Lecture 7

Transmission Modes,

and

Multiplexing

Topics in This Lecture

- Transmission modes
  - Serial vs parallel
- **little- and big-endian**
- **Asynchronous** transmission
- **synchronous** transmission
- Communication **modes**
- Modulation and de-modulation
- **Multiplexing/demultiplexing**

- Chapters 9, 11

Transmission Modes

- Definition: A *transmission mode* is the manner in which data is sent over the underlying medium
- Transmission modes can be divided into two fundamental categories:
- Serial — one bit is sent at a time
  - Serial transmission is further categorized according to timing of transmissions
- Parallel — multiple bits are sent at the same time

Parallel Transmission

- Parallel transmission allows transfers of multiple data bits at the same time over separate media
- In general, parallel transmission is used with a wired medium that uses multiple, independent wires
- Furthermore, the signals on all wires are synchronized so that a bit travels across each of the wires at precisely the same time
- Engineers use the term parallel to characterize the wiring

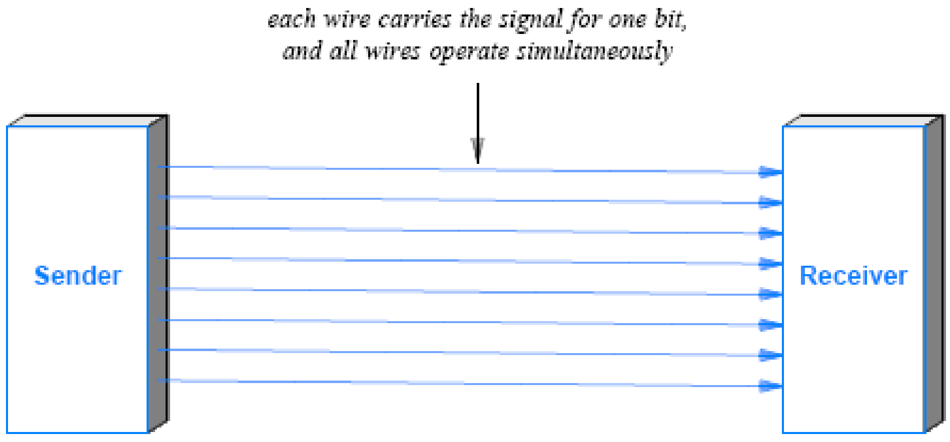


Figure 9.2 Illustration of parallel transmission that uses 8 wires to send 8 bits at the same time.

Parallel Transmission

- The figure omits two important details:

  (1) In addition to the parallel wires that each carry data, a parallel interface usually contains other wires that allow the sender and receiver to coordinate (control circuits)

  (2) To make installation and troubleshooting easy, the wires for a parallel transmission system are placed in a single physical cable

- A parallel mode of transmission has two chief advantages:

  (1) High speed: it can send N bits at the same time

  - a parallel interface can operate N times faster than an equivalent serial interface

  (2) Match to underlying hardware: Internally, computer and communication hardware uses parallel circuitry

  - a parallel interface matches the internal hardware well

Serial Transmission

- Serial transmission
  - sends one bit at a time
- It may seem that anyone would choose parallel transmission for high speeds
  - However, most communication systems use serial mode
- There are two main reasons

  (1) serial networks can be extended over long distances at less cost

  (2) using only one physical wire means that there is never a timing problem caused by one wire being slightly longer than another
- Sender and receiver must contain hardware that converts data between the parallel form used in the device and the serial form used on the wire

Serial Transmission

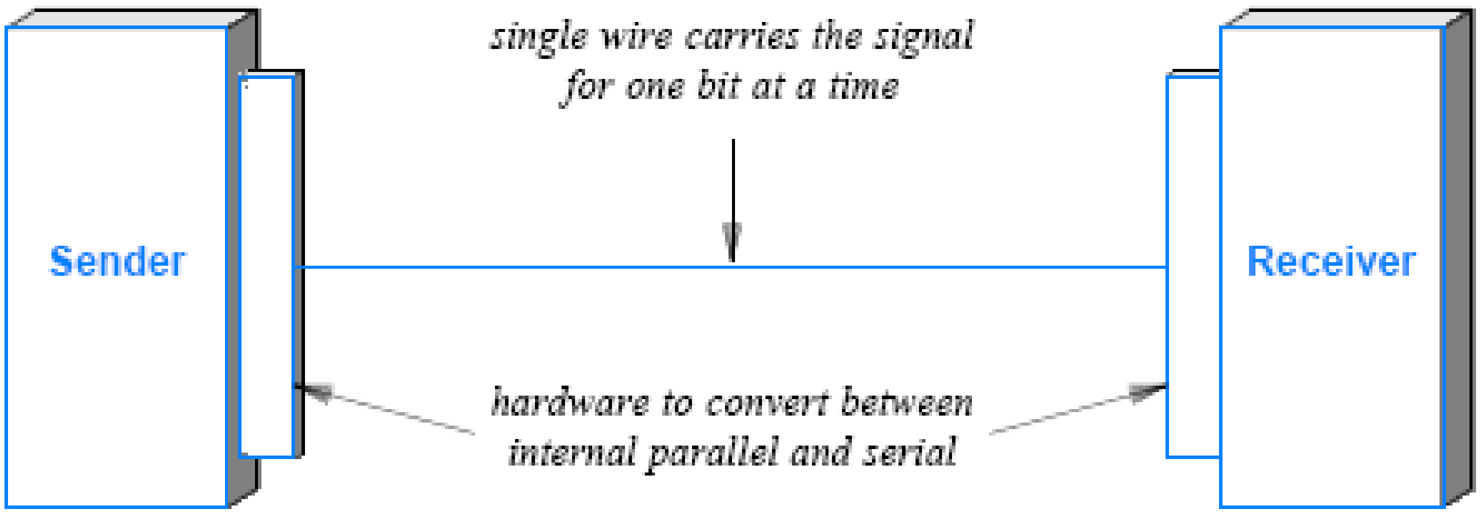


Figure 9.3 Illustration of a serial transmission mode.

