

8051 Microcontroller



Introduction to Microcontrollers

Microprocessor vs. Microcontroller

Microprocessor

- CPU is stand-alone, RAM, ROM, I/O, timer are separate
- Designer can decide on the amount of ROM, RAM and I/O ports.
- Expansive
- Versatility
- General-purpose

Microcontroller

- CPU, RAM, ROM, I/O and timer are all on a single chip
- Fix amount of on-chip ROM, RAM, I/O ports
- For applications in which cost, power and space are critical
- Not Expansive
- Single-purpose

8051 CPU Operation

1. Features

2. Pin Diagram

3. Block Diagram

8051 Microcontroller

- Intel introduced 8051, referred as MCS- 51, in 1981.
- The 8051 is an **8-bit processor**
 - The CPU can work on only 8 bits of data at a time
- The 8051 became widely popular after allowing other manufactures to make and market any flavor of the 8051.

8051 Family

- The 8051 is a subset of the 8052
- The 8031 is a ROM-less 8051
 - Add external ROM to it
 - You lose two ports, and leave only 2 ports for I/O operations

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

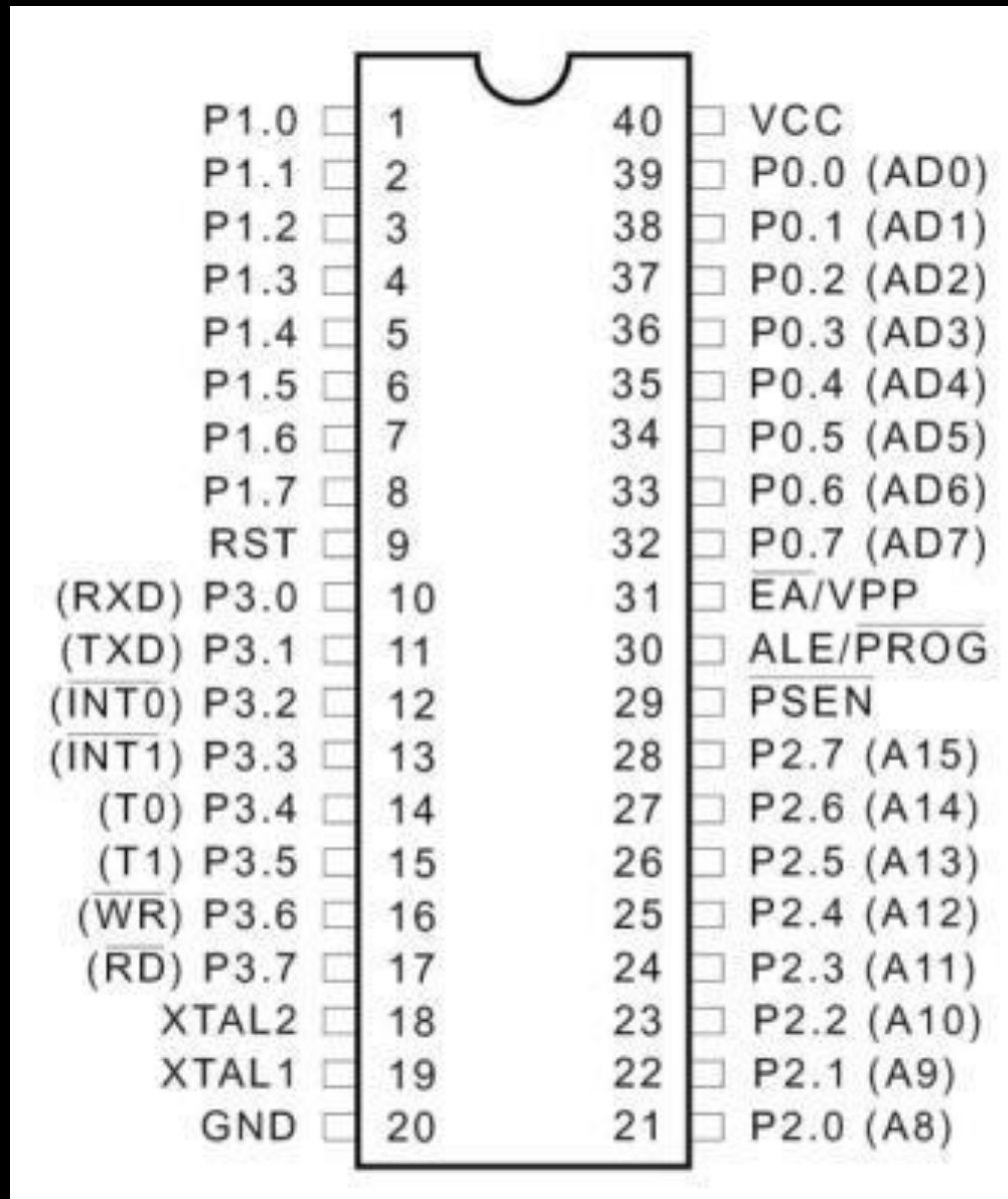
8051 Features

- *64KB Program Memory address space*
- *64KB Data Memory address space*
- *4K bytes of on-chip Program Memory*
- *128 bytes of on-chip Data RAM*
- *32 bidirectional and individually addressable I/O lines*
- *Two 16-bit timer/counters*
- *Full duplex UART*
- *6-source/5-vector interrupt structure with two priority levels*
- *On-chip clock oscillator*

Pin Description of the 8051

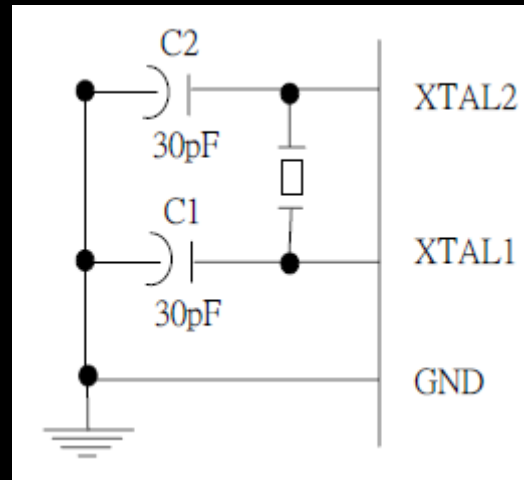
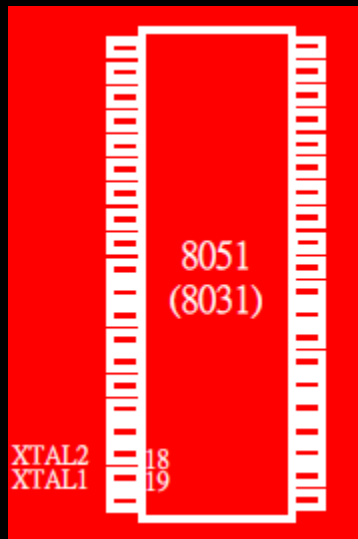
- 8051 family members (e.g., 8751, 89C51, 89C52, DS89C4x0)
 - Have **40 pins** dedicated for various functions such as I/O, RD, WR, address, data, and interrupts.
 - Come in different packages, such as
 - *DIP(dual in-line package),*
 - *QFP(quad flat package), and*
 - *LLC(leadless chip carrier)*
- Some companies provide a 20-pin version of the 8051 with a reduced number of I/O ports for less demanding applications

Pin Diagram of the 8051



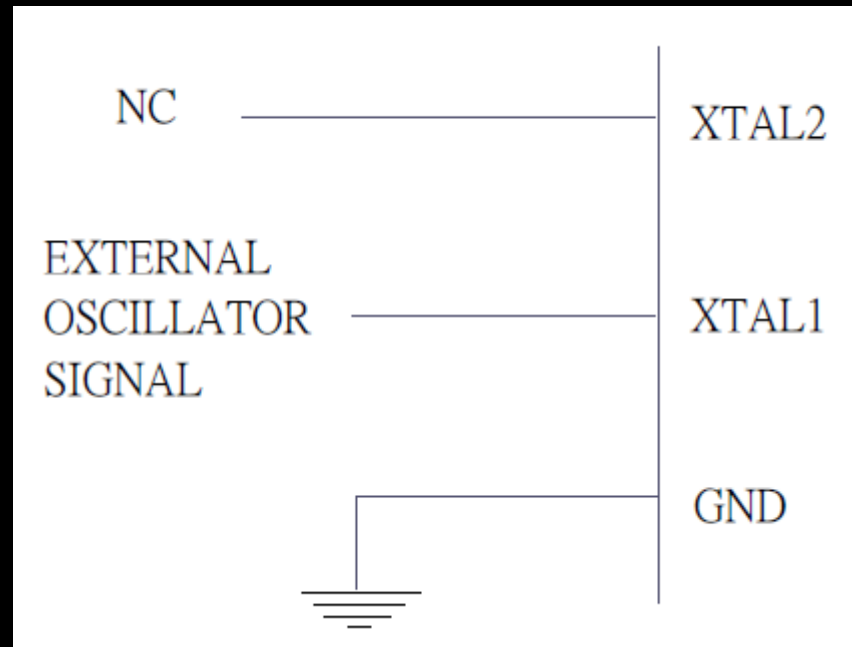
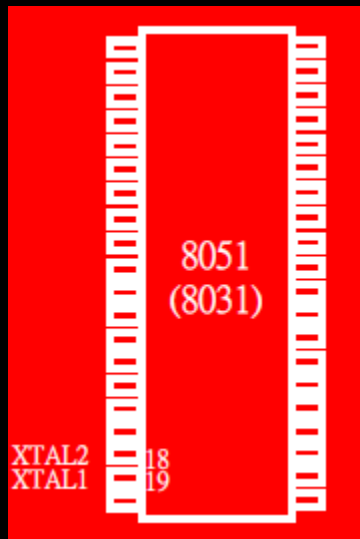
XTAL1 and XTAL2

- The 8051 has an on-chip oscillator but requires an external clock to run it
 - A quartz crystal oscillator is connected to inputs XTAL1 (pin19) and XTAL2 (pin18)
 - The quartz crystal oscillator also needs two capacitors of 30 pF value



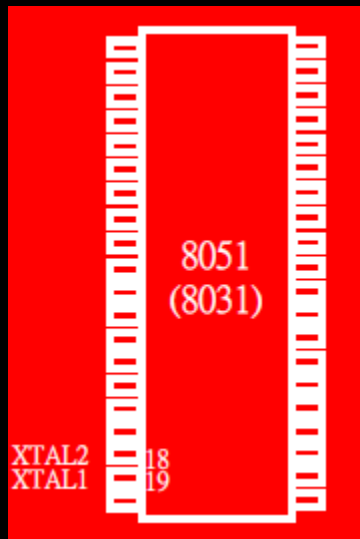
XTAL1 and XTAL2

- If you use a frequency source other than a crystal oscillator, such as a TTL oscillator:
 - It will be connected to XTAL1
 - XTAL2 is left unconnected



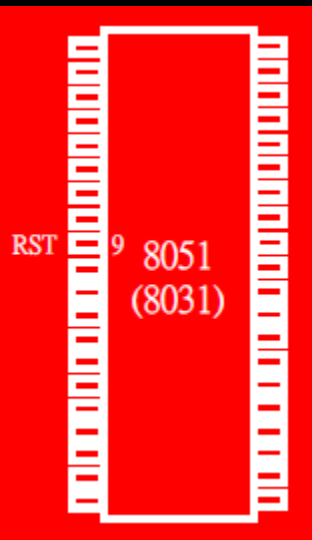
XTAL1 and XTAL2

- The **speed of 8051** refers to the maximum oscillator frequency connected to XTAL.
- We can observe the frequency on the XTAL2 pin using the oscilloscope.



RST

- RESET pin is an input and is active high (normally low)
- Upon applying a high pulse to this pin, the microcontroller will reset and terminate all activities
- This is often referred to as a power-on reset
- Activating a power-on reset will cause all values in the registers to be lost



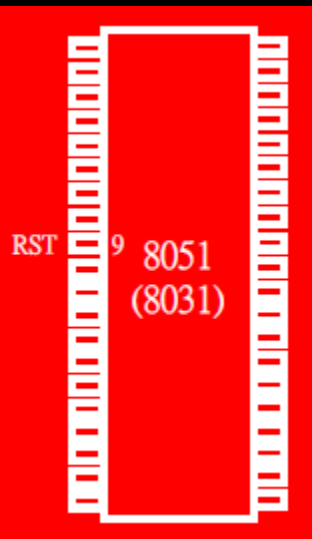
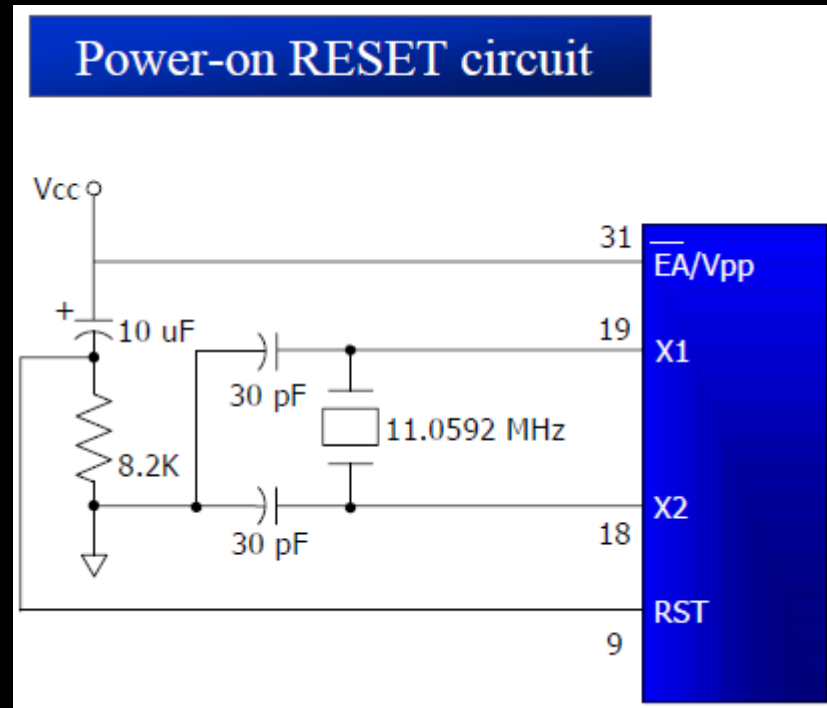
RESET value of some 8051 registers

we must place the first line of source code in ROM location 0

Register	Reset Value
PC	0000
DPTR	0000
ACC	00
PSW	00
SP	07
B	00
P0-P3	FF

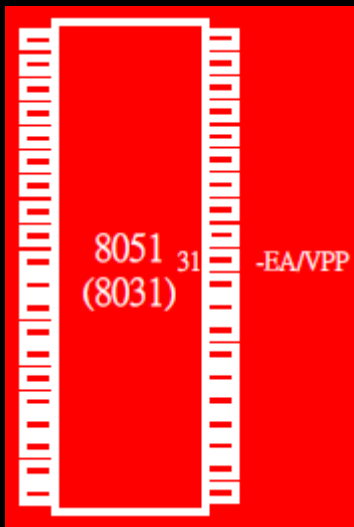
RST

- In order for the RESET input to be effective, it must have a minimum duration of 2 machine cycles.
- In other words, the high pulse must be high for a minimum of 2 machine cycles before it is allowed to go low.



