CodeGuide

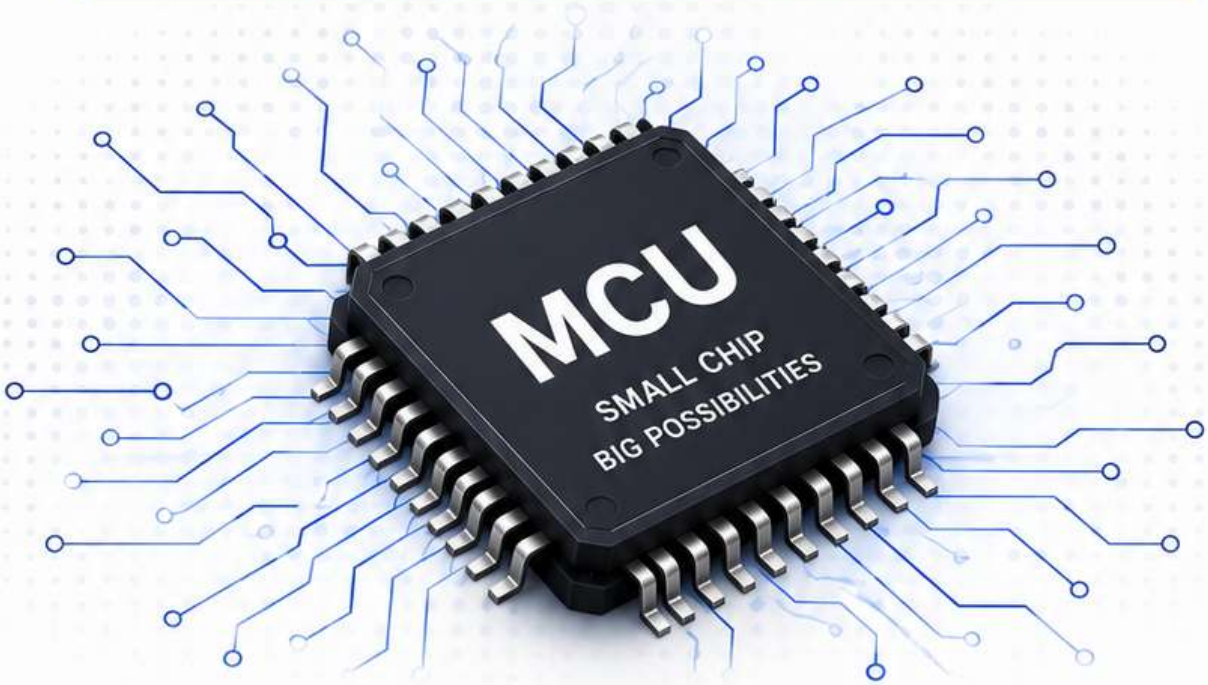
LEARN • BUILD • INNOVATE

QUICK NOTES FOR SMART LEARNING

EMBEDDED SYSTEM

MICROCONTROLLER NOTES

SIMPLE NOTES • CLEAR CONCEPTS • PRACTICAL APPROACH



CORE CONCEPTS
Understand the Fundamentals

ARCHITECTURE
Explore Internal Structure & Units

PROGRAMMING
Learn Efficient Embedded Coding

PERIPHERALS
Timers, ADC, UART, GPIO & More

APPLICATIONS
Real World Use in Embedded Systems

EXAM READY
Quick Revision & Practice

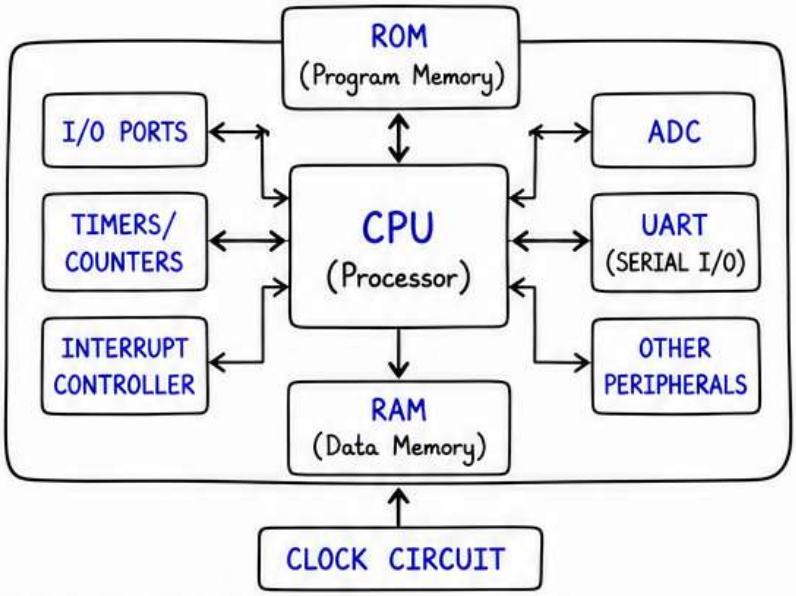
UNDERSTAND TODAY, DESIGN TOMORROW.

