V UNIT

TOPICS

- INTRODUCTION TO PIC MICROCONTROLLER
- CHARACTERISTICS OF PIC MICROCONTROLLER
- PIC MICROCONTROLLER FAMILIES
- INTERNAL ARCHITECTURE OF PIC 16F877A
- MEMORY ORGANIZATION
- PARALLEL AND SERIAL INPUT AND OUTPUT
- TIMERS OF PIC 16F877
- INTERRUPTS OF PIC16F877A

INTRODUCTION TO PIC MICROCONTROLLER

- PIC is a Peripheral Interface Microcontroller which was developed in the year 1993 by the General Instruments Microcontrollers.
- It is controlled by software and programmed in such a way that it performs different tasks and controls a generation line.
- PIC microcontrollers are used in different new applications such as smart phones, audio accessories and advanced medical devices.
- There are many PICs available in the market ranging from PIC16F84 to PIC16C84. These types of PICs are affordable flash PICs.

- Microchip has recently introduced flash chips with different types, such as 16F628, 16F877 and 18F452.
- The 16F877 costs twice the price of the old 16F84, but it is eight times more than the code size, with more RAM and much more I/O pins, a UART, A/D converter.
- The PIC microcontroller is based on RISC architecture. Its memory architecture follows the Harvard pattern of separate memories for program and data, with separate buses.

CHARACTERISTICS OF PIC MICROCONTROLLER

- RISC architecture
  - Only 35 instructions to learn
  - All single-cycle instructions except branches
- Operating frequency 0-20 MHz
- Precision internal oscillator
  - Factory calibrated
  - Software selectable frequency range of 8MHz to 31KHz
- Power supply voltage 2.0-5.5V
  - Consumption: 220uA (2.0V, 4MHz), 11uA (2.0 V, 32 KHz) 50nA (stand-by mode)
- Power-Saving Sleep Mode

- Brown-out Reset (BOR) with software control option
- 35 input/output pins
  - High current source/sink for direct LED drive
  - Software and individually programmable pull-up resistor
  - Interrupt-on-Change pin
- 8K ROM memory in FLASH technology
  - Chip can be reprogrammed up to 100.000 times
- In-Circuit Serial Programming Option
  - Chip can be programmed even embedded in the target device
- 256 bytes EEPROM memory
  - Data can be written more than 1.000.000 times
- 368 bytes RAM memory

- A/D converter:
  - 14-channels
  - 10-bit resolution
- 3 independent timers/counters
  - Watch-dog timer
- Analogue comparator module with
  - Two analogue comparators
  - Fixed voltage reference (0.6V)
  - Programmable on-chip voltage reference
- PWM output steering control
- Enhanced USART module
  - Supports RS-485, RS-232 and LIN2.0
  - Auto-Baud Detect
- Master Synchronous Serial Port (MSSP)
  - supports SPI and I2C mode

PIC MICROCONTROLLER FAMILIES

- The 8-Bit family has four categories:
  - Baseline (12-bit wide program memory)
  - Mid-Range (14-bit wide program memory)
  - Enhanced Mid-Range (Enhanced 14-bit wide program memory)
  - High End (16-bit wide program memory)

Baseline

- Baseline PIC microcontrollers utilize a 12-bit instruction word. Baseline has the simplest architecture of the 8-bit family and therefore are the easiest to work with and understand.
- They feature:
  - Simple 33 (12-bit wide) instruction set for ease of use and quick development
  - 2 K word (3 KB) addressable program memory

- 144 bytes RAM (max)
- 2 level hardware stack
- 1 (8-bit) file select register
- Multiple product options and easy migration
- Smallest form factors available

- The Baseline can be recognized by their part number structure. 10Fxxx, 12Fxxx and 16Fxxx.

Mid-Range

- Mid-Range PIC Microcontrollers are the next tier in performance and has features from the Baseline PIC microcontrollers.
- Utilizing a 14-bit instruction word, these peripheral-rich devices are ideal for many applications that require a more memory.

Features of mid range:

- 35 (14-bit wide) easy instructions to learn
- 8 K word (14 KB) addressable program memory
- 368 bytes RAM (max)
- 8 level hardware stack
- 1 (9-bit) file select register
- Hardware interrupt handling

Enhanced Mid-Range

- The Enhanced core adds more program memory and higher operating speeds. They also feature the highest accuracy, highest frequency internal oscillators.
- The Enhanced Mid-Range can be recognized by their part number structure12F1xxx and 16F1xxx

Features of enhanced mid range:

- 49 (14-bit wide) assembly commands
- 32 K word (56 KB) addressable program memory
- 4 KB RAM (max)
- 16 level hardware stack
- 2 (16-bit) file select registers
- Hardware interrupt handling with content save
- Advanced feature set, multiple serial communications, and motor control capability