Serial Transmission

- The hardware needed to convert data between an internal parallel form and a serial form can be straightforward or complex
- In the simplest case, a single chip that is known as a Universal Asynchronous Receiver and Transmitter (**UART**) performs the conversion
- A related chip, Universal Synchronous-Asynchronous Receiver and Transmitter (**USART**) handles conversion for synchronous network

Transmission Order: Bits and Bytes

- In serial mode, when sending bits, which bit should be sent across the medium first?
- Consider an integer: Should a sender transmit
  - the Most Significant Bit (MSB)
  - or the Least Significant Bit (LSB) first?
- We use the term **little-endian** to describe a system that sends the LSB first
- We use the term **big-endian** to describe a system that sends the MSB first
- Either form can be used, but the sender and receiver must agree

Transmission Order: Bits and Bytes

- The order in which bits are transmitted does not settle the entire question of transmission order
  - Data in a computer is divided into bytes, and each byte is further divided into bits (typically 8 bits per byte)
  - it is possible to choose a byte order and a bit order independently
  - For example, Ethernet technology specifies that data is sent byte big-endian and bit little-endian

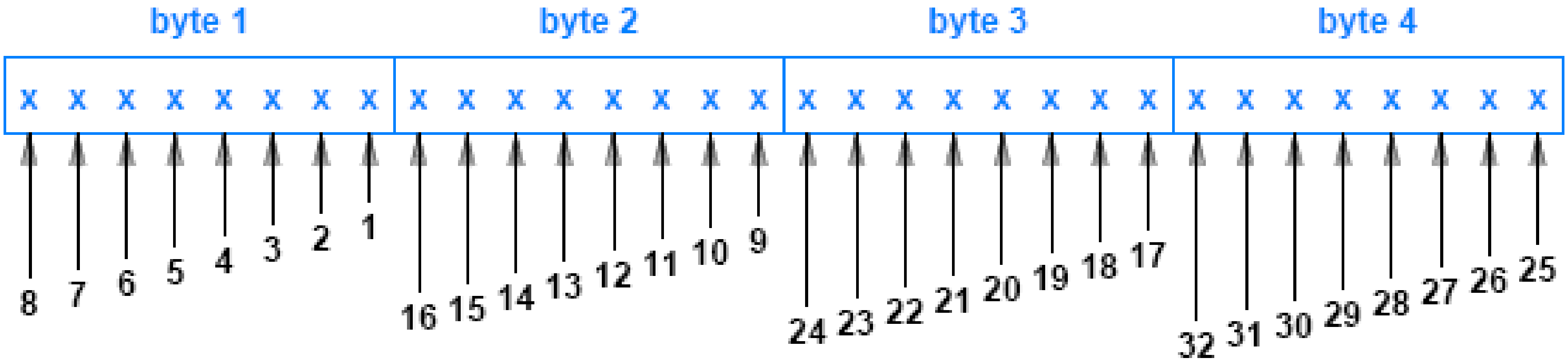


Figure 9.4 Illustration of byte big-endian, bit little-endian order in which the least-significant bit of the most-significant byte is sent first.

Timing of Serial Transmission

- Serial transmission mechanisms can be divided into three broad categories (depending on how transmissions are spaced in time):
- Asynchronous transmission can occur at any time
  - with an arbitrary delay between the transmission of two data items
- Synchronous transmission occurs continuously
  - with no gap between the transmission of two data items

Asynchronous Transmission

- It is asynchronous if the system allows the physical medium to be idle for an arbitrary time between two transmissions
- The asynchronous style of communication is well-suited to applications that generate data at random
  - (e.g., a user typing on a keyboard or a user that clicks on a link)

Asynchronous Transmission

- The disadvantage of asynchrony arises from the lack of coordination between sender and receiver
  - While the medium is idle, a receiver cannot know how long the medium will remain idle before more data arrives
- Asynchronous technologies usually arrange for a sender to transmit a few extra bits for each data item
  - to inform the receiver that a data transfer is starting
  - extra bits allow the receiver to synchronize with the incoming signal
  - the extra bits are overhead

RS-232 Asynchronous Character Transmission

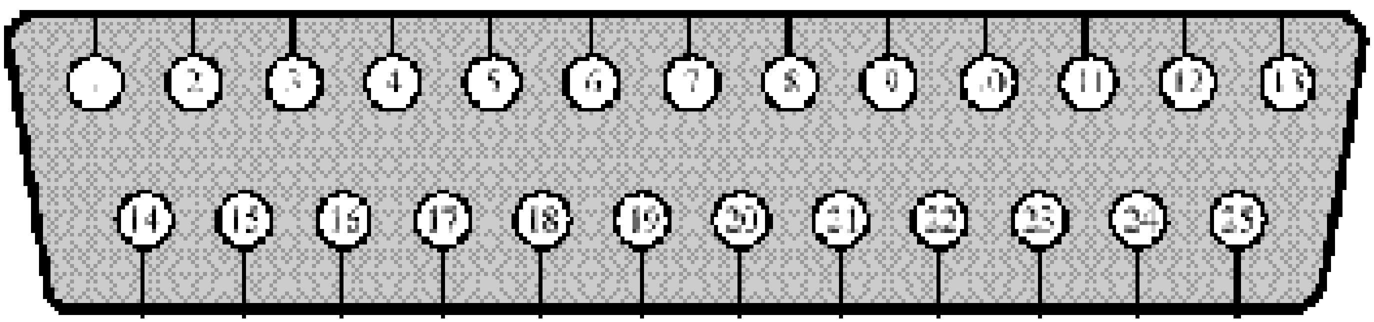
- It is standardized by the Electronic Industries Alliance (EIA)
  - It has become the most widely used for character communication
  - Known as **RS-232-C**, and commonly abbreviated RS-232
- EIA standard specifies the details, such as
  – physical connection size (max cable length 50 feet long)
  – electrical details (range between -15v +15v)
  – the line coding being used
  – It can be configured to control the exact number of bits per second
  – It can be configured to send 7-bit or 8-bit characters

