



INTELLIGENT PEN

By

G.SAKTHIVEL

Reg No.:0710107083

P.SANTHANAKHAJAN

Reg No.:0710107086

S.VIJAYKRISHNAN

Reg No.: 0710107113

K.VISAGAN

Reg No.: 0710107115

of

KUMARAGURU COLLEGE OF TECHNOLOGY

(An Autonomous Institution affiliated to Anna University of Technology, Coimbatore)

COIMBATORE - 641049

A PROJECT REPORT

Submitted to the

FACULTY OF ELECTRONICS AND COMMUNICATION ENGINEERING

In partial fulfillment of the requirements

for the award of the degree

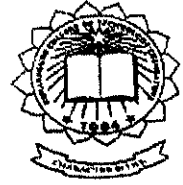
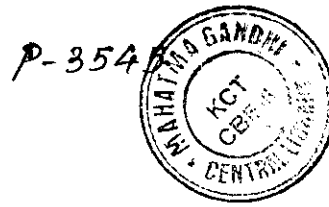
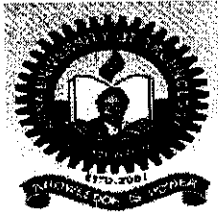
of

BACHELOR OF ENGINEERING

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

APRIL 2011



INTELLIGENT PEN

By

G.SAKTHIVEL

Reg No.:0710107083

P.SANTHANAKHAJAN

Reg No.:0710107086

S.VIJAYKRISHNAN

Reg No.: 0710107113

K.VISAGAN

Reg No.: 0710107115

of

KUMARAGURU COLLEGE OF TECHNOLOGY

(An Autonomous Institution affiliated to Anna University of Technology, Coimbatore)

COIMBATORE - 641049

A PROJECT REPORT

Submitted to the

FACULTY OF ELECTRONICS AND COMMUNICATION ENGINEERING

In partial fulfillment of the requirements

for the award of the degree

of

BACHELOR OF ENGINEERING

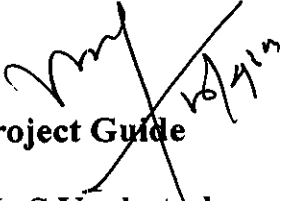
IN

ELECTRONICS AND COMMUNICATION ENGINEERING

APRIL 2011

BONAFIDE CERTIFICATE

Certified that this project report entitled "INTELLIGENT PEN" is the bonafide work of **Mr.G.SAKTHIVEL, Mr.P.SANTHANAKHAJAN, Mr.S.VIJAYKRISHNAN, and Mr.K.VISAGAN** who carried out the research under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.


Project Guide

Mr.S.Venkatesh


Head of the Department

Dr. Rajeswari Mariappan

The candidates with university Register no. 0710107083, 0710107086, 0710107113, 0710107115 are examined by us in the project viva-voce examination held on19.04.11.....


Internal Examiner


External Examiner

ACKNOWLEDGEMENT

A project of this nature needs co-operation and support from many for successful completion. In this regards, we would like to express my thanks and appreciation to the many people who have contributed to the successful completion of this project.

We express my profound gratitude to our beloved Director **Dr.J.Shanmugam**, Kumaraguru College of Technology for his kind support and necessary facilities to carry out the work.

We would like to thank **Dr.S.Ramachandran., Principal** for providing us an opportunity to carry out this project work.

We express my gratitude to **Dr.Rajeswari Mariappan**, Head of the Department, Electronics and Communication Engineering, who gave her continual support for me throughout the course of this project.

Our heartfelt thanks to **Ms.K.Kavitha M.E.**, Assistant Professor and Project Coordinator, for her contribution and innovative ideas at various stages of the project and for her help to successful completion of this project work.

I express my gratitude to **Mr. S.Venkatesh M.E.**, Assistant Professor and my internal project guide, for his valuable guidance, innovative ideas and constant encouragement throughout this project.

I express my sincere gratitude to my family members, friends and to all my staff members of Electronics and Communication Engineering department for their support throughout the course of my project.

Last but not the least I would like to thank the God almighty without whose grace nothing would have been possible so far.

ABSTRACT

In any mobile phone, to write, people have two clear choices at the moment – they either use a keypad or a touch screen stylus. Many people get discouraged with current phones and their small keys. As phones get smaller, this frustration will only grow.

How about creating a third option that allows people to write in air? The concept is simple. The entire phone should be moved through the air to write. So making a “3” shape in the air sees that letter appear on the screen. The idea has been made possible by the growing use of accelerometers inside devices that can track the movement being carried out with the device – something used so effectively in the Nintendo Wii and the iPhone. In fact the project uses the same accelerometer LIS302DL that was used in highly successful “Apple iPhone”.

The project is about designing a creative pen-style hardware and analysis software for the recognition of simple air written characters. The hardware has a 3-dimensional acceleration sensor, a microcontroller with I²C serial communication port, and does not need any touching screen or keypad. Motion through air is converted as a set of points in space and to recognize a character a unique software algorithm verifies the pattern of motion points against a stored form and a display shows the character recognized in the process

TABLE OF CONTENTS

S.NO	CAPTION	PAGE.NO
	ABSTRACT	iii
	LIST OF FIGURES	vi
	CHAPTER I	
1.1	INTRODUCTION	1
	CHAPTER II	
2.1	OBJECTIVE	2
2.2	BLOCK DIAGARM	3
2.3	EMBEDDED SYSTEM	4
2.4	TYPES OF EMBEDDED SYSTEMS	4
2.5	APPLICATIONS OF EMBEDDED SYSTEMS	4
2.6	CHARACTERSITICS	5
2.7	THE FUTURE OF EMBEDDED SYSTEMS	6
2.8	INTRODUCTION TO MICROCONTROLLER	7
2.9	MICROCONTROLLER WITH ITS BASIC ELEMENTS AND INTERNAL CONNECTIONS	11
2.10	CISC AND RISC	12
2.11	PIC MICROCONTROLLER'S CORE FEATURES	14
2.12	PERIPHERAL FEATURES	15
2.13	MEMORY ORGANIZATION	15
2.14	I/O PORTS	16
2.15	TIMER0 MODULE	18
2.16	MASTER SYNCHRONOUS SERIAL PORT (MSSP) MODULE	19
2.17	PIN DIAGRAM	22
	CHAPTER III	
3.1	INTRODUCTION TO I ² C	23
3.2	THE I ² C BUS PROTOCOL	25
3.3	DESIGNER BENEFITS	25
3.4	I ² C BUS CONCEPT	26
3.5	I ² C SERIAL EEPROM	29
3.6	BUS CHARACTERISTICS	31
3.7	WRITE OPERATION	33
3.8	READ OPERATION	35
3.9	NOISE PROTECTION	36
3.10	PIN DESCRIPTIONS	36

CHAPTER IV		
4.1	MEMS ACCELEROMETER	38
4.2	MEMS ACCELEROMETER LIS302DL	39
4.3	FEATURES	40
4.4	PIN DESCRIPTION	42
4.5	TERMINOLOGY	42
4.6	IC INTERFACE	44
4.7	POWER SUPPLY	45
CHAPTER V		
5.1	DISPLAY UNIT	47
5.2	LCD OPERATION	47
5.3	LCD PIN DESCRIPTIONS	47
5.4	STEPS TO INTERFACE LCD WITH PIC MICROCONTROLLER	50
CHAPTER VI		
6.1	CIRCUITS	52
6.2	DEVELOPMENT TOOLS FOR PIC MICROCONTROLLERS	53
6.3	PHASES OF COMPILER	55
CHAPTER VII		
7.1	CONCLUSION	64
7.2	ADVANTAGES	65
7.3	ENHANCEMENTS	65
CHAPTER VIII		
8.1	REFERENCES	66

LIST OF FIGURES

FIG NO.	CAPTION	PAGE NO
1	BLOCK DIAGARM	3
2	EMBEDDED SYSTEM DESIGN AND DEVELOPMENT LIFE CYCLE	5
3	BLOCK DIAGRAM OF MEMORY UNIT	8
4	REPRESENTATION OF BUS	9
5	WATCH DOG TIMER	10
6	ANALOG TO DIGITAL CONVERTER	11
7	BLOCK REPRESENTATION OF MICROCONTROLLER	12
8	CISC AND RISC	13
9	PIN DIAGRAM OF PIC	22
10	EXAMPLE OF I ² C BUS CONFIGURATION	28
11	BLOCK DIAGRAM OF I ² C SERIAL EEPROM	30
12	DATA TRANSFER SEQUENCE IN SERIAL BUS	32
13	BYTE WRITE	34
14	PAGE WRITE	34
15	BLOCK DIAGRAM OF MEMS	41
16	TOP VIEW AND BOTTOM VIEW	41
17	THE BASIC POWER SUPPLY SCHEMATIC	46
18	FIT HEAT SINK	46
19	LCD INTERFACE	51
20	CIRCUIT DIAGRAM	52

CHAPTER I

1.1 INTRODUCTION:

Context awareness is an emerging application area with the aim of easing human computer interaction (HCI). In the case of a mobile device the HCI can be tedious given the physical size limitation both in the keyboard and screen. If the mobile terminal can aware of the user's current context then it could react in some appropriate manner to suit the user without the need of user interaction. Since gestures are commonly used in daily life, gesture based interaction can be one of novel interaction ways that users want.

To implement the gesture-based interaction, many different techniques, such as vision-based gesture interaction, touch-based gesture interaction have been utilized .In recent years, a new kind of interaction technology that recognizes users' movement has emerged due to the rapid development of sensor technology. An accelerometer measures the amount of acceleration of a device in motion. Analysis of acceleration signals enables three kinds of gesture interaction methods: tilt detection, shake detection and gesture recognition. Although in the literature there are already exist some approaches of using acceleration signals for gestures recognition, most work focuses on recognizing the simple gestures such as Arabic numerals, simple linear movements and direction.

CHAPTER II

2.1 OBJECTIVE:

In this project, the hardware and software system will be designed to recognize 10 numerals at a very high recognition rates, between 90 - 100%. Although the database is quite small, it clearly demonstrates the usefulness of the acceleration-based air-written character recognition system without any touch screen or keypad.

The project creates a useful application with this. Once you typed a number in air, just make a simple tap. The system recognizes this and alerts you after the specified number of minutes through a vibrating motor like the one used in mobile phones.

This system will be called as "Force pen". With current accelerometers the system can only recognize a single character at a time with a pause before starting the next one. As accelerometer technology advances joined letter writing should become possible.

2.2 BLOCK DIAGRAM:

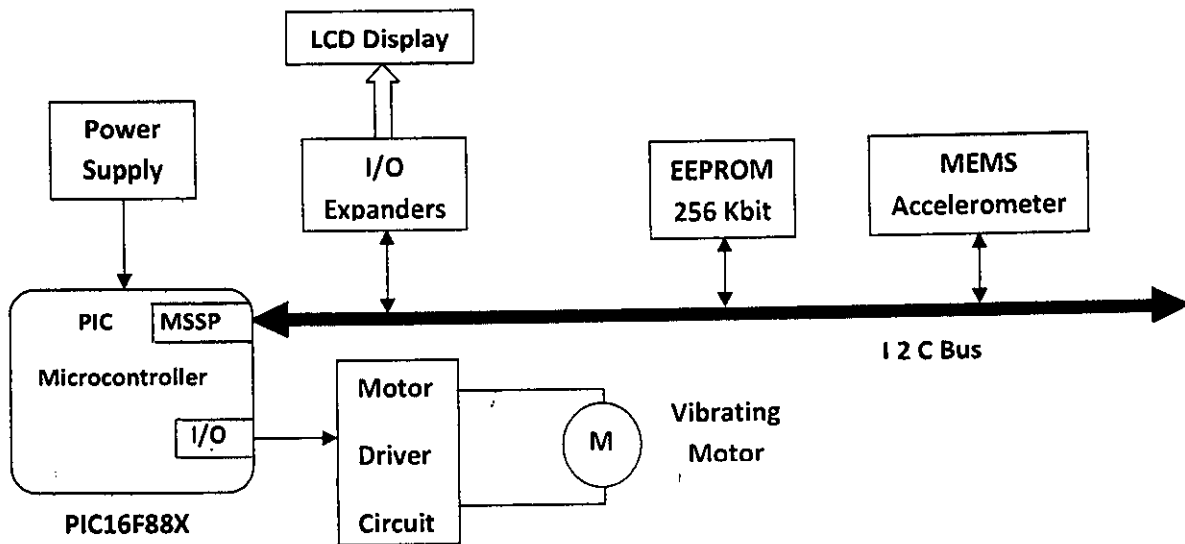


FIG 1: Block diagram

2.3 EMBEDDED SYSTEM:

A general definition of embedded systems is: embedded systems are computing systems with tightly coupled hardware and software integration, which are designed to perform a dedicated function. In some cases, embedded systems can function as standalone systems.

Real-time systems are defined as those systems in which the overall correctness of the system depends on both the functional correctness and the timing correctness. The timing correctness is at least as important as the functional correctness.

2.4 TYPES OF EMBEDDED SYSTEM:

Embedded System is broadly categorized as

- Stand alone embedded system

Example: Washing Machine,

- Networking embedded system

Example: Network Printer

2.5 APPLICATION OF EMBEDDED SYSTEM:

In real life we are using so many embedded systems for example. Home application (micro oven, washing machine, security system DVD, Mp3 player etc,) Air craft, missiles, automotive, nuclear research, personal use (mobile phone, I pod)

2.6 CHARACTERISTICS:

- Embedded systems are designed to do some specific task, rather than be a general-purpose computer for multiple tasks. Some also have real-time performance constraints that must be met, for reason such as safety and usability
- Embedded systems are not always separate devices. Most often they are physically built-in to the devices they control.
- The software written for embedded systems is often called firmware, and is stored in read-only memory or Flash memory chips rather than a disk drive. It often runs with limited computer hardware resources: small or no keyboard, screen, and little memory.

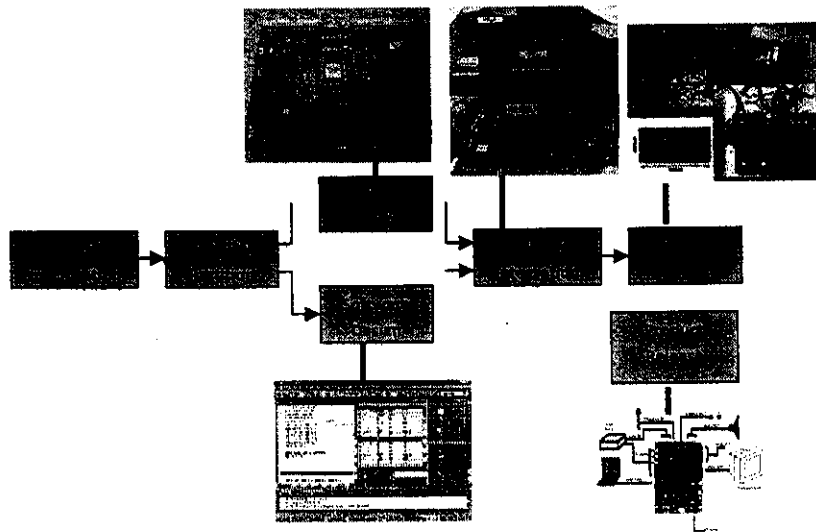


FIG 2: Embedded System Design and Development Life Cycle

2.7 THE FUTURE OF EMBEDDED SYSTEMS:

Until the early 1990s, embedded systems were generally simple, autonomous devices with long product lifecycles. In recent years, however, the embedded industry has experienced dramatic transformation, as reported by the Gartner Group, an independent research and advisory firm, as well as by other sources:

- Product market windows now dictate feverish six- to nine-month turnaround cycles.
- Globalization is redefining market opportunities and expanding application space.
- Connectivity is now a requirement rather than a bonus in both wired and emerging wireless technologies.
- Electronics-based products are more complex.
- Interconnecting embedded systems are yielding new applications that are dependent on networking infrastructures.
- The processing power of microprocessors is increasing at a rate predicted by Moore's Law, which states that the number of transistors per integrated circuit doubles every 18 months.

