8051 Interrupts

Interrupt Fundamentals

- During program execution, if peripheral device needs service from microcontroller device will generate interrupt and then gets the service from microcontroller.
- When peripheral device activate the interrupt signal, the processor branches to a program called interrupt service routine.
- After executing the interrupt service routine the processor returns to the main program.

Steps taken by processor while processing an interrupt:

1. It completes the execution of current instruction.
2. PSW is pushed to stack.
3. PC content is pushed to stack.
4. Interrupt flag is reset.
5. PC is loaded with ISR address.

ISR will always ends with RET instruction. The execution of RET instruction results in –

i. POP the current stack top to PC.

ii. POP the current stack top to PSW.

Interrupt Fundamentals

- Whenever any peripheral device needs microcontroller’s service, the device notifies the microcontroller by sending it an interrupt signal.
- An interrupt is an external or internal event that interrupts the microcontroller to inform it that a device needs it service.
- Interrupt can be a hardware or a software interrupt.
- Interrupts can be disabled or enabled.

Interrupt Fundamentals

- Upon receiving an interrupt signal, microcontroller finishes the instruction execution and serves the device.
- The program which is associated with the interrupt is called the interrupt service routine.
- For every interrupt, there is a fixed location in memory for ISR.
- The group of memory locations set aside to hold the addresses of ISRs is called interrupt vector table.

Attending Interrupts

1. The microcontroller finishes the instruction it is executing and saves the address of the next instruction (PC) on the stack.
2. It also saves the current status of all the interrupts internally.
3. It gets the address of the ISR from the interrupt vector table and jumps to it.
4. It starts to execute the interrupt service subroutine until it reaches the last instruction of the subroutine which is RETI (return from interrupt).
5. Upon executing the RETI instruction, the microcontroller returns to the place where it was interrupted.

