

DESIGN AND CONSTRUCTION OF A COMPUTER-AIDED
AUTOMATIC WATER PUMP ACTUATOR

BY

ADI TERNGU

(97/5907EE)

A PROJECT REPORT SUBMITTED TO
ELECTRICAL AND COMPUTER ENGINEERING DEPARTMENT
SCHOOL OF ENGINEERING AND ENGINEERING TECHNOLOGY
FEDERAL UNIVERSITY OF TECHNOLOGY, MINNA
NIGER STATE, NIGERIA.

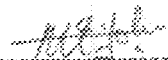
IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE
AWARD OF BACHELOR OF ENGINEERING [B. ENG] DEGREE IN
ELECTRICAL AND COMPUTER ENGINEERING

SEPTEMBER 2003


CERTIFICATION

This is to certify that this project work was carried out by Adl Terngu with registration number 97/5907EE under the supervision of Engr. Nwohu M. N. and submitted to the Electrical and Computer Engineering Department, Federal University Of Technology, Minna.

Engr. M.N. Nwohu
(Project Supervisor)

 22/10/15
Signature and Date

Engr. M.N. Nwohu
(Ag. Head of Department)

 22/10/15
Signature and date

(External Examiner)

Signature and date

DEDICATION

This piece of work is dedicated to God Almighty for his invaluable love, guidance and protection, and to my parents Mr. and Mrs. Linus S. Adi for their concerted efforts towards the successful completion of this project.

ACKNOWLEDGEMENT

I am very much indebted to my mother, who has always been there for me, supporting me morally, financially and spiritually. I cannot thank you enough. All other members of my family are not left out either, your individual and collective contributions are innumerable.

My deep appreciation goes to my supervisor, Engr. Nwohu M.N. for his kindness, attention, constructive criticism, advice and understanding. The precious time he spent in going through my manuscript is acknowledged. I am glad to have worked under him.

I wish to also acknowledge the enormous contribution of my HOD Dr. Y.A. Adediran and all the lecturers who taught me one ~~why~~^{or} the other.

My appreciation is further extended to Kunle Elijah, David Nmadu, Sesugh Anongo, Emem Etukudo, John Salibu, Odianosun Okosun, Solomon Baba, Henry Toby, Terver Kurugh Shagu Julius (Jnr.), Sesugh Beba, Teghtegh Zawua, and others that cannot be mentioned here due to lack of space, for their various contributions.

Finally, I would like to express my profound gratitude to God for his immeasurable blessings and mercy, which saw me through all endeavours during my academic pursuit at all levels.

TABLE OF CONTENTS

	Page
Title Page - - - - -	- i
Certification - - - - -	- ii
Dedication - - - - -	- iii
Acknowledgment - - - - -	- iv
Table of contents - - - - -	- v
Abstract - - - - -	- vii

CHAPTER ONE: GENERAL INTRODUCTION

1.1 Introduction - - - - -	- 1
1.2 Aims and Objectives - - - - -	- 2
1.3 Literature Review - - - - -	- 3
1.4 Project Outline - - - - -	- 5

CHAPTER TWO: SYSTEM DESIGN AND ANALYSIS

2.1 HARDWARE DESIGN - - - - -	- 6
2.1.1 Power Supply - - - - -	- 6
2.1.2 Water Detection Unit - - - - -	- 10
2.1.3 Input/Output buffers - - - - -	- 13
2.1.4 The Personal Computer - - - - -	- 14
2.1.5 Power/Cable Detection Unit - - - - -	- 18
2.1.6 Switching Unit - - - - -	- 18
2.1.7 LED Display Unit - - - - -	- 20

2.1.8 Alarm Unit	-	-	-	-	-	-	-	-	21
2.2 SOFTWARE DESIGN	-	-	-	-	-	-	-	-	23
2.2.1 Graphical User Interface (GUI)	-	-	-	-	-	-	-	-	24
2.2.2 Input/Output Routines	-	-	-	-	-	-	-	-	26
2.2.3 Keyboard and Mouse Routines	-	-	-	-	-	-	-	-	26

CHAPTER THREE: CONSTRUCTION AND TESTING

3.1 Tools and Materials	-	-	-	-	-	-	-	-	29
3.2 Hardware Construction	-	-	-	-	-	-	-	-	29
3.3 Project Casing	-	-	-	-	-	-	-	-	30
3.4 System Coupling	-	-	-	-	-	-	-	-	30
3.5 Construction Precautions	-	-	-	-	-	-	-	-	32
3.6 Testing and Results	-	-	-	-	-	-	-	-	32
3.7 Problems Encountered	-	-	-	-	-	-	-	-	33

CHAPTER FOUR: CONCLUSION AND RECOMMENDATION

4.1 Conclusion	-	-	-	-	-	-	-	-	34
4.2 Recommendation	-	-	-	-	-	-	-	-	34
References	-	-	-	-	-	-	-	-	35
Appendix A: Software GUI Screen Captures	-	-	-	-	-	-	-	-	36
Appendix B: Driver software	-	-	-	-	-	-	-	-	38
Appendix C: Complete Circuit Diagram of the Design	-	-	-	-	-	-	-	-	47

ABSTRACT

The Computer-Aided Automatic Water Pump Actuator is a system designed to monitor the level of water in a tank and operate the pump appropriately, it can also be used for any other liquid of resistance under $900k\Omega$ (i.e. between the maximum separation of the sensing probes). Another application of the device is to automatically switch ON/OFF any electric device connected to it, based on time sharing, thus giving it a wide range of application both in industries and at home.

It consists of two major parts: (1) the hardware, which connects to the microcomputer through the parallel port, and (2) the software, stored in memory on the computer, which drives the hardware and gives the system its high level of intelligence and versatility.

The hardware is designed using Medium Scale Integration (MSI) integrated circuits (ICs) and discrete components like resistors and capacitors, and could be divided into the following functional blocks: power supply unit, water sensing unit, input/output buffers, switching unit, light emitting diode (LED) indicators, alarm unit and the Personal Computer (PC).

The software is written in C++, one of the most powerful object oriented programming language available. It has rich Graphical User Interface (GUI) including buttons and mouse routines for easy usage, as is found in windows programming.

CHAPTER ONE

GENERAL INTRODUCTION

1.1 INTRODUCTION

Water is a basic necessity of life and most of the government organization strive to provide potable water to the public. However, despite concerted effort by government to make water available to all her citizens, the supply of water to most village and cities is grossly inadequate.

There are usually cases of water being supplied once or twice in a week and at low pressures that can hardly flow on the first floor in a storey building. This informs the decision of many homes, hospitals, industries etc to install pumps and overhead, surface or underground tanks for water storage.

The installation of pumps in buildings to pump water to an overhead tank for storage and redistribution requires an operator to monitor the water level in the tank, which may be tens of metres above the ground level, and operate the pump as appropriate. Besides the discouraging height at which the tank may stand, the monotonous nature of the work causes boredom and fatigue in human operators with subsequent deterioration in performance. Thus, the stress in operation causes a fall in efficiency since it is impossible to standardize the behaviour of human operators in handling manual control of water pump. Eventually, this led to the conception of the idea of use of microcomputer in the monitoring exercise and operation of the water pump to ease human operator of his problem.

The Computer-Aided Automatic Water Pump Actuator is designed to monitor the water level in a tank and switch the pump ON/OFF when the level of water reaches predetermined

levels in the tank, and alert the operator of possible dangers if any part of the system fails, thus eliminating the drawbacks associated with manual pump operation.

Furthermore, having interfaced the device with a microcomputer, it utilizes the high processing power of the computer, making the external hardware a simpler circuit than it could have been, and can also be adopted for other applications such as automatic control of other electrical devices like security lights, room heaters/coolers etc on time sharing basis. It also has the capability of controlling liquids other than water, which can conduct electricity and whose conductivity is within an acceptable range. This is very useful particularly in chemical/petrochemical industries.

1.2 AIMS AND OBJECTIVES

The design and construction of the Computer-Aided Automatic Water Pump Actuator is aimed at achieving the following goals:

1. It introduces the student to the practical application of the theory of courses taught in the classroom. Such courses as Analogue, Digital and Power Electronics, Microcomputer Hardware Technology and Programming, Circuit Theory, Laboratory Practical, etc.
2. It exposes the student to how hardware devices can be interfaced with the microcomputer and be programmed to perform desired tasks appropriately. It also helps the student to appreciate the efficiency of microcomputer beyond desktop publishing.
3. In another perspective, the device so produced is aimed at reducing the cost of labour for employers and monotony for operators, both in industrial and home applications.

1.3 LITERATURE REVIEW

Automation describes systems in which automatic or programmed devices can operate independently or nearly independently of human control. Such systems are designed to extend the capability of machines to perform tasks formerly done by human beings and to control sequences of operation without human intervention. In the fields of communications, aviation and 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. [4]

Essential to all automatic-control mechanisms is the feedback principle, which enables a designer to endow a machine with the capacity for self-correction. A feedback loop is a mechanical, pneumatic or electronic device that senses or measures a physical quantity such as position, temperature, size, or speed, compares it with a pre-established value, and takes whatever pre-programmed action necessary to maintain the measured quantity within the limits of the acceptable standard. An outstanding early example is the flyball governor, invented in 1788 by the Scottish engineer, James Watt, to control the speed of steam engine.

Russian Polzunov I. invented the first historical feedback system for water level control in 1765. It consisted of a float attached to a lever, which in turn controlled a valve. The system was used in a boiler where the float detects the water level and controls the valve that covers the water inlet in the boiler. [4]

Since then, several other approaches to controlling water flow in a container have been developed. One of such is the float switch contact used to operate a motor that pumps water into a tank. Float switches are generally designed with two sets of contacts: (1) Normally open (NO) and (2) Normally closed (NC). Since the pair of normally open and

normally closed contacts of the float switch can either open or close contacts with changes in liquid level. variations in the application of the control device are possible. For example, when normally closed contacts are used, the pump motor will continue to pump water until the water level rises high enough to cause contacts to open and switch off the pump. The installation of such systems are however clumsy. [3]

Magnetic control has been used in some places in order to separate the control equipment from the pump motor. Although this approach is more expensive than the float contact switch, it is neater and satisfies certain industrial and commercial installation requirements (i.e. certain electrical control equipment should be located in one area away from the load device). Simple Electromagnetic devices like solenoids, contactors and magnetic motor starters are used to effect the controlling process.

With the introduction of the first microprocessor by Intel Corporation in 1971 and first personal computer (PC) by IBM ten years later (1981), control systems have been developed as dedicated equipment incorporating a micro-controller or as peripherals interfaced with the PC where the PC does most of the processing work and issues out control commands as pre-programmed. These systems greatly facilitate the use of feedback loops in control processes. The Computer-Aided Automatic Water Pump Actuator is one of such systems; it senses the water level through the combination of sensing probes and IC's. The signals from the sensing unit constitutes the feedback loop which is sent into the microcomputer for processing and comparison against pre-set values and consequently, the controlling software sends out control signals to switch ON/OFF the pump, alarm and indicator LEDs.

1.4 PROJECT OUTLINE

Chapter one: This chapter gives general overview of the project. The general introduction to the project, literature review that highlights previous works on the subject. Aims and objectives of the project are also contained in this chapter.

Chapter two: This contains the detailed design, analysis and considerations. The principles of operation and design calculations of all the units that make up the system are discussed in this chapter.

