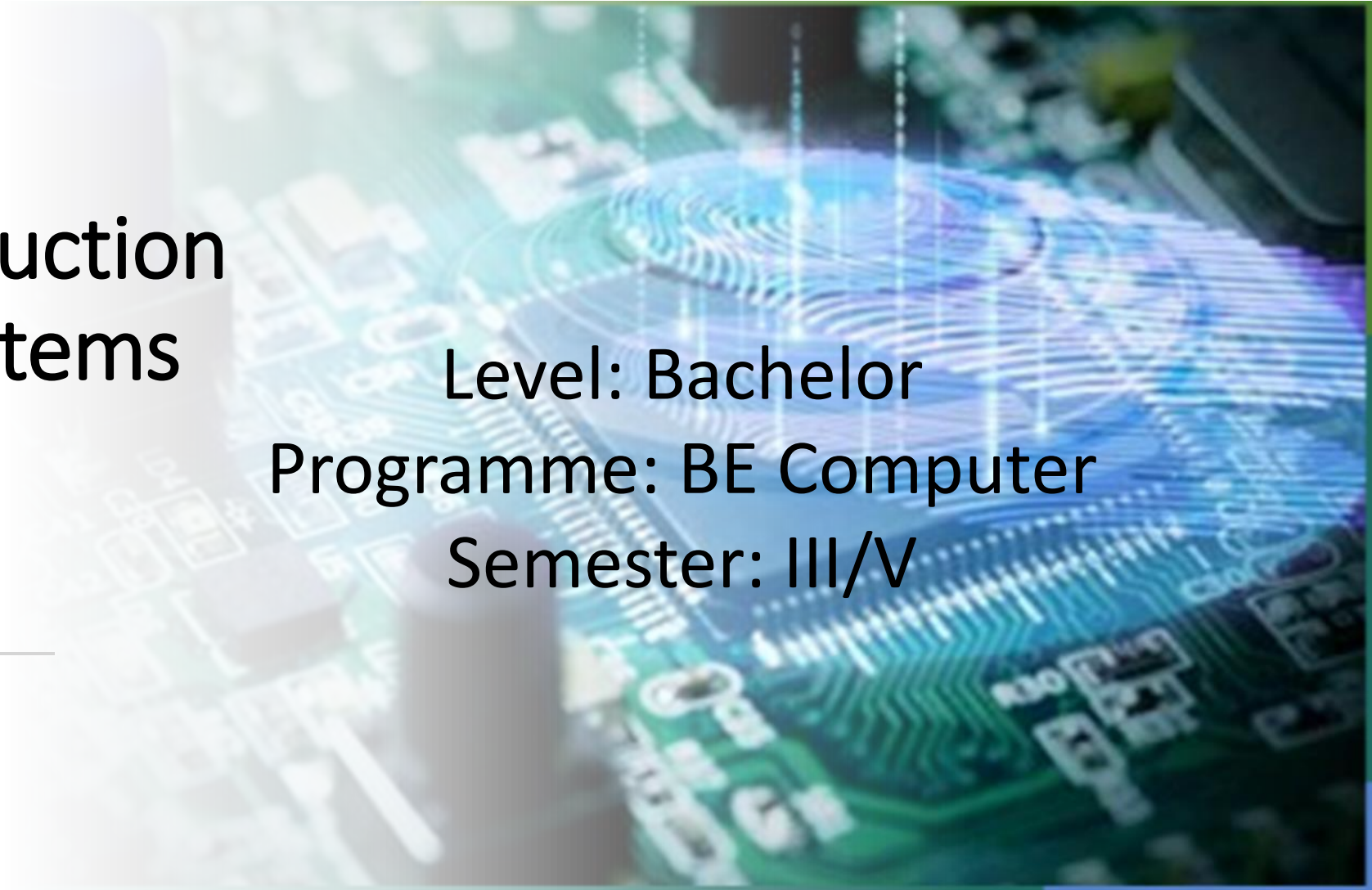


# Embedded

## Chapter 1. Introduction to Embedded Systems (3 hrs)



Level: Bachelor  
Programme: BE Computer  
Semester: III/V

---

Ishwor Koirala, Ph.D.

Instructor

# Outline

## Introduction

## Books

## Course Contents

- 1. Introduction to Embedded system (3 hrs)**
2. Programming for Embedded systems (5 hrs)
3. Real-time operating systems (RTOS) (5 hrs)
4. Embedded System Design using VHDL (5 hrs)
5. Communications Protocols (3 hrs)
6. Pheripherals and Interfacing (4 hrs)
7. Internet of Things (IoT)and Embedded system (3hrs)

# Introduction to Embedded System

- Any device that includes a computer but is not itself a general-purpose computer(Mainframe or Desktop).
- Respond to the external events.(e.g. someone pushes an elevator button.)

