DESIGN AND CONSTRUCTION OF MICROCONTROLLER BASED QUEUE MANAGER WITH PC BASED GRAPHICAL USER INTERFACE

BY

UDENSI DAVID EMEKA

2006/24429EE

ELECTRICAL AND COMPUTER ENGINEERING

F.U.T MINNA, NIGER STATE

NOVEMBER, 2011

DESIGN AND CONSTRUCTION OF MICROCONTROLLER BASED QUEUE MANAGER WITH PC BASED GRAPHICAL USER INTERFACE

BY

UDENSI DAVID EMEKA

2006/24429EE

A THESIS SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING, FEDERAL UNIVERSITY OF TECHNOLOGY MINNA

NOVEMBER, 2011

DEDICATION

I dedicate this project to the Almighty God who gave me strength and grace to accomplish it, and to my parents and entire family for their support and encouragement through it all.

DECLARATION

I UDENSI DAVID EMEKA, declare that this work was done by me and has never been presented elsewhere for the award of degree. I also relinquish the copyright to the Federal University of technology, Minna

UDENSI DAVID EMEKA	MR. ADEJO ACHONU
(Name of Student)	(Name of Supervisor)
[illegible] 9/11/2011	[illegible] 9/11/2011
(Signature and Date)	(Signature and Date)
ENGR. A. G. RAJI	Engr. Dr. G. I. Igbal
(Name of H.O.D)	(Name of External Examiner)
[illegible] (March 15, 2012)	[illegible] 6/3/2012
(Signature and Date)	(Signature and Date)

ACKNOWLEDGMENT

God is faithful; the successful accomplishment of this project was totally his doing.

I acknowledge the graceful supervision and intellectual personality of my supervisor Mr. Achonu Adejo, taking me through each step of this project; never too harsh and never too tolerable either, thank you sir, I couldn't have asked for a better supervisor. Thank you Mrs. E. A. Asindi, for your motherly touch throughout this period, and to all dear staff of the department; the H.O.D Engr. A. G. Raji, Mr. Micheal David, Engr. (Mrs) C. O. Alenoghena but to mention a few.

I must also acknowledge my friends, colleagues and loved ones; my good friends Ufedojo Opaluwa, Mary Idakwo, Elijah Ishyaku, Odili Cynthia, Itohan Enabulu, Dawar Katfun, Opute Bright to mention a few and my best friend Haruna Angyu, my roommates Chinedu, Epelle, Samuel, Usman, Charles and Chris. I would also use this medium to acknowledge the fellowship to which I belong in – Fellowship of Christian Students (FCS) and the Drama unit for their love and spiritual upbringing throughout my stay in the university. And to all those not mentioned who have positively affected me I hold you dearly in my heart

ABSTRACT

The microcontroller based Queue manager with PC based graphical user interface (GUI), is all about the utilization of simple electrical and computer components and modules to create a system to manage a departmental registration. A programmed microcontroller connected to an LCD and push buttons forms the queue manager which performs the actual queue management, while a computer installed with the departmental registration software performs the actual registration.

TABLE OF CONTENT

Cover page i

Title page ii

Dedication iii

Declaration iv

Acknowledgement v

Abstract vi

Table of Content vii

List of Figures viii

List of Tables ix

Chapter One: Introduction

1.1.0 Introduction 1

1.2.0 Objectives 2

1.3.0 Methodology 2

1.4.0 Sources of Materials 3

1.5.0 Scope of Project 3

1.6.0 Block Diagram 4

1.7.0 Flow Chart of Registration GUI..7

1.8.0 Project Outline..8

Chapter Two: Theoretical Background

2.1.0 Historical Background..9

2.1.1 History of Queuing Systems..10

2.1.2 History of People Counting Systems..11

2.2.0 Queue Management Concept..11

2.2.1 System Content..13

2.2.2 Stand Alone Queue System..13

2.2.3 Advanced Queue System..14

2.2.4 Centralized Control Queue System..15

2.3.0 Review of Related work..16

2.4.0 Complete Circuit Diagram..18

Chapter Three: Design and Implementation

3.1.0 Analysis and Design..19

3.2.0 Power Supply Unit..19

LIST OF TABLES

Table 3.1: Pin configuration of an LCD

bring about queue management with little or no time delay and a quick response to requests and queries, which also has visual monitoring and control parts. This project presents the design of a MICROCONTROLLER BASED QUEUE MANAGER.

1.2.0 OBJECTIVES

The objectives of this project include:

1. To design a device that aids management of queues in institutions
2. Assist in time management for people on the queue
3. Aid in controlling public aggressiveness by people on queues
4. Improve the technology of queue management in the country and beyond.
5. To design a demonstrative departmental registration portal (Graphic User Interface/GUI) on a computer

1.3.0 METHODOLOGY

The different steps involved in the execution of this project include:

a. Design and construction of the electronic circuit
b. Programming of the microcontroller
c. Design of the demonstrative registration GUI
d. Connection to a PC and printer
e. Testing

The process of the design of the circuit would include the purchase of the adequate, effective and efficient components, which according to research must reach a certain ratings of voltage, current, frequency etc. Programming the microcontroller entails choosing a proper programming language or assembly language (C++ in this case). The next phase is that of connecting the circuit to a computer (Registration point) and the graphical user interface. Then finally the last phase would be the testing of the entire system, using a test-queue.

1.4.0 SOURCES OF MATERIAL

The following are the sources of materials used for this project:

1. The internet
2. The school and departmental library
3. Other external libraries and
4. Purchased book from bookshops and stationeries

The components mainly were obtained from electrical components shops

1.5.0 SCOPE OF PROJECT

This project—a microcontroller based queue manager, comprises of two parts; the physical electronic aspect and the computer graphical user interface (GUI). It is applicable in a wide variety of situations that may be encountered in business, commerce, industry, healthcare, public service and engineering. Applications are frequently encountered in customer service situations as well as transport and telecommunication [1]. The scope of this project is broad and can be applied from simple queues to really complex queues. Some of the areas where this project can be applied are enumerated below:

1. Registration queues
2. Bank queues
3. Restaurant queues

4. Bus queues
5. Airports
6. Post offices
7. Hospital queues
8. Any organization that has to do with ticket issuing

The above mainly touches aspects that are related to the physical electronic aspect, for the GUI aspect of the project the scope of application are enumerated below:

1. Departmental registration e.g. Electrical Computer Department
2. School library
3. Any other database associated application having to do with queues

From the above description, this project will find relevance across extensive areas, covering wide applications in different types of queues

1.6.0 BLOCK DIAGRAM

The different parts of the project can be viewed in the block diagram shown below

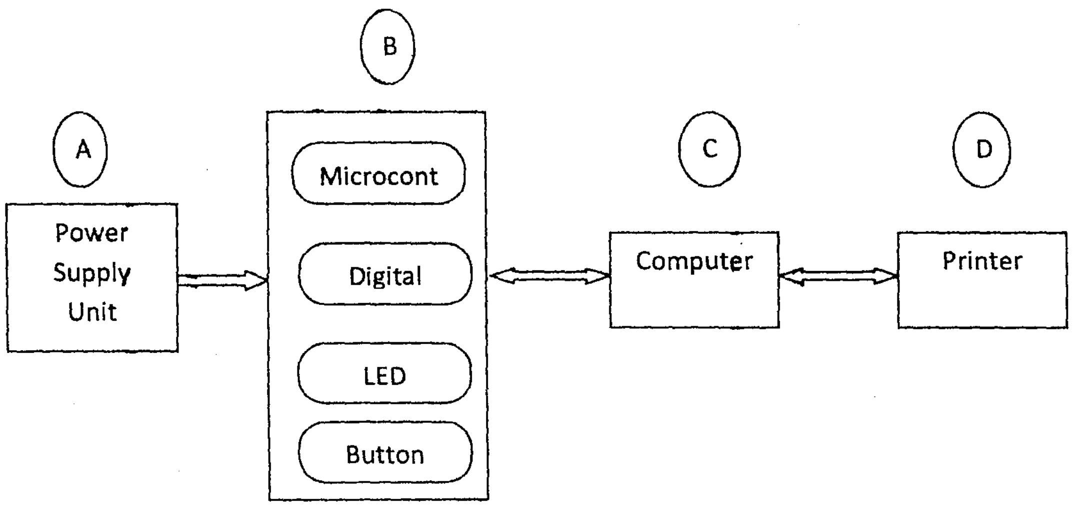

Fig 1.0 Block diagram of Project

The entire project is in 3 parts as is seen in the block diagram above:

A. Power supply unit: supplies the entire project with 5Vdc voltage
B. Controller unit: comprises of the physical design which contains the microcontroller, in this block there are buttons and LED display to indicate the functionality of the entire system
C. The Computer Unit: the microcontroller unit is connected to the computer. Also in this block, the registration GUI is designed and also a database bank for storing the data of the registration
D. The Printer Unit: Prints the tickets for the on-button customer i.e. the customer that is presently at the microcontroller unit end

This project comprises of many electrical power components (e.g. transformer, rectifier, resistors, capacitors, transistors), digital electronic components (e.g. Liquid crystal display—LCD) and integrated circuit or microcomputer components (e.g, the microcontroller—8051, frequency crystal) and works with basic physical electrical principles such as voltage, currents and resistance.

The queue manager operates on an input/output basis; centered on a programmed 8051 microcontroller. The system receives the input from the client on the device by the push of the *entrance button*. This action then trigger an output from the microcontroller to the computer system connected together via an RS 232 (Recommended Serial 232) and this input to the computer system can be used to output a print command to a printer to print a ticket which would contain a number or any other form of identification for the client. The client then is at liberty to wait without being on a physical queue until the time for he/she is to be served. This is possible because the client has been registered in the memory of the microcontroller and these numbers (after many other client registrations) are displayed on the liquid crystal display (LCD). The microcontroller using the FIFO (first-in-first-out) approach starts from the number of the first client that was registered in its memory down to the last client that was registered (this number is equivalent to the total length of the queue).

