

**Design and Construction of a  
Distributed Computer Controlled  
Switching System**

**Ubah Thomas Ubah**

**(2001/12131EE)**

**A Thesis submitted to the  
Department of Electrical and  
Computer Engineering, Federal  
University of Technology, Minna,  
Niger State.**

**November, 2007**

## **DEDICATION**

I hereby dedicate this project to my Saviour Jesus Christ, to the parents that God gave me;

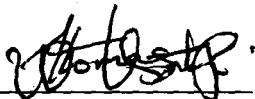
Col and Mrs. JIP Ubah, my brother and my beloved sisters.

## Attestation/Declaration

I, Ubah Thomas Ubah, declare that this work was done by me and has never been presented elsewhere for the award of a degree. I also hereby relinquish the copyright to The Federal University of Technology, Minna.

UBAH THOMAS UBAH

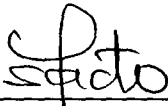
*(Name of student)*

  
\_\_\_\_\_

*(Signature and date)*

DR. J. TSADO

*(Name of supervisor)*

 3/12/07  
\_\_\_\_\_

*(Signature and date)*

ENGR. M. D. ABDULLAHI

*(Name of H.O.D)*

\_\_\_\_\_

*(Signature and date)*

\_\_\_\_\_

*(Name of external examiner)*

\_\_\_\_\_

*(Signature and date)*

## **ACKNOWLEDGEMENT**

Firstly I would like acknowledge God, who brought me into this school, gave me a purpose to live and the spirit of a sound mind.

I like to thank my parents, Col and Mrs. JIP Ubah, for all their support through my years in this institution my sisters, especially Enada Winifred Yusuf, for believing in me.

I like to appreciate all members of Electrical and Computer Engineering, class of 2007, for their corporation. To my friends, for giving me the right competition that helped me grow and learn.

Thanks to members of YWAP, for the times we've shared, and to Blessing Adaeze Ifemene for encouraging me in my endeavours.

Finally I like to thank all lecturers of the Department of Electrical and Computer Engineering, especially my supervisor, Dr. Jacob Tsado, for his support and kindness.

## **ABSTRACT**

A prototype of a distributed computer controlled switching system is presented in this project. The design takes advantage of the communication capabilities of the microcomputer and its ability to interface with external devices to implement the system.

The completion of this project results in a switching system that can be controlled from any computer on the internet.

# Table of Contents

Dedication .....	ii
Declaration .....	iii
Acknowledgement .....	iv
Abstract .....	v
Table of Contents.....	vi
List of Figures.....	viii
List of Tables.....	ix
Chapter One: Introduction.....	1
1.1 Goal.....	3
1.2 Features.....	3
1.3 Methodology.....	4
1.4 Scope of Study.....	4
Chapter Two: Literature Review.....	6
2.1 Historical Background.....	6
2.1.1 Solid State Switches.....	6
2.1.2 Embedded Systems.....	7
2.1.3 Computer Based Control .....	9
2.1.4 Computer Networks and the Internet .....	10
2.2 Computer Control Applications .....	12
2.3 Theoretical Background .....	12
2.4 Overview of System Operations .....	13
2.4.1 Network Implementation .....	14
2.4.2 Switching Circuitry .....	15
2.4.3 Software Implementation .....	16
Chapter Three: Design and Implementation .....	18

Chapter Three: Design and Implementation .....	18
3.1 Hardware Implementation .....	18
3.1.1 +5V Power Supply .....	18
3.1.2 Microcontroller Software Control Module .....	20
3.1.3 Opto-triac .....	24
3.1.4 RS-232 Interface .....	26
3.1.5 System Status Reporting .....	27
3.2 Software Implementation .....	27
3.2.1 Application Server .....	27
3.2.2 Application Client .....	28
3.2.3 Apache HTTP Server .....	28
3.3 Design Calculations .....	29
3.3.1 Reset Current Selection .....	29
3.3.2 Current Limiting resistance .....	30
3.3.3 Relay Selection .....	30
3.3.4 Power Triac Selection .....	31
3.3.5 Transistor Relay Driver .....	31
Chapter Four: Tests, Results and Discussion .....	33
4.1 Testing Procedures .....	33
4.1.1 Voltage Regulator .....	33
4.1.2 Microcontroller .....	33
4.1.3 Electronic Switches .....	34
4.1.4 Software .....	34
4.2 Results .....	34
4.3 Discussions .....	35
Chapter Five: Conclusion and Recommendation .....	36
5.1 Conclusion .....	36
5.2 Recommendations .....	36

## List of Figures

Figure 2.1 Operational Block Diagram of the System .....	13
Figure 3.1 5v Power Supply .....	19
Figure 3.2 On Board Switch Control .....	21
Figure 3.3 Command Byte used for Control Signal .....	22
Figure 3.4 Opto-Triac (moc3023) .....	24
Figure 3.5 Triac Switched Load .....	25
Figure 3.6 Pulse Modulated Wave .....	25
Figure 3.7 Logic Level Converter for the RS-232 .....	26
Figure 3.8 R-C Combination for Reset .....	29
Figure 3.9 Opto-Triac Arrangement .....	30
Figure 3.10 Relay Driver .....	31
Figure 3.11 Switching Circuitry .....	32
Figure 4.1 Sketch of oscilloscope waveform .....	35



## List of Tables

Table 3.1 Command Byte Usage .....	23
Table 4.1 Test Results for Voltage Regulator .....	34
Table 4.2 Results of microcontroller test .....	35

# **CHAPTER ONE**

## **INTRODUCTION**

To control a device, machine or system means that you are capable of making the system to operate in a particular way [1]. This pattern of operation could be predefined or spontaneous.

This control can be in the form of switching a device on or off, regulating the operations of a machine or setting the operating conditions or constraints by which a given system should work by.

Over the years, different ways of controlling devices or systems have been developed. Some systems use on board controls in the form of buttons, joysticks or even touch screens to control the operations of disparate systems. These controls may be used to control the movement of a device by moving a joystick or perform an action by pushing a button.

Other systems employ the use of automated system control. Here a system operates on a set of predefined directives. These directives are usually preprogrammed into the device before operation begins. Some systems also, include specialized hardware for control and monitoring. Such hardware are usually driven by components referred to as embedded systems [2].

