constantly being built and launched into orbit.
A GPS satellite weighs approximately 2,000 pounds and is about 17 feet across with the solar panels extended.

Transmitter power is only 50 watts or less.

The orbits are arranged so that at anytime, anywhere on Earth, there are at least four satellites "visible" in the sky.
All satellites broadcast at the same two frequencies, 1.57542 GHz (L1 signal) and 1.2276 GHz (L2 signal).
The satellite network uses a CDMA spread- spectrum technique where the low-bitrate message data is encoded with a high-rate pseudo-random (PRN) sequence that is different for each satellite.

The receiver must be aware of the PRN codes for each satellite to reconstruct the actual message data. The C/A code, for civilian use, transmits data at 1.023 million chips per second, whereas the P code, for U.S. military use.

3.8.3 Control and Monitoring Stations on Earth:

These stations monitor the GPS satellites, checking both their operational health and their exact position in space. The master ground station transmits corrections for the satellite's ephemeris constants and clock offsets back to the satellites themselves. The satellites can then incorporate these updates in the signals they send to GPS receivers. There are five monitor stations: Hawaii, Ascension Island, Diego Garcia, Kwajalein, and Colorado Springs.

3.8.4 The GPS Receivers:

Receiver determines location, speed, direction, and time
3 satellite signals are necessary to locate the receiver in 3D space 4th satellite is used for time accuracy
Position calculated within sub-centimeter scale
Individuals may purchase GPS handsets that are readily available through commercial retailers. Equipped with these GPS receivers, users can accurately locate where they are and easily navigate to where they want to go, whether walking, driving, flying, or boating.

Today's GPS receivers are extremely accurate, thanks to their parallel multi-channel design. Garmin's 12 parallel channel receivers are quick to lock onto satellites when first turned on and they maintain strong locks, even in dense foliage or urban settings with tall buildings. Certain atmospheric factors and other sources of error can affect.

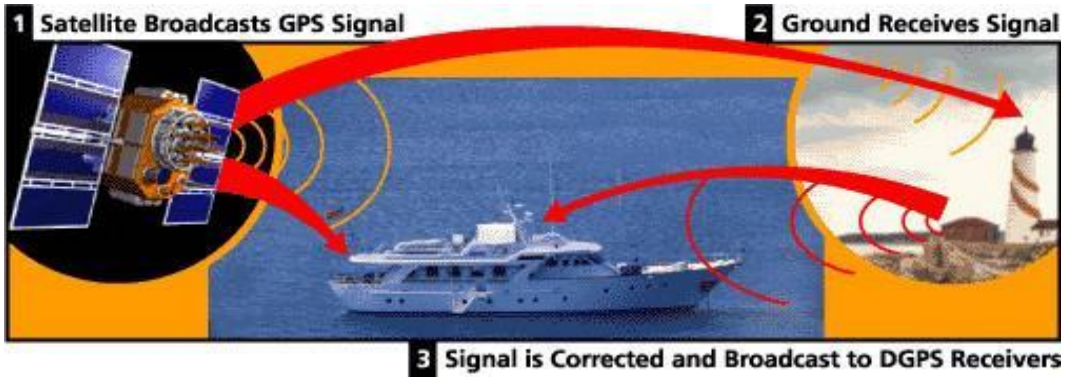

Fig3.15: Newer Garmin GPS Receivers with WAAS

The user segment is composed of hundreds of thousands of U.S. and allied military users of the secure GPS Precise Positioning Service, and tens of millions of civil, commercial and scientific users of the Standard Positioning Service. In general, GPS receivers are composed of an antenna, tuned to the frequencies transmitted by the satellites, receiver-processors, and a highly stable clock (often a crystal oscillator).

They may also include a display for providing location and speed information to the user. A receiver is often described by its number of channels: this signifies how many satellites it can monitor

The Global Positioning System is vast, expensive and involves a lot of technical ingenuity, but the fundamental concepts at work are quite simple and intuitive.

When people talk about "a GPS," they usually mean a GPS receiver. The Global Positioning System (GPS) is actually a constellation of 24 Earth-orbiting satellites The U.S. military developed and implemented this satellite network as a military navigation system, but soon opened it up to everybody else.

