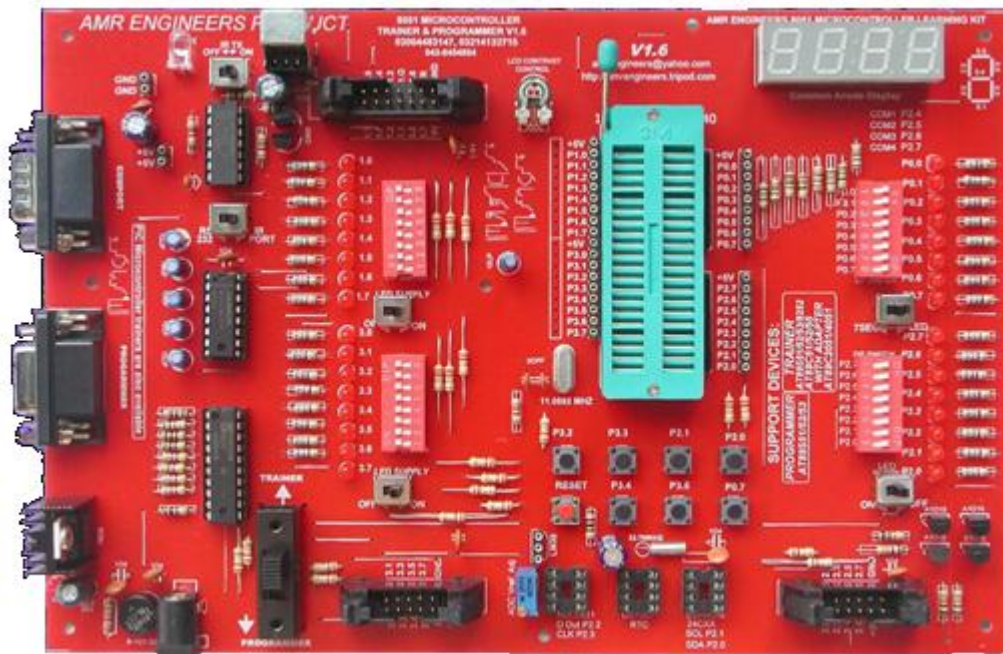


EE – 323L

Microprocessor Interfacing Lab



Faculty of Electronic Engineering
Ghulam Ishaq Khan Institute of Engineering Sciences & Technology

Table of Contents

Lab - 1: Introduction to Microcontrollers and LED Pattern using 8051.....	6
1.1 Objective.....	6
1.2 Pre-lab Reading	6
1.3 Résumé Of Theory	6
1.4 Activities and Exercises.....	7
1.5 Assessment Sheet.....	7
Lab - 2: Seven Segment Display and Clock using Timers in 8051	8
2.1 Objective.....	8
2.2 Pre-lab Reading	8
2.3 Résumé Of Theory	8
2.4 Activities and Exercises.....	10
2.5 Assessment Sheet.....	10
Lab - 3: Interrupts Programming and Serial Communication in 8051	11
3.1 Objective.....	11
3.2 Pre-lab Reading	11
3.3 Résumé Of Theory	11
3.4 Activities and Exercises.....	14
3.5 Assessment Sheet.....	14
Lab - 4: LCD and Keypad Interfacing with 8051	15
4.1 Objective.....	15
4.2 Pre-Lab Reading.....	15
4.3 Résumé Of Theory	15
4.4 Activities and Exercises.....	18
4.5 Assessment Sheet.....	18
Lab - 5: PIC Microcontroller Families Introduction and PIC18F4550.....	19
5.1 Objective.....	19
5.2 Pre-Lab Reading.....	19
5.3 Résumé Of Theory	19
5.4 Activities and exercise	21
5.5 Assessment Sheet.....	21
Lab - 6: Timers/Counters Programming in PIC18F4550.....	22
6.1 Objective.....	22
6.2 Pre-Lab Reading.....	22
6.3 Résumé of Theory.....	22
6.4 Activities and Exercise	25
6.5 Assessment Sheet.....	25
Lab - 7: Interrupts Programming in PIC18F4550	26
7.1 Objective.....	26
7.2 Pre-Lab Reading.....	26
7.3 Résumé Of Theory	26
7.4 Activities and Exercise	30
7.5 Assessment Sheet.....	30
Lab - 8: ADC Programming in PIC18F4550	31
8.1 Objective.....	31
8.2 Pre-Lab Reading.....	31
8.3 Résumé Of Theory	31

8.4	Activities and Exercise	34
8.5	Assessment Sheet	34
Lab - 9:	Capture Compare PWM (CCP) Programming in PIC18F4550	35
9.1	Objective	35
9.2	Pre-Lab Reading	35
9.3	Résumé Of Theory	35
9.4	Activities and Exercise	37
9.5	Assessment Sheet	37
Lab - 10:	Serial Communication using PIC18F4550	38
10.1	Objective	38
10.2	Pre-Lab Reading	38
10.3	Résumé Of Theory	38
10.4	Activities and Exercise	43
10.5	Assessment Sheet	43
Lab - 11:	USB Communication using PIC16F4550	44
11.1	Objective	44
11.2	Pre-Lab Reading	44
11.3	Résumé Of Theory	44
11.4	Activities and Exercise	47
11.5	Assessment Sheet	47
Lab - 12:	Doing a joint small project in lab 😊.....	48
12.1	Objective	48
12.2	Pre-Lab Reading	48
12.3	Résumé Of Theory	48
12.4	Activities and Exercise	48
12.5	Assessment Sheet	48

Instructions for Students

1. Attendance is mandatory for students in all the labs. If a student is absent from a lab due to any reason, he/she will have to get written permission of the Dean to perform that lab. The Dean may allow students to perform lab if he finds that the student has a genuine excuse.
2. Students should bring their text books to the lab, so that they can refer to theory, diagrams, and instruction sets whenever required.
3. Labs will be graded in double entry fashion; one entry in the assessment sheet given at the end of every lab and another entry in the instructor's record. This system of keeping records will keep students aware of their performance throughout the lab.
4. The assessment sheet at the end of every lab looks like this:

Name: _____

Reg. No. : _____

Date of Lab: _____

Problem Number	1 -	
	2 -	
	3 -	
Lab Performance	Working	
	Viva	
Total Score in Lab - 1		
Instructor's Verification		

5. The tentative marks distribution for final grade is as follows:
 - Lab Performance – 30 %
 - Mid Exam – 20%
 - Project – 35%
 - Lab Final – 15 %

(This marking scheme can be altered by the lab instructor if found necessary)

Lab - 1: Introduction to Microcontrollers and LED Pattern using 8051

1.1 OBJECTIVE

- To have a general knowledge about different families of famous microcontrollers and discuss the major differences and special features of each
- To program 8051 microcontroller and make different LED glowing patterns using loops and subroutines.

1.2 PRE-LAB READING

- Chapter 1, 2 , 3, Mazidi, "The 8051 Microcontroller and Embedded Systems"
- <http://en.wikipedia.org/wiki/Microcontroller>

1.3 RÉSUMÉ OF THEORY

Three most famous microcontroller families are Intel 8051, Microchip PIC and Atmel AVR. 8051 is the simplest and easiest of all. It is most suited for beginners to get experience in this field. 8051 has limited code ROM and built-in peripherals and thus cannot be used for advanced applications. Microchip PIC and AVR are advanced level microcontrollers with a very developed database and support. They have a large number of advanced built-in peripherals like ADC, CCP, USB, Bluetooth etc. They have a large code ROM and thus can serve applications requiring big codes and high processing powers. Both these families have a drawback that their development environments are somewhat complicated to use as compared to 8051 and their user interface is also complex.

Today, we will start our experiments with 8051 microcontroller. The basic aim of this lab is to get you familiar with the Integrated Development Environment (IDE), development board and programmer we are going to use. The MIDE-51 is freeware Integrated Development Environment (IDE) for MCS-51 microcontroller. The full package already comes with an assembler and a simulator.

The commands we are going to use in this lab are:

ORG, JMP, DJNZ, MOV, ADD, SUBB, INC, DEC, SET, CLR, CPL, END

The function of these assembly language commands will be explained in the pre-lab lecture.

1.4 ACTIVITIES AND EXERCISES

1. Run the following code in the simulator and fill the blank spaces in which you are required to write the values of register A, R0, R1, PC and Sp at each step of the code.