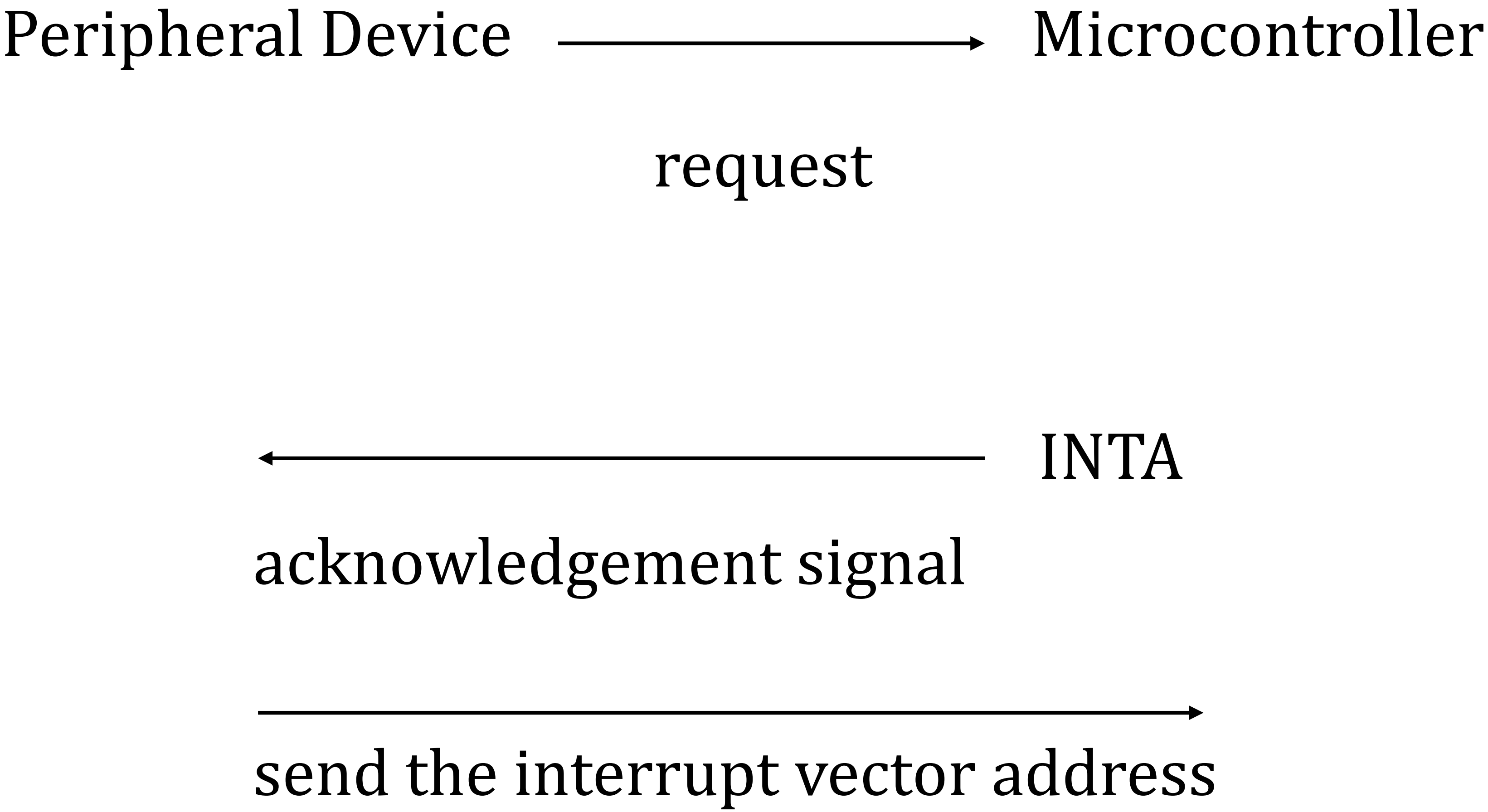
8051 Interrupts

There are six interrupts in 8051.

- Power on reset
- Two external interrupts (INT0’ and INT1’): P3.2 and P3.3
- Two timer interrupt (for Timer0 and Timer1)
- Serial port interrupt (for both transmit and receive)

Classification of interrupts:

1. **External and Internal interrupts:** External interrupts are those initiated by peripheral devices through the external pins of microcontroller. Internal interrupts are activated by internal peripherals of microcontrollers.
2. **Maskable and Non-maskable interrupts:** Maskable interrupts can be disabled by the processor using program. Programmer cannot disable non-maskable interrupts by program.
3. **Vectored and Non-vectored interrupts:** In vectored interrupts, the starting address (vector address) is predefined. In non-vectored interrupts the starting address is provided by the peripherals.



Peripheral Device

Microcontroller

request

INTA

acknowledgement signal

send the interrupt vector address

8051 Interrupt Structure:

There are five interrupts. Two external interrupts and three internal interrupts. All these interrupts are maskable and vectored interrupts.

- External interrupt 0 (0003 H) – Highest Priority
- Timer 0 interrupt (000B H)
- External interrupt 1 (0013 H)
- Timer 1 interrupt (001B H)
- Serial interrupt (0023 H) – Lowest Priority

8051 makes use of two registers to deal with interrupts:

1. IE – Interrupt Enable
2. IP – Interrupt Priority

Both are 8-bit registers.

Interrupt Table

Interrupt	Type of Interrupt	Pin/SFR bit	ROM Location
Reset	Hardware	9	0000 H
INT0	Hardware	P3.2 (12)	0003 H
Timer0 (TF0)	Software	TCON.5	000B H
INT1	Hardware	P3.3 (13)	0013 H
Timer1 (TF1)	Software	TCON.7	001B H
Serial Com (RI and TI)	Software	SCON.0 and SCON.1	0023 H

Enable/Disable Interrupt

- Upon reset, all interrupts are disabled (masked)
- The interrupts must be enabled by software in order for the microcontroller to respond to them
- IE (Interrupt Enable) register is responsible for enabling (unmasking) and disabling (masking) the interrupts

IE : Interrupt Enable Register (Bit Addressable)

If the bit is 0, the corresponding interrupt is disabled. If the bit is 1, the corresponding interrupt is enabled.