High-End

- These parts have their own prefix namely, PIC18. This family combines the maximum level of performance and integration with the ease-of-use of an 8-bit architecture.
- With up to 16 MIPS of processing power, PIC18 Microcontrollers feature advanced peripherals, such as CAN, USB, Ethernet, LCD

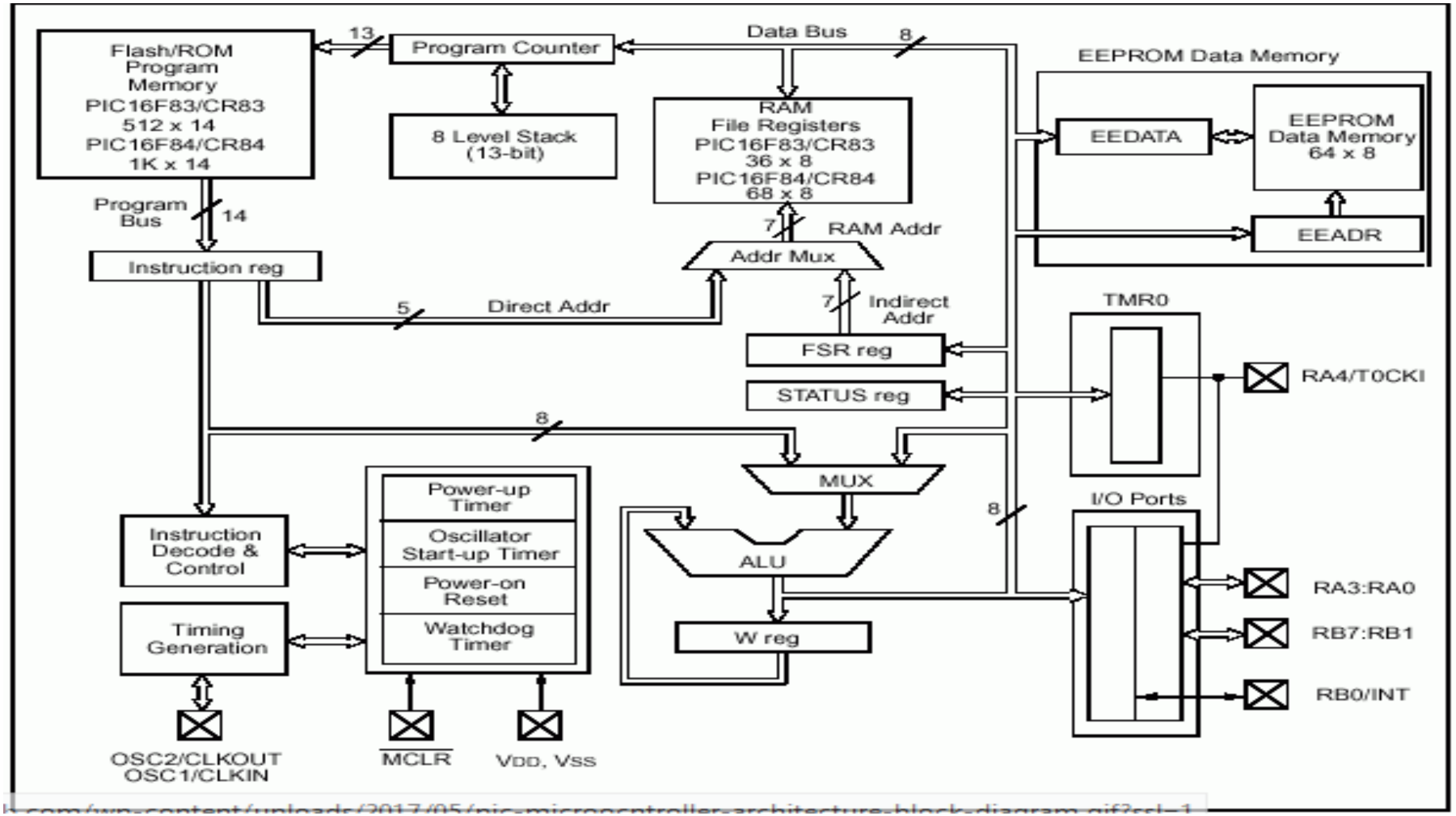
- The High-End devices can be recognized by their part number structure18Fxxxx, 18FxxJxx and 18FxxKxx.
- The architecture is optimized for C programming.

