DESIGN AND CONSTRUCTION OF A PC BASED HOME AUTOMATION SYSTEM

BY

KPAM J. SOKPO

(98/7244EE)

DEPARTMENT OF ELECTRICAL/COMPUTER ENGINEERING

SCHOOL OF ENGINEERING AND ENGINEERING TECHNOLOGY

FEDERAL UNIVERSITY OF TECHNOLOGY,

MINNA, NIGER STATE,

NIGERIA.

NOVEMBER 2004

DESIGN AND CONSTRUCTION

OF A

PC BASED HOME AUTOMATION SYSTEM

BY

SOKPO, KPAM JIMIN

98/7244EE

A PROJECT REPORT SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING, SCHOOL OF ENGINEERING AND ENGINEERING TECHNOLOGY, FEDERAL UNIVERSITY OF TECHNOLOGY, MINNA

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF BACHELOR OF ENGINEERING DEGREE (B.ENG) IN ELECTRICAL AND COMPUTER ENGINEERING.

NOVEMBER 2004

DECLARATION

I, **Kpam J. Sokpo** hereby declare that this project presented here as partial fulfillment for the award of a Bachelors in Electrical/Computer Engineering Degree has not been hitherto presented anywhere to the best of my knowledge.

--	08/12/2004 ------------------------------
Signature	Date

CERTIFICATION

This is to certify that this project work was carried out and submitted by Mr. Kpam J. Sokpo (98/7244EE) of the Department of Electrical/Computer Engineering, School of Engineering and Engineering Technology, Federal University of Technology, Minna.

ENGR. T. ASULA
(Supervisor)

14. 2004 Dec

Signature and Date

ENGR. M.D. ABDULLAHI
(Head of Department)

31/01/2005

Signature and Date

DR. T.O. AKINBULIRE
(External Examiner)

Signature and Date

DEDICATION

My Life, all I am and all I will ever be, I humbly dedicate to Him who is worthy of my all- ***Jesus Christ.***

ACKNOWLEDGEMENT

This project would not have been possible without the input of the wonderful people listed below. Each of them has contributed to its success in an immense and immeasurable way, be it academic or personal, and for that reason, I owe them a debt of gratitude.

I would like to thank my Heavenly Father and my Saviour for the new life He has given me in Christ. I am so glad that I have a hope in Him!

To my mum, Barrister (Mrs.) L.M. Sokpo, I feel merely saying 'thank you' is grossly inadequate to express what I feel. The love and support she has given me is beyond measure. Without her I would not have lasted the first year of engineering, let alone, made it to this point. I want to boldly say it, 'You are the greatest mum in the world and I love you very much.'

To my very resourceful supervisor, Engr. T. Asula, I'm very grateful for his patience and gentle corrections during the course of this thesis.

Thank you, Mr. Gbile Akanni for such inspiration and strength as you give me. You have been a father to me indeed.

I would like to thank my Aunties, Barrister (Mrs.) Elizabeth Kohol and Professor (Mrs.) Charity Angya, and my uncle, Dr. Samuel Abiem, for their moral and financial support.

I am grateful to my siblings, Hembadoon, Shagba, Sema, Mbawuese and Sesugh for their support and patience. I love you all very much.

I would also like to mention my 'Daddy' and 'Mummy', Dr. and Mrs. Michael Onimole. They have been a great source of support.

To Mr. Tope Adeniyi and Mr. Tony Ayabam, I want to say a big 'thank you and God bless you' for all the times you inspired me and made me think constructively.

I must not leave out my wonderful friends, Alozie, Afam jr., Akintola, Ibrahim, Bisi, Uko, Chinelo, Ichehor, Nancy, Bukky, Phemmy, Tosin, Joey, Batur, Benjamin, Kayode, Dimeji, Fred, Ikenna, Solomon, Mike, Lifidi, e.t.c… you all know who you are… Your support and laughter has kept me sane.

ABSTRACT

The PC Based Home Automation System presented in this project is an integration of 3 sub-systems:- (i) an automatic doorbell/visitor alert system, (ii) an automatic timed lighting system and (iii) a safe-room infrared motion detector alarm system all connected to the parallel port of a Personal Computer (PC).

Sub-system 1: Input is taken from a momentary pressure switch which is placed under the doormat of the model house into status line **S3** of the parallel port and via software control using Visual Basic programming, a doorbell/visitor alert and a porch light connected to output line **DO** is activated.

Sub-system 2: The system is programmed to automatically switch the perimeter/security lights ON and OFF at pre-determined times of the day. This is done using the timer function in Visual Basic to compare the set times with the system clock and then a 1 or 0 is sent to **D1** to either switch it ON and OFF.

Sub-system 3: An Infrared emitter is connected to output line **D3** while a photodiode is connected to input line, **S4.** When the IR alarm is activated, a high bit (1) is sent to **D3**, activating the IR emitter. The emitter in turn energizes the photodiode. The photodiode sends a 0 or 1 back into the parallel port, depending on whether the alarm has been software activated or not. When the system is breached due to detected motion within the region of IR emitter and receiver coupling, the software program activates output line **D2** to sound an alarm.

Table of contents

CHAPTER ONE

INTRODUCTION

1.1 GENERAL INTRODUCTION

Automation, as an aspect of engineering is the system of manufacturing engineering design that extends the capacity of machines to perform certain tasks formerly done by humans, and to control sequences of operations without human intervention. The term automation has also been used to describe non-manufacturing systems in which programmed or automatic devices can operate independently or nearly independently of human involvement. In the fields of communication, aviation, astronautics, for example, such devices as automatic telephone switching equipment, automatic pilots and automated guidance and control systems are used to perform various operations much faster and better than could be accomplished by human beings.

Robotics and Automatic Control of devices and systems, though a more recent area of research and development, is becoming one of the very high points of man's technological development. With so much attention being given to this field of engineering, it is not surprising that most factories, medical facilities, satellite and space probe systems, intelligent buildings et cetera, are becoming fully automated and computerized.

1.1.1 WHY AUTOMATION?

The obvious question now is, why automation? What would it achieve? Automation of certain systems is carried out for the purpose of achieving greater efficiency and ease of operation. An automated process eliminates most of the running costs associated with labour the process would have otherwise incurred. It also eliminates most of the problems that arise due to errors in judgment and fluctuating moods and emotions which humans are often so prone to, thereby giving higher efficiency and security of operation.

The applications of automation and robotics are innumerable and still growing by the day. But for the purpose of this project, the focus will be on its application in an 'intelligent house'. Certain activities in a house (e.g. switching of devices ON and OFF, opening and unlocking of doors and gates, security, lighting control, et cetera) do not necessarily need any human involvement. The reason being that once certain conditions are met, the process could automatically be carried out without anyone being present. This offers greater security of operation and hence, higher efficiency. Why employ the services of a retinue of security guards whom you would need to be paying salaries periodically when you can simply install a system that may probably be more efficient in service delivery and yet would incur minimal maintenance costs? Or why put your entire house at the risk of being gutted down by a fire or being broken into by burglars when you can have a fire and intruder detection system that would automatically dial the fire service or police help-lines and play a pre-recorded panic message, thereby giving you a good head start in the solution of your problem? A very simple and inexpensive computer interfaced system can achieve all these and much more.

This project will focus mainly on the use of a personal computer (PC) via its parallel port to demonstrate how a building, especially a house could be automated. The possibilities of such a system are almost limitless. Photosensors could be used to adjust the lighting in a room to take advantage of daylight. When tied to a home automation system, heating, cooling, and ventilation systems can be adjusted to account for passive solar heat gains. Systems connected to a home automation system can also be turned on by telephone, so the home is comfortable when the owner arrives. The block diagram of the system is illustrated below:-

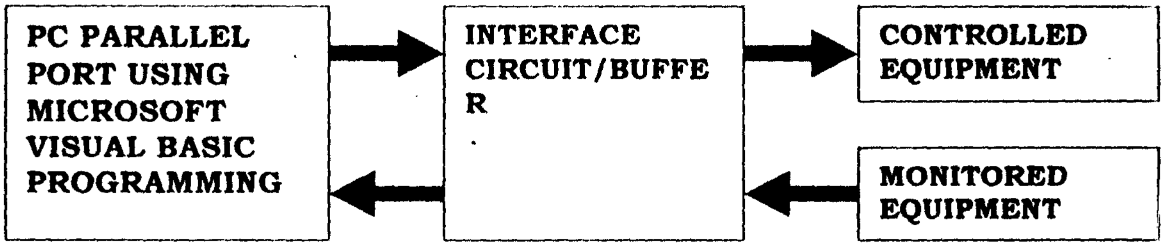

PC PARALLEL PORT USING MICROSOFT VISUAL BASIC PROGRAMMING
INTERFACE CIRCUIT/BUFFE R
CONTROLLED EQUIPMENT
MONITORED EQUIPMENT

Fig. 1.1: A simple block diagram of the system

1.2 LITERATURE REVIEW

1.2.1 HOME AUTOMATION SYSTEMS

Since the mid-1980s, several major participants in the housing, large appliance, and electrical control industries have pursued the idea of an integrated home automation system. Ideally, such a system monitors and operates many different functions within the home. Lighting, heating, cooling, ventilation, appliances, entertainment, and security can all be operated automatically. The homeowner programs, controls, or monitors the house by a computer or even by telephone. These systems have many similarities with building automation systems used in commercial buildings. Systems for the residential market, however, require mass production and simple installation before they will begin to make major inroads to the residential market. A minimal amount of custom engineering and installation is desirable. In many cases, automatic control is economically justified for only a few functions.

1.2.2 TYPES OF SYSTEMS

There are three types of home automation controls: individually controlled devices, distributed-control systems, and centrally controlled systems.

Individual devices control only one appliance or function. Examples include programmable setback thermostats, motion detectors, occupancy sensors, photocell lighting controls, and timers. Individual control devices have a wide variety of successful applications. These range from outdoor lighting to security sensors. The familiar television remote control unit often falls into this category. Remote control devices are not truly home automation devices, however, since they require conscious thought and human effort (however small) to operate.

A distributed control system uses standard power line wiring, telephone wire (4 pair), video wire (dual coaxial), radio frequency (RF) signals, and infrared (IR) signals. This makes it easier to retrofit the system into existing homes.

Microchip controls (actuators, interfaces, and sensors) must be installed in appliances or outlets. The system allows individual appliances to communicate with each other over the existing electrical wiring without a central controller (although keyboard entry is possible using telephones or personal computers). Users can use a television set to monitor the system's status. These form a local area network or home LAN. Compatible appliances are necessary, but currently it is up to the individual manufacturers to decide how to make them compatible. The major development in this direction has come from the Electronic Industries Association (EIA). The Association developed a standard communications protocol (CEBus) which will allow appliances and modems from different manufacturers to communicate with each other. Individual semiconductor manufacturers have developed microchips that could be installed in appliances.

