

DESIGN AND CONTRUCTION
OF
A MICROCOMPUTER BASED MICROCONTROLLER
PROGRAMMER

BY

ADEYEMI AKINSANYA KAYODE

99/8047EE

DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING
SCHOOL OF ENGINEERING AND ENGINEERING TECHNOLOGY
FEDERAL UNIVERSITY OF TECNOLOGY
MINNA, NIGER STATE.

A PROJECT SUBMITTED IN PARTIAL FULFILMENT
OF THE REQUIREMENT FOR AWARD OF BACHELOR
OF ENGINEERING (B.Eng) DEGREE, IN THE DEPARTMENT
OF ELECTRICAL AND COMPUTER ENGINEERING
FEDERAL UNIVERSITY OF TECHOLOGY, MINNA.

NOVEMBER 2005.

Declaration

I do hereby declare that this project was wholly presented by me, ADEYEMI AKINSANYA of matriculation number 99/8047EE, under the supervision of Engr. Emmanuel Eronu. This project was presented to the department of Electrical and Computer Engineering, during the 2004/2005 academic session.

ADEYEMI AKINSANYA KAYGDE (99/8047EE)

Adeyemi Ak.
Signature

Dec 2, 2005.
Date

Certification

This is to certify that this project work was carried out by Mr. Adeyemi Akinsanya Kayode of matriculation number 99/8047EE, in the department of Electrical and Computer Engineering, Federal University of Technology Minna under the supervision of Engr. Emmanuel Eronu. The project has been prepared in accordance with the specification governing the presentation of a B.Eng Degree in Electrical and Computer Engineering, Federal University of Technology, Minna.

ENGR. E. ERONU



Supervisor's Signature

5-12-2025
Date

ENGR. M.D. ABDULLAHI



H.O.D's Signature

27/07/25
Date

External Examiner's Signature

Date

Dedication

This project is dedicated to the following people who had made positive impact in my life but they had passed to the great beyond:

My father, in person of Mr. Adeyemi Theophilus Adegboye, Daddy, your death has created a vacuum, which nobody could ever fill rather to take solace in the exemplary life you lived. Daddy, every member of the family is still missing your company and putting our hope and trust in our heavenly father, who dies not and hoping that things shall be well.

My beloved lecturers who had left this sinful world to join their creator, Dr. Udokwe(Mechanical department) and Engr. Azula (Electrical department). Sir, though you are gone but your effort in building courageous and competent Engineers for this great country of ours, Nigeria, bear you the witness that you had not only come to this world but you made a remarkable contribution to the development of this country.

Finally, my fellow colleagues who had slept in the Lord: Achi Mugu, Funmilayo Fashona and Felicia Tsado. Although, these wonderful and gallant soldiers were lost in the battle but they put up a good fight that made them incommensurable with their mates and remembered by friends. Adieu

Acknowledgement

My first appreciation goes to my creator, the omnipotent, omniscience, the king of kings, the almighty God, which His name is endless and His reign is forever. I'm grateful God for seeing me through my programme, He made me to sail through despite the ups and downs that I faced and put a new song into my mouth.

As it is being said that behind every successful man there is a woman, also behind every graduand there is a family. Mummy, Brothers and Sisters, I appreciate the supports you have given me for making my dream to come to reality. I want you to know that I love you and care for you and I will always have you at the back of my heart.

Also, I will like to acknowledge the supports these following people have given me: Mr. & Mrs. Oyelami, Mr. & Mrs. Adebiji, Mr. & Mrs. Hemen, Isaac Hemen, Joseph Okeh, Christopher Aweto, Christopher Adeboye, Mustapha Umar, Mr. & Mrs. Oluwole, Mr. & Mrs. Adejumo and others. Please those whose names are not mentioned, your incomparable supports are immensely appreciated, it not done deliberately but out of oversight.

My final appreciation goes to the staff of Electrical & Computer Engineering department of the university, especially my supervisor who have granted me his listening ear and encouraged me toward successful completion of this project, thanks Sir for your unquantifiable supports, really you are a good mentor. I thank every member of this department for your contribution in training another competent Engineer for the world.

TABLE OF CONTENTS

| Contents | Page |
|--|------------|
| Title page | I |
| Declaration | II |
| Certification | III |
| Dedication | IV |
| Acknowledgement | V |
| Abstract | VI |
| Table of Contents | VII - VIII |
| Chapter One: | |
| 1.0 Introduction | 1 |
| 1.1 Project Aim/Objective | 2 |
| 1.2 Project Layout | 2 - 3 |
| Chapter Two: Literature Review | |
| 2.0 Microcomputer Review | 4 - 5 |
| 2.1 Interfacing Review | 5 - 6 |
| 2.2 Microcontroller | 7 |
| Chapter Three: Project Design and Analysis | |
| 3.0 Introduction | 8 |
| 3.1 Power Unit | 9 |
| 3.2 Transformer | 9 - 10 |
| 3.3 Rectification | 10 - 11 |
| 3.4 Filtering\Smoothing | 11 - 14 |
| 3.5 Voltage Regulator | 14 |
| 3.5.1 Linear Voltage Regulator | 14 - 15 |

| | |
|---|---------|
| 3.5.2 Switching Voltage Regulator | 15 - 18 |
| 3.6 Surge Protection | 18 - 19 |
| 3.7 Fusing | 19 - 20 |
| 3.8 Microcontroller Chips | 20 - 27 |
| 3.9 Microcomputer\Interface..... | 27 - 30 |
| Chapter Four: Construction, Testing and Result | |
| 4.1 Construction | 31 |
| 4.1.1 Hardware Module | 31 - 33 |
| 4.1.2 Software Module | 33 |
| 4.1.3 Components Layout | 33 - 34 |
| 4.1.4 Construction Tools | 34 - 35 |
| 4.1.5 Construction Diagram | 35 |
| 4.2 Project Testing | 35 |
| 4.3 Result\Discussion of Result | 36 |
| Chapter Five: Conclusion and Recommendation | |
| 5.1 Conclusion | 37 |
| 5.2 Recommendation | 37 |
| Reference | 38 |
| Appendices | |
| Appendix A Circuit Diagrams | 39 - 40 |
| Appendix B Myfirmware Program | 41-42 |
| Appendix C Mycode Program | 43 - 44 |

Abstract

The aim of this project is to design and construct a microcomputer based microcontroller programmer for 8051 family.

This is being achieved with the aid of firmware on the programmer and a microcomputer based software. The software was developed using Visual Basic(VB), VB was chosen because it has graphics user interface(GUI) facilities, which made the project interactive with the User. On the other hand, the firmware on the programmer was developed using C-language. C-language was taking because it is structural based than BASIC language and assembling language.

With this microcomputer based microcontroller programmer, the users can program their microcontroller chips with their codes and this will not only encourage Nigerians in designing a more effective and reliable automated system that can improve the economy but also brings out the ingenuity of Nigerians in digital circuit design. This microcontroller programmer can only be used any microcontroller that has the same pins and programming configurations with the family, so avail yourself the opportunity this project has brought.

CHAPTER ONE

1.0 Introduction

Over a decade now, microcomputer have been within the reach of many people and its importance that we can all testify to cannot be over-emphasized. Is its importance in data processing at offices and homes or its communication importance, as it's being used in both internet and intranet data transfer is to mention, that is known to a number of people or its industrial usefulness is to agree? Despite its numerous usefulness, few people bother to know what makes up a microcomputer and how they can use it to control external devices with the aid of software, running on the system.

Microcomputer is made-up of hardware and software. The hardware are Microprocessor, Memory unit and Input/Output device unit [1],[2]. The microprocessor is the heart of microcomputer; it functions like the central processing unit of any computer [1]. The memory unit handles the storing of program and data by the microprocessor while the input/output device unit aids the communication of microcomputer with external devices [2].

Knowing the microcomputer constituents, this project would use microcomputer in-conjunction with microcontroller programmer to program a microcontroller chip. The programmer is connected to the microcomputer through one of its serial ports of input/output device unit and communicates with an aid of a microcomputer-based software developed using Visual Basic language. Whenever the microcomputer is to be used to program a chip, the software is invoked. The mode of data transmission is serial, via D9 RS232 cable.

A programmer is a device used in programming memory chip that is to be used for a special purpose. The programmed chip may have to control other electronics components or devices with little or no human interference. When a chip is designed for this purposed, it is called controller which is the main ingredient of automation and the program running on the chip is called Firmware while the programmer is called controller programmer.

The programmer of this project is microcontroller programmer for **8051 family** that have 40-pin configuration but it can also program any other chip that has the pins configuration as the 8051 family.

The microcontroller programmer has two main modules; these are Power module and Chip module. The power module generated the power supply for the programmer while the chip module handled the programming of the microcontroller chip to be programmed.

1.1 Aim and Objectives

The aim of this project is to design and construct a microcomputer based microcontroller programmer for 8051 family and its derivatives.

The objectives are as follow:

- i. To stimulate the industrial production of microcontroller programmer which will not only improve the economy of this country but also encourage the use of microcontroller.
- ii. To stimulate the use of microcontroller in automation, especially using 8051 family.
- iii. To produce a low cost programmer for the use of Nigerians, in order to stimulate their interest in using microcontroller.
- iv. Challenging Nigeria undergraduate to design a universal microcontroller programmer.

