DESIGN AND CONSTRUCTION OF A MICROPROCESSOR – BASED CALCULATOR

BY

SULE USMAN

(93/3705)

A PROJECT SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE AWARD OF A DEGREE OF BACHELOR OF ENGINEERING.(B. Eng.) DEGREE IN THE

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

SCHOOL OF ENGINEERING AND ENGINEERING TECHNOLOGY

FEDERAL UNIVERSITY OF TECHNOLOGY MINNA

MARCH, 2000.

DECLARATION

I hereby declare that this project work presented for the degree of Bachelor of Engineering has not been presented either wholly or partially for any other degree elsewhere. I also declared that this work was carried out by me under the supervision of Mr. Attah of Electrical and Computer Engineering Federal University of Technology, Minna during 1998/1999 academic session.

---------------------------- [illegible] 18/9/2000

SULE USMAN SIGN/DATE

CERTIFICATION

This is to certify that this project titled design and construction of a microprocessor based calcul;ator was carried out by Sule Usman under the supervision of Mr. Paul Attah and submitted to Electrical and computer Engineering department, Federal University of technology, Minna, in partial fulfillment of the requirement for the award of Bachelor of Engineering (B. Eng.) degree in Electrical and Computer Engineering.

MR. PAUL ATTAH

SUPERVISOR

[illegible] 24/03/2000

SIGN/DATE

DR. ENGR. A.Y. ADEDIRAN
HEAD OF DEPARTMENT

[illegible] [illegible]

SIGN/DATE

Dr. J. O. ONI

EXTERNAL EXAMINER

[illegible] 15/4/2000

SIGN/DATE

DEDICATION

This project work is dedicated to my parents ;

Alh. Ahmed Usman and Hajiya Balikisu Usman.

AKNOWLEDGEMENT

All praise is due to the Almighty God, the creator of life and death, who gave me life and the will-power to be able to start this programme as well as finishing it successfully.

I deeply appreciate, and say thank you for all the love and support my Mum and Dad have given to me all my life. I am most thankful to my step mother Hajiya Zainab Usman, for all her love, words of encouragement and support towards my success in my programme.

I wish to thank my supervisor: Mr. Paul Attah, for his patience and understanding. He supervised this work with enthusiasm and consideration. To him I am grateful.

My gratitude also goes to Mr. Danjuma whose assistance toward the success of this programme is of great importance. I would like to acknowledge the Head of the Department. Dr (Engr) A.Y. Adediran for his guidance and encouragement I also wish to thank all the staff of my department for their co-operation.

To my brothers: Salisu, Mohammed (Alhaji) Mahmood, Heavy, Baba-omchi, Adazaki, Adagimba, Temim, Jibrin, Gambo, Aboobakar and lots more. Thank you all for your co-operation and being wonderful.

To my sisters: Ramatu, Amina, Adazaki, Kande, Lami, Maryam Jibrin, Asabe, Aishat, Talatu, Hussaina and Hassana, and lots more. I will love to say thank you all.

To my friends: El-kabir (the chairman), Bonu, A.Y.U., Tanko, A.T. Ahmed, Hamza, El-Amin, Soole (Azooks), Doctor and host of others, I am grateful for your understanding and words of encouragement.

Finally, to my beloved friends: Ahmed Sadiq to whom I am indebted to for he made this project a success and my project colleaques El-amin and Soole (Azooks) I will love to say thank you all for a job well done.

ABSTRACT

A microprocessor-based system depends solely on the software in the ROM. which is connected to the microprocessor. There exist different types of ROM but the best for this kind of work is the erasable programmable Read only memory (EPROM). The EPROM could be erased and reprogrammed. The software which are programmed into the EPROM are set of instruction which tells the microprocessor what to do. In lieu of this a microprocessor-based calculator could be upgraded in other words its function could be changed to suite your taste. Hence, a microprocessor-based calculator is advantageous over its counterpart that is not software dependent {i.e electronic calculator based on hardware only}.

The calculator has been designed to perform four primary functions. Addition. subtraction, multiplication and division of four digit numbers.

Chapter I " The introduction" . This chapter gives an insight to the origination of calculator and its technological advancement to present date. Chapter 2 "The System hardware" and software design contains the description of the various system hardware (components) and software used in the design. Chapter 3 "System development" explains how the hardware was constructed and the software compiled and integrated into the system.

TABLE OF CONTENTS

CHAPTER ONE

INTRODUCTION:

1.1 Historical Background

Since the dawn of civilization man has been faced with the problems of performing calculations and recording data. Primitive man using fingers performed the first form of calculation. However, fingers cannot store numbers and one of the early ways of achieving this was by carving notches on bone or wood. The resulting "tally sticks" as they are called date back over 30,000years.

Another form of early counting device was the abacus – a frame of rods or wires with beads attached, which according to their position, represent different values; calculations are performed by moving the beads about according to a set of rules.

In 1643, a French mathematician and philosopher, Blaise Pascal invented the first mechanical calculating machine (calculator). The Pascaline, as it was called, could add and with an adjustment, subtract. Gottfried Lelbuiz realized that pascaline's great weakness was that multiplication could only be achieved by repeated additions and division could only be achieved by repeated subtractions. He overcomes this problem by inventing the stepped wheel with nine teeth of different lengths, which made multiplication a distinct operation. In 1820, Charles Xaver Thomas, improved previous models by the addition of a hand crank drive mechanism. Through out the latter half of the nineteenth century, and well into the twentieth century, mechanically based calculators continued to be developed. The comptometer, a key operated adding machine, was invented by Dorr Felt (USA) in 1884 and in 1892 William Seward Burrough, also in the USA, developed a 90-key machine with a nine decimal digit capacity.

The next major improvement came with the electrification of calculating machines (calculator), a step that greatly increased the speed at which computation can be done. In the 60's the first practical electronic calculators, using transistors and computer industry technique were introduced.

The world's first microprocessor, the intel 4004, designed by Robert Noyce and his team in 1971 had not come about as a conscious effort on intel's part to develop a computer on a chip but more as a result of there desire to produce a flexible set of component parts for a range of programmable calculators. The introduction of this miniaturized circuit has made possible the development of small versatile calculators powered by battery(ies). Continuous technical developments made calculators inexpensive and capable of solving large varieties of numerical problems. They came into widespread use that the abacus, adding machines and slide rules became virtually archaic.

1.2 Literature Review

The problems of most electronic circuit in recent times have been overcome by the introduction of digital components, which has the ability for efficiency and high precision. Analogue systems though may not be completely ruled out entirely in its own merit cannot match the speed and accuracy of the digital systems.

A calculator contains three main components: A keyboard, the central processing unit (CPU) and a numerical display. The CPU, which is made up of microelectronic chip and the display, are powered either by batteries or by household electricity supply. The keypad consists of a set of switches that route electrical signals representing instructions and numerical information to the chip. The chip is an integrated circuit made with metal oxide semi conductors that is designed to carry out the functions of the calculator.

