P - 2005

ELECTRODERMAL ACTIVITY METER

A PROJECT REPORT

Submitted by

M.ABARNA RAJESWARI	71203106001
P.REVATHY	71203106043
C.SUGANYA	71203106052
SUMAIAH SHAHEEM	71203106054

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

ELECTRONICS AND COMMUNICATION ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY,

COIMBATORE

ANNA UNIVERSITY:CHENNAI 600 025

APRIL 2007

BONAFIDE CERTIFICATE

Certified that this project report **"ELECTRODERMAL ACTIVITY METER"** is the bonafide work of "M.ABARNA RAJESWARI, P.REVATHY, C.SUGANYA, SUMAIAH SHAHEEM" who carried out the project work under my supervision.

SIGNATURE

Dr.(Mrs)RAJESWARI MARIAPPAN

HEAD OF THE DEPARTMENT,
ECE DEPARTMENT,
KUMARAGURU COLLEGE
OF TECHNOLOGY,
COIMBATORE- 641006.

SIGNATURE

Mrs. S. KAVITHA

SENIOR LECTURER,
ECE DEPARTMENT,
KUMARAGURU COLLEGE
OF TECHNOLOGY,
COIMBATORE-641006

The candidates with university register nos.71203106001, 71203106043, 71203106052, and 71203106054 were examined in the project viva-voce examination held on 23/4/07

INTERNAL EXAMINER

EXTERNAL EXAMINER

ABSTRACT

ABSTRACT

The objective of our project is to measure the magnitude of the electrical conductance in a person's skin which is directly correlated to their emotional state. The short term changes in electrical conductance also correlate with a person's mental response.

We implemented an Electro Dermal Activity (EDA) meter designed to measure skin conductance. Our project finds potential application in psychology, hydration measurements etc.

Skin conductance is measured using surface electrodes. The voltage across the skin is given to the Wheatstone bridge circuit. The output from the Wheatstone bridge is sent to the voltage follower. The signal obtained is then digitized using an inbuilt ADC and processed in a PIC microcontroller. The digital output is displayed in the LCD and the real time graph is plotted in PC.

ACKNOWLEDGEMENT

ACKNOWLEDGEMENT

We extend our sincere thanks to our Principal **Dr.Joseph V. Thanikal, M.E., Ph.D., PDF., CEPIT., Kumaraguru College Of Technology,Coimbatore** for his incredible support for all our toil regarding the project.

We are deeply obliged to **Dr. Rajeswari Marippan, Professor and Head of the Department of Electronics and Communication Engineering**, for her concern and implication during the project course.

We intend to express our heartiest thanks to **Prof. K.Ramprakash**, Project coordinator who have helped us to overcome the perplexity while choosing the project.

We articulate our thanks to our guide **Mrs.S.Kavitha**, Senior Lecturer who rendered her valuable guidance throughout our project path and support to perform our project work extremely well.

We are also thankful to all the faculty members and non teaching staffs of the department of Electronics and Communication Engineering for providing us the technical support for our project.

We express our humble gratitude and thanks to our parents and friends who have supported and helped us to complete the project fruitfully, support and cooperation throughout our project.

CERTIFICATE

CONTENTS

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF FIGURES

LIST OF TABLES

LIST OF TABLES

INTRODUCTION

1. 1 INTRODUCTION TO TOPIC

Every organ in human body has its own characteristics. Skin is an organ of the integumentary system made up of multiple layers of epithelial tissues that guard underline muscles and organs. It plays a most important role in protecting against pathogens. The other main functions are insulation and temperature regulation sensation and vitamin D and vitamin B synthesis. the skin surface has an area of 1.5 to 2 sqm. Most of it's between 2 to 3 mm thick.

Every organ in human body conducts voltage in the range of mv. Medical experiments have shown that the magnitude of the electrical conductance in a person's skin is directly correlated to their emotional state. The short term changes in electrical conductances also correlate with a person's mental response. We implemented an EDA meter designed to measure skin conductance.

Measurement of skin conductance is often used in psychology for quantifying a person's reaction to different stimulus. It is also often implemented in a traditional lie detectors. The current theory is that a person under arousal generates sweat at the skin that changes the skin's electrical properties.

In our project, Skin conductance is measured using surface electrodes. The voltage across the skin is given to the Wheatstone bridge circuit. The output from the Wheatstone bridge is sent to the voltage follower. The signal obtained is then digitized using an inbuilt ADC and processed in a PIC microcontroller. The digital output is displayed in the LCD and the real time graph is plotted in PC.

BLOCK DIAGRAM &CIRCUIT DIAGRAM

2. BLOCK DIAGRAM &CIRCUIT DIAGRAM

2.1 LOGICAL STRUCTURE

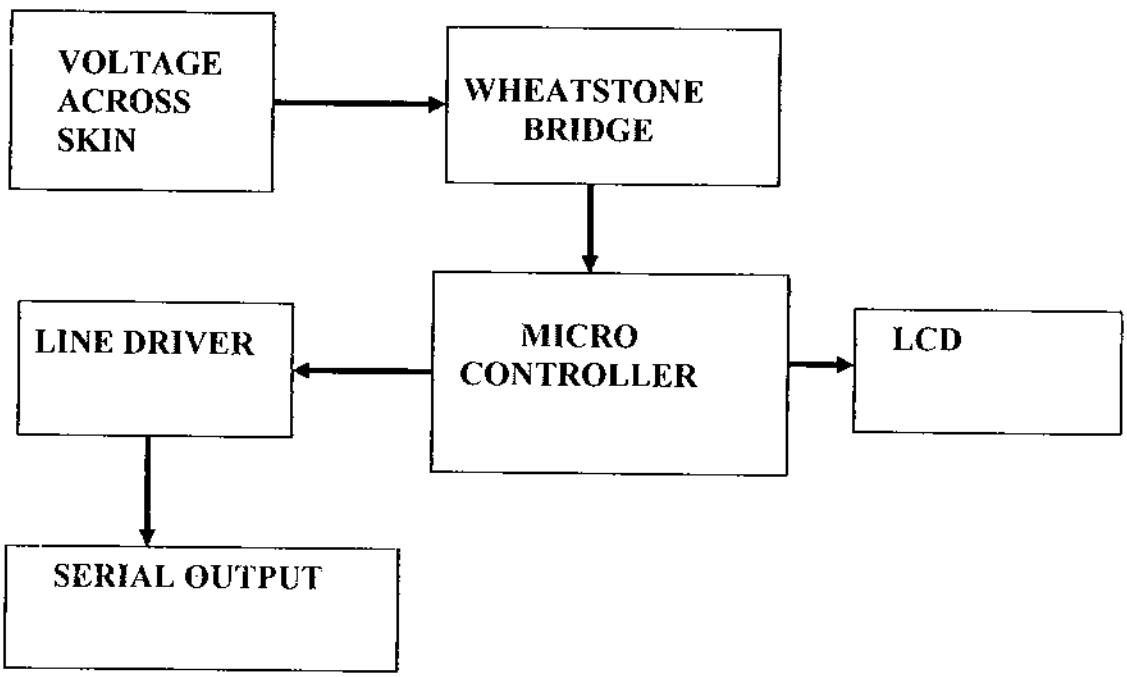
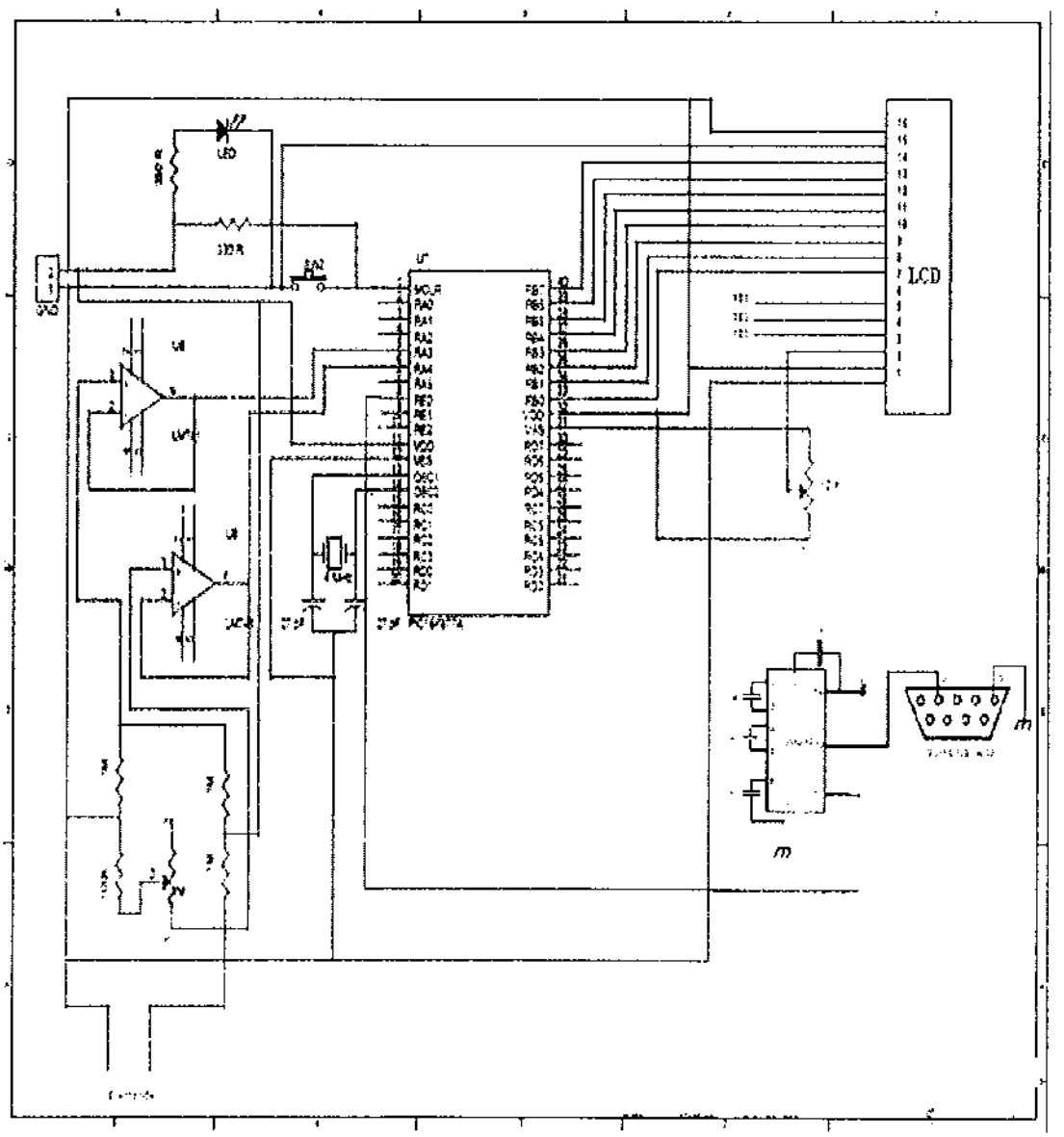


Fig 2.1 Logical structure

Our GSR design is based around the PIC microcontroller. Skin conductance reading are taken from a test subject and are given to the Wheatstone bridge circuit , and then given to the voltage follower. We send the signal to the PIC microcontroller through Port A2 and A3, the input to the ADC. We programmed the PIC for 10-bit ADC to increase the resolution of data. The data collected under the different modes are processed and output to the LCD. We also have a serial connection that can be connected to a computer. A C program reads from the serial port using a Line Driver

MAX 232. It then collects the skin conductance values and an output to a graph is generated.

2.2 CIRCUIT DIAGRAM:



2.2.1 COMPONENTS USED

- 5V Power Supply
- Copper Velcro Electrode
- Wheatstone Bridge
- Voltage follower
- PIC Microcontroller
- Liquid Crystal Display (LCD)
- Line Driver MAX 232
- RS 232 D-9 Connector

CIRCUIT DESCRIPTION OF ELECTRODERMAL ACTIVITY METER

3. CIRCUIT DESCRIPTION OF ELECTRODERMAL ACTIVITY METER

3.1 COPPER VELCRO ELECTRODE

It is popularly known as a lie detector, but is also used in Biofeedback conditioning. The theory is that; the more relaxed you are the dryer your skin is and so the higher the skin's electrical resistance. When you are under stress your hand sweats and then the resistance goes down.

3.1.1 DESIGN OF COPPER VELCRO ELECTRODE

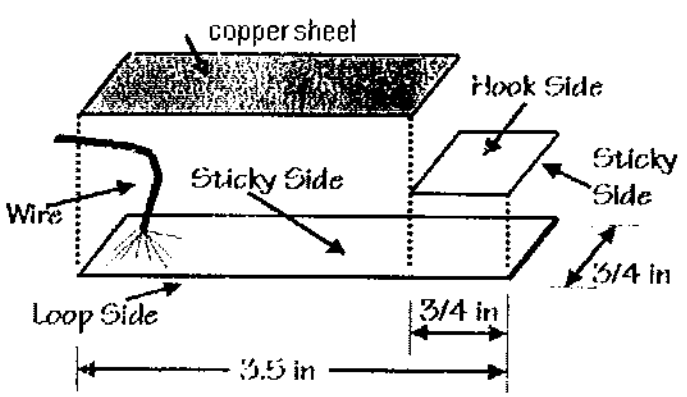


Fig 3.1.1 Electrode design

We used the Loop side of the Velcro (looks like carpet) for the part that wraps around the finger, and small square of the Hook side (stiffer plastic) to act as a catch. The stripped and frayed end of

the zero resistance wire is sandwiched between the Loop sticky side and the Aluminum Foil that covers the rest of the probe.