MICROCONTROLLER

1. WHAT IS MICROCONTROLLER?

- A microcontroller is a single chip computer.
- It contains CPU, Memory and Peripherals on the same chip.
- Used to control specific tasks in embedded systems.

2. MICROCONTROLLER BLOCK DIAGRAM



3. KEY FEATURES

- Single chip solution.
- Low power consumption.
- High speed and reliable.
- On-chip memory and I/O.
- Suitable for dedicated control applications.

4. MAIN COMPONENTS

- CPU : Executes instructions and controls operations.
- Memory : ROM for program, RAM for data.
- I/O Ports : Interface with external devices.
- Timers : Used for timing, counting and delays.
- ADC : Converts analog signals to digital.
- UART : Serial communication interface.
- Interrupt Controller : Handles interrupt requests.
- Clock Circuit : Provides system clock.

5. MICROCONTROLLER vs MICROPROCESSOR

FEATURE	MICROCONTROLLER	MICROPROCESSOR
Definition	Single chip computer	CPU only on a chip
Memory	On-chip ROM, RAM	External memory required
I/O Ports	On-chip I/O ports	External I/O required
Application	Dedicated control applications	General purpose applications
Cost & Size	Low cost, small size	High cost, larger size
Power Consumption	Low	High

6. KEY TAKEAWAY



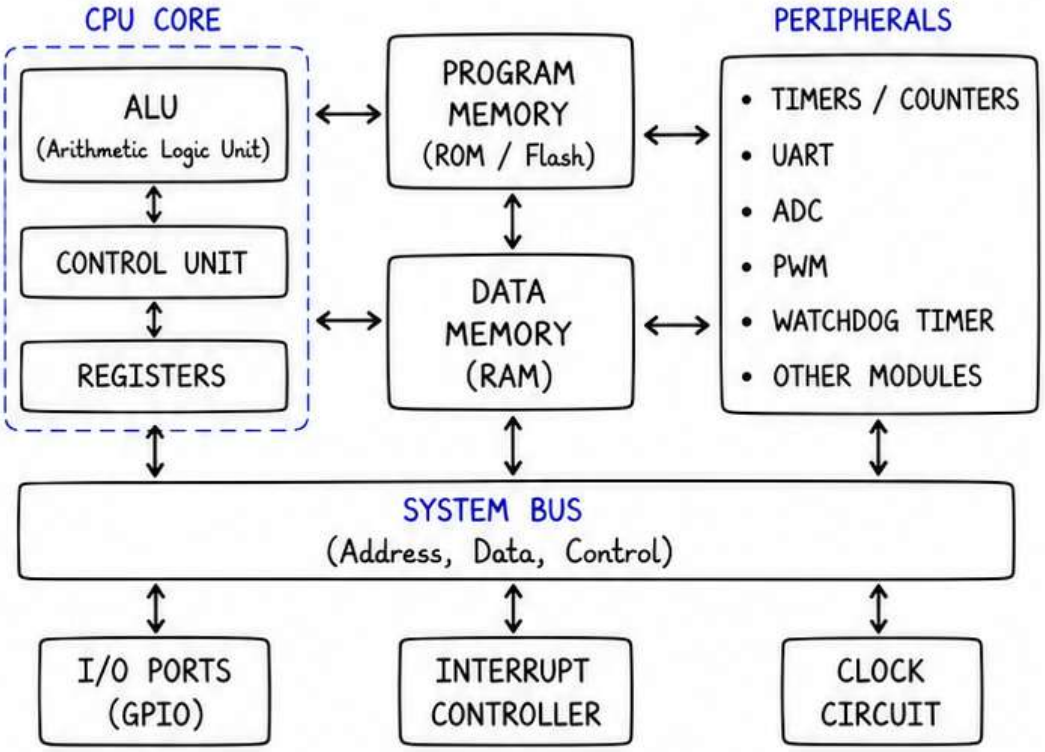
- ☑ Microcontroller is a single chip system with CPU, memory and peripherals.
- ☑ It is used in embedded systems for specific control tasks.
- ☑ Offers low cost, low power and high reliability.
- ☑ Essential components: CPU, Memory, I/O, Timers, ADC, UART, Interrupts, Clock.

