Chapter Two

Hardware Issue

8051 Microcontroller

Microcontroller(MCU)

- A microcontroller is an integrated circuit (IC) or system on chip which is small, low cost and self contained computer
  - designed to handle a specific task in embedded systems like receiving remote signals etc.
- MCU can be programmed to do a specific task based on it's instruction set and capabilities.
- The general microcontroller consists of the processor, the memory (RAM, ROM, EPROM), Serial ports, peripherals (timers, counters), etc.
- A microcontroller processes data given to it's input pins using it's CPU and gives output via output pins.

CPU families used in microcontrollers

- Microcontrol
  architecture.
- All microcc
  - contain
    code-co
  - they ma
    number
- There are
  Intel 8051, M

Intel 4004	4 bit (2300 PMOS trans, 108 kHz)	1971
Intel 8048	8 bit	1976
Intel 8031	8 bit (ROM-less)	
Intel 8051	8 bit (Mask ROM)	
Microchip PIC16C64	8 bit	1985
Motorola 68HC11	8 bit (on chip ADC)	
Intel 80C196	16 bit	
Atmel AT89C51	8 bit (Flash memory)	
Microchip PIC 16F877	8 bit (Flash memory + A	

programmed by the integrated circu
manufacturer (rather than by the use

e the

d

type of memory that functions similarly to RAM and ROM. write data he data isn't lost when the power is turned off

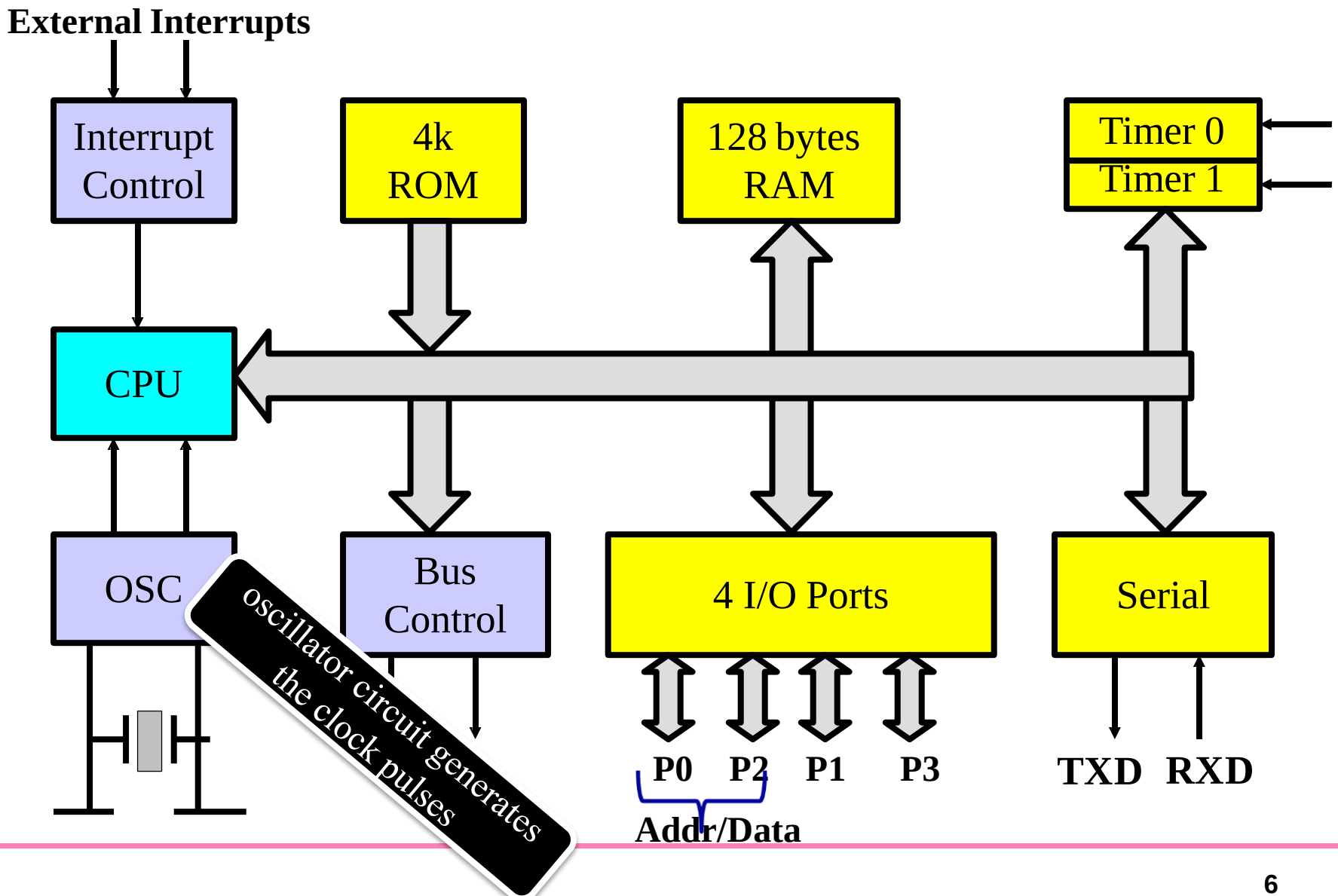
CPU families used in microcontrollers

- In 1981, Intel Corporation introduced an 8-bit microcontroller called the 8051 (Intel refers to it as MCS-51)
- The 8051 microcontroller family became widely popular after Intel allowing other manufactures to make and market any flavor of the 8051.
  - but remaining code-compatible.
- The 8051 family has the largest number of diversified suppliers: Intel, Atmel, Philips, AMD, Infineon and others.

