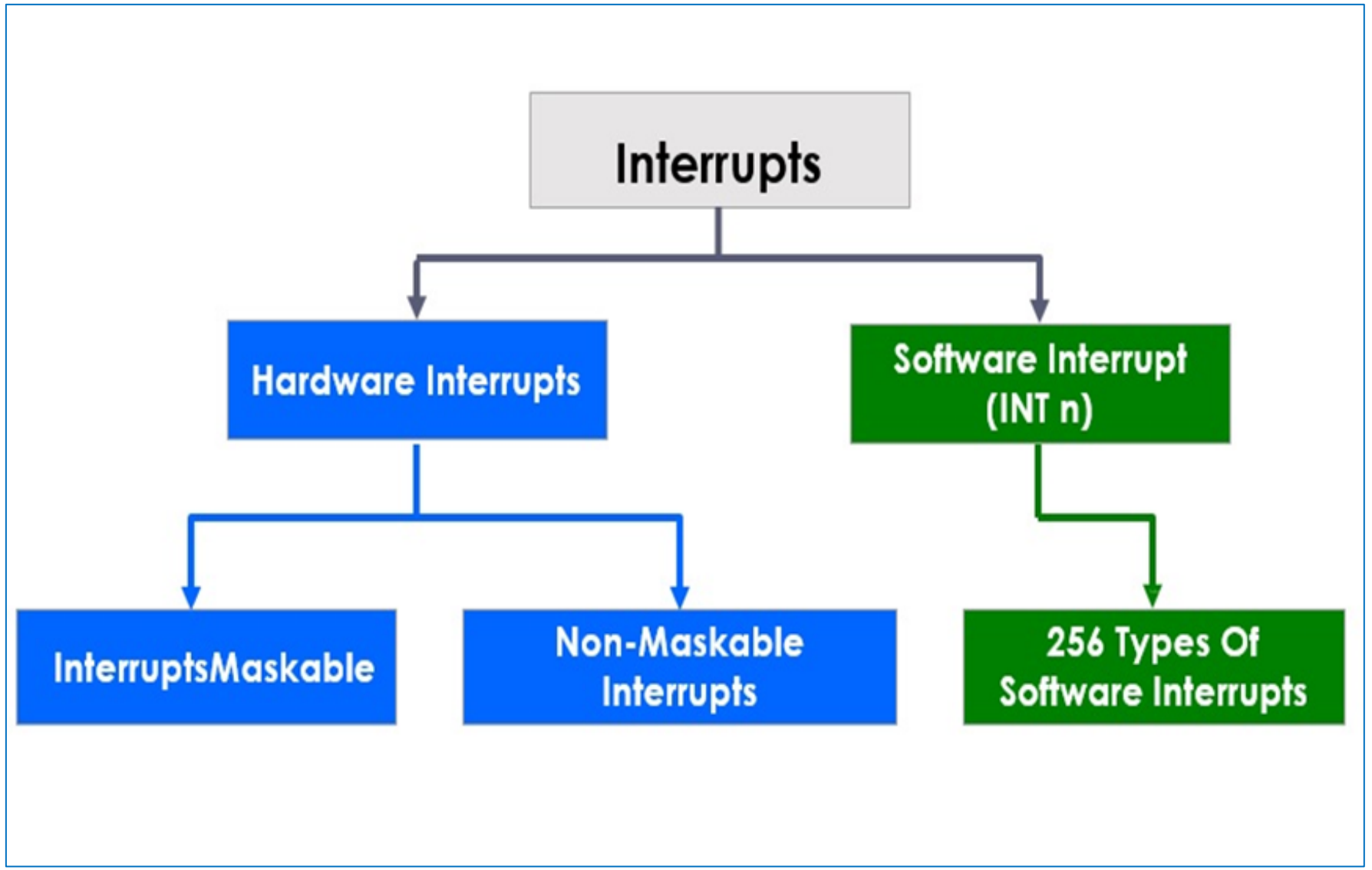
Interrupts

- An interrupt is a condition that halts the microprocessor temporarily to work on a different task and then return to its previous task. Interrupt is an event or signal that request to attention of CPU. This halt allows peripheral devices to access the microprocessor.
- Whenever an interrupt occurs the processor completes the execution of the current instruction and starts the execution of an Interrupt Service Routine (ISR) or Interrupt Handler.
- ISR is a program that tells the processor what to do when the interrupt occurs. After the execution of ISR, control returns back to the main routine where it was interrupted.