Integral circuits contains thousands of transistor called Logic gates – Logic gates are device that produce electrical signal of high or low voltage in a way that depend on the voltage of the signal or signals they received as input. Advances in integrated circuits technology lead to the ability to integrate larger numbers of logic gate on a single integrated circuit chip. Small scale Integration, SS1, allowed up to 12 gates to be integrated on a single chip. ,This resulted in IC's, that provide the basic logic functions, AND, OR INVERT and so on and the basic storage elements D and J – K flip-flops. Medium scale Integration, MS1, allow from 13 to 99 gates or gate equivalent to be integrated on a single chip. Since the number of pins associated with an IC package is limited, it was necessary to interconnect these logic gates on the IC to implement single functions such as counters, decoders, registers comparators and adders. When the capability to integrate 100 or more logic gates on a single chip, large scale integration, LSI, was achieved, it became possible to have an ALU and control on a single chip and this IC is known as a MICROPROCESSOR

A microprocessor-based calculator is a calculator whose function depends on the software that is programmed into the memory of the microprocessor. Unlike the electronic calculator, the microprocessor-based calculator's hardware is its skeleton (which cannot do anything of its own), while the software is its brain. If the area of application is to be altered, the previous software or instruction sets in the EPROM of the system are erased by exposing the contents to high intensity ultraviolet light, via a quartz and the new software or instruction set are programmed into the EPROM. The microprocessor-based calculator can be upgraded to work as a microprocessor-based single board microcomputer.

CHAPTER TWO

THE SYSTEM HARDWARE

2.1 System hardware specification

The specification are as follows:-

i) The INTEL 8085, an 8-bit processor

ii) Crystal of 6.125MH_z

iii) An EPROM 2716 – 2k by 8-bit

iv) An SRAM 6116 – 2k by 8-bit

v) Common cathode display – 4 digit 7 –segment multiplexed display

vi) Keypad – "4 x 4" matrix keypad

vii) Connector cables – secondary winding of a transformer

viii) A +5volt regulated power supply.

2.2 The modular concept

This involves the breaking of a system into modules. Each module performs a particular function and together they satisfy the requirement of the system. This system is made up of three basic modules and a power supply unit. The modules are:-

i) The processor module

ii) The input – keypad module

iii) The output – display module

2.3 System functional block

The calculator is based on the intel 8085 microprocessor. It is an 8-bit microprocessor. The clock logic of the 8085 is provided on the chip, hence a simple crystal is used to produce the necessary clock signal to the microprocessor. The microprocessor is automatically reset at power ON by an RC circuit connected to the RESET IN input of the 8085. Once the system is operating, it can be reset manually by the push button switch. The address bus and data bus of the microprocessor is buffered using the buffer blocks. The buffer for the data bus is bi-directional as the data bus is used to send and receive data from the microprocessor. The buffer for the address bus is unidirectional.

The time multiplexed address/data pins is externally latched and held during the remainder of the memory reference to provide address bits. The 8-bit latch –latches the address information from the address/data pins when clocked by the address latch enable signal, ALE.

The 8085 generate two control pulses to indicate whether it is reading, RD or writing, ¯WR, an external register. Another control signal IO/M, generated by the 8085 indicates whether the microprocessor wants to read or write memory or I/O.

The EPROM 2716, 2k by 8-bits EPROM – is the resident read only memory of the system the EPROM maintained stored data even when power was removed from the system. The program was stored in this type of memory and was available for execution as soon as power was applied to the system. RWM(RAM) 6116, is the memory that is used temporarily during the execution of the system – this memory can be written to and as well read from hence the name RWM (read write memory). The output of this system is a 4-digit seven-segment display. The output has two ports one of the ports, provide the digit select and the other port provides the segment select. The input of this system is a 16-button keypad ("4 x 4 matrix scan keypad") that uses the scan matrix technique. Just like the output, the input has two ports, one of the port is used for row scanning and the other for column scanning. The diagram is shown in fig.2.3

2.4 The INTEL 8085

The intel 8085 is an 8-bit microprocessor. It is a single chip NMOS device contained in a 40-pin dual in line package.

The 8085 operate on a single 5v power supply connected at Vcc, power supply ground is connected to Vss. The clock logic of the 8085 is provided on the chip so that only a simple crystal of 6.125mH$_z$ is needed. The frequency of the internal clock generator, which synchronized the operation of the 8085 is determined by the crystal connected at pins X_1 and X_2. The input frequency of 6.125mH$_z$ by the crystal is divided by 2 using a divide by 2 counter (7474) to give the processor internal operating frequency.

When power is applied to the microprocessor, the various registers and flip-flops assume random states, and its operation is unpredictable. Therefore, the microprocessor must be reset when it is first powered, in order to fetch the first instruction. The 8085 is automatically reset at power ON by an RC circuit connected to the RESET IN input of the processor. When power is first applied to the circuit, the voltage across the capacitor is O volt, the capacitor charges at a rate determined by RC to a final voltage of Vcc. Note that values of R and C are selected to maintain RESET IN at the logic O for the required amount of time. Once the system is operating, it can be reset manually by the