If past trends give any indication of the future, then as technology evolves, embedded software will continue to proliferate into new applications and lead to smarter classes of products. With an ever-expanding marketplace fortified by growing consumer demand for devices that can virtually run themselves as well as the seemingly limitless opportunities created by the Internet, embedded systems will continue to reshape the world for years to come.

2.8 INTRODUCTION TO MICROCONTROLLERS:

Circumstances that we find ourselves in today in the field of microcontrollers had their beginnings in the development of technology of integrated circuits. This development has made it possible to store hundreds of thousands of transistors into one chip. That was a prerequisite for production of microprocessors, and the first computers were made by adding external peripherals such as memory, input-output lines, timers and other. Further increasing of the volume of the package resulted in creation of integrated circuits. These integrated circuits contained both processor and peripherals. That is how the first chip containing a microcomputer, or what would later be known as a microcontroller came about.

2.8.1 MEMORY UNIT:

Memory is part of the microcontroller whose function is to store data. For a certain input we get the contents of a certain addressed memory location and that's all. Two new concepts are brought to us: addressing and memory location. Memory consists of all memory locations, and addressing is nothing but selecting one of them.

This means that we need to select the desired memory location on one hand, and on the other hand we need to wait for the contents of that location. Besides reading from a memory location, memory must also provide for writing onto it. This is done by supplying an additional line called control line. We will designate this line as R/W (read/write). Control line is used in the following way: if $r/w=1$, reading is done, and if opposite is true then writing is done on the memory location.

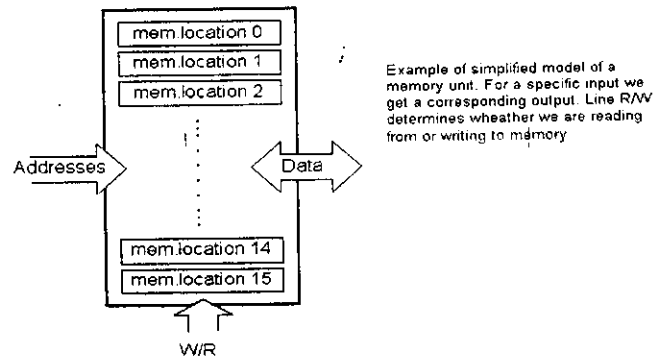


FIG 3: Block diagram of memory unit

2.8.2 CENTRAL PROCESSING UNIT:

Let add 3 more memory locations to a specific block that will have a built in capability to multiply, divide, subtract, and move its contents from one memory location onto another.

Registers are therefore memory locations whose role is to help with performing various mathematical operations or any other operations with data wherever data can be found. Look at the current situation. We have two independent entities (memory and CPU) which are interconnected, and thus any exchange of data is hindered, as well as its functionality.

2.8.3 BUS:

That "way" is called "bus". Physically, it represents a group of 8, 16, or more wires. There are two types of buses: address and data bus. The first one consists of as many lines as the amount of memory we wish to address and the other one is as wide as data, in our case 8 bits or the connection line. First one serves to transmit address from CPU memory, and the second to connect all blocks inside the microcontroller.

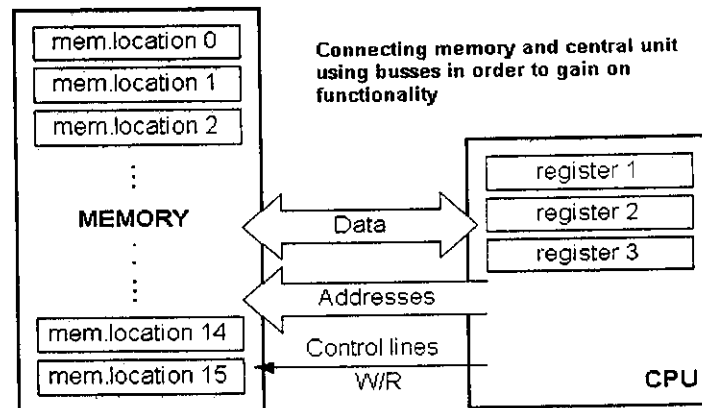


FIG 4: Representation of Bus

2.8.4 INPUT-OUTPUT UNIT:

Those locations we've just added are called "ports". There are several types of ports: input, output or bidirectional ports. When working with ports, first of all it is necessary to choose which port we need to work with, and then to send data to, or take it from the port. When working with it the port acts like a memory location. Something is simply being written into or read from it, and it could be noticed on the pins of the microcontroller.

2.8.5 SERIAL COMMUNICATION:

As we have separate lines for receiving and sending, it is possible to receive and send data (info.) at the same time. So called full-duplex mode block which enables this way of communication is called a serial communication block. Unlike the parallel transmission, data moves here bit by bit, or in a series of bits what defines the term serial communication comes from.

After the reception of data we need to read it from the receiving location and store it in memory as opposed to sending where the process is reversed. In order for this to work, we need to set the rules of exchange of data. These rules

are called protocol. Data goes from memory through the bus to the sending location, and then to the receiving unit according to the protocol.

2.8.6 TIMER UNIT:

The timer block this can give us information about time, duration, protocol etc. The basic unit of the timer is a free-run counter which is in fact a register whose numeric value increments by one in even intervals, so that by taking its value during periods T1 and T2 and on the basis of their difference we can determine how much time has elapsed. This is a very important part of the microcontroller whose understanding requires most of our time.

2.8.7 WATCHDOG:

One more thing is requiring our attention is a flawless functioning of the microcontroller during its run-time.

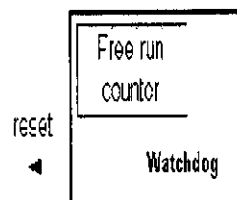


FIG 5: Watch dog time

Suppose that as a result of some interference (which often does occur in industry) our microcontroller stops executing the program, or worse, it starts working incorrectly. However, there is no reset button we can push on the microcontroller and thus solve our problem. To overcome this obstacle, we need to introduce one more block called watchdog.

2.8.8 ANALOG TO DIGITAL CONVERTER:

As the peripheral signals usually are substantially different from the ones that microcontroller can understand (zero and one), they have to be converted into a pattern which can be comprehended by a microcontroller.

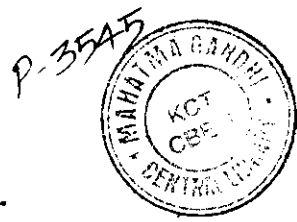
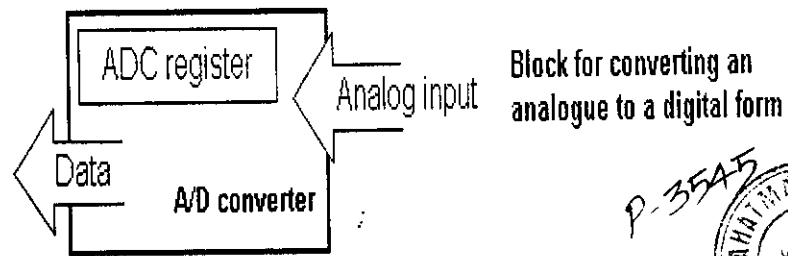


FIG 6: Analog to Digital converter

2.9 MICROCONTROLLER WITH ITS BASIC ELEMENTS AND INTERNAL CONNECTIONS:

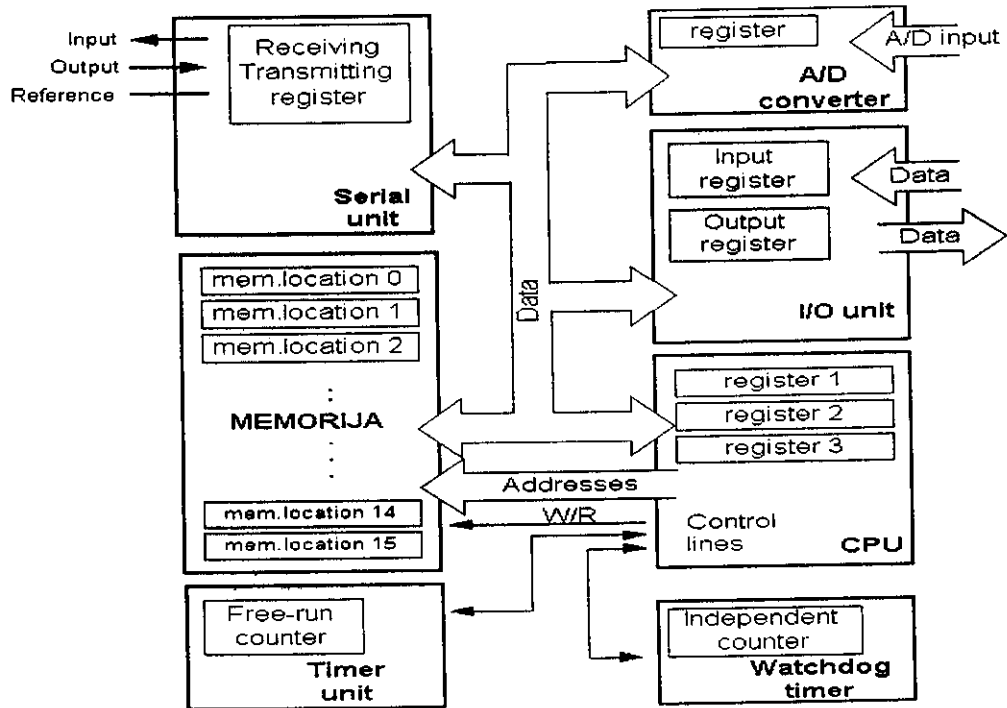


FIG 7: Block representation of the microcontroller

2.10 CISC AND RISC:

Harvard architecture is a newer concept than Von-Neumann's. It rose out of the need to speed up the work of a microcontroller. In Harvard architecture, data bus and address bus are separate. Thus a greater flow of data is possible through the central processing unit, and of course, a greater speed of work. Separating a program from data memory makes it further possible for instructions not to have to be 8-bit words. It is also typical for Harvard architecture to have fewer instructions than von-Neumann's, and to have instructions usually executed in one cycle.

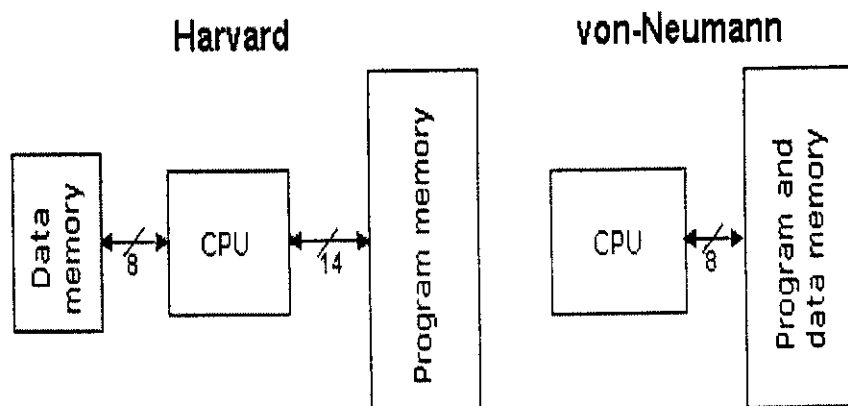


FIG 8: CISC and RISC

Microcontrollers with Harvard architecture are also called "RISC microcontrollers". RISC stands for Reduced Instruction Set Computer. Microcontrollers with von-Neumann's architecture are called 'CISC microcontrollers'. Title CISC stands for Complex Instruction Set Computer. Since PIC16F877 is a RISC microcontroller, that means that it has a reduced set of instructions, more precisely 35 instructions. All of these instructions are

executed in one cycle except for jump and branch instructions. PIC16F87 usually reaches results of 2:1 in code compression and 4:1 in speed in relation to other 8-bit microcontrollers in its class.

2.11 PIC MICROCONTROLLER (PIC16F87X) CORE FEATURES:

- High-performance RISC CPU
- Only 35 single word instructions
- All single cycle instructions except for program branches which are two cycle
- Operating speed: DC - 20 MHz clock input DC - 200 ns instruction cycle
- Up to 8K x 14 words of FLASH Program Memory,
- Up to 368 x 8 bytes of Data Memory (RAM)
- Up to 256 x 8 bytes of EEPROM data memory
- Interrupt capability (up to 14 sources)
- Eight level deep hardware stack
- Direct, indirect and relative addressing modes
- Power-on Reset (POR)
- Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Programmable code-protection
- Power saving SLEEP mode
- Selectable oscillator options
- Low-power, high-speed CMOS FLASH/EEPROM technology
- Fully static design
- In-Circuit Serial Programming (ICSP) via two pins
- Single 5V In-Circuit Serial Programming capability

- In-Circuit Debugging via two pins
- Processor read/write access to program memory
- Wide operating voltage range: 2.0V to 5.5V
- High Sink/Source Current: 25 mA
- Commercial and Industrial temperature ranges
- Low-power consumption:
 - < 2 mA typical @ 5V, 4 MHz
 - 20 mA typical @ 3V, 32 kHz

2.12 PERIPHERAL FEATURES:

- Timer0: 8-bit timer/counter with 8-bit prescaler
- Timer1: 16-bit timer/counter with prescaler, can be incremented during sleep via
 - ✓ External crystal/clock
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- Two Capture, Compare, PWM modules
 - ✓ Capture is 16-bit, max. Resolution is 12.5 ns
 - ✓ Compare is 16-bit, max. Resolution is 200 ns
 - ✓ PWM max. Resolution is 10-bit
- 10-bit multi-channel Analog-to-Digital converter
- Synchronous Serial Port (SSP) with SPI (Master Mode) and I2C (Master/Slave)
- Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI) with 9-bit address detection

- Parallel Slave Port (PSP) 8-bits wide, with external RD, WR and CS controls (40/44-pin only)

2.13 MEMORY ORGANIZATION:

There are three memory blocks in each of these PICmicro MCUs. The Program Memory and Data Memory have separate buses so that concurrent access can occur.

2.13.1 PROGRAM MEMORY ORGANIZATION:

The PIC16F87X devices have a 13-bit program counter capable of addressing an 8K x 14 program memory space. The PIC16F877/876 devices have 8K x 14 words of FLASH program memory and the PIC16F873/874 devices have 4K x 14. Accessing a location above the physically implemented address will cause a wraparound. The reset vector is at 0000h and the interrupt vector is at 0004h.

2.13.2 DATA MEMORY ORGANIZATION:

The data memory is partitioned into multiple banks which contain the General Purpose Registers and the Special Function Registers. Bits RP1(STATUS<6>) and RP0 (STATUS<5>) are the bank select bits.

Each bank extends up to 7Fh (128 bytes). The lower locations of each bank are reserved for the Special Function Registers. Above the Special Function Registers are General Purpose Registers, implemented as static RAM.

2.14 I/O PORTS:

Some pins for these I/O ports are multiplexed with an alternate function for the peripheral features on the device. In general, when a peripheral is enabled, that pin may not be used as a general purpose I/O pin.

2.14.1 PORTA AND THE TRISA REGISTER:

PORTA is a 6-bit wide bi-directional port. The corresponding data direction register is TRISA. Setting a TRISA bit (=1) will make the corresponding PORTA pin an input (i.e., put the corresponding output driver in a hi-impedance mode). Clearing a TRISA bit (=0) will make the corresponding PORTA pin an output (i.e., put the contents of the output latch on the selected pin). Reading the PORTA register reads the status of the pins, whereas writing to it will write to the port latch. Pin RA4 is multiplexed with the Timer0 module clock input to become the RA4/T0CKI pin. The RA4/T0CKI pin is a Schmitt Trigger input and an open drain output. All other PORTA pins have TTL input levels and full CMOS output drivers. Other PORTA pins are multiplexed with analog inputs and analog VREF input. The operation of each pin is selected by clearing/setting the control bits in the ADCON1 register (A/D Control Register1)

2.14.2 PORTB AND THE TRISB REGISTER:

PORTB is an 8-bit wide, bi-directional port. The corresponding data direction register is TRISB. Setting a TRISB bit (=1) will make the corresponding PORTB pin an input. Clearing a TRISB bit (=0) will make the corresponding PORTB pin an output. Three pins of PORTB are multiplexed

with the Low Voltage Programming function; RB3/PGM, RB6/PGC and RB7/PGD. Each of the PORTB pins has a weak internal pull-up. A single control bit can turn on all the pull-ups. This is performed by clearing bit RBPU (OPTION_REG<7>). The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on a Power-on Reset. Four of PORTB's pins, RB7:RB4, have an interrupt on change feature. Only pins configured as inputs can cause this interrupt to occur. The input pins (of RB7:RB4) are compared with the old value latched on the last read of

PORTB. The "mismatch" outputs of RB7:RB4 are OR'ed together to generate the RB Port Change Interrupt with flag bit RBIF (INTCON<0>).

