CHAPTER 1

INTRODUCTION

1.1 Introduction to Robotics

Robot is defined as a mechanical design that is capable of performing human tasks or behaving in a human-like manner. Building a robot requires expertise and complex programming. It's about building systems and putting together motors, flame sensors and wires, among other important components. A fire fighter robot is one that has a small fire extinguisher added to it. By attaching a small fire extinguisher to the robot, the automation put out the fire by human controlling. This paper covers the design and construction of a robot that is able to sense and extinguish fire. This robot implements the following concepts: environmental sensing, proportional motor control. This robot processes information from its various sensors and key hardware elements via microcontroller. It uses thermistors or ultraviolet or visible sensors to detect the fire accident. A robot capable of extinguishing a simulated tunnel fire, industry fire and military applications are designed and built. Ultraviolet sensors/thermistors/flame sensors will be used for initial detection of the flame. Once the flame is detected, the robot sounds the alarm with the help of buzzer provided to it, the robot actuates an electronic valve releasing sprinkles of water on the flame. The project helps to generate interests as well as innovations in the fields of robotics while working towards a practical and obtainable solution to save lives and mitigate the risk of property damage.

Fire fighters face risky situations when extinguishing fires and rescuing victims, it is an inevitable part of being a fire fighter. In contrast, a robot can function by itself or be controlled from a distance, which means that fire fighting and rescue activities could be executed without putting fire fighters at risk by using robot technology instead. In other words, robots decrease the need for fire fighters to get into dangerous situations. This robot provides fire protection when there is a fire in a tunnel or in an industry by using automatic control of robot by the use of microcontroller in order to reduced loss of life and property damage. This robot uses dc motors, castor wheel, microcontroller, sensors, pump and sprinkler. Microcontroller is the heart of the project. Microcontroller controls all the parts of the robot by the use of programming. In this robot as the fire sensor senses the fire, it sends the signal to microcontroller; since the signal of the sensor is very weak the amplifier is used

so that it can amplify the signal and sends it to microcontroller. As soon as microcontroller receives the signal a buzzer sounds, the buzzer sound is to intimate the occurrence of fire accident. After the sounding of the buzzer microcontroller actuates the driver circuit and it drives the robot towards fire place, as the robot reaches near the fire microcontroller actuates the relay and pump switch is made ON and water is sprinkled on the fire through the sprinkler.

1.2 What is Robotics?

Robotics is a branch of applied science, the popular conception of which came not from science, but from drama, fiction and cinema. The word "robot" was first used in 1921 by Czech playwright Karel Capek in his play "Rossum's Universal Robots" where robots were machines resembling human beings except that they were exceptionally hardworking. The word "Robotics" which means the study of robots, was later coined in 1942 by science fiction writer Isaac Asimov in his story "Runaround" where he put forward three "laws" of robotics. Science fiction writers including Asimov and film-makers used the concept of robots widely and projected robots as human-like mechanical "beings" with tremendous physical and intellectual capabilities, compared to which even the most sophisticated robots of today will look very primitive.

This budding of the new science in the cradles of arts had two-fold results. On one side, robotics got a natural terminology straight from human anatomy with words like arm, shoulder, elbow, wrist, hand, finger, leg, knee, ankle, foot etc and an ideal system, namely the human body, to get new ideas and to evaluate the performance of existing system. On the other hand, a myth was created in the minds of lay- men regarding human-like machines called robots, the sophistication of which is quite phenomenal. To many people, the word "robot" gives rise to a mental picture of a metallic human of tremendous strength and a picture of an actual robot would be rather disappointing. An actual industrial robot can, of course, look similar to a human arm, in basic mechanical architecture. For example, the motion capabilities of a six degrees of freedom PUMA robot can be explained in analogy with movements at shoulder, elbow and wrist of a human arm. Many other robots, however, depart from this analogy to different extents depending on their architecture, though they perform the same jobs as PUMA. The four principal components of a robot, namely the manipulator, the controller, the sensors and the actuators roughly resemble in function (though not in appearance) the human arm, brain, sense organs and muscles.

A standard definition (given by Robot Institute of America) describes a robot as a "reprogrammable multifunctional manipulator". In that perspective, hard automation systems and numerically con- trolled (NC) machines do not fall within the scope of robotics. Teleoperators or telerobots also fall near the border line. Though pro- grammable machines existed even in the 19th century, the science of robotics came into being in the last 50 years through the epoch-making developments of first telerobot to handle radioactive material (world war II), first servoed electric-powered teleoperators (1947–48), NC machines (1952), first reprogrammable robot (1954) and the installation of the first robot (1961). By the 1970's, robotics emerged as an independent field of study. Nowadays, robots are used for material handling, welding, spray-painting, teleoperations (in inaccessible and or hazardous places), assembly, machining etc.

With this introduction to the field of robotics, let us have a look at the recent trends in robotic research and application, which can be described under the following broad headings.

1.3 Types of Robot

- **Redundant Robots**

Six degrees of freedom are, in principle, enough to manipulate objects in space with three possible independent translations and three independent rotations. But, with a given architecture of a robot arm and a given working environment, restrictions of workspace, dexterity and obstacles call for additional degree(s) of free- dom. In such cases, as a human being uses additional freedoms of the body to supplement the capabilities of the arm for enhancing the reach, manipulate objects comfortably and reach below the table or around the corner objects, additional degrees of freedom can be provided in the robot arm with extra joints and links. Such robots are called redundant robots (because they use more inputs than necessary) and are used for the purposes of workspace enhancement and avoidance of singularities and obstacles. With a redundant robot, a particular point can be reached in infinite number of ways — to choose one of those infinite ways is the problem of redundancy resolution, which is solved by optimizing the performance.

- **Space Robots**

Robots in space applications are light, can handle greater masses and have a special characteristic that, unlike robots on earth, their frames are not fixed, rather they float with the rest of the robot, together with the space vehicle.

- **Flexible Robots**

Truly speaking, all solid bodies are flexible. Conventional modelling of robot manipulators needs to consider the links of a robot as rigid, for which the deflections have to be negligible from the viewpoint of positional accuracy. Consequently, the links are to be designed stronger than necessary and heavy. But, from a physical point of view, it is not necessary and we should not mind the links being flexible as long as they are within elastic limits and we know their behaviour. So, the recent interest has been to work with flexible robots and to take advantage of their light weight by incorporating their flexibility into the mathematical model which, of course, complicates the dynamics of the system --- a price to be paid for the advantage gained.

- **Parallel-actuated Robots and Closed-loop Robots**

The traditional serial chain robots, due to their cantilever structure, have less load carrying capacity. Actuations off the base aggravate this problem and make the robot bulky. Consequently, the serial robots tend to bend at high load and vibrate at high speed. Though they possess a large workspace, the positioning capability is rather poor. So, where high load carrying capacity and precise positioning is of prime concern, an alternative is provided by parallel-actuated and closed-loop robots which have attracted tremendous research interest in the last 15 years. As a human being uses both arms to handle a heavy load, three fingers in parallel for doing a precise work like writing and as animal body is supported on four legs with provision of in-parallel actuation at the leg joints, robot manipulators also can be designed with the end effector (hand) connected to the frame by parallel chains of joints and links having the actuations distributed among the various chains or legs. The most celebrated among the parallel manipulators is the six-degrees-of- freedom parallel manipulator called the Stewart platform which has its end-effector connected to the ground by six extensible legs having ball-socket joints at the ends, the extensions of the legs being done by six linear actuators. Parallel robots, in general, provide high structural rigidity and load carrying capacity, good positioning capabilities and have less vibration. But, they generally have

restricted workspaces and their kinematics and dynamics is quite complicated to study and analyse. Typical applications of parallel robots include applications where high load capacity and precise positioning are required, use as an assembly workstation, dexterous wrist and micromanipulators. The application of the concept of parallel actuation has uses in cooperating robots and in multi-fingered gripping and manipulation.

- **Walking Robots**

While manipulation robots manipulate objects by utilizing the freedom available at the joints, mobile robots can carry objects to greater distances by body movement. In ordinary life, we use trains, automobiles and animals for conveyance. Similarly, in robotics, we have tracked, wheeled and legged vehicles. Though all of these have their own applications, walking machines have enjoyed the maximum research interest due to their versatility over terrain irregularities and greater mobility, and work is mostly focused on machines walking on two to six legs. Till now, most of these walking machines have succeeded mostly in laboratory conditions and have achieved little breakthrough on completely unstructured ground, but the attempts in this direction promise a high potential. Recently BARC has developed a walking machine with six legs (presented in National Convention on Industrial Problems in Machines and Mechanisms 1994) which moves forward and can take turns also, but the walking speed is quite low. A challenging field of research is biped locomotion which gives rise to a problem of stability, which is evident as equilibrium of a body with less than three supports is precarious. The ease of biped locomotion in human beings can be attributed to their erect body structure, the extent and nature of the surface of the foot and an extremely smart nervous system — conditions simulating which in machines is a really challenging task.

- **Model-based Control**

The traditional control strategy of robot manipulators is completely error driven and shows poor performance at high-speeds, when the high dynamic forces act as disturbances. The current trend is towards model-based control, where the dynamic forces are incorporated in the control strategy as feed forward gains and feed- back compensations along with the servo-controller which is required only to take care of external noise and other factors not included in the dynamic model of the robot. As is expected, the model-based control scheme exhibits better performance, but demands higher computational load in real time. A particular

area of model-based control is adaptive control, which is useful when the dynamic parameters of the robot are not well-known a priori. The controller adapts itself during execution of tasks and improves the values of the dynamic parameters.

- **Force Control**

Conventional control schemes primarily concentrate on position control. But certain tasks (e.g. cleaning a window pane) requires the maintenance of some required contact forces. Cur- rent trend is to control the force in such directions (in such applications) and position in other directions. The use of compliance (flexibility) also is getting popular in control. In applications like insertion of a peg into a hole, if the positional accuracy is poor, the forces stemming from such errors tend to correct the position, if some compliance is provided at the wrist.

1.4 Robot Intelligence and Vision

Intelligence, as pertaining to human beings, is a very subtle and profound quality which cannot be learnt or taught (though can be enhanced in scope and application) and cannot be described in terms of logic and knowledge alone. Again, knowledge is much more than a collection of pieces of information. As the so-called machine intelligence of the present time basically stems from the information and logic coded in computer, we cannot expect true 'intelligence' in a robot right now in the sense of the robot having true 'understanding'. Still, the intelligent robots, as they are called, constitute an extremely popular topic in robotics, because they can perform various tasks in an apparently intelligent manner.

For example, machine intelligence can be applied for path planning and obstacle avoidance by maintaining (and updating) a complete CAD model of the working environment such that the robot navigates through obstacles without collision. As sight, sound and touch assist human intelligence so far as the knowledge of the external world is concerned, robots also are equipped with sensors (optical, ultrasonic, tactile) for feedback. When a robot has to work in a poorly structured or un- structured environment (the environment not being completely known a priori), the robot intelligence heavily relies on 'vision', which is the process of extracting, characterizing and interpreting information from two-dimensional images of the three-dimensional world. In the whole processing of an image, the interpretation of a scene is the most difficult task and requires highest level of intelligence.

The maturity of this technique is expected to provide tremendous decision-making capabilities to the robots.

The dream of perfection for the science of robotics is a system equipped with high-performance sensors and stereo vision, such that it can work in unstructured environment where changes in scenario may be rapid and unexpected. In addition, certain degree of fault-tolerance is desirable with decision-making capabilities such that it can perform tasks assigned to it as long as they are physically possible.

1.5 History of Robotics

In 1927 the Maschinenmensch ("machine-human") gynoid humanoid robot (also called "Parody", "Futura", "Robotrix", or the "Maria impersonator") was the first depiction of a robot ever to appear on film was played by German actress Brigitte Helm in Fritz Lang's film Metropolis.

In 1942 the science fiction writer Isaac Asimov formulated his Three Laws of Robotics.

In 1948 Norbert Wiener formulated the principles of cybernetics, the basis of practical robotics.

Fully autonomous robots only appeared in the second half of the 20th century. The first digitally operated and programmable robot, the Unimate, was installed in 1961 to lift hot pieces of metal from a die casting machine and stack them. Commercial and industrial robots are widespread today and used to perform jobs more cheaply, or more accurately and reliably, than humans. They are also employed in jobs which are too dirty, dangerous, or dull to be suitable for humans. Robots are widely used in manufacturing, assembly, packing and packaging, transport, earth and space exploration, surgery, weaponry, laboratory research, safety, and the mass production of consumer and industrial goods.

Robotics is based on two enabling technologies: Telemanipulators and the ability of numerical control of machines.

Telemanipulators are remotely controlled machines which usually consist of an arm and a gripper. The movements of arm and gripper follow the instructions the human gives through his control device. First telemanipulators have been used to deal with radio-active material.

Numeric control allows controlling machines very precisely in relation to a given coordinate system. It was first used in 1952 at the MIT and lead to the first programming language for machines (called APT: Automatic Programmed Tools).

The combination of both of these techniques leads to the first programmable telemanipulator. The first industrial robot using these principles was installed in 1961. These are the robots one knows from industrial facilities like car construction plants.

The development of mobile robots was driven by the desire to automate transportation in production processes and autonomous transport systems.New forms of mobile robots have been constructed lately like insectoid robots with many legs modelled after examples nature gave us or autonomous robots for underwater usage.

Humanoid robots are being developed since 1975 when Wabot-I was presented in Japan. The current Wabot-III already has some minor cognitive capabilities. Another humanoid robot is "Cog", developed in the MIT-AI-Lab since 1994. Honda's humanoid robot became well known in the public when presented back in 1999. Although it is remote controlled by humans it can walk autonomously (on the floor and stairs).

1.6 What is a Robot?

Robots are physical agents that perform tasks by manipulating the physical world. They are equipped with sensors to perceive their environment and effectors to assert physical forces on it (covered in more detail in next section). As mentioned before Robots can be put into three main categories: manipulators, mobile robots and humanoid robots.

- **Robotics and AI**

Artificial intelligence is a theory. The base object is the agent who is the "actor". It is realized in software. Robots are manufactured as hardware. The connection between those two is that the control of the robot is a software agent that reads data from the sensors decides what to do next and then directs the effectors to act in the physical world.

- **Sensor**

A sensor is a device that measures a physical quantity and converts it into a signal which can be read by an observer or by an instrument. For example, a mercury-in-glass

thermometer converts the measured temperature into expansion and contraction of a liquid which can be read on a calibrated glass tube. A thermocouple converts temperature to an output voltage which can be read by a voltmeter. For accuracy, most sensors are calibrated against known standards.

- **Effectors**

Effectors are the means by which robots manipulate the environment, move and change the shape of their bodies

In robotics, end effector is the device at the end of a robotic arm, designed to interact with the environment. The exact nature of this device depends on the application of the robot.

In the strict definition, which originates from serial robotic manipulators, the end effectors means the last link (or end) of the robot. At this endpoint the tools are attached. In a wider sense, end effectors can be seen as the part of a robot that interacts with the work environment. This does not refer to the wheels of a mobile robot or the feet of a humanoid robot which are also not end effectors—they are part of the robot's mobility.