Overview of 8051 family

- The 8051 is an 8-bit processor and it has
  - 128 bytes of RAM
  - 4K bytes of on-chip ROM
  - Two 16-bit timers
  - One serial port
  - Four I/O ports, each 8 bits wide

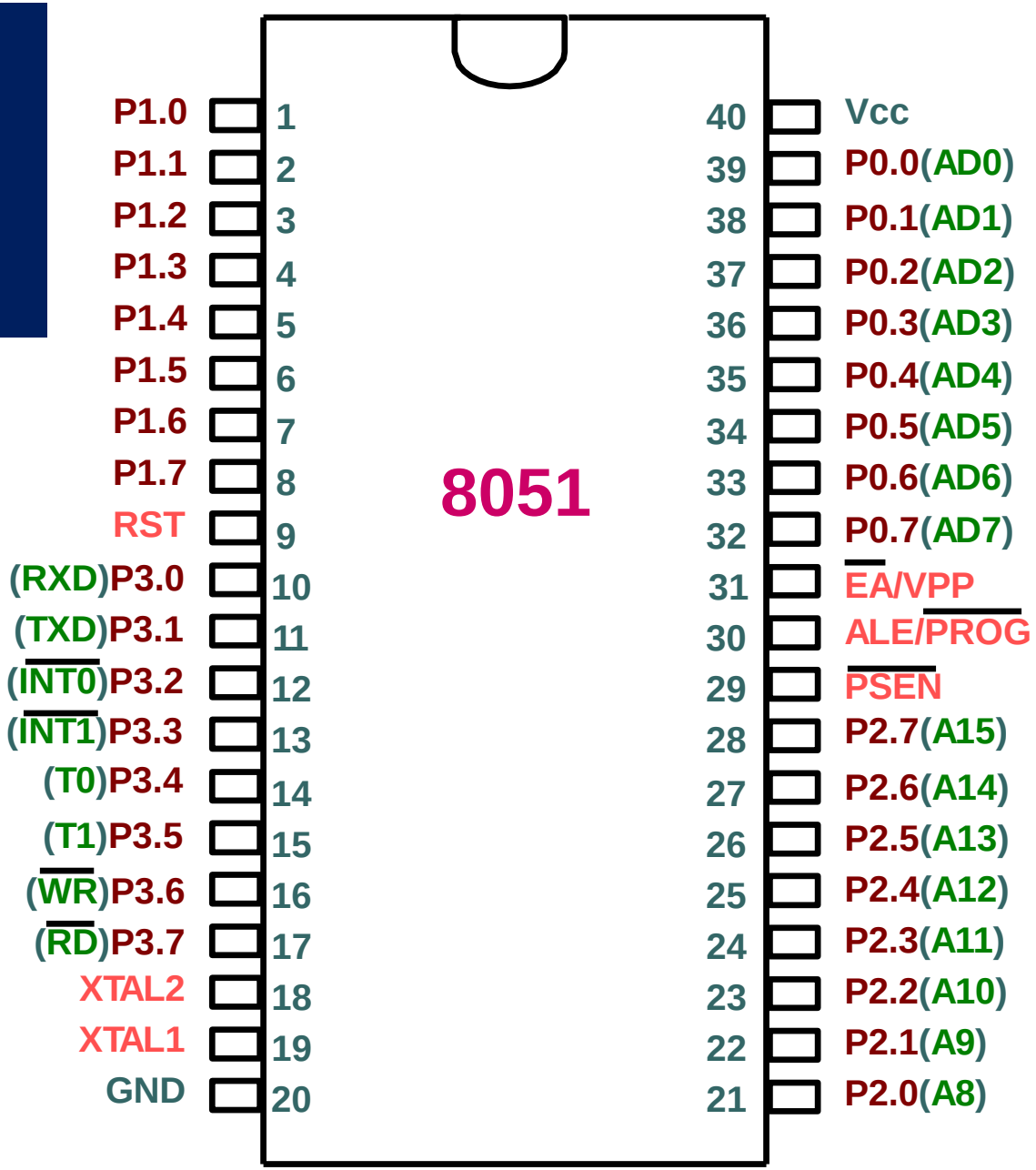
8051 Microcontroller



8051 Family

Feature	8051	8052	8031
ROM (on-chip program space in bytes)	4K	8K	0K
RAM (bytes)	128	256	128
Timers	2	3	2
I/O pins	32	32	32
Serial port	1	1	1
Interrupt sources	6	8	6

8051 pin diagram



I/O Ports

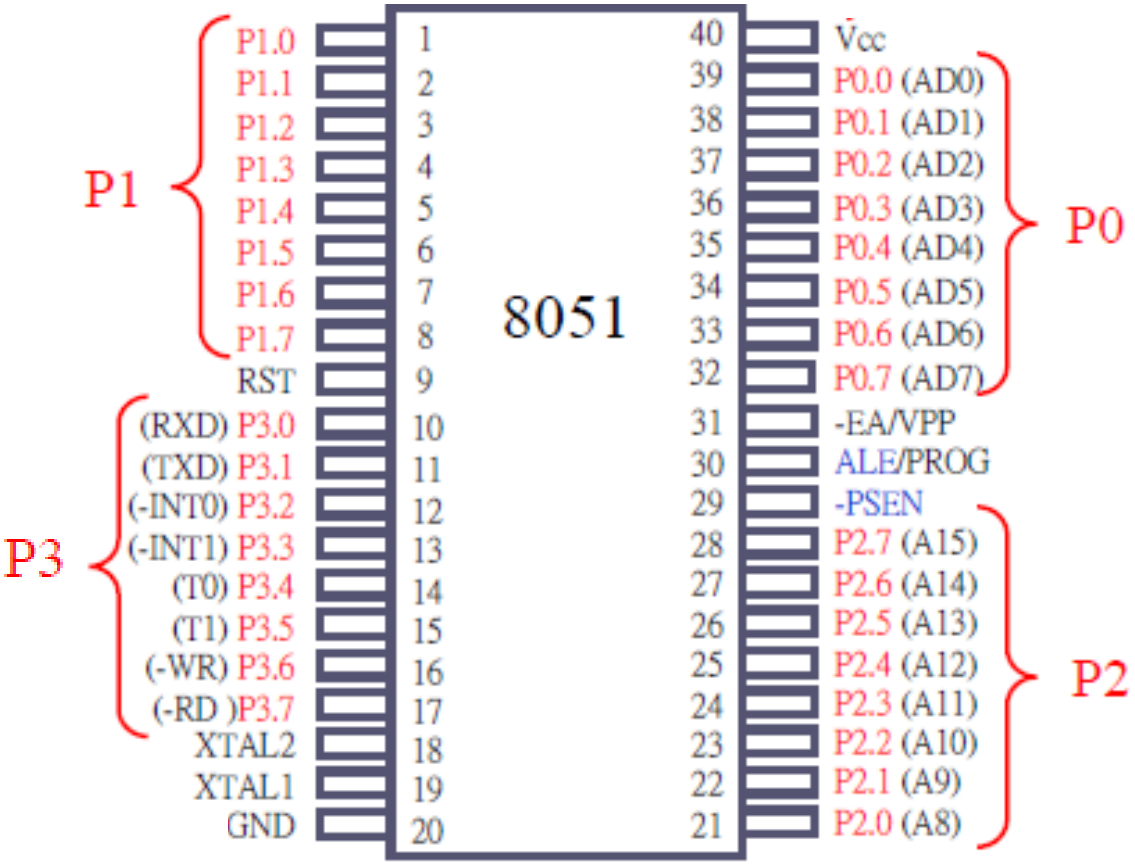
- In order to make the microcontroller useful, it is necessary to connect it to peripheral devices.
- The Peripherals of an embedded processor can either be on the same chip as the processor or can be connected externally through the I/O pins.(network, display port..)
- Each microcontroller has one or more registers (called a port) connected to the microcontroller pins.