EA	–	–	ES	ET1	EX1	ET0	EX0

EA	IE.7	Disables all interrupts. If EA = 0, no interrupt will be acknowledged. If EA = 1, interrupt source is individually enable or disabled by setting or clearing its enable bit.
-	IE.6	Not implemented, reserved for future use*.
-	IE.5	Not implemented, reserved for future use*.
ES	IE.4	Enable or disable the Serial port interrupt.
ET1	IE.3	Enable or disable the Timer 1 overflow interrupt.
EX1	IE.2	Enable or disable External interrupt 1.
ET0	IE.1	Enable or disable the Timer 0 overflow interrupt.
EX0	IE.0	Enable or disable External Interrupt 0.

Hardware Interrupts: INT0’ and INT1’

Bit	D7	D6	D5	D4	D3	D2	D1	D0
TCON	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

Four bits (lower nibble) of TCON register are used for external interrupts INT0’ and INT1’.

IT1/IT0: Level/Edge triggering of external interrupt.

- ❑ 0 – Low logic level triggered;
- ❑ 1 – Falling edge triggered;

IE1/IE0: Set when INT1/INT0 is detected.

Timer Interrupts:

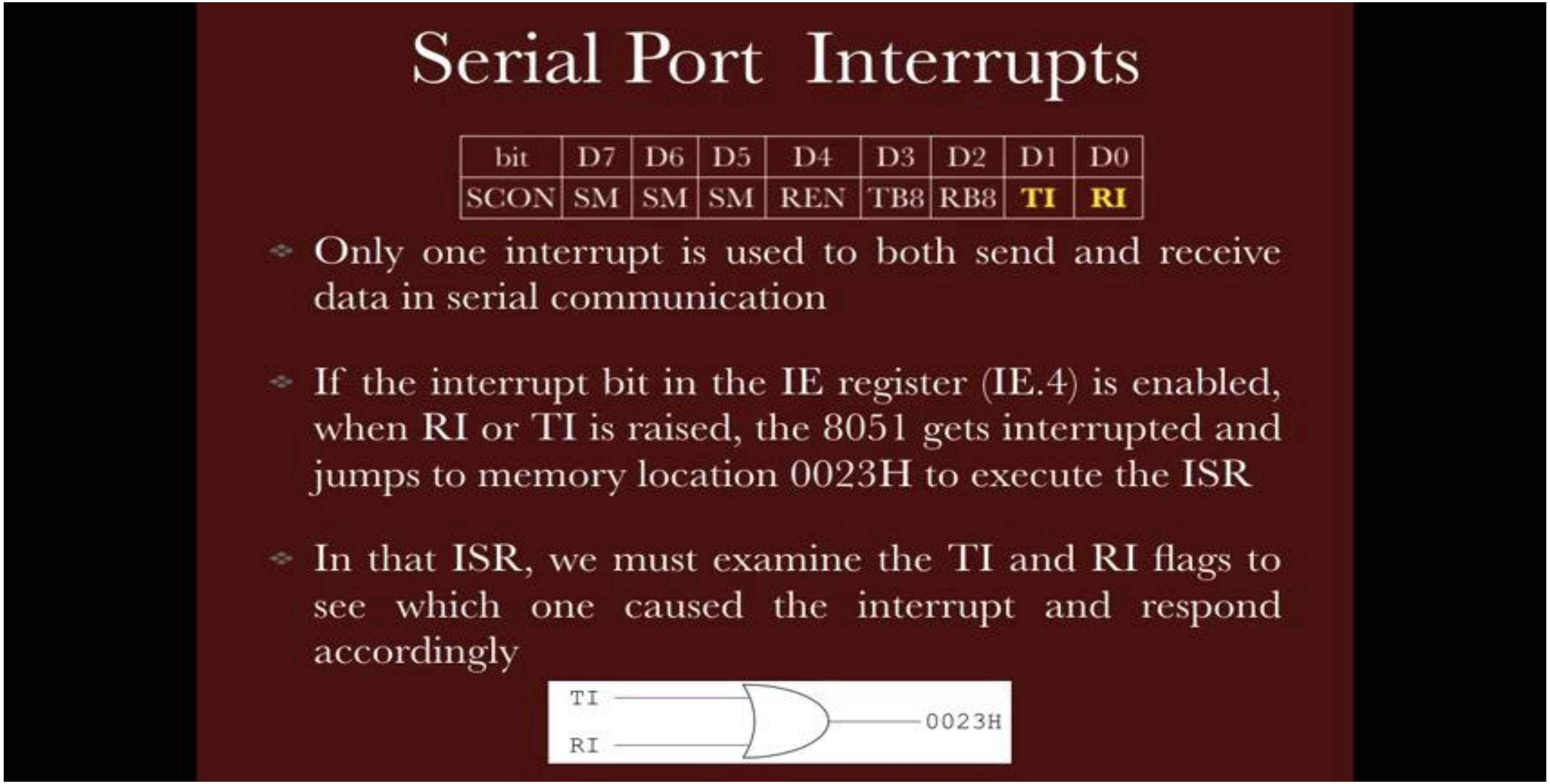
Bit	**D7**	**D6**	**D5**	**D4**	**D3**	**D2**	**D1**	**D0**
TCON	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