1.2 Project Layout

This project write-up contained five chapters; the chapter one comprised of the main introduction of the project, aim and objectives and project layout. Chapter two is literature review of microcomputer, microcontroller and interfacing. Chapter three is the project analysis, this review the theories employed in the course of executing and reason behind the choice of the materials used. Chapter four handled the construction, testing and

result/discussion of result while the chapter five which is the last chapter that cared for the conclusion, recommendation and references.

CHAPTER TWO

Literature Review

2.0 Microcomputer Review

The advent of large-scale integrated circuit (LSI) ushered in the development of microprocessor and microprocessor based system, which in return have brought advance improvement to the digital circuit design [1]. Microprocessor is the Central Processing Unit (CPU) of a microcomputer or it is the central processing unit on-chip [1],[2].

Microcomputer consists of hardware and software, which are also subdivided into subgroups, for instance, the main hardware are microprocessor unit (MPU), memory unit and input/output device unit (I/O) [1]. A typical block diagram of the connection of a microcomputer hardware is shown below:

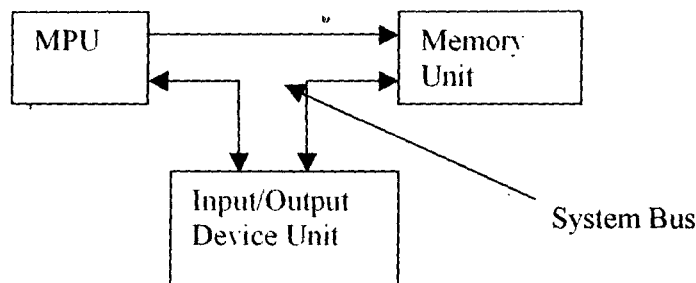


Figure2. 1: Typical Diagram of A Microcomputer Hardware Connection

The microprocessor unit performs both the arithmetic and logical operation and controls other devices because it contains both the Arithmetic\Logical Unit (ALU) and Control Unit like a CPU of any computer. The data and microprocessor's instructions reside in memory until required by the processor [1]. Most Input\Output devices look like memories to microprocessor because it can store to output device and read data from input devices, but the main difference between memory unit and input\output device is that

input/output device unit is associated with external device in the outside world of the microcomputer.

The System Bus connects the various components of a microcomputer, as shown in figure 2.1. A bus is a collection of wires on which electrical signals pass between components in the system. There are three major types of busses used by microcomputer: these are Address bus, Data bus and Control bus [1],[2],[6]. Data bus is used to shuffle data between the various components of a microcomputer. Address bus is used to locate the memory location or input/output device that the microprocessor is communicating with. The Control bus is an eclectic collection of signals that control how the processor communicates with the rest of the system. The control bus handles the system clocking of the microcomputer system. The system clock is an electrical signal on the control bus that alternates between '0' and '1' at a specific periodic rate (see figure 2.2).

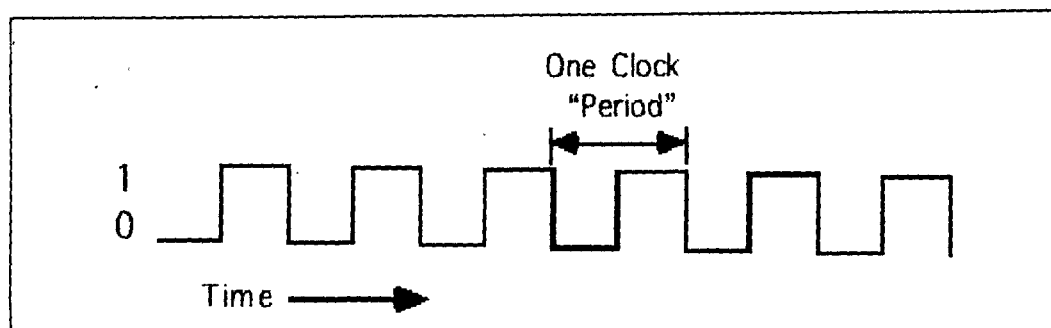


Figure 2.2 A system clock waveform

2.1 Interfacing

Communication between microprocessor and other hardware takes place through one or more Interface. An interface converts data from a form acceptable to the sending device to a form that is also acceptable to the receiving device and adjusts for any speed difference of these devices [2],[4]. The device from which the data is taken from is called Data Terminal Equipment (DTE) e.g. IBM PC, while where the data is to be used is called Data

Communication Equipment (DCE), this is usually a modem. The figure 3 shows the block diagram of the connection of the DTE to DCE.

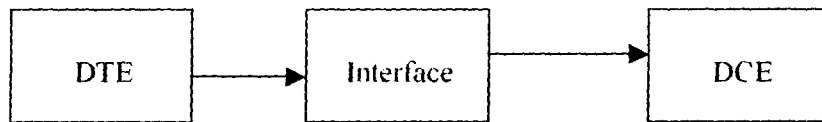


Figure 3: Direction of signal flow

The device that have the ability to transfer and receive data can either be used as DTE or DCE, this includes microcomputers, programmers etc.

Interfacing is beyond direct connection of the DTE and DCE; it also involves using communication protocol and transmission mode. Communication protocol is a set of rules or conventions agreed by the devices involving in the exchanging of data, to ensure that data being exchanged are interpreted correctly [4]. Software is often used to achieve this. Software are series of instructions (program) given to microprocessor for a specific purpose.

The mode of transmission of data is the form by which data are transmitted. The mode could either be parallel or serial transmission mode. The parallel mode involves transmitting all bits in a word at the same time while serial transmission involves transmitting the bits of a word one at a time [2],[4]. Although, parallel transmission mode is the fastest but most costly mode to achieved.

In microcomputer, parallel port uses parallel transmission mode while the serial ports and USB ports use the serial transmission mode. These modes employ three basic Input\Output methods by which data are transmitted between the peripheral devices and memory location. These methods are Programmable input\output, Interrupt input\output and Direct Memory Access (DMA) methods [2].

The programmable input\output method uses microprocessor directly to control data transfer and input\output operation. It uses command like PRINT, LPRINT, INPUT, OUT etc as in BASIC language and GET, PUT as in x86 assembly language. The interrupt

input/output method allows microprocessor to carry on other activities while an input/output operation is taking place.

2.2 Microcontroller Review

The using of a single chip in circuitry which simplified circuit design while adding flexibility and additional functionality have been the driven force behind the development of microcontroller in the mid-1970s [5],[10]. The chip was designed basically as calculator based processor with small Random Access Memory (RAM), data memories and a handful of input/output ports like computers.

The advancement of microelectronics in silicon technology development have made possible the development of more powerful microcontroller of 8-bit, 16-bit etc. In addition to their improved instruction sets, microcontrollers have on-chip counter/timers interrupt facilities and input/output handlings. Despite the improvement, the on-chip memory capacity was still small and not adequate for many applications. Although, there was on-chip Ultraviolet Erasable Programmable Read Only Memory (UVEPROM), which simplified the product development time considerably and allow their use in low-volume application but there was still room for improvement.

In view of the early produced microcontrollers' shortcoming, Intel introduced in the early 1980s a family of microcontrollers called 8051 family from the MCS48 family [10]. The 8051 family has all other properties of the early microcontroller and on-chip Electrical Erasable Programmable Read Only Memory (EEPROM) instead of the UVEPROM and more RAM [5], [10]. Since its introduction, it has been one of the most popularly used microcontroller and has been secondly sourced by many manufacturers, such as Atmel etc. the 8051 family, currently have different version and greatly improved capabilities like on-chip analogue to digital converters (ADC), considerably large size of ROM and RAM, pulse width modulation on outputs and flash memories that can be erased and reprogrammed electrically.

CHAPTER THREE

Project Design and Analysis

3.0 Introduction

The special memory devices used in automation are programmed using special devices called Programmers. But, as the memory devices have different configuration and programming requirement so also, the programmers are not of the same configuration. Programmer is named after the device it used in programming e.g. EPROM programmer, microcontroller programmer etc.

This chapter shall give exclusive description of the various components and modules used in the design of a microcomputer based 8051 family microcontroller programmer.

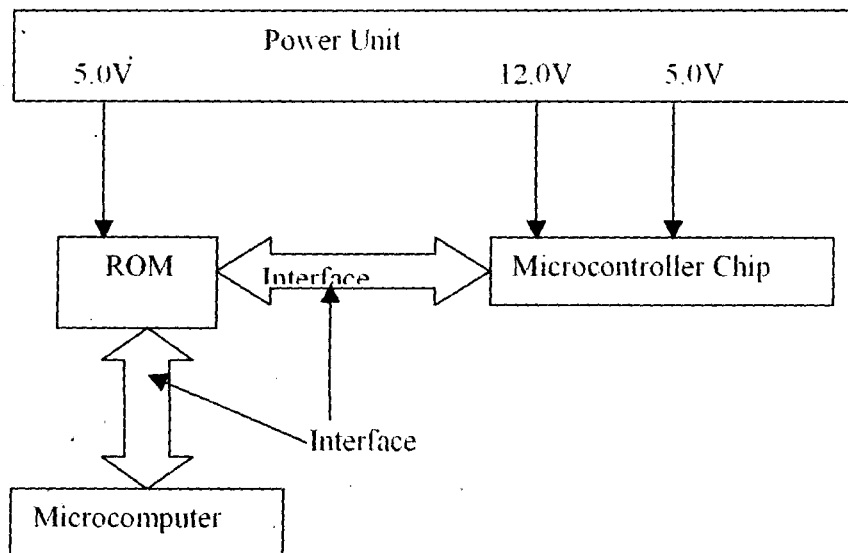


Figure 3.1 Block diagram of a microcomputer based microcontroller Programmer

3.1 Power Unit

Most electronics equipment and circuit need small amount of AC supply that is being transmitted from the power station. As the result, there is need for a device that can bring down, i.e. step-down, the voltage supplied by the power station to an appreciable value that can be rectified by electronics components such as diodes, thyristor for electronics use. The device for this purpose is called Step-down Transformer.