ORG 00H	
MOV R0, #22H	R0= -----, R1= -----, A= -----, PC= -----
MOV R1, 00H	R0= -----, R1= -----, A= -----, PC= -----
MOV A, #13H	R0= -----, R1= -----, A= -----, PC= -----
ADD A, R1	R0= -----, R1= -----, A= -----, PC= -----
MOV R1, A	R0= -----, R1= -----, A= -----, PC= -----
PUSH 1	R0= -----, R1= -----, SP= -----, PC= -----
PUSH 0	R0= -----, R1= -----, SP= -----, PC= -----
POP 1	R0= -----, R1= -----, SP= -----, PC= -----
POP 0	R0= -----, R1= -----, SP= -----, PC= -----
END	

Table 1-1

2. You are required to write the code which shows up-counter from 0 to 255 on Port1. Show output to your instructor.
3. Write a code which generates a square wave of central frequency 10 KHz and duty cycle of 50%. Explain mathematical calculations for time setting and show the output wave on oscilloscope

1.5 ASSESSMENT SHEET

Name: _____

Reg. No. : _____

Date of Lab: _____

Problem Number	1 -	
	2 -	
	3 -	
Lab Performance	Working	
	Viva	
Total Score in Lab		
Instructor's Verification		

Lab - 2: Seven Segment Display and Clock using Timers in 8051

2.1 OBJECTIVE

- To use subroutines and advanced commands and programming techniques
- To have knowledge about and use Timers in 8051
- Use the conventional 7-Segment display with and without the decoder IC

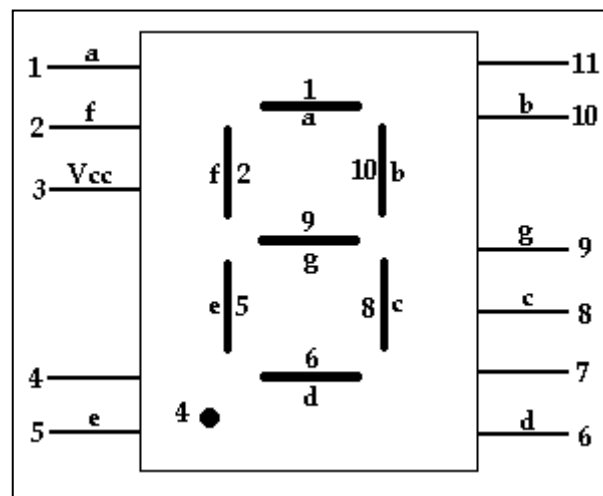


Figure 2-1: Pin Configuration

2.2 PRE-LAB READING

Following Commands from datasheet of 8051:

SJMP, LJMP, ACALL, LCALL, RET, JNB, JB, DJNZ, MOV, ADD, SUBB, INC, DEC, SET, SETB, CLR, CPL

2.3 RÉSUMÉ OF THEORY

This lab covers the control transfer instructions available in 8051 Assembly language. In the 8051, the loop action is performed by the instruction "DJNZ reg, label". Another control transfer instruction is the CALL instruction, which is used to call a subroutine. Command RET is used to return from the last called subroutine to the place of the code from where the subroutine was called.

ABOUT TIMERS:

The 8051 microcontroller has two timers: Timer 0 and Timer 1. They are used to generate time delay. Both Timer 0 and Time 1 are 16 bits wide. Since the 8051 has an 8-bit architecture, each 16-bit timer is accessed as two separate registers of low byte and high byte. In Timer use, the 8051's crystal is used as the source of the frequency.

ABOUT COUNTERS:

The 8051 microcontroller has two counters. They are used to count events happening outside the microcontroller. When timer/counter is used as a counter, it is a pulse

outside the 8051 that an increment the TH, TL registers. In counter mode, TMOD and TH, TL registers are the same as for the timer.

TIMER MODES:

There are four timer modes in 8051. A timer or counter function and modes are selected by writing appropriate bits in the SFR (special function register) , called the timer mode register (TMOD) whereas the control of timer operation is done through the SFR , called the timer control register (TCON) . These SFRs are shown in following tables.

TCON (Timer Control Register):

TCON register is bit addressable, as indicated below:

TCON.7	TCON.6	TCON.5	TCON.4	TCON.3	TCON.2	TCON.1	TCON.0
TF1	TR1	TF0	TRO	IE1	IT1	IE0	IT0

TCON.7	TF1	Timer 1 overflow flag, set when timer overflows
TCON.6	TR1	Timer 1 run control bit
TCON.5	TF0	Timer 0 overflow flag, set when timer 0 overflows
TCON.4	TRO	Timer 0 run control bit
TCON.3	IE1	Interrupt 1
TCON.2	IT1	Timer interrupt 1
TCON.1	IE0	Interrupt 0 flag
TCON.0	IT0	Timer 0 interrupt ,IT=0 , low level trigger ,IT0=1 edge trigger

TMOD (Timer Mode Register)

TMOD register is bit addressable, as indicated below:

TMOD.7	TMOD.6	TMOD.5	TMOD.4	TMOD.3	TMOD.2	TMOD.1	TMOD.0
GATE	C/T	M1	M0	GATE	C/T	M1	M0

GATE	if 1, Timer_ x is enabled , if INT_ x =1 and TR_ x=1 x=0 for timer 0 and x=1 for timer 1		
C/T	0=Timer ,1=counter		
M1 M0 Timer mode is determined by these bits.	M1	M0	
	0	0	
	0	1	
	1	0	
	1	1	

2.4 ACTIVITIES AND EXERCISES

1. Write and run an assembly language code to display 0 to 99 on two 7-segment displays using 2 output ports and without using a decoder IC. The increment should be after every 1/3 second.
2. Write a code to make a real time clock using six seven segment displays connected to 3 output ports of the microcontroller through 6 BCD to 7 Segment Decoder ICs. Use Timers to create delay.

2.5 ASSESSMENT SHEET

Name: _____

Reg. No. : _____

Date of Lab: _____

Problem Number	1 -	
	2 -	
	3 -	
Lab Performance	Working	
	Viva	
Total Score in Lab		
Instructor's Verification		

Lab - 3: Interrupts Programming and Serial Communication in 8051

3.1 OBJECTIVE

- To get familiar with Interrupts in 8051 and function of their SFRs.
- To get familiar with PWM and its real life applications in DC Motor Control
- To use the UART of 8051 for Serial Communication between PC and controller

3.2 PRE-LAB READING

- Chapter 10, 11, Mazidi, "The 8051 Microcontroller and Embedded Systems"
- <https://en.wikipedia.org/wiki/RS-232>
- https://en.wikipedia.org/wiki/Pulse-width_modulation

3.3 RÉSUMÉ OF THEORY

INTERRUPTS:

A single microcontroller can serve several devices. There are two ways to do that: interrupts or polling. In the interrupt method, whenever any device needs its service, the device notifies the microcontroller by sending it an interrupt signal. Upon receiving an interrupt signal, the microcontroller interrupts whatever it is doing and serves the device. The program associated with the interrupt is called the interrupt service routine (ISR) or interrupt handler. In polling, the microcontroller continuously monitors the status of a given device; when the status condition is met, it performs the service. After that, it moves on to monitor the next device until each one is serviced. There are five interrupts in 8051 which are shown in the following table.

Interrupt	Flag Affected	Vector	Cause Of Interrupt (If Enabled)
External interrupt 0	IE0	003H	A high to low transition on pin INT0
Timer /counter 0 interrupt	TF0	00BH	Overflow of timer/counter 0
External interrupt 1 INT pin	IE1	0013H	A high to low transition on pin INT1
Timer /counter 1 interrupt	TF1	001BH	Overflow of timer /counter 1
Serial port	RI+TI	0023H	Either TI or RI flag is set

INTERRUPT ENABLE REGISTER (IE):

The interrupt enable register allows the programmer to enable interrupts as needed. This register IE is bit addressable. Enable all (EA) bit allows disabling the whole interrupt operation. If cleared. Thus, it acts as a master control bit for any of the interrupts. The advantage of this facility is that, just by a single bit, one can control over all the interrupt sources. For any particular interrupt to occur, bit EA and corresponding bit must be set. For example in case of serial interrupt bit EA and bit ES must be set.

IE.7	IE.6	IE.5	IE.4	IE.3	IE.2	IE.1	IE.0
EA	ES	ET1	EX1	ET0	EX0

EA (Enable All)	0 = Disable all interrupts, 1= Allows each of individual interrupts to be enabled.
----------------------------	---

ES	Enable/Disable serial port interrupts, 0=Disable , 1= Enable (provided EA=1)
ET1	Enable/Disable timer interrupt 1; 0=Disable , 1= Enable (provided EA=1)
EX1	Enable/Disable External interrupt 1; 0=Disable , 1= Enable (provided EA=1)
ET0	Enable/Disable timer interrupt 0; 0=Disable , 1= Enable (provided EA=1)
EX0	Enable/Disable external interrupt 1; 0=Disable , 1= Enable (provided EA=1)

INTERRUPT PRIORITIES:

User can program the interrupt priority levels by setting or clearing the bits in SFR called interrupt priority register (IP). IP register is also bit addressable. If the bit is set then that particular interrupt will have high priority.

A high interrupt can interrupt the low priority interrupt. High priority interrupt will be served first. If the same priority level interrupts request simultaneously, then within each priority level there is a polling structure due to the inherent priority in the order. Shown in the following table

IP.7	IP.6	IP. 5	IP.4	IP.3	IP.2	IP.1	IP.0
X	X	PT2	PS	PT1	PX1	PT0	PX0

IP.5	PT2
IP.4	PS	Serial interrupt priority
IP.3	PT1	Timer 1 interrupt
IP.2	PX1	External interrupt
IP.1	PT0	Timer 0 interrupt
IP.0	PX0	External interrupt 0

Note that the priority within level structure is used only to distinguish the requests of the same priority levels. The first instruction enables both interrupts, namely, the external interrupt 1 and the timer 1 interrupt. The instruction SETB PT1 assigns high priority to the timer interrupt. So, if both of them request simultaneously, then the timer interrupt will be served first.