Chapter three: This chapter covers the details of construction and testing procedures employed to achieve the final product. It spans soldering of components on the veroboard, case construction, testing, troubleshooting and results obtained. Difficulties encountered are also listed here.

Chapter four: Contains the conclusion drawn from the results of testing, with reference to the objectives and goals of the project, and recommendations.

Reference, which lists the books and materials consulted and appendices that consist of final circuit diagram of the project, screen captures of control software graphical user interface (GUI), and program codes, follow chapter four as the concluding part of this project report.

CHAPTER TWO

SYSTEM DESIGN AND ANALYSIS

The design approach of Computer-Aided Automatic Water Pump Actuator is simple. The system is made up of two major sections-the hardware and the software. The discussion is thus, grouped into these sections.

2.1 HARDWARE DESIGN

The hardware comprises all the physical components of all units that make the system. The design therefore, involves the analysis, calculations and considerations that led to the choice of components for each unit.

2.1.1 POWER SUPPLY UNIT

All electronic devices require a direct current (dc) voltage source to operate. Very often, a circuit that converts the readily available alternating current (ac) supply to dc voltages is used. This system uses one of such circuits to provide the required voltages for operation. Figure 2.1 shows a block diagram of the power supply unit and the waveform at each stage.

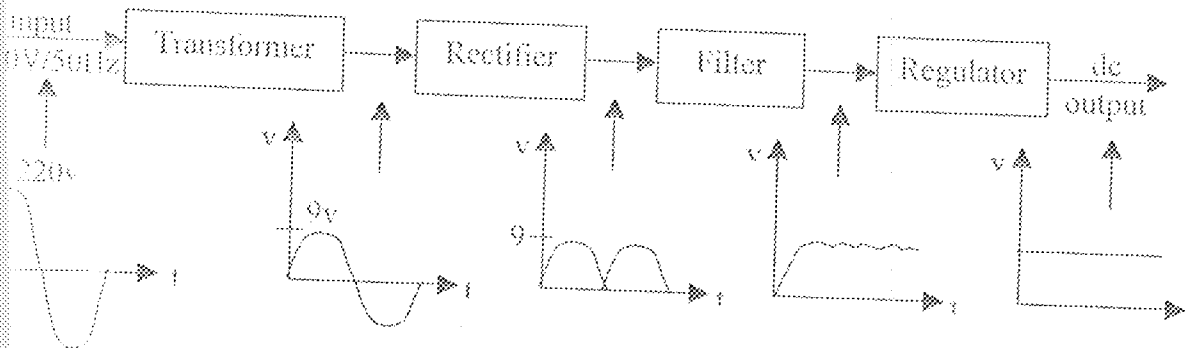


Fig. 2.1 Power supply block diagram and waveforms.

The Transformer

The first stage of the power supply design involves the stepping down of the 220V ac from the mains to about 9V ac with the aid of 220V/9V, 500mA transformer, whose current capacity is enough to drive the entire circuit.

The transformer is an electrical device that provides physical isolation between the 220V ac mains and the rest of the hardware. The only link is by means of magnetic flux, thus eliminating the risk of shock. It consists of two coils, the primary (input) winding and secondary (output) winding. Figure 2.2 shows the circuit symbol of a transformer. The ratio of the primary voltage V_1 to that of the secondary V_2 is equal to the number of turns in primary n_1 winding to that in the secondary winding n_2 :

$$\frac{V_1}{V_2} = \frac{n_1}{n_2} \quad \text{--- (2.1)}$$

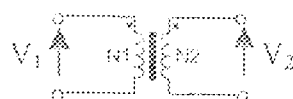


Fig. 2.2 Transformer circuit symbol.

A fuse (3A) is incorporated at the primary side to protect the transformer and the rest of the circuit from excessive current from the mains.

The Rectifier

The rectifier converts the 9V ac voltage from the transformer into a pulsating dc voltage and the process is called rectification.

A full-wave bridge rectifier is used for the rectification, it consist of four IN4001 diodes in the arrangement shown in figure 2.3 below.

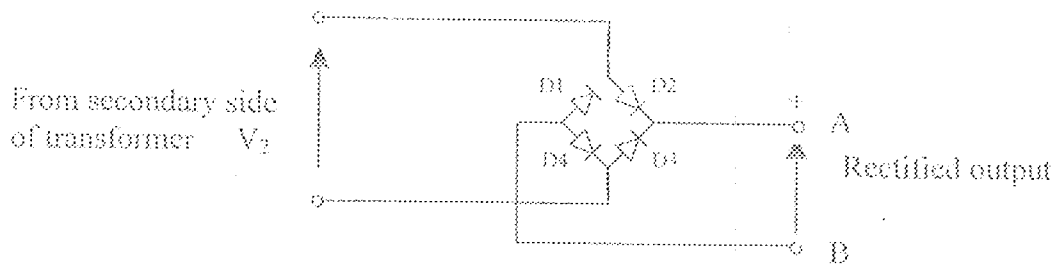


Fig. 2.3 Full-wave bridge rectifier.

During the positive half cycle, diodes D2 and D3 are forward biased and current flows through any load connected at terminals AB. In the negative half cycle, diodes D1 and D4 are forward biased. Since the load current is in the same direction in both half cycles, the full-wave rectified signal appears across the load. The average dc voltage V_{dc} across AB is:

$$V_{dc} = \frac{2V_{2(\text{peak})}}{\pi} = \frac{2\sqrt{2}V_{rms}}{\pi} = \frac{2 \times \sqrt{2} \times 9}{\pi} = 8.10V$$

Where $V_{2(\text{peak})}$ and V_{rms} are the peak output and the root mean square voltages of the secondary winding of the transformer respectively.

The diodes were so chosen such that their peak inverse voltage (PIV) rating is greater than $V_{2(\text{peak})}$, so that they do not break down when reverse biased.

The Filter

The pulsating dc voltage from the rectifier is only suitable for limited applications such as charging batteries and running dc motors. Most electronic circuits require dc voltage that is constant in value. A filter is used to convert the full-wave rectified signal into a constant dc voltage.

Capacitive filtering is adopted in this design where a large electrolytic capacitor is connected across the rectifier output. The capacitor charges up during the diode conduction period to the peak value, and when the rectifier voltage falls below this value, the capacitor

discharges through the load, so that the load receives almost steady dc voltage. The discharge time constant, which is the time taken for the capacitor to drop to 33% of the peak value is given thus:

$$\tau_d = R_L C \text{ ----- (2.2)}$$

Where R_L is load resistance, C is capacitance.

Since R_L is a constant for any given circuit, it follows that the larger the C , the smaller the ripple voltage. A 2200 μ F, 25V capacitor was chosen for this circuit, which is large enough for the intended purpose.

The Regulator

The output voltage of the filter capacitor varies when load current or the input voltage varies. This effect is also undesirable.

Two monolithic voltage regulator IC chips, 7806 and 7805 are used to supply steady 6V and 5V to drive the switching unit and energize the rest of the circuit respectively. These regulator chips supply the rated voltage with a wide range of voltage input (7V to 35V) and variations in the load current. Small capacitors (1 μ F) are connected at their outputs to filter out any ripples left on the supply line. Figure 2.4 is a complete circuit diagram of the power supply unit.

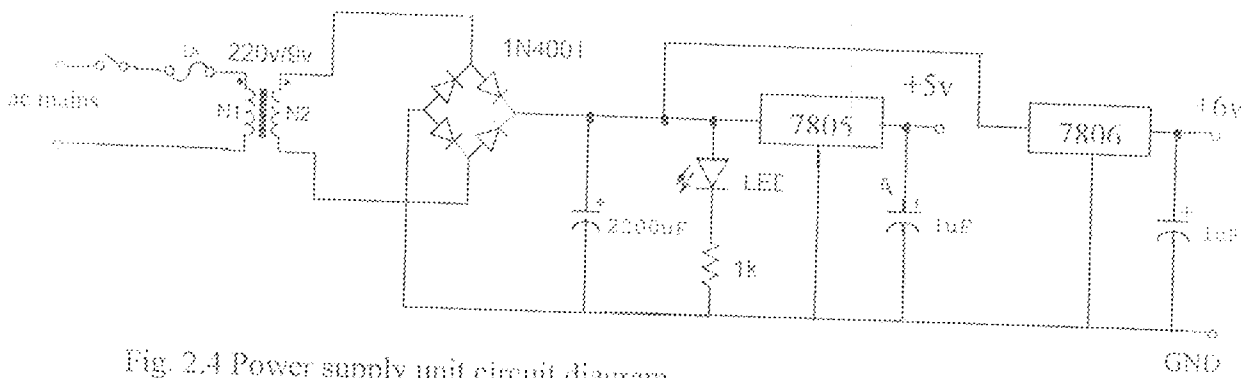


Fig. 2.4 Power supply unit circuit diagram.

2.1.2 WATER DETECTOR UNIT

The Computer-Aided Automatic Water Pump Actuator uses electrical transducers to convert water level in the tank into electrical signals. The electrical signals in this case are dc voltage levels (LOW and HIGH). The transducers used are probes which are simply conductors lowered into the tank at desired levels, with the common 'COM' probe at the bottom. They operate based upon the conductivity of water.

All aqueous solutions conduct electricity to various degrees. The conductivity of liquids varies with temperature, volume and separation height of the measuring probes. Tap water, for instance, has a conductivity of about 50 μ S/cm at 25°C (equivalent of 20k Ω /cm). Table 2.1 lists typical conductivity for various solutions at 25°C.

Four probes are used in arrangement shown in figure 2.5. The common 'COM' probe at the bottom of the tank is connected to the circuits ground. The LOW probe is suspended at the minimum desired water level in the tank; the HIGH probe is placed at maximum desired water level, while the overflow probe is placed at a level that would indicate possibility of water spillage from the tank.

Table 2.1. Typical conductivity of solutions at 25°C.