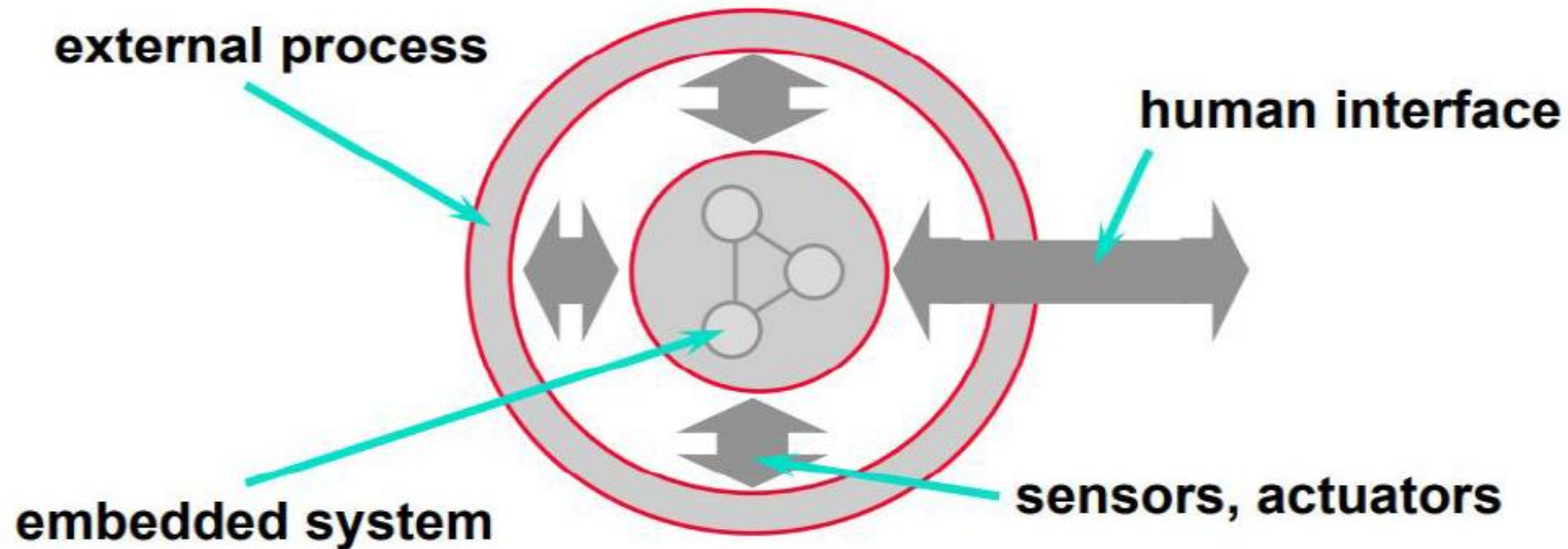


Figure: Embedded System

# Embedded System:

- An embedded system is a special-purpose system in which the computer is completely encapsulated by the device it controls.
- Unlike a general-purpose computer, such as a personal computer, an embedded system performs pre-defined tasks, usually with very specific requirements.
- Since the system is dedicated to a specific task, design engineers can optimize it, reducing the size and cost of the product.
- Embedded systems are often mass-produced, so the cost savings may be multiplied by millions of items
- Handheld computers or PDAs are generally considered embedded devices because of the nature of their hardware design, even though they are more expandable in software terms.
- This line of definition continues to blur as devices expand. 2

# Characteristics of Embedded System:

The main characteristics of embedded system are:

## **Single-functioned**

- Executes a single program, repeatedly

## **Tightly-constrained**

- Low cost, low power, small, fast, etc.

## **Real-time**

- Must compute certain results in real-time without delay.
- In Hard Real time any delay in deadline could result in a catastrophe.

# Characteristics of Embedded System: Contd...

## **Must be Dependable**

- Reliability, Maintainability, Availability, Safety, Security.