Another feature of the queue manager is that it utilizes a simple Computer Aided design Graphical User Interface, using java (a 3rd generation programming language) and a netbeans 6.7 compiler the GUI was designed. Since the focus of the queue manager was to manage queues, a real-time queuing situation was needed and so a GUI for departmental registration was incorporated in the project. The GUI contain courses data for each level in the department (from 100-500 level); once a client (in this case a student) gets past the queue manager he inputs his departmental details in the program (GUI) and then proceeds to selecting core courses and carry-over courses (if any), the program also monitors the credit load of the courses being selected and once they exceed the highest credit load for that particular level it indicates and stops the registration, it also gives information on the status of the student; either *'IGS'* or *'DEF'*. When the selection is over the student registration data is stored in a database and could be retrieved any time if needed

Java object-oriented programming language introduced in 1995 by Sun Microsystems, Inc, facilitates the distribution of both data and small applications programs. Java applications do not interact directly with a computer's central processing unit (CPU) or operating system and are therefore platform independent, meaning that they can run on any type of personal computer, workstation, or mainframe computer [7].

1.7.0 FLOW CHART OF REGISTRATION GUI:

Under the computer unit, the flow chart for the registration GUI is shown below

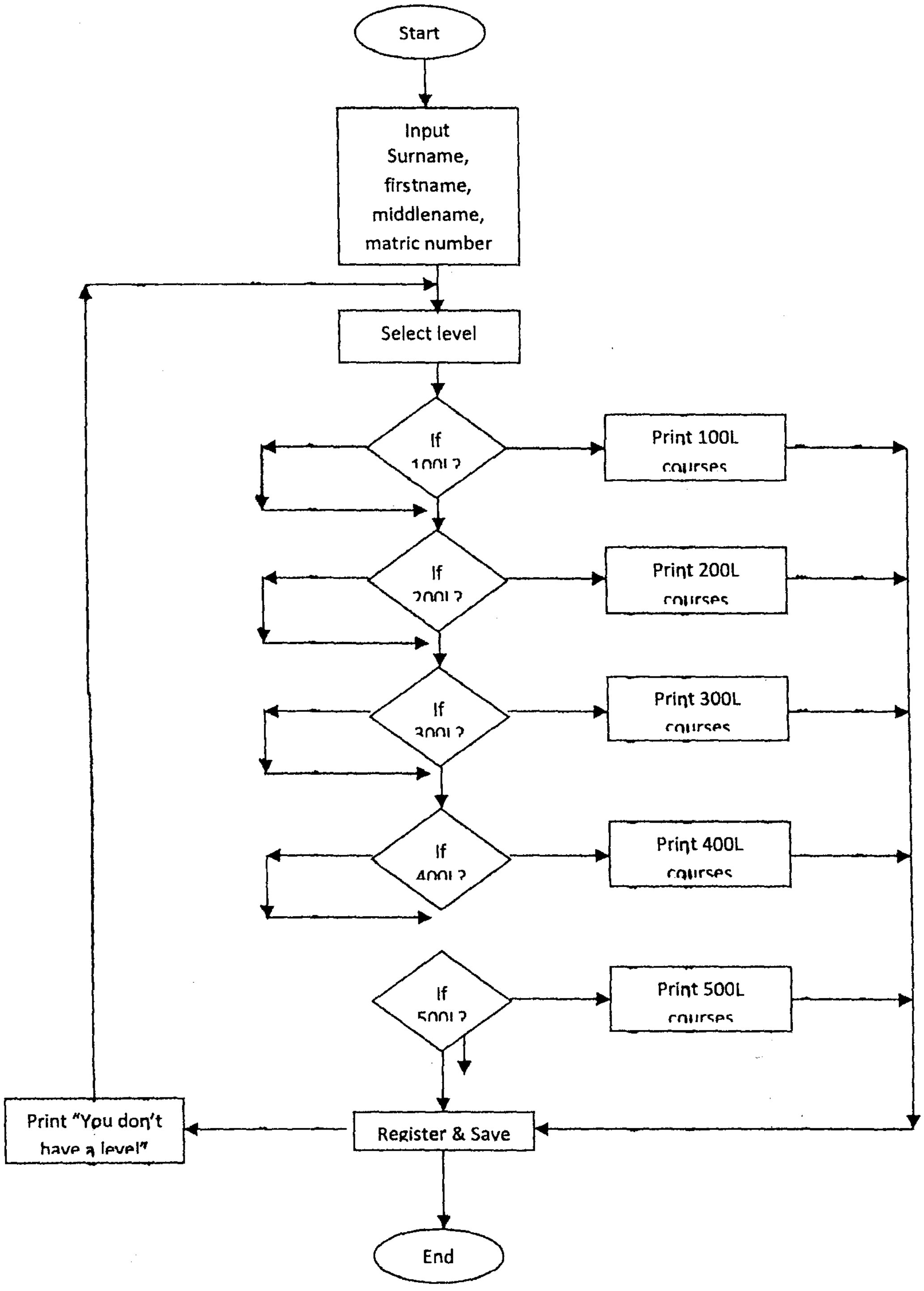

Fig 1.1 Block Diagram of Graphical User Interface

1.8.0 PROJECT OUTLINE:

This project is divided into five chapters, chapter one is introductory chapter,
Chapter two deals with literature review, chapter three is the design procedure, chapter four is connected with the analysis and discussion of results. The last chapter which is five and it involves conclusion, limitation, recommendations and useful suggestion for further studies.

CHAPTER TWO

LITERATURE REVIEW

2.1.0 HISTORICAL BACKGROUND:

The history of queue management goes back to primitive man; even in the days of Noah there was a need for queue management in leading all the animals to the ark.

Fig 2.0: Noah's Ark

The first use of the term queue occurred in 1951 in the journal of the Royal Statistical Society. David G. Kendall (England, 1918-2007) published an article "Some Problems in the Theory of Queues". On page 153, the first line of the third paragraph he refers to a "simple queuing system," though there were other articles on the subject much earlier than his—even using the word "queue", but not using the word "queuing" or the phrase "queuing system"[8][9].

2.1.1 HISTORY OF QUEUING SYSTEMS

According to Avi Mandelbaum, Conny Palm was the first to consider customer abandonment of a queue. In 1953, David G. Kendall introduced the A/B/C type queuing notation. "The study of queues with multiple servers dates back over fifty years to the seminal work and Wolfowitz (1955)." (Mor Harchol-Balter) one of the standard ways of setting up integral equations to describe certain queuing systems was to use "Lindley's integral equation" (D. V Lindley, 1952) [8]. "The theory of continuous time storage models was initiated by P.A.P Moran, J. Gami and N.U Prabhu during 1956-63. The book "Queues, Inventory, and Maintenance" by Philip M. (McCord) Morse, was published in 1958 and is considered the first textbook on queuing. The concept of reneging (or queuing abandonment) was introduced by Conny Palm (1937) and was rediscovered and named by Frank Haight (1958)[8]. Haight also introduced the concept of bulking and parallel queues (1958). H. White and L.S. Christie were the first to consider severe breakdown (1958) [8]. Other early researchers in severe interruptions include Miller (1960), Gaver (1962), Avai-Itzhak and Naor (1961), Keilson and Kooharian (1960). Cohen (1958) seems to have been the first to consider retrial queues. It is curios that there are two individuals named Jackson with Major contributions to queuing networks. They are RRP Jackson (Ray Jackson, first queuing publication 1954) and JP Jackson (James Jackson, first queuing publication 1957) [8]. According to RRP Jackson the product rule is Jackson's theorem (R.R.P.); Jackson's networks are due to Jackson's (J.R.) [8]. The proof of little's formula (so called because it was first proved by John Little) was published in 1961. The use of queuing for computer performance began around 1970; D.R Cox introduced "Supplementary variable technique" (1965) to analyze queues, Percy Brill introduced the "Level Crossing Method (1975), Robert Coper did some of the earliest work on (1969-1970) on Polling Models [8]. In particular the "vacation model" was introduced (where it was used as a component in the waiting time analysis of polling model), and a special case of decomposition theorem was given. Richard Larson developed the "queue inference engine" (1990), Gelenbe (1991) introduced the concept of negative customer and G-network. Marcel Neuts introduced the matrix analysis method (1981). John Frank Charles Kingman introduced "algebra of queues" (1966) [8] [9].

2.1.2 HISTORY OF PEOPLE COUNTING SYSTEMS

Major advancements in people counters has been based on the resolution and the sophistication of the technology employed, the simplest form of human counters began as a single horizontal infrared beam placed at an entrance linked to a PC, and counted an increment or decrement when the link is broken. It progressed from infrared based counters to computer versions which typically used either closed-circuit television camera or an IP camera to count and monitor the number of customers. It then moved to thermal imaging systems which used array sensors that detect heat sources rather than using cameras as in computer versions. Modern people counting systems are referred to as Synthetic Intelligence Counters; the system employs IR transceivers to create count zones at ankle heights [10].

Though this evolution affected queuing systems on a quite small level, it was mainly used for security and intelligence crime detection systems, but they can be used for the purpose of queue management in commercial and other related fields.