Solution	Conductivity (μ S/cm)	Solution	Conductivity (μ S/cm)
Ultra pure water	0.055	Ground water	20,000
Boiled water	1.0	1.0M KCl	111,342
Tap water	50	10% NaOH	355,000
Sea water	50,000	10% Sulphuric acid	432,000

(Sensorex: www.sensorex.com/conductivity.htm)

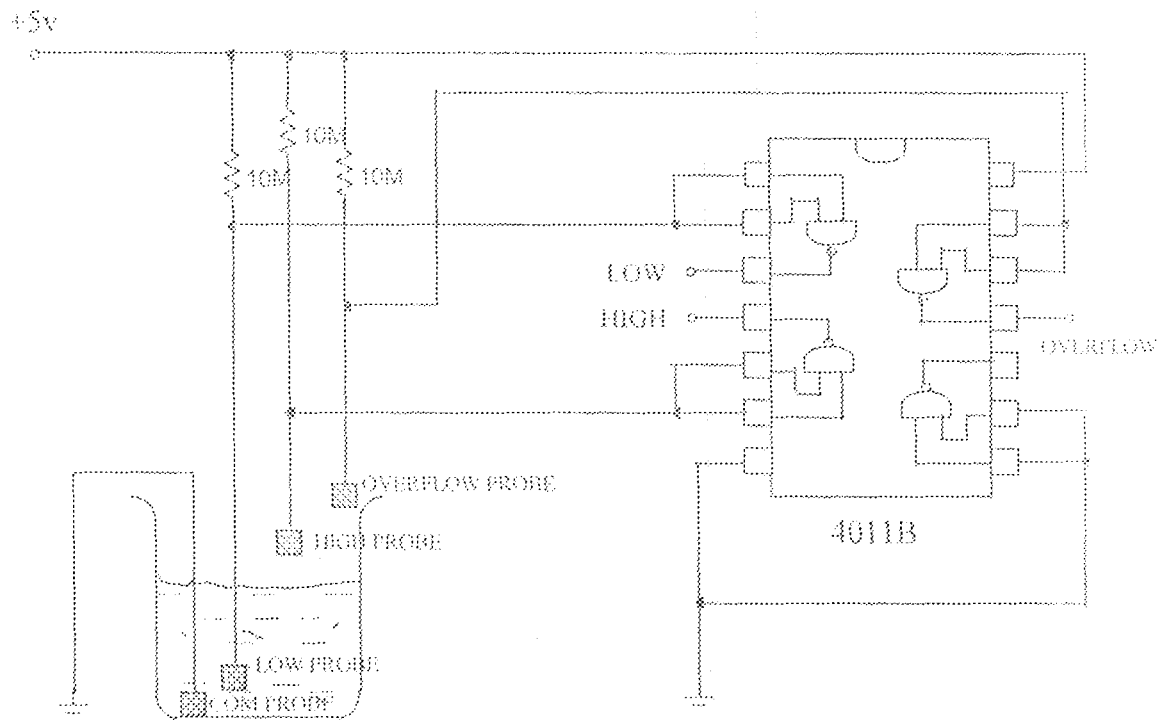


Fig. 2.5 Water detection unit's circuit diagram.

The two inputs into each NAND gate in the 4011B chip are connected together, making the gate act as an inverter. If any probe is out of water, Vcc is linked through the 10MΩ resistor to the input of the NAND gate to which the probe is connected, which is inverted to produce a LOW output, indicating that the probe is out of water. When the water level rises and touches any probe, the input to which the probe is connected is effectively linked through water to the COM probe, which is at ground potential. The input is thus grounded, and when inverted produces a HIGH output to indicate the probe is in water. In reality, the 10MΩ resistor and the resistance of water between the probe in question and the COM probe constitute a voltage divider across the input to the NAND gate, thus V_{in} gives the voltage at the input pin when a probe is in water.

$$V_{in} = \frac{R_w}{R_w + 10M\Omega} \times 5V \quad \text{----- (2.3)}$$

Where R_w is the resistance of water between the probes. As earlier stated, the resistance of water R_w depends on the distance of separation of the probes, hence there is an upper limit to the separation height between each probe and COM. This limit depends on the fixed resistance and the maximum voltage $V_{IL(max)}$ required to provide a valid LOW input for the 4011B chip. From the data sheet of the chip, $V_{IL(max)} = 1.5V$. This CMOS chip is actually chosen to provide this large margin as compared to TTL's 0.8V. The maximum water resistance (maximum separation height between OVERFLOW and COM), for a given liquid at a given temperature can be evaluated by substituting $V_{IL(max)}$ in equation (2.3) thus.

$$R_{w(max)} = \frac{10 M\Omega \times V_{IL(max)}}{5 - V_{IL(max)}} = \frac{10 M\Omega \times 1.5}{5 - 1.5} = 4.29 M\Omega$$

For tap water at 25°C, maximum height of tank in which the system would function

optimally would be: $\frac{4.29 M\Omega}{20 k\Omega/cm} = 214.3 cm \approx 2m$

The truth table below shows all the possible combinations of the states of the probes and expected response from the computer.

Table 2.2 Truth table of the system.

Overflow	HIGH	LOW	Pump	Alarm
0	0	0	ON	OFF
0	0	1	RETAIN	OFF
0	1	0	X	X
0	1	1	OFF	OFF
1	0	0	X	X
1	0	1	X	X
1	1	0	X	X
1	1	1	OFF	ON

RETAIN means the pump's present status should be maintained (i.e. if it is off, it should remain off until the water level falls below the low probe and if it is on, it should

continue to pump until the water level reaches the high probe). X stands for "DON'T CARE". Under normal operations, such states would never occur. They are therefore used to flag water detection errors.

2.1.3 INPUT/OUTPUT BUFFERS

A buffer is any logic circuit designed to have a greater output current and/or voltage capability than an ordinary logic circuit. Input/output buffers are very important when interfacing an external device with the computer. They give electrical protection to both the computer and the device in the event of current or voltage surge. Again, in a computer system where several devices use a common data bus, the outputs of the buffers are disabled when the processor is not reading data from the device, so as to avoid data conflict (jam) on the bus.

Two copies of 74LS125, a tri-state buffer chip with three possible states (low, high and high impedance), are used as input and output buffers. The 74LS125 chip contains four non-inverting tri-state buffers; the structure of one is shown in figure 2.6.

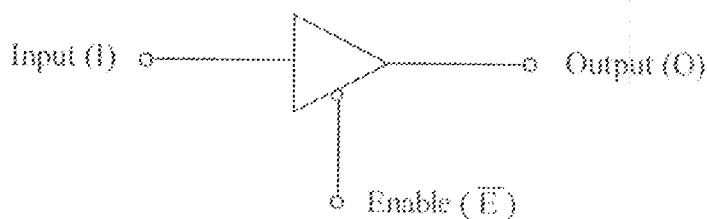


Fig. 2.6 Non-inverting buffer with active low enable.

The water level signals from the water detection unit are connected to the inputs of the buffers while the control signal to enable and disable the input buffers come from the computer.

No special interfacing circuitry is required to connect the CMOS 4011B output to the TTL 74LS125 inputs because each output is driving only one input. And from the data sheets, the output of 4011B can source 0.4mA in the high state and sink 0.4mA in the low state, while the 74LS125 input can sink 20µA in the high state and source 0.4mA in the low state. Also, the output voltages of the 4011B are 4.95V(min) in the high state and 0.05V(max) in the low state, which are just suitable for driving the 74LS125 in a predictable manner.

The signals from the computer to control the switching unit, alarm and LED displays are passed through the output buffers. The enable inputs of these buffers are hard-wired active by grounding them so that the output is always either HIGH or LOW. However, since the inputs of these buffers are driven from the computer, if the cable is not (properly) connected, these inputs will be open. An open TTL input always assumes a high state, which will enable all the units the buffers are driving. This is taken care of by biasing these inputs low through a resistor connected between them and ground. The resistor is chosen such that the voltage drop across it by the current from the input in the low state I_L will not exceed the maximum allowed voltage for a low state $V_{L(max)}$. Thus, the largest value of R is given by:

$$R_{(max)} = \frac{V_{L(max)}}{I_L} = \frac{0.8V}{0.4mA} = 2k\Omega \quad (\text{Values taken from data sheet}).$$

1.8kΩ resistor was chosen. [5]

2.1.4. THE PERSONAL COMPUTER (PC)

This unit forms the heart of the Computer-Aided Water pump Actuator; the microcomputer is used as a control tool. The digital signals from the input buffers representing the status of the probes are transferred into the microcomputer through the parallel port. The microcomputer then interprets these signals as guided by the control

software, and gives out the control signals to operate the pump, alarm and indicator LEDs (as desired). The computer is also capable of detecting power availability in the external device and proper cable connection.

The Parallel Port (Standard)

The parallel port was originally designed by IBM Corporation to drive printers. The standard parallel port is accessible through a 25-pin D-type female connector at the back of the PC. The sole purpose was to interface with the de-facto centronics printer. Instead of building a clean direct interface that relied on software to invert signals they used hardware inverters in a most unusual fashion. But it is the standard, anyone who intends to use the port for one's application must understand the structure. The parallel port has been used to interface many commercial and home built devices with the computer.

It was chosen for this project instead of the serial port for the following reasons:

1. A byte of data or more can be simultaneously transferred into the computer unlike the serial RS-232 port that transfers one bit at a time.
2. Interfacing using the parallel port is relatively easier and requires less external hardware.

DOS supports up to three parallel ports, which are assigned the following handles; LPT1, LPT2 and LPT3. Each port requires three consecutive input/output addresses to select all the registers (also called ports), these are Base, Base+1 and Base + 2 addresses. [10]

The parallel port has three commonly used base addresses; these are listed in table 2.3. LPT1 is normally assigned base 378h while LPT2 is assigned 278h, however this may not always be the case. The actual values on each machine can be checked from a lookup table provided by BIOS. When BIOS assigns addresses to printer devices, it stores the addresses at

specific locations in memory, so they can be found using appropriate program codes. Table 2.4 shows addresses in BIOS lookup table. [10]

At base address, 8 bits are available as outputs on pins 2 to 9; this is called the data port. The output is latched with the input/output (I/O) write pulse and is always active. The original IBM card used a 74LS3374 tri-state device that has its output enable pin hard-wired active. This means the port is strictly an output port. However newer ports have bi-directional feature that allows the input of data from the external world through this same port. This is not utilized in this project anyway.

At base + 1, there are five input bits from D3 to D7; this port is called the status port. The first three bits D0 to D2 are reserved, and therefore not available on the DB-25 connector. They are gated on the bus with I/O read pulse. Bit D7 (pin 11) is hardware inverted; software can invert this bit if necessary. Bit D6 (pin 10) can also be used to generate hardware interrupt.

At base + 2, called the control port, several options exist. It is a four bit output or four bit input or can be configured as any mixture of input and output. This is possible because the output is open collector. They are available from D0 to D3, all of which are hardware inverted except D2. D5 or D6 is used internally to control bi-direction of the DATA port. D7 is reserved. Some cards have the open collector outputs pulled up to Vcc internally. A 4.7kΩ resistor is usually used externally to pull-up these pins, just in case there is no internal pull-up.

In summary, the parallel port is capable of 8 to 12 output bits and 5 to 9 or 5 to 17 input bits for uni-directional and bi-directional ports respectively. All inputs and outputs behave as TTL devices.

Table 2.3 common base addresses for parallel ports.

Base address	Comment
3BCb	Used for parallel ports that were incorporated onto video cards.
378b	Usual address for LPT1
278b	Usual address for LPT2

Table 2.4 LPT addresses in BIOS data area.

