3

Some Protocols for Industrial Wired-Networks

Dr. H. K. Verma

Distinguished Professor (EEE)

Sharda University, Greater Noida

(Formerly: Deputy Director and Professor of Instrumentation

Indian Institute of Technology Roorkee)

website : profhkverma.info

CONTENTS

Network Classification and Requirements

A. Business Data Networks

- High data rates

B. Industrial Data Networks

- Enterprise-level network
  - High data rates
- Control-level network
  - Medium data rates
- Device-level network
  - Low data rates
  - Low latency
  - High data security
  - High network security
  - Low Power Consumption (for WSN)

Important Protocols for Industrial Wired-Networks

1. RS-422
2. RS-485
3. Modbus
4. Foundation Fieldbus
5. DNP (Distributed Network Protocol)
6. Ethernet
7. Ethernet/IP
8. HART (Highway Addressable Remote Transmitter)
9. CAN Bus (Controller Area Network)
10. Profibus (Process Field Bus)
11. LON (Local Operation Network)
12. BAC-Net (Building Automation Control Network)

Note: Protocols at serial numbers 1 to 5 and 8 are explained in these notes.

RS422 Serial Interface Protocol

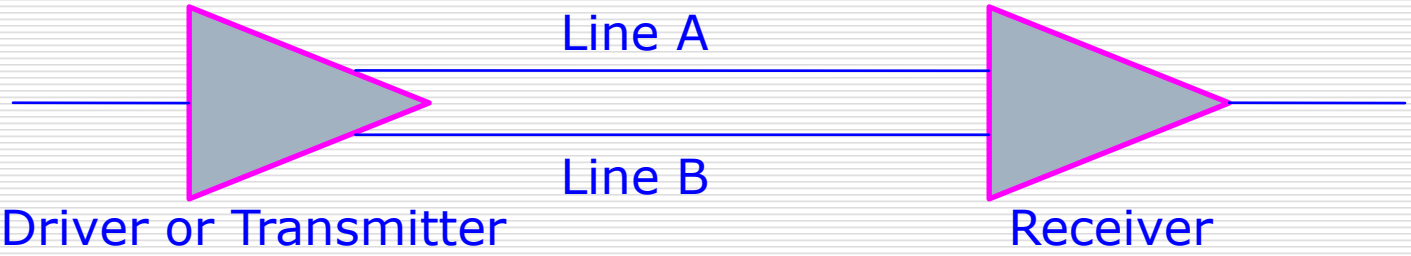
- RS-422 is “Recommended Standard for “Electrical Characteristics of Balanced-Voltage Digital-Interface Circuits”
- Published jointly by TIA (Telecommunications Industry Association) and EIA (Electronic Industries Alliance), so also known as TIA-422 or EIA-422 standard
- Specifies balanced mode of signals (differential signals) for transmission
- Connectors not specified
- Differential signals help nullifying:
  a) Effects of ground shifts
  b) Effects of common-mode noise signals induced on wires
- Consequently, superior performance as compared to single-ended signal transmission in terms of:
  - Longer distance transmission
  - Higher data rate transmission

Main Specifications

- Cable: 2 Unshielded twisted pairs (UTPs)
- Maximum data rate: 10 Mbps @ cable length upto 12m
- Maximum cable length: 1200m @ 100 kbps
- Specified for multi-drop (party-line) applications
- One driver transmits on a bus of upto 10 receivers
- Does not support a truly multi-point network (consisting of multiple drivers and multiple receivers on a single bus)
- Supports full-duplex communication only
- Supports "Master-Slave" mode of communication only

Balanced Mode Transmission

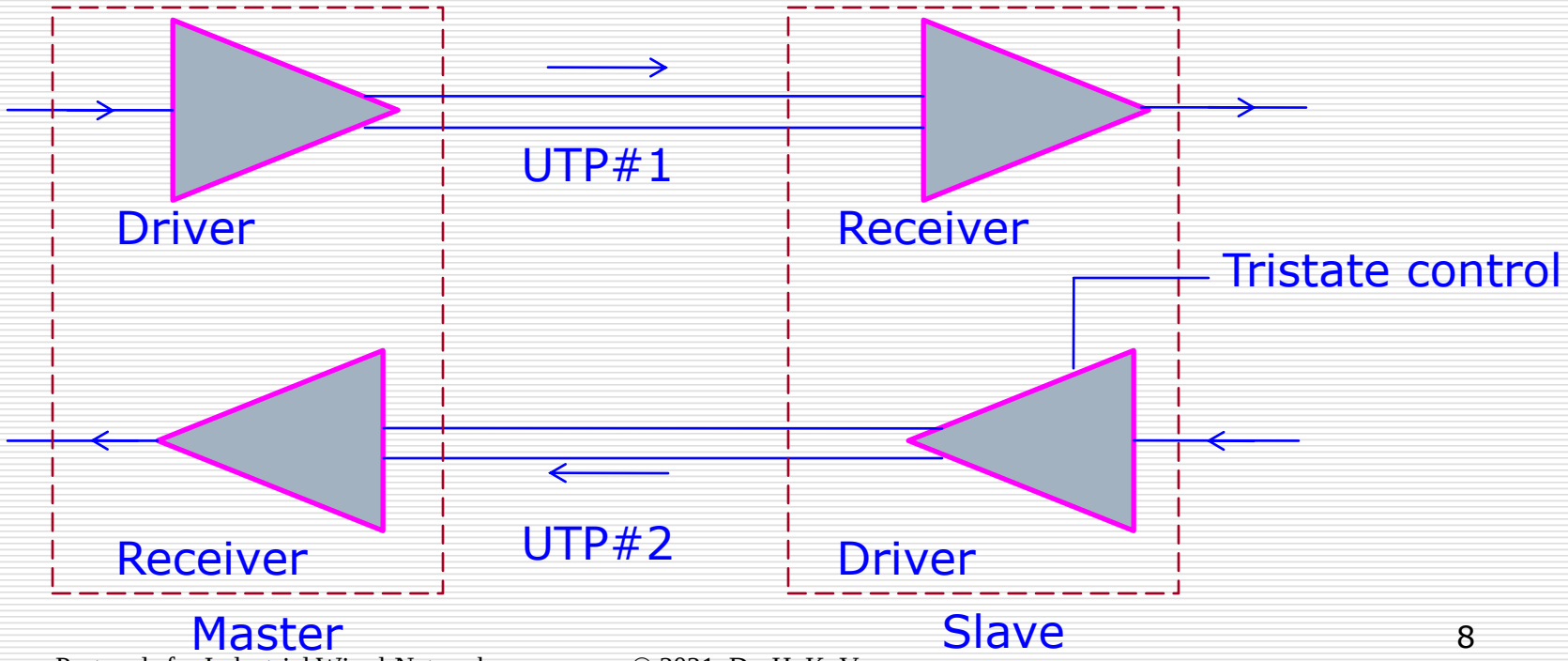
- Driver (transmitter) converts SE signal to differential signal
- Receiver translates differential signal back to SE signal
- CM noise induced in two lines is rejected by receiver
- Twisted pair provides transposition of lines A & B, which minimizes differential-mode noise.



- Signal Levels:
  - Logic 0 : Line B more positive than Line A
  - Logic 1 : Line A more positive than Line B
- Maximum differential voltage permitted on the lines: ±12V
- Maximum CM voltage permitted on the lines: ±7V
- Minimum output of driver: ±1.5V
- Minimum differential voltage detected by receiver: ±0.2V

Full-Duplex Transmission

- Two UTPs or 4 wires used
- Signal in either direction is transmitted over a separate pair
- Master always initiates dialogue on one UTP
- The addressed (polled) slave responds on the other UTP
- Driver of master is always enabled, hence needs no tri-state capability
- Drivers of slaves should have tri-state capability



RS-485 Field Network Protocol

- RS-485 is “Recommended Standard for Electrical Characteristics of Generators and Receivers for use in Balanced Digital Multipoint Systems”
- Published jointly by TIA (Telecommunications Industry Association) and EIA (Electronic Industries Alliance), so also known as TIA-485 or EIA-485
- Having support of both TIA and EIA, it is a de-facto standard communication protocol
- Specifies balanced mode of signal transmission
- Supports multi-drop or bus topology
- Very common protocol for distributed data acquisition on field networks/device-level networks.

Main Specifications

- The protocol specifies:
  (a) MAC Layer (Sub-layer of Data-Link Layer)
  (b) Physical Layer
- MAC protocol specified is Master-slave protocol.
- Physical Layer specifies the following:
  i. Signal transmission mode
  ii. Communication modes
  iii. Signal transmission medium
  iv. Data transmission rate
  v. Cable length
  vi. Data-signal relationship
  vii. Number of drivers and receivers

  Note: Connectors are not specified.

MAC Layer Details

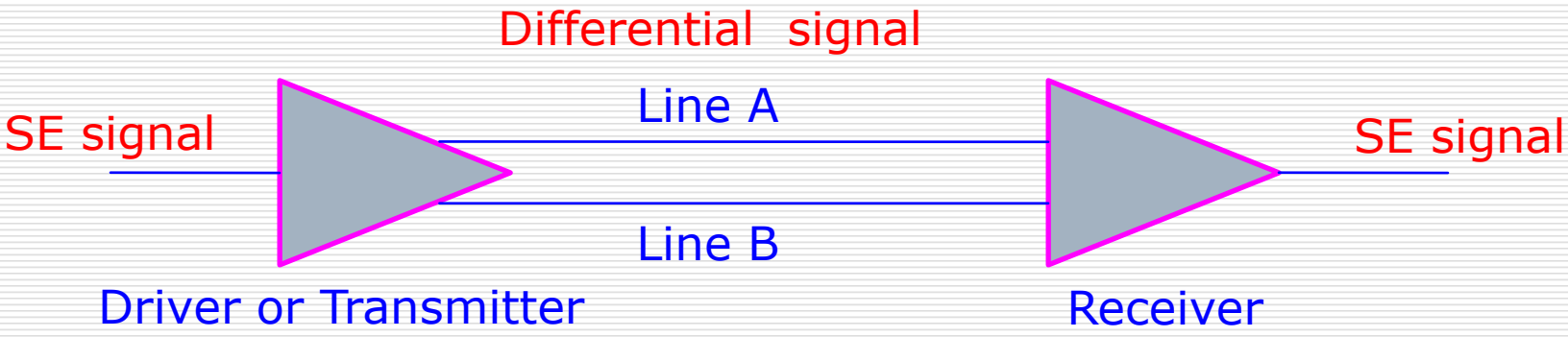
- MAC Protocol specified is Master-Slave protocol
- Master-slave protocol:
  a) It is the simplest MAC protocol
  b) Communication is always initiated by the master (that is, the master node)
  c) The master 'requests' a slave (that is, a slave node) through polling to send data
  d) The polled slave 'replies' by transmitting the required data
  e) Slaves never initiate a dialogue
  f) Slaves never communicate with each other
- Master-slave protocol eliminates any possibility of collisions on the data network.

Physical Layer Details

- **Signal transmission mode**: RS-485 supports balanced mode of transmission (differential signals are transmitted).
- **Communication modes**: Supports (allows) full-duplex as well as half-duplex mode of communication.
- **Signal transmission medium**:
  a) The protocol specifies UTP cable
  b) Two twisted pairs required for full-duplex transmission
  c) Single twisted pair required for half-duplex transmission
- **Data transmission rate and cable-length**:
  (a) Maximum data rate: 10 Mbps for cable lengths upto 12 m
  (b) Maximum cable length: 1200 m for data rates upto 100 kbps
- **Number of drivers and receivers**: Supports (allows) upto 32 drivers and 32 receivers on the network.
- **Data-signal relationship**: Explained in a later slide.

Balanced-Mode Transmission

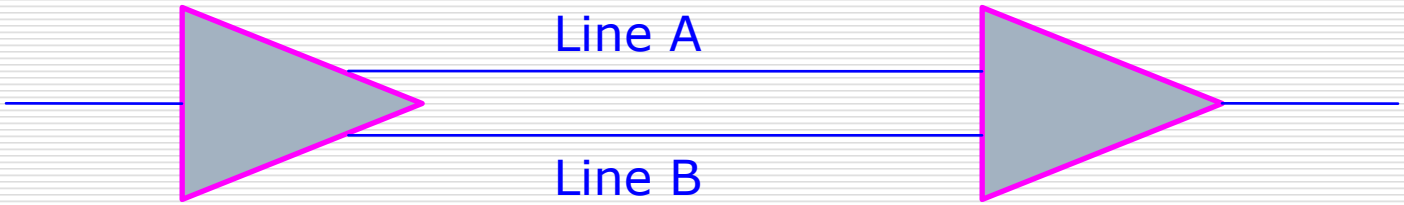
- Driver (transmitter) converts SE signal to differential signal
- Receiver translates differential signal back to SE signal.



- Lines A and B constitute one twisted pair of a UTP cable
- Twisted pair provides transposition of lines A & B, which minimizes differential-mode noise.
- Receiver has high sensitivity to differential-mode (DM) input but strongly rejects any common-mode (CM) input.
- Therefore, CM noise that may be induced in the twisted pair of lines would be strongly rejected by the receiver.

Data-Signal Relationship

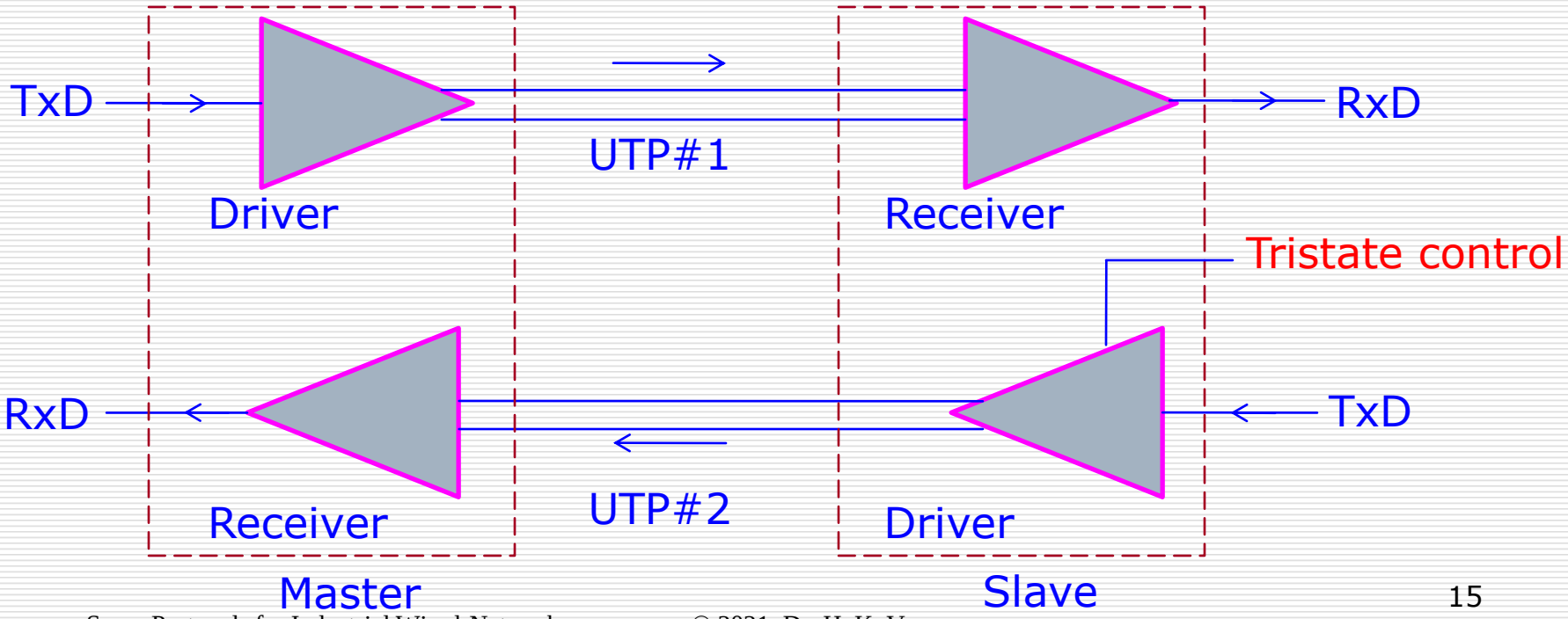
- Data-signal relationship:
  - Data logic 0 : Line B more positive than Line A
  - Data logic 1 : Line A more positive than Line B



- Maximum DM voltage permitted on the lines: ±12V
- Maximum CM voltage permitted on the lines: ±7V
- Minimum DM output of driver: ±1.5V
- Minimum DM voltage detected by receiver: ±0.2V
- This ensures sufficient margin for a reliable data transmission even if there is a severe degradation of signals across the cable and connectors.