SERIAL COMMUNICATION:

In serial communication, the data is sent one bit at a time in contrast to parallel communication, in which the data is sent a byte or more at a time. Serial communication of 8051 is the topic of our lab. The 8051 has serial communication capability built into it, thereby making possible fast data transfer using only a few wires. 8051 supports a full duplex serial port. 8051 has TXD and RXD pins for transmission and reception of serial data respectively. The 8051 serial communication is supported by RS232 standard. Communication between two microcontrollers and multiprocessor communication is also possible. In serial transmission, baud rate is one important factor. The baud rate is reciprocal of the time to send one bit. Baud rate is need not be equal to number of bits per second .This is because each byte is preceded by a start bit and followed by a one stop bit .The start and stop bits are used to synchronize the serial receivers . The data byte is always transmitted with least

significant bit first. For error checking purpose it is possible to include a parity bit as well, just prior to the sop bit .Thus the bits are transmitted at specific time intervals determined by the baud rate. For error free serial communication it is necessary that the baud rate, the number of data bits, the number of stop bits, and the presence and absence of a parity bit along with its status be the same at transmitter and receiver ends.

The basic mechanism of serial transmission is that a data byte in a parallel form is converted into serial data stream. Along with some more bits like start, stop and parity bits, a serial data frame is sent over a line. There are four modes of serial data transmission in 8051. In each of these modes, it is important to decide the baud rate, the way in which serial data frame is sent and any other information, etc. What is common in all these modes is the use of SFR called “SBUF”, for transmission as well as reception. The data to be transmitted must be transferred to SBUF.

SCON Register:

One more SFR that controls the serial communication is the serial control register SCON. Bits SM0 and SM1 in SCON define serial port mode. Bit SM2 enables the multiprocessor communication in mode 2 and 3. Transmission is initiated by the execution of any instruction that uses SBUF as the destination.

SCON.7	SCON.6	SCON.5	SCON.4	SCON.3	SCON.2	SCON.1	SCON.0
SM0	SM1	SM2	REN	TB8	RB8	TI	RI

SCON bit	Description
SM0	Serial communication mode
SM1	Serial communication mode
SM2	In mode 2 and 3 , if set , this will enable multiprocessor communication
REN	(Receive enable) Enable serial reception
TB8	This is the 9 th data bit that is transmitted in modes 2 and 3
RB8	9 th data bit that is received in modes 2 and 3.it is not used in mode 0 .in mode 1, if SM2 =0,then RB8 is the stop bit that is received
TI	Transmit interrupt flag ,set by hardware , must be cleared by software
RI	Receive interrupt flag , set by hardware , must be cleared by software

SM0	SM1	MODE	Description	Baud Rate
0	0	0	8-bit shift register mode	$f_{osc} / 12$
0	1	1	8-bit UART	Variable (set by timer 1)
1	0	2	9-bit UART	$f_{osc} / 164$ or $f_{osc} / 32$
1	1	3	9-bit UART	Variable (set by timer 1)

3.4 ACTIVITIES AND EXERCISES

1. Write a program to control speed of a DC motor using External and Timer Interrupts. When you press the external interrupt 0 button once, the motor starts running at 50% speed, when pressed 2nd time, the speed becomes 70%, when pressed thrice, speed becomes 100% and when pressed 4th time, the motor stops and the cycle continues.
2. Write a code to send the values to the table of 3 up to multiples of 10 to the Hyper Terminal of the PC at 9600 Baud Rate using Serial Interrupt.
3. Write a code to receive data from the Hyper Terminal of PC at 9600bps and show it on P2 plus bounce it back to PC using Serial Interrupt.

3.5 ASSESSMENT SHEET

Name: _____

Reg. No. : _____

Date of Lab: _____

Problem Number	1 -	
	2 -	
	3 -	
Lab Performance	Working	
	Viva	
Total Score in Lab		
Instructor's Verification		

Lab - 4: LCD and Keypad Interfacing with 8051

4.1 OBJECTIVE

- To Interface components like LCD and KEYPAD with 8051 micro controller

4.2 PRE-LAB READING

- Chapter 12, Mazidi, "The 8051 Microcontroller and Embedded Systems"
- Data sheet of LCD and Algorithm of Keypad from the chapter.

4.3 RÉSUMÉ OF THEORY

LCD (LIQUID CRYSTAL DISPLAY):

LCD's have found widespread use in the recent years. A normal LCD has 16 pins. Out of these pins, eight are data pins, three pins are for backlight and contrast setting and two pins are for power supply. The rest three pins are control pins. Among these three pins RS is to select between commands or data register. R/W pin is to read from LCD or write on LCD. E pin is used to latch in the information present on LCD data lines. Following diagram shows LCD connected to microcontroller.

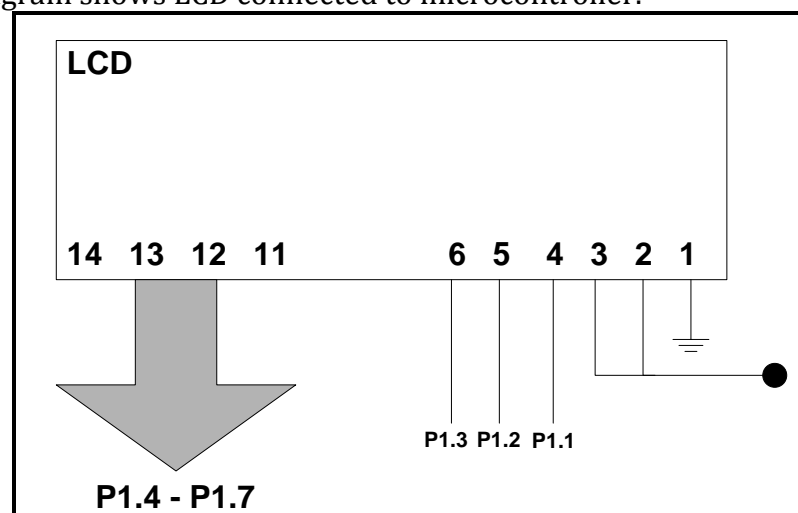


Figure 4-1

As shown in figure only four data ports are connected with microcontroller. Therefore, do not forget to initialize LCD for 4 bit data length instead of eight.

PIN DESCRIPTIONS FOR LCD:

Pin	Symbol	I/O	Description
Pin1	VSS	---	Ground
Pin2	VCC	---	+5 power supply
Pin3	VEE	---	Power supply to control contrast
Pin4	RS	I	RS=0 to select command register RS=1 to select data register
Pin5	R/W	I	R/W=0 for write R/W=1 for read
Pin6	E	I/O	Enable

Pin7	DB0	I/O	The 8 bit data bus
Pin8	DB1	I/O	The 8 bit data bus
Pin9	DB2	I/O	The 8 bit data bus
Pin10	DB3	I/O	The 8 bit data bus
Pin11	DB4	I/O	The 8 bit data bus
Pin12	DB5	I/O	The 8 bit data bus
Pin13	DB6	I/O	The 8 bit data bus
Pin14	DB7	I/O	The 8 bit data bus

To display letters and numbers 0--9 to these pins while making RS=1. There are also instruction command codes that can be sent to the LCD to clear the display or force the cursor to the home position or blink the cursor .The following table lists the instruction command codes.

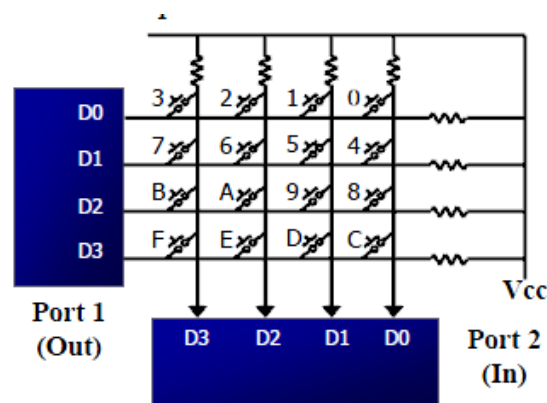
LCD COMMAND CODES:

Code (Hex)	Command to LCD instruction Register
1	Clear display screen
2	Return home
4	Decrement cursor (shift cursor to left)
5	Shift display right
6	Increment cursor (Shift cursor to 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
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 first line
C0	Force cursor to beginning of second line
38	2 lines and 5x7 matrix