RS-232

- Standard
  - specifies mechanical/electrical/functional details
- RS-232
  - Connecting DTE and DCE
    - Data terminal equipment, *computer for example*
    - Data circuit-terminating equipment, *modem for example*
  - **Serial**: one bit by one bit
  - Asynchronous: sender and receiver do not synchronize before transmission
- Used in connection to keyboard/mouse, modem, COM port
  - Low speed data communication short distance (15 meters)

Mechanical Specification

- Dimension and shape
  - 25 pin connector
  - 9 pin connector widely used in PC
  - Some adapters are required in practice



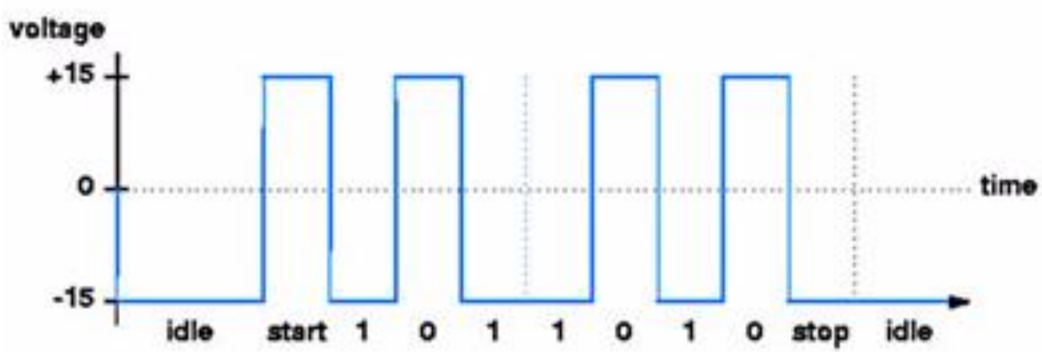
Electrical Specification

- Digital signal encoding
  - Binary 1: a negative voltage in range -15 – -3v
    - Also called “***mark***”, “***off***”
  - Binary 0: a positive voltage in range +3 – +15v
    - Also called “***space***”, “***on***”
- TTL(Transistor Transistor Logic)
  - Used inside the computer
  - Binary 1: a positive voltage in range 0.8 – 5v
  - Binary 0: a positive voltage in range 0 – 0.8v
- Conversion required between TTL and RS-232

Asynchronous Communication

- Asynchronous
  - Sender and receiver do not synchronize before each transmission
- Sender and receiver must agree on
  - Number of bits per character (or byte)
  - Duration of each bit (data rate)
- Receiver
  - Does not know when a character will arrive
  - May wait forever
- To ensure meaningful exchange
  - Start bit before character, used by receiver for bit/byte synchronization
  - Bit by bit to send all bits for a character
  - One or more stop bits after character
- Word format
  - 7-N-1   7 data bits per character, no parity, 1 stop bit

Illustration of RS-232



- Start bit
  - Same as *0*
  - Not part of data
- data bits
  - Number of bits for a character
  - LSB is sent first, MSB bit is sent last
- Parity check bit
  - Optional, for error detection
- Stop bits (can be 1, 1.5, or 2 bits)
  - Same as *1*
  - Follows data

- Word format

  7-N-1 7 data bits, no parity, 1 stop bit

  Example: '-'

- Questions
  - Explain 8-N-1, 7-E-1, 7-N-2
  - Draw wave diagram for '5' in 7-E-1 format (ASCII code is 35h)

Functional Specification

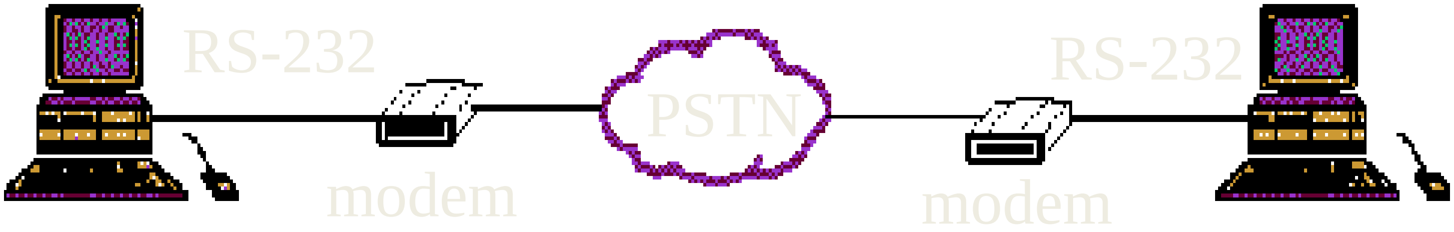
- Function and meaning of each circuit
- Main circuits in RS-232
  - TxD (Tranasmit data) DTE $\rightarrow$ DCE
    - It is in Mark state (off) when no data is available
  - RxD (Receive data) DTE $\leftarrow$ DCE
  - DTR (Data terminal ready) DTE $\rightarrow$ DCE
    - DTE tells it is alive and well, it is set to be on when power is on
  - DSR (Data set ready) DTE $\leftarrow$ DCE
    - DCE tells it is alive and well, it is set to be on when power is on
  - RTS (Request to send) DTE $\rightarrow$ DCE
    - DTE tells DCE that it has data to send by setting it to be on
  - CTS (Clear to send) DTE $\leftarrow$ DCE
    - DCE tells DTE it is ready to accept data by setting it to be on

