

# **Professional Elective-II: Embedded Systems (PECCSE601B)**

## **Module - II**

The typical embedded system

Prepared by – Dr. Susmita Biswas

Associate Professor

University of Engineering and Management, Kolkata

# **processors**

## Processor

General Purpose Processor (GPP)

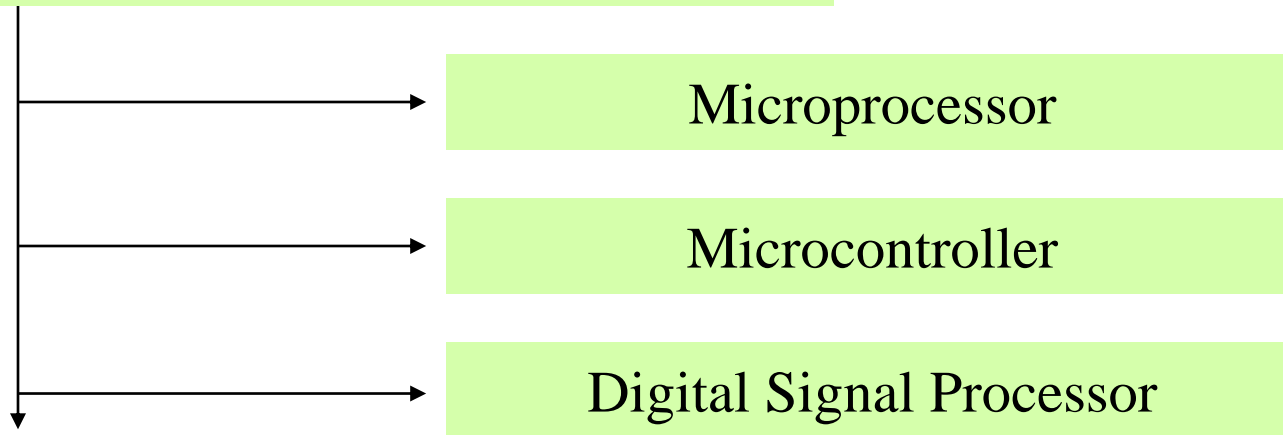
Application Specific System Processor (ASSP)

Application Specific Instruction Processor (ASIP)

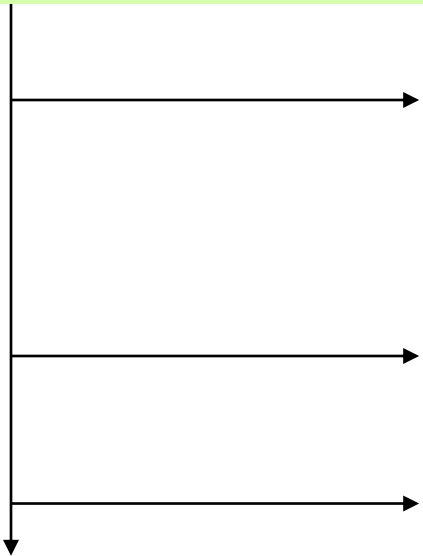
Programmable Logic Device (PLD)

Commercial Off-The-Shelf (COTS) components

## General Purpose Processor (GPP)



## Microprocessor



8-bit (Internal bus is 8-bit, register is 8-bit, ALU also designed for 8 bit operation): Intel 8085 microprocessor

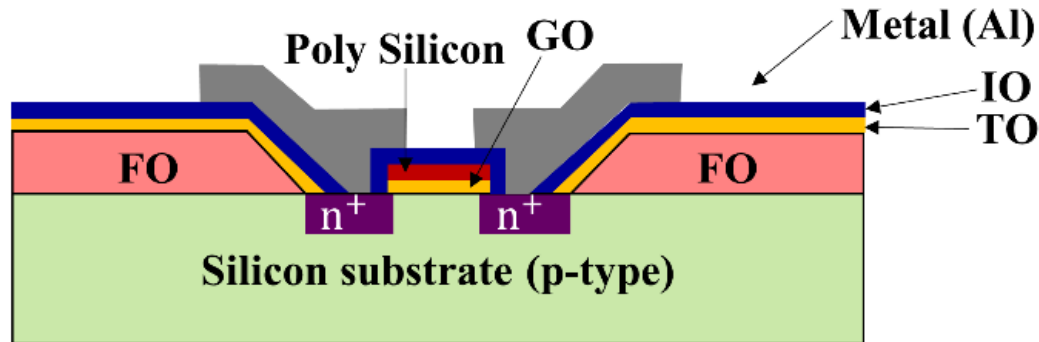
16-bit: Intel 8086, 8088 microprocessor

32-bit: Intel 80x86, ARM processor

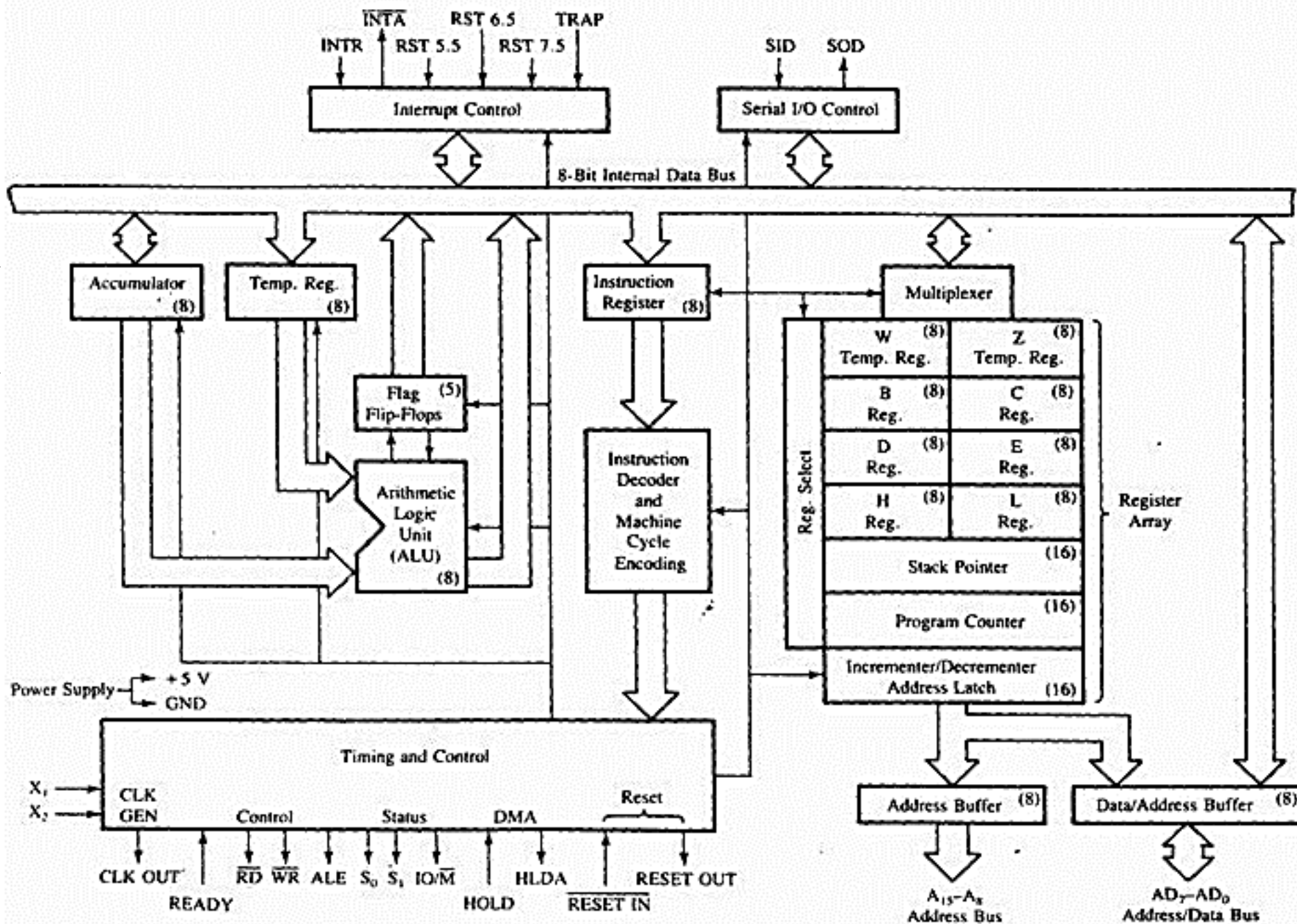
Microcontroller: Intel 8051, ARM (Advanced RISC Machine), PIC (Peripheral Interface Controller)

# Intel 8085 microprocessor architecture

- The Intel 8085 microprocessor is an **8-bit microprocessor** designed by Intel in 1977 using **NMOS (n-channel metal-oxide semiconductor) technology**.



- Its architecture consists of several key components, including the accumulator, registers, program counter, stack pointer, instruction register, flags register, data bus, address bus, and control bus.
- The 8085 is based on **Von-Neumann architecture**, where the data and instructions are in the same memory space without any distinction between them.
- The Intel 8085 comes as a 40-pin IC package.



# Intel 8085 microprocessor architecture

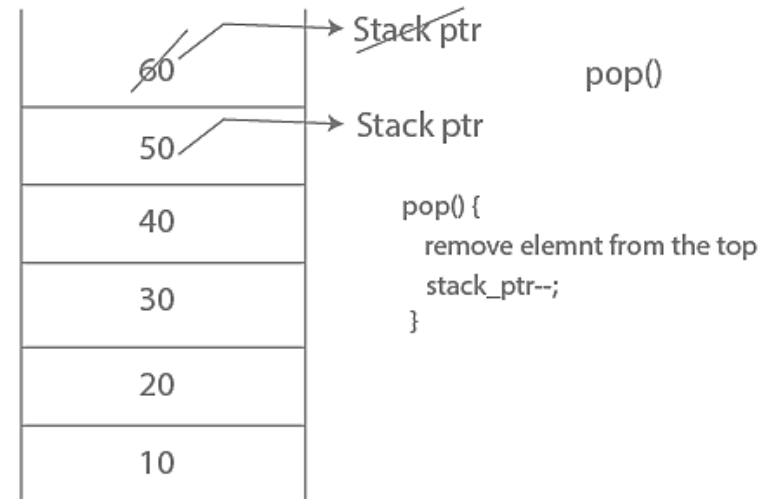
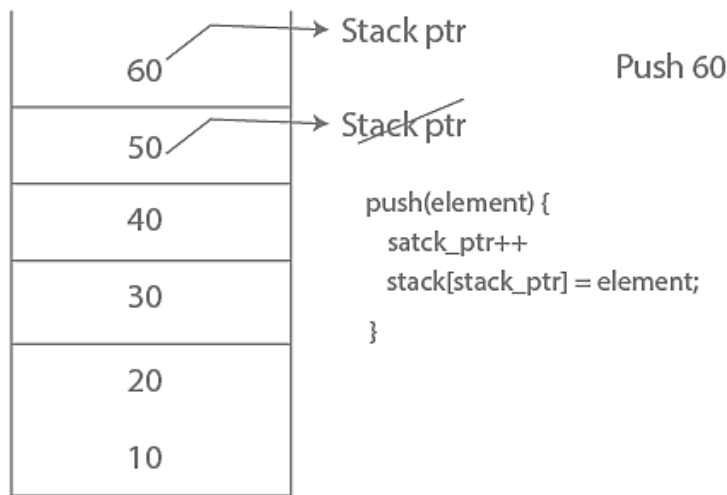
1. **Arithmetic and Logic Unit (ALU):** Arithmetic Operations, Logical operations, Bit-Shifting Operations.
2. **Accumulator:** Accumulator is used to perform I/O, arithmetic, and logical operations. It is connected to ALU and the internal data bus. For all arithmetic operations Accumulator's 8-bit pin will always be connected with ALU and in most-of times results of all the operations, carried by different instructions, will be stored in the accumulator after operation performance.
3. **General Purpose Registers:** There are six general-purpose registers. These registers can hold 8-bit values. These 8-bit registers are B,C,D,E,H,L. These registers work as 16-bit registers when they work in pairs like B-C, D-E, and H-L.