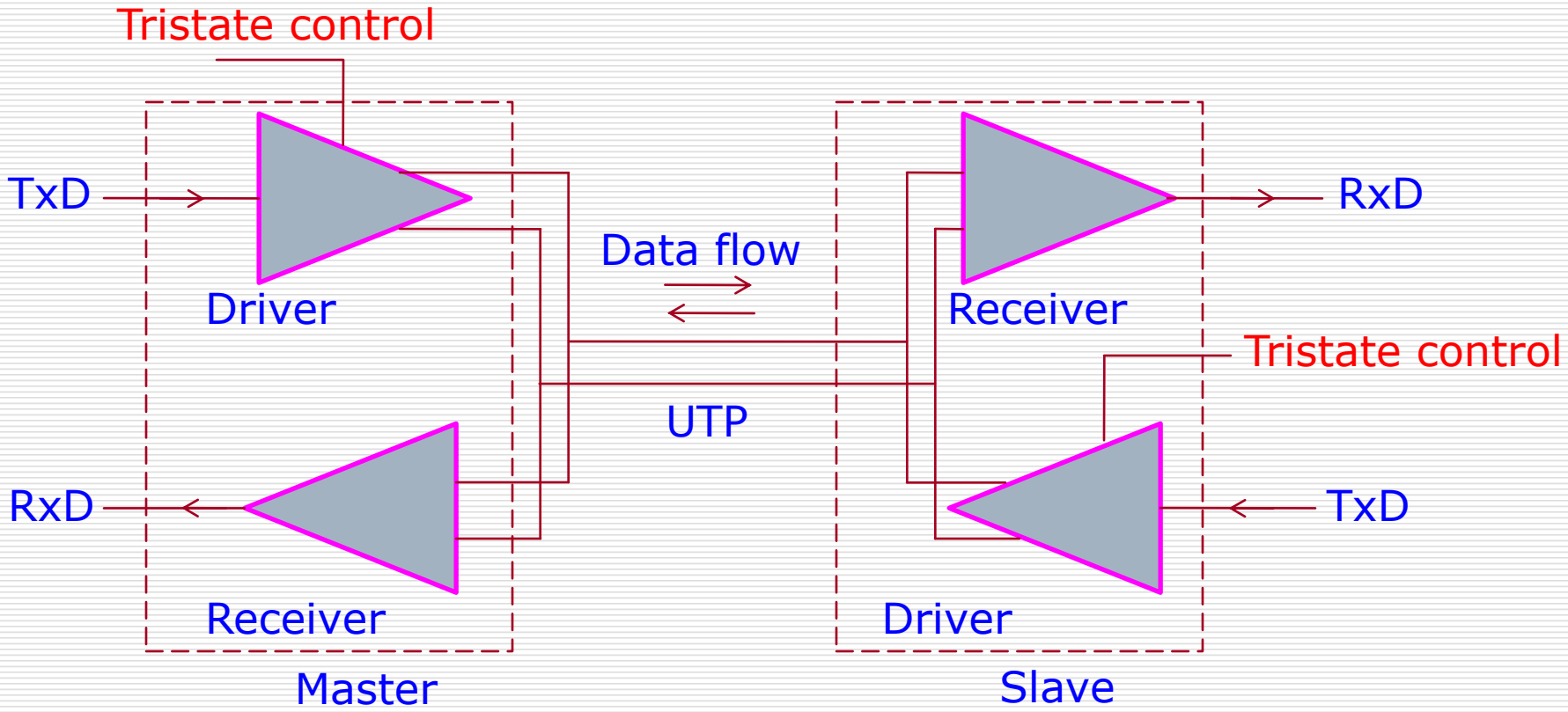
Transmission in Full-Duplex Mode

- Two UTPs or 4 wires used
- Signal in either direction is transmitted over a separate pair
- Master always initiates dialogue on one UTP
- The addressed (polled) slave responds on the other UTP
- Driver of master is always enabled, hence needs no tri-state capability.
- Drivers of slaves should have tri-state capability



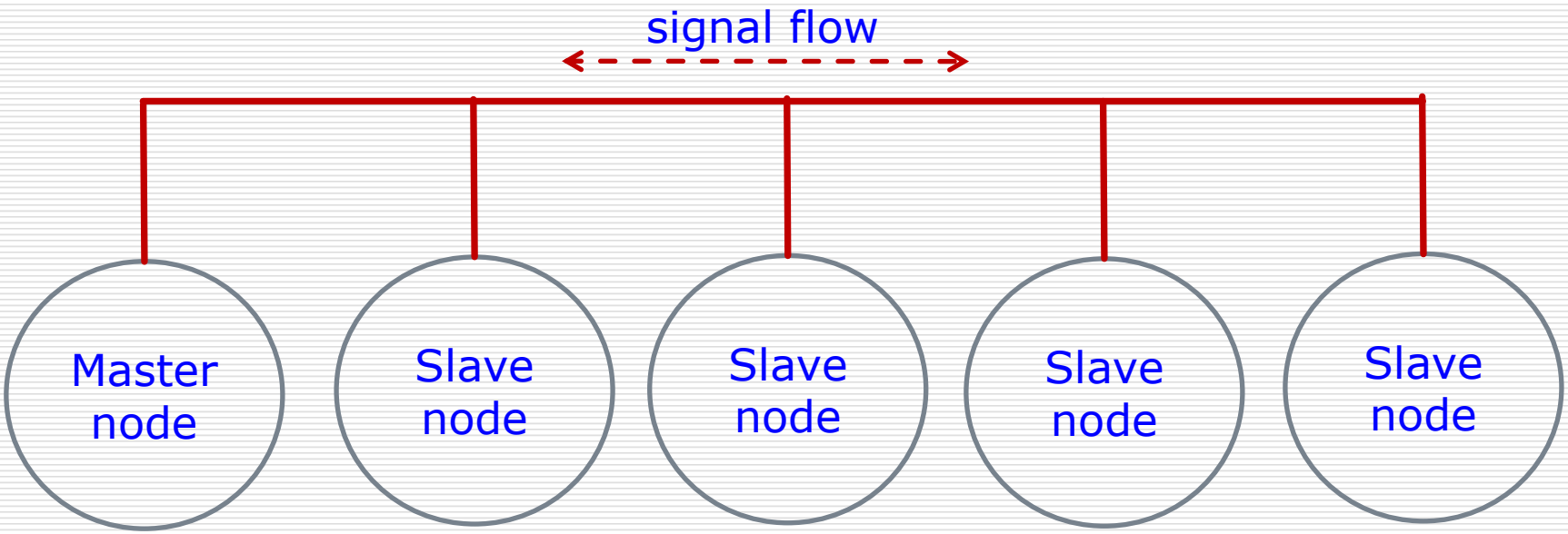
Transmission in Half-Duplex Mode

- Only one UTP or 2 wires are required.
- All drivers, including master, must have tri-state capability.



Basic Bus Structure

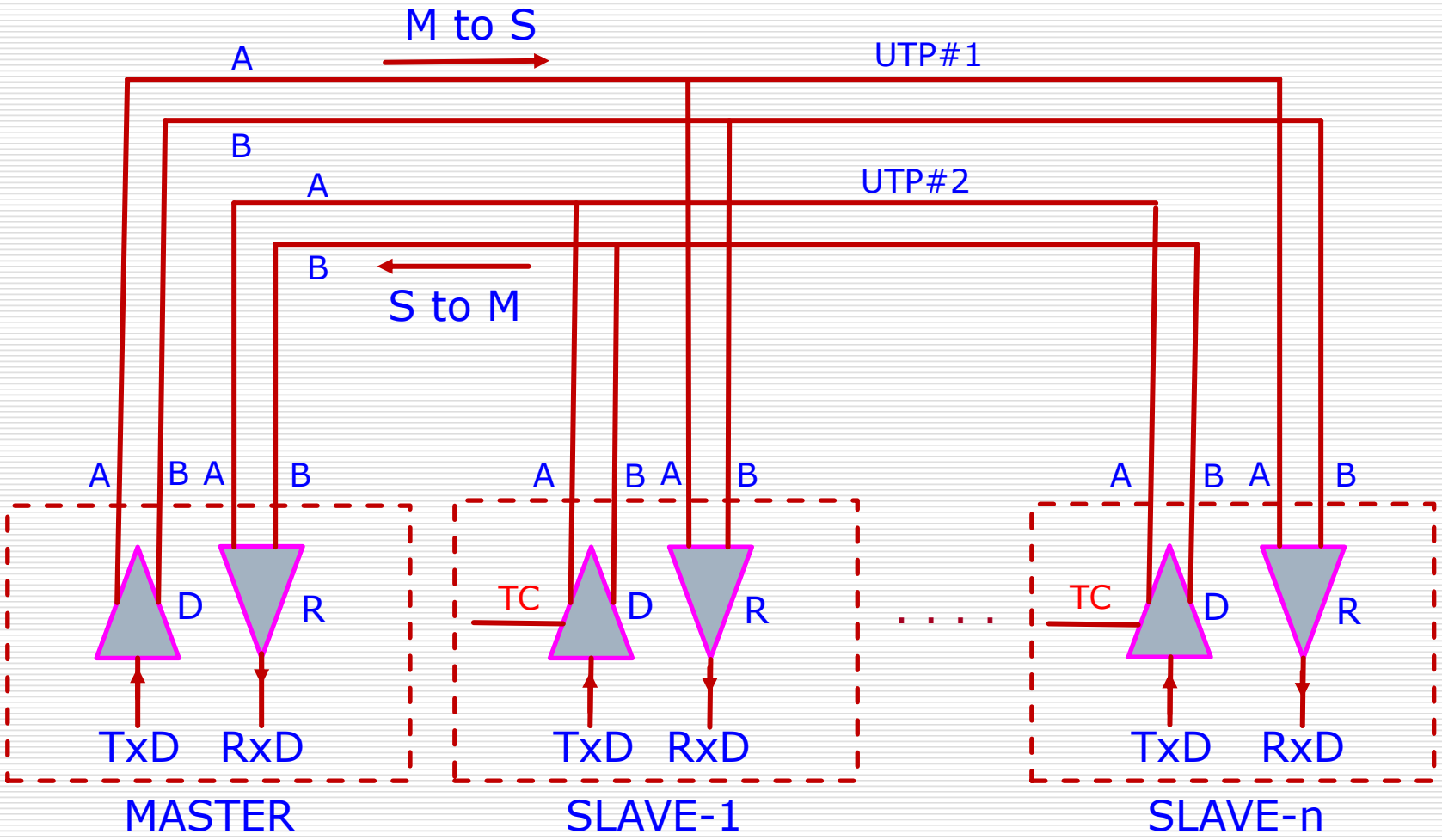
- Basic bus structure in RS-485 is shown below
- There is one master node and several slave nodes, all communicating through a bus
- In full-duplex mode of communication, bus comprises 2 UTPs
- In half-duplex mode, bus comprises just a single UTP.



Full-Duplex Bus Structure

- Full-duplex bus structure in RS-485 is shown in next slide.
- The bus comprises two UTPs
- UTP-1 transmits signals from the transmitter of master node to receivers of all slave nodes.
- UTP-2 transmits signals from the transmitters of slave nodes to the receiver of the master node.

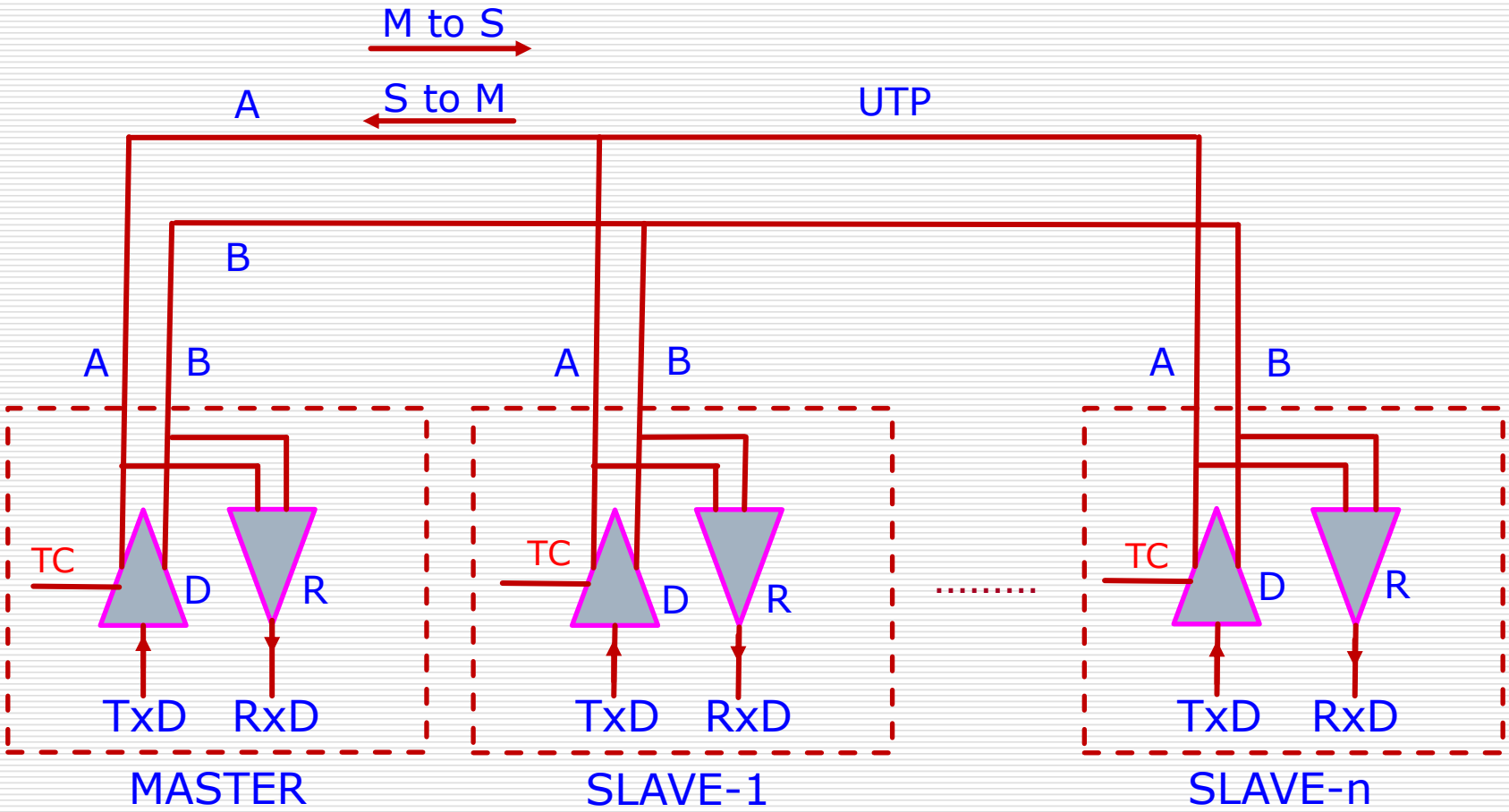
Full-Duplex Bus Structure



Half-Duplex Bus Structure

- Half-duplex bus structure in RS-485 is shown in next slide
- The bus comprises a single UTP
- At one time, the UTP transmits signals from the transmitter of master node to receivers of all slave nodes.
- **Later on**, the same UTP transmits signals from the transmitter of a slave node to the receiver of the master node.

Half-Duplex Bus Structure



D – Driver or Transmitter R – Receiver TC- Tristate Control

Merits of RS485 Protocol

1. Only one UTP may be used for signal transmission.
2. Daisy-chain wiring can be used for connecting nodes, thereby avoiding the need of taps or stubs.
3. Master-slave protocol of MAC is fully secured against any collisions on the network.
4. No restriction on the type of the connector to be used.
5. Use of repeaters every 1.2 km length of cable allows extension of the cable as much as necessary.

Limitations of RS485 Protocol

1. The protocol cannot be used for networking of mobile nodes.
2. The protocol cannot be used all alone as it does not specify Application Layer.
3. The protocol supports only bus topology.
4. The number of the nodes is limited to 32 only.
5. As the cable length is increased, the data rate needs to be reduced in the same proportion for a satisfactory performance of the network. [The product of cable length in km and data rate in kbps should not exceed 120]

Modbus Protocol

- Data communication protocol suitable for field-device networks
- Developed and introduced in 1978 by AEG Modicon (which is now Schneider Electric) for use with its PLCs.
- In 2004, Schneider Electric transferred its rights and responsibilities on Modbus protocol to Modbus Organization
- Modbus Organization is an association of the users and suppliers of Modbus-compliant components.
- Since 2004, Modbus Organization has been developing and promoting Modbus protocol.
- It is now a de-facto industry standard for field level networks.
- ***It is now an open data communication protocol.***
- Application Layer of Modbus is fully specified, while master-slave protocol is specified for MAC (medium access control).
- So, Modbus needs to be used in conjunction with other network protocols to take care of physical and other layers, as necessary.