2.14.3 PORTC AND THE TRISC REGISTER:

PORTC is an 8-bit wide, bi-directional port. The corresponding data direction register is TRISC. Setting a TRISC bit (=1) will make the corresponding PORTC pin an input. Clearing a TRISC bit (=0) will make the corresponding PORTC pin an output. When the I2C module is enabled, the PORTC (3:4) pins can be configured with normal I2C levels or with SMBUS levels by using the CKE bit (SSPSTAT <6>).

2.14.4 PORTD AND TRISD REGISTERS:

PORTD is an 8-bit port with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output. PORTD can be configured as an 8-bit wide microprocessor port (parallel slave port) by setting control bit PSPMODE (TRISE<4>). In this mode, the input buffers are TTL.

2.14.5 PORTE AND TRISE REGISTER:

PORTE has three pins, RE0/RD/AN5, RE1/WR/AN6 and RE2/CS/AN7, which are individually configurable as inputs or outputs. These pins have Schmitt Trigger input buffers. I/O PORTE becomes control inputs for the microprocessor port when bit PSPMODE (TRISE<4>) is set. In this mode, the input buffers are TTL. PORTE pins are multiplexed with analog inputs. When selected as an analog input, these pins will read as '0's. TRISE controls the direction of the RE pins, even when they are being used as analog inputs.

2.15 TIMER0 MODULE:

The Timer0 module timer/counter has the following features:

- 8-bit timer/counter
- Readable and writable
- 8-bit software programmable prescaler
- Internal or external clock select
- Interrupt on overflow from FFh to 00h
- Edge select for external clock

Timer mode is selected by clearing bit T0CS (OPTION_REG<5>). In timer mode, the Timer0 module will increment every instruction cycle (without prescaler). If the TMR0 register is written, the increment is inhibited for the following two instruction cycles.

The user can work around this by writing an adjusted value to the TMR0 register. Counter mode is selected by setting bit T0CS (OPTION_REG<5>). In counter mode, Timer0 will increment either on every rising or falling edge of pin RA4/T0CKI.

2.16 MASTER SYNCHRONOUS SERIAL PORT (MSSP) MODULE:

The Master Synchronous Serial Port (MSSP) module is a serial interface useful for communicating with other peripheral or microcontroller devices. These peripheral devices may be serial EEPROMs, shift registers, display drivers, A/D converters, etc. The MSSP module can operate in one of two modes:

- Serial Peripheral Interface (SPI).
- Inter-Integrated Circuit (I2C).

2.16.1 SPI Mode:

The SPI mode allows 8 bits of data to be synchronously transmitted and received simultaneously. All four modes of SPI are supported. To accomplish communication, typically three pins are used:

- Serial Data out (SDO)
- Serial Data in (SDI)
- Serial Clock (SCK)

Additionally, a fourth pin may be used when in a slave mode of operation: Slave Select (SS) to enable the serial port, MSSP Enable bit, SSPEN (SSPCON<5>) must be set. To reset or reconfigure SPI mode, clear bit SSPEN, re-initialize the SSPCON registers, and then set bit SSPEN.

Figure shows the block diagram of the MSSP module when in SPI mode. This configures the SDI, SDO, SCK and SS pins as serial port pins. For the pins to behave as the serial port function, some must have their data direction bits (in the TRIS register) appropriately programmed. That is:

- SDI is automatically controlled by the SPI module

- SDO must have TRISC<5> cleared
- SCK (Master mode) must have TRISC<3> cleared
- SCK (Slave mode) must have TRISC<3> set
- SS must have TRISA<5> se

2.16.2 MSSP I2C Operation:

The MSSP module in I2C mode fully implements all master and slave functions (including general call support) and provides interrupts-on-start and stop bits in hardware to determine a free bus (multi-master function). The MSSP module implements the standard mode specifications, as well as 7-bit and 10-bit addressing

Two pins are used for data transfer. These are the SCL pin, which is the clock, and the SDA pin, which is the data. The SDA and SCL pins are automatically configured when the I2C mode is enabled. The SSP module functions are enabled by setting SSP Enable bit SSPEN (SSPCON<5>). The SSPCON register allows control of the I2C operation. Four mode selection bits (SSPCON<3:0>) allow one of the following I2C modes to be selected:

- I2C Slave mode (7-bit address)
- I2C Slave mode (10-bit address)
- I2C Master mode, clock = OSC/4 (SSPADD +1)

The SSPSTAT register gives the status of the data transfer. SSPBUF is the register to which the transfer data is written to or read from. In receive operations; the SSPBUF and SSPSR create a doubled buffered receiver. This allows reception of the next byte to begin before reading the last byte of received data. When the complete byte is received, it is transferred to the SSPBUF register and flag bit SSPIF is set.

2.17 PIN DIAGRAM:

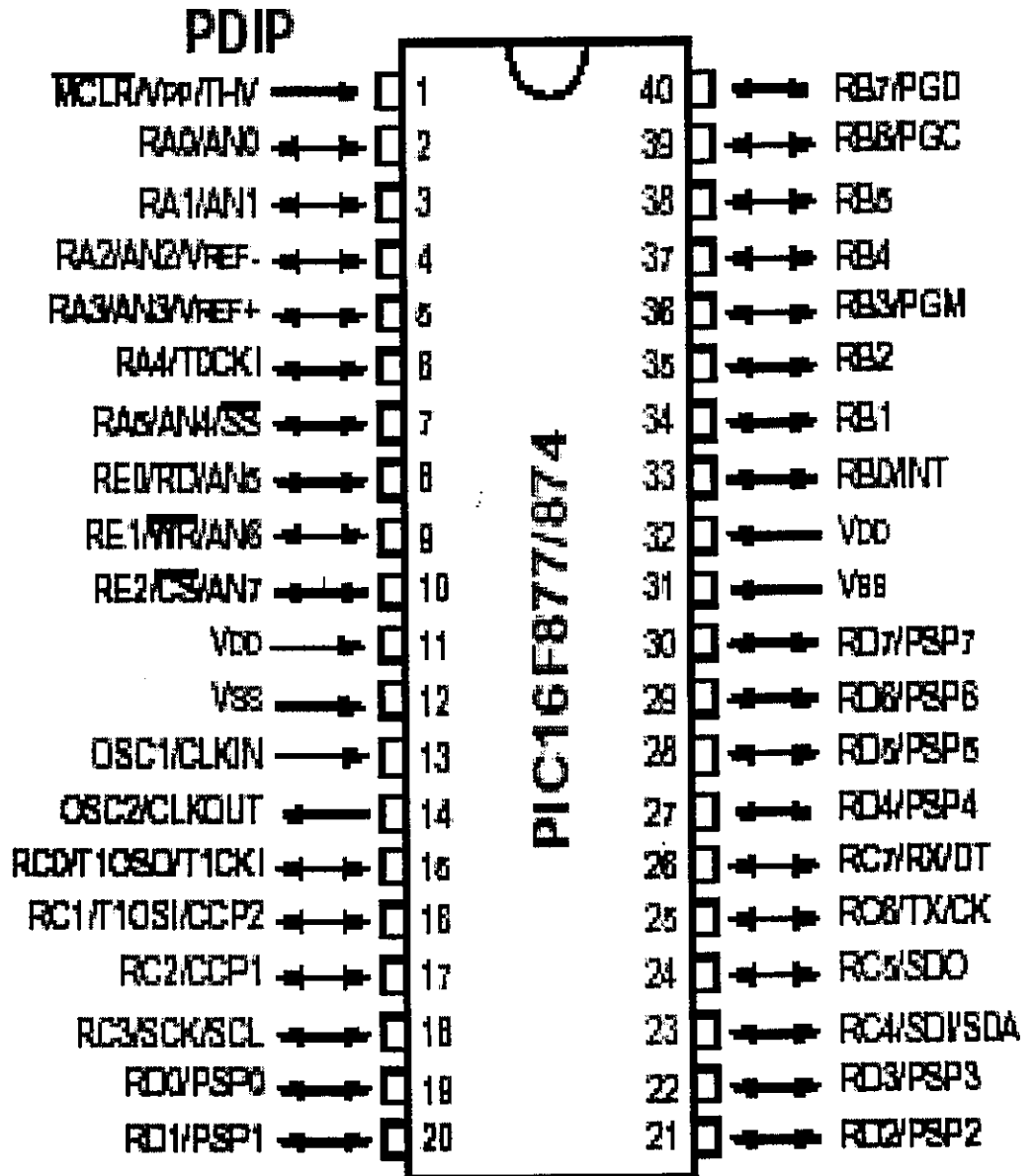


FIG 9: Pin diagram of PIC

CHAPTER III

3.1 INTRODUCTION TO I²C :

I²C is a multi-master serial computer bus invented by Philips that is used to attach low-speed peripherals to a motherboard, embedded system, or cellphone. The name stands for Inter-Integrated Circuit and is pronounced I-squared-C and also, incorrectly, I-two-C.

I²C uses only two bidirectional open-drain lines, Serial Data (SDA) and Serial Clock (SCL), pulled up with resistors. The I²C reference design has a 7-bit address space with 16 reserved addresses, so a maximum of 112 nodes can communicate on the same bus. The maximum number of nodes is obviously limited by the address space, and also by the total bus capacitance of 400 pF. To maximize hardware efficiency and circuit simplicity, Philips developed a simple bi-directional 2-wire bus for efficient inter-IC control.

This bus is called the Inter IC or I2C-bus. All I2C-bus compatible devices incorporate an on-chip interface which allows them to communicate directly with each other via the I2C-bus. This design concept solves the many interfacing problems encountered when designing digital control circuits. Here are some of the features of the I2C-bus:

- Only two bus lines are required; a serial data line (SDA) and a serial clock line (SCL)
- Each device connected to the bus is software addressable by a unique address and simple master/slave relationships exist at all times; masters can operate as master-transmitters or as master-receivers

- It's a true multi-master bus including collision detection and arbitration to prevent data corruption if two or more masters simultaneously initiate data transfer
- Serial, 8-bit oriented, bi-directional data transfers can be made at up to 100 kbit/s in the Standard-mode, up to 400 kbit/s in the Fast-mode, or up to 3.4 Mbit/s in the High-speed mode

For 8-bit oriented digital control applications, such as those requiring microcontrollers, certain design criteria can be established:

- A complete system usually consists of at least one microcontroller and other peripheral devices such as memories and I/O expanders
- The cost of connecting the various devices within the system must be minimized
- A system that performs a control function doesn't require high-speed data transfer
- Overall efficiency depends on the devices chosen and the nature of the interconnecting bus structure.

To produce a system to satisfy these criteria, a serial bus structure is needed. Although serial buses don't have the throughput capability of parallel buses, they do require less wiring and fewer IC connecting pins. However, a bus is not merely an interconnecting wire, it embodies all the formats and procedures for communication within the system.

Devices communicating with each other on a serial bus must have some form of protocol which avoids all possibilities of confusion, data loss and blockage of information. Fast devices must be able to communicate with slow devices. A procedure has also to be devised to decide which device will be in control of the bus and when. And, if different devices with different clock

speeds are connected to the bus, the bus clock source must be defined. All these criteria are involved in the specification of the I2C-bus.

3.2 THE I²C BUS PROTOCOL:

The I2C bus physically consists of 2 active wires and a ground connection. The active wires, called SDA and SCL, are both bi-directional. SDA is the Serial Data line, and SCL is the Serial Clock line.

Every device hooked up to the bus has its own unique address, no matter whether it is an MCU, LCD driver, memory, or ASIC. Each of these chips can act as a receiver and/or transmitter, depending on the functionality. Obviously, an LCD driver is only a receiver, while a memory or I/O chip can be both transmitter and receiver.

The I2C bus is a multi-master bus. This means that more than one IC capable of initiating a data transfer can be connected to it. The I2C protocol specification states that the IC that initiates a data transfer on the bus is considered the Bus Master. Consequently, at that time, all the other ICs are regarded to be Bus Slaves.

3.3 DESIGNER BENEFITS:

I2C-bus compatible ICs allow a system design to rapidly progress directly from a functional block diagram to a prototype. Moreover, since they 'clip' directly onto the I2C-bus without any additional external interfacing, they allow a prototype system to be modified or upgraded simply by 'clipping' or 'unclipping' ICs to or from the bus. Here are some of the features of I2C-bus compatible ICs which are particularly attractive to designers:

- Functional blocks on the block diagram correspond with the actual ICs; designs proceed rapidly from block diagram to final schematic.
- No need to design bus interfaces because the I2C-bus interface is already integrated on-chip.
- Integrated addressing and data-transfer protocol allow systems to be completely software-defined
- The same IC types can often be used in many different applications
- Design-time reduces as designers quickly become familiar with the frequently used functional blocks represented by I2C-bus compatible ICs
- ICs can be added to or removed from a system without affecting any other circuits on the bus
- Fault diagnosis and debugging are simple; malfunctions can be immediately traced
- Software development time can be reduced by assembling a library of reusable software modules.

3.4 I²C BUS CONCEPT:

The I2C-bus supports any IC fabrication process (NMOS, CMOS, bipolar). Two wires, serial data (SDA) and serial clock (SCL), carry information between the devices connected to the bus. Each device is recognized by a unique address (whether it's a microcontroller, LCD driver, memory or keyboard interface) and can operate as either a transmitter or receiver, depending on the function of the device. Obviously an LCD driver is only a receiver, whereas a memory can both receive and transmit data. In addition to transmitters and receivers, devices can also be considered as masters or slaves when performing data transfers (see Table 1). A master is the device which initiates a data transfer

on the bus and generates the clock signals to permit that transfer. At that time, any device addressed is considered a slave.

Table 1 Definition of I²C-bus terminology

TERM	DESCRIPTION
Transmitter	The device which sends data to the bus
Receiver	The device which receives data from the bus
Master	The device which initiates a transfer, generates clock signals and terminates a transfer
Slave	The device addressed by a master
Multi-master	More than one master can attempt to control the bus at the same time without corrupting the message
Arbitration	Procedure to ensure that, if more than one master simultaneously tries to control the bus, only one is allowed to do so and the winning message is not corrupted
Synchronization	Procedure to synchronize the clock signals of two or more devices

The I²C-bus is a multi-master bus. This means that more than one device capable of controlling the bus can be connected to it. The possibility of connecting more than one microcontroller to the I²C-bus means that more than one master could try to initiate a data transfer at the same time.

To avoid the chaos that might ensue from such an event - an arbitration procedure has been developed. This procedure relies on the wired-AND connection of all I²C interfaces to the I²C-bus. If two or more masters try to put information onto the bus, the first to produce a 'one' when the other produces a 'zero' will lose the arbitration. The clock signals during arbitration are a

synchronized combination of the clocks generated by the masters using the wired-AND connection to the SCL line. Generation of clock signals on the I2C-bus is always the responsibility of master devices; each master generates its own clock signals when transferring data on the bus. Bus clock signals from a master can only be altered when they are stretched by a slow-slave device holding-down the clock line or by another master when arbitration occurs.