First four bits (higher nibble) of TCON register are used for timer interrupts.

TR1/TR0: Turning on/off Timer 1/0.

TF1/TF0: Overflow of Timer 1/0.

If the timer interrupt in the IE register is enabled, whenever TF is raised, the microcontroller jumps to the interrupt vector table to service the ISR.



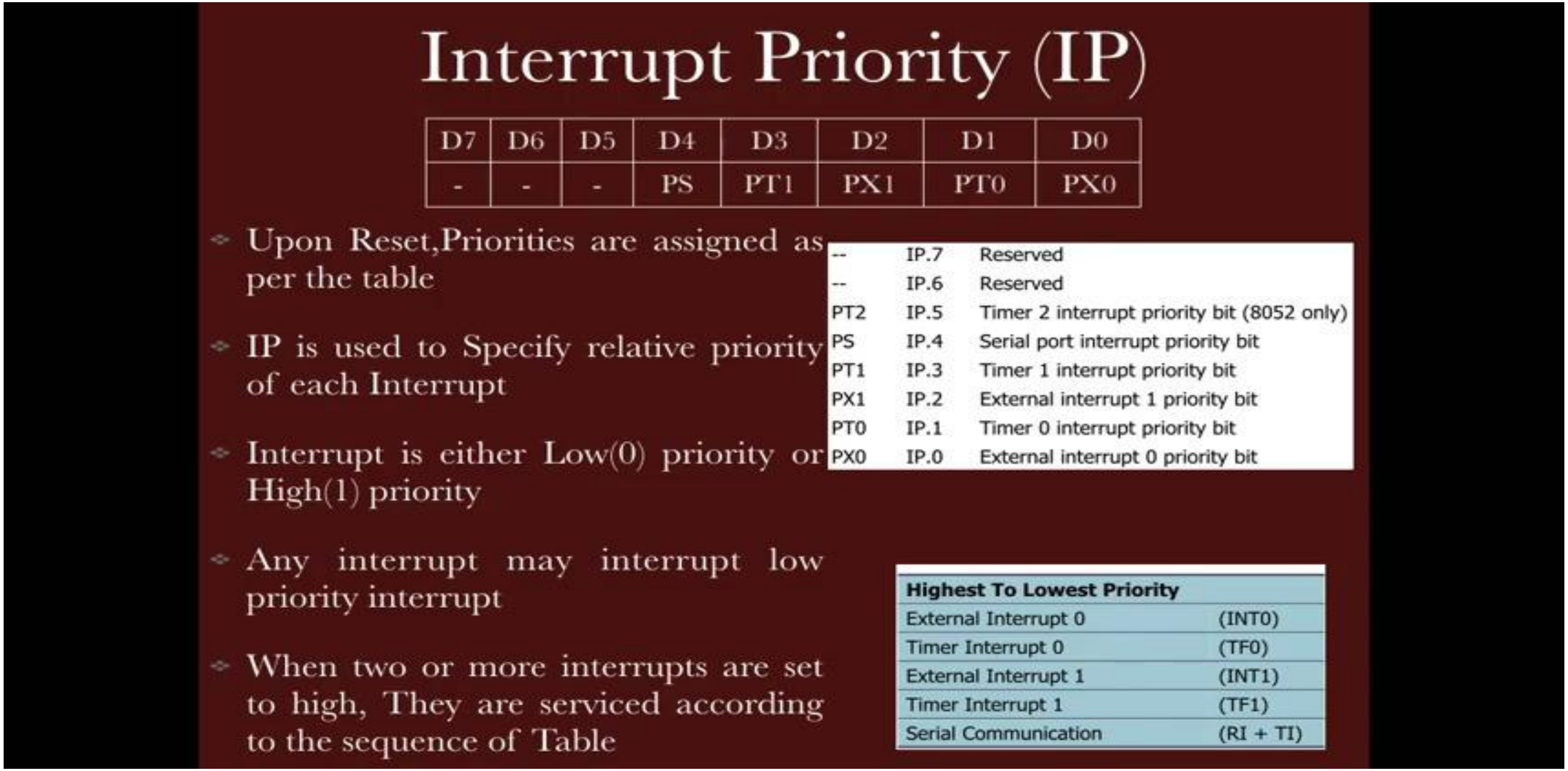
Serial Port Interrupts

bit	D7	D6	D5	D4	D3	D2	D1	D0
SCON	SM	SM	SM	REN	TB8	RB8	**TI**	**RI**

- Only one interrupt is used to both send and receive data in serial communication

- If the interrupt bit in the IE register (IE.4) is enabled, when RI or TI is raised, the 8051 gets interrupted and jumps to memory location 0023H to execute the ISR

- In that ISR, we must examine the TI and RI flags to see which one caused the interrupt and respond accordingly