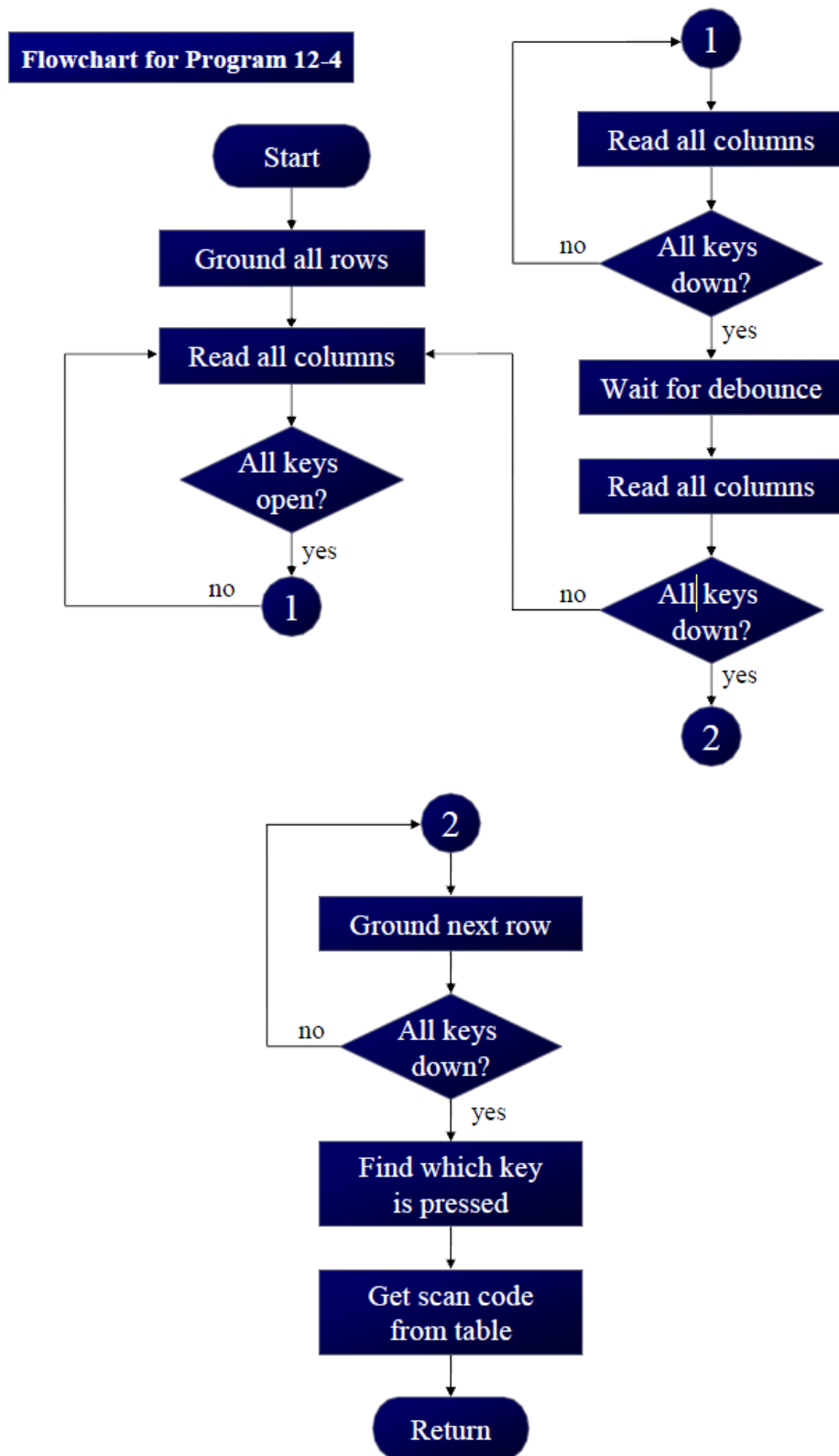
BUSY FLAG:

We use RS=0 to check the busy flag bit to see if the LCD is ready to receive the information. The busy flag is D7 and can be read when R/W=1 and RS=0. If D7 =0 the LCD is ready to receive the information otherwise LCD is busy in taking care of internal operations if R/W =1, RS=0.

KEYPAD:



Flowchart for Program 12-4



4.4 ACTIVITIES AND EXERCISES

1. Write a Program to take alpha-numeric input from the Keypad and show it on LCD

4.5 ASSESSMENT SHEET

Name: _____

Reg. No. : _____

Date of Lab: _____

Problem Number	1 -	
	2 -	
	3 -	
Lab Performance	Working	
	Viva	
Total Score in Lab		
Instructor's Verification		

Lab - 5: PIC Microcontroller Families Introduction and PIC18F4550

5.1 OBJECTIVE

To get familiar with PIC family of microcontrollers, it's features, and programming environment.

5.2 PRE-LAB READING

- Given handouts in previous lab on how to get started with MPLab.
- http://en.wikipedia.org/wiki/PIC_microcontroller

5.3 RÉSUMÉ OF THEORY

A summary of **PIC18F4550** microcontroller features is given below:

- USB V2.0 Compliant
- Interface for Off-Chip USB Transceiver
- RUN, IDLE and SLEEP Modes
- Four Crystal modes, including High Precision PLL for USB
- Internal Oscillator Block: 8 user-selectable frequencies, from 31 kHz to 8 MHz
- Three External Interrupts
- Four Timer modules (Timer0 to Timer3)
- Up to 2 Capture/Compare/PWM (CCP) modules:
 - Capture is 16-bit, max. resolution 5.2 ns (TCY/16)
 - Compare is 16-bit, max. resolution 83.3 ns (TCY)
 - PWM output: PWM resolution is 1 to 10-bit
- Enhanced Capture/Compare/PWM (ECCP) module:
 - Multiple output modes
 - Selectable polarity
 - Programmable dead time
 - Auto-shutdown and auto-restart
- Enhanced USART module:
 - LIN bus support
- Master Synchronous Serial Port (MSSP) module supporting 3-wire SPI (all 4 modes) and I2C™ Master and Slave modes
- 10-bit, up to 13-channel Analog-to-Digital Converter module (A/D) with Programmable Acquisition Time
- Dual Analog Comparators with Input Multiplexing

OSCILLATOR OPTIONS:

Users can program the FOSC3:FOSC0 Configuration bits to select one of these modes:

1. XT Crystal/Resonator
2. XTPLL Crystal/Resonator with PLL enabled
3. HS High-Speed Crystal/Resonator
4. HSPLL High-Speed Crystal/Resonator with PLL enabled

5. EC External Clock with FOSC/4 output
6. ECIO External Clock with I/O on RA6
7. ECPLL External Clock with PLL enabled and FOSC/4 output on RA6
8. ECPIO External Clock with PLL enabled, I/O on RA6
9. INTHS Internal Oscillator used as microcontroller clock source, HS Oscillator used as USB clock source
10. INTXT Internal Oscillator used as microcontroller clock source, XT Oscillator used as USB clock source
11. INTIO Internal Oscillator used as microcontroller clock source, EC Oscillator used as USB clock source, digital I/O on RA6
12. INTCKO Internal Oscillator used as microcontroller clock source, EC Oscillator used as USB clock source, FOSC/4 output on RA6

REGISTER 2-2: OSCCON: OSCILLATOR CONTROL REGISTER

R/W-0	R/W-1	R/W-0	R/W-0	R ⁽¹⁾	R-0	R/W-0	R/W-0
IDLEN	IRCF2	IRCF1	IRCF0	OSTS	IOFS	SCS1	SCS0
bit 7							bit 0

- bit 7 **IDLEN:** Idle Enable bit
 1 = Device enters Idle mode on **SLEEP** instruction
 0 = Device enters Sleep mode on **SLEEP** instruction
- bit 6-4 **IRCF2:IRCF0:** Internal Oscillator Frequency Select bits
 111 = 8 MHz (INTOSC drives clock directly)
 110 = 4 MHz
 101 = 2 MHz
 100 = 1 MHz⁽³⁾
 011 = 500 kHz
 010 = 250 kHz
 001 = 125 kHz
 000 = 31 kHz (from either INTOSC/256 or INTRC directly)⁽²⁾
- bit 3 **OSTS:** Oscillator Start-up Time-out Status bit⁽¹⁾
 1 = Oscillator Start-up Timer time-out has expired; primary oscillator is running
 0 = Oscillator Start-up Timer time-out is running; primary oscillator is not ready
- bit 2 **IOFS:** INTOSC Frequency Stable bit
 1 = INTOSC frequency is stable
 0 = INTOSC frequency is not stable
- bit 1-0 **SCS1:SCS0:** System Clock Select bits
 1x = Internal oscillator
 01 = Timer1 oscillator
 00 = Primary oscillator

(Setting the bits IRCF0, IRCF1 and IRCF2 will activate internal oscillator at 8MHz)

5.4 ACTIVITIES AND EXERCISE

1. Write a simple program to continuously flash all PORTS LEDs one by one in a line with a 0.4sec on time for each LED (LED's are active LOW on this trainer board)

5.5 ASSESSMENT SHEET

Name: _____

Reg. No. : _____

Date of Lab: _____

Problem Number	1 -	
	2 -	
	3 -	
Lab Performance	Working	
	Viva	
Total Score in Lab		
Instructor's Verification		

Lab - 6: Timers/Counters Programming in PIC18F4550

6.1 OBJECTIVE

Use the PIC18F4550 timers and the various modes in which the timers can be used.

6.2 PRE-LAB READING

- PIC18F4550 datasheet Chapter 11, 12, 13, 14 (Timer0, Timer1, Timer2, Timer3)

6.3 RÉSUMÉ OF THEORY

PIC18F4550 has four timers, 0, 1, 2 and 3. Timer 2 is an 8-bit timer while all other are 16-bit timers. All of them can be used as either a timer or a counter and all have their own peripheral interrupt sources.

TIMER 0:

REGISTER 11-1: T0CON: TIMER0 CONTROL REGISTER

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0
bit 7							bit 0

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

bit 7	TMR0ON: Timer0 On/Off Control bit 1 = Enables Timer0 0 = Stops Timer0
bit 6	T08BIT: Timer0 8-Bit/16-Bit Control bit 1 = Timer0 is configured as an 8-bit timer/counter 0 = Timer0 is configured as a 16-bit timer/counter
bit 5	T0CS: Timer0 Clock Source Select bit 1 = Transition on T0CKI pin 0 = Internal instruction cycle clock (CLKO)
bit 4	T0SE: Timer0 Source Edge Select bit 1 = Increment on high-to-low transition on T0CKI pin 0 = Increment on low-to-high transition on T0CKI pin
bit 3	PSA: Timer0 Prescaler Assignment bit 1 = Timer0 prescaler is NOT assigned. Timer0 clock input bypasses prescaler. 0 = Timer0 prescaler is assigned. Timer0 clock input comes from prescaler output.
bit 2-0	T0PS2:T0PS0: Timer0 Prescaler Select bits 111 = 1:256 Prescale value 110 = 1:128 Prescale value 101 = 1:64 Prescale value 100 = 1:32 Prescale value 011 = 1:16 Prescale value 010 = 1:8 Prescale value 001 = 1:4 Prescale value 000 = 1:2 Prescale value

(TMR0L and TMR0H are the Timer 0 8-bit Registers that hold the value of the present count of timer.)