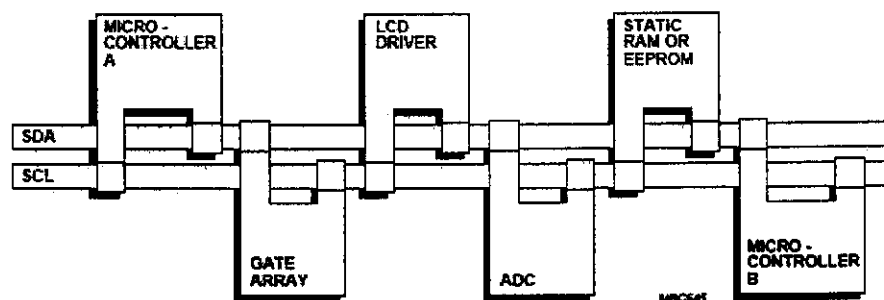


FIG 10: Example of I²C bus configuration using two micro controllers

Both SDA and SCL are bi-directional lines, connected to a positive supply voltage via a current-source or pull-up resistor (see Fig.3). When the bus is free, both lines are HIGH. The output stages of devices connected to the bus must have an open-drain or open-collector to perform the wired-AND function. Data on the I2C-bus can be transferred at rates of up to 100 kbit/s in the Standard-mode, up to 400 kbit/s in the Fast-mode, or up to 3.4 Mbit/s in the High-speed mode. The number of interfaces connected to the bus is solely dependent on the bus capacitance limit of 400 pF.

3.5 I²C SERIAL EEPROM:

The Microchip Technology Inc. 24LC04B/08B is a 4K or 8K bit Electrically Erasable PROM. The device is organized as two or four blocks of 256 x 8-bit memory with a 2-wire serial interface. Low voltage design permits operation down to 2.5 volts with typical standby and active currents of only 5mA and 1 mA respectively. The 24LC04B/08B also has a page-write capability for up to 16 bytes of data. The 4LC04B/08B is available in the standard 8-pin DIP and both 8-lead and 14-lead surface mount SOIC packages.

3.5.1 FEATURES:

- Single supply with operation down to 2.5V
- Low power CMOS technology
- 1 mA active current typical
- 10mA standby current typical at 5.5V
- 5mA standby current typical at 3.0V
- Organized as two or four blocks of 256 bytes(2 x 256 x 8) and (4 x 256 x 8)
- 2-wire serial interface bus, I²C compatible
- Schmitt trigger, filtered inputs for noise suppression
- Output slope control to eliminate ground bounce
- 100 kHz (2.5V) and 400 kHz (5V) compatibility
- Self-timed write cycle (including auto-erase)
- Page-write buffer for up to 16 bytes
- 2 ms typical write cycle time for page-write
- Hardware write protect for entire memory
- Can be operated as a serial ROM
- Factory programming (QTP) available

- ESD protection > 4,000V
- 1,000,000 erase/write cycles guaranteed
- Data retention > 200 years
- 8-pin DIP, 8-lead or 14-lead SOIC packages
- Available for extended temperature ranges

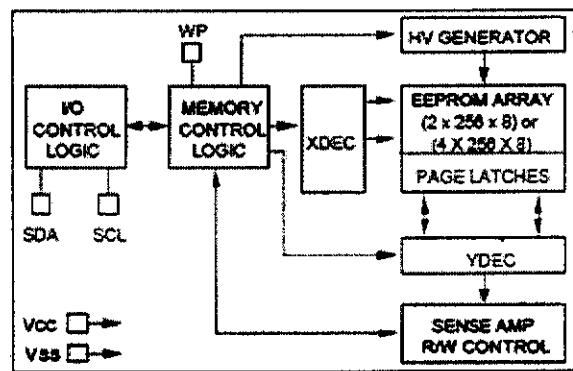


FIG 11: Block diagram of i^2c serial EEPROM

3.5.2 FUNCTIONAL DESCRIPTION:

The 24LC04B/08B supports a Bi-directional 2-wire bus and data transmission protocol. A device that sends data onto the bus is defined as transmitter, and a device receiving data as receiver. The bus has to be controlled by a master device which generates the serial clock (SCL), controls the bus access, and generates the START and STOP conditions, while the 24LC04B/08B works as slave. Both, master and slave can operate as transmitter or receiver but the master device determines which mode is activated.

3.6 BUS CHARACTERISTICS:

The following bus protocol has been defined:

- Data transfer may be initiated only when the bus is not busy.
- During data transfer, the data line must remain stable whenever the clock line is HIGH. Changes in the data line while the clock line is HIGH will be interpreted as a START or STOP condition. Accordingly, the following bus conditions have been defined.\

3.6.1 BUS NOT BUSY (A):

Both data and clock lines remain HIGH.

3.6.2 START DATA TRANSFER (B):

A HIGH to LOW transition of the SDA line while the clock (SCL) is HIGH determines a START condition. All commands must be preceded by a START condition.

3.6.3 STOP DATA TRANSFER (C):

A LOW to HIGH transition of the SDA line while the clock (SCL) is HIGH determines a STOP condition. All operations must be ended with a STOP condition.

3.6.4 DATA VALID (D):

The state of the data line represents valid data when, after a START condition, the data line is stable for the duration of the HIGH period of the clock signal. The data on the line must be changed during the LOW period of the clock signal. There is one clock pulse per bit of data. Each data transfer is initiated with a START condition and terminated with a STOP condition. The

number of the data bytes transferred between the START and STOP conditions is determined by the master device and is theoretically unlimited, although only the last 16 will be stored when doing a write operation. When an overwrite does occur it will replace data in a first in first out fashion.

3.6.5 ACKNOWLEDGE:

Each receiving device, when addressed, is obliged to generate an acknowledge after the reception of each byte. The master device must generate an extra clock pulse which is associated with this acknowledge bit. The device that acknowledges, has to pull down the

SDA line during the acknowledge clock pulse in such a way that the SDA line is stable LOW during the HIGH period of the acknowledge related clock pulse. Of course, setup and hold times must be taken into account. A master must signal an end of data to the slave by not generating an acknowledge bit on the last byte that has been clocked out of the slave. In this case, the slave must leave the data line HIGH to enable the master to generate the STOP condition.

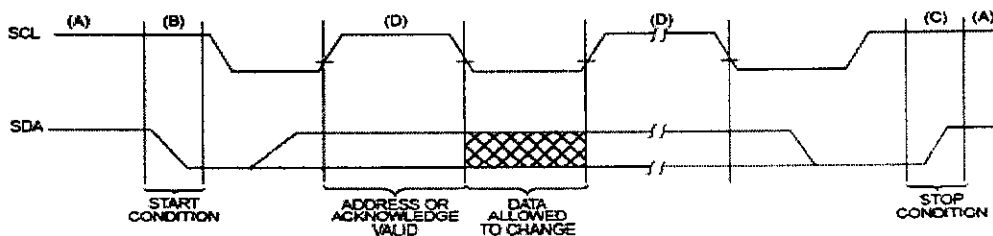


FIG 12: Data transfer sequence in serial bus

3.6.6 DEVICE ADDRESSING:

A control byte is the first byte received following the start condition from the master device. The control byte consists of a 4-bit control code, for the

24LC04B/08B this is set as 1010 binary for read and writes operations. The next three bits of the control byte are the block select bits (B2, B1, B0). B2 is a don't care for both the 24LC04B and 24LC08B; B1 is a don't care for the 24LC04B. They are used by the master device to select which of the two or four 256 word blocks of memory are to be accessed. The last bit of the control byte defines the operation to be performed. When set to one a read operation is selected, when set to zero a write operation is selected.

3.7 WRITE OPERATION:

3.7.1 BYTE WRITE:

Following the start condition from the master, the device code (4 bits), the block address (3 bits), and the R/W bit which is a logic low is placed onto the bus by the master transmitter. Therefore the next byte transmitted by the master is the word address and will be written into the address pointer of the 24LC04B/08B. After receiving another acknowledge signal from the 24LC04B/08B the master device will transmit the data word to be written into the addressed memory location. The 24LC04B/08B acknowledges again and the master generates a stop condition. This initiates the internal write cycle, and during this time the 24LC04B/08B will not generate acknowledge signals.

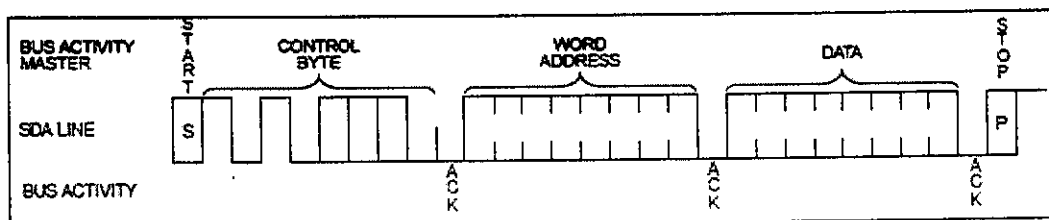


FIG 13: Byte write

3.7.2 PAGE WRITE:

The write control byte, word address and the first data byte are transmitted to the 24LC04B/08B in the same way as in a byte write. But instead of generating a stop condition the master transmits up to 16 data bytes to the 24LC04B/08B which are temporarily stored in the on-chip page buffer and will be written into the memory after the master has transmitted a stop condition. After the receipt of each word, the four lower order address pointer bits are internally incremented by one. The higher order seven bits of the word address remains constant. If the master should transmit more than 16 words prior to generating the stop condition, the address counter will roll over and the previously received data will be overwritten. As with the byte write operation, once the stop condition is received an internal write cycle will begin

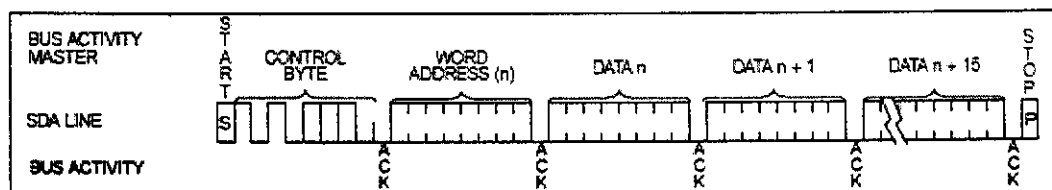


FIG 14: Page write

3.7.3 ACKNOWLEDGE POLLING:

Since the device will not acknowledge during a write cycle, this can be used to determine when the cycle is complete (this feature can be used to maximize bus throughput). Once the stop condition for a write command has been issued from the master, the device initiates the internally timed write cycle. ACK polling can be initiated immediately. This involves the master sending a

start condition followed by the control byte for a write command ($R/W = 0$). If the device is still busy with the write cycle, then no ACK will be returned. If the cycle is complete, then the device will return the ACK and the master can then proceed with the next read or write command.

3.7.4 WRITE PROTECTION:

The 24LC04B/08B can be used as a serial ROM when the WP pin is connected to VCC. Programming will be inhibited and the entire memory will be write-protected.

3.8 READ OPERATION:

Read operations are initiated in the same way as write operations with the exception that the R/W bit of the slave address is set to one. There are three basic types of read operations: current address read, random read, and sequential read.

3.8.1 CURRENT ADDRESS READ:

The 24LC04B/08B contains an address counter that maintains the address of the last word accessed, internally incremented by one. Therefore, if the previous access (either a read or write operation) was to address n , the next current address read operation would access data from address $n + 1$. Upon receipt of the slave address with R/W bit set to one, the 24LC04B/08B issues an acknowledge and transmits the 8-bit data word.

3.8.2 RANDOM READ:

Random read operations allow the master to access any memory location in a random manner. To perform this type of read operation, first the word address must be set. This is done by sending the word address to the

24LC04B/08B as part of a write operation. After the word address is sent, the master generates a start condition following the acknowledge. The 24LC04B/08B will then issue an acknowledge and transmits the 8-bit data word. The master will not acknowledge the transfer but does generate a stop condition and the 24LC04B/08B discontinues transmission.

3.8.3 SEQUENTIAL READ:

Sequential reads are initiated in the same way as a random read except that after the 24LC04B/08B transmits the first data byte, the master issues an acknowledge as opposed to a stop condition in a random read. This directs the 24LC04B/08B to transmit the next sequentially addressed 8-bit word. To provide sequential reads the 24LC04B/08B contains an internal address pointer which is incremented by one at the completion of each operation. This address pointer allows the entire memory contents to be serially read during one operation.

3.9 NOISE PROTECTION:

The 24LC04B/08B employs a VCC threshold detector circuit which disables the internal erase/write logic if the VCC is below 1.5 volts at nominal conditions. The SCL and SDA inputs have Schmitt trigger and filter circuits .

3.10 PIN DESCRIPTIONS:

3.10.1 SDA SERIAL ADDRESS/DATA INPUT/OUTPUT:

This is a Bi-directional pin used to transfer addresses and data into and data out of the device. It is an open drain terminal, therefore the SDA bus requires a pullup resistor to VCC (typical 10KW for 100 kHz, 2 KW for 400 kHz). For normal data transfer SDA is allowed to change only during SCL low.

Changes during SCL high are reserved for indicating the START and STOP conditions.

3.10.2 SCL SERIAL CLOCK:

This input is used to synchronize the data transfer from and to the device.

3.10.2.1 WP:

This pin must be connected to either VSS or VCC. If tied to VSS, normal memory operation is enabled (read/write the entire memory). If tied to VCC, WRITE operations are inhibited. The entire memory will be write-protected. Read operations are not affected. This feature allows the user to use the 24LC04B/08B as a serial ROM when WP is enabled (tied to VCC).

3.10.2.2 A0, A1, A2:

These pins are not used by the 24LC04B/08B. They may be left floating or tied to either VSS or VCC.

CHAPTER IV

4.1 MEMS ACCELEROMETER:

An accelerometer is a device for measuring acceleration and gravity induced reaction forces. Single- and multi-axis models are available to detect magnitude and direction of the acceleration as a vector quantity. Accelerometers can be used to sense inclination, vibration, and shock. They are increasingly present in portable electronic devices.

Modern accelerometers are often small micro electro-mechanical systems (MEMS), and are indeed the simplest MEMS devices possible, consisting of little more than a cantilever beam with a proof mass (also known as seismic mass). Mechanically the accelerometer behaves as a mass-damper-spring system; the damping results from the residual gas sealed in the device. As long as the Q-factor is not too low, damping does not result in a lower sensitivity.

Under the influence of gravity or acceleration the proof mass deflects from its neutral position. This deflection is measured in an analog or digital manner. Most commonly the capacitance between a set of fixed beams and a set of beams attached to the proof mass is measured. This method is simple and reliable; it also does not require additional process steps making it inexpensive. Integrating piezoresistors in the springs to detect spring deformation, and thus deflection, is a good alternative, although a few more process is needed. For very high sensitivities quantum tunneling is also used; this requires specific fabrication steps making it more expensive. Optical measurement has been demonstrated on laboratory scale.

Another, far less common, type of MEMS-based accelerometer contains a small heater at the bottom of a very small dome, which heats the air inside the

dome to cause it to rise. A thermocouple on the dome determines where the heated air reaches the dome and the deflection off the center is a measure of the acceleration applied to the sensor.

Most micromechanical accelerometers operate in-plane, that is, they are designed to be sensitive only to a direction in the plane of the die. By integrating two devices perpendicularly on a single die a two-axis accelerometer can be made. By adding an additional out-of-plane device three axes can be measured.

Micromechanical accelerometers are available in a wide variety of measuring ranges, reaching up to thousands of g's. The designer must make a compromise between sensitivity and the maximal acceleration that can be measured.

4.2 MEMS Accelerometer LIS302DL:

The LIS302DL is an ultra compact low-power three axes linear accelerometer. It includes a sensing element and an IC interface able to provide the measured acceleration to the external world through I2C/SPI serial interface. The sensing element, capable of detecting the acceleration, is manufactured using a dedicated process developed by ST to produce inertial sensors and actuators in silicon. The IC interface is manufactured using a CMOS process that allows to design a dedicated circuit which is trimmed to better match the sensing element characteristics. The LIS302DL has dynamically user selectable full scales of $\pm 2g/\pm 8g$ and it is capable of measuring accelerations with an output data rate of 100Hz or 400Hz. A self-test capability allows the user to check the functioning of the sensor in the final application. The device may be configured to generate inertial wake-up/free-fall interrupt signals when a programmable acceleration threshold is crossed at least in one of the three axes.