3.2 Step-down transformer

A transformer has two terminals named primary and secondary terminals. The primary terminal serves as the input while its secondary is the output terminal, as shown in figure 3.2

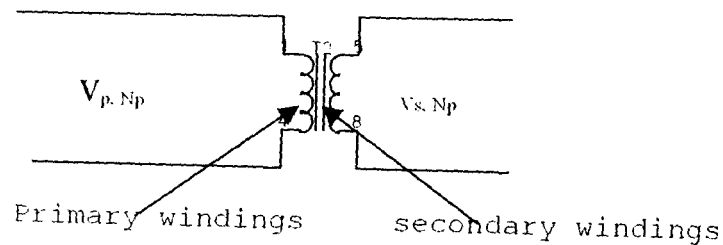


Figure 3.2 A Transformer Schematic Diagram

V_p and N_p are the primary voltage and number of turns at the primary windings, respectively.

V_s and N_s are the secondary voltage and number of turns at the secondary windings, respectively.

The voltage at each terminal is proportional to the number of turns at terminal.

$$\text{i.e } V_p \propto N_p \Rightarrow V_p = K_p N_p \dots \dots \dots (3.1)$$

$$\text{Also } V_s \propto N_s \Rightarrow V_s = K_s N_s \dots \dots \dots (3.2)$$

Assuming a lossless transformer

$$\Rightarrow K_s = K_p ; \text{ dividing eqn2 by eqn1}$$

$$\therefore \frac{N_s}{N_p} = \frac{V_s}{V_p} \dots \dots \dots (3.3)$$

$\frac{N_s}{N_p}$ is called Turns ratio and it can also be called Transfer function of a lossless

transformer $\frac{V_s}{V_p}$

when $N_s < N_p$ the transformer is called Step-down transformer

when $N_s > N_p$ the transformer is called Step-up transformer

this project used 220\18V step-down transformer

$$\Rightarrow \frac{N_s}{N_p} = \frac{V_s}{V_p} = (18 \div 220) \text{ volt/turn}$$

3.3 Rectification

The conversion of an ac voltage into a dc voltage by eliminating the negative half cycle of the AC voltage is known as Rectification [7] and the circuit used in achieving this is called Rectifier circuit. Semiconductor devices such as diodes are the main circuit components of a rectifier circuit.

Although there are numerous rectifier circuits available but a Full wave bridge rectifier was used in this project, for its ability to produce approximate varying and reference voltage. The rectifier circuit and its waveforms are shown in figure 3.3.1 and 3.3.2

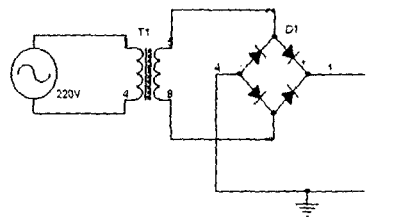


Figure 3.3.1 A Rectifier Circuit Diagram

The output secondary voltage was an ac voltage V_s with its waveform shown in figure 3.3.1 b

$$V_s = V_{\max} \sin \omega t \dots \dots \dots (3.4)$$

where

V_s ——— Average secondary voltage

V_{max} _____ Maximum secondary voltage

From figure 3.3.1a, the diode D1 and D3 conduct during the positive half cycle of the ac while the D2 and D4 are OFF. During the conduction of diodes D1 and D3, they act like a closed switch while the D2 and D4 act like an opened switch. In the negative half cycle, the diodes D2 and D4 conduct while the D1 and D3 are OFF.

The average output voltage of the rectifier circuit when D1 and D3 or D2 and D4 conduct is given by:

$$V_{dc} = 0.636(V_{max} - V_d)$$

$$\text{Where } V_{max} = \sqrt{2} \times V_s \dots\dots\dots (3.5)$$

$$= \sqrt{2} \times 17.5 = 24.75V$$

V_d _____ voltage dropped across the two diodes conducting during the half cycle

$$V_d = 0.6V$$

$$V_{dc} = 0.636(24.75 - 0.6) = 0.636 \times 24.15$$
$$= 15.37V$$

A bridge rectifier (3N532) of 50V breakdown voltage was used in this project, in order to reduce the circuitry and improve the stability and reliability of the rectifier of this project. Also, its breakdown voltage is greater than the peak inverse peak (PIV), which has to be twice the V_{max}

$$\text{i.e. PIV} = 2 \times V_{max} \dots\dots\dots (3.6)$$

$$= 2 \times 24.75$$

$$= (49.75 < 50)V$$

3.4 Filtering or Smoothing

There is need to improve the dc voltage of the rectifier circuit output, the circuit used in achieving this is called Filter circuit. The main function of a filter circuit is to minimize the fluctuating content in the rectifier output called Ripple [7]. Ripple is the ac component

that is contained in the rectifier circuit output. This type of output is not useful for driving sophisticated electronic circuits or equipment like programmer because they need a very steady dc power supply that approaches the smoothness of a battery's output [7]. The block diagram of achieving a filtered dc power supply is shown in figure3.4

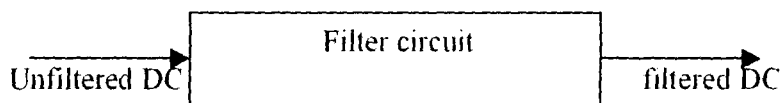
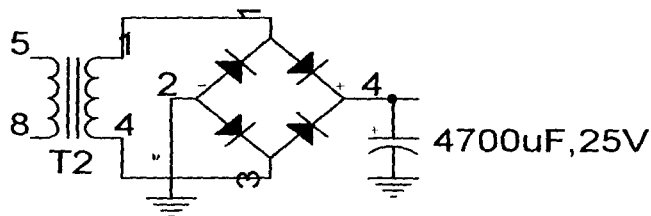


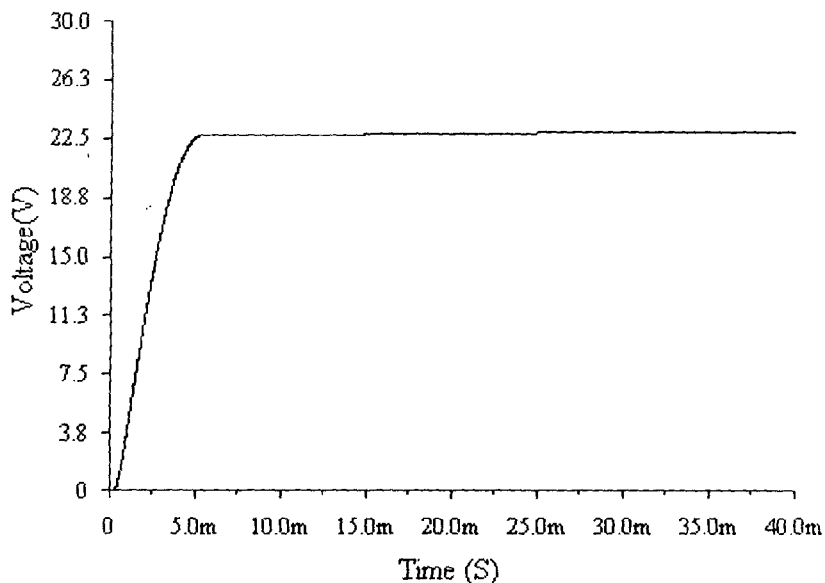
Figure3.4 A Filter block diagram

The output of the rectifier is passed through a filter circuit, which filters the ripple in the output to a very appreciable steady dc supply

There are number of filter circuits such as Shunt capacitor, Choke input or LC and PI circuits [7] and PI circuit has been adjudged the best of all but this project used the Shunt capacitor circuit (figure3.4.2), because it is cheap and simple to construct.



(a) Filter Circuit



(b) Filtered DC waveform

Figure 3.4.2

The value of the capacitor used was obtained as follow:

$$Q = CV \dots\dots\dots (3.7)$$

Where Q ——— charge across the capacitor

C ——— capacitance of the capacitor

$$V = V_{rmax}$$

V_{rmax} ——— maximum ripple voltage

Also, $Q = I \times T$

$$I = I_{max}$$

I_{max} ——— maximum current

$$I_{max} = \sqrt{2} \times I_s \dots\dots\dots (3.8)$$

I_s ——— Average Secondary output current = 500mA

T ——— period of the supply

$$T = \frac{1}{2 \times f} \dots\dots\dots (3.9)$$

f ——— Frequency of the ac supply = 50Hz

$$\therefore I \times T = CV$$

$$\therefore C = \frac{I \times T}{V}$$

$$V_{rmax} = 10\% \times (V_{max} - V_d) \text{ (by standard) [6]} \dots\dots\dots (3.10)$$

$$\Rightarrow V_{rmax} = 10\% \times 24.15 = 2.42V$$

$$I_{max} = \sqrt{2} \times 0.5 = 0.71A$$

$$T = \frac{1}{2 \times 50} = 0.01s$$

$$\therefore C = \frac{0.71 \times 0.01}{2.42} = 2.94E-3 f$$

$$= 2940 \mu f$$

by standard, the minimum tolerance value is 20% of the capacitance[6] of the evaluated

$$\begin{aligned} \text{i.e. } C &= \frac{120}{100} \times 2.94\text{E-}3 & \text{XI} \\ &= 3.53\text{E-}3\text{f} \\ &= 3530 \mu\text{f} \end{aligned}$$

But $3530 \mu\text{f}$ is not available in market and also, it has been proved that increasing the capacitance of a capacitor[6],[7]:

- Increases V_{dc} towards the maximum voltage V_{max}
- Reduces the magnitude of the ripple voltage
- Reduces the time flow of current pulse through diodes
- Increases the peak current in the diode

In view of these, $4700 \mu\text{f}$, 25V capacitor was used as the shunt capacitor. The circuit and the waveform are show in figure3.4.2

3.5 Voltage Regulator

The dc supply voltage changes with changes in load or input voltage, as a result of this the dc power supply becomes unregulated power supply [7] but electronics devices like programmers need a steady dc power supply therefore, there is need to stabilize or regulate the power supply. A stabilized or regulated power supply can be obtained from the unregulated supply by using a Voltage Regulator circuit.