## **Reactive**

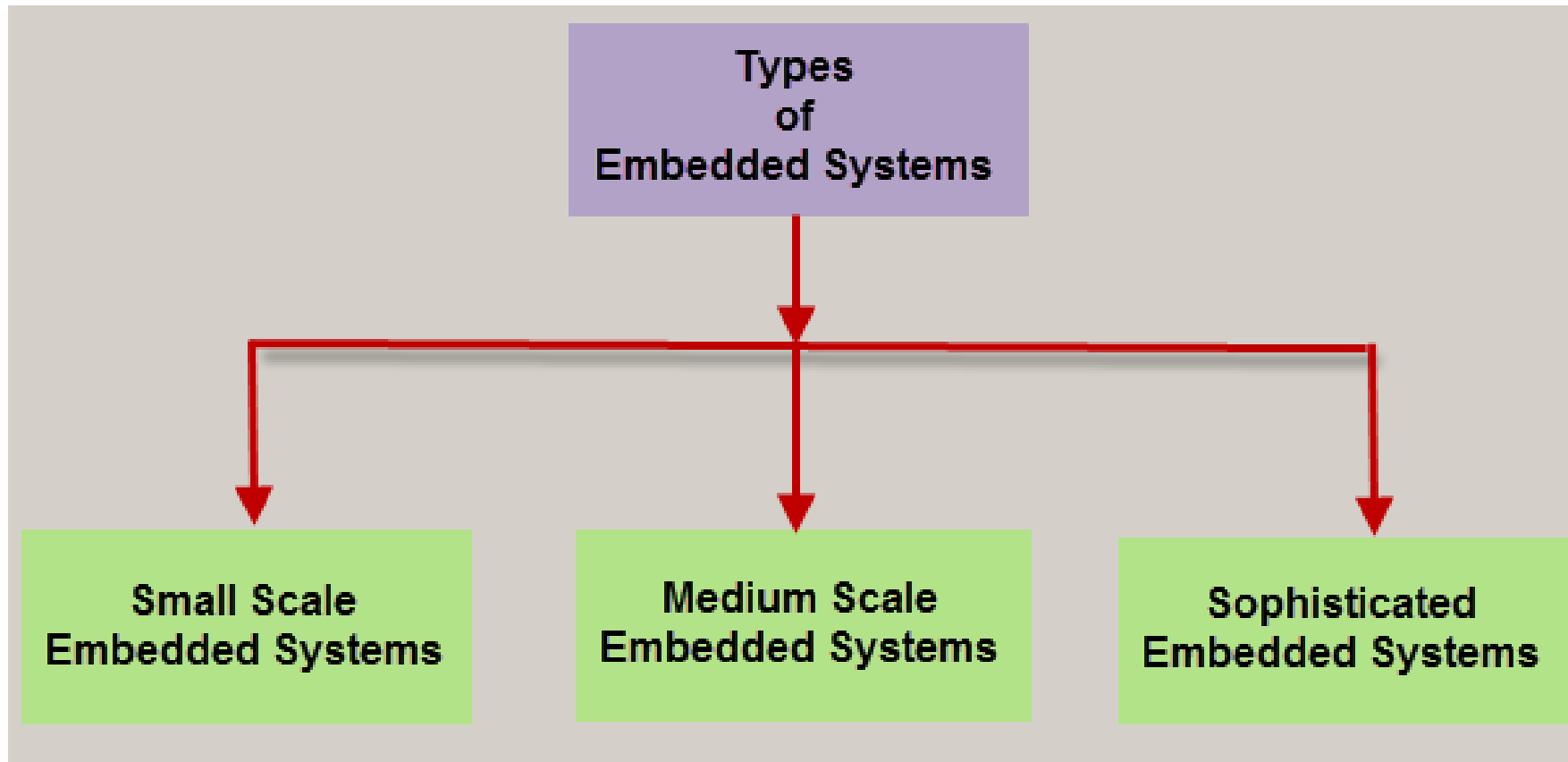
- Continually reacts to changes in the system's environment.

## **Sophisticated Functionality**

- Often have to run sophisticated or multiple algorithms for example cellphones, laser printer.

# Classification of Embedded System:

- Embedded systems are classified into three:



# Classification of Embedded System:

Embedded systems are classified into three:

## **Small Scale Embedded Systems:**

- Single 8-bit or 16-bit Microcontroller.
- Little Hardware and Software complexity.
- They may be battery operated.
- Usually C may be used to develop such system.
- Programming Tools:
  - Editor, Assembler and Cross Assembler.
- Example: Stepper motor controller for a robotics system, Washing system, Computer Mouse, Printer, Scanner, Remote controller etc....



# Classification of Embedded System:

## Medium Scale Embedded Systems:

- Single or few 16 or 32-bit microcontrollers or Digital Signal Processors(DSP) or Reduced Instruction Set Computers(RISC).
- Both Hardware and Software Complexity.
- Programming tools
  - RTOS, Source Code Engineering tool, Simulator, Debugger and Integrated Development Environment(IDE).
- Example: Banking systems like ATM and Credit card transactions, Entertainment systems like Video and Music system etc....

# Classification of Embedded System:

## Sophisticated ES:

- Enormous Hardware and Software complexity.
- May need scalable processor or configurable processor and programming logic arrays.
- Constrained by the processing speed available in their hardware units.
- Example: Real time video processing system, Speech or Multimedia processing system.
- Programming Tools
  - For these systems may not be readily available at reasonable cost or may not be available at all.
  - A compiler or retargetable compiler might have to be developed for this.

# Essential Components of ES

- Embeds hardware.
  - Processors, Timers, Interrupt Controller, I/O devices, Memories, Ports etc.
- Embeds main application software generally into flash or ROM and the application software performs concurrently the number of tasks.
- Embeds a real time operating system, which supervises the application software tasks running on the hardware and organizes the accesses to the system resources according to priorities and timing constraints of tasks in the system.

# Overview of Processors in ES

- General Purpose Processor
  - Microprocessor
    - Intel : 8085, 8086, 80186, 80188, 80286, 80386
    - Motorola: 6800, 6809, G3, G4, G5
- Microcontroller
  - Intel: 8032, 8051, 8052
  - PIC: 8-bit PIC 16, PIC18, 16-bit DSPIC33/PIC24
  - Motorola: 6811
- Embedded Processor
  - AndeScore N9/10/12, ARM 7/9/11, Intel i960, AMD 29050.
- Digital Signal Processor
  - PAC, TMS320CXX, SHARC, Motorola 5600XX.