Start address	Function
0000:0408	LPT1's Base address
0000:040A	LPT2's Base address
0000:040E	LPT3's Base address

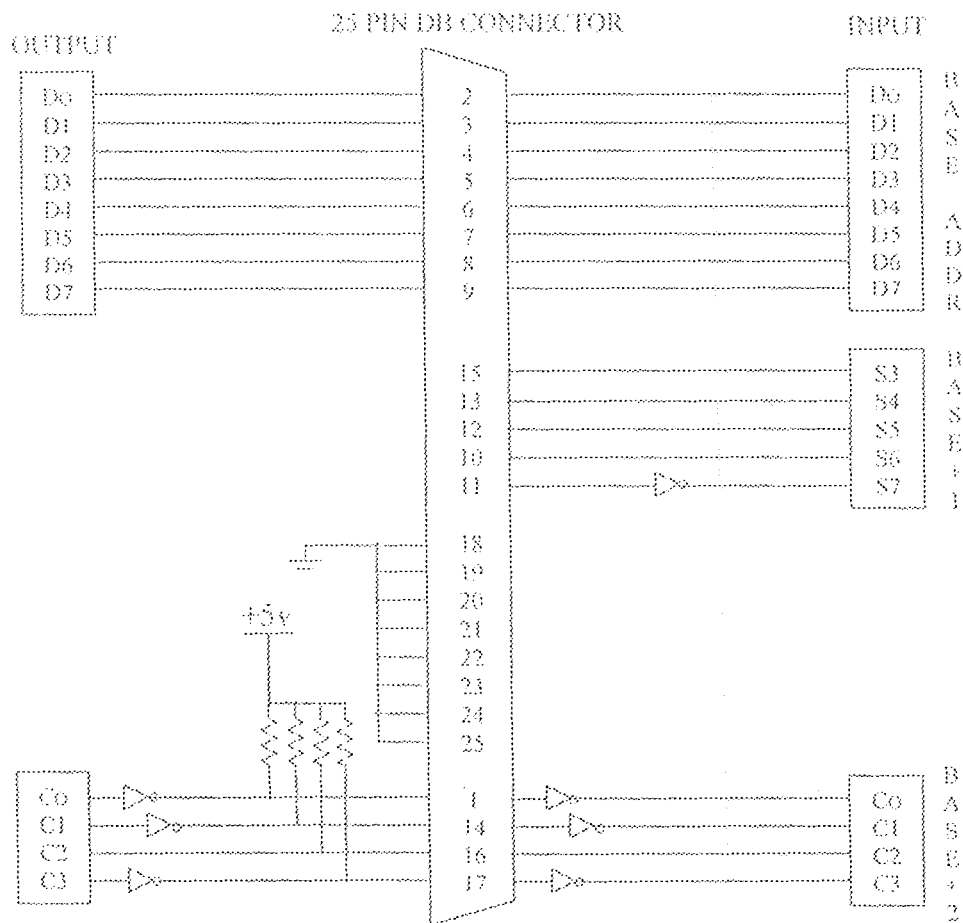


Fig. 2.7 Parallel port hardware.

2.1.5 POWER/CABLE DETECTION UNIT

When interfacing an external device with a microcomputer, it is usually necessary to include some circuitry that will enable the microcomputer to know whether there is power supply to the external device and also if it is properly connected to the parallel port or not. This improves the efficiency and reliability of the system.

In the Computer-Aided Automatic Water Pump Actuator, bit 6 of the STATUS port (pin 12) was grounded directly while bit 7 (pin 11) was grounded through a C9013 transistor switch whose base current is supplied by the supply voltage V_{cc} through a 1k resistor as shown in figure 2.8. When the cable is not (properly) connected, these pins will assume 01 status. If the cable is connected but the power is off, they are 00. Finally, they assume 10 when the cable is connected and the power is on. Hence, the exact problem is detected and the operator alerted. No further processing takes place until the fault is cleared.

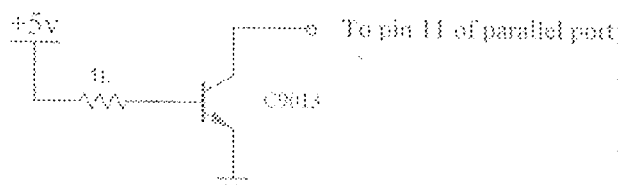


Fig. 2.8 Power detection unit.

2.1.6 SWITCHING UNIT

This unit performs the major task of switching the pump ON and OFF. It is driven from pin 2 of the parallel port. The power supply for this unit is +6V, basically because of the rating of the relay used. Transistor switching is used to provide the current necessary to energize the relay coil. Two general purpose transistors, 2N2222A in the arrangement shown in figure 2.9 eliminates the possibility of false triggering due to reverse bias current, and improve the switching speed.

The relay is an electromagnetic switch. It has an inductor coil which when energized closes a switch, and opens it when the energizing current is removed. The relay is used in this design such that +6V dc voltage is used to switch on 220V ac supply, which drives the pump. If the inductor coil is energized, current flows through the coil storing energy in it. When the current is removed, the stored up energy must drop rapidly to zero depending on coil resistance, producing negative voltage transient in the inductor. If an alternate path were not provided, this excessive current would destroy the transistors. A freewheeling 1N4001 diode is connected across the coil to conduct away this current and thus protect the transistors. The relay rating is 6V, 100Ω, 10A.

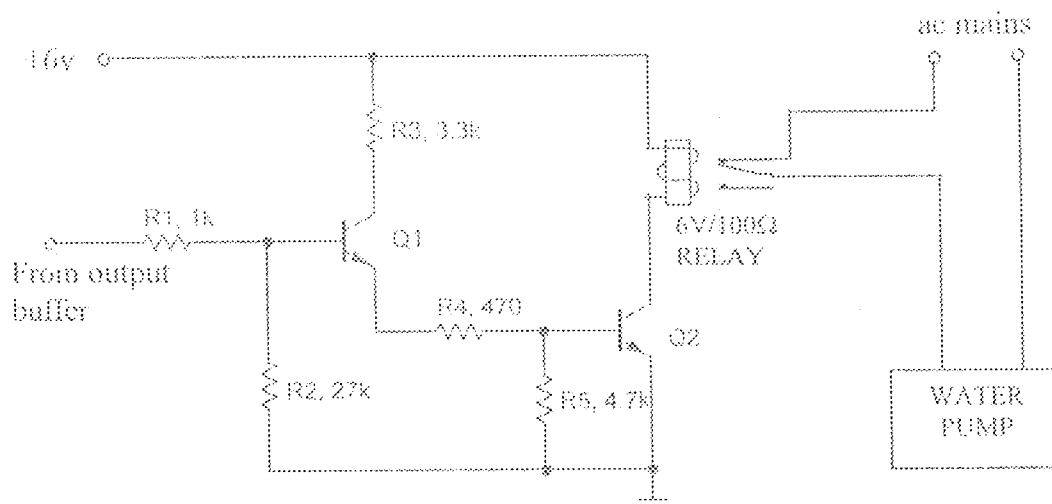


Fig. 2.9 Switching unit's circuit diagram.

The calculations for the unit are as follows:

The minimum HIGH output voltage from the output buffer: $V_{OH(min)}$ is 3.5V (data sheet)

R1 and R2 form a voltage divider across the base of transistor Q1 that gives base voltage V_{B1} of:

$$V_{B1} = \frac{R2}{R1 + R2} \times V_{OH(min)} = \frac{27k\Omega}{(27 + 1)k\Omega} \times 3.5V = 3.375V$$

The emitter voltage V_{E1} of transistor Q1 is:

$$V_{E1} = V_{B1} - V_{BE} \quad (V_{BE} = 0.7V \text{ - diode drop})$$

$$V_{E1} = 3.375 - 0.7 = 2.675V$$

R4 and R5 constitute voltage divider across the base of transistor Q2, the base voltage V_{B2} is:

$$V_{B2} = \frac{R4}{R4 + R5} \times V_{out(max)} = \frac{4.7k\Omega}{(4.7 + 0.47)k\Omega} \times 2.675V = 2.43V$$

The base current is:

$$I_{B2} = \frac{V_{B2}}{R4 + R5} = \frac{2.43V}{(470k + 0.47)k\Omega} = 5.2\mu A$$

The collector current is:

$$I_{C2} = h_{FE} I_{B2} \quad (h_{FE} \text{ - transistor current gain} = 239 \text{ as read from the multimeter})$$

$$I_{C2} = 239 \times 5.2\mu A = 1.2 \text{ mA}$$

The maximum current that can flow in the collector is:

$$I_{C2(max)} = V_{CC}/R_C \quad (R_C = 100\Omega \text{ - coil resistance})$$

$$I_{C2(max)} = 6/100 = 0.6 \text{ mA}$$

Since $I_{C2} \gg I_{C2(max)}$, the transistor will be driven into hard saturation when turned on and cut-off when turned off. [1]

2.1.7 LED DISPLAY UNIT

This unit shows the relative level of water in the tank. It consists of three light emitting diodes (LEDs), orange, green and red. When the water level is below the LOW probe, the orange LED is switched on. When the water level is between the LOW and HIGH probes, the green LED is on, while the red one is turned on when the water level is above the

HIGH probe. All of them are switched on if the water level reaches the OVERFLOW probe. The orange and the green LEDs will be turned on when there is pump failure. In the automatic device control mode, only the green LED is used. It is turned on if the device is on and off when the device is off.

2.1.8 ALARM UNIT

This is designed to give an audible tone upon the occurrence of a serious problem in the system; like possibility of water spillage from the tank and pump failure. Once the alarm is triggered, it remains on until it gets the attention of the operator who will then turn it off by simply acknowledging the accompanying error message displayed on the screen, using the appropriate command (click OK or press ENTER key).

The alarm unit is designed using a commonly available 555-timer chip, configured to operate as an astable multivibrator. It produces a high frequency signal filtered by a capacitor and sent to an 8Ω speaker. The pin-out diagram of this timer chip is shown figure 2.10

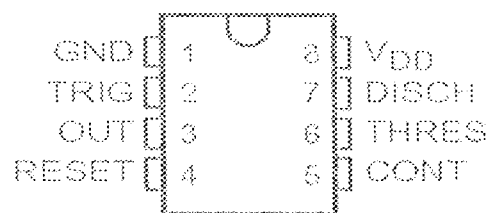


Fig 2.10 Pin-out diagram of the 555-timer chip.

The 555-timer has a trigger level of approximately one-third of the supply voltage and a threshold voltage level equal to approximately two-third V_{cc} . These levels can be altered by use of the control voltage terminal (CONT). When the trigger input (TRIG) falls below the trigger level, an internal flip-flop is set and the output (OUT) goes high. If TRIG is above the trigger level and the threshold input (THRES) is above the threshold level, the flip-flop is

reset and the output goes low. The reset input (RESET) can override all the inputs and can be used to initiate a new timing cycle. If the RESET is low, the flip-flop is reset and the output is low. Whenever the output is low, a low impedance path is provided between the discharge terminal (DISCH) and ground. All unused inputs are linked to an appropriate logic level to prevent false triggering.

Figure 2.11 shows the circuit diagram for the alarm unit. In the astable mode, when power is first turned on, the capacitor C3 will be uncharged, therefore, both the trigger and threshold inputs will be near zero, causing the output to switch high. That also causes an open circuit at DISCH terminal and allows the capacitor C3 to begin charging through resistors R1 and R2. As soon as the charge on C3 reaches 2/3 of the supply voltage, the internal flip-flop resets, causing the output to switch low and also establish a low impedance to ground at DISC terminal. The capacitor now begins to discharge through R2. As soon as the voltage across C3 drops to 1/3 of the supply voltage, the control flip-flop again sets and the output goes high. DISCH is again an open circuit and C3 begins to charge. The cycle continues to repeat with C3 alternately charging and discharging. The resulting output is a continuous stream of rectangular pulses. A 0.01mF capacitor is connected between CONT and GND to prevent false triggering.

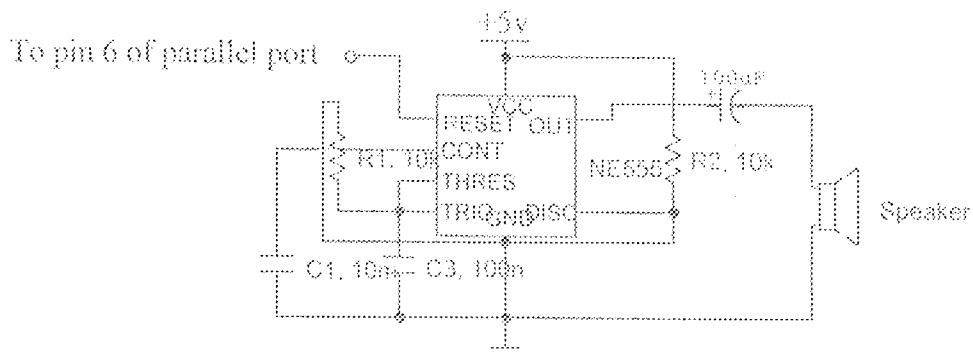


Fig. 2.11 Alarm unit.