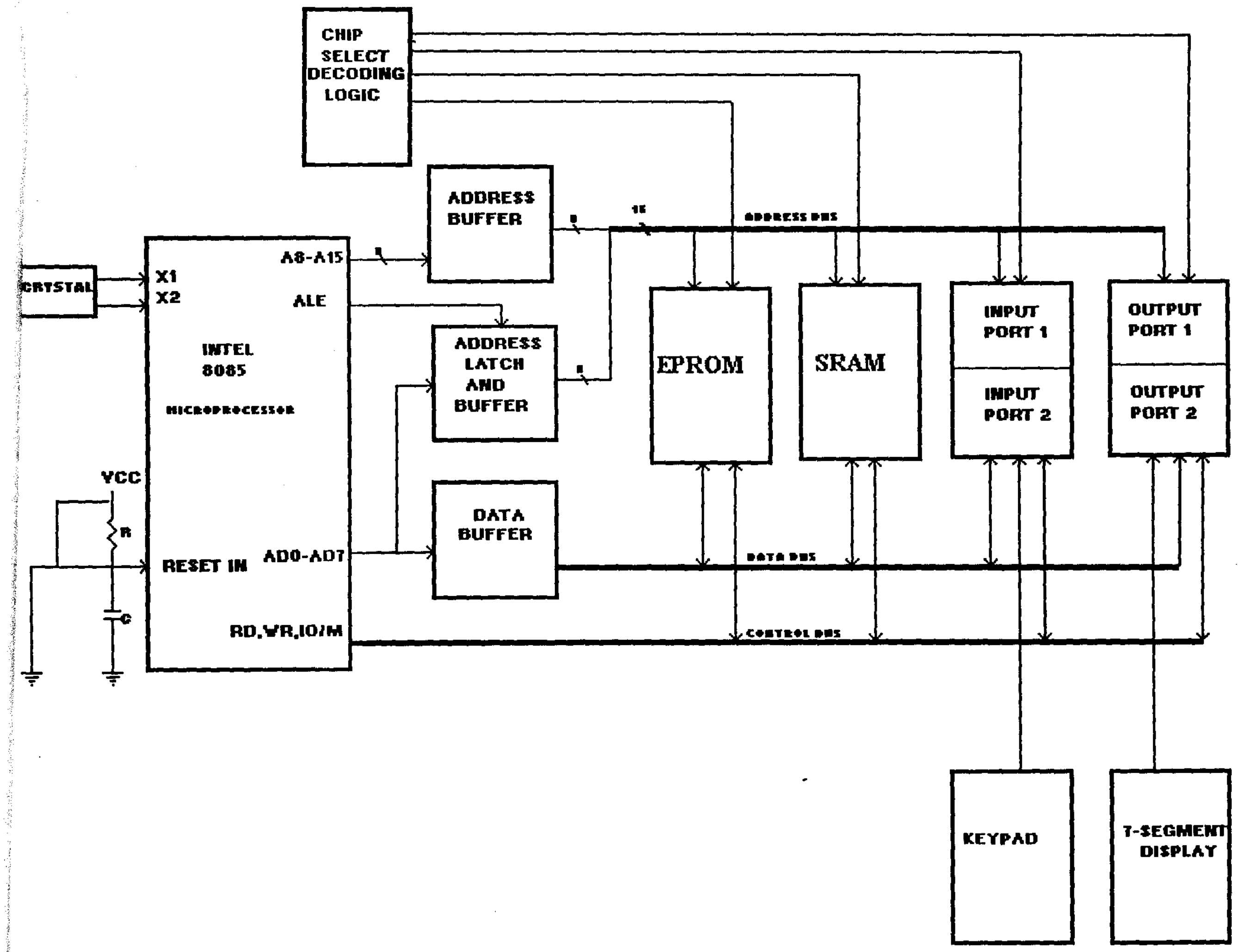


FIG 2·3 BLOCK DIAGRAM OF A MICROPROCESSOR BASED DIGITAL CALCULATOR push button switch, pressing the switch, short the capacitor and discharge it when RESET IN is logic 0, RESET OUT is logic 1. RESET OUT is used to reset external devices in the microprocessor system.

The 8085 generate two control pulses to indicate whether it is reading, RD, or writing WR, an external register. Another control signal, IO/M generated by processor indicates whether the microprocessor wants to read or write memory or input or output. When this signal IO/M, is 0 memory is being referenced; when it is logic 1, input/output is being referenced.

2.5 INTEL 8085 Bus buffering and control

The 8085 is capable of directly addressing up to 64k memory locations with its 16-bit address. Eight of the 16 bits, $A_8 - A_{15}$ are provided directly on the address pins, $A_8 - A_{15}$. The other 8-bits $A_0 - A_7$, are provided on the bi-directional address/data pins, $AD_0 - AD_7$. The address data pins are time multiplexed. At times carrying addresses at other times carrying data. A 74ALS573 octal transparent latch demultiplexes the low address byte $A_0 - A_7$. The address latch Enable ALE, clocks an external register that latches the low order address byte. Address lines are unidirectional and as such there buffering is straightforward. Address buffering for $A_8 - A_{15}$ is provided by a 74LS244 octal buffer. Address lines are unidirectional because it is used only for addressing of memory and I/O devices when used to address memory, ideally, the address bus contains a 16 bit address which allow any of 64k different memory location to be read from or written to by the intel 8085. And when used to address the I/O devices, the address bus contain an 8-bit I/O address that is used to one of 256 different I/O devices.

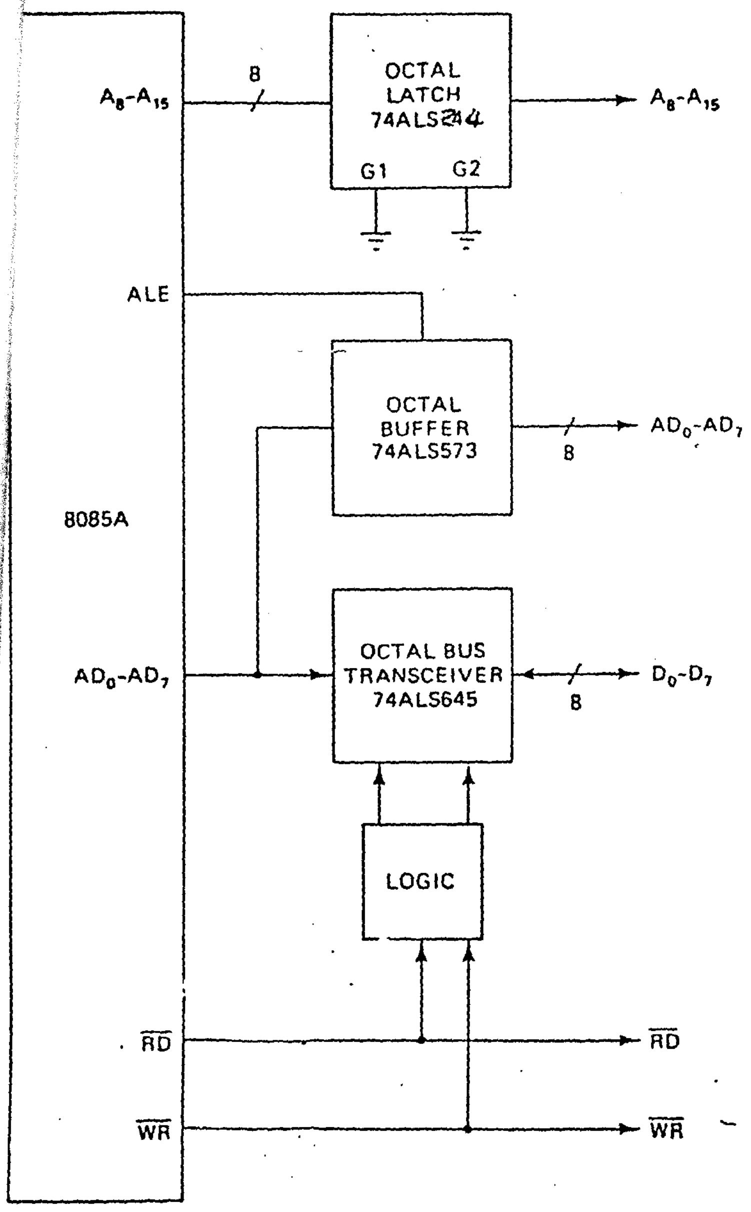


FIG 2·5 8085A with buffered address, data, and control bus.

The Data bus, $D_0 - D_7$ from the address/data bus are used to transfer data between the microprocessor and the memory or I/O. The data bus is an 8-bit bus. The data bus lines are bi-directional, their buffering requires special consideration. The buffering in this case is implemented using a 74LS645 octal bus transceiver.

The 8085 generate two control pulses to indicate whether it is reading RD, or writing WR an external register. Another control signal,10/M generated by the 8085 indicates whether the microprocessor wants to read or write memory or input/output. When this signal is logic O^0 memory is referenced, when it is logic 1, input/out is being referenced. The signals RD, WR and IO/M are used together in the system design to control the reading and writing of external memory and input/output.

The 8085 do not provide control output specifically for controlling a bus transceiver. However, with a minimal amount of logic, the existing control and status was used. The control thus: G was kept at logic 0 and controls the direction of the buffer using RD thus the buffer is enabled and in the output direction until a RD strobe occurs. During the RD strobe, the buffer is in input direction.

The diagram of fig. 2.5 illustrates 8085 bus buffering and control.