- **Characteristic of robot**

A robot has these essential characteristics:

- Sensing First of all your robot would have to be able to sense its surroundings. It would do this in ways that are not unsimilar to the way that you sense your surroundings. Giving your robot sensors: light sensors (eyes), touch and pressure sensors (hands), chemical sensors (nose), hearing and sonar sensors (ears), and taste (tongue) will give your robot awareness of its environment.
- Movement a robot needs to be able to move around its environment. Whether rolling on wheels, walking on legs or propelling by thrusters a robot needs to be able to move. To count as a robot either the whole robot moves, like the Sojourner or just parts of the robot moves, like the Canada Arm.

CHAPTER 2

FIRE DETECTOR AND EXTINGUISHER ROBOT

2.1 Block Diagram

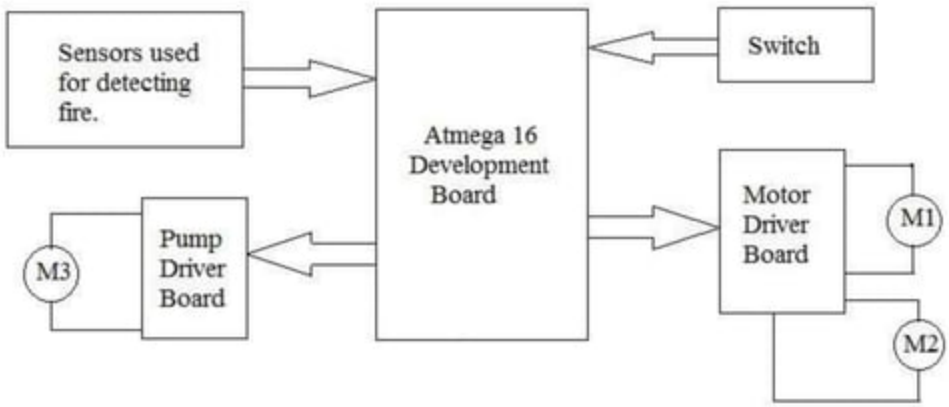

Figure 2.1 Block Diagram of Fire Detector & Extinguisher Robot

2.2 Explanation of Block Diagram

Fire Detector and Extinguisher Robot is operated to detect the fire and also to extinguish it. It can be operated in two modes one is manual mode and other is autonomous mode. Manual mode is operated using joysticks and for autonomous mode there is no human intervention. In manual mode direction of the robot is controlled using joysticks, even pump is operated manually. In autonomous mode IR sensors are used to detect the fire and robot is coded accordingly to move in the direction of detected fire. In this robot has a switch which is used to switch between manual and autonomous mode.

The main components are :

- IR sensors
- Joysticks
- Motor drivers
- DC motors
- Pump

- **IR Sensors:**

IR Photo Diodes are used as sensors. These sensors are connected to ADC pins of the microcontroller. IR Photo diodes are connected in reverse bias to be operated as photo diodes otherwise they will be operated as normal diodes. The voltage variations of the sensors is given as inputs to the Micro controller and accordingly the robot is operated.

- **Joysticks:**

The direction of the robot is controlled using joysticks. The terminals of joysticks is connected to ADC pins of micro controller. Joysticks are usually the potentiometer, the potential variations is given as an input to micro controller and robot is operated accordingly.

- **Motor Drivers:**

Motor drivers are used to describe the direction of movement of the robot. It is used to give high voltage and high current as an output to run the motors which are used in the project for the movement of the robot.

- **DC Motors:**

In this project we use simple DC motor for the rotation of the wheel which are responsible for the movement of the robot. Usually DC motors convert electrical energy into mechanical energy.

- **Pump:**

Pump is a mechanical device which is used to pump water on to the fire to extinguish it. It uses a simple motor to pump water.

CHAPTER 3

SOFTWARE DESCRIPTION

3.1 Codevision AVR

CodeVision AVR an integrated software development environment for microcontrollers family AVR firm Atmel .

CodeVision AVR includes the following components:

- compiler C like language for the AVR;
- compiler assembler for AVR;
- initial program code generator that allows to initialize the peripheral devices;
- interaction with the module development board STK-500;
- module interaction with the programmer ;
- source code editor with syntax highlighting;
- terminal

CodeVisionAVR output files are:

- HEX, BIN or ROM-file to download to the microcontroller through a programmer.
- COFF - file containing information for the debugger.
- OBJ - the file in which to store the intermediate code compilation, the so-called object code.

CodeVisionAVR is a commercial software. There is a free trial version with limited number of possibilities, in particular, the size of the code is limited to 4 kilobytes and includes a number of libraries.

As of November 2012 the latest version is 2.6. C compiler, which is part CodeVisionAVR, has some differences from the AVR-GCC (WinAVR), including its own syntax, the set of supported microcontrollers series (the latest version supports including a series ATXMega), as well as different in speed generates the output code.

❖ **AVR**

AVR - a family of eight-bit microcontrollers firm Atmel . Year of development - 1996 .

History of the AVR architecture

The idea of developing a new RISC -core belongs to two students Norwegian University of Science and Technology (NTNU) in the Norwegian city of Trondheim (Trondheim) - Alf Bogen (Alf-Egil Bogen) and Vegard Vollenu (Vegard Wollen). In 1995 Bogen Vollen and decided to offer the American corporation Atmel , which was known for its chips with Flash-memory , releasing a new 8-bit RISC -microcontroller and implement it Flash-memory programs on a single chip with a computational kernel.

The idea was approved by Atmel Corp., and it was decided to immediately invest in this development. In late 1996, was released experienced microcontroller AT90S1200, and in the second half of 1997 Atmel Corporation has started mass production of a new family of microcontrollers, their advertising and technical support.

The new kernel has been patented and was named AVR. There are several interpretations of the acronym. Someone claims that this Advanced Virtual RISC, others believe that there has not been without ALF Egil Bogen Vegard Wollan RISC.

Family of microcontrollers

Standard family:

- tinyAVR (ATtiny xxx):

  Flash memory up to 16 KB; SRAM up to 512 b; EEPROM 512 b;
  The number of lines input-output 4-18 (total pins 6-32);
  Limited set of peripherals .

- megaAVR (ATmega xxx):

  Flash memory up to 256 KB; SRAM 16KB; EEPROM up to 4 KB;
  The number of lines IO 23-86 (28-100 total number of terminals);
  Hardware multiplier;
  Extended command system and peripherals.

- AVR XMEGA (ATxmega xxx):

  Flash memory up to 384 KB; SRAM to 32KB; EEPROM up to 4 KB;
  Four DMA -controller;
  Innovative Event System.

CodeVisionAVR is a complete set of tools designed for rapid and efficient software development for the Atmel AVR microcontrollers.

CodeVisionAVR is the only IDE on the market that features an Automatic Program Generator (CodeWizardAVR) for the new ATxmega devices and the only C compiler that supports the reduced core (ATtiny4, ATtiny5, ATtiny9, ATtiny10, ATtiny20, ATtiny40) chips.

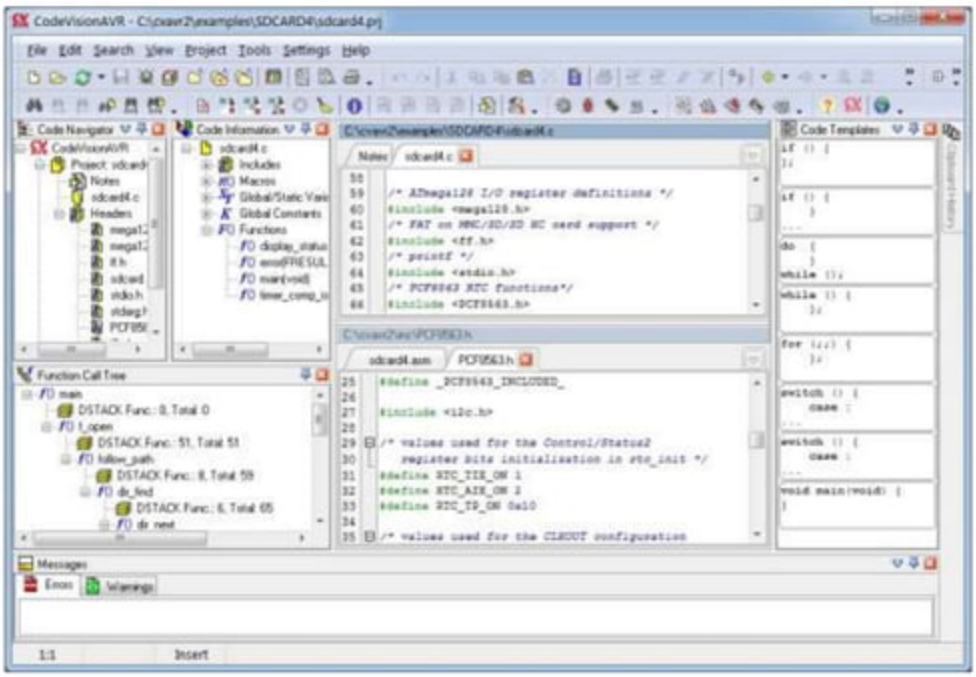

Figure 3.1 Codevision AVR

Main features

- Editor with auto indentation, syntax highlighting for both C and AVR assembler, function parameters and structure/union members autocomplete.
- High perfomance ANSI C compiler with specific extensions that greatly simplify embedded system design:
  - Simple accessing of the EEPROM memory area, allows you to directly place there any variable, even complex structures
  - Simple accessing of the FLASH memory area, allows you to directly place there any constant, even complex structures

- Bit level access to I/O registers
- Interrupt support
- GPIOR register support
- Possibility to insert assembler code directly in the C source file
- VERY EFFICIENT USE OF RAM: Constant character strings are stored only in FLASH memory and aren't copied to RAM, like in other compilers for the AVR
- C Source level debugging, with COFF symbol file generation, allows variable watching in Atmel's AVR Studio Debugger
- Fully compatible with Atmel's In-Circuit Emulators
- 4 memory models:
- TINY (8 bit data pointers for chips with up to 256 bytes of RAM)
- SMALL (16 bit data pointers for chips with more than 256 bytes of RAM)
- MEDIUM (for chips with 128k of FLASH)
- LARGE (for chips with 256k or more FLASH).

The MEDIUM and LARGE memory models allow full FLASH addressing for chips like ATmega128, ATmega1280, ATmega2560, etc, the compiler handling the RAMPZ register totally transparently for the programmer.

This feature is available as Standard in CodeVisionAVR, at no additional costs.

Powerful optimizations:

- Peephole optimizer
- Advanced variables to register allocator, allows very efficient use of the AVR architecture
- Common Block Subroutine Packing (what our competition calls "Code Compressor"), replaces repetitive code sequences with calls to subroutines.
- This optimizer is available as Standard in CodeVisionAVR, at no additional costs, not like in our competitor's products.
- Common sub-expression elimination
- Loop optimization
- Branch optimization
- Subroutine call optimization

- Cross-jumping optimization
- Constant folding
- Constant literal strings merging
- Store-copy optimization
- Dead code removing optimization
- User selectable optimization for code Size or Speed
- Rich set of libraries for embedded systems:
- Alphanumeric LCD modules
- MMC/SD/SD HC FLASH Memory Card drivers, FAT access libraries
- Philips I2C bus
- National Semiconductor LM75 Temperature Sensor
- Philips PCF8563, PCF8583, Dallas Semiconductor DS1302 and DS1307 Real Time Clocks
- Dallas Semiconductor 1 Wire protocol
- Dallas Semiconductor DS1820/DS1822 Temperature Sensors
- Dallas Semiconductor DS1621 Thermometer/Thermostat
- SPI
- Power management
- Delays

3.2 Protius ISIS

Proteus is a software for microprocessor simulation, schematic capture, and printed circuit board (PCB) design. It is developed by Labcenter Electronics.

The XGameStation Micro Edition was designed using Labcenter's Proteus schematic entry and PCB layout tools.

System components

ISIS Schematic Capture - a tool for entering designs.

- PROSPICE Mixed mode SPICE simulation - industry standard SPICE3F5 simulator combined with a digital simulator.
- ARES PCB Layout - PCB design system with automatic component placer, rip-up and retry auto-router and interactive design rule checking.

- VSM - Virtual System Modelling lets cosimulate embedded software for popular micro-controllers alongside hardware design.
- System Benefits Integrated package with common user interface and fully context sensitive help.
- With additional modules, ISIS is also able to simulate the behaviour of a microcontroller (PIC, Atmel, 8051, ARM, HC11 ...) and its interaction with the components that surround it.

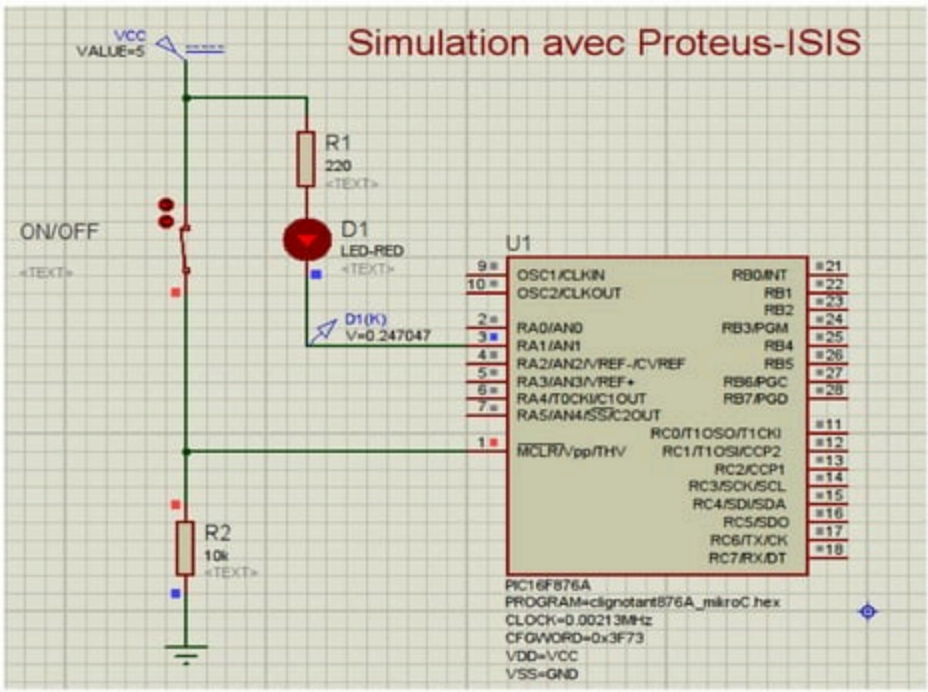

Figure 3.2 Simulation using Protius ISIS

The Proteus Design Suite is wholly unique in offering the ability to co-simulate both high and low-level micro-controller code in the context of a mixed-mode SPICE circuit simulation. With this Virtual System Modelling facility, you can transform your product design cycle, reaping huge rewards in terms of reduced time to market and lower costs of development.

Proteus Virtual System Modelling (VSM) combines mixed mode SPICE circuit simulation, animated components and microprocessor models to facilitate co-simulation of complete microcontroller based designs. For the first time ever, it is possible to develop and test such designs before a physical prototype is constructed.