# Overview of Processors in ES Contd...

- Application-Specific Processor:
  - Dedicated to specific tasks like image compression and provides faster solution.
  - Used as an additional processing unit for running the application in place of using embedded software.
  - Example: IIM7100, W3100A.
- Single-Purpose Processor
  - Digital circuit designed to execute only a single program.
  - Contains only the components to execute a single program.
  - No program memory.
  - Example: coprocessor, accelerator or peripheral.

# Hardware Units in Embedded System

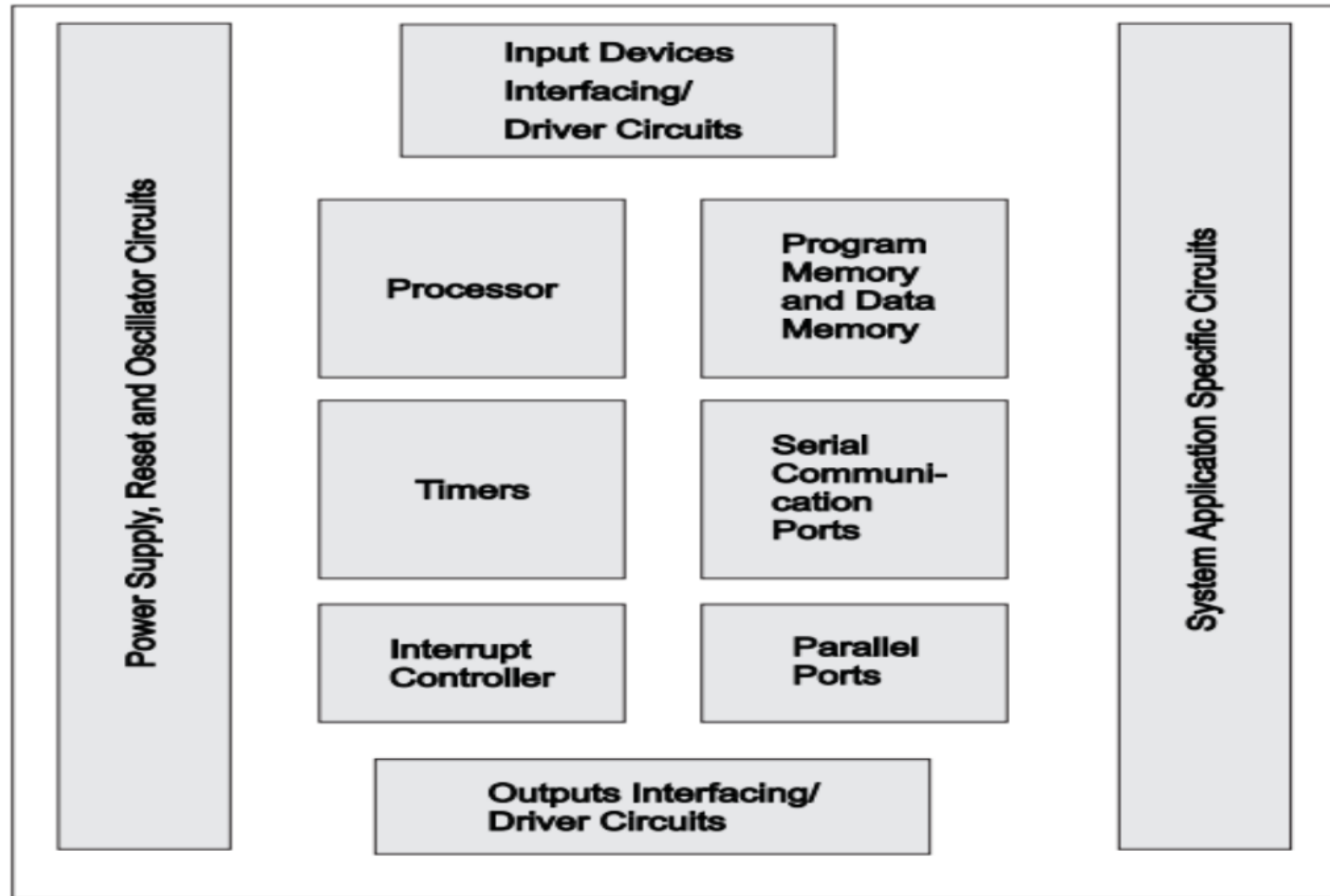


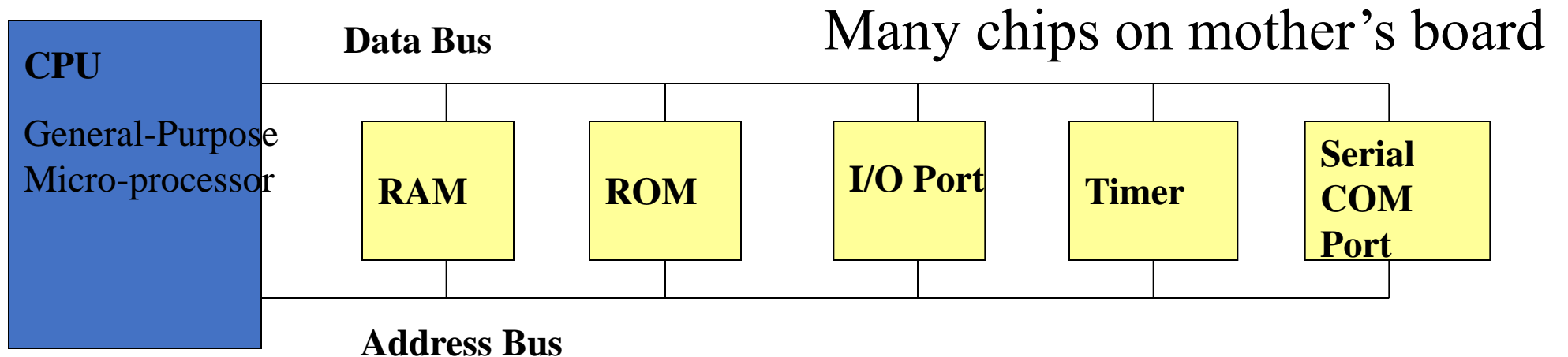
Figure *Hardware units in Embedded system*

# Application of Embedded system

- ✓ Automotive Electronics
- ✓ Aircraft Electronics
- ✓ Trains
- ✓ Telecommunication
- ✓ Medical Systems
- ✓ Military Applications