Remote controls are also used for device control. These controls could be either wired or wireless. For the wired, the remote control is attached to the controlled system via a cable; this cable could be extended to ease movement of personnel. The wireless

remote controls make use of either infrared or radio frequency (RF) signals to transmit control commands to the controlled system.

More recently, use of computers has also been incorporated to aid in the control of devices and systems [3]. These computers range from function specific computers, which were designed and programmed to carry out a specific function, to general purpose computers, which can be extended using electronic hardware and software to control or monitor a specific system.

Although, all the aforementioned methods of control are effective and find usage in varying areas of device control, the use of computers have proven to be the most flexible of all. This proves to be true since computers are extensible through the use of specialized hardware and custom written software to fit the role of various tasks.

The use of computers to control and monitor virtually all devices, including home appliances, security systems, factory processes, etc, is becoming increasingly popular. Computers add an extra level of effectiveness based on the fact that all controlled devices can be administered from a single location while using a single host computer as your command console. This of course assumes that all controlled devices are connected to that one computer via extension hardware.

Even with this, however, comes the restriction of a single command or control point. There may be a need for flexibility of control positions, especially in cases of emergency, were the control point may not be readily accessible.

The advent of digital computer networks and the internet looks to solve the problem of the single control point. Since in correctly configured computer networks, different computers can communicate with each other almost seamlessly, specialized

software and hardware can be designed to work with networks in such a way that a command console can be readily accessed from any computer on the network, usually through a web interface [4].

Many programming languages, especially high level languages, have the capability to communicate across a computer networks. These make it suitable to write software which will act as a user interface or control console, that can be accessed on any computer on the network, as an application software or as the content of a web browser. These types of software are usually written in the form of a client and server, where the server is located on a system that can access the controlled devices and the client is usually a user interface accessible from the network.

With the above scenario, it will be possible to control devices not just from one location but from virtually all computers which are connected in a network pool or possibly the internet.

## **1.1 GOAL**

This project is aimed at creating a switching system, which will be accessible and controllable over a digital computer network or the internet. Certain appliances will be connected to a host computer, which will serve as a server, and a control console, which will be accessible from any system on the network, via a web browser, will be used to control the on or off states of the appliances from the remote system

## **1.2 FEATURES**

- ❖ Capability to control devices from any location on a network.
- ❖ A control console or user interface accessible on all computers on the network.

- ❖ Interface circuit that will be connected to the server system and serve as the switch for the controlled devices.
- ❖ An on board set of controls to control the devices in case server computer is offline.

### **1.3 METHODOLOGY**

The various loads will be connected to a computer terminal through a custom built interface circuitry, which will act as the switching or control device. The computer will employ the use of its RS-232 interface, which is a serial communications port for sending and receiving control and status bytes.

This computer will also run an application, which acts as a server software. This software directly communicates with the serial port and will be used to send and receive data to and from the serial port. Also a web server which will be used to serve the page containing the client or command console will also be installed on the system.

The server system will be connected to a network, where other computers on the network will be able to access the command console user interface, which will be used to control the devices.

This software, both client and server, will be written in the java programming language and the web server of choice will be the Apache HTTP server.

### **1.4 SCOPE OF STUDY**

This project will be limited to controlling three devices. Two of which will be static loads. By static it is meant that they have only two states, on and off. The other load

will be dynamic in the sense that it should be able to work using varying voltage levels.

The type of load can be a regulated load such as a fan or a dimming light.

Since the internet might not be accessible due to time and expense, a small network of three computers will be used to simulate the internet.

## **CHAPTER TWO**

### **LITERATURE REVIEW**

A control system is one which undertakes some function, checks its success and takes further action until the objective is attained [5]. There are usually implemented using either hardware or software singularly, or as a hybrid of both. Control systems can be used to control various quantities which range from physical quantities, electrical appliances or even industrial machinery and processes.

Microcomputers can be added to control systems to extend the functionality of the system. This extension may include improvements in performance of system and increased flexibility, since microcomputers can be programmed to carry out different tasks such as data analysis, and control of physical, electrical or mechanical quantities [3].

The control system implemented in this project uses the microcomputer to control the state of a set of electrical appliances. It also takes advantage of the computers communication abilities to implement a distributed control scheme, where any system on a network can control the electrical appliances. To investigate the workings of the system, a brief discussion of certain vital modules are carried out below.

#### **2.1 HISTORICAL BACKGROUND**

Historical background of the various components and their developments over the years are as follows;

##### **2.1.1 Solid State Switches**

Switches are commonly used to operate electric lights, permanently connected equipment or electrical outlets. Although these switches are usually two state, to switch

the device off and on, there are switches that allow varying current levels to enter the device, these switches are called dimmer switches.

With the introduction of the transistor in the early 1950's, and especially with the development of integrated circuits from the early 1960's onwards [6], designers of electronic equipment, computers and instrumentation have increasingly demanded more efficient and flexible power control over their equipment. With the three-terminal transistor we can make an electric switch, which can be controlled by another electrical switch [7].

Although the transistor was sufficient for most switching tasks, it had limited power capabilities and low switching speeds. The need for higher power control and better switching speeds were needed, as such the thyristor family of switches were born. SCRs (silicon controlled rectifiers) as there are commonly called are 4-layer solid state device that controls current flow.

The SCR was developed by a team of power engineers led by Gordon Hall and commercialized by Frank W. "Bill" Gutzwiller in 1957 [8]. Since then, immense effort have been put into the research of development of better semi-conductor switches. As such, electronic switches now provide immense flexibility in terms of different interfaces for its operations, such as touch plates, soft-touch controls, pressure / light sensor based control, interactive touch-screens (which are widely used in aircraft for lighting control) etc.

### **2.1.2 Embedded Systems**

An embedded system is a special-purpose computer system designed to perform one or a few dedicated functions [4]. It is usually embedded as part of a complete device



including hardware and mechanical parts. In contrast, a general-purpose computer, such as a personal computer, can do many different tasks depending on programming.

In the earliest years of computers in the 1940s, computers were sometimes dedicated to a single task, but were too large to be considered "embedded". Over time however, the concept of programmable controllers developed from a mix of computer technology, solid state devices, and traditional electromechanical sequences.