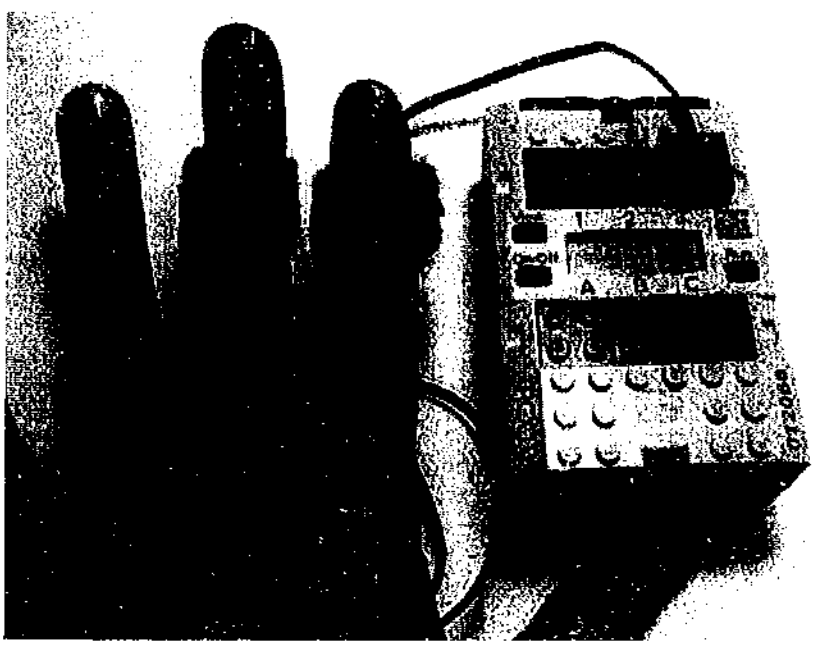


Fig 3.1.2 Electrode

3.2 WHEAT STONE BRIDGE

The Bridge has four resistive arms, a source of emf (a battery) and a null detector usually a galvanometer or other sensitive current meter. The current through the galvanometer depends on the potential difference between point c and d. The bridge is said to be balanced when the potential difference across the galvanometer is Zero volt so that there is no current through the galvanometer.

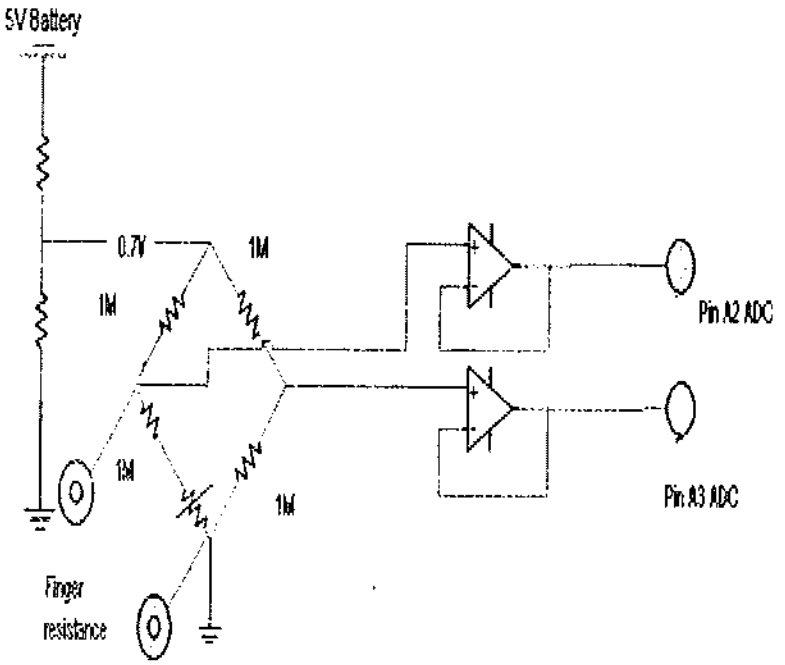


Fig 3.2.1 Wheatstone Bridge Circuit

3.3 VOLTAGE FOLLOWER

- Output voltage follows the input voltage
- Unity Gain amplifier
- Input impedance is very high (MΩ range)
- Output impedance is Zero
- It can be used as a buffer for impedance matching

3.4 INTRODUCTION TO PIC MICROCONTROLLER

3.4.1 PIN DIAGRAM OF PIC16F877:

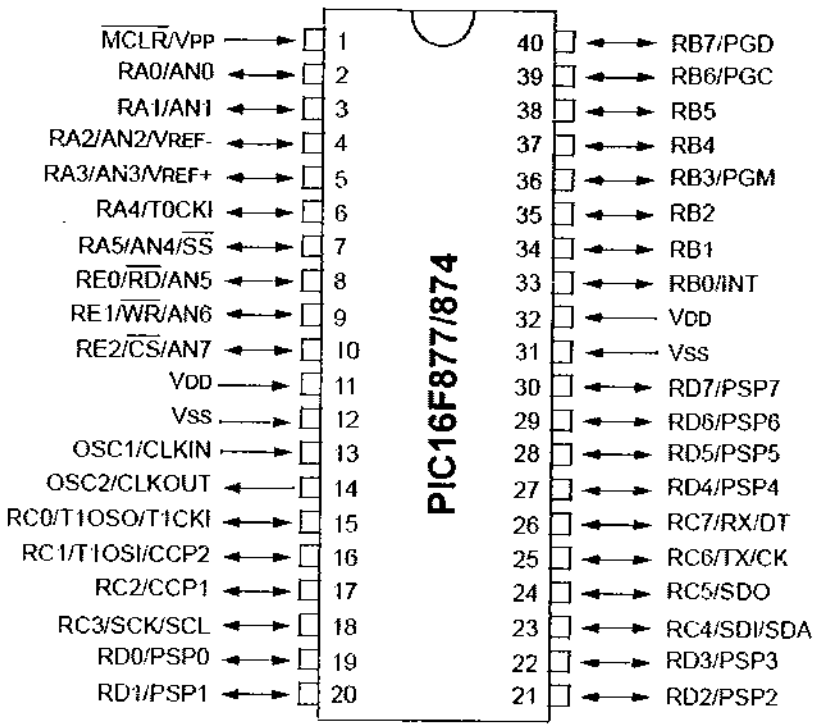


Fig 3.4.1 Pin diagram

P-2005

3.4.2 MICROCONTROLLER CORE FEATURES:

- High-performance RISC CPU
- Only 35single word instructions to learn
- All single cycle instructions except for program branches which are two cycle
- Operating speed: DC-20MHz clock input. DC 200 ns instruction cycle
- Up to 8k×14 words of Flash Program Memory,
  Up to 368×8Bytes of Data Memory (RAM)
  Up to 256×8Bytes of EEPROM Data Memory

- Pin out compatible to the PIC16373/74/76/77
- Interrupt capability (Up to 14 internal/external interrupt 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 EPROM/EEPROM technology
- Fully static design
- In-Circuit Debugging via 2 points
- Processor read/write access to program memory
- Wide operating voltage range:2.5V to 5.5v
- High sink/Source Current:2.5Ma
- Commercial and Industrial temperature ranges
- Low-power consumption:
    - < 2mA typical @ 5V,4MHz
    - 20mA typical @ 3V,32KHz
    - < 1mA typical standby current

3.4.3 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 12.5ns,
  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 (I^2)C (Master/Slave)
- Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI) 9-bit address detection
- Parallel Slave Port (PSP) 8-bits wide, with external RD, WR and CS controls (40/44-pin only)
- Brown-out detection circuitry for Brown-out Reset (BOR)

3.5 ANALOG TO DIGITAL CONVERSION (A/D)

The analog-to-digital (A/D) converter module has up to eight analog inputs. The A/D allows conversion of an analog input signal to a corresponding 8-bit digital number. The output of the sample and hold is the input into the converter, which generates the result via successive approximation. The analog reference voltage is software selectable to either the device's positive supply voltage (VDD) or the voltage level on the VREF pin. The A/D converter has a unique feature of being able to operate

while the device is in SLEEP mode. The A/D module has three registers. These registers are

1. A/D Result Register (ADRES)
2. A/D Control Register0 (ADCON0)
3. A/D Control Register1 (ADCON1)

The ADCON0 register controls the operation of the A/D module. The ADCON1 register configures the functions of the port pins. The I/O pins can be configured as analog inputs (one I/O can also be a voltage reference) or as digital I/O.Fig.3.5 shows the A/D block diagram.

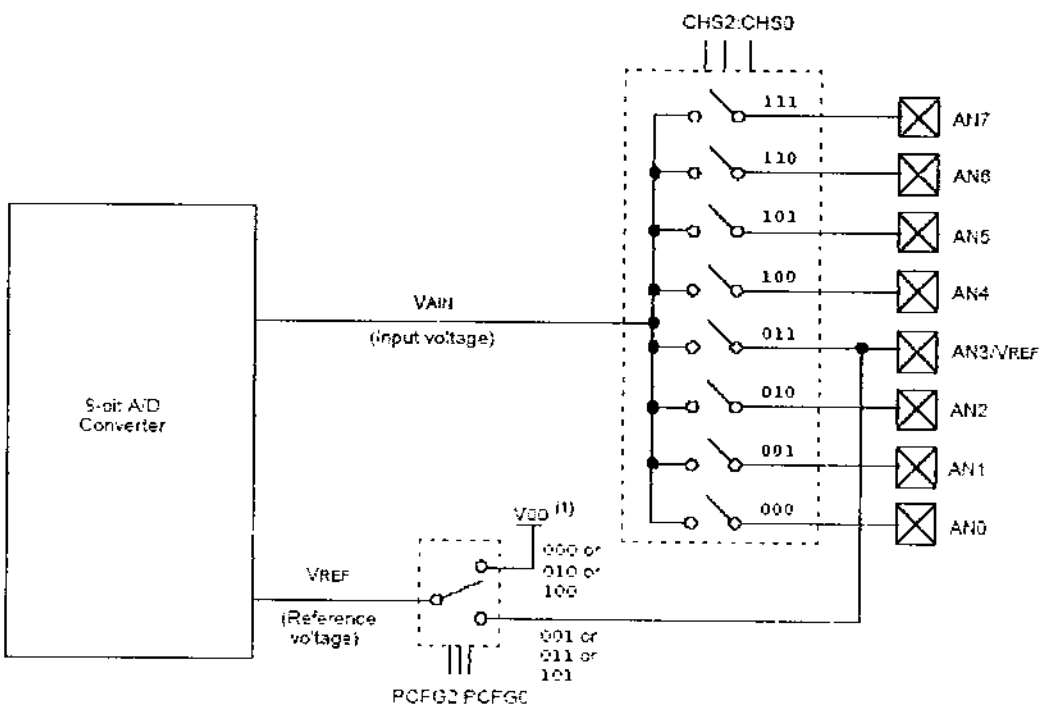


Fig 3.5 Block Diagram of A/D

3.6 UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECIVER TRANSMITTER (USART)

Micro controllers have also proven to be quite popular recently. Many of these have in built SCI (Serial Communications Interfaces), which can be used to talk to the outside world. Serial Communication reduces the pin count of these MPU's. Only two pins are commonly used, Transmit Data (TXD) and Receive Data (RXD) compared with at least 8 pins.

In this mode, the USART uses standard non-return-tozero (NRZ) format (one START bit, eight or nine data bits, and one STOP bit). The most common data format is 8-bits. An on-chip, dedicated, 8-bit baud rate generator can be used to derive standard baud rate frequencies from the oscillator. The USART transmits and receives the LSb first. The transmitter and receiver are functionally independent, but use the same data format and baud rate. The baud rate generator produces a clock, either x16 or x64 of the bit shift rate, depending on bit BRGH (TXSTA<2>). Parity is not supported by the hardware, but can be implemented in software (and stored as the ninth data bit). Asynchronous mode is stopped during SLEEP. Asynchronous mode is selected by clearing bit SYNC (TXSTA<4>).

The USART Asynchronous module consists of the following important elements:

- Baud Rate Generator
- Sampling Circuit
- Asynchronous Transmitter
- Asynchronous Receiver

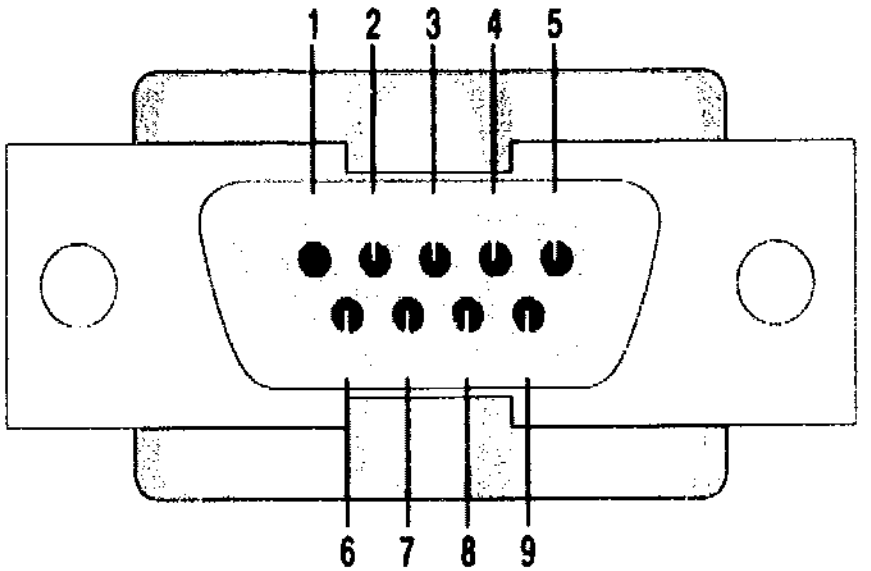
3.7. SERIAL PORT STANDARD

Master microcontroller to PC communication is established using RS232 protocol. RS232 is a serial protocol mainly used for PC communication. The maximum distance it can run is 15 meters.

3.7.1 RS 232 Connector Pin Assignment

RS 232 connector was originally developed to use 25 pins. In this, pin-out provisions were made for a secondary RS232 communication channel. Now a days, the smaller 9-pin version is most commonly used which is shown in the fig 3.7.1 and fig 3.7.2.

Fig 3.7.1 PC Com Port - EIA-574 RS-232/V.24 pin out on a DB-9 pin Used for Asynchronous Data



Pin	Signal	Pin	Signal
1	Data Carrier Detect	6	Data Set Ready
2	Received Data	7	Request to Send
3	Transmitted Data	8	Clear to Send
4	Data Terminal Ready	9	Ring Indicator
5	Signal Ground		

Fig: 3.7.2 Pin out diagram of RS 232 port

Electronic data communications between elements will generally fall into two broad categories, single-ended and differential. RS232 (single-ended) was introduced in 1962, and despite rumors for its early demise, has remained widely used through the industry.