ISIS Schematic Capture

ISIS lies at the heart of the Proteus system, and is far more than just another schematics package. It combines a powerful design environment with the ability to define most aspects of the drawing appearance. Whether your requirement is the rapid entry of complex designs for simulation and PCB layout, or the creation of attractive schematics for publication, ISIS is the tool for the job.

ARES PCB Layout Software

ARES is the PCB editing software that is included in the Proteus Design Suite. It is fully compatible with ISIS Schematic Capture.

3.3 Sinaprog

Sinaprog is a software which helps in the interfacing between the Codevision AVR and the Atmega microcontroller to dump the program in it.

This software is free domain software which is used to burn the hex file or the Program in the microcontroller. This program is easy to use because of its GUI. This program uses AVR DUDE software as the underlying operating program. Operating frequencies and other .Fuses can be set from this interface.

Figure 3.3 Sinaprog tool

CHAPTER 4

HARDWARE DESCRIPTION

4.1 Atmega 16 Microcontroller

Circumstances that we find ourselves in today in the field of microcontrollers had their beginnings in the development of technology of integrated circuits. This development has made it possible to store hundreds of thousands of transistors into one chip. That was a prerequisite for production of microprocessors, and the first computers were made by adding external peripherals such as memory, input-output lines, timers and other. Further increasing of the volume of the package resulted in creation of integrated circuits. These integrated circuits contained both processor and peripherals. That is how the first chip containing a microcomputer, or what would later be known as a microcontroller came about.

Microprocessors and microcontrollers are widely used in embedded systems products. Microcontroller is a programmable device. A microcontroller has a CPU in addition to a fixed amount of RAM, ROM, I/O ports and a timer embedded all on a single chip. The fixed amount of on-chip ROM, RAM and number of I/O ports in microcontrollers makes them ideal for many applications in which cost and space are critical.

A microcontroller often serves as the "brain" of a mechatronic system. Like a mini, self-contained computer, it can be programmed to interact with both the hardware of the system and the user. Even the most basic microcontroller can perform simple math operations, control digital outputs, and monitor digital inputs. As the computer industry has evolved, so has the technology associated with microcontrollers. Newer microcontrollers are much faster, have more memory, and have a host of input and output features that dwarf the ability of earlier models. Most modern controllers have analog-to-digital converters, high-speed timers and counters, interrupt capabilities, outputs that can be pulse-width modulated, serial communication ports, etc.

Features

- Advanced RISC Architecture
- Up to 16 MIPS Throughput at 16 MHz
- 16K Bytes of In-System Self-Programmable Flash
- 512 Bytes EEPROM

- 1K Byte Internal SRAM
- 32 Programmable I/O Lines
- In-System Programming by On-chip Boot Program
- 8-channel, 10-bit ADC
- Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes
- One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture
- Four PWM Channels
- Programmable Serial USART
- Master/Slave SPI Serial Interface
- Byte-oriented Two-wire Serial Interface
- Programmable Watchdog Timer with Separate On-chip Oscillator
- External and Internal Interrupt Sources

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 conventional

Block Diagram

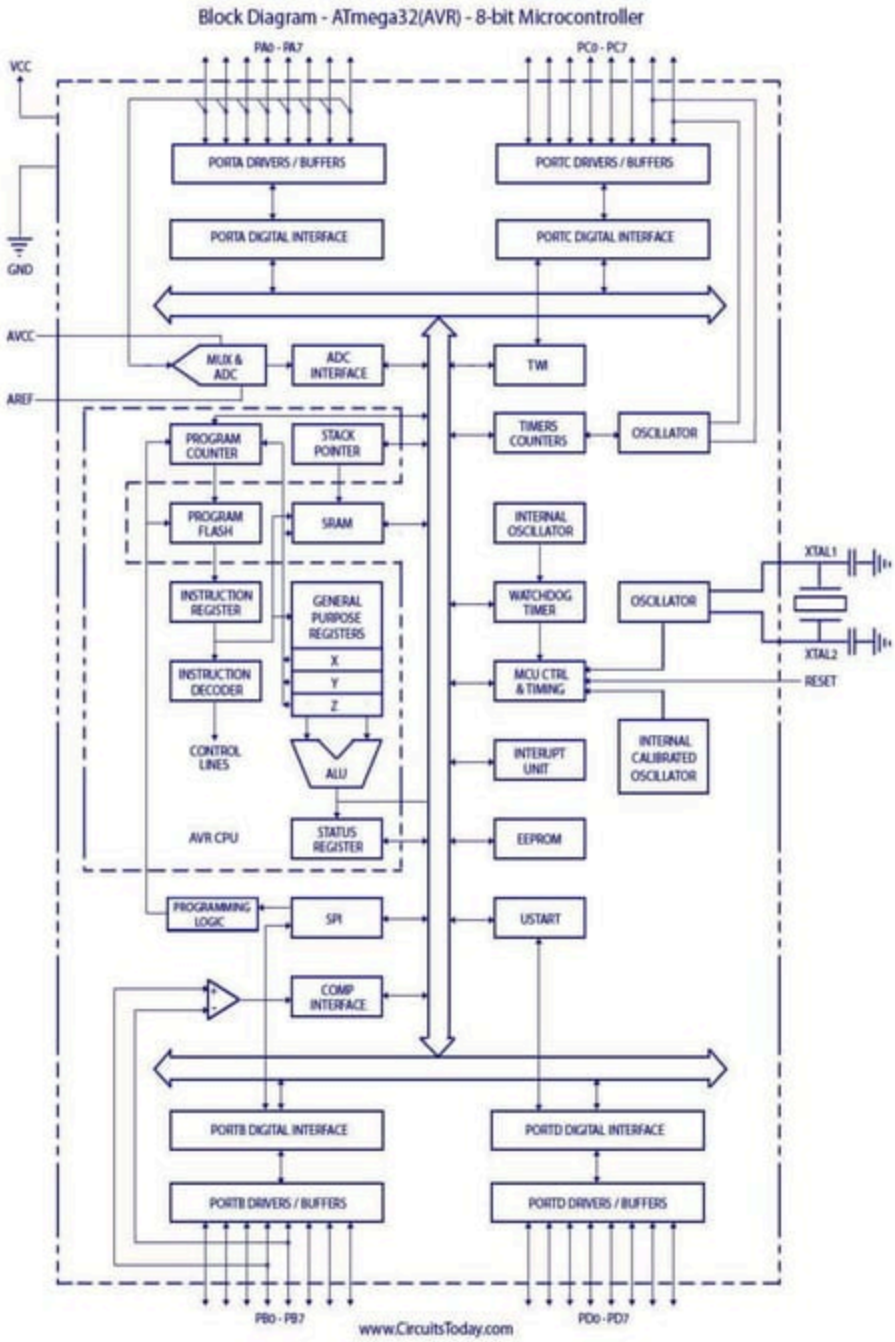

Figure 4.1 Block Diagram of Atmega 16

CISC microcontrollers

The ATmega16 provides the following features: 16 Kbytes of In-System Programmable Flash Program memory with Read-While-Write capabilities, 512 bytes EEPROM, 1 Kbyte SRAM, 32 general purpose I/O lines, 32 general purpose working registers, a JTAG interface for Boundary scan, On-chip Debugging support and programming, three flexible Timer/Counters with compare modes, Internal and External Interrupts, a serial programmable USART, a byte oriented Two-wire Serial Interface, an 8-channel, 10-bit ADC with optional differential input stage with programmable gain (TQFP package only), a programmable Watchdog Timer with Internal Oscillator, an SPI serial port, and six software selectable power saving modes. The Idle mode stops the CPU while allowing the USART, Two-wire interface, A/D Converter, SRAM, Timer/Counters, 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 External Interrupt 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 Reduction 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. In Extended Standby mode, both the main Oscillator and the Asynchronous Timer continue to run.

The device is manufactured using Atmel's high density nonvolatile memory technology. The On chip ISP Flash allows the program memory to be reprogrammed in-system through an SPI serial interface, by a conventional nonvolatile memory programmer, or by an On-chip Boot program 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 ATmega16 is a powerful microcontroller that provides a highly-flexible and cost-effective solution to many embedded control applications.

The ATmega16 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.

- **AVR CPU Core**

Introduction

The main function of the CPU core is to ensure correct program execution. The CPU must therefore be able to access memories, perform calculations, control peripherals, and handle interrupts.

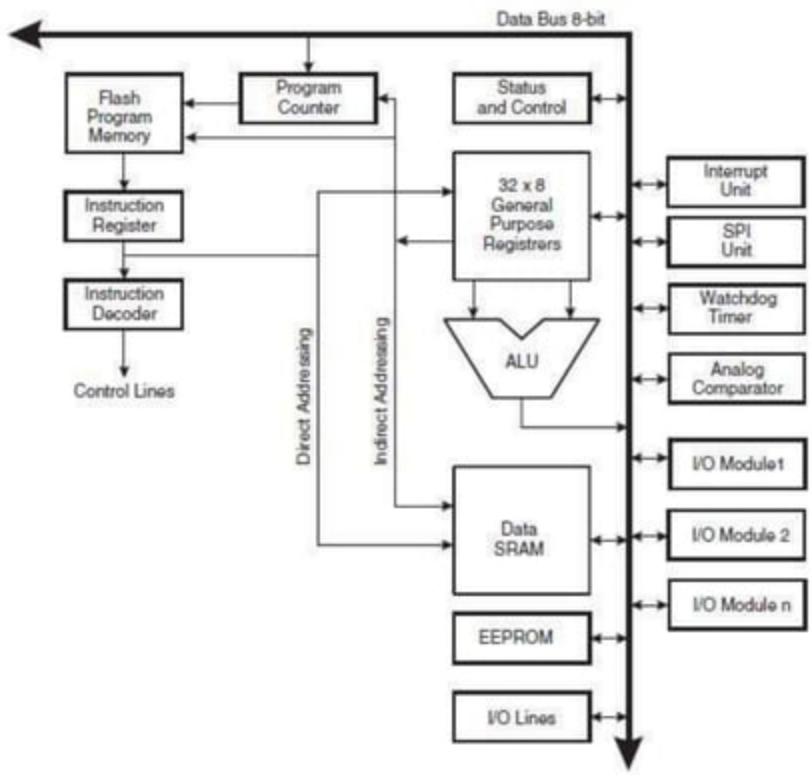

Figure 4.2 Architecture of AVR CPU Core

In order to maximize performance and parallelism, the AVR uses a Harvard architecture -with separate memories and buses for program and data. Instructions in the program memory are executed with a single level pipelining. While one instruction is being executed, the next instruction is pre-fetched from the program memory. This concept enables

instructions to be executed in every clock cycle. The program memory is In-System Reprogrammable Flash memory.

The fast-access Register File contains 32 × 8-bit general purpose working registers with a single clock cycle access time. This allows single-cycle Arithmetic Logic Unit (ALU) operation. In a typical ALU operation, two operands are output from the Register File, the operation is executed, and the result is stored back in the Register File – in one clock cycle.

Pin Configuration

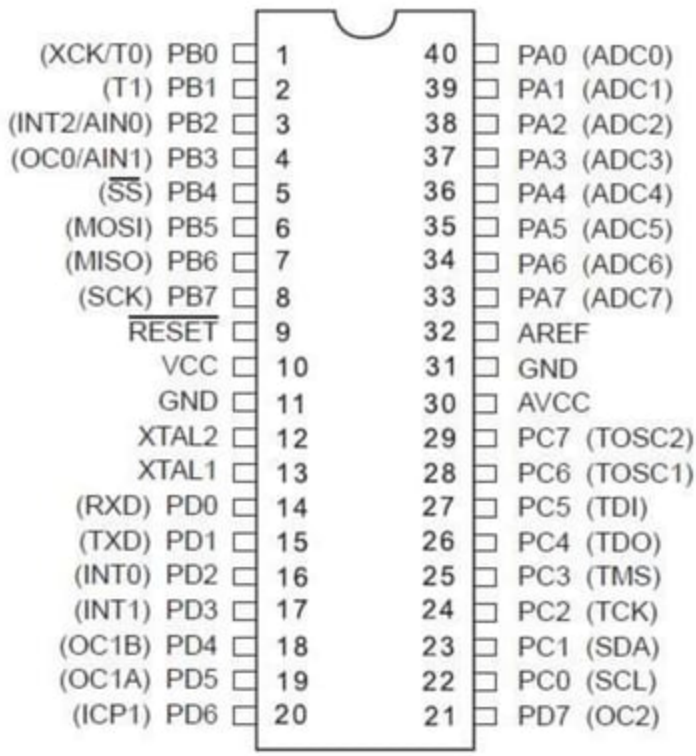

Figure 4.3 Pin Configuration of Atmega 16

Six of the 32 registers can be used as three 16-bit indirect address register pointers for Data Space addressing – enabling efficient address calculations. One of the these address pointers can also be used as an address pointer for look up tables in Flash Program memory. These added function registers are the 16-bit X-register, Y-register, and Z-register.

The ALU supports arithmetic and logic operations between registers or between a constant and a register. Single register operations can also be executed in the ALU. After an arithmetic operation, the Status Register is updated to reflect information about the result of the operation.

Pin Descriptions:

- **VCC:** Digital supply voltage. (+5V)
- **GND:** Ground. (0 V) Note there are 2 ground Pins.
- **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. 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.
- **Port B (PB7 - PB0):** Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). Port B also serves the functions of various special features of the ATmega16 as listed on page 58 of datasheet.
- **Port C (PC7 - PC0):** Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). Port C also serves the functions of the JTAG interface and other special features of the ATmega16 as listed on page 61 of datasheet. 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.
- **Port D (PD7 - PD0):** Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). Port D also serves the functions of various special features of the ATmega16 as listed on page 63 of datasheet.
- **RESET:** 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.
- **XTAL1:** External oscillator pin 1
- **XTAL2:** External oscillator pin 2
- **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 ADC is not used. If the ADC 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.

Digital Input Output Port:

Atmega16 has 32 I/O (Input/Output) pins grouped as A, B, C & D with 8 pins in each group. This group is called as PORT.

- PA0 - PA7 (PORTA)
- PB0 - PB7 (PORTB)
- PC0 - PC7 (PORTC)
- PD0 - PD7 (PORTD)

These are additional function that pin can perform other than I/O. Some of them are

- ADC (ADC0 - ADC7 on PORTA)
- UART (Rx,Tx on PORTD)
- TIMERS (OC0 - OC2)
- SPI (MISO, MOSI, SCK on PORTB)
- External Interrupts (INT0 - INT2)

Registers :

All the configurations in microcontroller is set through 8 bit (1 byte) locations in RAM (RAM is a bank of memory bytes) of the microcontroller called as Registers. All the functions are mapped to its locations in RAM and the value we set at that location that is at that Register configures the functioning of microcontroller. There are total 32 x 8bit registers in Atmega-16. As Register size of this microcontroller is 8 bit, it called as 8 bit microcontroller.

- **ADC**

An analog-to-digital converter (abbreviated ADC, A/D or A to D) is a device that converts a continuous physical quantity (usually voltage) to a digital number that represents the quantity's amplitude.

The conversion involves quantization of the input, so it necessarily introduces a small amount of error. Instead of doing a single conversion, an ADC often performs the conversions ("samples" the input) periodically. The result is a sequence of digital values that have converted a continuous-time and continuous-amplitude analog signal to a discrete-time and discrete-amplitude digital signal.