I/O Ports

- I/O pins are generally grouped into ports of 8 pins, which can be accessed with a single byte access.
- Pins can either be input only, output only, or most commonly— bidirectional, that is, capable of both input and output.
- Apart from their I/O capabilities, most pins have one or more alternate functions to save pins and keep the chip small.
- The application programmer can select which function should be used for the pin by enabling the functionality within the appropriate module.

8051 I/O ports

- 8051 MCU has four ports P0, P1, P2, and P3 and each use 8 pins



8051 I/O ports

- **Port 0** （pins 32-39） : P0（P0.0-P0.7）
  - 8-bit R/W - allowing it to be used as both address and data
  - Or acts as **low order address bus** and **data bus** for external memory design.
- **Port 1** （pins 1-8） : P1（P1.0-P1.7）
  - Only 8- P1 is a true I/O port as it doesn't have any alternative functions as in P0.
  - if logic zero (0) is applied to the I/O port it will act as an output pin and if logic one (1) is applied the pin will act as an input pin

8051 I/O ports

- **Port 2** （pins 21-28） : P2（P2.0-P2.7）
  - 8-bit R/W –
  - Or only **high byte** of the **address bus** for external memory design(A8-A15).

8051 I/O ports

- **Port 3** （pins 10-17） : P3（P3.0～P3.7）
  - General Purpose I/O: if not used by the internal peripherals (timers) or external interrupts.

Port Pin	Alternate Function
P3.0	RXD (serial input port)
P3.1	TXD (serial output port)
P3.2	$\overline{\text{INT0}}$ (external interrupt 0)
P3.3	$\overline{\text{INT1}}$ (external interrupt 1)
P3.4	T0 (Timer 0 external input)
P3.5	T1 (Timer 1 external input)
P3.6	$\overline{\text{WR}}$ (external data memory write strobe)
P3.7	$\overline{\text{RD}}$ (external data memory read strobe)

Receive data

Other pins

- $\overline{EA}$（pin 31）：external access
  - If we have to use multiple memories then the application of logic 1 to this pin instructs the Microcontroller to read data from both memories: first internal and then external.
- $\overline{PSEN}$（pin 29）：program store enable
  - This is an output pin and Systems to allow storage of program code in external ROM..
- ALE（pin 30）：address latch enable
  - When ALE = 1, then the P0 pins work as Data bus and when ALE = 0, then the P0 pins act as Address bus.
  - This pin is used to distinguish between memory chips when multiple memory chips are used.

Other pins

Vcc – Pin 40 provides supply to the Chip and it is +5 V.

Gnd – Pin 20 provides ground for the Reference.

XTAL1, XTAL2 (Pin no 18 & Pin no 19) – CPU executing an instruction takes a certain number of clock cycles

- These are referred as to as machine cycles

- The length of machine cycle depends on the frequency of the crystal oscillator connected to 8051
- In original 8051, one machine cycle lasts 12 oscillator periods
- **8051 has an on-chip oscillator. It needs an external crystal that decides the operating frequency of the 8051.**
- Oscillators are **responsible for supplying the clock signals in microcontrollers**.
- **T0 and T1(timers): A timer 16bit register uses the frequency of the internal clock signal, and generates delay. to measure time and count external events.**

Other pins

(RXD) :10th pin is RXD (serial data receive pin) which is for serial input. Through this input signal microcontroller receives data for serial communication.

(TXD) :11th pin is TXD (serial data transmit pin) which is serial output pin

(WR') :16th pin is for external memory write i.e. writing data to the external memory.

(RD') :17th pin is for external memory read i.e. reading data from external memory.

