CONTROLLING HOME APPLIANCES USING INTERNET

Project Report submitted in partial fulfillment of the requirement for the degree of

Bachelor of Technology

In

Information Technology

Under the Supervision of

Mr. Amol Vasudeva

Assistant Professor

By

ANIRUDH KAUSHIK

(111478)

Jaypee University of Information and Technology

Waknaghat, Solan– 173234, Himachal Pradesh

Certificate

This is to certify that project report entitled "**Controlling Home Appliances Using Internet**", submitted by **Anirudh Kaushik** in partial fulfillment for the award of degree of Bachelor of Technology in Information Technology in Jaypee University of Information Technology, Waknaghat, Solan has been carried out under my supervision.

This work has not been submitted partially or fully to any other University or Institute for the award of this or any other degree or diploma.

Date: 8th May 2015 **Mr. Amol Vasudeva**

(Assistant Professor)

Acknowledgement

No venture can be completed without the blessings of the Almighty. I consider it my bounded duty to bow to Almighty whose kindness has always inspired me on the right path. I would like to take the opportunity to thank all the people who helped us in any form in the completion of my project "**Controlling Home Appliances using Internet**".

It would not have been possible to see through the project without the guidance and constant support of my project guide Mr. Amol Vasudeva. For his coherent guidance, careful supervision, critical suggestions and encouragement I feel fortunate to be taught by him.

I would like to express my gratitude towards Brig. (Retd) Balbir Singh, Director JUIT, for having trust in me. I owe my heartiest thanks to Prof. Dr. RMK Sinha Dean (CSE and IT) and Head Dept. of IT and Prof. Brig. (Retd.) S.P. Ghrera (HOD, CSE Dept.), who has always inspired me to work towards my aim. Also, I am grateful to the whole faculty who helped us in making us believe that we could do this project.

Date: 8th May 2015 Anirudh Kaushik

Table of Contents

List of Figures

PROBLEM STATEMENT

- To design an embedded system that can be interfaced with a PC.
- To design an electrical circuit to control the switching of electrical appliances via embedded system and hence via PC.
- To design a relational database system that manages the whole data flow between the user action and the action taken by the embedded system.
- To make a web user interface for controlling the whole system by any of the internet protocol (HTTP, FTP,TELNET).

ABSTRACT

The development of digital information has led the rapid change in human lifestyle. The use of electricity is very important as one of the main source of energy that is vital in today modern life.

Some kinds of mechanism using available technology could be used to reduce wastage in electricity usage. Thus a prototype based on a microcontroller device using Internet is developed.

An apparatus and method for controlling a home appliance network is disclosed, in which a home server is established at low cost so that an external user can easily control the operation of home appliances. It can automatically control any electrical equipment at home remotely through internet. Hence the electrical energy saving in daily life can be made more efficient and effective.

The purpose of using Internet is to provide widest coverage at an effective cost. Therefore the use of Internet would facilitate in controlling the electrical device at home from long distance and low in maintenance and independent from any physical geographical boundary.

At the present time, people use electrical energy as one of the main source of power of energy to operate any electrical device or appliance. Most of the people turn on the light for 24 hours per day when they are away from home. Leaving the light turned on continuously, lead to energy waste. Thus this project is proposed to develop a system is to facilitate the home owner to optimize usage of electricity remotely using Internet.

The motivation is to facilitate the users to automate their homes having ubiquitous access. The system provides availability due to development of a low cost system. The followings are the objectives of the research project to ensure it meets the aim.

- To design an embedded system that can be interfaced with a PC.
- To design an electrical circuit to control the switching of electrical appliances via embedded system and hence via PC.
- To design a relational database system that manages the whole data flow between the user action and the action taken by the embedded system.
- To make a web user interface for controlling the whole system by any of the internet protocol (HTTP, FTP, and TELNET).

CHAPTER 1

INTRODUCTION TO HOME AUTOMATION

1.1AUTOMATION

1.1.1 What is Automation?

Automation or automatic control is the use of various control systems for operating equipment such as machinery, processes in factories, boilers and heat treating ovens, switching in telephone networks, steering and stabilization of ships, aircraft and other applications with minimal or reduced human intervention. Some processes have been completely automated.

The biggest benefit of automation is that it saves labor however it is also used to save energy and materials and to improve quality, accuracy and precision.

The term automation, inspired by the earlier word automatic (coming from automaton), was not widely used before 1947, when General Motors established the automation department. It was during this time that industry was rapidly adopting feedback controllers, which were introduced in the 1930s.

Automation has been achieved by various means including mechanical, hydraulic, pneumatic, electrical, electronic and computers, usually in combination. Complicated systems, such as modern factories, airplanes and ships typically use all these combined techniques.

1.1.2 Types of Automation

i. **Open and Closed loop**

All the elements constituting the measurement and control of a single variable are called a control loop. Control that uses a measured signal, feeds the signal back and compares it to a set point, calculates and sends a return signal to make a correction, is called closed loop control. If the controller does not incorporate feedback to make a correction then it is open loop. An operator monitoring signals from various sensors and manually making corrections either physically, such as turning the handle on a valve, or remotely, such as using a dial on a control panel, is performing open loop control. Timers and sequence controllers using logic, such as those on an elevator, are also open loop.

ii. Feedback Control

Feedback control is accomplished with a controller. To function properly, a controller must provide correction in a manner that maintains stability. The theoretical basis of feedback control is control theory, which also covers servomechanisms, which are often part of an automated system.

Maintaining stability is a principal objective of control theory. Stability means that the system should not oscillate excessively around the set point or get into a situation where it shuts down or runs away.

As an example of feedback control, consider a steam coil air heater in which a temperature sensor measures the temperature of the heated air, which is the measured variable. This signal is constantly "fed back" to the controller, which compares it to the desired setting (set point). The controller calculates the difference (error), then calculates a correction and sends the correction signal to adjust the air pressure to a diaphragm that moves a positioner on the steam valve, opening or closing it by the calculated amount.

The complexities of this are that the quantities involved are all of different physical types; the temperature sensor signal may be electrical or pressure from an enclosed fluid, the controller may employ pneumatic, hydraulic, mechanical or electronic techniques to sense the error and send a signal to adjust the air pressure that moves the valve.

The first controllers used analog methods to perform their calculations. Analog methods were also used in solving differential equations of control theory. The electronic analog computer was developed to solve control type problems and electronic analog controllers were also developed.[3] Analog computers were displaced by digital computers when they became widely available.

Common applications of feedback control are control of temperature, pressure, flow, and speed.

iii. Sequential control and logical sequence or system state control

Sequential control may be either to a fixed sequence or to a logical one that will perform different actions depending on various system states. An example of an adjustable but otherwise fixed sequence is a timer on a lawn sprinkler.

States refer to the various conditions that can occur in a use or sequence scenario of the system. An example is an elevator, which uses logic based on the system state to perform certain actions in response to its state and operator input. For example, if the operator presses the floor n button, the system will respond depending on whether the elevator is stopped or moving, going up or down, or if the door is open or closed, and other conditions.

An early development of sequential control was relay logic, by which electrical relays engage electrical contacts which either start or interrupt power to a device. Relays were first used in telegraph networks before being developed for controlling other devices, such as when starting and stopping industrial-sized electric motors or opening and closing solenoid valves. Using relays for control purposes allowed event-driven control, where actions could be triggered out of sequence, in response to external events. These were more flexible in their response than the rigid single-sequence cam timers. More complicated examples involved maintaining safe sequences for devices such as swing bridge controls, where a lock bolt needed to be disengaged before the bridge could be moved, and the lock bolt could not be released until the safety gates had already been closed.

The total number of relays, cam timers and drum sequencers can number into the hundreds or even thousands in some factories. Early programming techniques and languages were needed to make such systems manageable, one of the first being ladder logic, where diagrams of the interconnected relays resembled the rungs of a ladder. Special computers called programmable logic controllers were later designed to replace these collections of hardware with a single, more easily re-programmed unit.

In a typical hard wired motor start and stop circuit (called a control circuit) a motor is started by pushing a "Start" or "Run" button that activates a pair of electrical relays. The "lock-in" relay locks in contacts that keep the control circuit energized when the push button is released. (The start button is a normally open contact and the stop button is normally closed contact.) Another relay energizes a switch that powers the device that throws the motor starter switch (three sets of contacts for

three phase industrial power) in the main power circuit. (Note: Large motors use high voltage and experience high in-rush current, making speed important in making and breaking contact. This can be dangerous for personnel and property with manual switches.) All contacts are held engaged by their respective electromagnets until a "stop" or "off" button is pressed, which de-energizes the lock in relay. See diagram: Motor Starters Hand-Off-Auto With Start-Stop (Note: The above description is the "Auto" position case in this diagram).

Commonly interlocks are added to a control circuit. Suppose that the motor in the example is powering machinery that has a critical need for lubrication. In this case an interlock could be added to insure that the oil pump is running before the motor starts. Timers, limit switches and electric eyes are other common elements in control circuits.

Solenoid valves are widely used on compressed air or hydraulic fluid for powering actuators on mechanical components. While motors are used to supply continuous rotary motion, actuators are typically a better choice for intermittently creating a limited range of movement for a mechanical component, such as moving various mechanical arms, opening or closing valves, raising heavy press rolls, applying pressure to presses.

iv. Computer Control

Computers can perform both sequential control and feedback control, and typically a single computer will do both in an industrial application. Programmable logic controllers (PLCs) are a type of special purpose microprocessor that replaced many hardware components such as timers and drum sequencers used in relay logic type systems. General purpose process control computers have increasingly replaced standalone controllers, with a single computer able to perform the operations of hundreds of controllers. Process control computers can process data from

a network of PLCs, instruments and controllers in order to implement typical (such as PID) control of many individual variables or, in some cases, to implement complex control algorithms using multiple inputs and mathematical manipulations. They can also analyze data and create real time graphical displays for operators and run reports for operators, engineers and management.

Control of an automated teller machine (ATM) is an example of an interactive process in which a computer will perform a logic derived response to a user selection based on information retrieved from a networked database. The ATM process has a lot of similarities to other online transaction processes. The different logical responses are called scenarios. Such processes are typically designed with the aid of use cases and flowcharts, which guide the writing of the software code.

1.1.3 History

The earliest feedback control mechanism was used to tent the sails of windmills. It was patented by Edmund Lee in 1745.

The centrifugal governor, which dates to the last quarter of the 18th century, was used to adjust the gap between millstones.The centrifugal governor was also used in the automatic flour mill developed by Oliver Evans in 1785, making it the first completely automated industrial process. The governor was adopted by James Watt for use on a steam engine in 1788 after Watt's partner Boulton saw one at a flour mill Boulton & Watt were building.

The governor could not actually hold a set speed; the engine would assume a new constant speed in response to load changes. The governor was able to handle smaller variations such as those caused by fluctuating heat load to the boiler. Also, there was a tendency for oscillation whenever there was a speed change. As a consequence, engines

equipped with this governor were not suitable for operations requiring constant speed, such as cotton spinning.

Several improvements to the governor, plus improvements to valve cut-off timing on the steam engine, made the engine suitable for most industrial uses before the end of the 19th century. Advances in the steam engine stayed well ahead of science, both thermodynamics and control theory.

The governor received relatively little scientific attention until James Clerk Maxwell published a paper that established the beginning of a theoretical basis for understanding control theory. Development of the electronic amplifier during the 1920s, which was important for long distance telephony, required a higher signal to noise ratio, which was solved by negative feedback noise cancellation. This and other telephony applications contributed to control theory. Military applications during the Second World War that contributed to and benefited from control theory were fire-control systems and aircraft controls. The word "automation" itself was coined in the 1940s by General Electric. The so-called classical theoretical treatment of control theory dates to the 1940s and 1950s.

Relay logic was introduced with factory electrification, which underwent rapid adaption from 1900 through the 1920s. Central electric power stations were also undergoing rapid growth and operation of new high pressure boilers, steam turbines and electrical substations created a large demand for instruments and controls.

Central control rooms became common in the 1920s, but as late as the early 1930s, most process control was on-off. Operators typically monitored charts drawn by recorders that plotted data from instruments. To make corrections, operators manually opened or closed valves or turned switches on or off. Control rooms also used color coded lights to send signals to workers in the plant to manually make certain changes.

Controllers, which were able to make calculated changes in response to deviations from a set point rather than on-off control, began being introduced the 1930s. Controllers allowed manufacturing to continue showing productivity gains to offset the declining influence of factory electrification.

In 1959 Texaco's Port Arthur refinery became the first chemical plant to use digital control. Conversion of factories to digital control began to spread rapidly in the 1970s as the price of computer hardware fell.