An ADC is defined by its bandwidth (the range of frequencies it can measure) and its signal to noise ratio (how accurately it can measure a signal relative to the noise it introduces). The actual bandwidth of an ADC is characterized primarily by its sampling rate, and to a lesser extent by how it handles errors such as aliasing. The dynamic range of an ADC is influenced by many factors, including the resolution (the number of output levels it can quantize a signal to), linearity and accuracy (how well the quantization levels match the true analog signal) and jitter (small timing errors that introduce additional noise). The dynamic range of an ADC is often summarized in terms of its effective number of bits(ENOB), the number of bits of each measure it returns that are on average not noise. An ideal ADC has an ENOB equal to its resolution. ADCs are chosen to match the bandwidth and required signal to noise ratio of the signal to be quantized. If an ADC operates at a sampling rate greater than twice the bandwidth of the signal, then perfect reconstruction is possible given an ideal ADC and neglecting quantization error. The presence of quantization error limits the dynamic range of even an ideal ADC, however, if the dynamic range of the ADC exceeds that of the input signal, its effects may be neglected resulting in an essentially perfect digital representation of the input signal.

An ADC may also provide an isolated measurement such as an electronic device that converts an input analog voltage or current to a digital number proportional to the magnitude of the voltage or current. However, some non-electronic or only partially electronic devices, such as rotary encoders, can also be considered ADCs. The digital output may use different coding schemes. Typically the digital output will be a two's complement binary number that is proportional to the input, but there are other possibilities. An encoder, for example, might output a Gray code.

The inverse operation is performed by a digital-to-analog converter (DAC).

The resolution of the converter indicates the number of discrete values it can produce over the range of analog values. The resolution determines the magnitude of the quantization error and therefore determines the maximum possible average signal to noise ratio for an ideal ADC without the use of oversampling. The values are usually stored electronically in binary form, so the resolution is usually expressed in bits. In consequence, the number of discrete values available, or "levels", is assumed to be a power of two. For example, an ADC with a resolution of 8 bits can encode an analog input to one in 256 different levels, since 28

= 256. The values can represent the ranges from 0 to 255 (i.e. unsigned integer) or from −128 to 127 (i.e. signed integer), depending on the application.

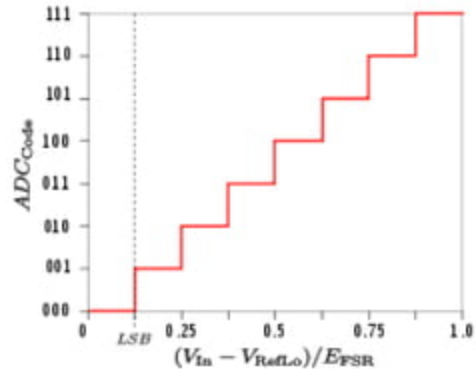

Figure 4.4 An 8-lavel coding scheme

Resolution can also be defined electrically, and expressed in volts. The minimum change in voltage required to guarantee a change in the output code level is called the least significant bit (LSB) voltage. The resolution Q of the ADC is equal to the LSB voltage. The voltage resolution of an ADC is equal to its overall voltage measurement range divided by the number of discrete values:

$$Q = \frac{E_{\text{FSR}}}{2^M - 1},$$

where M is the ADC's resolution in bits and E_{FSR} is the full scale voltage range (also called 'span'). E_{FSR} is given by

$$E_{\text{FSR}} = V_{\text{RefHi}} - V_{\text{RefLow}},$$

where V_{RefHi} and V_{RefLow} are the upper and lower extremes, respectively, of the voltages that can be coded.

Normally, the number of voltage intervals is given by

$$N = 2^M - 1,$$

where M is the ADC's resolution in bits.

That is, one voltage interval is assigned in between two consecutive code levels

Example:

- Coding scheme as in figure 1 (assume input signal x(t) = Acos(t), A = 5V)
- Full scale measurement range = -5 to 5 volts
- ADC resolution is 8 bits: 28 – 1 = 256 – 1 = 255 quantization levels (codes).
- ADC voltage resolution, Q = (10 V − 0 V) / 255 = 10 V / 255 ≈ 0.039 V ≈ 39 mV.

In practice, the useful resolution of a converter is limited by the best signal-to-noise ratio (SNR) that can be achieved for a digitized signal. An ADC can resolve a signal to only a certain number of bits of resolution, called the effective number of bits (ENOB). One effective bit of resolution changes the signal-to-noise ratio of the digitized signal by 6 dB, if the resolution is limited by the ADC. If a preamplifier has been used prior to A/D conversion, the noise introduced by the amplifier can be an important contributing factor towards the overall SNR.

Quantization error :

Quantization error is the noise introduced by quantization in an ideal ADC. It is a rounding error between the analog input voltage to the ADC and the output digitized value. The noise is non-linear and signal-dependent.

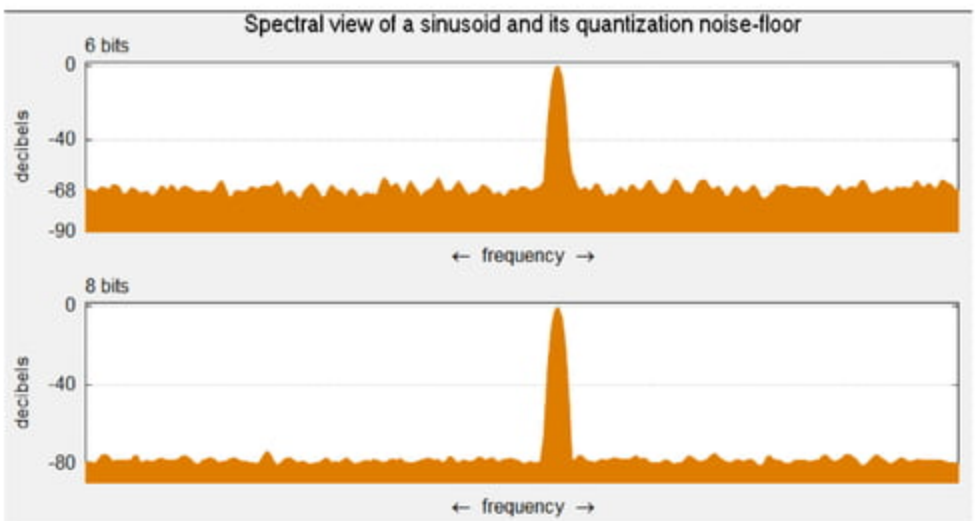

Figure 4.5 Quantization errors of different quantizing bit

In an ideal analog-to-digital converter, where the quantization error is uniformly distributed between −1/2 LSB and +1/2 LSB, and the signal has a uniform distribution covering all quantization levels, the Signal-to-quantization-noise ratio (SQNR) can be calculated from

SQNR= 20 log_10 ⁡〖2^Q 〗≈ 6.02.Q dB

Where Q is the number of quantization bits. For example, a 16-bit ADC has a maximum signal-to-noise ratio of 6.02 × 16 = 96.3 dB, and therefore the quantization error is 96.3 dB below the maximum level. Quantization error is distributed from DC to the Nyquist frequency, consequently if part of the ADC's bandwidth is not used (as inoversampling), some of the quantization error will fall out of band, effectively improving the SQNR. In an oversampled system, noise shaping can be used to further increase SQNR by forcing more quantization error out of the band.

Accuracy:

An ADC has several sources of errors. Quantization error and (assuming the ADC is intended to be linear) non-linearity are intrinsic to any analog-to-digital conversion.

These errors are measured in a unit called the least significant bit (LSB). In the above example of an eight-bit ADC, an error of one LSB is 1/256 of the full signal range, or about 0.4%.

Non-linearity:

All ADCs suffer from non-linearity errors caused by their physical imperfections, causing their output to deviate from a linear function (or some other function, in the case of a deliberately non-linear ADC) of their input. These errors can sometimes be mitigated by calibration, or prevented by testing.

Important parameters for linearity are integral non-linearity (INL) and differential non-linearity (DNL). These non-linearities reduce the dynamic range of the signals that can be digitized by the ADC, also reducing the effective resolution of the ADC.

Jitter:

When digitizing a sine wave

$$x(t) = A\sin(2\pi f_0 t)$$

the use of a non-ideal sampling clock will result in some uncertainty in when samples are recorded. Provided that the actual sampling time uncertainty due to the clock jitter is Δt, the error caused by this phenomenon can be estimated as

$$E_{ap} \leq |x'(t)\Delta t| \leq 2A\pi f_0 \Delta t$$

This will result in additional recorded noise that will reduce the effective number of bits (ENOB) below that predicted by quantization error alone.

Output size (bits)	Signal Frequency						
	1 Hz	1 kHz	10 kHz	1 MHz	10 MHz	100 MHz	1 GHz
8	1,243 µs	1.24 µs	124 ns	1.24 ns	124 ps	12.4 ps	1.24 ps
10	311 µs	311 ns	31.1 ns	311 ps	31.1 ps	3.11 ps	0.31 ps
12	77.7 µs	77.7 ns	7.77 ns	77.7 ps	7.77 ps	0.78 ps	0.08 ps
14	19.4 µs	19.4 ns	1.94 ns	19.4 ps	1.94 ps	0.19 ps	0.02 ps
16	4.86 µs	4.86 ns	486 ps	4.86 ps	0.49 ps	0.05 ps	–
18	1.21 ns	121 ps	6.32 ps	1.21 ps	0.12 ps	–	–

20	304 ps	30.4 ps	1.58 ps	0.16 ps	–	–	–

Table 4.1 Errors in time for different quantization lavel

The error is zero for DC, small at low frequencies, but significant when high frequencies have high amplitudes. This effect can be ignored if it is drowned out by the quantizing error. Jitter requirements can be calculated using the following formula:

$$\Delta t < \frac{1}{2^q \pi f_0},$$

where q is the number of ADC bits.

Clock jitter is caused by phase noise. The resolution of ADCs with a digitization bandwidth between 1 MHz and 1 GHz is limited by jitter.

When sampling audio signals at 44.1 kHz, the anti-aliasing filter should have eliminated all frequencies above 22 kHz. The input frequency (in this case, < 22 kHz kHz), not the ADC clock frequency, is the determining factor with respect to jitter performance.

Sampling rate:

The analog signal is continuous in time and it is necessary to convert this to a flow of digital values. It is therefore required to define the rate at which new digital values are sampled from the analog signal. The rate of new values is called the sampling rate or sampling frequency of the converter.

A continuously varying bandlimited signal can be sampled (that is, the signal values at intervals of time T, the sampling time, are measured and stored) and then the original signal can be exactly reproduced from the discrete-time values by an interpolation formula. The accuracy is limited by quantization error. However, this faithful reproduction is only possible if the sampling rate is higher than twice the highest frequency of the signal. This is essentially what is embodied in the Shannon-Nyquist sampling theorem.

Since a practical ADC cannot make an instantaneous conversion, the input value must necessarily be held constant during the time that the converter performs a conversion (called the conversion time). An input circuit called a sample and hold performs this task—in most cases by using a capacitor to store the analog voltage at the input, and using an electronic switch or gate to disconnect the capacitor from the input. Many ADC integrated circuits include the sample and hold subsystem internally.

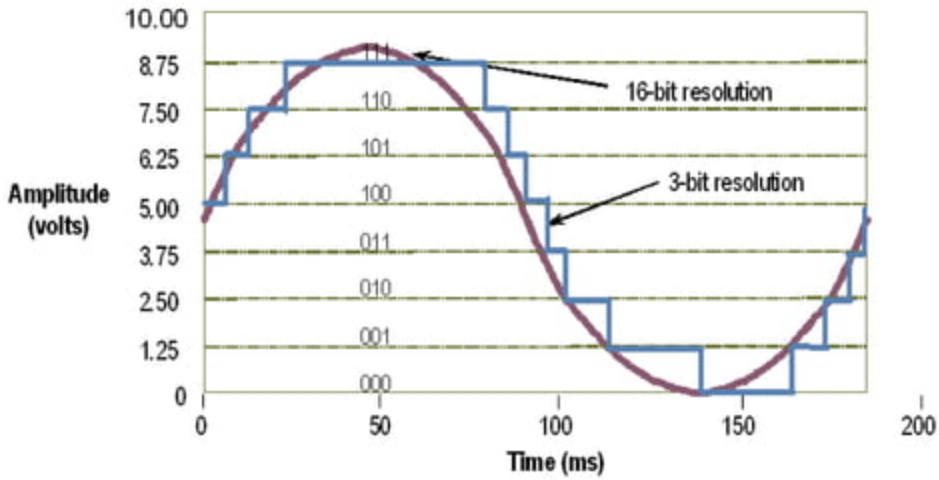

Figure 4.6 Analog to Digital Conversion

2.2 Atmega 16 Development Board User Manual

Overview:

The Atmega 16 Development Board provides a simple and low cost platform for rapidly developing applications based on 40 pin Atmega 16 microcontroller. The board has just the bare minimal external circuitry that is needed to upload and execute programs for compatible AVR microcontrollers. Board can work on 9V to 12V DC supply. It has switches for reset and power. LCD display is used for programming debugging purposes. The I/O s taken on 14 pin FRC connectors are compatible with motor driver circuits of the L298 Driver boards. Besides, the board also has facility to drive 4 servo motors.2 are dedicated for this purpose and the other two are multiplexed with the ADC pins. Additional I/O s are accessible on relimated connectors for custom applications. The firmware is developed in Win-Avr and is loaded using Sina-Prog programming utility.

Contents:

➢ **Specifications:**

- Input Voltage: 9V – 12V DC
- Power Status LED
- Standard 10 pin FRC connector compatible with STK500 programmer
- Can support 3 motor driver circuits (Independent control of 6 DC motors)
- Support for independent control of 4 high torque servo motors
- LCD display for program debugging
- Provision for driving two motors using the relay motor drivers
- Facility for 2 interrupts
- Accessible pins for ADC applications and the facility to use USART for communication.

➢ **PCB Contents:**

- Atmega 16 development board.
- Atmega 16 IC.

➢ **Using the board:**

- Power Supply

The board consists of a 2 pin relimated male connector which is used to connect battery to the development board. The power switch is used to apply this battery power to 7805 voltage regulator which outputs regulated 5V DC for the microcontroller. Once you have the board powered on correctly you will see the power status LED light up. Please observe polarity given on board when connecting

the battery. Any misconnection might lead to permanent damage to the development board.

- Programmer FRC

  This 10 pin FRC is used to load the firmware into the Atmega 128 IC using STK500 programmer. Firmware is loaded using Sina-Prog software tool.

- Motor Driver FRCs

  The Development board has three 14 pin FRC connectors. All of them are compatible with the L298 Motor Driver boards of Robolab Technologies. They are used to connect external motor driver circuits to the development board. Each motor driver provided can drive 2 DC motors. Hence, on one 14 pin FRC you can independently control 2 DC motors using the L298 Motor driver.

- Servo motor connectors

  The two 3 pin relimated connectors associated with 6 pins viz. PB1, PB3 and PA0, PA1 are dedicated for servo motor applications. Servo motor requires a separate high current 5V/6V DC power source. An additional 2 pin relimated connector along with a slider switch is used to connect the same to the development board.