The time taken to charge from $1/3V_{cc}$ to $2/3V_{cc}$ is:

$$\begin{aligned}t_{ON} &= 0.693 (R1+R2) C3 \\ &= 0.693 (10k+10k) \times 0.1mF = 1.4ms\end{aligned}$$

The time taken to discharge from $2/3V_{cc}$ to $1/3V_{cc}$ is:

$$\begin{aligned}t_{OFF} &= 0.693 \times R2 \times C3 \\ &= 0.693 \times 10k \times 0.1mF = 0.7ms\end{aligned}$$

Signal period T is:

$$T = t_{ON} + t_{OFF} = 1.4 + 0.7 = 2.1ms$$

Frequency of oscillation f is:

$$f = 1/T = 1 / 2.1ms = 476.2Hz$$

The duty cycle D is:

$$D = (t_{ON} / T) \times 100\% = (1.4 / 2.1) \times 100\% = 66.7\%$$

The $100\mu F$ capacitor between the timer output and speaker filters off dc components of the output signal and the spikes generated at the signal peaks.

2.2 SOFTWARE DESIGN

Software is a program, stored in memory, which directs the computer hardware on what to do at any given point in time. In every real time control, the ingenuity in the software design determines the intelligence of the system.

Computer-aided water pump actuator driving software was written in object oriented programming language, C++. The design can be divided into three sections viz; the graphical user interface (GUI), the input/output (I/O) routines and the mouse/keyboard routines.

2.2.1 THE GRAPHICAL USER INTERFACE (GUI)

The GUI is the graphic display on the monitor to ensure easy and effective communication between the computer (software) and the user. It is through the GUI that the user interacts with the computer to effect efficient operation of the system. The graphic displays are designed using the rich graphic commands in Borland Graphics Interface (BGI) incorporated in the Borland C++ compiler. Some screen captures of these graphics are included in the appendix.

The Welcome Window: - This is the first screen display when the program is started. It contains a welcome message, the title of the software, name of designer and supervisor of the project.

There are three buttons. The "EXIT" to terminate the program, "MENU" to jump straight to the menu screen and "CONTINUE" to move to the next screen, which is the introduction.

The Introduction Screen: - This window contains an introduction to the system design and how it operates. The page also has three buttons. The "EXIT" and "MENU" buttons perform same tasks as described in welcome window above. The "CONTINUE" buttons takes the user to the quick guide.

The Quick Guide: - this screen gives useful hints that will help the user to use the software efficiently as is found in any software application. It contains tips on setting up the hardware, description of button functions, entering commands and figures from the mouse and keyboard, and understanding screen displays. The "EXIT" button on this screen terminates the application. Either "MENU" or "CONTINUE" takes the user to the menu screen.

The Menu Window: - This displays the menu for the user to choose a mode of operation or choose to quit the program. There are two modes of operation of the system – the automatic water-pump control and the automatic device control modes.

The Automatic Water-Pump Control Mode: - This screen displays a tank with water running into it from a pipe to depict the kind of operation going on. However, since the running water does not actually indicate pump operation, there is a pump status display on the screen to show that the pump is ON or OFF. There are also buttons like “ON” to switch on the pump irrespective of the water level in the tank, OFF to force the pump off, “RESET” to clear the over riding effect of the ON and OFF buttons and the return control to automatic operation, the “MENU” button to return to the menu, and exit to quit the program.

Automatic Device Control Mode: - In this mode, the system can be used to switch any electric device ON/OFF based on timing. The screen display provides windows for entering the ON period, OFF period and DELAY time. It also displays the date and time that the control operation is initiated so that the user can recollect and then predict the status of the system at any time. Here again, there are ON/OFF override buttons and the “RESET” button. Clicking the reset button in this mode requires that new periods be entered. Clicking the “MENU” and “EXIT” buttons return the user to the menu and quit the program respectively.

Error Message Boxes: - The program is designed to trap errors and inform the user of such errors. Serious errors like possible overflow from the tank and pump failure are accompanied with an alarm while other less serious errors are simply displayed on the screen with the aid of a dialogue box carrying the message. Each message box has “OK” button, which is used to acknowledge the error. If the problem is not solved before clicking the “OK” button, the message will persist. Errors trapped by the software include mouse error (not detected), no

power supply to the external device, improper cable connection, water-overflow, pump failure and water detection errors.

2.2.2 INPUT/OUTPUT ROUTINES

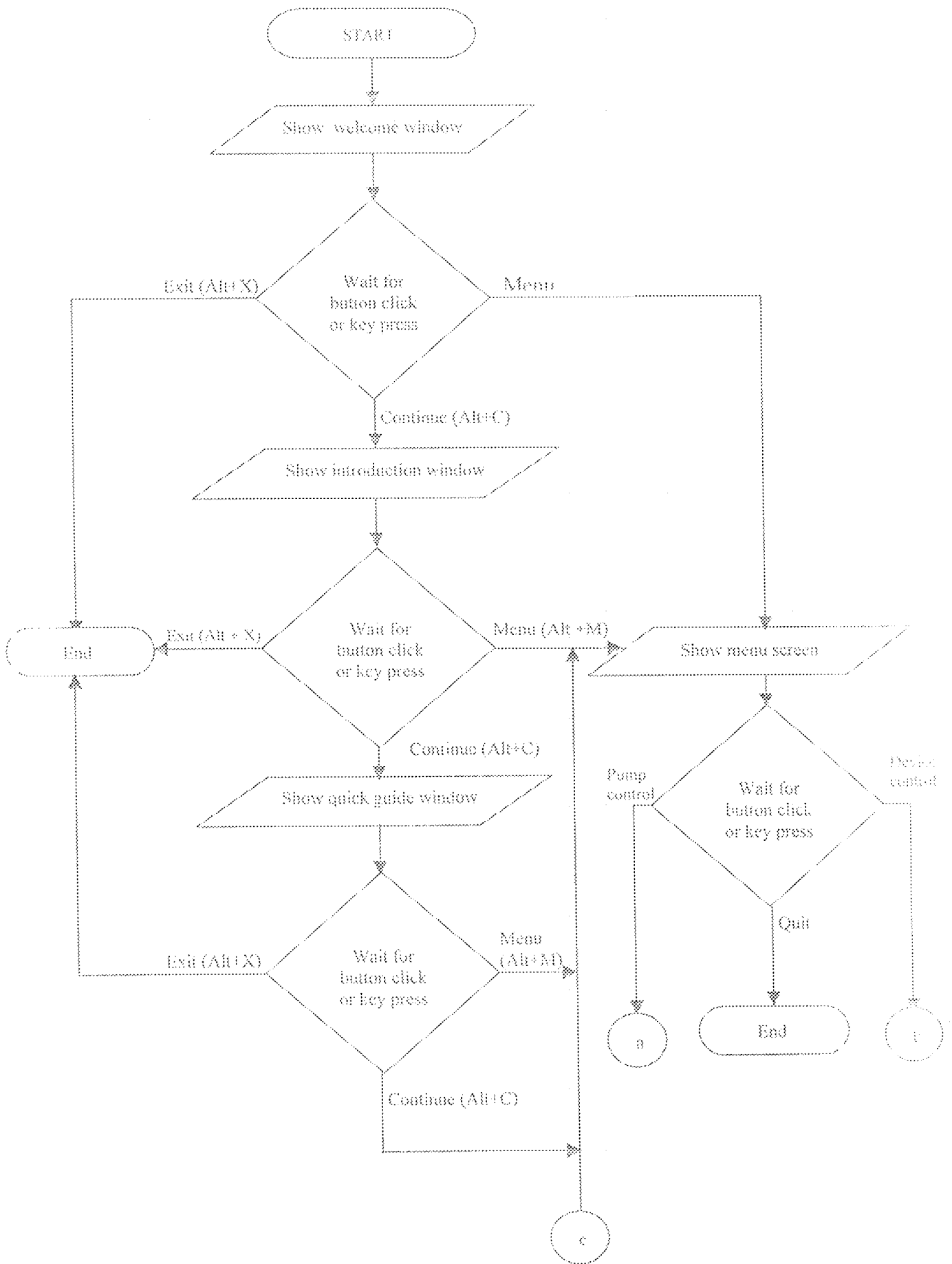
Input and output routines involve the communication between the microprocessor and the external device via the parallel port. It includes the acquisition of probes status from the input buffers through the status port and sending out of control signals via the DATA port to switch ON/OFF the pump/electrical device, alarm and LEDs.

The parallel port is accessed using the *inportb (Address)* and *outportb (Address, value)* C++ commands. Prior to input from the input buffers, its outputs are enabled through the CONTROL port, and disabled immediately after the data has been read.

The low probe is connected to bit 3, HIGH to bit 4 and overflow to bit 5, while the power detection transistor is connected to bit 7, all of the STATUS port. Bit 0 of the DATA port drives the switching unit, bit 1 to bit 3 drive the LEDs while bit 4 drives the alarm unit. The enables of the input buffers are all linked to bit 0 of the CONTROL port.

2.2.3 KEYBOARD AND MOUSE ROUTINES

Mouse and keyboard are very necessary for the user to be able to interact with the software. In order to be able to detect all keyboard presses including the non-ASCII keys like the "Alt" key, the *__bios__ keyboard (cmd)* command is used instead of *getch()*. Besides clicking, buttons can be activated by pressing Alt + underlined letter on the button of interest, as is found in windows programs. This feature allows the user to be able to use the program even when the mouse is not available. Timing period are also entered through the keyboard.