MICROCONTROLLER ARCHITECTURE

1. OVERVIEW

- Microcontroller integrates CPU, memory, I/O and peripherals on a single chip.
- Designed for dedicated control applications.
- All components communicate over internal buses.

2. MICROCONTROLLER ARCHITECTURE (BLOCK DIAGRAM)



3. CPU CORE

- ALU : Performs arithmetic and logical operations.
- Control Unit : Fetches instructions, decodes and controls execution.
- Registers : Store data, addresses and status.

4. MEMORY

- Program Memory (ROM / Flash) : Stores instructions.
- Data Memory (RAM) : Stores variables and data during execution.

5. PERIPHERALS (ON-CHIP MODULES)

- Timers / Counters : Used for timing, counting and event generation.
- UART : Serial communication.
- ADC : Converts analog signals to digital.
- PWM : Generates pulse width modulated signals.
- Watchdog Timer : Resets system if program hangs.
- Other Modules : SPI, I^2C, Comparator, etc.

6. I/O PORTS (GPIO)

- Connect microcontroller with external devices.
- Each pin can be configured as:
  - Input
  - Output
  - Alternate Function
- Digital signals are read from or written to these pins.

7. INTERNAL BUS STRUCTURE

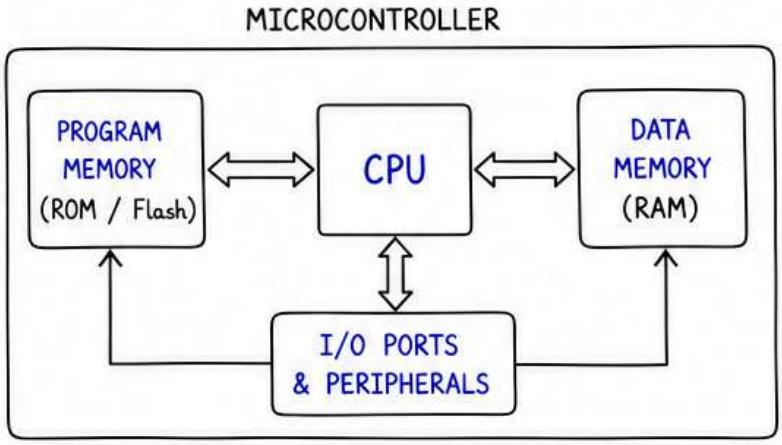
Address Bus	Carries address from CPU to select memory or I/O location.
Data Bus	Carries data between CPU, memory and peripherals.
Control Bus	Carries control signals (Read, Write, Interrupt, Reset, etc.).

MEMORY ORGANIZATION

1. WHAT IS MEMORY ORGANIZATION?

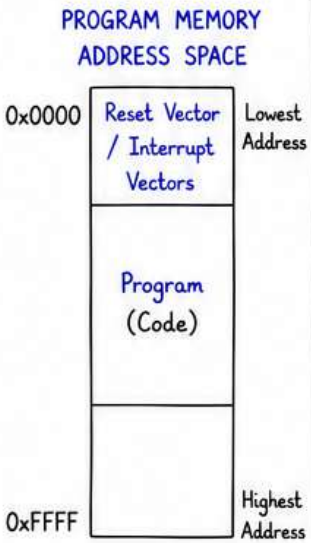
- Refers to the way memory is arranged and accessed by a microcontroller.
- Includes program memory, data memory and their address mapping.
- Proper organization allows efficient storage and fast data access.