2.2.0 QUEUE MANAGEMENT CONCEPT

In queuing system, there are basically 6 types of queue models which include [2]:

1. SPF (Shortest processed first)
2. FIFO (First In First Out)

2.2.1 SYSTEM CONCEPT:

In summary, the concept of this project is to produce a device that can perform the following function in solving the problem of queuing:

- Ability to recognize and identify new comers to the queue
- Perception of waiting time should be managed.
- The process must be clearly Identified; start, present and end points must be visible.
- Should be able to deliver tickets to enhance identification of customer
- The process must include positive feedback of progress.
- Achieve communication with a Computer System

In the Queue Management System, many products have been produced according to increase in the need for effectiveness in queue flow. The following are some of the existing queue management systems [2]:

1. The Stand Alone Queue System
2. Advance Queue System and
3. Centralized Control Queue System.

2.2.2 STAND ALONE QUEUE SYSTEM

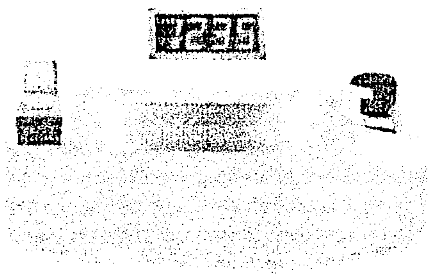

Fig 2.1: Stand Alone Queue System

Fig 2.1 shows the system of a queue processor based solution. This system design is based on the single service and single counter operation. For the operation of this system, this system operates by displaying numbers in sequential or randomized order. This system is best use in application of small business, clinic & single payment counter. Fig 2.2 below shows the example of operation in Stand Alone Queue System [2].

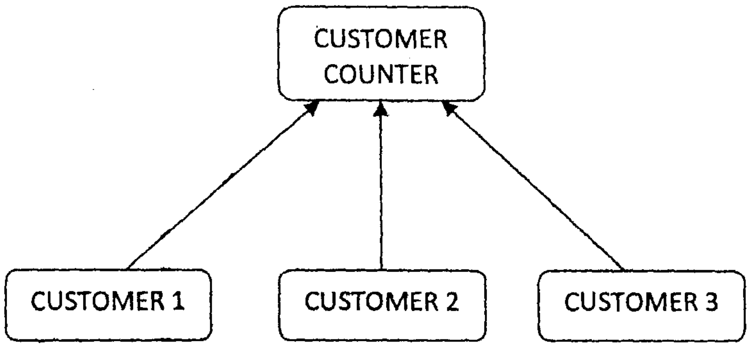

Fig 2.2: Operation of Stand Alone Queue System

In the Stand Alone Queue System, one counter operates. The entire customer will be managed at the same counter with the single service operation. The model used is the FIFO (First in First Out) queue model. With this concept, the entire customer will be treated equally. The Stand Alone Queue System is best described with a single department service operation.

2.2.3 ADVANCE QUEUE SYSTEM

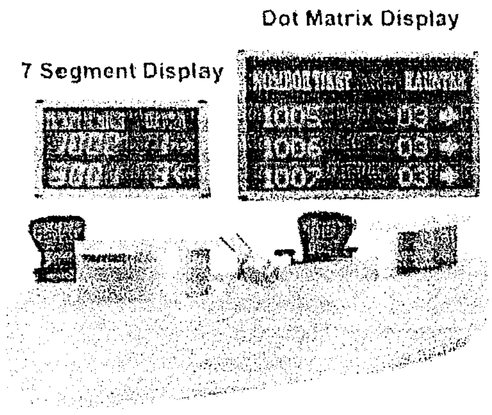

Fig 2.3: Advance Queue System

Fig 2.3 shows a system of queue processor based solution. This system design can support up to 32 service points. It provides useful queue features and comprehensive reports; it also provides information concerning other service points that are in the system and assigns customers to free service points. The application of this system is best used for service center, bank, hospital, pharmacy, or any organization with multiple counters. It also provides real-time status monitoring for the queue management analysis. Fig 2.4 below shows the example of the Advance Queue Management Systems operations [2].

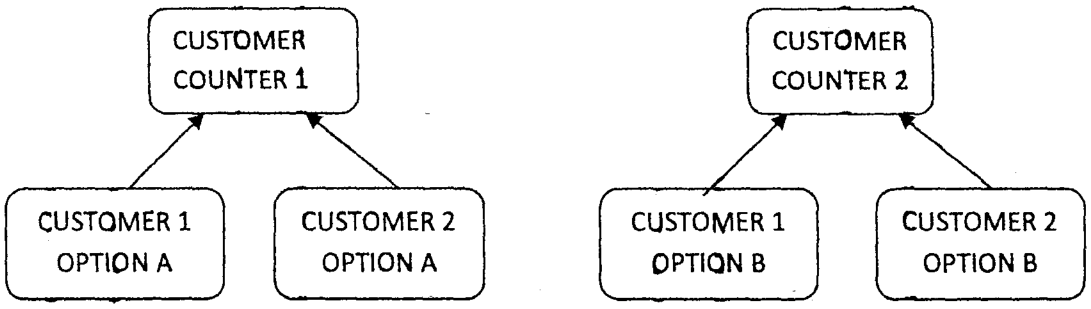

Fig 2.4: Operation of Advance Queue System

The Advance Queue System is the system that consists of multiple service operation and multiple counters. In this system, the customer will be arranged according to their service operation. This system operation is efficient when it has to manage many customers with many service operations. This system operation is best described as the single departmental operation with multiple counters.

2.2.4 CENTRALIZED CONTROL QUEUE SYSTEM

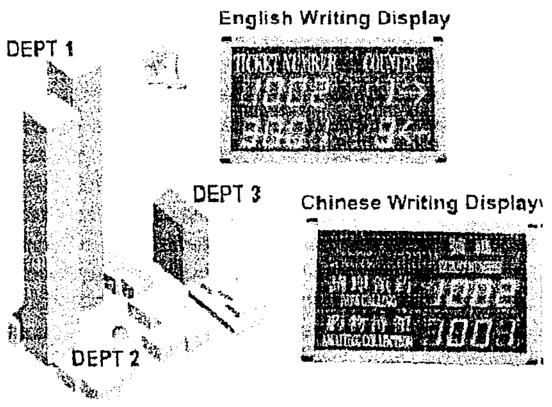

Fig 2.5: Centralized Control Queue System

Fig 2.5 shows a system of a high-end PC-based solution. This system queue server will be able to support up to 20 departments (containing several systems). Each of the departments can have up to 32 services and 60 counters. The system is network-compatible; this means each department can be located at different building or even different geographical area which is connected through LAN or INTRANET. This system also provides real-time status monitoring [2].

2.3.0 REVIEW OF RELATED WORKS

MOHAMMED FAZLI BIN ALIAS, of the Faculty of Electrical and Electronics Engineering, University of Malaysia Pahang worked on a Front Desk Customer Service for Queue Management System. The Queue Management System was designed to manage 18 customers with single department and multiple counters, with two operations and four customer counters in this system. He used a Seven Segment Display to display the related information from the system and a Sound Module (Buzzer). The major component of his project was a PIC microcontroller that was programmed using PIC Basic Language for programming the system of Queue Management System to achieve the entire project objective. The major drawback of the project was that it couldn't attend to more than 18 clients/students, but I have used a AT89C52 microcontroller which can be used for a larger number and also it utilizes a PC based GUI [2]

OJEKUNLE PETER, of the faculty of Electrical and Computer Engineering, Federal University of Technology Minna Nigeria worked on an automated people counting system, using a programmed AT89C2051 microcontroller triggered by light sensors (LED and LDR) station before and after a door (which could be used as a service door), the increment in the number of persons passing through the door helps perform the counting, this increment is displayed on a Liquid Crystal Display. It employed a 240V ac main source stepped down and regulated to 5V dc. The major drawback is that it cannot be sufficiently used to manage queues because the system does not have any allowance for identifying the customers/students after counting [10]