- ✓ Authentication Systems
- ✓ Consumer Electronics e.g MP3 Audio, digital Camera, home electronics etc.
- ✓ Robotics etc...

# General-purpose microprocessor

- CPU for Computers
- No RAM, ROM, I/O on CPU chip itself
- Example : Intel's x86, Motorola's 680x0

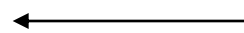
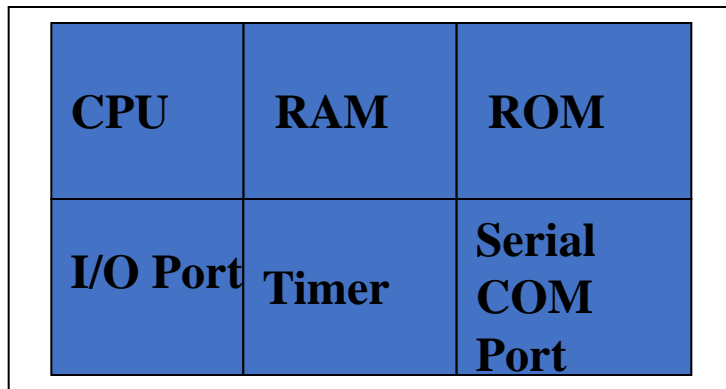


General-Purpose Microprocessor System



# Microcontroller

- A smaller computer
- On-chip RAM, ROM, I/O ports, timers etc
- Example : Motorola's 6811, **Intel's 8051**, Zilog's Z8 and PIC 16X



A single chip  
Microcontroller

# 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.
- expensive

## Microcontroller

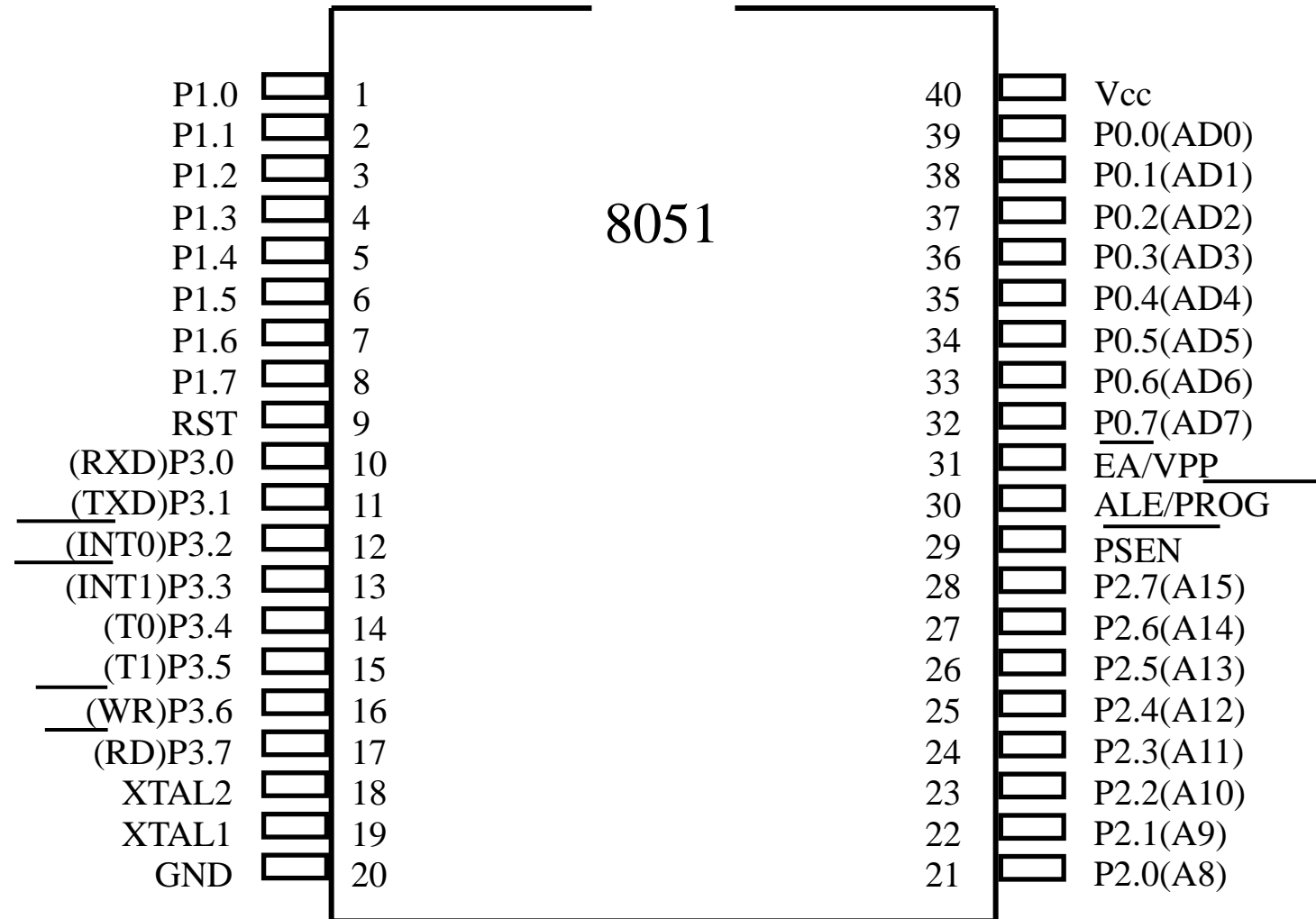
- CPU, RAM, ROM, I/O and timer are all on a single chip
- fix amount of on-chip ROM, RAM, I/O ports
- for applications in which cost, power and space are critical