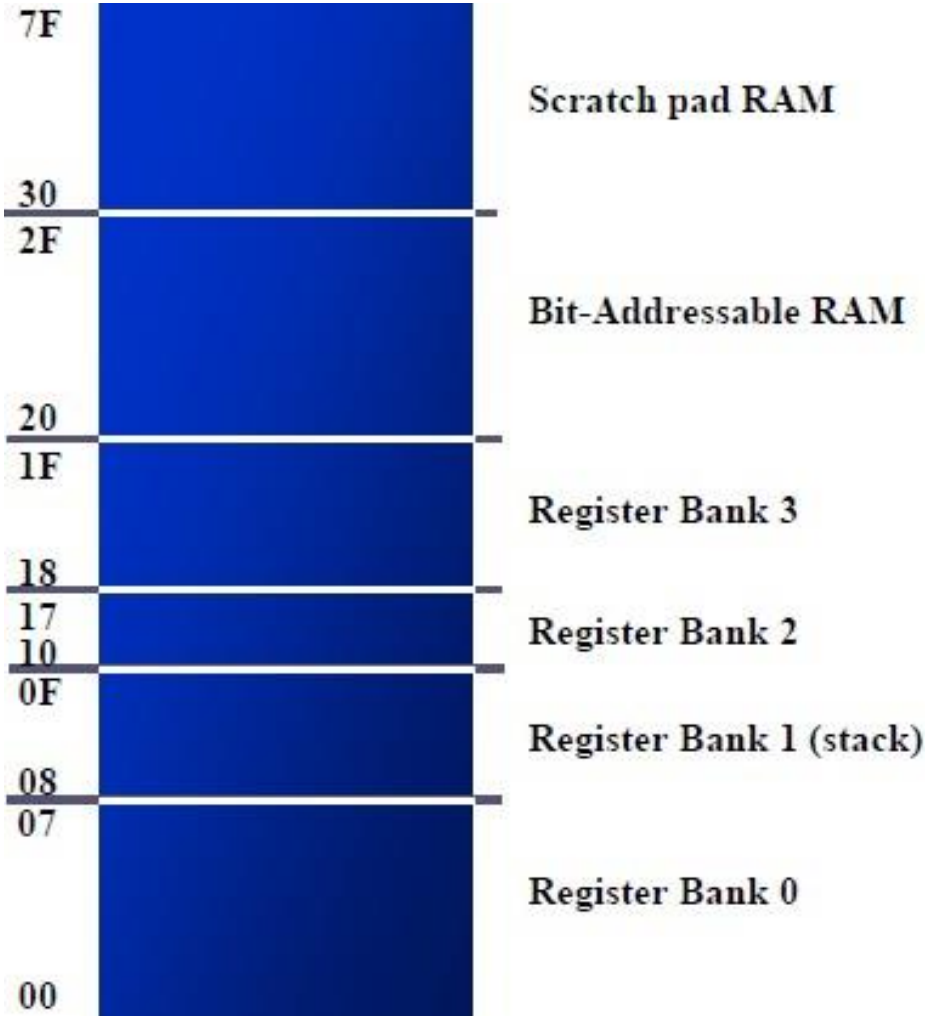
ROM in 8051

- The original 8051 microcontroller has 4K bytes on-chip ROM.
- No member of 8051 family can have more than 64K bytes ROM:
  - The program counter is a 16-bit register

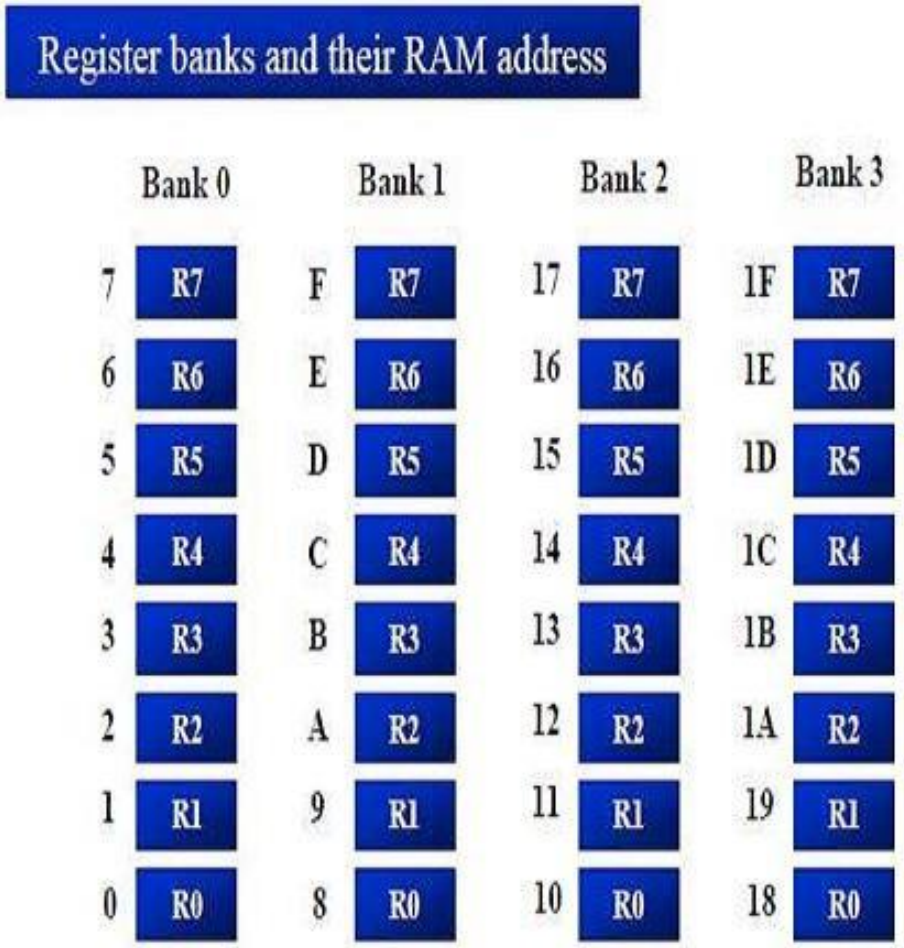
RAM in 8051

- There are 128 bytes of RAM in the 8051(Assigned addresses 00 to 7FH)
- The 128 bytes are divided into three different groups as follows:
  1. A total of 32 bytes from locations 00 to 1FH(0-31) are set aside for **register banks** and the **stack**
  2. A total of 16 bytes from locations 20H to 2FH (32-47) are set aside for **bit-addressable** read/write memory
  3. A total of 80 bytes from locations 30H to 7FH are used for read and write storage, called **scratch pad.**

RAM of 8051


7F
30
2F
20
1F
18
17
10
0F
08
07
00
Scratch pad RAM
Bit-Addressable RAM
Register Bank 3
Register Bank 2
Register Bank 1 (stack)
Register Bank 0

Register Banks



Register Banks

- The 32 bytes from address 00 to 1F hex are divided into 4 banks of registers in which each bank has 8 registers, R0-R7.
- Register bank 0 is the default when 8051 is powered up.
  - That is by default RAM locations 0, 1,2,3,4,5,6 and 7 are accessed with names R0, R1, R2, R3, R4, R5, R6 and R7 when programing the 8051.
- We can switch to other banks by use of the PSW(program status word) register.
  - Bits D4 and D3 of the PSW are used to select the desired register bank.