2. MEMORY ORGANIZATION (GENERAL)



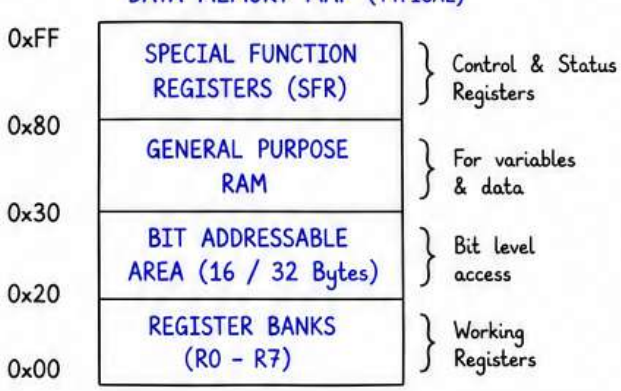
3. PROGRAM MEMORY

- Stores program / instructions.
- Usually non-volatile (ROM, Flash, EEPROM).
- Address space is separate from data memory in most MCUs.
- Contents are fixed or written infrequently.



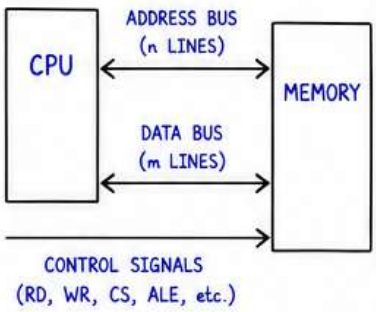
4. DATA MEMORY

- Stores data, variables, registers, stack.
- Usually volatile (RAM).
- Divided into different regions.



5. ADDRESSING IN MEMORY

- Microcontroller uses address bus to select memory location.
- If n address lines are used, total addressable locations = 2^n.
- Program and data memory may have separate address spaces.



6. TYPES OF MEMORY

TYPE	DESCRIPTION
ROM / MASK ROM	Program stored permanently.
PROM	Programmable once.
EPROM	UV erasable, reprogrammable.
EEPROM / Flash	Electrically erasable, reprogrammable.
RAM	Volatile memory for data.

7. KEY TAKEAWAY



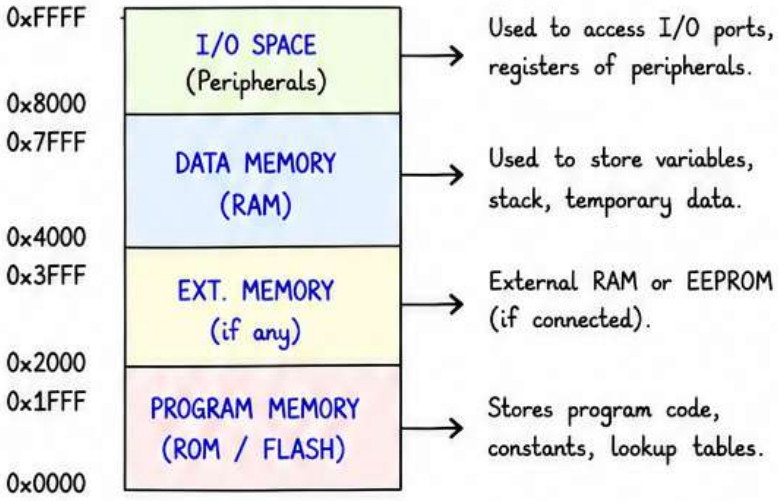
- ☑ Memory organization defines how program and data are stored and accessed.
- ☑ Program memory is non-volatile and stores instructions.
- ☑ Data memory (RAM) is volatile and used for variables, stack and registers.
- ☑ Addressing and mapping ensure efficient and organized memory access.

MEMORY MAP & I/O

1. MEMORY MAP

- Memory map shows how the address space of a microcontroller is divided.
- It defines where Program Memory, Data Memory and I/O registers are located.
- Helps CPU to access memory and peripherals correctly.

2. TYPICAL MEMORY MAP (Example)



3. I/O MAPPING METHODS

- I/O devices can be mapped in two ways.

1) MEMORY MAPPED I/O

- I/O registers are mapped in memory address space.
- Accessed using normal memory instructions (LD, ST).

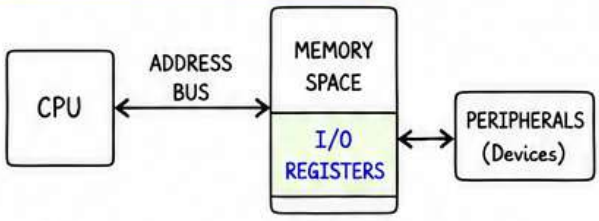
2) ISOLATED (PORT MAPPED) I/O

- I/O registers are in a separate I/O address space.
- Accessed using special I/O instructions (IN, OUT).