# Comparison of the 8051 Family Members

## 3 FAMILY MEMBERS

Feature	8051	8052	8031
ROM (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

# Pin Description of the 8051



# Pin Description of the 8051

Stack Pointer (SP) – it contains the address of the data item on the top of the stack. Stack may reside anywhere on the internal RAM. On reset, SP is initialized to 07 so that the default stack will start from address 08 onwards.

- Data Pointer (DPTR) – DPH (Data pointer higher byte), DPL (Data pointer lower byte). This is a 16 bit register which is used to furnish address information for internal and external program memory and for external data memory.

- Program Counter (PC) – 16 bit PC contains the address of next instruction to be executed. On reset PC will set to 0000. After fetching every instruction PC will increment by one.

# Pin Description of the 8051

**Pins 1-8: PORT 1.** Each of these pins can be configured as an input or an output.

- **Pin 9 RESET.** A logic one on this pin disables the microcontroller and clears the contents of most registers. In other words, the positive voltage on this pin resets the microcontroller. By applying logic zero to this pin, the program starts execution from the beginning.

- **Pins 10-17 PORT 3.** Similar to port 1, each of these pins can serve as general input or output. Besides, all of them have alternative functions.

- **Pin 10 RXD.** Serial asynchronous communication input or Serial synchronous communication output.

# Pin Description of the 8051

- **Pin 11 TXD.** Serial asynchronous communication output or Serial synchronous communication clock output.
- **Pin 12 INT0.** External Interrupt 0 input.
- **Pin 13 INT1.** External Interrupt 1 input.
- **Pin 14 T0.** Counter 0 clock input.
- **Pin 15 T1.** Counter 1 clock input
- **Pin 16 WR.** Write to external (additional) RAM.
- **Pin 17 RD.** Read from external RAM
- **Pin 18, 19 XTAL2, XTAL1.** Internal oscillator input and output. A quartz crystal which specifies operating frequency is usually connected to these pins.
- **Pin 20 GND.** Ground.