EA'

- EA', “external access”, is an input pin and must be connected to Vcc or GND
- The 8051 family members all come with on-chip ROM to store programs and also have an external code and data memory.



- Normally EA pin is connected to Vcc
- EA pin must be connected to GND to indicate that the code or data is stored externally.

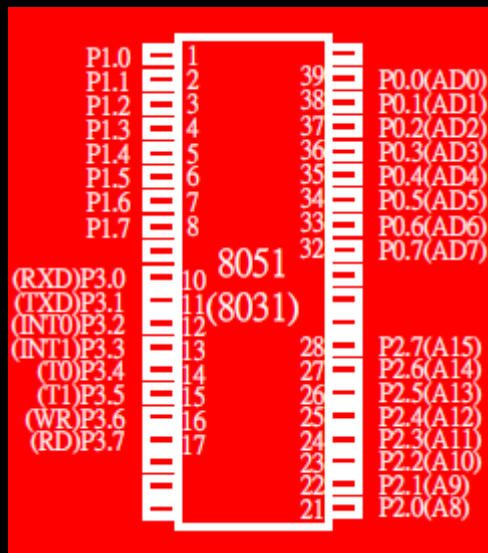
PSEN' and ALE

- PSEN, “program store enable”, is an output pin
- This pin is connected to the OE pin of the external memory.
- For External Code Memory, PSEN' = 0
- For External Data Memory, PSEN' = 1
- ALE pin is used for demultiplexing the address and data.



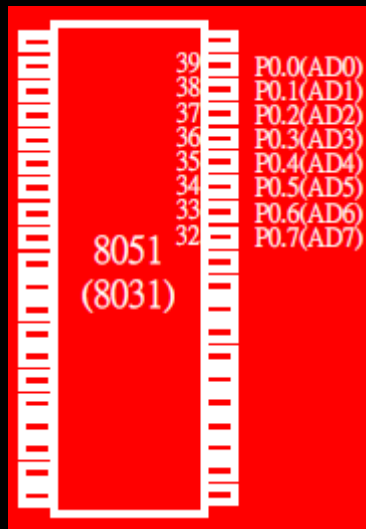
I/O Port Pins

- The four 8-bit I/O ports **P0**, **P1**, **P2** and **P3** each uses 8 pins.
- All the ports upon RESET are configured as output, ready to be used as input ports by the external device.

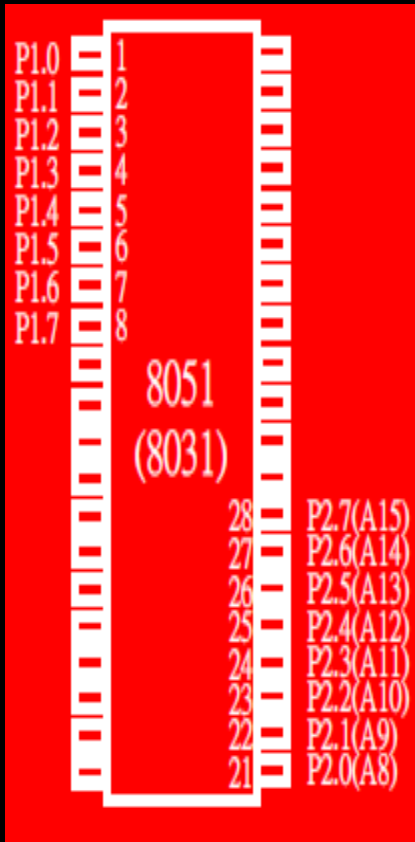


Port 0

- Port 0 is **also** designated as **AD0-AD7**.
- When connecting an 8051 to an external memory, port 0 provides both address and data.
- The 8051 multiplexes address and data through port 0 to save pins.
- **ALE** indicates if P0 has address or data.
 - When $ALE=0$, it provides data $D0-D7$
 - When $ALE=1$, it has address $A0-A7$



Port 1 and Port 2



- In 8051-based systems with no external memory connection:
 - Both P1 and P2 are used as simple I/O.
- In 8051-based systems with external memory connections:
 - Port 2 must be used along with P0 to provide the 16-bit address for the external memory.
 - P0 provides the lower 8 bits via A0 – A7.
 - P2 is used for the upper 8 bits of the 16-bit address, designated as A8 – A15, and it cannot be used for I/O.

Port 3

P3 Bit	Function	Pin
P3.0	RxD	10
P3.1	TxD	11
P3.2	$\overline{\text{INT0}}$	12
P3.3	$\overline{\text{INT1}}$	13
P3.4	T0	14
P3.5	T1	15
P3.6	$\overline{\text{WR}}$	16
P3.7	$\overline{\text{RD}}$	17

Serial communications

External interrupts

Timers

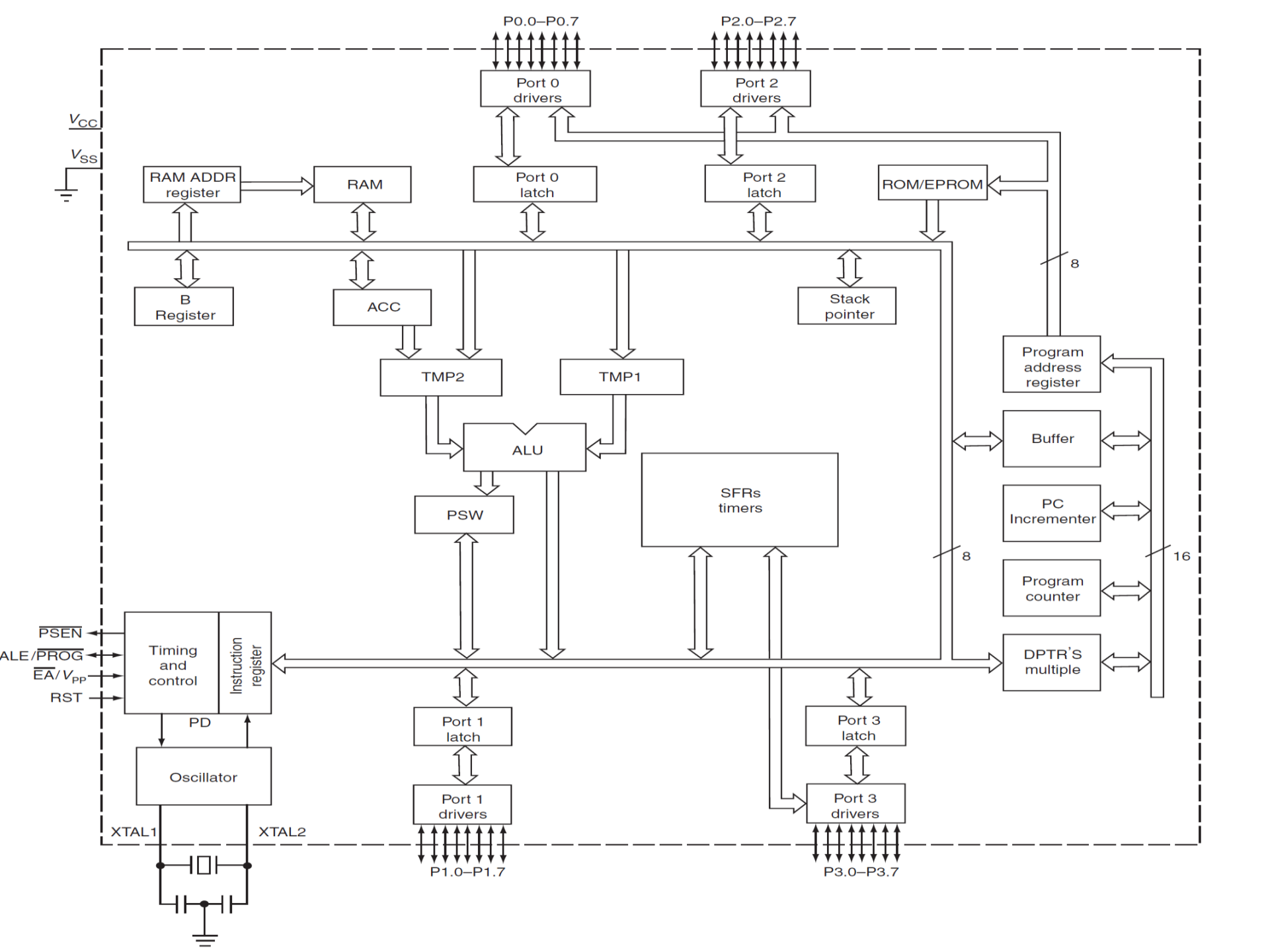
Read/Write signals of external memories

Pin Description Summary

PIN	TYPE	NAME AND FUNCTION
Vss	I	Ground: 0 V reference.
Vcc	I	Power Supply: This is the power supply voltage for normal, idle, and power-down operation.
P0.0 - P0.7	I/O	Port 0: Port 0 is an open-drain, bi-directional I/O port. Port 0 is also the multiplexed low-order address and data bus during accesses to external program and data memory.
P1.0 - P1.7	I/O	Port 1: Port 1 is an 8-bit bi-directional I/O port.
P2.0 - P2.7	I/O	Port 2: Port 2 is an 8-bit bidirectional I/O. Port 2 emits the high order address byte during fetches from external program memory and during accesses to external data memory that use 16 bit addresses.
P3.0 - P3.7	I/O	Port 3: Port 3 is an 8 bit bidirectional I/O port. Port 3 also serves special features as explained.

Pin Description Summary

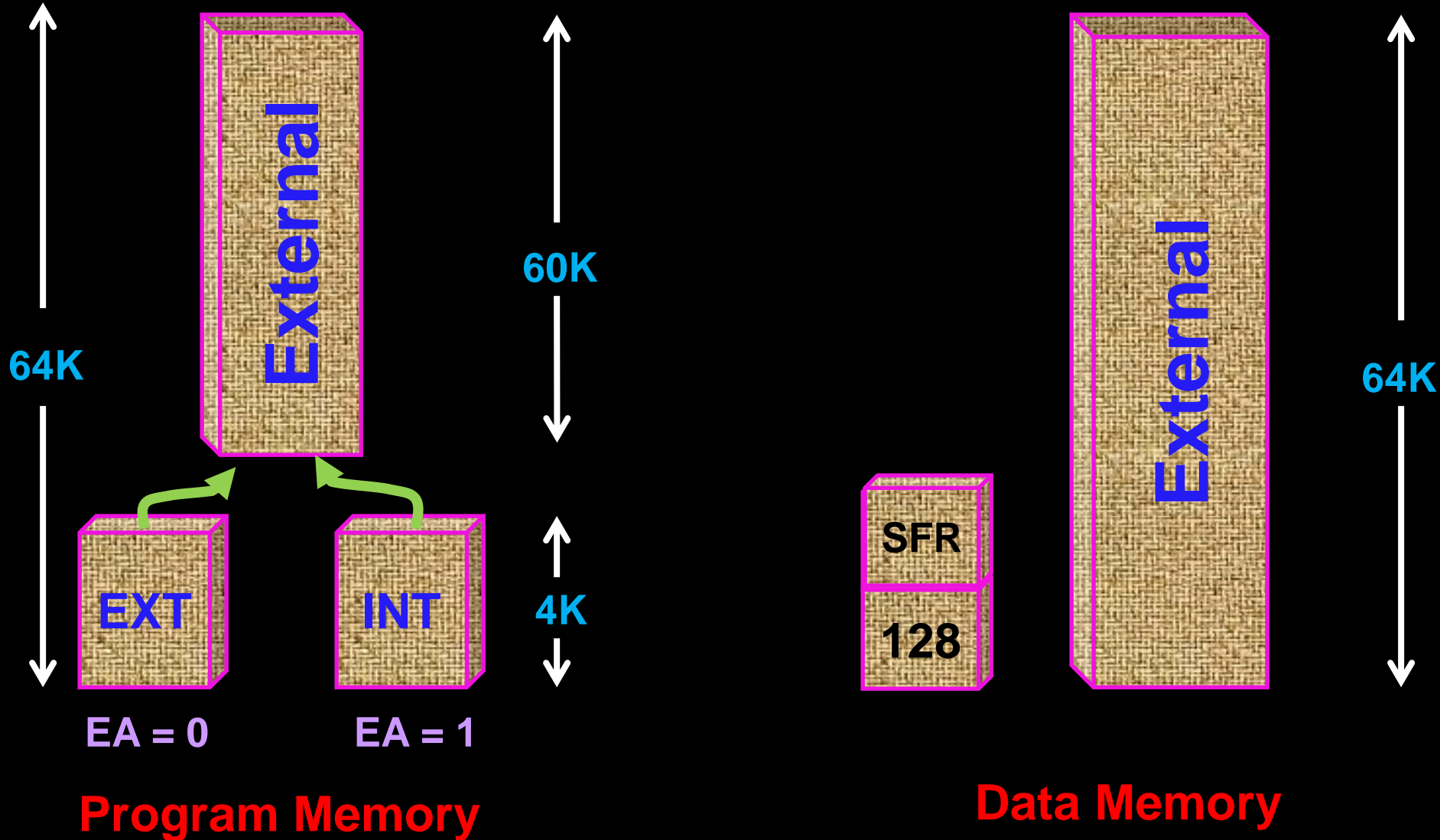
PIN	TYPE	NAME AND FUNCTION
RST	I	Reset: A high on this pin for two machine cycles while the oscillator is running, resets the device.
ALE	O	Address Latch Enable: Output pulse for latching the low byte of the address during an access to external memory.
PSEN*	O	Program Store Enable: The read strobe to external program memory. When executing code from the external program memory, PSEN* is activated twice each machine cycle, except that two PSEN* activations are skipped during each access to external data memory.
EA*/VPP	I	External Access Enable/Programming Supply Voltage: EA* must be externally held low to enable the device to fetch code from external program memory locations. If EA* is held high, the device executes from internal program memory. This pin also receives the programming supply voltage Vpp during Flash programming. (applies for 89c5x MCU's)