TABLE 11-1: REGISTERS ASSOCIATED WITH TIMER0

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
TMR0L	Timer0 Register Low Byte								52
TMR0H	Timer0 Register High Byte								52
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
INTCON2	RBPUR	INTEDG0	INTEDG1	INTEDG2	—	TMR0IP	—	RBIP	51
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	52
TRISA	—	TRISA6 ⁽¹⁾	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	54

TIMER 1:**REGISTER 12-1: T1CON: TIMER1 CONTROL REGISTER**

R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNCR	TMR1CS	TMR1ON
bit 7							bit 0

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

bit 7	RD16: 16-Bit Read/Write Mode Enable bit 1 = Enables register read/write of Timer1 in one 16-bit operation 0 = Enables register read/write of Timer1 in two 8-bit operations
bit 6	T1RUN: Timer1 System Clock Status bit 1 = Device clock is derived from Timer1 oscillator 0 = Device clock is derived from another source
bit 5-4	T1CKPS1:T1CKPS0: Timer1 Input Clock Prescale Select bits 11 = 1:8 Prescale value 10 = 1:4 Prescale value 01 = 1:2 Prescale value 00 = 1:1 Prescale value
bit 3	T1OSCEN: Timer1 Oscillator Enable bit 1 = Timer1 oscillator is enabled 0 = Timer1 oscillator is shut off The oscillator inverter and feedback resistor are turned off to eliminate power drain.
bit 2	T1SYNCR: Timer1 External Clock Input Synchronization Select bit When TMR1CS = 1: 1 = Do not synchronize external clock input 0 = Synchronize external clock input When TMR1CS = 0: This bit is ignored. Timer1 uses the internal clock when TMR1CS = 0.
bit 1	TMR1CS: Timer1 Clock Source Select bit 1 = External clock from RC0/T1OSO/T13CKI pin (on the rising edge) 0 = Internal clock (Fosc/4)
bit 0	TMR1ON: Timer1 On bit 1 = Enables Timer1 0 = Stops Timer1

TABLE 12-2: REGISTERS ASSOCIATED WITH TIMER1 AS A TIMER/COUNTER

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
PIR1	SPPIF ⁽¹⁾	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	54
PIE1	SPPIE ⁽¹⁾	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	54
IPR1	SPPIP ⁽¹⁾	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	54
TMR1L	Timer1 Register Low Byte								52
TMR1H	Timer1 Register High Byte								52
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNCR	TMR1CS	TMR1ON	52

TIMER 2:

REGISTER 13-1: T2CON: TIMER2 CONTROL REGISTER

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7							bit 0

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

bit 7 **Unimplemented:** Read as '0'
 bit 6-3 **T2OUTPS3:T2OUTPS0:** Timer2 Output Postscale Select bits
 0000 = 1:1 Postscale
 0001 = 1:2 Postscale
 •
 •
 •
 1111 = 1:16 Postscale
 bit 2 **TMR2ON:** Timer2 On bit
 1 = Timer2 is on
 0 = Timer2 is off
 bit 1-0 **T2CKPS1:T2CKPS0:** Timer2 Clock Prescale Select bits
 00 = Prescaler is 1
 01 = Prescaler is 4
 1x = Prescaler is 16

TABLE 13-1: REGISTERS ASSOCIATED WITH TIMER2 AS A TIMER/COUNTER

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	51
PIR1	SPPIF ⁽¹⁾	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	54
PIE1	SPPIE ⁽¹⁾	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	54
IPR1	SPPIP ⁽¹⁾	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	54
TMR2	Timer2 Register								52
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	52
PR2	Timer2 Period Register								52

6.4 ACTIVITIES AND EXERCISE

1. Write a program to create delay of 1sec using Timer 1 and flash LEDs of PORTD.
2. Write a program to use Timer 2 as a Counter and continuously display the count on PORTB.
3. Write a program to measure frequency of a waveform (1Hz to 10KHz) and display the result on PORTB.

6.5 ASSESSMENT SHEET

Name: _____

Reg. No. : _____

Date of Lab: _____

Problem Number	1 -	
	2 -	
	3 -	
Lab Performance	Working	
	Viva	
Total Score in Lab		
Instructor's Verification		

Lab - 7: Interrupts Programming in PIC18F4550

7.1 OBJECTIVE

To get familiar with the interrupt sources present in PIC18F4550

7.2 PRE-LAB READING

- PIC18F4550 datasheet Chapter 9

7.3 RÉSUMÉ OF THEORY

PIC18F4550 has various hardware and software interrupt sources like External Interrupts, Timer Interrupts, Serial Interrupt, USB Interrupt, ADC Interrupt etc.

The interrupts like those of Timers, USB and ADC are called Peripheral Interrupts and the others are general interrupts. To enable interrupts, the Global Interrupt Enable (GIE) bit has to be set and to enable peripheral interrupts, Peripheral Interrupt Enable (PEIE) bit has to be set.

Following are the interrupt enable bits and interrupt flags of various sources:

- **TIMER 0:** TMR0IE (Timer 0 Interrupt Enable), TMR0IF (Timer 0 Interrupt Flag)
- **TIMER 1:** TMR1IE (Timer 1 Interrupt Enable), TMR1IF (Timer 1 Interrupt Flag)
- **TIMER 2:** TMR2IE (Timer 2 Interrupt Enable), TMR2IF (Timer 2 Interrupt Flag)
- **TIMER 3:** TMR3IE (Timer 3 Interrupt Enable), TMR3IF (Timer 3 Interrupt Flag)
- **ADC:** ADIE (A/D Interrupt Enable), ADIF (A/D Interrupt Flag)
- **External Interrupt 0:** INT0IE (Interrupt Enable), INT0IF (Interrupt Flag)
- **External Interrupt 1:** INT1IE (Interrupt Enable), INT1IF (Interrupt Flag)
- **External Interrupt 2:** INT2IE (Interrupt Enable), INT2IF (Interrupt Flag)

There are two Interrupt Service Routines (ISR) for the interrupts in PIC. One is low_ISR (vector address 0x018) for low priority interrupts and the other one is high_ISR (vector address 0x008) for high priority interrupts (This facility is NOT available in HiTech Compiler). If no priority is set for any interrupt or just one interrupt is being used, we can use the general ISR without mentioning high or low. User has to check manually using the Interrupt Indicating Flags within the ISR that which interrupt has caused the program to jump to ISR.

The format for writing the ISR is as follows:

```
void interrupt ISR(void) //general interrupt ISR
{
    if(ADIF = 1)
        {GO = 1;
        PORTD = ADRESH;
        }
}
```

REGISTER 9-1: INTCON: INTERRUPT CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF ⁽¹⁾
bit 7							bit 0

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

bit 7	GIE/GIEH: Global Interrupt Enable bit <u>When IPEN = 0:</u> 1 = Enables all unmasked interrupts 0 = Disables all interrupts <u>When IPEN = 1:</u> 1 = Enables all high priority interrupts 0 = Disables all high priority interrupts
bit 6	PEIE/GIEL: Peripheral Interrupt Enable bit <u>When IPEN = 0:</u> 1 = Enables all unmasked peripheral interrupts 0 = Disables all peripheral interrupts <u>When IPEN = 1:</u> 1 = Enables all low priority peripheral interrupts 0 = Disables all low priority peripheral interrupts
bit 5	TMR0IE: TMR0 Overflow Interrupt Enable bit 1 = Enables the TMR0 overflow interrupt 0 = Disables the TMR0 overflow interrupt
bit 4	INT0IE: INT0 External Interrupt Enable bit 1 = Enables the INT0 external interrupt 0 = Disables the INT0 external interrupt
bit 3	RBIE: RB Port Change Interrupt Enable bit 1 = Enables the RB port change interrupt 0 = Disables the RB port change interrupt
bit 2	TMR0IF: TMR0 Overflow Interrupt Flag bit 1 = TMR0 register has overflowed (must be cleared in software) 0 = TMR0 register did not overflow
bit 1	INT0IF: INT0 External Interrupt Flag bit 1 = The INT0 external interrupt occurred (must be cleared in software) 0 = The INT0 external interrupt did not occur
bit 0	RBIF: RB Port Change Interrupt Flag bit ⁽¹⁾ 1 = At least one of the RB7:RB4 pins changed state (must be cleared in software) 0 = None of the RB7:RB4 pins have changed state

REGISTER 9-2: INTCON2: INTERRUPT CONTROL REGISTER 2

R/W-1	R/W-1	R/W-1	R/W-1	U-0	R/W-1	U-0	R/W-1
$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	—	TMR0IP	—	RBIP
bit 7							bit 0

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

- bit 7 **$\overline{\text{RBPU}}$** : PORTB Pull-up Enable bit
 1 = All PORTB pull-ups are disabled
 0 = PORTB pull-ups are enabled by individual port latch values
- bit 6 **INTEDG0**: External Interrupt 0 Edge Select bit
 1 = Interrupt on rising edge
 0 = Interrupt on falling edge
- bit 5 **INTEDG1**: External Interrupt 1 Edge Select bit
 1 = Interrupt on rising edge
 0 = Interrupt on falling edge
- bit 4 **INTEDG2**: External Interrupt 2 Edge Select bit
 1 = Interrupt on rising edge
 0 = Interrupt on falling edge
- bit 3 **Unimplemented**: Read as '0'
- bit 2 **TMR0IP**: TMR0 Overflow Interrupt Priority bit
 1 = High priority
 0 = Low priority
- bit 3 **Unimplemented**: Read as '0'
- bit 2 **TMR0IP**: TMR0 Overflow Interrupt Priority bit
 1 = High priority
 0 = Low priority
- bit 1 **Unimplemented**: Read as '0'
- bit 0 **RBIP**: RB Port Change Interrupt Priority bit
 1 = High priority
 0 = Low priority