Here registers **W and Z are reserved registers**. We can't use these registers in arithmetic operations. It is reserved for microprocessors for internal operations like swapping two 16-bit numbers. We know that to swap two numbers we need a third variable hence here W-Z register pair works as **temporary registers** and we can swap two 16-bit numbers using this pair.



# Intel 8085 microprocessor architecture

- Program Counter:** Program Counter holds the address value of the memory to the next instruction that is to be executed. It is a 16-bit register. Program Counter(PC) works in random memory locations.
- Stack Pointer :** It is a 16-bit special register. A stack pointer stores the memory address of the last data element added to the stack or, in some cases, the first available address in the stack. Stack pointer works in a continuous and contiguous part of the memory. In stack, the content of the register is stored that is later used in the program. This pointer is very useful in stack-related operations like PUSH, POP, and nested CALL requests initiated by Microprocessor.



# Intel 8085 microprocessor architecture

6. **Flag Register:** Flag Register contains 8-bit out of which 5-bits are important. Those 5 flip-flops are set/reset after an operation according to data condition of the result in accumulator or other register.
- **Carry Flag (CY):** Set ( $q=1$ ) when arithmetic operation results in a carry otherwise reset ( $q=0$ ).
  - **Parity Flag (P):** Set ( $q=1$ ) when result has an even number of 1, otherwise reset ( $q=0$ ).
  - **Auxiliary Carry Flag (AC):** Set ( $q=1$ ) when in an arithmetic operation carry is generated by digit D3 and propagated to digit D4 otherwise reset ( $q=0$ ).
  - **Zero Flag (Z):** Set ( $q=1$ ) when result of an arithmetic operation is zero otherwise reset ( $q=0$ ).
  - **Sign Flag (S):** Set ( $q=1$ ) when sign bit (MSB) of the result of an arithmetic operation is 1 (negative result) otherwise reset ( $q=0$ ).

# Intel 8085 microprocessor architecture

- 7. Instruction register and decoder:** It is an 8-bit register that holds the instruction that is just fetched from the memory and being decoded and executed by the processor.
- 8. Timing and control unit:** The timing and control unit comes under the CPU section, and it controls the flow of data from the CPU to other devices. There are certain timing and control signals like Control signals, DMA Signals, RESET signals and Status signals.
  - **ALE** – It is an **Address Latch Enable** signal. It goes high (1) during first T state of a machine cycle and enables the lower 8-bits of the address, otherwise data bus is activated.
  - **IO/ $\bar{M}$**  – It is a status signal which determines whether the address is for input-output or memory. When it is high(1) the address on the address bus is for input-output devices. When it is low(0) the address on the address bus is for the memory.
  - **SO, S1** – These are status signals. They distinguish the various types of operations such as halt, reading, instruction fetching or writing.

- **$\overline{\text{RD}}$**  – It is a signal to control READ operation. When it is low the selected memory or input-output device is read.
- **$\overline{\text{WR}}$**  – It is a signal to control WRITE operation. When it goes low the data on the data bus is written into the selected memory or I/O location.
- **READY** – It senses whether a peripheral is ready to transfer data or not. If READY is high(1) the peripheral is ready. If it is low(0) the microprocessor waits till it goes high. It is useful for interfacing low speed devices.
- **$\overline{\text{RESET IN}}$**  – When the signal on this pin is low(0), the program-counter is set to zero.
- **RESET OUT** – This signal indicates that the MPU is being reset. The signal can be used to reset other devices.
- **HOLD** – It indicates that another device is requesting the use of the address and data bus. Having received HOLD request the microprocessor relinquishes the use of the buses as soon as the current machine cycle is completed. Internal processing may continue. After the removal of the HOLD signal the processor regains the bus.
- **HLDA** – It is a signal which indicates that the hold request has been received after the removal of a HOLD request, the HLDA goes low.

# Intel 8085 microprocessor architecture

## 9. System bus:

- The **data bus** is **8-bit bidirectional** and carries the data which is to be stored. The 8 data lines are manipulating 8-bit data ranging from 00 to FF i.e. ( $2^8 = 256$ ) numbers from 0000 0000 -1111 1111.
- The **address bus** is **16-bit unidirectional** and carries the location where information (instruction/ data) is to be stored. MPU carries 16-bit address i.e.  $2^{16} = 65,536$  or 64K memory locations.
- The **control bus** is having various **single lines** used for sending control signals in the form of the pulse to the memory and I/O devices.

**10. Interrupt control:** Whenever a microprocessor is executing the main program and if suddenly an interrupt occurs, the microprocessor shifts the control from the main program to process the incoming request. After the request is completed, the control goes back to the main program.

There are 5 interrupt signals in 8085 microprocessors. :

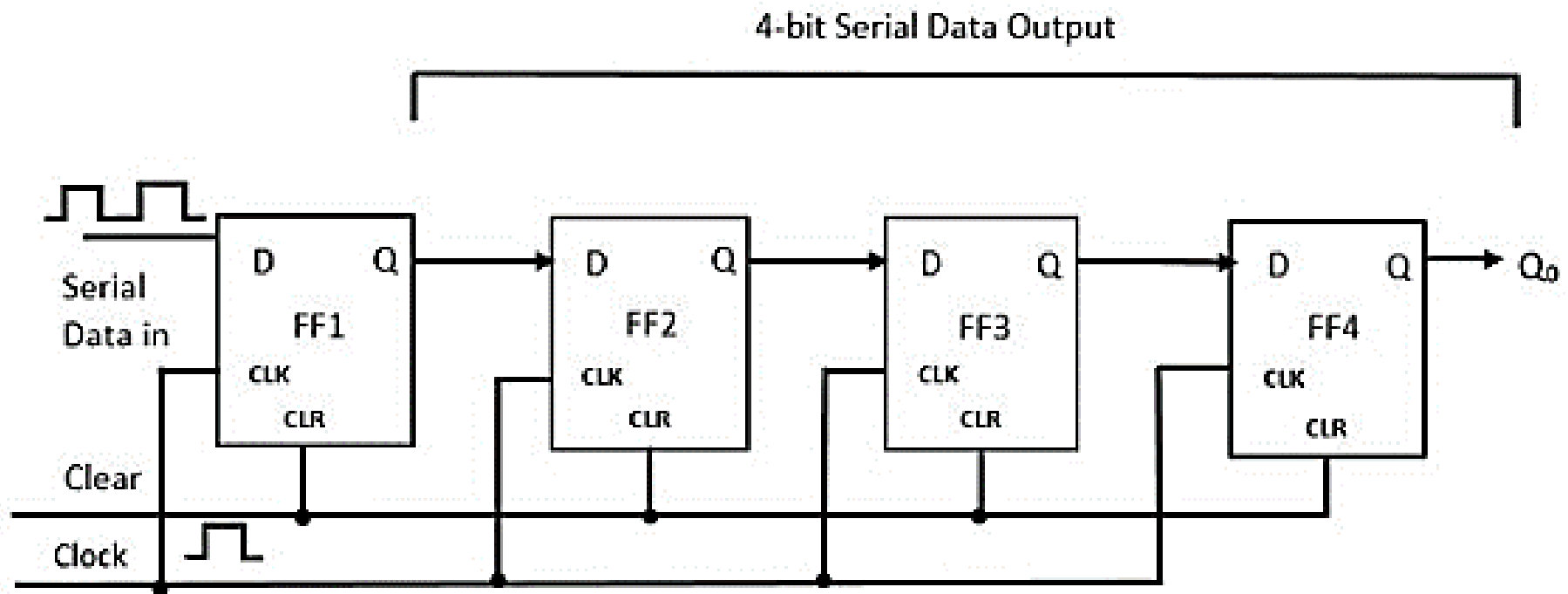
- **TRAP:** The TRAP interrupt is a **non-maskable interrupt** that is generated by an external device, such as a power failure or a hardware malfunction. The TRAP interrupt has the **highest priority and cannot be disabled or ignored by the instructions of CPU**.
- **RST 7.5:** The RST 7.5 interrupt is a **maskable** interrupt (can be disabled or ignored by the instructions of CPU) that is generated by a software instruction. It has the second highest priority.
- **RST 6.5:** The RST 6.5 interrupt is a **maskable** interrupt that is generated by a software instruction. It has the third highest priority.
- **RST 5.5:** The RST 5.5 interrupt is a **maskable** interrupt that is generated by a software instruction. It has the fourth highest priority.
- **INTR:** The INTR interrupt is a **maskable** interrupt that is generated by an external device, such as a keyboard or a mouse. It has the **lowest priority** and can be disabled. INTA is an interrupt acknowledgement sent by the microprocessor after INTR is received.

## 11. Serial Input/output control:

- It controls the serial data communication by using Serial input data and Serial output data. **Serial Input/Output control in the 8085 microprocessor refers to the communication of data between the microprocessor and external devices in a serial manner, i.e., one bit at a time.**
- The 8085 has a serial I/O port (SID/SOD) for serial communication. The **SID (Serial Data Input)** pin is used for serial input. In SID, the **RIM (Read Interrupt Mask)** instruction is initiated to input data in a serial manner. **RIM** instruction reads the current value of the **IMR (Interrupt Mask Register)** and copies it to the accumulator. The IMR is a register that controls which interrupts are allowed to be processed by the microprocessor.
- The **SOD (Serial Data Output)** pin is used for serial output. The **SIM (Set Interrupt Mask)** instruction should be initiated in order to output data in serial manner. SIM takes one byte as an operand and sets the corresponding bits in the IMR based on the value of the operand.

# 11. Serial Input/output control:

- The timing and control of serial communication is managed by the 8085's internal circuitry (Timing and control unit). The 8085 also has two special purpose registers, the **Serial Control Register (SC)** and the **Serial Shift Register (SS)**, which are used to control and monitor the serial communication. These are nothing but shift registers.



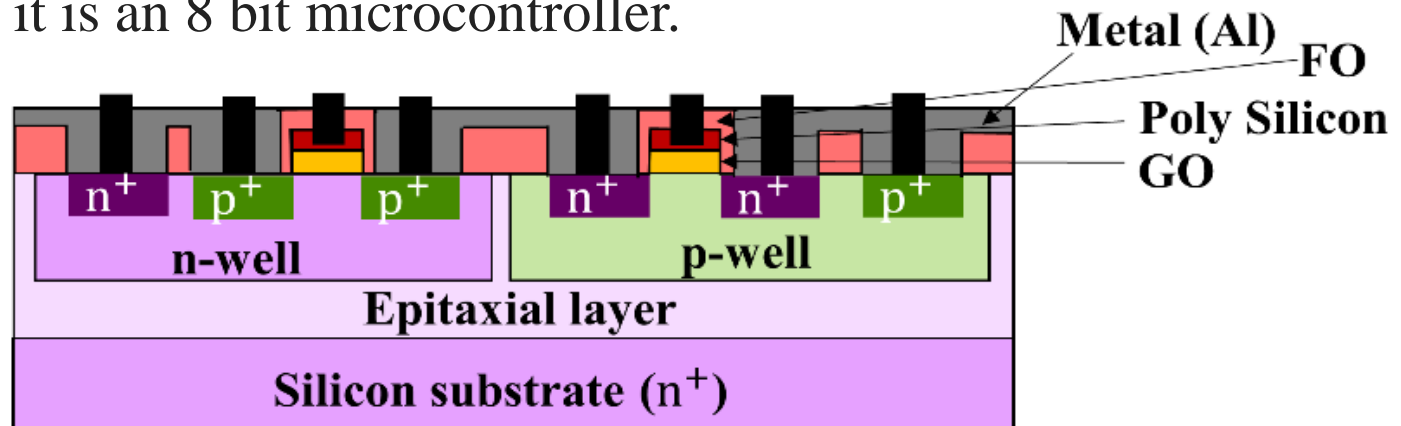
(16) (PDF) Area and Speed Efficient Layout Design of Shift Registers using Nanometer Technology (researchgate.net)



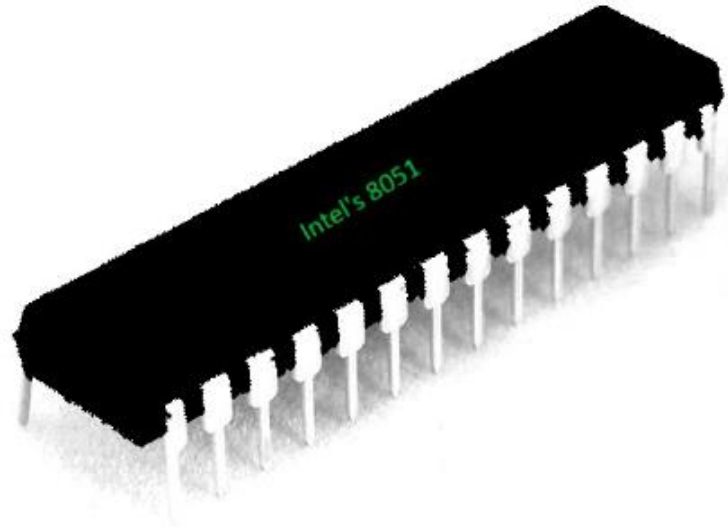
# Intel 8051 microcontroller architecture

- When microprocessor and additional outside circuitry (memory, input/output devices, timers) are **built in a single semiconductor chip (System on a Chip (SoC))**, this is called as **microcomputer**. Microcomputers are used in control applications – they are also designated as **microcontroller**.
- The 8051 Microcontroller is an **8-bit microcontroller designed by Intel in 1981**. Number of transistors utilized in this microcontroller is ~10 K.