Program Status Word (PSW)

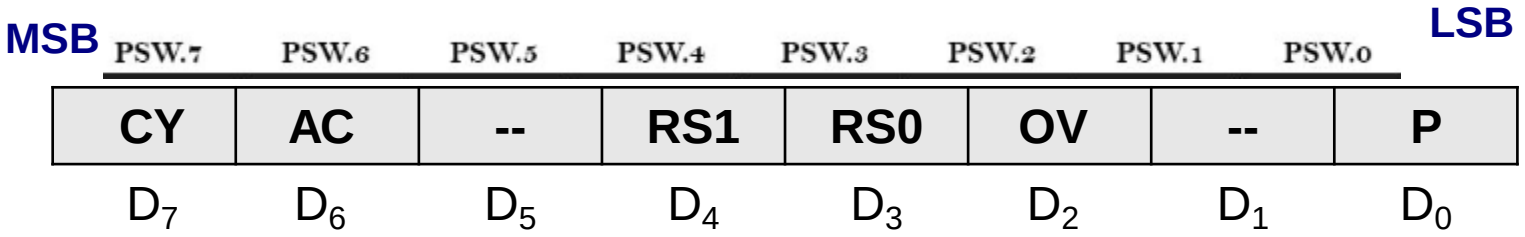
- Upon RESET, bank 0 is selected cause it is default
- We can select any other banks using the bit-addressability of the PSW.

CY	AC	--	RS1	RS0	OV	--	P

RS1	RS0	Register Bank	Address
0	0	0	00H - 07H
0	1	1	08H - 0FH
1	0	2	10H - 17H
1	1	3	18H - 1FH

Program Status Word (PSW)

- The program status word (PSW) register, also referred to as the flag register, is an 8 bit register.
- Only 4 bits are used
  - CY (carry), AC (auxiliary carry), P (parity), and OV (overflow).
  - The PSW3 and PSW4 are designed as RS0 and RS1, and are used to *change the register banks.*



Program Status Word (PSW) Register

Program Status Word (PSW)

- The carry flag (CY): there is a carry out of MSB or not.
  - Set whenever there is a carry out from the D7 bit
  - It is affected after an 8-bit addition or subtraction.
- Auxiliary carry (AC)
  - indicates a carry from D3 to D4 during addition and subtraction operations.
- *The overflow flag (OV)*
  - set whenever the result of a signed number operation is too large, causing the high-order bit to overflow into the sign bit.

Program Status Word (PSW)

- The parity flag (P)
  - The parity flag reflects the number of 1s in the A (accumulator) register only.
  - If the register A contains an odd number of 1s, then P = 1.

    And P = 0 if A has an even number of 1s.

Exercise:

Show the status of CY ,AC and P flags after the addition of 9CH and 64H in the following instruction. After Addition A=00H, C=1

MOV A, #9CH

ADD A, # 64H

CY = 1 since there is a carry beyond the D7 bit.
AC = 1 since there is a carry from the D3 to the D4 bit.
P = 0 since the accumulator has an even number of 1s (it has zero 1s).

Stack in 8051

- The stack is a section of RAM used by the CPU to store information temporarily.
- The register used to access the stack is called the stack pointer (SP) register.

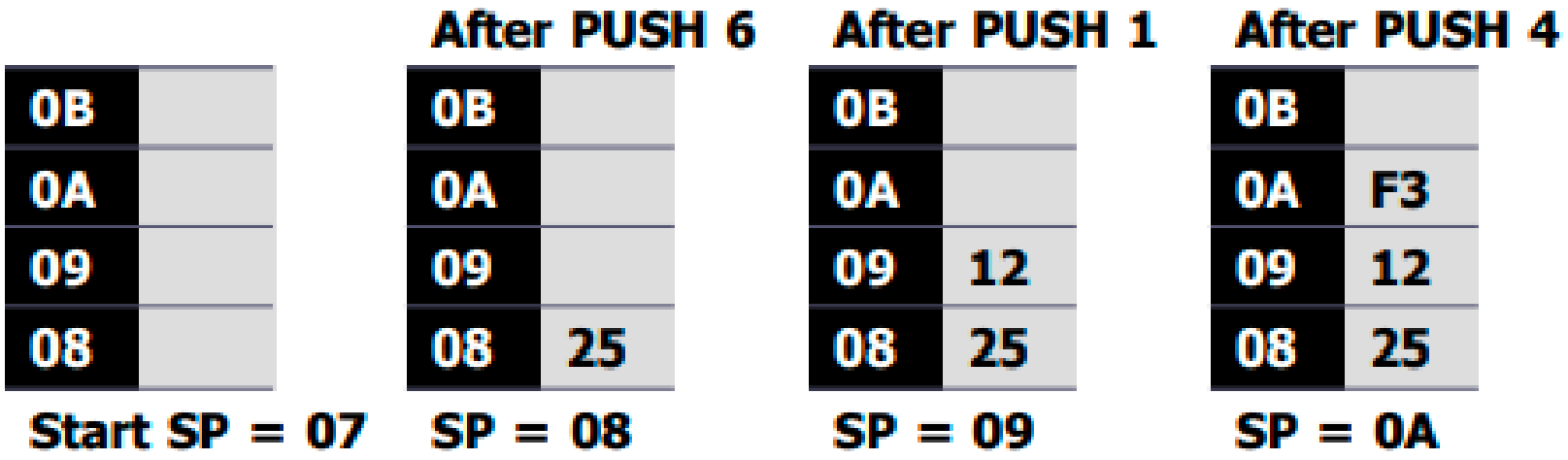
- The storing of a CPU register in the stack is called a PUSH
  - SP is pointing to the last used location of the stack
  - As we push data onto the stack, the SP is incremented by one
- Loading the contents of the stack back into a CPU register is called a POP
  - With every pop, the top byte of the stack is copied to the register specified by the instruction and the stack pointer is decremented one.