REGISTER 9-3: INTCON3: INTERRUPT CONTROL REGISTER 3

R/W-1	R/W-1	U-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
INT2IP	INT1IP	—	INT2IE	INT1IE	—	INT2IF	INT1IF
bit 7							bit 0

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

bit 7	INT2IP: INT2 External Interrupt Priority bit 1 = High priority 0 = Low priority
bit 6	INT1IP: INT1 External Interrupt Priority bit 1 = High priority 0 = Low priority
bit 5	Unimplemented: Read as '0'
bit 4	INT2IE: INT2 External Interrupt Enable bit 1 = Enables the INT2 external interrupt 0 = Disables the INT2 external interrupt
bit 3	INT1IE: INT1 External Interrupt Enable bit 1 = Enables the INT1 external interrupt 0 = Disables the INT1 external interrupt
bit 2	Unimplemented: Read as '0'
bit 1	INT2IF: INT2 External Interrupt Flag bit 1 = The INT2 external interrupt occurred (must be cleared in software) 0 = The INT2 external interrupt did not occur
bit 0	INT1IF: INT1 External Interrupt Flag bit 1 = The INT1 external interrupt occurred (must be cleared in software) 0 = The INT1 external interrupt did not occur

9.9 PORTB Interrupt-on-Change

An input change on PORTB<7:4> sets flag bit, RBIF (INTCON<0>). The interrupt can be enabled/disabled by setting/clearing enable bit, RBIE (INTCON<3>). Interrupt priority for PORTB interrupt-on-change is determined by the value contained in the interrupt priority bit, RBIP (INTCON2<0>).

7.4 ACTIVITIES AND EXERCISE

1. Write a program to count the number of people entering and leaving a room using two IR sensors at the gate. Use this data to turn on lights if anyone is inside the room and turn them off when the room is empty. Also light and LED indicating that the room is full when 20 people are inside. Use timer/counter interrupts and/or external interrupts to perform the task.

7.5 ASSESSMENT SHEET

Name: _____

Reg. No. : _____

Date of Lab: _____

Problem Number	1 -	
	2 -	
	3 -	
Lab Performance	Working	
	Viva	
Total Score in Lab		
Instructor's Verification		

Lab - 8: ADC Programming in PIC18F4550

8.1 OBJECTIVE

To get familiar with built-in ADC module of PIC18F4550 and its features.

8.2 PRE-LAB READING

- PIC18F4550 datasheet Chapter 21

8.3 RÉSUMÉ OF THEORY

PIC18F4550 has a 13-Channel built-in ADC with a resolution of 10-bits for each channel. The ADC digital result is stored in two registers, ADRESH and ADRESL. ADC has its own peripheral interrupt source that can inform about completion of A/D conversion.

The module has five registers:

- A/D Result High Register (ADRESH)
- A/D Result Low Register (ADRESL)
- A/D Control Register 0 (ADCON0)
- A/D Control Register 1 (ADCON1)
- A/D Control Register 2 (ADCON2)

(ADRESH and ADRESL are the registers in which the converted digital data is stored.)

REGISTER 21-1: ADCON0: A/D CONTROL REGISTER 0

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	CHS3	CHS2	CHS1	CHS0	GO/DONE	ADON
bit 7							bit 0

bit 7-6	Unimplemented: Read as '0'
bit 5-2	CHS3:CHS0: Analog Channel Select bits 0000 = Channel 0 (AN0) 0001 = Channel 1 (AN1) 0010 = Channel 2 (AN2) 0011 = Channel 3 (AN3) 0100 = Channel 4 (AN4) 0101 = Channel 5 (AN5) ^(1,2) 0110 = Channel 6 (AN6) ^(1,2) 0111 = Channel 7 (AN7) ^(1,2) 1000 = Channel 8 (AN8) 1001 = Channel 9 (AN9) 1010 = Channel 10 (AN10) 1011 = Channel 11 (AN11) 1100 = Channel 12 (AN12) 1101 = Unimplemented ⁽²⁾ 1110 = Unimplemented ⁽²⁾ 1111 = Unimplemented ⁽²⁾
bit 1	GO/DONE: A/D Conversion Status bit <u>When ADON = 1:</u> 1 = A/D conversion in progress 0 = A/D Idle
bit 0	ADON: A/D On bit 1 = A/D converter module is enabled 0 = A/D converter module is disabled

REGISTER 21-2: ADCON1: A/D CONTROL REGISTER 1

U-0	U-0	R/W-0	R/W-0	R/W-0 ⁽¹⁾	R/W ⁽¹⁾	R/W ⁽¹⁾	R/W ⁽¹⁾
—	—	VCFG0	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

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

bit 7-6 **Unimplemented:** Read as '0'

bit 5 **VCFG0:** Voltage Reference Configuration bit (VREF- source)

1 = VREF- (AN2)

0 = VSS

bit 4 **VCFG0:** Voltage Reference Configuration bit (VREF+ source)

1 = VREF+ (AN3)

0 = VDD

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

PCFG3: PCFG0	AN12	AN11	AN10	AN9	AN8	AN7 ⁽²⁾	AN6 ⁽²⁾	AN5 ⁽²⁾	AN4	AN3	AN2	AN1	AN0
0000 ⁽¹⁾	A	A	A	A	A	A	A	A	A	A	A	A	A
0001	A	A	A	A	A	A	A	A	A	A	A	A	A
0010	A	A	A	A	A	A	A	A	A	A	A	A	A
0011	D	A	A	A	A	A	A	A	A	A	A	A	A
0100	D	D	A	A	A	A	A	A	A	A	A	A	A
0101	D	D	D	A	A	A	A	A	A	A	A	A	A
0110	D	D	D	D	A	A	A	A	A	A	A	A	A
0111 ⁽¹⁾	D	D	D	D	D	A	A	A	A	A	A	A	A
1000	D	D	D	D	D	D	A	A	A	A	A	A	A
1001	D	D	D	D	D	D	D	A	A	A	A	A	A
1010	D	D	D	D	D	D	D	D	A	A	A	A	A
1011	D	D	D	D	D	D	D	D	D	A	A	A	A
1100	D	D	D	D	D	D	D	D	D	D	A	A	A
1101	D	D	D	D	D	D	D	D	D	D	D	A	A
1110	D	D	D	D	D	D	D	D	D	D	D	D	A
1111	D	D	D	D	D	D	D	D	D	D	D	D	D

A = Analog input

D = Digital I/O

REGISTER 21-3: ADCON2: A/D CONTROL REGISTER 2

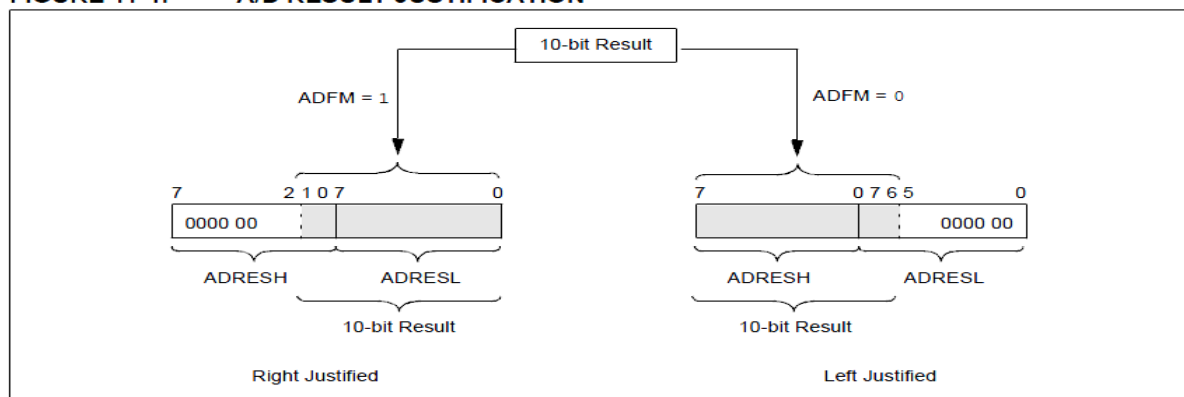
R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0
bit 7							bit 0

- bit 7 **ADFM**: A/D Result Format Select bit
 1 = Right justified
 0 = Left justified
- bit 6 **Unimplemented**: Read as '0'
- bit 5-3 **ACQT2:ACQT0**: A/D Acquisition Time Select bits
 111 = 20 TAD
 110 = 16 TAD
 101 = 12 TAD
 100 = 8 TAD
 011 = 6 TAD
 010 = 4 TAD
 001 = 2 TAD
 000 = 0 TAD⁽¹⁾
- bit 2-0 **ADCS2:ADCS0**: A/D Conversion Clock Select bits
 111 = FRC (clock derived from A/D RC oscillator)⁽¹⁾
 110 = FOSC/64
 101 = FOSC/16
 100 = FOSC/4
 011 = FRC (clock derived from A/D RC oscillator)⁽¹⁾
 010 = FOSC/32
 001 = FOSC/8
 000 = FOSC/2

The following steps should be followed to perform an A/D conversion:

- Configure the A/D module:
 - Configure analog pins, voltage reference and digital I/O (ADCON1)
 - Select A/D input channel (ADCON0)
 - Select A/D acquisition time (ADCON2)
 - Select A/D conversion clock (ADCON0)
 - Turn on A/D module (ADCON0)
- Configure A/D interrupt (if desired):
 - Clear ADIF bit
 - Set ADIE bit
 - Set GIE bit
- Wait the required acquisition time (if required).
- Start conversion:
 - Set GO/DONE bit (ADCON0 register)
- Wait for A/D conversion to complete, by either:
 - Polling for the GO/DONE bit to be cleared
 OR
 - Waiting for the A/D interrupt
- Read A/D Result registers (ADRESH:ADRESL); clear bit ADIF, if required.
- For next conversion, go to step 1 or step 2, as required. The A/D conversion time per bit is defined as TAD. A minimum wait of 3 TAD is required before the next acquisition starts.