The first recognizably modern embedded system was the Apollo Guidance Computer, developed by Charles Stark Draper at the MIT Instrumentation Laboratory. At the project's inception, the Apollo guidance computer was considered the riskiest item in the Apollo project. The use of the then new monolithic integrated circuits, to reduce the size and weight, increased this risk.

The first mass-produced embedded system was the Autonetics D-17 guidance computer for the Minuteman missile, released in 1961. It was built from transistor logic and had a hard disk for main memory. When the Minuteman II went into production in 1966, the D-17 was replaced with a new computer that was the first high-volume use of integrated circuits. This program alone reduced prices on quad NAND gate ICs from \$1000/each to \$3/each, permitting their use in commercial products.

Since these early applications in the 1960s, embedded systems have come down in price. There has also been an enormous rise in processing power and functionality. For example the first microprocessor was the Intel 4004, which found its way into calculators and other small systems, but required external memory and support chips.

In 1978 National Engineering Manufacturers Association released the standard for a programmable microcontroller. The definition was an almost any computer-based

controller. They included single board computers, numerical controllers, and sequential controllers in order to perform event-based instructions.

By the mid-1980s, many of the previously external system components had been integrated into the same chip as the processor, resulting in integrated circuits called microcontrollers [4], and widespread use of embedded systems became feasible.

As the cost of a microcontroller fell below \$1, it became feasible to replace expensive knob-based analog components such as potentiometers and variable capacitors with digital electronics controlled by a small microcontroller with up/down buttons or knobs. By the end of the 80s, embedded systems were the norm rather than the exception for almost all electronics devices, a trend which has continued since.

### **2.1.3 Computer Based Control**

At the outset of computer engineering development in the 1950s the main attention was concentrated on the creation of computers which could solve complex mathematical problems. Computers then were stationary and intended for successive or batch problem solution irrelatively to the real time and dynamic parameter changes. Later the machines formed the most widespread class of universal computers for various applications.

By the late 1950s the defence industry and Ministries of Defence took an interest in computers for data processing and control in military systems. The nature of such applications considerably differed from the computational problems which had become traditional by the time.

In the 1960s and 1970s military system consumers were continuously extending the range of computing tasks and stepped up the requirements for solution quality. The trend was accompanied with rapid increase in computer program capacity and complexity. Low labour efficiency made it necessary to actively train new specialists and augment the work force.

Technological progress and element base development lagged behind the requirements for the military computer memory and performance, which were determined by new requirements and tasks. Computer control functions were also becoming more and more complex and critical. This caused gain in requirements for control systems quality, reliability and security.

Many military control systems based on mobile computers were aimed at small-scale (separate specimens, tens or hundreds of pieces) production and use. Therefore, programmers made use of ingenious designs and ignored hardware and software unification and standardization. Consumers did not coordinate the specifications for computer aids, each of which had to be individually adjusted to consumer's tasks. Hence there emerged a real "zoo" of different computers and software complexes often solving the very same problems.

#### **2.1.4 Computer Networks and the Internet**

Research on dividing information into packets and switching them from computer to computer began in the 1960s. The U.S. Department of Defense Advanced Research Projects Agency (ARPA) funded a research project that created a packet switching network known as the ARPANET. ARPA also funded research projects that produced two satellite networks. In the 1970s ARPA was faced with a dilemma: Each of its

networks had advantages for some situations, but each network was incompatible with the others. ARPA focused research on ways that networks could be interconnected, and the Internet was envisioned and created to be an interconnection of networks that use TCP/IP protocols. In the early 1980s a group of academic computer scientists formed the Computer Science NETWORK, which used TCP/IP protocols. Other government agencies extended the role of TCP/IP by applying it to their networks: The Department of Energy's Magnetic Fusion Energy Network (MFENet), the High Energy Physics NETWORK (HEPNET), and the National Science Foundation NETWORK (NSFNET) [9].

In the 1980s, as large commercial companies began to use TCP/IP to build private internets, ARPA investigated transmission of multimedia—audio, video, and graphics—across the Internet. Other groups investigated hypertext and created tools such as Gopher that allowed users to browse menus, which are lists of possible options. In 1989 many of these technologies were combined to create the World Wide Web. Initially designed to aid communication among physicists who worked in widely separated locations, the Web became immensely popular and eventually replaced other tools. Also during the late 1980s, the U.S. government began to lift restrictions on who could use the Internet, and commercialization of the Internet began. In the early 1990s, with users no longer restricted to the scientific or military communities, the Internet quickly expanded to include universities, companies of all sizes, libraries, public and private schools, local and state governments, individuals, and families.

Now embedded systems take advantage of the internet and create web based control interfaces [4]. This avoids the cost of a sophisticated display, yet provides complex input and display capabilities when needed, on another computer. This is

successful for remote, permanently installed equipment. In particular, routers take advantage of this ability.

## **2.2 Computer Control Applications**

The use of computers for control purposes is now very widespread and has cut across virtually all areas of our everyday working. In industries computers and embedded systems are used in controlling and monitoring manufacturing processes. Here the embedded system does the control of the machinery and sends feedback to a computer monitoring station, from which changes that be made to the variables of the production cycle.

Computer based automation has also found its way to the home. More recently, home automation and control systems such as the X10 control module are used to control and monitor electrical appliances in the home [11]. Even security systems are controlled using specialized hardware in conjunction with computers and software.

The military has also found a soft spot for the use of computers in their operations, which range from manufacture processes to communication, control of surveillance systems to control of general military infrastructure. Computer control has particularly found a home in unmanned aircraft control and missile guidance systems [10].

## **2.3 Theoretical Background**

The entire system is made up of a switching circuit, a client/server software implementation, a web server and a computer network. The computer network is used to connect different systems together so as to enable communication and sharing of certain computer resources which include; files, printers, etc.

Web servers are used to publish content, located on a host system, over the internet or any network. This content could be static or dynamic HTML pages, or a link to a downloadable file. In this project it is used to publish a user interface which will be used as a control console.

The user interface is part of a custom written program which includes an application server and a client. The client in essence is the user interface that will be accessible over the network on web pages. The server part listens for method calls from the client. These method calls contain a command value intended for the switching circuit. The command values are received from the client and sent over the serial port interface of the host computer to the switching circuit.