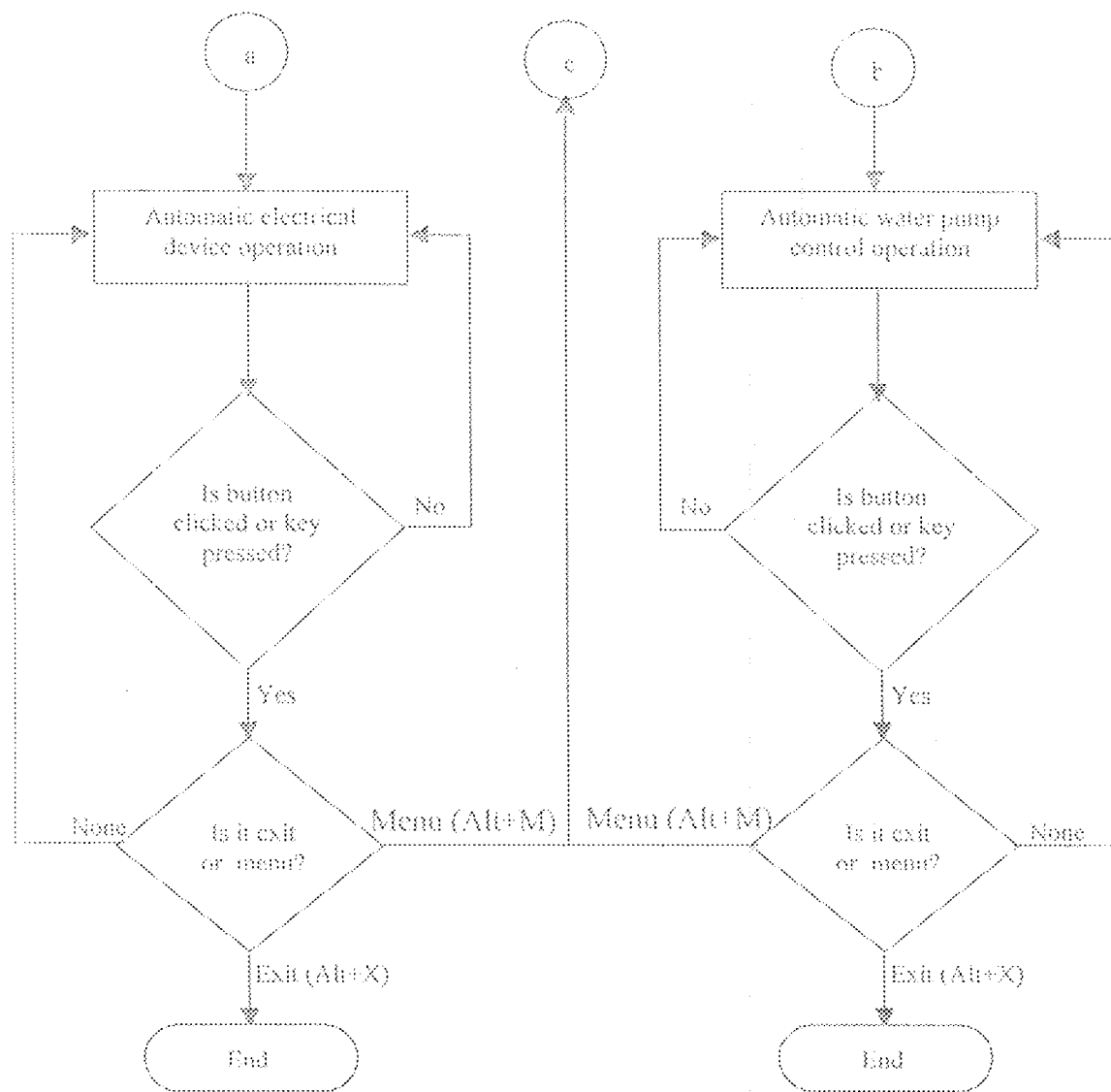


Fig2.12 Program flowchart (simplified)

The mouse routines were incorporated using a mouse library shareware, which is added to the project. Mouse activities like left clicking in a particular area on the screen (button) were then associated with certain program routines to carry out a pre-defined task.

The program was compiled into a stand-alone executable so that the C++ Integrated Development Environment (IDE) will not be required to run the program. This means that the program can run on any computer, with or without C++ compiler installed. Due to lack of space, only a selected section of the program is included in the appendix. The flowchart for the program is shown in figure 2.12.

CHAPTER THREE

CONSTRUCTION AND TESTING

3.1 TOOLS AND MATERIALS

During the construction of Computer-Aided Automatic Water Pump Actuator, some tools and materials were used. Some of them are briefly discussed here.

1. Breadboard - This is a board on which a circuit is set up temporarily to ascertain that it is working according to design before it is transferred to a veroboard.
2. Veroboard - This is a perforated plastic, overlaid with strips of metal conductors. It is used for permanent construction of the project prototype.
3. Lead - This is a metal with low melting point. It is used to hold components and connecting wires in place on the veroboard.
4. Soldering iron - It provides the heat needed to melt the lead onto the veroboard when connected to ac mains.
5. Lead sucker - This is used to suck-up molten lead from the veroboard when de-soldering a connection.
6. Digital multi-meter - This is a multi-function electronic measuring instrument employed in the measurement of voltages, resistances, transistor current gains, and checking for continuity.

Others are pliers, wire cutters, drilling machine, file, chisel, scissors, hammer and bending machine.

3.2 HARDWARE CONSTRUCTION

Careful planning of the circuit layout and simple wiring minimized errors and made troubleshooting easier. All ICs used in the design were aligned in the same direction for a

logical signal flow and this also made it easy to keep track of pin numbers during soldering and troubleshooting.

Each unit is soldered independently and tested to ensure that it is working fine. All the units are then interconnected together as specified in the design. All the connections to the device are made detachable by using connectors at the back panel of the system casing. This made the device to be portable.

All lines to and from the computer are connected to a standard DB-25 male connector. The probes use a 4-pin connector. Both the power input and output use 3-pin sockets, but of different sizes.

3.3 PROJECT CASING

Metal is used to case the project. The dimensions are 134mm x 142mm x 132mm. The front panel is cut at convenient portions for the power switch, the fuse, power and water level indicator LEDs, and perforated around the speaker position to allow passage of sound of alarm. The cover has holes drilled on its sides for ventilation.

The back panel has provisions for the DB-25 male connector, 4-pin connector for the probes, 3-pin power supply socket and 3-pin output socket. The entire casing is sprayed with auto-base paint. Each component on the front and back panel is labeled for easy identification.

3.4 SYSTEM COUPLING

The transformer is mounted, using screws, at the centre of the casing so as to balance its weight. The circuit board is mounted in a vertical position so as to create reasonable space inside the device for interconnecting wires. The sockets, switch, fuse and connectors are

firmly screwed on the casing. The LEDs are put in place using gum. The strong magnet on the speaker made it easy to stick to the metal case firmly.

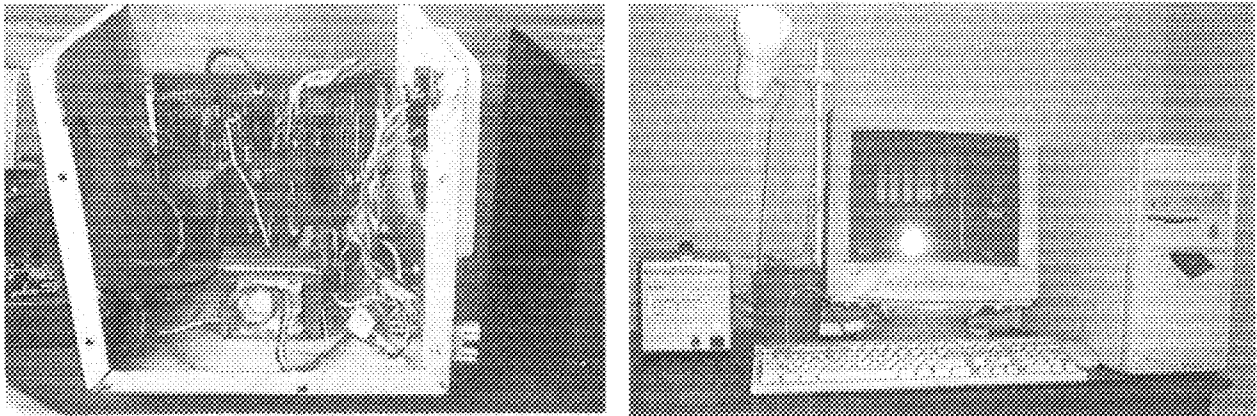


Fig. 3.1 Photograph of the device (interior view and testing setup)

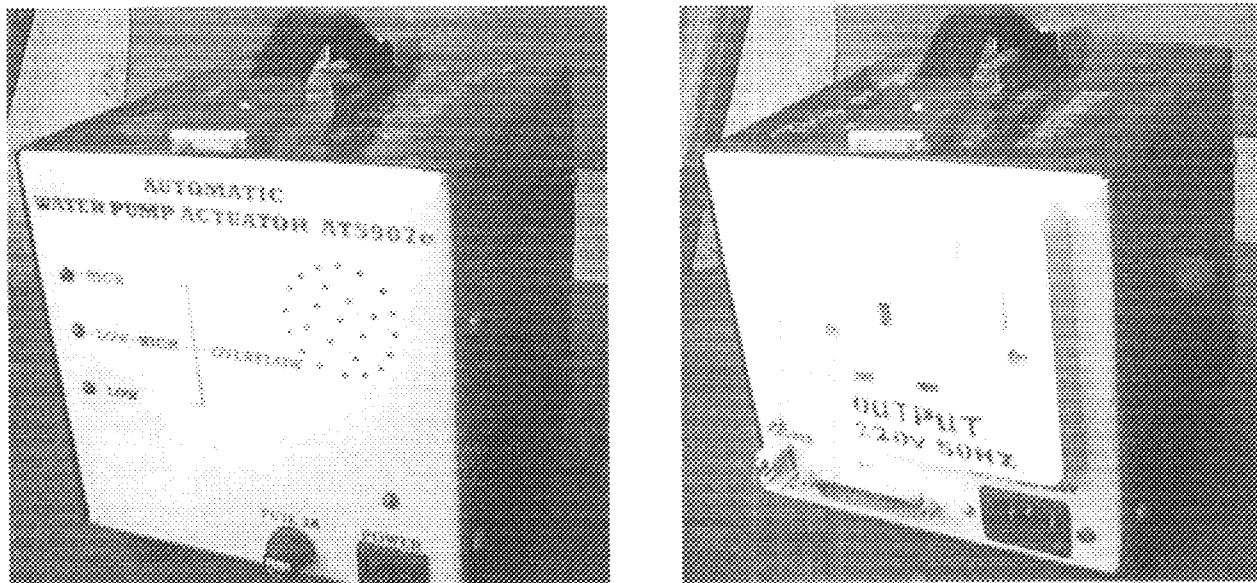


Fig. 3.2 Photograph of the device (front and back panel view)

3.5 CONSTRUCTION PRECAUTIONS

1. All soldered joints were tested for continuity so as to avoid open circuits.
2. All excessive lead were removed to avoid bridges (short circuits) on the board.
3. Polarities of electrolytic capacitors and pin configuration of transistors and ICs were checked properly before soldering.
4. ICs were mounted on IC sockets to avoid over-heating them and for easier troubleshooting.
5. Excessive heating of the components was avoided so that they do not burn out.

3.6 TESTING AND RESULT

The system was setup by connecting the power supply cable, parallel port cable and water sensing probes to their respective connecting points/outlets at the back of the device. A bowl of water was provided too. In the absence of a water pump, a light bulb was used as load.

The device was switched on and the program was launched. In the automatic water pump control mode, when no probe was dipped in water, the light bulb and the orange LED came on. When the LOW probe was dipped in the water, the orange LED went off and the green one switched on. The light bulb remained on. The HIGH probe was then lowered in to the water, the light bulb switched off and the LEDs switched from green to red. In order to test the system response during overflow, the OVERFLOW probe was put into the water, the light bulb remained off, all the LEDs and the alarm came on. Consequently, an error message was displayed on the screen, which disappeared when the OK button was clicked. The probes

were then removed in turn and the processes reversed with the light bulb remaining off until the LOW probe was removed from the water.

The program mode was changed to automatic electric device control. Testing periods of 2, 1 and 2 were entered for the on, off and delay respectively. The light bulb and green LED came on after two minutes, stayed on for two minutes, went off for one minute and came on again for two minutes and the circle continued until it was terminated.

The power supply was intentionally switched off in the external device and the power detection error message was displayed on the screen. It refused to disappear when the OK button was clicked until the power was restored. The cable was disconnected and the cable connection error message showed up. It remained until the cable was reconnected.