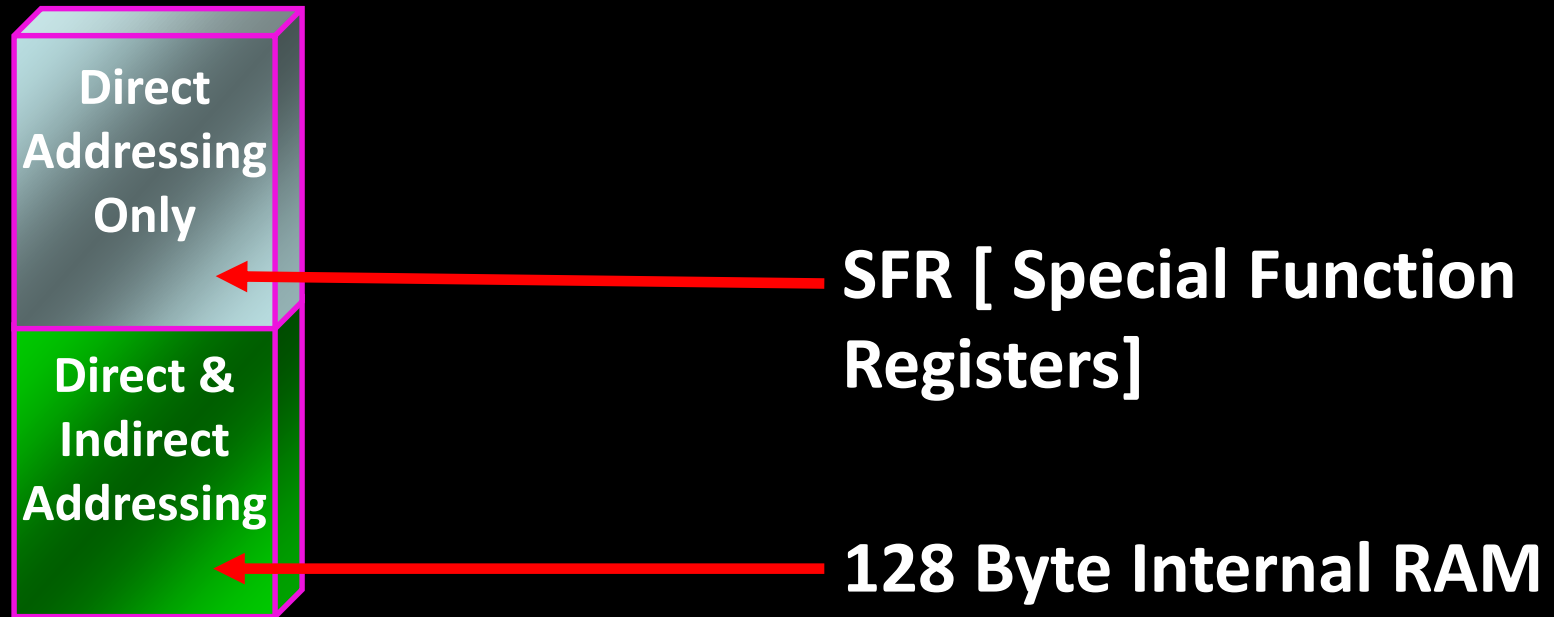
8051

Memory Space

8051 Memory Structure



Internal RAM Structure

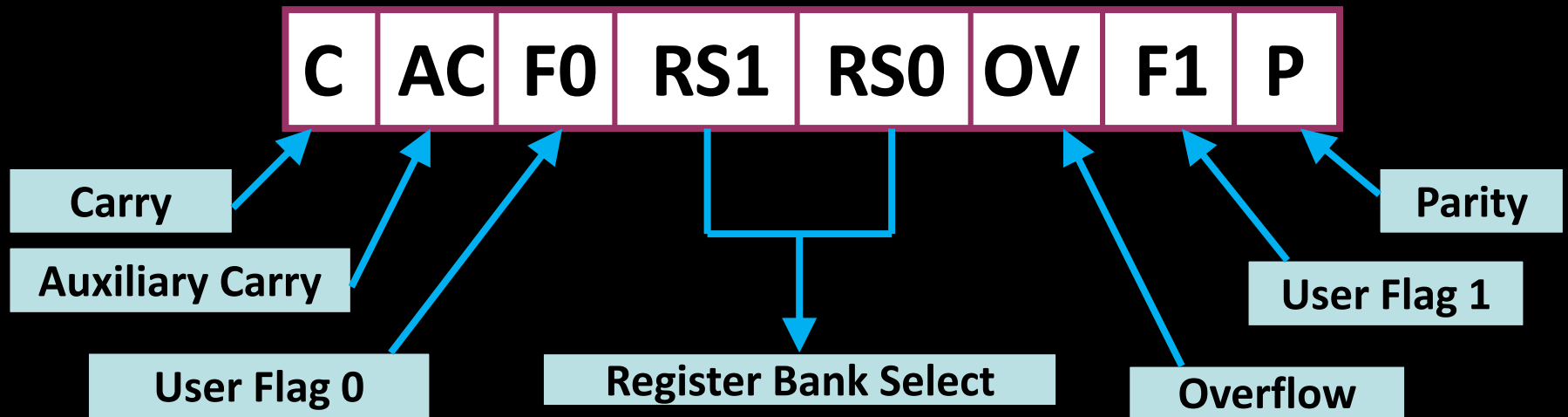


Special function register

80	PO
81	SP
82	DPL
83	DPH
87	PCON
88	TCON
89	TMOD
8A	TL0
8B	TL1
8C	TH0
8D	TH1

90	P1
98	SCON
99	SBUF
A0	P2
A8	IE
B0	P3
B8	IP
D0	PSW
E0	ACC
F0	B

Program Status Word [PSW]



8051 instructions that affects flag

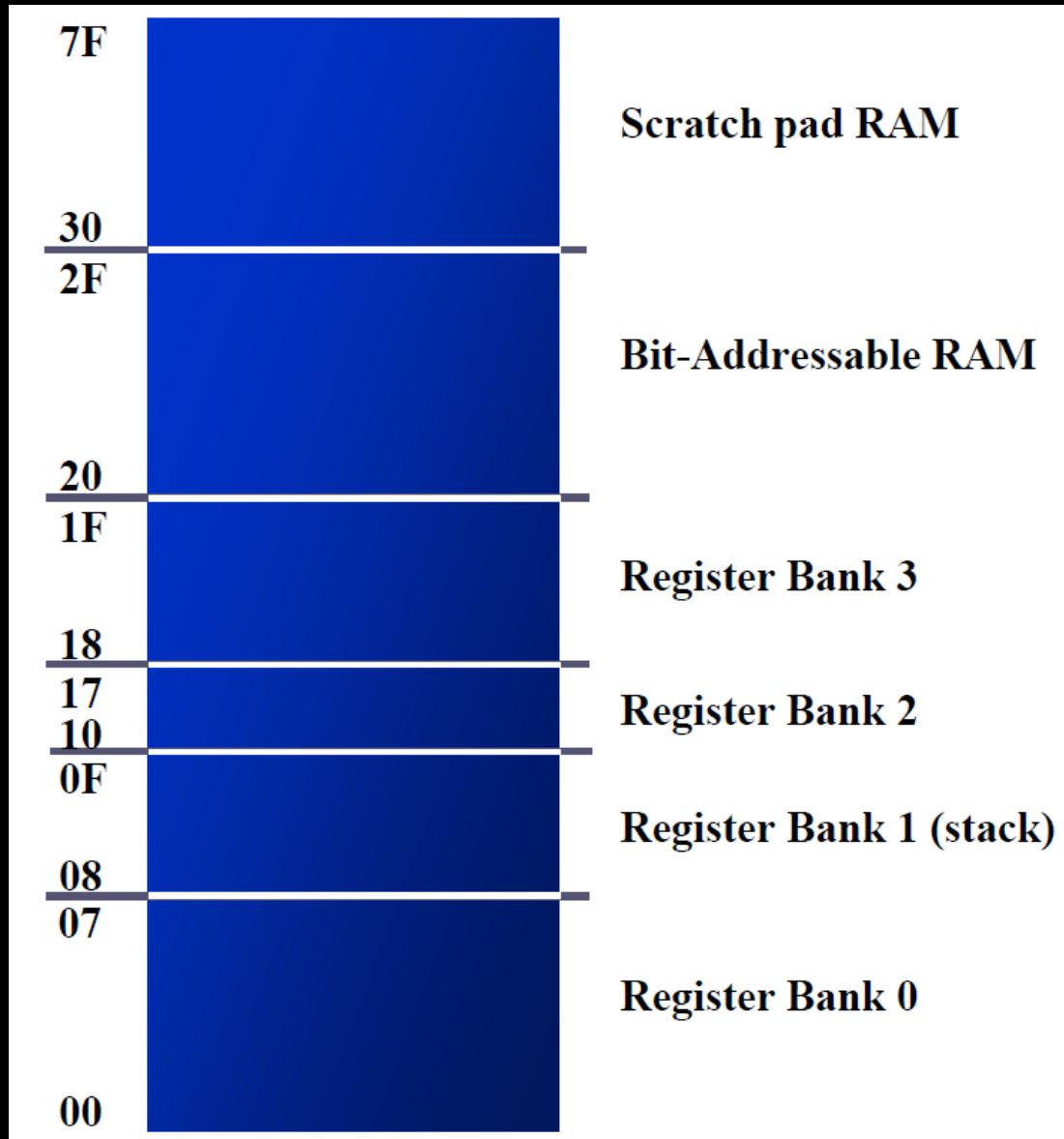
Instruction	CY	OV	AC
ADD	X	X	X
ADDC	X	X	X
SUBB	X	X	X
MUL	0	X	
DIV	0	X	
DA	X		
RPC	X		
PLC	X		
SETB C	1		
CLR C	0		
CPL C	X		
ANL C, bit	X		
ANL C, /bit	X		
ORL C, bit	X		
ORL C, /bit	X		
MOV C, bit	X		
CJNE	X		

128 Byte RAM

- There are 128 bytes of RAM in the 8051.
 - Assigned addresses 00 to 7FH
- The **128 bytes** are divided into **3 different groups** as follows:
 1. A total of **32 bytes** from locations 00 to 1FH hex are set aside for *register banks* and the *stack*.
 2. A total of **16 bytes** from locations 20H to 2FH 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*.



