UART Interfacing on eYFi-Mega Board

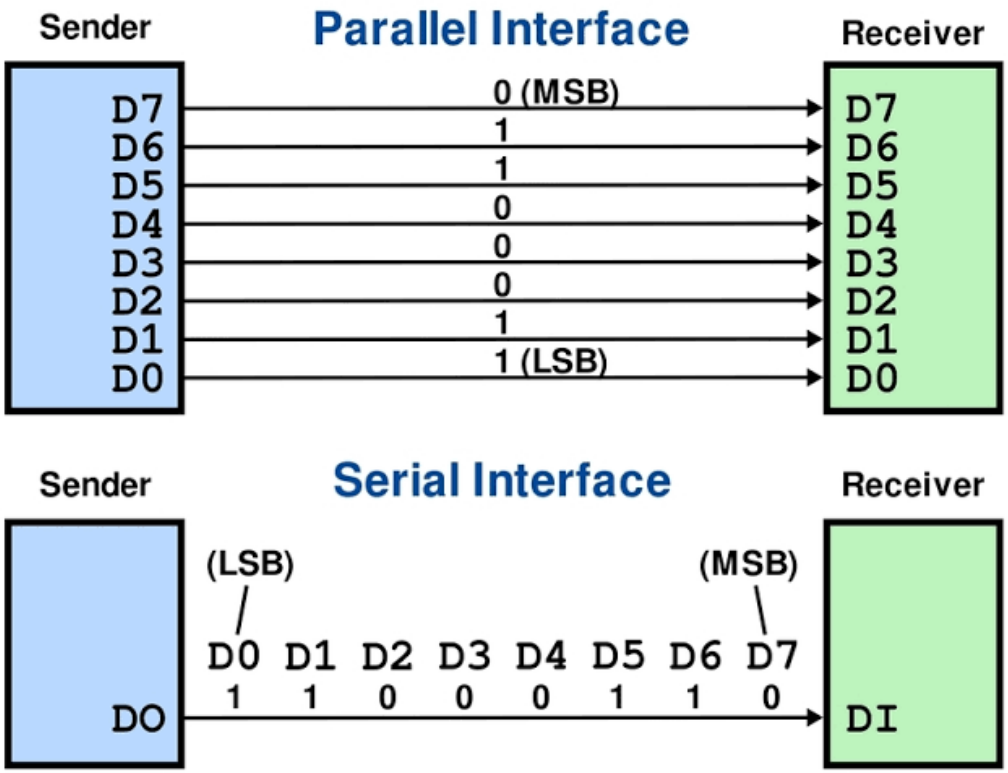
e-Yantra Team

Embedded Real-Time Systems (ERTS) Lab
Indian Institute of Technology, Bombay

IIT Bombay
July 6, 2022

Parallel v/s Serial Interface

Parallel v/s Serial Interface



Motivation for Serial Interface

Motivation for Serial Interface

- Sending data a single bit at a time

Motivation for Serial Interface

- Sending data a single bit at a time
- Why not send data in parallel?
- Why bother with parallel to serial and serial to parallel conversions?

Motivation for Serial Interface

- Sending data a single bit at a time
- Why not send data in parallel?
- Why bother with parallel to serial and serial to parallel conversions?

Several situations where serial interface is preferable:

- Transmitter and receiver are remote

Motivation for Serial Interface

- Sending data a single bit at a time
- Why not send data in parallel?
- Why bother with parallel to serial and serial to parallel conversions?

Several situations where serial interface is preferable:

- Transmitter and receiver are remote
- Difficult to ensure all bits on parallel bus have same delay

Motivation for Serial Interface

- Sending data a single bit at a time
- Why not send data in parallel?
- Why bother with parallel to serial and serial to parallel conversions?

Several situations where serial interface is preferable:

- Transmitter and receiver are remote
- Difficult to ensure all bits on parallel bus have same delay
- Difference in delay comparable to data rate

Motivation for Serial Interface

- Sending data a single bit at a time
- Why not send data in parallel?
- Why bother with parallel to serial and serial to parallel conversions?