4. MEMORY MAPPED vs ISOLATED I/O

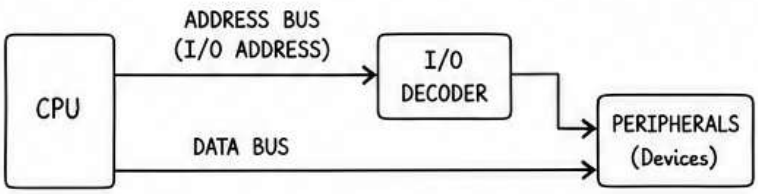
MEMORY MAPPED I/O	ISOLATED (PORT MAPPED) I/O
I/O uses memory address space.	I/O uses separate I/O address space.
Accessed using normal memory instructions.	Accessed using special I/O instructions.
Simple hardware.	Extra hardware (I/O decoder) required.
Less number of I/O addresses.	More number of I/O addresses.
Example: ARM Cortex-M	Example: 8085, 8051

5. MEMORY MAPPED I/O (Example)



- I/O registers occupy some addresses in memory space.
- CPU reads/writes I/O using normal memory instructions.

6. ISOLATED I/O (PORT MAPPED) (Example)



- I/O registers are in separate I/O address space.
- CPU uses IN/OUT instructions to access I/O.

KEY TAKEAWAY

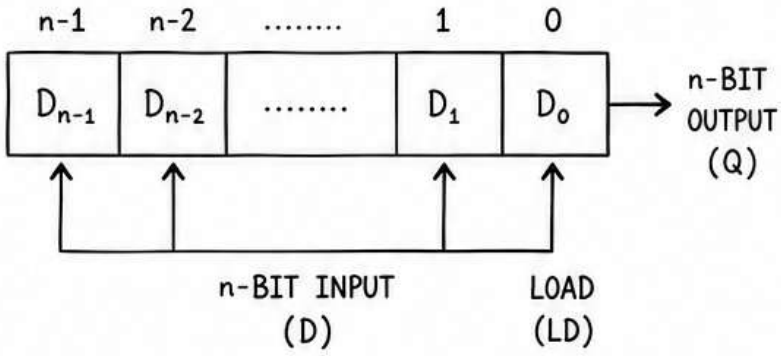
- ☑ Memory map shows the allocation of Program memory, Data memory and I/O space.
- ☑ I/O can be Memory Mapped or Isolated (Port Mapped).
- ☑ Memory mapped I/O uses memory instructions, isolated I/O uses special I/O instructions.
- ☑ Proper memory mapping ensures efficient access to memory and peripherals.

REGISTER AND REGISTER OPERATION

1. REGISTER

- A register is a small, high speed storage location inside CPU.
- Used to store data, addresses, instructions and status.
- Built using flip-flops.
- Key registers : ACC, PC, IR, MAR, MDR, SP, FLAGS.

2. REGISTER STRUCTURE (n-BIT)



3. TYPES OF REGISTERS

- General Purpose Registers (GPR) – For data storage and operations.
- Special Purpose Registers – Have dedicated functions.
- Flag Registers – Store status flags.

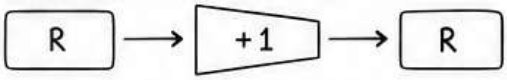
4. REGISTER OPERATIONS

- Data can be transferred, processed or shifted inside a register.
- Basic operations are performed by the ALU using register operands.

OPERATION	DESCRIPTION
LOAD (LD)	Transfer data from bus to register.
READ (RD)	Transfer data from register to bus.
CLEAR (CLR)	Set all bits of register to 0.
SET	Set all bits of register to 1.
INCREMENT (INC)	Increase register value by 1.
DECREMENT (DEC)	Decrease register value by 1.
SHIFT LEFT (SHL)	Shift all bits towards MSB.
SHIFT RIGHT (SHR)	Shift all bits towards LSB.