8051 RAM with addresses



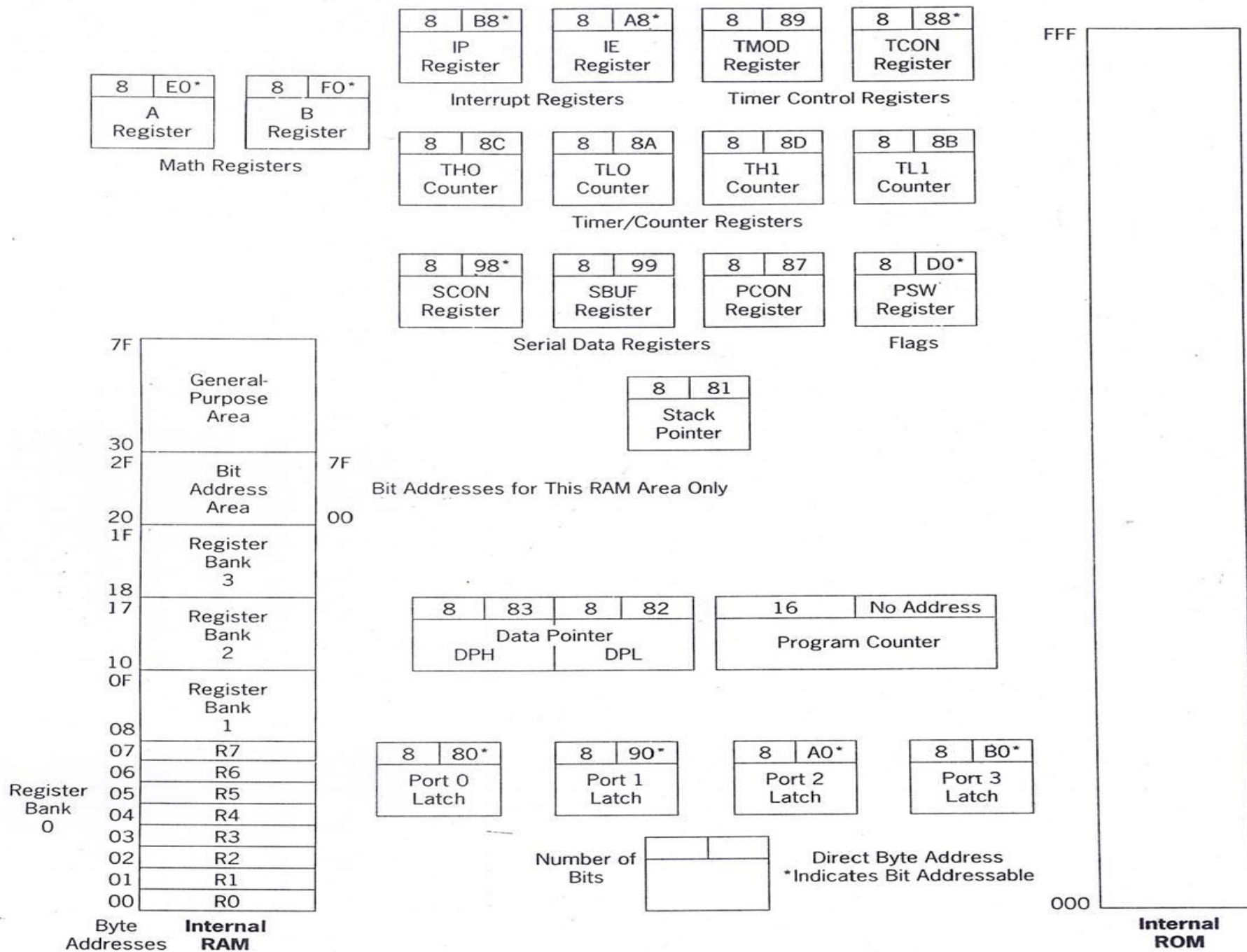
8051 Register Bank Structure



8051 Register Banks with address

Register bank 0		Register bank 1		Register bank 2		Register bank 3	
00	R0	08	R0	10	R0	18	R0
01	R1	09	R1	11	R1	19	R1
02	R2	0A	R2	12	R2	1A	R2
03	R3	0B	R3	13	R3	1B	R3
04	R4	0C	R4	14	R4	1C	R4
05	R5	0D	R5	15	R5	1D	R5
06	R6	0E	R6	16	R6	1E	R6
07	R7	0F	R7	17	R7	1F	R7

Figure 3.1b 8051 Programming Model



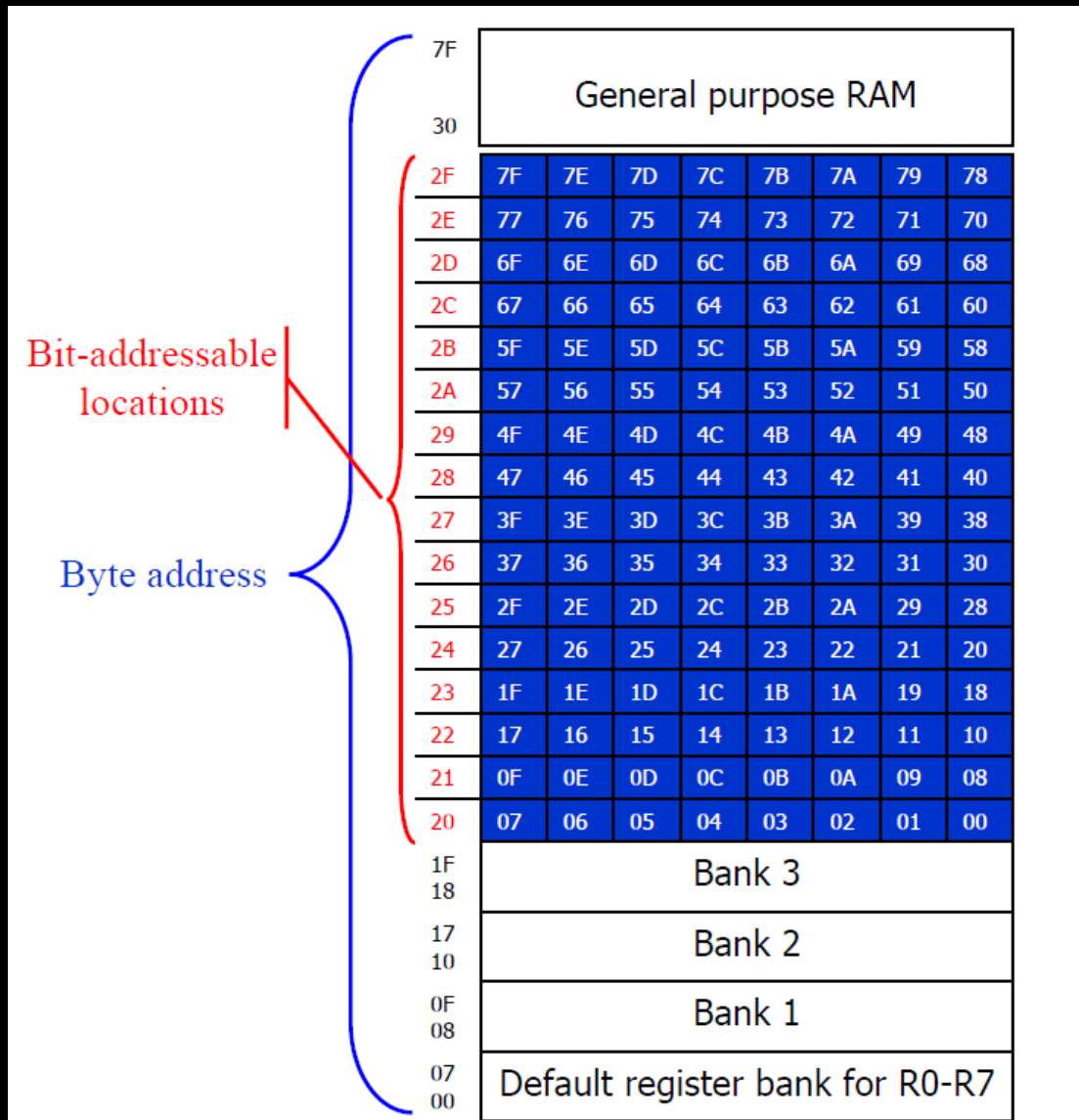
8051 Stack

- The **stack** is a section of RAM used by the CPU to store information **temporarily**.
 - This information could be data or an address
- The register used to access the stack is called the **SP (stack pointer) register**
 - The stack pointer in the 8051 is **only 8 bit wide**, which means that it can take value of 00 to FFH
 - When the 8051 is powered up, the SP register contains value 07
 - RAM location 08 is the first location begin used for the stack by the 8051

8051 Stack

- 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
 - This is **different** from many microprocessors
- 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** once

Bit Addressable & Byte Addressable



Single bit Instructions

Single-Bit Instructions

Instruction	Function
SETB bit	Set the bit (bit = 1)
CLR bit	Clear the bit (bit = 0)
CPL bit	Complement the bit (bit = NOT bit)
JB bit, target	Jump to target if bit = 1 (jump if bit)
JNB bit, target	Jump to target if bit = 0 (jump if no bit)
JBC bit, target	Jump to target if bit = 1, clear bit (jump if bit, then clear)

Bit Addressable Programming

- **Example:** Find out to which by each of the following bits belongs. Give the address of the RAM byte in hex

(a) SETB 42H, (b) CLR 67H, (c) CLR 0FH (d) SETB 28H, (e) CLR 12, (f) SETB 05

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

8051 Peripheral Overview

1. Timers

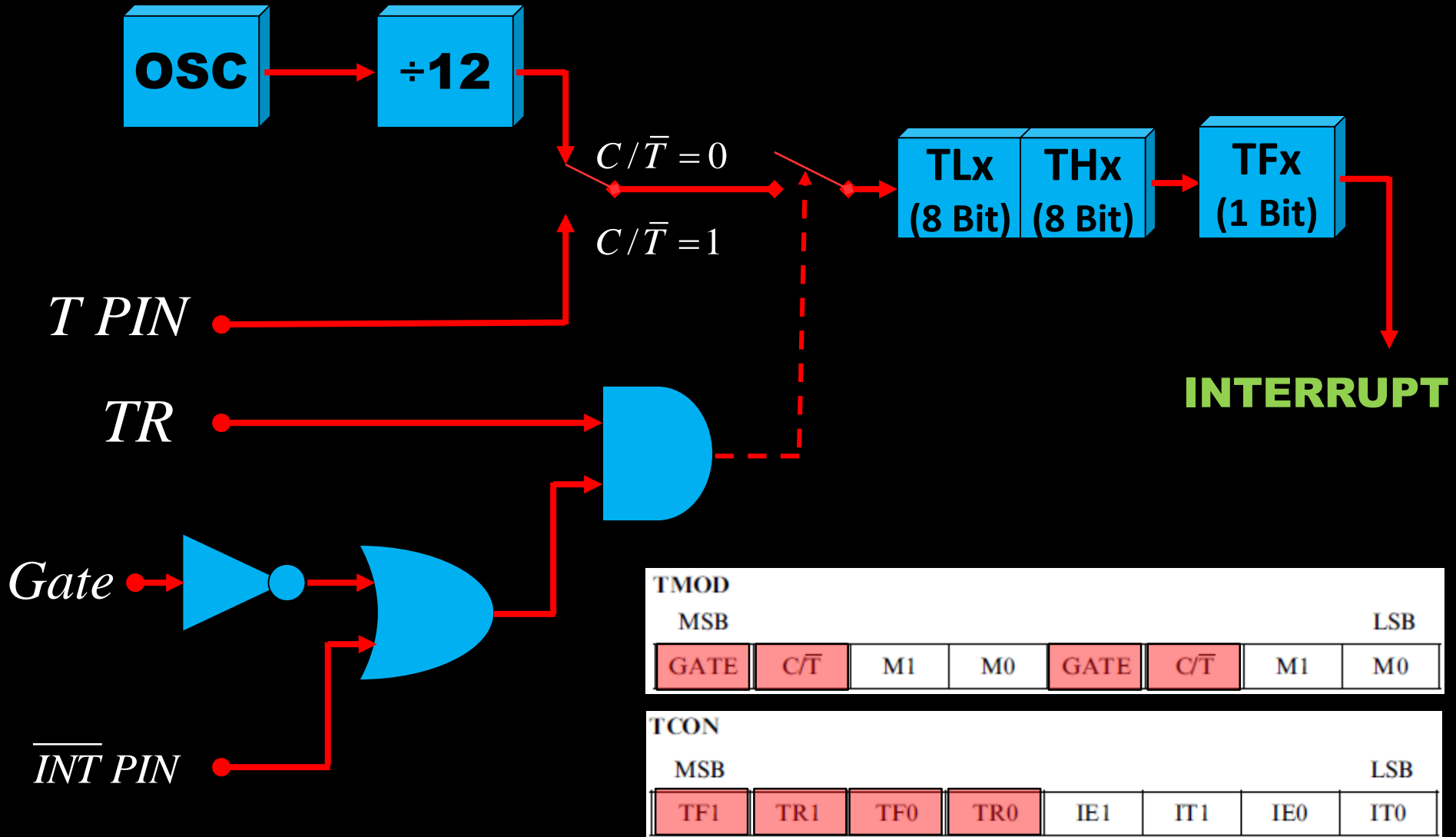
2. Serial Port

3. Interrupts

8051

TIMERS

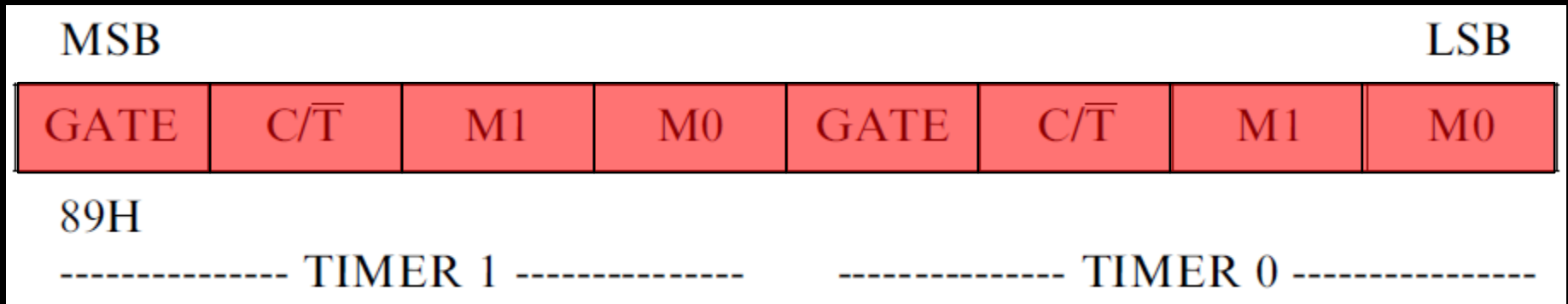
8051 Timer/Counter



TMOD							
MSB				LSB			
GATE	C/\bar{T}	M1	M0	GATE	C/\bar{T}	M1	M0

TCON							
MSB				LSB			
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

TMOD Register



GATE:

When set, timer/counter x is enabled, if INTx pin is high and TRx is set.