Several situations where serial interface is preferable:

- Transmitter and receiver are remote
- Difficult to ensure all bits on parallel bus have same delay
- Difference in delay comparable to data rate

How do we send data serially?

Motivation for Serial Interface

- Sending data a single bit at a time
- Why not send data in parallel?
- Why bother with parallel to serial and serial to parallel conversions?

Several situations where serial interface is preferable:

- Transmitter and receiver are remote
- Difficult to ensure all bits on parallel bus have same delay
- Difference in delay comparable to data rate

How do we send data serially?

Transmitter and Receiver must agree on the order in which the bits will be sent.

Synchronization

- Must match the data out rate (at Tx) and data in rate (at Rx)

Synchronization

- Must match the data out rate (at Tx) and data in rate (at Rx)
- Adjust clock phase at Rx to sample serial line when stable and not when data is changing

Synchronization

- Must match the data out rate (at Tx) and data in rate (at Rx)
- Adjust clock phase at Rx to sample serial line when stable and not when data is changing

How can this be ensured?

Synchronization

- Must match the data out rate (at Tx) and data in rate (at Rx)
- Adjust clock phase at Rx to sample serial line when stable and not when data is changing

How can this be ensured?

- Synchronous communication *OR*

Synchronization

- Must match the data out rate (at Tx) and data in rate (at Rx)
- Adjust clock phase at Rx to sample serial line when stable and not when data is changing

How can this be ensured?

- Synchronous communication *OR*
- Asynchronous communication

Synchronous v/s Asynchronous

Synchronous v/s Asynchronous

Parameters	**Synchronous**	**Asynchronous**
Clock signal	Required	Not required
Overhead bits	Not required	Required
Data transmission speed	Fast	Slow
Data Tx/Rx	Blocks or frames	Bytes or character
Examples	I^2C, SPI	UART

What is UART?

What is UART?

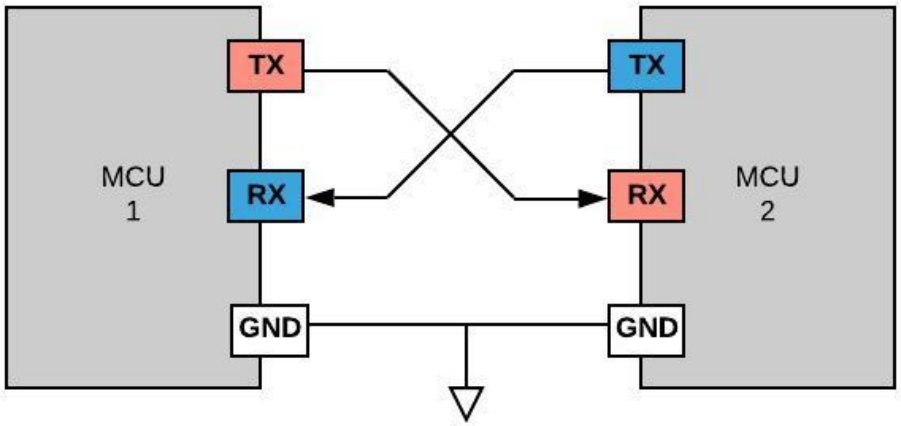
- **Universal Asynchronous Receiver Transmitter (UART)** is used to communicate data between micro-controller and PC or other devices.

What is UART?

- **Universal Asynchronous Receiver Transmitter (UART)** is used to communicate data between micro-controller and PC or other devices.
- UART requires two lines for communication
  1. Receive - RX
  2. Transmit - TX

What is UART?

- **Universal Asynchronous Receiver Transmitter (UART)** is used to communicate data between micro-controller and PC or other devices.
- UART requires two lines for communication
  1. Receive - RX
  2. Transmit - TX



Working of UART

Working of UART

- An external clock signal is not required.