Independent channels are established for two-way (full-duplex) communications. The RS232 signals are represented by voltage levels with respect to a system common (power / logic ground). The "idle" state (MARK) has the signal level negative with respect to common, and the "active" state (SPACE) has the signal level positive with respect to common. RS232 has numerous handshaking lines (primarily used with modems), and also specifies a communications protocol.

The RS-232 interface presupposes a common ground between the DTE and DCE. This is a reasonable assumption when a short cable connects the DTE to the DCE, but with longer lines and connections between devices that may be on different electrical busses with different grounds, this may not be true.

RS232 data is bi-polar.... +3 TO +12 volts indicate an "ON or 0-state (SPACE) condition" while A -3 to -12 volts indicates an "OFF" 1-state (MARK) condition.... Modern computer equipment ignores the negative level and accepts a zero voltage level as the "OFF" state. In fact, the "ON" state may be achieved with lesser positive potential. This means circuits powered by 5 VDC are capable of driving RS232 circuits directly; however, the overall range that the RS232 signal may be transmitted/received may be dramatically reduced.

The output signal level usually swings between +12V and -12V. The "dead area" between +3v and -3v is designed to absorb line noise. In the various RS-232-like definitions this dead area may vary. For instance, the definition for V.10 has a dead area from +0.3v to -0.3v. Many receivers designed for RS-232 are sensitive to differentials of 1v or less.

This can cause problems when using pin-powered widgets - line drivers, converters, modems etc. These types of units need enough voltage & current to power them self's up. Typical USART (the RS-232 I/O chip) allows up to 50ma per output pin - so if the device needs 70ma to run we would need to use at least 2 pins for power. Some devices are very efficient and only require one pin (some times the Transmit or DTR pin) to be high - in the "SPACE" state while idle.

An RS-232 port can supply only limited power to another device. The number of output lines, the type of interface driver IC, and the state of the output lines are important considerations. Data is transmitted and received on pins 2 and 3 respectively.

3.8 IC MAX 232

This IC is popularly known as a voltage shifter. It's a 16- pin IC used for the PC to microcontroller communication. This IC is connected between router and PC. Pin diagram of MAX 232 is shown in fig 3.8.

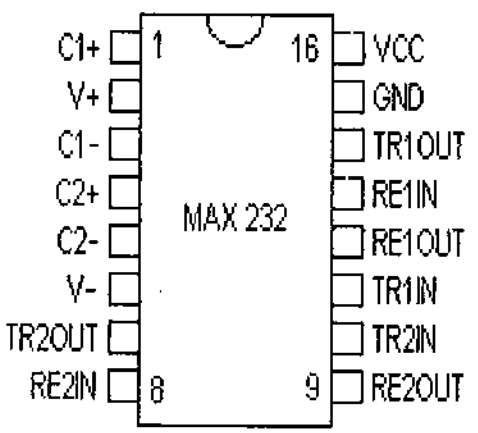


Fig 3.8.1 Pin diagram of MAX 232

The voltage level of microcontroller is 0 TO 5 Volts. In order to make communication between PC and microcontroller voltage conversion is essential. MAX 232 achieves this voltage conversion. This internally consists of diodes, logic gates, and capacitors. MAX 232 modifies the voltage level with respect to PC and vice versa. Fig.3.8.2 shows the logic diagram of MAX 232 IC.

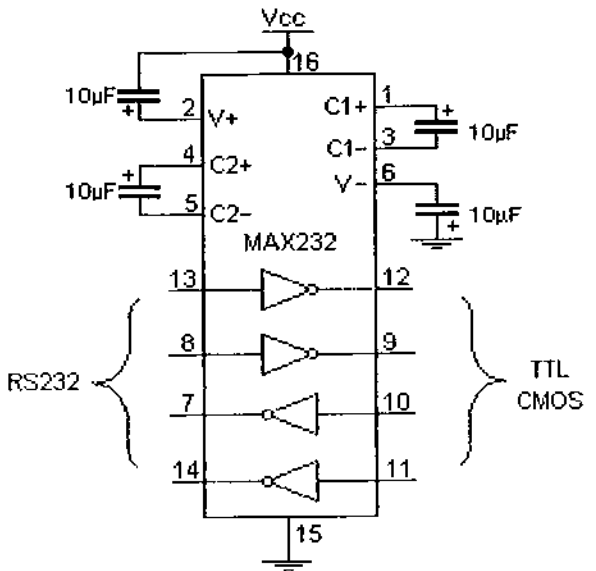


Fig 3.8.2 Logic diagram of MAX 232

3.9 LCD INTERFACING

The most common application of liquid crystal technology is in liquid crystal displays (LCDs). From the ubiquitous pocket calculator to an advanced VGA computer screen, this type of display has evolved into an important and versatile interface.

A liquid crystal display consists of an array of tiny segments (called pixels) that can be manipulated to present information. This basic idea is common to all displays, ranging from simple calculators to a full color LCD television.

3.9.1 STEPS TO BE FOLLOWED WHILE PROGAMMING

- Initializing the LCD.
- Clearing the LCD.
- Positioning the cursor.

- LCD
- Displaying a string.
- Displaying a screen.
- "Wiping" a screen clear, then "wiping" another on in its place.
- Displaying multiple pages of text.
- Using special user-defined characters.
- Implementing a menu.
- Flashing text.
- Centering text on a line.

3.9.2 LCD PIN DETAILS

PIN NUMBER	SYMBOL	LEVEL	I/O	FUNCTIONS
1	V_{SS}	-	-	POWER SUPPLY(GND)
2	V_{CC}	-	-	POWER SUPPLY(+5V)
3	V_{EE}	-	-	CONTRAST ADJUST
4	RS	0/1	I	0=INSTRUCTION INPUT 1=DATA INPUT
5	R/W	0/1	I	0=WRITE TO LCD MODULE 1=READ TO LCD MODULE
6	E	1,1->0	I	ENABLE SIGNAL
7	DB_0	0/1	I/O	DATA BUS LINE 0 (LSB)
8	DB_1	0/1	I/O	DATA BUS LINE 1

9	DB_2	0/1	I/O	DATA BUS LINE 2
10	DB_3	0/1	I/O	DATA BUS LINE 3
11	DB_4	0/1	I/O	DATA BUS LINE 4
12	DB_5	0/1	I/O	DATA BUS LINE 5
13	DB_6	0/1	I/O	DATA BUS LINE 6
14	DB_7	0/1	I/O	DATA BUS LINE 7

Table 3.9.2 Pin Description of an 14pin LCD

RELATIONSHIP OF DESIGN TO EXISTING STANDARDS

4.1 RELATIONSHIP OF DESIGN TO EXISTING STANDARDS

Since this device is connected to a human, we had to comply to the class safety standard that there should never be a connection from the human to the 120V power grid. Furthermore, we made sure that the electrode placement would never be an opposite sides of the body given that even small currents can cause fibrillation in the heart.

OUTPUT PLOT

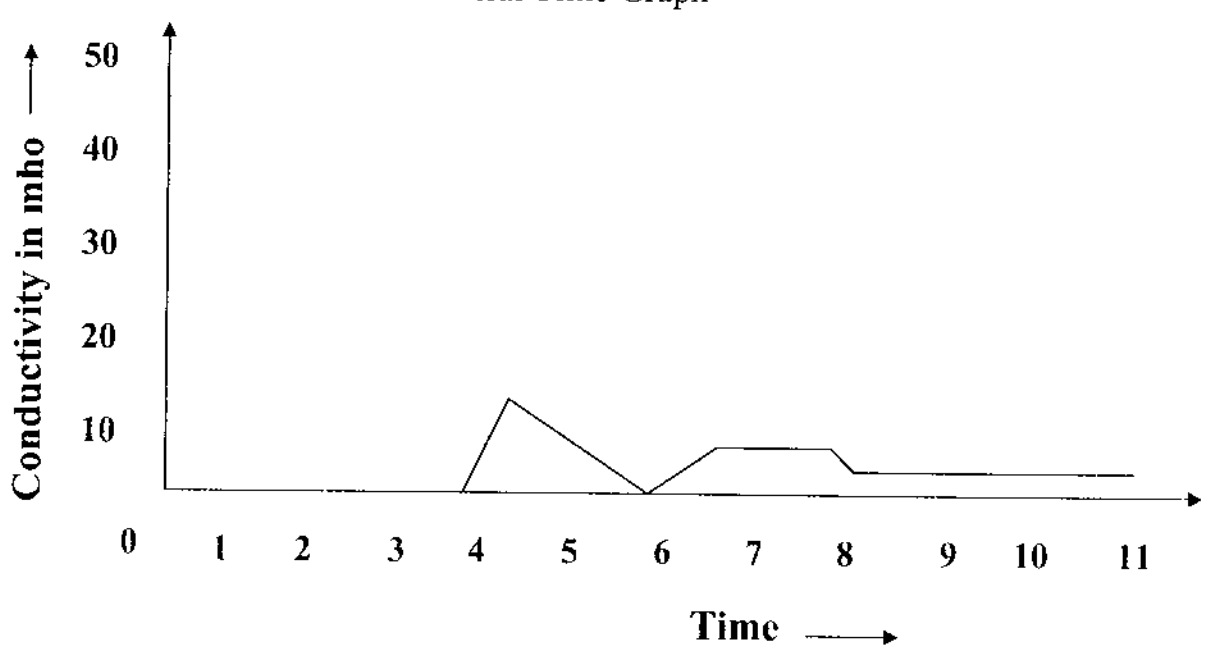
OUTPUT PLOT

EDA sensor

CONDUCTIVITY : 0.75 mho

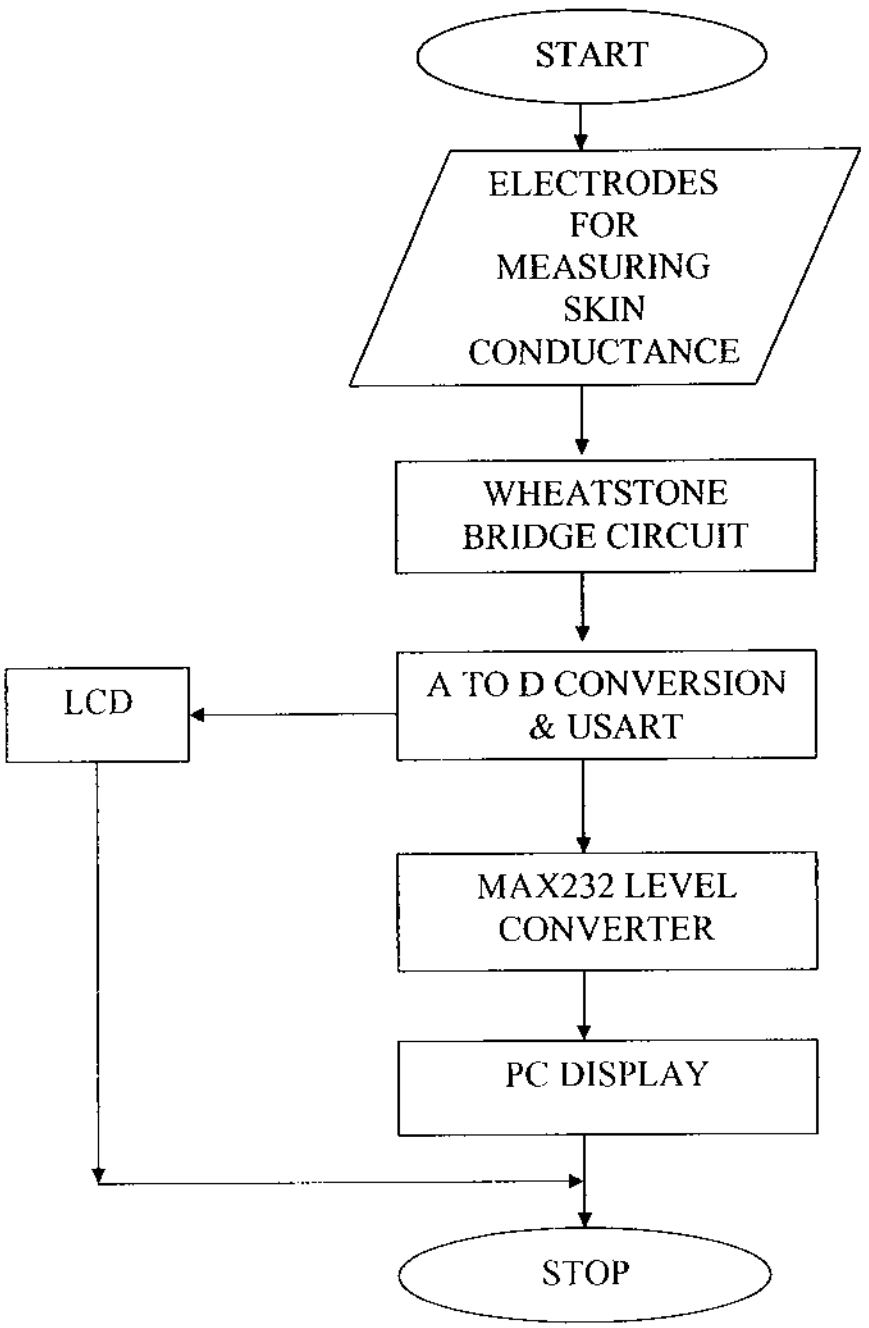
RESISTIVITY : 10.90 ohm

Real Time Graph



PROCESS FLOW CHART

FLOWCHART


START
ELECTRODES FOR MEASURING SKIN CONDUCTANCE
WHEATSTONE BRIDGE CIRCUIT
A TO D CONVERSION & USART
LCD
MAX232 LEVEL CONVERTER
PC DISPLAY
STOP

CONCLUSION

CONCLUSION

We conclude that the measurement of skin conductance gives the emotional status of the person. It finds application in Psychology, Hydration measurements and Lie detection in its advanced form.

Electro dermal activity meter, with its high lability, freely varying activity and ability to map the orienting response has shown to be promising in measurement of direct mental interaction with living systems.