- External interrupts

  A 4 pin relimated connector is provided that can be utilized to use 2 interrupt pins PD2 and PD3. A rising or falling edge (as per set in the program) will cause the controller serve the interrupt service routine for the corresponding interrupt, deviating from the ongoing program flow.

- Proximity and Pneumatics

  A 6 pin relimated connector is provided that is compatible with both the proximity interfacing board as well as the pneumatics driver board. The pins used are PIND3, PIND4, PIND5 and PIND6.

- ADC

  The PORTA of Atmega 16 microcontroller can be configured for ADC applications. The device can serve up to 8 independent analog channels. These 8 pins (PORTA0 to PORTA7) are accessible 14 pin FRC on the top of the board. The ADC parameters like frequency, mode of conversion, Ref voltage are configured in the program.

- LCD

A 14 pin FRC connector labeled as "LCD/L 298" is used to interface standard 16*2 LCD display to Atmega 16 microcontroller. The LCD library is designed such that only 4 of the 8 data lines of LCD are required when printing any character on LCD, thus minimizing the pin requirement. The same connector can also be used for motor driver application if desired.

- I2C connector

  The 4 pin relimated connector with PORTC0 and PORTC1 pins can be used for I2C communication if configured for the same. Many digital sensors like gyroscope can be interfaced with the development board through this connector.

- UART connector

  The 4 pin relimated connector labeled USART with PORTD0 and PORTD1 pins can be used for UART communication if configured for the same. Many wireless modules like bee, Bluetooth, RF module can be interfaced with the development board via this connector.

➢ **Firmware:**

- Open Sinaprog GUI.
- Specify the address of the Hex/EEP File that we want to burn in Microcontroller.
- Press Search Button for initial checking and initialization.
- Press Program under Flash section if the file format is .hex else Press Program under EEPROM section if the file format is .eep

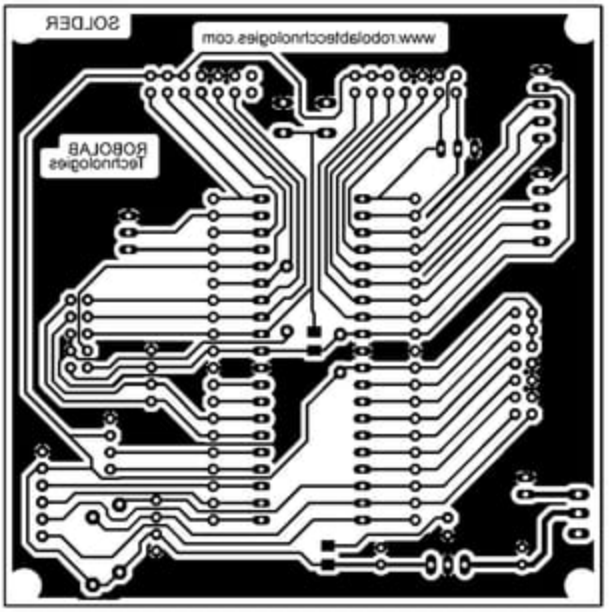

Figure 4.7 Atmega 16 Development Board Routing

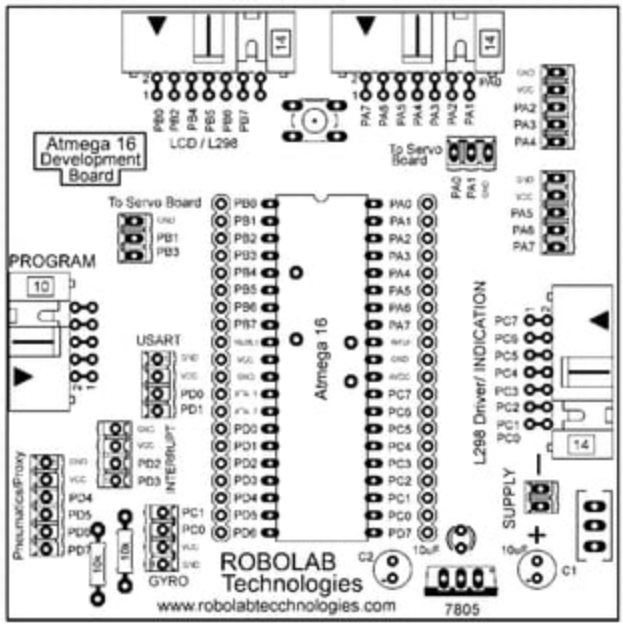

Figure 4.8 Atmega 16 Development Board Pin Out

2.3 Programmer

The AVR Programmer allows to painlessly transfer hex programs to most ATMEL AVR microcontrollers without sacrificing budget and time. It is more reliable than most other simple AVR programmers available out there and can be built in very short amount of time.

AVR programmer consists of in-circuit serial programmer (dongle) and small pcb with a DIP socket where you can fit your microcontroller and have it quickly programmed.

These are simple AVR programmers. The four different programmers for various envilonments: LPT controlled parallel programmer, LPT controlled ISP adapter, COM controlled ISP adapter and COM controlled generic SPI bridge. Additionaly, COM controlled adapters can be used as a communication cable between host PC and target board, it is useful for debugging.

These AVR programmers have no controller on the programmer and directly controlled with port signals. Therefore, programming time is reduced shorter than any other programmers because there is not any communication delay such as command/result sequense between PC and programmer. These programmer can also be used with AVR studio.

Programming method on the AVR

AVR has two different programming modes called Parallel Programming Mode (Parallel Mode) and Serial Downloading Mode (ISP mode).

At the Parallel Mode, the device to be programmed is put on the programmer's socket and +12 volts programming voltage is required to its RESET pin. Communicating between the programmer and the device is done in parallel programming commands, so that the programing speed is two times faster than ISP mode. This programming mode is used to pre-program many devices or/and ISP mode cannot use due to the board design. However, most programmers except STK500 seem not to support this programming mode.

At the ISP Mode, the device communicates via its SPI interface to program and verify. This mode requires only three signal lines without +12 volts programming voltage, so that it can also program in the terget system, this is called ISP (In-System Programming). However, the ISP mode cannot change fuse bits at some devces, and some devices don't have ISP feature. Such devices must be programmed in Parallel Mode. But most AVR programmers use this programming mode even if it has a socket, they have the same problem.

Additionally, there is serial programming mode using +12V programming voltage that called High-Voltage Serial Programming Mode. This programming mode is equivalent to the Paralell Mode and available in only 8/14 pin devices. For details on the each programming mode, please refer to the device data sheets.

Figure 4.9 Programmer

USB programmer for Atmel AVR controllers

Features

- Works under multiple platforms. Linux, Mac OS X and Windows are tested.
- No special controllers or smd components are needed.
- Programming speed is up to 5kBytes/sec.
- SCK option to support targets with low clock speed (< 1,5MHz).
- Planned: serial interface to target (e.g. for debugging).

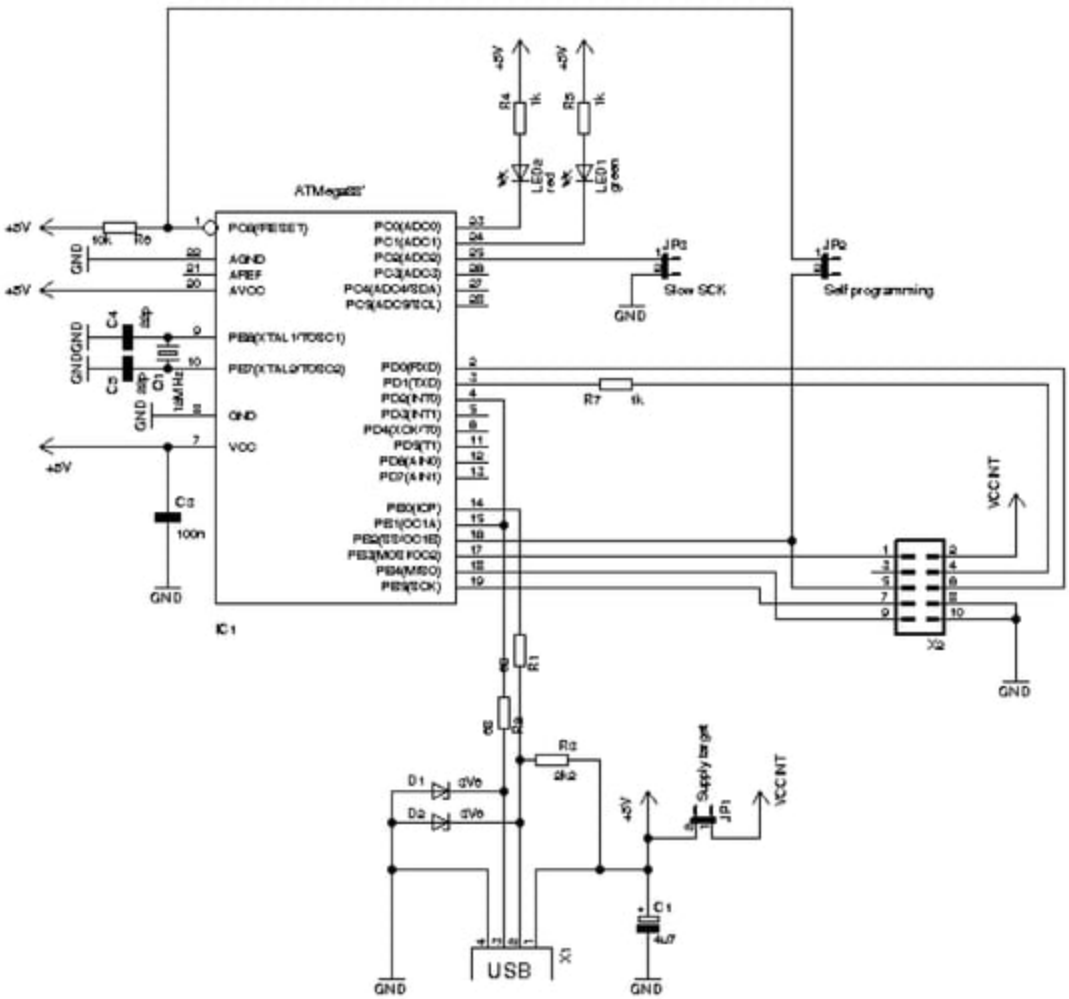

Figure 4.10 Internal circuit diagram of Programmer

Recent desktop PCs in the market tend to omit legacy ports (COM/LPT), especially, the legacy ports on the notebook PC is completely eliminated and they have been replaced with the USB ports. The USBspi is a general purpose SPI bridge attached to USB port. On

the SPI bridge R4, an AVR specific command is added and the programming time (Erase+Write+Verify) of 6.6 seconds at 32 Kbytes is achieved. It is two times faster than COM port adapter.

2.4 Motor Driver

1. L293D:

Description

L293D is a typical Motor driver or Motor Driver IC which allows DC motor to drive on either direction. L293D is a 16-pin IC which can control a set of two DC motors simultaneously in any direction. It means that you can control two DC motor with a single L293D IC. Dual H-bridge Motor Driver integrated circuit (IC).

The l293d can drive small and quiet big motors as well, check the Voltage Specification at the end of this page for more info.

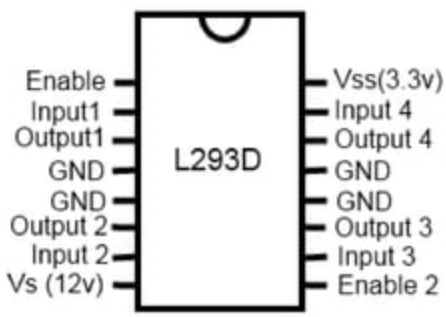

Figure 4.11 Pin Configuration of L293D

Concept

It works on the concept of H-bridge. H-bridge is a circuit which allows the voltage to be flown in either direction. As you know voltage need to change its direction for being able to rotate the motor in clockwise or anticlockwise direction, Hence H-bridge IC are ideal for driving a DC motor.

In a single l293d chip there two h-Bridge circuit inside the IC which can rotate two dc motor independently. Due its size it is very much used in robotic application for controlling DC motors. Given below is the pin diagram of a L293D motor controller.

There are two Enable pins on l293d. Pin 1 and pin 9, for being able to drive the motor, the pin 1 and 9 need to be high. For driving the motor with left H-bridge you need to enable pin 1 to high. And for right H-Bridge you need to make the pin 9 to high. If anyone of the either pin1 or pin9 goes low then the motor in the corresponding section will suspend working. It's like a switch.

Working of L293D

The there 4 input pins for this l293d, pin 2,7 on the left and pin 15 ,10 on the right as shown on the pin diagram. Left input pins will regulate the rotation of motor connected across left side and right input for motor on the right hand side. The motors are rotated on the basis of the inputs provided across the input pins as LOGIC 0 or LOGIC 1.

In simple to provide Logic 0 or 1 across the input pins for rotating the motor.

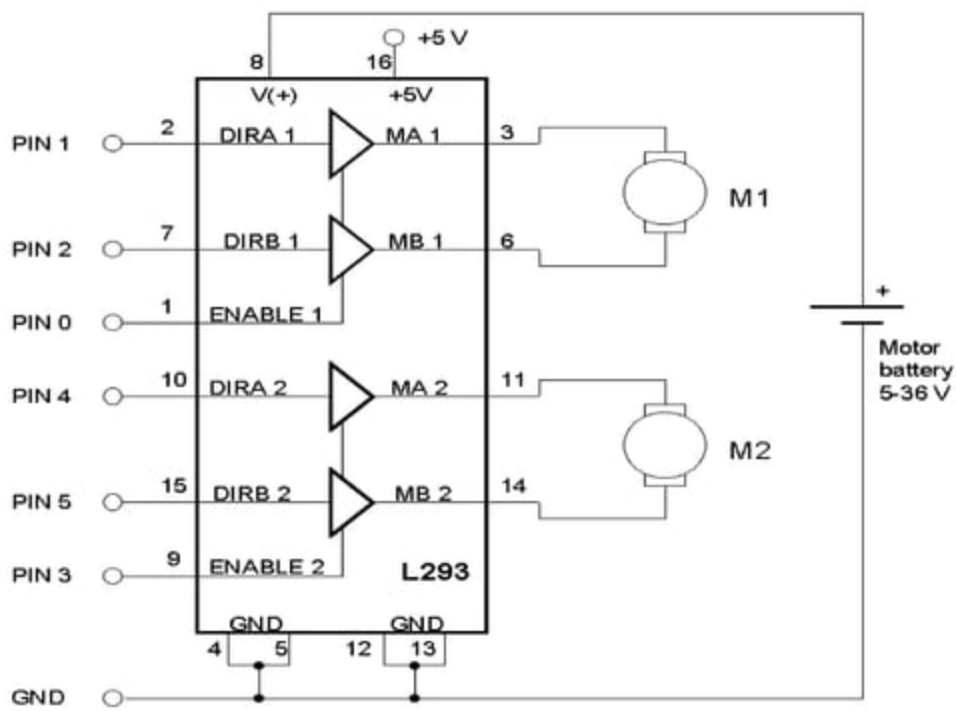

Figure 4.12 Internal Architecture of L293D

L293D Logic Table

Lets consider a Motor connected on left side output pins (pin 3,6). For rotating the motor in clockwise direction the input pins has to be provided with Logic 1 and Logic 0.