Modbus Object Types

- Objects are the **entities within a Modbus slave** participating in monitoring and/or control.
- Four **prominent types** of objects are specified in Modbus:

S. No.	Object Type	Access	Size	Address Space
1	Discrete input or contact	Read only	1 bit	10001 - 19999
2	Discrete output or coil	Read-write	1 bit	00001 - 09999
3	Input register	Read only	16 bits	30001 – 39999
4	Holding register	Read-write	16 bits	40001 - 49999

Modbus Functions & their Codes

- Examples of some **commonly used** Modbus functions along with their codes are given below:

Function	Function Code
Read multiple discrete outputs or coils	01
Read multiple discrete inputs or contacts	02
Read multiple holding registers	03
Read multiple input registers	04
Write single discrete output or coil	05
Write single holding register	06
Write multiple discrete outputs or coils	15
Write multiple holding registers	16

Modbus Variants

- Depending on the Physical Layer and MAC protocol used along with the Application Layer of Modbus protocol, it has by now several variants.
- **Open protocols**: Important Modbus variants available as open protocols, and discussed in detail in these notes, are:
  - ***i. Modbus over serial line***
  - ***ii. Modbus TCP/IP or Modbus TCP***
- **Proprietary protocols**: Some examples of Modbus variants available as proprietary protocols are listed below, but not discussed in these notes:
  - i. Modbus Plus
  - ii. Pemex Modbus
  - iii. Enrion Modbus

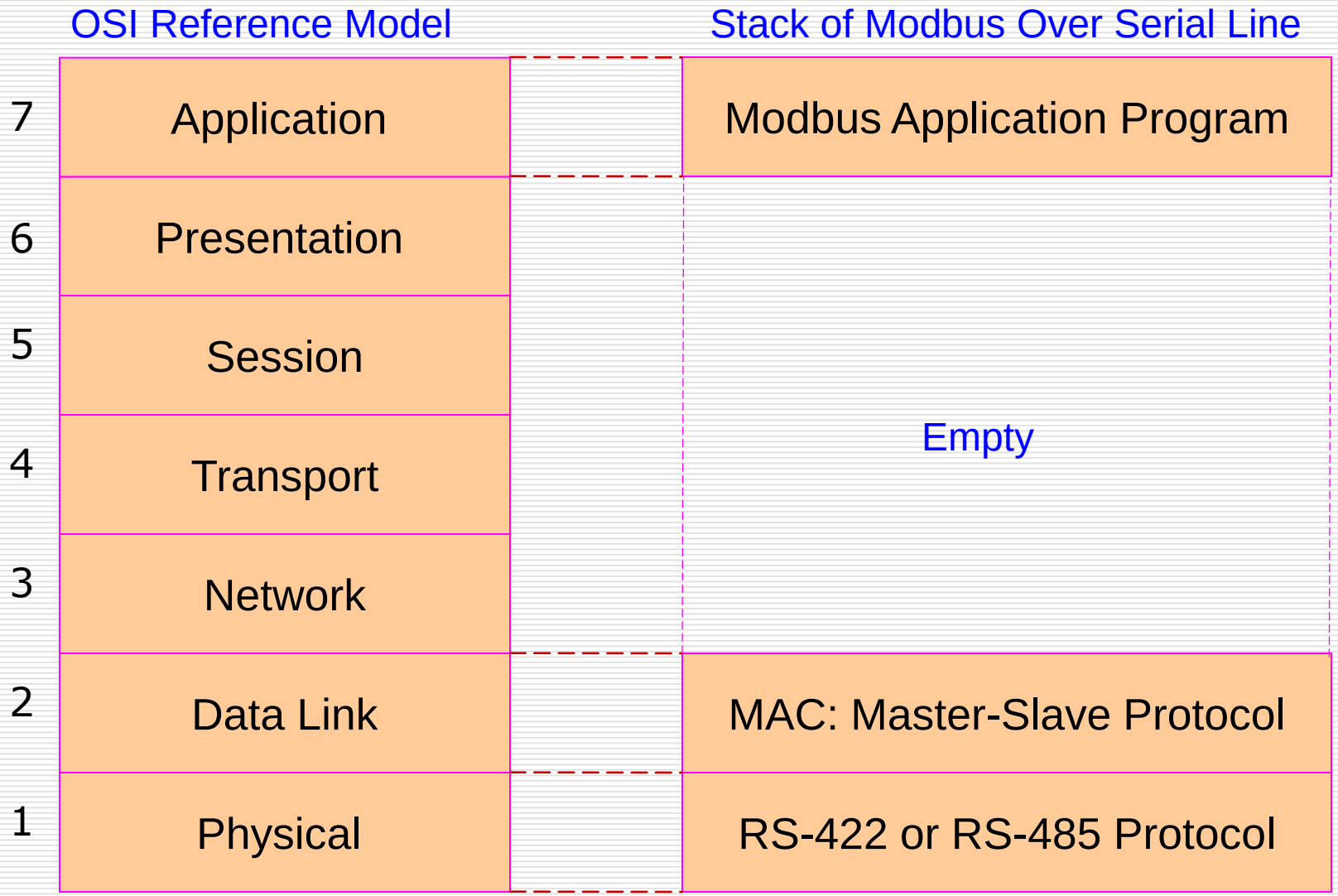
Modbus Over Serial Line

- As mentioned earlier, Modbus standard fully specifies the Application Layer and specifies master-slave protocol for MAC.
- "Modbus over serial line" is the variant of Modbus protocol compatible with the physical layer and MAC protocol of certain **serial-line communication protocols** suitable for device level networks.
- Two popular combinations of Modbus and serial-line communication are:

  a) Modbus / RS-422
  b) Modbus / RS-485

- Both RS-422 and RS-485 support master-slave protocol for medium access control (MAC).

Protocol Stack of Modbus Over Serial Line



Master-Slave Protocol

- Communication is always initiated by the master (that is, the master node)
- The master ‘requests’ a slave (that is, a slave node) through polling to send data
- The polled slave ‘replies’ or ‘responds‘ by transmitting the required data
- Slaves never initiate a dialogue
- Slaves never communicate with each other
- Master node initiates only one transaction at a time.

Request/Response Modes

- Two modes of request/response are used by "Modbus over serial line":
  a) Unicast mode
  b) Broadcast mode

(a) Unicast Mode

I. **Request**: Master addresses **an individual slave** by **polling technique.** It sends a "request message" containing address of the slave.

II. **Response**: Only the addressed/polled slave replies to the master. It sends a "response message" to the master

(b) Broadcast Mode

I. Master addresses **all slaves** connected to the bus. It sends (broadcasts) a "broadcast message" for a "writing function", for example "reset" message to them.

II. All slaves accept the message and act upon it.

III. Slaves don't send any "response message" back to master.

Modbus Address Space

- Address size is 8 bits
- Accordingly, total address space is 256 addresses.

Address	Purpose
0	To address all slaves in “broadcast mode”
1-247	To address individual slaves in “unicast mode”
248-255	Reserved

Variants of "Modbus over Serial Line"

- Depending on the **style or method of data representation and data transmission** used, following are the variants of "Modbus over serial line".
- Both the variants are ***open protocols***.

a) Modbus/RTU

- It uses RTU style of data representation and transmission
- RTU stands for "remote terminal unit", widely used in distributed control.
- Error control technique used: Cyclic redundancy check (CRC).

b) Modbus/ASCII

- It uses ASCII code for data representation and transmission.
- ASCII stands for "American Standard Code for Information Interchange".
- Error control technique used: Longitudinal redundancy check (LRC) or check-sum.

Modbus/RTU Protocol

1. Data Representation

2. Data Transmission

3. Message Format

Data Representation in Modbus/RTU

- Modbus/RTU uses 8-bit data bytes.
- Hexadecimal characters (0-9 and A-F) are used for representation of numbers.
- Two hexadecimal characters are accommodated in one byte.
- **Very compact data representation**
- Therefore, Modbus/RTU is the most commonly used Modbus variant.

Data Transmission in Modbus/RTU (1)

- Data transmission takes place in **asynchronous mode**.
- Modbus message is transmitted continuously without any time gap (that is, without any idle period) between bytes.
- But, two messages must be separated by an idle period of 28 bits or longer.

Data Transmission in Modbus/RTU (2)

- Each data byte may be followed by a parity bit, but the same is not mandatory.
- 11 bits are transmitted per data byte, whether parity bit is used or not.
- Bit sequence for data transmission if parity is used:

Start bit	8 data bits (LSB ….. MSB)	Parity bit	Stop bit

- Bit sequence for data transmission if parity is not used:

Start bit	8 data bits (LSB ….. MSB)	Stop bit	Stop bit

Message Format in Modbus/RTU

- Each Modbus/RTU message is comprised of 4 fields as under:

Field	Master-to-Slave Message (Request)	Slave-to-Master Message (Response)
1. Address	Slave address	Slave address
2. Function Code	Indicates to slave the kind of action to perform	Indicates the kind of response
3. Data	Request parameters	Response parameters & values
4. Error Check Code	CRC	CRC

- Format of "Message Frame" and size of each field in it are:

No. of bytes:	1	1	n (0-252)	2
Fields:	Address	Function Code	Data	CRC

Modbus/ASCII Protocol

1. Data Representation

2. Data Transmission

3. Message Format

Data Representation in Modbus/ASCII

- Modbus/ASCII uses 7-bit ASCII code
- Each character is represented by seven bits.
- Characters are numerals, alphabetic characters, special characters (like ‘.’ ‘,’ ‘:’) and commands (like LF, CR).
- Code is very inefficient
- Hence, Modbus/ASCII is not a popular Modbus variant.

Data Transmission in Modbus/ASCII (1)

- Data transmission takes place in **asynchronous mode**.
- Modbus message is transmitted continuously without any time gap (idle periods) between characters.
- There may or may not be any time gap (idle period) between two messages.

Data Transmission in Modbus/ASCII (2)

- Character bits may be followed by a parity bit, but the same is not mandatory.
- 10 bits are transmitted per character, whether parity bit is used or not.
- Bit sequence for data transmission if parity is used:

Start bit	7 data bits (LSB ….. MSB)	Parity bit	Stop bit

- Bit sequence for data transmission if parity is not used:

Start bit	7 data bits (LSB ….. MSB)	Stop bit	Stop bit

Message Format in Modbus/ASCII

- "Modbus/ASCII Message" has 6 fields as under:

Field	Master-to-Slave Message (Request)	Slave-to-Master Message (Response)
1. Start delimiter	: (colon)	: (colon)
2. Address	Slave address	Slave address
3. Function code	Indicates to slave the kind of action to be performed	Indicates the kind of response
4. Data	Request parameters	Response parameters & values
5. Error check code	LRC or check-sum	LRC or check-sum
6. End delimiter	CR/LF pair	CR/LF pair

- Format of "Message Frame" and size of each field in it are:

No. of characters:	1	2	2	n (0-2x252)	2	2
Fields:	Start delimiter	Address	Function code	Data	LRC	End delimiter CR, LF

Modbus TCP/IP Protocol

1. Highlights

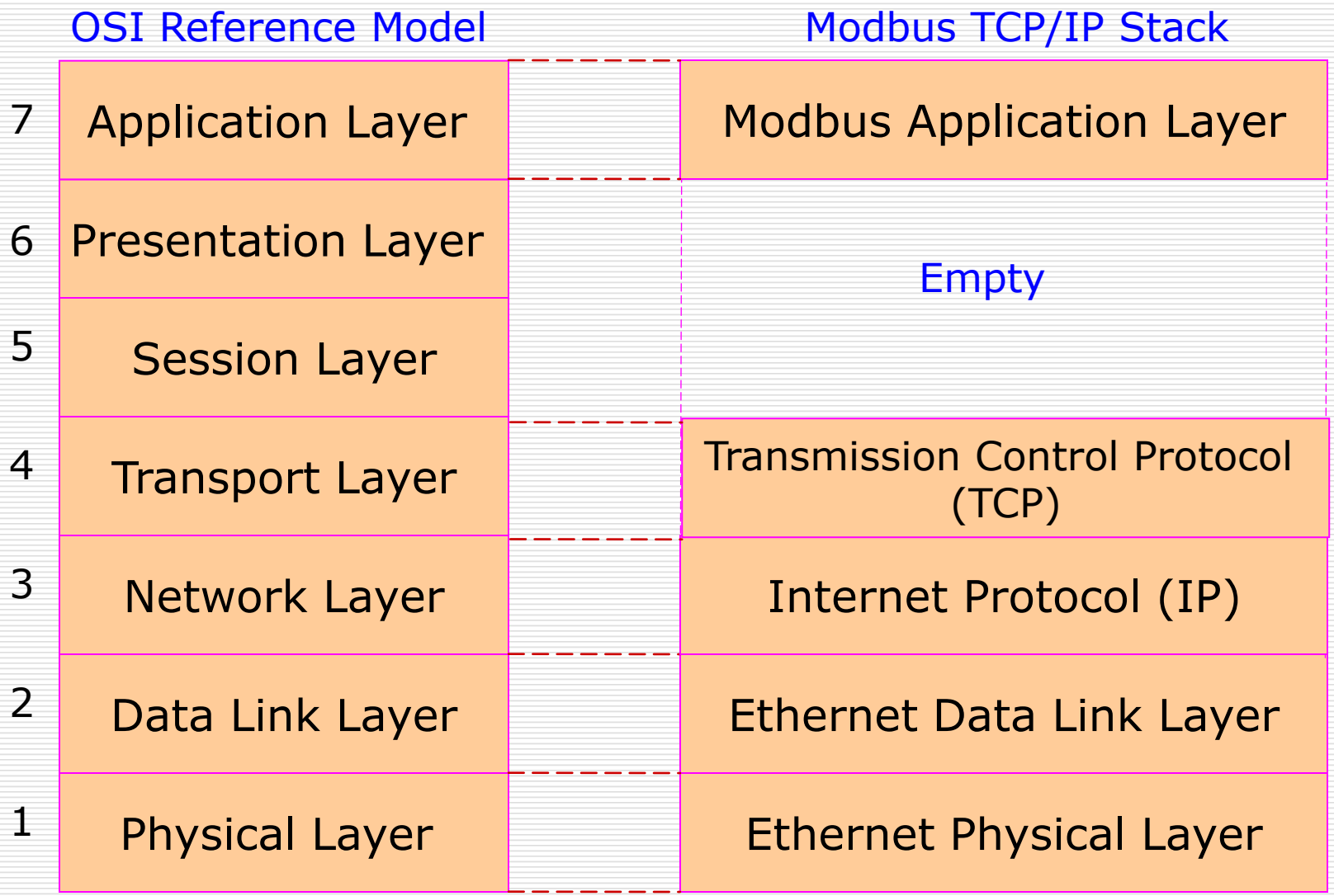
2. Protocol Stack

3. Message Format

Modbus TCP/IP or Modbus TCP

- Application layer is completely specified by Modbus.
- Communication takes place over a network operating on **TCP/IP protocol suite.**
- TCP/IP protocol suite specifies Transport and Network layers and comprises the following major protocols:
  - Transmission control protocol (TCP)
  - User datagram protocol (UDP)
  - Internet protocol (IP)
  - Address resolution protocol (ARP)
- Very often used with **Ethernet protocol**, which specifies Physical and Data Link layers.
- The master-slave relationship becomes **server-client relationship** here.
- **No error check code** (like CRC or checksum) is required in the Application Layer as the lower layers provide checksum protection.

Protocol Stack of Modbus TCP/IP



Message Format in Modbus TCP/IP

- "Modbus TCP/IP Message" has 6 fields as under:

Field	Request Message	Response Message
1. Transaction identifier	For synchronization between messages of server and client	
2. Protocol identifier	0 for Modbus TCP/IP	
3. Length	Number of remaining bytes in this frame	
4. Unit identifier	Slave address behind a gateway or some other unit (255, if not used)	
5. Function code	Indicates the kind of action by addressed device	Indicates the kind of response
6. Data	Request parameters	Response parameters & values

- Format of "Message Frame" and size of each field in it are:

No. of bytes:	2	2	2	1	1	n
Fields:	Transaction identifier	Protocol identifier	Length	Unit identifier	Function code	Data

Merits of Modus

1. As Modbus was developed by the automation industry with industrial automation applications in mind, its use in industrial data networks is straight forward, needing no adoptions.
2. It is an open protocol and there are no royalty issues.
3. The protocol is easy to deploy and maintain.
4. It allows movement of raw bits or words without placing many restrictions on the software designer.