APPENDIX

APPENDIX – I

PROGRAMMING

CONTROLLER CODE USING MP CODE

```
#include<htc.h>
#include<stdio.h>
#include<string.h>
#include<math.h>

void configLCD();
void commandLCD();
void dataLCD();
void clearDisp();
void writeLCD(char ch,unsigned char r,unsigned char c);
void writeStrLCD(const char str[17],char l,char c);

void initSerial();
void sendByte(unsigned char c);

void initADC();
void selChannel(char chan);

void delayuS(int n);
void delaymS(int n);

void selBank(char n);

bit ADConvFlag = 0;
bit recFlag = 0;

unsigned char SampleCount = 0;
unsigned char rdat = 0;

bank1 unsigned char buffer[25] = "";
bank1 unsigned char buffer1[25] = "";

unsigned char Channel = 0;
```

```
unsigned char i = 0,len = 0;

unsigned int cdata = 0,pdata = 0;
unsigned int temp = 0;

signed int Chan2 = 0, Chan3 = 0,Diff = 0;

signed long R = 0, Rx = 0;

float fDiff = 0.0f;

void main()
{
	initADC();
	initSerial();
	selBank(0);
	configLCD();
	clearDisp();
	selChannel(2);
	Channel = 2;
	delaymS(50);

	writeStrLCD("Test",0,1);

	while(1)
	{
		ADCON0 = 0x05;

		if(ADConvFlag)
		{
			if(Channel == 2)
			{
				Chan2 = cdata;
				Chan2 = (long) Chan2 * 500/1023;

				sprintf(buffer,"A2: %d%c%d ",
Chan2/100,'.',Chan2%100);
				writeStrLCD(buffer,0,0);

				selChannel(3);
```

```
                Channel = 3;
                delaymS(50);
            }

            else if(Channel == 3)
            {
                Chan3 = cdata;
                Chan3 = (long) Chan3 * 500/1023;
                sprintf(buffer,"A3: %d%c%d ",
Chan3/100,'.',Chan3 %100);
                writeStrLCD(buffer,1,0);
                selChannel(2);
                Channel = 2;
                delaymS(50);
            }

            delaymS(50);
            ADConvFlag = 0;

        }

        Diff =  (int) Chan3 - Chan2;

        if(Diff < 0)
        {
            Diff = Diff * (-1);
            sprintf(buffer,"D:-%d%c",(int) Diff/100,'.');

            if((Diff % 100) < 10)
            {
                strcat(buffer,"0");
                sprintf(buffer1,"%d",(int) Diff%100);
                strcat(buffer,buffer1);
            }
            else
            {
                sprintf(buffer1,"%d",(int) Diff%100);
                strcat(buffer,buffer1);
            }
        }
```

```
else
{
        sprintf(buffer,"D:%d%c",(int) Diff/100,'.');
        if((Diff % 100) < 10)
        {
                strcat(buffer,"0");
                sprintf(buffer1,"%d",(int) Diff%100);
                strcat(buffer,buffer1);

        }
        else
        {
                sprintf(buffer1,"%d",(int) Diff%100);
                strcat(buffer,buffer1);
        }
}

writeStrLCD(buffer,0,9);

if(recFlag)
{
        if(rdat == 'R')
        {
                sendByte(0xFF);
                sendByte((Chan3 >> 8) & 0xFF);
                sendByte(Chan3 & 0xFF);

                //sendByte(0xFF);
                sendByte((Chan2 >> 8) & 0xFF);
                sendByte(Chan2 & 0xFF);

                /*TXREG = 0xFF;
                TXEN = 1;
                asm("nop");
                asm("nop");

                TXREG = Diff >> 8;
                TXEN = 1;
                asm("nop");
                asm("nop");
```

```
                        TXREG = Diff & 0xFF;
                        TXEN = 1;
                        asm("nop");
                        asm("nop");*/

                    }

                    recFlag = 0;
                }

        }

}

void interrupt interISR()
{
        if(ADIF)
        {
                IRP = 0;
                RP1 = 0;
                RP0 = 0;

                ADIF = 0;

                while(ADCON0 & 0x04)
                {
                        NOP();
                }

                //SampleCount = 1;

                ADCON0 = 0x01;
                temp = ADRESH;
                temp = temp << 8;

                IRP = 0;
                RP1 = 0;
                RP0 = 1;
```

```
        cdata = ADRESL;
        cdata = cdata | temp;

        cdata = cdata & 0x3FF;

        //ADConvFlag = 1;

        if(pdata == cdata)
        {
                SampleCount++;
        }
        else
        {
                pdata = cdata;
                SampleCount = 0;
        }

        if(SampleCount > 6)
        {
                ADConvFlag = 1;
                SampleCount = 0;
        }
        else
        {
                ADCON0 = 0x05;
        }
}

if(TXIF)
{
        while(!TRMT);
        TXEN = 0;
        TXIF = 0;
}

if(RCIF)
{
        rdat = RCREG;
        recFlag = 1;
        RCIF = 0;
```

```
        }

}

void configLCD()
{
        selBank(1);
        //ADCON1 = 0x02;        // On not using ADC it is must to configure
ADCON1

        TRISB = 0x00;
        TRISE = 0x00;

        selBank(0);

        PORTB = 0x38;
        commandLCD();
        delaymS(60);

        PORTB = 0x0C;
        commandLCD();
        delaymS(60);

        PORTB = 0x06;
        commandLCD();
        delaymS(60);

        PORTB = 0x01;
        commandLCD();
        delaymS(60);

        PORTB = 0x80;
        commandLCD();
        delaymS(60);
}

void commandLCD()
{
        selBank(0);
        PORTE = 0x04;
```

```
        delaymS(1);
        PORTE = 0x00;
}

void dataLCD()
{
        selBank(0);
        PORTE = 0x05;
        delaymS(1);
        PORTE = 0x01;
}

void clearDisp()
{
        PORTB = 0x01;
        commandLCD();
        delaymS(60);
}

void writeLCD(char ch,unsigned char r,unsigned char c)
{
        unsigned char ad = 0;

        r = r % 2;
        c = c % 0x10;

        switch(r)
        {
                case 0:
                        ad = 0x80;
                break;

                case 1:
                        ad = 0xC0;
                break;
        }

        ad = ad + c;

        PORTB = ad & 0xFF;
```

```
        commandLCD();
        delaymS(60);

        PORTB = ch & 0xFF;
        dataLCD();
        delaymS(60);

}

void writeStrLCD(const char str[17],char l,char c)
{
        char i = 0;

        while(str[i])
        {
                writeLCD(str[i],l,c + i);
                i++;
        }
}

void initSerial()
{
        selBank(1);
        GIE = 1;
        PEIE = 1;

        TXIE = 1;
        RCIE = 1;

        TXSTA = 0x04;
        SPBRG = 0x19;    // 0x81 -- 9600 for 20 MHz
                                // 0x4D -- 9600 for 12 MHz
                                // 0x19 -- 9600 for  4 MHz

        selBank(0);
        RCSTA = 0x90;
}

void sendByte(unsigned char c)
{
```

```
        TXREG = c;
        TXEN = 1;

        asm("NOP");
        asm("NOP");
}

void initADC()
{
        selBank(1);
        TRISA = 0x1F;
        INTCON = 0xc0;
        PIE1 = 0x40;
        selBank(0);
        ADRESH = 0x00;
        ADCON0 = 0x01;
        selBank(1);
        ADCON1 = 0x82;
        ADRESL = 0x00;
}

void selChannel(char chan)
{
        selBank(0);

        CHS0 = chan & 0x01;
        chan = chan >> 1;
        CHS1 = chan & 0x01;
        chan = chan >> 1;
        CHS2 = chan & 0x01;
}

void delayuS(int n)
{
        while(n > 0)
        {
                n--;
        }
}
```

```
void delaymS(int n)
{
	while(n > 0)
	{
		n--;
		delayuS(54);		// for 4 MHz XT
		//delayuS(164);		// for 12 MHz HS
	}
}

void selBank(char n)
{
	switch(n)
	{
		case 0:
			IRP = 0;
			RP1 = 0;
			RP0 = 0;
		break;

		case 1:
			IRP = 0;
			RP1 = 0;
			RP0 = 1;
		break;

		case 2:
			IRP = 1;
			RP1 = 1;
			RP0 = 0;
		break;

		case 3:
			IRP = 1;
			RP1 = 1;
			RP0 = 1;
		break;
	}
}
```

C PROGRAM

```
#include <dos.h>
#include <stdio.h>
#include <conio.h>
#include <math.h>
#include <graphics.h>
#include <stdlib.h>


#define PORT1 0x3F8  /* Port Address Goes Here */
#define INTVECT 0x0C /* Com Port's IRQ here (Must also change PIC
setting) */

unsigned int bufferin = 0;
float ss = 0;
unsigned int count = 0, bufferout = 0;
unsigned char bufff[22],*ch;
unsigned char buffer[1025];
unsigned char mani1[34],flag = 0;
unsigned char in [25];
unsigned int t1=0,m1=0,l1=0,l2=0;
float RR = 0.00f,RX = 0.00f,CY = 0.00f;
int s1= 0;
float s11=0,t11=0,m11=0,l11=0,l21=0;
int gd = DETECT, gm;
unsigned int n,k=19,t=0,m,n1,*sample;
unsigned char mani[12];
char mani2[15],mani3[15];
float sum0 = 0, sum1 = 0, sum2 = 0,sum0sq = 0,sum1sq = 0,sum2sq =
0,dxdy = 0,dydz = 0, dxsq = 0,dysq = 0,dzsq = 0;

float
a[75]={0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0
.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0
.0};

float s11;
```

```
void interrupt (*oldport1isr)();

char receive();

void interrupt PORT1INT()  /* Interrupt Service Routine (ISR) for PORT1
*/
{
        int c ;
        count = 0;
        flag = 1 ;
        do
        {
                c = inportb(PORT1 + 5);

                if (c & 1)
                {
                        buffer[count++] = inportb(PORT1);
                }

                delay(6);

        } while (c & 1);
        buffer[count] = '\0';

        outportb(0x20,0x20);
}

void config_port1()
{
        outportb(PORT1 + 1 , 0);
        oldport1isr = getvect(INTVECT);
        setvect(INTVECT, PORT1INT);
        outportb(PORT1 + 3 , 0x80);
        outportb(PORT1 + 0 , 0x0C);
        outportb(PORT1 + 1 , 0x00);
        outportb(PORT1 + 3 , 0x03);
        outportb(PORT1 + 2 , 0xC7);
        outportb(PORT1 + 4 , 0x0B);
        outportb(0x21,(inportb(0x21) & 0xEF));
```

```
        outportb(PORT1 + 1 , 0x01);

}

app_stres(float *a,int d1)
{
        int i;
        static int ll1=59;
        k=59;

        for(i=0;i<k;i++)
        {
                a[i]=a[i+1];
        }
        a[k]=d1;
        ll1--;
        return 1;
}

void stres(float *a,float d1)
{
        int i;
        n=59;
        n1=59;

        setcolor(0);
        moveto(110,390);
        line(110,390,110,(390-(2*a[0])));
        moveto(110,(390-(2*a[0])));

        for(t=((520-110)/n1),i=1;i<=n;i++)
        {
                lineto(110+t,(390-(2*a[i])));
                t+=((520-110)/n1);

        }

        app_stres(a,d1);

        settextstyle(2,0,4);
```

```
        setcolor(14);


        line(110,390,497,390);
        line(110,390,110,(390-150));        // Graph axis Design
        outtextxy(450,410,"TIME ->");

        settextstyle(2,1,4);
//      outtextxy(85,240,"VOLTAGE ->");
        settextstyle(2,0,5);
        outtextxy(270,205,"Real Time Graph");
        settextstyle(2,0,4);
        moveto(110,390);
        moveto(110,(390-(2*a[0])));
        for(t=((520-110)/n1),i=1;i<=n;i++)
        {
                lineto(110+t,(390-(2*a[i])));
                t+=((520-110)/n1);
        }
        //      delay  (150);
}               .


void main(void)
{
        int i,j;

        config_port1();
        initgraph(&gd,&gm,"");
        clearviewport();
        setmode(1,1);
        setcolor(WHITE);
        rectangle(7,2,getmaxx()-17,getmaxy()-17);
        rectangle(10,5,getmaxx()-20,getmaxy()-20);

//      line((getmaxx()+20)/2,5,(getmaxx()+20)/2,getmaxy()-20);
//      line(10,(getmaxy()+20)/3,getmaxx()-20,(getmaxy()+20)/3);

        setcolor(14);
        settextstyle(2,0,4);
```

```
//      settextstyle(0,0,0);
        settextstyle(2,0,5);
        outtextxy(270,205,"Real Time Graph");
        settextstyle(2,0,4);
        settextstyle(1,0,0);            // Mind switch title Design
        setcolor(9);
        outtextxy(235,24,"EDA sensor");
        setcolor(15);
        outtextxy(236,25,"EDA sensor");
        setcolor(9);
        outtextxy(237,26,"EDA sensor");

        settextstyle(2,0,4);
        setcolor(14);

        // 1st graph axis Design
        settextstyle(2,0,4);
        setcolor(14);


        line(110,390,497,390);
        line(110,390,110,(390-150));      // Graph axis Design
        outtextxy(450,410,"TIME ->");

        settextstyle(2,1,4);
        outtextxy(78,255,"Conductivity in MHO ->");
        settextstyle(2,0,4);


        for(j=0,i=0;j<13;i+=30,j++)
        {
                sprintf(ch,"%d",j);         // Co-ordinates Design
                outtextxy(110+i,393,ch);
        }

        for(j=10,i=30;j<=50;i+=28,j+=10)
        {
                sprintf(ch,"%d",j);
                outtextxy(94,390-i,ch);
        }
```

```
        settextstyle(0,0,1);
        outtextxy(107,235,"^");
        settextstyle(2,0,4);
        outtextxy(498,384,">");
        setcolor(14);
        while(!kbhit())
        {

                outportb(PORT1,'R');
//              while(flag != 1);
                delay(50);
                if(flag == 1)
                {
                        flag = 0;
                        if(count == 5)
                        {
                                static int s2 = 0;
                                float ss,sss,diff = 0.00;
                                unsigned char aa[20] = {0};
                                count = 0;

                                s1 = buffer[1] & 0xff;
                                s1 = (s1 << 8) & 0xff00;
                                s1 = s1 | (buffer[2] & 0xff);

                                s2 = buffer[3] & 0xff;
                                s2 = (s2 << 8) & 0xff00;
                                s2 = s2 | (buffer[4] & 0xff);

                                diff =(float) (s1-s2) / 100.00f ;


//                              s11 = ( (float) (s1 * 5.0f) / 1023.01f);

                        //      s11 = ( (float) s1 / 100.0f);

                                settextstyle(2,0,5);
                                setcolor(14);
                                outtextxy(180,120,"CONDUCTITITY : ");
                                outtextxy(180,140,"RESISITIVITY : ");
```

```
                        if(diff < 1)
                        {
                        RR =  (float) ( (2.0f * diff) + 5.0f) / (5.0f - diff) ;

                        if(RR != 1)
                                RX = RR / (1.2 - RR);

                        CY = (1.0f / RX) * 35;

                        if( (CY >= -5.0) && (CY  <= 50.0f))
                        {
                                setcolor(0);
                                sprintf(aa,"%2.2f ohm",sss);
                                outtextxy(290,120,bufff);
                                outtextxy(290,140,aa);
                                setcolor(14);
                                sprintf(bufff,"%2.2f mho",(float)(CY/10));
                                sprintf(aa,"%2.2f ohm",RX);
                                outtextxy(290,120,bufff);
                                outtextxy(290,140,aa);
                                ss= CY;
                                sss = RX;
                        }

                        strcpy(buffer,"");
                        if( (CY >= -5.0) && (CY  <= 50.0f))
                        {
                                settextstyle(2,0,5);
                                stres(a,CY);
                        }
                        }
                }
        }
    }
    outportb(PORT1 + 1 , 0);        /* Turn off interrupts - Port1 */
    outportb(0x21,(inportb(0x21) | 0x10));  /* MASK IRQ using PIC */

    setvect(INTVECT, oldport1isr); /* Restore old interrupt vector */
    closegraph();
}
```