1.1.4 Advantages & Disadvantages

The main advantages of automation are:

- Increased throughput or productivity.
- Improved quality or increased predictability of quality.
- Improved robustness (consistency), of processes or product.
- Increased consistency of output.
- Reduced direct human labor costs and expenses.

The following methods are often employed to improve productivity, quality, or robustness

- Install automation in operations to reduce cycle time.
- Install automation where a high degree of accuracy is required.
- Replacing human operators in tasks that involve hard physical or monotonous work.
- Replacing humans in tasks done in dangerous environments (i.e. fire, space, volcanoes, nuclear facilities, underwater, etc.)
- Performing tasks that are beyond human capabilities of size, weight, speed, endurance, etc.
- Economic improvement: Automation may improve in economy of enterprises, society or most of humanity. For example, when an enterprise invests in automation, technology recovers its investment; or when a state or country increases its income due to automation like Germany or Japan in the 20th Century.
- Reduces operation time and work handling time significantly.
- Frees up workers to take on other roles.

- Provides higher level jobs in the development, deployment, maintenance and running of the automated processes.

The main disadvantages of automation are:

- Security Threats/Vulnerability: An automated system may have a limited level of intelligence, and is therefore more susceptible to committing errors outside of its immediate scope of knowledge (e.g., it is typically unable to apply the rules of simple logic to general propositions).
- Unpredictable/excessive development costs: The research and development cost of automating a process may exceed the cost saved by the automation itself.
- High initial cost: The automation of a new product or plant typically requires a very large initial investment in comparison with the unit cost of the product, although the cost of automation may be spread among many products and over time.

1.2 INTERNET

1.2.1 Terminology

The Internet is a global system of interconnected computer networks that use the standard Internet protocol suite (TCP/IP) to link several billion devices worldwide. It is an international network of networks that consists of millions of private, public, academic, business, and government packet switched networks, linked by a broad array of electronic, wireless, and optical networking technologies. The Internet carries an extensive range of information resources and services, such as the inter-linked hypertext

documents and applications of the World Wide Web (WWW), the infrastructure to support email, and peer-to-peer networks for file sharing and telephony.

The origins of the Internet date back to research commissioned by the United States government in the 1960s to build robust, fault-tolerant communication via computer networks. While this work, together with work in the United Kingdom and France, led to important precursor networks, they were not the Internet. There is no consensus on the exact date when the modern Internet came into being, but sometime in the early to mid-1980s is considered reasonable. From that point, the network experienced decades of sustained exponential growth as generations of institutional, personal, and mobile computers were connected to it.

The funding of a new U.S. backbone by the National Science Foundation in the 1980s, as well as private funding for other commercial backbones, led to worldwide participation in the development of new networking technologies, and the merger of many networks. Though the Internet has been widely used by academia since the 1980s, the commercialization of what was by the 1990s an international network resulted in its popularization and incorporation into virtually every aspect of modern human life. As of June 2012, more than 2.4 billion people—over a third of the world's human population—have used the services of the Internet; approximately 100 times more people than were using it in 1995.Internet use grew rapidly in the West from the mid-1990s to early 2000s and from the late 1990s to present in the developing world. In 1994 only 3% of American classrooms had access to the Internet while by 2002 92% did.

Most traditional communications media including telephone, music, film, and television are being reshaped or redefined by the Internet, giving birth

to new services such as voice over Internet Protocol (VoIP) and Internet Protocol television (IPTV). Newspaper, book, and other print publishing are adapting to website technology, or are reshaped into blogging and web feeds. The Internet has enabled and accelerated new forms of human interactions through instant messaging, Internet forums, and social networking. Online shopping has boomed both for major retail outlets and small artisans and traders. Business-to-business and financial services on the Internet affect supply chains across entire industries.

The Internet has no centralized governance in either technological implementation or policies for access and usage; each constituent network sets its own policies. Only the overreaching definitions of the two principal name spaces in the Internet, the Internet Protocol address space and the Domain Name System, are directed by a maintainer organization, the Internet Corporation for Assigned Names and Numbers (ICANN). The technical underpinning and standardization of the core protocols (IPv4 and IPv6) is an activity of the Internet Engineering Task Force (IETF), a non-profit organization of loosely affiliated international participants that anyone may associate with by contributing technical expertise.

CHAPTER 2

REVIEW OF LITERATURE

2.1 MICROCONTROLLER

FIG:1 ATMEGA 16 MICROCONTROLLER (*www.mikrocontroller.net*)

A microcontroller (sometimes abbreviated μC, uC or MCU) is a small computer on a single integrated circuit containing a processor core, memory, and programmable input/ output peripherals. Program memory in the form of NOR flash or OTP.

ROM is also often included on chip, as well as a typically small amount of RAM.

Microcontrollers are designed for embedded applications, in contrast to the microprocessors used in personal computers or other general purpose applications.

Microcontrollers are used in automatically controlled products and devices, such as automobile engine control systems, implantable medical devices, remote controls, office machines, appliances, power tools, toys and other embedded systems. By reducing the size and cost compared to a design that uses a separate microprocessor, memory, and input/output devices, microcontrollers make it economical to digitally control even more devices and processes. Mixed signal microcontrollers are common, integrating analog components needed to control non digital electronic systems.

2.1.1 Family of Microcontrollers

As of 2011 there are several dozen microcontroller architectures and vendors including:

- ARM core processors (from many vendors)
- Atmel AVR (8-bit), AVR32 (32-bit), and AT91SAM (32-bit)
- Cypress Semiconductor PSoC (Programmable System-on-Chip)
- Freescale Cold Fire (32-bit) and S08 (8-bit)
- Freescale 68HC11 (8-bit)
- Intel 8051
- Infineon: 8, 16, 32 Bit microcontrollers
- MIPS

- Microchip Technology PIC, (8-bit PIC16, PIC18, 16-bit dsPIC33 / PIC24), (32-bit PIC32)
- NXP Semiconductors LPC1000, LPC2000, LPC3000, LPC4000 (32-bit), LPC900, LPC700(8-bit).
- Parallax Propeller
- PowerPC ISE
- Rabbit 2000 (8-bit)
- Renesas RX, V850, Hitachi H8, Hitachi SuperH (32-bit), M16C (16-bit), RL78, R8C,78K0/78K0R (8-bit).
- Silicon Laboratories Pipelined 8051 Microcontrollers
- STMicroelectronics ST8 (8-bit), ST10 (16-bit) and STM32 (32-bit)
- Texas Instruments TI MSP430 (16-bit)
- Toshiba TLCS-870 (8-bit/16-bit).

Application of Microcontrollers

- Automotive
- Building Automation
- Home Appliances
- Home Entertainment
- Industrial Automation
- Lighting
- Smart Energy
- Mobile Electronics
- PC Peripherals
- Internet-of-Things

2.1.2 AVR Family

The AVR is a modified Harvard architecture 8-bit RISC single chip microcontroller which was developed by Atmel in 1996. The AVR was one of the first microcontroller families to use on chip flash memory for program storage, as opposed to one-time programmable ROM, EPROM, or EEPROM used by other microcontrollers at the time.

Atmega16

ATmega16 is an 8-bit high performance microcontroller of Atmel's Mega AVR family with low power consumption. Atmega16 is based on enhanced RISC (Reduced Instruction Set Computing) architecture with 131 powerful instructions. Most of the instructions execute in one machine cycle. Atmega16 can work on a maximum frequency of 16MHz.

ATmega16 has 16 KB programmable flash memory, static RAM of 1 KB and EEPROM of 512 Bytes. The endurance cycle of flash memory and EEPROM is 10,000 and 100,000, respectively.

ATmega16 is a 40 pin microcontroller. There are 32 I/O (input/output) lines which are divided into four 8-bit ports designated as PORTA, PORTB, PORTC and PORTD.

ATmega16 has various in-built peripherals like USART, ADC, Analog Comparator, SPI, JTAG etc. Each I/O pin has an alternative task related to in-built peripherals. The following table shows the pin description of ATmega16.

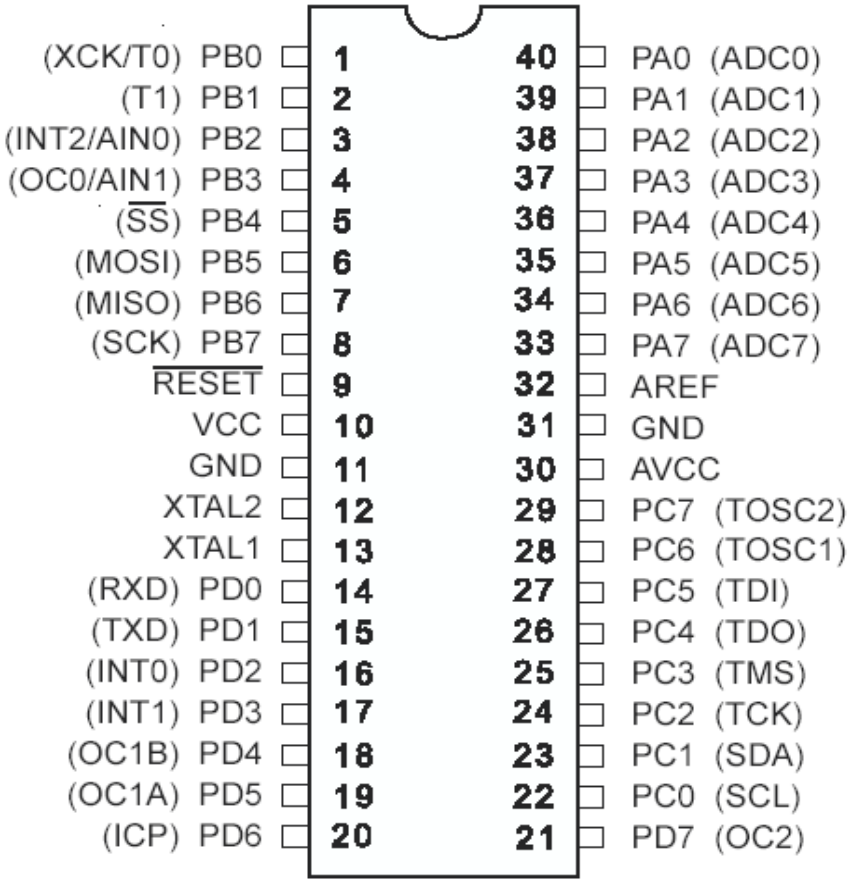

Fig: 2 ATMEGA 16 Pin Diagram (*www.mikrocontroller.net***)**

Pin No.	Pin Name	Description	Alternate Function
1	(XCK/T0) PB0	I/O PORTB,Pin 0	**T0:Timer External Counter Input XCK:USART External Clock I/O**
2	**(T1)PB1**	**I/O PORTB,Pin 1**	**T1:Timer1 External Counter Input**
3	(INT2/AIN0) PB2	I/O PORTB, Pin 2	AIN0: Analog Comparator Positive I/P INT2: External Interrupt 2 Input
4	(OC0/AIN1) PB3	I/O PORTB, Pin 3	AIN1: Analog Comparator Negative I/P OC0 : Timer0 Output Compare Match Output

5	(SS) PB4	I/O PORTB, Pin 4	In System Programmer (ISP) Serial Peripheral Interface (SPI)
6	(MOSI) PB5	I/O PORTB, Pin 5	In System Programmer (ISP) Serial Peripheral Interface (SPI)
7	(MISO) PB6	I/O PORTB, Pin 6	In System Programmer (ISP) Serial Peripheral Interface (SPI)
8	(SCK) PB7	I/O PORTB, Pin 7	In System Programmer (ISP) Serial Peripheral Interface (SPI)
9	RESET	Reset Pin, Active Low Reset	
10	Vcc	Vcc = +5V	
11	GND	GROUND	
12	XTAL2	Output to Inverting Oscillator Amplifier	
13	XTAL1	Input to Inverting Oscillator Amplifier	
14	(RXD) PD0	I/O PORTD,Pin 0	USART Serial Communication Interface
15	(TXD) PD1	I/O PORTD, Pin 1	USART Serial Communication Interface
16	(INT0) PD2	I/O PORTD, Pin 2	External Interrupt INT0
17	(INT1) PD3	I/O PORTD, Pin 3	External Interrupt INT1
18	(OC1B) PD4	I/O PORTD, Pin 4	PWM Channel Outputs
19	(OC1A) PD5	I/O PORTD, Pin 5	PWM Channel Outputs
20	(ICP) PD6	I/O PORTD, Pin 6	Timer/Counter1 Input Capture Pin
21	PD7 (OC2)	I/O PORTD, Pin 7	Timer/Counter2 Output Compare Match Output
22	PC0 (SCL)	I/O PORTC, Pin 0	TWI Interface
23	PC1 (SDA)	I/O PORTC, Pin 1	TWI Interface
24	PC2 (TCK)	I/O PORTC, Pin 2	JTAG Interface
25	PC3 (TMS)	I/O PORTC, Pin 3	JTAG Interface
26	PC4 (TDO)	I/O PORTC, Pin 4	JTAG Interface
27	PC5 (TDI)	I/O PORTC, Pin 5	JTAG Interface
28	PC6 (TOSC1)	I/O PORTC, Pin 6	Timer Oscillator Pin 1