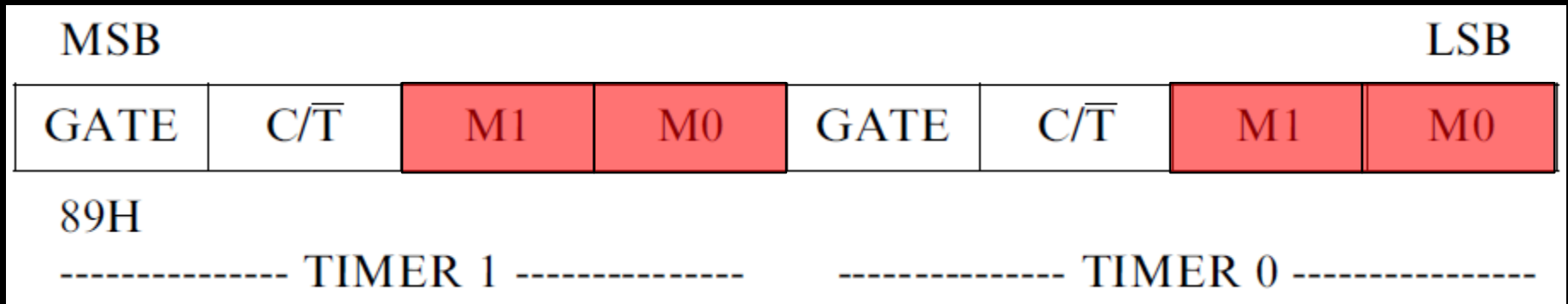
When cleared, timer/counter x is enabled, if TRx bit set.

C/T*:

When set, counter operation (input from Tx input pin).

When cleared, timer operation (input from internal clock).

TMOD Register



The TMOD byte is not bit addressable.

M1	M0	Operation
0	0	8048 8-bit timer TLx serves as 5-bit prescaler
0	1	16-bit timer/counter. THx and TLx are cascaded. No prescaler
1	0	8-bit autoreload timer/counter. THx contents loaded into TLx when it overflows
1	1	TL0 is 8-bit counter controlled by timer 0 control bits. TH0 is 8-bit timer controlled by timer 1 control bits
1	1	Timer 1 off

TCON Register

MSB				LSB			
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
8FH	8EH	8DH	8CH	8BH	8AH	89H	88H

Bit	Function
TF1/0	Timer 1/0 overflow flag. Set by hardware on timer/counter overflow. Cleared when CPU vectors to interrupt routine
TR1/0	Timer 1/0 run control bit. Set/cleared by software to turn timer/counter on/off
IE1/0	Interrupt 1/0 edge flag. Set by hardware when external interrupt edge detected. Cleared when interrupt processed
IT1/0	Interrupt 1/0 control bit. Set/cleared by software to specify falling edge/low level triggered external interrupts

8051 TIMERS

Timer 0

Mode 0

Mode 1

Mode 2

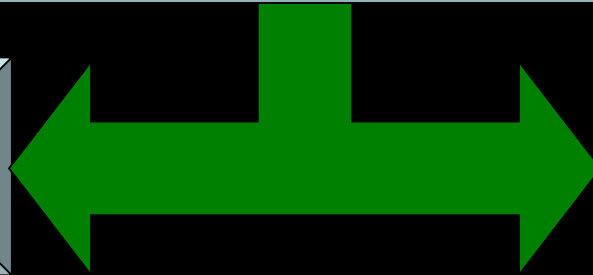
Mode 3

Timer 1

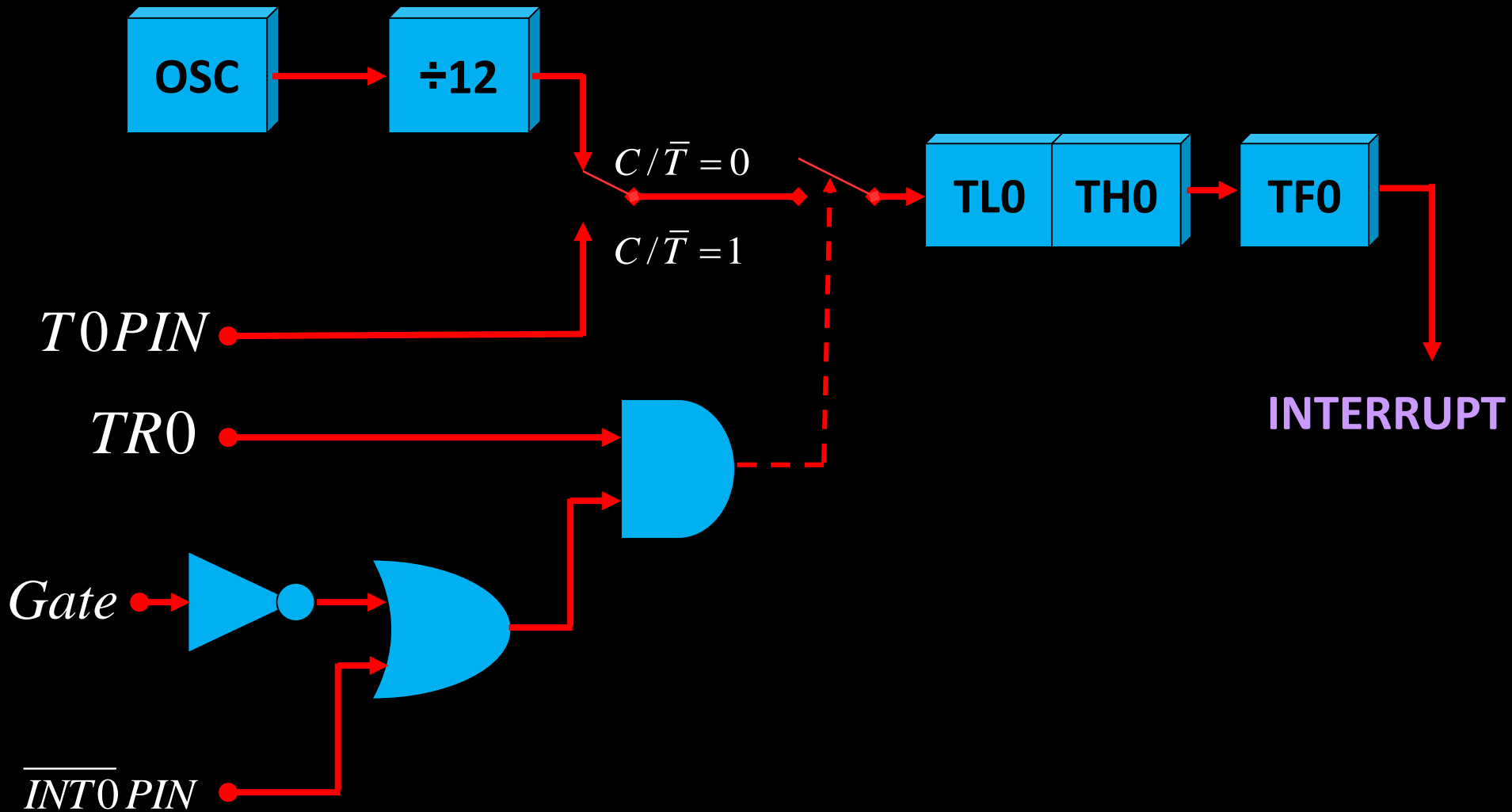
Mode 0

Mode 1

Mode 2

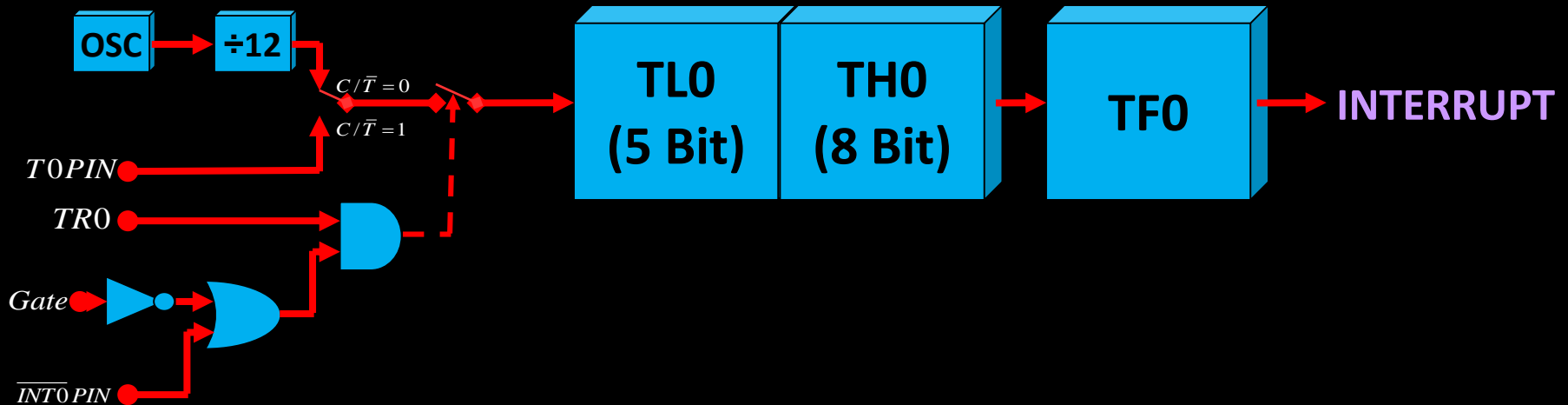


TIMER 0



TIMER 0 – Mode 0

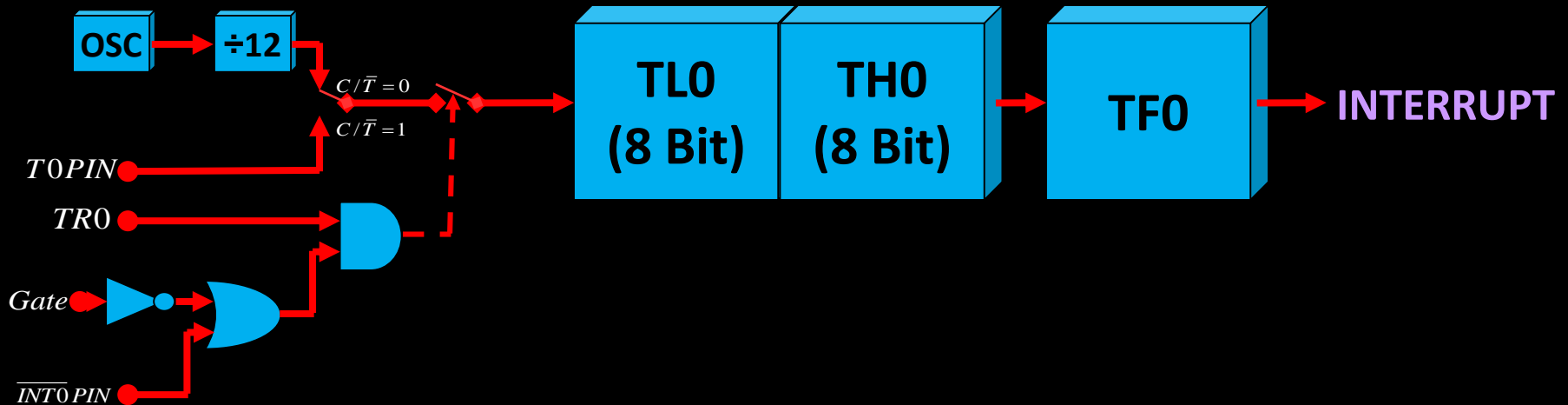
13 Bit Timer / Counter



Maximum Count = 1FFFh (0001111111111111)

TIMER 0 – Mode 1

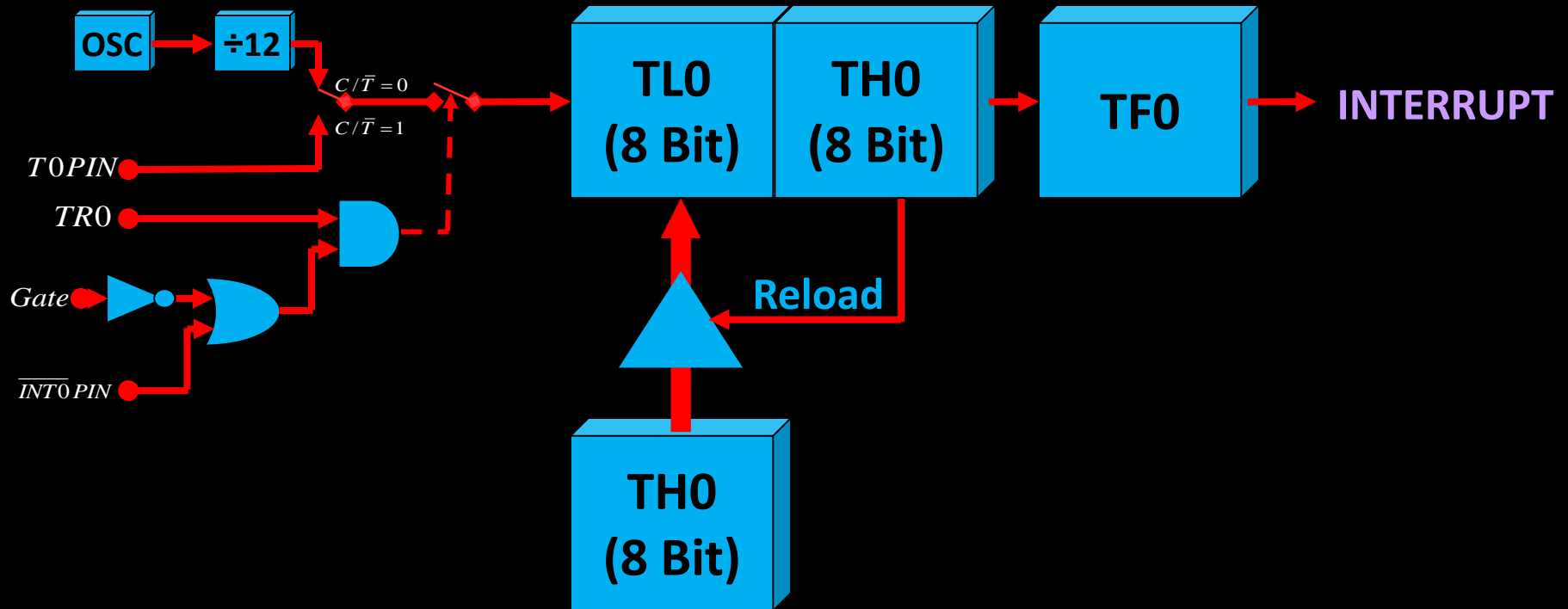
16 Bit Timer / Counter



Maximum Count = FFFFh (1111111111111111)