APPENDIX-II DATA SHEETS

MICROCHIP

PIC16F87X

28/40-Pin 8-Bit CMOS FLASH Microcontrollers

Devices Included in this Data Sheet:

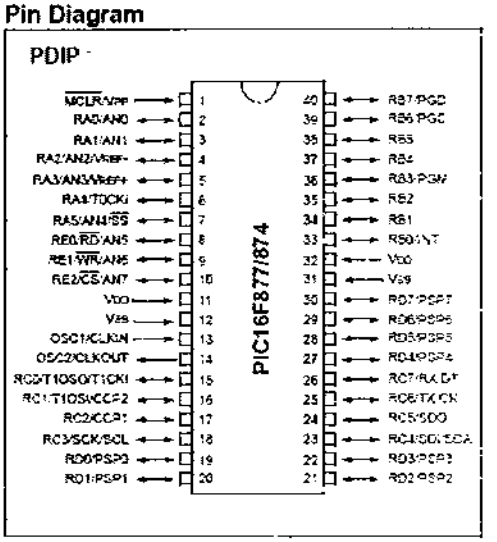
- PIC16F873
- PIC16F874
- PIC16F876
- PIC16F877

Microcontroller Core Features:

- High performance RISC CPU
- Only 35 single word instructions to learn
- 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
- Pinout compatible to the PIC16C73B/74B/76/77
- 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, Industrial and Extended temperature ranges
- Low-power consumption:
  - < 0.6 mA typical @ 3V, 4 MHz
  - 20 µA typical @ 3V, 32 kHz
  - < 1 µA typical standby current

Pin Diagram

PDIP

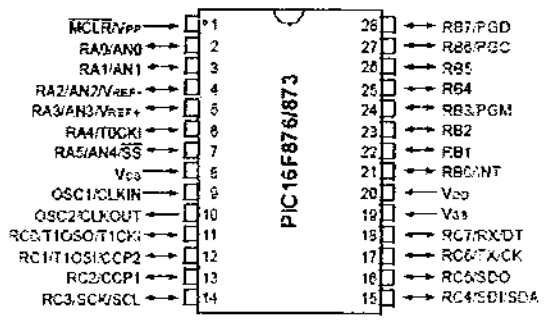


Peripheral Features:

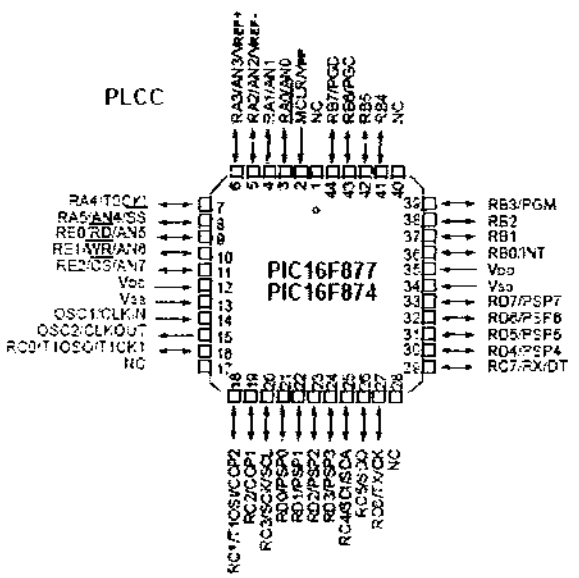
- Timer0: 8-bit timer/counter with 8-bit prescaler
- Timer1: 16-bit timer/counter with prescaler, can be incremented during SLEEP via external crystal/clock
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- Two Capture, Compare, PWM modules
  - Capture is 16-bit, max. 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)
- Brown-out detection circuitry for Brown-out Reset (BOR)

Pin Diagrams

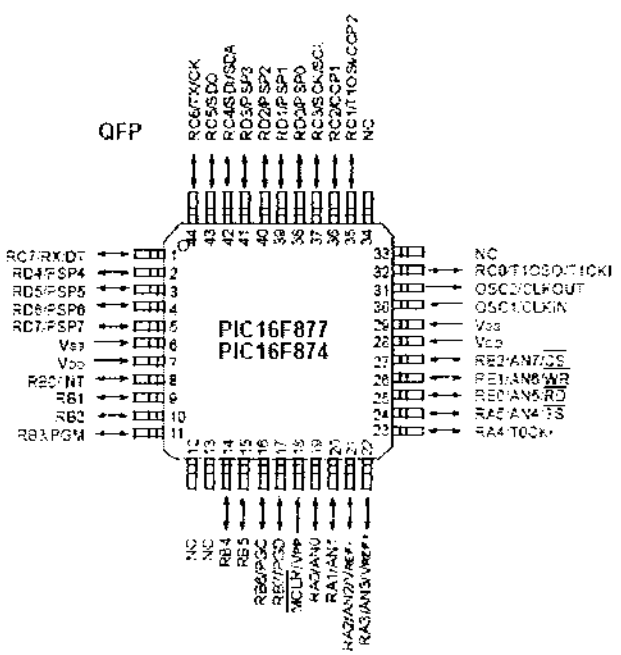
PDIP, SOIC



PLCC



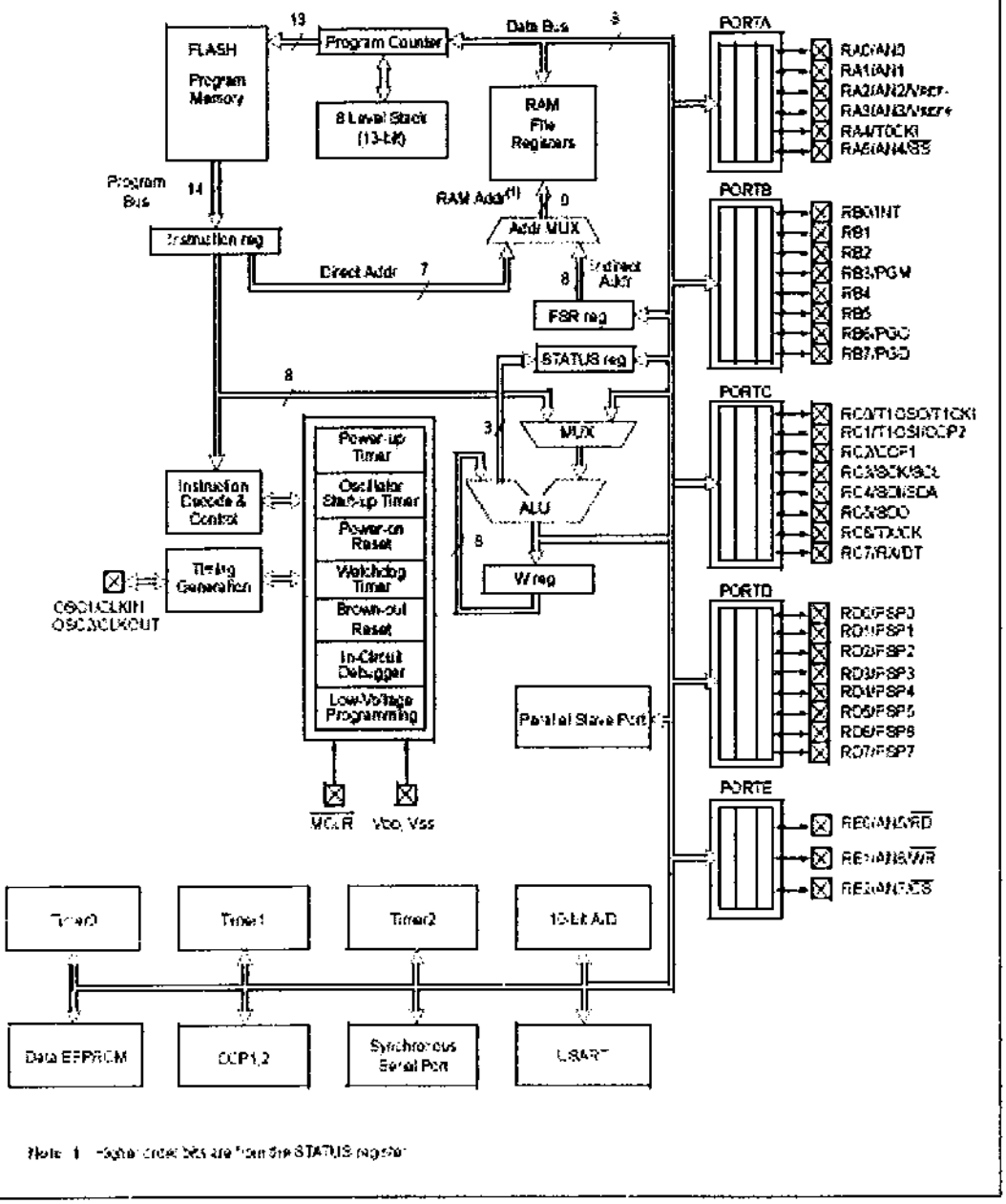
QFP



Key Features PICmicro™ Mid-Range Reference Manual (DS33023)	PIC16F873	PIC16F874	PIC16F876	PIC16F877
Operating Frequency	DC - 20 MHz	DC - 20 MHz	DC - 20 MHz	DC - 20 MHz
RESETS (and Delays)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)
FLASH Program Memory (14-bit words)	4K	4K	8K	8K
Data Memory (bytes)	192	192	368	368
EEPROM Data Memory	128	128	256	256
Interrupts	13	14	13	14
I/O Ports	Ports A,B,C	Ports A,B,C,D,E	Ports A,B,C	Ports A,B,C,D,E
Timers	3	3	3	3
Capture/Compare/PWM Modules	2	2	2	2
Serial Communications	MSSP, USART	MSSP, USART	MSSP, USART	MSSP, USART
Parallel Communications	—	PSP	—	PSP
10-bit Analog-to-Digital Module	5 input channels	8 input channels	5 input channels	8 input channels
Instruction Set	35 instructions	35 instructions	35 instructions	35 instructions

FIGURE 1-2: PIC16F874 AND PIC16F877 BLOCK DIAGRAM

Device	Program FLASH	Data Memory	Data EEPROM
PIC16F874	4K	192 Bytes	128 Bytes
PIC16F877	8K	368 Bytes	256 Bytes



Note 1: Higher order bits are from the STATUS register.

TABLE 1-2: PIC16F874 AND PIC16F877 PINOUT DESCRIPTION

Pin Name	DIP Pin#	PLCC Pin#	QFP Pin#	I/O/P Type	Buffer Type	Description
OSC1/CLKIN	13	14	30	I	ST/CMOS(4)	Oscillator crystal input/external clock source input.
OSC2/CLKOUT	14	15	31	O	—	Oscillator crystal output. Connects to crystal or resonator in crystal oscillator mode. In RC mode, OSC2 pin outputs CLKOUT which has 1/4 the frequency of OSC1, and denotes the instruction cycle rate.
$\overline{MCLR}$/VPP	1	2	18	I/P	ST	Master Clear (Reset) input or programming voltage input. This pin is an active low RESET to the device.
						PORTA is a bi-directional I/O port.
RA0/AN0	2	3	19	I/O	TTL	RA0 can also be analog input0.
RA1/AN1	3	4	20	I/O	TTL	RA1 can also be analog input1.
RA2/AN2/VREF-	4	5	21	I/O	TTL	RA2 can also be analog input2 or negative analog reference voltage.
RA3/AN3/VREF+	5	6	22	I/O	TTL	RA3 can also be analog input3 or positive analog reference voltage.
RA4/T0CKI	6	7	23	I/O	ST	RA4 can also be the clock input to the Timer0 timer/counter. Output is open drain type.
RA5/$\overline{SS}$/AN4	7	8	24	I/O	TTL	RA5 can also be analog input4 or the slave select for the synchronous serial port.
						PORTB is a bi-directional I/O port. PORTB can be software programmed for internal weak pull-up on all inputs.
RB0/INT	33	36	8	I/O	TTL/ST(1)	RB0 can also be the external interrupt pin.
RB1	34	37	9	I/O	TTL	
RB2	35	38	10	I/O	TTL	
RB3/PGM	36	39	11	I/O	TTL	RB3 can also be the low voltage programming input.
RB4	37	41	14	I/O	TTL	Interrupt-on-change pin.
RB5	38	42	15	I/O	TTL	Interrupt-on-change pin.
RB6/PGC	39	43	16	I/O	TTL/ST(2)	Interrupt-on-change pin or In-Circuit Debugger pin. Serial programming clock.
RB7/PGD	40	44	17	I/O	TTL/ST(2)	Interrupt-on-change pin or In-Circuit Debugger pin. Serial programming data.