3.9 INTRODUCTION TO GSM:

Global System for Mobile (GSM) is a second generation cellular standard developed to cater voice services and data delivery using digital modulation.

Fig3.16: SIM900A GSM Module

GSM MODEM

Architecture

Technical Specifications

Frame Structure

Channels

Security

Characteristics and features

Applications

3.9.1 GSM-History:

- Developed by Group Special Mobile (founded 1982) which was an initiative of CEPT (Conference of European Post and Telecommunication)
- Aim: to replace the incompatible analog system.

- Presently the responsibility of GSM standardization resides with special mobile group under ETSI (European telecommunication Standards Institute).
- Full set of specifications Phase-I became available in 1990.

- Under ETSI, GSM is named as "Global System for Mobile communication".
- Today many providers all over the world use GSM (more than 135 Countries in Asia, Africa, Europe, Australia, America).

- More than 1300 million subscribers in world and 45 million subscribers in India.

GSM supports voice calls and data transfer speeds of up to 9.6kbit/s, together with the transmission of SMS (Short Message Service). GSM operates in the 900MHz and 1.8GHz bands in Europe and the 1.9GHz and 850MHz bands in the US. The 850MHz band is also used for GSM and 3G in Australia, Canada and many South American countries. This gives consumers seamless and same number connectivity in more than 218 countries.

In 1980's the analog cellular telephone systems were growing rapidly all throughout Europe, France and Germany. Each country defined its own protocols and frequencies to work on. For example, UK used the Total Access Communication System (TACS), USA used the AMPS technology and Germany technology. None of these systems were interoperable and also they were analog in nature.

In 1982 the Conference of European Posts and Telegraphs (CEPT) formed a study group called the GROUPE SPECIAL MOBILE (GSM) The main area this focused on was to get the cellular system working throughout the world, and ISDN compatibility with the ability to incorporate any future enhancements.

3.9.2 Technical Specifications of GSM Module:

PCB Size	71.4mm X 66.0mm X1.6mm
Indicators	PWR, status LED, net LED
Power Supply	5V
Communication protocol	UART
RoHS	Yes
Baud Rate	115200

GSM was originally defined for the 900 MHz range but after some time even the 1800 MHz range was used for cellular technology. The 1800 MHz range has its architecture and specifications almost same to that of the 900 MHz GSM technology but building the Mobile exchanges is easier and the high frequency Synergy effects add to the advantages of the 1800 MHz range.

Architecture and Building Blocks GSM is mainly built on 3 building blocks.

GSM Radio Network – This is concerned with the signaling of the system. Handovers occur in the radio network. Each BTS is allocated a set of frequency channels.
GSM Mobile Switching Network – This network is concerned with the storage of data required for routing and service provision.
GSM Operation and Maintenance – The task carried out by it include Administration and commercial operation, Security management, Network configuration, operation, performance management and maintenance tasks.