Main Circuits (continued)

- **DCD** (Data Carrier Detect) DTE ← DCE
  - DCE tells DTE that it detects carrier
- **RI (** Ring Indicator) DTE ← DCE
- **SIG** (Signal Ground)
- **GND**(Protection Ground) or SHD
- Circuits in 9-pin connector
  - 1 DCD 2 RxD 3 TxD 4 DTR 5 SIG
  - 6 DSR 7 RTS 8 CTS 9 RI

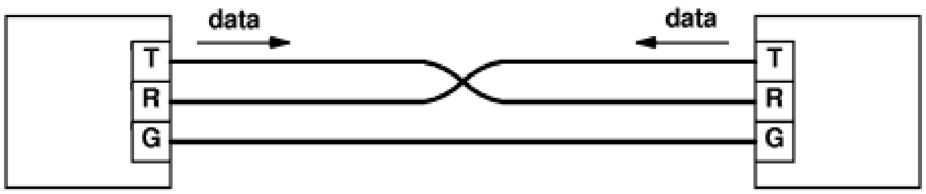
Connecting Two Computers

- **Via public switched telephone network**
  - Modem is required
- **Direct connection**
  - **Null modem is used**



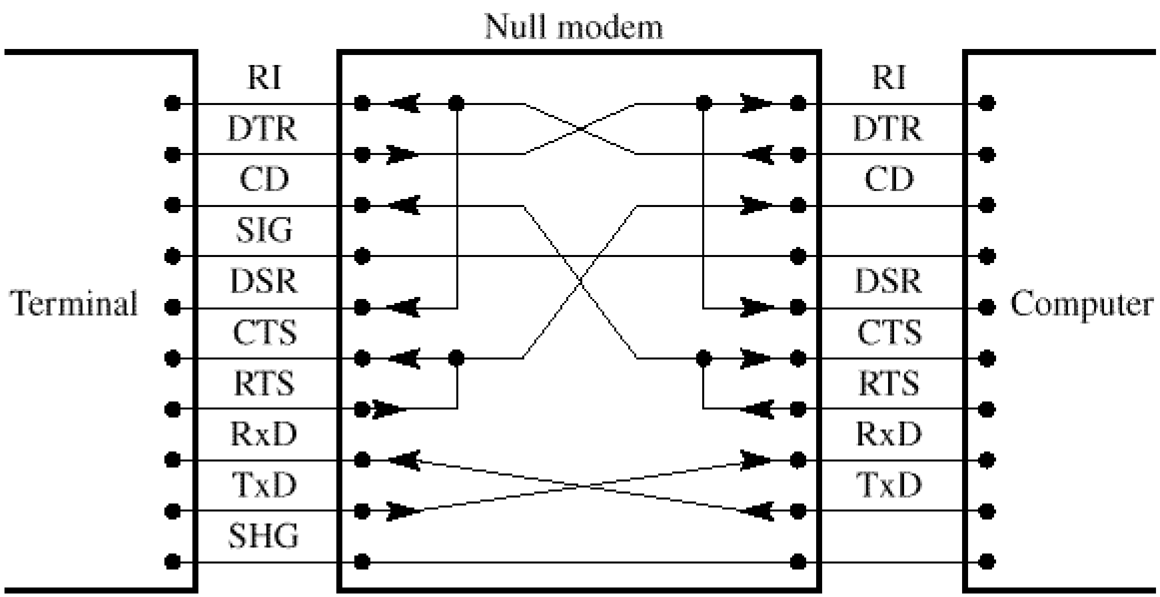
Null modem (1)

- Allow two DTEs communicate directly
- **TxD and RxD must be swapped**
- **One simple example:**



Null modem (2)

- Another null modem example
  - **More circuits involved**
- More on nullmodem or other cables
  - http://www.nullmodem.com/



Flow Control

- Local Flow Control
  - Avoid overflow of data
  - Control data flow between DTE and DCE
- XON/XOFF or software flow control
  - send XOFF to tell the other to stop sending
  - send XON to tell the other to resume sending
  - XON ASCII value 17, XOFF 19
  - used for text communication
- Hardware Flow Control
  - Usually use RTS and CTS to do local flow control
- if possible, choose hardware flow control

Examples

- Draw the waveform diagram that results when the "HKr" is sent in ASCII across an RS-232 connection. Suppose the word format is 7-N-1.

- Suppose one sent 10000 7-bit characters across an RS-232 connection that operated at 9600 bps. How long would the thransmission require?

- What is the concept "Null Modem" ?
- List main circuits found in the RS-232 interface.

Synchronous Transmission

- A synchronous mechanism transmits bits of data continually
  - with no idle time between bits
  - after transmitting the final bit of one data byte, the sender transmits a bit of the next data byte
- The sender and receiver constantly remain synchronized
- Compare the 8-bit characters on
  - an asynchronous system as illustrated in Figure 9.5
  - and a synchronous system as illustrated in Figure 9.6
- Each character sent using RS-232 requires an extra start bit and stop bit
  - meaning that each 8-bit character requires a minimum of 10 bit times, even if no idle time is inserted
- On a synchronous system
  - each character is sent without start or stop bits