Stack in 8051

Show the stack and stack pointer from the following. Assume the default stack area.

```
MOV R6, #25H
MOV R1, #12H
MOV R4, #0F3H
PUSH 6
PUSH 1
PUSH 4
```

Solution:

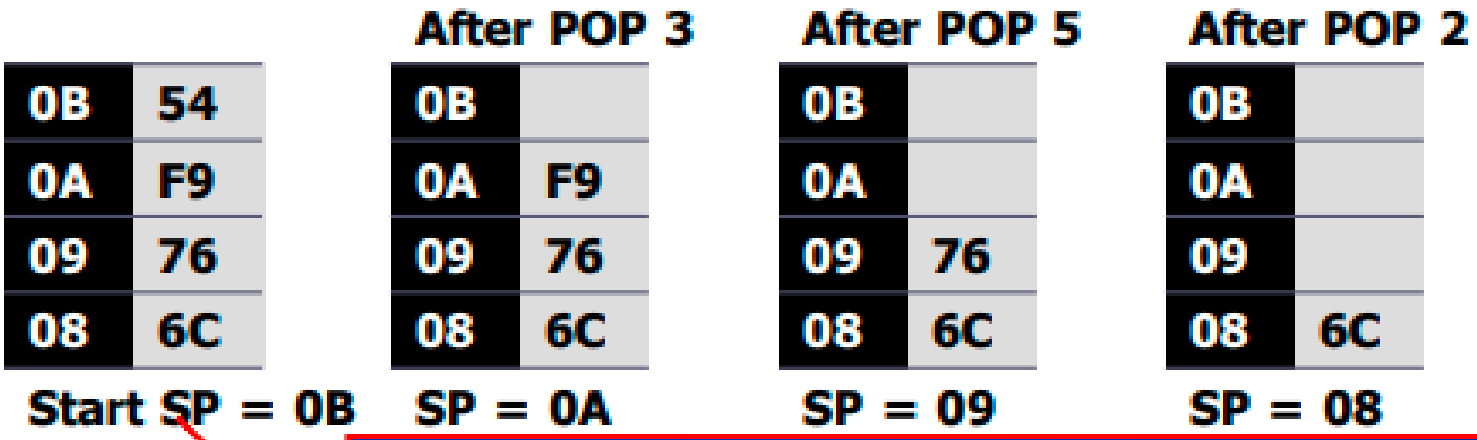


Stack in 8051

Examining the stack, show the contents of the register and SP after execution of the following instructions. All value are in hex.

```
POP     3          ; POP stack into R3
POP     5          ; POP stack into R5
POP     2          ; POP stack into R2
```

Solution:



Because locations 20-2FH of RAM are reserved for bit-addressable memory, so we can change the SP to other RAM location by using the instruction "MOV SP, #XX"

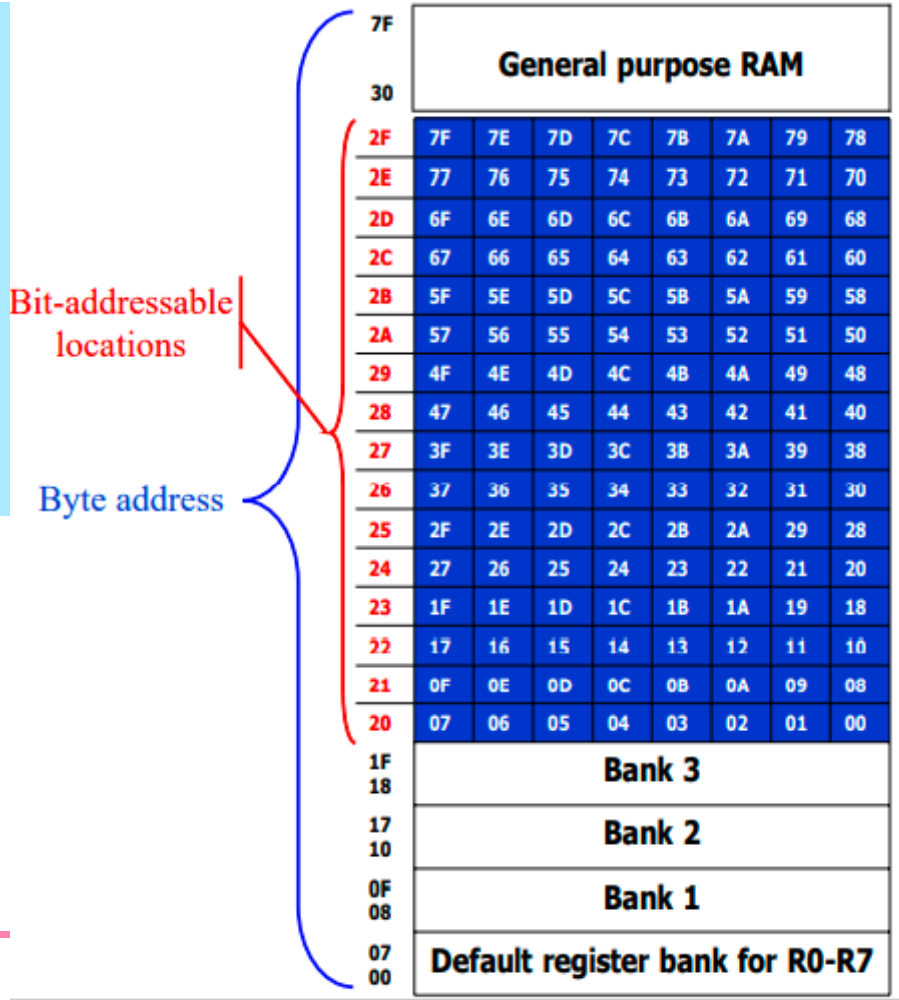
bit-addressable RAM

- Of the 128-byte internal RAM of the 8051, only 16 bytes from 20H to 2FH are bit-addressable.
  - Provide 128 bits (16*8) of RAM bit-addressability
  - They are address as 0 to 127 Decimal (00h-7Fh)
  - The bit addresses 0-7 are the first byte of internal RAM location 20H and bit addresses 8-0FH are second byte of internal RAM location 21H.

bit-addressable RAM

- Find out to which byte each of the following bits belongs. Give the address of the RAM

a) SETB 42H
b) CLR 67H
c) CLR 0F
d) SETB 28B
e) CLR 12
f) CLR 05H



Cont….