A centrally controlled communication system routes signals between a central computer and appliance controllers or environmental sensors. The main advantage of this system is that it can control some "dumb" appliances as well as "smart" appliances. If the controller fails, however, the whole system fails.

The major distinction in "smart" home technology is the way electricity is distributed throughout the home. A central control system allots incoming household electricity to a distribution unit in each room of the house. The distribution unit (or network box) does not provide power to the room's outlets indiscriminately, as in a conventional home. The new outlets contain microprocessor chips that only provide power upon request by a "smart" appliance. "Smart" appliances have microprocessor chips that enable them to "communicate" their identity, power demands, and functional status to the network box when the appliance is plugged in. If the computer system determines that all is well, the network box sends power to that outlet. If the network senses potential danger, such as a frayed cord, or appliance incompatibility, the system denies power to the outlet. An outlet is only "live" when utilized by a compatible appliance.

These systems perform similar functions with natural gas appliances. Gas outlets can be placed strategically throughout the house, using easy-to-install semi-rigid stainless steel piping. Any appliance requiring gas, like a stove, outdoor grill, or clothes' dryer, could be "plugged in," accessing gas from the system as it requires it. The "smart gas" outlets constantly monitor for leaks, improper connections, and other malfunctions, and shut off gas to the outlet when the situation is unsafe. The most attractive feature of this system is that the interactive household system would automatically detect smoke and shut off the flow of gas from the main valve. This prevents gas-fueled house fires or explosions.

The idea of a "smart house" package for new homes was conceived by the National Association of Home Builders (NAHB) in 1984. An offshoot of that project called the Smart House Limited Partnership has continued research and development. They currently sell a complete wiring system for new smart house construction. Although they have developed and marketed a number of complete intelligent systems, with a variety of options, complete systems can be very expensive.

The idea behind this work is to design a system that is affordable and yet offers a good deal of control. This design is an application of the centrally controlled

Adeyemi, Habeeb Dayo of the Department of Electrical & Computer Engineering, F.U.T. Minna, did a similar work under the supervision of Engineer Emmanuel Eronu in September of 2003. His work, which was titled, "Design and Construction of Security Monitoring System Computer Interface (SMSCI)" was also based on the use of a PC parallel port interfaced to an infrared motion detector to monitor the security of a building. In his design, the Microcomputer monitors the doors, windows and roof of a model building using infrared proximity detectors which are connected to the PC via the status line of the printer port.

The work presented here is an upgrade on his work. This project attempts to make better use of the capabilities of the parallel port by integrating both the Data and Status lines into the design.

1.3 AIMS AND OBJECTIVES

The modeled automated house we will be considering in this project will have the following characteristics: -

❖ Lights will be switched ON and OFF at pre-determined times of the day.

❖ Given certain conditions, for instance, when there is unauthorized intrusion, the system generates an alarm.

❖ An automatic visitor alert (doorbell) is used to alert the occupants of the house of the presence of any person at the front door.

❖ All the above named processes will be carried out automatically with minimal human involvement.

1.4 PROJECT OVERVIEW (CHAPTER REVIEW)

Now that the problem and its solution have been introduced, past work has been reviewed and the aims of the project have been clearly stated, the remaining sections will cover in depth, the research, technology and reasoning processes that were made in designing the automated house.

The theory of operation behind Parallel Port Interfacing is covered in **Chapter 2**. This chapter contains a brief overview of all the theory that was researched during the design process. The solution is then derived and broken down into its key modules and subsystems. The reason for this is to give the reader a proper background on any technical terms and statements made in the course of the project. Each of the components used is carefully considered so as to understand their applications in this design.

The next chapter (**Chapter 3**) derives the required physical specifications, and the fundamental design decisions made in achieving the solution. The information from **chapters 1, 2 and 3** is compiled and processed and actuated. The construction of the physical system is looked into with some detail. After the hardware issues, the software implementation is also considered. Attempt is made to explain the effect of hardware on the software solution and visa versa. The chapter concludes with a development process of how the code for the control of the parallel port interface circuit was written.

Chapter 4 includes the results and performance evaluation of the automated house prototype. This chapter looks at how well the proposed solution fits the specifications given in the previous sections and conclusions are drawn from these results. The chapter also covers the lessons learnt by the author, and how these lessons can be used in future engineering arenas.

Included also in this chapter are the proposed recommendations for future developments and the conclusion to this project. Recommendations are made on how this product could be improved to better serve the needs of the market and how this product would be sold if it was put into production. The chapter concludes with an overview of the entire project and its achievements.

CHAPTER TWO

SYSTEM DESIGN AND ANALYSIS

This chapter derives the required physical specifications, and the fundamental design decisions made in designing the-PC based home automation system.

2.1 THEORY OF OPERATION

The application of the Microcomputer (PC) in the control of external devices lay at the heart of this project. The PC-based home automation system uses the Microcomputer's peripheral addressing system together with the capabilities of the parallel port. There are 3 separate sub-systems incorporated into this design.

Sub-system 1: A momentary strain switch is placed under the doormat of the model house. This switch is connected to **S3**, an input line on the parallel port. When the switch is activated by someone standing on it, a high bit (5Vdc) input is sent into the status line, **S3** of the parallel port. The system then interprets it and via software control, sends a high bit (5Vdc) to an output line of the parallel port, **D0**, to activate an alarm and switch on the porch lights to illuminate whosoever is standing at the door. This way, an alert is sent into the house alerting the occupants of the house of the presence of a visitor on the premises.

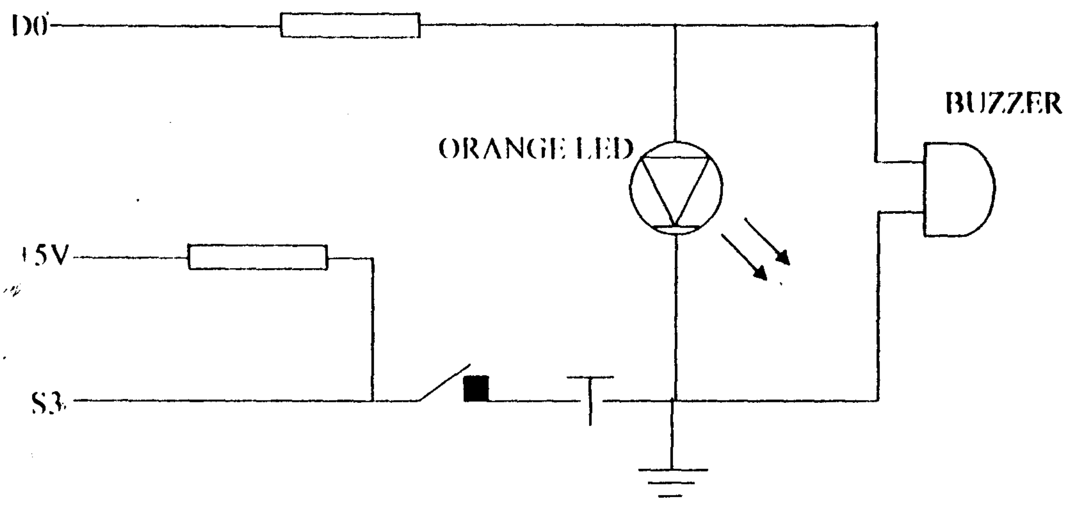


Fig. 2.1: Circuit Diagram of Automatic Door Bell System (Sub-system 1)

Sub-system 2: Using the timer function, two time values are set which the system compares with the system time to switch lights **ON** and **OFF**. One of the timer settings calls up the system to output a high bit (5Vdc) to the data line **(D1)** thereby switching **ON** the lights and the other time setting calls the system to send a low bit (5Vdc) to **D1**, thereby switching **OFF** the lights.

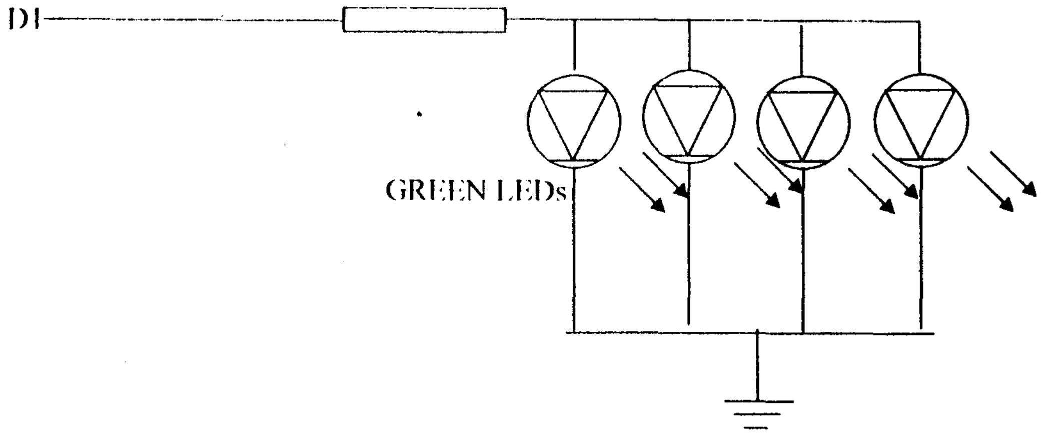


Fig. 2.2: Circuit Diagram of Automatic Lighting System (Sub-system 2)

Sub-system 3: The third sub-system uses an output line **(D3)** from the parallel port to activate an infrared emitter which then energizes the infrared transceiver. The output of the IR transceiver is connected to the input of the parallel port. Whenever the infrared circuit is activated by sending a high bit to output line 3 **(D3)** of the parallel port, the emitter conducts and begins to emit infrared light waves in the direction of the receiver. When the receiver conducts, it sends a high bit which the computer interprets and via programming sets another output line, **D2** low i.e. 0v. As soon as the input into the parallel port goes low as a result of motion detected within the operating region of the infrared motion detector, a high bit (which is electrically 5Vdc) is sent out through output line **D2** to activate a burglar alarm. The alarm does not stop until the RESET button on the graphic user interface provided on the system is clicked upon.