2.6 System memory

The system memory include the permanent memory EPROM and the temporary memory RWM (RAM). The ultraviolet erasable and electrically reprogrammable read only memory EPROM 2716 a 2k by 8-bits EPROM is widely used for storing program and constants in microprocessor system. EPROM are nonvolatile; therefore they can store the application program in a dedicated microprocessor system and the program will

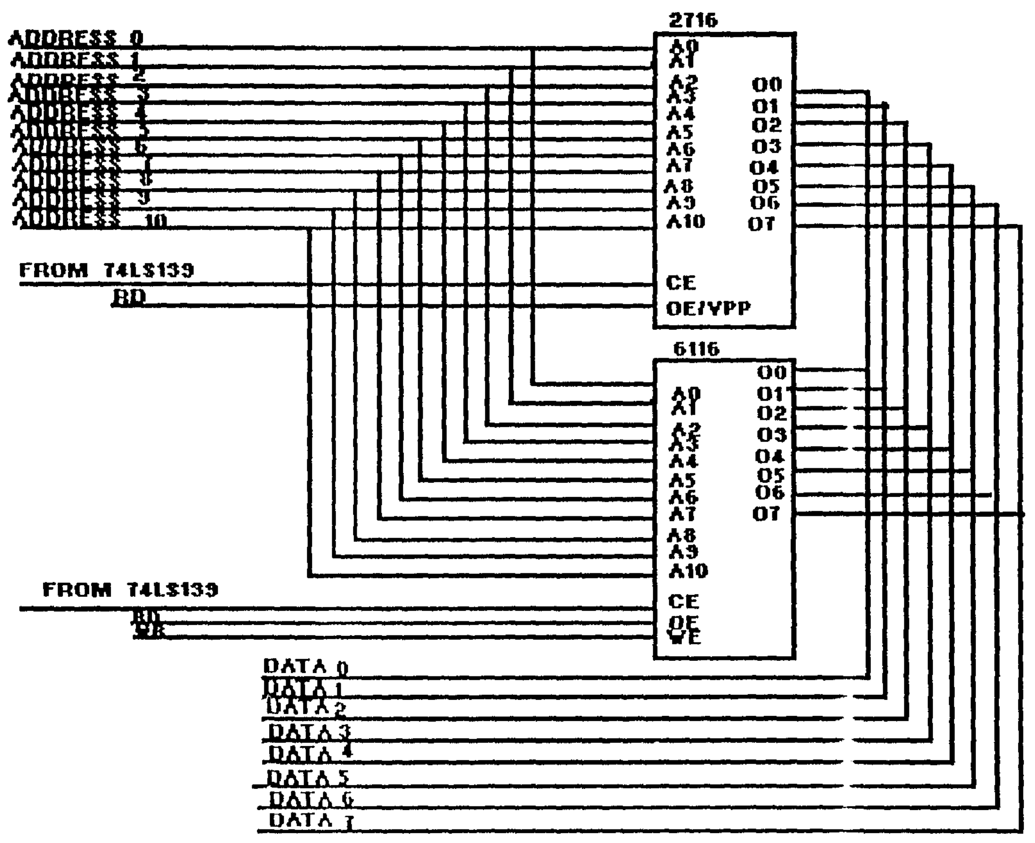


FIG 2.6 SYSTEM MEMORY

always be ready for execution. EPROM contains the monitor software or instruction required for running the system. The EPROM address bus is connected to the output of the address buffers, while the output of the EPROM is connected to the output of the data buffer. The EPROM, which is a 2k by 8-bits EPROM is capable of accessing $2^{11} = 2048$ memory location, hence only eleven address lines from the address buffers are inputted into the EPROM. The EPROM is a read only memory it has no WE (write enable) input. Two control lines are used to operate the EPROM. The two control lines are CE and OE. The chip Enable input, CE is similar in function to the CS input of an SRAM. RD is connected to the OE while the chip select decoder is connected to CE.

Both erasure and programming are done with the EPROM removed from the microprocessor system. Erasure is accomplished by shining ultraviolet light through the quart window of the EPROM package and into the IC chip. EPROM programming is done with an instrument designed for that purpose, an EPROM programmer.

The easiest type of RWM to use in a microprocessor system is the static random access memory, SRAM. An SRAM is volatile. It will retain its data only while its operation power is maintained. An SRAM 6116, 2k 8-bits RAM is used just like EPROM, the SRAM address bus is connected to the output of the address buffers while the output of the SRAM is connected to the output of the data buffer. The SRAM 6116 is a 2k by 8-bit RAM and capable of accessing $2^{11} = 2048$ memory location, hence only eleven address lines from the address buffer are connected to the SRAM.

Three control lines CS, WE and OE are used to control inputs to cause write or read operation to take place. CS is connected to one of the output from the chip select decoder (74LS139).

WE is connected to a write strobe WR from the microprocessor while the OE is connected to a read strobe RD from the microprocessor.

EPROM 2716 and RAM 6116 are both of 24 dual in line package.

Fig. 2.6.1 Illustrate the system memory and its connection.

2.7 System input/output port

The term port is applied to the circuitry that provides the opening or gateway for the transfer of data between the microcomputer or microprocessor-based system and the external world.

There are four input/output ports. Two are used for the display while the remaining two are used for the keypad. The display uses port 1 and port 2, which are both 8-bit latches (74LS373). (Port 1 is for digit select, while port 2 is for segment select. The output of the data buffer (of which the output of the memory is also connected) is the input of the two ports 1 and 2. The output of port 1 is connected to segments a,b,c,d,e,f and g. Hence port 1 is used for digit select. The common cathode pin is connected to a pnp transistor, which is also connected to port 2.

The output controls of both chips are connected to ground so that the latches are transparent. The latches are required to be transparent because every time the latches are selected, the information at the input pins are immediately needed to be transferred to the output pins. Since the ports are latches, they will hold on to the last information until a new information is sent.

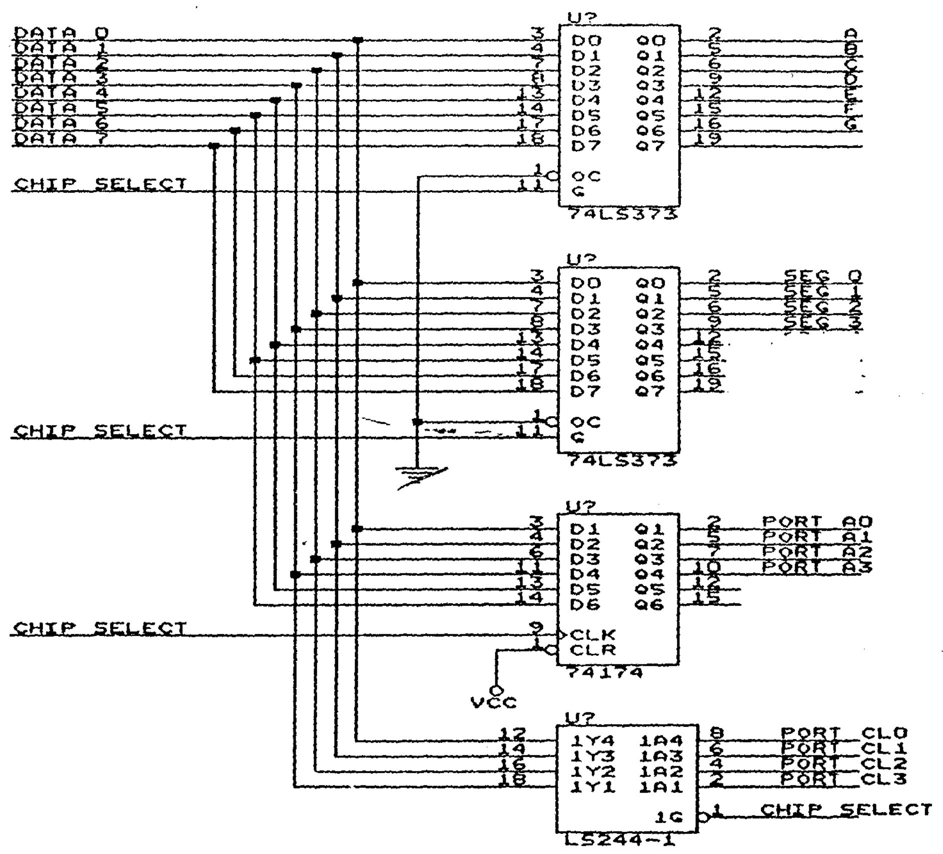


FIG 2·7 SYSTEM INPUT / OUTPUT PORTS

The keypad uses port 3 and port 4 , port 3 is the keypad latch (74LS174) while port 4 is the keypad buffer (74LS244). The input to the latch is the output from the data buffer. The output of the latch is connected to the column switches of the keypad. The input to the buffer is the row switches of the keypad connected through a pull-up resistor. The output of the buffer is connected to the output of the data buffer.