FIGURE 11-4: A/D RESULT JUSTIFICATION



8.4 ACTIVITIES AND EXERCISE

1. Use ADC interrupt to continuously send upper 8-bits of digital data from Channel 0 to PORTB, Channel 1 to PORTC and Channel 2 to PORTD.

8.5 ASSESSMENT SHEET

Problem Number	1 -	
Lab Performance	Working	
	Viva	
Total Score in Lab		
Instructor's Verification		

Lab - 9: Capture Compare PWM (CCP) Programming in PIC18F4550

9.1 OBJECTIVE

To know what is Capture/Compare/PWM is and use the CCP module of PIC18F4550

9.2 PRE-LAB READING

- PIC18F4550 datasheet Chapter 15, 16

9.3 RÉSUMÉ OF THEORY

PIC18F4550 has one built-in ECCP and one CCP module. These provide the user with an opportunity to “Capture/Read” the instantaneous value of Timer upon a rising or falling edge on an external pin, or, to set/clear an output pin upon comparing the value of timer with our desired given value, or, to generate a variable PWM out.

Each Capture/Compare/PWM (CCP) module contains
a 16-bit register which can operate as a:

- 16-bit Capture register
- 16-bit Compare register
- PWM Master/Slave Duty Cycle register

TABLE 15-2: INTERACTIONS BETWEEN CCP1 AND CCP2 FOR TIMER RESOURCES

CCP1 Mode	CCP2 Mode	Interaction
Capture	Capture	Each module can use TMR1 or TMR3 as the time base. The time base can be different for each CCP.
Capture	Compare	CCP2 can be configured for the Special Event Trigger to reset TMR1 or TMR3 (depending upon which time base is used). Automatic A/D conversions on trigger event can also be done. Operation of CCP1 could be affected if it is using the same timer as a time base.
Compare	Capture	CCP1 be configured for the Special Event Trigger to reset TMR1 or TMR3 (depending upon which time base is used). Operation of CCP2 could be affected if it is using the same timer as a time base.
Compare	Compare	Either module can be configured for the Special Event Trigger to reset the time base. Automatic A/D conversions on CCP2 trigger event can be done. Conflicts may occur if both modules are using the same time base.
Capture	PWM ⁽¹⁾	None
Compare	PWM ⁽¹⁾	None
PWM ⁽¹⁾	Capture	None
PWM ⁽¹⁾	Compare	None
PWM ⁽¹⁾	PWM	Both PWMs will have the same frequency and update rate (TMR2 interrupt).

REGISTER 15-1: CCPxCON: STANDARD CCPx CONTROL REGISTER

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
— ⁽¹⁾	— ⁽¹⁾	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0
bit 7							bit 0

bit 7-6 **Unimplemented:** Read as '0'⁽¹⁾

bit 5-4 **DCxB1:DCxB0:** PWM Duty Cycle Bit 1 and Bit 0 for CCPx Module

Capture mode:

Unused.

Compare mode:

Unused.

PWM mode:

These bits are the two LSBs (bit 1 and bit 0) of the 10-bit PWM duty cycle. The eight MSBs of the duty cycle are found in CCPR1L.

bit 3-0 **CCPxM3:CCPxM0:** CCPx Module Mode Select bits

0000 = Capture/Compare/PWM disabled (resets CCPx module)

0001 = Reserved

0010 = Compare mode: toggle output on match (CCPxIF bit is set)

0011 = Reserved

0100 = Capture mode: every falling edge

0101 = Capture mode: every rising edge

0110 = Capture mode: every 4th rising edge

0111 = Capture mode: every 16th rising edge

1000 = Compare mode: initialize CCPx pin low; on compare match, force CCPx pin high (CCPxIF bit is set)

1001 = Compare mode: initialize CCPx pin high; on compare match, force CCPx pin low (CCPxIF bit is set)

1010 = Compare mode: generate software interrupt on compare match (CCPxIF bit is set, CCPx pin reflects I/O state)

1011 = Compare mode: trigger special event, reset timer, start A/D conversion on CCP2 match (CCPxIF bit is set)

11xx = PWM mode

15.4.1 PWM PERIOD

The PWM period is specified by writing to the PR2 register. The PWM period can be calculated using the following formula:

EQUATION 15-1:

$$\text{PWM Period} = [(PR2) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value})$$

$$\text{PWM Resolution (max)} = \frac{\log\left(\frac{F_{OSC}}{F_{PWM}}\right)}{\log(2)} \text{ bits}$$

15.4.2 PWM DUTY CYCLE

The PWM duty cycle is specified by writing to the CCPRxL register and to the CCPxCON<5:4> bits. Up to 10-bit resolution is available. The CCPRxL contains the eight MSBs and the CCPxCON<5:4> bits contain the two LSBs. This 10-bit value is represented by CCPRxL:CCPxCON<5:4>. The following equation is used to calculate the PWM duty cycle in time:

EQUATION 15-2:

$$\text{PWM Duty Cycle} = (\text{CCPRxL:CCPxCON<5:4>}) \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value})$$

CCPRxL and CCPxCON<5:4> can be written to at any time, but the duty cycle value is not latched into CCPRxH until after a match between PR2 and TMR2 occurs (i.e., the period is complete). In PWM mode, CCPRxH is a read-only register.

TABLE 15-4: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 40 MHz

PWM Frequency	2.44 kHz	9.77 kHz	39.06 kHz	156.25 kHz	312.50 kHz	416.67 kHz
Timer Prescaler (1, 4, 16)	16	4	1	1	1	1
PR2 Value	FFh	FFh	FFh	3Fh	1Fh	17h
Maximum Resolution (bits)	10	10	10	8	7	6.58

15.4.4 SETUP FOR PWM OPERATION

The following steps should be taken when configuring the CCP module for PWM operation:

1. Set the PWM period by writing to the PR2 register.
2. Set the PWM duty cycle by writing to the CCPRxL register and CCPxCON<5:4> bits.
3. Make the CCPx pin an output by clearing the appropriate TRIS bit.
4. Set the TMR2 prescale value, then enable Timer2 by writing to T2CON.
5. Configure the CCPx module for PWM operation.

9.4 ACTIVITIES AND EXERCISE

1. Use ADC interrupt and CCP1 & CCP2 Modules to design a variable PWM Duty-Cycle system that captures the ADC value from Channel 0 and Channel 1 and use them as the duty cycle of PWM1 and PWM2 respectively after a proper scaling factor.
2. Use the CCP1 module in Capture mode (falling edge) and CCP2 module in Compare mode (set output) and show your results on any suitable port.

9.5 ASSESSMENT SHEET

Name: _____

Reg. No. : _____

Date of Lab: _____

Problem Number	1 -	
	2 -	
	3 -	
Lab Performance	Working	
	Viva	
Total Score in Lab		
Instructor's Verification		

Lab - 10: Serial Communication using PIC18F4550

10.1 OBJECTIVE

To use the USART in PIC18F4550 for Serial Communication with PC

10.2 PRE-LAB READING

- PIC18F4550 datasheet Chapter 20

10.3 RÉSUMÉ OF THEORY

PIC18F4550 has a built-in USART (Universal Synchronous Asynchronous Receiver Transmitter). It is one of the two Serial Communication Modules of PIC18F4550 (2nd being the MSSP).

The EUSART module includes the following capabilities:

- Full-duplex asynchronous transmit and receive
- Two-character input buffer
- One-character output buffer
- Programmable 8-bit or 9-bit character length
- Address detection in 9-bit mode
- Input buffer overrun error detection
- Received character framing error detection
- Half-duplex synchronous master
- Half-duplex synchronous slave
- Programmable clock polarity in synchronous modes

The EUSART module implements the following additional features, making it ideally suited for use in Local Interconnect Network (LIN) bus systems:

- Automatic detection and calibration of the baud rate
- Wake-up on Break reception
- 13-bit Break character transmit

REGISTER 20-1: TXSTA: TRANSMIT STATUS AND CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN ⁽¹⁾	SYNC	SENDB	BRGH	TRMT	TX9D
bit 7							bit 0

bit 7	CSRC: Clock Source Select bit <u>Asynchronous mode:</u> Don't care. <u>Synchronous mode:</u> 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⁾ 1 = Transmit enabled 0 = Transmit disabled
bit 4	SYNC: EUSART Mode Select bit 1 = Synchronous mode 0 = Asynchronous mode
bit 3	SENDB: Send Break Character bit <u>Asynchronous mode:</u> 1 = Send Sync Break on next transmission (cleared by hardware upon completion) 0 = Sync Break transmission completed <u>Synchronous mode:</u> Don't care.
bit 2	BRGH: High Baud Rate Select bit <u>Asynchronous mode:</u> 1 = High speed 0 = Low speed <u>Synchronous mode:</u> 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 address/data bit or a parity bit.