Features:

- 83 (16-bit wide) assembly instructions
- Up to 2 MB addressable program memory
- 4 KB RAM (max)
- 32 level hardware stack
- 1 (8-bit) file select register
- Integrated 8x8 hardware multiply
- Highest performance 8-bit architecture

	Baseline Architecture	Mid-Range Architecture	Enhanced Mid-Range Architecture	PIC18 Architecture
Pin Count	6-40	8-64	8-64	18-100
Interrupts	No	Single interrupt capability	Single interrupt capability with hardware context save	Multiple interrupt capability with hardware context save
Performance	5 MIPS	5 MIPS	8 MIPS	Up to 16 MIPS
Instructions	33, 12-bit	35, 14-bit	49, 14-bit	83, 16-bit
Program Memory	Up to 3 KB	Up to 14 KB	Up to 28 KB	Up to 128 KB
Data Memory	Up to 138 Bytes	Up to 368 Bytes	Up to 1,5 KB	Up to 4 KB
Hardware Stack	2 level	8 level	16 level	32 level
Features	▪ Comparator ▪ 8-bit ADC ▪ Data Memory ▪ Internal Oscillator	In addition to Baseline: ▪ SPI/I²C™ ▪ UART ▪ PWMs ▪ LCD ▪ 10-bit ADC ▪ Op Amp	In addition to Mid-Range: ▪ Multiple Communication Peripherals ▪ Linear Programming Space ▪ PWMs with Independent Time Base	In addition to Enhanced Mid-Range: ▪ 8x8 Hardware Multiplier ▪ CAN ▪ CTMU ▪ USB ▪ Ethernet ▪ 12-bit ADC
Highlights	Lowest cost in the smallest form factor	Optimal cost to performance ratio	Cost effective with more performance and memory	High performance, optimized for C programming, advanced peripherals

INTERNAL ARCHITECTURE OF PIC 16F877A



Central Processing Unit (CPU):

- PIC microcontroller's CPU consists of
  - Arithmetic logic unit (ALU)
  - Memory unit (MU)
  - Control unit (CU)
  - Accumulator
- ALU is used for arithmetic operations and for logical decisions.
- Memory is used for storing the instructions after processing.
- Control unit is used to control the internal and external peripherals which are connected to the CPU and accumulator is used for storing the results.

Program Counter & 8-Level Stack:

- Program Counter loads the address of the next instruction to be executed. This PC contains 8-level stack.
- 8-level stack indicate the long stack for multiple interrupt. For example, the return address of the interrupt can be stored in this 8-level stack memory.

RAM File Registers:

- All the general purpose registers that are used with PIC microcontroller will be stored in this RAM file registers.

Flash Program Memory:

- Entire program related instructions are available in this area. The instructions can be fetched from this area and can be given to instruction register.

- The box containing the Power up timer, Oscillator Start up timer, Watch Dog Timer, Power-On Reset is referred to as internal control circuitry.

FSR Register:

- FSR stands for File Select Register. It is used for indirect or indexed addressing of the other file registers, particularly the GPRs (General Purpose Registers).

I/O Ports:

- There are three I/O ports that can be used with PIC microcontroller which are used to communicate with outside world i.e. PORT A, PORT B, PORT C

OSC2 & OSC1:

- Used to generate different oscillator frequencies for the microcontroller.