29	PC7 (TOSC2)	I/O PORTC, Pin 7	Timer Oscillator Pin 2
30	AVcc	Voltage Supply=Vcc for ADC	
31	GND	GROUND	
32	AREF	Analog Reference Pin for ADC	
33	PA7 (ADC7)	I/O PORTA, Pin 7	ADC Channel 7
34	PA6 (ADC6)	I/O PORTA, Pin 6	ADC Channel 6
35	PA5 (ADC5)	I/O PORTA, Pin 5	ADC Channel 5
36	PA4 (ADC4)	I/O PORTA, Pin 4	ADC Channel 4
37	PA3 (ADC3)	I/O PORTA, Pin 3	ADC Channel 3
38	PA2 (ADC2)	I/O PORTA, Pin 2	ADC Channel 2
39	PA1 (ADC1)	I/O PORTA, Pin 1	ADC Channel 1
40	PA0 (ADC0)	I/O PORTA, Pin 0	ADC Channel 0

Table 1 Pin Description of Atmega16

2.2 Development Board

The NEXTSAPIENS Development Board Features are:

- 40 Pin Atmel ATmega16microcontroller with internal system clock upto 8 MHz and externally upto 16 MHz
- 16 KB Flash RAM memory for programs
- 1 KB of SRAM
- 512 of EEPROM
- One 6x1 Pin SPI Relimate Header
- Eight 3x1 Pin Relimate header inputs for 8 analog sensors
- One 16 Pin header to connect 16*2 alphanumeric LCD
- Two onboard L293D drivers for motors (upto 600 mA per channel)
- Dual 7805 Voltage regulator
- Dual power input options (Through molex connector or through DC Jack)
- Two programmable Micro-Switches

- Two programmable LEDs
- Two DPDT switches (one for power on/off and one for reset)
- MAX 232 Level shifter for RS232 communication
- One 3x1 Pin relimate header for RS2332 communication
- Four 8 Pin bergistick headers (male) from each port of ATmega16/32
- Wide input power range from 7 volts to 24 volts at 1.5-2 Amps
- Board size of 6 x 3 inches, designed for educational and hobby purpose, on high quality PCB

2.2.1 LCD

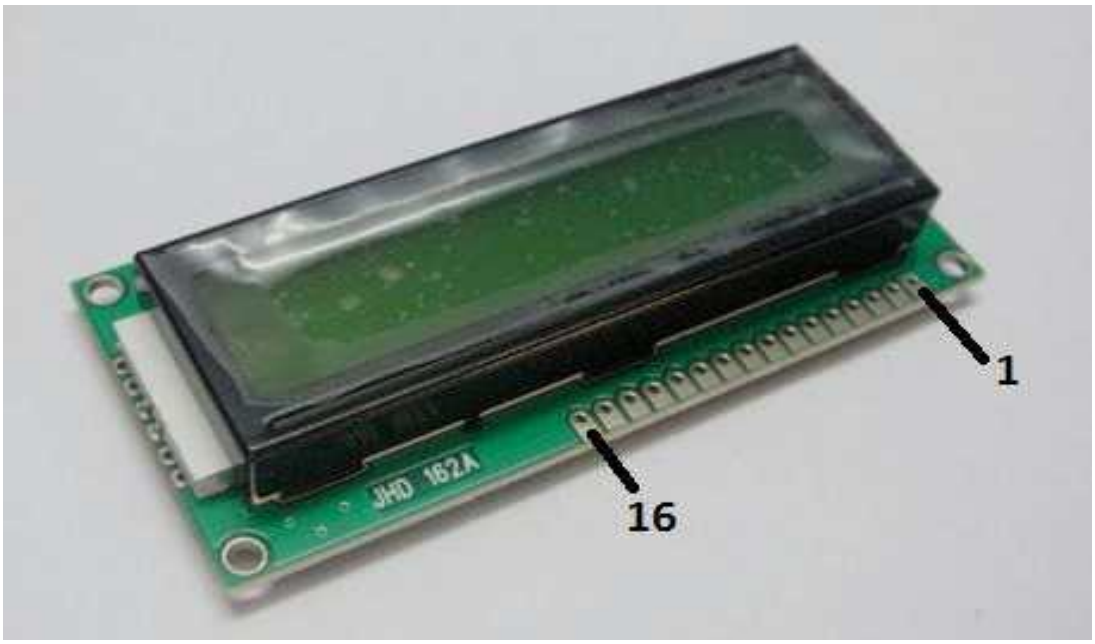

Figure 3: 16*2 LCD Panel (*www.circuitstoday.com*)

LCD is Liquid Crystal Display that uses the light modulating properties of liquid crystals. LCD displays utilize two sheets of polarizing material with a liquid crystal solution between them. An electric current passed through the liquid causes the crystals to align so that light cannot pass through them. Each crystal, therefore, is like a shutter, either allowing light to pass through or blocking the light.

An LCD monitor consists of five layers: a backlight, a sheet of polarized glass, a "mask" of pixels, a layer of liquid crystal solution responsive to a wired grid of x, y coordinates, and a second polarized sheet of glass. By manipulating the orientations of crystals through precise electrical charges of varying degrees and voltages, the crystals act like tiny shutters, opening or closing in response to the stimulus, thereby allowing degrees of light that have passed through specific colored pixels to illuminate the screen, creating a picture.

Each pixel of an LCD typically consists of a layer of molecules aligned between two transparent electrodes, and two polarizing filters, the axes of transmission of which are (in most of the cases) perpendicular to each other. With no actual liquid crystal between the polarizing filters, light passing through the first filter would be blocked by the second (crossed) polarizer. The surfaces of the electrodes that are in contact with the liquid crystal material are treated so as to align the liquid crystal molecules in a particular direction. This treatment typically consists of a thin polymer layer that is uni-directionally rubbed using, for example, a cloth. The direction of the liquid crystal alignment is then defined by the direction of rubbing. Electrodes are made of a transparent conductor called Indium (ITO). The Liquid Crystal Display is intrinsically a “passive” device; it is a simple light valve. The managing and control of the data to be displayed is performed by one or more circuits commonly denoted as LCD drivers.

Before applying an electric field, the orientation of the liquid crystal molecules is determined by the alignment at the surfaces of electrodes. In a twisted nematic device (still the most common liquid crystal device), the surface alignment directions at the two electrodes are perpendicular to each other, and so the molecules arrange themselves in a helical structure, or twist.

This reduces the rotation of the polarization of the incident light, and the device appears grey. If the applied voltage is large enough, the liquid crystal molecules in

the center of the layer are almost completely untwisted and the polarization of the incident light is not rotated as it passes through the liquid crystal layer. This light will then be mainly polarized perpendicular to the second filter, and thus be blocked and the pixel will appear black. By controlling the voltage applied across the liquid crystal layer in each pixel, light can be allowed to pass through in varying amounts thus constituting different levels of gray.

The optical effect of a twisted nematic device in the voltage-on state is far less dependent on variations in the device thickness than that in the voltage-off state. Because of this, these devices are usually operated between crossed polarizers such that they appear bright with no voltage (the eye is much more sensitive to variations in the dark state than the bright state). These devices can also be operated between parallel polarizers, in which case the bright and dark states are reversed. The voltage-off dark state in this configuration appears blotchy, however, because of small variations of thickness across the device.

Both the liquid crystal material and the alignment layer material contain ionic compounds. If an electric field of one particular polarity is applied for a long period of time, this ionic material is attracted to the surfaces and degrades the device performance. This is avoided either by applying an alternating current or by reversing the polarity of the electric field as the device is addressed (the response of the liquid crystal layer is identical, regardless of the polarity of the applied).

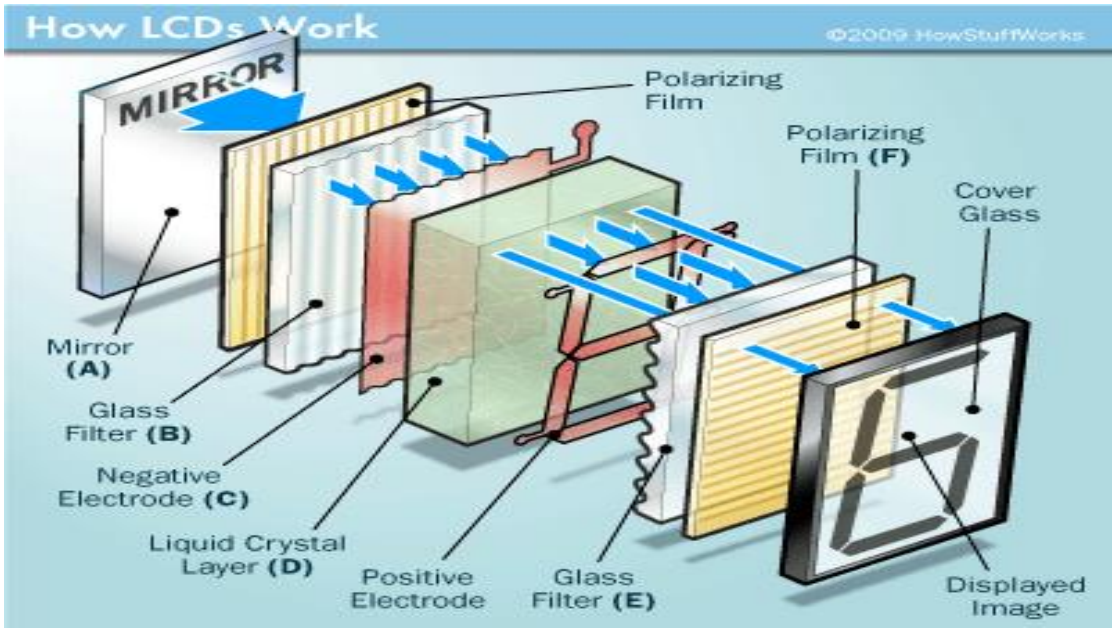

Figure 4: Working of LCD Panel (*www.circuitstoday.com*)

Types of LCD'S

There are mainly two types of LCD

- Passive Display
- Active Display

Passive Display- Passive displays are widely used with segmented digits and characters for small readouts in devices such as calculators, fax machines and remote controls, most of which are monochrome or have only a few colors.

Active Display- Used in all LCD TVs and desktop computer monitors and 99.9% of all laptops, active displays are essentially "active matrix" displays and almost always color. The reason for the 99.9% is that OLED is emerging.

Characteristics of LCD:

Here by 16*2 we mean that there are 16 characters can be displayed in one line and 2 means there are 2 lines in our display.

Features of 16*2 character LCD-

- 5 x 8 dots with cursor
- Built-in controller (KS 0066 or Equivalent)
- + 5V power supply (Also available for + 3V)
- 1/16 duty cycle
- B/L to be driven by pin 1, pin 2 or pin 15, pin 16 or A.K (LED)
- N.V. optional for + 3V power supply

Connections of LCD:

Pin 1 of LCD to ground.

Pin 2 of LCD to Vcc (5V).

Pin 3 of LCD to ground.

Pin 4 of LCD to PortB.2 (2nd pin of port B).

Pin 5 of LCD to ground.

Pin 6 of LCD to PortB.3.

Pin 7-10 are left floating (open).

Pin 11 of LCD to PortB.4.

Pin 12 of LCD to PortB.5.

Pin 13 of LCD to PortB.6.

Pin 14 of LCD to PortB.7.

Pin 15 of LCD to Vcc (5V).

Pin 16 of LCD to ground.

Syntax for Configuring LCD:

CONFIG LCD = LCD_type

CONFIG LCDPIN = PIN, DB4= PN,DB5=PN, DB6=PN, DB7=PN, E=PN, RS=PN

LCD_type – It is the type of LCD you want to configure. It can be: 40 * 4, 16 * 1, 16 * 2, 16* 4, 16 * 4, 20 * 2 or 20 * 4 or 16 * 1a or 20*4A.

Config Lcdpin - Override the LCD-PIN select options.

Now to configure our 16*2 alphanumeric LCD we use the following command:-

Config LCD = 16*2

Config Lcdpin = Pin, Db4 = PortB.4, Db5 = PortB.5, Db6 = PortB.6, Db7 = PortB.7, E = PortB.3 , Rs = PortB.2

2.3 Programming In BASCOM

2.3.1 MAIN SYNTAX

$regfile = "m16def.dat"

(It refers to the name of register file. The register files are stored in the BASCOM-AVR application directory with .DAT extension).The register file holds information about the chip such as the internal registers and interrupts addresses. Since we are using Atmega16

Microcontroller, we will define $regfile= “m16def.dat”)

$crystal = 1000000

(1000000 is 1 MHz frequency that a user can set freely for the microcontroller. It defines the clock speed at which you want to run your microcontroller.)