Solution:

(a) D2 of RAM location 28H

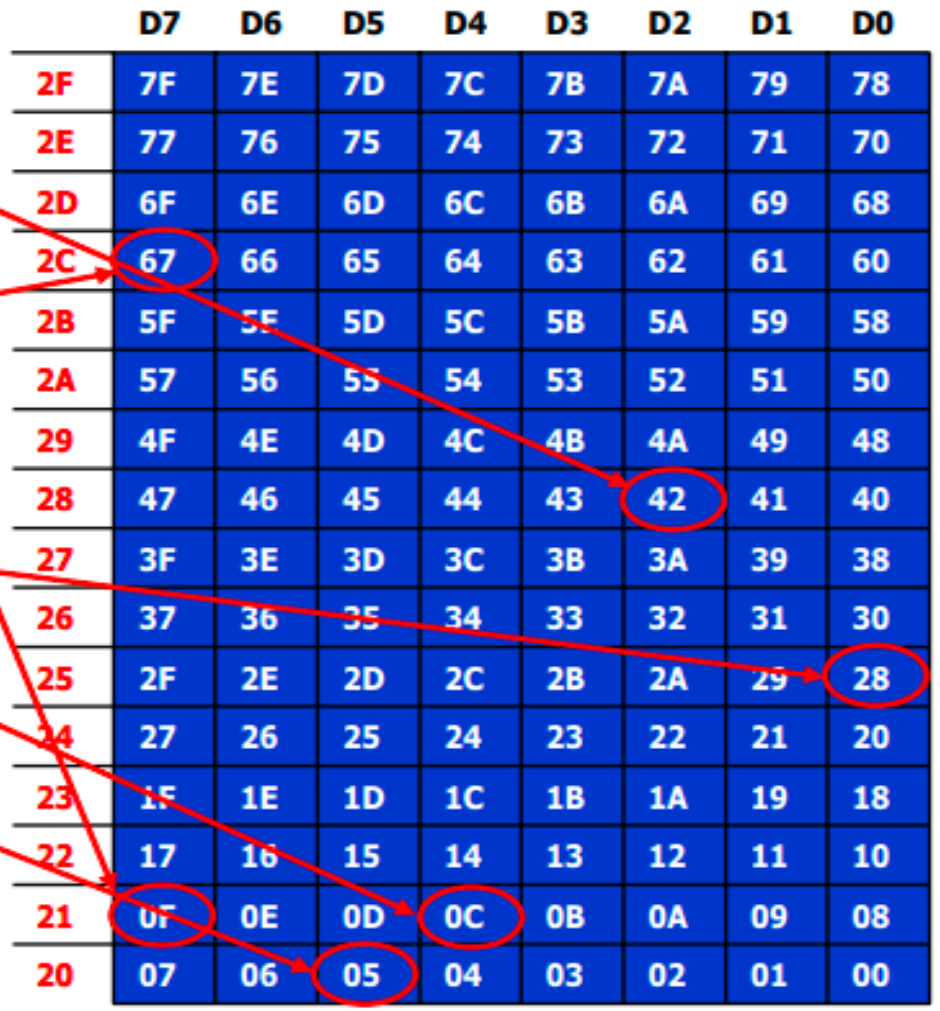
(b) D7 of RAM location 2CH

(c) D7 of RAM location 21H

(d) D0 of RAM location 25H

(e) D4 of RAM location 21H

(f) D5 of RAM location 20H



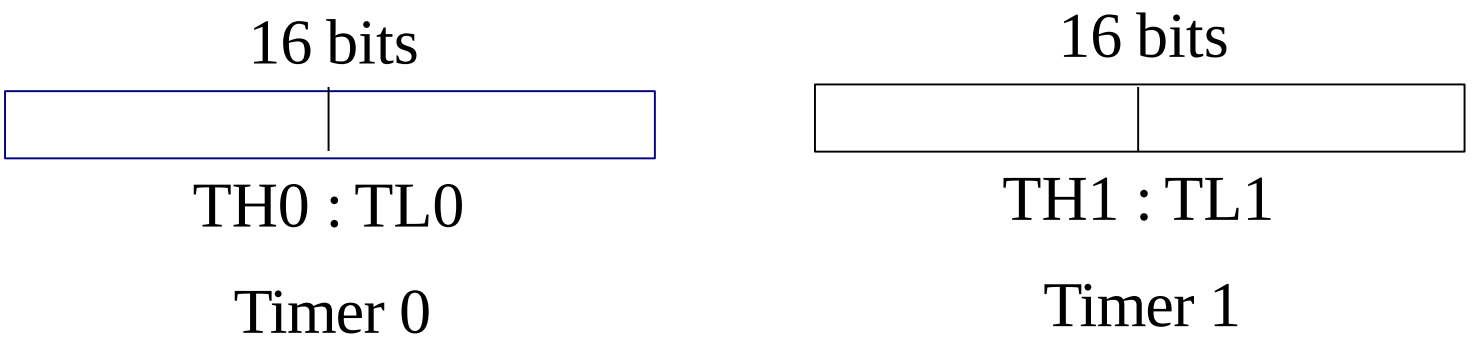
	D7	D6	D5	D4	D3	D2	D1	D0
2F	7F	7E	7D	7C	7B	7A	79	78
2E	77	76	75	74	73	72	71	70
2D	6F	6E	6D	6C	6B	6A	69	68
2C	67	66	65	64	63	62	61	60
2B	5F	5E	5D	5C	5B	5A	59	58
2A	57	56	55	54	53	52	51	50
29	4F	4E	4D	4C	4B	4A	49	48
28	47	46	45	44	43	42	41	40
27	3F	3E	3D	3C	3B	3A	39	38
26	37	36	35	34	33	32	31	30
25	2F	2E	2D	2C	2B	2A	29	28
24	27	26	25	24	23	22	21	20
23	1F	1E	1D	1C	1B	1A	19	18
22	17	16	15	14	13	12	11	10
21	0F	0E	0D	0C	0B	0A	09	08
20	07	06	05	04	03	02	01	00

Timers and Counters

- Timers and counters are distinguished from one another largely by their use, not by their logic.
- Both are built from adder logic with registers to hold the current value, with an increment input that adds one to the current register value.
  - A timer has its count connected to a periodic clock signal to measure time intervals.
  - A counter has its count input connected to an aperiodic signal in order to count the number of occurrences of some external event.

