UNIT 5

INTERFACING TECHNIQUES FOR MANAGEMENT OF EXTERNAL DEVICES

1.0 Introduction

Microcontrollers are the heart of modern mechatronic systems. However, a microcontroller alone cannot perform useful tasks unless it communicates with external devices such as memory chips, sensors, displays, motors, and data converters.

Interfacing is the process of connecting external hardware devices to a microcontroller so that information can be exchanged between them.

Examples of interfaced devices include:

- External memory
- Sensors
- ADCs (Analog-to-Digital Converters)
- DACs (Digital-to-Analog Converters)
- Keyboards
- LCD displays
- Seven-segment displays
- Stepper motors
- DC motors

This unit explains how these devices are connected and controlled by a microcontroller-based system.

2.0 Interfacing to External Memory

2.1 Meaning of External Memory

External memory refers to memory devices connected outside the microcontroller chip to increase storage capacity.

Examples:

- RAM (Random Access Memory)
- ROM (Read Only Memory)
- EEPROM

Why External Memory is Needed

Some applications require:

- Large program storage
- Large data storage
- Data logging
- Database storage

The internal memory of a microcontroller may not be sufficient.

2.2 External Memory Connection

External memory is connected through:

1. Address Bus

Used to select a memory location.

Example:

If address = 1000H, the memory location 1000H is selected.

2. Data Bus

Transfers data between memory and microcontroller.

3. Control Signals

Examples:

- RD (Read)
- WR (Write)
- CS (Chip Select)

2.3 Memory Read Operation

During reading:

1. Microcontroller sends address.
2. RD signal becomes active.

3. Memory places data on data bus.
4. Microcontroller receives data.

Example

Reading temperature data stored in RAM.

2.4 Memory Write Operation

During writing:

1. Microcontroller sends address.
2. Data is placed on data bus.
3. WR signal becomes active.
4. Data is stored in memory.

Example

Saving sensor readings in RAM.

3.0 Interfacing with 8255 Programmable Peripheral Interface (PPI)

3.1 What is 8255?

The 8255 is a programmable input/output interface chip used to expand the number of I/O ports.

It provides:

- Port A
- Port B
- Port C

Each port has 8 bits.

Therefore:

8 + 8 + 8 = 24 I/O lines

3.2 Need for 8255

Microcontrollers sometimes have limited I/O pins.

The 8255 helps connect:

- Keyboards
- LEDs
- Displays
- Relays
- Sensors

3.3 Internal Structure of 8255

The 8255 contains:

Port A

8-bit port

Port B

8-bit port

Port C

8-bit port

Control Register

Used to configure ports as:

- Input
- Output

3.4 Operating Modes of 8255

Mode 0

Simple Input/Output.

Example:

Reading switches and driving LEDs.

Mode 1

Input/Output with handshaking.

Used when synchronization is required.

Mode 2

Bidirectional communication.

Port A can transmit and receive data.

Applications of 8255

- Industrial control systems
- Traffic light systems
- Data acquisition systems
- Mechatronic automation systems

4.0 Interfacing with Sensors

4.1 What is a Sensor?

A sensor is a device that detects a physical quantity and converts it into an electrical signal.

Examples:

Sensor	Measures
Temperature Sensor	Temperature
Pressure Sensor	Pressure
Light Sensor	Light intensity
Proximity Sensor	Distance/Object presence
Humidity Sensor	Humidity

4.2 Sensor Interfacing Process

Step 1

Sensor detects physical quantity.

Step 2

Sensor produces electrical signal.

Step 3

Signal is sent to microcontroller.

Step 4

Microcontroller processes data.

Step 5

Action is taken.

Example: Temperature Control System

1. Sensor measures temperature.
2. Signal sent to microcontroller.
3. Microcontroller compares with set value.
4. Fan is switched ON if temperature is high.

4.3 Types of Sensor Outputs

Analog Output

Continuous signal.

Example:

0–5V temperature signal.

Digital Output

Only:

- 0
- 1

Example:

Motion detector.

5.0 ADC Interfacing

5.1 What is ADC?

ADC means:

Analog-to-Digital Converter

It converts analog signals into digital data that a microcontroller can understand.

Why ADC is Needed

Most sensors produce analog signals.

Microcontrollers process digital data.

Therefore conversion is necessary.

ADC Conversion Process

Sensor Output (Analog)

↓

ADC

↓

Digital Data

↓

Microcontroller

Example

Temperature Sensor Output:

2.5V

ADC converts it to:

10000000 (digital value)

Microcontroller processes the data.