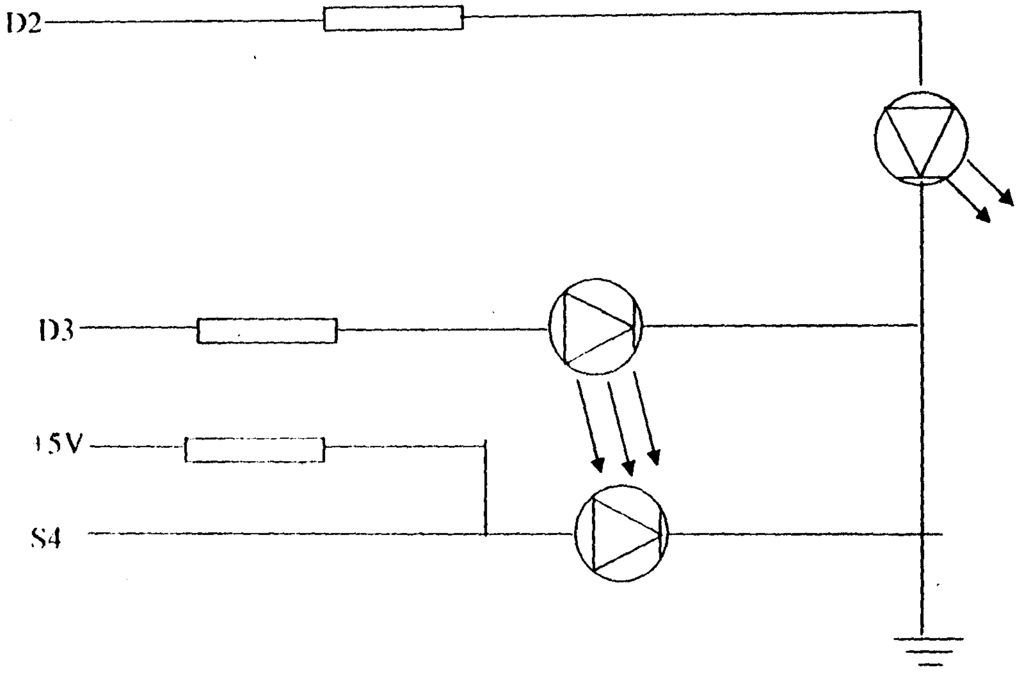


Fig 2.3: Circuit Diagram of Infrared Motion Detector (Sub-system 3)

2.2 THE MICROCOMPUTER

The Microcomputer serves as the main control unit of the entire system. Input from the outside world is processed by the microcomputer and compared with some preset conditions in the software programming. Depending on the conditions which have been met by the input signals, the PC responds via software control by sending out a specific output.

In its basic form, a modern PC system of today bears little difference from IBM's original design- a collection of components, both internal and external, interconnected by a series of electrical data highways over which data travels as it completes the processing cycle that transforms it from an item of input to an item of output. These "buses", as they are called, connect all the PC's internal components and external devices and peripherals to its Central Processing Unit (CPU) and main memory (RAM).

Communication with each of these components is made possible by a unique address assigned to each device by the microprocessor- an I/O port number. The PC has a built-in listing of all I/O units, each of which has their own port address- a "zip code" of some sorts.

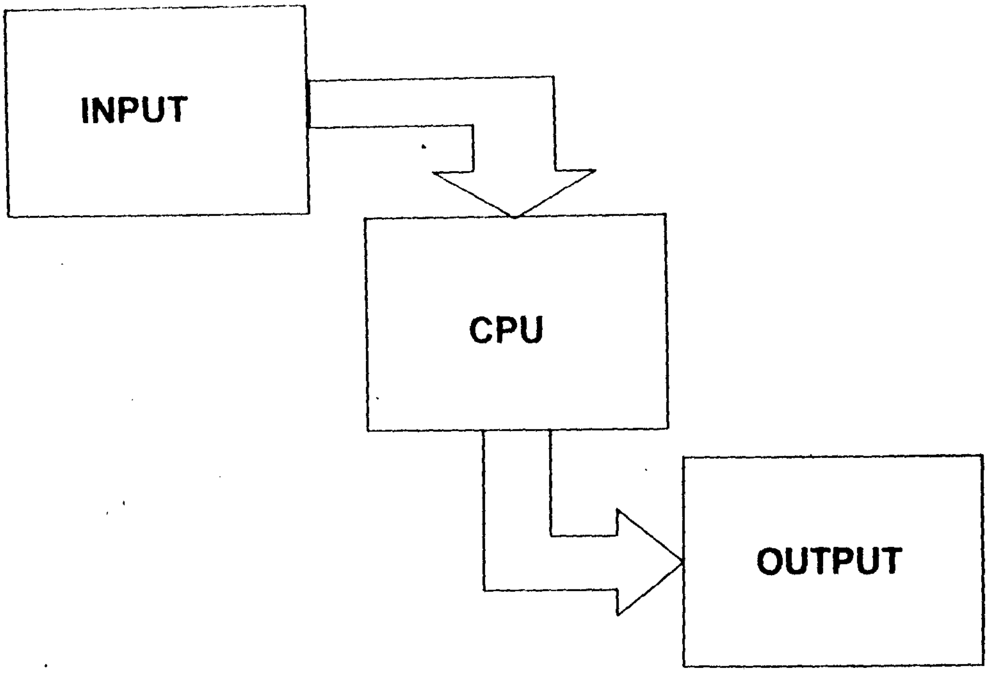


Fig. 2.4: A simple block diagram illustrating the Microcomputer's processing cycle

In this project, the outside world is interfaced with the Microcomputer via the Parallel Port. The Parallel Port has a 25-pin connector. Its pin configuration is given below in **Table 2.1.**

For the purposes of this project, only pins 2, 3, 4, 5, 13 and 15 were put to use. The remaining output lines were grounded while the input lines were left open.

2.3 PARALLEL PORTS

The Parallel Port is normally found on the back of nearly all IBM compatible PCs. It was specifically designed to attach printers with a parallel port interface, but it can be used as a general input/output port for any device or application that matches its input/output capabilities. It has 12 TTL-buffer output points, which are latched and can be written to and read from under program control using the Microprocessor's **In** or **Out** instruction. The adapter also has five steady-state input points that may be read using the processor's **In** instruction. In addition, one input can also be used to create a processor interrupt. This interrupt can be enabled and disabled under program control.

The input/output signals are made available at the back of the adapter through a right-angled, PCB-mounted, 25-pin, D-type female connector. This connector protrudes through the rear panel of the system, where a cable may be attached.

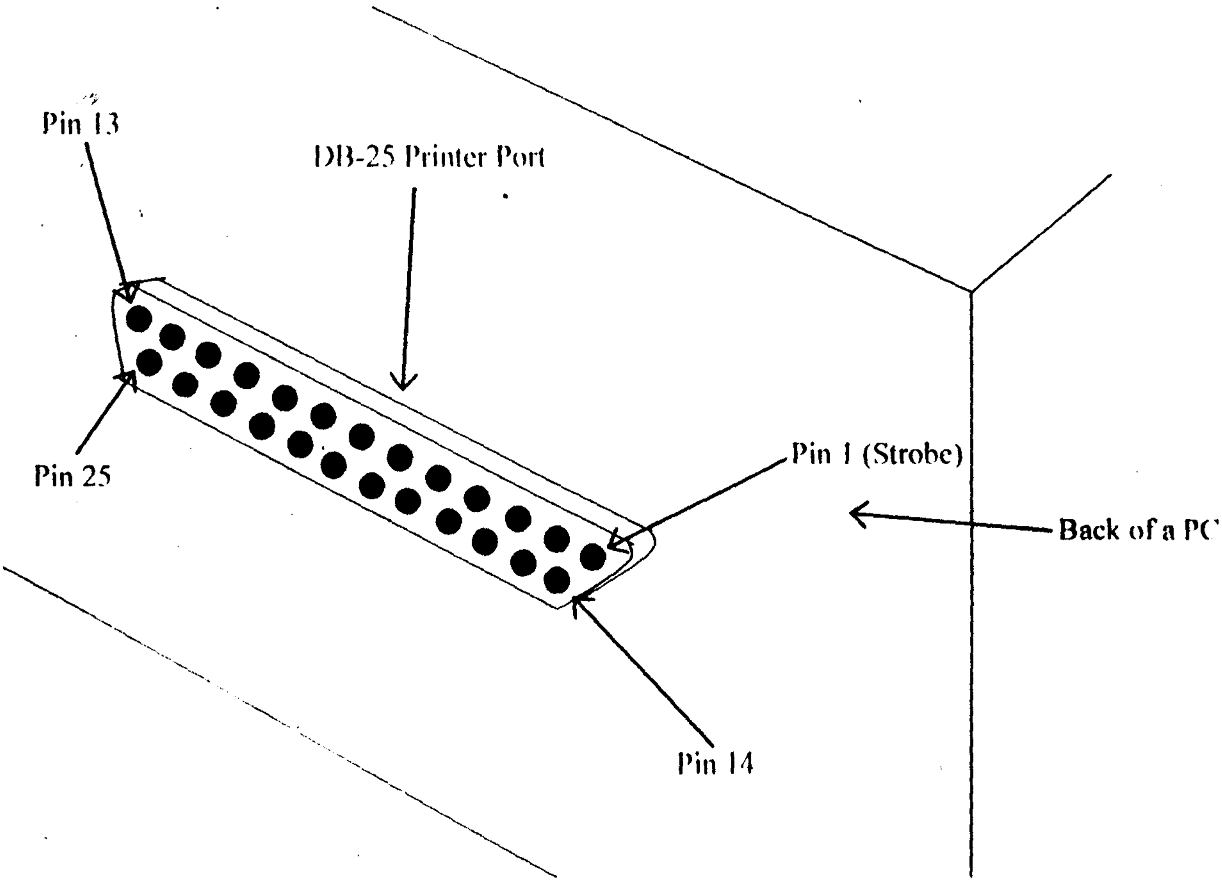


Fig. 2.5: Diagram of the D-25 port on the back of a PC

2.3.1 HARDWARE PROPERTIES

Shown below is a table of the "Pin Outs" of the D-Type 25 Pin connector and the Centronics 34 Pin connector. The D-Type 25 pin connector is the most common connector found on the Parallel Port of the computer, while the Centronics Connector is commonly found on printers.