Working of UART

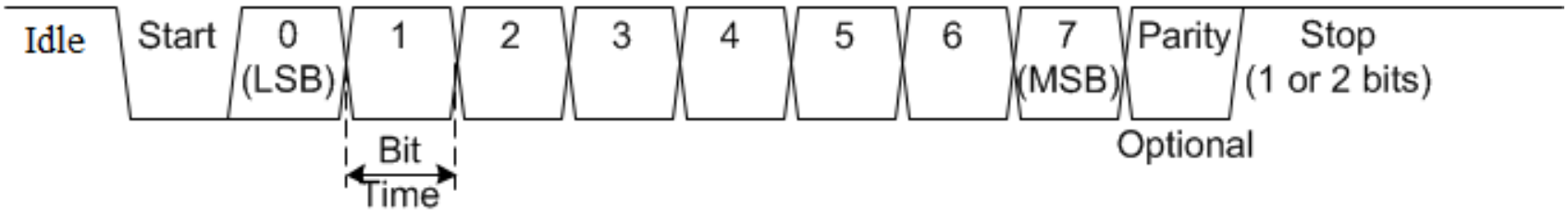
- An external clock signal is not required.
- Extra rules or mechanisms are needed to ensure reliable, error-free sending and receiving of data, which are:

Working of UART

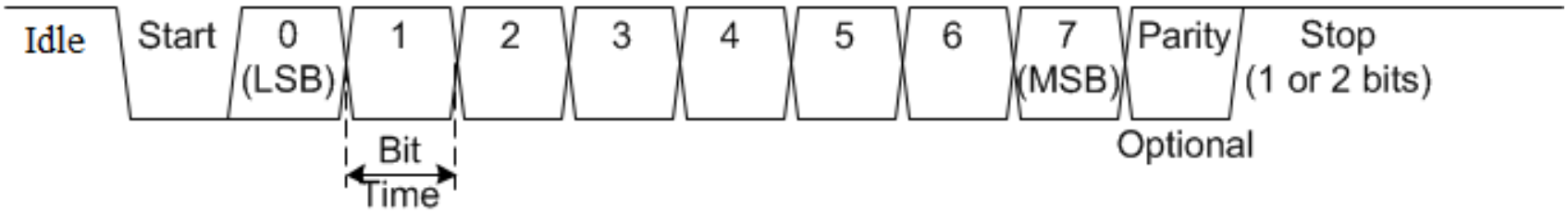
- An external clock signal is not required.
- Extra rules or mechanisms are needed to ensure reliable, error-free sending and receiving of data, which are:
  1. Data Packet
     - Synchronization Bits
     - Data Bits
     - Parity Bits
  2. Baud Rate

Data Packet

Data Packet

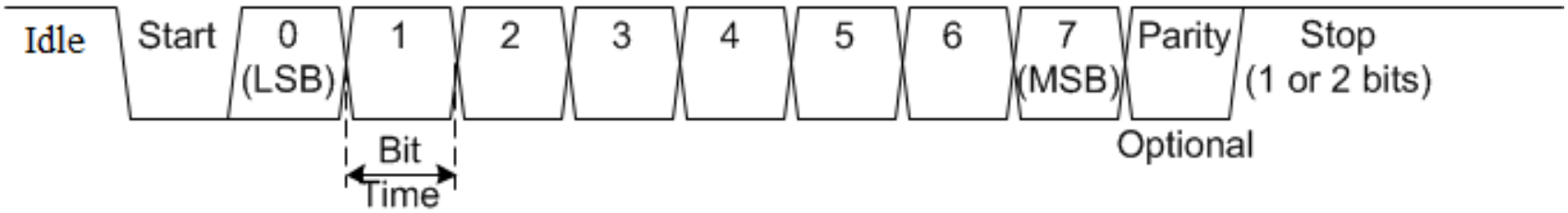

Idle
Start
0
(LSB)
1
2
3
4
5
6
7
(MSB)
Parity
Stop
(1 or 2 bits)
Optional
Bit
Time