The switching circuit is driven by a 2051 microcontroller and its duty is to receive command values, decode them and perform certain device control tasks based on the command value. When this is done, it sends an acknowledgement to the host computer which in turn sends it to clients as status information.

## 2.4 Overview of System Operation

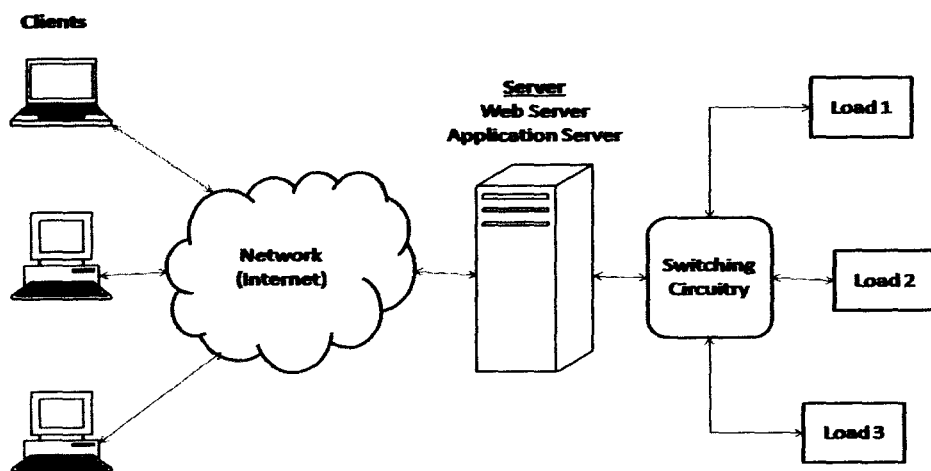


Fig 2.1 operational block diagram of system

The totality of the system can be split into three effective working parts, viz;

- i. The network Implementation
- ii. The switching circuitry
- iii. The software implementation

#### **2.4.1 Network Implementation**

A network is a system used to link two or more computers by wires, cables, or a telecommunications system in order to exchange data. Network users are able to share files, printers, and other resources, send electronic messages, and run programs on other computers [1, 9].

A network has three layers of components: application software, network software, and network hardware [9]. Application software consists of computer programs that interface with network users and permit the sharing of information, such as files, graphics, and video, and resources, such as printers and disks. One type of application software is called client-server. Client computers send requests for information or requests to use resources to other computers, called servers that control data and applications. Another type of application software is called peer-to-peer. In a peer-to-peer network, computers send messages and requests directly to one another without a server intermediary. This project uses the client server approach, where the server consists of an application server and a web server, which is used to distribute the client page.

Network software consists of computer programs that establish protocols, or rules, for computers to talk to one another. These protocols are carried out by sending and receiving formatted instructions of data called packets. Protocols make logical connections between network applications, direct the movement of packets through the

physical network, and minimize the possibility of collisions between packets sent at the same time. The two major protocols used by the network implementation are, TCP/IP for the data communication, and HTTP, used by the web server to deliver HTML pages.

Network hardware is made up of the physical components that connect computers. Two important components are the transmission media that carry the computer's signals, typically on wires or fiber-optic cables, and the network adapter, which accesses the physical media that link computers, receives packets from network software, and transmits instructions and requests to other computers. Transmitted information is in the form of binary digits, or bits (1s and 0s), which the computer's electronic circuitry can process.

#### **2.4.2 Switching Circuitry**

The switching circuit is made up of an AT89C2051 microcontroller which is used to receive and decode command signals sent from the server computer via its RS-232 interface. The microcontroller is equipped with a full duplex UART or serial port, which enables it receive and send serial data [12]. These command signals are byte-wise data and contain bits which indicate whether a particular device is to be switched off or on. A logical 1 turns a device on while a logical 0 turns the device off.

The components used in switching the devices off and on include electronic relays and triacs. The relay contacts are opened and closed by using a transistor to supply current to the coil of the relay. The transistor is switched on and off by the microcontroller, and the state of the transistor depends on the control bit sent to the microcontroller from the server machine.



The triacs are driven by an opto-triac. When there is a logical low output from the microcontroller's pin. The LED will be able to sink enough current that is capable of putting it on. When the LED emits light, the diac turns on and conducts enough current to trigger the triac, which subsequently allows the flow of current to the connected load. On the event of a logic high appearing on the microcontroller's pin, the LED turns off. This puts off the diac and subsequently the triac. When the triac is off, no current flows through the load and thus it is turned off.

### **2.4.3 Software Implementation**

The software written totally in the Java programming language takes advantage of Java's Remote Method Invocation (RMI) API (application programming interface) for its client/server architecture implementation.

The Java Remote Method Invocation (RMI) system allows an object running in one Java virtual machine to invoke methods on an object running in another Java virtual machine. RMI provides for remote communication between programs written in the Java programming language [13]. RMI applications often comprise two separate programs, a server and a client. A typical server program creates some remote objects, makes references to these objects accessible, and waits for clients to invoke methods on these objects. A typical client program obtains a remote reference to one or more remote objects on a server and then invokes methods on them. RMI provides the mechanism by which the server and the client communicate and pass information back and forth. Such an application is sometimes referred to as a distributed object application.

Distributed object applications need to do the following:

- **Locate remote objects:** applications can use various mechanisms to obtain references to remote objects. For example, an application can register its remote objects with RMI's simple naming facility, the RMI registry. Alternatively, an application can pass and return remote object references as part of other remote invocations [14].
- **Communicate with remote objects:** Details of communication between remote objects are handled by RMI. To the programmer, remote communication looks similar to regular Java method invocations [14].
- **Load class definitions for objects that are passed around:** because RMI enables objects to be passed back and forth, it provides mechanisms for loading an object's class definitions as well as for transmitting an object's data [14].

In this project, the RMI server is located on the server machine, and is used to create a remote object which is made accessible to clients. It also is used to configure and initialize the serial port in preparation for communication with the interfacing circuit. The RMI client is a Java applet located on top a HTML page, which is published with the aid of a web server. The client obtains the reference of the server object and call methods which send control signals to the serial port of the server computer.

## **CHAPTER THREE**

### **DESIGN AND IMPLEMENTATION**

The entire TCP/IP based switch control system is essentially split into two major subsystem blocks:

- i. A Hardware module comprising of the interfacing circuitry and control devices
- ii. A software module, which is a client/server software implementation, used as a user interface for controlling the connected devices.