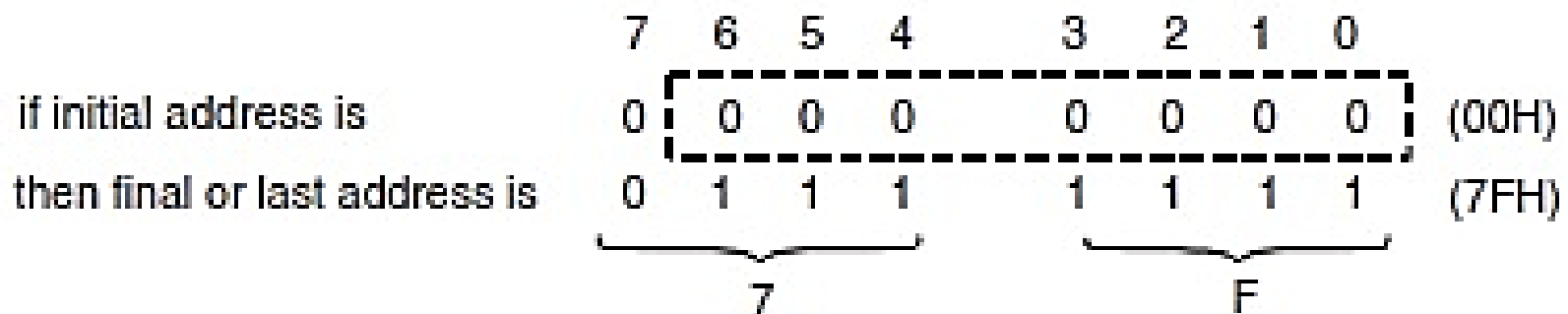
Normally, this microcontroller was developed using **NMOS** technology, which requires more power to operate. Therefore, Intel redesigned Microcontroller 8051 using **CMOS (Complementary Metal Oxide Semiconductor)** technology and their updated versions came with a letter C in their name, for instance an 80C51 it is an 8 bit microcontroller.



# Intel 8051 microcontroller architecture



- It is based on a **Harvard architecture**.
- It includes 128 ( $2^7$ ) bytes of Random Access memory (RAM).



- It includes 4KB read only memory (ROM) is available for program storage. This is used for permanent data storage.

$$4\text{KB} = 2^2 \cdot 2^{10} \text{ B (since 1KB} = 2^{10} \text{ B)}$$

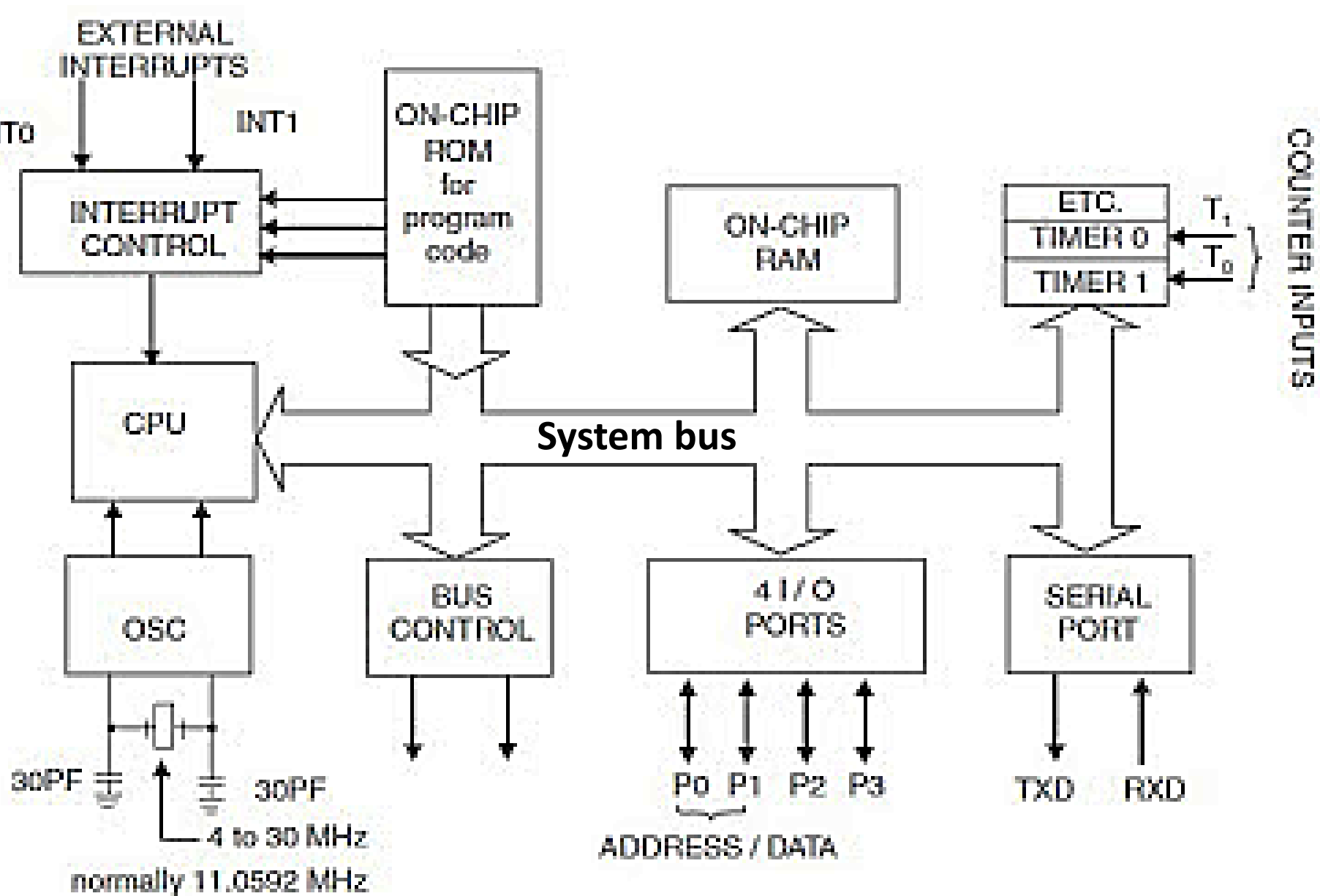
$$= 2^{12} \text{ Byte}$$

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Starting Address	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	(0000H)
Last Address	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	(0FFFH)
	0				F				F				F				

- It includes 4KB of on-chip flash memory, and an 8-bit CPU with an instruction set that includes a variety of arithmetic and logic operations.
- It has four parallel 8-bit ports that are programmable and addressable based on the requirement.
- It also includes two 16-bit timers and an on-chip crystal oscillator of frequency 12 MHz.
- A UART (Universal Asynchronous Receiver/Transmitter) receives serial data and stores it as parallel data (usually one byte), and takes parallel data and transmits it as serial data.

# Intel 8051 microcontroller architecture

MICROCONTROLLER 8051 ARCHITECTURE ~ LEARN ABOUT ELECTRONICS (electronicsfrom.blogspot.com)



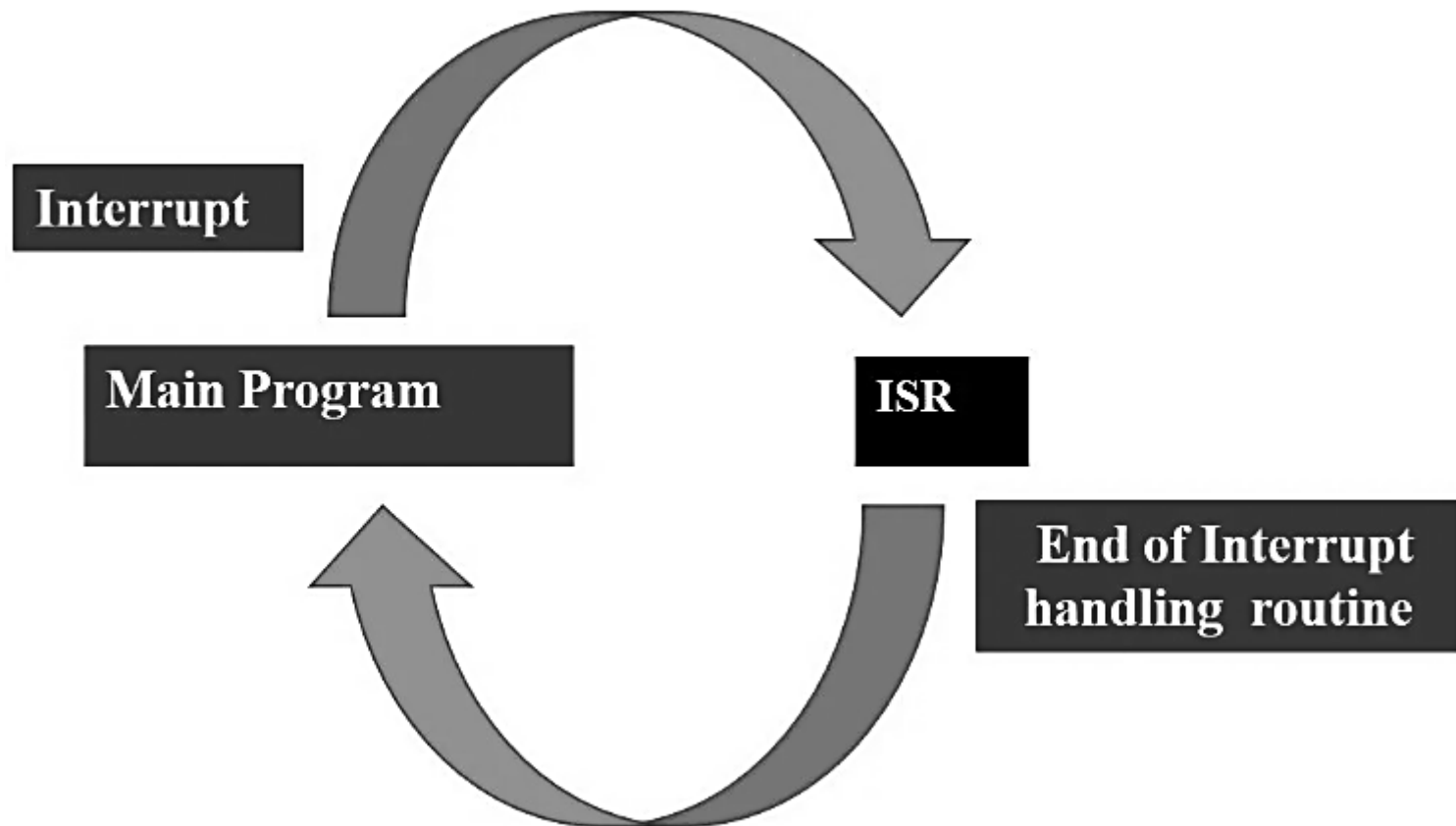
# Intel 8051 microcontroller architecture

**Central Processor Unit (CPU):** As we know that the CPU is the **brain** of any processing device of the microcontroller. **It monitors and controls all operations that are performed on the Microcontroller units.** The User has no control over the work of the CPU directly . It reads program written in **program memory (ROM)**, fetch data from **data memory (RAM)** and executes them and do the expected task of that application. Generated output will be stored in data memory (RAM) or transferred to the output port.

