# Microprocessors & Interfacing

*Serial Input/Output (I)*

Lecturer : Annie Guo

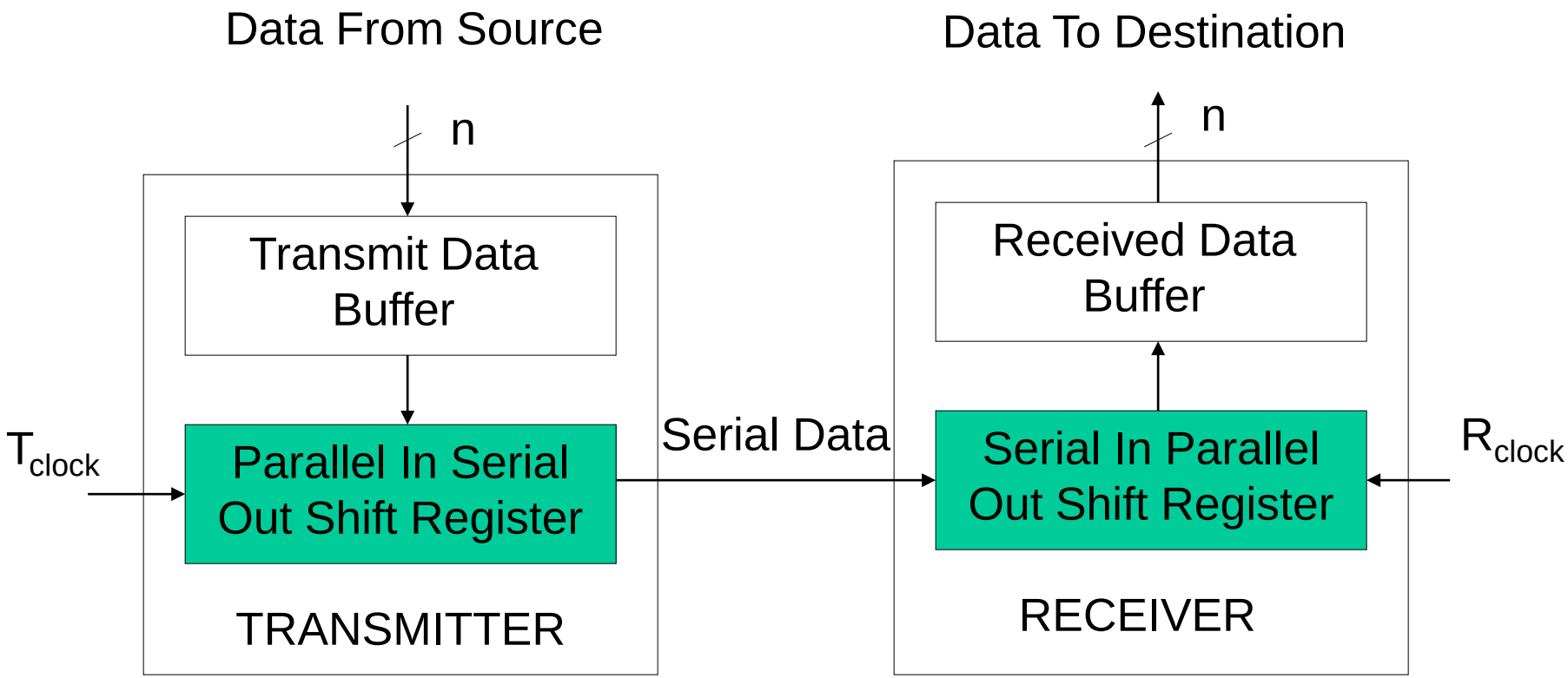
# Lecture Overview

- Serial Communication
  - Concepts
  - Standards*

# Why Serial I/O?

- Problems with Parallel I/O:
  - Needs one wire for each bit.
  - When the source and destination are far from each other the parallel cable can be bulky and expensive.
  - Susceptible to reflections and induced noises for long distance communication.
- Serial I/O overcomes these problems.

# Serial Communication System Structure



# Serial Communication System Structure (cont.)

- At the communication source:
  - The parallel interface transfers data to the transmit data buffer.
  - The data is loaded into the Parallel In Serial Out (PISO) register and $T_{clock}$ shifts the data bits out from the register to the receiver.

# Serial Communication System Structure (cont.)

- At the communication destination:
  - $R_{clock}$ shifts each bit received into the Serial In Parallel Out (SIPO) register.
  - After all data bits have been shifted in, they are transferred to the received data buffer.
  - The data in the received data buffer can be read by an input operation via the parallel interface.