The texts below expand on how the above mentioned modules were implemented.

#### **3.1 Hardware Implementation**

The hardware module of the system can further be split into a combination of the following subsystems:

- i. A +5V regulated power supply.
- ii. The microcontroller/software control block.
- iii. A 240v ac power switch for static and dynamic loads.

##### **3.1.1 +5V Regulated Power Supply**

This was effected by using a 12V 0.5A step down transformer wired to a full wave bridge rectifier as specified below.

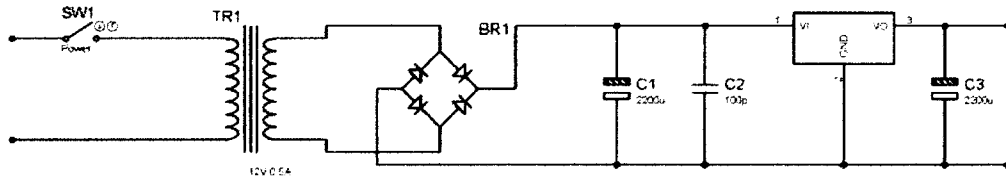


Fig 3.1 5v power supply

The 12V secondary voltage is rectified using four silicon diodes encapsulated in plastic. The 12V rms voltage is converted to:

$$V_{peak} = [(V_{rms}\sqrt{2}) - 1.4]v \dots\dots\dots 3.1$$

Where;

$V_{rms}$  = secondary rms voltage = 12V

$\sqrt{2}$  = rms to peak conversion factor.

1.4 = 2 diode forward voltage drop.

Therefore for this circuit;

$$V_{peak} = (12\sqrt{2}) - 1.4 = 15.5v \dots\dots\dots 3.2$$

The rectified dc voltage is smoothed by a 25v 2200µF capacitance in parallel with a 0.1µF capacitance. The value of the capacitance was chosen to give the minimum ac ripple on the rectified voltage.

Using the equation

$$cv = It \dots\dots\dots 3.3$$

$$c\Delta v = t\Delta I \dots\dots\dots 3.4$$

Where;

C = smoothing capacitance

$\Delta v$  = maximum allowable ac ripple

$$t = 1/f = 1/100s$$

$\Delta I$  = current change during circuit operation.

Neglecting  $t$ ,

$$c\Delta v = \Delta I$$

$$\Delta v = \Delta I/c$$

From above, we have that

$$\Delta v \propto 1/c$$

To keep  $\Delta v$  minimum,  $c$  has to be as high as possible. Thus, a 2200 $\mu$ F capacitor was chosen as the smoothing capacitor.

The rectified voltage is regulated to +5v by a 7805 regulator to produce a steady 5v supply that is used to run the various digital subsystems.

### 3.1.2 Microcontroller/Software Module

The assembly is run by an AT89C2051, which is an 8-bit microcontroller with 128 bytes of RAM, 15 I/O (input/output) lines on port 1 and port 3, and numerous on-chip peripherals, including two 16-bit timers, full duplex UART, etc [12].

The microcontroller was programmed with assembly language, using the 2051 instruction set, as this provided the fastest speed of instruction execution.

Running on a 12 MHz clock frequency, the controller is capable of executing 1 million instructions in 1 second. The flow of program execution is detailed below;

1. System reset
2. Initialize system variables and hardware
3. Clear command control byte.
4. Scan Keypad

5. Check Serial port for command from host computer
6. If no command is received go begin again from 4 above.
7. If keypad is pressed, get the variable to adjust i.e. load 1 on/off, load 2 on/off, system power on/off, etc.
8. Update variable and system status.
9. If command byte received on serial port, decode the byte and update system status.
10. Complement command byte and send to host computer as acknowledgement.
11. Loop back to 4 above.

User inputs are provided into the system kernel via external switches designated as shown below:

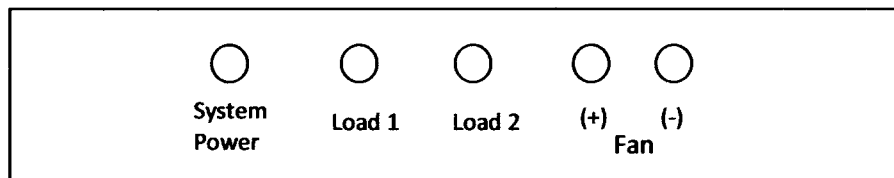


Fig 3.2 on board switch controls

For system power, load 1 and load 2, individual LEDs are provided to indicate their status; the associated LED glows to indicate whether system power, load 1 or load 2, is on.

Since load 1 and load 2 is either on or off, electromechanical relays have been provided to effect their switching. The system power is switched on or off by a power triac which is gated on by an optotriac. The dynamic load is controlled by pulse width modulation using an optotriac and a 15A power triac.

## System Initialization

Upon exiting the reset procedure, the controller starts instruction execution at hexadecimal address 0000h. For the software implementation, interrupts were not used, instead a polling system was implemented. Though polling wastes execution time, it was implemented based on the fact that the controller is only concerned with executing a small block of instructions. The software overhead associated with interrupt processing is therefore eliminated.

During initialization, all input pins are set to a logical 1 state (high), the relays are deactivated, the LEDs are turned off, the timer 1 is configured as an 8-bit auto reload timer, loaded with hexadecimal value 0FDh, producing a serial transfer rate of 2403 bps. This value is also used to set the baud rate of the serial port on the host computer system.

The slow communication speed (2403bps) does not in any way affect performance, since only byte-wise data packets are sent to or from the computer system.

## Command/Control Byte

The control byte is expounded upon as shown below, consisting of bit-wise and nibble-wise control signals.