**Interrupts:** The **interrupt is a signal emitted by hardware or software when a process or an event needs immediate attention.** It alerts the microcontroller to a high-priority process requiring interruption of the current working process. microcontroller services the interrupt by executing a subroutine called **interrupt service routine (ISR)**. The feature of Interrupt is very useful as it helps in case of emergency operations. An Interrupts gives us a mechanism to put on hold the ongoing operations, execute a subroutine and then again resumes to another type of operations.

# Intel 8051 microcontroller architecture

**Interrupts:** The Microcontroller 8051 can be configured in such a way that it temporarily terminates or pause the main program at the occurrence of interrupts. When a subroutine is completed, Then the execution of main program starts.



# Intel 8051 microcontroller architecture

**Interrupt Vector Table:** A fixed memory area is assigned for each interrupt inside the microcontroller. The **Interrupt Vector Table** contains the **starting address of the memory location of every interrupt**.

When an interrupt occurs, the controller transfers the contents of the program counter onto the stack. Then it jumps to the memory location, which is specified by the **Interrupt Vector Table (IVT)**.

The code written in this memory area by the programmer starts its execution. We refer to this code as an **Interrupt Service Routine (ISR)** or an interrupt handler on the 8051 microcontroller.

Interrupts	Memory Location	Pin	Flag Clearing
Reset	0000	9	Auto
Timer0	000B		Auto
Timer1	001B		Auto
INT0	0003	12	Auto
INT1	0013	13	Auto
Serial com	0023		Cleared by programmer

# Intel 8051 microcontroller architecture

**8051 Interrupt Types:** The 8051 microcontroller can recognize **six different types of events**. These events can request the microcontroller to temporarily stop the execution of the current program and instead run a special block of code first. The interrupt sources present in 8051 microcontrollers are:

**Reset interrupt:** When the **reset pin activates**, the **program execution flow jumps to execute code from the 0000H memory location**. Generally, we don't use this pin. It is also known as power-on reset.

**Timer0 overflow interrupt (TF0) and timer1 overflow interrupt (TF1):** Two timers (T0 and T1) are present in the 8051 microcontroller and are responsible for a Timer interrupt. **A timer interrupt informs the microcontroller that the corresponding timer has finished counting**. Memory locations 000BH and 001BH in the interrupt vector table belong to Timer0 and Timer1, respectively.



# Intel 8051 microcontroller architecture

## 8051 Interrupt Types:

**External hardware interrupt (INT0 and INT1):** There are two external interrupts (INT0 and INT1) to serve external devices. **Pin numbers 12 and 13 in port 3 are for the external hardware interrupts.** Both of these interrupts are **active low**. An **external interrupt informs the microcontroller that an external device needs routine service**. Memory locations 0003H and 0013H in the interrupt vector table belong to INT0 and INT1, respectively.

**Serial communication interrupt (RI/TI):** This interrupt is useful for serial communication. It has a single interrupt that belongs to both receive and transmit. **When enabled, it notifies the controller whether a byte has been received or transmitted.** This interrupt vector table assigns location 0023H to this interrupt.

# Intel 8051 microcontroller architecture

**Memory:** Intel 8051 microcontroller is based on a **Harvard architecture**. The memory which is used to store the program of the microcontroller is known as **code memory or Program memory**. It is known as **ROM** memory. Microcontroller also requires a memory to store data or operands temporarily. The **data memory** of the 8051 is used to store data temporarily for operation is known **RAM** memory. 8051 microcontroller has 4KB of code memory or program memory, that has 4KB ROM and also 128 bytes of data memory of RAM.

# Intel 8051 microcontroller architecture

**BUS:** Basically **Bus** is a collection of wires which work as a communication channel or medium for transfer of **Data**. These buses consists of 8, 16 or more wires of the microcontroller. Thus, these can carry 8 bits,16 bits simultaneously.

**Address Bus:** Microcontroller 8051 has a **16 bit address bus** for transferring the data. It is used to address memory locations and to transfer the address from CPU to Memory of the microcontroller. **It has four addressing modes that are Immediate addressing modes, Bank address (or) Register addressing mode, Direct Addressing mode, Register indirect addressing mode.**

**Data Bus:** Microcontroller 8051 has 8 bits of the data bus, which is used to carry data of particular applications.

# Intel 8051 microcontroller architecture

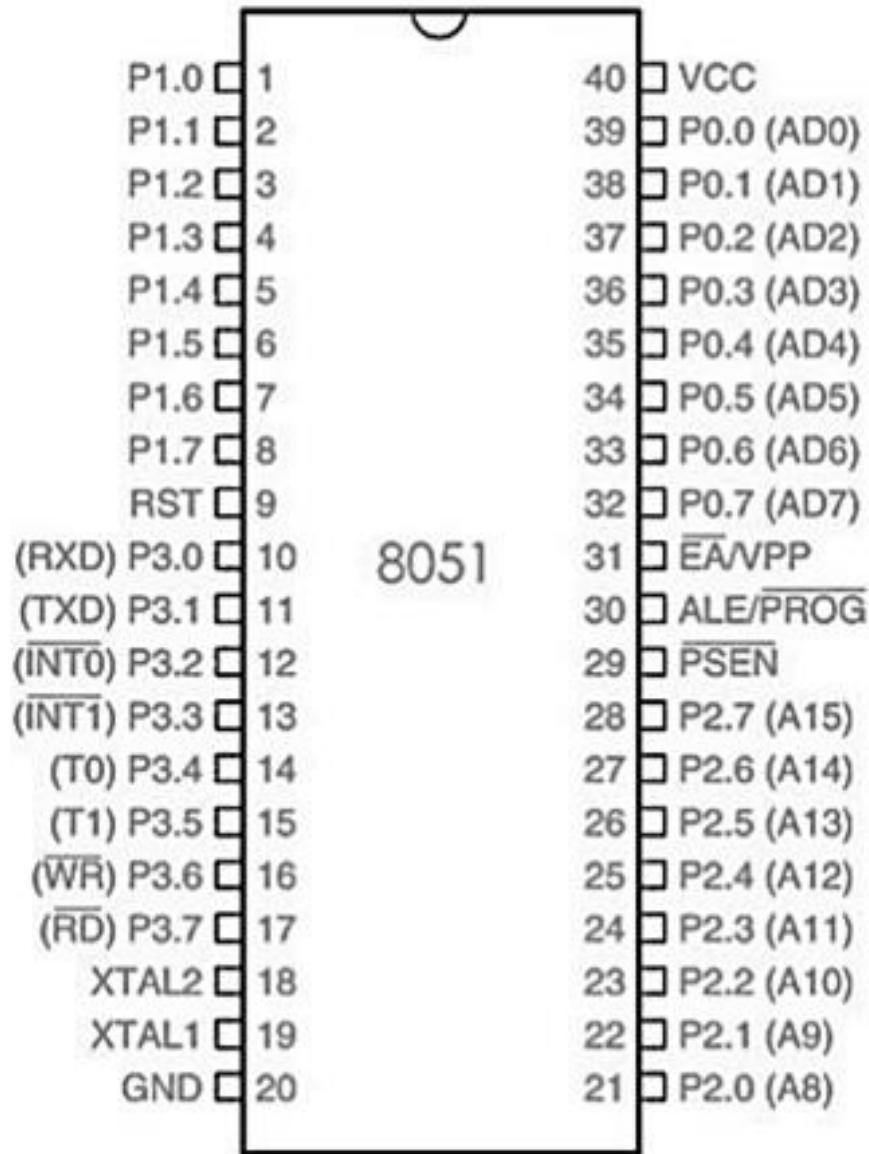
**Oscillator:** Generally, we know that the microcontroller is a computing device, therefore it requires clock pulses for its operation of microcontroller applications. For this purpose, microcontroller 8051 has an **on-chip crystal oscillator of frequency 12 MHz** which works as a **clock source for Central Processing Unit** of the microcontroller. The output pulses of oscillator are stable. Therefore, it enables synchronized work of all parts of the 8051 Microcontroller.

**Input/Output Port:** Normally microcontroller is used in embedded systems to control the operation of machines in the microcontroller. Therefore, to connect it to other machines, devices or peripherals we require I/O interfacing ports in the microcontroller interface. For this purpose microcontroller 8051 has 4 input, output ports to connect it to the other peripherals

# Intel 8051 microcontroller architecture

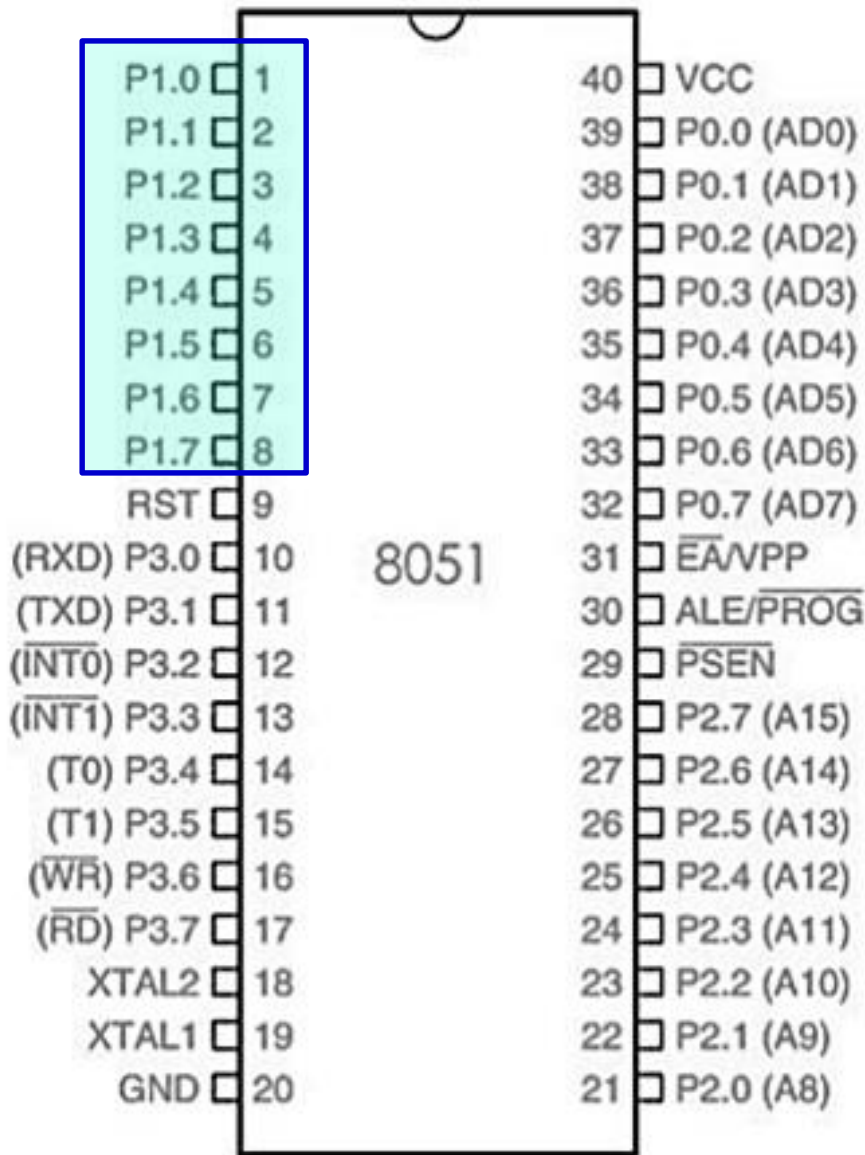
8051 microcontroller has two 16 bit timers and counters. These counters are again divided into a 8 bit register. The timers are used for measurement of intervals to determine the pulse width of pulses.