# Pin Description of the 8051

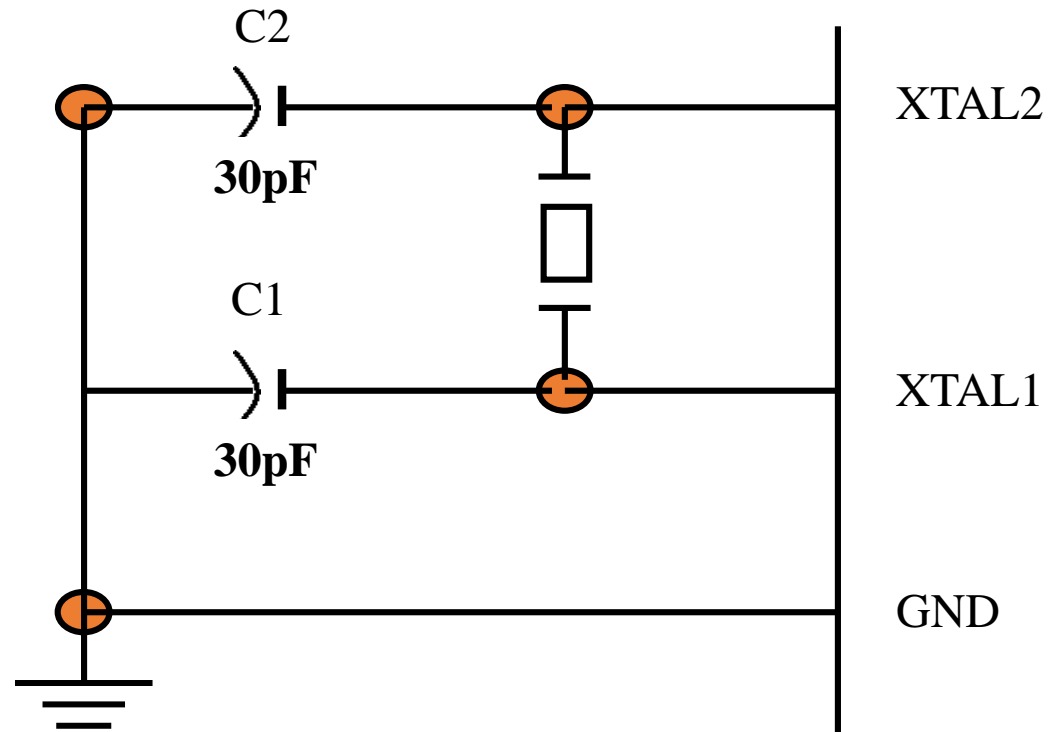
- **Pin 21-28 Port 2.** If there is no intention to use external memory then these port pins are configured as general inputs/outputs. In case external memory is used, the higher address byte, i.e. addresses A8-A15 will appear on this port. Even though memory with capacity of 64Kb is not used, which means that not all eight port bits are used for its addressing, the rest of them are not available as inputs/outputs.
- **Pin 29 PSEN.** If external ROM is used for storing program then a logic zero (0) appears on it every time the microcontroller reads a byte from memory.



# Pin Description of the 8051

- **Pin 30 ALE.** Prior to reading from external memory, the microcontroller puts the lower address byte (A0-A7) on P0 and activates the ALE output. After receiving signal from the ALE pin, the external latch latches the state of P0 and uses it as a memory chip address. Immediately after that, the ALE pin is returned its previous logic state and P0 is now used as a Data Bus.
- **Pin 32-39 PORT 0.** Similar to P2, if external memory is not used, these pins can be used as general inputs/outputs. Otherwise, P0 is configured as address output (A0-A7) when the ALE pin is driven high (1) or as data output (Data Bus) when the ALE pin is driven low (0).
- **Pin 40 VCC.** +5V power supply.

# Using a quartz crystal oscillator



# Registers

A
B
R0
R1
R2
R3
R4
R5
R6
R7

DPTR

DPH	DPL
-----	-----

PC

PC
----

Some 16-bit Register of 8051

Some 8-bit Registers  
of 8051

# Some Simple Instructions

**MOV dest,source** ; dest = source

MOV A,#72H ;A=72H

MOV R4,#62H ;R4=62H

MOV B,0F9H ;B=the content of F9'th byte of RAM

MOV DPTR,#7634H

MOV DPL,#34H

MOV DPH,#76H

MOV P1,A ;Move A to port 1

ADD A, Source                    ;A=A+SOURCE

ADD            A,#6                    ;A=A+6

ADD            A,R6                    ;A=A+R6

ADD            A,0F3H                ;A=A+[0F3H]