Limitations of Modbus

1. The protocol cannot be used all alone as it specifies Application Layer alone.
2. Since Modbus was designed in the late 1970s to communicate with PLCs, the data types supported are limited to those used with PLCs at the time.
3. In the master/slave protocol, used in Modbus/RTU and Modbus/ASCII, there is no way for a slave node (field device) to report on its own to the master node.

Overview of Error Control in Data Communication

1. Error Control in Data Communication
2. Error-Detection Approach
3. Error-Detection Techniques:
   - Parity-Checking Technique
   - CRC Technique
   - Checksum or LRC Technique
4. Error-Correction Approach

Error Control in Data Communication

- Data can be corrupted due to occurrence of errors during transmission on data communication links and networks.
- Error control process is aimed at taking care of such errors.
- For reliable data communication, errors must be detected and, if possible, corrected.
- Types of data errors:
  a) **Single-Bit Error**: Only one bit in the "data unit" goes wrong (that is, either a '0' becomes '1' or a '1' becomes '0') during transmission
  b) **Multi-Bit Error**: Several bits in the "data unit" go wrong during transmission.
- Two approaches are used for error control:
  A. Error detection approach
  B. Error correction approach

"Error-Detection Approach" to Error Control

- This approach consists in receiver checking the received data unit for any error that may have occurred during transmission, using redundancy.
- **Redundancy** is the concept of using extra bits (called redundant or check bits) for use in error detection. The transmitter adds appropriate redundant bits to the data unit and transmits to the receiver. The receiver uses these bits to check whether any error occurred during the transmission:
  - If the receiver finds no error, it accepts the data unit.
  - If the receiver detects an error, it rejects the data unit and requests the transmitter to retransmit the original data unit.
- It should be noted that on a noisy transmission medium, repeated re-transmissions may become necessary. So, a successful data transmission could take a long time, or even never occur.
- Error detection techniques:
  i. Parity checking
  ii. Cyclic Redundancy Check (CRC)
  iii. Checksum or Longitudinal redundancy check (LRC)

Parity Checking Technique

- An extra bit, called **parity check bit**, is added at the end of the data unit that indicates whether the number of '1' bits in the augmented data is even (for even parity) or odd (for odd parity).
- If a single bit gets changed during transmission, the parity at the receiver end will fail. Thus, a single-bit error can be detected by the receiver based on this parity check.
- Parity checking is not very robust error detection technique, since if the number of bits changed during transmission is even, the parity check will pass and thus the error will not be detected.
- **The data units are generally the individual characters.**
- The technique is generally used with asynchronous data transmission.

CRC Technique

- Cyclic Redundancy Check (CRC) is a very efficient error-detection technique based again on redundancy.
- It is based on binary division of the data unit by a certain divisor. The remainder bits (called as CRC bits) are appended to the data unit at its end and transmitted to the receiver.
- The receiver divides the received data unit by the same divisor.
  - If the remainder is zero, it means there is no error and, therefore, the data unit is accepted.
  - If the remainder is not zero, it is concluded that an error has occurred during transmission.
- The divisor is called as **CRC generator**.
- **The data unit is the whole message or a data packet being transmitted.**

Check-Sum or LRC Technique

- The check-sum technique of error detection is also based on the concept of redundancy.
- Transmitter uses a “checksum algorithm” to generate a checksum, which is appended to the data unit before transmitting it.
- The simplest checksum algorithm is the so-called longitudinal redundancy check or LRC. It breaks the data unit into "words" of n bits (for example, 8-bit bytes) and then computes the Exclusive-OR (XOR) of all those words. The result, called as checksum, is appended to the data unit as an extra word.
- To check the integrity of received data, the receiver computes the XOR of all its words, including the checksum word.
  - If the result is a word consisting of n zeros, then there is no error and hence the data is accepted.
  - If the result is not a word consisting of n zeros, the receiver concludes that a transmission error has occurred.

"Error-Correction Approach" to Error-Control

- Error-correction approach enables the receiver to not only detect but also locate the error, if it has occurred during transmission.
- Once the location of error is known, the bit in that location is complemented to get the correct data unit.
- One such **error-correction technique** or **error-correction code** is the **Hamming Code**.
  - It works only for single-bit errors.
  - In case the data suffers a multi-bit error during transmission, the Hammimng code will only add one more error.
  - Redundant or check bits are inserted in the original data unit. These check bits are so distributed and computed that the error in different locations produces a different numbers. This resulting number gives the location of the erroneous bit.

Foundation Fieldbus

- **Fieldbus Protocols** are the data network protocols that have been developed specially for connecting **intelligent field devices** to their host controllers in plants (or on factory floors) for **distributed control applications.**
- Most of them conform to IEC-61158, either fully or in respect of the Application Layer.
- **Foundation Fieldbus (FF)** is one of the **open-architecture** fieldbus protocols.
- Developed in 1997, FF is a de-facto industry standard.
- Fieldbus Foundation, a world-wide consortium of manufacturers and industry groups, specifies Foundation Fieldbus (FF) and certifies products to be compliant with it.
- FF is now widely used the world over in control of industrial processes, specially **continuous processes.**

Highlights of IEC-61158

- IEC-61158 is the International Standard for "Digital Data Communications for Measurement and Control – Fieldbus for Use in Industrial Control Systems" (Highlights of the standard are given in next slide).
- It lays down following minimum requirements for Fieldbus Protocols:

1. Multi-drop operation
2. At least 30 devices in the network
3. Operation in hazardous areas using intrinsic safety techniques
4. Communication speed of 31.25 kbps in intrinsically-safe mode
5. Communication speed of 1 Mbps or higher in non-intrinsically-safe mode
6. Interoperability between IEC61158-compliant devices from different manufacturers.

Important Fieldbus Protocols

S. No.	Fieldbus	Developer	Year	Application Area
1	Modbus	AEG Modicon	1978	Process Control
2	MAP(2)	General Electric	1980	Manufacturing plants
3	LON(3)	Echelons	1991	Building automation Energy distribution
4	Profibus(4)	Siemens	1994	Process control
5	SDS(5)	Honeywell	1994	Process control
6	CAN(6)	Bosch	1995	Automobiles
7	Foundation Fieldbus	Fieldbus Foundation	1997	Process control

(2) Manufacturing Automation Protocol, (3) Local Operation Network
(4) Process Control Fieldbus, (5) Small Distributed System
(6) Controller Area Network

Foundation Fieldbus Variants

Foundation Fieldbus (FF) has two variants:

(a) FF-H1

- Low-speed variant
- Operates at 31.25 kbps
- Provides intrinsic-safety option.

(b) FF-HSE (High-Speed Ethernet)

- High-speed variant
- Operates at 10/100/1000 Mbps
- Does not provide intrinsic-safety option.

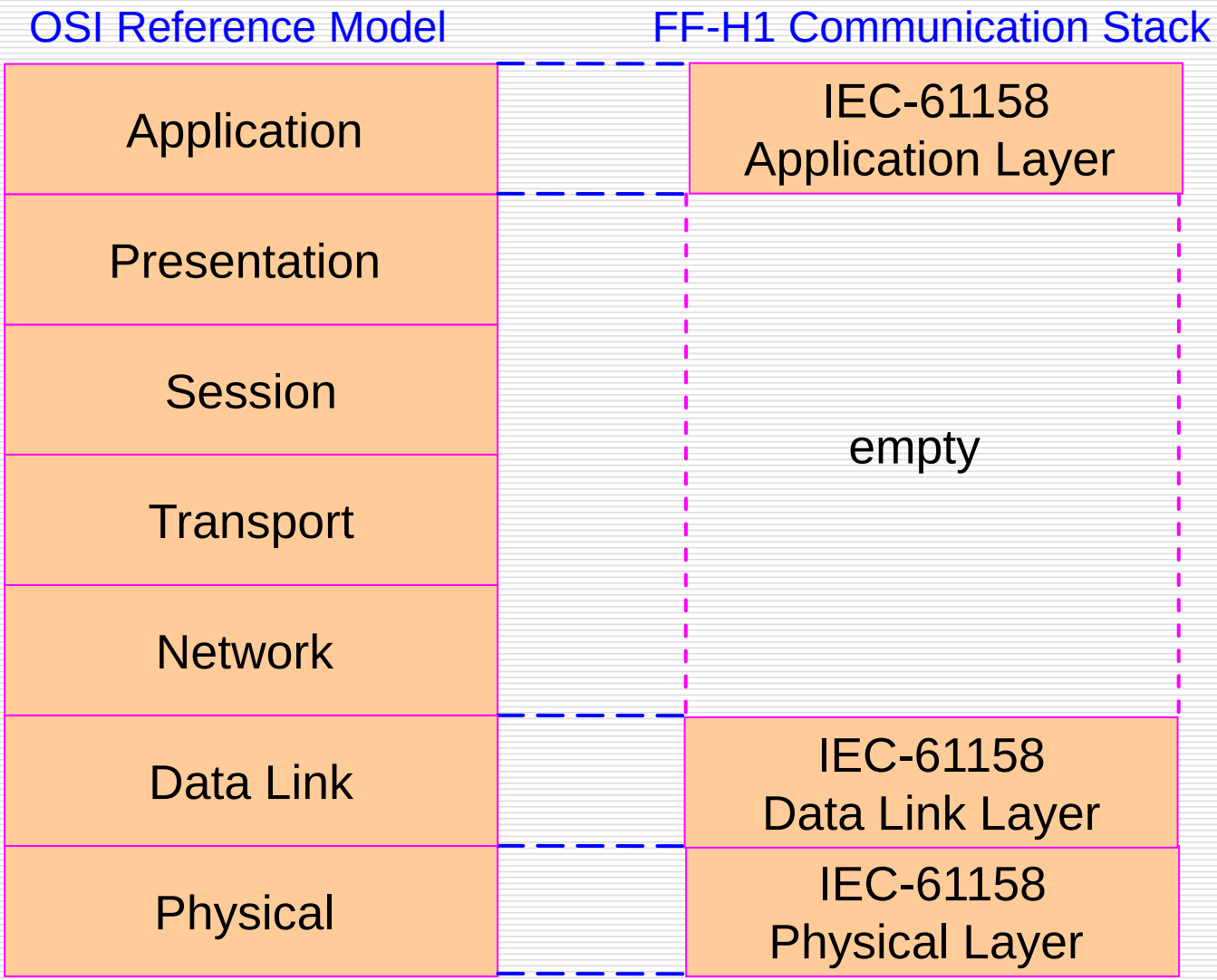
FF-H1 Protocol

1. Highlights
2. Communication Stack
3. Physical Layer
4. Frame Formats
5. MAC Protocol
6. Application Layer
7. Merits

Highlights of FF-H1 Protocol

- It is a low-speed variant of Foundation Fieldbus.
- Speed specified is 31.25 kbps
- Generally used to connect field devices to their host systems, that is controllers
- Provides communication and power over a single twisted-pair.
- Works in both conventional and intrinsic-safety (IS) applications.
- It fully conforms to IEC-61158 standard.
- It is a type-1 fieldbus in terms of IEC-61158
- The protocol defines the following three layers of OSI model and all of them conform to IEC-61158 standard:
  - ✓ Physical Layer
  - ✓ Data Link Layer
  - ✓ Application Layer

FF-H1 Communication Stack

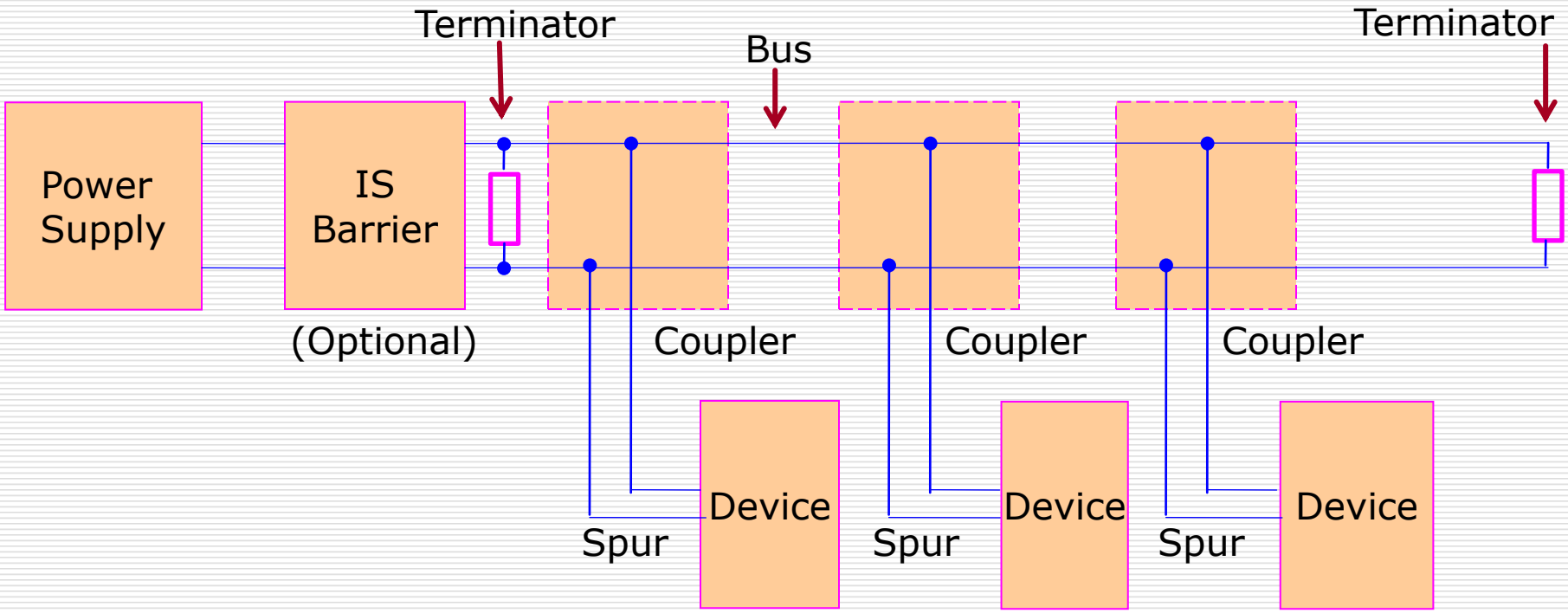


Physical Layer of FF-H1

Following aspects are specified:

- **Data Rate**: 31.25 kbps
- **Network Topologies Supported**:
  - Bus and Tree network topologies
  - Network structures shown in next two slides
- **Type of cable**: Both UTP and STP cables can be used
- **Maximum total length of cable:**
  - 1900 m with STP cable or 400 m with UTP cable
  - Total cable length = Length of trunk + length of all spurs
- Signal strength and encoding
- Physical-frame format