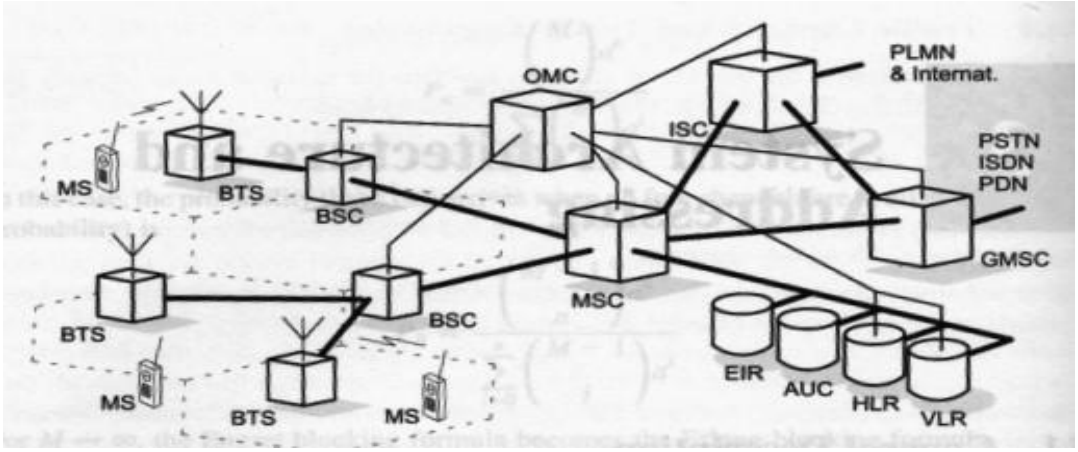

Fig3.17: Basic Block Diagram of GSM Module

Signaling Schemes and Ciphering Codes Used:

GSM is digital but voice is inherently analog. So the analog signal has to be converted and then transmitted. The coding scheme used by GSM is RPE-LTP (Rectangular pulse Excitation – Long Term Prediction). The voice signal is sampled at 8000 bits/sec and is quantized to get a 13-bit resolution corresponding to a bit rate of 104 Kbits/sec. This signal is given to a speech coder (codec) that compresses this speech into a source-coded speech signal of 260 bit blocks at a bit rate of 13 Kbit/sec. The codec achieves a compression ratio of 1:8.

Ciphering Codes:
MS Authentication algorithm's. These algorithms are stored in the SIM and the operator can decide which one it prefers using.

A3/8:
The A3 generates the SRES response to the MSC's random challenge, RAND which the MSC has received from the HLR. The A3 algorithm gets the RAND from the MSC and the secret key Ki from the SIM as input and generated a 32- bit output, the SRES response. The A8 has a 64-bit Kc output.

A5/1 (Over the Air Voice Privacy Algorithm):
The A5 algorithm is the stream cipher used to encrypt over the air transmissions. The stream cipher is initialized for every frame sent with the session key Kc and the no. of frames being decrypted / encrypted. The same Kc key is used throughout the call but different 22-bit frame is used.

3.10 INRODUCTION TO PUSH BUTTON:

A push-button or simply button is a simple switch mechanism for controlling some aspect of a machine or a process. Buttons are typically made out of hard material, usually plastic or metal. The surface is usually flat or shaped to accommodate the human finger or hand, so as to be easily depressed or pushed. Buttons are most often biased switches, though even many un-biased buttons (due to their physical nature) require a spring to return to their

un-pushed state. Different people use different terms for the "pushing" of the button, such as press, depress, mash, and punch.

3.10.1 Uses:

In industrial and commercial applications, push buttons can be linked together by a mechanical linkage so that the act of pushing one button causes the other button to be released. In this way, a stop button can "force" a start button to be released. This method of linkage is used in simple manual operations in which the machine or process have no electrical circuits for control.

Red pushbuttons can also have large heads (called mushroom heads) for easy operation and to facilitate the stopping of a machine. These pushbuttons are called emergency buttons and are mandated by the electrical code in many jurisdictions for increased safety. This large mushroom shape can also be found in buttons for use with operators who need to wear gloves for their work and could not actuate a regular flush push button. The source of the energy to illuminate the light is not directly tied to the contacts on the back of the pushbutton but to the action the pushbutton controls. In popular culture, the phrase the button (sometimes capitalized) refers to a (usually fictional) button that a military or government leader could press to launch nuclear weapons.