Thresholds and timing of interrupt generators are completely programmable by the end user on the fly. The LIS302DL is available in plastic Thin Land Grid Array package (TLGA) and it is guaranteed to operate over an extended temperature range from -40°C to $+85^{\circ}\text{C}$. The LIS302DL belongs to a family of products suitable for a variety of applications:

- Free-Fall detection
- Motion activated functions
- Gaming and Virtual Reality input devices
- Vibration Monitoring and Compensation

4.3 FEATURES:

- Three axes
- SPI/I2C digital interface
- Innovative embedded functionalities
- Two highly flexible and programmable interrupt request outputs
- Programmable thresholds and timing of interrupt signals
- 2.16V to 3.6V supply voltage
- 1.8V compatible I/Os
- $<1\text{mW}$ power consumption
- Temperature range -40 to $+85^{\circ}\text{C}$
- $\pm 2/\pm 8\text{g}$ selectable full scale range
- Embedded high pass filter
- Embedded self-test
- 10,000g high shock survivability
- LGA package $3\times 5\times 0.9\text{mm}^3$

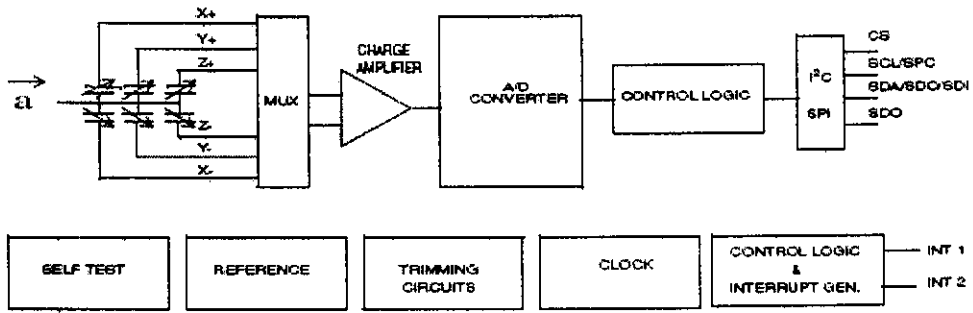


FIG 15: Block diagram of MEMS



FIG 16: Top view and bottom view

4.4 PIN DESCRIPTION:

Pin#	Name	Function
1	Vdd_IO	Power supply for I/O pins
2	GND	0V supply
3	Reserved	Connect to Vdd
4	GND	0V supply
5	GND	0V supply
6	Vdd_	Power supply
7	CS	SPI enables I2C/SPI mode selection
8	INT 1	Inertial interrupts 1
9	INT 2	Inertial interrupts 2
10	GND	0V supply
11	Reserved	Connect to Gnd
12	SDO	SPI Serial Data Output
13	SDA	I2C Serial Data (SDA)
14	SCL	I2C Serial Clock (SCL)

TABLE 2:Pin description

4.5 TERMINOLOGY:

4.5.1 SENSITIVITY:

Sensitivity describes the gain of the sensor and can be determined e.g. by applying 1 acceleration to it. As the sensor can measure DC accelerations this can be done easily by pointing the axis of interest towards the center of the earth, noting the output value, rotating the sensor by 180 degrees (point to the sky) and noting the output value again. By doing so, $\pm 1g$ acceleration is applied to the sensor. Subtracting the larger output value from the smaller one and

dividing the result by 2 leads to the actual sensitivity of the sensor. This value changes very little over temperature and also very little over time. The Sensitivity Tolerance describes the range of Sensitivities of a large population of sensor.

4.5.2 ZERO-G LEVEL:

Zero-g level Offset (Off) describes the deviation of an actual output signal from the ideal output signal if there is no acceleration present. A sensor in a steady state on a horizontal surface will measure 0g in X axis and 0g in Y axis whereas the Z axis will measure 1g. The output is ideally in the middle of the dynamic range of the sensor (content of OUT registers 00h, data expressed as 2's complement number). A deviation from ideal value in this case is called Zero-g offset. Offset is to some extent a result of stress to a precise MEMS sensor and therefore the offset can slightly change after mounting the sensor onto a printed circuit board or exposing it to extensive mechanical stress. Offset changes little over temperature; see "Zero-g level change vs. temperature".

4.5.3 FUNCTIONALITY:

The LIS302DL is a ultracompact, low-power, digital output 3-axis linear accelerometer packaged in a LGA package. The complete device includes a sensing element and an IC interface able to take the information from the sensing element and to provide a signal to the external world through an I2C/SPI serial interface.

4.5.4 SENSING ELEMENT:

A proprietary process is used to create a surface micro-machined accelerometer. The technology allows to carry out suspended silicon structures which are attached to the substrate in a few points called anchors and are free to move in the direction of the sensed acceleration. To be compatible with the traditional packaging techniques a cap is placed on top of the sensing element to avoid blocking the moving parts during the molding phase of the plastic encapsulation. When acceleration is applied to the sensor the proof mass displaces from its nominal position, causing an imbalance in the capacitive half-bridge. This imbalance is measured using charge integration in response to a voltage pulse applied to the sense capacitor. At steady state the nominal value of the capacitors are few pF and when acceleration is applied the maximum variation of the capacitive load is in FFrange.

4.6 IC INTERFACE:

The complete measurement chain is composed by a low-noise capacitive amplifier which converts into an analog voltage the capacitive unbalancing of the MEMS sensor and by analog-to-digital converters. The acceleration data may be accessed through an I2C/SPI interface thus making the device particularly suitable for direct interfacing with a microcontroller. The LIS302DL features a Data-Ready signal (RDY) which indicates when a new set of measured acceleration data is available thus simplifying data synchronization in digital system employing the device itself. The LIS302DL may also be configured to generate an inertial Wake-Up and Free-Fall interrupt signal accordingly to a programmed acceleration event along the enabled axes. Both Free-Fall and Wake-Up can be available simultaneously on two different pins.

4.7 POWER SUPPLY:

A power supply provides a constant output regardless of voltage variations. "Fixed" three-terminal linear regulators are commonly available to generate fixed voltages of plus 3 V, and plus or minus 5 V, 9 V, 12 V, or 15 V when the load is less than about 7 amperes.

The "78xx" series (7805, 7812, etc.) regulate positive voltages while the "79xx" series (7905, 7912, etc.) regulate negative voltages. Often, the last two digits of the device number are the output voltage; eg, a 7805 is a +5 V regulator, while a 7915 is a -15 V regulator. The 78xx series ICs can supply up to 1.5 Amperes depending on the model.

4.7.1 FEATURES:

- Output Current up to 1A
- Output Voltages of 5, 6, 8, 9, 10, 12, 15, 18, 24
- Thermal Overload Protection
- Short Circuit Protection
- Output Transistor Safe Operating Area Protection

When you have a requirement for a project of say 12V, or even 5V if it's a digital project, then these are the types you use. 7805 or 7812 are the types. There are of course negative voltage regulators with the numbers 79XX which are substantially the same as those discussed here excepting they are negative. We will not consider them further. Assume your project calls for a basic fixed 12V D.C. to operate. Looking back to our earlier tutorial we apply all the same principles. Look at the original schematic.

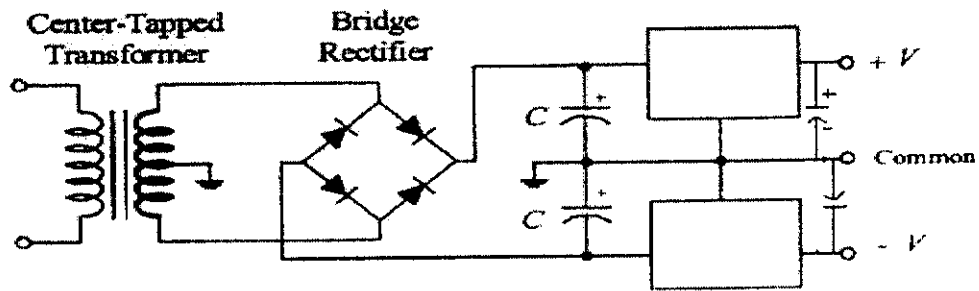


FIG 17: The basic power supply schematic

To obtain a DC power supply with both positive and negative output voltages, a center-tapped transformer is used, where a third wire is attached to the middle of the secondary winding and it is taken as the common ground point. Then voltages from the opposite ends of the winding will be positive or negative with respect to this point.

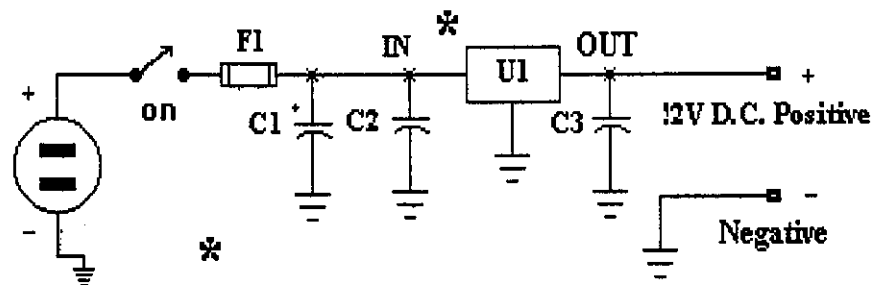


FIG 18: Fit heat sink

4.7.2 APPLICATIONS:

- Part of electronics devices, small laboratory power supply
- Power supply voltage: Unregulated DC 8-18V power supply
- Power supply current: Needed output current + 5 mA
- Component costs: Few dollars for the electronics components .

CHAPTER V

5.1 DISPLAY UNIT:

This section describes the operation modes of LCDs, then describes how to program and interface an LCD to PIC Microcontroller.

5.2 LCD OPERATION:

In recent years the LCD is finding widespread use replacing LEDs (seven-segment LEDs or other multi segment LEDs). This is due to the following reasons:

- The declining prices of LCDs.
- The ability to display numbers, characters, and graphics. This is in contrast to LEDs, which are limited to numbers and a few characters.
- Incorporation of a refreshing controller into the LCD, thereby relieving the CPU of the task of refreshing the LCD. In contrast, the LED must be refreshed by the CPU (or in some other way) to keep displaying the data.
- Ease of programming for characters and graphics.

5.3 LCD PIN DESCRIPTIONS:

The LCD discussed in this section has 14 pins. The function of each pin is given in the table below.

V_{CC} , V_{SS} , AND V_{EE} :

While V_{CC} and V_{SS} provide +5V and ground, respectively, V_{EE} is used for controlling LCD contrast.

RS, REGISTER SELECT :

There are two very important registers inside the LCD. The RS pin is used for their selection as follows. If RS=0, the instruction command code register is selected, allowing the user to send a command such as clear display, cursor at home, etc. If RS=1 the data register is selected, allowing the user to send data to be displayed on the LCD.

R/W, READ/WRITE:

R/W input allows the user to write information to the LCD or read information from it. R/W=1 when reading; R/W=0 when writing.

E, ENABLE:

The enable pin is used by the LCD to latch information presented to its data pins. When data is supplied to data pins, a high-to-low pulse must be applied to this pin in order for the LCD to latch in the data present at the data pins. This pulse must be a minimum of 450ns wide.

D0-D7:

The 8-bit data pins, D0-D7, are used to send information to the LCD or read the contents of the LCD's internal registers.

To display letters and numbers, we send ASCII codes for the letters A-Z, a-z, and numbers 0-9 to these pins while making RS=1.

We also use RS=0 to check the busy flag bit to see if the LCD is ready to receive information. The busy flag is D7 and can be read when R/W=1 and RS=0, as follows: if R/W=1, RS=0. When D7=1 (busy flag=1), the LCD is

busy taking care of internal operations and will not accept any new information.

Code(Hex)	Command to LCD Instruction Register
1	Clear display screen
2	Return home
4	Decrement cursor (shift cursor to left)
6	Increment cursor (shift cursor to right)
5	Shift display right
7	Shift display left
8	Display off, cursor off
A	Display off, cursor on
C	Display on, cursor off
E	Display on, cursor blinking
F	Display off, cursor blinking
10	Shift cursor position to left
14	Shift cursor position to right
18	Shift the entire display to the left
1C	Shift the entire display to the right
80	Force cursor to beginning of 1 st line
C0	Force cursor to beginning of 2 nd line
38	2 lines and 5x7 matrix

TABLE 3: HEX code for LCD display

Pin No	Symbol	Details
1	GND	Ground
2	Vcc	Supply Voltage +5V
3	Vo	Contrast adjustment
4	RS	0->Control input, 1-> Data input
5	R/W	Read/ Write
6	E	Enable
7 to 14	D0 to D7	Data
15	VB1	Backlight +5V
16	VB0	Backlight ground

TABLE 4: PIN details of LCD

5.4 STEPS TO INTERFACE LCD WITH PIC MICROCONTROLLER:

STEP 1: Identify:

Determine what you want LCD are available in many flavors which are specified as follows 16x1 , 16x2 , 20x2 in the format AxB where A is the number of columns (chatters) and B is the number of Rows (lines) An LCD might also be Back lit .

STEP 2 : Connect :

Most of the LCD's follow the standard Hitachi Pin out which is simple.

STEP 3 : Interface :

Now connect pins RS, RW, E, D0 - D7 to pins on the micro controller. Let's suppose I connect Data bus on port A and the RS, RW, E on port B. (you can save pins by using LCD in Nibble Mode (4 data pins) and permanently grounding the RW line (always in write mode)). Now we'll see how to go from simple switching it on to graphics on the LCD.

An Intelligent LCD Need Only a few Commands and data to function
Command Set for the LCD.

