Sub: Embedded System (KOE-062)

Notes: Unit-2

Topic

Embedded Networking: Introduction

Network: In a network, a group of computers and peripheral devices are connected. The smallest possible network is the one in which two computers are connected.
Networking: Networking is nothing but implementing tools and tasks for linking computers so that they can share resources over the network.

The embedded system was originally designed to work on a single device. However, in the current scenario, the implementation of different networking options has increased the overall performance of the embedded system in terms of economy as well as technical considerations.
The most efficient types of the network used in the embedded system are BUS network and an Ethernet network.
A BUS is used to connect different network devices and to transfer a huge range of data, for example, serial bus, I2C bus, CAN bus, etc.
The Ethernet type network works with the TCP/IP protocol.
Examples of embedded networking include CAN, I2C, Component, sensor, and serial bus networking.

Different types of networks generally have the following points in common:

- **Servers:** These are computers that provide the main information.
- **Clients:** These are computers or other devices that get access to shared resources.
- **Connection medium:** Connection medium defines the interlinking of different devices.
- **Shared data:** It refers to the information that is transmitted in a network and received by the clients.
- **Printers and other shared peripherals:** Peripherals (devices) that are connected to the client machines for processing and obtaining the information.

Types of Networks

There are different types of networks used in embedded systems, depending on the requirements of the application. Some of the commonly used networks in embedded systems are:

1. Local Area Network (LAN) - This type of network is used for devices that are located in close proximity to each other, typically within a single building or campus. LANs are used for communication between devices like computers, printers, and servers.

2. Wireless Sensor Network (WSN) - WSNs are used for applications that require communication between a large number of small devices over a wireless medium. These networks are commonly used in applications like home automation, industrial automation, and smart cities.

3. **Industrial Ethernet** - This type of network is used in industrial applications where high-speed communication is required between devices like programmable logic controllers (PLCs), human-machine interfaces (HMIs), and sensors. Industrial Ethernet provides high-speed communication and can operate in harsh environments.

4. **Cellular Networks** - Cellular networks are used in embedded systems that require communication over long distances. These networks are commonly used in applications like fleet management, remote monitoring, and surveillance.

Networking Protocols

Networking protocols are used to establish communication between devices in a network. There are different types of protocols used in embedded systems, depending on the network type and the application requirements. Some of the commonly used protocols are:

1. Transmission Control Protocol/Internet Protocol (**TCP/IP**) - TCP/IP is the most widely used protocol in the world and is used for communication between devices on the internet. TCP/IP provides reliable and secure communication between devices.

2. User Datagram Protocol (**UDP**) - UDP is a lightweight protocol that is used for applications that require low latency and do not require reliable communication. UDP is commonly used in applications like streaming media and gaming.

3. Hypertext Transfer Protocol (**HTTP**) - HTTP is used for communication between web servers and web clients. HTTP is used to transfer data between the server and the client and is used in applications like web browsing and e-commerce.

4. Message Queue Telemetry Transport (**MQTT**) - MQTT is a lightweight protocol that is used for communication between IoT devices. MQTT is commonly used in applications like home automation and smart cities.

Networking in Embedded Systems Design

Networking in embedded systems design requires careful consideration of factors like network type, communication protocol, and hardware design. Some of the important design considerations are:

1. **Power consumption** - Networking in embedded systems can consume a significant amount of power, which can be a challenge for battery-powered devices. Designers need to carefully balance the power consumption of the networking hardware and the application requirements.