Fig3.18: Push Button Switch

3.11 FLOW CHART:

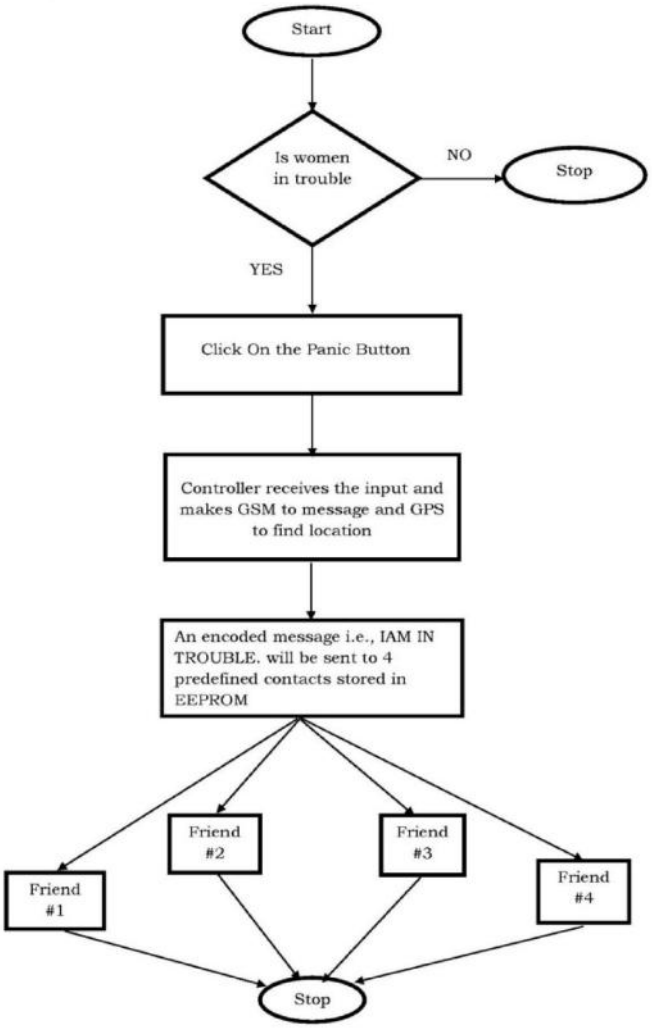

Fig3.19: Flow Chart

CHAPTER 4
SOFTWARE SPECIFICATIONS

4.1 ARDUINO IDE:

The Arduino integrated development environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in the programming language Java. It is used to write and upload programs to Arduino compatible boards, but also, with the help of 3rd party cores, other vendor development boards.

The source code for the IDE is released under the GNU General Public License, version 2. The Arduino IDE supports the languages C and C++ using special rules of code structuring.The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main() into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution.The Arduino IDE employs the program avrdude to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

4.2 SOFTWARE DESCRIPTION:

- Check out: http://arduino.cc/en/Guide/HomePage
- Download & install the Arduino environment (IDE)
- Connect the board to your computer via the USB cable
- If needed, install the drivers (not needed in lab)
- Launch the Arduino IDE
- Select your board
- Select your serial port
- Upload the program