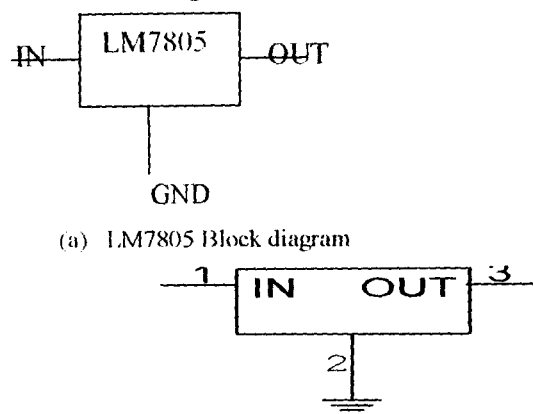
A regulator circuit is an electronics circuit that is capable of providing a nearly constant dc power supply even when there are variations in load or input voltage [7]. There are numerous regulator circuits but this project employed the used of Linear and Switching regulator circuits to obtain stable 5V and 12V, needed for programming.

3.5.1 Linear Regulated circuit

In linear regulators, the transistor operates somewhere between saturation and cut-off. There are two types of these regulators; these are Series and Shunt regulators [7]. Series

regulator was used in this project. The linear series regulators are regulator integrated circuits (IC) that have three terminal pins [6], where one of the pins serves as the input for the unregulated voltage (Pin 1), centre pin (Pin 2) is the ground and the third pin (Pin 3) serves as the output voltage terminal i.e. regulated voltage. The ICs have their output voltage fixed and they are of 78xx and 79xx series. The 78xx series' output voltage are positive voltage while the 79xx series' outputs voltage are negative voltage. Also, the last two digits (xx) indicate the voltage that will be supplied by the regulator.

A LM7805 was used to achieve the 5volts needed by the chips and the transceiver buffer IC MAX232. Its block diagram and schematic diagram are shown in figure 3.5.1.



(a) LM7805 Block diagram

(b) A LM7805 Schematic diagram

Figure 3.5.1(a) LM7805 and (b) LM7805 Schematic Diagram

3.5.2 Switching Regulator

In switching regulator s, the transistor operates like a switch i.e. it is either saturated or cut-off. They are of three basic types and they work as their names imply, these are Step-up, step-down and inverting types. This project used step-up switching regulator circuit in generating the programming voltage i.e. 12V.

In actualizing this, an adjustable voltage regulator LM317T was used in-conjunction with resistors to generate a 12V regulated voltage. The block diagram of the connection is shown in figure3.5.2a while the schematic is shown in figure3.5.2b

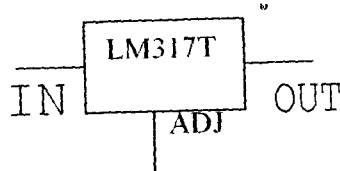


Figure 3.5.2(a) Block diagram

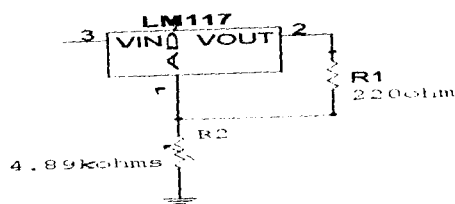


Figure 3.5.2(b) A LM317 Schematic diagram

The relationship between the voltage generated and the resistors is given as

$$V_{out} = 1.25(1 + R_2/R_1) \dots \dots \dots (3.11)$$

By standard the value of R_1 is 240Ω [6] or below it. A 220Ω resistor was used because the 240Ω was not available in market. The value of resistor R_2 was evaluated from eqn1 as follow:

$$V_{out} = 12V$$

$$R_1 = 220 \Omega$$

$$\therefore 12 = 1.25(1 + R_2/220)$$

$$\therefore R_2 = 220(12/1.25 - 1)$$

$$= 1892 \Omega$$

$$\approx 2k \Omega$$

But, this 12V won't be generated until a 2N2222 transistor that is connected as shown in figure3.5.3 is switched ON.

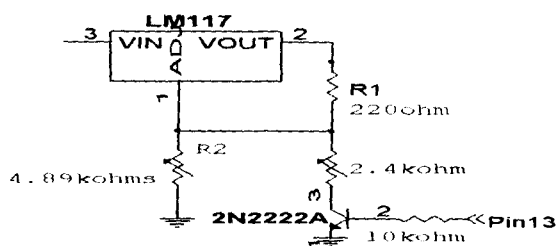


Figure 3.5.3 A Switching Regulator Circuit

2N2222 transistor is an NPN transistor and it is being used in the circuit as a logical switch that its operation base on the signal it receives from the Pin13 of My firmware(ROM). When the logic signal of Pin13 is '1', it will make the transistor to be ON which will then generate the programming voltage 12V. But, when the logic signal is '0', the transistor will

be OFF and this will the output voltage V_{out} of the regulator to be 5V. When the transistor is OFF, the resistor R_t will be in parallel with the resistor R_2 and their effective resistance in combination with resistor R_1 , generate the 5V.

The value of the resistor R_t was obtained as follow:

Using (3.11)

$$\text{i.e. } V_{out} = 1.25(1 + R'/R_1) \dots\dots\dots (3.12)$$

$$R' = R_1(V_{out}/1.25 - 1)$$

Where

$$V_{out} = 5V$$

$$R_1 = 220 \Omega$$

$$R' = R_2 || R_t$$

$$\begin{aligned} \therefore R' &= 220(5/1.25 - 1) \\ &= 220(4 - 1) \\ &= 220 \times 3 = 660 \Omega \end{aligned}$$

$$\text{but } R' = R_2 R_t / (R_2 + R_t)$$

$$\Rightarrow R_2 R_t / (R_2 + R_t) = 660$$

$$\Rightarrow R_2 R_t = 660(R_2 + R_t)$$

$$\text{where } R_2 = 2k \Omega$$

$$\therefore (2E3/660)R_t = 2E3 + R_t$$

$$3.03R_t = 2E3 + R_t$$

$$(3.03 - 1)R_t = 2E3$$

$$2.03R_t = 2E3$$

$$R_t = 2E3/2.03$$

$$= 985.2 \approx 1k \Omega$$

For a practical transistor, the V_{ce} can never be zero when the transistor switches ON rather it will add 0.6V to the output voltage V_{out} make it to be 12.6V [6],[7]. To avoid this and enhance accuracy and convenience, variable resistor of $4.8k\Omega$ was used as R_2 while variable resistor of $2.4k\Omega$ was put in place of R_1 . the variable resistor of R_2 was used to set the programming voltage while the Pin 13 logic signal was '1' and variable resistor of R_1 was used to set the 5V needed by the chips when the logic signal was '0'.

3.6 Surge Protection

Capacitors were connected across voltage input of chips, as to protect each chip against power surge. The minimum capacitance value of each capacitor used was obtained as follow;

Recall, $Q = CV$

Where Q ——— charge across the capacitor

C ——— capacitance of the capacitor

V ——— input voltage (chip voltage) = 5V

Also, $Q = I \times T$

I ——— chip current = 20mA

T ——— Time constant

$$T = \frac{1}{2 \times f}$$

For a 240V, 50Hz supply, a surge of a fraction of period (say one-tenth) then the time constant becomes

f ——— Frequency of the ac supply = 50Hz

$$T = \frac{1}{2 \times 50 \times 10} = 1\text{ms}$$

$$\therefore I \times T = CV$$

$$\therefore C = \frac{I \times T}{V}$$

$$\therefore C = \frac{20 \times 1}{5} = 4 \mu f$$

With the minimum tolerance value of 20% of the capacitance of the evaluated

$$\begin{aligned} \text{i.e. } C &= \frac{120}{100} \times 4 \\ &= 4.8 \mu f \end{aligned}$$

This means that the minimum capacitance of the capacitor required is $4.8 \mu f$ but $10 \mu f$ was used in this project to enhance better performance.

3.7 Fusing

Protection of circuit components against over-current is an ingredient of good circuit design and an attribute of a good Engineer. The designing of an electrical circuit in which if there is a fault in any section of the circuit, the nearest protective device(Fuse) is cut-off which makes other parts of the circuit safe and unaffected by the fault is called Discrimination. Effective discrimination can only be achieved, if the right protective device(fuse) is used. The current rating of this device must be twice or more than the current of the circuit but must less than the current rating of the connecting wire.