# Pin diagram of 8051 Microcontroller



**40 - PIN DIP**

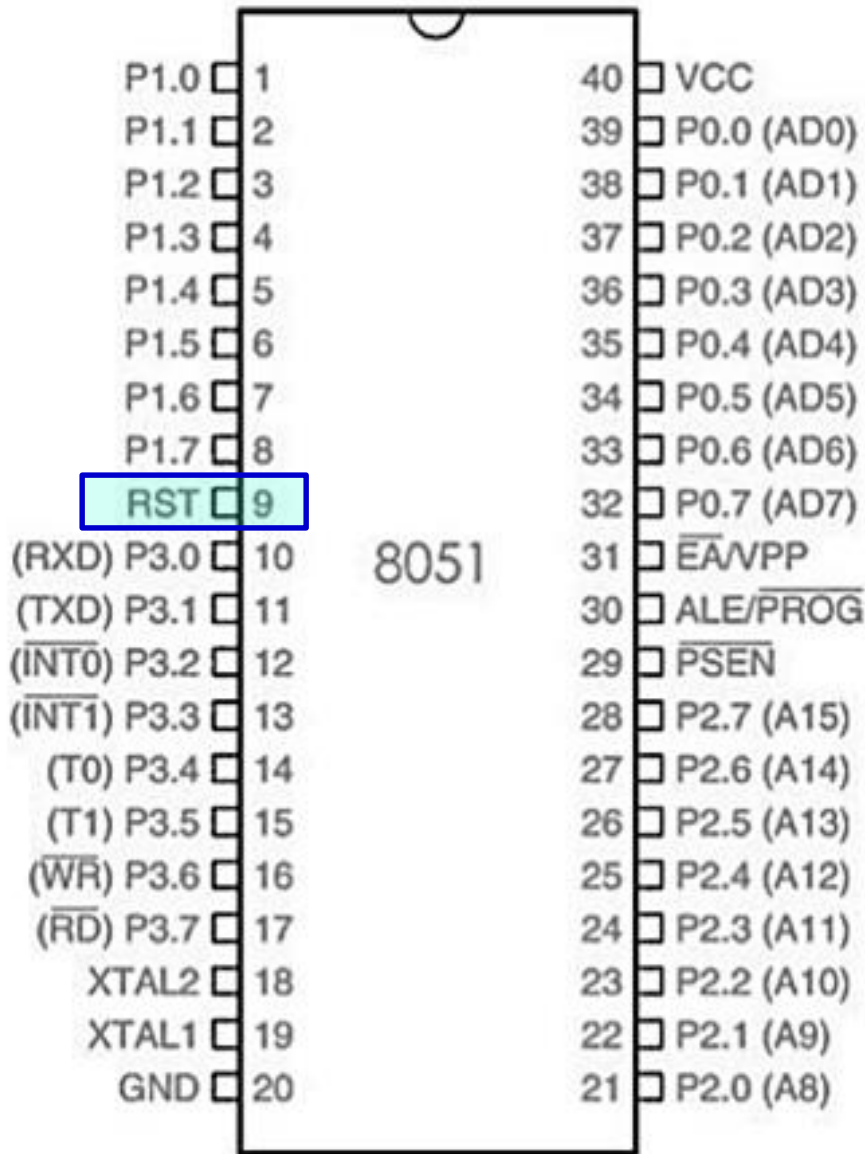
# Pin diagram of 8051 Microcontroller



**40 - PIN DIP**

• **Pin 1 to Pin 8 (Port 1) – Pin 1 to Pin 8 are assigned to Port 1 for simple I/O operations.** They can be configured as input or output pins depending on the logic control i.e. 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. These pins are also referred to as P1.0 to P1.7 (where **P1** indicates that it is a pin in port 1 and the number after ‘.’ tells the pin number i.e. 0 indicates first pin of the port. So, P1.0 means first pin of port 1, P1.1 means second pin of the port 1 and so on). These pins are bidirectional pins.

# Pin diagram of 8051 Microcontroller

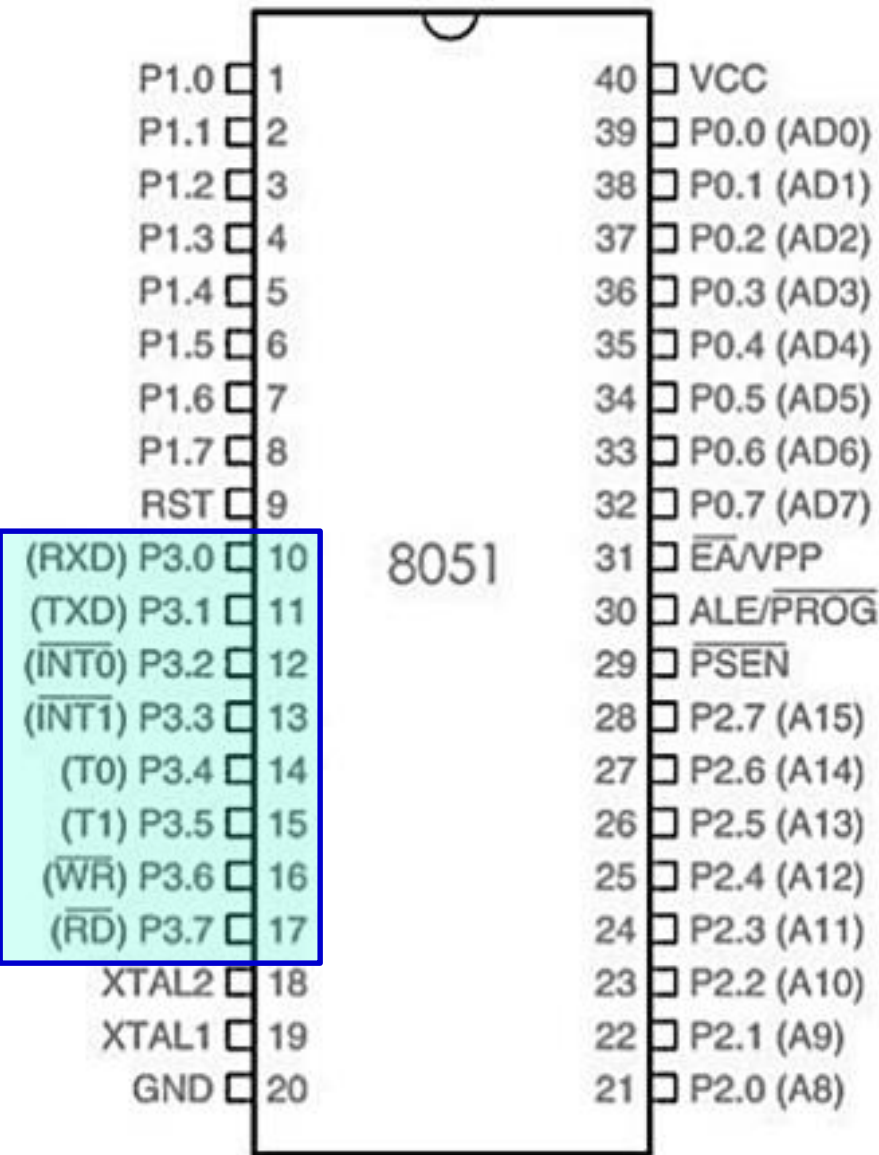


• **Pin 9 (RST) – Reset pin.** It is an **active-high, input pin**. Therefore if the **RST pin is high for a minimum of 2 machine cycles**, the microcontroller will reset i.e. it will close and terminate all activities. It is often referred as “power-on-reset” pin because it is used to reset the microcontroller to its initial values when power is on (high).

**40 - PIN DIP**



# Pin diagram of 8051 Microcontroller



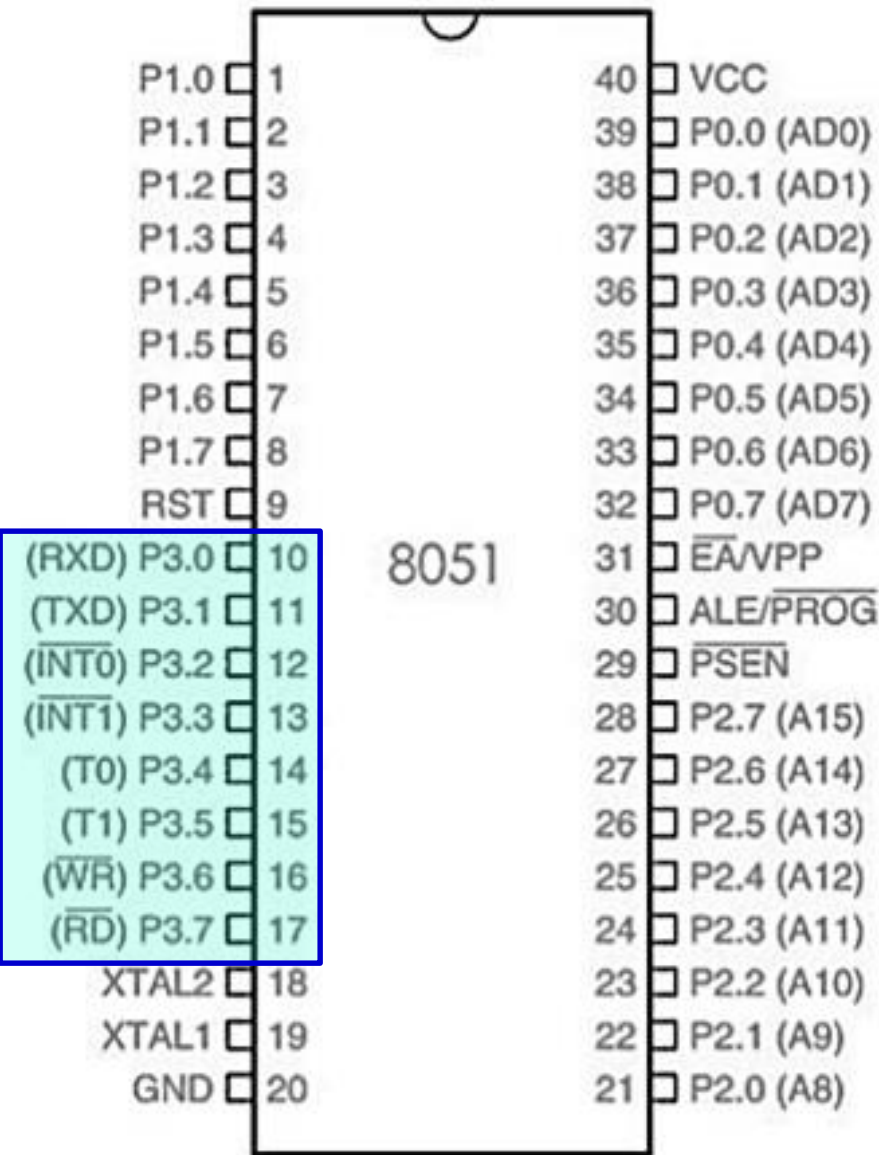
40 - PIN DIP

**Pin 10 to Pin 17 (Port 3)** – Pin 10 to pin 17 are port 3 pins which are also referred to as P3.0 to P3.7. These pins are similar to port 1 and can be used as universal input or output pins. These pins are bidirectional pins. These pins also have some additional functions which are as follows:

**P3.0 (RXD):** 10th pin is RXD (**serial data receive pin**) which is for **serial input**. Through this input signal microcontroller receives data for serial communication.

**P3.1 (TXD):** 11th pin is TXD (**serial data transmit pin**) which is **serial output pin**. Through this output signal microcontroller transmits data for serial communication.