5.2 Parallel ADC Interfacing

In parallel ADC:

All bits are transferred simultaneously.

Advantages

- Fast conversion
- High speed

Disadvantages

- Requires many wires
- More expensive

Applications:

- Industrial monitoring
- Data acquisition systems

5.3 Serial ADC Interfacing

In serial ADC:

Bits are transferred one after another.

Advantages

- Fewer wires
- Simple connection

Disadvantages

- Slower

Applications:

- Embedded systems
- Portable devices

6.0 DAC Interfacing

6.1 What is DAC?

DAC means:

Digital-to-Analog Converter

It converts digital data into analog signals.

Conversion

Digital Data

↓

DAC

↓

Analog Voltage

Why DAC is Needed

Many devices require analog signals.

Examples:

- Audio speakers
- Motor speed control
- Signal generators

Example

Digital Input:

11111111

DAC Output:

5V

Applications

- Audio systems
- Robotics
- Industrial control
- Process control systems

7.0 Keyboard Interfacing

7.1 What is a Keyboard?

A keyboard is an input device used to enter data into a microcontroller system.

Examples:

- Computer keyboard
- Matrix keypad
- Calculator keypad

7.2 Matrix Keyboard

Common arrangement:

4 × 4 Matrix

Contains:

- Rows
- Columns

Keys are identified by row-column intersections.

Working Principle

1. Microcontroller scans rows.
2. Columns are monitored.
3. Pressed key is detected.
4. Corresponding code is generated.

Applications

- ATM machines
- Security systems
- Calculators
- Industrial control panels

8.0 Seven-Segment Display Interfacing

8.1 What is a Seven-Segment Display?

A display made of seven LEDs arranged to form digits.

Segments:

a, b, c, d, e, f, g

Displayed Digits

Can display:

0–9

Example:

Digit 8 lights all segments.

Types

Common Cathode

All cathodes connected together.

Common Anode

All anodes connected together.

Applications

- Digital clocks
- Counters
- Calculators
- Measurement instruments

9.0 LCD Display Interfacing

9.1 What is LCD?

LCD stands for:

Liquid Crystal Display

Used to display:

- Text
- Numbers
- Symbols

Advantages of LCD

- Low power consumption
- Easy to read
- Lightweight
- Compact

Common LCD Pins

Pin	Function
VCC	Power
GND	Ground
RS	Register Select
RW	Read/Write
EN	Enable
D0–D7	Data Lines

LCD Working

1. Microcontroller sends command.
2. LCD receives command.
3. Data is displayed on screen.

Applications

- Industrial instruments
- Digital meters
- Embedded systems
- Medical equipment

10.0 Stepper Motor Interfacing

10.1 What is a Stepper Motor?

A stepper motor rotates in fixed angular steps.

Unlike DC motors, it moves step by step.

Characteristics

- Precise positioning
- High accuracy
- Easy control

Working Principle

Microcontroller sends pulses.

Each pulse causes one step movement.

Example:

200 steps/revolution

One step:

360° ÷ 200

= 1.8°

Applications

- CNC machines
- 3D printers
- Robotics
- Automated systems

11.0 DC Motor Interfacing

11.1 What is a DC Motor?

A DC motor converts electrical energy into rotational mechanical energy.

Why Interfacing is Needed

Microcontroller output current is very small.

A DC motor requires larger current.

Therefore a driver circuit is used.

Examples:

- L293D
- L298N

Working Principle

1. Microcontroller sends control signal.
2. Driver amplifies signal.
3. Motor rotates.

Direction Control

Forward Direction

Motor rotates clockwise.

Reverse Direction

Motor rotates anticlockwise.

Speed Control

Speed is controlled using:

PWM (Pulse Width Modulation)

Higher duty cycle → Higher speed

Lower duty cycle → Lower speed

Applications

- Electric vehicles
- Robots
- Conveyor belts
- Fans
- Pumps

12.0 Summary

In a mechatronic system, a microcontroller must communicate with external devices through interfacing techniques.

The major interfacing techniques studied are:

1. Interfacing to external memory.
2. Interfacing with 8255 PPI.

3. Interfacing with sensors.
4. Parallel and serial ADC interfacing.
5. DAC interfacing.
6. Keyboard interfacing.
7. Seven-segment display interfacing.
8. LCD display interfacing.
9. Stepper motor interfacing.
10. DC motor interfacing.

These interfacing methods allow a microcontroller to monitor, control, and automate real-world systems, making them essential in modern mechatronics and embedded system applications.