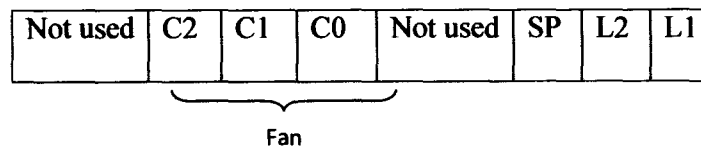


Fig 3.3 command byte used for control signal

Table 3.1 command byte usage

Bits	Usage
7	Not used
6	Speed Setting C2 (0/1)
5	Speed Setting C1 (0/1)
4	Speed Setting C0 (0/1)
3	Not used
2	System power on/off (0/1)
1	Load 2 on/off (0/1)
0	Load 1 on/off (0/1)

From above, setting bits 0 to 2 enable or disable the associated function. Bits 4 to 6 however, establish the voltage level of the regulated load (fan). On the host system software, the voltage (fan speed) was set to a maximum of 5. On the microcontroller, the fan speed was multiplied by two to control the ac circles via pwm (pulse width modulation) circuitry.

If bit 2 (system power) of the control byte is cleared to zero, the entire system is turned off, and can be turned on via the embedded keypad, or the user interface made available over the network.

If any button is pressed, say load 1 on/off control, bit 0 of the control byte is complemented (i.e. its logic level is inverted), and the system is updated. The same also applies to load 2 and the system power.



For the regulated load however, the control value is calculated by shifting bits 4 to 6 three places to the right, effectively multiplying it by two to produce a value in the range of 0 to 10 for controlling the ac cycles.

The pulse width modulation scheme is illustrated as shown below. If the speed setting is zero, the optotriac is turned off; if the speed setting is 2 ( $1 \times 2$ ), for every ten ac cycles, two ac cycles are passed through the triac and eight ac cycles are blocked, if the speed setting is 4 ( $2 \times 2$ ), 4 ac cycles are passed to the load and 6 cycles are blocked. The blocking and passing of ac cycles is done by an optotriac and a power triac subsystem.

### 3.1.3 Optotriac

An optotriac is a semiconductor device comprising an infrared LED encapsulated in a package in close proximity with a diac.

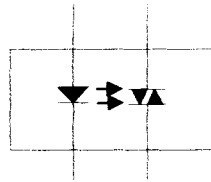


Fig 3.4 an opto-triac (moc3023)

The encapsulated diac is switched on when the light is enough to initiate conduction in the diac. Once triggered on, the diac conducts heavily, sourcing current into the gate of the connected power triac to energize the load.

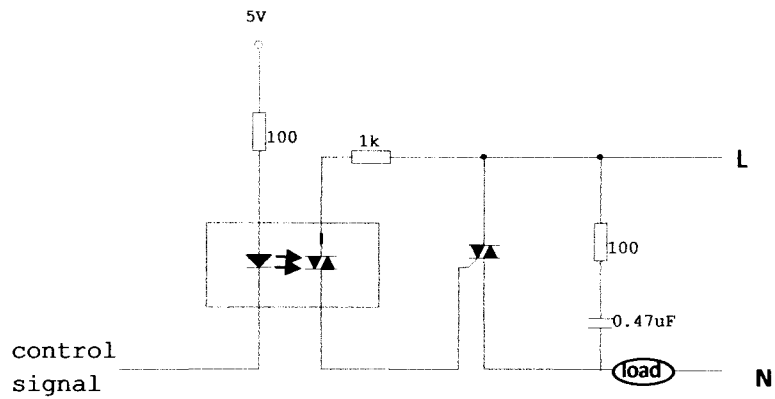


Fig 3.5 triac switched load

The optotriac is driven on/off by a dedicated port pin, P3.4 in the case of system power the on time of the diode is directly controlled by the duty cycle of the pulse width modulated waveform at the pin.

To switch the triac on, the port pin is taken low, set to a logic low, forward biasing the internal LED and turning on the diac, which consequently turns on the connected triac via its gate. The diac and hence the controlled triac, remains on until the pulse width modulation drive goes high to switch off the LED. The diac and triac both turn off when current through them falls below holding current, turning the connected load off.

The waveform for pulse width modulation is shown below.

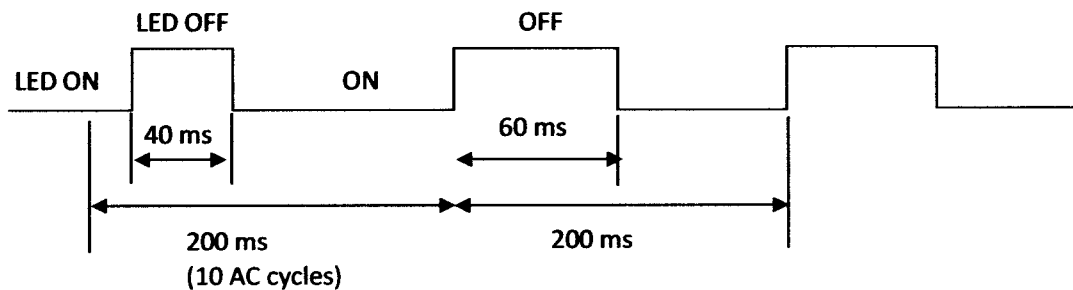


Fig 3.6 pulse modulated wave

Thus by varying the ratio of the on/off times of the pulse width modulation drive, various levels of average power can be developed for the connected load, in this case the fan's armature winding.

### 3.1.4 RS-232 Interface

The microcontroller's UART interfaces with the host system's serial interface via an RS-232 logic level converter circuit shown below.

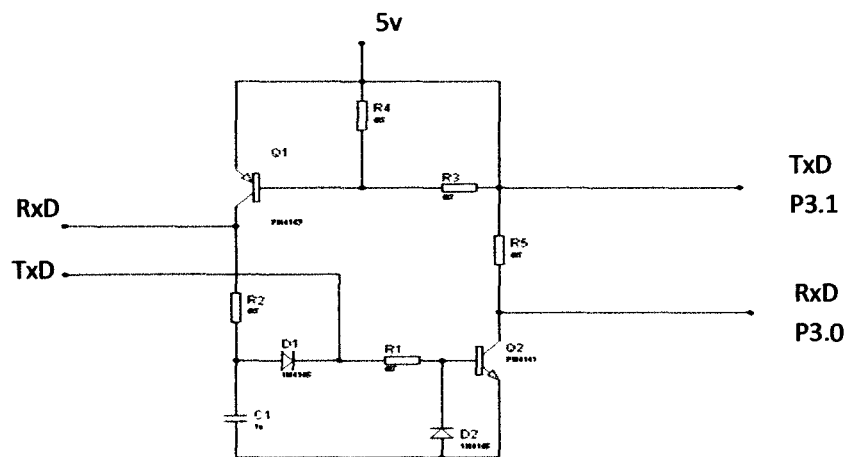


Fig 3.7 logic level converter for the RS-232

As the RS-232 interface uses non-standard logic levels for signaling, a TTL/CMOS to 232 converter has to be incorporated on the board to enable UART communication. Data from the microcontroller is level-shifted by Q1 and sent to the serial port on the host computer via Q1; data from the host computer system is received by the controller at pin P3.0 of the microcontroller after being level translated by Q2. Although there are standard ICs such as the MAX232, which can be used for logic level conversion between the RS-232 interface and other devices, there were not used based on low reliability of the available products.