Synchronous Transmission

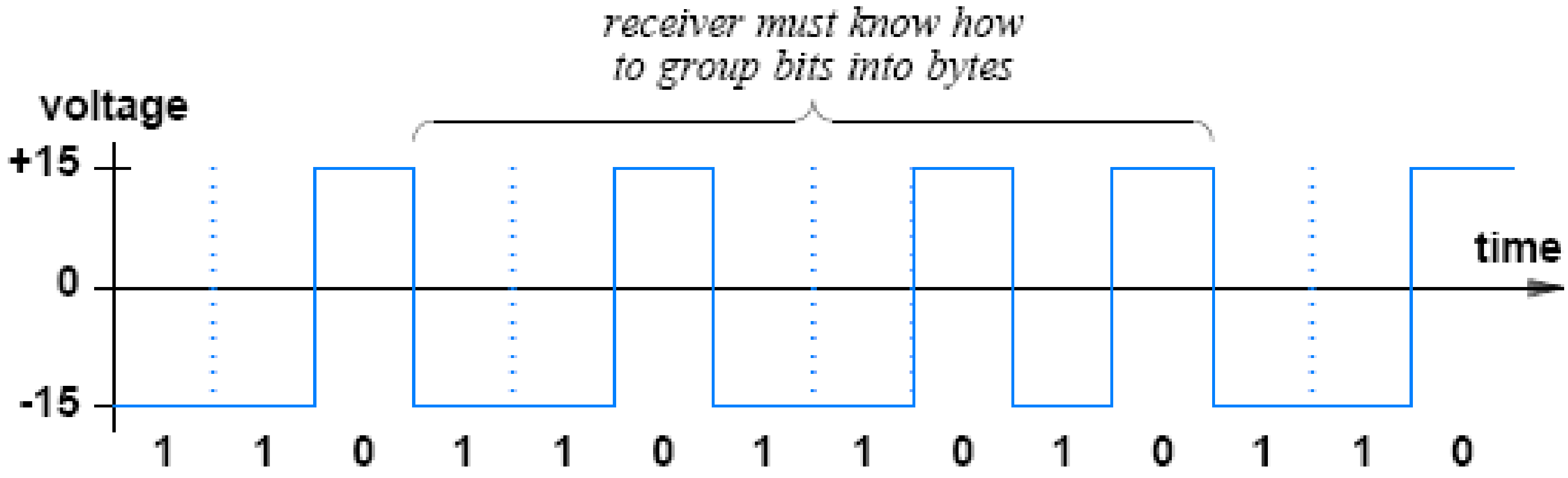


Figure 9.6 Illustration of synchronous transmission where the first bit of a byte immediately follows the last bit of the previous byte.

Bytes, Blocks, and Frames

- If the underlying synchronous mechanism must send bits continually
  - What happens if a sender does not have data ready to send at all times?
  - The answer lies in a technique known as framing:
    - an interface is added to a synchronous mechanism that accepts and delivers a block of bytes known as a frame
  - To ensure that the sender and receiver stay synchronized
    - a frame starts with a special sequence of bits
  - Most synchronous systems include an idle sequence (or idle byte)
    - that is transmitted when the sender has no data to send

Bytes, Blocks, and Frames

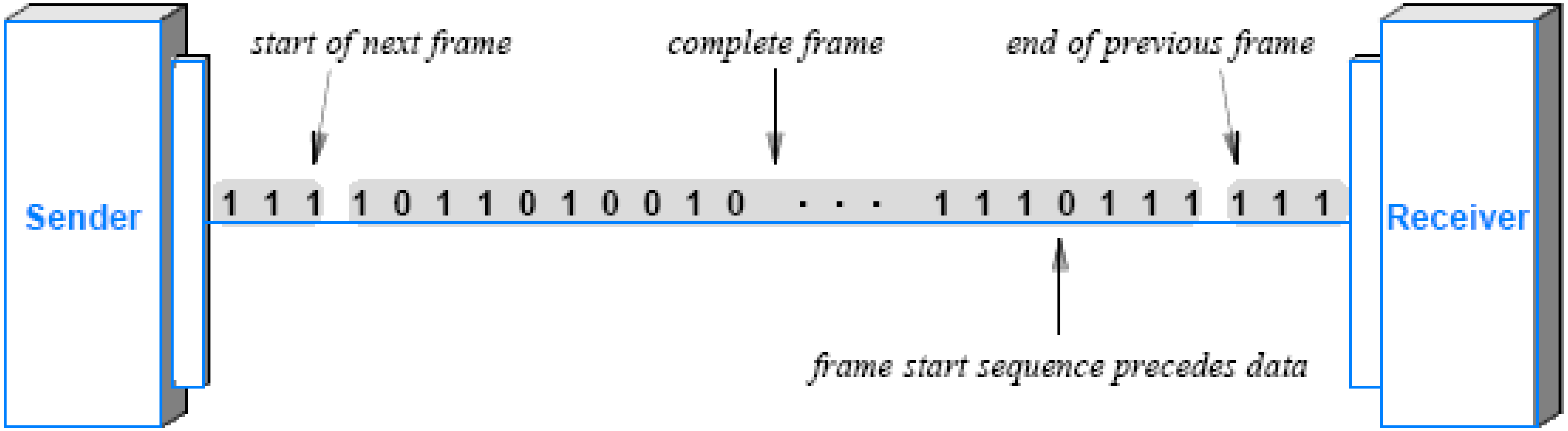


Figure 9.7 Illustration of framing on a synchronous transmission system.

Simplex, Half-Duplex, and Full-Duplex Transmission

- A communications channel is classified as one of three types:
  - Simplex: send or receive; one direction only
  - Full-Duplex: send and receive; two directions
  - Half-Duplex: send and receive; one direction at any time