2.8 The chip select logic

Each of the microprocessor peripheral devices: EPROM, RAM, INPUT PORTS AND OUTPUT PORTS are connected to the chip select decoder which is itself connected to the most significant bit or least significant bits as the case may be. The peripheral devices receives an address sent out on the buss simultaneously so it is essential to ensure that only the correct device responds by accepting or sending data. This arbitration is achieved by means of the chip select decoder. The chip select decoder used here is the 74LS139, a dual 2 to 4 binary decoder and the four output are used to control the chip select CS pins on each of the peripheral devices, only one of which will be act activated at a time.

To select any of the four ports, a 74LS139 is used. The least significant bits of the address lines are connected to the input of the decoder. These bits are address lines from the microprocessor. Also connected to the input of the decoder is a control strobe IO/M from the microprocessor. The outputs of the decoder are each connected to the CS pin of the ports.

The 74LS139 is a 2-line to 4-line decoder, hence the second lines is used to select the memory. The most significant bit of the address line is connected to the select pin of the

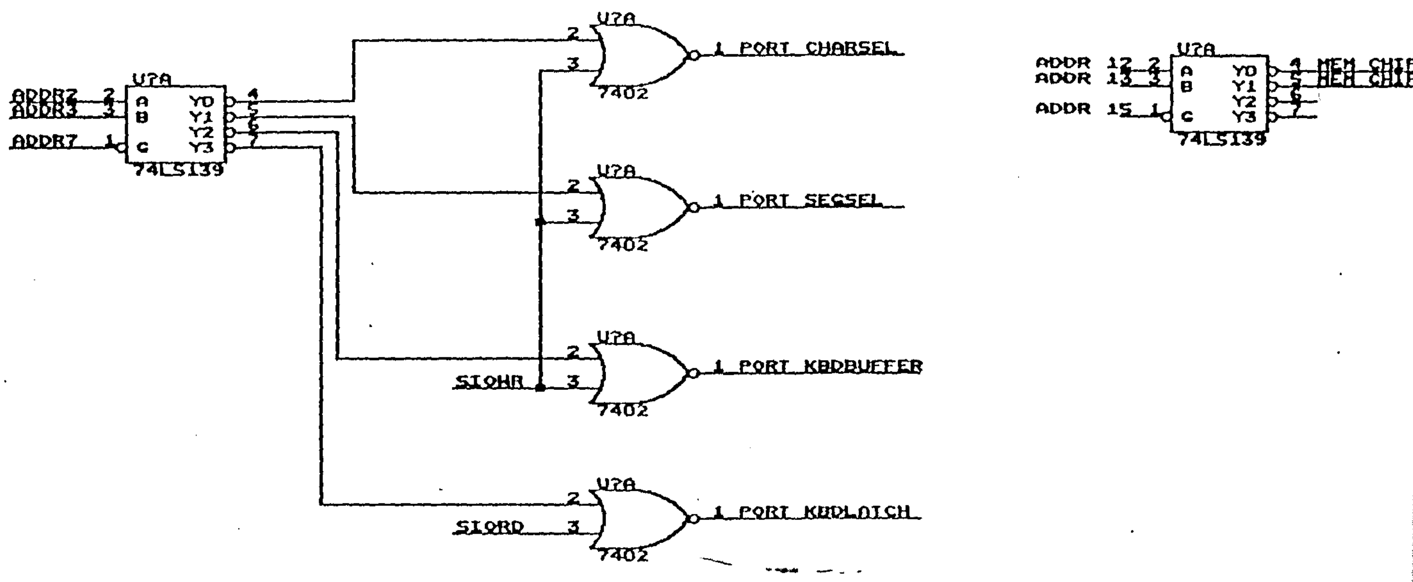


FIG 2·8 CHIP SELECT LOGIC

decoder, also connected to the enable pin of the decoder (2G) is the 10/M pin from the microprocessor. The outputs of the decoder 1Y and 2Y, are used to uniquely select the memory.

2.9 Seven – segment multiplexed display

Seven-segment displays are available in two basic types: the common anode and common cathode displays. The common cathode displays are designed such that the 0V (ground) is applied to the cathode of all the internal LED's and +5v(vcc) is applied to a segment through a current limiting resistor. The current limiting resistors are used to prevent the LED's from burning out due to the passage of current higher than its rated current. Common anode display functions almost the same way except that the anodes are here connected to +5v and 0v is applied to a segment through a current limiting resistor to light it. Each LED requires a current of about 10MA to fully illuminate a segment.

For this project, a 4-digit common cathode display is used. All the segment of each digit are connected to their corresponding segment in the other 7 segment digit of the display, thereby giving a common output a-g. The ground of each of these digits are not connected together. Instead, each of them are connected to the segment select latch port (2).

Ordinarily to light up all the digits of a 4-digit display, a very high quantity of current is required which may be impossible to provide. Hence, multiplexed display is used in the case of multiplexed display, at any particular instant only one LED is powered on and the rest are off. Every few microseconds a different LED is powered so as to reduce the consumption of current.

This multiplexed display is achieved by using two ports, the segment select port and the digit select port. The segment select port the seven-segment code for the digit to be displayed. The digit select port selects the particular digit whose code is sent to the segment select port. For the next digit, its 7-segment code is sent to port 2 and the digit select port (port 1) select the digit to be displayed.

2.10 The Unencoded keypad

A keypad is a very important interface to a microprocessor-based system because information required by the system is usually entered through the keypad. Hence for a microprocessor-based calculator, the construction of a proper keyboard and interfacing is very necessary to ensure that the right data is punched into the microprocessor in order to avoid distorted calculation.

There are different techniques of entering data into the microprocessors but this project the scan matrix keypad technique is used.

Here the keys are arranged in a matrix form that is in columns and rows. In this project sixteen keys arranged in four rows and four columns. The matrix keypad is interfaced using two ports: and input port and output port are connected to the output port and columns are connected to the input port.

The software technique called the matrix scan is used to read the data from the keypad. In a matrix scan, rows are grounded by outputting O'S to all the rows through the output port and a key closure is checked by reading the data on columns through the input port. The columns are rows are connected only when the key is pressed; otherwise they are disconnected and remain high (+5v).