A number of ways are used to attain a good discrimination, such as fusing factor, diversity factor etc. the fusing factor is often used in electronic circuit and this project wasn't an exception. Fusing factor is obtained as follow;

$$\text{Fusing factor} = \frac{\text{minimum fusing current}}{\text{Circuit current rating}} \dots\dots\dots(3.13)$$

The more closer is the fusing factor to unity the more protected is the circuit. The fusing factor of this project was obtained as follow;

$$\text{Circuit current rating} = I_{\max} = 0.71 A$$

$$\text{Current rating of the fuse used} = 1 A$$

$$\begin{aligned} \therefore \text{fusing factor} &= \frac{1}{0.71} \\ &= 1.04 \end{aligned}$$

3.8 Microcontroller Chips

A microcontroller is a microcomputer on-chip. It has features of a typical microcomputer exception of not having the capacity to run more than one program. This program is stored on a memory integrated circuit called Erasable programmable Read Only Memory, which could either be Ultraviolet Erasable programmable Read Only Memory (UVEPROM) or Electrically Erasable programmable Read Only Memory (EEPROM), on the microcontroller chip itself before is being transferred on to the memory code.

A typical microcontroller has of three main memories; these are ROM, RAM and program code memory[5],[8]. The ROM stored the instruction sets (code) of the microcontroller which are in assembly language, the RAM is a series of registers used for moving data within the microcontroller, the number of the instruction sets determines the number of the RAM capacity. Program memory code stored the code for execution. The capacity of the program memory code is determines by the number of the address lines, if the number of the address lines is n , the number of the program memory code is 2^n .

There are a number of microcontroller families available with different instruction set, configuration and other features but all of them are used for controlling devices which is the brain of system automation. Out of this number of microcontrollers, this project shall used 8051 family to implement the designing and construction of a microcomputer based microcontroller programmer. This family is chosen because its members are easily to get, they have Universal Asynchronous/Synchronous Receiver and Transmitter (UART) feature incorporated that can be activated with a little codes, cheap and popularly used in industry, the table below shown some of the 8051 family variant [5].

Table 3.1 Popular 8051-variants

| Devices | Flash (bytes) | RAM (bytes) | Timers/Counter | Interrupt |
|----------------|----------------------|--------------------|-----------------------|------------------|
| 80C51 | 4K OTP | 128 | 2 | 5 |
| 80C52 | 8K OTP | 256 | 3 | 6 |
| 87C51 | 4K UVEPROM | 128 | 2 | 5 |
| 87C52 | 8K UVEPROM | 256 | 3 | 6 |
| 89LV51 | 4K EEPROM | 128 | 2 | 5 |
| 89LV52 | 8K EEPROM | 256 | 3 | 6 |
| 89C51 | 4K EEPROM | 128 | 2 | 5 |
| 89C52 | 8K EEPROM | 256 | 3 | 6 |
| 89C1051 | 1K EEPROM | 64 | 1 | 3 |
| 89C2051 | 2K EEPROM | 128 | 2 | 5 |

The microcontroller contains a number of different registers, part of which is the Special Function Registers (SFRs)[8]. Most of the SFRs are used for special functions as their names implied. Part of the SFRs are the Ports (P0-P3), TMOD, TCON, SCON, PSW etc. but out of these TMOD, SCON, TCON, PCON and T2CON if available are used to activate the UART feature of the microcontroller. The TCON and T2CON handles the timers, SCON handles the bit-mode of the serial communication and TMOD determines the timer mode, their respective table [5],[8],[9],[10], is shown below:

Table 3.2 TCON Table

| Bit | Name | Timer |
|-----|------|-------|
| 7 | TF1 | 1 |
| 6 | TR1 | 1 |
| 5 | TF0 | 0 |
| 4 | TR0 | 0 |

Table 3.3 SCON Table

| | | TCON | | | | SCON | | | |
|--------|----------|--|-----|-----|-----|------|-----|----|----|
| | | SM0 | SM1 | SM2 | REN | TB8 | RB8 | TI | RI |
| Symbol | Position | Name and Significance | | | | | | | |
| SM0 | SCON.7 | Serial port mode bit 0 (see table below). | | | | | | | |
| SM1 | SCON.6 | Serial port mode bit 1 (see table below). | | | | | | | |
| SM2 | SCON.5 | Enables the multiprocessor communication feature in Modes 2 and 3. In Mode 2 or 3, if SM2 is set to 1, then RI will not be activated if the received 9th data bit (RB8) is 0. In Mode 1, if SM2 = 1, then RI will not be activated if a valid stop bit was not received. In Mode 0, SM2 should be 0. | | | | | | | |
| REN | SCON.4 | Enables serial reception. Set by software to enable reception. Clear by software to disable reception. | | | | | | | |
| TB8 | SCON.3 | The 9th data bit that will be transmitted in Modes 2 and 3. Set or clear by software. | | | | | | | |
| RB8 | SCON.2 | In Modes 2 and 3, the 9th data bit that was received. In Mode 1, if SM2 = 0, RB8 is the stop bit that was received. In Mode 0, RB8 is not used. | | | | | | | |
| TI | SCON.1 | Transmit interrupt flag. Set by hardware at the end of the 8th bit time in Mode 0, or at the beginning of the stop bit in the other modes, in any serial transmission. Must be cleared by software. | | | | | | | |
| RI | SCON.0 | Receive interrupt flag. Set by hardware at the end of the 8th bit time in Mode 0, or halfway through the stop bit time in the other modes, in any serial reception (except see SM2). Must be cleared by software. | | | | | | | |

Where SM0, SM1 specify the serial port mode as follows:

| SM0 | SM1 | Mode | Description | Baud Rate |
|-----|-----|------|----------------|--|
| 0 | 0 | 0 | Shift Register | fixed ($f_{osc}/12$) |
| 0 | 1 | 1 | 8-bit UART | variable (set by timer) |
| 1 | 0 | 2 | 9-bit UART | fixed ($f_{osc}/64$ or $f_{osc}/32$) |
| 1 | 1 | 3 | 9-bit UART | variable (set by timer) |

Table 3.4 TMOD Table

| (MSB) | | | | (LSB) | | | |
|---------|---|------|--|--|-----|----|----|
| GATE | C/T | M1 | M0 | GATE | C/T | M1 | M0 |
| Timer 1 | | | | Timer 0 | | | |
| GATE | Gating control when set. Timer/Counter x is enabled only while INTx pin is high and TRx control pin is set. When cleared, Timer x is enabled whenever TRx control bit is set. | | | Timer 0 gate bit Timer 0 counter/timer select bit | | | |
| C/T | Timer or Counter Selector cleared for Timer operation (input from internal system clock). Set for Counter operation (input from Tx input pin). | | | Timer 0 M1 bit Timer 0 M0 bit | | | |
| M1 | Mode bit 1 | | | | | | |
| M0 | Mode bit 0 | | | | | | |
| M1 | M0 | Mode | Operating Mode | | | | |
| 0 | 0 | 0 | 13-bit Timer Mode. 8-bit Timer/Counter THx with TLx as 5-bit prescaler. | | | | |
| 0 | 1 | 1 | 16-bit Timer Mode. 16-bit Timer/Counter THx and TLx are cascaded; there is no prescaler. | | | | |
| 1 | 0 | 2 | 8-bit Auto Reload. 8-bit auto-reload Timer/Counter THx holds a value which is to be reloaded into TLx each time it overflows. | | | | |
| 1 | 1 | 3 | Split Timer Mode. (Timer 0) TL0 is an 8-bit Timer/Counter controlled by the standard Timer 0 control bits. TH0 is an 8-bit timer only controlled by Timer 1 control bits. | | | | |
| 1 | 1 | 3 | (Timer 1) Timer/Counter 1 stopped. | | | | |

| Timer SFR | Purpose | Address | Bit-Addressable |
|-----------------------|---------------------------|---------|-----------------|
| TCON | Control | 88H | Yes |
| TMOD | Mode | 89H | No |
| TL0 | Timer 0 low-byte | 8AH | No |
| TL1 | Timer 1 low-byte | 8BH | No |
| TH0 | Timer 0 high-byte | 8CH | No |
| TH1 | Timer 1 high-byte | 8DH | No |
| T2CON ⁽¹⁾ | Timer 2 control | 88H | Yes |
| T2MOD ⁽¹⁾ | Timer 2 Mode | 89H | No |
| RCAP2L ⁽¹⁾ | Timer 2 low-byte capture | CAH | No |
| RCAP2H ⁽¹⁾ | Timer 2 high-byte capture | CBH | No |
| TL2 ⁽¹⁾ | Timer 2 low-byte | COH | No |
| TH2 ⁽¹⁾ | Timer 2 high byte | CDH | No |

Note: 1. AT89C52 only.

For serial communication, the Timer1 is used, the TMOD is also set and the SM0 and SM1 are configured to the desired mode and the baud rate is obtained [8],[9],[10] as follow:

$$\text{Mode 0 baud rate} = \text{oscillator frequency} / 12 \dots\dots\dots (3.14)$$

$$\text{Mode 2 baud rate} = 2^{\text{SMOD}} \times \text{oscillator} / 64 \dots\dots\dots (3.15)$$

$$\begin{aligned} \text{Also, mode 1,3 baud rate} &= 2^{\text{SMOD}} \times \text{Timer1 overflow rate} / 32 \dots\dots\dots (3.16) \\ &= 2^{\text{SMOD}} \times \text{oscillator frequency} / (32 \times 12 \times (256 - \text{TH1})) \end{aligned}$$

but, if the baud rate is known then the TH1 reload value can be obtained as follow:

$$\text{i.e. TH1} = 256 - 2^{\text{SMOD}} \times \text{oscillator frequency} / (384 \times \text{baud rate})$$

SMOD is the state of the SMOD of the PCON, which could either be 1 or 0. Although, the PCON is not addressable but if the SMOD, which is the bit7 is set the baud rate is double.