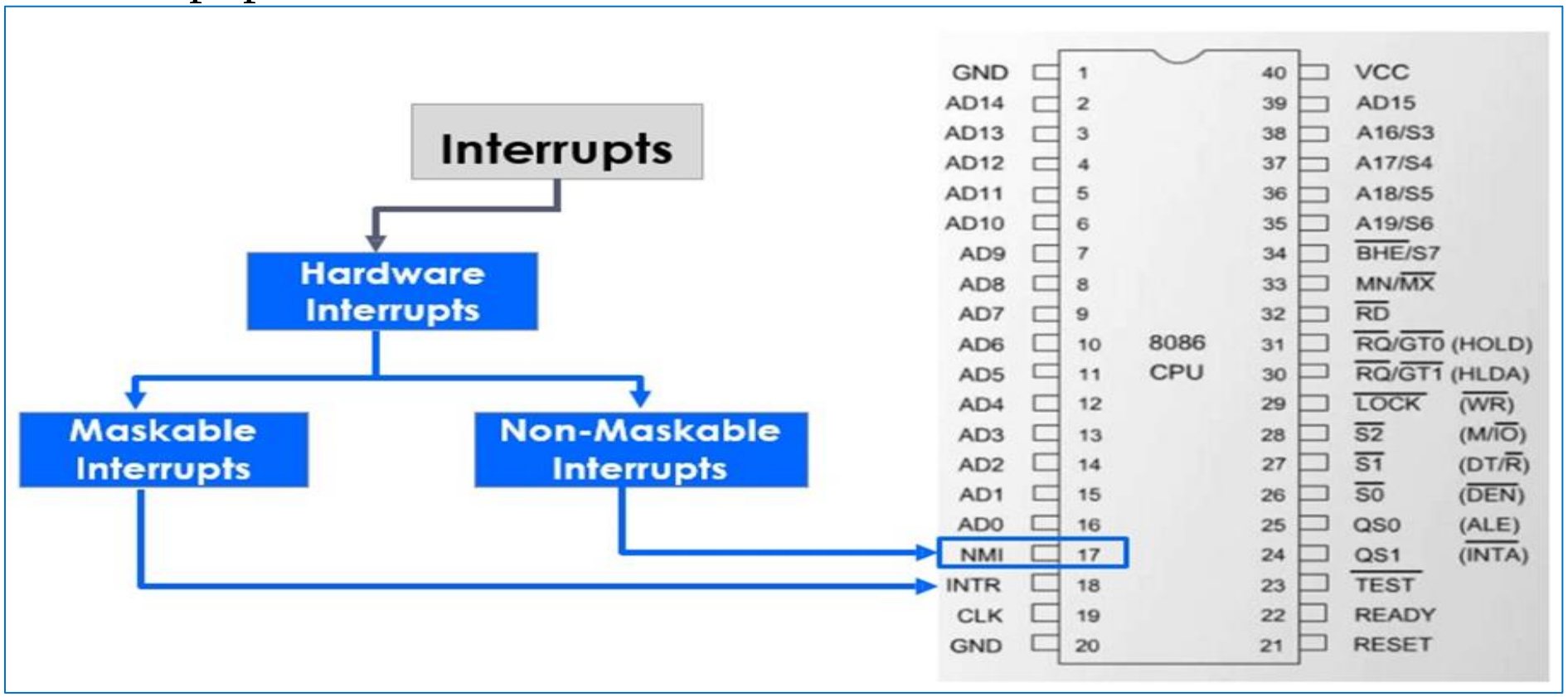
Interrupts

- Broadly the interrupts are divided into two types. They are External (Hardware) Interrupts and Internal (Software) Interrupts.
- The hardware interrupts are classified as non-maskable and maskable interrupts.
- The hardware interrupt is caused by any peripheral device by sending a signal through a specified pin to the microprocessor.
- Whereas internal interrupts are initiated by the state of the CPU (e.g. divide by zero error) or by an instruction. So, the software interrupt is one which interrupts the normal execution of a program of the microprocessor.
- The 8086 has two hardware interrupt pins namely NMI and INTR. In the two, the NMI is a non-maskable interrupt and the INTR interrupt request is a maskable interrupt which has lower priority .The third pin associated with the hardware interrupts are the INTA called interrupt acknowledge.