2.4.0 COMPLETE CIRCUIT DIAGRAM

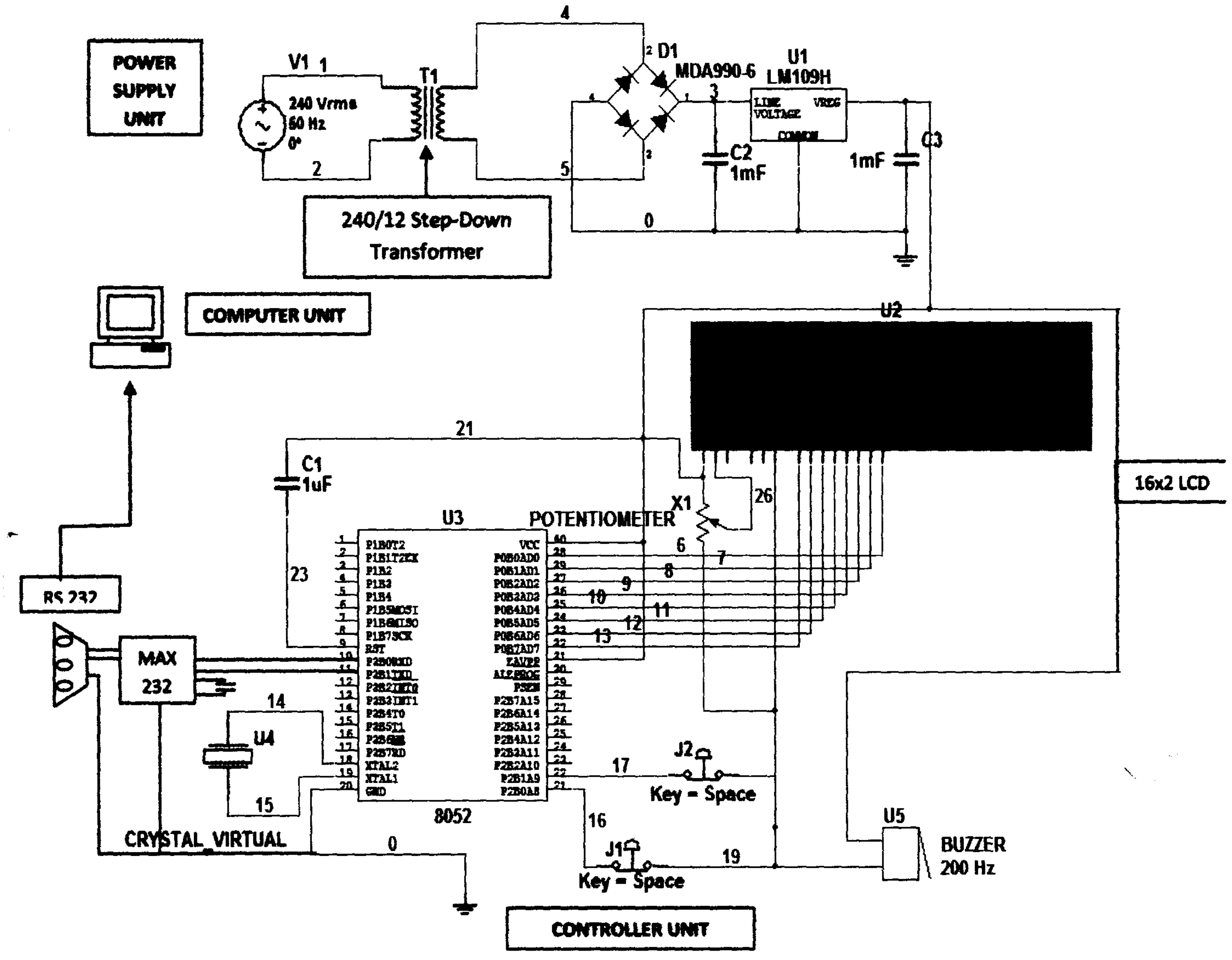

Fig 2.6: Microcontroller Based Queue Manager with PC Based GUI Circuit Diagram

CHAPTER THREE

DESIGN AND IMPLEMENTATION

3.1.0 ANALYSIS AND DESIGN

The block diagram of the system is shown below. The system consists of 4 basic units; the power supply unit, the controller unit, the computer unit and the printer unit. This chapter gives a thorough and clear analysis of the design of each of the modules that make up the Queue Manager and the Registration Graphical User Interface.

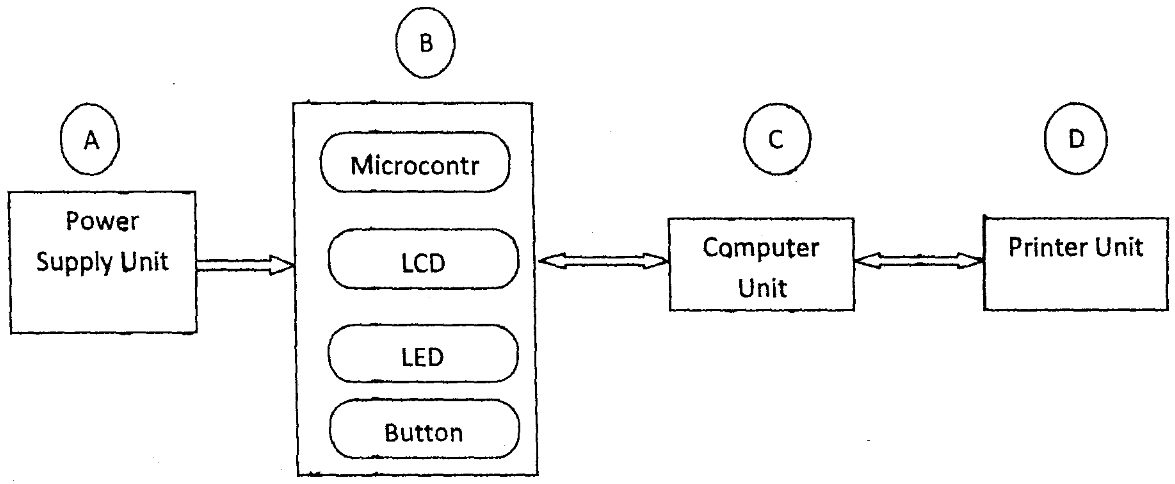

Fig 3.0: Block diagram of Project

A: Power Supply Unit

B: Controller Unit

C: Computer Unit

D: Printer Unit

3.2.0 THE POWER SUPPLY UNIT

Power supply can be in the form of alternating current or a direct current. Since the system requires a steady dc power supply, an AC source had to be converted to dc for the system to be powered efficiently. For an alternately current power supply to become a direct current power supply, it must undergo some processes which include voltage transformation, voltage rectification, filtration and regulation. The power supply unit supplies the whole system with the appropriate regulated biasing voltage. The power supply circuit comprises of a step-down transformer to step down the source voltage, a rectifier which converts the alternating current signal into direct current, a filter capacitor, a 5V voltage regulator IC which regulates the direct current to an exact value of 5V

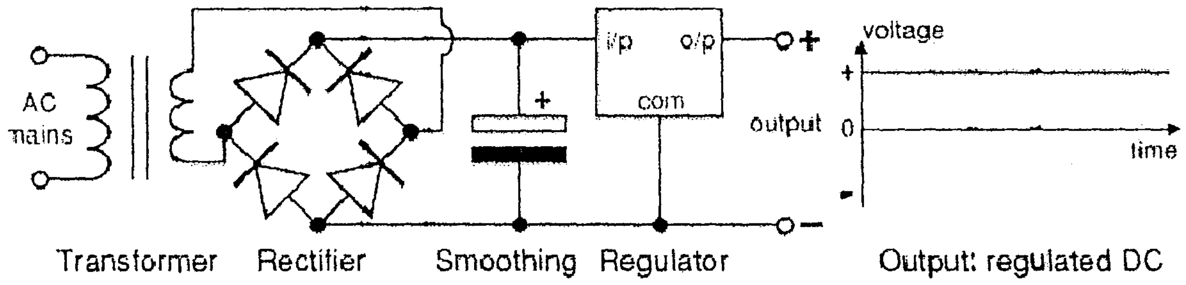

Fig 3.1: schematic circuit diagram for power supply unit

3.2.1 Supply Input:

The supply of the circuit is the conventional Nigerian PHCN rating where Voltage = 240V, Current = 500mA, Frequency = 50Hz

3.2.2 Voltage Transformation:

The voltage transformation was achieve using 12V — 0.5A step-down transformer to step down AC mains supply from the normal 240V, 50Hz Nigerian supply

For the transformation of the transformer follow the following formula [11]:

$$\frac{Np}{Ns} = \frac{\mathrm{Vp}}{Vs} = \frac{\mathrm{Is}}{\mathrm{Ip}}$$

...... Equation 3.1

Where:

N_p= Number of turns in the primary

N_s= Number of turns in the secondary

V_p = Primary voltage

V_s = Secondary voltage

I_p = Primary current

I_s = Secondary current

Readings:

The following are actual readings from the project.

Current rating = 500mA

Primary Voltage rating = 240Vrms

Secondary Voltage = 12Vrms

Substituting the above circuit readings in the equation below:

Vpeak = √2 × Vrms ,........ Equation 3.2

= √2 × 12 = 16.97V

3.2.3 Voltage Rectification:

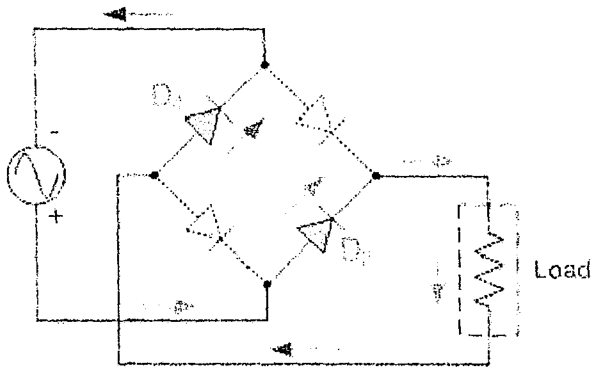

Fig 3.2: Voltage rectification circuit diagram

Rectification is the process where the AC voltage is converted into DC voltage in this case the output is the output at the transformer. The process is accomplished by using rectifier diodes which permit the flow of current in one direction only. Four IN4001 diodes connected in the bridge circuit is used to accomplish this in this system, though there exist rectifier bridge ICs (which were actually employed in this project)

$V_{in} = V_{peak} = \sqrt{2} \times V_{rms}$ .... Equation 3.3

And the average DC voltage of the bridge rectifier is

$V_{dc}\ (peak) = V_{rms}\sqrt{2} - 1.4$... [10] .... Equation 3.4

Where 1.4 is the forward voltage drop in two adjacent diodes of the rectification

Readings:

Substituting the value of the Vrms reading read from the transformer to the rectifier bridge into Equation 3.4; we get the following

$V_{dc}\ (peak) = 12\sqrt{2} - 1.4$

$= 15.56V$

3.2.4 Voltage Filtration:

The process involves the removal of AC ripples from the rectified voltage. It involves the use of a capacitor to filter off the ripples which are embedded on the DC output. The ripples have a frequency that is twice the frequency of the 240V AC utility supply. The filter capacitor ripple is given as [10]:

$V_{ripple} = I/fC$ .... Equation 3.5

Where:

$V_{ripples}$ = peak to peak voltage

I = Current (DC)

f = ripple frequency

C = capacitance of capacitor

Readings:

The ripple voltage has a frequency that is twice the frequency of the 240V, 50Hz rating of the AC utility supply. Therefore:

Frequency $f = 2 \times 50 = 100Hz$

$V_{ripples}$ = peak-peak voltage = 15.56V

Current = 500mA

V ripples = I/fC

Capacitance C needed to filter the ripple is thus:

C = I/fV

$C = 500mA/100 \times 15.56$

$C = 321\mu F$

This is the minimum value needed to maintain a regulated 5V output, but to ensure sufficient filtering of the rectified AC power a 1000μF is used.in this project

3.2.5 Voltage Regulation:

In this project, a 5V DC supply voltage (operational) is required for the main circuit. Therefore, a positive voltage regulator IC is used which provides a constant 5V. For a 5V regulated output, the input voltage is a minimum of 7V (at least 2V overhead is needed to maintain the output regulated). Voltage regulator ICs are available with fixed (typically 5V, 12V, 15V) or variable output voltage. They are also rated by the maximum current they can pass. Most regulators include some automatic protection from excessive current ('overload protection') and overheating ('thermal protection'). Many of the fixed voltage regulator ICs have 3 leads and look like power transistor. This project made use of an LM7805 i.e. a fixed regulator

3.2.6 Supply Unit Output:

The process of filtering by use of a capacitor is not sufficient to completely eliminate AC ripples from the DC output of the rectifier bridge; this is because the capacitors used have the ability to charge and discharge with time [12]; therefore a second capacitor is used across the outputs of the power supply unit.