This project made used of two AT89C52s, one as ROM and the remaining was used to test the programmer. The ROM is being used because the microcomputer communicates the programmer using serial transmission mode. So that with the aid of the firmware running on the ROM, the chip can be programmed using parallel transmission mode since microcontroller cannot be programmed through its serial port. The firmware was developed in C-language and compiled using SDDC compiler to microcontroller language i.e. assembly language. Although implementing a program using high level language consumes a lot of memory space compare to using assembly language but its readability, portability and maintainability in developing a complex project like this cannot be compromised to its size, this is the reason for chosen C-language.

The baud rate at which the programmer is communicating with the microcomputer based program was ensured to be same and this was incorporated into the firmware. The serial mode of the ROM was configured as follow:

$$\text{Recall, TH1} = 256 - 2^{\text{SMOD}} \times \text{oscillator frequency} / (384 \times \text{baud rate})$$

$$\text{Oscillator frequency} = 11059.20\text{KHz}$$

$$\text{Baud rate} = 9.60\text{KHz}$$

$$\text{SMOD} = 0$$

$$\therefore \text{TH1} = 256 - 2^0 \times 11059.2 / (384 \times 9.6)$$

$$\therefore = 256 - 3$$

$$\therefore = 253$$

$$\therefore = 0 \times \text{FD (hexadecimal)}$$

The serial mode was 1, that is SM1 which is the bit 6 of the SCON have to be enable also the REN and the TI that are the bit4 and 1, respectively are also enable because the programmer and microcomputer are using software handshaking. The value of the SCON was obtained as follow

- To enable SM1 that is the bit6 needs a value of $2^6 = 64$
- To enable REN that is bit4 needs a value of $2^4 = 16$
- To enable TI that is bit1 needs a value of $2^1 = 2$

Therefore, the value needs to configure the SCON = $64 + 16 + 2$

$$= 82 = 0x52(\text{hexadecimal})$$

Also, TMOD was configured to activate the Timer1 into 8-bit autoloader mode that would reload the TH1 with 0xFD, whenever the timer overflows but this can only be implemented if the Timer1 is activated and this done by enabling TR1, i.e. TR1 = 1.

All this are implemented in the firmware as follow:

```
SCON = 0x52;
```

```
TMOD = 0x20;
```

```
TH1 = 0xFD;
```

```
TR1 = 1;
```

The algorithms used for programming the microcontroller chips [8],[9],[10], is as follow:

- Power-up sequence:
- Apply power between VCC and GND pins and set RST and XTAL1 to GND
- Set pin RST to 'H' and set pin P3.2 to 'H'
- Apply the appropriate combination of 'H' or 'L' logic levels to pins P3.3, P3.4, P3.5, P3.7 to select one of the programming operations shown in the PEROM Programming Modes table.






To Program and Verify the Array:

- Apply data for Code byte at location 0000H to P1.0 to P1.7.
- Raise RST to programming voltage(12V) to enable programming.
- Pulse P3.2 once to program a byte in the PEROM array or the lock bits. The byte-write cycle is self-timed and typically takes 1.2 ms.
- To verify the programmed data, lower RST from 12V to logic 'H' level and set pins P3.3 to P3.7 to the appropriate levels. Output data can be read at the port P1 pins.
- To program a byte at the next address location, pulse XTAL1 pin once to advance the internal address counter. Apply new data to the port P1 pins.
- Repeat steps 5 through 8, changing data and advancing the address counter for the entire program memory(8kbyte) array or until the end of the object file is reached.
- Power-off sequence: set XTAL1 to 'L' set RST to 'L'
- Program Verify: If the lock bits have not been programmed code data can be read back via the data lines for verification:
- Reset the internal address counter to 0000H by bringing RST from 'L' to 'H'.
- Apply the appropriate control signals for Read Code data and read the output data at the port P1 pins.
- Pulse pin XTAL1 once to advance the internal address counter.
- Read the next code data byte at the port P1 pins.
- Repeat steps 3 and 4 until the entire array is read.

Flash Programming Modes:

- The internal FLASH address counter is reset to 0000H on the rising edge of RST and is advanced by a positive pulse at XTAL1 pin.
- Chip Erase requires a 10-ms PROG pulse.
- P3.1 is pulled Low during programming to indicate RDY/BSY.

Table 3.5 Programming mode

| Mode | RST | PSEN | ALE/PROG | EA/V _{pp} | P2.6 | P2.7 | P3.6 | P3.7 |
|---------------------|---------|------|--|--------------------|------|------|------|------|
| Write Code Data | H | L |  | H:12V | L | H | H | H |
| Read Code Data | H | L | H | H | L | L | H | H |
| Write Lock | Bit - 1 | H |  | H:12V | H | H | H | H |
| | Bit - 2 | H |  | H:12V | H | H | L | L |
| | Bit - 3 | H |  | H:12V | H | L | H | L |
| Chip Erase | H | L |  (t) | H:12V | H | L | L | L |
| Read Signature Byte | H | L | H | H | L | L | L | L |

- Chip Erase: The entire PEROM array and the Lock Bits are erased electrically by using the proper combination of control signals and by holding P3.2 low for 10 ms. The code array is written with all "1"s in the Chip Erase operation and must be executed before any nonblank memory byte can be re-programmed.

3.9 Microcomputer Interface

Interfacing is all about using computer with external devices or human, via protocol. Protocol is a set of rules or convention agreed upon by computer and the devices for communicating, this is to ensure that data exchanged are interpreted correctly.

Microcomputer communicates with the external devices through its Input/Output device units. Although, the Input/Output device units are seeing as memory but the designer always designated some location for this units, so that any device connected to any of the units can be used to control the operation of the microcomputer or the device, i.e. interfacing, by system Engineer. These designated location are referred to as Input/Output units but they are generally called Ports and these ports are classified to USB, Parallel and Serial ports(see below for their diagrams).

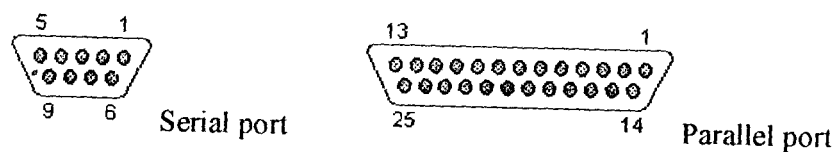


Figure 3.6 Microcomputer Ports

As the microprocessor coordinates whatever goes on in the microcomputer, it determines what happens at the ports through the a bus called System bus, via the protocol. A bus is a collection of wires on which electrical signals pass between components in the system. The system bus comprises of three buses, these are Data bus, Address bus and Control bus.

The data bus is used by microprocessor to shuffle data between various memory locations. The size of this bus varies from one microprocessor to another. The size is determined from the number of lines(bits) on the bus and this is also used to name the processor eg. Eight lines processor is called 8-bit microprocessor ,sixty-four lines is called Pentium

The address bus is used by microprocessor to locate the destination of the data. The number of address line determines the size of the maximum memory location that the microprocessor can access. If number of the address lines is n , the maximum memory locations the processor can only access is 2^n bytes locations, see the table below for better understanding.

Table 3.7 Some Common Microprocessors

| Microprocessor | Data Bus Size | Address Bus Size | Maximum memory Capacity | In computer Language |
|--------------------|---------------|------------------|-------------------------|----------------------|
| 8088 | 8 | 20 | 1,048,576 | 1Mbytes |
| 8086 | 16 | 20 | 1,048,576 | 1Mbyte |
| 80188 | 8 | 20 | 1,048,576 | 1Mbytes |
| 80186 | 16 | 20 | 1,048,576 | 1Mbytes |
| 80286 | 16 | 24 | 16,777,216 | 16Mbytes |
| 80486 | 32 | 32 | 4,294,976,296 | 4Gbytes |
| 80586/Pentium(PRO) | 64 | 32 | 4,294,976,296 | 4Gbyte |

$$M = 2^{20}$$

$$G = 2^{30}$$

The control bus is used by the microprocessor to control the data on the data bus. The two common lines to all microprocessor on the data bus, are Read and Write signal lines. The Read line makes the microprocessor to receive data on the data bus from a specific memory location while the Write line makes the microprocessor to write data on the data bus to memory location on the address bus. Other lines on the control bus are Interrupt, Status lines etc, but with a good understanding of how to use them, they can be used to control connected external devices to the ports.

If a parallel port, i.e. parallel mode of transmission, is to be used to communicate with external device, all the three buses that make up the system bus are incorporated directly in the protocol because all the pins connected to these buses are made visible at the port and this makes the circuitry of the external device complex. But, if serial port, i.e. serial mode of data transmission is to be used, what are needed to be done are to identify the four memory locations allotted for serial ports(COM1 – COM4), number of the communicating

bits and the baud rate. All these are to be incorporated in the protocol while the address bus and control bus can be implemented externally, by using counters, decoders and part of the serial port pins to trigger them.