Bus Structure of FF-H1



Device: Field device or Controller

Intrinsic-safety (IS) Barrier: For operation in hazardous areas only

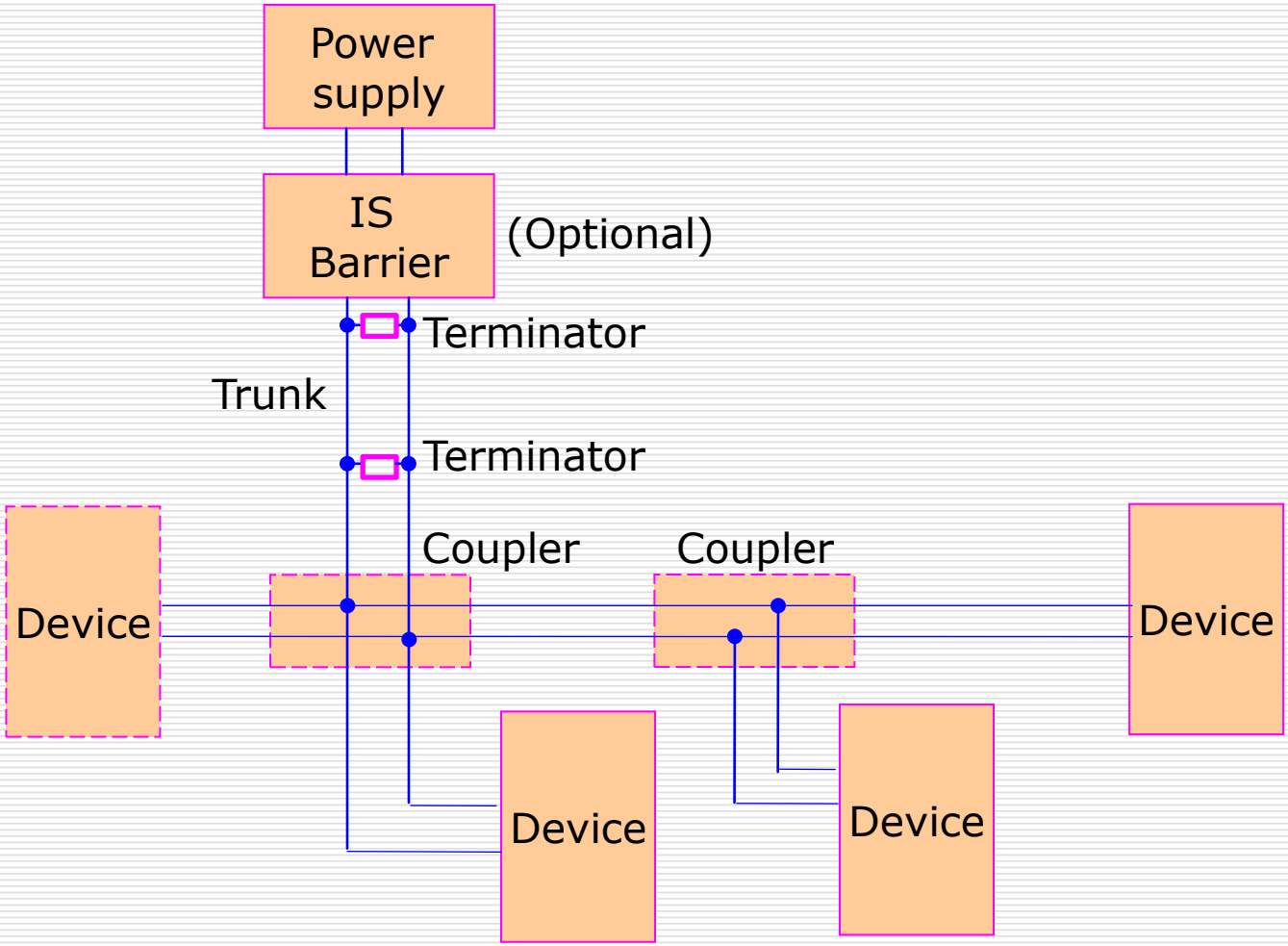
Spur: Connection > 1m

Splice: Connection ≤ 1m

Passive coupler: To connect field devices

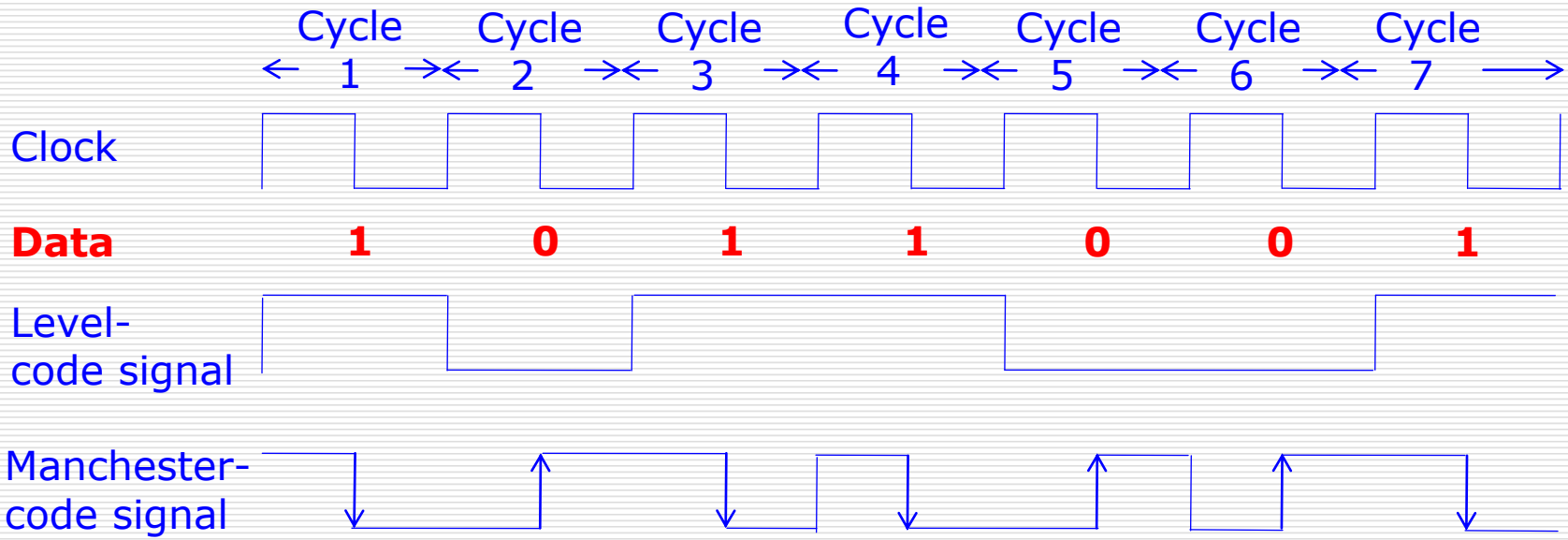
Active coupler: To extend spur length

Tree Structure of FF-H1

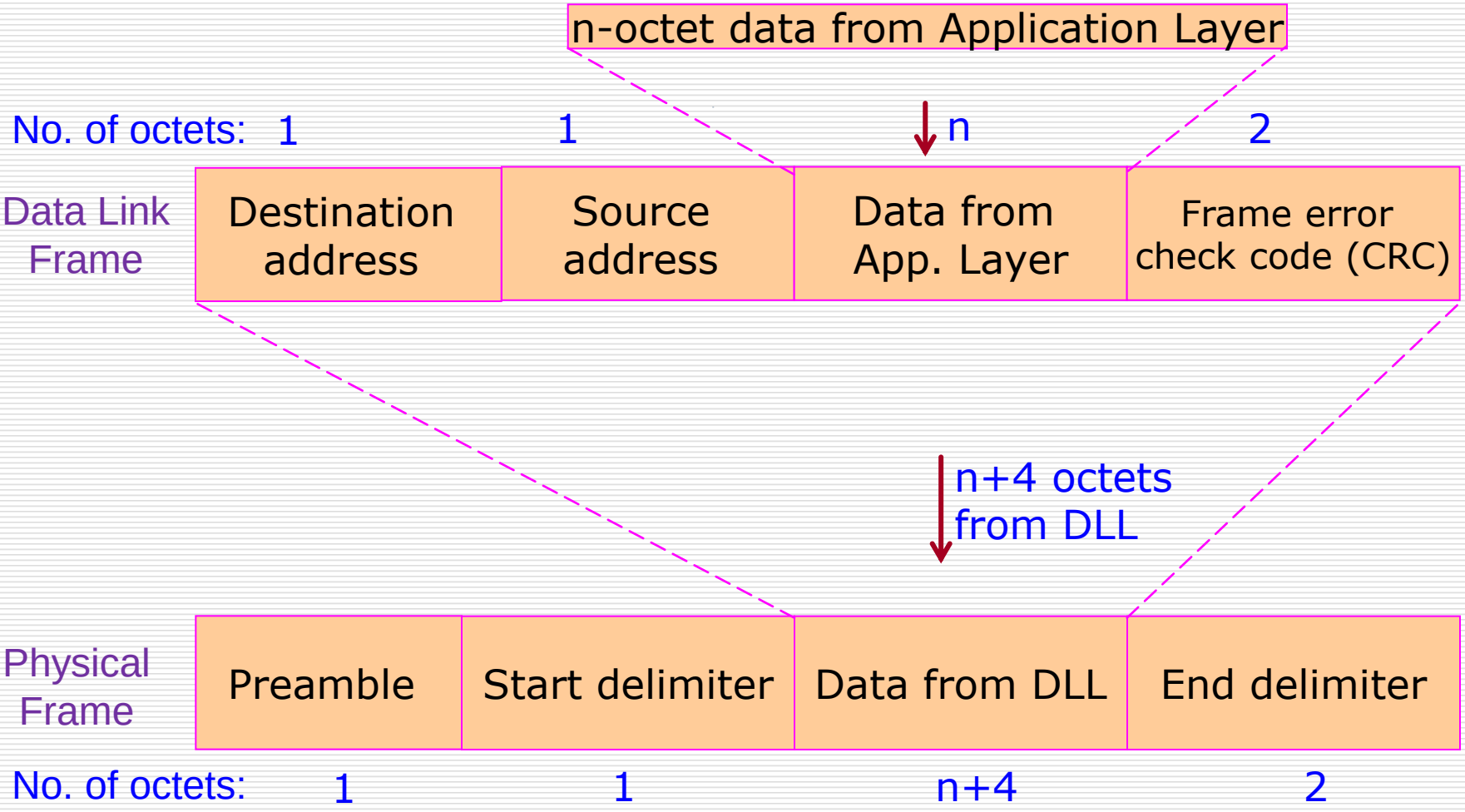


Signal Strength and Encoding in FF-H1

- Transmitter output : 15-20 mA p-p
- Receiver Sensitivity: 150 mV
- Encoding Method : Manchester code -
  ‘0’ represented by mid-cycle rising edge
  ‘1’ represented by mid-cycle falling edge



Frame Formats of FF-H1



MAC Protocol of FF-H1

- MAC protocol supports multiple masters.
- So, it is based on fusion of two principles as under:

a) Token-passing principle

- Used by masters to initiate communication
- After communication is finished by the current master, it passes the token to the next master.

b) Polling principle

- Used by the master to poll slaves
- The master requests, the addressed slave responds.

- So, all frames contain source address (SA) as well as destination address (DA).

Application Layer of FF-H1

- As mentioned earlier, Application Layer of FF-H1 conforms to IEC-6118.
- Usually, a device has a set of functions it can perform. These functions are represented as **function blocks** within the device.
- A function block can be thought of as a processing unit.
- These function blocks are used as building blocks in defining a monitoring and control application.
- IEC-61158 defines a standard set of function blocks, including 10 for basic control and 19 for advanced control. Next slide describes some of them.
- Manufacturer of a device can define additional function blocks for the device.
- In addition to function blocks, the Application Layer of IEC-61158 makes use of resource blocks and transducer blocks.
- **Resource block** specifies the general characteristics of a resource (for example, a device). This includes the device type and revision, manufacturer's ID, etc.
- **Transducer blocks** read directly from physical sensors into function blocks.

Standard Function-Blocks

- Some of the standard function blocks meant for basic control and input/output (I/O) functions are described below as examples:
- **AI (Analog Input) Block:** It reads data from a single analog input channel and performs filtering and scaling of this raw data to engineering units. Supports limit alarming.
- **AO (Analog Output) Block:** It writes data to an analog output channel and supports switching from manual to automatic mode of control. It also reacts if communication between it and the upstream block fails.
- **PID (Proportional–Integral–Derivative) Block:** It implements a PID control algorithm. It has to be connected to an upstream block (such as an AI block) and a downstream block (such as an AO block) before it can be used for control.
- **DI (Discrete Input) Block:** It reads data from discrete input channels and performs filtering and processing of this raw data. Supports limit alarming.
- **DO (Discrete Output) Block:** It writes to a discrete output channel and supports cascade initialization to allow upstream control blocks to determine the current state of the process before assuming control. It also reacts if communication between it and the upstream block fails.

Merits of FF-H1

1. It is an open protocol and there are no royalty issues.
2. Operation takes place with just two wires that carry power as well as communication signals.
3. Unlike RS485 and Modbus, FF-H1 can be deployed all alone, as it specifies all the three essential layers of a network communication system.
4. As it conforms to the common international standard for Fieldbuses (IEC-61158), it meets the requirement of interoperability between devices from different manufacturers.
5. With intrinsic safety techniques, FF-H1 ensures satisfactory network operation in hazardous areas.
6. FF-H1 supports tree as well as bus topology of networks.
7. Signal-edge based Manchester coding used in FF-H1, provides a higher data security against noise than the signal-level based data coding techniques.

FF-HSE Protocol

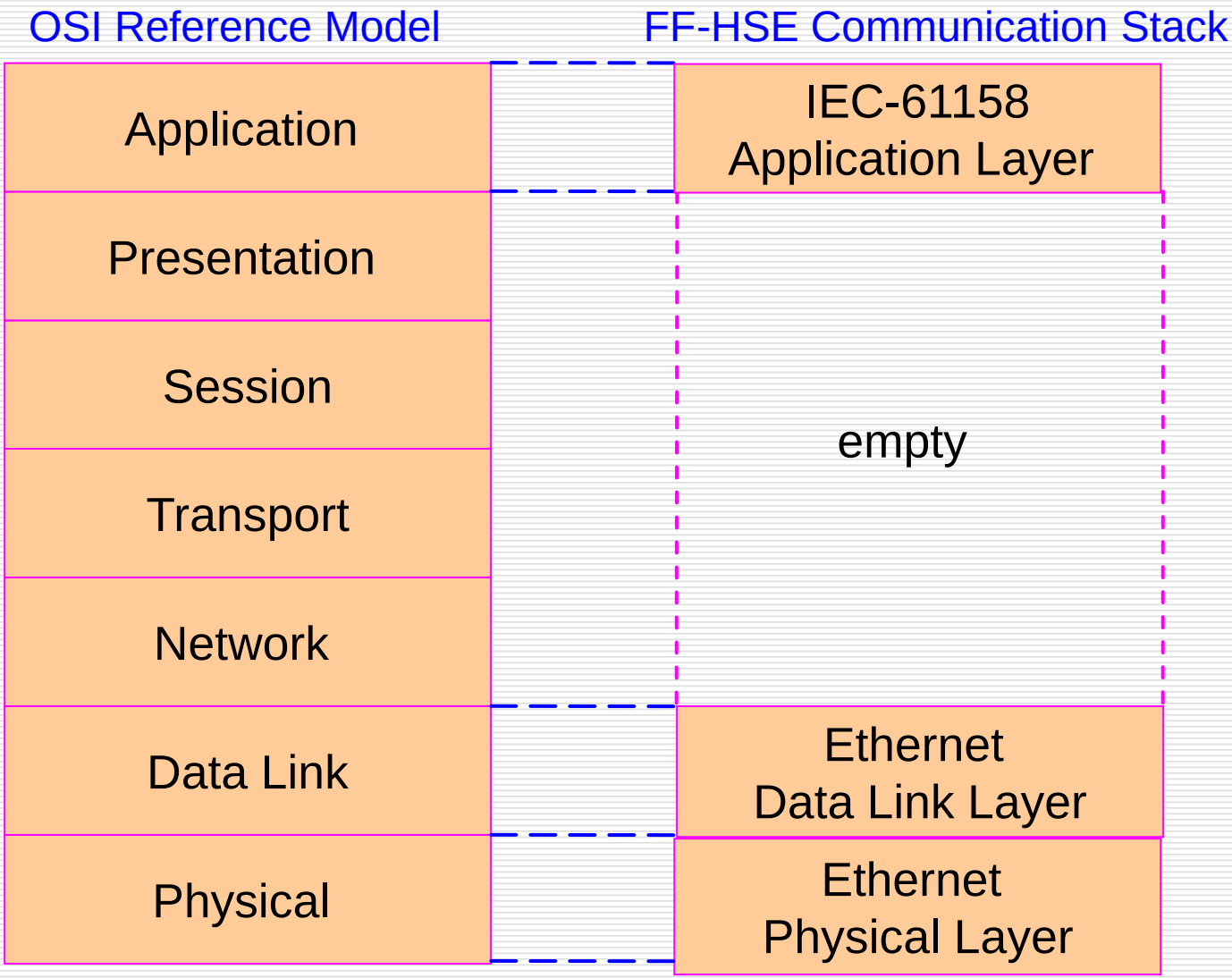
1. Highlights

2. Communication Stack

Highlights of FF-HSE Protocol

- It is a high-speed variant of Foundation Fieldbus.
- Application Layer of FF-HSE, which conforms to IEC-61158, is combined with Physical and Data link Layers conforming to Ethernet protocol / IEEE-802.3 standard.
- Speeds specified are those of the Ethernet / IEEE 802.3, namely 10/100/1000 Mbps
- Generally used to create a network of control terminals, work-stations, controllers, input/output units, PLCs and gateways etc. in the control room of an industrial plant.
- It is a type-5 fieldbus in terms of IEC-61158
- STM can be either STP copper cable or optical fibre cable as per IEEE 802.3 standard.
- Separate copper wire pair is used for powering the devices / equipment connected on the network.
- Does not have intrinsic-safety option.