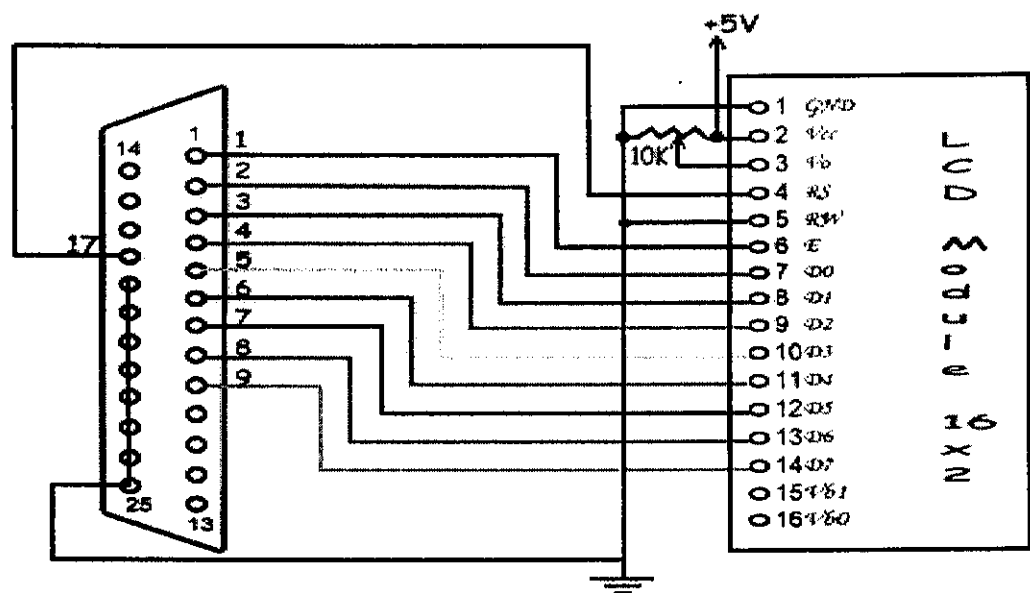


FIG 19: LCD interface

CHAPTER VI

6.1 CIRCUITS:

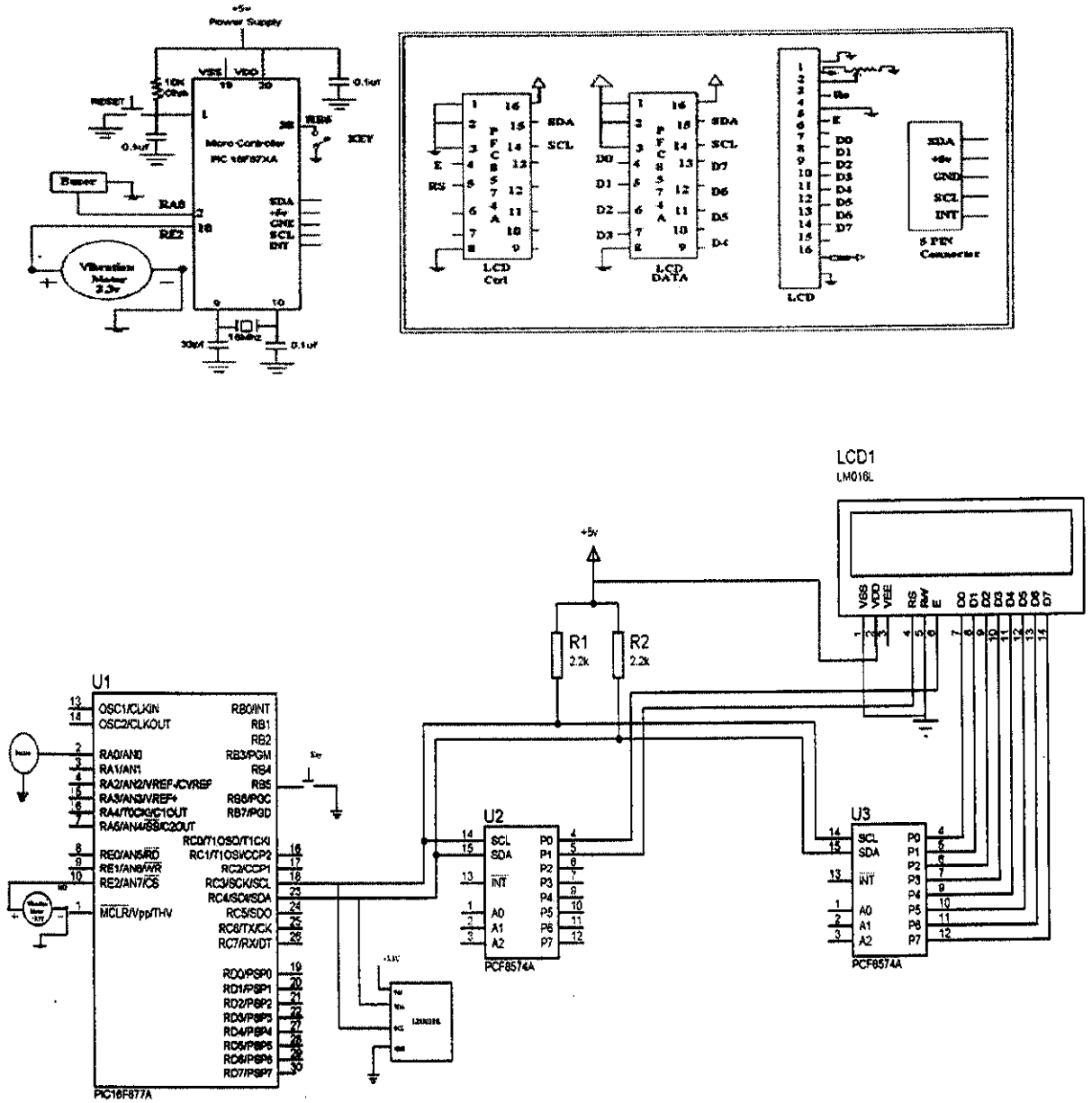


FIG 20: Circuit diagram

6.2 DEVELOPMENT TOOLS FOR PIC MICROCONTROLLERS:

6.2.1 MPLAB IDE:

MPLAB IDE is an easy to learn and use integrated development environment (IDE). The integrated development environment is an application that has multiple functions for firmware development. The MPLAB IDE integrates a compiler, an assembler, a project manager, an editor, a debugger, a simulator and an assortment of other tools within one window application. The IDE provides firmware development engineers the flexibility to develop and debug the firmware for microchips PIC microcontroller (MCU) families.

Steps to create firmware for an embedded system using MPLAB 6.43

- 1) Open the MPLAB 6.43 from the **Start**→ **Programs**→ **Microchip MPLAB IDE**→**MPLAB IDE**.
- 2) Select **Project Wizard** from the **Project** menu. This wizard helps you to create and configure a new MPLAB project. Click **Next**.
- 3) Select a device. For example: **PIC16F877A**.
- 4) Select a Language Tool suite. For example: **Microchip MPASM Tool suite**.
- 5) Name your project and select a project directory.
- 6) Add any existing file to your project (optional).
- 7) Click **Finish** to create a new project. A new Workspace will be created and the new project added to that workspace.
- 8) To write a source file for your project, select **File**→ **New** option. A new text editor is created for entering the assembly language or C language code.

- 9) After completion of entering the code, save it with the extension <file name>.asm (for assembly language) or <file name>.c (for C language).
- 10) Add source code to your project by selecting **Add Files to Project** option from the **Project** menu.
- 11) Assemble or compile the project by choosing **Build All** option from the **Project** menu.
- 12) If you have written your program without errors you will get a message **Build Succeeded** else **build Failed** along with errors and their types. A hexadecimal file of your project is created with the extension **.hex**

Now you have machine language version of your project ready for fusing it into the program memory of the PIC microcontroller. The downloading of the hex file can be done using software called **ICPROG**.

6.2.2 IC-PROG(Hex code programmer):

IC-prog is windows based software to control a development programmer for pic microcontroller. To operate the software, a basic knowledge about electronics and windows OS is necessary. In order for this software to operate you have to attach a programmer to your computer and set up the hardware and the software appropriately. IC-prog has been designed as a universal programming application for all programmers.

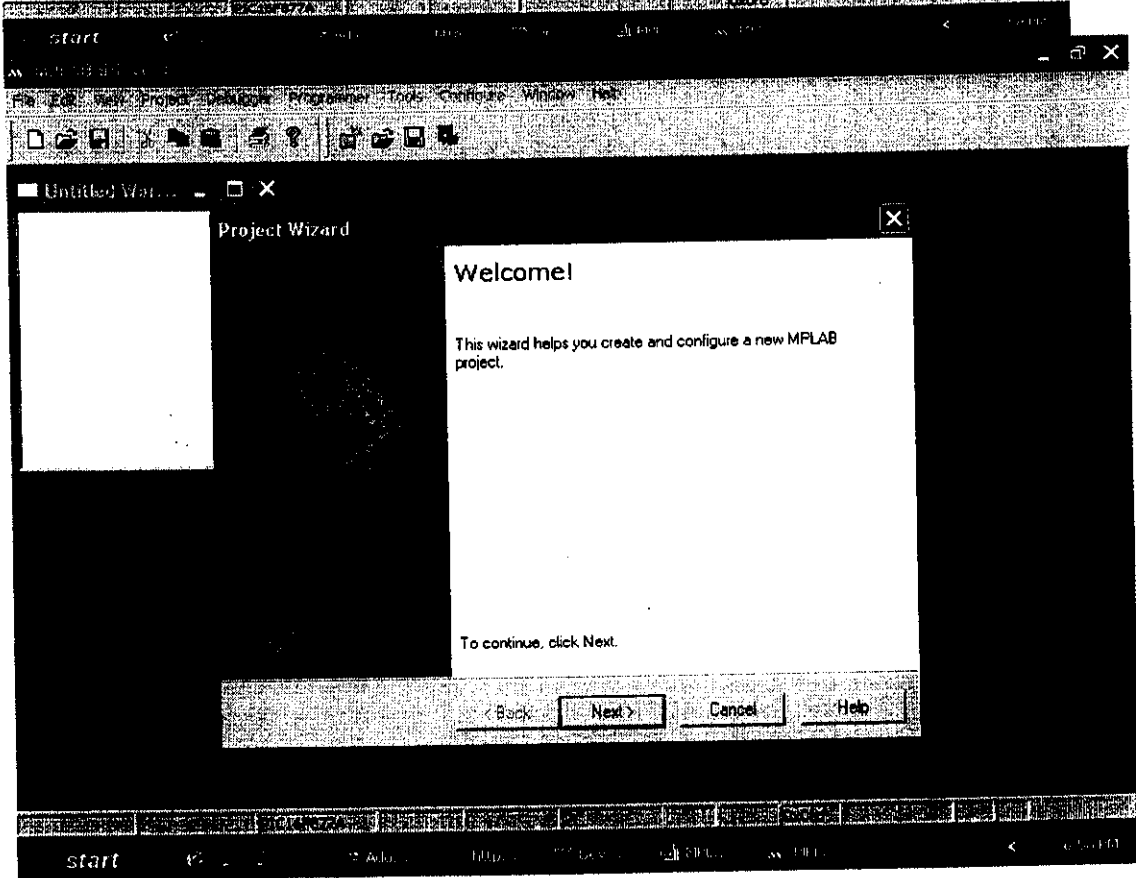
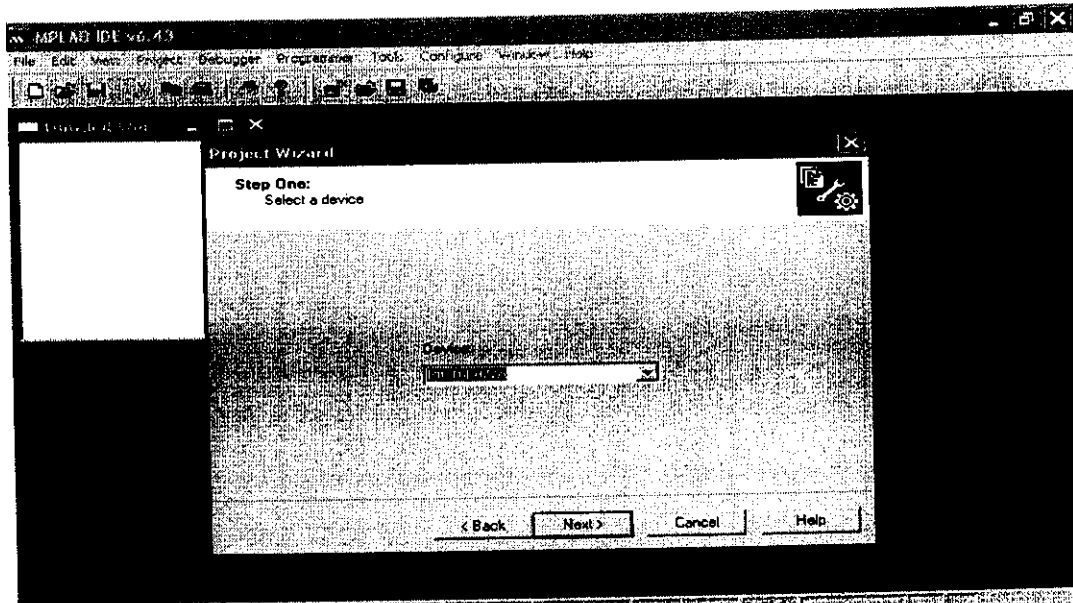
6.2.2.1 STEPS TO FUSE THE HEX CODE INTO MC CHIP USING IC PROG:

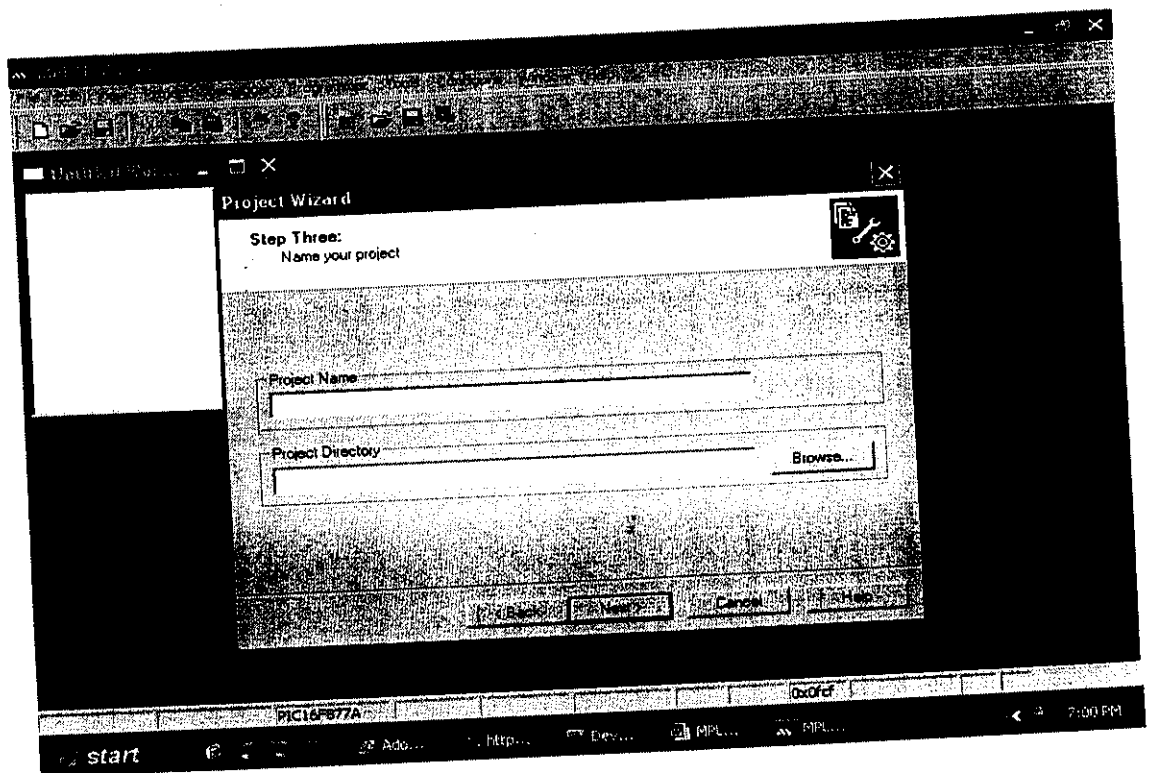
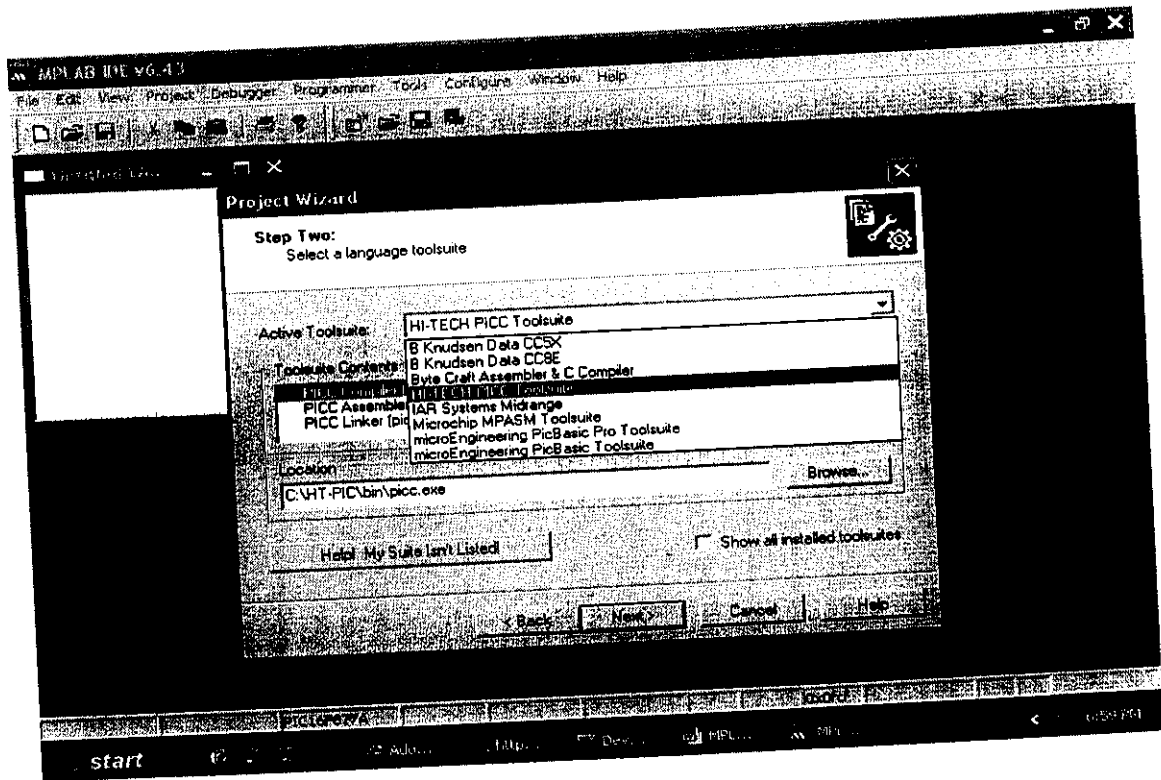
- Select the target device from the device list. For Example:16F877A
- Select the oscillator type as **HS**, write enable for **0000-0FFFh** and enable the fuses **PWRT** and disable all other fuses.