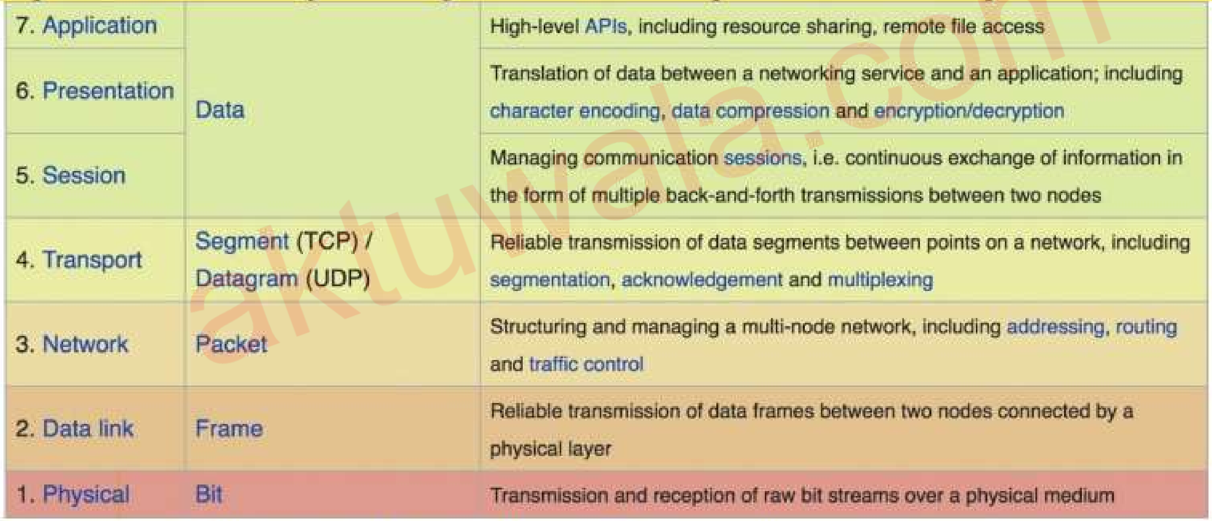
2. **Security** - Embedded systems are often used in applications where security is critical, like medical devices and industrial automation. Designers need to ensure that the networking hardware and protocols used provide adequate security.

3. **Real-time requirements** - Some applications, like industrial automation and control, have strict real-time requirements. Designers need to ensure that the networking hardware and protocols used can meet these requirements.

The OSI Model

Our discussion of embedded networking begins with an overview of computer networking systems and how they function. The earliest conceptual model of computer networks was developed by the International Organization for Standardization (ISO) in 1984 and is known as the Open System Interconnection (OSI) model.

The OSI model itself is conceptual in nature - it does not include any actual specifications for network implementation. However, the OSI model does provide a framework for understanding the components of a complete network communication system. As we will see, many of today's most commonly implemented networking technologies use features and protocols that reflect parts of the OSI model.



7. Application	Data	High-level APIs, including resource sharing, remote file access
6. Presentation		Translation of data between a networking service and an application; including character encoding, data compression and encryption/decryption
5. Session		Managing communication sessions, i.e. continuous exchange of information in the form of multiple back-and-forth transmissions between two nodes
4. Transport	Segment (TCP) / Datagram (UDP)	Reliable transmission of data segments between points on a network, including segmentation, acknowledgement and multiplexing
3. Network	Packet	Structuring and managing a multi-node network, including addressing, routing and traffic control
2. Data link	Frame	Reliable transmission of data frames between two nodes connected by a physical layer
1. Physical	Bit	Transmission and reception of raw bit streams over a physical medium

The OSI model defines seven-layer architecture for a complete communication system:

1.2. I/o Device Port & Buses:

A connection point that acts as interface between the computer and external devices like mouse, printer, modem, etc. is called **port**. Ports are of two types −

- **Internal port** − It connects the motherboard to internal devices like hard disk drive, CD drive, internal modem, etc.
- **External port** − It connects the motherboard to external devices like modem, mouse, printer, flash drives, etc.

1. SERIAL DATA COMMUNICATION - BASICS

Within a micro-computer system, the data transfer is in parallel because it is the fastest method. But transferring the data over long distances, the parallel data transmission requires too many wires and it is complicated and expensive. Therefore, the data to be sent for long distances is converted into serial form so that it can be sent on a single wire. At the destination, the received serial data is converted into parallel form so that it can be easily transferred on the micro-computer buses.