Config LCD = 16 * 2

(It is the type of LCD you want to configure. It can be: 40 * 4, 16 * 1, 16 * 2, 16 * 4, 16 * 4, 20 *2 or 20 * 4 or 16 * 1a or 20*4A.

16*2 is the LCD type which means this LCD prints 16 characters per two lines.)

Config Lcdpin = Pin, Db4 = PortB.4, Db5 = PortB.5, Db6 = PortB.6, Db7 = PortB.7, E = PortB.3, R = PortB.2

(This line gives a description about the LCD pin connections with the Microcontroller Ports.)

Config ADC = Single, Prescaler = Auto, Reference = AVcc
(ADC – It defines the Running mode. Its value is SINGLE.
PRESCALER - A numeric constant for the clock divider. Use AUTO to let the compiler generate the best value depending on the XTAL.

REFERENCE - Some chips like the M163 have additional reference options. Its value may be OFF, AVCC or INTERNAL.

Single means instructing the ADC to fetch the value only when its asked to.
"Auto" means that the ADC can automatically set its frequency in regard with the microcontroller frequency in the program.
Reference is set as AVcc because Aref voltage is referred from the voltage supply to the ADC's i.e. AVcc.)

Start ADC

(StartAdc id=s the command given to initialize the ADCs.)
Config Timer1 = Pwm, Pwm = 8, Prescale = 1, Compare A Pwm = Clear Down, Compare B
Pwm = Clear Down

(We use PWM to control the motor speed which is of 8 bit that is why we set PWM= 8, PWM works on the same frequency as the Microcontroller, and the Channels A & B are set as clear down to vary the speeds from 0 to 255 in increasing order.)

StartTimer1
(StartTimer1 is used to start the PWM channels.)

DEFINING VARIABLES

SYNTAX:

DIM (var) as type Var- Name of Variable

Type - Bit, Byte, Word, Integer, Long, Single, Double or String

Example

Dim A as Integer

Dim B as String * 8

First statement is defining a variable as integer and second one is defining B variable as string of 8 characters long. Other than Integer and String there are many data types available in BASCOM.

START & CLEAR COMMANDS

Start Command: This command is use to start the specified device.

Syntax

START device

Device - TIMER0, TIMER1, COUNTER0 or COUNTER1, WATCHDOG, AC (Analog comparator power) or ADC (A/D converter power)

Example – Start ADC

CLS Command: Clear the LCD display and set the cursor to home.

Syntax/ Example – Cls

LOOPS

If-Else statement, Loops and Select-case statement

BASCOM allows using all types of loops in the program like do, while and for. Concept of using these loops is same as using them in other languages like C. Given below are syntaxes of all loops you can use in BASCOM –

1. Do Loop

```
Do
    <Statement>

Loop
```

2. If-else statement

```
If (condition) then
    <Statements>
Else
    <Statements>
End if
```

GETADC COMMAND

This command is used to take input from the analog sensor connected to the development board.

This command retrieves the analog value from channel 0-7 of port A. The range of analog value is from 0 to 1023.

var = GETADC (channel [, offset])

Var- The variable in which the value will be stored.

Channel – It is the pin no of port A to which analog sensor is connected.

Offset – It is an optional numeric variable that specifies gain or mode.

Example

L = Getadc (2)

Here, in above example, the analog value of the input provided by the sensor connected to pin 2 of port A is stored in variable L.

LCD COMMANDS

It is used to display a constant or variable on LCD screen.

SYNTAX:

LCD x

X - Variable or constant to be displayed on LCD

For displaying string / text, Use LCD “text”

For displaying variable, Use LCD A (A refers to the variable)

For displaying text/variable in next line, we use command LOWERLINE

Example

LCD A; “hello”

Lowerline

LCD “Next sapiens”

Output on LCD will be: “Value of Variable A”, Hello Next sapiens

WAITMS & PWMXX COMMAND

Waitms command: Suspends program execution for a given time in ms.

SYNTAX:

WAITMS mS

mS- The number of milliseconds to wait. (1-65535)

Example: Waitms 200

PWMXX command: It is used to set the speed of motor

SYNTAX:

PwmXX = value

XX- it is the channel of a motor

Value – any integer value ranging from 0 to maximum speed.

Example

Pwm1a = 180

PORTX.Y COMMAND

PORTX.y command: it is used to set the direction of the motor

SYNTAX:

PORTX.y = value

X.y - 'X' as port number and 'y' as pin number

Value - 0 for clock rotation and 1 for ant clock rotation

Example:

PortD.3=1

2.4 MAX 232

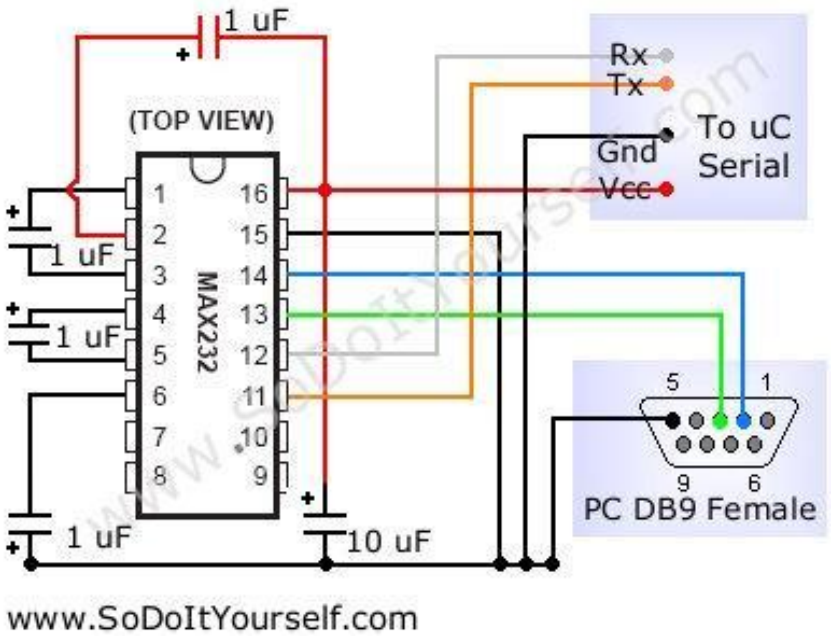

Figure 5: Circuit Diagrams showing MAX 232 connections

(*www.SoDoItYourself.com***)**

The MAX232 is an integrated circuit that converts signals from an RS-232 serial port to signals suitable for use in TTL compatible digital logic circuits. The MAX232 is a dual driver/receiver and typically converts the RX, TX, CTS and RTS signals.

The drivers provide RS-232 voltage level outputs (approx. ± 7.5 V) from a single + 5 V supply via on-chip charge pumps and external capacitors. This makes it useful for implementing RS-232 in devices that otherwise do not need any voltages outside the 0 V to + 5 V range, as power supply design does not need to be made more complicated just for driving the RS-232 in this case.

The receivers reduce RS-232 inputs (which may be as high as ± 25 V), to standard 5 V TTL levels. These receivers have a typical threshold of 1.3 V, and a typical hysteresis of 0.5 V.

The later MAX232A is backwards compatible with the original MAX232 but may operate at higher baud rates and can use smaller external capacitors – 0.1 μF in place of the 1.0 μF capacitors used with the original device.

The newer MAX3232 is also backwards compatible, but operates at a broader voltage range, from 3 to 5.5 V. Serial RS-232 (V.24) communication works with voltages (between -15V ... -3V are used to transmit a binary '1' and +3V ... +15V to transmit a binary '0') which are not compatible with today's computer logic voltages. On the other hand, classic TTL computer logic operates between 0V ... +5V (roughly 0V ... +0.8V referred to as low for binary '0', +2V ... +5V for high binary '1'). Modern low-power logic operates in the range of 0V ... +3.3V or even lower.

So, the maximum RS-232 signal levels are far too high for today's computer logic electronics, and the negative RS-232 voltage can't be grokked at all by the computer logic. Therefore, to receive serial data from an RS-232 interface the voltage has to be reduced, and the 0 and 1 voltage levels inverted. In the other direction (sending data from some logic over RS-232) the low logic voltage has to be "bumped up", and a negative voltage has to be generated too.

The MAX232 from Maxim was the first IC which in one package contains the necessary drivers (two) and receivers (also two), to adapt the RS-232 signal voltage levels to TTL logic. It became popular, because it just needs one voltage (+5V) and

generates the necessary RS-232 voltage levels (approx. -10V and +10V) internally. This greatly simplified the design of circuitry.

Circuitry designers no longer need to design and build a power supply with three voltages (e.g. -12V, +5V, and +12V), but could just provide one +5V power supply, e.g. with the help of a simple 78x05 voltage regulator.

The MAX232 has a successor, the MAX232A. The ICs are almost identical, however, the MAX232A is much more often used (and easier to get) than the original MAX232, and the MAX232A only needs external capacitors 1/10th the capacity of what the original MAX232 needs.

It should be noted that the MAX232 (A) is just a driver/receiver. It does not generate the necessary RS-232 sequence of marks and spaces with the right timing, it does not decode the RS-232 signal, it does not provide a serial/parallel conversion. All it does is to convert signal voltage levels. Generating serial data with the right timing and decoding serial data has to be done by additional circuitry, e.g. by a 16550 UART or one of these small micro controllers (e.g. Atmel AVR, Microchip PIC) getting more and more popular.

2.5 JSP Module

2.5.1 JAVA SERVER PAGES

Java Server Pages (JSP) is a technology that helps software developers create dynamically generated web pages based on HTML, XML, or other document types. Released in 1999 by Sun Microsystems, JSP is similar to PHP, but it uses the Java programming language.

To deploy and run Java Server Pages, a compatible web server with a servlet container, such as Apache Tomcat or Jetty, is required.

Architecturally, JSP may be viewed as a high-level abstraction of Java servlets. JSPs are translated into servlets at runtime; each JSP servlet is cached and re-used until the original JSP is modified.

JSP can be used independently or as the view component of a server-side model–view–controller design, normally with JavaBeans as the model and Java servlets (or a framework such as Apache Struts) as the controller. This is a type of Model 2 architecture.

JSP allows Java code and certain pre-defined actions to be interleaved with static web markup content, with the resulting page being compiled and executed on the server to deliver a document. The compiled pages, as well as any dependent Java libraries, use Java byte code rather than a native software format. Like any other Java program, they must be executed within a Java virtual machine (JVM) that integrates with the server's host operating system to provide an abstract platform-neutral environment.

JSPs are usually used to deliver HTML and XML documents, but through the use of Output Stream, they can deliver other types of data as well.

The Web container creates JSP implicit objects like pageContext, servletContext, session, request & response.

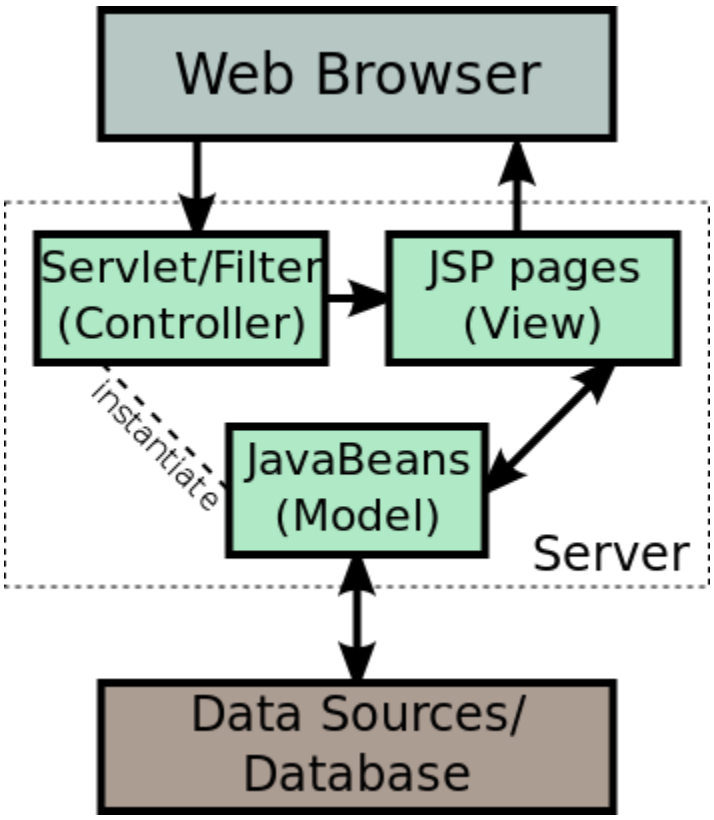

Figure 6: JSP Model 2 Architecture (*www.java-samples.com***)**

2.5.2 SYNTAX

JSP pages use several delimiters for scripting functions. The most basic is <% ... %>, which encloses a JSP scriptlet. A scriptlet is a fragment of Java code that is run when the user requests the page. Other common delimiters include <%= ... %> for expressions, where the scriptlet and delimiters are replaced with the result of evaluating the expression, and directives, denoted with <%@ ... %>.