Simplex, Half-Duplex, and Full-Duplex Transmission

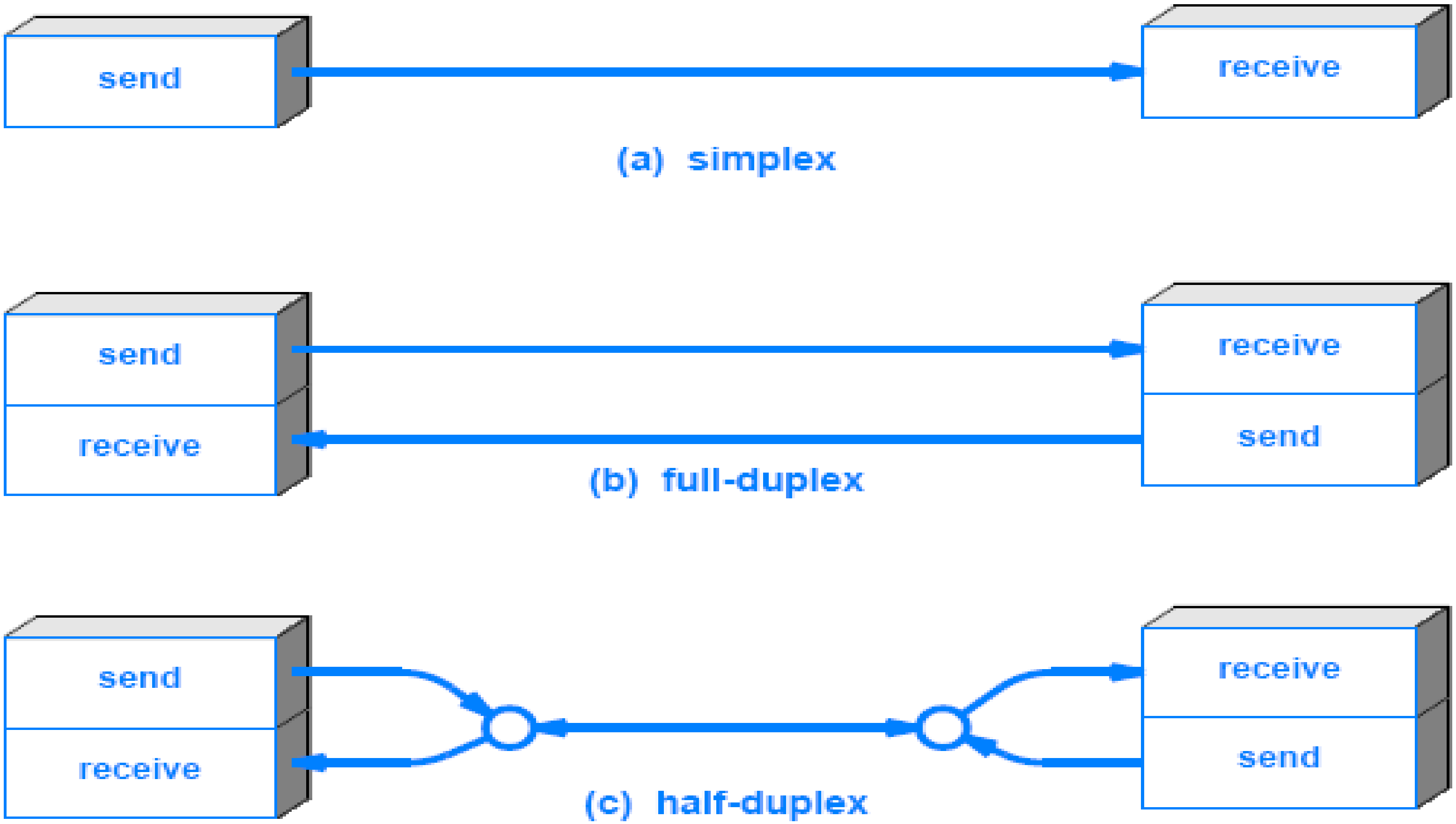


Figure 9.8 Illustration of the three modes of operation.

The Concept of Multiplexing

Definition: Multiplexing refers to the combination of information streams from multiple sources for transmission over a shared medium.

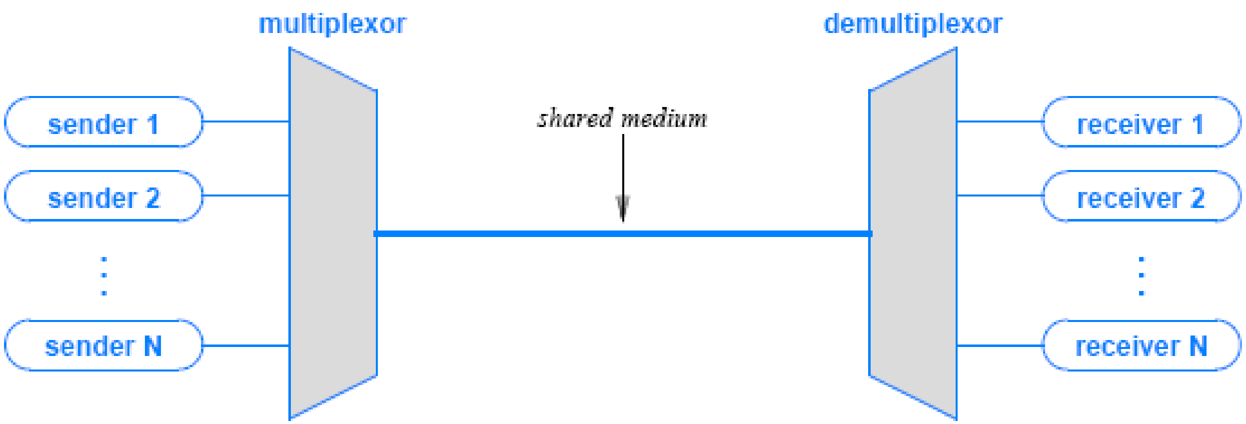


Figure 11.1 The concept of multiplexing in which independent pairs of senders and receivers share a transmission medium.