FF-HSE Communication Stack

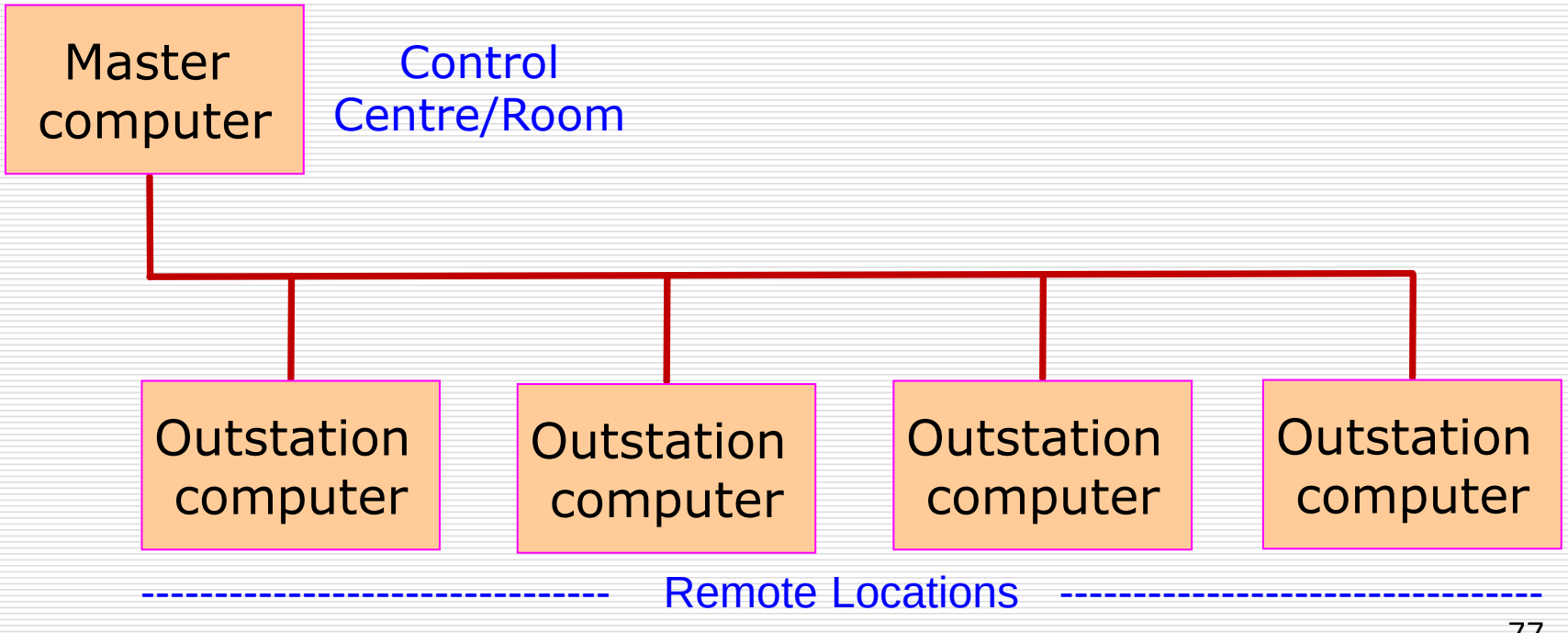


Distributed Network Protocol (DNP)

- Distributed Network Protocol (DNP) is a communication protocol designed specially for (a) **data acquisition** and (b) **control** on wired-networks in :
  - **A. Distributed control system (DCS),** and
  - **B. Supervisory control and data acquisition (SCADA) system.**
- It has been used primarily by electrical and other utility industries.
- It is an open (non-proprietary) protocol that is available to anyone on the web site www.dnp.org.
- Details of DNP will be taken up with particular reference to ***its popular version 3.3, known as DNP3.***

Concept of Master and Outstation Computers

- The DNP protocol is a set of rules:
  - For communicating "**data**" and "**control commands**"
  - Between a "**master computer**" (located in the control centre/room) and "**outstation computers**" (located remotely from the control room, distributed in the controlled plant/system).
- The concept is illustrated below:



Information Communicated by Master Computer

Master computer sends following types of information to outstation computers:.

a) **Binary control commands**, like close or trip a breaker, start or stop a motor.

b) **Analog values** to set regulated variables (i.e. **"set-points"** of controllers), for example the desired voltage level or pressure level.

c) All **requests** to outstation computers for obtaining **data** from them.

d) Information meant for **synchronizing time and date** between the master computer and outstation computers.

Information Communicated by Outstation Computers

Outstation computers send following types of information

to the master computer:

a) **Binary data** that conveys the status of two-state devices, such as whether a circuit breaker is closed or tripped.

b) **Analog data** that conveys values of analog variables like voltage, current, power, water level, temperature, etc.

c) **Counter data** that conveys the integrated values, such as energy.

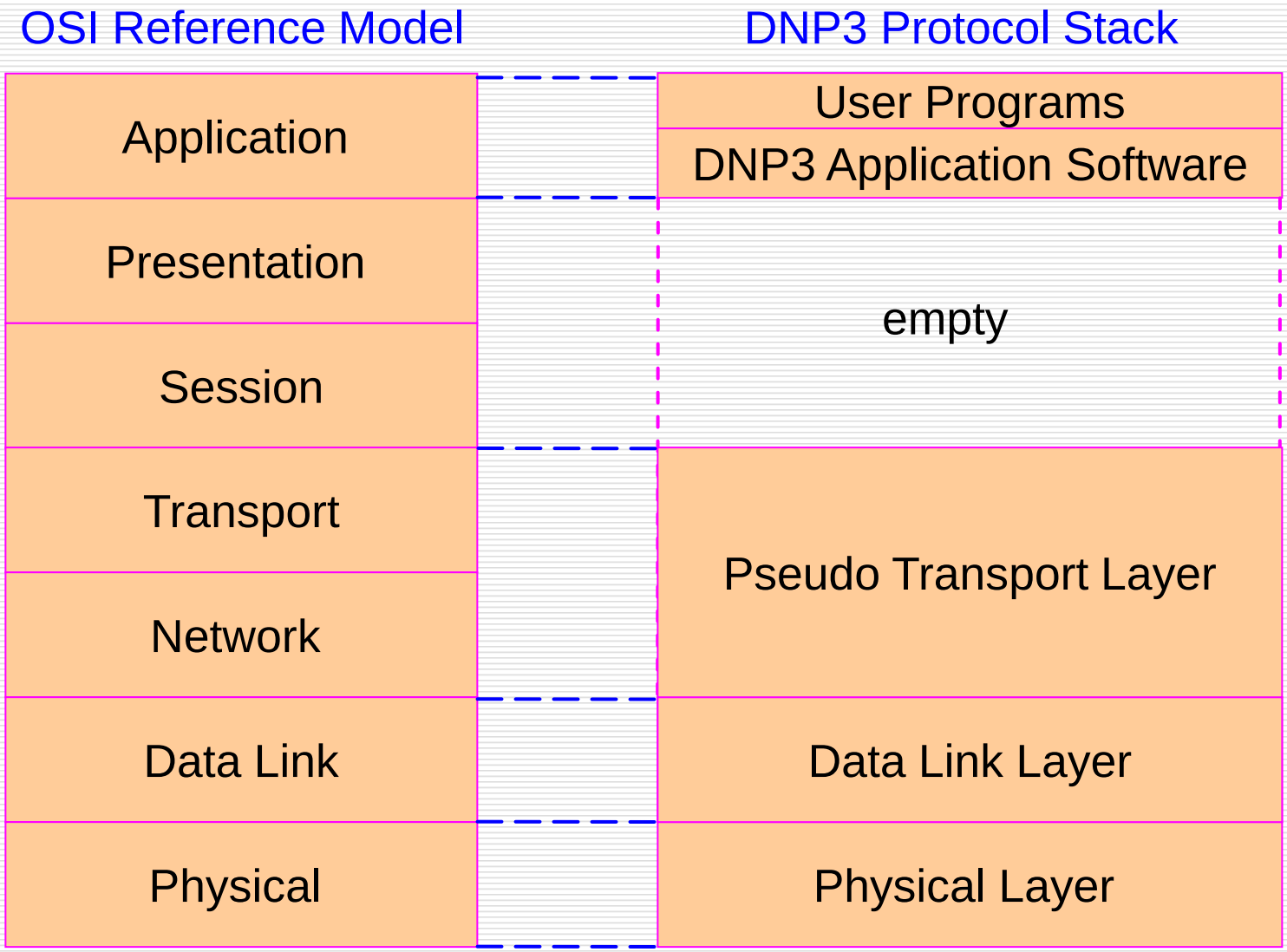
d) All **responses** to the master computer (that can include **any type of data** in general).

e) Information meant for **synchronizing time and date** between the master computer and the outstation computers.

DNP3 Protocol Structure

- With reference to the 7-layer OSI model, DNP3 protocol structure **has only four layers** (see next slide).
- The Application Layer of OSI model is split into two sub-layers: “User Software” or “User Programs” and “DNP3 Application Software”.
- The Transport and Network Layers of OSI model are combined into a single layer, called “Pseudo Transport Layer”.
- The Data Link Layer is well defined.
- The MAC and Physical Layer are not rigidly defined.

DNP3 Protocol Stack



Application Layer of DNP3

- As stated earlier, Application Layer is comprised of two sub-layers:
  a) **User Programs** at the top, and
  b) **DNP3 Application Software** below it.
- **Role of User Programs (UPs):**
  - **UPs in Master Computer:** Generation of "**control commands**" and **"requests"** to be sent to outstation computers and updating the database using the **"responses"** received from outstation computers.
  - **UPs in Outstation Computer:** Preparing **"response"** to be sent to master computer using its own database.
- **Role of DNP3 Application Software (AS):**
  - **AS in Master Computer**: **Transmission** of control commands and requests to outstation computers and **receiving** responses from outstation computers, **through the layers below it**.
  - **AS in Outstation Computers**: **Receiving** requests from the master computer and **transmission** of responses to the master computer, **through the layers below it.**

Pseudo-Transport Layer of DNP3

- The layer combines some of the functions of Transport Layer and a few functions of Network Layer of the OSI model.
- Since NOT all the functions of transport layer of OSI model are performed by it, it has been called as *Pseudo-transport Layer (PTL).*
- When sending data, PTL has the responsibility of breaking long messages coming from Application Layer into smaller packets and passing them down to DLL.
- The maximum size of a packet allowed is 250 octets (bytes).
- When receiving data, PTL reassembles packets coming from DLL into a single message and passes it on to Application Layer.

Data-Link Frame-Format of DNP3

- DLL adds header and CRC bits to the data packet received from PTL and sends it as Data-Link Frame to Physical Layer.
- Thus, a "DNP3 Data-Link Frame" consists of two fields:
  a) Header (fixed length of 10 octets)
  b) Data Section (variable length)

- The format of DNP3 Data-Link Frame is shown below:

Field:	Header	Data Section
Field size:	Fixed –10 octets	Variable

'Header' of DNP3 Data-Link Frame

- As stated earlier, Header has a fixed size of 10 octets (bytes).
- It has 6 fields as shown below:

Field:	Sync	Length	Link control	Destination address	Source address	CRC
Size (octets):	2	1	1	2	2	2

- Various fields of Header have following meanings:

Field	Meaning/ Purpose
Sync	Helps the receiver determine where the frame begins
Length	Specifies the number of data octets in Data Section
Link control	Data link control information
Destination address	Address of destination device
Source address	Address of source device
CRC	CRC for the header

'Data Section' of DNP3 Data-Link Frame

- The data section contains the user data passed down from PTL as a data packet and CRC bits computed and added by Data Link Layer.
- It is organized as multiple blocks, where each block can have a maximum of 16 bytes of user data followed by 2 bytes of CRC.
- As an example, the data section for 250 bytes of user data will have 282 bytes in all, organized as under:

User data (Block 1)	CRC for Block 1	.....	User data (Block 15)	CRC for Block 15	User data (Block 16)	CRC for Block 16
16	2		16	2	10	2

Sizes in octets or bytes

Length of Data Section = (16 data bytes + 2 CRC bytes) per block X 15 blocks + 10 (remaining data bytes) + 2 CRC bytes = 282 bytes

Addressing in DNP3

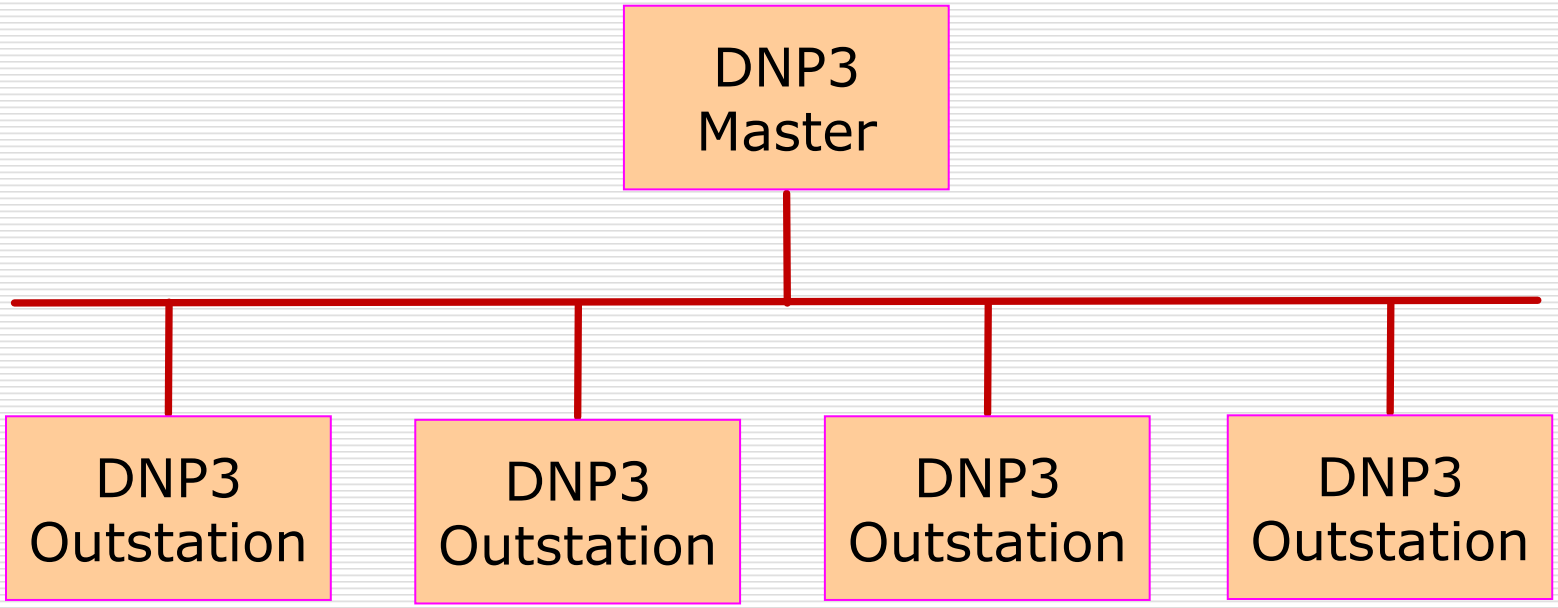
- The destination address specifies which DNP3 device should process the data, and the source address identifies which DNP3 device sent the message.
- Having both destination and source addresses satisfies essential requirement of peer-to-peer communication, because the receiving device knows where to direct its responses.
- Address size is 2 bytes, allowing a total of 65,536 addresses.
- These 65,536 addresses are assigned as under:
  - 65,520 addresses are available to address individual devices (thus allowing upto 65,520 devices in the network).
  - 3 addresses are reserved by DNP3 to denote an 'all-call' message, in which case the frame should be processed by all receiving devices.
  - One address is a universal address.
  - 12 addresses are reserved for special needs in the future.

Physical Layer and MAC

- The physical layer and MAC are not rigidly specified in DNP3.
- They support a number of communication protocols, signal transmission media and system architectures or topologies as under:
- **Communication protocols supported by DNP3:**
  - Serial Communication Protocols: RS232, RS422 and RS485
  - TCP/IP
- **Signal transmission media supported by DNP3:**
  - Copper cable
  - Optical fibre cable
  - Telephone line
- **System architectures or topologies supported by DNP3:**
  1. Multi-drop system/topology
  2. One-on-one system or point-to-point topology
  3. Hierarchical system
  4. Data collection system

Multi-Drop System

- This is the common type of system architecture/topology.
- One master computer communicates with multiple outstation computers/devices.
- The communication medium used is:
  a) Copper cable
  b) Optical fibre cable, or
  c) Multi-dropped telephone line.