5. COMMON REGISTER OPERATIONS (EXAMPLES)

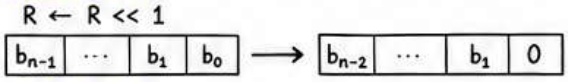
- Increment : $R \leftarrow R + 1$



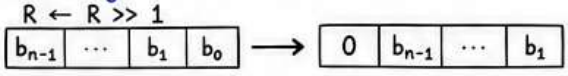
- Clear : $R \leftarrow 0$



- Shift Left : $R \leftarrow R << 1$



- Shift Right : $R \leftarrow R >> 1$



6. KEY TAKEAWAY



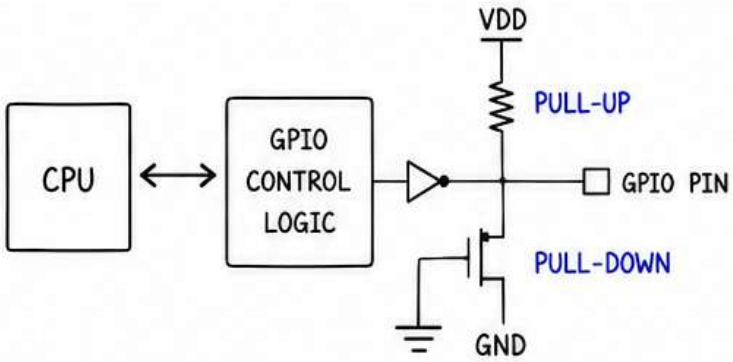
- ☑ Register is a fast storage location inside CPU.
- ☑ Registers hold data, addresses, instructions and status.
- ☑ Basic operations: Load, Read, Clear, Set, Increment, Decrement, Shift.
- ☑ Special purpose registers control the working of microcontroller.

GPIO (GENERAL PURPOSE INPUT/OUTPUT)

1. WHAT IS GPIO?

- GPIO pins are programmable pins of a microcontroller.
- Can be configured as Input or Output.
- Used to interface switches, LEDs, sensors, relays, etc.
- Controlled through GPIO registers.

2. GPIO PIN STRUCTURE (TYPICAL)



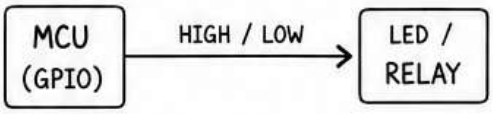
- Pull-up : Keeps pin HIGH when not driven.
- Pull-down : Keeps pin LOW when not driven.

3. GPIO MODES

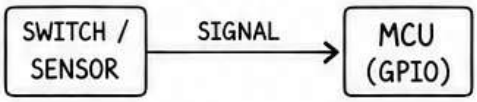
MODE	DESCRIPTION
INPUT	Pin reads external signal from device.
OUTPUT	Pin drives HIGH or LOW to external device.
ALTERNATE FUNCTION	Pin works as peripheral function (UART, SPI, I²C etc.).
ANALOG	Pin used as analog input (ADC).

4. GPIO OPERATION

- As OUTPUT
  - Write to Output Data Register (ODR)
    1 → HIGH, 0 → LOW

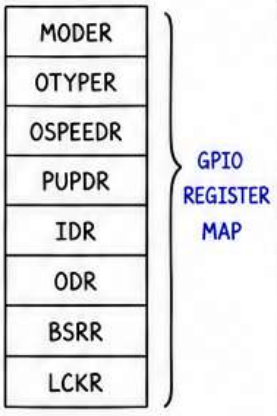


- As INPUT
  - Read from Input Data Register (IDR)
    Reads current logic level on the pin.



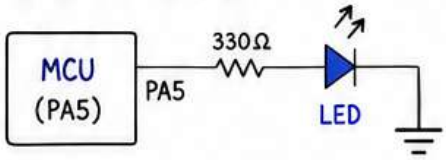
5. GPIO REGISTERS (TYPICAL)

- MODER : Mode register (Input/Output/Alt/Analog)
- ODR : Output Data Register (write data)
- IDR : Input Data Register (read data)
- BSRR : Bit Set/Reset Register (set or reset pin)
- PUPDR : Pull-up / Pull-down Register



6. EXAMPLE

- LED connected to PA5 pin.
- Configure PA5 as output.
- Write 1 → LED ON
  Write 0 → LED OFF



KEY TAKEAWAY

- ☑ GPIO pins are configurable as input or output.
- ☑ Used to communicate with external devices.
- ☑ Controlled through GPIO registers.
- ☑ Important in embedded system for real world interfacing.