3.7 PROBLEMS ENCOUNTERED

During the testing period, it was discovered at some point that the system was no longer responding to changes in the water level. It was checked and discovered that there was no continuity in the common probe. It was fixed and the system functioned normally thereafter.

Many problems were encountered during the programming stage. They were however solved. Most often, no section runs as desired at first attempt. Debugging would eventually make it run as expected.

Drilling and cutting out of the metal parts of the casing to desired shapes and filing the edges wasn't an easy task at all, particularly, in the absence of precision machines.

CHAPTER FOUR

CONCLUSION AND RECOMMENDATION

4.1 CONCLUSION

From the results of the tests carried out, the computer-Aided Automatic Water Pump Actuator performed its expected functions satisfactorily. The components used in constructing the device are inexpensive and commonly available, and with the wide spread use of computers together with an easy-to-use program, the device will be useful in homes, industries, hospitals, schools etc to make water available at reduced cost and minimum energy input.

The practical implementation of the design made the student to become more familiar with electronic components and improve the student's skills and techniques in handling construction tools.

In summary, the aims and objectives of the project have been achieved, though, with little problems that were encountered but resolved.

4.2 RECOMMENDATION

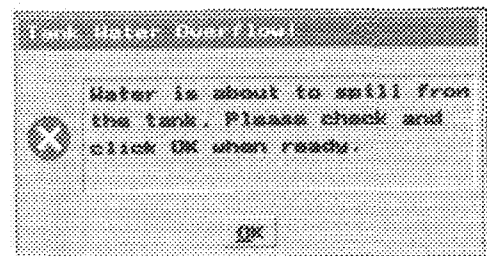
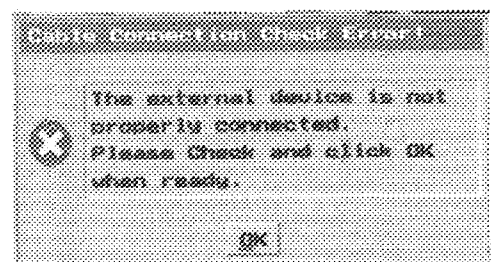
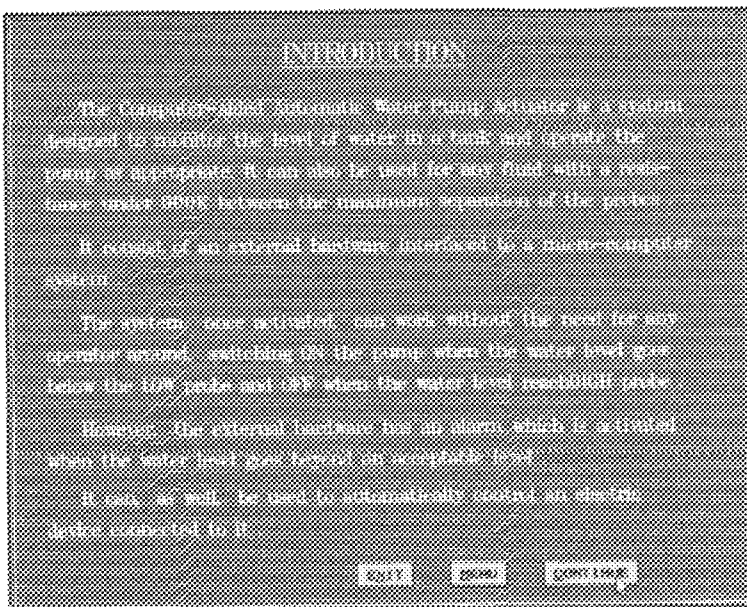
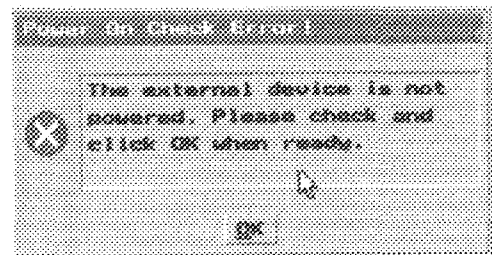
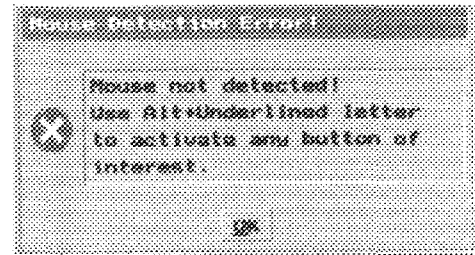
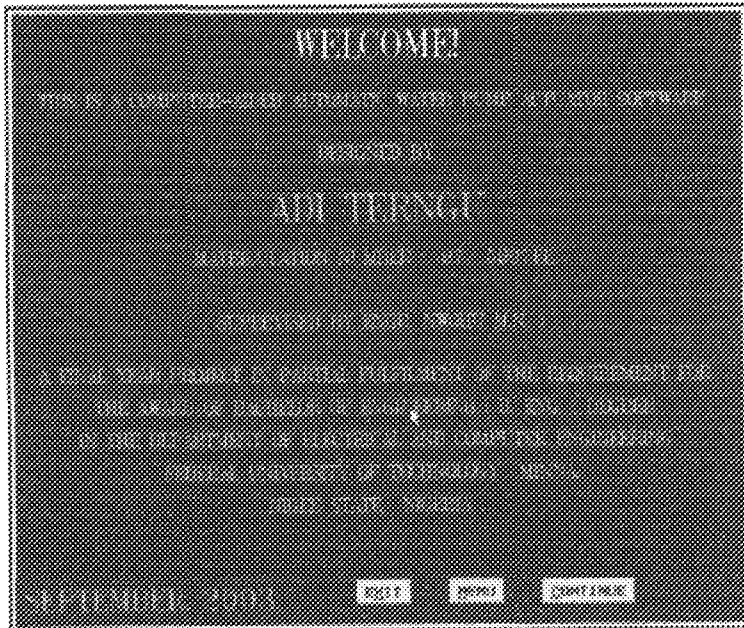
In this design, the water level was measured at discrete points only. This can be improved by using differential approach like varying resistance with liquid level. This will make it possible to know the exact liquid level in the container at any point in time and communicated to the user using appropriate displays on the computer screen.

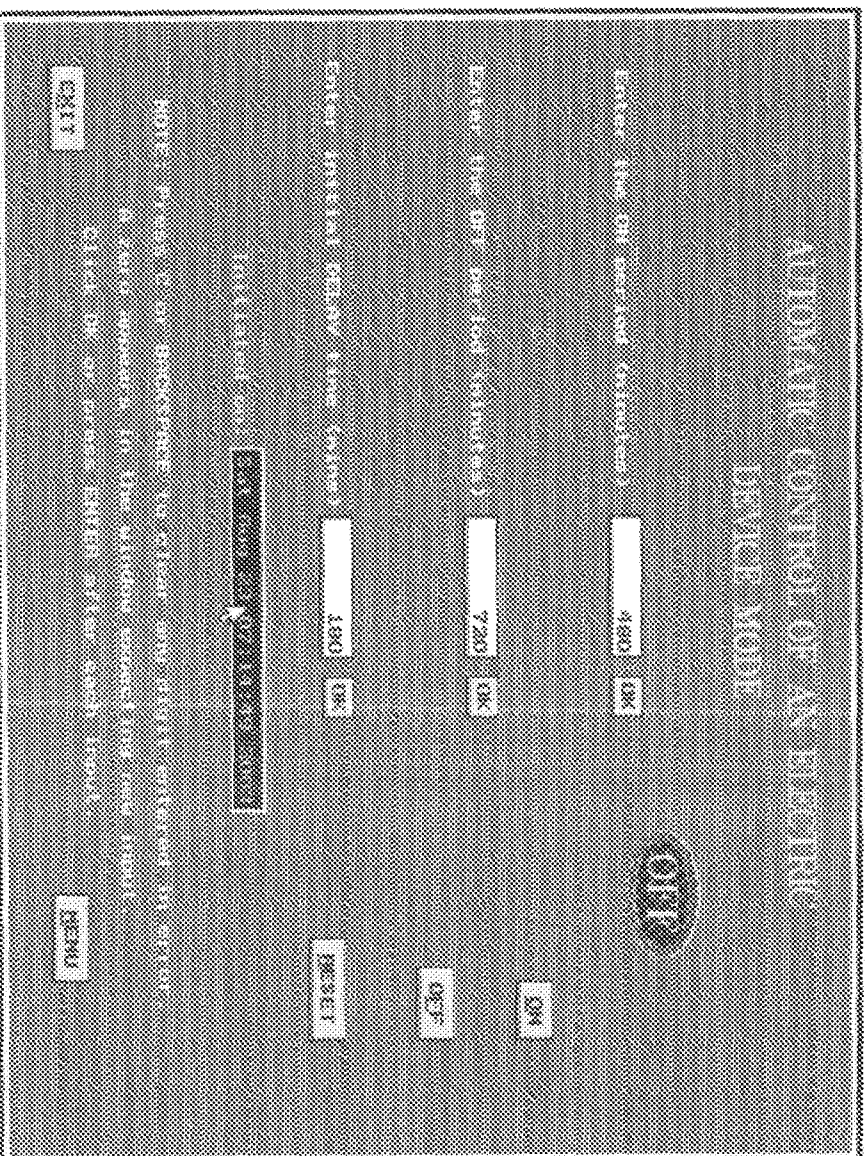
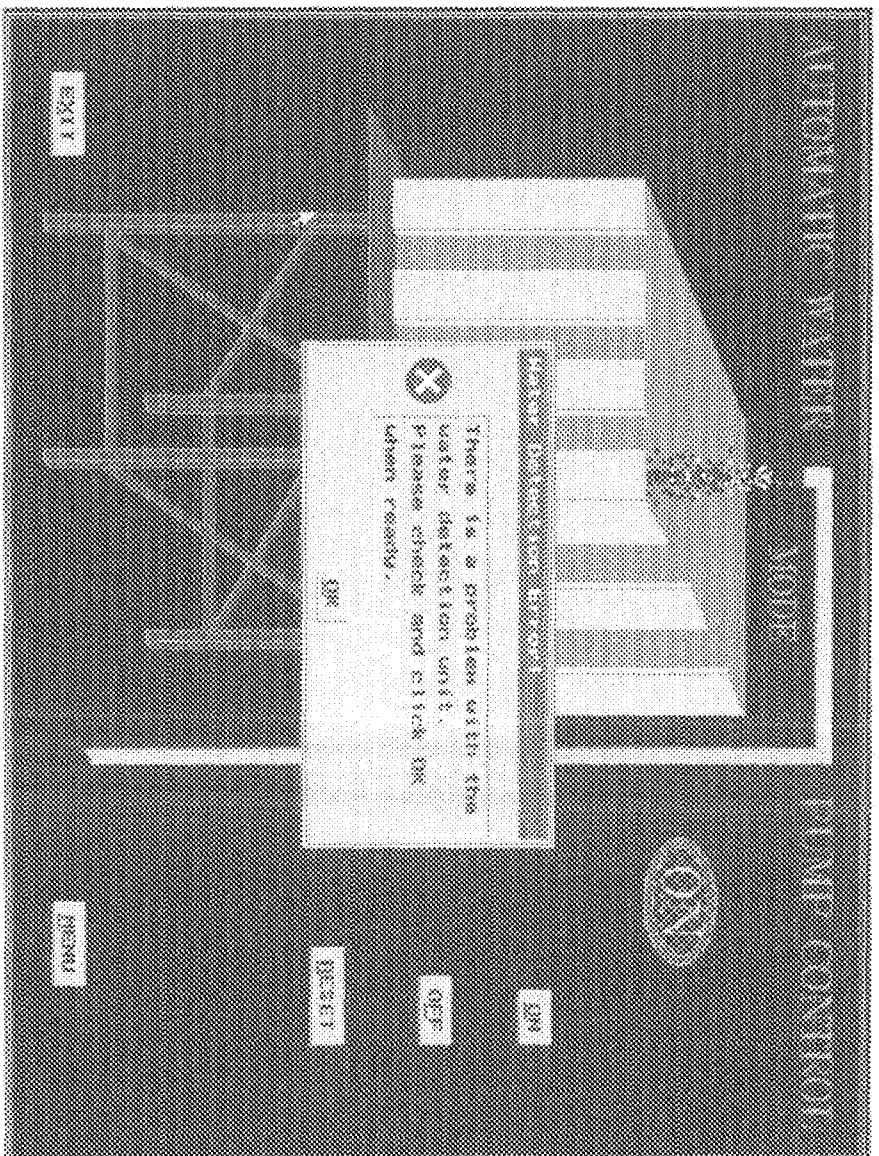
More emphasis was placed on automatic water pump control than the automatic electric device control during the programming due to lack of time. The device control mode of operation can be further worked on to bring out more functionality from the device when operated in this mode.