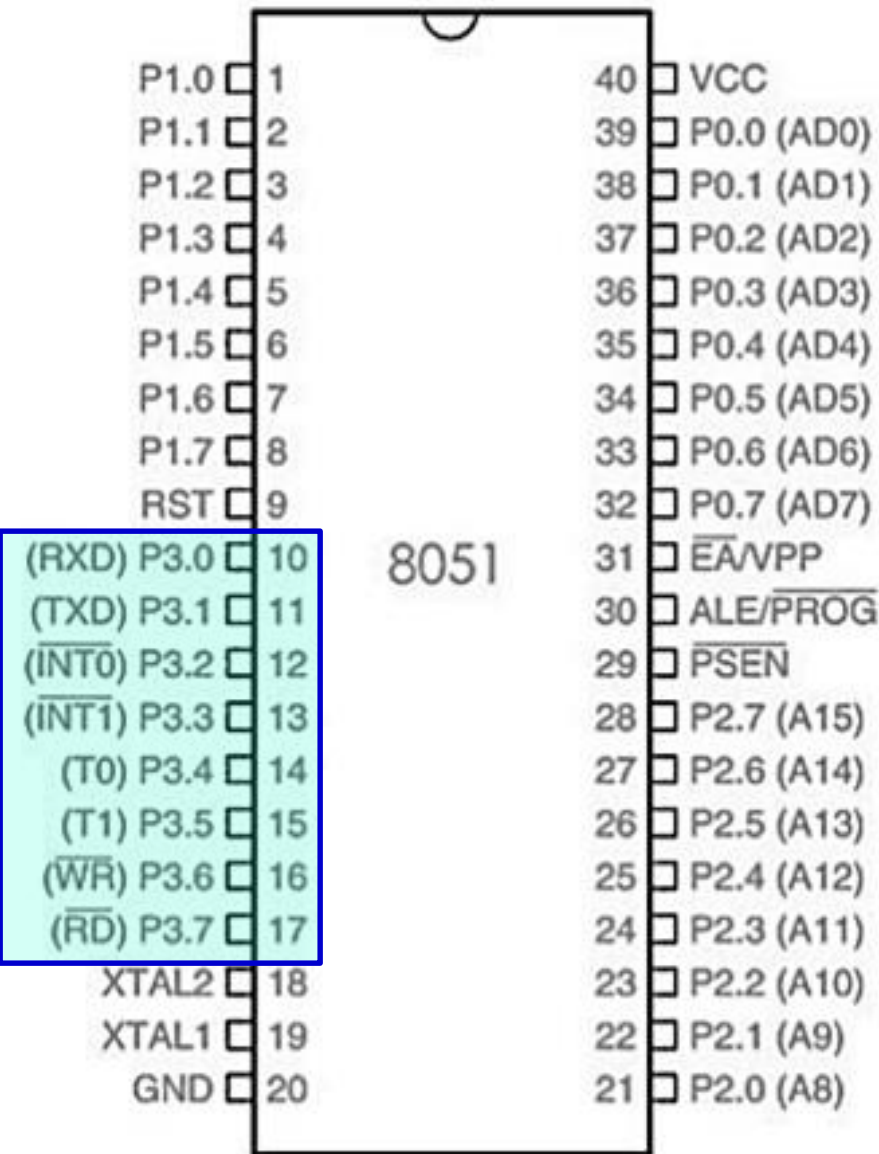
# Pin diagram of 8051 Microcontroller



**P3.2 and P3.3 ( $\text{INT0}'$ ,  $\text{INT1}'$ )** : 12th and 13th pins are for **External Hardware Interrupt 0 and Interrupt 1 respectively**. When this interrupt is activated (i.e. when it is low), 8051 gets interrupted in whatever it is doing and jumps to the vector value of the interrupt (**0003H for INT0 and 0013H for INT1**) and starts performing Interrupt Service Routine (ISR) from that vector location.

**P3.4 and P3.5 (T0 and T1)** : 14th and 15th pin are for **Timer 0 and Timer 1 external input**. They can be connected with **16 bit timer/counter**.

# Pin diagram of 8051 Microcontroller

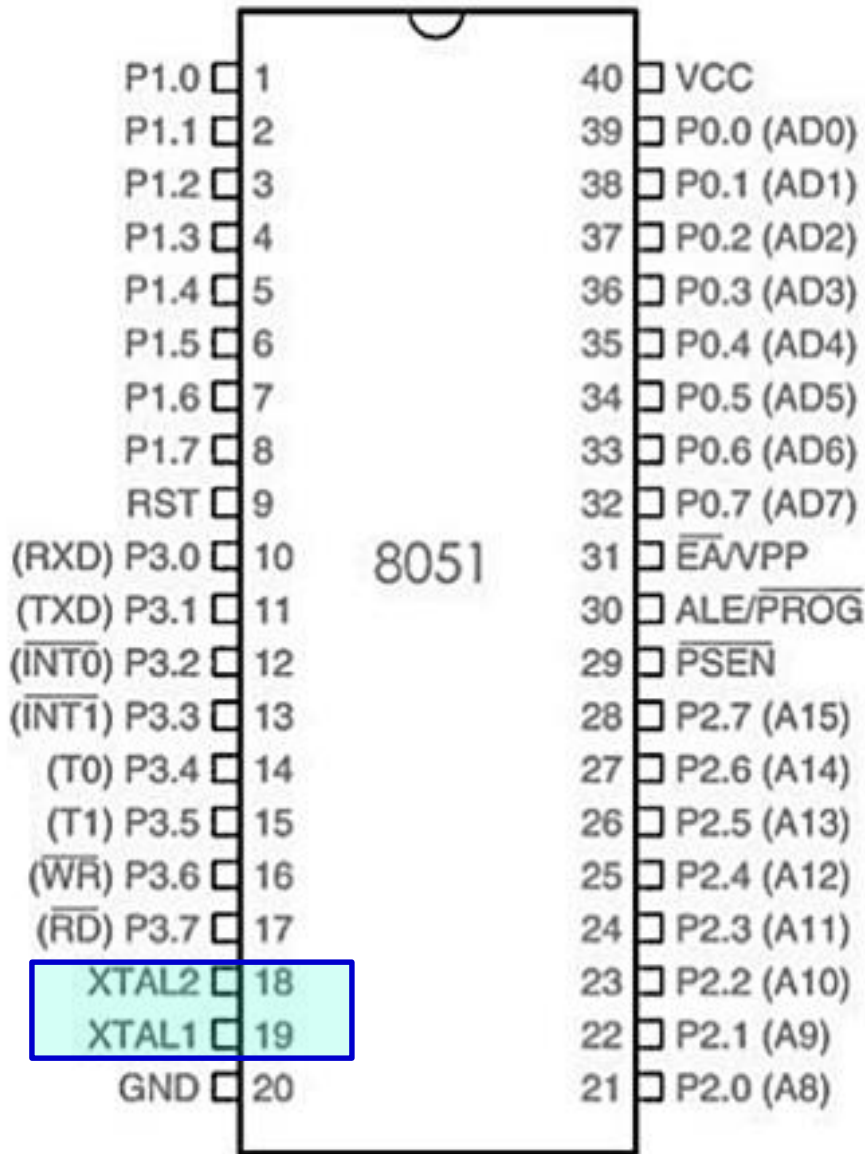


**P3.6 (WR')** : 16th pin is for **external memory write** i.e. writing data to the external memory.

**P3.7 (RD')** : 17th pin is for **external memory read** i.e. reading data from external memory.

**40 - PIN DIP**

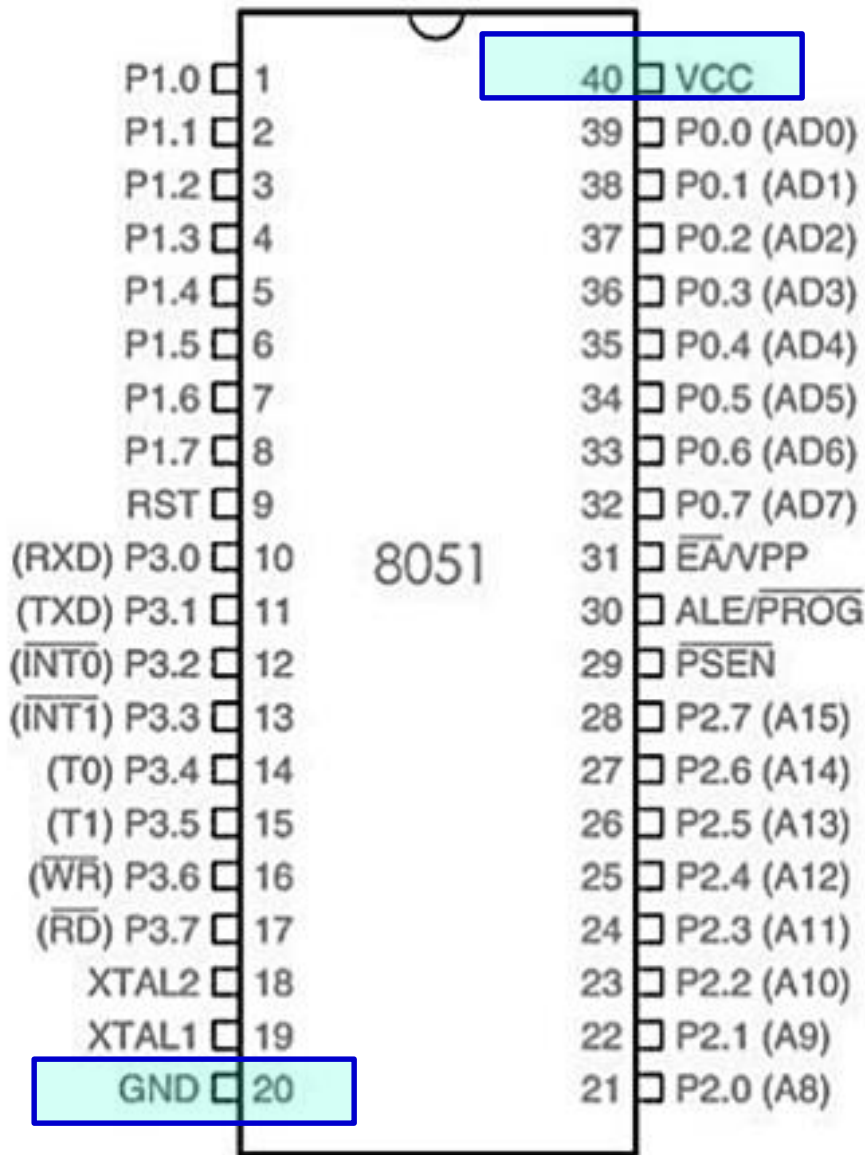
# Pin diagram of 8051 Microcontroller



**Pin 18 and Pin 19 (XTAL2 And XTAL1)** – These pins are **connected to an external oscillator** which is generally a quartz crystal oscillator. They are used to provide an external clock frequency of **4MHz to 30MHz**.