Legend: I = Input O = Output I/O = Input/output P = power
— = Not used TTL = TTL input ST = Schmitt Trigger input

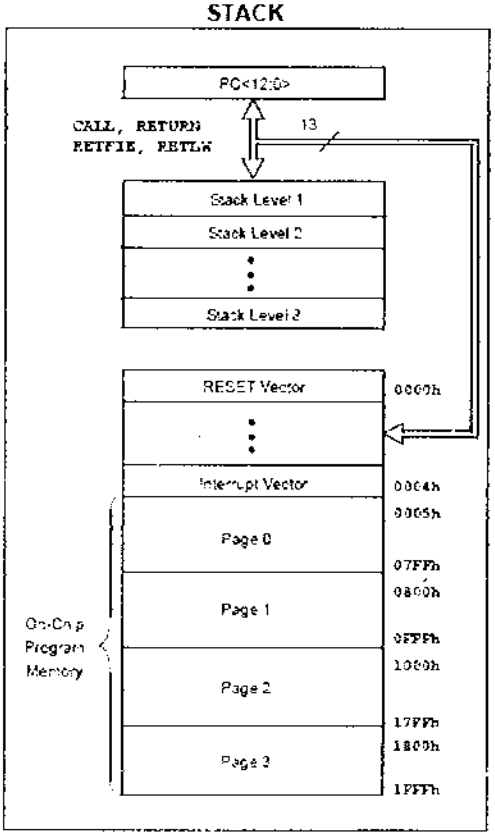
Note 1: This buffer is a Schmitt Trigger input when configured as an external interrupt.
2: This buffer is a Schmitt Trigger input when used in Serial Programming mode.
3: This buffer is a Schmitt Trigger input when configured as general purpose I/O and a TTL input when used in the Parallel Slave Port mode (for interfacing to a microprocessor bus).
4: This buffer is a Schmitt Trigger input when configured in RC oscillator mode and a CMOS input otherwise.

2.0 MEMORY ORGANIZATION

There are three memory blocks in each of the PIC16F87X MCUs. The Program Memory and Data Memory have separate buses so that concurrent access can occur and is detailed in this section. The EEPROM data memory block is detailed in Section 4.0.

Additional information on device memory may be found in the PICmicro™ Mid-Range Reference Manual, (DS33023).

FIGURE 2-1: PIC16F877/876 PROGRAM MEMORY MAP AND STACK

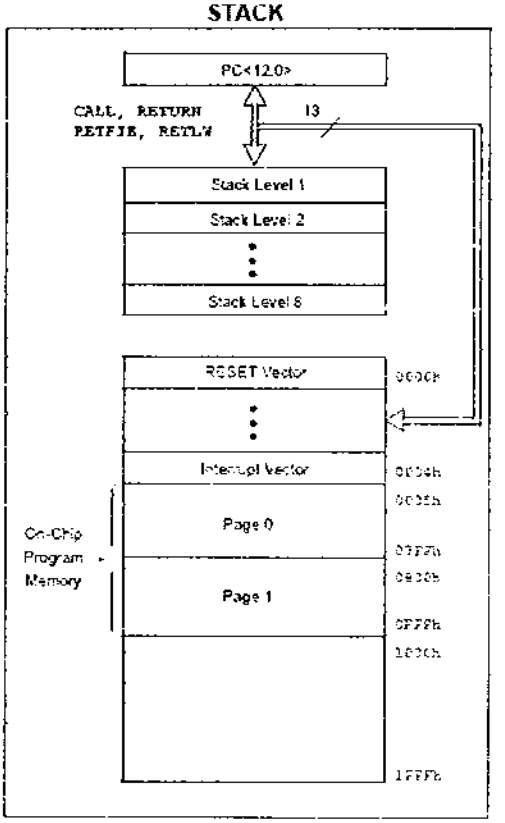


2.1 Program Memory Organization

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.

FIGURE 2-2: PIC16F874/873 PROGRAM MEMORY MAP AND STACK



2.2 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.

RP1:RP0	Bank
00	0
01	1
10	2
11	3

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. All implemented banks contain Special Function Registers. Some frequently used Special Function Registers from one bank may be mirrored in another bank for code reduction and quicker access.

Note:	**EEPROM Data Memory description can be found in Section 4.0 of this data sheet.**

2.2.1 GENERAL PURPOSE REGISTER FILE

The register file can be accessed either directly, or indirectly through the File Select Register (FSR).

2.2.2 SPECIAL FUNCTION REGISTERS

The Special Function Registers are registers used by the CPU and peripheral modules for controlling the desired operation of the device. These registers are implemented as static RAM. A list of these registers is given in Table 2-1.

The Special Function Registers can be classified into two sets: core (CPU) and peripheral. Those registers associated with the core functions are described in detail in this section. Those related to the operation of the peripheral features are described in detail in the peripheral features section.

TABLE 2-1: SPECIAL FUNCTION REGISTER SUMMARY

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Details on page:
Bank 0											
00h(3)	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	27
01h	TMR0	Timer0 Module Register								xxxx xxxx	47
02h(1)	PCL	Program Counter (PC) Least Significant Byte								0000 0000	26
03h(3)	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	18
04h(3)	FSR	Indirect Data Memory Address Pointer								xxxx xxxx	27
05h	PORTA	—	—	PORTA Data Latch when written: PORTA pins when read						--0x 0000	29
06h	PORTB	PORTB Data Latch when written: PORTB pins when read								xxxx xxxx	31
07h	PORTC	PORTC Data Latch when written: PORTC pins when read								xxxx xxxx	33
08h(4)	PORTD	PORTD Data Latch when written: PORTD pins when read								xxxx xxxx	35
09h(4)	PORTE	—	—	—	—	—	RE2	RE1	RE0	---- -xxx	36
0Ah(1,5)	PCLATH	—	—	—	Write Buffer for the upper 5 bits of the Program Counter					---0 0000	26
0Bh(3)	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	20
0Ch	PIR1	PSPIF(2)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	22
0Dh	PIR2	—	(5)	—	EEIF	BCLIF	—	—	CCP2IF	-r-0 0--0	24
0Eh	TMR1L	Holding register for the Least Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	52
0Fh	TMR1H	Holding register for the Most Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	52
10h	T1CON	—	—	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON	--00 0000	51
11h	TMR2	Timer2 Module Register								0000 0000	55
12h	T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	55
13h	SSPBUF	Synchronous Serial Port Receive Buffer/Transmit Register								xxxx xxxx	70, 73
14h	SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	67
15h	CCPR1L	Capture/Compare/PWM Register1 (LSB)								xxxx xxxx	57
16h	CCPR1H	Capture/Compare/PWM Register1 (MSB)								xxxx xxxx	57
17h	CCP1CON	—	—	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	58
18h	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	96
19h	TXREG	USART Transmit Data Register								0000 0000	99
1Ah	RCREG	USART Receive Data Register								0000 0000	101
1Bh	CCPR2L	Capture/Compare/PWM Register2 (LSB)								xxxx xxxx	57
1Ch	CCPR2H	Capture/Compare/PWM Register2 (MSB)								xxxx xxxx	57
1Dh	CCP2CON	—	—	CCP2X	CCP2Y	CCP2M3	CCP2M2	CCP2M1	CCP2M0	--00 0000	58
1Eh	ADRESH	A/D Result Register High Byte								xxxx xxxx	116
1Fh	ADCON0	ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/$\overline{DONE}$	—	ADON	0000 00-0	111

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved. Shaded locations are unimplemented, read as '0'.

Note 1: The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<12:8> whose contents are transferred to the upper byte of the program counter.
2: Bits PSPIE and PSPIF are reserved on PIC16F873/876 devices; always maintain these bits clear.
3: These registers can be addressed from any bank.
4: PORTD, PORTE, TRISD, and TRISE are not physically implemented on PIC16F873/876 devices, read as '0'.
5: PIR2<6> and PIE2<6> are reserved on these devices; always maintain these bits clear.

TABLE 2-1: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Details on page:
Bank 1											
80h(3)	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	27
81h	OPTION_REG	$\overline{RBPU}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	19
82h(3)	PCL	Program Counter (PC) Least Significant Byte								0000 0000	26
83h(3)	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	18
84h(3)	FSR	Indirect Data Memory Address Pointer								xxxx xxxx	27
85h	TRISA	—	—	PORTA Data Direction Register						--11 1111	29
86h	TRISB	PORTB Data Direction Register								1111 1111	31
87h	TRISC	PORTC Data Direction Register								1111 1111	33
88h(4)	TRISD	PORTD Data Direction Register								1111 1111	35
89h(4)	TRISE	IBF	OBF	IBOV	PSPMODE	—	PORTE Data Direction Bits			0000 -111	37
8Ah(1,3)	PCLATH	—	—	—	Write Buffer for the upper 5 bits of the Program Counter					---0 0000	26
8Bh(3)	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	20
8Ch	PIE1	PSPIE(2)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	21
8Dh	PIE2	—	(5)	—	EEIE	BCLIE	—	—	CCP2IE	-r-0 0--0	23
8Eh	PCON	—	—	—	—	—	—	$\overline{POR}$	$\overline{BOR}$	---- --qq	25
8Fh	—	Unimplemented								—	—
90h	—	Unimplemented								—	—
91h	SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000	69
92h	PR2	Timer2 Period Register								1111 1111	65
93h	SSPADD	Synchronous Serial Port (I^2C mode) Address Register								0000 0000	73, 74
94h	SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	66
95h	—	Unimplemented								—	—
96h	—	Unimplemented								—	—
97h	—	Unimplemented								—	—
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	95
99h	SPBRG	Baud Rate Generator Register								0000 0000	97
9Ah	—	Unimplemented								—	—
9Bh	—	Unimplemented								—	—
9Ch	—	Unimplemented								—	—
9Dh	—	Unimplemented								—	—
9Eh	ADRESL	A/D Result Register Low Byte								xxxx xxxx	116
9Fh	ADCON1	ADFM	—	—	—	PCFG3	PCFG2	PCFG1	PCFG0	0--- 0000	112

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as '0', r = reserved. Shaded locations are unimplemented, read as '0'.

Note 1: The upper byte of the program counter is not directly accessible. PCLATH is a holding register for the PC<12:8> whose contents are transferred to the upper byte of the program counter.

2: Bits PSPIE and PSPIF are reserved on PIC16F873/876 devices; always maintain these bits clear.

3: These registers can be addressed from any bank.

4: PORTD, PORTE, TRISD, and TRISE are not physically implemented on PIC16F873/876 devices; read as '0'.

5: PIR2<6> and PIE2<6> are reserved on these devices; always maintain these bits clear.

USART

TABLE 10-5: REGISTERS ASSOCIATED WITH ASYNCHRONOUS TRANSMISSION

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other RESETS
0Bh, 8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	R0IF	0000 000x	0000 000u
0Ch	PIR1	PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
18h	RCSTA	SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D	0000 -00x	0000 -00x
19h	TXREG	USART Transmit Register								0000 0000	0000 0000
8Ch	PIE1	PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented locations read as '0'. Shaded cells are not used for asynchronous transmission.
Note 1: Bits PSPIE and PSPIF are reserved on the PIC16F873/876; always maintain these bits clear.

TABLE 10-7: REGISTERS ASSOCIATED WITH ASYNCHRONOUS RECEPTION

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other RESETS
0Bh, 8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	R0IF	0000 000x	0000 000u
0Ch	PIR1	PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
18h	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
1Ah	RCREG	USART Receive Register								0000 0000	0000 0000
8Ch	PIE1	PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented locations read as '0'. Shaded cells are not used for asynchronous reception.
Note 1: Bits PSPIE and PSPIF are reserved on PIC16F873/876 devices; always maintain these bits clear.

REGISTER 10-1: TXSTA: TRANSMIT STATUS AND CONTROL REGISTER (ADDRESS 98h)

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D
bit 7							bit 0

bit 7 **CSRC:** Clock Source Select bit

Asynchronous mode:
Don't care

Synchronous mode:
1 = Master mode (clock generated internally from BRG)
0 = Slave mode (clock from external source)

bit 6 **TX9:** 9-bit Transmit Enable bit
1 = Selects 9-bit transmission
0 = Selects 8-bit transmission

bit 5 **TXEN:** Transmit Enable bit
1 = Transmit enabled
0 = Transmit disabled

Note: SREN/CREN overrides TXEN in SYNC mode.

bit 4 **SYNC:** USART Mode Select bit
1 = Synchronous mode
0 = Asynchronous mode

bit 3 **Unimplemented:** Read as '0'

bit 2 **BRGH:** High Baud Rate Select bit

Asynchronous mode:
1 = High speed
0 = Low speed

Synchronous mode:
Unused in this mode

bit 1 **TRMT:** Transmit Shift Register Status bit
1 = TSR empty
0 = TSR full

bit 0 **TX9D:** 9th bit of Transmit Data, can be parity bit

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 10-2: RCSTA: RECEIVE STATUS AND CONTROL REGISTER (ADDRESS 18h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
bit 7							bit 0

bit 7 **SPEN**: Serial Port Enable bit
1 = Serial port enabled (configures RC7/RX/DT and RC6/TX/CK pins as serial port pins)
0 = Serial port disabled

bit 6 **RX9**: 9-bit Receive Enable bit
1 = Selects 9-bit reception
0 = Selects 8-bit reception

bit 5 **SREN**: Single Receive Enable bit
Asynchronous mode:
Don't care
Synchronous mode - master:
1 = Enables single receive
0 = Disables single receive
This bit is cleared after reception is complete.
Synchronous mode - slave:
Don't care

bit 4 **CREN**: Continuous Receive Enable bit
Asynchronous mode:
1 = Enables continuous receive
0 = Disables continuous receive
Synchronous mode:
1 = Enables continuous receive until enable bit CREN is cleared (CREN overrides SREN)
0 = Disables continuous receive

bit 3 **ADDEN**: Address Detect Enable bit
Asynchronous mode 9-bit (RX9 = 1):
1 = Enables address detection, enables interrupt and load of the receive buffer when RSR<8> is set
0 = Disables address detection, all bytes are received, and ninth bit can be used as parity bit

bit 2 **FERR**: Framing Error bit
1 = Framing error (can be updated by reading RCREG register and receive next valid byte)
0 = No framing error

bit 1 **OERR**: Overrun Error bit
1 = Overrun error (can be cleared by clearing bit CREN)
0 = No overrun error

bit 0 **RX9D**: 9th bit of Received Data (can be parity bit, but must be calculated by user firmware)

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

TABLE 11-2: REGISTERS/BITS ASSOCIATED WITH A/D

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on MCLR, WDT
08h,88h, 108h,188h	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
8Ch	PIE1	PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
1Eh	ADRESH	A/D Result Register High Byte								xxxx xxxx	uuuu uuuu
9Eh	ADRESL	A/D Result Register Low Byte								xxxx xxxx	uuuu uuuu
1Fh	ADCON0	ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/DONE	—	ADON	0000 00-0	0000 00-0
9Fh	ADCON1	ADFM	—	—	—	PCFG3	PCFG2	PCFG1	PCFG0	--0- 0000	--0- 0000
85h	TRISA	—	—	PORTA Data Direction Register						--11 1111	--11 1111
05h	PORTA	—	—	PORTA Data Latch when written: PORTA pins when read						--0x 0000	--0u 0000
89h(1)	TRISE	IBF	OBF	IBOV	PSPMODE	—	PORTE Data Direction bits			0000 -111	0000 -111
09h(1)	PORTE	—	—	—	—	—	RE2	RE1	RE0	---- -xxx	---- -uuu

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'. Shaded cells are not used for A/D conversion
Note 1: These registers/bits are not available on the 28-pin devices.