TIMER 0 – Mode 2

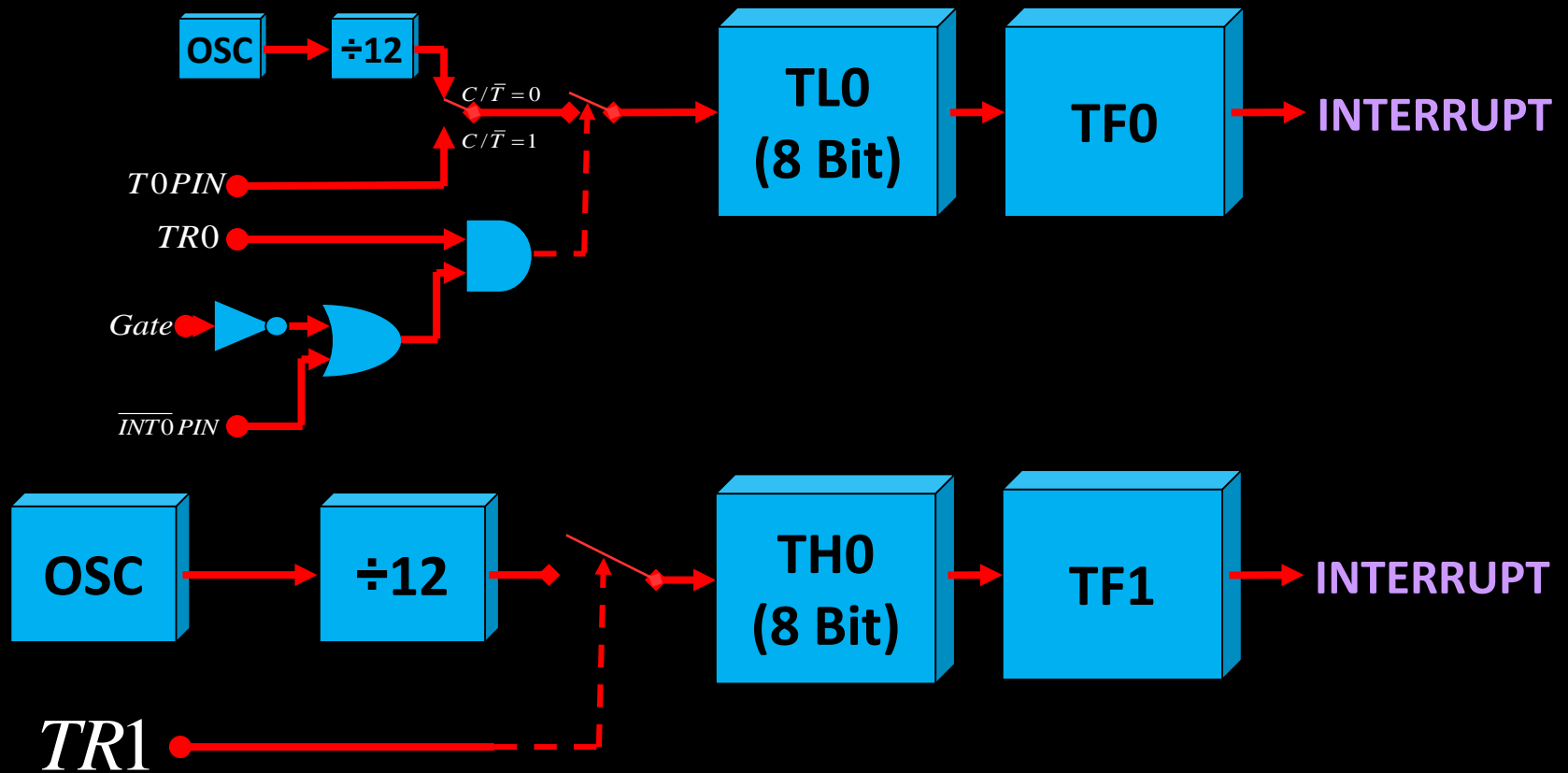
8 Bit Timer / Counter with AUTORELOAD



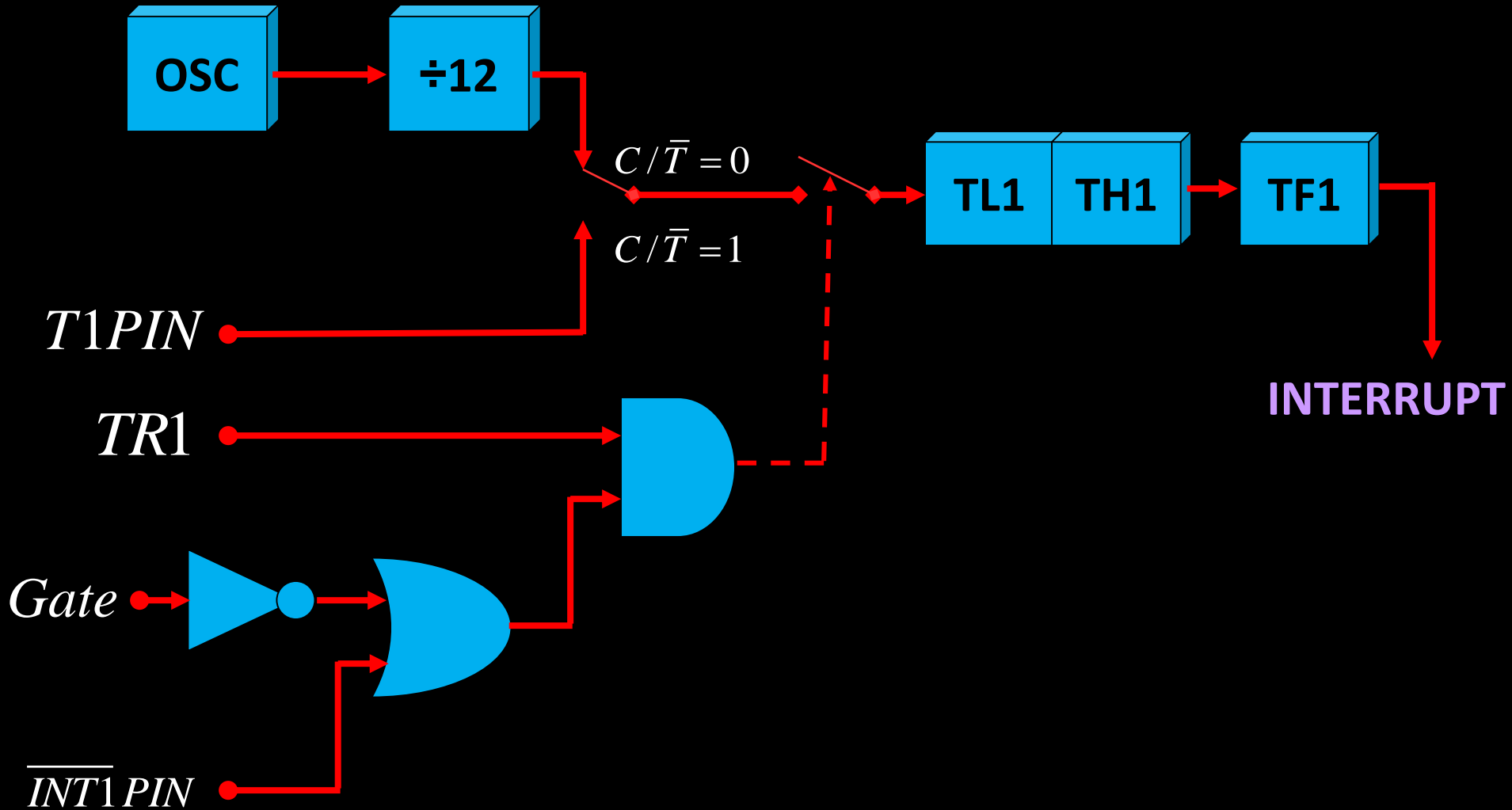
Maximum Count = FFh (11111111)

TIMER 0 – Mode 3

Two - 8 Bit Timer / Counter

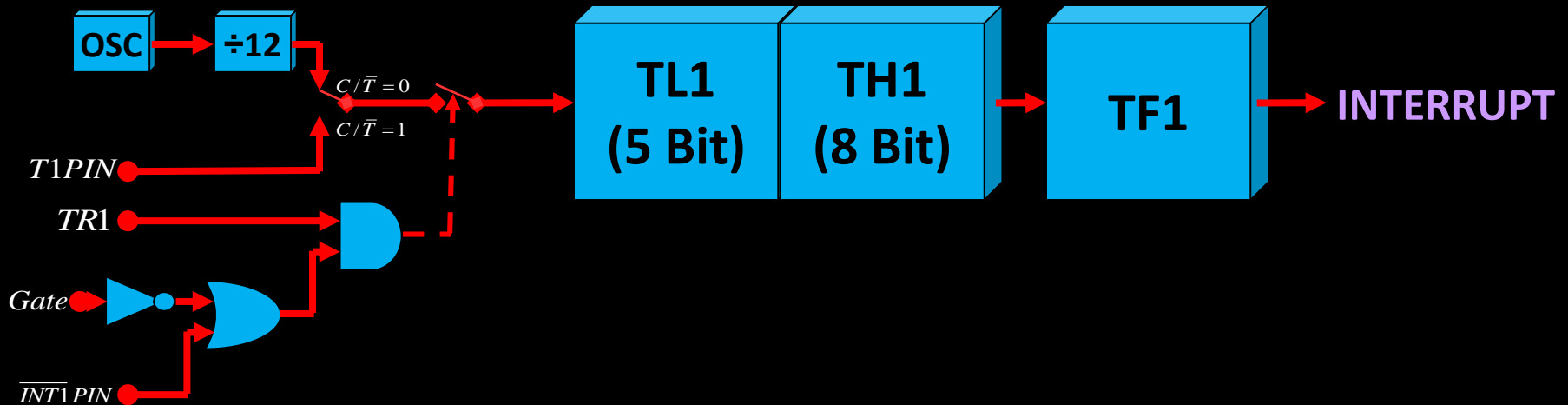


TIMER 1



TIMER 1 – Mode 0

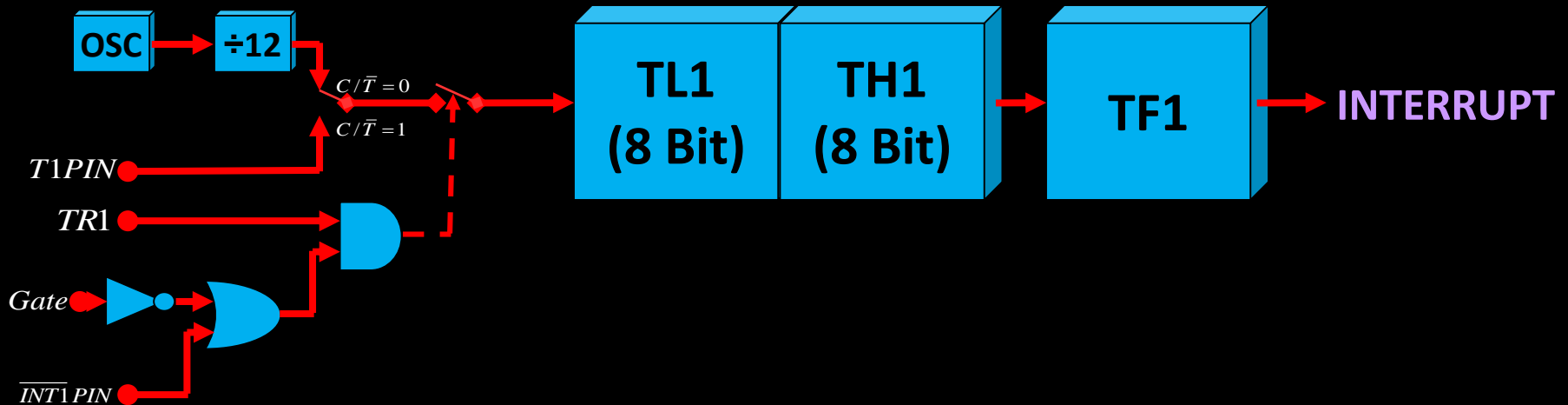
13 Bit Timer / Counter



Maximum Count = 1FFFh (0001111111111111)

TIMER 1 – Mode 1

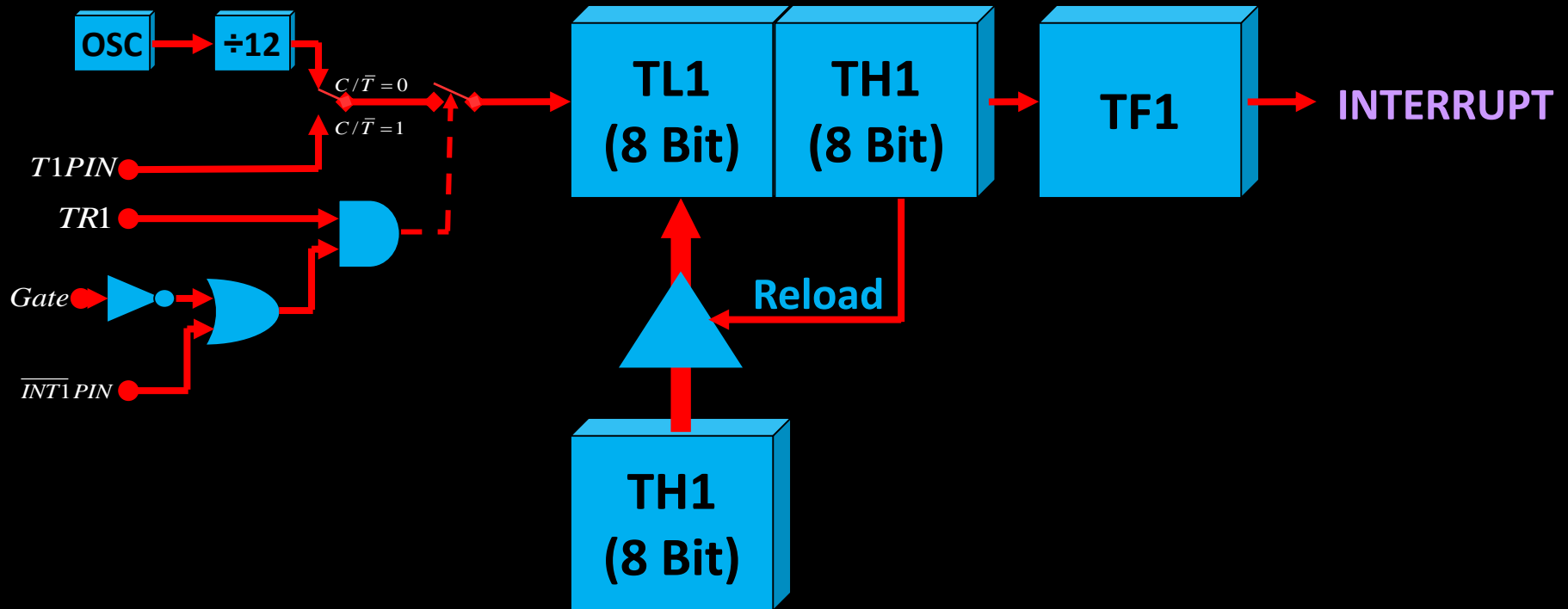
16 Bit Timer / Counter



Maximum Count = FFFFh (1111111111111111)

TIMER 1 – Mode 2

8 Bit Timer / Counter with AUTORELOAD



Maximum Count = FFh (11111111)

Programming Timers

- **Example:** Indicate which mode and which timer are selected for each of the following.