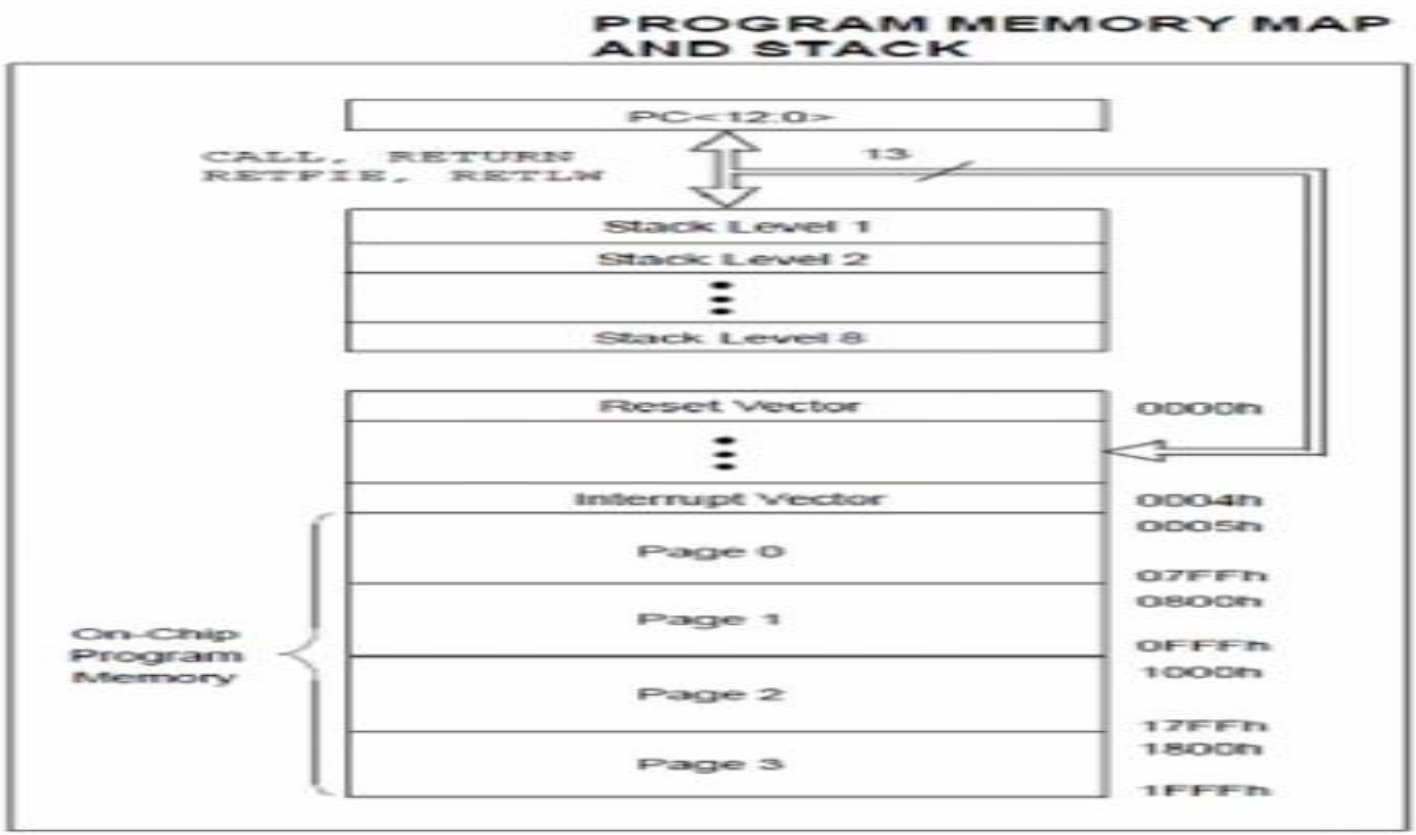
MEMORY ORGANIZATION

- The memory of a PIC 16F877 chip is divided into 3 sections. They are
  - Program memory
  - Data memory and
  - Data EEPROM

Program memory:

- Program memory contains the programs that are written by the user. The program counter (PC) executes these stored commands one by one.
- PIC16F877 devices have a 13 bit wide program counter that is capable of addressing 8K×14 bit program memory space.
- This memory is primarily used for storing the programs that are written (burned) to be used by the PIC.

- These devices also have 8K*14 bits of flash memory that can be electrically erasable /reprogrammed.
- Each time we write a new program to the controller, we must delete the old one at that time. The figure below shows the program memory map and stack.



- Program counters (PC) is used to keep the track of the program execution by holding the address of the current instruction.
- The PC is automatically incremented to the next instruction during the current instruction execution.
- The PIC16F87XA family has an 8-level deep x 13-bit wide hardware stack.
- The stack space is not a part of either program or data space and the stack pointers are not readable or writable.
- In the PIC microcontrollers, this is a special block of RAM memory used only for this purpose.
- Each time the main program execution starts at address 0000 – Reset Vector. The address 0004 is “reserved” for the “interrupt service routine” (ISR).

PIC16F87XA Data Memory Organization

- The data memory of PIC16F877 is separated into multiple banks which contain the general purpose registers (GPR) and special function registers (SPR).
- According to the type of the microcontroller, these banks may vary.
- The PIC16F877 chip only has four banks (BANK 0, BANK 1, BANK 2, and BANK4). Each bank holds 128 bytes of addressable memory.
- The banked arrangement is necessary because there are only 7 bits are available in the instruction word for the addressing of a register, which gives only 128 addresses.
- The selection of the banks are determined by control bits RP1, RP0 in the STATUS registers

- Together the RP1, RP0 and the specified 7 bits effectively form a 9 bit address.
- The first 32 locations of Banks 1 and 2, and the first 16 locations of Banks2 and 3 are reserved for the mapping of the Special Function Registers (SFR's).

BANK	RP0	RP1
0	0	0
1	1	0
2	0	1
3	1	1

Data EEPROM and FLASH

- The data EEPROM and Flash program memory is readable and writable during normal operation (over the full VDD range).
- This memory is not directly mapped in the register file space. Instead, it is indirectly addressed through the Special Function Registers.
- There are six SFRs used to read and write this memory:
  - EECON1
  - EECON2
  - EEDATA
  - EEDATH
  - EEADR
  - EEADRH