Communications in Multi-Drop System

- The system may use either the simple master-slave mode of communication or peer-to-peer communication.

(a) Master-slave communication:

- Typically, the conversations take place between the master station computer and one outstation computer at a time.
- The master computer requests data from the first outstation computer/ device, then moves onto the next outstation computer/ device for its data, and continually interrogates all outstation computers/ devices in a cyclic order.
- All outstation computers hear a message from the master, but only the addressed one responds.

(b) Peer-to-peer communication

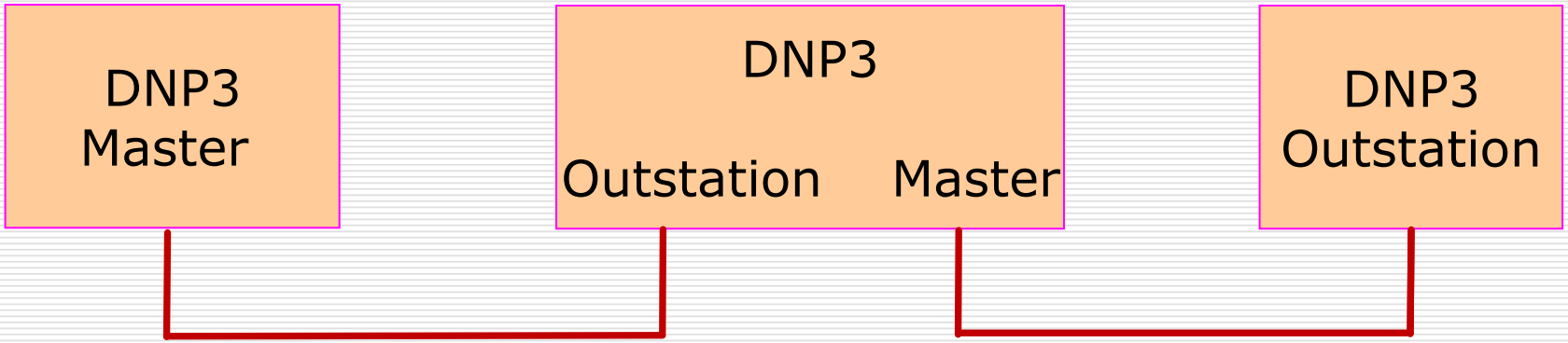
- A device operates as master for obtaining data from an outstation computer and/or sending commands to an outstation computer.
- Later, it may change its role to become an outstation computer to another device that now takes on the role of master.

One-on-One System

- The system has one master computer and one outstation computer as illustrated below.
- The physical connection between the two computers is typically:
  a) A dedicated line, or
  b) A dial-up telephone line.

Hierarchical System

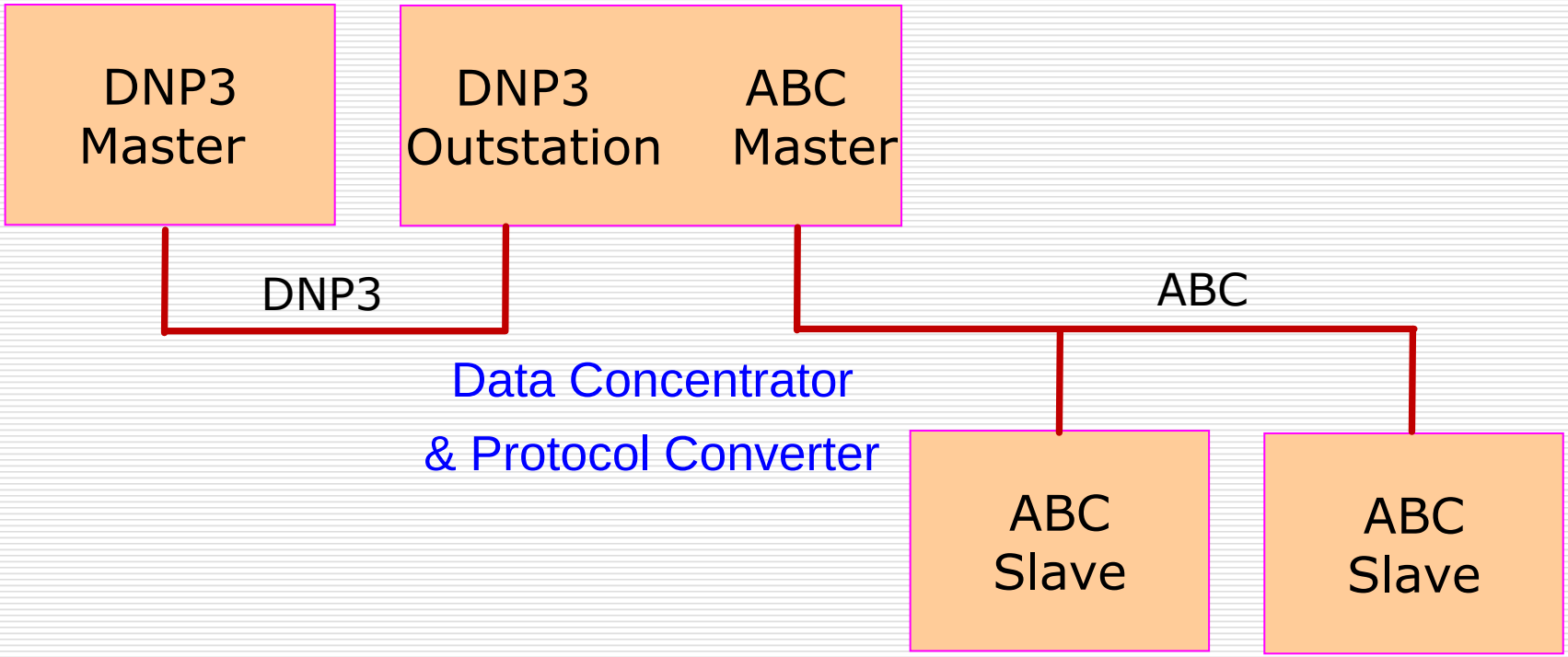
- Figure below shows a hierarchical system.
- The device in the middle acts as an outstation to the master at the left and as master with respect to the outstation on the right.
- The middle device is often termed a **sub-master**.



DNP3 Sub-Master.

Data Concentration System

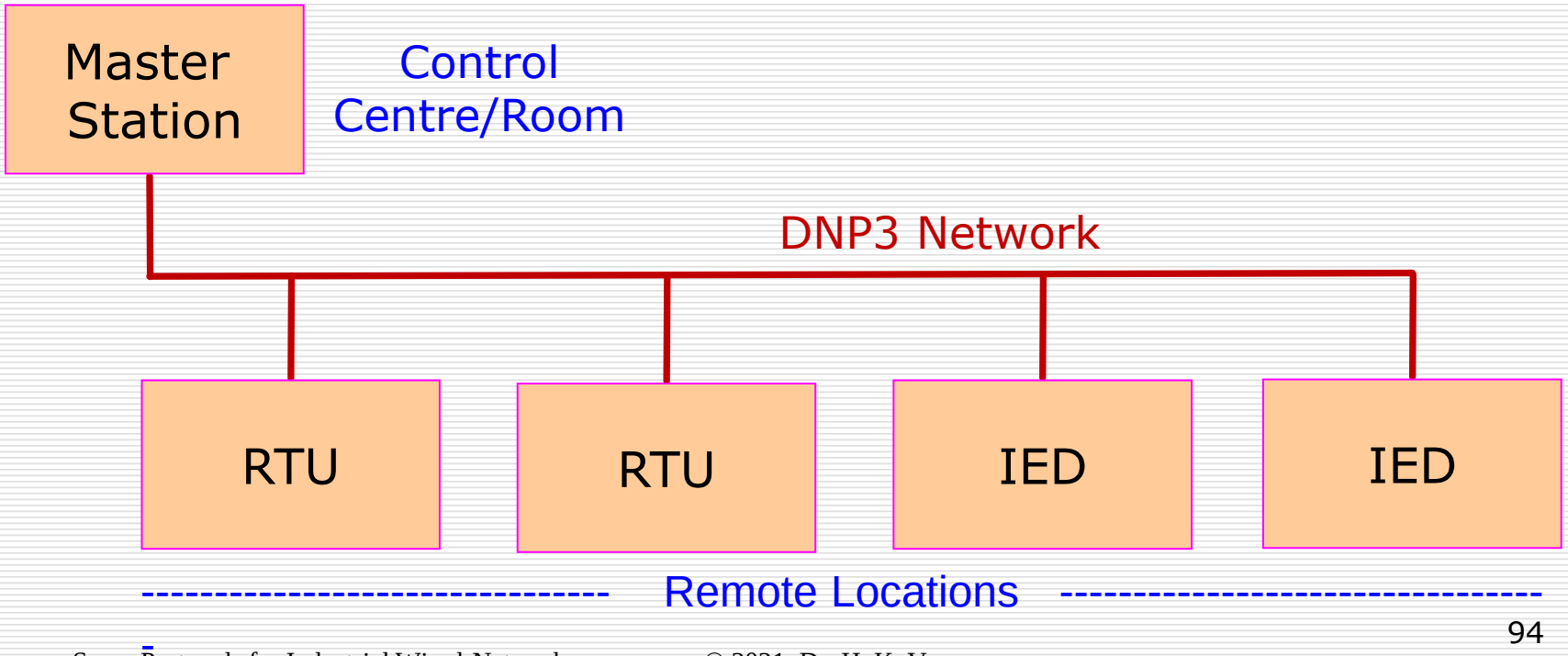
- The device in the middle acts as a master, working on some ABC protocol (different from DNP3), to collect data from its associated slave devices and sends the same to DNP3 master on left side, for which it acts like a DNP3 outstation computer.
- So it works both as data concentrator and as protocol converter.



DNP3-Based SCADA System

- Master Station* of the SCADA system works as the Master Computer for DNP3 communications.
- Remote Terminal Units* (RTUs) and Intelligent Electronic Devices* (IEDs) of the SCADA system work as Outstation Computers for DNP3 communications.

*For details, see http://profhkverma.info/wp/students/scada/



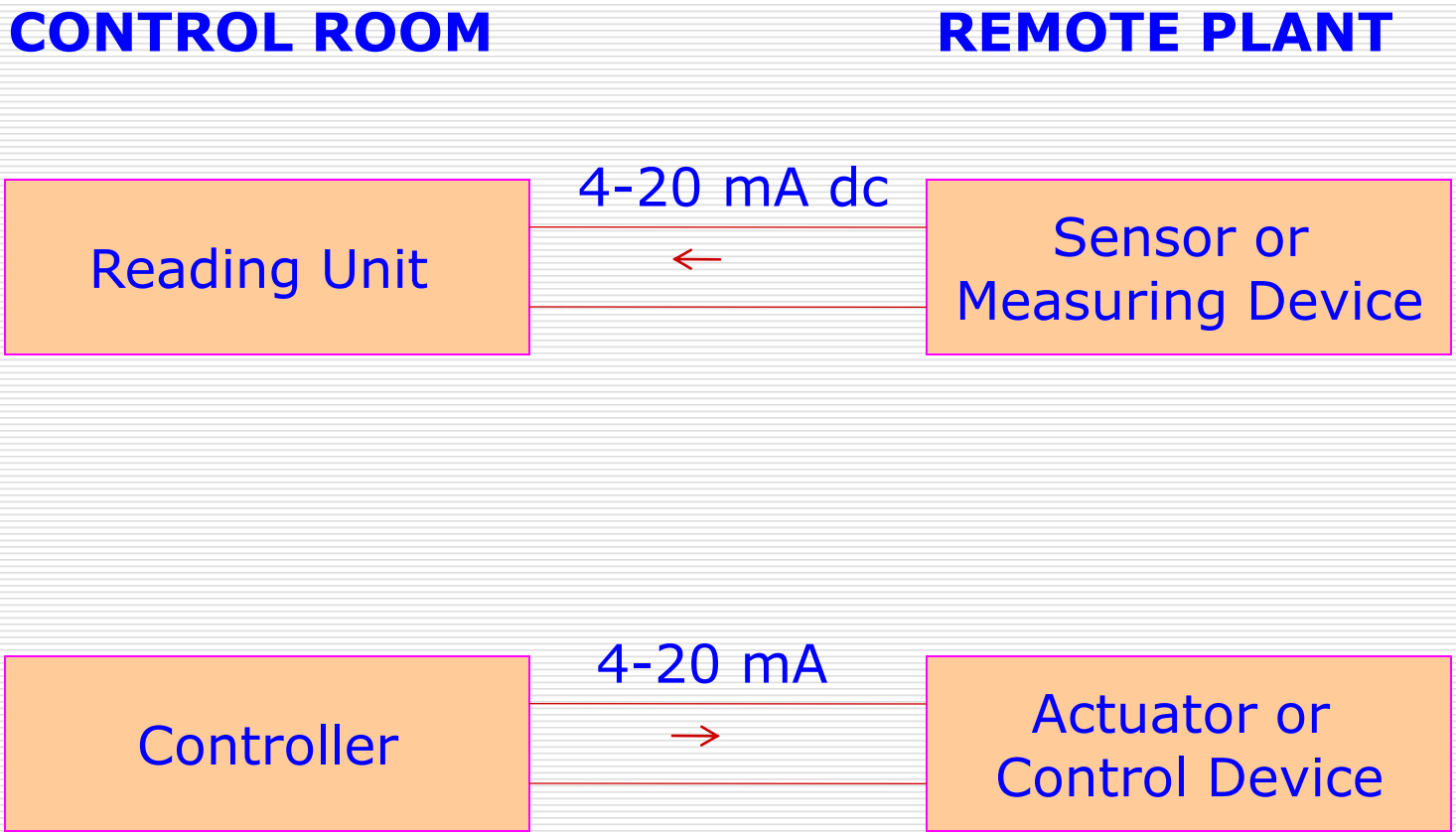
Merits of DNP3

1. DNP3 is an open protocol free from any licensing issues.
2. As DNP3 was designed and developed for data acquisition and control, it is one of the best protocols available for distributed control applications, specially in DCS and SCADA environment.
3. It transports data as generic values rather than being limited by any specified types of objects.
4. It does not impose any restriction on the size of message to be transferred, as long messages are broken into shorter packets for transmission.
5. The protocol supports communication on both, local area networks (LANs) and wide area networks (WANs)
6. Extensive use of CRC bits (16 CRC bits for every data block of 16 bytes) provides a high degree of assurance that communication errors, including multi-bit errors, in the data shall be detected.