The Concept of Multiplexing

Definition: Demultiplexing refers to the separation of a combination of information streams back into separate information streams

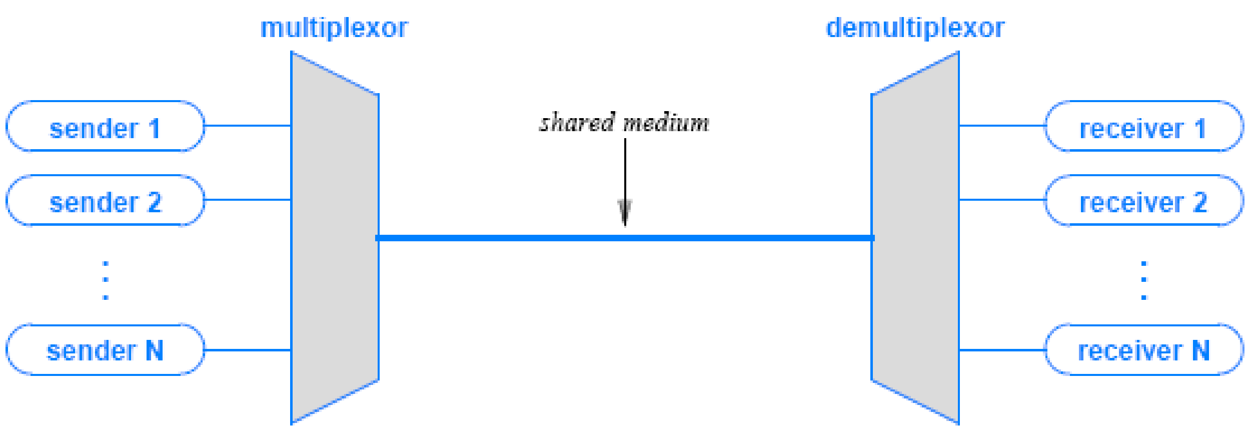


Figure 11.1 The concept of multiplexing in which independent pairs of senders and receivers share a transmission medium.

The Basic Types of Multiplexing

- There are four basic approaches
  - Frequency Division Multiplexing (FDM)
  - Time Division Multiplexing (TDM)
  - Wavelength Division Multiplexing (WDM)
  - Code Division Multiplexing (CDM)
- TDM and FDM are widely used
- WDM is a form of FDM used for optical fiber
- CDM is a mathematical approach used in cell phone mechanisms

Frequency Division Multiplexing

- Bandwidth is divided into N sections, one for each pair of sender and receiver
- Advantage of FDM arises from the simultaneous use of a transmission medium by multiple pairs of entities
- Mainly used in analog communications

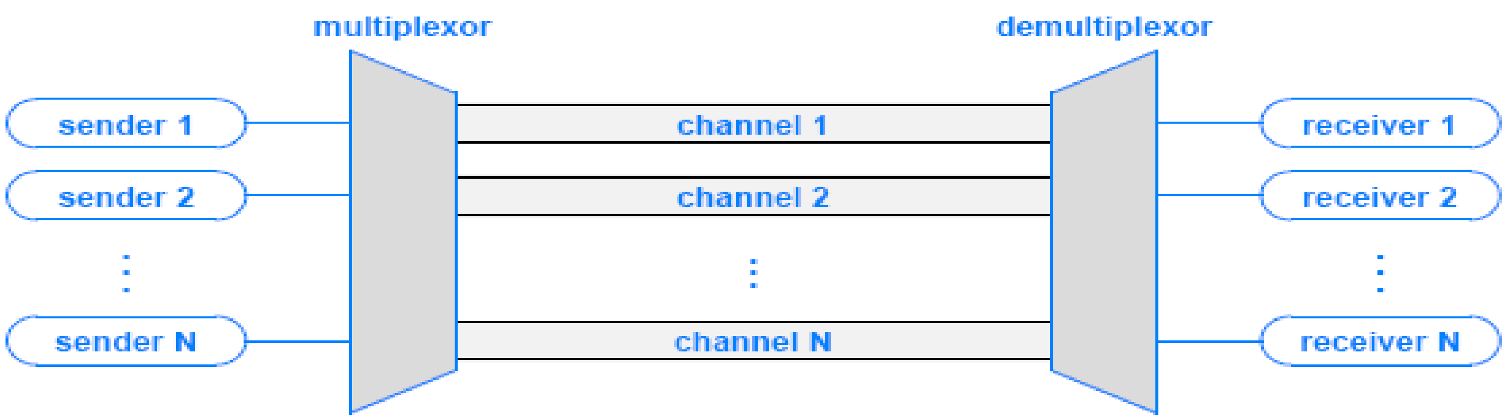


Figure 11.3 The conceptual view of Frequency Division Multiplexing (FDM) as providing a set of independent channels.

FDM example

Channel	Frequencies Used
1	100 KHz - 300 KHz
2	320 KHz - 520 KHz
3	540 KHz - 740 KHz
4	760 KHz - 960 KHz
5	980 KHz - 1180 KHz
6	1200 KHz - 1400 KHz

Figure 11.4 An example assignment of frequencies to channels with a guard band between adjacent channels.