**40 - PIN DIP**

# Pin diagram of 8051 Microcontroller

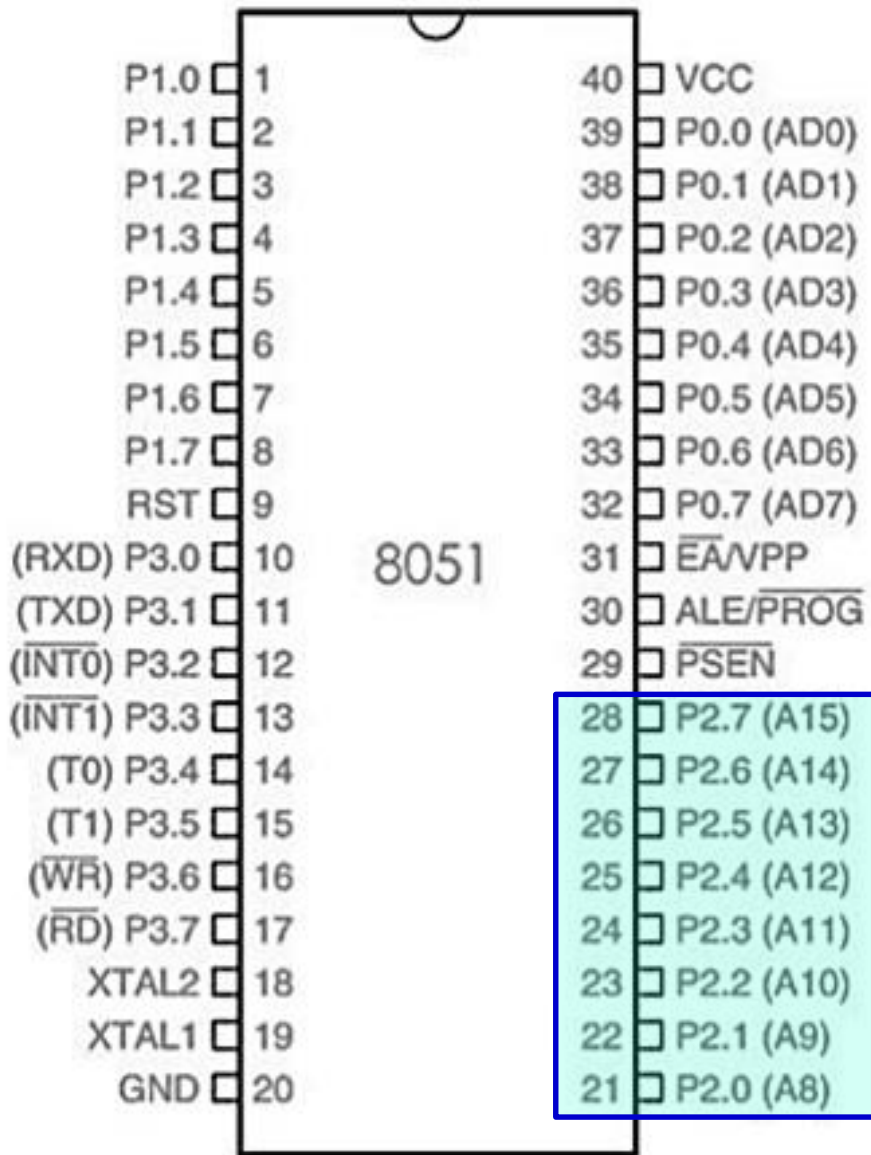


**Pin 20 (GND)** – This pin is connected to the ground. It has to be provided with 0V power supply. Hence it is connected to the negative terminal of the power supply.

**Pin 40 (VCC)** – This pin provides power supply voltage i.e. +5 Volts to the circuit.



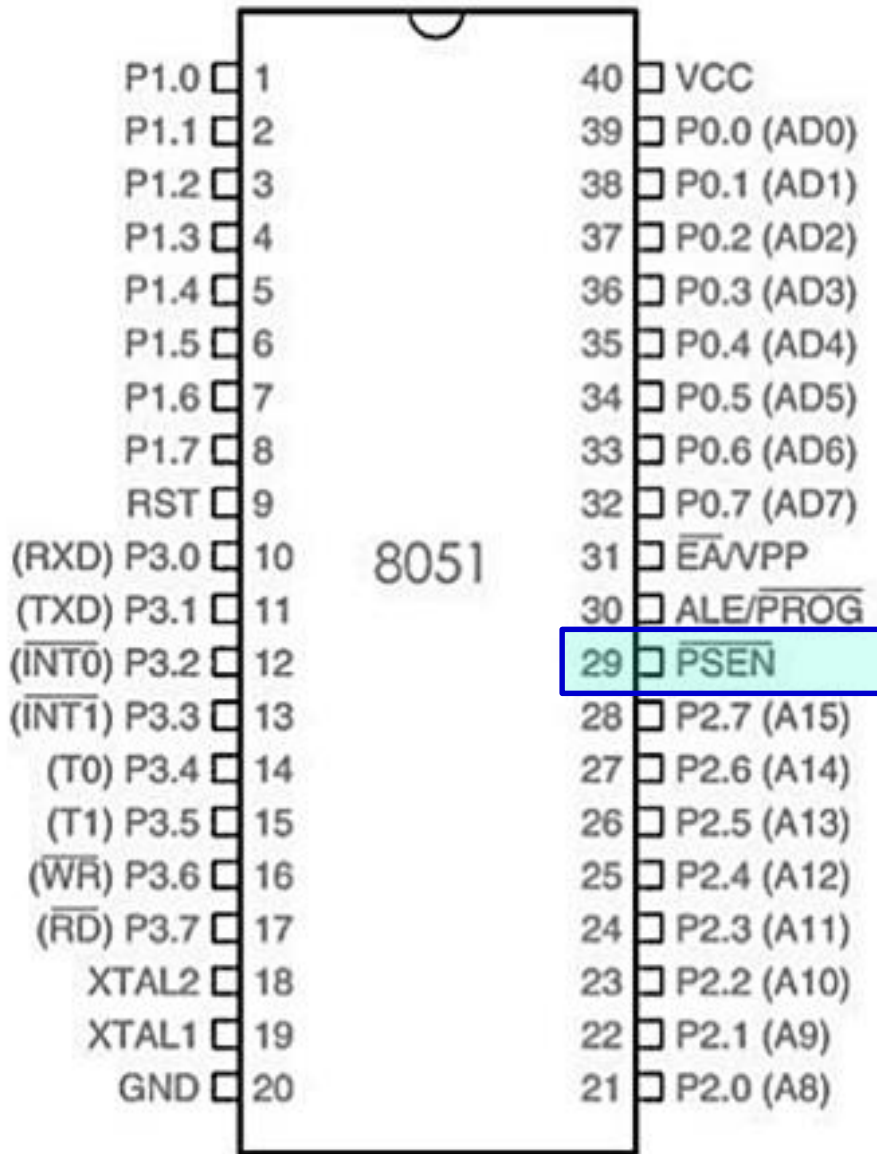
# Pin diagram of 8051 Microcontroller



**Pin 21 to Pin 28 (Port 2)** – Pin 21 to pin 28 are **port 2 pins** also referred to as P2.0 to P2.7. When additional external memory is interfaced with the 8051 microcontroller, pins of port 2 act as higher-order address bytes. These pins are bidirectional.

**40 - PIN DIP**

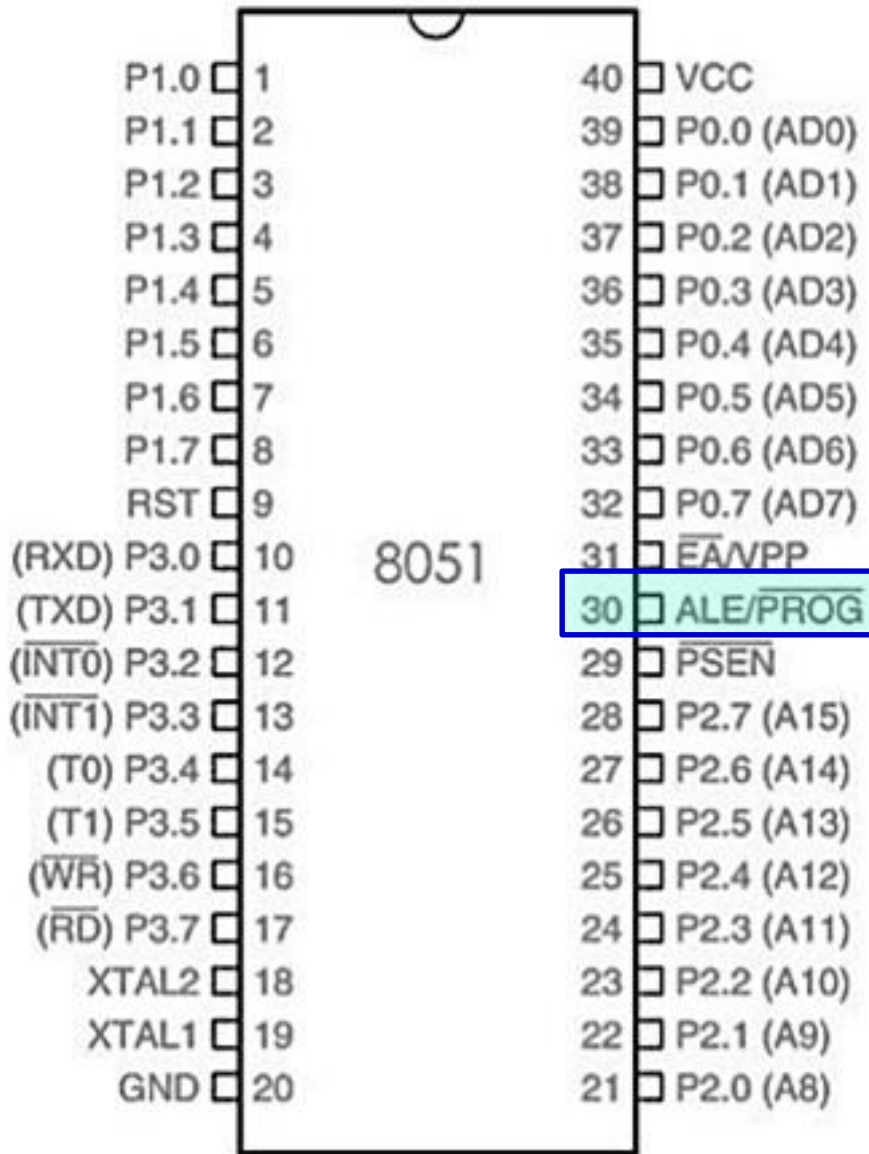
# Pin diagram of 8051 Microcontroller



**Pin 29 ( $\overline{\text{PSEN}}$ )** –  $\overline{\text{PSEN}}$  stands for **Program Store Enable**. It is output, active-low pin. This is used to read external memory.

**40 - PIN DIP**

# Pin diagram of 8051 Microcontroller



**Pin 30 (ALE/ PROG)** – ALE stands for **Address Latch Enable**. It is **input, active-high pin**. This pin is used to **distinguish between memory chips** when multiple memory chips are used. It is also used to **de-multiplex the multiplexed address and data signals available at port 0**. During flash programming i.e. Programming of EPROM, this pin acts as program pulse input (PROG).

**40 - PIN DIP**



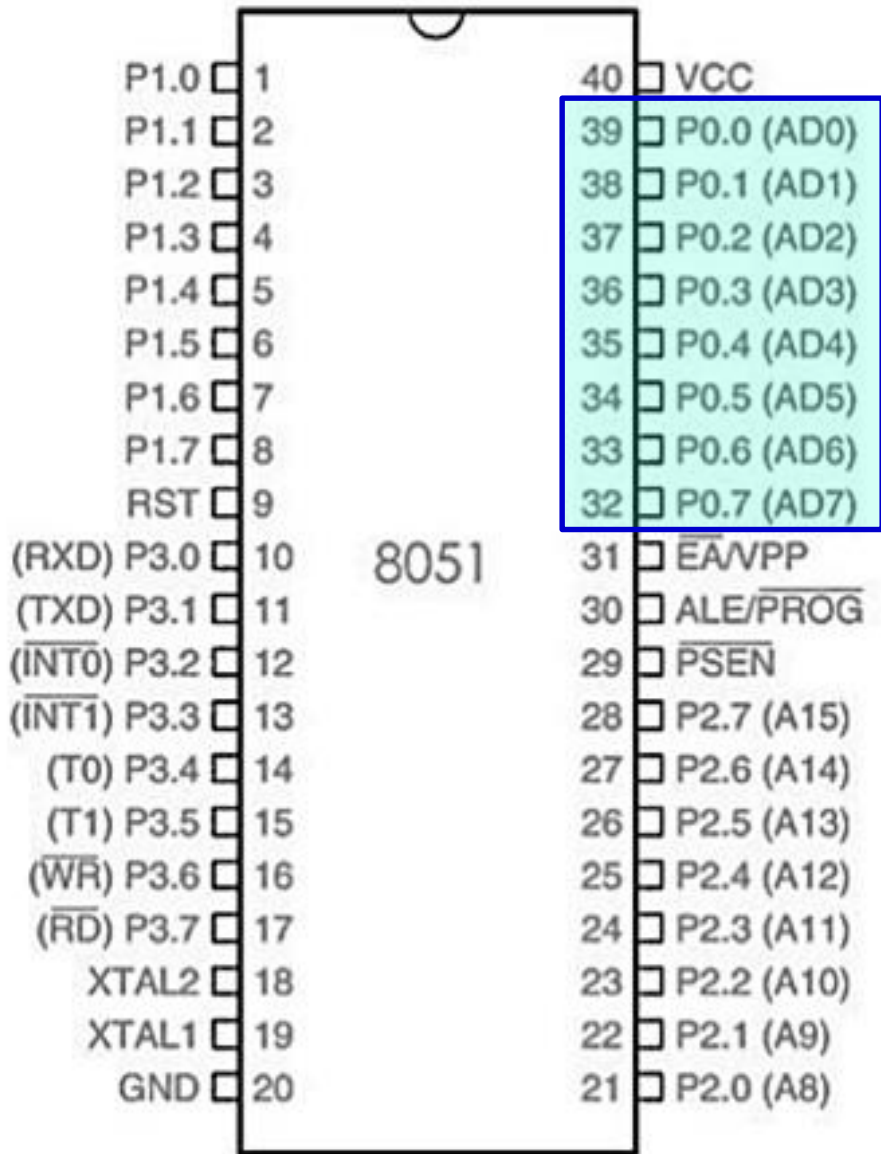
# Pin diagram of 8051 Microcontroller



**Pin 31 ( $\overline{\text{EA/VPP}}$ )** – EA stands for **External Access input**. It is **used to enable/disable external memory interfacing**. In 8051, EA is connected to Vcc as it comes with on-chip ROM to store programs.