Data Packet



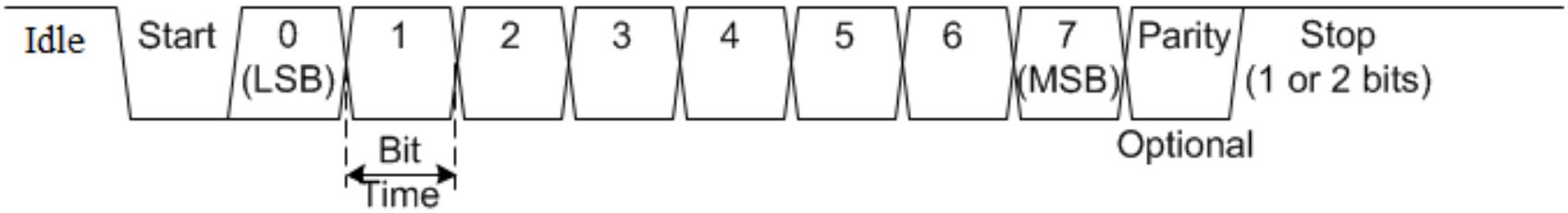
1. Synchronization Bits
   - Start (1 bit) - transition on idle data line from 1 to 0.
   - Stop (1-2 bit/s) - transition back to idle state, holding the line at 1.

Data Packet



1. Synchronization Bits
   - Start (1 bit) - transition on idle data line from 1 to 0.
   - Stop (1-2 bit/s) - transition back to idle state, holding the line at 1.
2. Data Bits
   - Number of data bits
   - Endianness of data bits

Data Packet



1. Synchronization Bits
   - Start (1 bit) - transition on idle data line from 1 to 0.
   - Stop (1-2 bit/s) - transition back to idle state, holding the line at 1.
2. Data Bits
   - Number of data bits
   - Endianness of data bits
3. Parity Bits
   - Low-level and simple form of error checking.
   - It can be odd or even.

Baud Rate

Baud Rate

- It indicates the rate at which data transfer will occur between two or more devices.

Baud Rate

- It indicates the rate at which data transfer will occur between two or more devices.
- The unit is **bits-per-second (bps)**.

Baud Rate

- It indicates the rate at which data transfer will occur between two or more devices.
- The unit is **bits-per-second (bps)**.
- Baud rates can take any value; but devices must agree to operate on same rate, as communication is asynchronous.

Baud Rate

- It indicates the rate at which data transfer will occur between two or more devices.
- The unit is **bits-per-second (bps)**.
- Baud rates can take any value; but devices must agree to operate on same rate, as communication is asynchronous.
- Commonly used baud rates are 1200, 2400, 4800, 9600, 19200, 38400, 57600, and 115200.

Baud Rate

- It indicates the rate at which data transfer will occur between two or more devices.
- The unit is **bits-per-second (bps)**.
- Baud rates can take any value; but devices must agree to operate on same rate, as communication is asynchronous.
- Commonly used baud rates are 1200, 2400, 4800, 9600, 19200, 38400, 57600, and 115200.
- These baud rates are achieved in micro-controller by dividing the clock frequency.

An Example

An Example

- Send the data "**Hi**" with UART configuration **9600-8N1**. Determine the number of packets transferred per second.

An Example

- Send the data "**Hi**" with UART configuration **9600-8N1**. Determine the number of packets transferred per second.
  1. **9600** ⇒ Baud Rate
  2. **8** ⇒ Number of data bits in a frame
  3. **N** ⇒ No Parity bits
  4. **1** ⇒ 1 Stop bit
  5. "**H**" ⇒ ASCII value = **0b01001000**
  6. "**i**" ⇒ ASCII value = **0b01101001**
  7. **Endianness** ⇒ Little-endian, by default