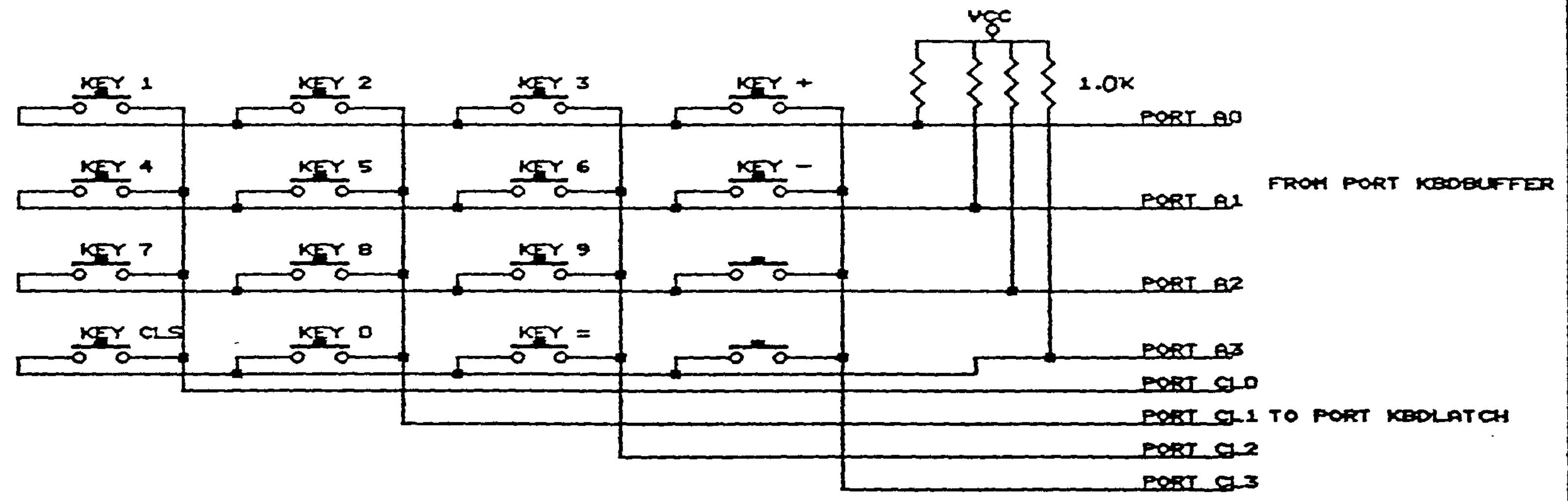


FIG 2·10 SCAN MATRIX KEYBOARD

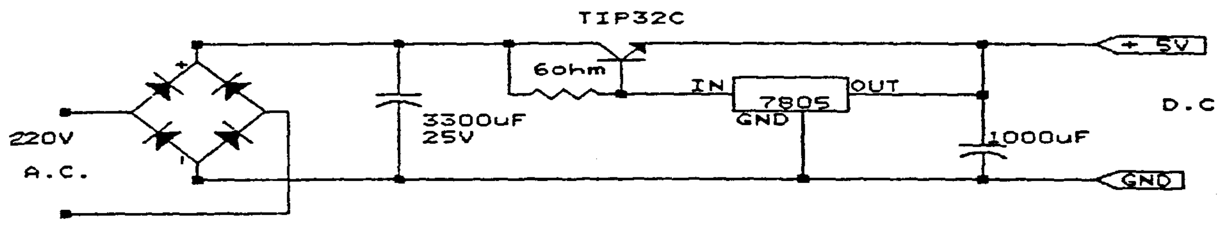
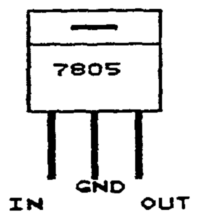


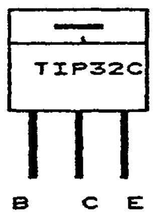
Fig 2.10 FIG 2·11 REGULATED POWER SUPPLY

VOLTAGE REGULATOR



INPUT VOLTAGE = 8V TO 40V
OUTPUT = 5V

POWER TRANSISTOR



COLLECTOR TO BASE VOLTS	= 130V
COLLECTOR TO EMITTER VOLTS	= 120V
BASE TO EMITTER VOLTS	= 5V
MAX. COLLECTOR CURRENT AMPS	=4 AMPS
MAX. DEVICE DISS. WATTS	= 0.5 WATTS
FREQ.IN MHz	= 4 MHz MIN.
CURRENT GAIN	= 75

During the scan, one row at a time is brought low, if a key in that row is pressed, then the corresponding column also go low, otherwise the columns connected to that row remain high and then the next row is driven low and vice vasa.

Four pull-up resistors are connected between the column lines and VCC so as to keep the line at a constant VCC. These pull-up resistors are chosen to be 1k-ohm each because they keep the constant current to a minimum value.

2.11 The power supply unit

The power requirement is very critical since this is a microprocessor-based system. A microprocessor is very sensitive to ripples and variation in power, hence a regulated power supply is used. The unregulated power is passed through a 7805 regulator to provide regulated 5 volt and this is current amplified by the power transistor TIP32C.

2.12 SYSTEM SOTFWARE DESIGN

2.2.0 Software modular concept

For the ease of developing and debugging the software, the software is structured into the main module and support modules, which could also be referred to as subroutine. The support modules include:-

i) mult/DiV.

ii) Binary to decimal converter; B1NTD

iii) BCD to binary conversions; BCDTB.

2.2.1 Machine and assembly language

The microprocessor responds to a listing OF operations that is called machine code. The machine code or machine program is usually stored in a linear memory. Each cell or byte of the linear memory stores an instruction or operation or data. Instruction are stored ready for execution in semiconductor memory devices, such as ROMS or RAMS which contain instruction and data in binary code where each bit is stored as either a binary "1" or binary "0" depending on the voltage level used. Although the microprocessor finds a diet of ones and zeros, most palatable, programmers do not and variety of alternative representation are used which are more simpler to remember and more compact to set down on paper. The two most popular codes for representing the binary information used by microprocessor are octal and hexadecimal which used base 8 and base 16 respectively in place of binary's base 2. Hexadecimal is the simplest to use and it is certainly the most popular scheme used in manufactures instructions set data. Its advantage is that each hexadecimal digit represents four binary digits and since microprocessor usually has word length of some multiple of four bits, hex coding is a convenient tool.

Program in hexadecimal notation is also very difficult to be understood by human being. This makes it tedious and not convenient for all but the simplest programs. And so manufacturers specially introduced a special program language. This program language is meant for translating the program written in words and phrases to the code the microprocessor understands (i.e machine code). Such a program is called ASSEMBLY LANGUAGE. Thus a program written in mnemonic form is called an assembly language.

An assembly language requires an ASSEMBLER (i.e a translating program) and a LOADER, which serve the purpose that large programs be set up quickly in the memory.

This level of automation functionally is called ASSEMBLY CODE, and is simply the automation of machine code programming by using a program called assembler. This still required the programmer to operate at the level of the instruction set but in this case permits him to enter instruction mnemonics directly without the need for encoding them into binary or hexadecimal format. The assembler program itself is used to process the list of mnemonics previously entered by the programmer into a text file created using a simple editor program. The mnemonic text file created by the programmer is called the source file, while the resulting machine code file produced by the assembler is called the object file.

The object file is generally in a format suitable for direct execution by the machine without further processing. The figure below illustrates the above translation.

2.2.2 Writing assembly language programs

Before writing the actual assembly language for the operation, there is the need to representing the program in schematic form called flowchart. Alternatively, the algorithm method could be used in place of flowcharting. The advantage of flowcharting or the algorithm drafting is basically to assist one in writing his program.

Assembly language programs are usually written in a form so that they can be translated to machine language by an assembler program. In this standard form, assembly language statements have four fields.

The first field is the LABEL FIELD. A level is a symbol or group of symbols used to represent an address that is not specifically known. Some assemblers require that labels have no more than five or six characters and that the first character be a letter. The label field is usually ended with a colon.

The second field is the OPCODE FIELD. This field contains the mnemonic for the instruction to be performed. Following the OPCODE field is the OPERAND FIELD. This field contains the registers, data, or address on which the instruction is to be performed. A comma is required between register initials or between register initial and a data byte. Between the opcode field and the operand field is an open (empty) space.