**40 - PIN DIP**

# Pin diagram of 8051 Microcontroller



**Pin 32 to Pin 39 (Port 0)** – Pin 32 to pin 39 are port 0 pins also referred to as P0.0 to P0.7. They are **bidirectional input/output pins**. Port 0 is also designated as **AD0-AD7** because 8051 multiplexes address and data through port 0 to save pins.

**40 - PIN DIP**

# Difference between RISC and CISC architecture

Parameters	RISC	CISC
Full form	Reduced Instruction Set Architecture	Complex Instruction Set Architecture
Instruction Size	Fixed size	Variable size
Instruction fetch time	Same for all instruction	Vary with respect to instructions
Instruction set	Small and simple	Large and complex
Dealing with	Registers	Registers as well as memory
Addressing modes	Less modes as most instructions are based on registers. Only the load and store operation deals with memory.	More modes as complex instructions are available with different verities.

# Difference between RISC and CISC architecture

Parameters	RISC	CISC
Number of registers	Many	Few
Compiler design	Simple: small instruction set and less number of addressing modes	Complex: instruction set is complex and many addressing modes are required
Program size	Long (Weak code density)	Small (Better code density): complex instructions are available
Number of operands	Fixed (Mainly in registers)	Variable (Can be in registers and memory)
Control unit	Hardware controlled	Micro program controlled: if we use hardware control it will be very costly

# Difference between RISC and CISC architecture

Parameters	RISC	CISC
Execution speed	Faster	Slower
Pipelining	Efficient: As instruction sizes are fixed	Inefficient: As instruction sizes are variable

# Pipelining

## Instruction execution without pipelining

<b>F1</b>	<b>E1</b>	<b>F2</b>	<b>E2</b>	<b>F3</b>	<b>E3</b>	<b>F4</b>	<b>E4</b>
-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------

## Instruction execution with pipelining

<b>F1</b>	<b>E1</b>	<b>E2</b>	<b>E3</b>	<b>F4</b>
	<b>F2</b>	<b>F3</b>	<b>F4</b>	

# ARM microcontroller architecture

**ARM** is short for “**Advanced RISC Machines**”. The ARM processor belongs to the family of CPUs which are based primarily on **Reduced Instruction Set Computer (RISC)**. The ARM processors could be of **32 bit or 64 bit**. The RISC processors are higher in speed because they perform a small number of instructions.

The ARM processors have a less number of transistors because they have a reduced instruction set, which allows a smaller size for the IC. Thereby being **space efficient** also. Most of the electronic devices such as **tablets, mobiles, smart phones and other mobile devices** consist of these processors.

**By combining the ARM microprocessor with RAM, ROM and other peripherals in one single chip, we get an ARM microcontroller, for example, LPC2148.**

**Developed by Acorn computer limited.**



# ARM microcontroller architecture

ARM architecture is basically a RISC architecture (large uniform register file, load/store architecture, simple addressing modes, uniform and fixed length instruction fields).

Advancements, that have done in ARM architecture compared to any other RISC architecture is that each instruction controls the ALU and shifter, auto increment and auto decrement addressing modes, multiple load/store architecture (conditional execution).

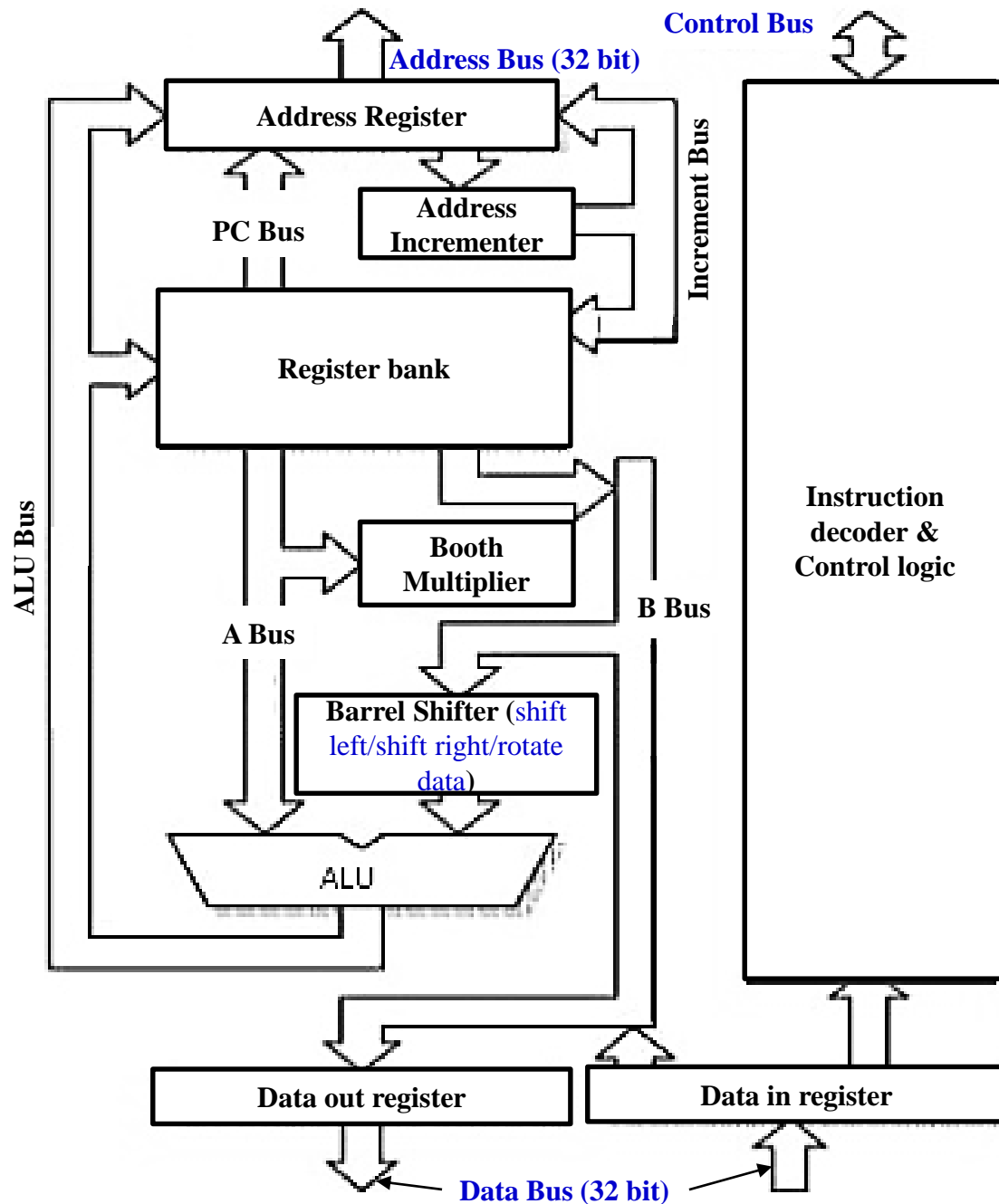
## Features of ARM architecture:

- High performance
- Low code size
- Low power consumption
- Consists of a 3 stage pipeline which fetches the instructions, then decodes it and then finally executes the instruction
- Low silicon area in IC chip

ARM microcontroller implements 2 type of instruction sets: 32 bit ARM instruction set and 16 bit thumb instruction set.



# ARM microcontroller architecture



# ARM microcontroller architecture

1. **Arithmetic and Logic Unit (ALU):** It is a combinational logic circuit that performs Arithmetic Operations and Logical operations.
2. **Booth multiplier:** It multiplies two signed binary numbers in 2's complement notation.
3. **Barrel shifter:** 32 bit barrel shifter is digital circuit (combinational) that can shift data (shift left/shift right) by a specific number of bits without the use of any sequential logic. Barrel shifter performs preprocessing of data before execution of a particular operation in ALU. Thus **it speeds up the execution process**.
4. **Control unit:** It is responsible for the system operation and it controls the flow of data between processor and other devices.

# ARM microcontroller architecture

5. **Register file:** ARM microcontroller has load (copy data from memory to register)/store (copy data from register to memory) **RISC architecture**. Here, arithmetic and logical operation only use register operands and cannot directly operate with memory locations.

**ARM Cortex M3 processor** consist of 37 register set (31 general purpose register (GPR) and 6 status register). These registers are part of the programmer's model.

- a. **Core Registers:** These are the fundamental registers used for general-purpose computation and control. They include:
- **General-purpose registers (R0-R12):** Used for temporary storage during arithmetic operations.
  - **R13: Stack Pointer (SP):** Points to the top of the stack.
  - **R14: Link Register (LR):** Stores the return address after a function call.
  - **R15: Program Counter (PC):** Holds the address of the next instruction to be executed.

# ARM microcontroller architecture

## b. Special Registers:

- **xPSR (Program Status Register):** Contains flags and status information.
- **MSP (Main Stack Pointer):** Used for exception handling.
- **PSP (Process Stack Pointer):** Used for thread mode.
- **Control:** Determines the privilege level and stack selection.

## c. Floating-Point Registers:

- **S0-S15:** Single-precision floating-point registers.
- **FPSCR (Floating-Point Status and Control Register):** Manages floating-point exceptions.

## d. Debug Registers:

- **DHCSR (Debug Halting Control and Status Register):** Controls debugging features.
- **DCRSR (Debug Core Register Selector Register):** Selects the register to access during debugging.

# Difference between 8051 and ARM microcontroller

8051 Microcontroller	ARM microcontroller
8 bit for standard core bus width is present in 8051 microcontroller – average performance.	32 bit/64 bit bus width is present in ARM microcontroller – high performance.
Average power consumption and less expensive.	Less power consumption and more expensive.
It is based on CISC Instruction set Architecture.	It is based on RISC Instruction Set Architecture.
Smaller number of built-in peripherals.	Greater number of built-in peripherals.
Von Neumann architecture.	Modified Harvard architecture.
Flash, ROM, SRAM memory is used in 8051 microcontroller.	Flash, EEPROM, SDRAM memory is used in ARM microcontroller.
UART, USART, I2C, SPI, communication protocols are used.	UART, USART, Ethernet, I2S, DSP, SPI, CAN, LIN, I2C communication protocols are used.

# Difference between 8051 and ARM microcontroller

8051 Microcontroller	ARM microcontroller
8051 microcontroller costs very low as compared to features provided.	ARM microcontroller costs low as compared to features provided.
Developed by Intel.	Developed by Acorn computer limited.