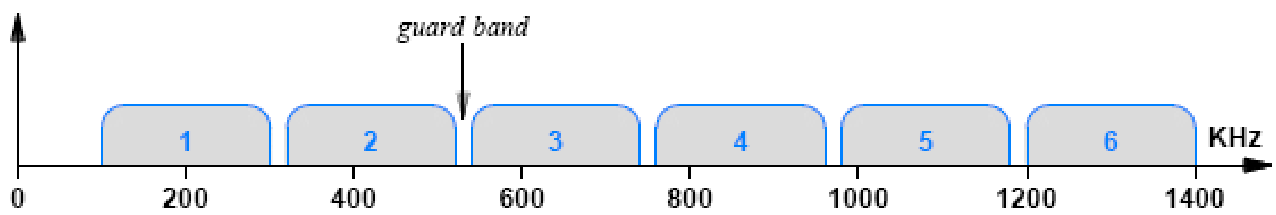
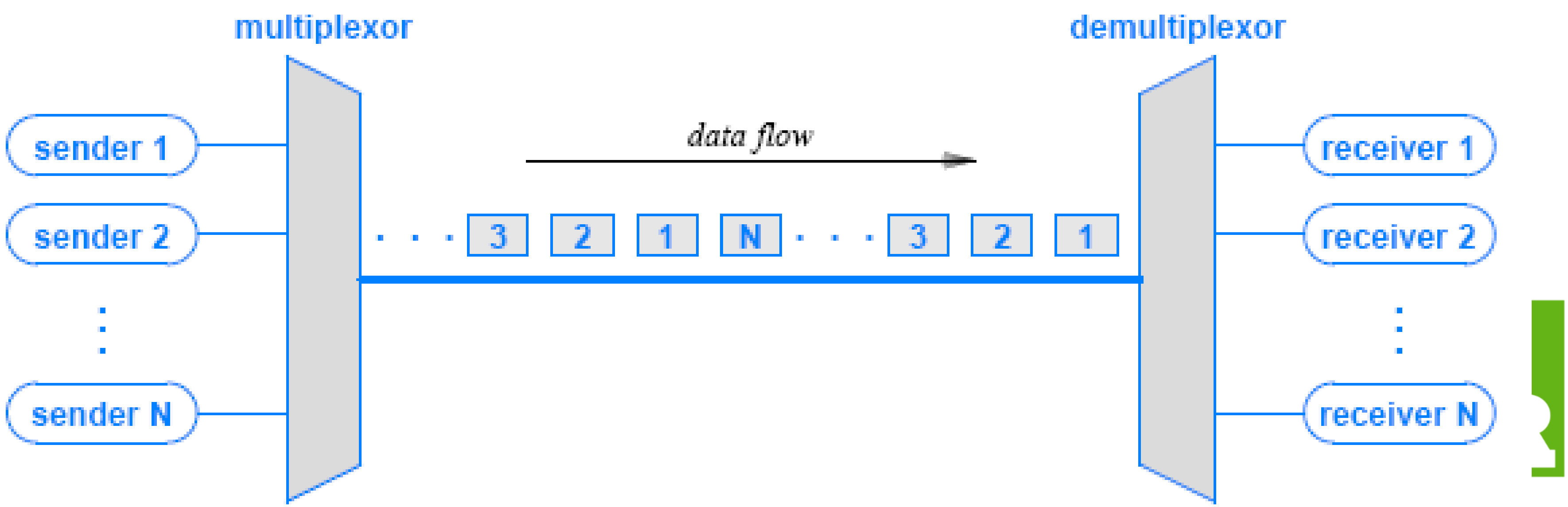


Figure 11.5 A frequency domain plot of the channel allocation from Figure 11.4 with a guard band visible between channels.

Time Division Multiplexing (TDM)

- Time is divided into time slots and each pair of sender and receiver communicate in the allocated slot.
  - Use Round-Robin method
- Used in data communications



Statistical TDM

skip any source that does not have data ready
eliminate unused slots

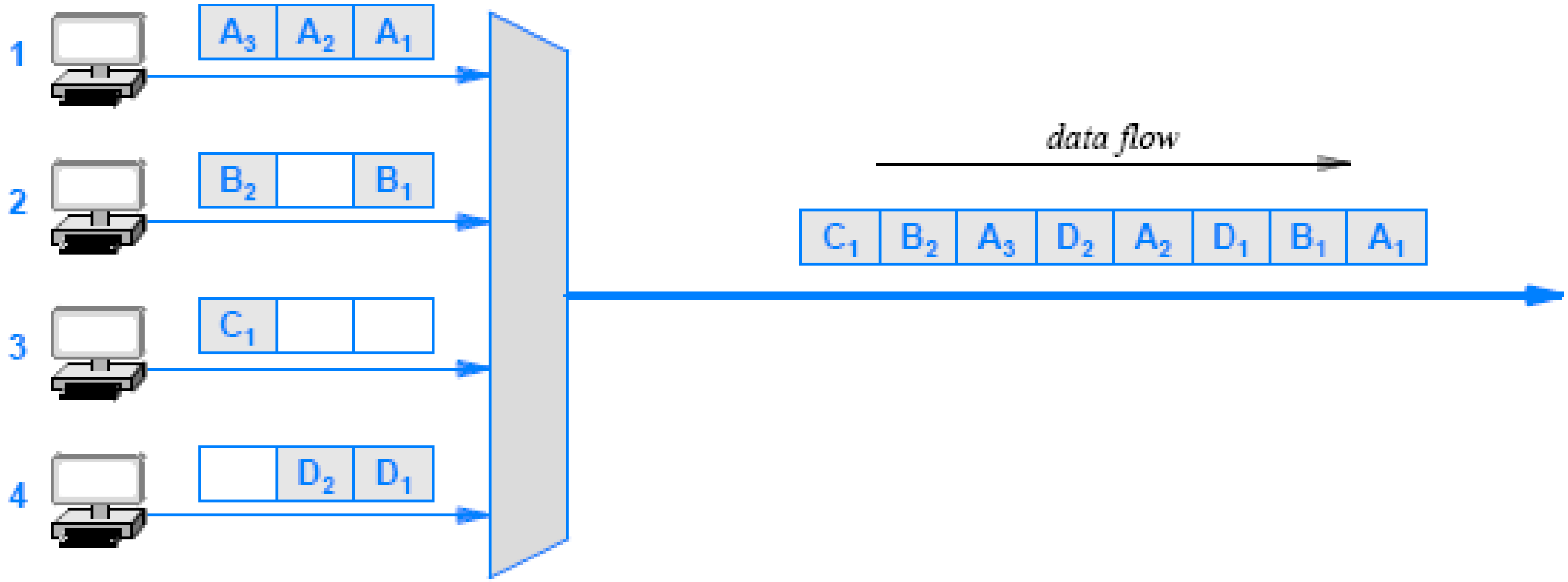


Figure 11.13 Illustration that shows how statistical multiplexing avoids unfilled slots and takes less time to send data.