Methods of serial data transmission:

(i) Simplex :

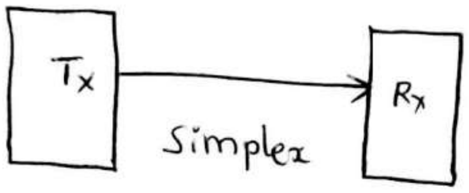
In this mode, the data is transmitted only in one direction over a single communication channel.

Ex: CPU to CRT display, Key board to CPU, Radio signal

(ii) Half-dupplex :

In this mode, the data is transmitted in both directions, but only one direction at a time. i.e., simultaneous data transfer is not possible

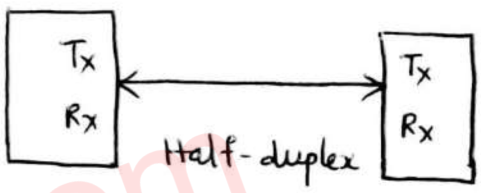
Ex: Walkie Talkie



(iii)Full-dupplex :

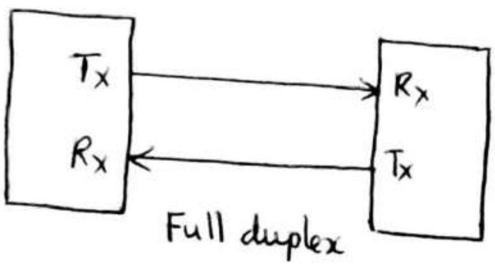
In this mode, the data transmission takes place in both directions simultaneously. It requires two channels.

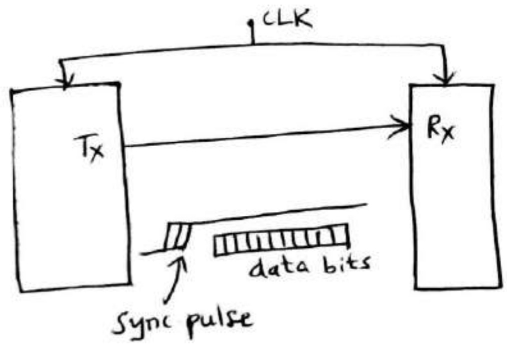
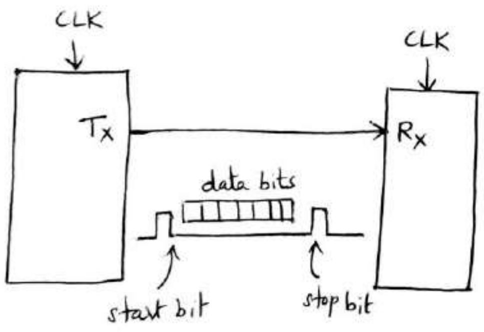
Ex: Telephone communication



Synchronous Vs Asynchronous data transfer modes

Synchronous data transmission	Asynchronous data transmission

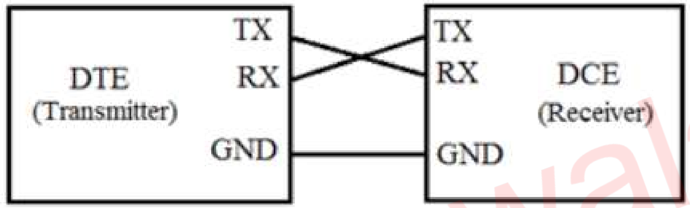


	 CLK CLK Tx Rx data bits start bit stop bit
Transmitter and Receiver are operated with same CLK frequency. SYNC pulses are required A group of characters can be transmitted after sending the SYNC pulses It is used in high speed data transmission Generally used between CPU and other devices on the same PCB, as the same power supply and CLK are used. Ex: SPI, I2C	Transmitter and Receiver can operated with different CLK frequency. START and STOP bits are required For each character, the START & STOP bits are required. It is used in low speed data transmission It is used to exchange data with other equipment such as PC. Ex: UART

What is RS232?