### **3.1.5 System Status Reporting**

The device status can be retrieved from the microcontroller system by issuing hex data 0xFF. In response to this, the controller sends the current control byte. On the host computer, the byte is received, decoded and the appropriate status is updated.

If the hardware is completely offline (not phased to ac power but connected to the host system), the system based software times out and an appropriate status message is issued.

## **3.2 Software Implementation**

The software designed using the client/server design approach, using Remote Method Invocation (RMI) as its communications protocol. It consists essentially of two main parts, a client and a server. It is written in the Java programming language which was picked for its simplicity, portability and network oriented nature.

### **3.2.1 Application Server**

This can also be referred to as the RMI server module. It consists of a user interface which enables the user to select a serial port to use for communication with the interfacing circuit. It also has a start and stop button, to start and stop.

When a port is selected and the start button clicked, the server attempts to configure the serial interface, to meet communication speeds of the microcontroller circuitry. It also opens a network connection for clients to be able to connect to over a network. Remote methods are provided for clients to use for communication with the interfacing circuit.

This server continually runs and listens for client connections, receives system status and provides methods for clients to communicate with the microcontroller system. This is effectively the bridge between the client and the microcontroller.

### **3.2.2 Application Client**

The application client is a java applet, which can be placed on a HTML (hypertext markup language) page and made accessible to other people via the use of a web server. It provides the user interface required by the user, to control the intended devices, in this case two static loads and one dynamic load.

On initialization, the client tries to get a reference to the server. After the reference has been gotten, it uses this reference to communicate with the microcontroller circuitry by calling methods made accessible to it by the server.

It periodically checks if the system status has changed and updates a status page indicating whether, the various devices are on or off.

It is good to note that the client will only work properly on a computer that has a Java Virtual Machine (JVM), which is the runtime environment for Java programs.

### **3.2.3 Apache HTTP Server**

The apache HTTP server is a web server and it is used to serve dynamic as well as static content over the internet. This content is usually in the form of web pages (static or dynamic HTML pages).

The web server is installed on and run from the server system and is used to serve the page containing the client to any computer connected to the same network as the server system.

### 3.3 Design Calculations

#### 3.3.1 Reset Current Selection

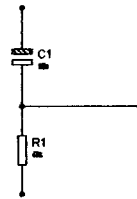


Fig 3.8 RC combination for reset

The RC combination must be capable of generating a frequency of at least 2 clock cycles. At 12MHz, a clock cycle is 1μs. Thus, the RC timing components must provide a valid logic 1 level for at least 2μs.

Using  $T = R \times C$  .....3.5

$$2 \times 10^6 = R \times C$$

R was chosen to be 47KΩ, which gives the value of C to be;

$$C = \frac{2 \times 10^6}{47 \times 10^3} = 42nF$$
 .....3.6

From the above calculation, the value of C must be greater than or equal to 42nF. A value of 10μF was selected, giving a delayed reset signal of;

$$T = 47 \times 10^3 \times 10 \times 10^{-6}$$

$$T = 0.47s$$

### 3.3.2 Current Limiting Resistance

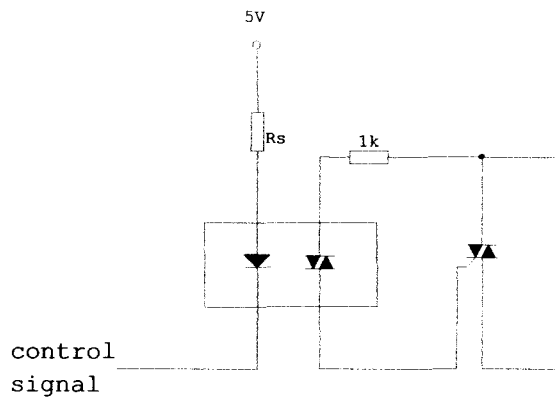


Fig 3.9 opto-triac arrangement

The maximum forward current for the integrated LED is 30mA at a forward voltage of 1.7V (gotten from the manufacturer's data sheet). Solving for the value of  $R_s$  yields.

$$R_s = \frac{V_s - V_{led}}{I_{led}} \dots\dots\dots 3.7$$

$$R_s = \frac{5 - 1.7}{0.03} = 110\Omega$$

The closest standard value to  $110\Omega$  is  $100\Omega$ , so it was chosen as the current limiting resistance value.

### 3.3.3 Relay Selection

The relays were chosen with contact current ratings of 6A, as this is the current drain of most household appliances.

### 3.3.4 Power Triac Selection

The power triac BT139 was chosen as the main AC power switching element as it can handle a continuous load current of up to 16A.

### 3.3.5 Transistor Relay Driver

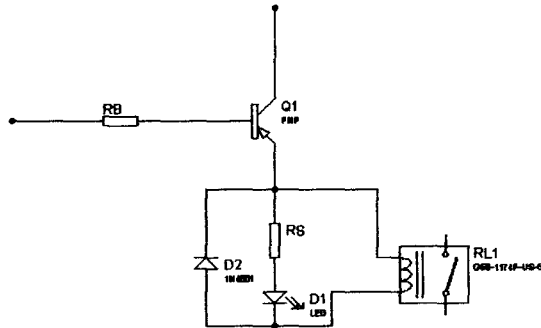


Fig 3.10 relay driver

Relay voltage = 5V

Relay coil resistance = 430 $\Omega$

Relay coil current =  $\frac{5}{430} = 12\text{mA}$

The 25A10159R PNP transistor has a typical gain of 200. For a relay current of 12mA;

$$I_B = \frac{12\text{mA}}{200} = 60\mu\text{A}$$





## **CHAPTER FOUR**

### **TEST, RESULTS AND DISCUSSION**

The design been completed, its workability needs to be tested. This is confirmed or done by subjecting the design to a series of test procedures. This procedures help in the test of each of the units and the results tabulated. This chapter gives the various test procedures and their corresponding results.