After transformation, rectification, filtration and regulation the output of the power supply unit is now 5V, 0.5A which is suitable to be passed on to the controller unit which utilizes a 5V voltage rating

3.3.0 THE CONTROLLER UNIT:

The controller unit is where actual arithmetic counting and display of values is done, the main components of this unit are the microcontroller – which performs the arithmetic counting, and the Liquid Crystal Display (LCD) – which displays the values sent by the microcontroller. Other peripheral connections are made to the microcontroller and the LCD which altogether make up the Controller Unit. The controller unit is made up of the following:

3.3.1 The Serial Port (RS 232):

Recommended Standard 232, is the traditional name for a series of standards for serial binary single-ended data and control signals connecting between a DTE (Data Terminal Equipment) and a DCE (Data Circuit-terminating Equipment) [14]. This device is the transport device which aids communication between the Controller Unit and the Computer Unit [14]. It is plugged in to the computer via the computer serial ports. The standard defines the electrical characteristics and timing of signals, the meaning of signals, and the physical size and pin-out of connectors.

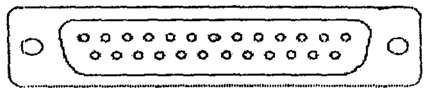

Fig 3.3: A 25 pin connector RS-232

The RS-232 standard defines the voltage levels that correspond to logical one and logical zero levels for the data transmission and the control signal lines. Valid signals are plus or minus 3 to 15 volts; the ±3 V range near zero volts is not a valid RS-232 level. The standard specifies a maximum open-circuit voltage of 25 volts: signal levels of ±5 V, ±10 V, ±12 V, and ±15 V are all commonly seen depending on the power supplies available within a device. The RS-232 (drivers and receivers) protect the system by its capability to withstand indefinite short circuit to ground or to any voltage level up to ±25 volts and it controls the slew rate (i.e. how fast the signal changes between levels) [14]

The RS 232 is soldered at the edge of the vero-board with about 50% (the part containing the 25 pins) of the device protruding outside the vero-board this is because it will aid easy connection to the computer unit, this protruding is visible outside the casing of the project

3.3.2 Max 232;

Next to the RS 232 is the MAX 232 IC. It is an integrated circuit that converts signal from RS 232 to signals suitable for use in TTL (transistor-transistor logic) compatible digital logic used by the counter [15]. The drivers provide RS-232 voltage level outputs (approx. ± 7.5 V) from a single + 5 V supply via on-chip charge pumps and external capacitors (as seen in the circuit diagram). This makes it useful for implementing RS-232 on the device (the queue manager) that otherwise do not need any voltages outside the 0 V to + 5 V range, as power supply design does not need to be made more complicated just for driving the RS-232 in this case [15].

The receivers reduce RS-232 inputs (which may be as high as ± 25 V), to standard 5 V TLL levels. These receivers have a typical threshold of 1.3 V, and a typical hysteresis of 0.5 V [15].

The receivers reduce RS-232 inputs (which may be as high as ± 25 V), to standard 5 V TLL levels. These receivers have a typical threshold of 1.3 V, and a typical hysteresis of 0.5 V [15].

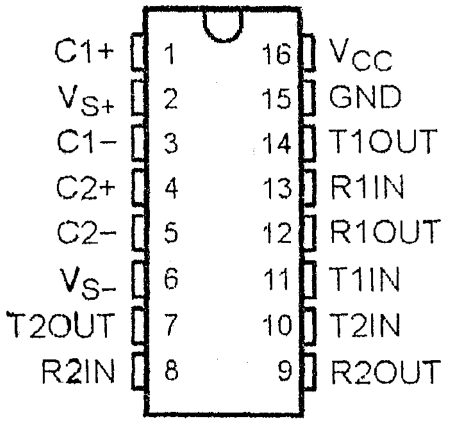

Fig 3.4: MAX-232 Chip

The MAX 232 is a dual driver/receiver that includes a capacitive voltage generator to supply the microcontroller from a single 5V supply. Each receiver converts the input to 5V TTL/CMOS levels. These receivers have threshold of 1.3V, a typical hysteresis of 0.5V, and can accept +/-30V inputs. Each driver converts TTL/CMOS input level into TIA/EIA-232-F levels [15].

3.3.3 The Microcontroller (AT89C52 Microcontroller):

There are different and diverse types of microcontrollers with different architectures, the list given below is of some few microcontrollers: Intel 8051, parallax propeller, Atmel AVR, PIC, Toshiba TLCS-870, NEC V850, Hitachi H8, Hitachi superH, Infineon Microcontroller, MIPS, Rabbit 2000, Zilog eZ8 etc [13].

The AT89C52 is a low-power, high-performance CMOS 8-bit microcomputer with 8K bytes of Flash programmable and erasable read only memory (PEROM). The device is manufactured using Atmel's high-density nonvolatile memory technology and is compatible with the industry standard 80C51 and 80C52 instruction set and pin out. The on-chip Flash allows the program memory to be reprogrammed in-system or by a conventional nonvolatile memory programmer. By combining a versatile 8-bit CPU with Flash on a monolithic chip, the Atmel AT89C52 is a powerful microcomputer which provides a highly-flexible and cost-effective solution to many embedded control applications [18].

The AT89C52 provides the following standard features:

- 8K bytes of in-system Reprogrammable flash memory.
- Endurance; 1000 write/Erase cycles.
- Full static Operation; 0Hz to 33 MHz
- Three level program memory lock.
- 256 × 8 bit internal RAM.
- 32 programmable I/O (Input/output) lines.
- Three 16-bit Timer/Counters.
- Eight interrupt sources.
- Programmable serial channels.
- Low-power Idle and Power-down mode.
- Watchdog Timer.
- Dual Data Pointer.
- Fast Programming Time.
- Flexible ISP Programming (Byte and Page mode).
- 4.0V to 5.5V Operating Range.
- Interrupt Recovering from Power-down Mode.

- Compactable with MCS-51 products.

3.3.4 Functionality in Controller Unit:

The AT89C52 is the main component in the controller unit, it receives voltage (5V) from the Power supply unit mainly at pin 40 (VCC), this voltage powers the microcontroller. Connected to pins 18 and 19 is the inverting oscillator amplifier crystal operating at 11.092 MHz shown in the figure below, this crystal gives accurate baud rate (data transmission speed) for serial communication. At pin 9 (RST/Reset pin) a 1µF capacitor is used in place a Schmitt-trigger circuit usually used to reset the microcontroller [13], (this circuit consists of a 1N933 diode in parallel to 10K ohm resistor, a 74HC14 NOT gate, a 10µF capacitors and a reset switch). This project doesn't require a reset button thus the 1µF capacitor is used which charges up to 5V and resets the microcontroller unit each time the device unit is turned on or powered up

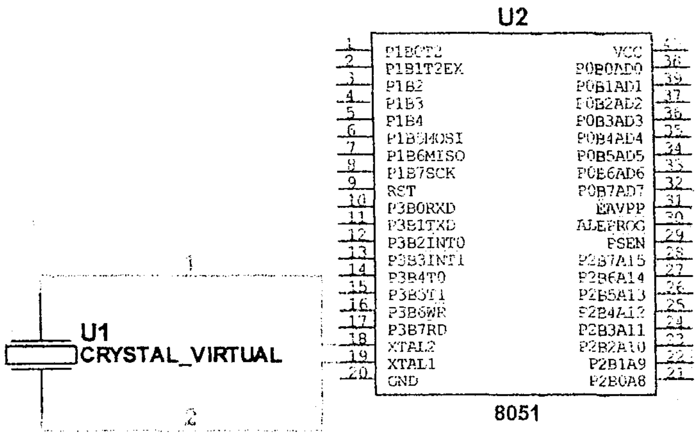

Fig 3.5: Inverting Oscillator Amplifier attached to the Microcontroller

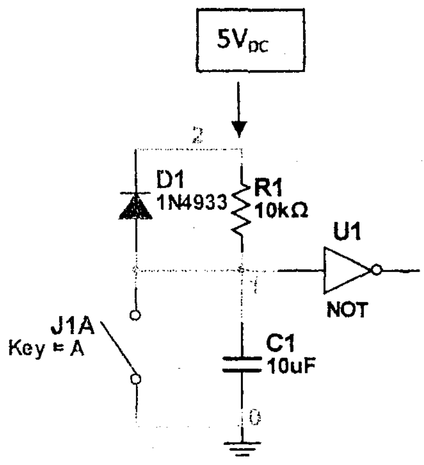

Fig 3.6: Schmitt-trigger reset Circuit

The power supply unit also supplies 5V to pin 31 (EA/External access) which must be kept high since the microcontroller is not booting from an external memory source. The microcontroller gets its input from the computer unit via the MAX 232 IC connected to pins 10 and 11 (P3.0 and P3.1 respectively). A pivotal function of the controller unit performed by the microcontroller is the storing of increments on the queue, recognition of the number of the student that is being attended to at a particular moment and the display of this values on the 16×2 LCD screen; all this is done by the microcontroller by receiving inputs or signals from buttons connected to pins 21, 22, 23 and 24 (ports P2.0, P2.1, P2.2 and P2.3 respectively) and outputting through pins 32, 33, 34, 35, 36, 37, 38 and 39 (ports P0.7, P0.6, P0.5, P0.4, P0.3, P0.2, P0.1, P0.0 respectively) connected to the LCD U2 (shown in the circuit).