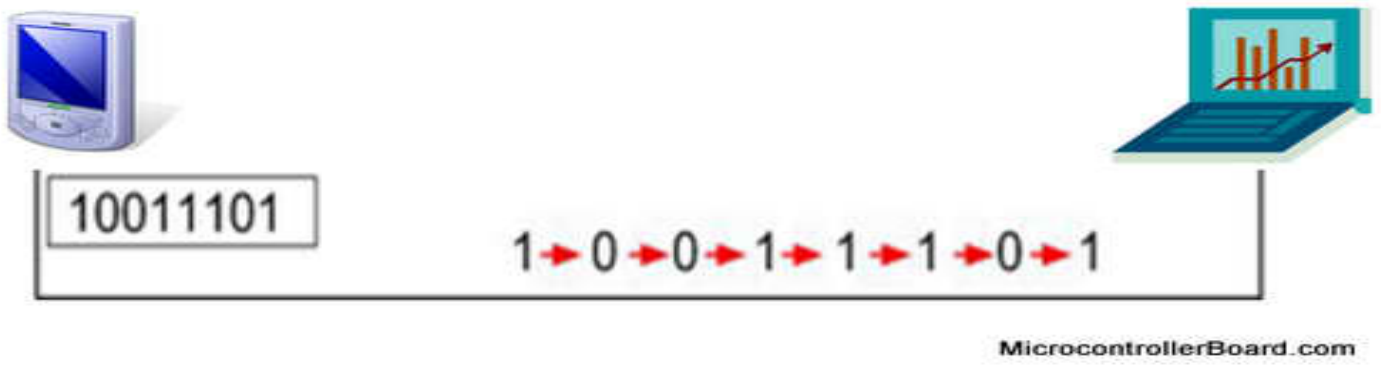
- The EEPROM data memory allows single-byte read and writes. The Flash program memory allows single-word reads and four-word block writes.
- Program memory write operations automatically perform an erase-before write on blocks of four words.
- A byte write in data EEPROM memory automatically erases the location and writes the new data (erase-before-write).
- The write time is controlled by an on-chip timer. The write/erase voltages are generated by an on-chip charge pump.

PARALLEL AND SERIAL INPUT AND OUTPUT

Feature	Serial Communication	Parallel Communication
Relative Speed	Faster	Slower
Distance Range	Much Farther	Shorter
Transfer Method	One bit is transmitted at a time	Bytes are transmitted in parallel, One byte or more at a time
Applications	Computer small peripherals, modules interfacing, and sensors measuring & sending simple data frames	Short distance High-Speed communication such as Computer printers, etc
Wires Inside	Few wires, all data bits pass only through the same data line	Multiple, more wires, each bit has a dedicated wire. So as to be transmitted all at once

Using the serial communication:

When using the serial communication we transmit the multi-bit word bit after bit (when at any given moment only one bit will pass).



Transmitting the word 10011101 using serial communication.

Using the parallel communication:

When using the parallel communication, however, the number of bits will be transmitted at once from one computer to the second computer.

Serial communication with PIC 16F877A:

- PIC16F877A comes with inbuilt USART which can be used for Synchronous/Asynchronous communication.

UART Registers

- The below table shows the registers for PIC16F877A UART.

Register	Description
TXSTA	Transmit Status And Control Register
RCSTA	Receive Status And Control Register
SPBRG	USART Baud Rate Generator
TXREG	USART Transmit Register. Holds the data to to be transmitted on UART
RCREG	USART Transmit Register. Holds the data received from UART

TXSTA							
7	6	5	4	3	2	1	0
CSRC	TX9	TXEN	SYNC	-	BRGH	TRMT	TX9D

CSRC: Clock Source Select bit
Asynchronous mode:Don't care.

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

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

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

BRGH: High Baud Rate Select bit
1 = High speed
0 = Low speed

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

TX9D: 9th bit of Transmit Data, can be Parity bit

RCSTA							
7	6	5	4	3	2	1	0
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D

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

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

SREN: Single Receive Enable bit
Asynchronous mode:Don't care.

CREN: Continuous Receive Enable bit
Asynchronous mode:
1 = Enables continuous receive
0 = Disables continuous receive

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 is set
0 = Disables address detection, all bytes are received and ninth bit can be used as parity bit

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

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

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

➢ Steps To Send Char

- ➢ Wait till the previous char is transmitted. TXIF will be set when the TXREG is empty.
- ➢ Clear the TXIF for next cycle.
- ➢ Load the new char to be transmitted into THR.

```
void UART_TxChar(char ch)
{
    while(TXIF==0);  // Wait till the transmitter register becomes empty
    TXIF=0;          // Clear transmitter flag
    TXREG=ch;        // load the char to be transmitted into transmit reg
}
```