Java code is not required to be complete or self-contained within its scriptlet element block, but can straddle markup content providing the page as a whole is syntactically correct. For example, any Java if/for/while blocks opened in one scriptlet element must be correctly closed in a later element for the page to successfully compile. Markup which falls inside a split block of code is subject to that code, so markup inside an if block will only appear in the output when the if condition evaluates to

true; likewise, markup inside a loop construct may appear multiple times in the output depending upon how many times the loop body runs.

2.5.3 COMPILER

A Java Server Pages compiler is a program that parses JSPs, and transforms them into executable Java Servlets. A program of this type is usually embedded into the application server and run automatically the first time a JSP is accessed, but pages may also be precompiled for better performance, or compiled as a part of the build process to test for errors.

Some JSP containers support configuring how often the container checks JSP files timestamps to see whether the page has changed. Typically, this timestamp would be set to a short interval (perhaps seconds) during software development, and a longer interval (perhaps minutes, or even never) for a deployed Web application

Servlet containers

- Apache Tomcat
- Jetty (web server)
- Oracle iPlanet Web Server
- GlassFish
- Websphere

2.5.4 SAMPLE CODE FOR WEBSITE

Admin Page:

```
<%@ page import="java.sql.*;"%>
 <%
 String s=request.getParameter("status");
 String user=(String)session.getAttribute("user");
Connection con;
Statement sta;
```

```
ResultSet rs;
PreparedStatement pstmt;
con=null;
sta=null;
rs=null;
pstmt=null;

//out.println("1");
   try
    {
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver"); //Load Drivers
    con = DriverManager.getConnection("jdbc:odbc:home;user=sa;password=.");
    pstmt = con.prepareStatement("UPDATE devices SET status = '"+s+"' where
    dev='fan'       and devid='1' and username='"+user+"'");
    //out.println("2");
     int r=pstmt.executeUpdate();
     //out.println("3");
              pstmt.close();
              //out.println("4");
              if (r>0)
              {
                       response.sendRedirect("userpage.jsp");
              }
              else
              {
                       response.sendRedirect("index.jsp");
              }
              con.close();
    }
              catch (Exception e)
```

```
            {
                System.err.println("Exception: "+e.getMessage());
            }
%>
</body>
</html>
```

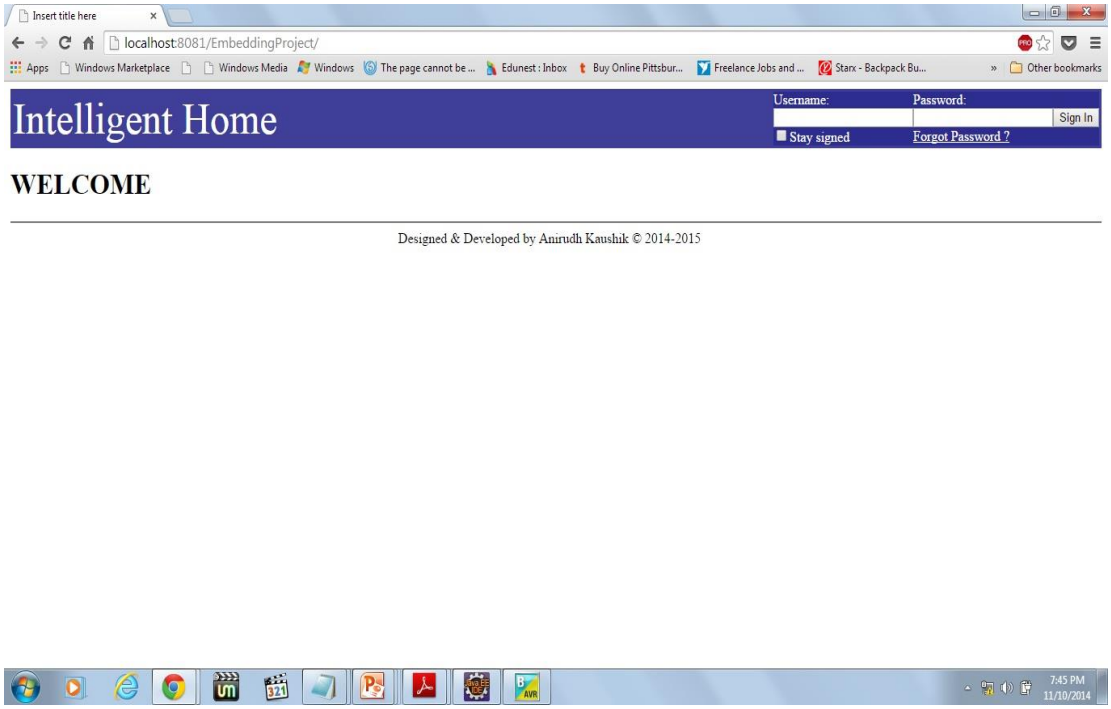

Figure 7: Homepage of the website (*screenshot*)

2.6 SOFTWARE MODEL

What is a software life cycle model?

A software life cycle model is either a descriptive or prescriptive characterization of how software is or should be developed.

What are the benefits of software life cycle model?

- It provides guideline to organize, plan, staff, budget, schedule and manage software project work over organizational time, space, and computing environments.
- It provides prescriptive outline for what documents to produce for delivery to client.
- It provides basis for determining what software engineering tools and methodologies will be most appropriate to support different life cycle activities.
- It provides the framework for analysing or estimating patterns of resource allocation and consumption during the software life cycle
- It provides basis for conducting empirical studies to determine what affects software productivity, cost, and overall quality.

The waterfall model often considered as classic approach to the system development life cycle, the waterfall model describes a development method that is linear and sequential. Waterfall development has distinct goals for each phase

of development. Imagine a waterfall on the cliff of a steep mountain. Once the water has flowed over the edge of the cliff and has begun its journey down the side of the mountain, it cannot turn back. It is the same with waterfall development. Once a phase of development is completed, the development proceeds to the next phase and there is no turning back. Waterfall Model is one of the most widely used Software Development Process. It is also called as "Linear Sequential model" or the "classic life cycle" or "iterative model". It is widely used in the commercial development projects. It is called so because here, we move to next phase (step) after getting input from previous phase, like in a waterfall, water flows down to from the upper steps.

PHASES

Analysis

- Feasibility study
- Requirements gathering
- Requirements specification preparation

Design

- Architecture
- Database design
- Prototypes and wireframe development
- Test plan preparation

Development

- Coding
- Unit testing

Testing & Quality Assurance

- Test cases preparation
- Testing and issue tracking
- Bug resolution

Launch and Maintenance

- Application delivery
- User acceptance
- Maintenance
- Support

Requirement Analysis and Software Definition

This is the first phase of waterfall model which includes a meeting with the customer to understand his requirements. This is the most crucial phase as any misinterpretation at this stage may give rise to validation issues later. The software definition must be detailed and accurate with no ambiguities. It is very

important to understand the customer requirements and expectations so that the end product meets his specifications.

In this project, requirements of a customer could be:

(I) Number of appliances the customer needs to attach
(II) What kind of appliances customer needs to attach.
(III) Website account needs to be setup.
(IV) Voltages at which the customer will operate the appliances attached to the board.

System Design

The customer requirements are broken down into logical modules for the ease of implementation. Hardware and software requirements for every module are identified and designed accordingly. Also the inter relation between the various logical modules is established at this stage. Algorithms and diagrams defining the scope and objective of each logical model are developed. In short, this phase lays a fundamental for actual programming and implementation.

In this project software requirements are identified as:

I. A platform to host the website developed using J2EE technology.
II. A server to connect the AVR-BOARD with the website hoisted.
III. A platform to embed the required code into the microcontroller.
IV. An interface to connect the hardware with the software.

Hardware requirements were identified as:

I. An AVR-BOARD
II. ATMEGA 16 Microcontroller
III. Connecting wires
IV. UART cable

V. A burner to burn the code. Etc

System Implementation

This is the software process in which actual coding takes place. A software program is written based upon the algorithm designed in the system design phase. A piece of code is written for every module and checked for the output.

In this project the software programs were divided into modules.

I. Website – JSP modules

   Number of JSP modules were combined to form the website like:

   a) User login Java servlet Page
   b) Home Java servlet Page
   c) Logout Java servlet Page
   d) Database entries

II. Embedding Code

III. Interface program to connect hardware with software

System Testing

The programmatically implemented software module is tested for the correct output. Bugs, errors are removed at this stage. In the process of software testing, a series of tests and test cases are performed to check the module for bugs, faults and other errors. Erroneous codes are rewritten and tested again until desired output is achieved.

Alpha testing was performed for this project.

Alpha testing is one of the most common software testing strategy used in software development. It is specially used by product development organizations. This test takes place at the developer's site. Developers observe the users and note problems .Alpha testing is testing of an application when development is about to complete. Minor design changes can still be made as a result of alpha testing.

Alpha testing is typically performed by a group that is independent of the design team, but still within the company.

System Deployment and Maintenance

This is the final phase of the waterfall model, in which the completed software product is handed over to the client after alpha, beta testing. After the software has been deployed on the client site, it is the duty of the software development team to undertake routine maintenance activities by visiting the client site.