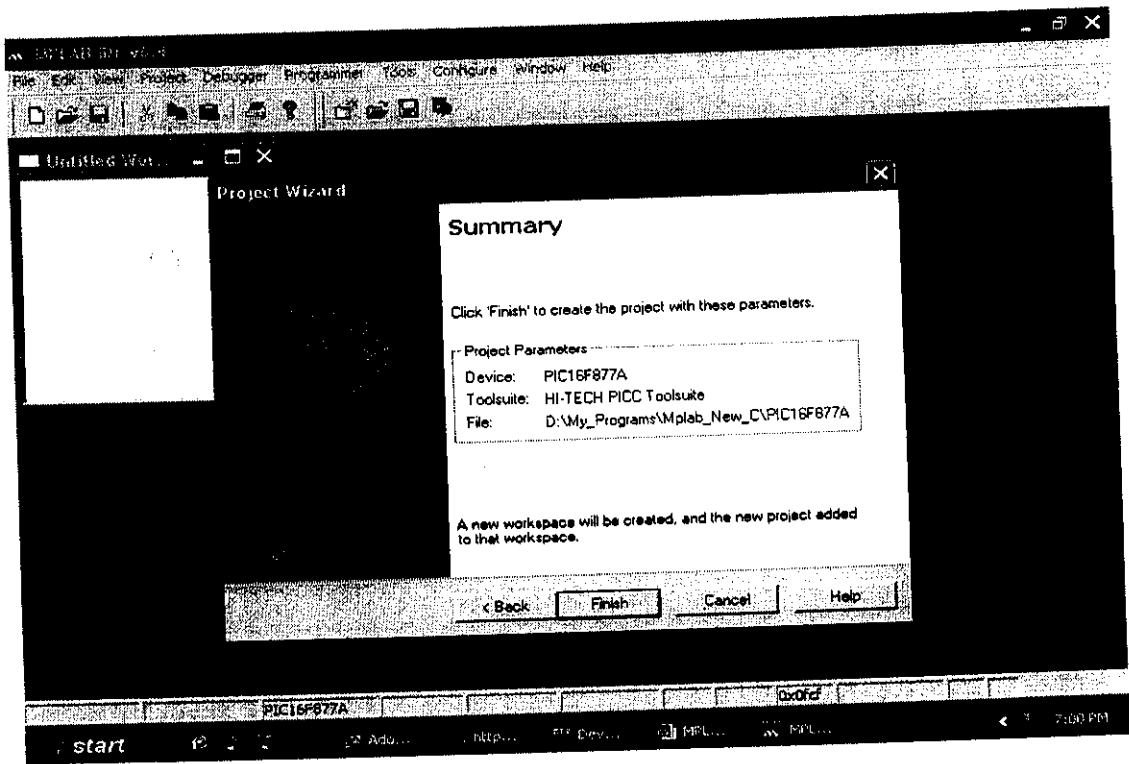
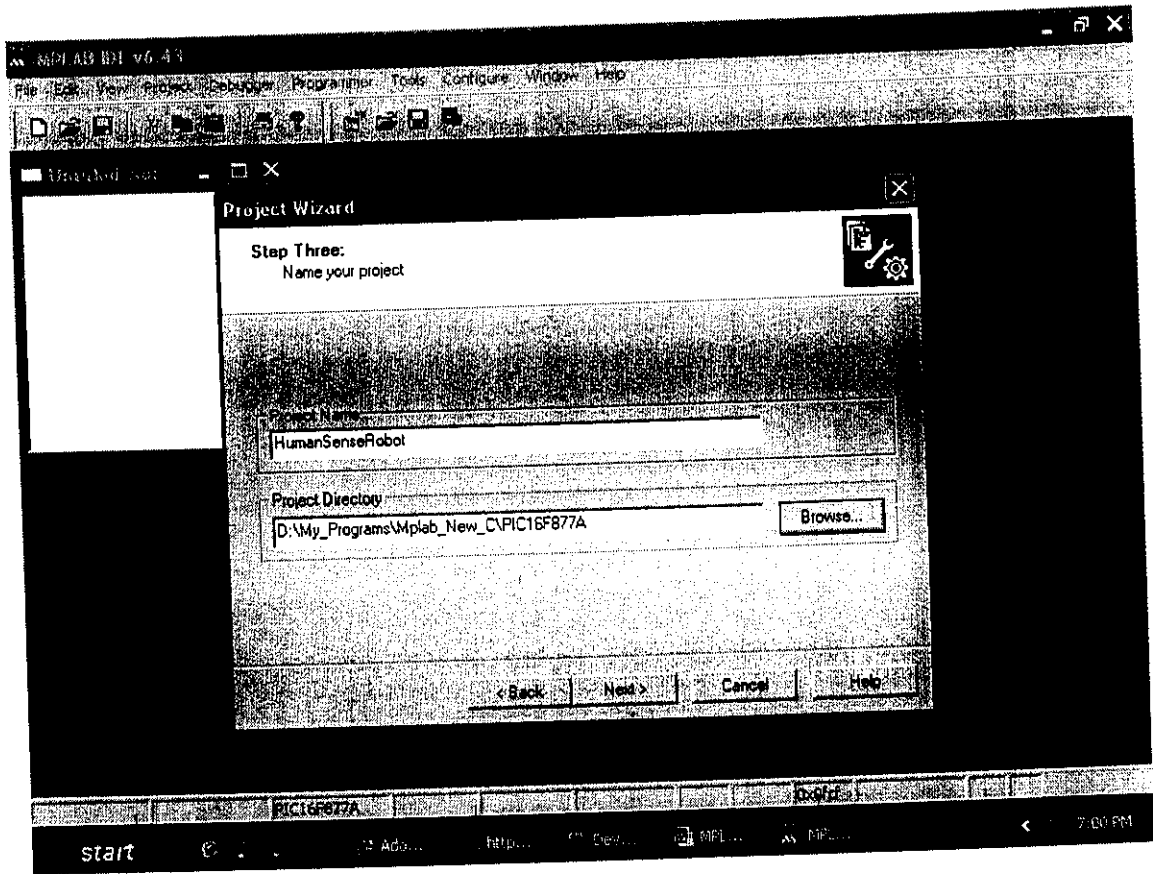
- Open the hex version of the source file from the directory by selecting **Open** command from the **File** menu.
- Choose **Program All** option from the **Command** menu. This will program the chip with the hex file of your project.
- Finally download the configuration settings for the device by selecting **Program**

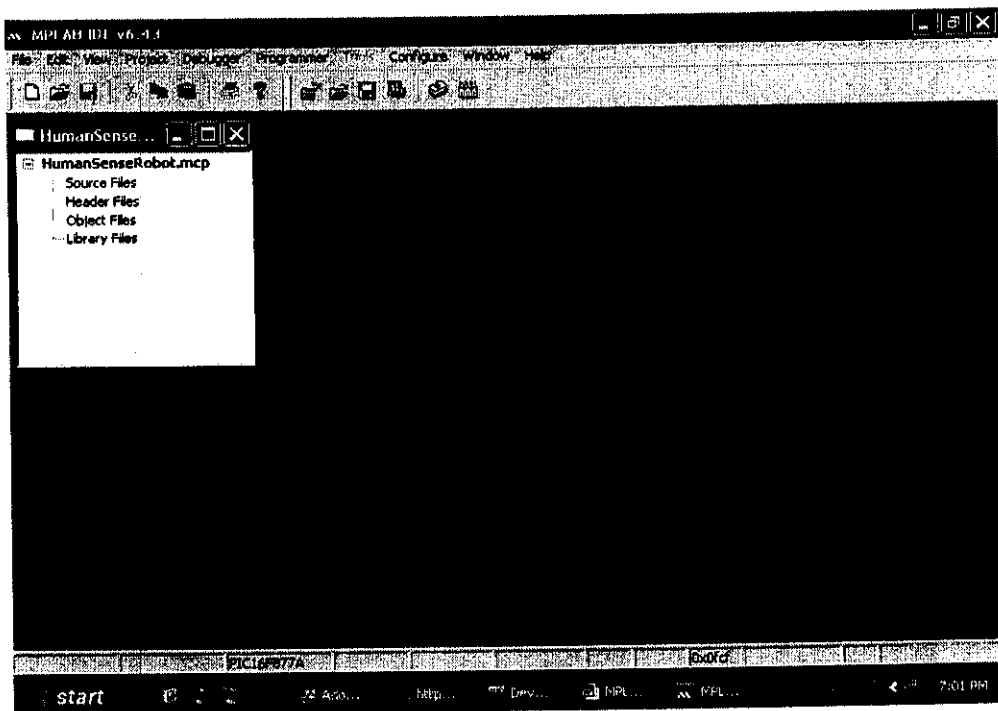
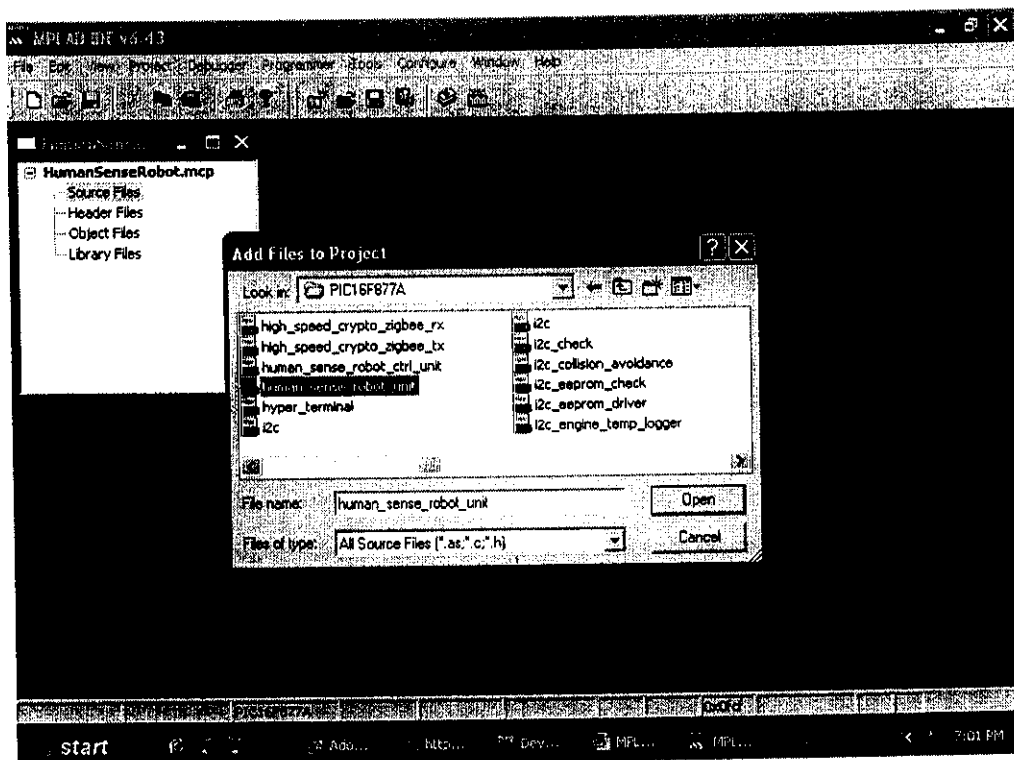
6.3 PHASES OF COMPILER

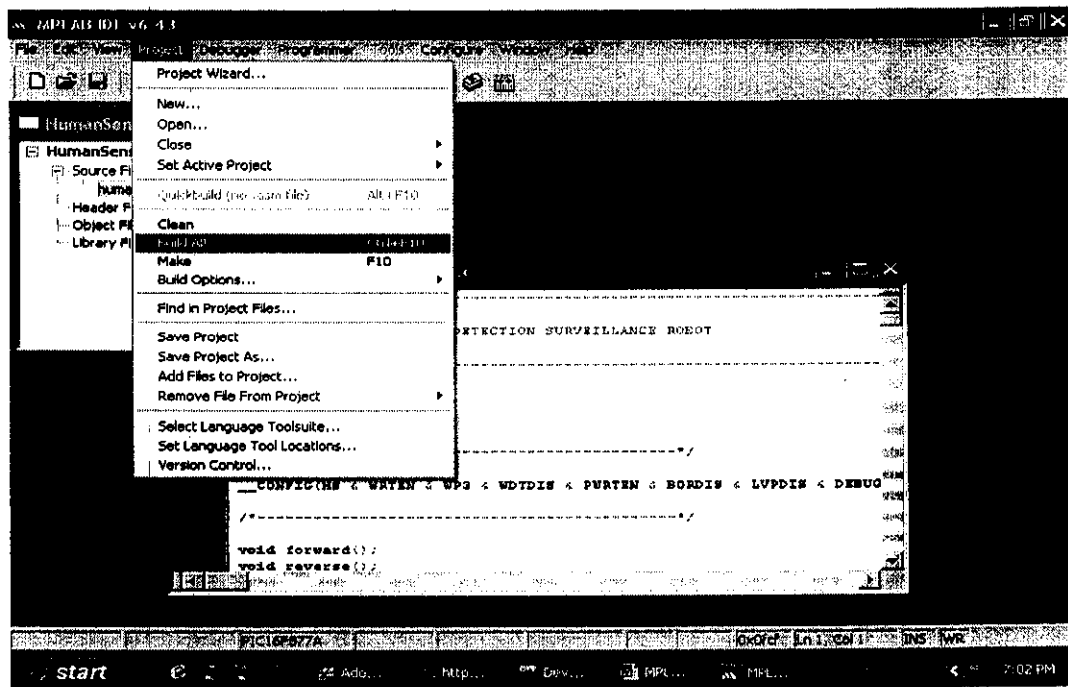
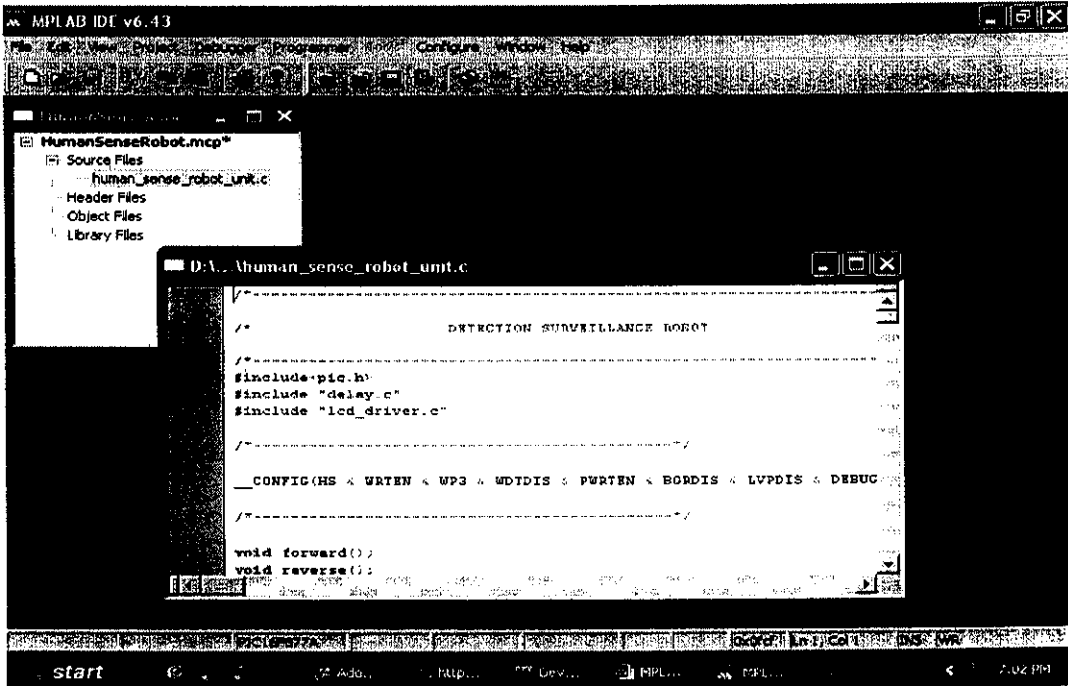