Pin No (D-Type 25)	Pin No (Centronics)	SPP Signal	Direction In/out	Register	Hardware Inverted
1	1	nStrobe	In/Out	Control	Yes
2	2	Data 0	Out	Data	No
3	3	Data 1	Out	Data	No
4	4	Data 2	Out	Data	No
5	5	Data 3	Out	Data	No
6	6	Data 4	Out	Data	No
7	7	Data 5	Out	Data	No
8	8	Data 6	Out	Data	No
9	9	Data 7	Out	Data	No
10	10	nAcknowledge	In	Status	No
11	11	Busy	In	Status	Yes
12	12	Paper-Out / Paper-End	In	Status	No

13	13	Select	In	Status	No
14	14	nAuto-Linefeed	In/Out	Control	Yes
15	32	nError / nFault	In	Status	No
16	31	nInitialize	In/Out	Control	No
17	36	nSelect-Printer / nSelect-In	In/Out	Control	Yes
18 - 25	19-30	Ground	Gnd		No

Table 2.1: Pin Assignments of the D-Type 25 pin Parallel Port Connector

The above table uses "n" in front of the signal name to denote that the signal is active low e.g. nError. If the printer has experienced an error, then this line is low. This line normally is high when the printer is functioning correctly. The "Hardware Inverted" means the signal is inverted by the Parallel card's hardware. Such an example is the 'Busy' line. If +5v (Logic 1) was applied to this pin and the status register read, it would return back a 0 in Bit 7 of the Status Register.

The output of the Parallel Port is normally TTL logic levels.

2.3.2 Centronics

Centronics is an early standard for transferring data from a host to the printer. The majority of printers use this handshake. This handshake is normally implemented using a Standard Parallel Port under software control. Shown below is a simplified diagram of the Centronics' Protocol.

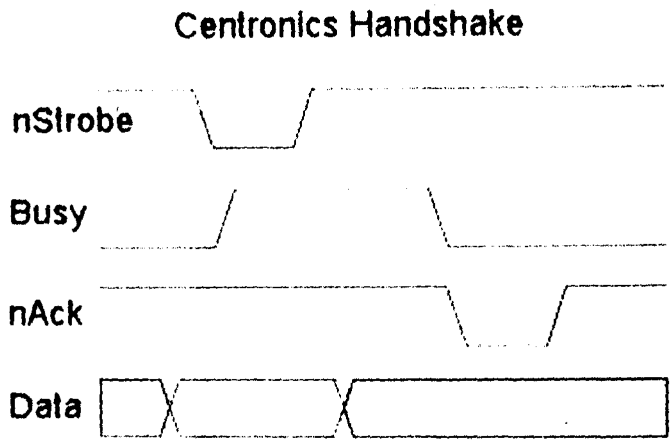


Fig. 2.6: Simplified Illustration of the Centronics' Handshake

Data is first applied on the Parallel Port pins 2 to 7. The host then checks to see if the printer is busy i.e. if the busy line is low. The program then asserts the strobe, waits a minimum of 1uS, and then de-asserts the strobe. Data is normally read by the printer/peripheral on the rising edge of the strobe. The printer will indicate that it is busy processing data via the Busy line. Once the printer has accepted data, it will acknowledge the byte by a negative pulse about 5µS on the nAcknowledge line.

2.3.3 Port Addresses

The Parallel Port has three commonly used base addresses: 3BCh, 378h and 278. **LPT1** is normally assigned base address 378h, while LPT2 is assigned 278h. 378h & 278h have always been commonly used for Parallel Ports (the lower case **h** denotes that it is a hexadecimal number). These addresses may vary from machine to machine.

When the computer is first turned on, **BIOS** (Basic Input/Output System) will determine the number of ports the PC has and will assign device labels **LPT1, LPT2** & **LPT3** to them. **BIOS** first looks at address 3BCh. If a Parallel Port is found here, it is assigned as **LPT1**, and then it searches at location 378h. If a Parallel card is found there, it is assigned the next free device label. This would be **LPT1** if a card wasn't found at 3BCh or **LPT2** if a card was found at 3BCh. The last port of call is 278h and follows the same procedure the other two ports. It is therefore possible to have a **LPT2** which is at 378h and not at the expected address 278h.

When **BIOS** assigns addresses to your printer devices, it stores the address at specific locations in memory, so we can find them.

Start Address in Memory	Function
0000:0408	LPT1's Base Address
0000:040A	LPT2's Base Address
0000:040C	LPT3's Base Address

Table 2.3: LPT Addresses in the BIOS Data Area

The above table, **Table 2.3**, shows the address at which we can find the Printer Port's addresses in the **BIOS** Data Area. Each address will take up 2 bytes.

2.3.4 Software Registers of the Parallel Port

Data Register

The base address, usually called the Data Port or Data Register is simply used for outputting data on the Parallel Port's data lines (Pins 2-9). This register is normally a write only port. If you read from the port, you should get the last byte sent. However if the printer port is bi-directional, you can receive data on this address. The Table below illustrates the configuration of the Data register/Port.

Offset	Name	Read/Write	Bit No.	Properties
Base + 0	Data Port	Write	Bit 7	Data 7
			Bit 6	Data 6
			Bit 5	Data 5
			Bit 4	Data 4
			Bit 3	Data 3
			Bit 2	Data 2
			Bit 1	Data 1
			Bit 0	Data 0

Table 2.4: Data Port

Status Register

The Status Port (base address +1) is a read only port. Any data written to this port will be ignored. The Status Port is made up of 5 input lines (Pins 10,11,12,13 & 15), an IRQ status register and two reserved bits. Bit 7 (Busy) is an active low input. E.g. If bit 7 happens to show a logic 0, this means that there is +5v at pin 11. This is the same with Bit 2 (nIRQ) of the status port. If this bit shows a '1', then an interrupt has not occurred. The Table Below illustrates the properties of the Status Register.

Offset	**Name**	**Read/Write**	**Bit No.**	**Properties**
Base + 1	Status Port	Read Only	Bit 7	Busy
			Bit 6	Ack
			Bit 5	Paper Out
			Bit 4	Select In
			Bit 3	Error
			Bit 2	IRQ (Not)
			Bit 1	Reserved
			Bit 0	Reserved

Table 2.5: Status Port

Control Register

The Control Port (base address +2) was intended as a write only port. When a printer is attached to the Parallel Port, four "controls" are used. These are Strobe, Auto Linefeed, Initialize and Select Printer, all of which are inverted except Initialize.

The printer would not send a signal to initialize the computer, nor would it tell the computer to use Auto linefeed. However these four outputs can also be used for inputs. These lines are often "open collector" outputs (or open drain for CMOS devices). This means that it has two states: a low state (0v) and a high impedance state (open circuit).

Bits 4 & 5 are internal controls. Bit four will enable the IRQ and Bit 5 will enable the bi-directional port meaning 8 bits can be inputted into the system using the output lines (**do-d7**). This mode is only possible if the printer port card supports bi-directional communication. Bits 6 & 7 are reserved. Any writes to these two bits will be ignored.

Offset	**Name**	**Read/Write**	**Bit No.**	**Properties**
Base + 2	Control Port	Read/Write	Bit 7	Unused
			Bit 6	Unused

			Bit 5	Enable Bi-Directional Port
			Bit 4	Enable IRQ Via Ack Line
			Bit 3	Select Printer
			Bit 2	Initialize Printer (Reset)
			Bit 1	Auto Linefeed
			Bit 0	Strobe

Table 2.6: Control Port

2.4 Parallel Port Interface Circuit

The Parallel Port Interface circuit was primarily designed to provide a link between the PC and the controlled equipment, and to also effectively protect the PC from damage. It consists of:

i. A DB-25 male connector

ii. 2 non-inverting hex buffers (74LS367)

iii. 3 nos. 6V Electromagnetic Relays

iv. 3 nos. IN4001 Diodes

v. 3 nos. 2N2222 Transistors

vi. 3 nos. 4.7 kΩ Resistors

vii. 3 nos. 470Ω Resistors

i) DB-25 Male Connector: This connector plugs into the DB-25 female connector located at the back of the PC. The pins of the male connector are numbered as shown below:

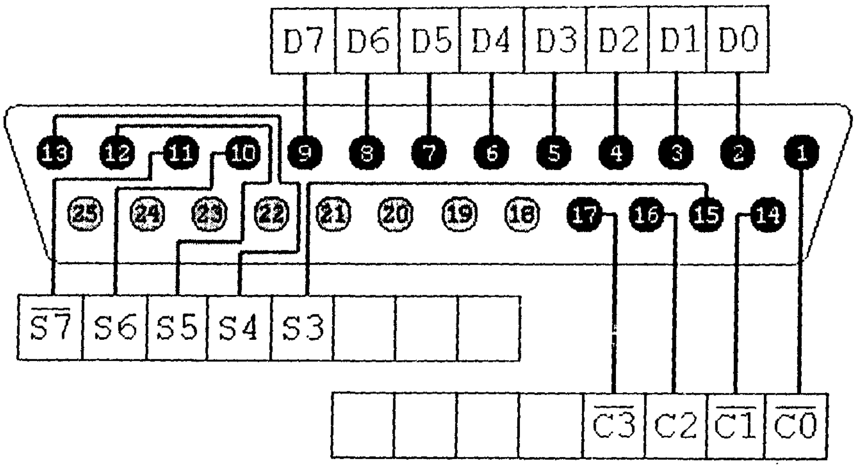


Fig. 2.7: Pin layout of the DB-25 male connector

i) Non-Inverting Hex Buffers: There are some hardware devices that always generate some output, such as registers. Many of these devices are often hooked to a bus, where only one device should write to at a time. Tri-state buffers allow us to control which device outputs its value to the bus, by allowing us to selectively control thc control bit of the tri-state buffer. These chips protect the motherboard from sourcing or sinking too much current. The Parallel Port can sink or source about 5mA of current. Most devices demand much more current than that and can thereby damage the motherboard. The 74LS367 serves as a protective buffer. If the connected device draws or sinks too much current, these chips would blow before surging the motherboard. They serve as a form of electrical 'fuses' in this application.