HART Protocol

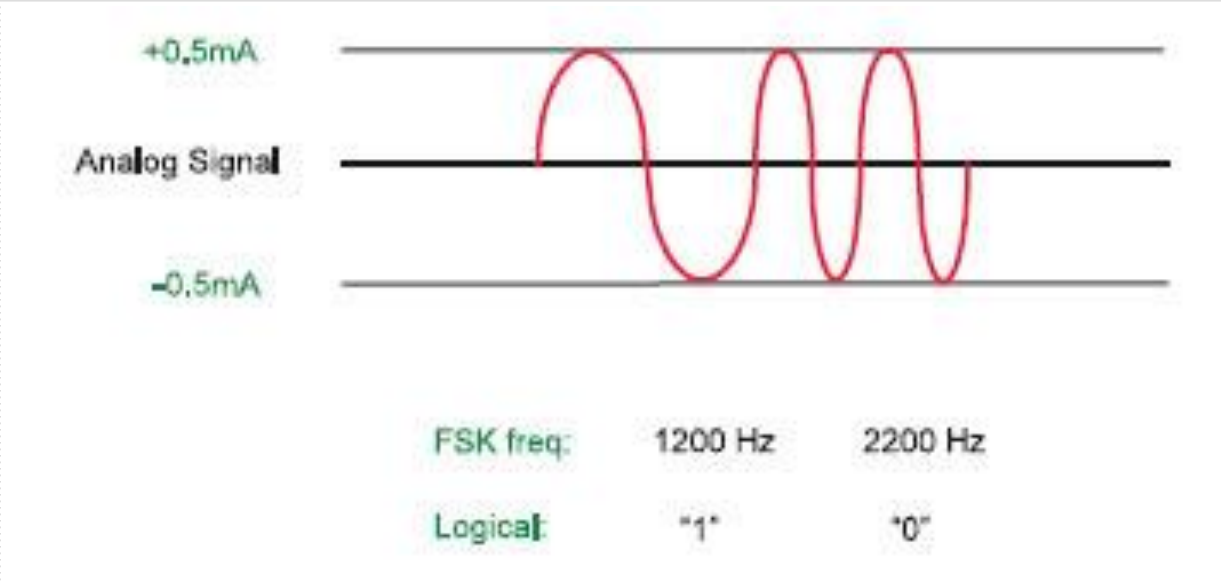
- A field communication protocol
- HART: Highway Addressable Remote Transducer
- HART protocol adds digital (data) communication to 4-20 mA field devices widely used in process automation
- Superimposes digital communication signal on analog communication signal of 4-20 mA on the same pair of wires
- Analog Communication: One-way only
- Digital Communication: Two-way (half duplex)
- Normally used for collecting data from field devices, configuring them and controlling them

4-20 mA Field Devices



Digital Communication Technique

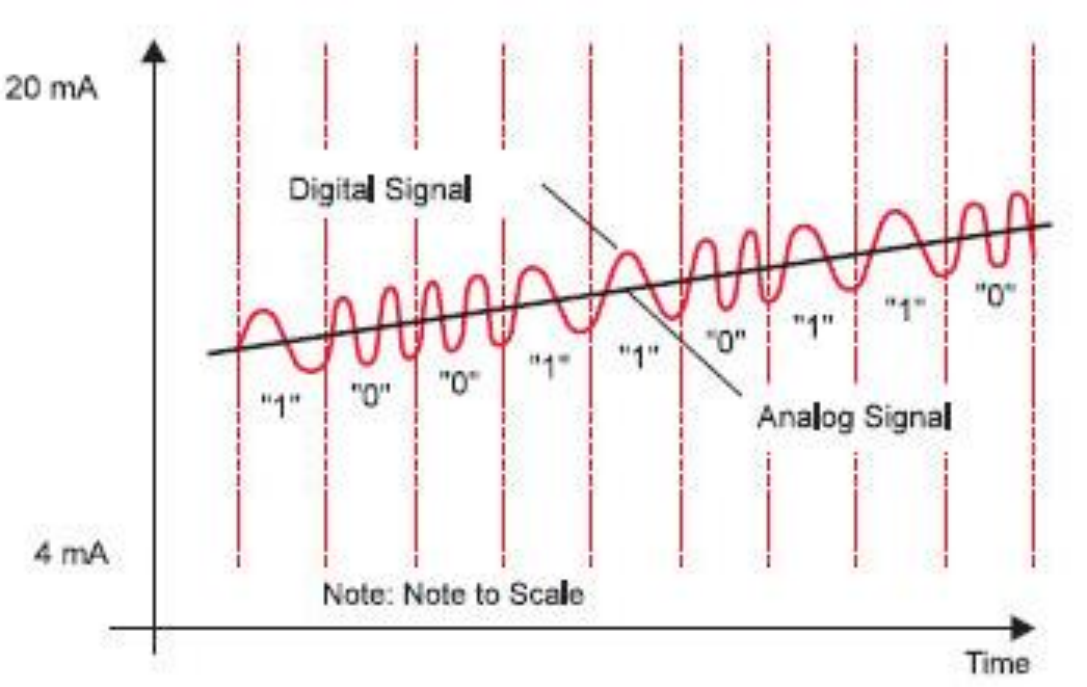
- Bell-202 FSK standard for digital communication
- Digital communication signal is low-level current carrier (0.5mA peak)
- Logic 1: 1200 Hz, Logic 0: 2200 Hz
- Maximum data rate: 1200 bps



Source: www.smar.com

Signal Superimposition

Digital (FSK) signal is superimposed on Analog (4-20 mA) signal



Source: www.smar.com

Cable Specifications

One UTP

- Maximum length of 1500 m

Alternatively, one STP

- Maximum length of 3000 m

Communication Configurations

HART protocol supports two com system configurations:

A. Point-to-Point Configuration

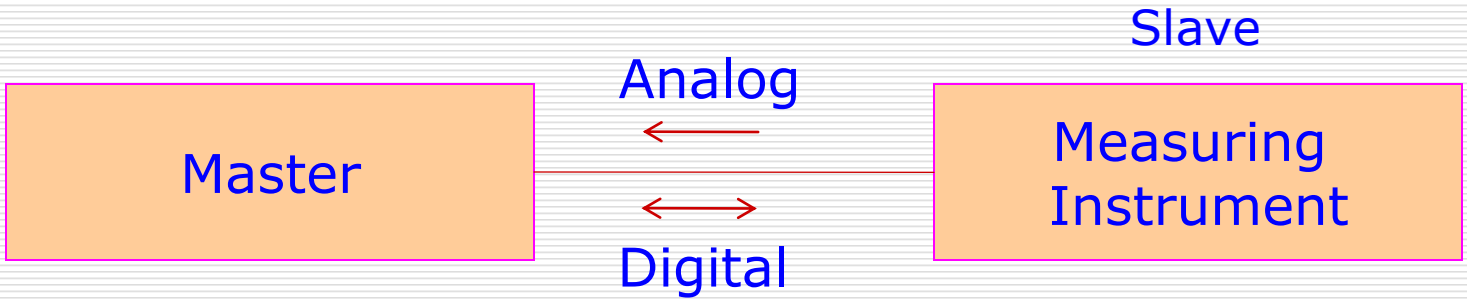
- Both analog and digital communications are supported
- One-to-one communication

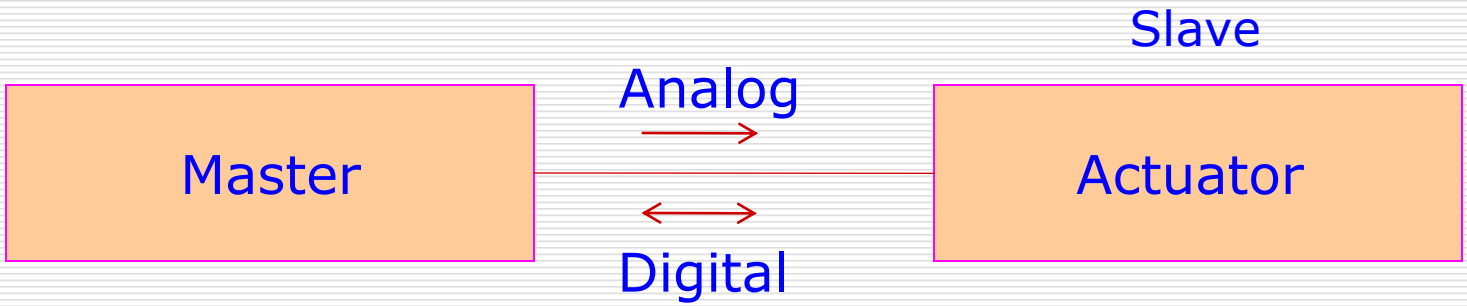
B. Multi-drop or Party-line configuration

- Only digital communication is supported
- Primary Master (e.g. PC, Controller)
- Secondary Master (e.g. Hand-held communicator)
- Slaves (e.g. Field devices)

Point-to-Point Configuration

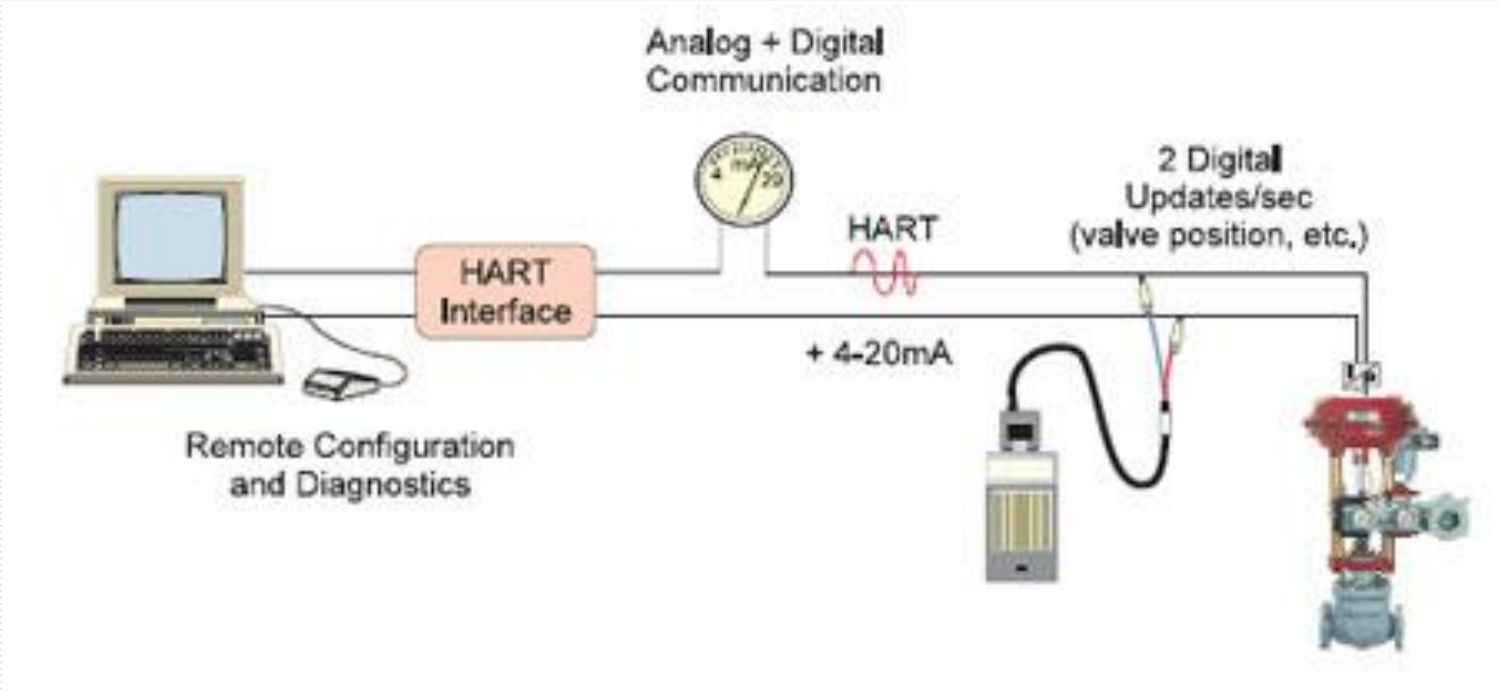
A: Master + Slave





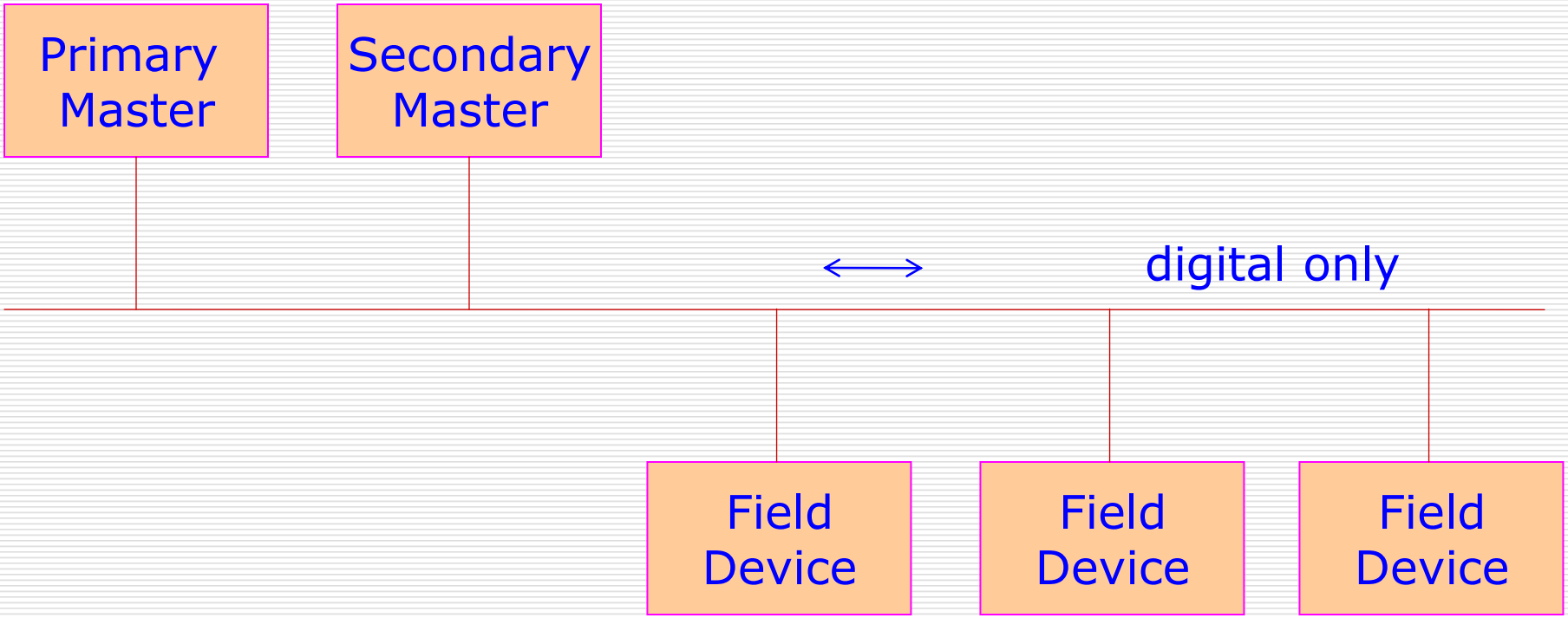
Point-to-Point Communication

B: Primary Master + Secondary Master + Slave



Source: www.smar.com

Multi-Drop or Party-Line Configuration



Digital Communication Modes

A. Normal Mode or
Poll-Response Mode

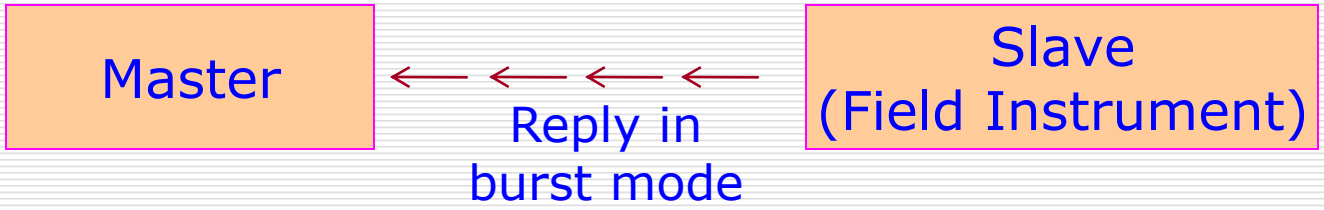
B. Burst Mode

Digital Communication in Normal Mode

- Configuration: Point-to-point or Multi-point
- Master sends command (request), slave responds (replies)
- Typically, 2 responses per second (maximum)

Digital Communication in Burst Mode

- Slave transmits messages continuously (e.g. values of a measurand)
- Used for fast updation of the values of measurands.



HART Command Set

HART command-set comprises three groups of commands:

A. Universal Commands

B. Common-Practice Commands

C. Device-Specific Commands

A. Universal Commands

”The commands that m*ust be recognized and implemented by every HART compliant field device”*

Examples:

- Read manufacturer’s name
- Read model
- Read serial number
- Read range
- Read process variable names
- Read current output

B. Common-Practice Commands

"The commands that are recognized and implemented by most HART-compliant field devices"

Examples:

- Read (upto 4) variables
- Perform calibration check
- Perform self-test
- Write damping time constant
- Write transmitter range
- Set fixed current output

Note: A HART device can handle upto 256 process variables and can communicate upto 4 process variables in one message

C. Device-Specific Commands

"The commands that are specific to a HART-compliant field device"

Examples:

- Start, stop or clear (totalizer)
- Select ^{0}C/^{0}F (temperature sensor)
- Read alarm set point (relay)
- Write alarm set point (relay)
- Tune a control parameter (control device)
- Select proportional / PID control (control device)

Benefits of HART Protocol

- Maintains 4-20 mA compatibility while allowing simultaneous digital communication
- Relatively easy to understand & use
- Allows manufacturers to add special features to existing field instrument designs
- HART devices can be added incrementally
- "No-risk" solution as integrity of 4-20 mA signal is not disturbed
- Saving in installation cost due to multi-drop capability
- Supported by large number of device/system manufacturers

Limitations of HART Protocol

- Applicable to 4-20 mA devices only
- Only for sensors/measuring instruments and actuators
- Multiple controllers not possible
- Wireless alternative not possible