Timers in 8051

- Original 8051 has 2 timers.
- Timers increment on each system clock.
- Timer registers (TH0, TL0, TH1, TL1) can be read or written to.
- Timer overflow can cause "interrupts" or set SFR bits high.



Interrupts

- There are two techniques which can be used by the processor to communicate with I/O(peripheral) devices:

***Polling* and *Interrupt*.**

  - The microcontroller continuously monitors the status of a given device.

  - **Polling:** In this technique the **processor polls the device (asks question) repeatedly** at regular intervals to check if the device has completed the given task or has any new task to execute.

  - **Interrupt: an external or internal event that interrupt the MCU to inform it that a device need services.**

  - An interrupt is a signal sent **from a peripheral to the processor**.

  - A peripheral may send an interrupt signal to a processor when it has some job to perform which requires the processor's intervention.

Interrupts Vs Polling

- An interrupt is like a shopkeeper. If one needs a service or product, he goes to him and tells him his needs.
  - In case of interrupts, when the flags or signals are received, they notify the controller that they need to be serviced.
- The polling method is like a salesperson. The salesman goes from door to door while requesting to buy a product or service.
  - Similarly, the processor keeps monitoring the flags or signals one by one for all devices and provides service to whichever component that needs its service.

Cont..

- Interrupt signals can cause a program to suspend itself temporarily to service the interrupt by branching into another program called Interrupt Service Routines (ISR) for the specified device which has caused the interrupt.
- When the ISR completes, the processor returns to the work that was interrupted.
- The interrupts can be either hardware interrupts or software interrupts.

Hardware Interrupts

- Hardware Interrupts: A hardware interrupt is an electronic alerting signal sent to the processor from an external device, like a disk controller or an external peripheral.
  - For example, when we press a key on the keyboard, it triggers hardware interrupt which causes the processor to read the keystroke.

Software Interrupts

- Software Interrupts: A software interrupt is caused either by an exceptional condition or a special instruction in the instruction set which causes an interrupt when it is executed by the processor.
  - For example, if the processor's arithmetic logic unit runs a command to divide a number by zero, to cause a divide-by-zero exception, thus causing the computer to abandon the calculation or display an error message.
  - Software interrupt instructions work similar to subroutine calls.

Interrupt Service Routine (ISR)

- For every interrupt, there must be an **interrupt service routine (ISR),** or interrupt handler.
- ISR examines an interrupt and determines how to handle it executes the handling, and then returns a logical interrupt value.
- When an interrupt occurs, the microcontroller runs the interrupt service routine.
- For every interrupt, there is a fixed location in memory that holds the address of its interrupt service routine, ISR.
  - The table of memory locations that holds the addresses of ISRs is called the **Interrupt Vector Table**.

Steps to Execute an Interrupt

- When an interrupt gets active, the microcontroller goes through the following steps:
  - The microcontroller closes the currently executing instruction and **saves** the address of the next instruction **(PC)** on the stack.
  - It jumps to the memory location of the interrupt vector table that holds the address of the interrupts service routine.

Steps to Execute an Interrupt

- The microcontroller gets the address of the ISR from the interrupt vector table and jumps to it.
  - It starts to execute the ISR until it reaches the last instruction of the subroutine which is RETI (return from interrupt).
- Upon executing the RETI instruction, the microcontroller returns to the location where it was interrupted.
  - First, it gets the program counter (PC) address from the stack by popping the top bytes of the stack into the PC.
  - Then, it start to execute from that address.

Enabling and Disabling an Interrupt

- Upon **Reset**, all the interrupts are disabled even if they are activated.
- The interrupts must be enabled using software in order for the microcontroller to respond to those interrupts.
- A special function register called IE (interrupt enable) register is responsible for enabling and disabling the interrupt.
  - IE is a bit-addressable register.

IE (Interrupt Enable) Register of 8051

MSB LSB

EA	--	--	ES	ET1	EX1	ET0	EX0

EA	**IE.7**	**Disables all interrupts**
--	**IE.6**	**Not implemented, reserved for future use**
ET2	**IE.5**	**Enables or disables timer 2 overflow or capture interrupt (8952)**
ES	**IE.4**	**Enables or disables the serial port interrupt**
ET1	**IE.3**	**Enables or disables timer 1 overflow interrupt**
EX1	**IE.2**	**Enables or disables external interrupt 1**
ET0	**IE.1**	**Enables or disables timer 0 overflow interrupt**
EX0	**IE.0**	**Enables or disables external interrupt 0**

Interrupt Priorities

- What if two interrupt sources interrupt at the same time?
- Each interrupt source can also be individually programmed to one of the two priority levels by setting or clearing a bit in the SFR named IP (Interrupt Priority).
- A low-priority interrupt can be interrupted by a high-priority interrupt.
- If two interrupt requests of different priority levels are received simultaneously, the request of higher priority is serviced.

Interrupt Priority in 8051

- If interrupt requests of the same priority level are received simultaneously, an internal polling sequence determines which request is serviced.
- When the 8051 is powered up, the priorities are assigned according to the following.

Highest To Lowest Priority	
External Interrupt 0	(INT0)
Timer Interrupt 0	(TF0)
External Interrupt 1	(INT1)
Timer Interrupt 1	(TF1)
Serial Communication	(RI + TI)

Further Reading ….

- Micro Controllers families
- How to use Assembly Language for microcontroller simulation
- Register Banks(stack in RAM)
- Interrupt priorities