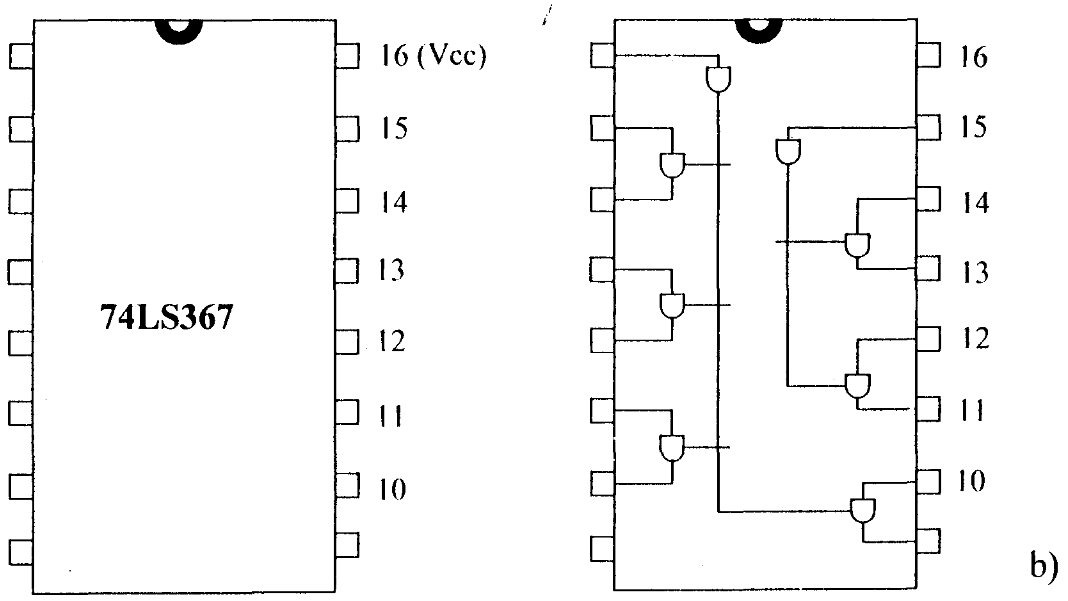


Fig. 2.8: Diagrams of a) Physical Pin Configuration and b) Tri-State Logic of the 74LS367

INPUTS		OUTPUT
A	$\overline{G}$	Y
LOW	LOW	LOW
HIGH	LOW	HIGH
X	HIGH	HI-Z

HIGH= High Logic Level

LOW= Low Logic Level

X= Either Low or High Logic

HI-Z= 3-State (Outputs are disabled)

Table 2.7: Truth table of the 74LS367 IC

ii) The Electromagnetic Relay: Electromagnetic Relays are electrically controlled switches. They consist of a multi-turn coil, wound on an iron core to form an electromagnet. When the coil is energized by passing current through it, the core becomes temporarily magnetized. The magnetized core attracts the iron armature. The armature is pivoted which allows it to operate one or more sets of contacts. When the coil is de-energized, the armature and contacts are released.

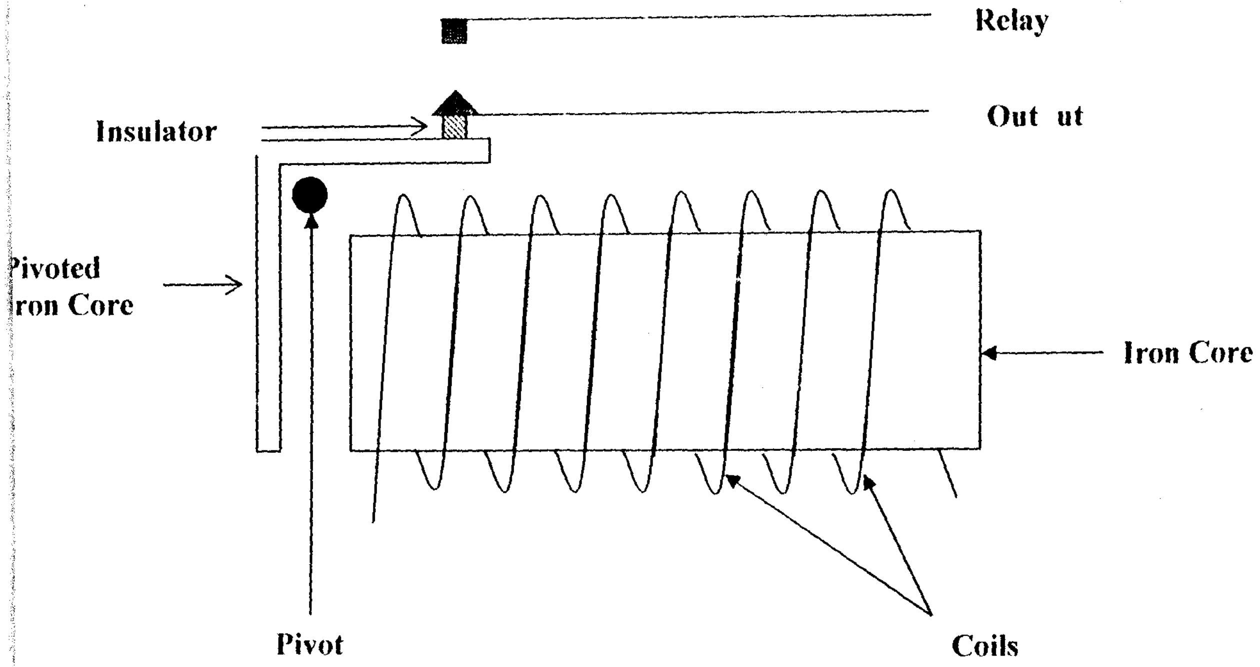
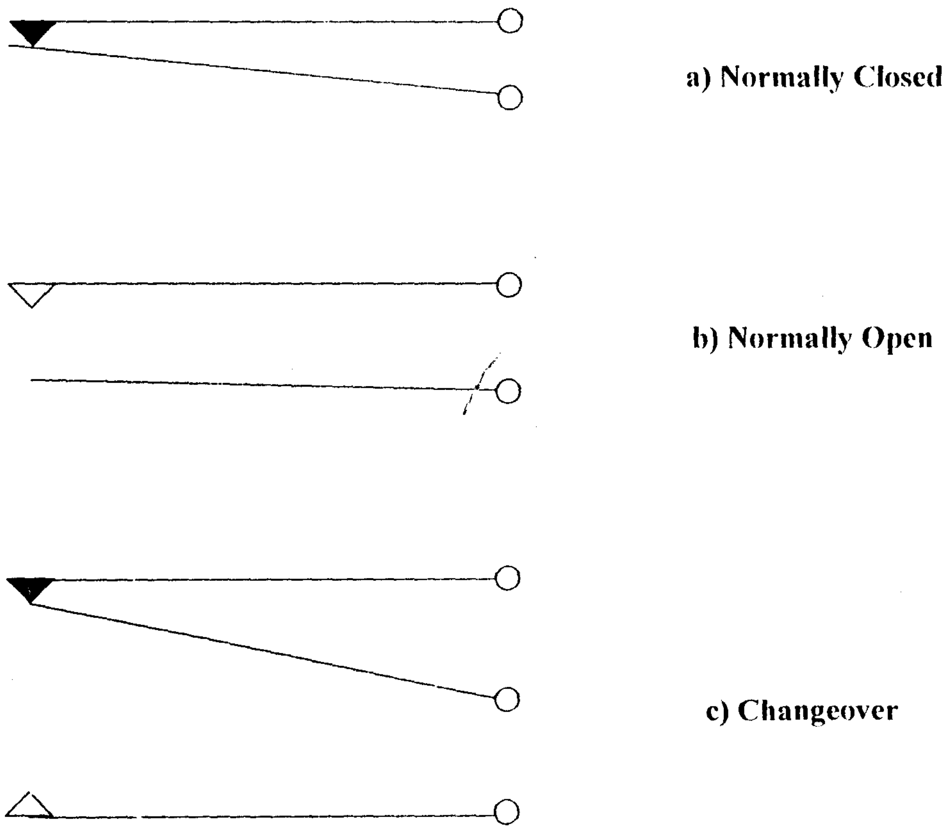


Fig. 2.9: Inner Structure of the Electromagnetic Relay

The coil can be energized from a low power source such as a transistor while the contacts can switch high powers such as the mains supply. Relays can generate a very high voltage across the coil when switched off. This can damage other components in the circuit. To prevent this, a diode is connected across the coil. The cathode of the diode is connected to the most positive end of the coil. The primary uses of relays are in remote switching and high-voltage (or high-current) switching. Because it is very important to keep electronic circuits electrically isolated from the ac power line, relays are used to switch ac while keeping the control signals electrically isolated.



a) Normally Closed

b) Normally Open

c) Changeover

Fig. 2.10: The different types of Relay Contacts

The spring sets (contacts) can be a mixture of N.O, N.C and C.O. The operating voltage of the relays used in this project is 6V.

iii) Diodes: These are two-terminal semiconductor (rectifying) devices that exhibit a non-linear current/voltage characteristic. The function of a diode is to allow current in one direction and to block current in the opposite direction. The terminals of a diode are called the anode and cathode. There are two kinds of semiconductor diodes: a P-N junction diode, which forms an electrical barrier at the interface between N- and P-type semiconductor layers, and a Schottky diode, whose barrier is formed between metal and semiconductor regions. They are formed either in germanium or silicon crystal.

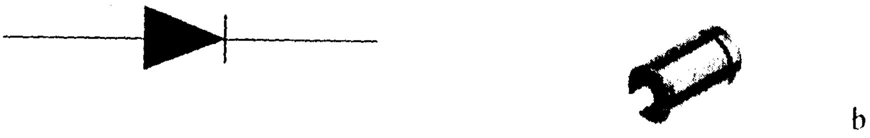


Fig 2.11: Diagrams of diode showing a) symbol and b) physical characteristics.

The arrowhead indicates the conventional direction of current when the device is forward-biased. They were used in the design of this system to protect the transistor and the PCs motherboard from back emf, which occurs when the relay is de-energized and its magnetic field collapses.

iv) Transistors

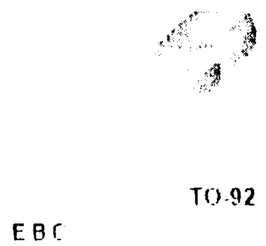


Fig. 2.12: Physical structure of the 2N2222 Transistor

One of the major applications of transistors is its use in switching a small control current so as to enable a much larger current to flow in another circuit. When the voltage applied to the base is 5V (i.e. when the logic signal is high), then

$$\mathbf{I_B = V_{in}/R_B} \qquad \text{(Given that RB=470}\Omega\text{ and } V_{in}\text{=5V)}$$

$$\mathbf{I_B=5/470=0.0106A=10.6mA}$$

This current is not sufficient to drive the relay. But on amplification to βI_B (where β, which is equivalent to the gain of the transistor, has a minimum value of 40), IC becomes:

$$\mathbf{I_C=\beta I_B=40x0.0106=0.4255A=425.5mA}$$

This current is sufficient to drive the relay and consequently, turns on the LEDs. As a switch, the transistor serves to isolate the input from the load.

2.4.1 Transistor Switching

Switching is examined here in the context of a bipolar junction transistor (2N2222) switching an inductive relay coil.