An Example

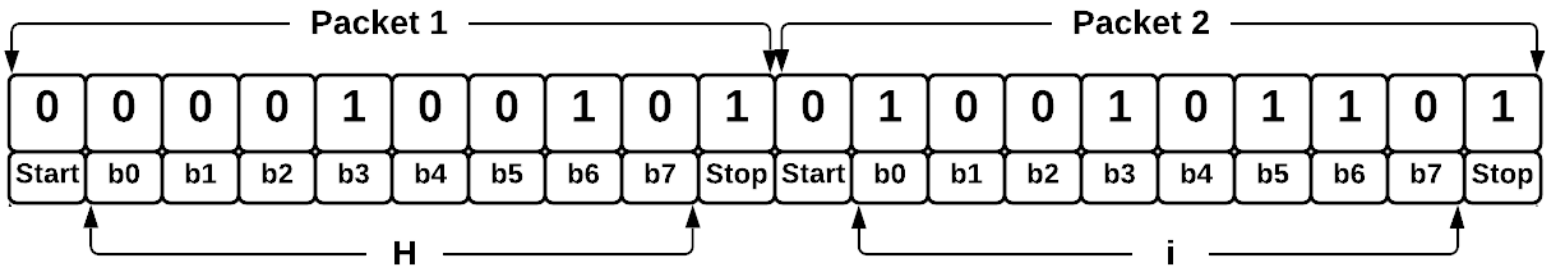
- Send the data "**Hi**" with UART configuration **9600-8N1**. Determine the number of packets transferred per second.

1. **9600** ⇒ Baud Rate
2. **8** ⇒ Number of data bits in a frame
3. **N** ⇒ No Parity bits
4. **1** ⇒ 1 Stop bit
5. "**H**" ⇒ ASCII value = **0b01001000**
6. "**i**" ⇒ ASCII value = **0b01101001**
7. **Endianness** ⇒ Little-endian, by default
8. **10 bits** per packet (1-Start, 8-Data and 1-Stop)
9. With Baud Rate = 9600 bps, **960 packets** are sent per sec

An Example

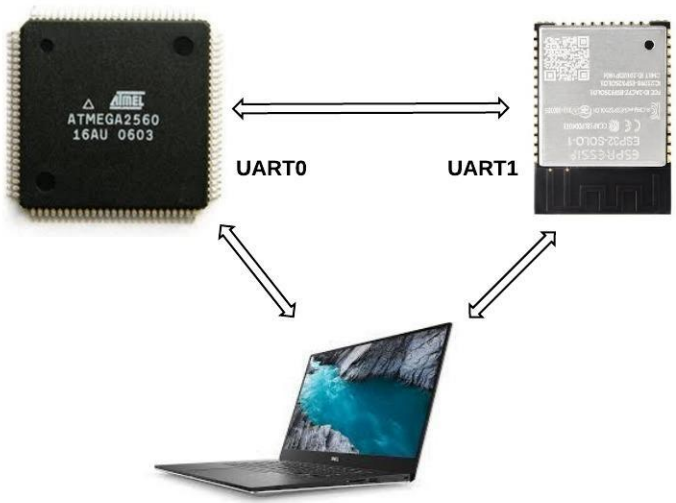
- Send the data "**Hi**" with UART configuration **9600-8N1**. Determine the number of packets transferred per second.

1. **9600** ⇒ Baud Rate
2. **8** ⇒ Number of data bits in a frame
3. **N** ⇒ No Parity bits
4. **1** ⇒ 1 Stop bit
5. "**H**" ⇒ ASCII value = **0b01001000**
6. "**i**" ⇒ ASCII value = **0b01101001**
7. **Endianness** ⇒ Little-endian, by default
8. **10 bits** per packet (1-Start, 8-Data and 1-Stop)
9. With Baud Rate = 9600 bps, **960 packets** are sent per sec