- Pin 2 = Logic 1 and Pin 7 = Logic 0 | Clockwise Direction
- Pin 2 = Logic 0 and Pin 7 = Logic 1 | Anticlockwise Direction
- Pin 2 = Logic 0 and Pin 7 = Logic 0 | Idle [No rotation] [Hi-Impedance state]
- Pin 2 = Logic 1 and Pin 7 = Logic 1 | Idle [No rotation]

In a very similar way the motor can also operated across input pin 15,10 for motor on the right hand side.

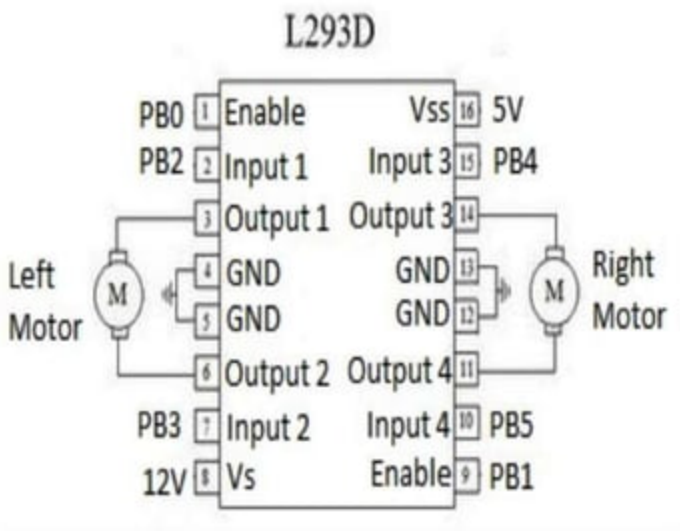

Figure 4.13 Pin Connection of L293D

Voltage Specification

VCC is the voltage that it needs for its own internal operation 5v; L293D will not use this voltage for driving the motor. For driving the motors it has a separate provision to provide motor supply VSS (V supply). L293d will use this to drive the motor. It means if you want to operate a motor at 9V then you need to provide a Supply of 9V across VSS Motor supply.

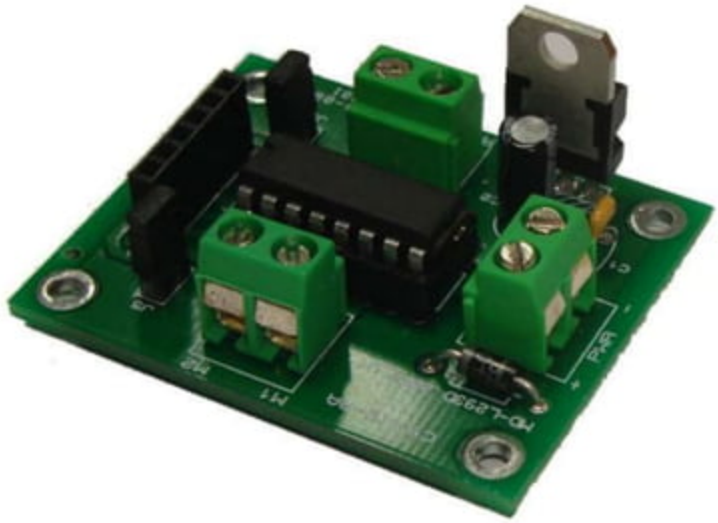

Figure 4.14 L293D Motor Driver

The maximum voltage for VSS motor supply is 36V. It can supply a max current of 600mA per channelSince it can drive motors Up to 36v hence you can drive pretty big motors with this l293d.

VCC pin 16 is the voltage for its own internal Operation. The maximum voltage ranges from 5v and upto 36v.

TIP: Don't Exceed the Vmax Voltage of 36 volts or it will cause damage.

2. Relay Motor Driver

A relay is an electrically operated switch. Many relays use an electromagnet to mechanically operate a switch, but other operating principles are also used, such as solid-state relays. Relays are used where it is necessary to control a circuit by a low-power signal (with complete electrical isolation between control and controlled circuits), or where several circuits must be controlled by one signal. The first relays were used in long distance telegraph circuits as amplifiers: they repeated the signal coming in from one circuit and re-transmitted it on another circuit. Relays were used extensively in telephone exchanges and early computers to perform logical operations.

A type of relay that can handle the high power required to directly control an electric motor or other loads is called a contactor. Solid-state relays control power circuits with no moving parts, instead using a semiconductor device to perform switching. Relays with calibrated operating characteristics and sometimes multiple operating coils are used to protect electrical circuits from overload or faults; in modern electric power systems these functions are performed by digital instruments still called "protective relays".

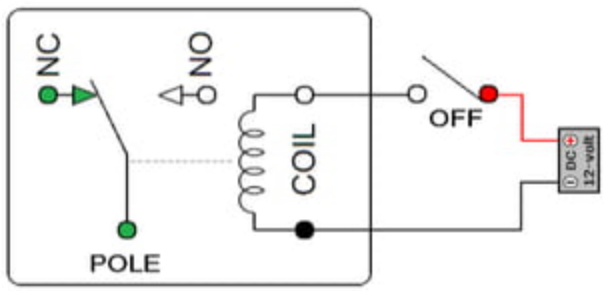

Figure 4.15 Internal Circuit Connection of Relay

Terms associated with relays:

- **Normally Open (NO):** contacts connect the circuit when the relay is activated; the circuit is disconnected when the relay is inactive.
- **Normally Closed(NC):** contacts disconnect the circuit when the relay is activated; the circuit is connected when the relay is inactive.
- **Change Over (CO):** Its the common contact.
- **COIL:** Its the electromagnet coil inside relay.

Figure 4.16 Relay Motor Driver

Relay triggering circuit:

Depending upon a relay's coil rating, some may require current greater than 100mA. If an IC cannot provide this much current, a transistor is used as a switch to trigger the relay as shown below. Don't avoid the protection diode (D1 shown in circuit) as it will protect transistor from back emf induced in relay coil.

Switching speed of a relay is slow, around 10ms. Relays are used to drive an AC load from a small DC circuit, or to drive a high current consuming motors. Have you noticed a sound of tic -tic while car wiper is on, This is the sound of relay inside the car that drives the wiper motor.

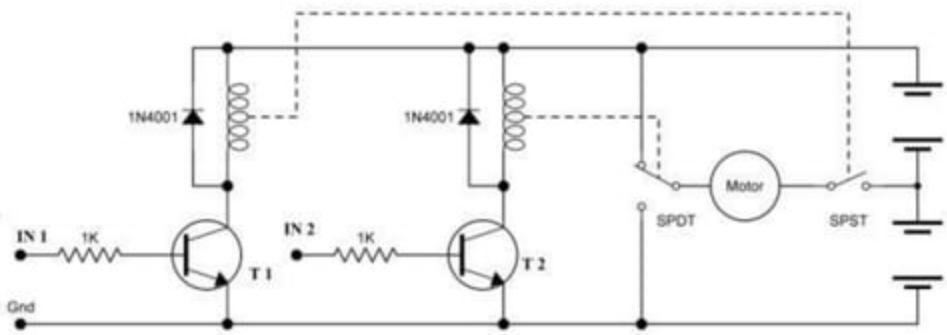

Figure 4.17 Circuit Connection of Relay Driver

2.5 IR Sensors

Infrared (IR) light is electromagnetic radiation with longer wavelengths than those of visible light, extending from the nominal red edge of the visible spectrum at 700 nanometers (nm) to 1 mm. This range of wavelengths corresponds to a frequency range of approximately 430 THz down to 300 GHz. Most of the thermal radiation emitted by objects near room temperature is infrared.

Infrared radiation was discovered in 1800 by astronomer William Herschel, who discovered a type of invisible radiation in the light spectrum beyond red light, by means of its effect upon a thermometer. Slightly more than half of the total energy from the Sun was eventually found to arrive on Earth in the form of infrared. The balance between absorbed and emitted infrared radiation has a critical effect on Earth's climate.

Infrared light is emitted or absorbed by molecules when they change their rotational vibrational movements. Infrared energy elicits vibrational modes in a molecule through a change in the dipole moment, making it a useful frequency range for study of these energy states for molecules of the proper symmetry. Infrared spectroscopy examines absorption and transmission of photons in the infrared energy range.

Infrared light is used in industrial, scientific, and medical applications. Night-vision devices using active near-infrared illumination allow people or animals to be observed without the observer being detected. Infrared astronomy uses sensor-equipped telescopes to penetrate dusty regions of space, such as molecular clouds; detect objects such as planets, and to view highly red-shifted objects from the early days of the universe. Infrared thermal-imaging cameras are used to detect heat loss in insulated systems, to observe changing blood flow in the skin, and to detect overheating of electrical apparatus.

Thermal-infrared imaging is used extensively for military and civilian purposes. Military applications include target acquisition, surveillance, night vision, homing and tracking. Humans at normal body temperature radiate chiefly at wavelengths around 10 µm (micrometers). Non-military uses include thermal efficiency analysis, environmental monitoring, industrial facility inspections, remote temperature sensing, short-ranged wireless communication, spectroscopy, and weather forecasting.

Light comparison

Name	Wavelength	Frequency (Hz)	Photon Energy (eV)
Gamma ray	less than 0.01 nm	more than 30 EHz	124 keV – 300+ GeV
X-Ray	0.01 nm – 10 nm	30 EHz – 30 PHz	124 eV – 124 keV
Ultraviolet	10 nm – 380 nm	30 PHz – 790 THz	3.3 eV – 124 eV
Visible	380 nm–700 nm	790 THz – 430 THz	1.7 eV – 3.3 eV
Infrared	700 nm – 1 mm	430 THz – 300 GHz	1.24 meV – 1.7 eV
Microwave	1 mm – 1 meter	300 GHz – 300 MHz	1.24 µeV – 1.24 meV
Radio	1 mm – 100,000 km	300 GHz – 3 Hz	12.4 feV – 1.24 meV

Table 4.2 Frequency band of different light

Natural infrared

Sunlight, at an effective temperature of 5,780 kelvins, is composed of nearly thermal-spectrum radiation that is slightly more than half infrared. At zenith, sunlight provides an irradiance of just over 1 kilowatts per square meter at sea level. Of this energy, 527 watts is infrared radiation, 445 watts is visible light, and 32 watts is ultraviolet radiation.

On the surface of Earth, at far lower temperatures than the surface of the Sun, almost all thermal radiation consists of infrared in various wavelengths. Of these natural thermal radiation processes only lightning and natural fires are hot enough to produce much visible energy, and fires produce far more infrared than visible-light energy.

In general, objects emit infrared radiation across a spectrum of wavelengths, but sometimes only a limited region of the spectrum is of interest because sensors usually collect radiation only within a specific bandwidth. Thermal infrared radiation also has a maximum emission wavelength, which is inversely proportional to the absolute temperature of object, in accordance with Wien's displacement law.

Figure 4.18 IR Sensor

Feature summary

- operating voltage: 4.5 V to 5.5 V
- average current consumption: 33 mA (typical)
- distance measuring range: 20 cm to 150 cm (8″ to 60″)
- output type: analog voltage
- output voltage differential over distance range: 2.0 V (typical)

- update period: 38 ± 10 ms
- package size: 29.5 × 13.0 × 21.5 mm (1.16″ × 0.5″ × 0.85″)
- weight: 4.8 g (0.17 oz)

Characteristics of IR Sensor

The relationship between the sensor's output voltage and the inverse of the measured distance is approximately linear over the sensor's usable range. The GP2Y0A02YK datasheet (703k pdf) contains a plot of analog output voltage as a function of the inverse of distance to a reflective object. You can use this plot to convert the sensor output voltage to an approximate distance by constructing a best-fit line that relates the inverse of the output voltage (V) to distance (cm).

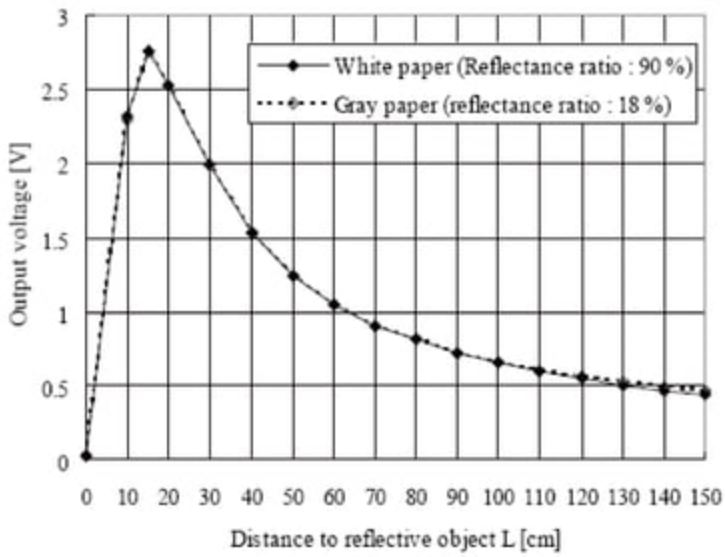

Figure 4.19 Linearizing of Output of Sensor

2.6 DC Motor

A DC motor in simple words is a device that converts direct current(electrical energy) into mechanical energy. It's of vital importance for the industry today, and is equally important for engineers to look into the working principle of DC motor in details that has been discussed in this article. In order to understand the operating principle of dc motor we need to first look into its constructional feature.

The very basic construction of a dc motor contains a current carrying armature which is connected to the supply end through commulator segments and brushes and placed within the north south poles of a permanent or an electro-magnet as shown in the diagram below.

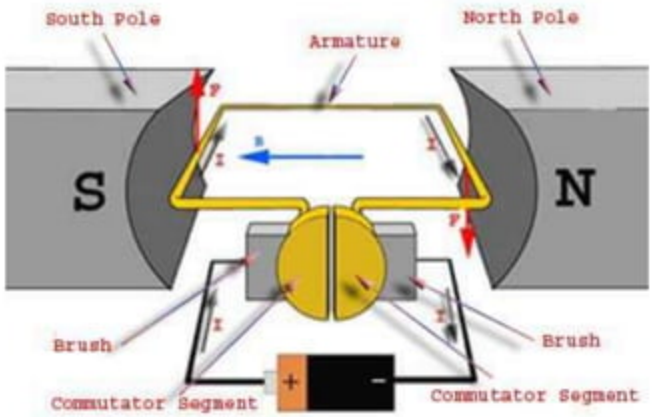

Figure 4.20 Construction of DC Motor

Now to go into the details of the operating principle of DC motor its important that we have a clear understanding of Fleming's left hand rule to determine the direction of force acting on the armature conductors of dc motor.

Fleming's left hand rule says that if we extend the index finger, middle finger and thumb of our left hand in such a way that the electric current carrying conductor is placed in a magnetic field (represented by the index finger) is perpendicular to the direction of current (represented by the middle finger), then the conductor experiences a force in the direction (represented by the thumb) mutually perpendicular to both the direction of field and the current in the conductor.