The 2N2222 transistor is an NPN bipolar junction transistor. It is made from the same N-Type and P-Type semiconductor materials used for diode fabrication and employs the same principle in its operation. However the transistor has two P-N junctions. It has an emitter made from N-type material and its base is a P-Type material for the transistor to operate, the emitter must be connected to negative, the base to the positive and the collector to the positive. Current flow between the emitter and base controls the current flow between emitter and collector. By regulating the current at the emitter-base junction, the amount of current allowed to pass from the emitter to the collector can be controlled.

iv) Resistors: Resistors with values of 1kΩ, 470Ω and 4.7kΩ were used as limiting resistors. The resistors are needed to limit the current taken from parallel port to a value that will acceptably light up normal LEDs and will still be safe (not overloading the parallel port chip). In a practical situation, after using the resistors, the output current will be few milli-amperes for the

LED, which will cause a typical LED to somewhat light up visibly, but not get the full brightness.

2.5 Input Devices

For input into the Parallel Port, a switch, an infrared emitter and an infrared transceiver were used. These devices were connected to the status lines of the Parallel Port.

i) The switch was used to provide an input into the status port to demonstrate what happens when a strain gauge under the doormat of a house is activated. It was fed with an external 5V from the regulated power supply. When the contacts of the switch are closed, a voltage level of 5V, which represents logic high, is sent into the parallel port. The computer program interprets this and then sends an output to **D0** lighting up the LED and ringing an alarm within the house. A second switch was used so that the automatic doorbell could be disabled whenever someone was going out the house. This second switch could be placed on the rim of the door.

ii) An infrared LED was used to activate the infrared security system. The infrared LED is connected to output line 4 (i.e. **D3**). A photodiode is connected to an input line (**S5**). Whenever the alarm is activated, the PC send out 5Vdc through pin 13 of the parallel port. This voltage energizes the

IR LED which in turn energizes the photodiode. The IR LED is placed about 4cm from the photodiode. Photodiodes are P-N junctions specifically designed to optimize their inherent photosensitivity. Photodiodes can be used in two ways -- in a photovoltaic (where it becomes a current source when illuminated), or photoconductive role. To use a photodiode in its photoconductive mode, the photodiode is reverse-biased; the photodiode will then allow a current to flow when it is illuminated. Note that phototransistors behave much like photodiodes, but with higher gain (i.e., a phototransistor allows more current to flow than would a photodiode in a photoconductive role).

CHAPTER THREE

Construction, Simulation and Results

3.1 Hardware Construction

Construction of the system was done using a Vero board and the various components were laid out according to the circuit diagram of each of the Sub-systems represented in **FIG. 2.1**, **2.2** and **2.3.**

3.1.1 CONSTRUCTION.

The initial stage of construction, which was the design and simulation, was done using an electronic computer aided design application known as MULTISIM Student Version developed by **Interactive Image Technologies Ltd. U.S.A. 1992 – 1999 ©**

The simulated components were then and soldered to a Vero board.

A Vero board is a board lined with parallel copper tracks. The circuit is wired as shown in the circuit from **FIG. 2.1**, **2.2** and **2.3.** The legs of each component were pushed through the holes on the top of the board and soldered to the bottom of the board where the copper tracks are lined up. To join the components according to the circuit, strings of wire (jumpers) were soldered to the components or to the tracks corresponding to where a component was initially soldered. Integrated circuit sockets were soldered to the board for the **74LS367** – Non-inverting Hex Tri-State Buffer chip. The sockets are 16 pin sockets. This enables easy troubleshooting and changing of the chips in the event of any of them failing.

3.1.2 SOLDERING TECHNIQUES EMPLOYED

The most fundamental skill to assemble any electronic project is soldering. The process of joining electrical parts together to form an electrical connection is called soldering. It involves using a mixture of molten lead and tin (solder) with a soldering iron (with a heating element inside of it).
A large range of soldering irons is available. The following are the characteristics that were watched out for when deciding on the type of soldering iron to be used.

Voltage: Most irons run from the mains at 240 V. However, low voltage types (e.g. 12V or 24V) generally form part of a "soldering station" and are designed to be used with a special controller made by the manufacturer.

Wattage: Typically, they may have a power rating of between 15-25 watts; a higher wattage means more power in reserve for coping with large joints.

Temperature control: The simplest and cheapest types don't have any form of temperature regulation. Simply plug in and switch on. Unregulated irons form an ideal general-purpose soldering iron for most work.

There is another type of soldering iron that does not run from the mains, they are called gas-powered soldering irons, rather, they use butane. They have a catalytic element which, once warmed up, continues to glow hot when gas passes over them.

Soldering irons are best used along with a heat-resistant bench-type holder, so that the hot iron can be safely kept there when not in between use.

A 240V 60W unregulated soldering iron was used in the process of constructing the circuit.

3.1.3 TESTING

Testing was done in stages using a digital multi-meter. The circuit was also tested at other times by directly plugging it to the PC and carefully watching its response.

3.1.4 POWER SUPPLY.

The voltage source from the supply is switched on to supply power to the circuit.

The output voltage across the regulator output pin was measured using a digital multi-meter with the ground being the ground pin of the voltage regulator. This voltage was found to be about 5.3V.

3.1.5 THE OUTPUT INDICATOR

The voltage across the each of the LEDs was found to be about 1.92V.A total of seven LEDs were driven from the output of the parallel port. The output unit was constructed using 4 green LEDs, 1 red LED, 1 orange LED, 1 infrared LED and 2 piezo-buzzers. The orange LED and a buzzer were connected to RY1 while the red LED and the other buzzer and were connected to RY3. The green LEDs were connected to RY2. The infrared LED was connected to output line 4 (**D3**).

3.1.6 THE INPUT UNIT

Two input devices were connected to the status lines of the parallel port. The input devices used were an infrared receiver and two switches. The infrared receiver used was photodiode. The two switches were connected in series and soldered to pin 15. An external 5V supply was connected to the switches and the end grounded. The infrared emitting diode was connected to an output **D3** in series with a 100Ω resistor to limit the device from sourcing too much current from the parallel port. The receiver (photodiode) was connected to pin 13 of the parallel port in series with a 10kΩ resistor.

3.1.7 MODEL HOUSE

The finished construction was encased in a model house. The green LEDs were placed on the sides and entrance to the model house to demonstrate how the security lights will be switched on. The red LED was placed on the grounds of the model, while the orange LED was placed on the front door of the house. The two buzzers were placed inside the model house. The infrared emitter and transceiver was mounted inside the house at a distance of about 4cm from each other.

3.2 SOFTWARE DESIGN

The program used for the system was designed using Visual Basic 6. It is an object-oriented language. Visual Basic 6.0 has a graphic user interface which makes it easier for most people to use and understand. Object oriented programming is a type of programming in which programmers define not only the data type of a data structure, but also the types of operations (functions) that can be applied to the data structure. In this way, the data structure becomes an *object* that includes both data and functions. In addition, programmers can create relationships between one object and another. For example, objects can *inherit* characteristics from other objects. One of the principal advantages of object-oriented programming techniques over procedural programming techniques is that they enable programmers to create modules that do not need to be changed when a new type of object is added. A programmer can simply create a new object that inherits many of its features from existing objects. This makes object-oriented programs easier to modify.

The first step in writing a instructions to carry out a task is to determine what the output should be- that is, exactly what the task should produce,. The second step is to identify the data or input, necessary to obtain the output, that is, to determine what formulas of doing things can be used to obtain the output.

The following step-by-step process was used in developing the software program for the home automation system:

1. Analysis: The problem was clearly defined. The input was defined, the desired output understood and a relationship between the output and input established.

2. Design: The solution to the stated problem was carefully planned. An algorithm representing the logical sequence of precise steps to solving the problem was defined. This included the use of flowcharts and pseudocodes.

3. Choice of Interface: The method of displaying the input and output was determined by way of selecting objects (text boxes, command buttons, etc) Appropriate command buttons were also developed to allow the user to control the program.

4. Coding: During this stage of the program development, the program code was written using Visual Basic code and entered into the computer. The algorithms devised in Step 2 along with knowledge of Visual Basic were applied in the coding.

5. Testing and Debugging Errors: Errors in the program were located and removed. Testing is the process of finding errors in a program, and debugging is the process of correcting the errors that were found.(An error in a program is called a bug). As the program was being typed, Visual Basic pointed out certain types of program errors. Other types of errors were detected by Visual Basic when the program was executed. However, many errors due to typing mistakes, flaws in the algorithm, or incorrect usages of the Visual Basic language rules were uncovered and corrected by careful detective work.

6. Completion of the Documentation: All the material that describes the program was carefully organized an documented. The documentation of a program is intended to allow another person, or the programmer at a later date, to understand the program. Internal documentation consists of statements in the program that were not executed, but point out the purposes of various parts of the program. Documentation might also consist of a detailed description of what the program does and how to use the programs (for instance, what type of input is expected). Types of documentation include the flowchart, pseudocode and the top-down chart that were used to construct the program. Although documentation is listed as the last step in the program development cycle, it actually took place all through the development process.

3.3 RESULTS.

The PC-based home automation system works on the principle of pre-programmed events as observed from the tests and simulations carried out. A common voltage level of 5Vdc was used throughout the circuit. Whenever the momentary switch, SW1, which is under the front doormat is pressed, the porch lights are switched on and the door bell rings within the house alerting the occupants of the house of the presence of someone at the front door. A second switch, SW2 is also used to show that whenever someone goes out of the door or when the door is opened, the switch which is placed on the door rim is de-activated, thereby disabling SW1.

3.4 DISCUSSION OF RESULTS

It was discovered that whenever the PC was powered up or reset and a POST (Power On Self Test) was carried out and the 3 Relays connected to D0, D1 and D2 were energized and the LEDs were automatically switched ON and then OFF after the POST was finished. The voltage from the parallel port was sufficient to switch the devices on and off.

CHAPTER FOUR

CONCLUSION AND RECOMMENDATION

4.1 Conclusion

The design of the project was quite simple and straightforward. The components used were readily available and affordable in the market. The objectives and targets that were earlier stated were achieved quite satisfactorily.

4.2 Recommendations

Firstly, I will recommend that better use be made of the capabilities of the parallel port by interfacing more devices. All eight of the output lines could be put to use. This would give greater control and capabilities. More input devices like a heat/smoke detector, a temperature sensor; e.t.c could also be interfaced to the system. Motor control could also be incorporated into the design. If more than 8 output lines and more than 5 input lines are desired, a decoder could be used to provide more flexibility.

Secondly, an X10 microcontroller chip could be used instead of the parallel port. It provides far greater control capabilities and functions through the already existing electrical wiring.