***RS232C "Recommended Standard 232C"** is the recent version of Standard 25 pin whereas, **RS232D** which is of 22 pins. In new PC's male D-type which is of 9 pins.*

RS232 is a standard protocol used for serial communication, it is used for connecting computer and its peripheral devices to allow serial data exchange between them. As it obtains the voltage for the path used for the data exchange between the devices. It is used in serial communication up to 50 feet with the rate of 1.492kbps. As EIA defines, the RS232 is used for connecting **Data Transmission Equipment (DTE)** and **Data Communication Equipment (DCE)**.



Universal Asynchronous Data Receiver &Transmitter (UART) used in connection with RS232 for transferring data between printer and computer. The microcontrollers are not able to handle such kind of voltage levels, connectors are connected between RS232 signals. These connectors are known as the **DB-9 Connector** as a serial port and they are of two type's **Male connector (DTE) & Female connector (DCE).**

Electrical Specifications

Let us discuss the electrical specifications of RS232 given below:

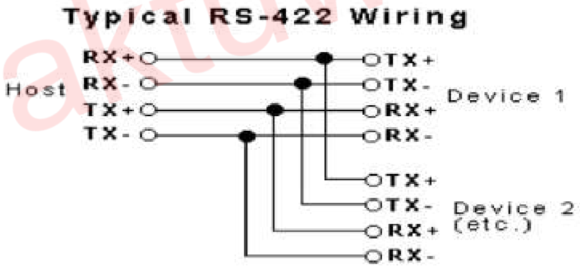
- **Voltage Levels:** RS232 also used as ground & 5V level. Binary 0 works with voltages up to +5V to +15Vdc. It is called as 'ON' or spacing (high voltage level) whereas Binary 1 works with voltages up to -5V to -15Vdc. It is called as 'OFF' or marking (low voltage level).
- **Received signal voltage level:** Binary 0 works on the received signal voltages up to +3V to +13 Vdc & Binary 1 works with voltages up to -3V to -13 Vdc.
- **Line Impedances:** The impedance of wires is up to 3 ohms to 7 ohms & the maximum cable length are 15 meters, but new maximum length in terms of capacitance per unit length.
- **Operation Voltage:** The operation voltage will be 250v AC max.
- **Current Rating:** The current rating will be 3 Amps max.
- **Dielectric withstanding voltage:** 1000 VAC min.
- **Slew Rate:** The rate of change of signal levels is termed as Slew Rate. With its slew rate is up to 30 V/microsecond and the maximum bitrate will be 20 kbps.

The ratings and specification changes with the change in equipment model.

What is RS 422

- is a telecommunications standard for binary serial communications between devices.
- is an updated version of the original serial protocol known as RS-232.
- One device will be known as the data terminal equipment (DTE) and the other device is known as data communications equipment (DCE).

- example is a serial link between the computer and printer, the computer is the DTE device and the printer is the DCE device.
- RS-422 is a balanced four wire system.
- Two wire is for DTE transmit signal to DCE, and other two wire is for DCE transmit signal to DTE.