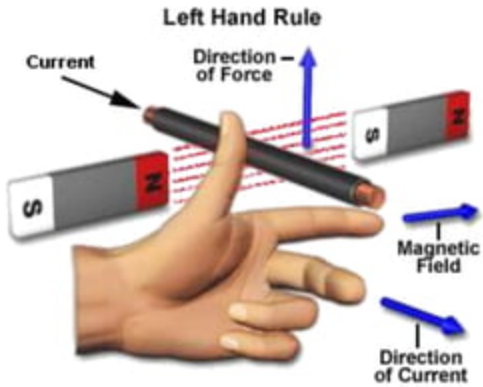

Figure 4.21 Fleming's Left Hand Rule

For clear understanding the principle of DC motor we have to determine the magnitude of the force, by considering the diagram below.

We know that when an infinitely small charge dq is made to flow at a velocity 'v' under the influence of an electric field E, and a magnetic field B, then the Lorentz Force dF experienced by the charge is given by:-

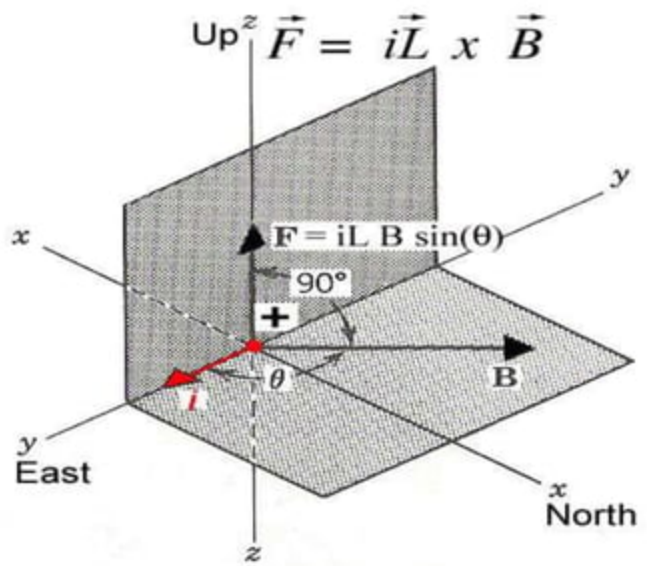

Figure 4.22 Magnetic Flux due to Current Flow

We know that when an infinitely small charge dq is made to flow at a velocity 'v' under the influence of an electric field E, and a magnetic field B, then the Lorentz Force dF experienced by the charge is given by:-

BIL

$dF = dq(E + v \times B)$

For the operation of dc motor, considering $E = 0$

$\therefore dF = dq\ v \times B$

i.e. it's the cross product of dq v and magnetic field B.

or $dF = dq\ (dL/dt) \times B$ $[v = dL/dt]$

Where dL is the length of the conductor carrying charge q.

Or $dF = (dq/dt)\ dL \times B$

or $dF = I\ dL \times B$ [Since, current $I = dq/dt$]

or $F = IL \times B = ILB \sin\theta$

or $F = BIL \sin\theta$

From the 1st diagram we can see that the construction of a DC motor is such that the direction of electric current through the armature conductor at all instance is perpendicular to the field. Hence the force acts on the armature conductor in the direction perpendicular to the both uniform field and current is constant.

i.e. $\theta = 90°$

So if we take the current in the left hand side of the armature conductor to be I, and current at right hand side of the armature conductor to be − I, because they are flowing in the opposite direction with respect to each other.

Then the force on the left hand side armature conductor, $Fl = BIL \sin 90° = BIL$

Similarly force on the right hand side conductor $Fr = B(-I)L.\sin 90° = -BIL$

$\therefore$ we can see that at that position the force on either side is equal in magnitude but opposite in direction. And since the two conductors are separated by some distance w = width of the armature turn, the two opposite forces produces a rotational force or a torque that results in the rotation of the armature conductor.

Now let's examine the expression of torque when the armature turn crate an angle of α with its initial position.

The torque produced is given by

Torque = force, tangential to the direction of armature rotation X distance.

Or $\tau = F\cos\alpha.w$

Or $\tau = BIL\ w\ \cos\alpha$

Where α is the angle between the plane of the armature turn and the plane of reference or the initial position of the armature which is here along the direction of magnetic field.

The presence of the term cosα in the torque equation very well signifies that unlike force the torque at all position is not the same. It in fact varies with the variation of the angle α. To explain the variation of torque and the principle behind rotation of the motor let us do a stepwise analysis.

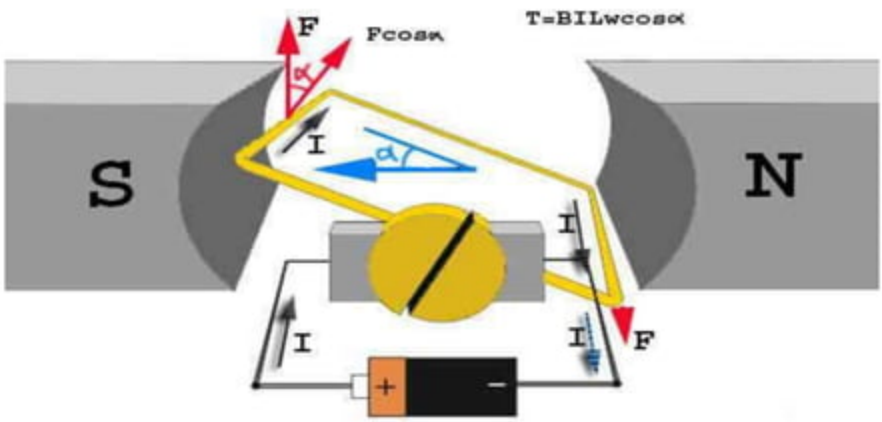

Figure 4.23 Direction of Rotation

Step 1:

Initially considering the armature is in its starting point or reference position where the angle $\alpha = 0$.

$\therefore \tau = BIL\ w\ \cos 0 = BILw$

Since α = 0, the term cos α = 1, or the maximum value, hence torque at this position is maximum given by τ = BILw. This high starting torque helps in overcoming the initial inertia of rest of the armature and sets it into rotation.

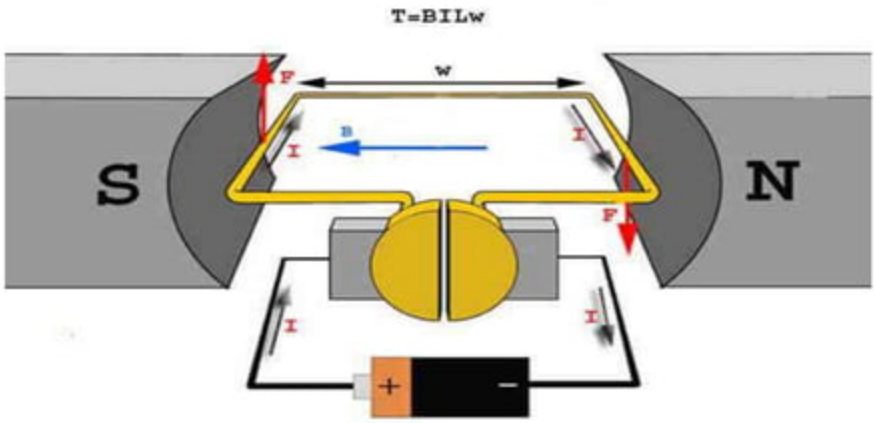

Figure 4.24 Direction of Rotation for α =0

Step 2:

Once the armature is set in motion, the angle α between the actual position of the armature and its reference initial position goes on increasing in the path of its rotation until it becomes 90° from its initial position. Consequently the term cosα decreases and also the value of torque.

The torque in this case is given by τ = BILwcosα which is less than BIL w when α is greater than 0°

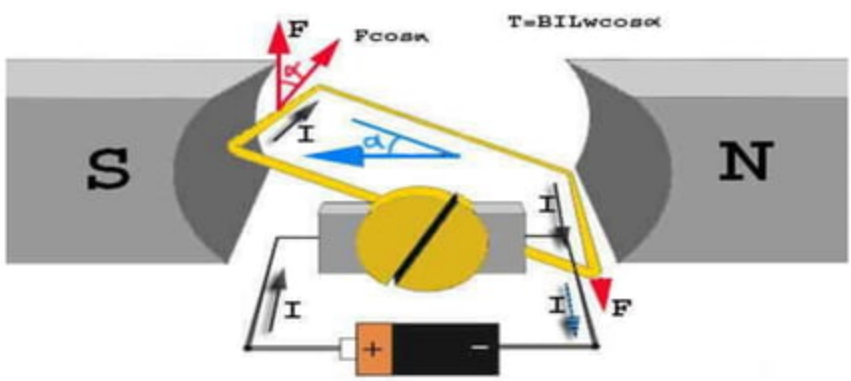

Figure 4.25 Direction of Rotation where α not 0

Step 3:

In the path of the rotation of the armature a point is reached where the actual position of the rotor is exactly perpendicular to its initial position, i.e. $\alpha = 90°$, and as a result the term $\cos\alpha = 0$.

The torque acting on the conductor at this position is given by.

$\tau = BILw\cos 90° = 0$

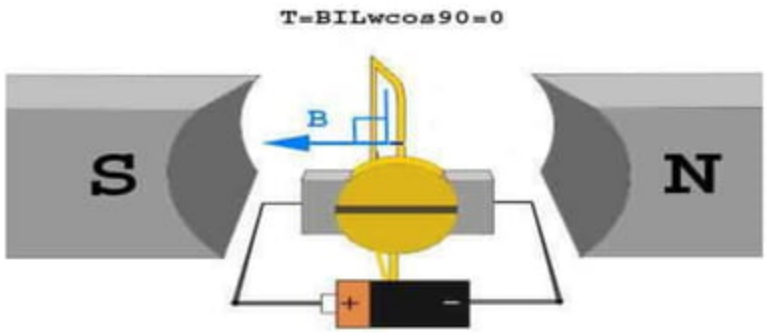

Figure 4.26 Direction of Rotation for α=90 degree

i.e. virtually no rotating torque acts on the armature at this instance. But still the armature does not come to a standstill, this is because of the fact that the operation of dc motor has been engineered in such a way that the inertia of motion at this point is just enough to overcome this point of null torque. Once the rotor crosses over this position the angle between the actual position of the armature and the initial plane again decreases and torque starts acting on it again.

Types

The direct current motor or the DC motor has a lot of application in today's field of engineering and technology. Starting from an electric shaver to parts of automobiles, in all small or medium sized motoring applications DC motors come handy. And because of its wide range of application different functional types of dc motor are available in the market for specific requirements.

The types of DC motor can be listed as follows

- ❖ DC motor
- ➢ Permanent Magnet DC Motor
- ➢ Separately Excited DC Motor
- ➢ Self Excited DC Motor
  - Shunt Wound DC Motor

- Series Wound DC Motor
- Compound Wound DC Motor
  - Cumulative compound DC motor
    - Short shunt DC Motor
    - Long shunt DC Motor
  - Differential Compound DC Motor
    - Short Shunt DC Motor
    - Long Shunt DC Motor

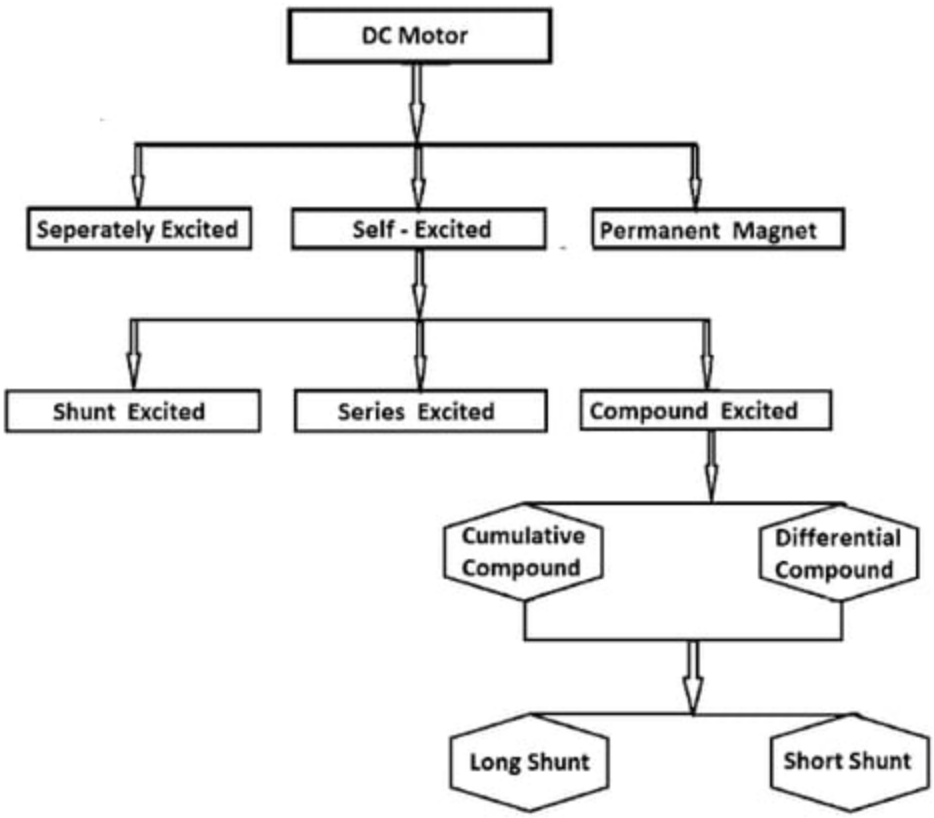

Figure 4.27 Types of DC Motor

2.7 Joystick

A joystick is an input device consisting of a stick that pivots on a base and reports its angle or direction to the device it is controlling. A joystick, also known as the control column, is the principal control device in the cockpit of many civilian and military aircraft, either as

a center stick or side-stick. It often has supplementary switches to control various aspects of the aircraft's flight.

Joysticks are often used to control video games, and usually have one or more push-buttons whose state can also be read by the computer. A popular variation of the joystick used on modern video game consoles is the analog stick. Joysticks are also used for controlling machines such as cranes, trucks, underwater unmanned vehicles, wheelchairs, surveillance cameras, and zero turning radius lawn mowers. Miniature finger-operated joysticks have been adopted as input devices for smaller electronic equipment such as mobile phones.

Figure 4.28 Image of Joystick

The basic idea of a joystick is to translate the movement of a plastic stick into electronic information a computer can process. Joysticks are used in all kinds of machines, including F-15 fighter jets, backhoes and wheelchairs. In this article, we'll be focusing on computer joysticks, but the same principles apply to other sorts of joysticks.

The various joystick technologies differ mainly in how much information they pass on. The simplest joystick design, used in many early game consoles, is just a specialized electrical switch.

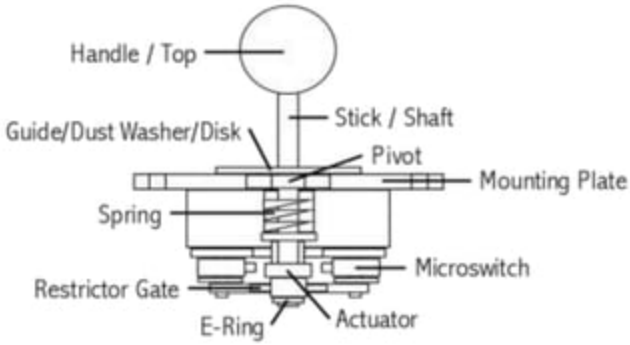

Figure 4.29 Structure of Joystick

This basic design consists of a stick that is attached to a plastic base with a flexible rubber sheath. The base houses a circuit board that sits directly underneath the stick. The circuit board is made up of several "printed wires," which connect to several contact terminals. Ordinary wires extend from these contact points to the computer.