(a) MOV TMOD, #01H (b) MOV TMOD, #20H (c) MOV TMOD, #12H

- **Solution:** We convert the value from hex to binary.

(a) TMOD = 00000001, mode 1 of timer 0 is selected.

(b) TMOD = 00100000, mode 2 of timer 1 is selected.

(c) TMOD = 00010010, mode 2 of timer 0, and mode 1 of timer 1 are selected.

Programming Timers

- Find the timer's clock frequency and its period for various 8051-based system, with the crystal frequency 11.0592 MHz when C/T bit of TMOD is 0.

- **Solution:**



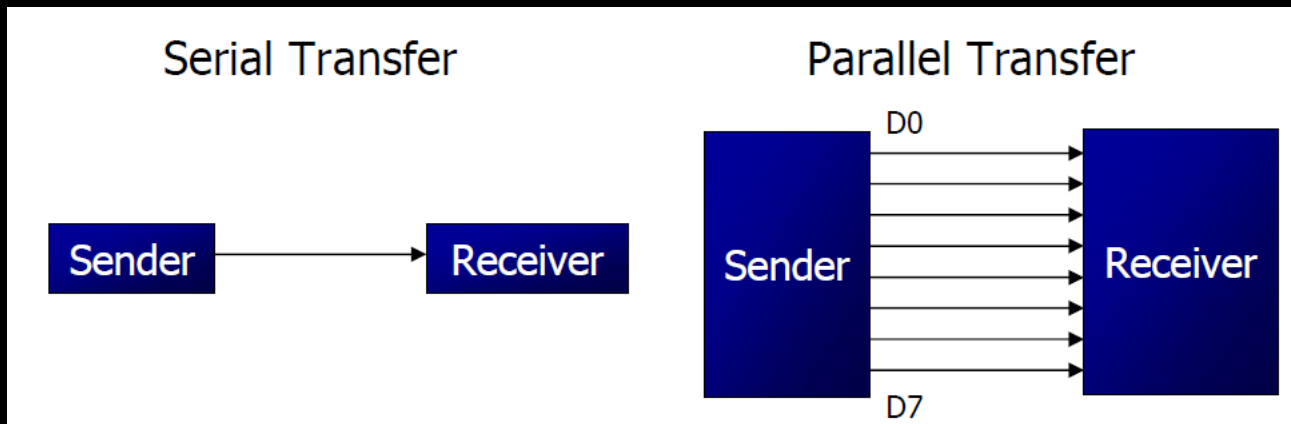
$$1/12 \times 11.0529 \text{ MHz} = 921.6 \text{ MHz};$$

$$T = 1/921.6 \text{ kHz} = 1.085 \text{ us}$$

8051 Serial Port

Basics of Serial Communication

- Computers transfer data in **two** ways:
 - **Parallel:** Often 8 or more lines (wire conductors) are used to transfer data to a device that is only a few feet away.
 - **Serial:** To transfer to a device located many meters away, the serial method is used. The data is sent one bit at a time.



Basics of Serial Communication

- Serial data communication uses **two** methods
 - **Synchronous** method transfers a block of data at a time
 - **Asynchronous** method transfers a single byte at a time

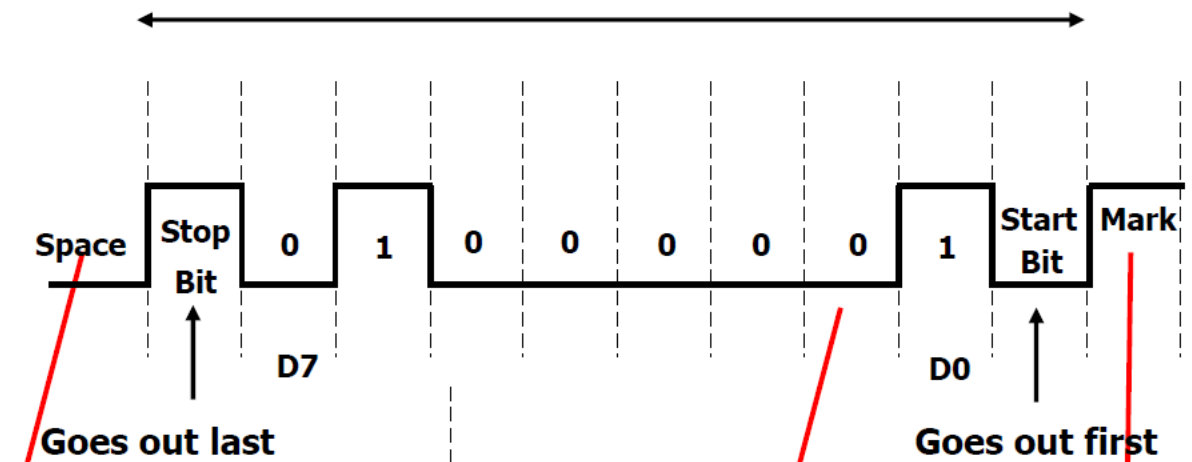
- There are **special IC's** made by many manufacturers for serial communications.
 - **UART** (universal asynchronous Receiver transmitter)
 - **USART** (universal synchronous-asynchronous Receiver-transmitter)

Asynchronous – Start & Stop Bit

- Asynchronous serial data communication is widely used for **character-oriented** transmissions
 - Each character is placed in between **start and stop bits**, this is called **framing**.
 - **Block-oriented** data transfers use the synchronous method.
- The start bit is always one bit, but the stop bit can be one or two bits
- The start bit is always a 0 (low) and the stop bit(s) is 1 (high)

Asynchronous – Start & Stop Bit

ASCII character "A" (8-bit binary 0100 0001)



The 0 (low) is referred to as *space*

The transmission begins with a start bit followed by D0, the LSB, then the rest of the bits until MSB (D7), and finally, the one stop bit indicating the end of the character

When there is no transfer, the signal is 1 (high), which is referred to as *mark*

Data Transfer Rate

- The rate of data transfer in serial data communication is stated in **bps (bits per second)**.
- Another widely used terminology for bps is **baud rate**.
 - It is modem terminology and is defined as **the number of signal changes per second**
 - In modems, there are occasions when a single change of signal transfers several bits of data
- As far as the **conductor wire** is concerned, **the baud rate and bps are the same**.

8051 Serial Port

- Synchronous and Asynchronous
- SCON Register is used to Control
- Data Transfer through TXd & RXd pins
- Some time - Clock through TXd Pin
- Four Modes of Operation:

Mode 0	:Synchronous Serial Communication
Mode 1	:8-Bit UART with Timer Data Rate
Mode 2	:9-Bit UART with Set Data Rate
Mode 3	:9-Bit UART with Timer Data Rate

Registers related to Serial Communication

1. SBUF Register

2. SCON Register

3. PCON Register

SBUF Register

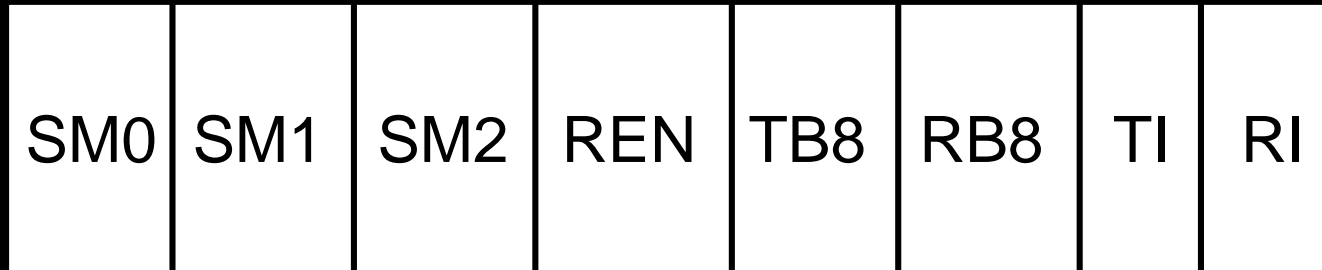
- **SBUF** is an **8-bit register** used solely for serial communication.
- For a byte data to be transferred via the **TxD line**, it must be placed in the **SBUF register**.
- The moment a byte is written into SBUF, it is framed with the start and stop bits and transferred serially via the TxD line.
- SBUF holds the byte of data when it is received by 8051 **RxD** line.
- When the bits are received serially via RxD, the **8051 deframes** it by eliminating the stop and start bits, making a byte out of the data received, and then placing it in SBUF.

SBUF Register

- **Sample Program:**

```
MOV SBUF,#'D'    ;load SBUF=44h, ASCII for 'D'  
MOV SBUF,A      ;copy accumulator into SBUF  
MOV A,SBUF      ;copy SBUF into accumulator
```

SCON Register



Serial Mode	Explanation
0	8-bit Shift Register
1	8-bit UART
2	9-bit UART
3	9-bit UART

Set to Enable Serial Data reception

Set when a Character received

Set when Stop bit Txed

Enable Multiprocessor Communication Mode

9th Data Bit Sent in Mode 2,3

9th Data Bit Received in Mode 2,3

8051 Serial Port – Mode 0

The Serial Port in Mode-0 has the following features:

1. Serial data enters and exits through RXD
2. TXD outputs the clock
3. 8 bits are transmitted / received
4. The baud rate is fixed at $(1/12)$ of the oscillator frequency

8051 Serial Port – Mode 1

The Serial Port in Mode-1 has the following features:

1. Serial data enters through RXD
2. Serial data exits through TXD
3. On receive, the stop bit goes into RB8 in SCON
4. **10 bits** are transmitted / received
 1. Start bit (0)
 2. Data bits (8)
 3. Stop Bit (1)
5. Baud rate is determined by the Timer 1 overflow rate.

8051 Serial Port – Mode 2

The Serial Port in Mode-2 has the following features:

1. Serial data **enters through RXD**
2. Serial data **exits through TXD**
3. 9th data bit (**TB8**) can be assign value 0 or 1
4. On receive, the 9th data bit goes into **RB8** in SCON
5. **11 bits** are transmitted / received
 - 1.Start bit (0)
 - 2.Data bits (9)
 - 3.Stop Bit (1)
6. **Baud rate** is programmable

8051 Serial Port – Mode 3

The Serial Port in Mode-3 has the following features:

1. Serial data **enters through RXD**
2. Serial data **exits through TXD**
3. 9th data bit (**TB8**) can be assign value 0 or 1
4. On receive, the 9th data bit goes into **RB8** in SCON
5. **11 bits** are transmitted / received
 - 1.Start bit (0)
 - 2.Data bits (9)
 - 3.Stop Bit (1)
6. **Baud rate** is determined by Timer 1 overflow rate.

Programming Serial Data Transmission

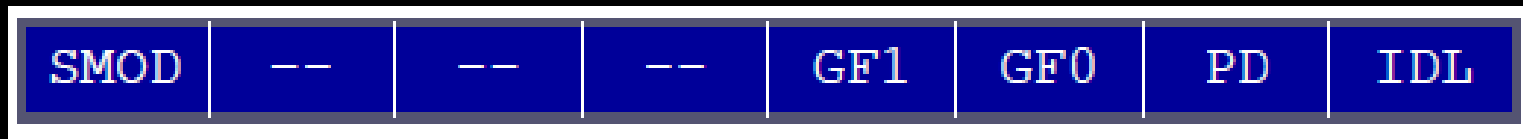
1. **TMOD register** is loaded with the value **20H**, indicating the use of timer 1 in mode 2 (8-bit auto-reload) **to set baud rate**.
2. The **TH1** is loaded with one of the values to set baud rate for serial data transfer.
3. The **SCON register** is loaded with the value **50H**, indicating serial mode 1, where an 8-bit data is framed with start and stop bits.
4. **TR1** is set to 1 to start timer 1
5. **TI** is cleared by **CLR TI** instruction
6. The character byte to be transferred serially is written into **SBUF register**.
7. The **TI flag bit** is monitored with the use of instruction **JNB TI, xx** to see if the character has been transferred completely.
8. To transfer the next byte, **go to step 5**

Programming Serial Data Reception

1. **TMOD register** is loaded with the value **20H**, indicating the use of timer 1 in mode 2 (8-bit auto-reload) **to set baud rate**.
2. **TH1** is loaded to set baud rate
3. The **SCON register** is loaded with the value **50H**, indicating serial mode 1, where an 8-bit data is framed with start and stop bits.
4. **TR1** is set to 1 to start timer 1
5. **RI** is cleared by **CLR RI** instruction
6. The **RI flag bit** is monitored with the use of instruction **JNB RI, xx** to see if an entire character has been received yet
7. **When RI is raised, SBUF** has the byte, its contents are moved into a safe place.
8. To receive the next character, **go to step 5**.