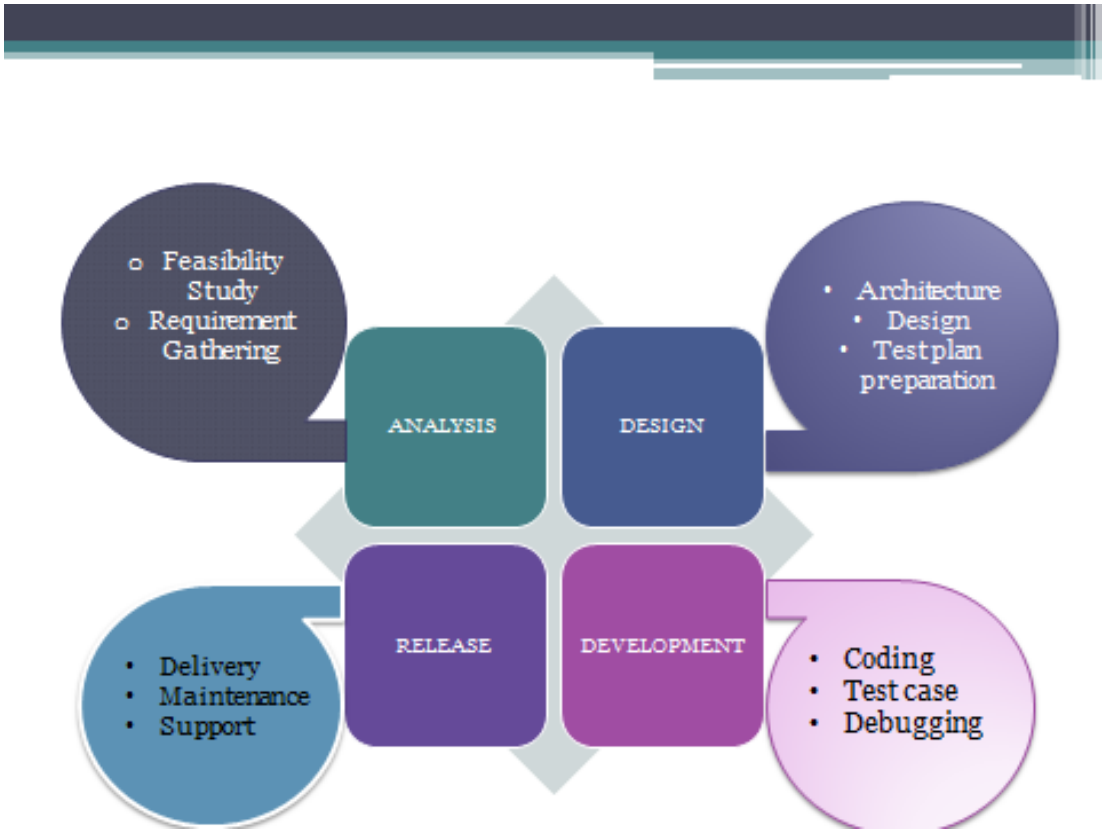

Figure 8: Waterfall Model (*software development-tutorials***)**

CHAPTER 3

METHODOLOGY FOR DESIGN

3.1 DATA FLOW DIAGRAM:

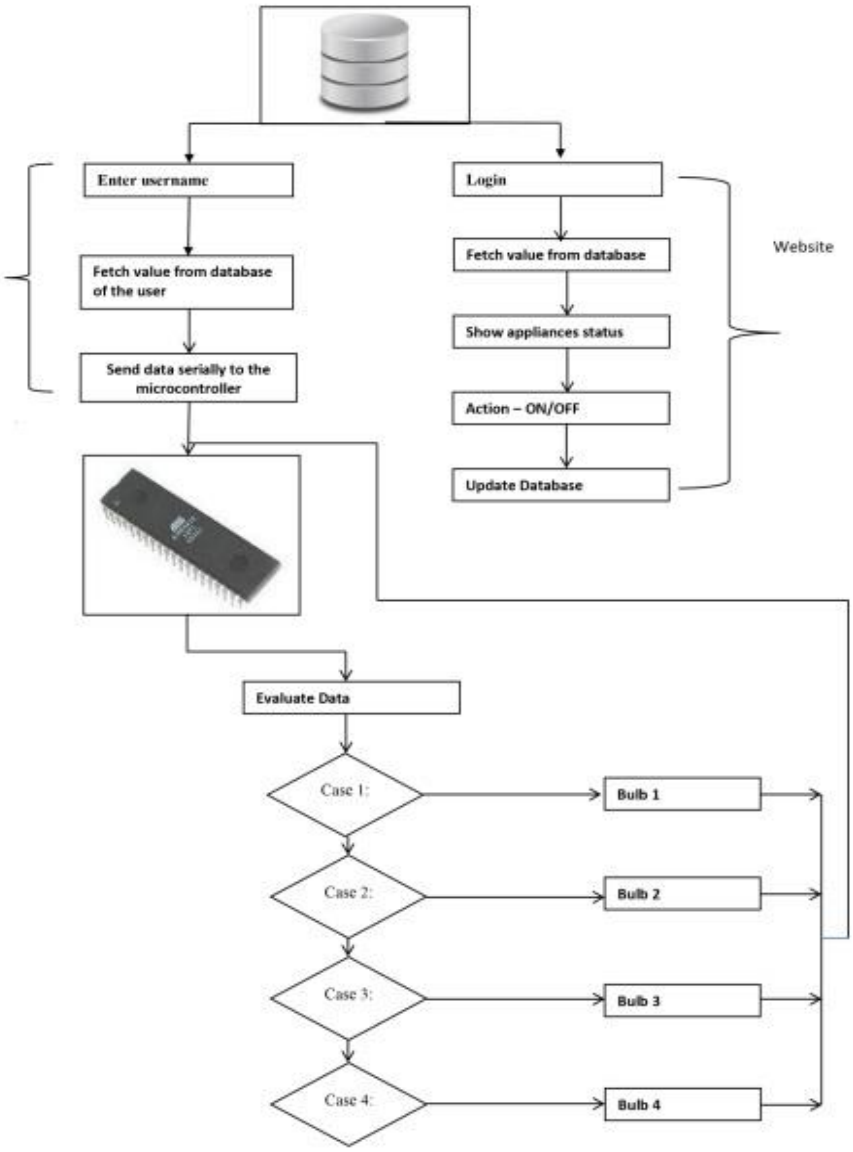

Figure 9: DFD of the whole system

3.2 ARCHITECTURE USED FOR THE PROJECT

4 TIER ARCHITECTURE USED FOR THIS SYSTEM

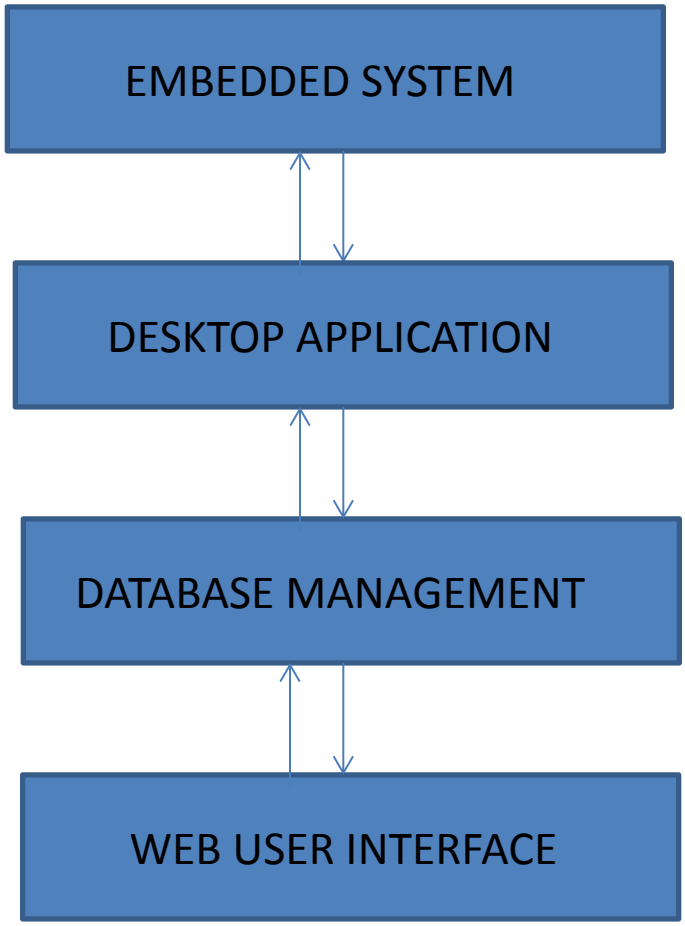

Figure 10: Tier Architecture

As Discussed in CHAPTER 3 Section 2.5 about the Web User Interface, Java server Pages were developed to make the interface and hosted on localhost server for this project. It provides a connection between User and Database in which status of the Appliance is stored and modified as per the user requirements. The Database is made in Oracle Database, through this the user can retrieve and store data via the Web interface provided.

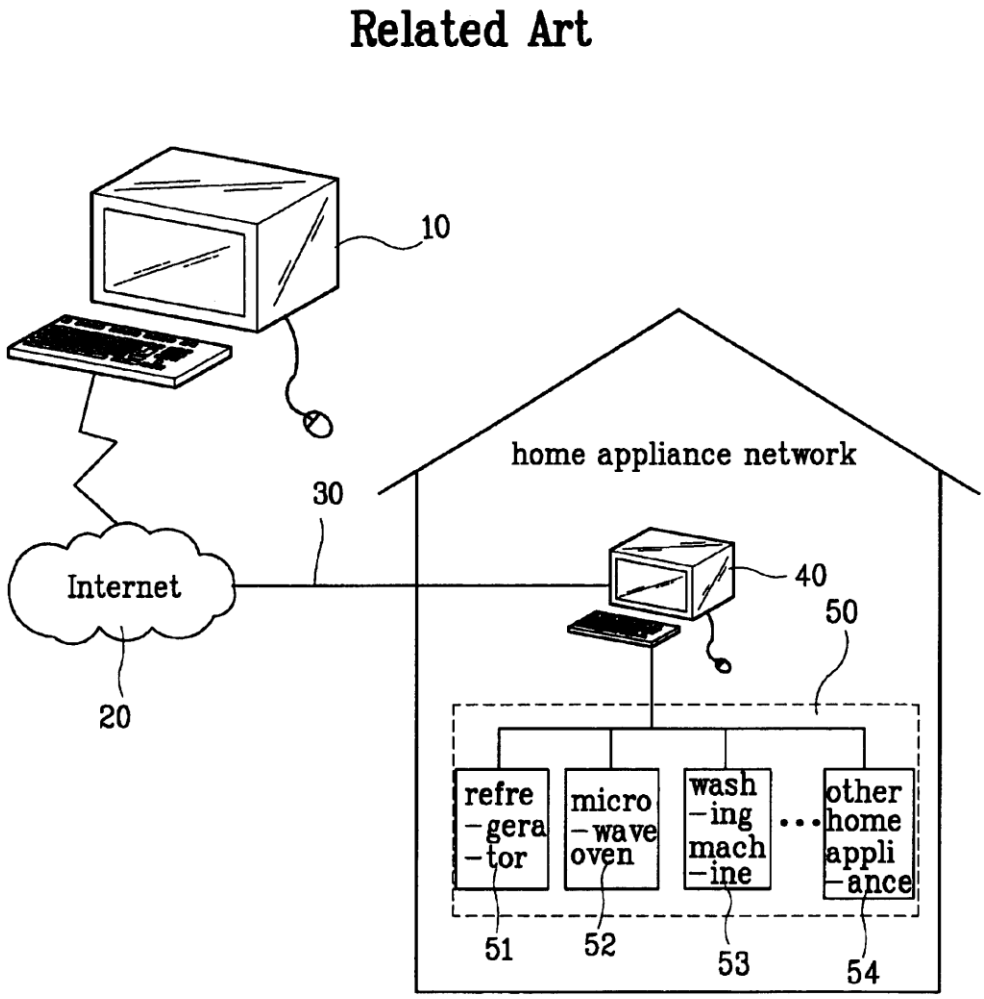

Figure 11: Mechanism

Embedded system:

- HARDWARE USED
    1. AVR BOARD
    2. LCD DISPLAY
    3. ATMEGA 16 MICROCONTROLLER
    4. MAX RS 232 IC

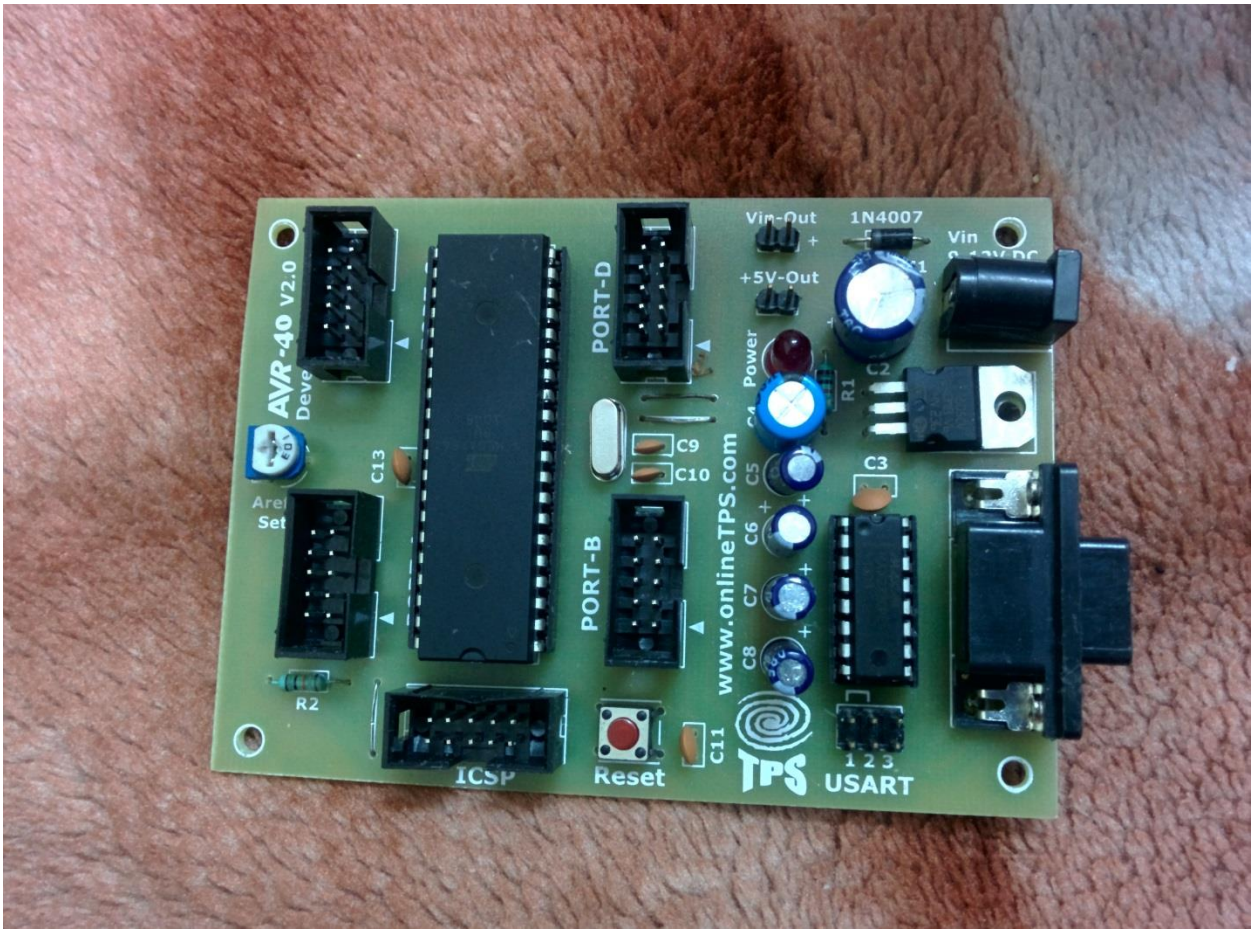

Figure 12: AVR Board (*self clicked*)

- SOFTWARE USED

1. BASCOM-AVR by MCS Electronic
2. AVR-DUDE
3. Eclipse-Kepler
4. Oracle

We used an AVR microcontroller board to control the relay action which will consequently control the switching of AC devices (home appliances). The microcontroller will communicate serially with the PC with a software installed on it that communicates with the database server containing the status information of the AC appliances.