REFERENCES

1. Theraja and Theraja — *A Textbook of Electrical Technology, S. Chand & Company Ltd., 1999. (Twenty-Second Edition) ISBN: 81-219-0289-4*

2. Horowitz and Hill — *Art of Electronics, Press Syndicate of The University of Cambridge, New York, 1995, ISBN 0- 521- 49846- 5.*

3. Schneider D.I. — *An Introduction to Programming Using Visual Basic 6.0, Fourth Edition, Pearson Custom Publishing, ISBN 0-536-60446-0*

4. http://www.hut.fi/parallel port output.htm

5. http://www.aaroncake.net/projects/comproom.htm

6. http://www.boondog.com//tutorials/parallel/parallel.htm

7. http://www.logix4u.net/

APPENDIX

A. Part List

i) Semiconductors

D1-D4 = 1n4001 silicon diode

IC1, IC2 = 74LS367 x 2

IC3 = 78L05 5-volt regulator ic

Led1 = Orange coloured LED

Led2-5 = Green coloured LED x 4

Led6 = red coloured LED

Q1, q2, q3 = 2N2222 NPN transistor

Ir_{LED} = Infrared LED

Ir_{PD} = photodiode

ii) Resistors

1kΩ x 6

470Ω x 3

4.7kΩ x 3

iii) Additional components

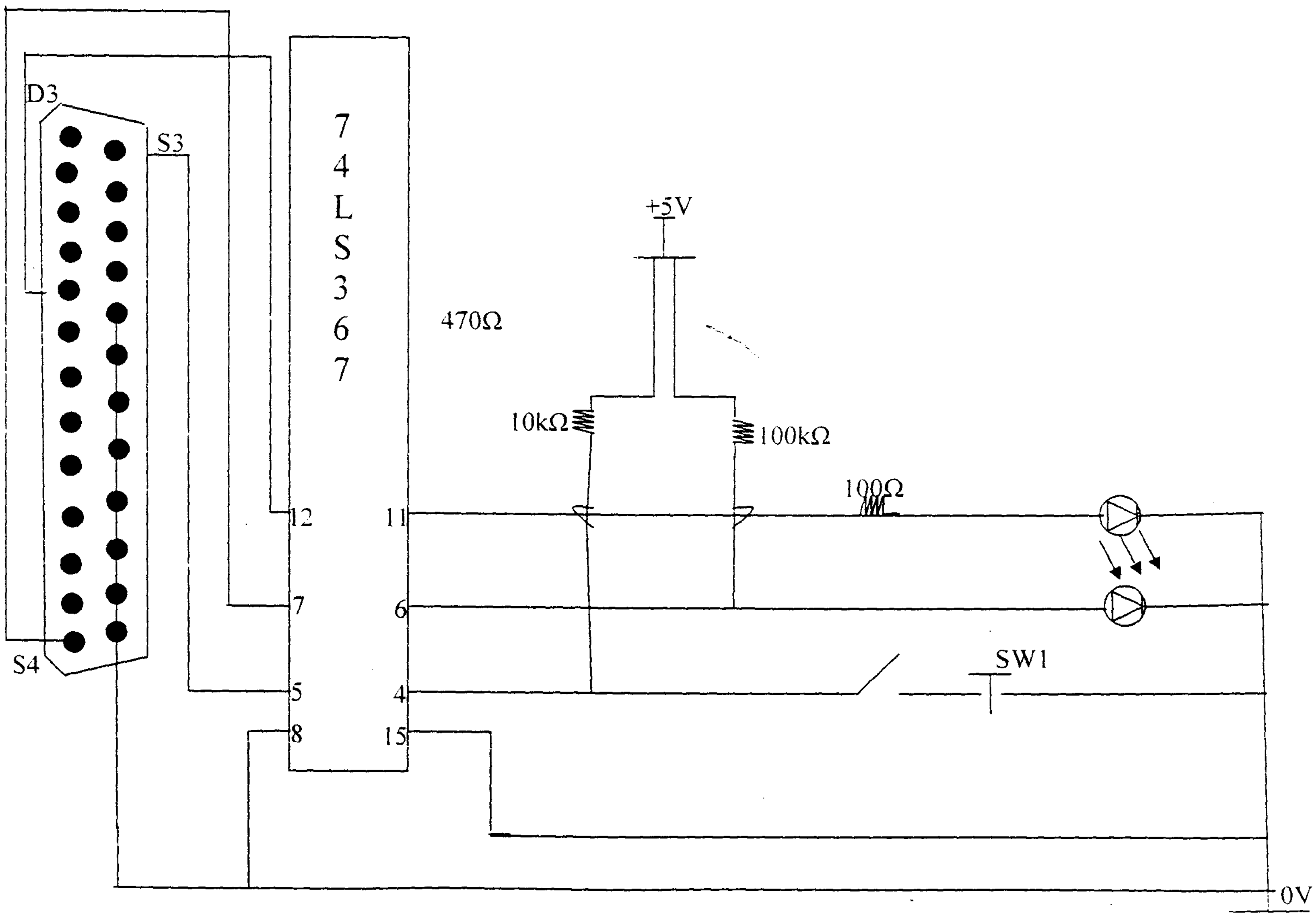
DB-25 Male Connector

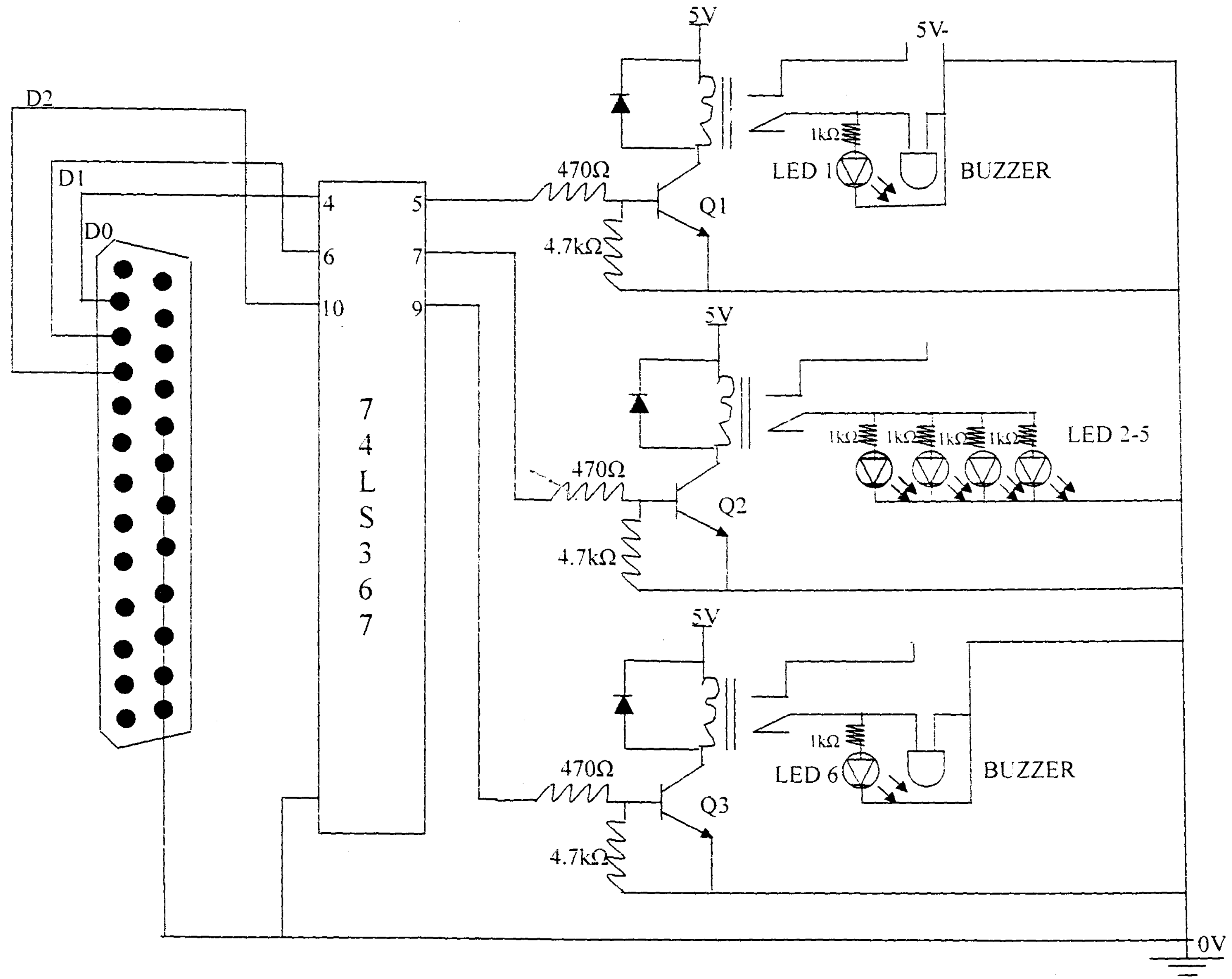
alrm1.alrm2 = Piezobuzzers x 2

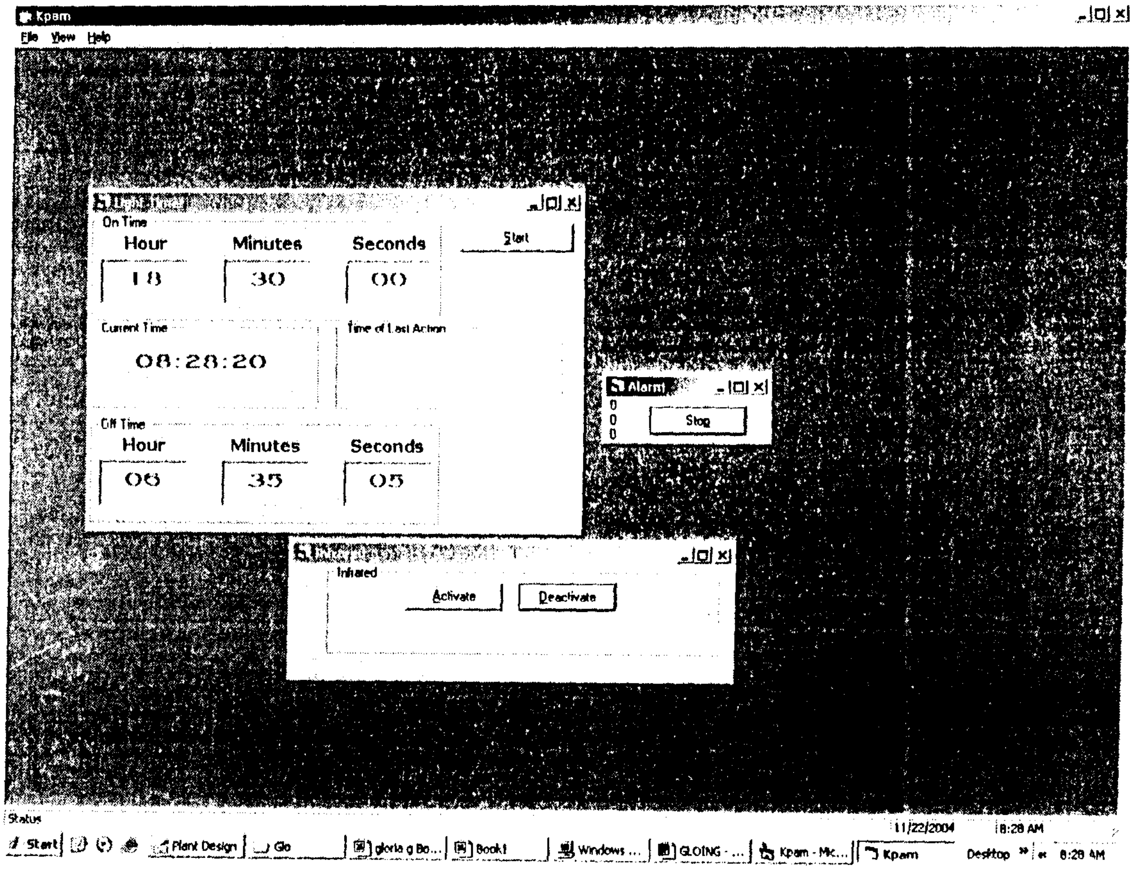
Ry1, Ry2, Ry3 = 6V, 10A Relay x 3

Sw1 = Momentary switch

Sw2 = SPST Switch


D3
S3
S4
74LS367
12
7
5
8
11
6
4
15
470Ω
+5V
10kΩ
100kΩ
100Ω
SW1
0V


5V
5V-
D2
D1
D0
4
6
10
5
7
9
7 4 L S 3 6 7
470Ω
4.7kΩ
Q1
1kΩ
LED 1
BUZZER
5V
470Ω
4.7kΩ
Q2
1kΩ
1kΩ
1kΩ
1kΩ
LED 2-5
5V
470Ω
4.7kΩ
Q3
1kΩ
LED 6
BUZZER
0V



On Time

Hour 18 Minutes 30 Seconds 00

Current Time 08:28:20

Time of Last Action

Off Time

Hour 08 Minutes 35 Seconds 05

Alarm 0 0 0

11/22/2004 8:28 AM

8:28 AM

C: Figure: Screenshot of the graphic user interface showing the automatic light switching system of the home automation system in its developmental stages.

VISUAL BASIC PROGRAM CODES

```
'Dim Alarm As Integer
Dim AlarmTime As String
Dim OffTime As String
Dim AlarmSounded As Integer
Private Declare Function Inp Lib "inpout32.dll" _
Alias "Inp32" (ByVal PortAdress As Integer) As Integer
Private Declare Sub out Lib "inpout32.dll" _
Alias "Out32" (ByVal PortAddress As Integer, ByVal Value As Integer)
Dim CurrentTime As String
Dim BaseAddress

Private Sub Form_Load()
   Me.Width = 6855
   Me.Height = 4695

   Now.Enabled = True
   tmrAlarm.Enabled = False
   BaseAddress = &H378

   'OutPort 888, 0
   'mine% = Inp(&H378)
   'mine% = BitStatus%(888, 7)
   'MsgBox mine%
End Sub
Private Function parrallelput(n As Integer)
   out BaseAddress, n
End Function
Private Function parallelget()
   parallelget = (Inp(BaseAddress + 1) Xor &H80)
End Function

Private Sub delay(n As Single)
   Dim start, check
   start = Timer
   Do Until check >= start + n * 0.001
      check = Timer
```

```
    Loop
End Sub

Private Sub Start_Click()
   AlarmTime = Format$(Val(txthour(0)) & ":" & Val(txtmin(0)) & ":" &
Val(txtsec(0)), "hh:mm:ss")
   OffTime = Format$(Val(txthour(1)) & ":" & Val(txtmin(1)) & ":" &
Val(txtsec(1)), "hh:mm:ss")
   AlarmSounded = True
   tmrTrigger.Enabled = True
End Sub
Private Sub stop_Click()
   tmrAlarm.Enabled = False
   tmrTrigger.Enabled = False
End Sub
Private Sub tmrAlarm_Timer()
   Beep
   parrallelput (1)          'put bit to D0
   'delay (250)
   parrallelput (0)          'put 0 to paralel port to clear the all data lines
   'delay (250)
End Sub

Private Sub tmrTrigger_Timer()
   Dim CurrentHour As Integer
   Dim CurrentMinute As Integer
   Dim CurrentSecond As Integer

   CurrentHour = Hour(CurrentTime)
   CurrentMinute = Minute(CurrentTime)
   CurrentSecond = Second(CurrentTime)
'LIGHTS ON
   If txthour(0) = "" Then
      AlarmTime = Format$((CurrentHour & ":" & Minute(AlarmTime) &
":" & Second(AlarmTime)), "hh:mm:ss")
   End If
   If txtmin(0) = "" Then
      AlarmTime = Format$((Hour(AlarmTime) & ":" & CurrentMinute &
":" & Second(AlarmTime)), "hh:mm:ss")
   End If
   If txtsec(0) = "" Then
      AlarmTime = Format$((Hour(AlarmTime) & ":" &
Minute(AlarmTime) & ":" & CurrentSecond), "hh:mm:ss")
   End If


'LIGHTS OFF
   If txthour(1) = "" Then
      OffTime = Format$((CurrentHour & ":" & Minute(OffTime) & ":" &
Second(OffTime)), "hh:mm:ss")