The printed wires form a simple electrical circuit made up of several smaller circuits. The circuits just carry electricity from one contact point to another. When the joystick is in the neutral position -- when you're not pushing one way or another -- all but one of the individual circuits are broken. The conductive material in each wire doesn't quite connect, so the circuit can't conduct electricity.

Each broken section is covered with a simple plastic button containing a tiny metal disc. When you move the stick in any direction, it pushes down on one of these buttons, pressing the conductive metal disc against the circuit board. This closes the circuit -- it completes the connection between the two wire sections. When the circuit is closed, electricity can flow down a wire from the computer (or game console), through the printed wire, and to another wire leading back to the computer.

2.8 Pump

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

Pumps operate by some mechanism (typically reciprocating or rotary), and consume energy to perform mechanical work by moving the fluid. Pumps operate via many

energy sources, including manual operation, electricity, engines, or wind power, come in many sizes, from microscopic for use in medical applications to large industrial pumps.

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

In biology, many different types of chemical and bio-mechanical pumps have evolved, and biomimicry is sometimes used in developing new types of mechanical pumps.

Figure 4.30 Pump

Pump efficiency

Pump efficiency is defined as the ratio of the power imparted on the fluid by the pump in relation to the power supplied to drive the pump. Its value is not fixed for a given pump, efficiency is a function of the discharge and therefore also operating head. For centrifugal pumps, the efficiency tends to increase with flow rate up to a point midway through the operating range (peak efficiency) and then declines as flow rates rise further. Pump performance data such as this is usually supplied by the manufacturer before pump selection. Pump efficiencies tend to decline over time due to wear(e.g. increasing clearances as impellers reduce in size).

When a system design includes a centrifugal pump, an important issue it its design is matching the head loss-flow characteristic with the pump so that it operates at or close to the point of its maximum efficiency.

Pump efficiency is an important aspect and pumps should be regularly tested. Thermodynamic pump testing is one method.

Pumping power

The power imparted into a fluid increases the energy of the fluid per unit volume. Thus the power relationship is between the conversion of the mechanical energy of the pump mechanism and the fluid elements within the pump. In general, this is governed by a series of simultaneous differential equations, known as the Navier–Stokes equations. However a more simple equation relating only the different energies in the fluid, known as Bernoulli's equation can be used. Hence the power, P, required by the pump:

$$P = \frac{\Delta P Q}{\eta}$$

where ΔP is the change in total pressure between the inlet and outlet (in Pa), and Q, the fluid flowrate is given in m^3/s. The total pressure may have gravitational, static pressure and kinetic energy components; i.e. energy is distributed between change in the fluid's gravitational potential energy (going up or down hill), change in velocity, or change in static pressure. η is the pump efficiency, and may be given by the manufacturer's information, such as in the form of a pump curve, and is typically derived from either fluid dynamics simulation (i.e. solutions to the Navier–Stokes for the particular pump geometry), or by testing. The efficiency of the pump depends upon the pump's configuration and operating conditions (such as rotational speed, fluid density and viscosity etc.)

$$\Delta P = \frac{(v_2^2 - v_1^2)}{2} + \Delta z g + \frac{\Delta p_{\text{static}}}{\rho}$$

For a typical "pumping" configuration, the work is imparted on the fluid, and is thus positive. For the fluid imparting the work on the pump (i.e. a turbine), the work is negative power required to drive the pump is determined by dividing the output power by the pump efficiency. Furthermore, this definition encompasses pumps with no moving parts, such as a siphon.

Description:

This vacuum pump really sucks! This pump operates at 12V and has enough suction for most small projects. We even used one to make our very own universal gripper using coffee grounds, balloon and some other parts.

Features:

- 12V motor
- 12W operation
- 1/4" barbs
- 0-16" Hg vacuum range

CHAPTER 5

CODE

```
#include <mega16.h>

#include <delay.h>

#define mode PIND.0
#define pump PIND.1



// Declare your global variables here

 void forward()     //Forward
 {
 PORTC.3=1;
 PORTC.6=1;

 PORTC.2=0;
 PORTC.4=1;
 PORTC.5=0;
 PORTC.7=1;
   }

 void reverse()     //Reverse
 {
 PORTC.3=1;
 PORTC.6=1;

 PORTC.2=1;
 PORTC.4=0;
 PORTC.5=1;
 PORTC.7=0;
 }
```

```
void brake()       //Brake
{
PORTC.3=1;
PORTC.6=1;

PORTC.2=1;
PORTC.4=1;
PORTC.5=1;
PORTC.7=1;
}
void radial_right()       //Right
{
PORTC.3=1;
PORTC.6=1;

PORTC.2=0;
PORTC.4=0;
PORTC.5=0;
PORTC.7=1;
}
void radial_left()         //Left
{
PORTC.3=1;
PORTC.6=1;

PORTC.2=0;
PORTC.4=1;
PORTC.5=0;
PORTC.7=0;
}

void pump_on()              //Pump on
{
PORTD.5=0;

PORTD.4=0;
PORTD.6=1;
}
 void pump_off()            //pump off
{
PORTD.5=0;

PORTD.4=1;
PORTD.6=1;
}

void main(void)
{
// Declare your local variables here
```

```
// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTA=0xFF;
DDRA=0x00;

// Port B initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTB=0x00;
DDRB=0x00;

// Port C initialization
// Func7=Out Func6=In Func5=Out Func4=Out Func3=In Func2=Out Func1=In
Func0=In
// State7=0 State6=P State5=0 State4=0 State3=P State2=0 State1=T State0=T
PORTC=0x48;
DDRC=0xB4;

// Port D initialization
// Func7=In Func6=Out Func5=In Func4=Out Func3=In Func2=In Func1=In
Func0=In
// State7=T State6=0 State5=P State4=0 State3=T State2=T State1=T State0=T
PORTD=0x03;   //00100001  // 0B 0000 0011
DDRD=0x70;   //01110000

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=FFh
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer1 Stopped
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
```

```
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer2 Stopped
// Mode: Normal top=FFh
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=0x00;
MCUCSR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;

// USART initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART Receiver: On
// USART Transmitter: On
// USART Mode: Asynchronous
// USART Baud Rate: 9600
UCSRA=0x00;
UCSRB=0x18;
UCSRC=0x86;
UBRRH=0x00;
UBRRL=0x33;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// ADC initialization
// ADC Clock frequency: 1000.000 kHz
// ADC Voltage Reference: AVCC pin
```

```
// ADC Auto Trigger Source: ADC Stopped
// Only the 8 most significant bits of
// the AD conversion result are used
ADMUX=ADC_VREF_TYPE & 0xff;
ADCSRA=0x83;


while(1)
{
if(mode==0)
{
   if(PIND.1==1)
   {
   PORTD.5=0;

    PORTD.4=1;
    PORTD.6=1;   //pump off
    }
    else
    {

    PORTD.5=0;  //enable

    PORTD.4=0;    //pump on
    PORTD.6=1;

    }

   if (read_adc(0)>200)   //LEFT MOTOR
     {
     PORTC.5=0;   // FRWD
     PORTC.7=1;

     }
     else  if (read_adc(0)<50)
     {
     PORTC.5=1;            // REVERSE
     PORTC.7=0;

     }
     else
      {
      PORTC.5=1;            // BRAKE
      PORTC.7=1;

       }
       if (read_adc(1)>200) //RIGHT  motor
       {
       PORTC.2=0;              //FRWG
       PORTC.4=1;
```

```
}
else if (read_adc(1)<50) //REVERSE
{
PORTC.2=1;
PORTC.4=0;

}
else
{
PORTC.2=1;
PORTC.4=1;

}
}

else
{                                   // auto mode

if (read_adc(6)<70)                 //front sensor
{
forward();
delay_ms(1000);
brake();
pump_on();
delay_ms(2000);
pump_off();
}

else if (read_adc(4)<70)            // left 1st sensor
{
radial_left();
while (read_adc(6)>70);
brake();
pump_on();
delay_ms(2000);
pump_off();

}

else if (read_adc(7)<70)            // right 1nd sensor
{
radial_right();
while (read_adc(6)>70);
brake();
pump_on();
delay_ms(2000);
pump_off();

}
```

```
    }
}       //while ends
```

CHAPTER 6

RESULT

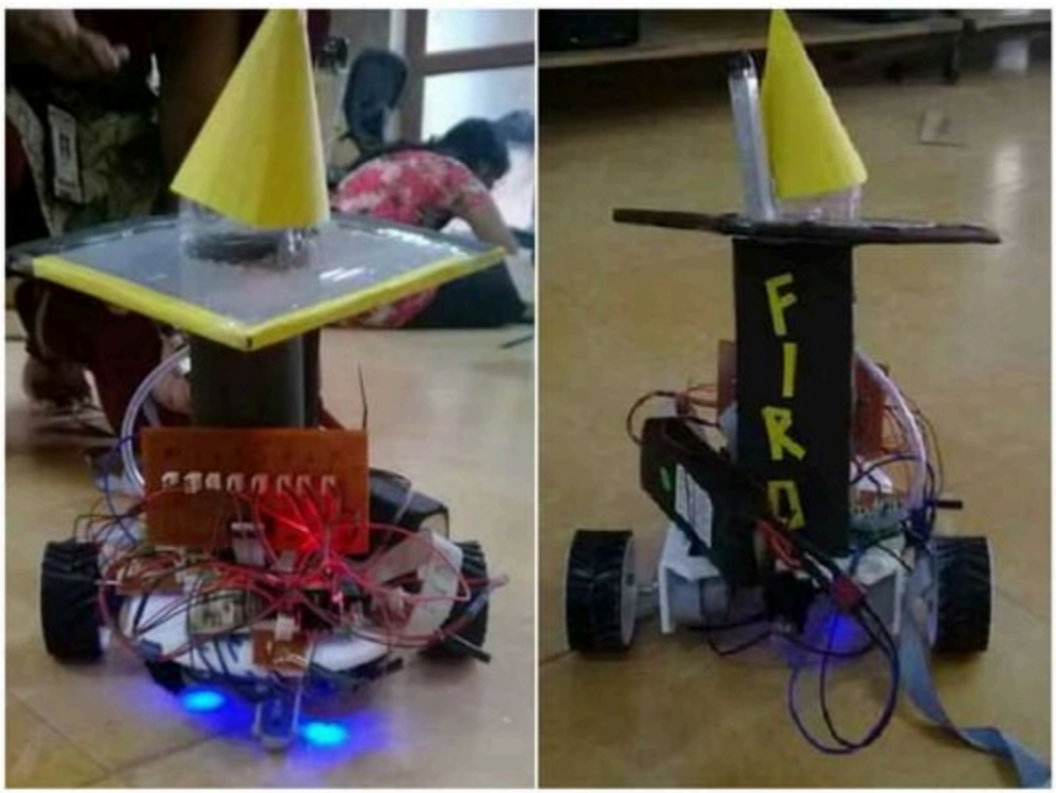

Figure 6.1 Photocopy of Fire Detector & Extinguisher Robot

The Fire Detector and Extinguisher Robot has been successfully executed. In its automatic mode it can detect the fire and try to extinguish it and also by using its manual mode we can extinguish the fire.

Applications:

- Can be used in record maintaining rooms where fire can cause lose of valuable data.
- Can be used in Server rooms for immediate action in case of fire
- Can be used in extinguishing fire where probability of explosion is high. For eg. Hotel kitchens, LPG/CNG gas stores, etc.
- Every working environment requiring permanent operator's attention. -At power plant control rooms. -At captain bridges. -At flight control centers.

Advantages:

- Prevention from dangerous incidents
- Minimization of
  - ecological consequences
  - financial loss
  - a threat to a human life
- Needs no micro-controller programming.
- The reconstruction of the curse of operator's work

Disadvantages:

- Doesn't predict nor interfere with operator's thoughts.
- Cannot force directly the operator to work

CHAPTER 7

FUTURE SCOPE & CONCLUSION

7.1 Future Scope

The project has been motivated by the desire to design a system that can detect fires and take appropriate action, without any human intervention. The development of sensor networks and the maturity of robotics suggests that we can use mobile agents for tasks that involve perception of an external stimulus and reacting to the stimulus, even when the reaction involves a significant amount of mechanical actions. This provides us the opportunity to pass on to robots tasks that traditionally humans had to do but were inherently life- threatening.

Fire-fighting is an obvious candidate for such automation. Given the number of lives lost regularly in fire- fighting, the system we envision is crying for adoption. Our experience suggests that designing a fire-fighting system with sensors and robots is within the reach of the current sensor network and mobile agent technologies. Furthermore, we believe that the techniques developed in this work will carry over to other areas involving sensing and reacting to stimulus, where we desire to replace the human with an automated mobile agent.

Of course, this project has only scratched the surface. As in the design simplifications and the implementation constraints in suggest, our project is very much a proof-of-concept. In particular, a practical autonomous fire-fighting system must include a collection of robots, communicating and cooperating in the mission; furthermore, such a system requires facilities for going through obstacles in the presence of fire, and ability to receive instructions on-the-fly during an operation. All such concerns were outside the scope of this project.

However, there has been research on many of these pieces in different contexts, e.g. coordination among mobile agents, techniques for detecting and avoiding obstacles, on-the-fly communication between humans and mobile agents, etc. It will be both interesting and challenging to put all this together into a practical, autonomous fire-fighting service.

7.2 Conclusion

This paper has presented a unique vision of the concepts which are used in this particular field. It aims to promote technology innovation to achieve a reliable and efficient outcome from the various instruments. With a common digitalized platform, these latest instruments will enable increased flexibility in control, operation, and expansion; allow for embedded intelligence, essentially foster the resilience of the instruments; and eventually benefit the customers with improved services, reliability and increased convenience. The nineties witnessed quantum leaps interface designing for improved man machine interactions. The Mechatronics application ensures a convenient way of simplifying the life by providing more delicate and user friendly facilities in computing devices. Now that we have proven the method, the next step is to improve the hardware. Instead of using cumbersome modules to gather information about the user, it will be better to use smaller and less intrusive units. The day is not far when this technology will push its way into your house hold, making you more lazy.

This paper presents the major features and functions of the various concepts that could be used in this field in detail through various categories. Since this initial work cannot address everything within the proposed framework and vision, more research and development efforts are needed to fully implement the proposed framework through a joint effort of various entities. This autonomous robot successfully performs the task of a fire fighter in a simulated house fire. Benefited from this technology, since the expense of activating other types of fire extinguishers may outweigh that of a robot, where product stock could be damaged by imprecise fire control methods.

REFERENCE

1. Robotics Fundamental Concepts And Analysis - Ashitava Ghosal
2. Robotics And Control – Mittal
3. Introduction to Robotics - SAEED B.NIKU
4. http://www.electrical4u.com/
5. http://electronics.howstuffworks.com/
6. http://elm-chan.org/
7. https://www.google.co.in
8. http://en.wikipedia.org
9. http://www.slideshare.net