This project used serial transmission mode and the protocol was developed using Visual Basic programming language. The serial transmission mode was used, so that the program can be run on any windows operating system without creating any Dynamic Link library (DLL), as the case using Visual Basic program over the parallel port on some windows operating system.

CHAPTER FOUR

Construction, Testing and Result

4.1 Construction

A number of construction methods are available but this project employed Top-Down design method, which involves breaking down the project design into modules or units for easy construction, testing and troubleshooting the circuit, in case of any fault.

The Top-down design method made this project to be divided into two main parts; these are hardware and software parts. The hardware was made into power unit, chip and communication modules.

4.1.1 Hardware Modules

Power Module: this handles the generation of 18V voltage supply, its rectification, filtration and regulation to the required programming voltage.

A transformer of voltage rating 220/18V ac was the first component installed and its secondary terminal wires (output 18V) were connected to the bridge rectifier, as shown in figure3.3.1a. This was done to convert the ac supply to dc supply.

The filtration of the ripple contained in the output of the rectifier circuit was done by connecting an electrolytic capacitor across the output of the rectifier circuit, as shown in figure3.4.2.

The output voltage of the capacitor was used to feed the adjustable regulator LM317T to obtain the programming voltage for the chip. This was achieved with the aid of an NPN transistor, 2N2222 that served as a switch for switching between 12V and 5V, as required by the chip. The switching of the transistor is base on the logic signal it receives from Pin13 of the myfirmware (ROM), where the logic signal '1' means 12V and logic signal '0' means 5V, as illustrated under the switching regulator.

Also, the output of the capacitor was used to feed a fixed regulator LM7805, which maintained 5V at its output terminal. The 5V generated would power the chips and the

transceiver buffer IC, MAX232. Finally the circuit was protected with 1A fuse on the live line of the transformer to the rectifier diode.

Chip Module: this module comprises the MAX232, ROM chip that controls the programming of the chip to be programmed and 40-pin socket IC, where the chip to be programmed would be inserted.

Communication Module: this is the last module of the hardware modules and it handles the exchanging of data between the microcomputer and programmer. This project was designed in a such way that microcomputer communicates with the ROM of the programmer using serial mode of transmission via the buffer IC MAX232 while the ROM communicates with the chip to be programmed by parallel mode of transmission. The serial mode was achieved by set the baud rate of both the microcomputer and ROM equal (11.0592MHz) and other properties with the aid of the programs running on ROM chip and microcomputer. The programs were able to communicate with the devices through the help of an RS232 (DB9) cable, which connected the microcontroller programmer and microcomputer.

Although, the RS232 cable and microcomputer serial port have nine pins each but, this project made use of three pins; transmitting (TD), Receiving (RD), Ground (GND) pins. The transmitting (TD) pin of the microcomputer is connected to receiving (Rx) pin of the ROM chip while its transmitting (Tx) is connected to the receiving (RD)pin of microcomputer, so that devices can be able to transmit and receive. The pin configuration is shown in the table below:

Table 4.1 Pin configuration table

| The pin function | Microcomputer pins | Microcontroller pins |
|------------------|--------------------|----------------------|
| Receiving pin | 2 | 10 |
| Transmitting pin | 3 | 11 |
| Ground pin | 5 | 20 |

4.1.2 Software Module

This module handles the programming of the chip, which is the combined effort of the firmware running on the ROM called "myfirmware"(see appendix B) and another one running on the microcomputer called "mycode"(see appendix C). The firmware was written in C-language, C-language was used because it is structural based language unlike the Assembly language that is object-based language. the program was compiled by using SDCC, which is one of the available freeware for 8051 C-complier. On the other hand, the microcomputer based program i.e. mycode, was developed using Visual Basic, a user interactive program. The Visual Basic was chosen because of its Graphic User Interface(GUI) that makes it user interactive program.

4.1.3 Components Layout

Before the active construction, the power unit module was simulated, using Multism while the software was also simulated, using the appropriate simulators.

During the active construction, the following steps were carried out while constructing the hardware:

The circuit was first built on a Breadboard before final design was done on the Vero board; this made circuit modification and fault location easy.

The connection of the circuit components was mapped out on a piece of paper, in order to minimize space and number of the connecting wires to be used.

The circuit was built in module to enhance easy identification components and each output of the modules was compared with the simulated result before proceeding to the next one. Adequate heat sinks were used for the regulators.

All the ICs were mounted on IC sockets that have been soldered to the Vero board, to ease replacement of the faulty or bad one in future.

Digital multimeter was used to check the continuity of the connecting wires and components where necessary.

The digital multimeter was also used to measure the voltage, current and resistance where necessary.

Alloyed lead was used and the soldering was carefully done. Razor blade was used to expose the copper surface before soldering was done.

The baud rate of both the microcomputer based program and the firmware were the same.

4.1.4 Construction Tools Used

During the modeling and construction, the following tools were used:

Microcomputer: This was used in designing and simulating the hardware and software.

Breadboard: The prototype was built on it

Veroboard: The circuit was built on it, permanently.

Connecting Wires: These were used for connecting the components.

Digital Multimeter: This was used to test the continuity of the lines, to measure voltage, current and resistance of various components in the circuit during construction.

Suction Tube: This was used to suck the desoldered leads away from unwanted areas.

Razor blade/cutting pliers: the razor was used to expose the copper surface while the pliers were used to cut the connecting wires and components' legs.

Screw Drivers: They were used to tie the screw that tied the circuit board and transformer to the stands in the casing.

Soldering iron (60W): this was used to melt the lead during soldering and desoldering.

4.1.5 Construction Diagram

The circuit diagram and its components are given in appendix A.

4.2 Testing

After the completion of the construction of the microcontroller programmer, the following tests were done:

The continuity of the connecting wires and components used was tested using multimeter.

The voltage at each module was measured using multimeter. The 5V and 12V were checked when none of the chips was inserted by applying 5V and 0V, respectively at pin13 of the ROM and measured the voltage at pin31 of socket of the chip that will be programmed. The result is shown in table 4.2.

The circuit was tested with the software by programming a chip that automatically switched ON a LED when there was darkness and OFF when there was light. The program code, written in assembly language is given below while its circuit diagram is shown in appendix A as circuit 2.

```
Org 0000
as: jnb p2.0,ay
    clr p0.0
    sjmp as
ay: setb p0.0
    sjmp as
    end
```

4.3 Result\Discussion of result

Every module of the programmer worked well, even the transistor that served as a switch between 12V and 5V, as its switching operation is shown in the table below;

Table 4.2 Transistor switching operation

| Voltage at Pin13 of the ROM chip(V) | Voltage at Pin31 of the target chip (V) | Remark, base on the transistor operation i.e. its switching operation |
|-------------------------------------|---|---|
| 0 | 12 | OFF |
| 5 | 5 | ON |

The programmer was able to program the test chip with the above codes.

The result obtained from the programming a microcontroller chip that controlled the lighting of a bulb, shows that the microcontroller programmer can be used to program any microcontroller of 8051 family or chip that have the same pins configuration with the 8051 family.

CHAPTER FIVE

Conclusion And Recommendation

5.1 Conclusion

The design and construction of a microcomputer based microcontroller programmer was successfully done.

The automatic lighting of the LED is an evidence that microcontroller is an important ingredient in system automation.

It can also be deduced from this project that industrial production of this microcomputer based microcontroller programmer will not only improve the economy of Nigeria but also aids the usage of microcontroller in automation in this country of ours.

5.2 Recommendation

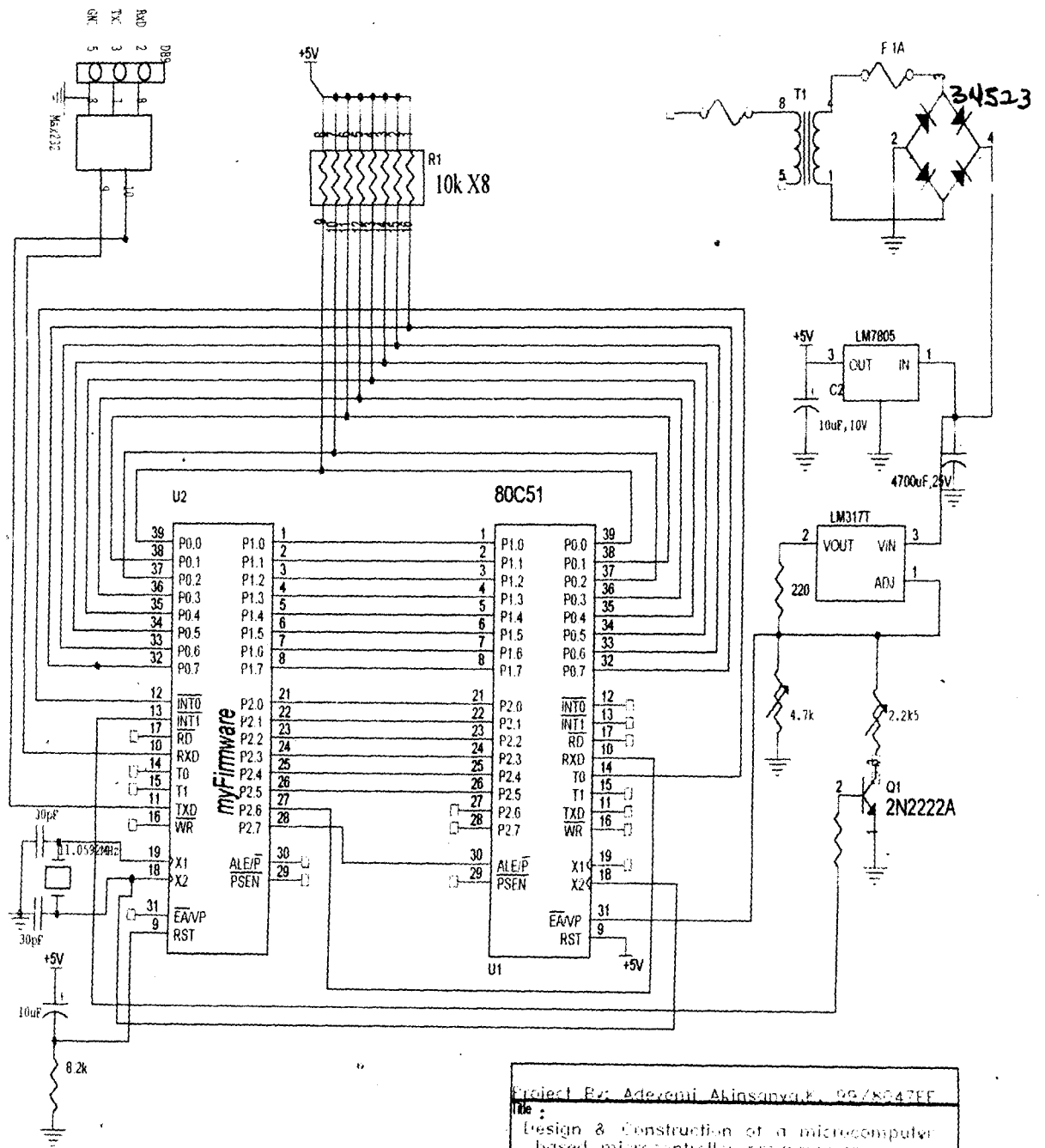
In view of the experience gained in the course of executing this project, the following recommendations are made:

- This project should be modified to suite the programming of Attiny and other pins configuration.
- Special attention should be given to the programming voltage of the chip to be programmed because different microcontrollers have different programming voltage so, the R_2 and R_1 should be used to set the required programming voltage
- Zener diode should not be used across the output of the adjustable regulator, so that the device can be used to program any microcontroller of the same pins configuration of different programming voltage by simply adjusting R_2 and R_1 .
- The program to be burnt on the chip should be simulated before it's being burnt onto the chip.

REFERENCES

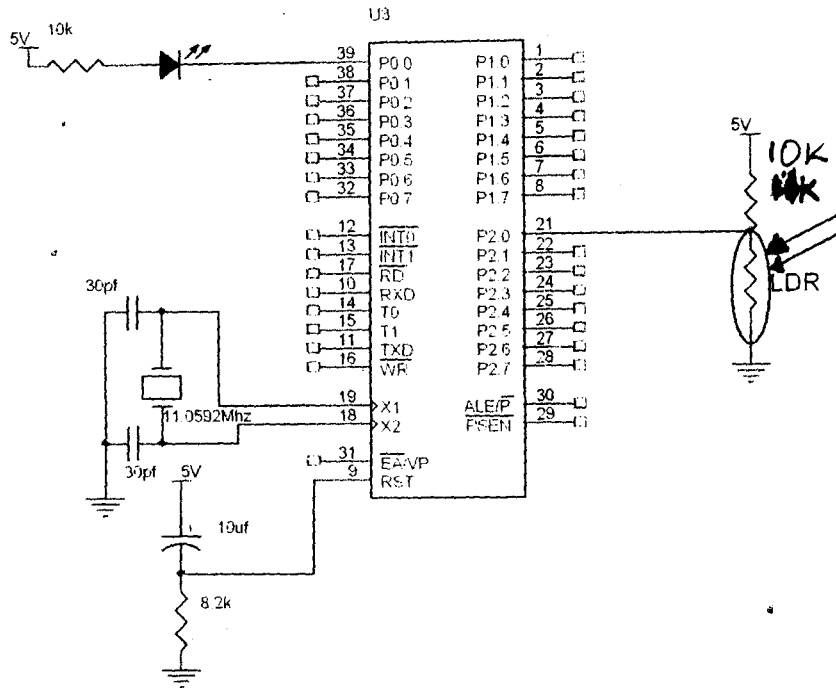
1. Y. Liu and G.A.Gibson, Microcomputer systems of the 8086/8088 family; architecture ,programming and design, Prentice-Hall, New Delhi, 1998, pp1-2.
2. W.M. Fuori and J. A. Lawrence, Computer and information processing, 2nd Ed., pp 6-34,
3. E.J. Laurie and R. D. Onatoro, Computers, automation and society, Irwin, 1979, pp15
4. K.G. Beauchamp, Computer communication, 2nd Edition, 1990, pp89-92
5. Dogan Ibrahim, Microcontroller project in C for the 8051, Biddles Ltd, Guildford and King's Lynn, Great Britain, 2000, pp1-13, pp147-151.
6. Paul Horowitz and Winfield Hill, The art of electronics, 2nd Edition, Cambridge University Press, Great Britain, 1995, pp 45-48, pp341-345, pp559-612, pp719-728.
7. B.L. Theraja and A.K. Theraja, A textbook of electrical technology, 22nd Edition, S. Chand & Company LTD., New Delhi, 1999, pp1699-1719.
8. <http://www.ATMEL.com>, "8051 Architecture ", May 2002.
9. <http://www.8052.com/forum>, " 8052 Tutorial & Reference", 1997-2004, pp5-49.10
10. [http:// www.epemag.wimbome.co.uk/resource](http://www.epemag.wimbome.co.uk/resource). "Everyday Practical Electronics", June 1998, pp 427-437.
11. NTE Electronics, "Semiconductors", 11th Edition, NTE Electronics Inc, 2002, pp1-12 -1-161.

Appendix A



| | | |
|---|-------------------------------------|------|
| Project By: Adeyemi Alinsanyak 09/8047EE | | |
| Title : Design & Construction of a microcomputer based microcontroller programmer | | |
| Size A | Document Number : Circuit Diagram 1 | Rev |
| Supervised By: Engr. Engru | | |
| Date: Sunday, November 27, 2005 | Sheet 1 | of 1 |

Circuit 2



| | | |
|---|--|--------|
| Project By: Adeyemi Akinsanmi, 2018047EE | | |
| Title : Design & Construction of a microcomputer based microcontroller programmer | | |
| Size A | Document Number : Circuit Diagram 2 | Rev |
| Supervised By: Engr. Egunnu | | |
| Date: Monday, November 28, 2005 | Sheet | 1 of 1 |

Appendix B

```
#include <at89x51.h>

#include <stdio.h>

#define xon 0x11

#define xoff 0x13

int i;

unsigned char ACCU,temp,blank,chip,VPP;

char command;

unsigned int address,checksum;

#define LM317 P3_5

#define LE P3_7

#define prog P2_7

#define Vpp P3_3

#define rdy P3_2

#define xtal P3_1

#define p26 P3_4

#define p27 P3_5

#define p36 P3_6

#define p37 P3_7

#define p10 P1_0

#define p11 P1_1

#define p12 P1_2

#define p13 P1_3

#define p14 P1_4

1ms()

{int i;for(i<130;i++)}

delay(int n) {int i;for(i=0;i<n;i++) 1ms();}

10us() {char i; for(i=0;i<20;i++)}

delay200us(){char i;for(i=0;i<20;i++)10us;}

pulseprog(){prog =0; delay200us();prog=1;delay200us}
```

```
read(){unsigned int i;if (command=='r'){chksum=0;Vpp=1;prog=1;p26=1;p27=1;p36=1;p37=1;
for(i=0;i<count)
main()
{i=0;count=0;
SCON=0x52;//8-bit UART mode
TMOD=0x20;//timer1 mode 2 autoreload
TH1=0xfd;//9600 N 8 1
TR1=1;//run timer1
getchar();write();read();}
```

Appendix C

MYCODE

```
Private Sub cmdRead_Click()
    Dim s As String
    Dim count As Integer
    Dialog.ShowOpen
    s = Dialog.FileName
    Prog1.Max = Len(s)
        MSComm1.Output = s
        With Prog1
            .Visible = True
            .Value = Prog1.Max Mod 10
        End With
    End Sub

Private Sub Form_Load()
    MSComm1.Settings = "9600,n,8,1"
    MSComm1.PortOpen = True
    MSComm1.Output = 13
    If MSComm1.Output = 10 Then
        cmdwrite.Visible = True
        cmdRead.Visible = True
    Else
        Label1.Caption = "Try to connect:COM1"
    End If
End Sub

Private Sub Form_Unload(Cancel As Integer)
    MSComm1.PortOpen = False
End Sub

Private Sub MSComm1_OnComm()
    Do While (MSComm1.Comm1Event = com1vReceive)
        msd = msd & MSComm1.Input
    End Do
End Sub
```

```
With Prog1
.Visible = True
.Value = Prog1.Max Mod 10
End With

Loop

Prog1.Visible = False

End Sub

Private Sub cmdRead_Click()

Dim s, msd As String

Dim count As Integer

Dialog.DialogTitle = "Reading Hex file"

Dialog.Filter = "Hex files (*.hex)|*.hex"

Dialog.ShowSave

Dialog.FileName = " "

s = Dialog.FileName

Open s For Binary Access Write As msd

Prog1.Max = MSComm1.InBufferSize

msd = MSComm1.Input

End Sub
```