REFERENCES

1. Malvino P.A (1999), "Electronic Principles", McGraw-Hill companies Inc, New York.
2. Horowitz P., Hill W., (1995), "The Art of Electronics", Cambridge University Press, Cambridge.
3. Rockis G., Mazar G., (1987), "Electrical Motor Controls", American Technical Publishers, New York.
4. Dorf R. C., Bishop R .H., (1998). "Modern Control Systems", Addison Wesley Longman Inc., England.
5. Tocci R. J., Widner N. S., (1998), "Digital Systems, Principles and Applications", Prentice Hall Inc., New Jersey.
6. Theraja B. L., Theraja B.K., (1999), "Electrical Technology", S. Chand & Company Ltd., Ram Nagar, New Delhi.
7. Stroustrup B., (1991), "The C++ Programming Language". Addison Wesley Longman Inc., England.
8. Stevens A., (1993), "Object-Oriented Programming in C++", MIS Press, Portland.
9. Lippman S. B.,(1991), "C++ Primer". Addison Wesley Longman Inc., England.
10. Peacock C., 2002, "Standard Parallel Port (SPP)", (<http://www.beyondlogic.org/spp/parallel.htm>).

APPENDIX A: SOFTWARE GUI SCREEN CAPTURES





APPENDIX B: DRIVER SOFTWARE

```
/* Project.h */  
#include <graphics.h>  
#include <conio.h>  
#include <string.h>  
#include <iostream.h>  
#include <stdlib.h>  
#include <dos.h>  
#include <time.h>  
#include <bios.h>  
  
#define ALT_D    0x2000  
#define ALT_F    0x2100  
#define ALT_M    0x3200  
#define BKSPC    8  
#define RETURN  13  
#define ENTER   0x1C0D  
#define ALT_C    0x2E00  
#define ALT_O    0x1800  
#define ALT_P    0x1900  
#define ALT_Q    0x1000  
#define ALT_R    0x1300  
#define ESC     0x011B  
#define ALT_X    0x2d00
```

```

#define DATA      0x378
#define STATUS     0x379
#define CONTROL    0x37A

enum buttonstatus { unpushed, pushed };
enum boolean { false, true };
enum hdir { hCENTER, hRIGHT, hLEFT};
enum devicestatus {Delay, off, on, OFF, ON};
enum mode {men, pum, dev, cont};

class Page
{
private:
    char line[70]; int xco, yco; int bckgrdcolor;
    int font, direction, size, color, mulx, divx;
    int horizjustify, vertjustify, muly, divy;
public:
    Page(int c=0){bckgrdcolor = c;}
    void drawborder(void); void setttext(void);
    void writetext(int, int, int, int, int, int, int, int, int);
    void writetext2(int, int, int, char**, int, int, int, int);
};

class Window
{
protected:

```

```

    Window* ptowner; char text[30];
    int left, top, right, bot, delta;
    int deltacolor, centercolor;

public:
    Window(char* tx, Window* ptoc, int l, int t, int r,
           int b, int dc, int cc);
    void display(void);
};

```

```

class button : public Window
{
private:
    int ch, outline; buttonstatus bstatus;
public:
    button(char* tz, Window* ptro, int l, int t, int r,
           int b, int c=l, int cc=BLACK, int dc=WHITE) :
        Window(tx, ptro, l, t, r, b, dc, LIGHTGRAY)
    { bstatus = unpushed; ch=c; outline=cc; }
    void click(void); void display(void);
};

```

```

class output : public Window
{
private:

```

```

        int horiz, textcolor;

public:
    output(char* tx, Window* ptrc, int l, int t, int r,
           int b, int dc=WHITE, int cc=BLUE, int hz=0, int d=0,
           int tc = 15) : Window(tx, ptrc, l, t, r, b, dc, cc)
        {horiz = hz; delta = d; textcolor = tc;}
    void show(char*);
};

class dialogobox : public Window
{
private:
    char** ptrmsg; char message[50];

public:
    dialogobox(char* tx, char** msg) : Window(tx, 0,
        180, 170, 460, 310, WHITE, LIGHTGRAY){ptrmsg=msg;}
    void display(void);
};

class waterdroplet
{
private:
    int sx, sy, xco, yco, oldx, oldy, speed, color;
    hdir horz;

public:
    waterdroplet(int x, int s, hdir h){xco=x,

```



```

        ycc=3, sx=z, sy=3, speed=s, horz=h, color=WHITE;}

        void erase(); void calculate();

        void drawhose(); void draw();

    };

void slidel(void); void slide2(void); void slide3(void);

void pauseslide(void); void menus(void); void pausomenu(void);

void status(devicestatus*, int=0); void Device(void);

void async(devicestatus*, int=0); void frame(int);

unsigned long GetTime(output*, button*, int); void pausedev();

void support(int, int, int=0); void drawtank(void);

void pausepump(void);

/* Project_app.cpp */

#include "project.h"

Window* const SCREEN = (Window*)0; const int  LINES = 26;

int maxx, maxy, TEXTWIDTH, TEXTHEIGHT, n=0;

unsigned long ONTIME, OFFTIME, STARTTIME;

char saveon[9], saveoff[9], savestart[9];

mode call;

button* ptrExit;  button* ptrMenu;  button* ptrExit;

button* ptrMenu;  button* ptrCont;  button* ptrOkay;

button* ptrok1;  button* ptrok2;  button* ptrok3;

button* ptron;  button* ptroff;  button* ptrreset;

```

```

button* ptrpump;  button* ptrquit;  button* ptrdevice;
output* ptrontime;  output* ptrstarttime;
output* ptrstarted;  output* ptrofftime;
char* TEXT[LINES]; char* GUIDE[20]; char* error[4];
char* detect[4]; char* power[4]; char* pump[3];
char* menu[4]; char* cable[4]; char* PUMPFailure[4];
char* DEVICE[9]; char* ALARM[2];
MouseEventHandler myHandler(ME_RELEASED, MouseHandler);

void main(void)
{
    registerbgidriver(EGAVGA_driver);
    registerbgifont(triplex_font);

    int driver=DETECT, mode;
    initgraph(&driver, &mode, "");

    TEXTWIDTH = textwidth("O"); TEXTHEIGHT = textheight("O");
    maxx=getmaxx();
    maxy=getmaxy();

    output_ontime("", SCREEN, 280,120, 360,140, BLACK, WHITE,
        RIGHT_TEXT, 2, BLACK),
        offtime("", &ontime, 0,80, 80,100, BLACK, WHITE,
        RIGHT_TEXT, 2, BLACK),

```

```

starttime("", &offtime, 0,80, 80,100, BLACK, WHITE,
          RIGHT__TEXT, 2,BLACK),
started("", &starttime, -33,50, 164,70, WHITE,
          BLACK, CENTER__TEXT, 2,LIGHTGREEN);
button ok1("OK", &ontime, 90,1, 88+3*TEXTWIDTH,18, 0),
        ok2("OK", &offtime, 90,1, 88+3*TEXTWIDTH,18, 0),
        ok3("OK", &starttime, 90,1, 88+3*TEXTWIDTH, 18, 0),
on("ON", &ok1, 168,50, 168+4*TEXTWIDTH,70, 1),
off("OFF", &ok2, 160,25, 160+5*TEXTWIDTH,45, 2),
reset("RESET", &ok2,144,65, 144+7*TEXTWIDTH,105, 1),
Exit("EXIT", SCREEN, 30,maxy-50,
      30+6*TEXTWIDTH,maxy-30, 2),
Menu("MENU", SCREEN, maxx-150,maxy-50, maxx-
      150+6*TEXTWIDTH,maxy-30);
button pump("", SCREEN, 140,150, 155,165, 0),
        device("", &pump, 0,55, 15,70, 0),
        quit("", &device, 0,65, 15,80, 0);
ptrontime = &ontime; ptrofftime = &offtime;
ptrstarttime = &starttime; ptrok1 = &ok1; ptrok2 = &ok2;
ptrok3 = &ok3; ptron = &on; ptroff = &off; ptrreset =
&reset; ptrExit =&Exit; ptrMenu =&Menu; ptrpump = &pump;
ptrdevice = &device; ptrquit = &quit;
ptrstarted = &started;

```

```

button exit("EXIT", SCREEN, 300, 440, 300+6*TEXTWIDTH, 460, 2),
      menu("MENU", &exit, 80, 0, 80+6*TEXTWIDTH, 20),
      Continue("CONTINUE", &menu, 80, 0,
              80+10*TEXTWIDTH, 20),
      okay("OK", 0, 304, 284, 336, 303, 1, LIGHTGRAY);
ptrexit = &exit; ptrmenu = & menu; ptrcont = &Continue;
ptrokay = &okay; boolean done = false;
while(1)
{
    if(mouse.Exists()) // check for mouse
    {
        mouse.InstallHandler(0xFF); // install event handler
        mouse.xyLimit(6, 634, 6, 474); mouse.Enable();
        slidel(); pauseslide(); break;
    }
    else
    {
        slidel();
        dialogbox mousererror("Mouse Detection
                               Error!", error);
        mousererror.display(); ptrokay->display();
        while(1)
        {
            int ch;

```

```

        if(_bios_keybrd(_KEYBRD_READY))
        ch = _bios_keybrd(_KEYBRD_READ);
        mouse.GetEvent();
        if((mouse.Released(LEFTBUTTON) &&
mouse.InBox(304,286,336,305)) || ch==ENTER
|| ch==AltO || ch==ESC)
        {
            ptrolay->click(); done = true; ch = 0;
            slidel(); pauseslide(); break;
        }
    }
}
if(done) break;
}
for(;;)
{
    if(call == men { menus();pausemenu(); }
    if(call == pum){ drawtank();pausepump();continue; }
    if(call == dev){ Device();pausedev();continue; }
    if(call == cont){ slide2();pauseslide();
        if(call==men) continue; slide3();pauseslide();
        call = men; }
}
}

```