Connection Diagram

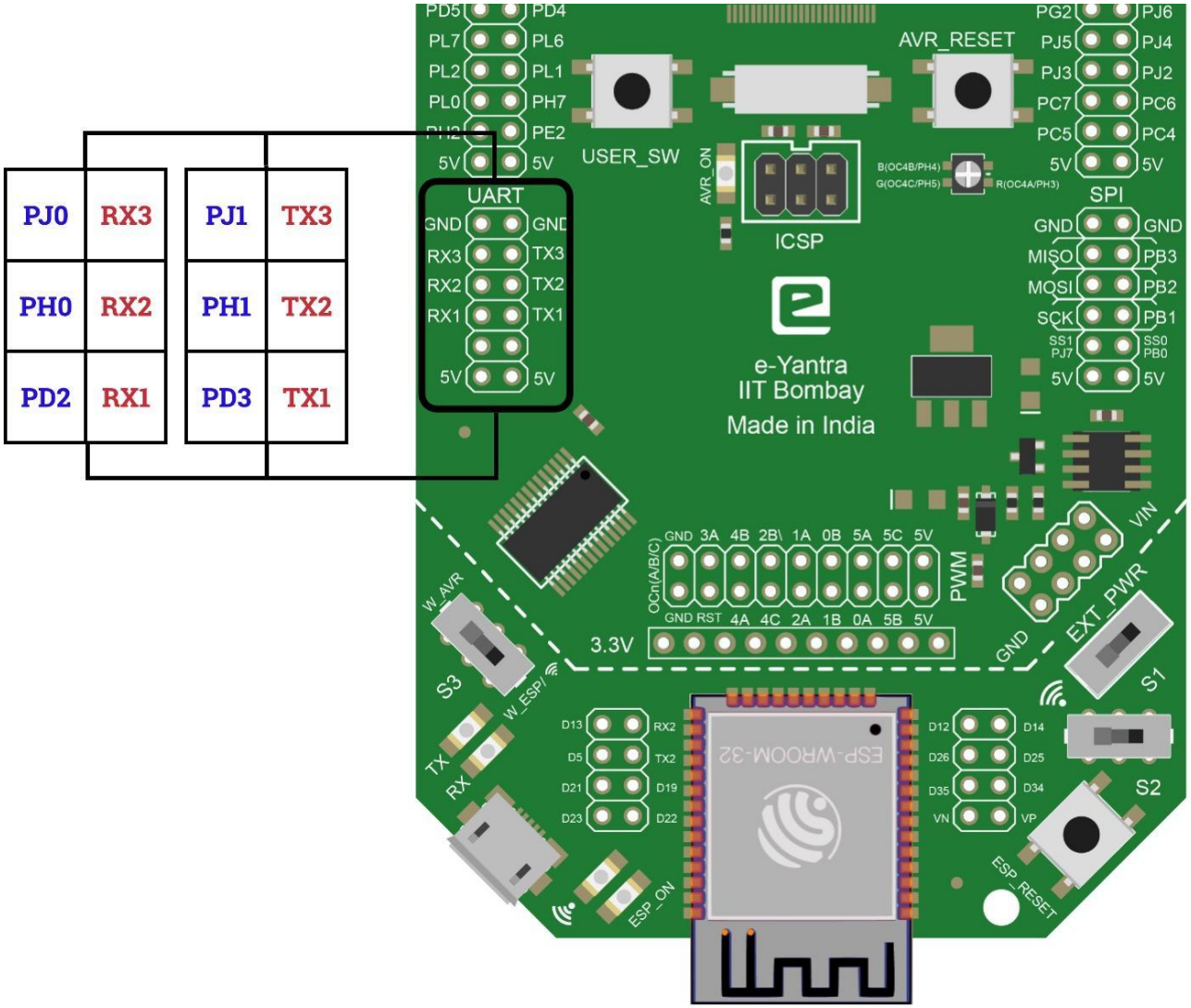
Connection Diagram



- Connection between ATmega2560 and PC
  1. TX0 → USB
  2. RX0 → USB
- Connection between ATmega2560 and ESP32
  1. ATmega2560:TX0 → ESP32:RX1
  2. ATmega2560:RX0 → ESP32:TX1

UART Header on eYFi-Mega Board

UART Header on eYFi-Mega Board



Problem Statement

Problem Statement

- Write a program to read character from UART and do the following according to the character received:

Character	Action
'F'	Decrement freq by 100 ms
'S'	Increment freq by 100 ms

Thank You!

Post your queries on: helpdesk@e-yantra.org