REGISTER 11-1: ADCON0 REGISTER (ADDRESS: 1Fh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	—	ADON
bit 7							bit 0

bit 7-6 **ADCS1:ADCS0**: A/D Conversion Clock Select bits
00 = Fosc/2
01 = Fosc/8
10 = Fosc/32
11 = FRC (clock derived from the internal A/D module RC oscillator)

bit 5-3 **CHS2:CHS0**: Analog Channel Select bits
000 = channel 0, (RA0/AN0)
001 = channel 1, (RA1/AN1)
010 = channel 2, (RA2/AN2)
011 = channel 3, (RA3/AN3)
100 = channel 4, (RA5/AN4)
101 = channel 5, (RE0/AN5)(1)
110 = channel 6, (RE1/AN6)(1)
111 = channel 7, (RE2/AN7)(1)

bit 2 **GO/$\overline{\text{DONE}}$**: A/D Conversion Status bit
If ADON = 1:
1 = A/D conversion in progress (setting this bit starts the A/D conversion)
0 = A/D conversion not in progress (this bit is automatically cleared by hardware when the A/D conversion is complete)

bit 1 **Unimplemented**: Read as '0'

bit 0 **ADON**: A/D On bit
1 = A/D converter module is operating
0 = A/D converter module is shut-off and consumes no operating current

Note 1: These channels are not available on PIC16F873/876 devices.

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 11-2: ADCON1 REGISTER (ADDRESS 9Fh)

U-0	U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	—	—	—	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

bit 7 **ADFM:** A/D Result Format Select bit
1 = Right justified. 6 Most Significant bits of ADRESH are read as '0'.
0 = Left justified. 6 Least Significant bits of ADRESL are read as '0'.

bit 6-4 **Unimplemented:** Read as '0'

bit 3-0 **PCFG3:PCFG0**: A/D Port Configuration Control bits:

PCFG3: PCFG0	AN7[1] RE2	AN6[1] RE1	AN5[1] RE0	AN4 RA5	AN3 RA3	AN2 RA2	AN1 RA1	AN0 RA0	VREF+	VREF-	CHAN/ Refs[2]
0000	A	A	A	A	A	A	A	A	VDD	VSS	8/0
0001	A	A	A	A	VREF+	A	A	A	RA3	VSS	7/1
0010	D	D	D	A	A	A	A	A	VDD	VSS	5/0
0011	D	D	D	A	VREF+	A	A	A	RA3	VSS	4/1
0100	D	D	D	D	A	D	A	A	VDD	VSS	3/0
0101	D	D	D	D	VREF+	D	A	A	RA3	VSS	2/1
011x	D	D	D	D	D	D	D	D	VDD	VSS	0/0
1000	A	A	A	A	VREF+	VREF-	A	A	RA3	RA2	6/2
1001	D	D	A	A	A	A	A	A	VDD	VSS	6/0
1010	D	D	A	A	VREF+	A	A	A	RA3	VSS	5/1
1011	D	D	A	A	VREF+	VREF-	A	A	RA3	RA2	4/2
1100	D	D	D	A	VREF+	VREF-	A	A	RA3	RA2	3/2
1101	D	D	D	D	VREF+	VREF-	A	A	RA3	RA2	2/2
1110	D	D	D	D	D	D	D	A	VDD	VSS	1/0
1111	D	D	D	D	VREF+	VREF-	D	A	RA3	RA2	1/2

A = Analog input D = Digital I/O

Note 1: These channels are not available on PIC16F873/876 devices.
2: This column indicates the number of analog channels available as A/D inputs and the number of analog channels used as voltage reference inputs.

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

19-4323; Rev 11; 2/03

MAXIM

+5V-Powered, Multichannel RS-232 Drivers/Receivers

General Description

The MAX220–MAX249 family of line drivers/receivers is intended for all EIA/TIA-232E and V.28/V.24 communications interfaces, particularly applications where ±12V is not available.

These parts are especially useful in battery-powered systems, since their low-power shutdown mode reduces power dissipation to less than 5µW. The MAX225, MAX233, MAX235, and MAX245/MAX246/MAX247 use no external components and are recommended for applications where printed circuit board space is critical.

Applications

Portable Computers

Low-Power Modems

Interface Translation

Battery-Powered RS-232 Systems

Multidrop RS-232 Networks

Features

Superior to Bipolar

- Operate from Single +5V Power Supply (+5V and +12V—MAX231/MAX239)
- Low-Power Receive Mode in Shutdown (MAX223/MAX242)
- Meet All EIA/TIA-232E and V.28 Specifications
- Multiple Drivers and Receivers
- 3-State Driver and Receiver Outputs
- Open-Line Detection (MAX243)

Ordering Information

PART	TEMP RANGE	PIN-PACKAGE
MAX220CPE	0°C to +70°C	16 Plastic DIP
MAX220CSE	0°C to +70°C	16 Narrow SO
MAX220CWE	0°C to +70°C	16 Wide SO
MAX220C/D	0°C to +70°C	Dice*
MAX220EPE	-40°C to +85°C	16 Plastic DIP
MAX220ESE	-40°C to +85°C	16 Narrow SO
MAX220EWE	-40°C to +85°C	16 Wide SO
MAX220EJE	-40°C to +85°C	16 CERDIP
MAX220MJE	-55°C to +125°C	16 CERDIP

Ordering information continued at end of data sheet.

**Contact factory for dice specifications.*

Selection Table

Part Number	Power Supply (V)	No. of RS-232 Drivers/Rx	No. of Ext. Caps	Nominal Cap. Value (µF)	SHDN & Three-State	Rx Active in SHDN	Data Rate (kbps)	Features
MAX220	+5	2/2	4	0.1	No	—	120	Ultra-low-power, industry-standard pinout
MAX222	+5	2/2	4	0.1	Yes	—	200	Low-power shutdown
MAX223 (MAX213)	+5	4/5	4	1.0 (0.1)	Yes	✔	120	MAX241 and receivers active in shutdown
MAX225	+5	5/5	0	—	Yes	✔	120	Available in SO
MAX230 (MAX200)	+5	5/0	4	1.0 (0.1)	Yes	—	120	5 drivers with shutdown
MAX231 (MAX201)	+5 and +7.5 to +13.2	2/2	2	1.0 (0.1)	No	—	120	Standard +5/+12V or battery supplies; same functions as MAX232
MAX232 (MAX202)	+5	2/2	4	1.0 (0.1)	No		120 (64)	Industry standard
MAX232A	+5	2/2	4	0.1	No	—	200	Higher slew rate, small caps
MAX233 (MAX203)	+5	2/2	0	—	No	—	120	No external caps
MAX233A	+5	2/2	0	—	No	—	200	No external caps, high slew rate
MAX234 (MAX204)	+5	4/0	4	1.0 (0.1)	No	—	120	Replaces 1488
MAX235 (MAX205)	+5	5/5	0	—	Yes	—	120	No external caps
MAX236 (MAX206)	+5	4/3	4	1.0 (0.1)	Yes	—	120	Shutdown, three state
MAX237 (MAX207)	+5	5/3	4	1.0 (0.1)	No	—	120	Complements IBM PC serial port
MAX238 (MAX208)	+5	4/4	4	1.0 (0.1)	No	—	120	Replaces 1488 and 1489
MAX239 (MAX209)	+5 and +7.5 to +13.2	3/5	2	1.0 (0.1)	No	—	120	Standard +5/+12V or battery supplies; single-package solution for IBM PC serial port
MAX240	+5	5/5	4	1.0	Yes	—	120	DIP or flatpack package
MAX241 (MAX211)	+5	4/5	4	1.0 (0.1)	Yes	—	120	Complete IBM PC serial port
MAX242	+5	2/2	4	0.1	Yes	✔	200	Separate shutdown and enable
MAX243	+5	2/2	4	0.1	No	—	200	Open-line detection simplifies cabling
MAX244	+5	8/10	4	1.0	No		120	High slew rate
MAX245	+5	8/10	0	—	Yes	✔	120	High slew rate, int. caps, two shutdown modes
MAX246	+5	8/10	0	—	Yes	✔	120	High slew rate, int. caps, three shutdown modes
MAX247	+5	8/9	0	—	Yes	✔	120	High slew rate, int. caps, nine operating modes
MAX248	+5	8/8	4	1.0	Yes	✔	120	High slew rate, selective half-chip enables
MAX249	+5	8/10	4	1.0	Yes	✔	120	Available in quad flatpack package

+5V-Powered, Multichannel RS-232 Drivers/Receivers

ABSOLUTE MAXIMUM RATINGS—MAX220/222/232A/233A/242/243

Supply Voltage (V_{CC}) ..... -0.3V to +6V
Input Voltages
T_{IN} ..... -0.3V to (V_{CC} + 0.3V)
R_{IN} (Except MAX220) ..... ±30V
R_{IN} (MAX220) ..... ±25V
T_{OUT} (Except MAX220) (Note 1) ..... ±15V
T_{OUT} (MAX220) ..... ±13.2V
Output Voltages
T_{OUT} ..... ±15V
R_{OUT} ..... -0.3V to (V_{CC} + 0.3V)
Driver/Receiver Output Short Circuited to GND ..... Continuous
Continuous Power Dissipation (T_A = +70°C)
16-Pin Plastic DIP (derate 10.53mW/°C above +70°C) 842mW
18-Pin Plastic DIP (derate 11.11mW/°C above +70°C) 889mW
20-Pin Plastic DIP (derate 8.00mW/°C above +70°C) 440mW
16-Pin Narrow SO (derate 8.70mW/°C above +70°C) 696mW
16-Pin Wide SO (derate 9.52mW/°C above +70°C) 762mW
18-Pin Wide SO (derate 9.52mW/°C above +70°C) 762mW
20-Pin Wide SO (derate 10.00mW/°C above +70°C) 800mW
20-Pin SSOP (derate 8.00mW/°C above +70°C) 640mW
16-Pin CERDIP (derate 10.00mW/°C above +70°C) 800mW
18-Pin CERDIP (derate 10.53mW/°C above +70°C) 842mW
Operating Temperature Ranges
MAX2_ _AC_ _, MAX2_ _C_ _ ..... 0°C to +70°C
MAX2_ _AE_ _, MAX2_ _E_ _ ..... -40°C to +85°C
MAX2_ _AM_ _, MAX2_ _M_ _ ..... -55°C to +125°C
Storage Temperature Range ..... -65°C to +160°C
Lead Temperature (soldering, 10s) ..... +300°C

Note 1: Input voltage measured with T_{OUT} in high-impedance state, $\overline{SHDN}$ or V_{CC} = 0V.
Note 2: For the MAX220, V+ and V- can have a maximum magnitude of 7V, but their absolute difference cannot exceed 13V.

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

ELECTRICAL CHARACTERISTICS—MAX220/222/232A/233A/242/243

(V_{CC} = +5V ±10%, C1–C4 = 0.1µF, MAX220, C1 = 0.047µF, C2–C4 = 0.33µF, T_A = T_{MIN} to T_{MAX}, unless otherwise noted.)

PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
RS-232 TRANSMITTERS						
Output Voltage Swing	All transmitter outputs loaded with 3kΩ to GND		±5	±8		V
Input Logic Threshold Low				1.4	0.8	V
Input Logic Threshold High	All devices except MAX220		2	1.4		V
	MAX220: V_{CC} = 5.0V		2.4			
Logic Pull-Up/Input Current	All except MAX220, normal operation			5	40	µA
	$\overline{SHDN}$ = 0V, MAX222/242, shutdown, MAX220			±0.01	±1	
Output Leakage Current	V_{CC} = 5.5V, $\overline{SHDN}$ = 0V, V_{OUT} = ±15V, MAX222/242			±0.01	±10	µA
	V_{CC} = $\overline{SHDN}$ = 0V, V_{OUT} = ±15V			±0.01	±10	
Data Rate				200	116	kbps
Transmitter Output Resistance	V_{CC} = V+ = V- = 0V, V_{OUT} = ±2V		300	10M		Ω
Output Short-Circuit Current	V_{OUT} = 0V			±7	±22	mA
RS-232 RECEIVERS						
RS-232 Input Voltage Operating Range					±30	V
RS-232 Input Threshold Low	V_{CC} = 5V	All except MAX243 $R2_{IN}$	0.8	1.3		V
		MAX243 $R2_{IN}$ (Note 2)	-3			
RS-232 Input Threshold High	V_{CC} = 5V	All except MAX243 $R2_{IN}$		1.8	2.4	V
		MAX243 $R2_{IN}$ (Note 2)		-0.5	-0.1	
RS-232 Input Hysteresis	All except MAX243, V_{CC} = 5V, no hysteresis in shdn.		0.2	0.5	1	V
	MAX243			1		
RS-232 Input Resistance			3	5	7	kΩ
TTL/CMOS Output Voltage Low	I_{OUT} = 3.2mA			0.2	0.4	V
TTL/CMOS Output Voltage High	I_{OUT} = -1.0mA		3.5	V_{CC} - 0.2		V
TTL/CMOS Output Short-Circuit Current	Sourcing V_{OUT} = GND		-2	-10		mA
	Shrinking V_{OUT} = V_{CC}		10	30		

ELECTRICAL CHARACTERISTICS—MAX220/222/232A/233A/242/243 (continued)

(V_{CC} = +5V ±10%, C1-C4 = 0.1µF, MAX220, C1 = 0.047µF, C2-C4 = 0.33µF, T_A = T_{MIN} to T_{MAX}, unless otherwise noted.)

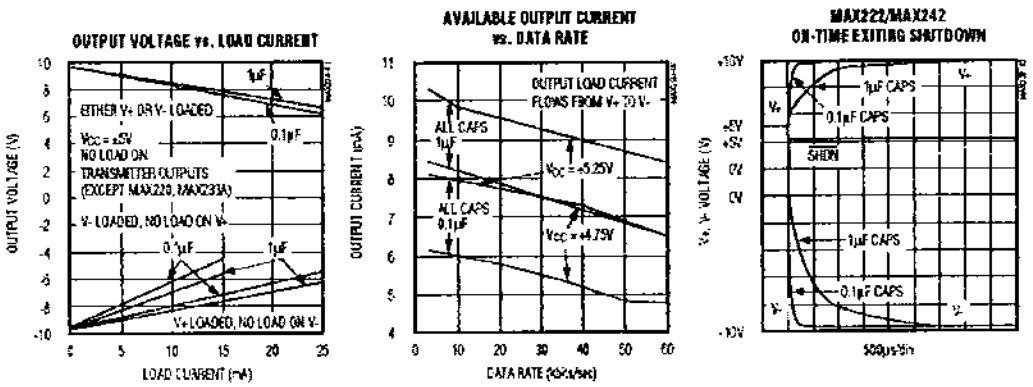
PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
TTL/CMOS Output Leakage Current	$\overline{SHDN}$ = V_{CC} or $\overline{EN}$ = V_{CC} ($\overline{SHDN}$ = 0V for MAX222), 0V ≤ V_{OUT} ≤ V_{CC}			±0.05	±10	µA
$\overline{EN}$ Input Threshold Low	MAX242			1.4	0.8	V
$\overline{EN}$ Input Threshold High	MAX242		2.0	1.4		V
Operating Supply Voltage			4.5		5.5	V
V_{CC} Supply Current ($\overline{SHDN}$ = V_{CC}), Figures 5, 6, 11, 19	No load	MAX220		0.5	2	mA
		MAX222/232A/233A/242/243		4	10	
	3kΩ load both inputs	MAX220		12		
		MAX222/232A/233A/242/243		15		
Shutdown Supply Current	MAX222/242	T_A = +25°C		0.1	10	µA
		T_A = 0°C to +70°C		2	50	
		T_A = -40°C to +85°C		2	50	
		T_A = -55°C to +125°C		35	100	
$\overline{SHDN}$ Input Leakage Current	MAX222/242				±1	µA
$\overline{SHDN}$ Threshold Low	MAX222/242			1.4	0.8	V
$\overline{SHDN}$ Threshold High	MAX222/242		2.0	1.4		V
Transition Slew Rate	C_L = 50pF to 2500pF, R_L = 3kΩ to 7kΩ, V_{CC} = 5V, T_A = +25°C, measured from +3V to -3V or -3V to +3V	MAX222/232A/233A/242/243	6	12	30	V/µs
		MAX220	1.5	3	30	
Transmitter Propagation Delay TLL to RS-232 (Normal Operation), Figure 1	t_{PHLT}	MAX222/232A/233A/242/243		1.3	3.5	µs
		MAX220		4	10	
	t_{PLHT}	MAX222/232A/233A/242/243		1.5	3.5	
		MAX220		5	10	
Receiver Propagation Delay RS-232 to TLL (Normal Operation), Figure 2	t_{PHLR}	MAX222/232A/233A/242/243		0.5	1	µs
		MAX220		0.6	3	
	t_{PLHR}	MAX222/232A/233A/242/243		0.6	1	
		MAX220		0.8	3	
Receiver Propagation Delay RS-232 to TLL (Shutdown), Figure 2	t_{PHLS}	MAX242		0.5	10	µs
	t_{PLHS}	MAX242		2.5	10	
Receiver-Output Enable Time, Figure 3	t_{ER}	MAX242		125	500	ns
Receiver-Output Disable Time, Figure 3	t_{DR}	MAX242		150	500	ns
Transmitter-Output Enable Time ($\overline{SHDN}$ Goes High), Figure 4	t_{ET}	MAX222/242, 0.1µF caps (includes charge-pump start-up)		250		µs
Transmitter-Output Disable Time ($\overline{SHDN}$ Goes Low), Figure 4	t_{DT}	MAX222/242, 0.1µF caps		600		ns
Transmitter + to - Propagation Delay Difference (Normal Operation)	t_{PHLT} - t_{PLHT}	MAX222/232A/233A/242/243		300		ns
		MAX220		2000		
Receiver + to - Propagation Delay Difference (Normal Operation)	t_{PHLR} - t_{PLHR}	MAX222/232A/233A/242/243		100		ns
		MAX220		225		

Note 3: MAX243 $R2_{OUT}$ is guaranteed to be low when $R2_{IN}$ is ≥ 0V or is floating.

+5V-Powered, Multichannel RS-232 Drivers/Receivers

Typical Operating Characteristics

MAX220/MAX222/MAX232A/MAX233A/MAX242/MAX243



APPENDIX-III

THINGS TRIED WHICH DID NOT WORK

While testing our design, we implemented several different electrode designs to find one that would give us the cleanest signal. We found that several things did not work. First a simple wire pressed against the skin was flimsy and provided too small a surface contact. Then, per the suggestion of a website, we built electrodes out of copper and Velcro wrapped around the fingers. The Wheatstone bridge, when kept open circuited gives shock. The differential amplifier used was not needed since PIC16F877 does all the work.

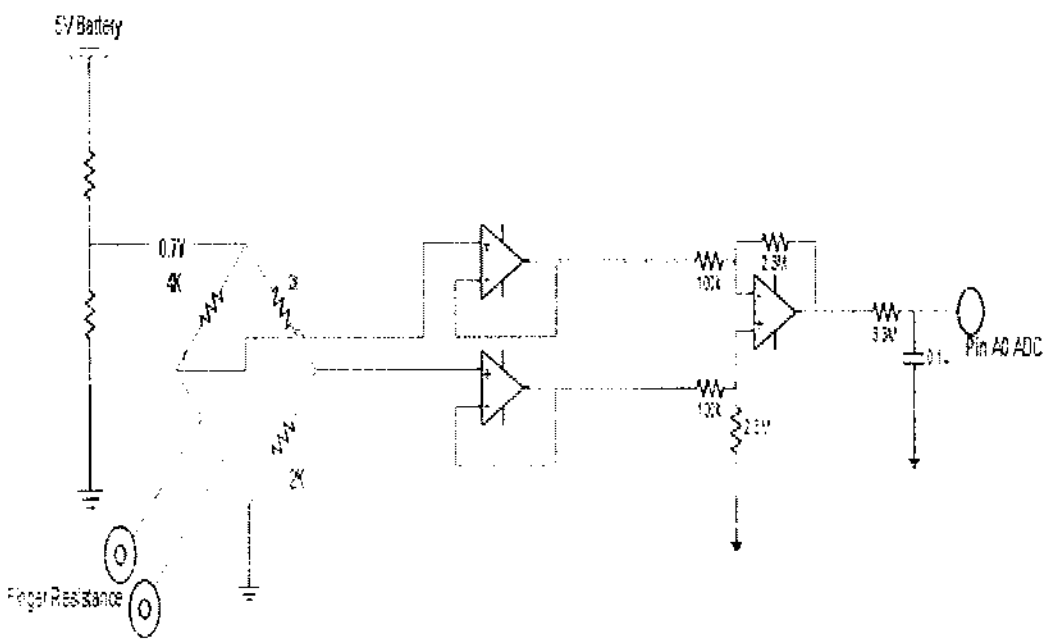
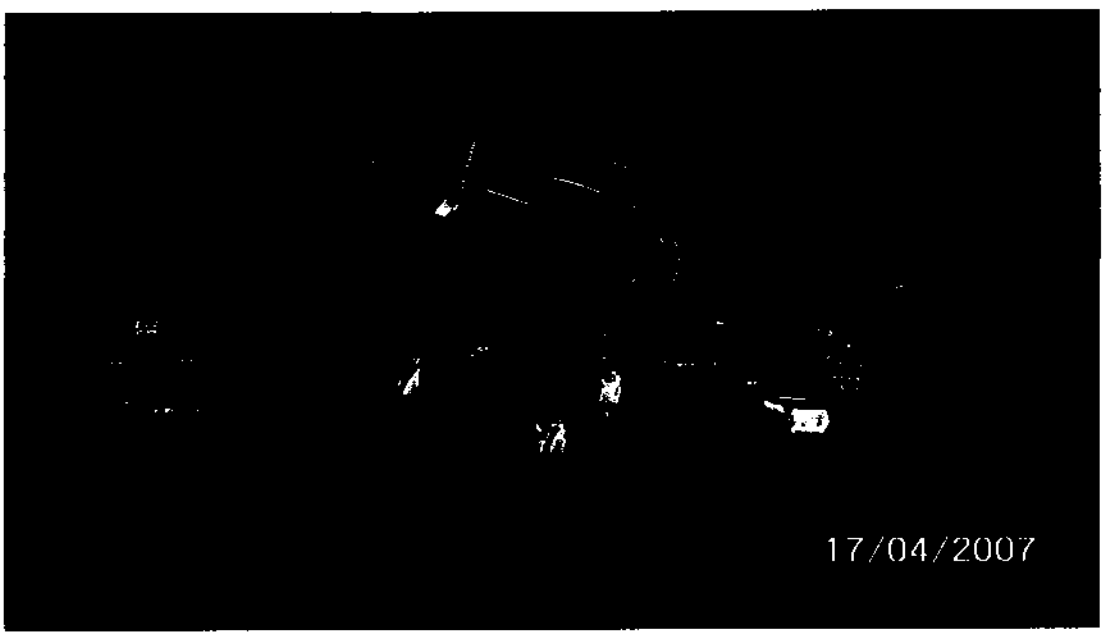
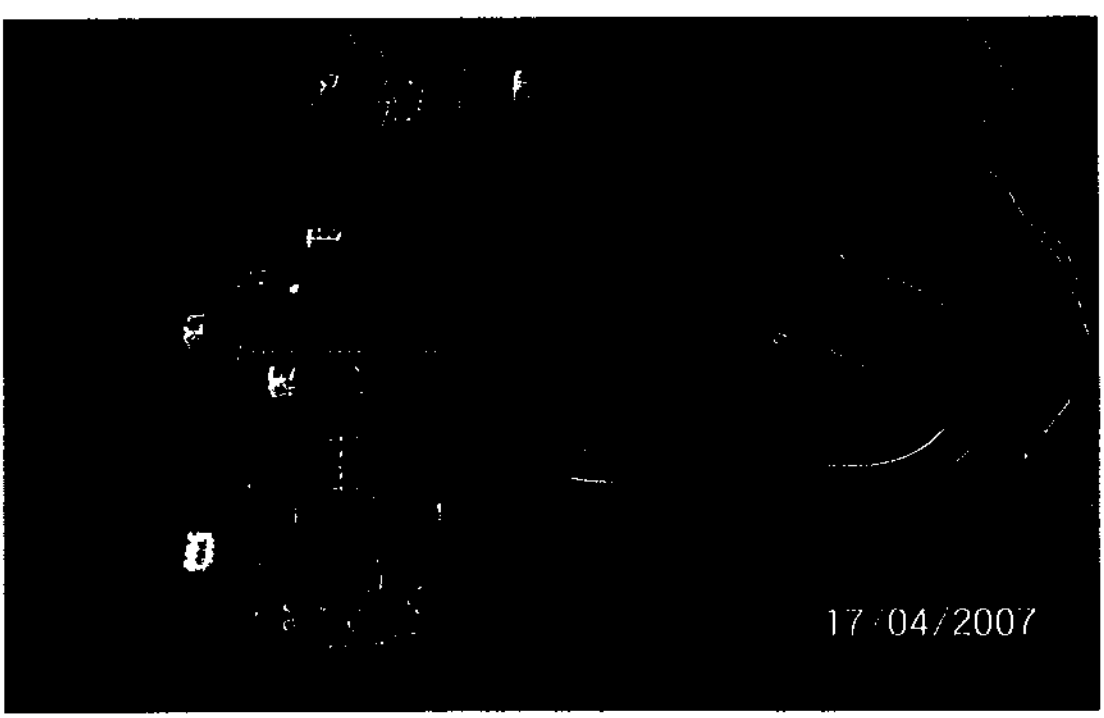


Fig 5.1 Amplification and filtering hardware circuit

PHOTOGRAPH



EDA MODULE



REFERENCES

REFERENCES

- John Peatman "EMBEDDED PIC MICROCONTROLLERS"
- Leslie Cromwell, Fred J. Weibell, Erich A. Pfeiffer "BIOMEDICAL INSTRUMENTATION AND MEASUREMENTS"

WEBSITES:

Data Sheets:

- www.microchip.com
- www.maxim-ic.com

Technical Reference:

- www.google.com
- www.howstuffswork.com