   End If
   If txtmin(1) = "" Then
      OffTime = Format$((Hour(OffTime) & ":" & CurrentMinute & ":" &
Second(OffTime)), "hh:mm:ss")
   End If

   If txtsec(1) = "" Then
      OffTime = Format$((Hour(OffTime) & ":" & Minute(OffTime) & ":"
& CurrentSecond), "hh:mm:ss")
   End If

   If (CurrentTime = AlarmTime) And Not AlamSounded Then
      AlarmSounded = True
      lblLastAlarm.Caption = CurrentTime
      OutPort 888, 1
      MsgBox "Lights On"
      'tmrAlarm.Enabled = True
      Beep
   End If

   If (CurrentTime = OffTime) And AlarmSounded Then
      AlarmSounded = False
      lblLastAlarm.Caption = OffTime
      OutPort 888, 0
      MsgBox "Lights Off"
      Beep
   End If
End Sub
Private Sub Now_Timer()
```

```
    CurrentTime = Time$
    lblTime(0).Caption = CurrentTime
End Sub

Private Sub OutPort(PortAddress%, OutNum%)
    PortState% = Inp(PortAddress%)
    PortNum% = PortState% + OutNum%
    If OutNum% = 0 Then
        out PortAddress%, 0
        Exit Sub
    End If
    out PortAddress%, PortNum%
End Sub
Function BitStatus(PortAddress%, BitYouWant%) As Integer
    If PortAddress% = 888 Then
        NumOfBits% = 8
    ElseIf PortAddress% = 889 Then
        NumOfBits% = 5
    Else
        NumOfBits% = 4
    End If

    ReDim PortBits(NumOfBits%) As Integer
    PortNum% = Inp(PortAddress%)

    For i = 1 To NumOfBits%
        PortBits%(i) = PortNum% Mod 2
        PortNum% = Fix(PortNum% / 2)
    Next i

    BitStatus% = PortBits%(BitYouWant%)
End Function



Private Declare Function Inp Lib "inpout32.dll" _
Alias "Inp32" (ByVal PortAdress As Integer) As Integer
Private Declare Sub out Lib "inpout32.dll" _
Alias "Out32" (ByVal PortAddress As Integer, ByVal Value As Integer)
Private Sub Form_Load()
```

```
   BaseAddress = &H378
End Sub
Private Function parrallelput(n As Integer)
   out BaseAddress, n
End Function

Private Function parallelget()
   parallelget = (Inp(BaseAddress + 1) Xor &H80)
End Function
Private Sub delay(n As Single)
   Dim start, check
   start = Timer
   Do Until check >= start + n * 0.001
      check = Timer
   Loop
End Sub

Private Sub Start_Click()
   'WRITE CODE TO ACTIVATE ALARM SYSTEM
End Sub

Private Sub stop_Click()
   'WRITE CODE TO STOP ALARM SYSTEM HERE
   trmAlarm.Enabled = False
   tmrTrigger.Enabled = False
End Sub

Private Sub tmrAlarm_Timer()
   Beep
   parrallelput (1)            'put bit to D0
   delay (250)
   parrallelput (0)            'put 0 to paralel port to clear the all data lines
   delay (250)
End Sub

Private Sub tmrTrigger_Timer()
   Dim valueRead As Integer
   valueRead = parrallelget
   Print valueRead              'Turn Switch on, read the number(XXX); turn it
off to read 2nd number
```

```
   If valueRead = 0 Then          'XXX is read from the form whenthe switch
is closed
      tmrAlarm.Enabled = True
   Else
      tmrAlarm.Enabled = False
   End If
End Sub

Private Declare Function Inp Lib "inpout32.dll" _
Alias "Inp32" (ByVal PortAdress As Integer) As Integer
Private Declare Sub out Lib "inpout32.dll" _
Alias "Out32" (ByVal PortAddress As Integer, ByVal Value As Integer)
Private Sub Form_Load()
   BaseAddress = &H378
End Sub
Private Function parrallelput(n As Integer)
   out BaseAddress, n
End Function

Private Function parallelget()
   parallelget = (Inp(BaseAddress + 1) Xor &H80)
End Function
Private Sub delay(n As Single)
   Dim start, check
   start = Timer
   Do Until check >= start + n * 0.001
      check = Timer
   Loop
End Sub

Private Sub stop_Click()
   trmAlarm.Enabled = False
   tmrTrigger.Enabled = False
End Sub

Private Sub tmrAlarm_Timer()
   Beep
   parrallelput (1)           'put bit to D0
   delay (250)
   parrallelput (0)           'put 0 to paralel port to clear the all data lines
   delay (250)
End Sub

Private Sub tmrTrigger_Timer()
   Dim valueRead As Integer
   valueRead = parrallelget
   Print valueRead              'Turn Switch on, read the number(XXX); turn it
off to read 2nd number
   If valueRead = 0 Then          'XXX is read from the form whenthe switch
is closed
      tmrAlarm.Enabled = True
   Else
      tmrAlarm.Enabled = False
   End If
End Sub
```