The last of these fields is the COMMENT FIELD, which is also important as it is used for reference in future time. It is started with a semicolon or an asterisk.

CHAPTER 3

SYSTEM DEVELOPMENT

3.1 Introduction

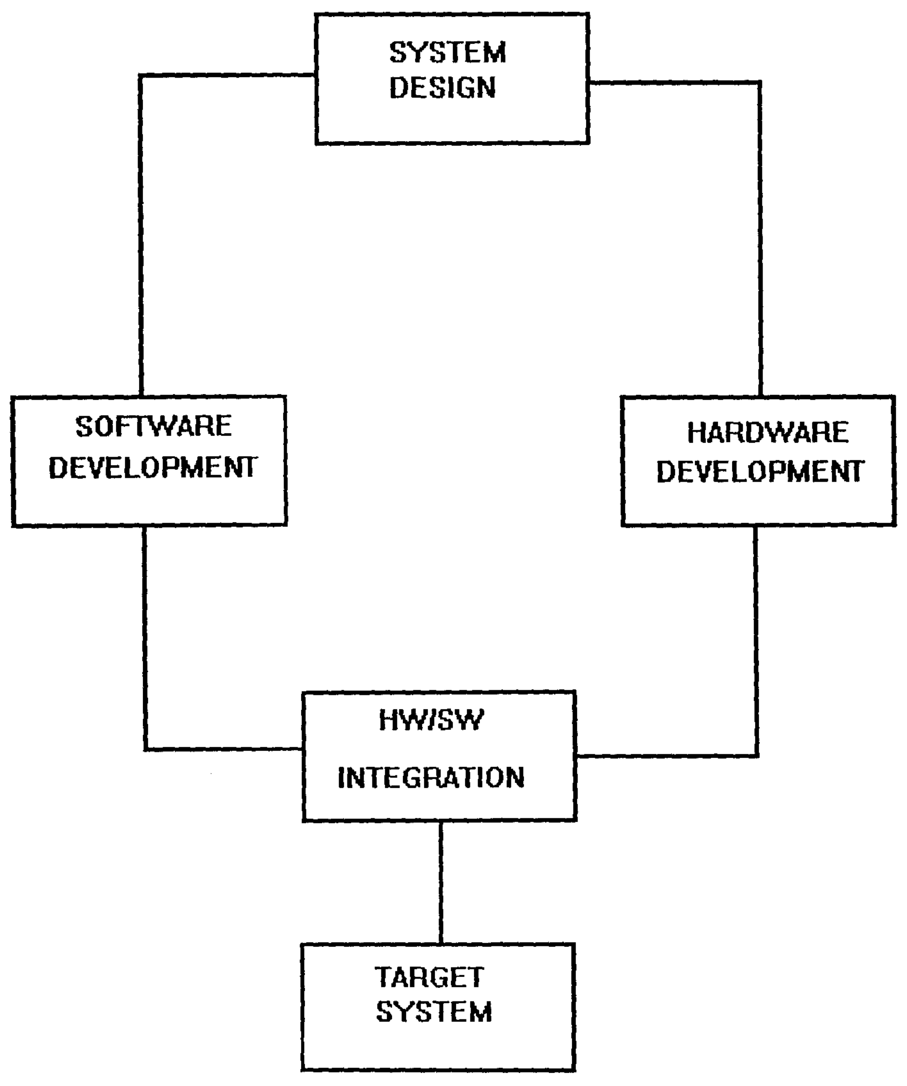
Once the architecture of the system under development has been determined, the detailed design of each portion of the system may begin. Software and hardware development may proceed somewhat independently during the early state of design. As the hardware systems are prototyped, test software may be installed to allow the processor system to be exercised and the functionality of I/O system may be checked likewise. The software designer may be able to check the logic functionality of selected algorithm before prototype hardware is available for test. As the hardware and software systems complete their preliminary testing phases. The time arrive for integrating the two subsystems. This phase can be one of the most difficult since both systems are as yet unproven.

3.2 Hardware development

3.2.1 Component layout

The component layout is an important consideration, which must be decided upon before any soldering work starts. The following consideration were made:-

i) Neatness:- in order to have a neat work, chips were well placed (positioned), so that the connecting cables does not crisscross one another and the connecting cables were made to follow a common path.



BLOCK DIAGRAM OF A SYSTEM DEVELOPMENT FIG 3.1

ii) capacity loading:- to reduce capacity loading of the system, chips are positioned to reduce the

length of the wires connecting the various chips..

iii) System enclosure:- the nature of the system enclosure determines the positions of some components like the connectors for the display and keyboard.

3.2.2 Construction tools and materials

i) Soldering Iron, 60watt

ii) Desoldering pump

iii) Solder

iv) Razor blade

v) Veroboard

vi) Connecting cables (secondary winding of a transformer)

vii) IC socket outlet (40 pins, 24 pins, 20 pins, 16 pins and 14 pins).

3.3 Construction steps

The construction was carried out in modules:-

a - the system processing module

b - the display module

c - the keypad module

d - power supply module

3.3.1 The system processing module:- Construction steps

i) Layouts:- the components are physically moved around on the board until a neat, orderly arrangement of chips was obtained.

ii) Sockets of equivalent pin outlet to the IC were soldered and documentation made on the physical position of every chip. This provided a record and guide for soldering pins and leads during the next step.

iii) The connection of the cables were made starting with the microprocessor unit followed by the latches and buffers, and followed by the memory and I/O ports comes next.

iv) After the connection of the I/O ports, the chip select decoder was also cabled to the memory and I/O ports via a NOR-gate.

3.3.2 The display unit - Construction steps

i) LED crystal were placed on the veroboard and soldered

ii) The display unit is the multiplexed type, hence all the segment that correspond to each other were connected to one another (i.e all the a'^{s} are connected together and vice-versa).

iii) Segments (a – g) were connected to the port charsel

iv) The common cathode pin was connected to the port segsel. This connection is made via a switching transistor.

3.3.3 Keypad module - Construction steps

i) The switches were placed on the veroboard to form a 4 x 4 matrix and then soldered.

ii) The horizontal switches are connected together while the vertical switches are connected together. Note that only one pin of the four pins was used.

iii) The horizon columns are connected to port (keyboard latch) while the vertical columns are connected to port (keyboard buffer) through a pull-up resistor.

3.4 Hardware testing

The test carried out involves the use of a locally produce logic probe (by connecting one arm of an LED to resistor of 150R and then finally to a long cable) to check the pjn out of a microprocessor if it is high or low. This is also called free running of a microprocessor. A digital multimeter was used to check if there was any open circuit or any bridging. The digital multimeter was also used to check the continuity of the soldered sockets before the chips were placed on it.

3.5 Software development

Software development is a critical part of the system development and takes a major development time of the system.

Microprocessor development system (MDS) was used. MDS is the lost system consisting of the hardware components and the software tools necessary to develop microprocessor based target system. It provide facility to design, develop and test software, hardware and the two interacting together.

MDS consist of text editor, assembler and a simulator/debugger all integrated into Integrated Development Environment IDE.

A simulated I/O device which can be configured to model the actual I/O device employed on the target system is also provided to allow the designer to have a real feel of his system behaviour even before its implementation.

The assembler used was 8085 cross assembler. It is very user-friendly software allowing programmers debug programs easily.