➢ Steps To Receive Char

- ➢ Wait till the Data is received. RCIF will be set once the data is received in RCREG register.
- ➢ Clear the receiver flag(RCIF) for next cycle.
- ➢ Copy/Read the received data from RCREG register.

```
char UART_RxChar()
{
    while(RCIF==0);    // Wait till the data is received
    RCIF=0;            // Clear receiver flag
    return(RCREG);     // Return the received data to calling function
}
```

```
#include<pic16f877a.h>
#define SBIT_TXEN     5
#define SBIT_SPEN     7
#define SBIT_CREN     4

void UART_Init(int baudRate)
{
    TRISC=0x80;             // Configure Rx pin as input and Tx as output
    TXSTA=(1<<SBIT_TXEN);  // Asynchronous mode, 8-bit data & enable transmitter
    RCSTA=(1<<SBIT_SPEN) | (1<<SBIT_CREN);  // Enable Serial Port and 8-bit continuous receive
    SPBRG = (20000000UL/(long)(64UL*baudRate))-1;      // baud rate @20Mhz Clock
}


void UART_TxChar(char ch)
{
    while(TXIF==0);     // Wait till the transmitter register becomes empty
    TXIF=0;             // Clear transmitter flag
    TXREG=ch;           // load the char to be transmitted into transmit reg
}

char UART_RxChar()
{
    while(RCIF==0);     // Wait till the data is received
    RCIF=0;             // Clear receiver flag
    return(RCREG);      // Return the received data to calling function
}


int main()
{
    char i,a[]={"Welcome to Pic Serial Comm, Type the char to be echoed: "};
    char ch;
    UART_Init(9600);           //Initialize the UART module with 9600 baud rate
    for(i=0;a[i]!=0;i++)
        UART_TxChar(a[i]); // Transmit predefined string
     while(1)
    {
        ch = UART_RxChar(); // Receive a char from serial port
        UART_TxChar(ch);    // Transmit the received char
    }
}
```

TIMERS OF PIC 16F877

- The PIC 16F877 basically has three timer modules. These timer module terminals are also multiplexed with other functions for handling alternate functions.
- These timer modules are usually denoted by the symbols TIMER-0, TIMER-1, and TIMER-2. These modules help to perform various timing and counting functions inside the chip.

TIMER-0 module

- The main timing/counting features of Timer-0 module are given below.
- Timer-0 module has built in 8 bit timer/counter
- It can be easily readable/writable
- Built in 8 bit software programmable pre-scalar functions
- Easily select internal/external clock pulses
- Interrupt with overflow from the value FFh to 00h
- Edge selection for external clock pulse

- The timer mode is normally selected by clearing the T0CS bit in the register.
- In Timer mode, when the Timer 0 Module increases with every instruction cycle, the TMR0 register is written.
- Counter mode is selected by setting bit T0CS in Counter mode.
- Timer 0 will increment either on every rising or falling edge of pin RA4/T0CKI.

REGISTERS ASSOCIATED WITH TIMER0

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TMR0	Timer0 Module Register							
INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF
OPTION_REG	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0

INTCON 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	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF

GIE: Global Interrupt Enable bit

PEIE: Peripheral Interrupt Enable bit

TMR0IE: TMR0 Overflow Interrupt Enable bit

INTE: RB0/INT External Interrupt Enable bit

RBIE: RB Port Change Interrupt Enable bit

TMR0IF: TMR0 Overflow Interrupt Flag bit

INTF: RB0/INT External Interrupt Flag control bit

RBIF: RB Port Change Interrupt Flag bit.

Bit 7 GIE: Global Interrupt Enable bit. If this bit is enable ('1'), which also enable all unmasked interrupts and if it is zero ('0'), which disable all interrupts.
1 = Enables all unmasked interrupts.
0 = Disables all interrupts.

Bit 6 (PEIE): this is a Peripheral Interrupt Enable bit which used for controlling peripheral interrupts. If this bit is enable('1'), also enable all unmasked peripheral interrupts and if it is disable('0'), also disable all active peripheral interrupt actions.
1 = Enables all unmasked peripheral interrupts
0 = Disables all peripheral interrupts

Bit 5 (TMR0IE): This is timer 0(TMR0) Overflow Interrupt Enable bit which control the overflow interrupt in timer 0.
1 = Enables the TMR0 interrupt
0 = Disables the TMR0 interrupt

Bit 4 (INTE): This is an RB0/INT External Interrupt Enable bit which used for enable/disable external interrupts.
1 = Enables the RB0/INT external interrupt
0 = Disables the RB0/INT external interrupt

Bit 3 (RBIE): RB Port Change Interrupt Enable bit which control PORTB change interrupt.
1 = Enables the RB port change interrupt.
0 = Disables the RB port change interrupt.

Bit 2 (TMR0IF): TMR0 Overflow Interrupt Flag bit which controls the overflow of timer 0.
1 = TMR0 register has overflowed [must be cleared in software]
0 = TMR0 register did not overflow

Bit 1 (INTF): RB0/INT External Interrupt Flag control bit.
1 = The RB0/INT external interrupt occurred (must be cleared in software)
0 = The RB0/INT external interrupt did not occur

Bit 0 (RBIF): RB Port Change Interrupt Flag bit.
1 = At least one of the RB7:RB4 pins changed state; a mismatch condition will continue to set that bit. Reading PORTB will end the mismatch condition and allow the bit to be cleared
[Must be cleared in software]

Option register:

- The option Register is a readable and writable register, which contains various control bits to configure the TMR0 prescaler, the external INT interrupt andTMR0 and the weak pull-ups on PORTB.