4.3 ARDUINO IDE:

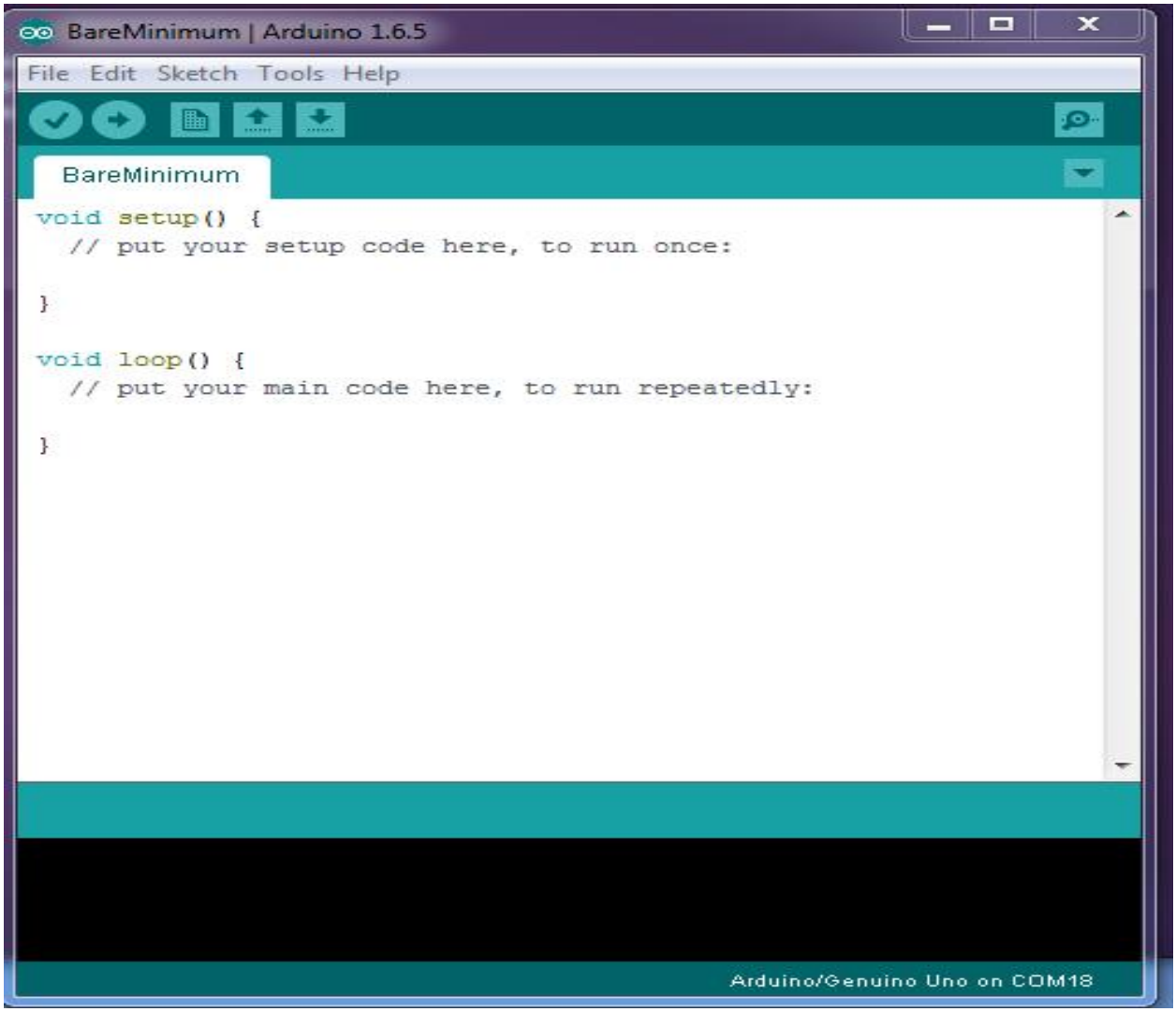

Fig4.1: Ardino IDE

The Arduino integrated development environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in the programming language Java. It originated from the IDE for the languages Processing and Wiring. It includes a code editor with features such as text cutting and pasting, searching and replacing text, automatic indenting, brace matching, and syntax highlighting, and provides simple one-click mechanisms to compile and upload programs to an Arduino board. It also contains a message area, a text console, a toolbar with buttons for

common functions and a hierarchy of operation menus. The source code for the IDE is released under the GNU General Public License, version 2.

The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main() into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program avrdude to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware

4.4 SKETCH:

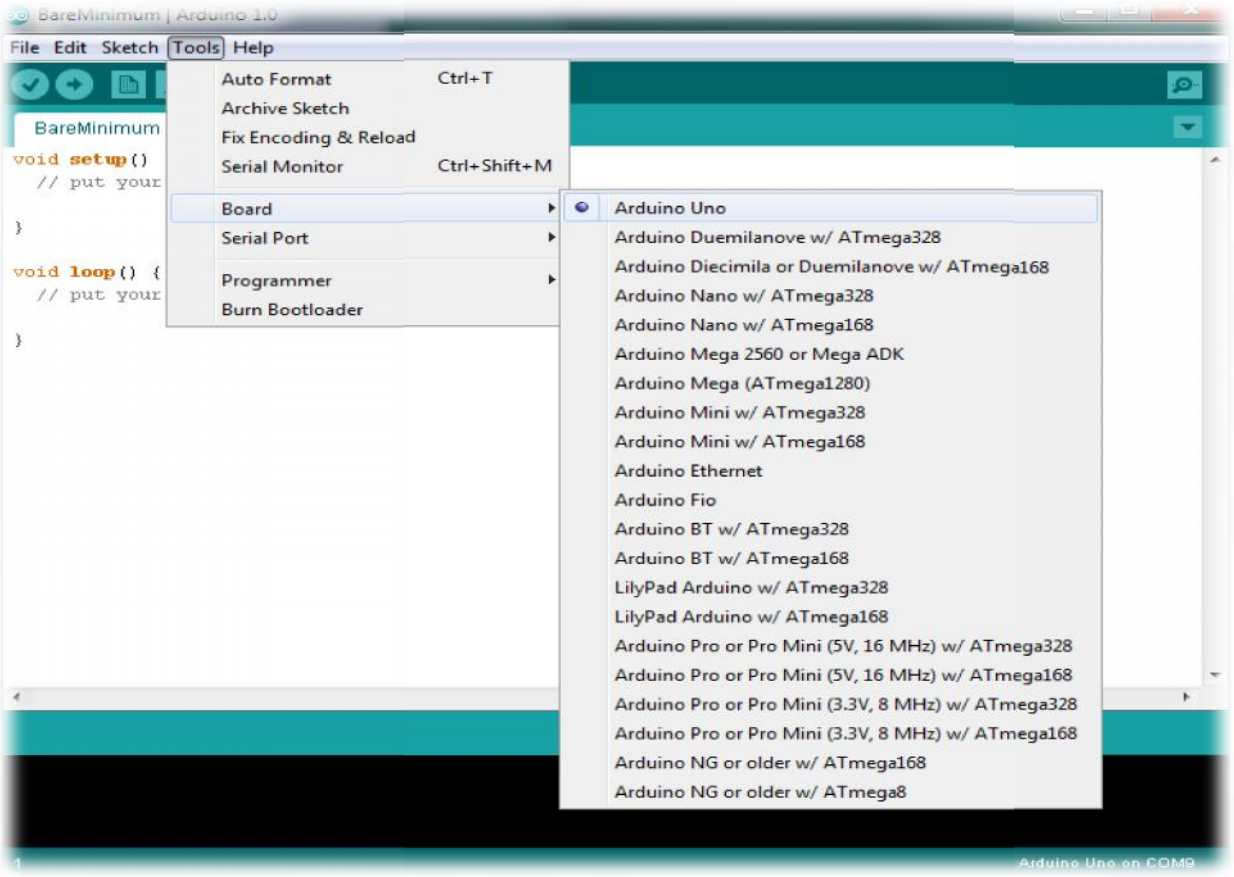

Fig4.2: Arduino Sktech

A sketch is a program written with the Arduino IDE.Sketches is saved on the development computer as text files with the file extension .ino. Arduino Software (IDE) pre-1.0 saved sketches with the extension .pde.

A minimal Arduino C/C++ program consists of only two functions:

setup(): This function is called once when a sketch starts after power-up or reset. It is used to initialize variables, input and output pin modes, and other libraries needed in the sketch.
loop(): After setup() function exits (ends), the loop() function is executed repeatedly in the main program. It controls the board until the board is powered off or is reset.

Once the Program is ready you need to upload the program into the arduino Board by selecting the board in tools and select the Serial port to which your board have selected.

CHAPTER 5

ADVANTAGES AND DISADVANTAGES

5.1 ADVANTAGES:

- Monitors all hazards and threats
- Alert message to mobile phone for remote information Mobile number.

5.2 DISADVANTAGES:

- When Power Is Off, Then The Total System Is Off, So Always Required Battery.
- GSM module should be handled smoothly.

5.3 APPLICATIONS:

- Auto motives and transport vehicles
- Security, Remote monitoring, Transportation and logistics
- This system is also can be interfaced with Vehicle airbag system.