The user will control his home appliances with the help of the website hosted on WWW. This website will automatically update the database as per the action performed by the user regarding the status information of his appliances.

Embedded code:

```
' This Code is Written By Anirudh Kaushik
$regfile = "M16def.dat "                    ' ATMEGA 16 library file
$crystal = 2000000                          ' Crystal Frequescy set to 2MHZ
$baud = 9600                                ' Baud Rate set To 9600
' LCD Configuration
Config Lcd = 16 * 2
Config Lcdpin = Pin , Db4 = Portb.4 , Db5 = Portb.5 , Db6 = Portb.6 , Db7 = Portb.7
, E = Portb.3 , Rs = Portb.2
' ADCD COnfiguration
Config Adc = Single , Prescaler = Auto , Reference = Avcc
Start Adc
' Timer COnfiguration
Config Timer1 = Pwm , Prescale = 1 , Pwm = 8 , Compare A Pwm = Clear Down ,
Compare B Pwm = Clear Down
Start Timer1
```

```
Dim A As String * 10
Config Portc = Output
Config Portd.2 = Output
Do
Cls
A = Waitkey()
Lcd A
Waitms 500
If A = "a" Then
Portc.0 = 1
Portc.2 = 0
Elseif A = "b" Then
Portc.0 = 0
Portc.2 = 0
Elseif A = "c" Then
Portc.1 = 1
Portc.2 = 0
Elseif A = "d" Then
Portc.1 = 0
Portc.2 = 0
End If
Loop
End
```

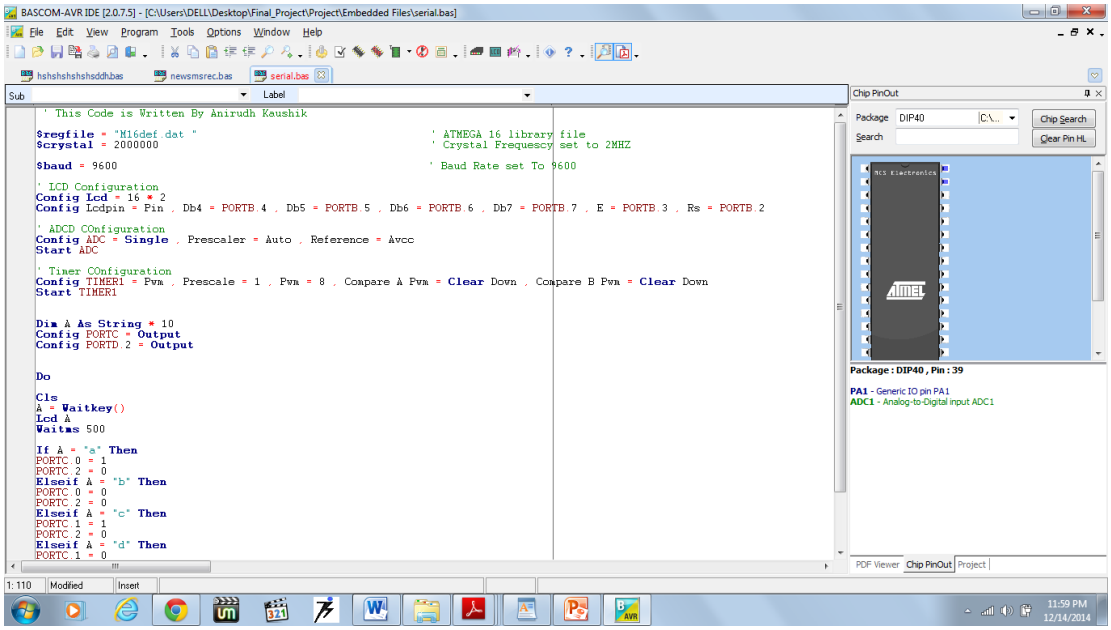

Figure 13: Screenshot of BASCOM-AVR Software Window

AVR-DUDE: AVRDUDE - AVR Downloader Uploader - is a program for downloading and uploading the on-chip memories of Atmel's AVR microcontrollers. It can program the Flash and EEPROM, and where supported by the serial programming protocol, it can program fuse and lock bits. AVRDUDE also supplies a direct instruction mode allowing one to issue any programming instruction to the AVR chip regardless of whether AVRDUDE implements that specific feature of a particular chip. AVRDUDE can be used effectively via the command line to read or write all chip memory types (eeprom, flash, fuse bits, lock bits, signature bytes) or via an interactive (terminal) mode. Using AVRDUDE from the command line works well for programming the entire memory of the chip from the contents of a file, while interactive mode is useful for exploring memory contents, modifing individual bytes of eeprom, programming fuse/lock bits, etc. AVRDUDE supports the following basic programmer types: Atmel's STK500, Atmel's AVRISP and AVRISP mkII devices, Atmel's JTAG ICE (both mkI and mkII, the latter also in ISP mode), appnote avr910, appnote avr109 (including the AVR Butterfly), serial bit-bang adapters, and the PPI (parallel port interface). PPI represents a class of simple programmers where the programming lines are directly connected to the PC parallel port. Several pin

configurations exist for several variations of the PPI programmers, and AVRDUDE can be be configured to work with them by either specifying the appropriate programmer on the command line or by creating a new entry in its configuration file.

All that's usually required for a new entry is to tell AVRDUDE which pins to use for each programming function.

The code is burnt via AVR-DUDE and uploaded into the hardware via USB to SERIAL PORT CONVERTOR.

3.3 POWER SECTION DESIGNING

- The Power section consists of Regulator 7805.
- The adaptor is connected to 220 volts AC household supply, and this supply is convertedto 12 volts DC supply.

The 12V DC is fed to a Diode, so that there is no backflow of current.

Further the supply is fed to a 0.33µF capacitor,to avoid voltage spike

Figure 14: Power supply through Diode (*en.wikepedia.org*)

This supply is then connected to the first pin of Voltage Regulator 7805.

- 7805 converts this supply into 5v
- The 5V supply is obtained from the 3rd pin of 7805
- Again the 5V supply is fed to 0.33µF capacitor.

An LED is connected to it to check whether the supply is available or not.

CHAPTER 4

RESULTS AND ANALYSIS

4.1 ANALYSIS

When the device is switched on the LCD displays “Initializing”, this means that the entire system is getting ready for functioning, also the microcontroller used in the Module will be ready to receive and transmit signals. It takes around 30 seconds for the whole system to initialize and after the process is over the LCD displays a message that to show that the system is ready for use.

A person can send the desired request to the Database which can be ON or OFF.

Sending an ‘ON’ request means the system will start the equipment for unlimited time, unless and until an ‘OFF’ message is fed to the system, and updates the database of the user that the device has been switched on.

Sending ‘OFF’ request switches the device off and updates the database of the user that the device has been switched off.

The flexibility to the users is that a user has the power to control the operation of the device, sitting in any corner of the globe totally just through the website.

4.2 RESULTS

This section describes the output of the implemented system. Several testing has been performed to ensure its executed and produce the intended result. The prototype system is designed such that user Login through the website and check the status of the appliances

which are connected through the AVR Board and then do the needful changes accordingly which then triggers the relay on the Home Automation Board. The incoming request is processed by the microcontroller and the action is then taken. The new status of the appliance is then updated in the database which can be viewed by the user. The status of the devices (turned ON or turned OFF) can be seen.

The system worked properly in nine tests out of ten. It reverted notifications and processed the commands.

CHAPTER 5

CONCLUSION

5.1 CONCLUSION

In the preceding chapters, we analyzed the working of Microcontrollers and identified some of its drawbacks. Along with this, we also implemented Embedding system for the working of the appliances. NetBeans was chosen as the platform to build and model the website. Website was developed using J2EE technology as explained in the previous chapter. Architecture and design is based upon the standards of the industry and the complete methodology is described as step by step procedure which provides insight into the strategy of the system design and the fine points that need to be considered while designing and implementing the whole project.

Actions performed through the website are well connected to the database which is further connected to an application running on the system situated at the desired household. The running application is well connected to the database and hence can detect the actions performed by the user through the website.

The application is then further connected to the AVR Board and hence conveys the signal of the desired action performed (ON/OFF) by the user on the other end. Appliances of the household are connected to the AVR-Board. The microcontroller installed on the AVR-Board will detect the signal and compute it accordingly and will perform the desired action required (ON/OFF).

I anticipate that the use of this application will cut down the energy consumption to a significant amount in different households and offices.

The project is being well made to tackle the energy consumption issues and user interface makes user to handle well the system and provides good communication between user and the system.

REFERENCES

[1] Sang Kyun Lee, Ki Tae Oh, Yeon Kyoung Lee, Chang Ho Kim, *"Home appliance networking system and method for controlling the same"*, Lg Electronics Inc. ,2006.

[2] Ryan J.L, "Home Autmation", *Electronics &Communication IEEE Journal*, colume 1, issue 4, Page 185-192, 2002.

[3] Inoue M, Uemura K, Minagawa Y, Esaki, Mitsunobu, Honda Y, "A Home Automation System", *Consumer Electronics, IEEE Transactions*, volume CE-31, issue 3, Page 516 – 527,2007.

[4] Mr. Abhishek Vichare, Ms. Shilpa Verma, "*Embedded Web Server for Home Appliances*", International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622, National Conference on Emerging Trends in Engineering & Technology (VNCET-30 Mar'12), 2012.

[5] Paul D. Arling, Jeremy Black, Christopher Chambers, Mauro Dresti, Patrick H. Hayes, Robert Lilleness, Wayne Scott, Allen Yuh, "*Home appliance control system and methods in a networked environment*", Universal Electronics Inc., 2006

[6] http://www.mikrocontroller.net/articles/AVRDUDE

[7] http://www.google.com/patents/US7136709

[8] http://www.ladyada.net/learn/avr/avrdude.html

[9] http://codeglobe.blogspot.in/2009/02/serial-port-communication-in-java.html

[10] http://www.ijera.com/special_issue/VNCET_Mar_2012/35.pdf

[11] http://www.academia.edu/4245054/ETHERNET_BASED_HOME_APPLIANCES_CONTROL

[12] http://corefuture.com/knowledgetour/remote-control-for-your-home-appliances-through-internet/

[13] http://en.wikipedia.org/wiki/Home_automation

APPENDIX – A

CODING

- **Website Modules**

  i. **Adminpage.jsp**

```
<table  width="100%" style="background: rgba( 44, 44, 144, 0.9);" >
<tr><td style="background: rgba( 44, 44, 144, 0.9);">
<jsp:include page="header.jsp"/>
</td>
</tr>
</table>
<%@ page import="java.sql.*;"%>
<%
{
Connection con;
Statement sta;
ResultSet rs;
String dev,devid,i,user,pass,notify;
int id;
con=null;
sta=null;
rs=null;

   try
        {
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver"); //Load Drivers
    con =
DriverManager.getConnection("jdbc:odbc:home;user=sa;password=.");
```

```
            sta=con.createStatement();
            rs=sta.executeQuery("SELECT * FROM DEVICES");
            out.println("<table cellpadding='2' cellspacing='0' border='1'
bgcolor='#e8eef7' valign='top'
width='106%'><tr><th>Device</th><th>Device
ID</th><th>Status</th><th>Action</th>");

            while(rs.next())
           {
                  out.println("<tr bgcolor='#ffffff'>");
                  dev=rs.getString(2);
                  devid=rs.getString(3);
                  i=rs.getString(4);
                  notify=rs.getString(5);
                  user=rs.getString(6);
                  pass=rs.getString(7);
                  //out.println(i);
                  out.println("<td>"+dev+"</td>");
                  out.println("<td>"+devid+"</td>");

                        if(i.equals("0"))
                        {
                              out.println("<td>OFF</td>");
out.println("<td><a
href='statusupdate.jsp?dev="+dev+"&devid="+devid+"&status=1'>ON</
a></td>");
                        }
                        else if(i.equals("1"))
                        {
                              out.println("<td>ON</td>");
```

```
out.println("<td><a
href='statusupdate.jsp?dev="+dev+"&devid="+devid+"&status=0'>OFF<
/a></td>");
                        }


                out.println("<td>"+notify+"</td>");
                out.println("<td>"+user+"</td>");
                out.println("<td>"+pass+"</td>");
                out.println("</tr>");



        }

                out.println("</table>");


                rs.close();
                sta.close();
                con.close();
        }
                catch (Exception e)
                {
                    System.err.println("Exception: "+e.getMessage());
                }
}
%>
<jsp:include page="footer.jsp"/>
```