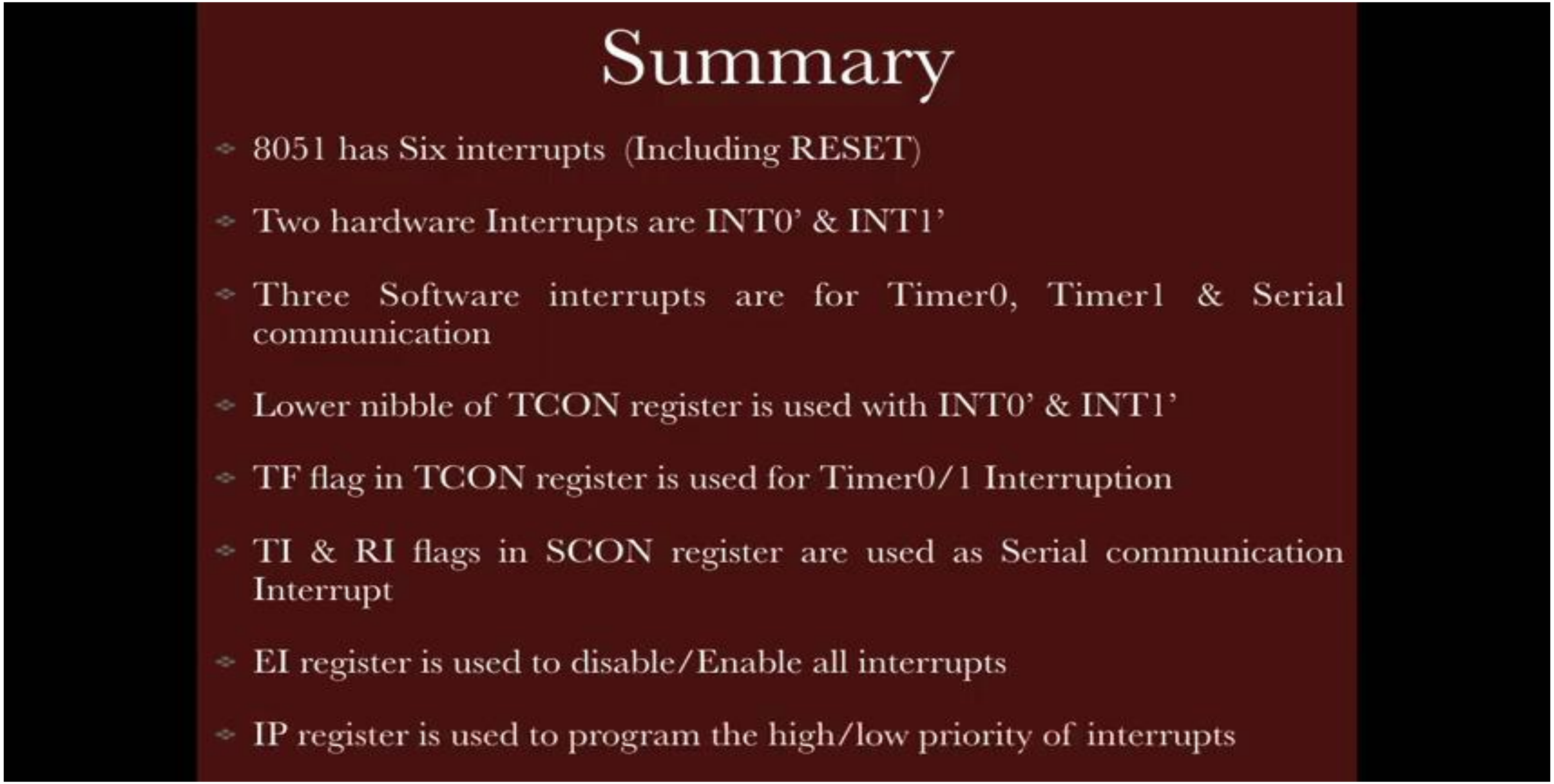
Interrupt Priority (IP)

D7	D6	D5	D4	D3	D2	D1	D0
-	-	-	PS	PT1	PX1	PT0	PX0

- Upon Reset,Priorities are assigned as per the table
- IP is used to Specify relative priority of each Interrupt
- Interrupt is either Low(0) priority or High(1) priority
- Any interrupt may interrupt low priority interrupt
- When two or more interrupts are set to high, They are serviced according to the sequence of Table

--	IP.7	Reserved
--	IP.6	Reserved
PT2	IP.5	Timer 2 interrupt priority bit (8052 only)
PS	IP.4	Serial port interrupt priority bit
PT1	IP.3	Timer 1 interrupt priority bit
PX1	IP.2	External interrupt 1 priority bit
PT0	IP.1	Timer 0 interrupt priority bit
PX0	IP.0	External interrupt 0 priority bit

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



Summary

- 8051 has Six interrupts (Including RESET)
- Two hardware Interrupts are INT0’ & INT1’
- Three Software interrupts are for Timer0, Timer1 & Serial communication
- Lower nibble of TCON register is used with INT0’ & INT1’
- TF flag in TCON register is used for Timer0/1 Interruption
- TI & RI flags in SCON register are used as Serial communication Interrupt
- EI register is used to disable/Enable all interrupts
- IP register is used to program the high/low priority of interrupts