3.3.5 Programming environment:

Traditionally most embedded systems and microcontrollers have been written in assembly language, however due to a variety of reasons most new microcontrollers are now expected to have a compiler available and are written in C language [17]. The programming language used in this project is the C programming language

A well written C program has the following advantages and these are the reasons the C programming language was chosen over the assembly language or any other programming language

1. It is more readable, they are very close to normal English language
2. It is more maintainable
3. They are non-procedural, it makes use of classes and methods that can be called at any point in the program
4. It is movable to other targets

The circuit design and Simulation was performed on Proteus. Proteus is software used for microprocessor simulation, schematic capture and printed circuit board (PCB) design. It is designed by Labcenter Electronics. The XGameStation micro edition was designed using Labcenter Proteus Schematic Entry and PCB Layout tools

The components of the Software include [16]:

1. ISIS Schematic capture: a tool for entering designs
2. PROSPICE Mixed mode SPICE simulation: industry standard SPICE3F5 simulator combined with a digital simulator

3. ARES PCB Layout: PCB design system with automatic component placer, rip-up and retry auto-router and interactive rule checking
4. VSM: Virtual System Modeling lets co simulate embedded software for popular microcontroller alongside hardware design
5. System Benefits: integrated package with common user and fully context sensitive help

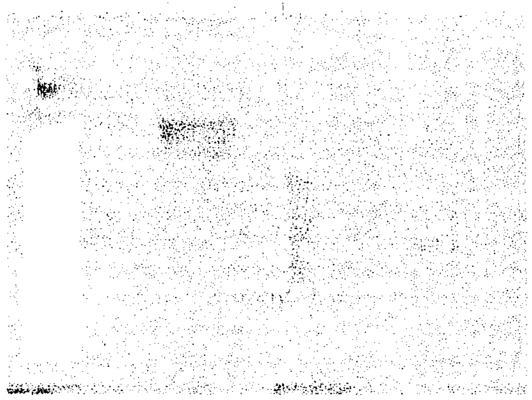

Fig 3.7: Proteus IDE

3.3.6 LCD (Liquid Crystal Display):

This is a thin, flat electronic visual display that uses the light modulating properties of liquid crystals (LCs), it is made up of any number of color or monochrome pixels arrayed in front of a light source or reflector, each column consists of a column of liquid crystal molecules suspended two transparent electrodes and two polarizing filters, the axis of polarity of which are perpendicular to each other [15]. The device can receive signals from the counter once any button is pushed and displays the result e.g. the increment or decrement in the line/queue, the maximum queue length allowable etc.

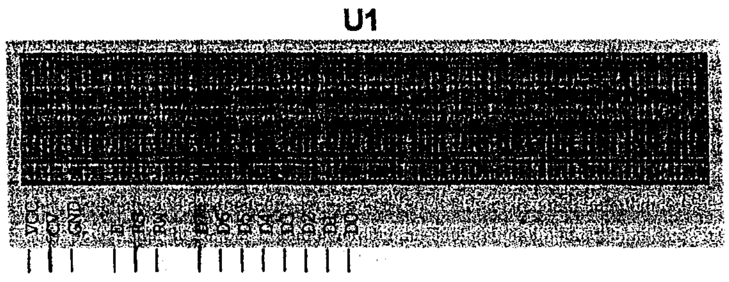

Fig 3.8: 16×2 Liquid Crystal Display (LCD)

The LCD uses a 16×2 segment display, it contains a power supply pin (VCC), its own oscillator to drive its segments, a contrast pin (CV) which allows for adjustment for best viewing under varying light conditions, viewing angles and temperatures, it contains other pins with various other functions, they are listed in the table below

The LCD also requires 3 control lines from the microcontroller [18]:

1. Enable (E): this line allows access to display through RW and RS lines. When this line is low, the LCD is disabled and ignores signals from RW and RS. When E line is high, the LCD checks the state of the two control lines and responds accordingly
2. Read/Write (RW): this line determines the direction of data between the LCD and microcontroller. When it is low data is written to the LCD. When it is high, a character is being read from the LCD
3. Register Select (RS): with the help of this line, the LCD interprets the type of data lines. When it is low, an instruction is being written to the LCD. When it is high, a character is being written to the LCD

Table 3.1: Pin Configuration of an LCD

Pin	Symbol	Input/output	Function
1	VCC	Input	Supply voltage (+5V)
2	CV	Input	Contrast adjustment
3	GND	Input	Signal ground
4	E	Input	Enable
5	RS	Input	Register select (1=data; 0=instruction register, busy flag/address counter)
6	RW	Input	Read (1)/write (0) select
7	D7	I/O	Data bit 7
8	D6	I/O	Data bit 6
9	D5	I/O	Data bit 5
10	D4	I/O	Data bit 4
11	D3	I/O	Data bit 3
12	D2	I/O	Data bit 2
13	D1	I/O	Data bit 1
14	D0	I/O	Data bit 0

3.3.7 DESIGN OF THE DISPLAY UNIT:

The variable resistor used on the 16×2 liquid crystal display LCD is to adjust the screen brightness (as stated above) which is connected between pins 1 and 2 (the VCC and CV respectively). For the LCD to be used, it has to be initialized i.e. some instructions has to be sent from the microcontroller to the LCD to set it ready to receive data, after this initialization then data can be sent

Sending data in the LCD are in two ways [15]:

1. 8-bit mode
2. One nibble mode

The 8-bit mode uses 8-bit of a port in the microcontroller to send data or instruction while the one nibble uses 4-bit to send data or instruction. In this project, the one nibble mode is employed therefore whatever is sent to the LCD is converted into 4-bits each before being sent one after the other

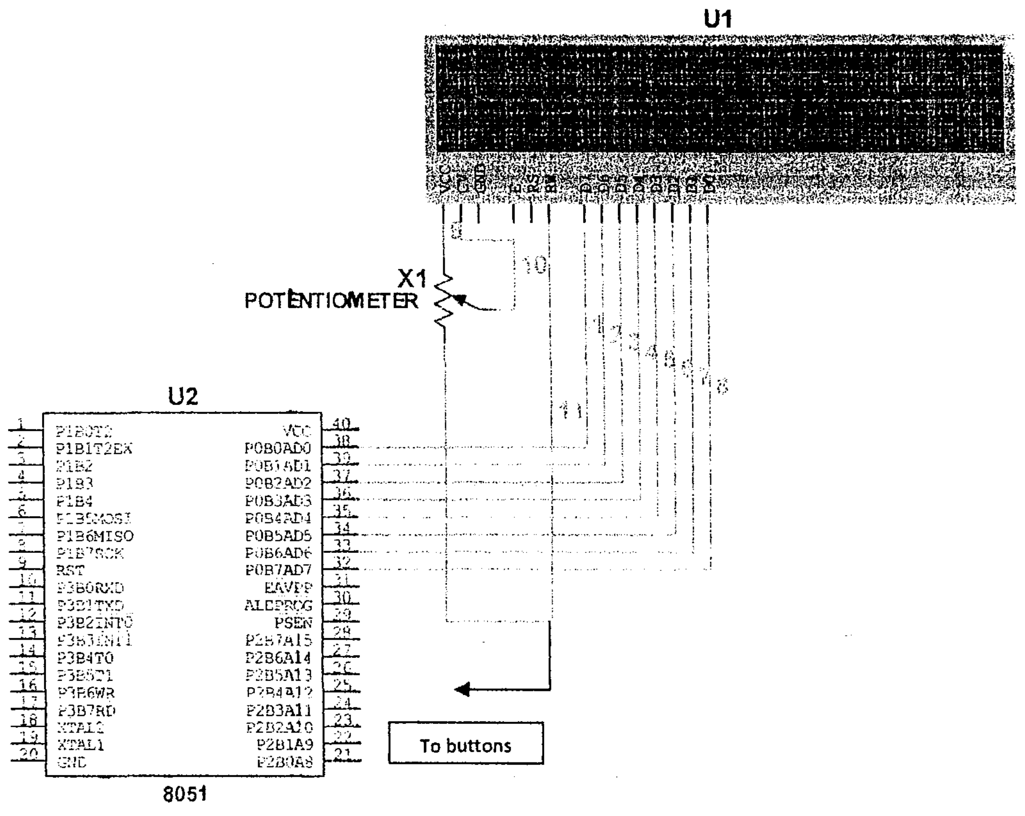

Fig 3.9: Interfacing the LCD with the microcontroller, circuit diagram

3.4.0 THE COMPUTER UNIT:

This unit is made up of the computer used in performing the actual registration of the students who have passed through the queue manager and the graphical user interface used by the computer to demonstrate the departmental registration.

3.4.1 REGISTRATION SOFTWARE DECRIPTION:

The Registration software posses the following features:

1. Contains Students data entry points for the various Levels
2. A management portal for staff use only
3. Efficient connection to suitable database, makes use of MySql database server

In programming the graphical user Interface, the following components were used:

1. Netbeans IDE (Integrated Drive Electronics) 6.7. with the following product features listed below

   Product Version: NetBeans IDE 6.7 (Build 200906241340)

   Java: 1.6.0_10-beta; Java HotSpot(TM) Client VM 11.0-b11

   System: Windows Vista version 6.1 running on x86; Cp1252; en_US (nb)

   The netbeans 6.7 is software that can be used to develop desktop, mobile and web applications with java, PHP, C/C++ and more. It runs on windows, Linux, Mac OS and Solaris. It is an open source product, free and can be downloaded from the internet from any of the following websites: www.netbeans.org, www.sun.org, www.java.net and any ORACLE corporation website. Other IDEs that can be used in compiling the GUI include JCreator, Notepad ++ etc.