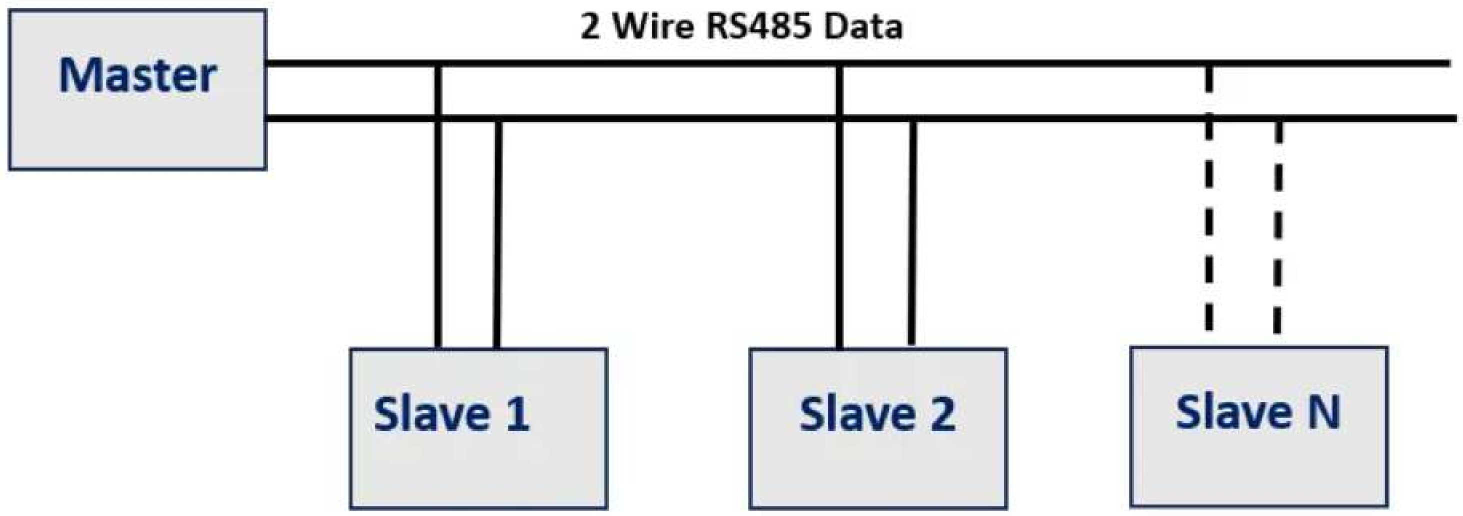


Specification of RS-422

Standard	EIA RS-422
Physical media	Twisted pair
Network topology	Point –to-point, multi-dropped
Maximum devices	10 (1 driver & 10 receivers)
Maximum distance	1200 meters (4000 feet)
Mode of operation	Differential
Maximum baud rate	100Kbps – 10Mbps
Voltage level	-6V to +6V (max voltage)
Mark(1)	Negative voltages
Space(0)	Positive voltages
Available signals	Tx+, Tx-, Rx+, Rx- (full duplex)
Connector types	Not specified, commonly screw terminals

What is RS485?

RS485 is a serial communication protocol. It is one of the most widely used communication protocols, especially in noisy industrial areas. Two devices can communicate with each other over a long distance (up to 1200m) using RS485 communication. Unlike RS232, RS485 has a 10 mbit/s data transfer rate. Lastly, it is master-slave communication so that each slave can communicate with the master. For each master, you can assign up to 32 slaves.



Why we need RS485 based Communication?

Firstly, in a large factory area, device-to-device communication is very challenging because the normal communication protocol can only handle a few meters; after that, data is destroyed and the data transfer rate is significantly reduced. To overcome this, RS485 is one of the best solutions because two or more devices can communicate up to 1200m with a 10 mbit/s high seed data transfer rate.

Secondly, an industrial area is a noisy area. The RS485 receiver decodes signals by comparing the voltage differences between its two wires. Although common-mode noise has limited influence on transmission due to the differential nature of the signal, it can still effectively corrupt data transmitted over the RS485 bus. Therefore, using RS485 communication, noise can be significantly reduced.

What is CAN Bus? An Overview

CAN Bus, short for Controller Area Network Bus, is a widely used communication protocol in the automotive and industrial automation industries. It is a network technology that allows various electronic devices and sensors to communicate with each other efficiently.

Understanding the Basics of CAN Bus

CAN Bus is a robust and reliable communication system that enables real-time data transmission between different components of a system. It was initially developed by Robert Bosch GmbH in the 1980s for in-vehicle communications.

Definition of CAN Bus

CAN Bus is an asynchronous serial communication protocol that uses a differential signalling scheme to transmit data over a twisted pair of wires. It employs a message-based communication model, where individual devices on the network can send and receive messages known as data frames.