Doubling Baud Rate

- There are two ways to increase the baud rate of data transfer
 1. By using a higher frequency crystal
 2. By changing a bit in the PCON register
- PCON register is an 8-bit register.

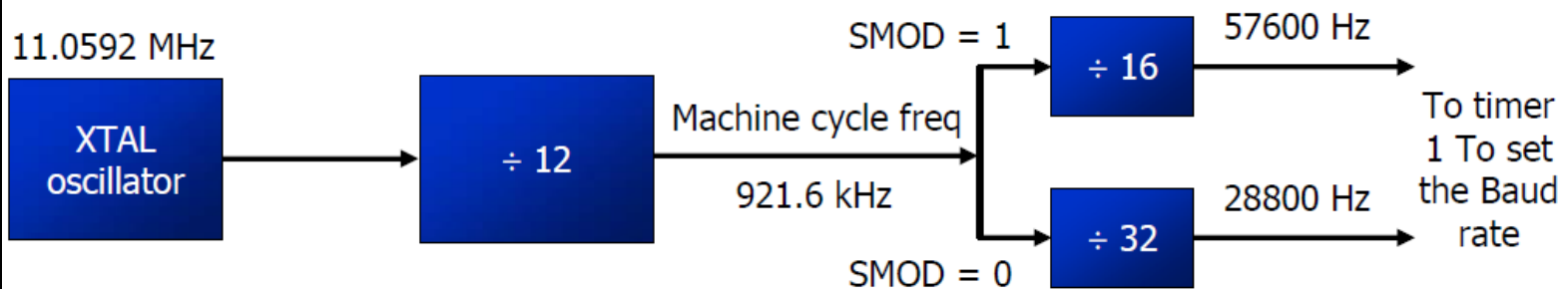


- When 8051 is powered up, **SMOD** is zero
- We can set it to high** by software and thereby **double** the baud rate.

Doubling Baud Rate (cont...)

```

MOV  A,PCON      ;place a copy of PCON in ACC
SETB ACC.7      ;make D7=1
MOV  PCON,A      ;changing any other bits
    
```



Baud Rate comparison for SMOD=0 and SMOD=1

TH1	(Decimal)	(Hex)	SMOD=0	SMOD=1
-3		FD	9600	19200
-6		FA	4800	9600
-12		F4	2400	4800
-24		E8	1200	2400

8051

Interrupts

INTERRUPTS

- An interrupt is an external or internal event that interrupts the microcontroller to inform it that a device needs its service
- A single microcontroller can serve several devices by two ways:
 1. Interrupt
 2. Polling

Interrupt Vs Polling

1. Interrupts

- 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 which is associated with the interrupt is called the **interrupt service routine (ISR)** or interrupt handler.

2. Polling

- The microcontroller **continuously monitors** the status of a given device.
- When the **conditions** met, it performs the service.
- After that, it moves on to monitor the **next device** until every one is serviced.

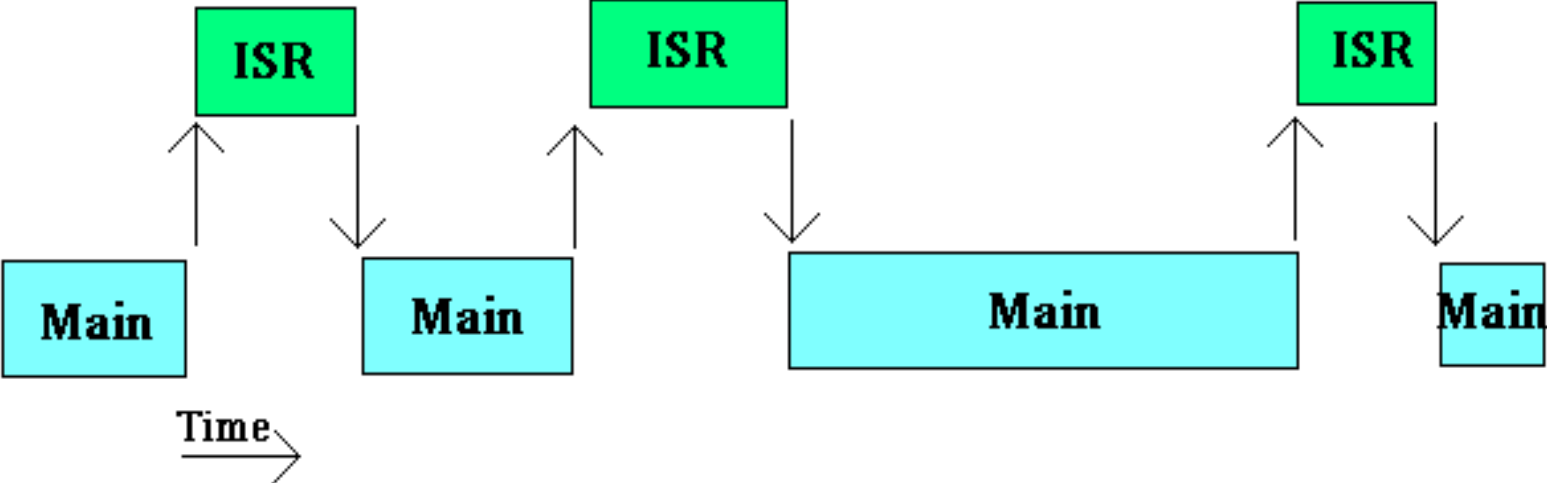
Interrupt Vs Polling

- The **polling method is not efficient**, since it wastes much of the microcontroller's time by polling devices that do not need service.
- The **advantage of interrupts** is that the microcontroller can serve many devices (not all at the same time).
- Each devices can get the attention of the microcontroller based on the **assigned priority**.
- For the polling method, it is **not possible** to assign priority since it checks all devices in a round-robin fashion.
- The microcontroller can also **ignore (mask)** a device request for service in Interrupt.

Program execution without intrrupts :



Program execution with intrrupts :



ISR : Intrrupt Service Routin

Six Interrupts in 8051

Six interrupts are allocated as follows:

1. Reset – power-up reset.

2. Two interrupts are set aside for the timers.

- one for timer 0 and one for timer 1

3. Two interrupts are set aside for hardware external interrupts.

- P3.2 and P3.3 are for the external hardware interrupts INT0 (or EX1), and INT1 (or EX2)

4. Serial communication has a single interrupt that belongs to both receive and transfer.

What events can trigger Interrupts?

- We can configure the 8051 so that any of the following events will cause an interrupt:
 - Timer 0 Overflow.
 - Timer 1 Overflow.
 - Reception/Transmission of Serial Character.
 - External Event 0.
 - External Event 1.
- We can configure the 8051 so that when Timer 0 Overflows or when a character is sent/received, the appropriate interrupt handler routines are called.

8051 Interrupt Vectors

INTERRUPT VECTORS

When the original 8051 and 8031 were introduced, only 5 interrupts were provided.

Interrupt Number	Interrupt Vector Address	Description
0	0003h	EXTERNAL 0
1	000Bh	TIMER/COUNTER 0
2	0013h	EXTERNAL 1
3	001Bh	TIMER/COUNTER 1
4	0023h	SERIAL PORT

8051 Interrupt related Registers

- The various registers associated with the use of interrupts are:
 - TCON - Edge and Type bits for External Interrupts 0/1
 - SCON - RI and TI interrupt flags for RS232
 - IE - Enable interrupt sources
 - IP - Specify priority of interrupts

Enabling and Disabling an Interrupt

- Upon **reset**, all interrupts are **disabled (masked)**, meaning that none will be responded to by the microcontroller if they are activated.
- The interrupts must be **enabled** by software in order for the microcontroller to respond to them.
- There is a register called **IE (interrupt enable)** that is responsible for enabling (unmasking) and disabling (masking) the interrupts.

Interrupt Enable (IE) Register



- **EA** : Global enable/disable.
- **---** : Reserved for additional interrupt hardware.
- **ES** : Enable Serial port interrupt.
- **ET1** : Enable Timer 1 control bit.
- **EX1** : Enable External 1 interrupt.
- **ET0** : Enable Timer 0 control bit.
- **EX0** : Enable External 0 interrupt.

```
MOV IE,#08h  
or  
SETB ET1
```

Enabling and Disabling an Interrupt

- **Example:** Show the instructions to (a) enable the serial interrupt, timer 0 interrupt, and external hardware interrupt 1 and (b) disable (mask) the timer 0 interrupt, then (c) show how to disable all the interrupts with a single instruction.
- **Solution:**
 - (a) **MOV IE,#10010110B** ;enable serial, timer 0, EX1
 - Another way to perform the same manipulation is:
 - **SETB IE.7** ;EA=1, global enable
 - **SETB IE.4** ;enable serial interrupt
 - **SETB IE.1** ;enable Timer 0 interrupt
 - **SETB IE.2** ;enable EX1
 - (b) **CLR IE.1** ;mask (disable) timer 0 interrupt only
 - (c) **CLR IE.7** ;disable all interrupts

Interrupt Priority

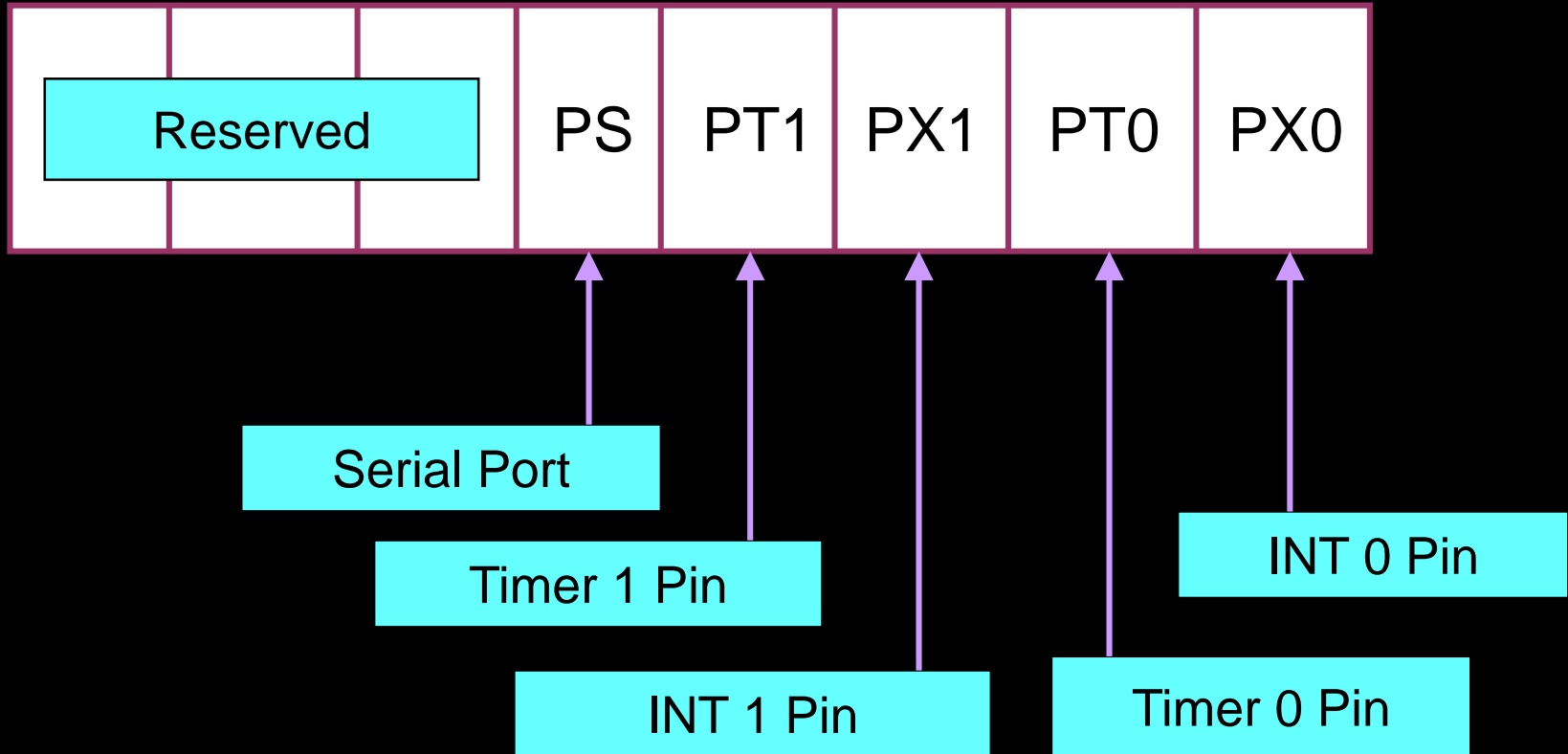
- When the 8051 is powered up, the priorities are assigned according to the following.
- In reality, the priority scheme is nothing but an internal polling sequence in which the 8051 polls the interrupts in the sequence listed and responds accordingly.

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

Interrupt Priority

- We can alter the sequence of interrupt priority by assigning a higher priority to any one of the interrupts by programming a register called **IP (interrupt priority)**.
- To give a higher priority to any of the interrupts, we make the **corresponding bit in the IP register high**.

Interrupt Priority (IP) Register



Priority bit=1 assigns high priority

Priority bit=0 assigns low priority