3.6 System software used

i) Pentium computer (Host system)

ii) EPROM programmer (courtesy Ahmed Sadiq – final year project 1999)

iii) 8085 cross assembler

3.7 Software testing

The simulator enables the host system to mimic the 8085 microprocessor, thereby allowing 8085 assembly language to run even in the absence of an actual 8085 microprocessor.

CHAPTER FOUR

ISCUSSION OF RESULTS, RECOMMENDATIONS AND CONCLUSION

ISCUSSION OF RESULT

The target system (completed project) met its basic requirement. The target system was initially designed r an 8-digit manipulation but along the line it was changed due to the inavailability of an EPROM eraser, ould in case of an error in the software design. Recall that the system software was tested using a simulated O device which was configured. This simulated I/O device in the host system is of 4-digit manipulation. Hence e target system was reduced to 4-digits.

The system software can be varied and interchanged to upgrade the system. This makes the project very satile. System design and construction was made using minimum number of components.

The power supply unit supplies a regulated (4.5 to 5.1) Vd.c. This fluctuation is as a result of the supply ltage from NEPA which ranges between 192 V and 238 V

ECOMMENDATION

The recommendation here is for further development work which can be carried out on this project. eing software dependent the system could be upgraded to perform logarithmic, trigonometric, Decay functions nd other arithmetic function, although an arithmetic processor called INTEL 8087- a numeric processor xtension (NPX) must be introduced to support the microprocessor.

I emplore the department to make available an EPROM programmer and Eraser to assist students in ~~iere~~ their project as microprocessor based project are varsatile and cost effective.

ONCLUSION

The cost of production of this project work is quite low if it will be upgraded and some support chips ntroduced so as to function as a programmable scientific calculator or a microcomputer. And as such, worth oing into production by the manufacturing industries.

REFERENCES

1 Microprocessor Data book by S.A. Money. Second edition 1990. (Blackwell scientific)

2 Software Engineers reference book by John A. McDermaid. First edition 1990. (Butterwortt Heinemann).

3 The Art of Electronics by Horowitz and W. Hill. Second edition 1989.(Cambridge University press)

4 Electronics Devices and circuit theory by Robert Boylestad, Louis Nahelsky. Fourth edition 1987.

5 Microprocesssor and Digital Systems by Douglas Hall. Second edition 1989.

6 Microprocessor and programmed logic by Keneth L. Short. Third edition 1988.

7 Microcomputer handout by J.A McGrindle. First edition 1985.

```
buffer:byte 0 ;KEYPAD INTERRUPT PROCEDURE
newkey:byte 0
buffer:byte 0
keycount:byte 0
org &h24
keypadintr:in 12
            sta buffer
            mvi a,1
            sta newkey
            ret
ReadKey:lda newkey;READ KEY-STROKE
            cpi 1
            jnz ReadKey
            xra a
             sta newkey
            lda buffer
            ret
numeric:push psw
                 lda keycount
                 cpi 4;if keycount=4 don't accept key
                  jz getout
                  cpi 0;if keycont=0 clear digits
                  cz cleardigits
                  cnz shiftdigits
                  lxi h,keycount
                  inr m
                  lda buffer
                  sta digit1
getout:        pop psw
                  ret
ClearDigits:push psw
                  mvi a,0
                  sta digit1
                  sta digit2
                  sta digit3
                  sta digit4
                  sta keycount
                  pop psw
                  ret
ShiftDigits:lda digit3
                 sta digit4
                 lda digit2
                 sta digit3
                 lda digit1,
                 sta digit2
                 ret
OutPut:push psw
            lda digit1
            out 4
            lda digit2
            out 3
            lda digit3
            out 2
```

```
op1:byte 0
op1+1: byte 0
op2:byte 0
op2+1:byte 0
Addition:lda op1
          lxi h,op2
          add m
          sta op1
          lda op1+1
          lxi h,op2+1
          adc m
          sta op1+1
          ret
Subtract:lda op1
          lxi h,op2
          sub m
          sta op1
          lda op1+1
          lxi h,op2+1
          sbb m
          sta op1+1
          ret
Digit4:byte 0
Digit3:byte 0
Digit2:byte 0
Digit1:byte 0
Operation:byte 0
Start:lxi sp,stack2
loop: call readkey
          cpi 10;if key is numeric
          cm numeric
          cpi 14
          jz equal
          cpi 15
          jz clear
          call output
          cpi 10
          jm loop
          sta operation
           lxi d,digit4;convert to binary
          call bcdtb
          shld op1
          xra a
          sta keycount
          call output
          jmp loop
Clear: xra a
             sta keycount
             call ClearDigits
             call output
             jmp loop
Equal:push psw
              lxi d,digit4;convert to binary
             call bcdtb
             shld op2
             lda operation
             cpi 10
          cz addition
          cpi 11
          cz subtract
          cpi 12
          cz multiply
          cpi 13
          cz divide
          pop psw
           lhld op1
           lxi d,digit4
           call bintd
          call output
          xra a
          sta keycount
          jmp loop
ck1:word 0
         word 0
         word 0
ck2:word 0
         word 0
         word 0
         word 0
         word 0
         word 0
         word 0
         word 0
         word 0
         word 0
ck:    word 0
         word 0
         word 0
         word 0
         word 0
         word 0
ck:  word 0
 start
```

```
;
Multiply:lxi h,op1
               mov e,m
               inx h
               mov d,m
               lda op2
               lxi h,0
               mvi b,8
mult:        dad h
               ral
               jnc chcnt
               dad d
chcnt:    dcr b
               jnz mult
               mov a,l
               sta op1
               mov a,h
               sta op1+1
             ret
divide:lhld op1
             lda op2
             mov c,a
             mvi b,8
div:      dad h
             mov a,h
             sub c
             jc cnt
             mov h,a
             inr l
cnt:    dcr b
           jnz div
           mov a,l
           sta op1
           mvi a,0
           sta op1+1
           ret
end
```

```
;BCD to binary conversion
BCDTB:lxi h,0 ; clear HL
             mvi c,4;set digit counter
loop1:     ldax d; load pointer to decimal number beffer
             add l; add decimal digit to HL
             mov l,a
             mvi a,0
             adc h
             mov h,a
             dcr c;check for last digit
             rz
             call ten;multiply HL by 10
             inx d
             jmp loop1
TEN:       push b;save counter
              dad h;HL*2
             push h ;save
             dad h;HL*4
             dad h;HL*8
             pop b;load bc with H*2
             dad b;HL*10
             pop b;restore counter
             ret
end
```

```
;Binary to Decimal converter
BINTD:mvi b,3;init digit cout B,N-1
            xchg
            shld DEC ;save pointer to digit storage area
            xchg
            lxi d,10;place powers of 10 constants on stack
            push d
            lxi d,100
            push d
            lxi d,1000
            push d
loop2:    pop d;get power of ten
            call digit ;sub returns digit in c
            push h;save binary difference
            lhld dec; get pointer to digit storage area
            mov m,c;store digit
            inx h;increase pointer
            shld dec;stor pointer
            pop h;get binary difference
            dcr b
            jnz loop2;more than 1 digit must still be determined
            mov c,l
            lhld dec ;get pointer to digit storage area
            mov m,c;store digit
            ret
DIGIT:mvi c,&hff;init c to -1
AGAIN:inr c
            mov a,l;subtract lowest power of 10 from binary number
            sub e
            mov l,a
           mov a,h
            sbb d
            mov h,a
            jnc again;if diff' is negative go back
            dad d ;if postive restore
            ret
dec: byte 0
dec+1:  byte 0
end
```