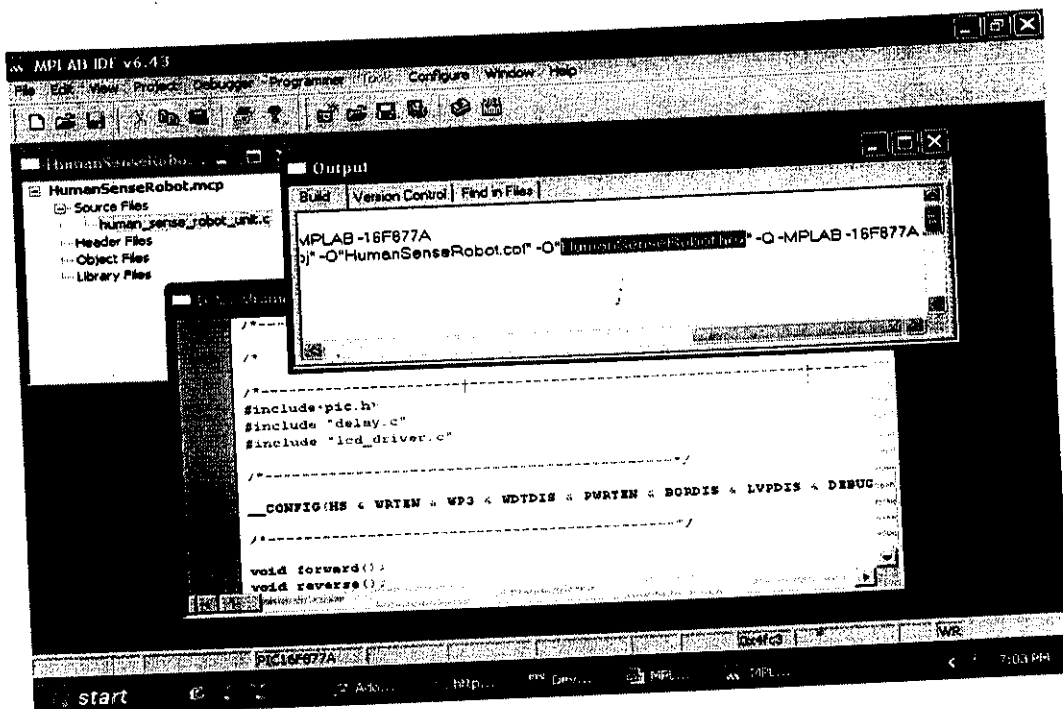
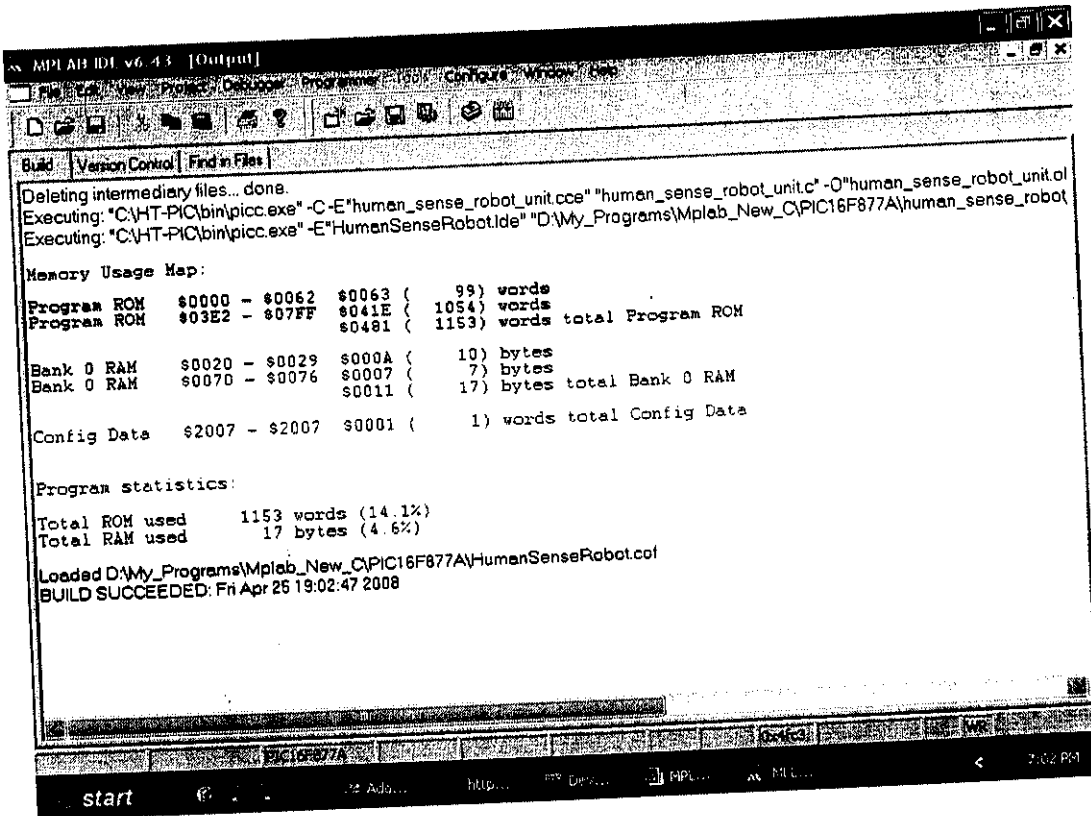


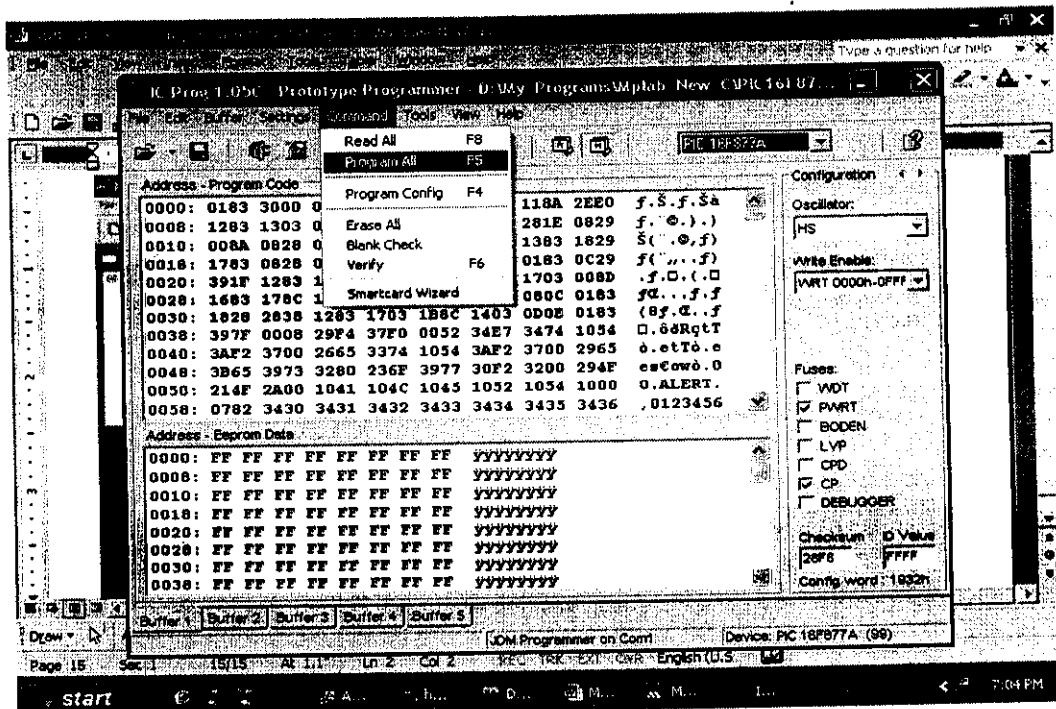
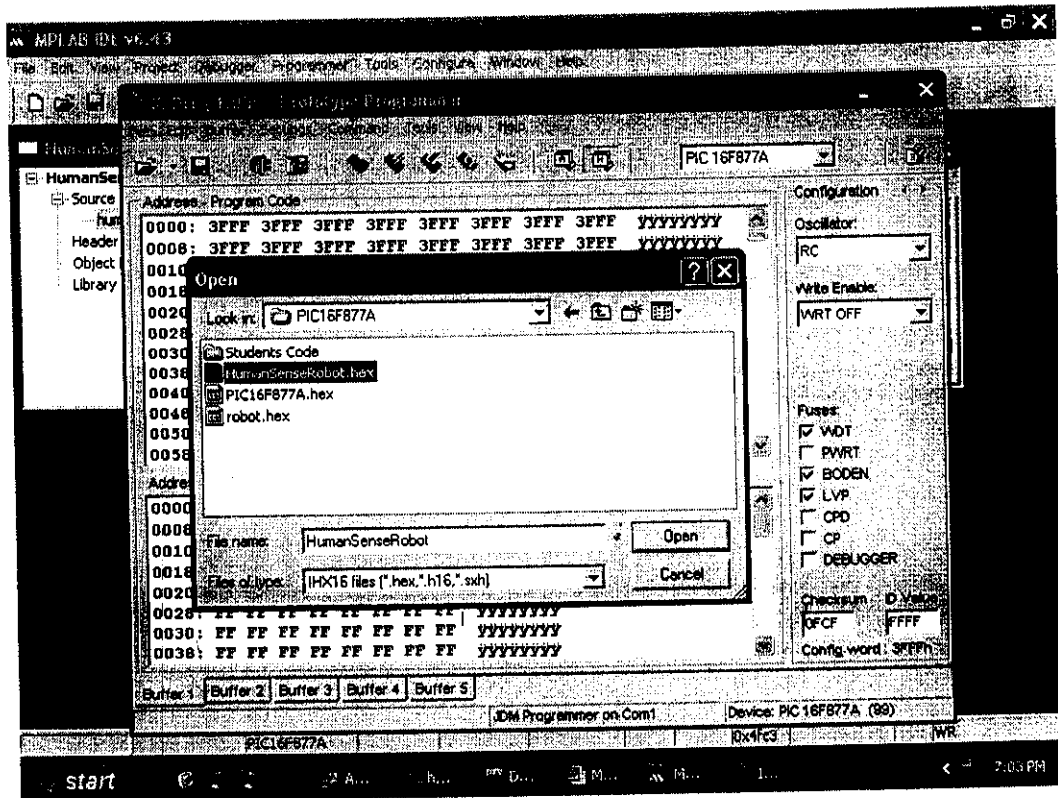












MPR AH - PICC Compilation Robot Project - Microsoft Word

Type a question for help

PIC16F877A

Address - Program Code

0000:	0183	3000	008A	2804	0183	120A	118A	2EED	f.S.f.Sa
0008:	1283	1303	00A8	1BA9	2816	1B29	281E	0829	f.(.f.)
0010:	008A	0828	0AA8	1903	0AA9	0082	1383	1829	S(.f.)
0018:	1783	0828	0AA8	0084	0800	0008	0183	0C29	f(.f.)
0020:	391F	1283	1703						
0028:	1693	178C	140C						
0030:	1828	2838	1283						
0038:	397F	0008	29F4						
0040:	3AF2	3700	2663						
0048:	3B63	3973	3280						
0050:	214F	2A00	1041						
0058:	0782	3430	3431						

Address - Eeprom Data

0000:	FF	FF	FF	FF	FF	FF	FF	FF	FFFFFFFF
0008:	FF	FF	FF	FF	FF	FF	FF	FF	FFFFFFFF
0010:	FF	FF	FF	FF	FF	FF	FF	FF	FFFFFFFF
0018:	FF	FF	FF	FF	FF	FF	FF	FF	FFFFFFFF
0020:	FF	FF	FF	FF	FF	FF	FF	FF	FFFFFFFF
0028:	FF	FF	FF	FF	FF	FF	FF	FF	FFFFFFFF
0030:	FF	FF	FF	FF	FF	FF	FF	FF	FFFFFFFF
0038:	FF	FF	FF	FF	FF	FF	FF	FF	FFFFFFFF

Device: PIC 16F877A

Programming Code (8192) bytes

Cancel

Configuration

Oscillator: HS

Write Enable: WRT 0000h-0FFF

Fuses:

- WDT
- PWRT
- BODEN
- LVP
- CP
- DEBUGGER

Checksum ID Value

28F6 FFFF

Config word: 1892h

Buffer 1 Buffer 2 Buffer 3 Buffer 4 Buffer 5

Programming JDM Programmer on Com1 Open File... Ctrl+O

Page 16 Sec 1 1616 AR 1.1 Ln 2 Col 2 REC TRK EAT OVR English (U.S.)

start

7:04 PM

CHAPTER VII

7.1 CONCLUSION:

In this Project, we propose a gesture recognition system based on a single tri-axis accelerometer mounted on a cell phone for human computer interaction. Three feature extract method, namely discrete cosine transform, Fast Fourier \ transform and combine wavelet packet decomposition with Fast Fourier transform are proposed. Classification of the gestures is performed with Support Vector Machine. Gesture recognition results are based on acceleration data collect from 67 subjects. The best recognize rate (87.36%) is achieved with wavelet-based method, while DCT coefficients and FFT coefficients produced accuracy of 85.16% and 86.92% respectively. Experimental results show that using DCT and FFT not only hold the primary information, but also reduce the dimensions of data. By using WPD, we can obtain decomposed signals that can efficiently represent the features of signal patterns. Gesture recognition based on single tri-axis accelerometer mounted on acell phone provides a novel human computer interaction.

7.2 ADVANTAGES:

- The Project uses inexpensive, easily deployable components.
- A simple setup that could be erected anywhere.
- There is always a provision for continuous improvement

7.3 ENHANCEMENTS

- It can be installed in mobile phones, so that visually challenged people can make calls easily.
- It can be enhanced to recognize characters.

CHAPTER VIII

8.1 REFERENCES:

[1] J.A. Flanagan and J. Mantyjarvi, "Unsupervised clustering of symbol strings and context recognition". ICDM, Maebashi, Japan. pp.171-178 .

[2] Eun-Seok Choi, Won-Chul Bang, et. al, "Beatbox Music Phone: Gesture Interactive Cell phone using Tri-axis Accelerometer," ICITIEEE Int. Conference on Industrial Technology, 2005.

[3] Sung-Jung Cho, Eunseok Choi, et. al., "Two-stage Recognition of Raw Acceleration Signals for 3-D Gesture- Understanding Cell Phones", 10th IWFHR, La Baule, France, Oct. 2006.

[4] Sung-Do Choi, A.S. Lee, Soo-Young Lee, "On-Line Handwritten Character Recognition with 3D Accelerometer", 2006 IEEE International Conference on Information Acquisition, pp.845-850, 2006

[5] S. Kallio, J. Kela and J. Mantyjarvi, "Online gesture recognition system for mobile interaction", 2003 IEEE International Conference on Systems, Man and Cybernetics, vol 3, pp.2070-2076.

[6] V. Vapnik. The nature of statistical learning theory, Springer Press, New York, 1999.

[7] N. Ahmed, T. Natarajan, and K. R. Rao. "Discrete Cosine Transform", IEEE Trans. on Computers, vol. 23, pp.90-93, Jan 1974.

[8] R. R. Coifman, Y. Meyer, and M. V. Wickerhauser, "Wavelet analysis and signal processing," in Wavelets and Their Applications, M. B. Ruskai, Ed. Boston: Jones and Bartlett, 1992.

[9] L. Deqiang, W. Pedrycz, and N. J. Pizzi, "Fuzzy wavelet packet based feature extraction method and its application to biomedical signal classification", IEEE Transactions on Biomedical Engineering, vol. 52, pp.1132-1139, 2005.