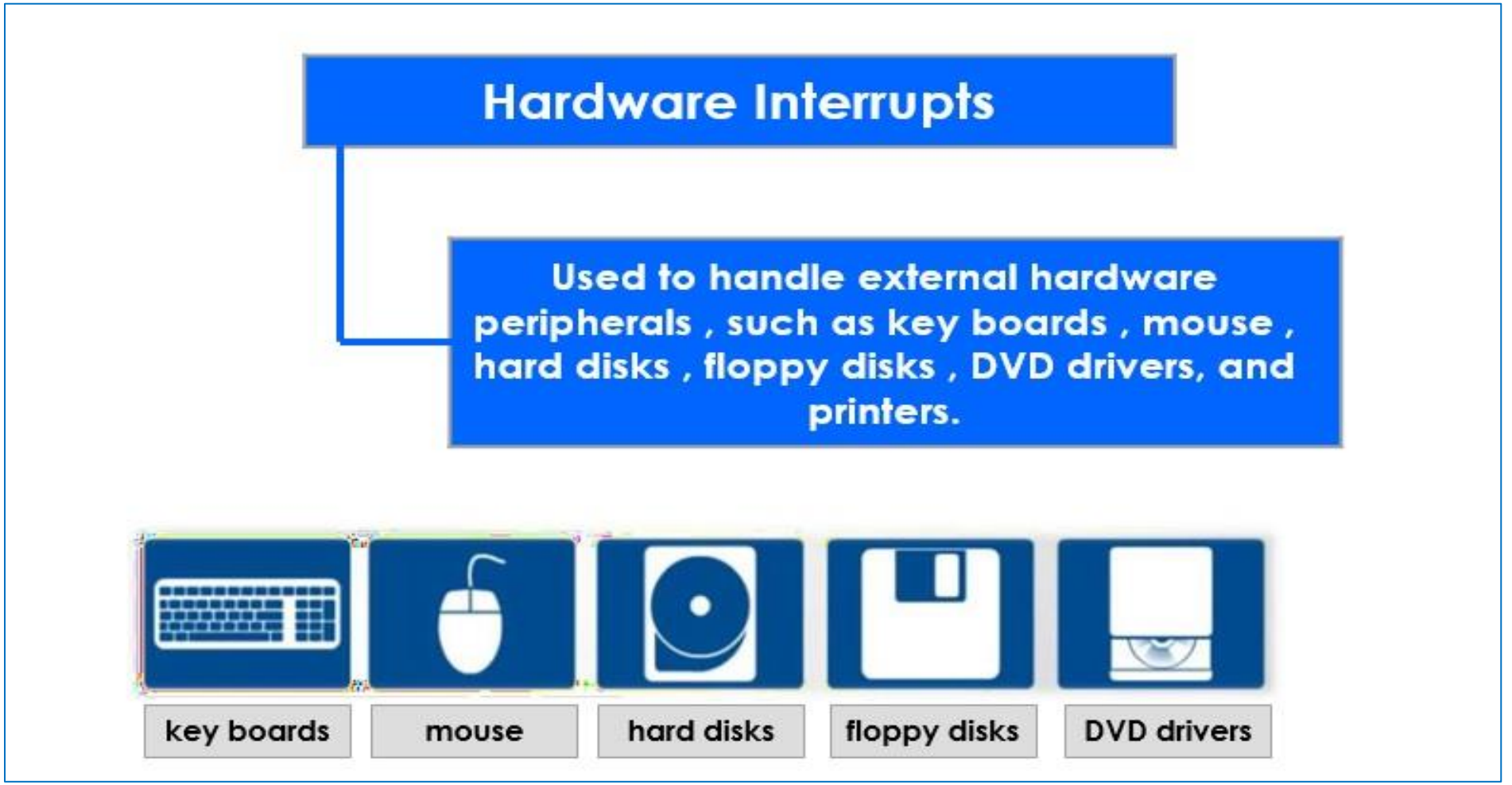

Interrupts
Hardware Interrupts
Software Interrupt (INT n)
InterruptsMaskable
Non-Maskable Interrupts
256 Types Of Software Interrupts

Hardware Interrupts

- The interrupts initiated by external hardware by sending an appropriate signal to the interrupt pin of the processor is called hardware interrupt. The 8086 processor has two interrupt pins INTR and NMI.



Hardware Interrupts



Hardware Interrupts

Maskable and Non-Maskable Interrupts

- The processor has the facility for accepting or rejecting hardware interrupts. Programming the processor to reject an interrupt is referred to as masking or disabling and programming the processor to accept an interrupt is referred to as unmasking or enabling.
- In 8086 the interrupt flag (IF) can be set to one to unmask or enable all hardware interrupts and IF is cleared to zero to mask or disable a hardware interrupts except NMI.
- The interrupts whose request can be either accepted or rejected by the processor are called maskable interrupts.