History and Development of CAN Bus

The development of CAN Bus technology can be traced back to the 1980s when Robert Bosch GmbH recognized the need for a reliable communication protocol for automotive applications. It was initially used in cars to replace complex wiring harnesses and to provide a more flexible and efficient means of interconnecting various electronic control units (ECUs).

What is Serial Peripheral Interface (SPI)?

A Serial Peripheral Interface (SPI) facilitates short-distance communication between peripheral integrated circuits and microcontrollers. In this article, we will understand the components of SPI, applications of Serial Peripheral Interface (SPI), and more.

What is SPI?

SPI stands for Serial Peripheral Interface. It is a protocol that is synchronous serial communication. It is used to communicate between the peripheral devices i.e. input and output devices and microcontrollers. It is allowed to transfer high-speed data. It is popular with digital communication applications and embedded systems. SPI can transfer the data and receive data from one device to another device at a time.

Components of SPI

Serial Peripheral Interface (SPI) is the process of synchronous serial communication protocol. It is mainly used for connecting the microcontrollers to peripheral devices like sensors, displays, and memory chips. It facilitates the full-duplex, synchronous serial communication between one or more slave devices and a microcontroller.

Components of SPI

I2C Communication Protocol

I2C stands for **Inter-Integrated Circuit.** It is a bus interface connection protocol incorporated into devices for serial communication. It was originally designed by Philips Semiconductor in 1982. Recently, it is a widely used protocol for short-distance communication. It is also known as Two Wired Interface(TWI).

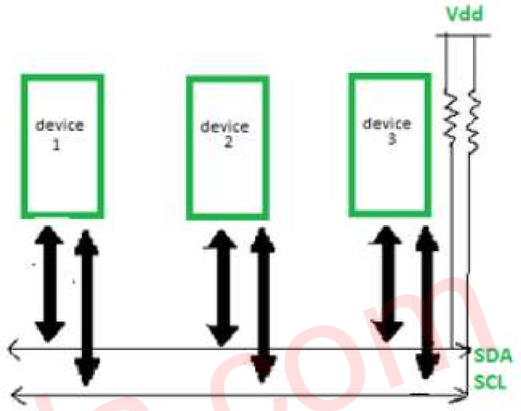
Working of I2C Communication Protocol :
It uses only 2 bi-directional open-drain lines for data communication called SDA and SCL. Both these lines are pulled high.

Serial Data (SDA) – Transfer of data takes place through this pin.
Serial Clock (SCL) – It carries the clock signal.

I2C operates in 2 modes –

- Master mode
- Slave mode

Each data bit transferred on SDA line is synchronized by a high to the low pulse of each clock on the SCL line.



According to I2C protocols, the data line can not change when the clock line is high, it can change only when the clock line is low. The 2 lines are open drain, hence a pull-up resistor is required so that the lines are high since the devices on the I2C bus are active low. The data is transmitted in the form of packets which comprises 9 bits. The sequence of these bits are –

1. **Start Condition** – 1 bit
2. **Slave Address** – 8 bit
3. **Acknowledge** – 1 bit

Advantages :

- Can be configured in multi-master mode.
- Complexity is reduced because it uses only 2 bi-directional lines (unlike SPI Communication).
- Cost-efficient.
- It uses ACK/NACK feature due to which it has improved error handling capabilities.

Limitations :

- Slower speed.
- Half-duplex communication is used in the I2C communication protocol.

Comparison between I2C and SPI Communication Protocols

Features	I2C Communication Protocol	SPI Communication Protocol
Number of wires	2 (SDA and SCL)	4 (MOSI, MISO, SCK, and SS)
Communication type	Half-duplex	Full-duplex
Maximum number of devices	Limited by addressing scheme	Limited by number of chip select (SS) lines
Data transfer speed	Slower	Faster
Error handling	Improved due to ACK/NACK feature	Not as robust
Cost	Cost-efficient due to fewer wires	More expensive due to additional wires
Complexity	Simpler due to fewer wires	More complex due to additional wires
Multi-master configuration	Yes	No
Synchronous communication	Yes	Yes