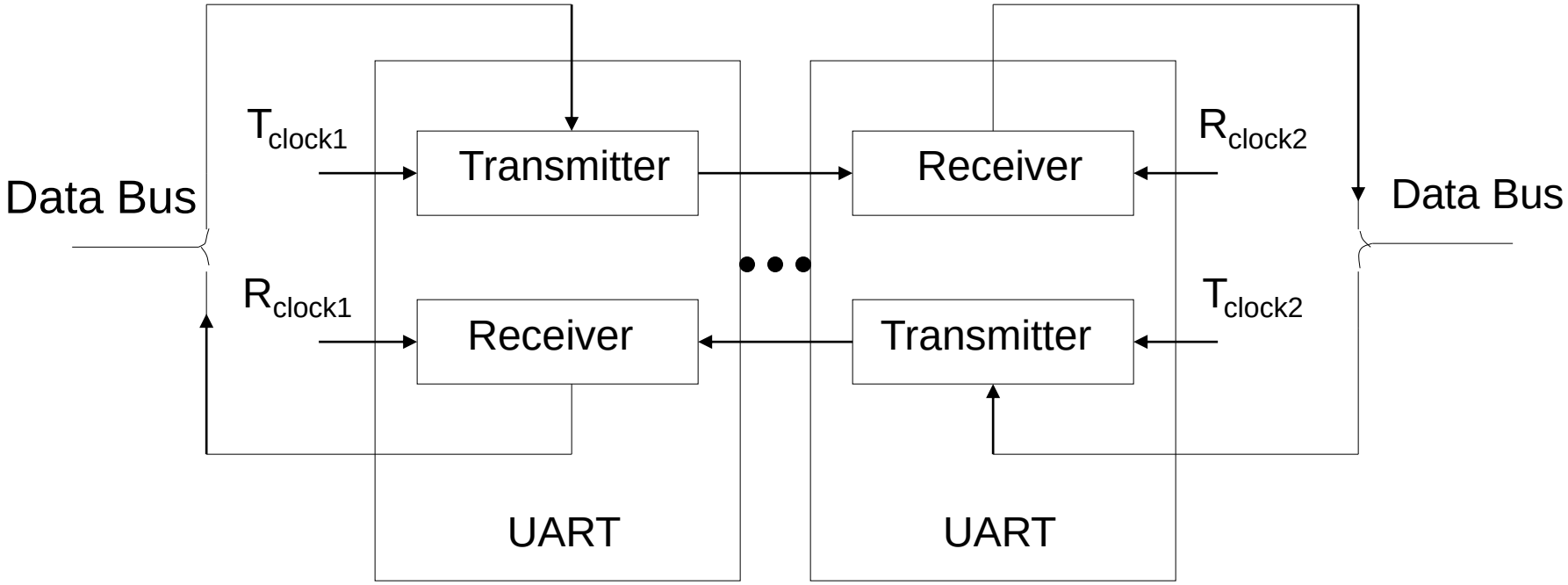
# Synchronous vs Asynchronous Transmission

- Synchronous
  - Transmitter and receiver clocks are synchronized
    - Need extra hardware for clock synchronization
  - Having faster data transfer rate
- Asynchronous
  - Transmitter and receiver use different clocks. No clock synchronization is required.
  - Used in many applications, such as keyboard, mouse and modem.
  - The rest of this lecture mainly focuses on Asynchronous communication

# UART

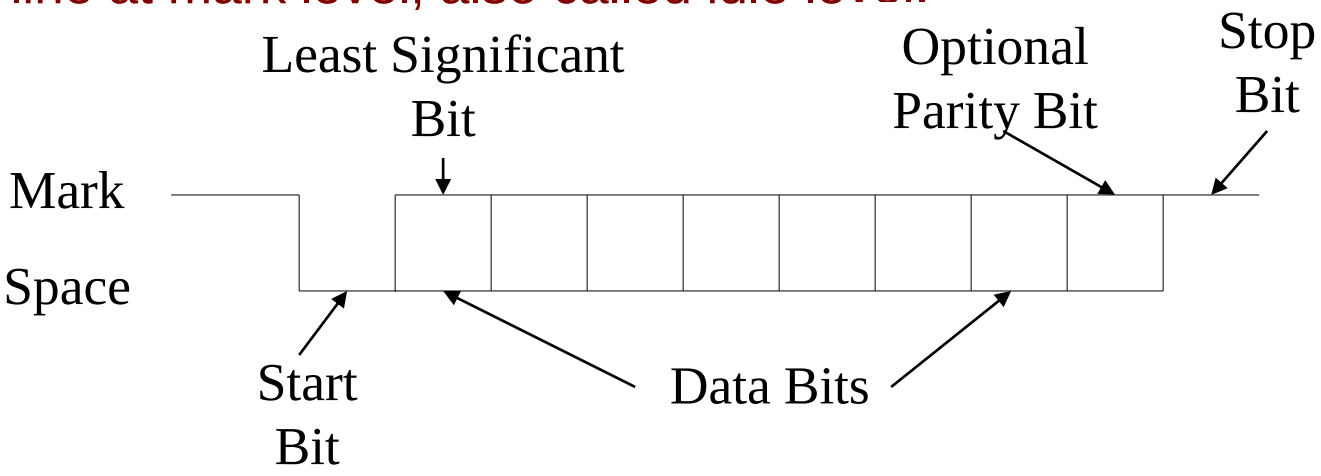
- UART is a basic serial communication hardware
- It has both transmitter and receiver
- Data are transmitted asynchronously
  - Clocks on both sides are not synchronized
  - But the receiver has a way to synchronise the data receiving operation with the data transmission operation
- Hence called UART (Universal Asynchronous Receiver/Transmitter).

# Communication Logic Structure with UART



# UART Data Frame Format

- Before transmission, data should be encoded
  - Use an encoding scheme, such as ASCII
- Each encoded data item is encapsulated with two types of bits
  - Start bit and stop bit
- ***Mark and space***: the logic one and zero levels are, respectively, called mark and space.
  - When the transmitter is not sending anything, it holds the line at mark level, also called idle level.



# UART Data Frame Format (cont.)

- Typical bits in data transmission:
  - Start bit:
    - When the transmitter has data to send, it first changes the line from the mark to the space level for one-bit time. This is to synchronise the receiver with the transmitter. When the receiver detects the start bit, it starts to clock in the serial data bits.
  - Data bits:
    - representing a data item, such as a character

# UART Data Frame Format (cont.)

- Typical bits in data transmission:
  - Parity bit: used to detect errors in the data
    - For odd parity: this bit added to the data to make the total number of 1s in the data odd
    - For even parity: this bit added to the data to make the total number of 1s in the data even.
    - E.g.

      ‘9’ → ASCII: 0x39=0b00111001
      P_even = 0
      P_odd = 1
  - Stop bit: added at the end of data frame.
    - It separates successive data transmissions.
    - Some systems require more than one stop bit.

# Data Transmission Rate

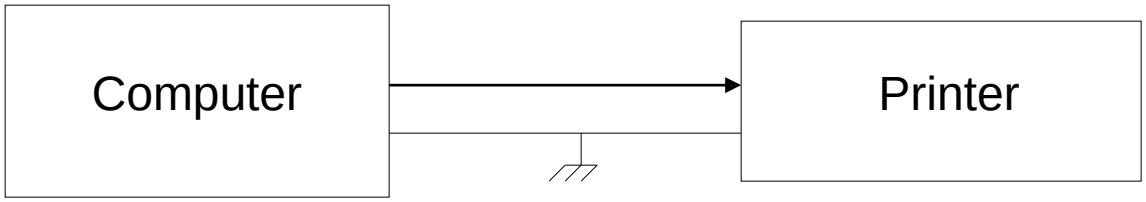
- The rate at which bits are transmitted, also called ***baud rate***, measured in *bits per second.*

# Communication Connection Types

- Three serial communication connection types:
  - Simplex
  - Full-duplex (FDX)
  - Half-duplex (HDX)

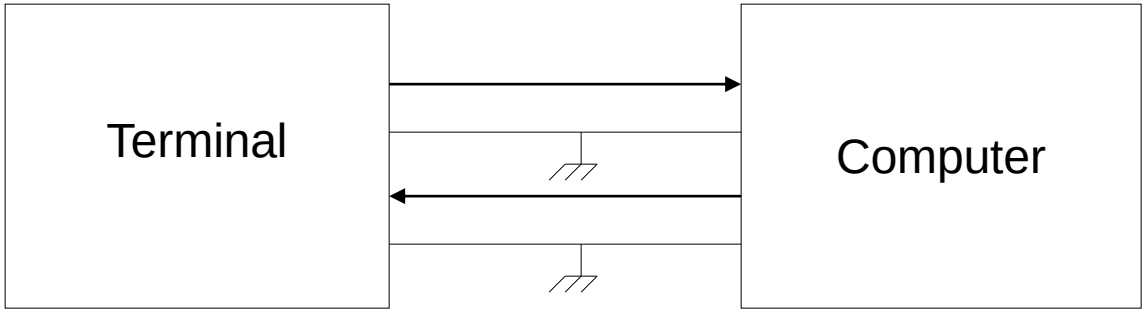
# Simplex Connection

- Data are sent in one direction only
  - For example, computer to a serial printer.
- Simple
  - If the sender does not send data faster than the receiver can accept it, no handshaking signals are required.



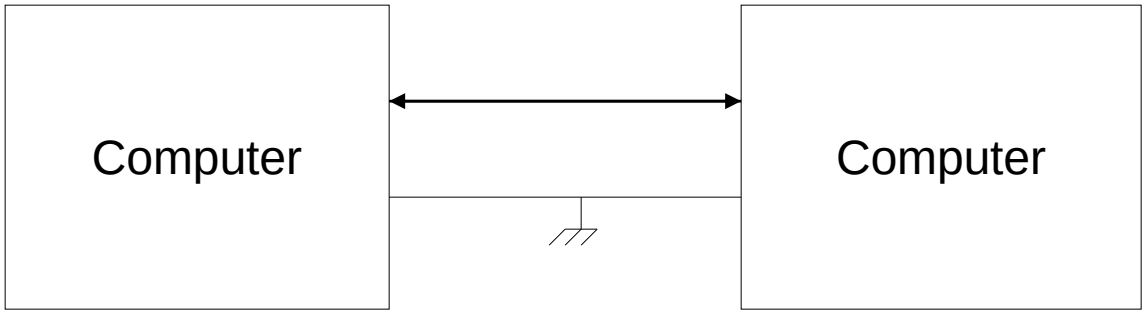
# Full-Duplex (FDX) Connection

- Data are transmitted in two directions, each with a separate data line.



# Half-Duplex (HDX) Connection

- Data are transmitted in two directions with only one data line.

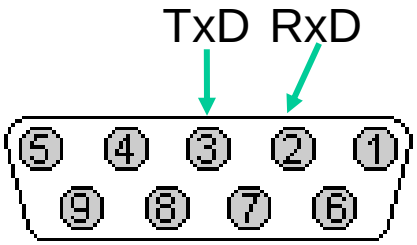


# Standards for the Serial I/O Interface*

- Interface standards are needed to allow different manufacturers' equipment to be interconnected
  - E.g. modem to computer
- Must define the following elements:
  - Handshaking signals
  - Direction of data flow
  - Types of communication devices.
  - Physical interface
  - Electrical signal levels.

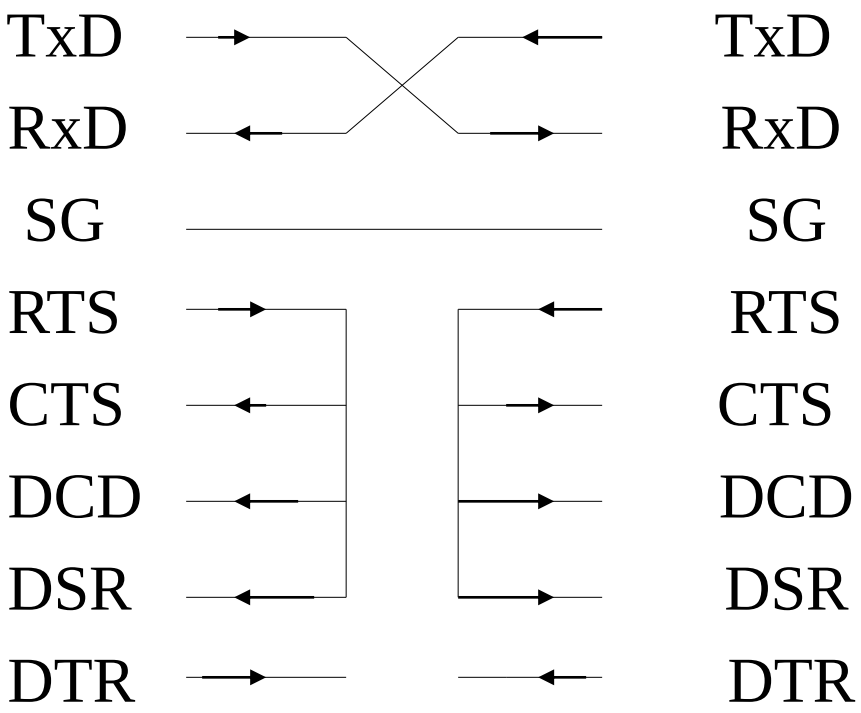
# Standards for the Serial I/O Interface*

- Typical standards include RS-232-C, RS-422, RS-423 and RS-485.
  - RS-232-C standard is used in most serial interface.
    - Standard interconnection cable should be used.
      - E.g. 9-pins
    - Can be in different connection types

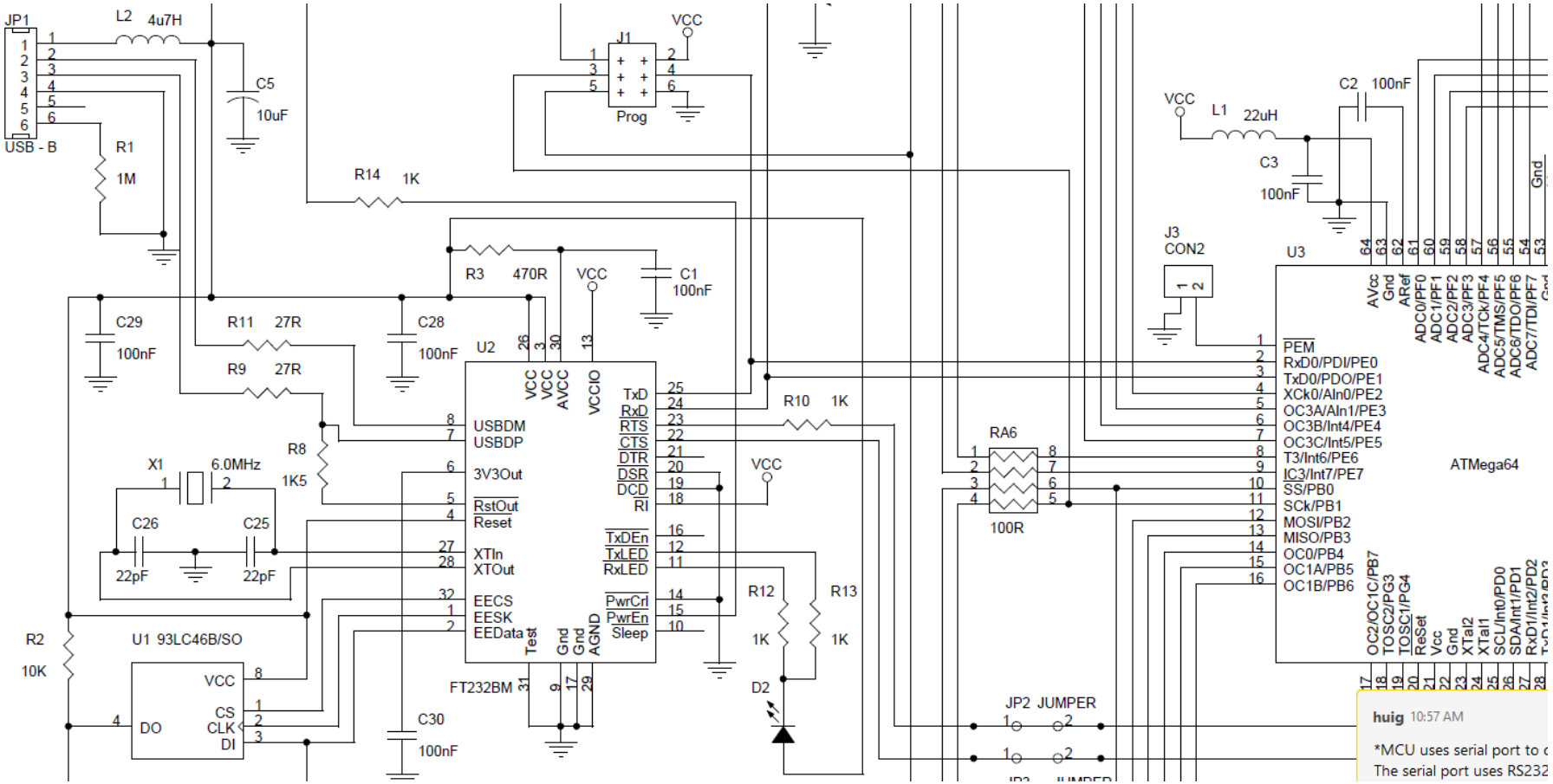


# RS-232-C Interconnection*

- Example: minimal null modem cable
  - Used in our lab board



# Example of UART to USB Connection*



# Reading Material

- Chapter 12: Serial Input/Output. Microcontrollers and Microcomputers by Fredrick M. Cady.
- Mega2560 Data Sheet
  - USART