CHAPTER 6

IMPLEMENTATION

6.1 SYSTEM IMPLEMENTATION:

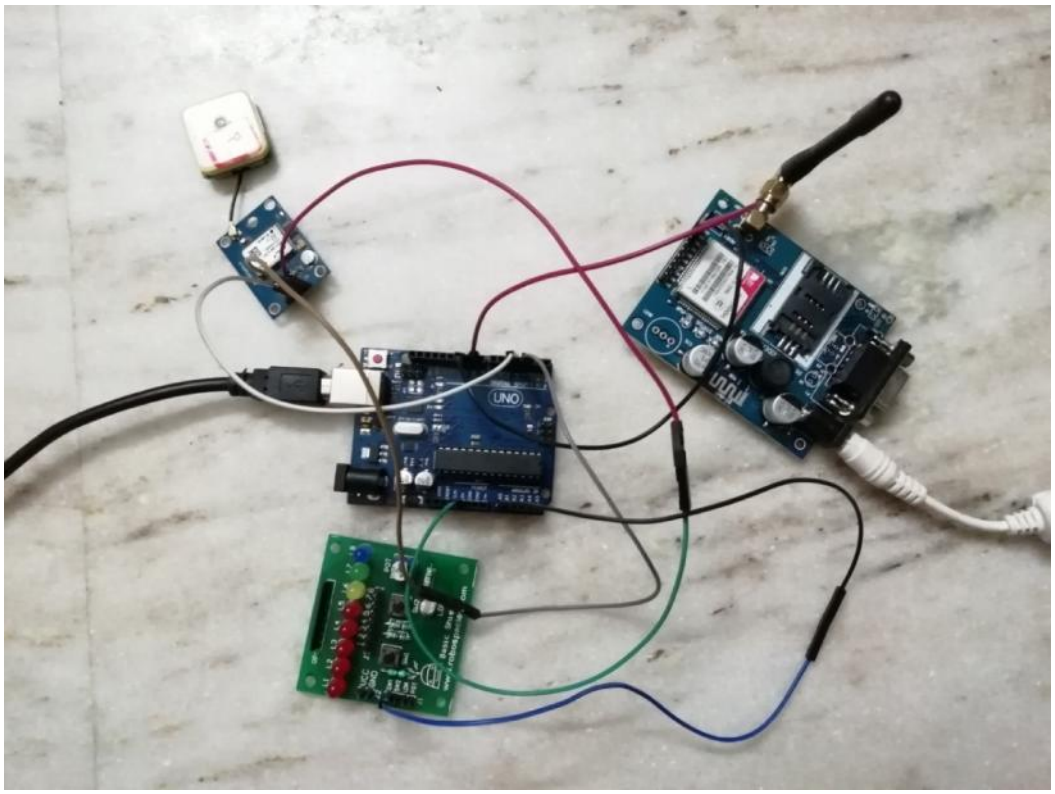

Fig6.1: System Implemented

6.2 PROJECT OUTPUT:

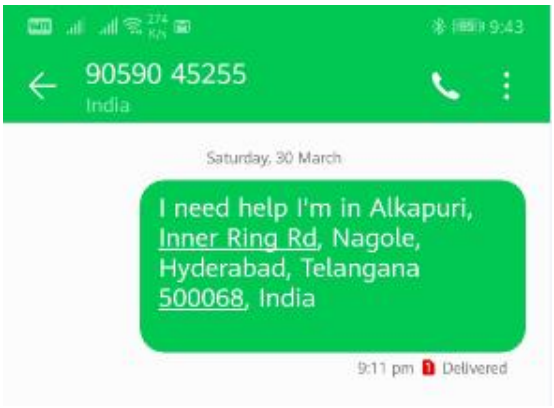

Fig6.2: project Output

CHAPTER 7
CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION:

In this project work, we have studied and Implemented a complete working model using ATMEGA328P Micro controller formally called as Arduino Uno Board. The Programming and Interfacing of Micro controller has been mastered during the implementation. This project work also included the study of GSM and GPS Modules in real time experience.

The Advantage of using this project , whenever the button is Pressed we will get the location of the Person from the GPS Module and also send the SMS to the family member through the GSM Module with the location. So, that we can know that the person is in threat and need help at the location.

7.2 FUTURE SCOPE:

- By encrypting the GOOGLE MAPS in the GPS sensor it can detect the area instead of latitude and longitude information.
- By using Nano sized materials, the kit size gets reduced.
- Using wireless GPS modem and wireless Panic button the carrying of the kit can be avoided.
- More effective system can be designed by adding MOTION DETECTOR Technology.