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
RBPU(inverting)	INTEDG	TOCS	TOSE	PSA	PS2	PS1	PS0

- **RBPU:** PORTB Pull-up Enable bit
- **INTEDG:** Interrupt Edge Select
- **T0CS:** Timer-0(TMR0) Clock Source Select
- **T0SE:** TMR0 Source Edge Select
- **PSA:** Prescaler Assignment bit
- **PS2 – PS0:** Prescaler Rate Select bits.

Bit 7 (RBPU): This is a PORTB Pull-up Enable bit. If this bit is '1' then PORTB pull-up function disabled. If this bit is '0', it enabled the pull-up function by individual port-latch values.

1 = PORTB pull-ups are disabled.

0 = PORTB pull-ups are enabled by individual port latch values.

Bit 6 (INTEDG): This is an Interrupt Edge Select bit. This bit decided if the interrupt is on either raisin edge or falling edge. The function of this bit is given below.

1 = Interrupt on rising edge of RB0/INT pin.

0 = Interrupt on falling edge of RB0/INT pin.

Bit 5 (T0CS): this is a timer-0(TMR0) Clock Source Select bit and its function is given below.

1 = Transition on RA4/T0CKI pin.

0 = Internal instruction cycle clock (CLK0).

Bit 4 (T0SE): TMR0 Source Edge Select bit which select the timer 0 source edge.

1 = Increment on high-to-low transition on RA4/T0CKI pin.

0 = Increment on low-to-high transition on RA4/T0CKI pin.

Bit 3 (PSA): Prescaler Assignment bit.

1 = Prescaler is assigned to the Watch Dog Timer (WDT).

0 = Prescaler is assigned to the Timer0 module.

Bit 2-0 (PS2:PS0): Prescaler Rate Select bits.

Timer 1 Module

- Timer 1 module is a 16 bit timer/counter unit. That is, it consists of two 8 bit (8+8) registers (TMR1H, TMR1L) which read and write easily.
- TMR1 register is a pair of TMR1H and TMR1L and also its value increment its value from 0000h to FFFFh and rolls over to 0000h.
- Timer 1 module basically operates in two different modes. They are
  - Timer mode
  - Counter mode
- The operating mode of timer 1 module is selected by using the clock select bit (TMR1CS), in timer mode. The timer 1 increases on every instruction cycle. But in counter mode, it increases on every rising edge of the external clock input.

- Timer 1 pin can be enabled/disabled easily by setting/clearing the control bit (TMR1ON).
- This timer1 pin also has an internal reset input function. It can be generated by either of the two CCP modules.

Timer 1 Operation in Timer Mode

- The Timer mode can be easily selected by clearing the TMR1CS bit.
- In this mode, the input clock to the timer is FOSC/4. The synchronize control bit, T1SYNC, has no effect since the internal clock is always in sync. Timer1 Operation in Synchronized.

Counter Mode

- The synchronized Counter mode is selected by setting timer 1 synchronized counter select bit (TMR1CS).
- In this mode, the timer increments on every rising edge of clock input on pin RC1/T1OSI/CCP2 when bit T1OSCEN is set, or on pin RC0/T1OSO/T1CKI when bit T1OSCEN is cleared.

Timer 1 Oscillator

- A crystal oscillator circuit is built-in between pins T1OSI (input) and T1OSO (amplifier output).

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
INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF
PIR1	PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF
PIE1	PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE
TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register							
TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register							

PIR1 Register:

- The PIR1 register contains the individual flag bits for the peripheral interrupt. The structure of PIR1 register is given below.

R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
PSPIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF

- **PSPIF:** Parallel Slave Port Read/Write Interrupt Flag
- **ADIF:** A/D Converter Interrupt Flag
- **RCIF:** USART Receive Interrupt Flag
- **TXIF:** USART Transmit Interrupt Flag
- **SSPIF:** Synchronous Serial Port (SSP) Interrupt Flag
- **CCP1IF:** CCP1 Interrupt Flag
- **TMR2IF:** TMR2 to PR2 Match Interrupt Flag
- **TMR1IF:** TMR1 Overflow Interrupt Flag

Bit 7 (PSPIF): Parallel Slave Port Read/Write Interrupt Flag bit.
1 = A read or a write operation has taken place (must be cleared in software)
0 = No read or write has occurred

Bit 6 (ADIF): A/D Converter Interrupt Flag bit that control the interrupt flag for that analog to digital converter.
1 = An A/D conversion completed
0 = The A/D conversion is not complete