#### **4.1 Testing Procedures**

Individual blocks are tested in order to aid in debugging the system. This is done before the various modules are put together.

##### **4.1.1 Voltage Regulator**

Testing the functionality of the voltage regulator is an important step. This will help to avoid damage to the equipment by feeding in too great a voltage. The test will be done by inputting the full voltage that is expected by the regulator, including voltage spikes. We will measure the output voltage with a multi-meter to ensure that it will be accurate and stable when the voltage regulator is plugged into our detection system. We will also measure the current output of the voltage regulator.

##### **4.1.2 Microcontroller**

For the microcontroller block, it will be connected to the serial port of a computer through a logic level converter. This is so that command signals can be sent from the computer to the microcontroller. A multi-meter is used to measure the various output pins, which would be connected to the switches, and checked to see if the required outputs correspond to the input command sent from the computer.

### 4.1.3 Electronic Switches

Both the solid state switches and electromechanical relay arrangements are tested to ascertain their behavior when a control signal is sent from the microcontroller. Their input signal (control signal) will be a 5v dc connected to the individual driver components and an ac voltage source will be connected to the power switching devices. When the required control signal is passed to the circuit drivers to close the switches, the outputs are measured to see if the input voltage and current are passed through the closed switch. This is repeated for a control signal to open the switch.

### 4.1.4 Software

To properly test the software module, a two computer network was configured, and the server software set up on one machine while the other machine will be used to access the client. An oscilloscope is connected to pin 3 of the serial port of the server computer, to check that the output pulses are equivalent to the command signals expected by the microcontroller circuit.

## 4.2 RESULTS

The output voltage of the voltage regulator (7805 IC) was measured and found to be 5v.

Table 4.1 Test Results for Voltage Regulator

Input Voltage	Output Voltage	Input Current	Output Current
11v	5v	0.5A	0.1A

The AT89C2051 microcontroller responded as expected when certain bytes were written from the serial port. Below is the tabulated result of the test.

Table 4.2 Results of microcontroller test

Serial Input (hex)	P3.2	P3.3	P3.4	P3.5	Meaning
0x04	1	1	0	1	System Power on
0x07	0	0	0	1	System power and load 1 and 2 on

Both the triac and relay switches responded well at the passing of the appropriate control signal, closing and opening the circuit as required.

The software was tested across a network and on operation the proper wave forms were seen on the oscilloscope screen. Below is a Sketch of the waveform when 0x04 was sent to the serial port by clicking the power on button.

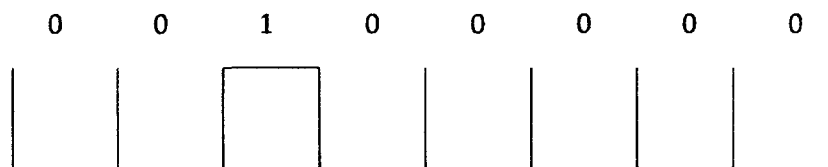


Fig 4.1 sketch of oscilloscope waveform

### 4.3 DISCUSSIONS

During the tests it was noticed that the software always threw an error, while trying to update status. It was found out that, this was due to the fact that the most significant bit (MSB) of the control byte was zero, and was usually truncated by the software. This was rectified by making the bit permanently 1.

## **CHAPTER FIVE**

### **CONCLUSION AND RECOMMENDATIONS**

#### **5.1 CONCLUSION**

The device constructed achieved its set objectives as it pertains to cost and functionality. However, it should be stated here that the device this project was aimed at fabricating was a prototype, a replica of the actual thing. It is economically viable to undertake certain projects this way since testing would not cost so much.

Any desire to implement an actual computer based control system would require a more flexible design of the interfacing circuit and the software interface, to enable the addition and removal of appliances without having to modify either one of the modules.

#### **5.2 RECOMMENDATIONS FOR FURTHER WORK**

The distributed computer controlled switch system should be improved upon to meet certain short comings like the inability to ascertain the working status of the controlled appliance. And also the inability to add extra appliances with changing the circuit and modifying the software should be considered.

The software could also be improved upon and features like timing operations added to include automation so that continual human interaction is unnecessary.

## REFERENCES

- [1] Encarta Dictionaries Microsoft • Encarta • 2007. © 1993-2006 Microsoft Corporation
- [2] Wajid Ali, "Embedded Systems", Term Paper, College of Information Technology, Punjab University, 2004.
- [3] Stephen E. Derenzo, *Practical Interfacing in the Laboratory*, Cambridge University Press, 2003, pp 360-503.
- [4] [[http://en.wikipedia.org/wiki/Embedded\\_system](http://en.wikipedia.org/wiki/Embedded_system)]
- [5] Edward Hughes, *Electrical and Electronic Technology*, 8<sup>th</sup> Ed. Pearson Education Ltd, 2004, pp 541-551.
- [6] [<http://www.steve-w.dircon.co.uk/fleadh/mphil/history.htm>]
- [7] [[http://nobelprize.org/educational\\_games/physics/transistor/history.html](http://nobelprize.org/educational_games/physics/transistor/history.html)]
- [8] [[http://en.wikipedia.org/wiki/Silicon-controlled\\_rectifier](http://en.wikipedia.org/wiki/Silicon-controlled_rectifier)]
- [9] Microsoft • Encarta • 2007. © 1993-2006 Microsoft Corporation
- [10] [<http://www.computer-museum.ru/english/index.htm>]
- [11] Chris Tackle and Lawrence Ricci, *Benchmarking Real-time Determinism in Microsoft Windows CE*, MSDN Library, 2002.
- [12] Atmel Corporation, *2051 Data Sheet*, 2005
- [13] Monica Pawlan, *Essentials of the Java Programming Language: A Hands-On Guide, Part 1*, Sun Microsystems inc, 2000.
- [14] Ann Wollrath and Jim Waldo, "Remote Method Invocation", *The Java Tutorial Continued*, Sun Microsystems Inc. 2006.