Hardware Interrupts

Maskable and Non-Maskable Interrupts

- The interrupts whose request has to be definitely accepted (or cannot be rejected) by the processor are called non-maskable interrupts. Whenever a request is made by non-maskable interrupt, the processor has to definitely accept that request and service that interrupt by suspending its current program and executing an ISR. In 8086 processor all the hardware interrupts initiated through INTR pin are maskable by clearing interrupt flag (IF). The interrupt initiated through NMI pin and all software interrupts are non-maskable.
- The programmer cannot control when a Non-Maskable Interrupts is serviced and the processor has to stop the main program to execute the NMI service routine.

Hardware Interrupts

Maskable and Non-Maskable Interrupts

- Maskable Interrupts the programmer can choose to mask specific interrupts and re-enable them later.
- Non-Maskable Interrupts used :

1. during power failure
2. during critical response time
3. during non-recoverable hardware errors
4. watchdog interrupt
5. during memory parity errors

Software interrupts

Coming to the software interrupts, 8086 can generate 256 interrupt types through the instruction INT n .Any of the 256 interrupt types can be generated by specifying the interrupt type after INT instruction. For example the first five types are as follows:

- TYPE 0 interrupt represents division by zero situation.
- TYPE 1 interrupt represents single-step execution during the debugging of a program.
- TYPE 2 interrupt represents non-maskable NMI interrupt.
- TYPE 3 interrupt represents break-point interrupt.
- TYPE 4 interrupt represents overflow interrupt.

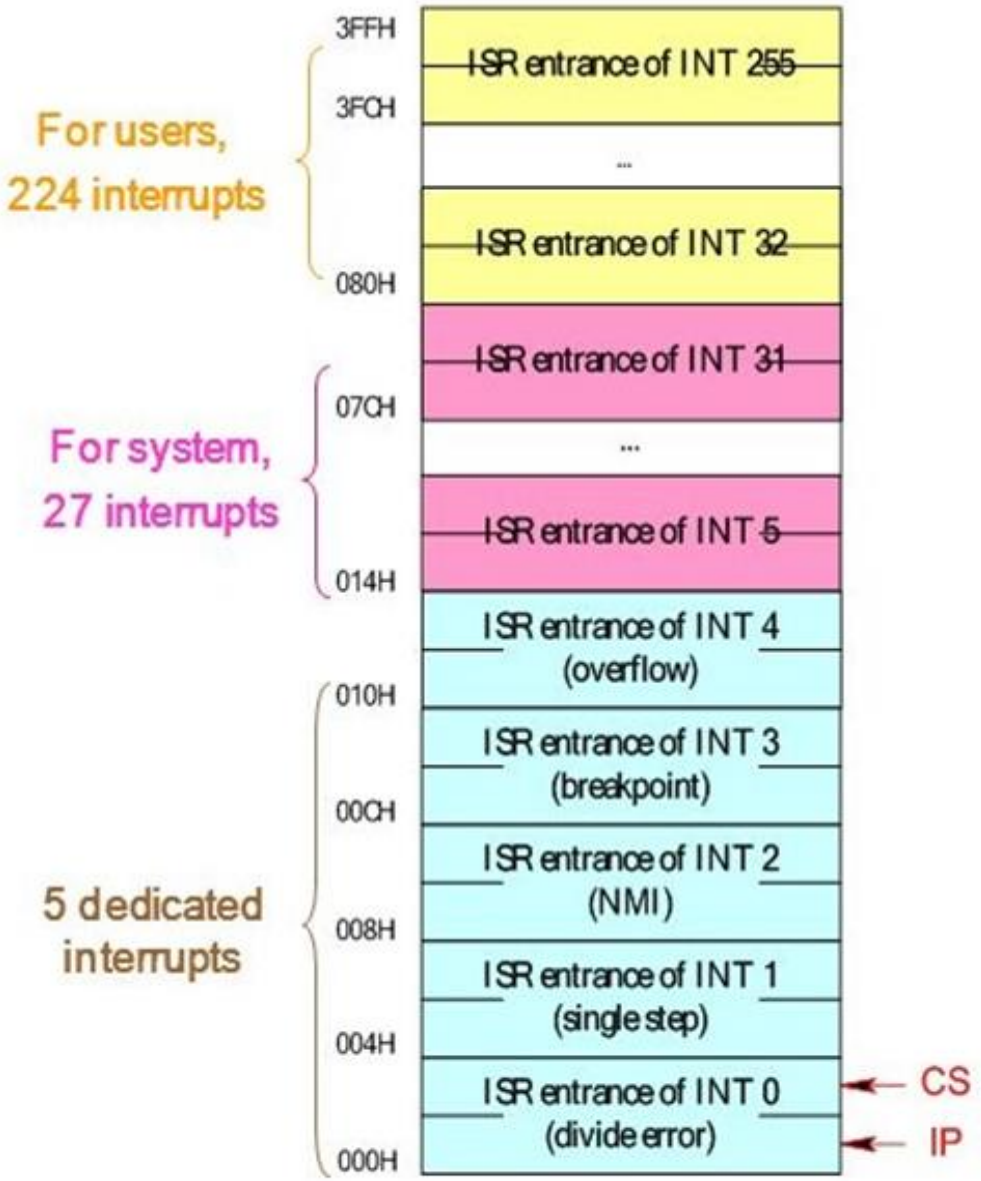
The interrupts from Type 5 to Type 31 are reserved for other advanced microprocessors, and interrupts from 32 to Type 255 are available for hardware and software interrupts.