Bit 5 (RCIF): USART Receive Interrupt Flag bit.
1 = The USART receive buffer is full
0 = The USART receive buffer is empty

Bit 4 (TXIF): USART Transmit Interrupt Flag bit.
1 = The USART transmit buffer is empty
0 = The USART transmit buffer is full

Bit 3 (SSPIF): Synchronous Serial Port (SSP) Interrupt Flag bit that control the SSP interrupt flag in a PIC.
1 = The SSP interrupt condition has occurred and must be cleared in software before returning from the Interrupt Service Routine.
0 = No SSP interrupt condition has occurred

Bit 2 (CCP1IF): CCP1 Interrupt Flag bit that control capture-compare-pulse width modulation interrupt flag. It works in three modes.
1.Capture mode:
1 = A TMR1 register capture occurred (must be cleared in software)
0 = No TMR1 register capture occurred
2.Compare mode:
1 = A TMR1 register compare match occurred (must be cleared in software)
0 = No TMR1 register compare match occurred
3.PWM mode

Bit 1 (TMR2IF): TMR2 to PR2 Match Interrupt Flag bit
1 = TMR2 to PR2 match occurred (must be cleared in software)
0 = No TMR2 to PR2 match occurred

Bit 0 (TMR1IF): TMR1 Overflow Interrupt Flag bit.
1 = TMR1 register overflowed (must be cleared in software)
0 = TMR1 register did not overflow

PIE1 Register

- The PIE1 register contains the individual enable bits for the peripheral interrupts. The structure of this register is shown below.

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE

- **PSPIE:** Parallel Slave Port Read/Write Interrupt Enable
- **ADIE:** A/D Converter Interrupt Enable
- **RCIE:** USART Receive Interrupt Enable
- **TXIE:** USART Transmit Interrupt Enable
- **SSPIE:** Synchronous Serial Port Interrupt Enable
- **CCP1IE:** Capture Compare Pulse 1 Interrupt Enable
- **TMR2IE:** TMR2 to PR2 Match Interrupt Enable
- **TMR1IE:** TMR1 Overflow Interrupt Enable

Bit 7 (PSPIE): this bit is the Parallel Slave Port Read/Write Interrupt Enable bit
1 = Enables the PSP read/write interrupt
0 = Disables the PSP read/write interrupt

Bit 6 (ADIE): A/D Converter Interrupt Enable bit which control the analog to digital converter interrupt.
1 = Enables the A/D converter interrupt
0 = Disables the A/D converter interrupt

Bit 5 (RCIE): USART Receive Interrupt Enable bit which control the USART data reception interrupt.
1 = Enables the USART receive interrupt
0 = Disables the USART receive interrupt

Bit 4 (TXIE): USART Transmit Interrupt Enable bit that control USART data transmission.
1 = Enables the USART transmit interrupt
0 = Disables the USART transmit interrupt

Bit 3 (SSPIE): Synchronous Serial Port Interrupt Enable bit that control SSP data interrupt.
1 = Enables the SSP interrupt
0 = Disables the SSP interrupt

Bit 2 (CCP1IE): CCP1 Interrupt Enable bit which control the capture-compare-pulse width modulation interrupt.
1 = Enables the CCP1 interrupt
0 = Disables the CCP1 interrupt

Bit 1 (TMR2IE): TMR2 to PR2 Match Interrupt Enable bit.
1 = Enables the TMR2 to PR2 match interrupt
0 = Disables the TMR2 to PR2 match interrupt

Bit 0 (TMR1IE): TMR1 Overflow Interrupt Enable bit that control the overflow interrupt of timer 1 module.
1 = Enables the TMR1 overflow interrupt
0 = Disables the TMR1 overflow interrupt

Timer 2 Module

- Timer 2 is an 8-bit timer with a prescaler and a postsaler. It can be used as the PWM (pulse width modulation) time base for the PWM mode of the CCP module(s).

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
INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF
PIR1	PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF
PIE1	PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE
TMR2	Timer2 Module's Register							

INTERRUPTS OF PIC16F877A

- Interrupts are special events that requires immediate attention, it stops a microcontroller/microprocessor from the running task and to serve a special task known as Interrupt Service Routine (ISR) or Interrupt Handler.

PIC 16F877A has the following 15 interrupt sources :

- External
- Timer 0
- Timer 1
- RB Port Change
- Parallel Slave Port Read/Write
- A/D Converter
- USART Receive
- USART Transmit
- Synchronous Serial Port
- CCP1 (Capture, Compare, PWM)
- CCP2 (Capture, Compare, PWM)
- TMR2 to PR2 Match
- Comparator
- EEPROM Write Operation
- Bus Collision

Registers Used for Interrupts

- INTCON
- OPTION_REG
- PIE1
- PIR1
- PIE2
- PIR2

- We shall consider the USART interrupt and the timer interrupt among the 15 interrupt sources.