Advantages of Netbeans 6.7 IDE:

The following are the advantages of the Netbeans 6.7 IDE; the reason why it was chosen for the purpose of this project:

a. Compatibility with java codes: the major programming language used in the programming of the GUI is java language; a 3rd generation, object-oriented, high level language. This IDE is very compatible with this language
b. Wider range of third party APIs: a wider range of more complicated and interesting projects can be carried out using this IDE. an important third party API used in this project is the *java communication API* used in establishing communication between the computer unit and the controller unit
c. Advanced database features

d. Drag and drop features

e. Object created on other graphics development software can be imported into the IDE e.g. button created on Photoshop and animations from KoolMoves and Adobe Flash

2. The programming computer: In programming the graphical user interface, due to the wide range of use of the IDE and programming language used many choice of computers can be used. In this project the computer system used was a HP 6735s model computer with the under listed features:

   a. Processor: AMD Sempron(tm) SI-40 2.00 GHz

   b. RAM: 2.00 GB

   c. System type: 32-bit Operating System

Requirements:

The following are the requirements needed in order to make the Graphical user interface:

1. Netbeans Integrated Drive Electronics (IDE)
2. Java development kit
3. Java Runtime Environment (RVM)
4. A Computer

These requirements were downloaded from the internet (from the above listed websites) they also come in updates and can be found in these websites

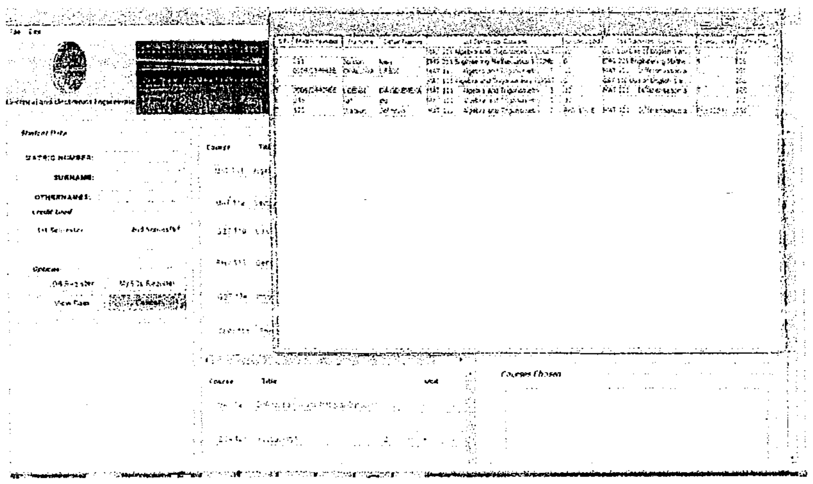

Fig 3.10: Registration GUI with Data Table

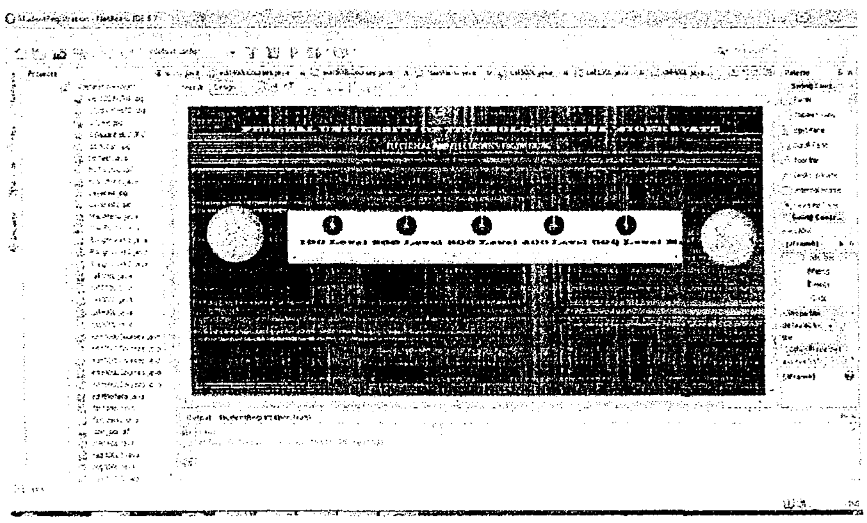

Fig 3.11 Netbeans 6.7 Integrated Development Environment (IDE)

3.4.2 INSTALLATION REQUIREMENT AND PROCEDURE

The software should be installed on a computer running on a 32bps or a 64bps speed, anything less than these may cause abnormalities in the running of the software.

The procedures for installation of the software are enumerated below;

1. Insert the CD into the CD-ROM of the system
2. Open the ECE REGISTRATION SOFTWARE folder
3. Install the MySQL server by following the instructions of the installer

4. Install the MySQL Administrator by following the instructions of the installer
5. Restoring the software backup: MySQL Administrator > Restore > Open Backup File > Go to directory on the CD > Select SW Backup > Restore.
6. You can now run the registration software.

NB: for any passwords during installations use *database*

3.5.0 THE PRINTING UNIT

The printing unit is the last part of the project, connected to the computer; it receives instructions from the computer to print a ticket for a person student who just joined the queue

CHAPTER FOUR

TESTING AND RESULTS

4.1.0 STAGES OF TESTING:

Different stages of testing was carried out on the project; the preliminary simulation and testing of the circuit on a circuit simulation software (Proteus) was the first stage of the testing – carried out even before actual construction of the circuit, then testing was done on bread board before being transferred to the Vero-board, the next step was to case the project and another process of final testing was carried out on the project. The graphical user interface of the demonstrative departmental registration was also tested all through its design

4.1.1 SIMULATION, AT89C52 PROGRAMMING AND DEBUGGING:

The Intel AT89C52 microcontroller was programmed according to the requirements of the project; the source code was carefully written to avoid logic error. The hex file from the source code was generated and transferred or burnt into the chip with the aid of a programmer. Before the burning of the hex file was burnt in the microcontroller, appropriate testing and debugging was carried out using the Proteus simulation application and due corrections were made to the coding

4.1.2 BREADBOARD TESTING:

The next stage of testing was carried out on the breadboard; this was also done before the burning of the hex file into microcontroller in case there were modifications to be made on the circuitry of the project. The main aim of the breadboard testing was to test the continuity, to calculate the ratings of the circuit and to fine out faults that are present in the circuit of the project; the instrument used in this stage is the multi-meter. Corrections were made in areas were faults were found. It was after this stage that the project was soldered and cased

4.1.3 SOLDERING, CASING AND POST-CASING TESTING:

After the breadboard testing was carried out the next stage was to solder the project on a Vero-board and cased, but before casing the project was also tested using the multi-meter to ensure appropriate current and voltage rating, at this stage no corrections were made because it was in accordance to the corrections and results gotten from the breadboard testing of the previous testing stage. Finally after the project had been cased it was tested with time successions this time around; to ensure proper workability of the project, this concluded the entire design process of the project.

4.1.4 GUI TESTING:

The demonstrative departmental graphical user interface (GUI) was also tested, due to the nature of the design of the GUI constant testing always required all through its design. The connection between the GUI and database bank was also tested.

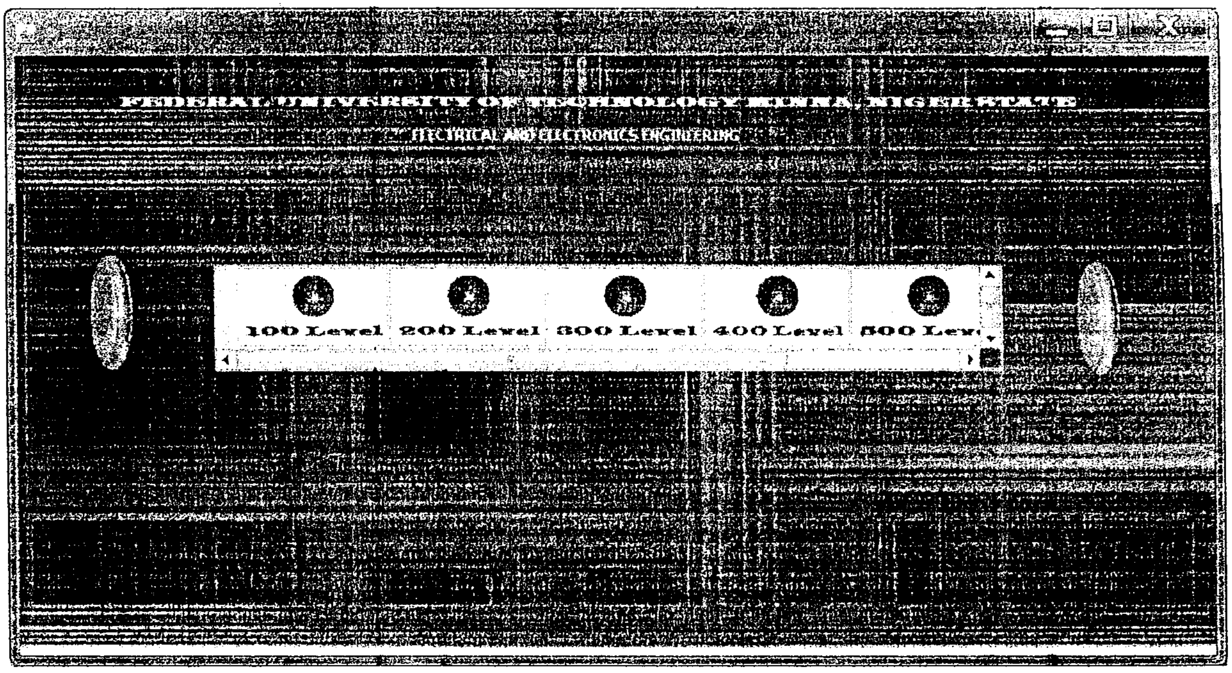
FEDERAL UNIVERSITY OF TECHNOLOGY MINNA, NIGER STATE
ELECTRICAL AND ELECTRONICS ENGINEERING
100 Level
200 Level
300 Level
400 Level
500 Lev