REGISTER 20-2: RCSTA: RECEIVE STATUS AND CONTROL REGISTER

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 RX/DT and TX/CK pins as serial port pins) 0 = Serial port disabled (held in Reset)
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 <u>Asynchronous mode:</u> Don't care. <u>Synchronous mode – Master:</u> 1 = Enables single receive 0 = Disables single receive This bit is cleared after reception is complete. <u>Synchronous mode – Slave:</u> Don't care.
bit 4	CREN: Continuous Receive Enable bit <u>Asynchronous mode:</u> 1 = Enables receiver 0 = Disables receiver <u>Synchronous mode:</u> 1 = Enables continuous receive until enable bit CREN is cleared (CREN overrides SREN) 0 = Disables continuous receive
bit 3	ADDEN: Address Detect Enable bit <u>Asynchronous mode 9-bit (RX9 = 1):</u> 1 = Enables address detection, enables interrupt and loads 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 <u>Asynchronous mode 9-bit (RX9 = 0):</u> Don't care.
bit 2	FERR: Framing Error bit 1 = Framing error (can be updated by reading RCREG register and receiving 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 This can be address/data bit or a parity bit and must be calculated by user firmware.

REGISTER 20-3: BAUDCON: BAUD RATE CONTROL REGISTER

R/W-0	R-1	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
bit 7							bit 0

- bit 7 **ABDOVF:** Auto-Baud Acquisition Rollover Status bit
1 = A BRG rollover has occurred during Auto-Baud Rate Detect mode (must be cleared in software)
0 = No BRG rollover has occurred
- bit 6 **RCIDL:** Receive Operation Idle Status bit
1 = Receive operation is Idle
0 = Receive operation is active
- bit 5 **RXDTP:** Received Data Polarity Select bit
Asynchronous mode:
1 = RX data is inverted
0 = RX data received is not inverted
Synchronous modes:
1 = CK clocks are inverted
0 = CK clocks are not inverted
- bit 4 **TXCKP:** Clock and Data Polarity Select bit
Asynchronous mode:
1 = TX data is inverted
0 = TX data is not inverted
Synchronous modes:
1 = CK clocks are inverted
0 = CK clocks are not inverted
- bit 3 **BRG16:** 16-Bit Baud Rate Register Enable bit
1 = 16-bit Baud Rate Generator – SPBRGH and SPBRG
0 = 8-bit Baud Rate Generator – SPBRG only (Compatible mode), SPBRGH value ignored
- bit 2 **Unimplemented:** Read as '0'
- bit 1 **WUE:** Wake-up Enable bit
Asynchronous mode:
1 = EUSART will continue to sample the RX pin – interrupt generated on falling edge; bit cleared in hardware on following rising edge
0 = RX pin not monitored or rising edge detected
Synchronous mode:
Unused in this mode.
- bit 0 **ABDEN:** Auto-Baud Detect Enable bit
Asynchronous mode:
1 = Enable baud rate measurement on the next character. Requires reception of a Sync field (55h); cleared in hardware upon completion.
0 = Baud rate measurement disabled or completed
Synchronous mode:
Unused in this mode.

TABLE 20-1: BAUD RATE FORMULAS

Configuration Bits			BRG/EUSART Mode	Baud Rate Formula
SYNC	BRG16	BRGH		
0	0	0	8-bit/Asynchronous	$F_{osc}/[64 (n + 1)]$
0	0	1	8-bit/Asynchronous	$F_{osc}/[16 (n + 1)]$
0	1	0	16-bit/Asynchronous	
0	1	1	16-bit/Asynchronous	$F_{osc}/[4 (n + 1)]$
1	0	x	8-bit/Synchronous	
1	1	x	16-bit/Synchronous	

Legend: x = Don't care, n = value of SPBRGH:SPBRG register pair

EXAMPLE 20-1: CALCULATING BAUD RATE ERROR

For a device with F_{osc} of 16 MHz, desired baud rate of 9600, Asynchronous mode, 8-bit BRG:

Desired Baud Rate = $F_{osc}/(64 ([SPBRGH:SPBRG] + 1))$

Solving for SPBRGH:SPBRG:

$$X = ((F_{osc}/\text{Desired Baud Rate})/64) - 1$$

$$= ((16000000/9600)/64) - 1$$

$$= [25.042] = 25$$

Calculated Baud Rate = $16000000/(64 (25 + 1))$

$$= 9615$$

Error = $(\text{Calculated Baud Rate} - \text{Desired Baud Rate})/\text{Desired Baud Rate}$

$$= (9615 - 9600)/9600 = 0.16\%$$

TABLE 20-3: BAUD RATES FOR ASYNCHRONOUS MODES

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 0											
	Fosc = 40.000 MHz			Fosc = 20.000 MHz			Fosc = 10.000 MHz			Fosc = 8.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	—	—	—	—	—	—	—	—	—	—	—	—
1.2	—	—	—	1.221	1.73	255	1.202	0.16	129	1201	-0.16	103
2.4	2.441	1.73	255	2.404	0.16	129	2.404	0.16	64	2403	-0.16	51
9.6	9.615	0.16	64	9.766	1.73	31	9.766	1.73	15	9615	-0.16	12
19.2	19.531	1.73	31	19.531	1.73	15	19.531	1.73	7	—	—	—
57.6	56.818	-1.36	10	62.500	8.51	4	52.083	-9.58	2	—	—	—
115.2	125.000	8.51	4	104.167	-9.58	2	78.125	-32.18	1	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 0								
	Fosc = 4.000 MHz			Fosc = 2.000 MHz			Fosc = 1.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.16	207	300	-0.16	103	300	-0.16	51
1.2	1.202	0.16	51	1201	-0.16	25	1201	-0.16	12
2.4	2.404	0.16	25	2403	-0.16	12	—	—	—
9.6	8.929	-6.99	6	—	—	—	—	—	—
19.2	20.833	8.51	2	—	—	—	—	—	—
57.6	62.500	8.51	0	—	—	—	—	—	—
115.2	62.500	-45.75	0	—	—	—	—	—	—

SAMPLE PROGRAM:

```
#include <htc.h>
void msdelay(unsigned int ms);
void main(void)
{
    SPEN=1;
    TXEN=1;
    SYNC=0;
    BRGH=1;
    CREN=1;
    SPBRG=25;
    TXREG=0x01;
    while(1)
    {
        while(TXIF==0);
        TXREG++;
    }
}
```

This program continuously sends incrementing values from 0-255 to PC via Serial Port.

10.4 ACTIVITIES AND EXERCISE

1. Write a Program to Read a Sin Wave of frequency (50Hz) at ADC Channel 0 using interrupt and continuously send the digital values to PC using Serial Interrupt. Calculate the Frequency of the input Sine Wave and Also Send it to PC Serially after 10 seconds.

10.5 ASSESSMENT SHEET

Name: _____

Reg. No. : _____

Date of Lab: _____

Problem Number	1 -	
	2 -	
	3 -	
Lab Performance	Working	
	Viva	
Total Score in Lab		
Instructor's Verification		

Lab - 11: USB Communication using PIC18F4550

11.1 OBJECTIVE

11.2 PRE-LAB READING

11.3 RÉSUMÉ OF THEORY

11.4 ACTIVITIES AND EXERCISE

11.5 ASSESSMENT SHEET

Name: _____

Reg. No. : _____

Date of Lab: _____

Problem Number	1 -	
	2 -	
	3 -	
Lab Performance	Working	
	Viva	
Total Score in Lab		
Instructor's Verification		

Lab - 12: Doing a joint small project in lab

12.1 OBJECTIVE

To make a joint project within lab addressing a real life task/problem using microcontrollers and interfacing circuits.

12.2 PRE-LAB READING

Data to be provided in the last lab.

12.3 RÉSUMÉ OF THEORY

Data to be provided in the last lab.

12.4 ACTIVITIES AND EXERCISE

1. To be announced on spot or in the last lab.

12.5 ASSESSMENT SHEET

Name: _____

Reg. No. : _____

Date of Lab: _____

Problem Number	1 -	
	2 -	
	3 -	
Lab Performance	Working	
	Viva	
Total Score in Lab		
Instructor's Verification		

Micro Lab Assessment

Rate the following out of 10 (for 10 being the best)

LAB ASSESSMENT:	RATING
1. Lab Contents.	[]
2. Usefulness of manual.	[]
3. No. of experiments.	[]
4. No. of tasks per lab.	[]
5. Effectiveness of quizzes/tasks/exams .	[]
6. Overall Lab Contents	[]

INSTRUCTOR RATING	ASSESSMENT
1. Punctuality.	[]
2. Lecture Preparation.	[]
3. Response towards students' questions .	[]
4. Communication at students' level.	[]
5. Skill in emphasizing main points.	[]
6. Effectiveness of examples.	[]
7. Arrangement of material.	[]
8. Availability other than lab hours.	[]
9. Overall Instructor Assessment	[]

ANY SUGGESTIONS, COMMENTS AND REMARKS FOR IMPROVEMENT:

ANY COMPLAINTS ABOUT LAB CONTENTS OR INSTRUCTOR:
