Introduction to USART

1. USART Definition

USART (Universal Synchronous/Asynchronous Receiver/Transmitter) is a communication protocol used to transfer data between electronic devices, such as microcontrollers, sensors, and other components. This protocol is highly flexible as it supports two main modes:

- **Synchronous**
- **Asynchronous**

2. UART vs. USART

While often used interchangeably, there is a technical difference:

- **UART (Universal Asynchronous Receiver/Transmitter):** Supports only asynchronous communication. It requires no clock signal as it relies on start/stop bits and pre-defined baud rates.
- **USART (Universal Synchronous/Asynchronous Receiver/Transmitter):** A superset of UART. It supports both asynchronous and synchronous modes. In synchronous mode, a dedicated clock pin (XCK) is used to synchronize data.

3. Operating Modes

A. Asynchronous Mode

In this mode, the USART module transmits data without an external clock signal. Synchronization is achieved using data frames consisting of:

- **Start Bit:** Indicates the beginning of transmission.
- **Data Bits:** Contains the primary information (typically 5-9 bits).
- **Parity Bit (Optional):** Used for error detection (Even, Odd, or None).
- **Stop Bit:** Indicates the end of transmission.

> **Characteristics:** Suitable for long-distance communication or between devices that do not share the same clock.

B. Synchronous Mode

Uses a clock signal to synchronize data transfer between the transmitter and the receiver.

- Requires the same clock configuration on both devices.
- Allows for faster and more reliable data transfer compared to asynchronous mode.

> **Characteristics:** Ideal for multimedia applications or high-speed bulk data transfers.

4. Configuration and Baud Rate

To use USART, several parameters must be defined:

1. **Baud Rate:** The speed of data transmission (bits per second).
2. **Data Format:** The number of data bits, parity, and stop bits.
3. **Interrupt:** Enables notifications when data is finished being sent or received.

5. USART and Arduino Serial Monitor

In the Arduino ecosystem (such as the Uno), the USART peripheral is the primary way the microcontroller communicates with your computer:

1. **Hardware Connection:** The ATmega328P uses its USART pins (TX on Pin 1, RX on Pin 0). These are connected to an onboard USB-to-Serial converter chip.
2. **Serial Monitor:** When you open the **Serial Monitor** or **Serial Plotter** in the Arduino IDE, it acts as the "Receiver" (RX) for data sent by the Arduino and the "Transmitter" (TX) for data you type in.
3. **Baud Rate Alignment:** For communication to work, the baud rate selected in the Serial Monitor (e.g., 9600) must match the baud rate configured in your code. If they do not match, you will see "garbage" characters or no data at all.

In Proteus, follow these steps to simulate using a Virtual Terminal to mimic Serial Monitor functionality:

1. Click on **Virtual Instruments Mode** in the left sidebar.
   picture 0
2. Select the **Virtual Terminal** component and place it on the schematic.
   picture 1
3. Add an **Arduino Uno** to your schematic. Then, connect the **TX** and **RX** pins of the Virtual Terminal to the **RX** and **TX** pins of the Arduino Uno respectively.
   picture 2
4. Double-click on the Virtual Terminal to set the **Baud Rate** (e.g., 9600) to match your code.
   picture 3

Baud Rate Calculation Formula (UBRR)

The **UBRR (USART Baud Rate Register)** value is calculated based on the CPU clock frequency (F_CPU) and the desired Baud Rate:

> **UBRR = (F_CPU / (16 * BAUD)) - 1**

Calculation Example:
If F_CPU = 16 MHz and the target BAUD = 9600 bps:

1. UBRR = (16,000,000 / (16 * 9600)) - 1
2. UBRR = 104.16 - 1
3. **UBRR ≈ 103** (Hex: 0x67)

Revision #5
Created 2026-02-09 14:15:12 UTC by BH
Updated 2026-02-09 14:39:37 UTC by BH