SUBB            A, Source            ;A=A-SOURCE

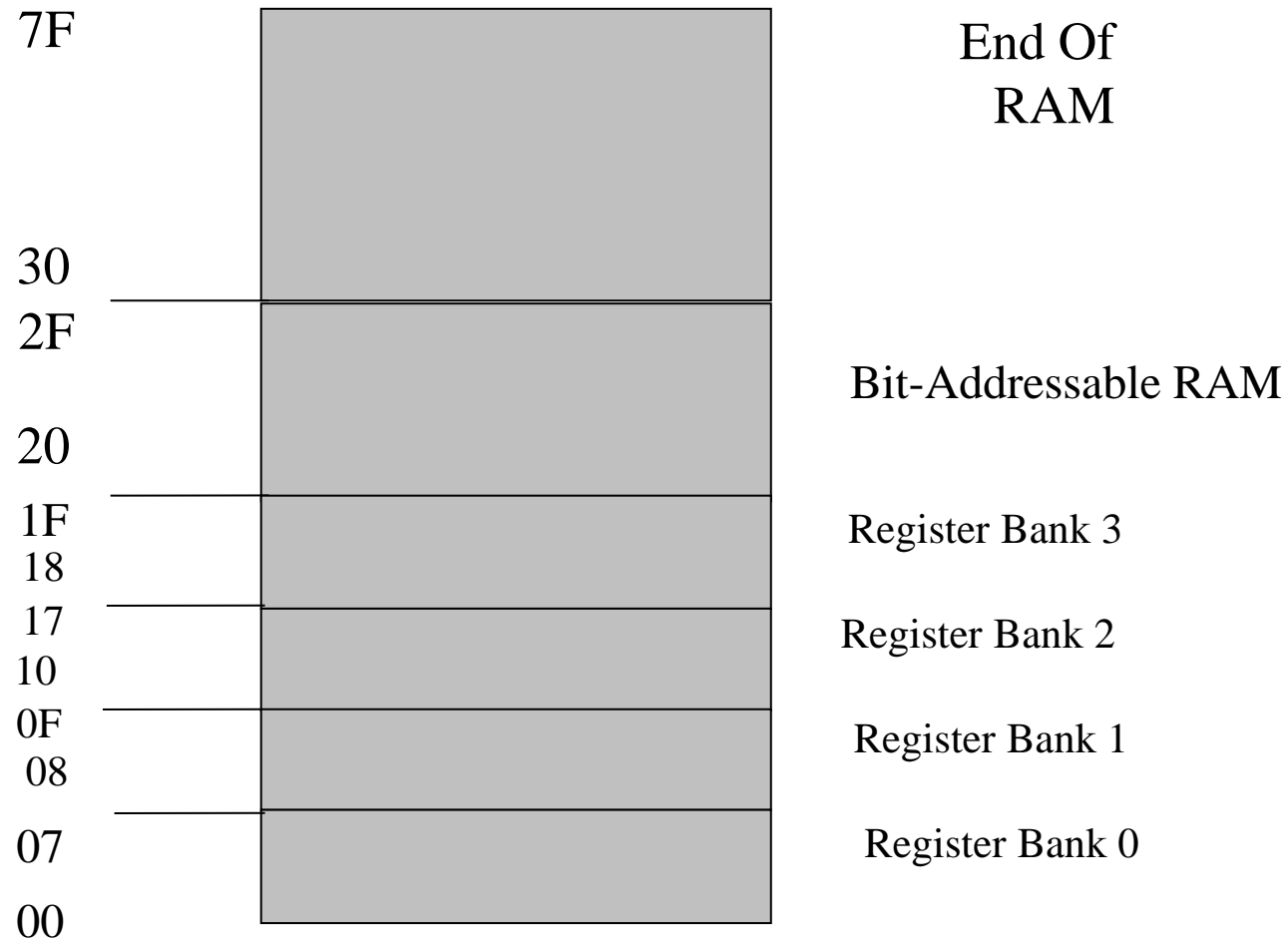
SUBB            A,#6                    ;A=A-6

SUBB            A,R6                    ;A=A-R6

▶ MUL      AB      ;A = A\*B  
MOV      A,#25H  
MOV      B,#65H  
MUL      AB                      ;25H\*65H=0E99  
                                    ;B=0EH, A=99H

▶ DIV      AB      ;A = A/B, B = A mod B  
MOV      A,#25  
MOV      B,#10  
DIV      AB                      ;A=2, B=5

# RAM memory space allocation in the 8051



# 8051 Flag bits and the PSW register

- Program Status Word (PSW) Register: An 8 bit register which contains the arithmetic status of ALU and the bank select bits of register banks.

CY	AC	F0	RS1	RS0	OV	--	P
----	----	----	-----	-----	----	----	---

<i>Carry flag</i>	PSW.7	<b>CY</b>
<i>Auxiliary carry flag</i>	PSW.6	<b>AC</b>
<i>Available to the user for general purpose</i>	PSW.5	<b>F0</b>
<i>Register Bank selector bit 1</i>	PSW.4	<b>RS1</b>
<i>Register Bank selector bit 0</i>	PSW.3	<b>RS0</b>
<i>Overflow flag</i>	PSW.2	<b>OV</b>
<i>User define bit</i>	PSW.1	<b>--</b>
<i>Parity flag Set/Reset odd/even parity</i>	PSW.0	<b>P</b>

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



# Addressing Modes

## 5 Addressing Modes

- Immediate
- Register
- Direct
- Register Indirect
- Indexed

# Immediate Addressing Mode

- ❑ Can be used to load information into any of the registers and memory location.
- ❑ Immediate data must be preceded by #.

MOV A,#65H

MOV R6,#65H

MOV DPTR,#2343H

MOV P1,#65H

# Register Addressing Mode

- ❑ Involves the use of registers to hold the data to be manipulated.

MOV R<sub>n</sub>, A            ;n=0,...,7

ADD            A, R<sub>n</sub>

MOV DPL, R6

# Direct Addressing Mode

❑ The data is in the RAM memory location & whose address is known.

MOV R0, 40H

MOV       56H, A

# Register Indirect Addressing Mode

✓ In this mode, register is used as a pointer to the data.

MOV A,@Ri           ; move content of RAM loc. Where address is held by  
Ri into A

MOV @R1,B

The registers that we can use in this mode be R0 and R1 only

# Indexed Addressing Mode And On-Chip ROM Access

- This mode is widely used in accessing data elements of look-up table entries located in the program (code) space ROM at the 8051

MOVC      A, @A+DPTR

A= content of address A +DPTR from ROM

## Note:

Because the data elements are stored in the program (code ) space ROM of the 8051, it uses the instruction MOVC instead of MOV. The “C” means code.

**THANKS**