ii. **Device.jsp**

```
<table border="1" valign="top" cellspacing="4" cellpadding="2">
<th></th>
<td>
<form name= "" method="post" action="test1.jsp">
ON<input type="radio" name="status" value="y" /><br>
OFF<input type="radio" name="status" value="n" />

</td>
<td>
 <%@ page import="java.sql.*;"%>
<%
Connection con;
Statement sta;
ResultSet rs;
PreparedStatement pstmt;
con=null;
sta=null;
rs=null;
pstmt=null;
String i;

   try
        {
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver"); //Load Drivers
    con =
DriverManager.getConnection("jdbc:odbc:home;user=sa;password=.");
                sta=con.createStatement();
```

```
                rs=sta.executeQuery("select * from devices where dev='fan'
and devid='1' ");
                while(rs.next())
                {
                        i=rs.getString(4);

                        if(i.equals("n"))
                        {
                                out.println("OFF");
                        }
                        else if(i.equals("y"))
                        {
                                out.println("ON");
                        }
                }
                sta.close();
                rs.close();



                con.close();
        }
                catch (Exception e)
                {
                    System.err.println("Exception: "+e.getMessage());
                }

%>
</td>
</tr>
```

```
</table>
<input type="submit" name="delete" value="Submit">
</form>
```

iii. **Footer.jsp**

```
<link rel="stylesheet" href="images/style.css">

 <body>

  <hr color="Gray">
  <center>
  Designed & Developed by
        Chirag Ahuja &copy; 2012-2013    
        </center>
 </body>
```

iv. **Headet.jsp**

```
<head>
  <title> </title>
 <link rel="stylesheet" href="images/style.css" type="style/css">
 </head>

<body>

 <div id="header-bar">
        <div class="header-logo">
        <font color="white" size="20px">Intelligent Home</font>
```

```
        </div>
</div>

 </body>
```

v. **Index.Jsp**

```
<table border="0" width="100%" style="background: rgba( 44, 44, 144,
0.9);">
<tr>
<td>
<jsp:include page="header.jsp"/>
</td>
<td align="right">
<jsp:include page="login.jsp"/>
</td>
</tr>
</table>
</td>
</tr>

<tr>
<td width="20%">
<h1>WELCOME</h1>

</td>
</tr>
```

```
<tr>
<td colspan="3">

<jsp:include page="footer.jsp"/>
</td>
</tr>
</table>
</body>
```

vi. **Layout.jsp**

```
<head>
<style>
#s a
{
        color:white;
        text-decoration:none;
}

#s a:visited
{
        color: white;
        text-decoration:none;
}

#s a:hover
{
        color: red;
        text-decoration:none;
```

```
}

</style>
</head>
<table width="100%" cellpadding="0" cellspacing="0" border="0"
height="20" >
        <tr>
                <td height="18" ><div id="header-bar">
<jsp:include page="header.jsp"/>
<div class="login">
<table border="0" style="margin-top:-10px;">
<tr>
<td>
<font color="white">Welcome 
<%@ page import="java.lang.*;"%>
<%
String name=(String)session.getAttribute("user");
String n=name.toUpperCase();
out.println(n);
%></font> 
</td>

<td>
<div style="width:50px; height:40px;">
<!--<jsp:include page="image.jsp"/> -->
</div>
</td>

<div id="s">
<td>
```

```
<a  href="#">Settings</a>
</td>

<td valign="center">
<a href="signout.jsp">Logout</a>
</td>
</div>
</tr>
</table>
</div>
</div>
</td>
        </tr>


<tr>
<td>
<table width="100%" cellpadding="0" cellspacing="0" border="0" >
        <tr>
<!--Links-->
<td width="120" valign="top">
<jsp:include page="links.jsp"/>
</td>


<td valign="top">
<!--Content-->
<table width="100%" cellpadding="0" cellspacing="0" border="0">
<tr>
```

```
<td height="600" valign="top">
```

vii. **Layoutdown.jsp**

```
<link rel="stylesheet" href="images/style.css">
</td>
</tr>
</table>

                </td>
        </tr>
</table>
</td>
</tr>

<tr>
<td valign="top">
<jsp:include page="footer.jsp"/>
</td>
</tr>
</table>
```

viii. **Links.jsp**

```
<%@ page import="java.sql.*;"%>
<%
Connection con;
Statement sta;
```

```
ResultSet rs;

con=null;
sta=null;
rs=null;
String username=request.getParameter("user");
String password=request.getParameter("pass");


   try
        {
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver"); //Load Drivers
    con =
DriverManager.getConnection("jdbc:odbc:chirag;user=sa;password=.");
          sta=con.createStatement();
          rs =  sta.executeQuery("SELECT * FROM LINKS");
          out.println("<table width='100%' cellpadding='2' cellspacing='0'
border='0'>");
          while(rs.next())
        {
                  out.println("<tr> <td>");
                  out.println("<a
href='"+rs.getString(2)+"'><b>"+rs.getString(3)+"</a>");
                  out.println("</tr></td>");

        }
                  out.println("</table>");

                  rs.close();
                  sta.close();
```

```
            con.close();
        }
            catch (Exception e)
            {
                System.err.println("Exception: "+e.getMessage());
            }



%>
```

ix. **Login.jsp**

```
<head>
<script type="text/javascript" src="js/login.js"></script>
</head>
<table border="0" valign="top" cellpadding="0px" cellspacing="0px"
style="text-color:'white'; background: rgba( 44, 44, 144, 0.9);">
<form name="login" method="get" >
        <tr>
        <td>
        <font color="white">Username:</font>
        </td>
        <td><font color="white">Password:</font>
        </td>
        </tr>

        <tr>
        <td>
        <input type="text" spellcheck="false" name="user">
```

```
</td>
<td>
<input type="password" name="pass">
</td>
<td>
<input type="Submit" name="Submit" value="Sign In"
onclick="onLogin();" />
</td>
</tr>

<tr>
<td>
<input type="checkbox" name="" id="" value="yes"><font
color="white">Stay signed</font>
</td>
<td>
<a href="#"><font color="white">Forgot Password ?</font></a>
</td>
</tr>
</form>
</table>
```

x. **Loginjsp.jsp**

```
<script language="Javascript">
function warn()
{
    alert("Wrong Username Password Entered");
}
```

```
</script>
<%@ page import="java.sql.*,java.util.*"%>
<%
String USERNAME=request.getParameter("user");
String PASSWORD=request.getParameter("pass");
if((USERNAME.equals("admin")) && PASSWORD.equals("admin"))
        {
                response.sendRedirect("adminpage.jsp");
        }
else{
try
{
boolean flag=false;
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
Connection
con=DriverManager.getConnection("Jdbc:Odbc:home","sa",".");
PreparedStatement ps=con.prepareStatement("select username,password
from devices where username=? and password=?");
ps.setString(1,USERNAME);
ps.setString(2,PASSWORD);
ResultSet rs=ps.executeQuery();
while(rs.next())
{
        flag = true;
}
if(flag)
{

        String name=request.getParameter("user");
        session.setAttribute("user",name);
```

```
response.sendRedirect("userpage.jsp");


}

else
{

response.sendRedirect("index.jsp");
%>
<%
}

}
catch(Exception e)
{
out.println(e.getMessage());
}
}
%>
```

xi. **Signout.jsp**

```
<%
session.invalidate();
response.sendRedirect("index.jsp");
%>
```

xii. StatusUpdate.jsp

```
<html>
```

```
<head>
 <title> Status Update </title>

</head>

<body>

<%@ page import="java.sql.*;"%>
<%
String s=request.getParameter("status");
String user=(String)session.getAttribute("user");



Connection con;
Statement sta;
ResultSet rs;
PreparedStatement pstmt;
con=null;
sta=null;
rs=null;
pstmt=null;

//out.println("1");
   try
        {
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver"); //Load Drivers
    con =
DriverManager.getConnection("jdbc:odbc:home;user=sa;password=.");
```

```
            pstmt = con.prepareStatement("UPDATE devices SET status =
'"+s+"' where dev='fan' and devid='1' and username='"+user+"'");
            //out.println("2");
             int r=pstmt.executeUpdate();
             //out.println("3");
                    pstmt.close();
                    //out.println("4");
                    if (r>0)
                    {
                            response.sendRedirect("userpage.jsp");
                    }
                    else
                    {
                            response.sendRedirect("index.jsp");

                    }

                    con.close();
            }
                    catch (Exception e)
                    {
                        System.err.println("Exception: "+e.getMessage());
                    }

 %>


 </body>
</html>
```

xiii. **Userpage.jsp**

```
<table  width="100%" style="background: rgba( 44, 44, 144, 0.9);" >
<tr><td style="background: rgba( 44, 44, 144, 0.9);">
<jsp:include page="header.jsp"/>
</td>


<td style="background: rgba( 44, 44, 144, 0.9);margin-top:-10px;"
align="right" >
<%
String user=(String)session.getAttribute("user");
out.println((user));
%>


<a href="signout.jsp">Logout</a>
</td>
</tr>
</table>
<%
if(request.getSession(true).getAttribute("user")==null)
{
        session.invalidate();
        response.sendRedirect("index.jsp");
}
else
%>
<h1>YOUR DEVICES INFORMATION</h1>
<div style="margin:10 50 5 400;">
```

```
<jsp:include page="userpage1.jsp"/>
</div>
<jsp:include page="footer.jsp"/>
```

xiv. **Userpage1.jsp**

```
<%@ page import="java.sql.*;"%>
<%
{

String user=(String)session.getAttribute("user");
Connection con;
Statement sta;
ResultSet rs;
String dev,devid,i;
int id;
con=null;
sta=null;
rs=null;

   try
        {
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver"); //Load Drivers
    con =
DriverManager.getConnection("jdbc:odbc:home;user=sa;password=.");
          sta=con.createStatement();
          rs=sta.executeQuery("SELECT * FROM DEVICES where
username='"+user+"'");
          out.println("<table cellpadding='2' cellspacing='0' border='1'
bgcolor='#e8eef7' valign='top'
```

```
width='106%'><tr><th>Device</th><th>Device
ID</th><th>Status</th><th>Action</th>");

          while(rs.next())
        {
                out.println("<tr bgcolor='#ffffff'>");
                dev=rs.getString(2);
                devid=rs.getString(3);
                i=rs.getString(4);
                //out.println(i);
                out.println("<td>"+dev+"</td>");
                out.println("<td>"+devid+"</td>");

                        if(i.equals("0"))
                        {
                                out.println("<td>OFF</td>");
out.println("<td><a
href='statusupdate.jsp?dev="+dev+"&devid="+devid+"&status=1'>ON</
a></td>");
                        }
                        else if(i.equals("1"))
                        {
                                out.println("<td>ON</td>");
out.println("<td><a
href='statusupdate.jsp?dev="+dev+"&devid="+devid+"&status=0'>OFF<
/a></td>");
                        }
```

```
                out.println("</tr>");



            }

                out.println("</table>");


                rs.close();
                sta.close();
                con.close();
            }
                catch (Exception e)
                {
                    System.err.println("Exception: "+e.getMessage());
                }
}
%>
```

❖ EMBEDDING CODING

```
 ' This Code is Written By Anirudh Kaushik
$regfile = "M16def.dat "                    ' ATMEGA 16 library file
$crystal = 2000000                          ' Crystal Frequescy set to 2MHZ
$baud = 9600                                ' Baud Rate set To 9600
' LCD Configuration
Config Lcd = 16 * 2
Config Lcdpin = Pin , Db4 = Portb.4 , Db5 = Portb.5 , Db6 = Portb.6 , Db7 = Portb.7
, E = Portb.3 , Rs = Portb.2
' ADCD COnfiguration
Config Adc = Single , Prescaler = Auto , Reference = Avcc
Start Adc
' Timer COnfiguration
Config Timer1 = Pwm , Prescale = 1 , Pwm = 8 , Compare A Pwm = Clear Down ,
Compare B Pwm = Clear Down
Start Timer1
Dim A As String * 10
Config Portc = Output
Config Portd.2 = Output
Do
Cls
A = Waitkey()
Lcd A
Waitms 500
If A = "a" Then
Portc.0 = 1
Portc.2 = 0
Elseif A = "b" Then
Portc.0 = 0
```

```
Portc.2 = 0
Elseif A = "c" Then
Portc.1 = 1
Portc.2 = 0
Elseif A = "d" Then
Portc.1 = 0
Portc.2 = 0
End If
Loop
End
```