REFERENCES

REFERENCE BOOKS:

- **ATMEGA328** Micro controller "**A Technical Reference Book**" by J.M Hughes.
- **"Internet of things"** technology Reference by Smart Bridge Educational Services pvt.ltd
- **"Power Electronics"** by M D Singh and K B Khanchandan
- **"Linear Integrated Circuits"** by D Roy Choudhry & Shail Jain
- **"Electrical Machines"** by S K Bhattacharya
- **"Electrical Machines II"** by B L Thereja

APPENDIX

```
#include <SoftwareSerial.h>
 #include <TinyGPS.h>
TinyGPS gps;

SoftwareSerial mySerial(7, 8);//(rx,tx)//GSM
SoftwareSerial ss(4, 3); //GPS

int button;

void setup()
{
  mySerial.begin(4800);   // Setting the baud rate of GSM Module
  ss.begin(4800);
  Serial.begin(4800);    // Setting the baud rate of Serial Monitor (Arduino)
  delay(100);
  pinMode(6,INPUT);
}



void loop()
{
 button=digitalRead(6);
 if(button == 0)
 {
  Serial.println("Button Pressed");
  sendlocation();
  SendMessage();
 }
```

```
}


 void SendMessage()
{
  Serial.println(".............................");
  mySerial.println("AT+CMGF=1");    //Sets the GSM Module in Text Mode
  delay(1000);  // Delay of 1000 milli seconds or 1 second
  mySerial.println("AT+CMGS=\"+918885544987\"\r"); // Replace x with mobile number
  delay(1000);
  mySerial.println("I'm in need ");// The SMS text you want to send
  delay(100);
  mySerial.println((char)26);// ASCII code of CTRL+Z
  delay(1000);
}


 void RecieveMessage()
{
  mySerial.println("AT+CNMI=2,2,0,0,0"); // AT Command to receive a live SMS
                                  //AT+CNMI=1,1,0,0,0 Set the new message
indicators.(CNMI=command name in text)

                                //AT+CNMI=<mode>,<mt>,<bm>,<ds>,<bfr>

//                                  <mode>=1 discard unsolicited result codes indication when TA –
TE link is reserved.(TE=terminal equipment)
//
//                                      <mt>=1 SMS-DELIVERs are delivered to the SIM and
routed using unsolicited code.
//
```

```
//                                          <bm>=0 no CBM indications are routed to the TE.
//
//                                           <ds>=0 no SMS-STATUS-REPORTs are routed.

//                                                 <bfr>=0 TA buffer of unsolicited result codes
defined within this command is flushed to the TE.

 delay(1000);
}

void sendlocation()
{
 bool newData = false;
 unsigned long chars;
 unsigned short sentences, failed;

 // For one second we parse GPS data and report some key values
 for (unsigned long start = millis(); millis() - start < 1000;)
 {
  while (ss.available())
  {
   char c = ss.read();
   // Serial.write(c); // uncomment this line if you want to see the GPS data flowing
   if (gps.encode(c)) // Did a new valid sentence come in?
    newData = true;
  }
 }

 if (newData)
 {
  float flat, flon;
```

```
  unsigned long age;
  gps.f_get_position(&flat, &flon, &age);
  Serial.print("LAT=");
  Serial.print(flat == TinyGPS::GPS_INVALID_F_ANGLE ? 0.0 : flat, 6);
  Serial.print(" LON=");
  Serial.print(flon == TinyGPS::GPS_INVALID_F_ANGLE ? 0.0 : flon, 6);
  Serial.print(" SAT=");
  Serial.print(gps.satellites() == TinyGPS::GPS_INVALID_SATELLITES ? 0 : gps.satellites());
  Serial.print(" PREC=");
  Serial.print(gps.hdop() == TinyGPS::GPS_INVALID_HDOP ? 0 : gps.hdop());
 }

 gps.stats(&chars, &sentences, &failed);
 Serial.print(" CHARS=");
 Serial.print(chars);
 Serial.print(" SENTENCES=");
 Serial.print(sentences);
 Serial.print(" CSUM ERR=");
 Serial.println(failed);
 if (chars == 0)
  Serial.println("** No characters received from GPS: check wiring **");
}
```