Fig 4.0: GUI during testing

4.2.0 PRECAUTION:

Throughout the design of this project the following precautions were taken while carrying out his project:

1. Proper and neat soldering was carried out to avoid unnecessary short circuiting on the board
2. Components soldered on the Vero board were spaced from each other to avoid complications
3. Appropriate testing of the project was carried out throughout the project work
4. The appropriate current and voltage rating was fed to the system
5. Appropriate ventilation was given to the casing frame in order to prevent overheating of the components

CHAPTER FIVE

CONCLUSION

5.1.0 PROBLEMS FACED:

The following were the problems faced while performing this project:

1. Liquid crystal display functions abnormally with breadboards, this contributed to delay during the breadboard stage
2. There was insufficient knowledge on how to Program the microcontroller, this contributed to delay
3. Creating communication between the device and the printer unit was a real challenge
4. Connecting the queue manager to a computer and establishing communication was a real challenge

5.2.0 RECOMMENDATION:

Since the project requires a lot of programming more time should be spent on this, creating communication between the device and computer GUI is a very vital part of the project and requires more attention, a suitable third party API should be applied in this area. As for the printing unit; communication is practically impossible to be established between it and the device except with the use of a computer, this should be looked into. Generally communication between the device and the computer should be established before final casing of the project

5.3.0 CONCLUSION:

The project was a successful one and in spite of the problems encountered, the project was an avenue for me to gain more knowledge on the issue of solving queue related problems with the aid of electrical, electronics and computerized means

REFERENCES

The following are the references for this project:

[1] en.wikipedia.org/wiki/Queueing_theory

[2] MOHAMMED FAZLI BIN ALIAS, Front Desk Customer Service for Queue Management System. Faculty of Electrical and Electronics Engineering, University of Malaysia Pahang

[3] www.freepatentsonline.com

[4] www.databyteindia.com

[5] www.ezinearticles.com

[6] Sanchez and Canton, Microcontroller: Programming the microchip PIC.

[7] Article on Java (Computer), Microsoft Encarta encyclopedia 2009, Microsoft Corporation

[8] http://web2.uwindsor.ca/math/hlynka/qhist.html

[9] www.mfn-consulting .com/html/historyAP.html

[10] OJEKUNLE SEYI, Design and construction of automated door opening and counting system. Federal University of technology Minna, November 2010

[11] B. L Theraja, A. K Theraja, A Textbook of Electrical Technology, S Chand and Company Ltd, Ram Nagar, New Delhi, 1999, pp 1123.

[12] B. L Theraja, A. K Theraja, A Textbook of Electrical Technology, S Chand and Company Ltd, Ram Nagar, New Delhi, 1999, pp 239.

[13] Jan Axelson, The microcontroller idea book; circuits, programs and applications featuring the 8052-BASIC microcontroller,

[14] www.en.wikipedia.org/wiki/RS-232

[15] www.en.wikipedia.org/wiki/LCD

[16] www.en.wikipedia.org/wiki/MAX-232

[17] Muhammad Ali Mazidi, Janice Mallispie Mazidi, The 8051 microcontroller and embedded systems,

[18] Richard Mann Imagecraft, How to program an 8-bit microcontroller, Atmel applications journal

APPENDIX

A. REGISTRATION SOFTWARE GRAPHICAL USER INTERFACE JAVA CODES

```
public class MainMenu extends javax.swing.JFrame {

  public MainMenu() {

    initComponents();

  }

  private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

    new call100L().show();

  }

  private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

    new call200L().show();

  }

  private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {

    new call300L().show();

  }

  private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {

    new call400L().show();
}

private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {

   new call500L().show();

}

private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {

   new manage().show();

}

public static void main(String args[]) {

   java.awt.EventQueue.invokeLater(new Runnable() {

       public void run() {

           new MainMenu().setVisible(true);

       }

   });

 }

 // Variables declaration - do not modify

 private javax.swing.JButton jButton1;

 private javax.swing.JButton jButton2;
private javax.swing.JButton jButton3;

private javax.swing.JButton jButton4;

private javax.swing.JButton jButton5;

private javax.swing.JButton jButton6;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel jLabel3;

private javax.swing.JLabel jLabel4;

private javax.swing.JLabel jLabel5;

private javax.swing.JLayeredPane jLayeredPane1;

private javax.swing.JPanel jPanel1;

private javax.swing.JScrollPane jScrollPane1;

private javax.swing.JToolBar jToolBar1;

// End of variables declaration
```

B. MICROCONTROLLER CODES

```
#include<reg51.h>
#include<stdio.h>
#include<intrins.h>
#define lcd_port P0
#define size 8
//#define t_base 250
#define scale 12.333333
sbit inc=P2^0;
sbit dec=P2^1;
sbit enq=P2^2;
sbit rs=P2^6;
sbit rw =P2^3;
sbit en= P2^5;
sbit alarm=P1^0;
sbit alarm2=P1^1;
sbit idi=P1^2;
unsigned int count=0;unsigned int wait=0;unsigned int max=1000;min=0;
void delay(unsigned int k)
{
unsigned int x,y;
for(x=0;x<k;x++){
for(y=0;y<800;y++);
                    }
}
```

```
/*
void serial_send(unsigned char *t)
{
while(*t!=0x0)
{
SBUF=*t;
while(!T1){   }
T1=0;
t++;
}}
 void init_serial()
{
SCON=0X50;
TMOD=0X20;
TH1=0XF3;
TR1=1;
T1=1;
EA=1;
ES=1;
}
 */
 void write(unsigned char c,unsigned char reg_select)
 {
 en=0;
 rw=0;
```

```
rs=reg_select;
lcd_port=c;
en=1;
en=0;
}
void lcd_data(unsigned char c)
{
        write(c,1);
        delay(1);
}
void lcd_cmd(unsigned char c)
{
        write(c,0);
        delay(1);
}
void clear_lcd(void)
{
        lcd_cmd(0x01);
}
void lcd_pos(unsigned char row, unsigned char pos)
{
        if(row==1)lcd_cmd(0x80+pos);
        if(row==2)lcd_cmd(0xc0+pos);
}
void lcd_string(unsigned char code *p
){
        while(*p)lcd_data(*p++);
}
void lcd(unsigned char data *ptr)
{
        while(*ptr)lcd_data(*ptr++);
}
void init_lcd(void)
{
        lcd_cmd(0x38);
        delay(1);
        lcd_cmd(0x38);
        delay(1);
        lcd_cmd(0x38);
        delay(1);
        lcd_cmd(0x0c);
        delay(1);
        lcd_cmd(0x01);
        delay(1);
        lcd_cmd(0x06);
        delay(1);
}
unsigned int update_count(void)
{
if(!inc)
```

```
{
if(count<1000)count++;
clear_lcd();
lcd_pos(1,0);
lcd_string("PLS WAIT PRNTING..");
idi=0;
delay(80);
//idi=1;
}
//if(!enq)
  alarm2=0;
  delay(50);
  alarm2=~alarm2;
  delay(50);
  if(count>5)
 //  alarm=0;
  //delay(150);
 // alarm=1;
  //delay(200);
  return (count);
 }
 void show_count(void)
 {
         unsigned char data lcd_buffer[10];
         unsigned int avg;
```

```
        unsigned int waiting_cnt;
        static unsigned int temp=0xffff;
        avg=update_count();
         if((!enq)&(count>5))
     waiting_cnt=min++;
        if(temp==avg)return;
        temp=avg;
                clear_lcd();
                lcd_pos(1,0);
                lcd_string("ATTENDING: ");
                sprintf(lcd_buffer,"%u",waiting_cnt);
                lcd(lcd_buffer);
                lcd_pos(2,0);
                lcd_string("LAST NO : ");
                sprintf(lcd_buffer,"%u",avg);
                lcd(lcd_buffer);
                delay(100);
        }
void show_logo(void)
{
        clear_lcd();
        lcd_pos(1,0);
        lcd_string("UDENSI D. EMEKA");
        lcd_pos(2,0);
        lcd_string("2006/24429EE ");
        delay(600);
        clear_lcd();
        lcd_pos(1,0);
        lcd_string("ELECT/COMPT.ENGR");
        delay(600);
        clear_lcd();
        lcd_pos(1,0);
        lcd_string("SUPERVISED  BY...");
        lcd_pos(2,0);
        lcd_string("MR ACHONU ADEJO.");
        delay(600);
        clear_lcd();
        lcd_pos(1,0);
        lcd_string("MCTRLLER. BASED");
        lcd_pos(2,0);
        lcd_string("QUEUE MANAGR&GUI");
        delay(600);
}
void sys_init(void)
{
        init_lcd();
        //show_ready();
        show_logo();
}
void show_ready(void)
```

```
{
        clear_lcd();
        lcd_pos(2,0);
        lcd_string(" SYSTEM READY!..");
        lcd_pos(1,0);
        lcd_string(" MAX COUNT=1000");
        delay(300);
}
void main(void)
{
        //      init_serial();
         sys_init();
        show_ready();
        while(1)
        {
          // serial_send();
                show_count();
        }
 }
```