Interrupt Vector Table

- **Interrupt Vector Table** on 8086 is a vector that consists of 256 total interrupts placed at first 1 kb of memory from 0000h to 03ffh, where each vector consists of segment and offset address for ISR.

- The call to interrupt service routine is similar to far procedure call.

- The size for each interrupt vector is 4 bytes (2 word in 16 bit), where 2 bytes (1 word) for segment and 2 bytes for offset of interrupt service routine address. So it takes 1024 bytes (1 kb) memory for interrupt vector table.



- 256 interrupts
  - 0 ~ 4 dedicated
  - 5 ~ 31reserved for system use
    - 08H ~ 0FH : 8259A
    - 10H ~ 1FH : BIOS
  - 32 ~ 255 reserved for users
    - 20H ~ 3FH : DOS
    - 40H ~ FFH : open

Interrupt Vector Table

When an interrupt occur, the following action are taken:

1. Push flag register on the stack
2. clear IF and TF
3. Push CS and IP register, on the stack
4. Load CS with the 16-bit data at memory address (INT-type *4+2)
5. Load IP with the 16 bit data at memory address (INT-type *4).

The last instruction of ISR is (IRET) instruction, it actions are:

1. POP the 16-value on top of stack into IP register
2. POP the 16-value on top of stack into CS register
3. POP the 16-value on top of stack into flag register.

How can more than one device interrupt?

- Computer typically have more than one I/O device requesting interrupt service, like keyboard, hard disk, floppy disk, printer all generate an INT when they required the attention to CPU.
- When more than one device INT CPU, we need a mechanism to priority these INT (if they come at the same time) and forward only one INT request at a time to the CPU while keeping other INT request pending for their service.

Examples of some interrupt instructions

1. INT 10h / AH=0Eh teletype output (print Al value)

- Ex: MOV AL, 'A'

  MOV AH, 0Eh

  INT 10h

2. INT 21h / AH=01h

Read character from standard input with echo, result is stored in AL.

- EX: MOV AH, 01h

  INT 21h

Examples of some interrupt instructions

3. INT 21h / AH=02h

Write character to standard output entry DL= character to write, after execution AL=DL

- EX:

```
MOV AH, 02h
MOV DL, 'a'
INT 21h
```

Ex: Write assembly program to read five value from keyboard and store it in array

```
Org 100h
Mov ah, 01h
Mov cx, 5
aa: int 21h
Mov a[si], al
Inc si
Loop aa
Ret
a db 5 dup (0)
```

Ex: Write assembly program to print the value of array a where a= 1, 2, 3, 4, 5

```
Org 100h
Mov ah, 2
Mov cx, 5
aa: Mov Dl, a[si]
Int 21h
Inc si
Loop aa
Ret
a db 1, 2, 3, 4, 5
```

Examples of some interrupt instructions

4. INT 33h (mouse driver interrupt)

INT 33h / AX= 0001h (show mouse pointer)

- Ex: Mov ax, 1
  Int 33h

5. INT 33h / AX=0002h (Hide visible mouse pointer)

Ex: Mov ax, 2
Int 33h