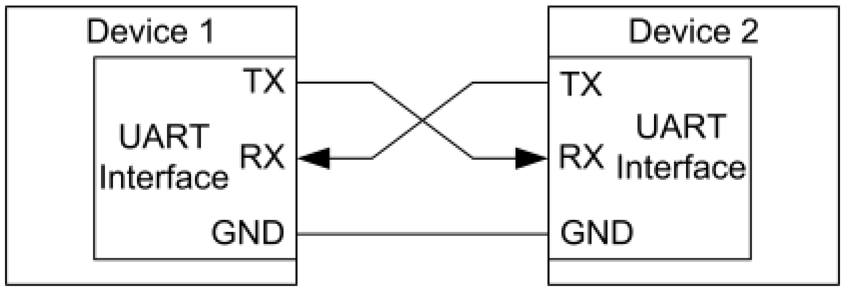
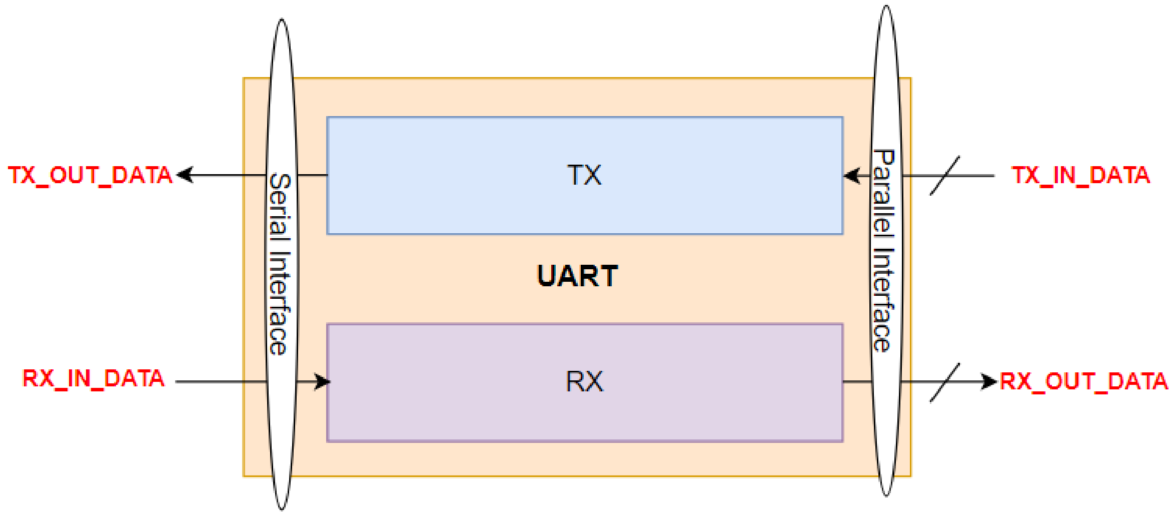
UART Receiver

Introduction: -

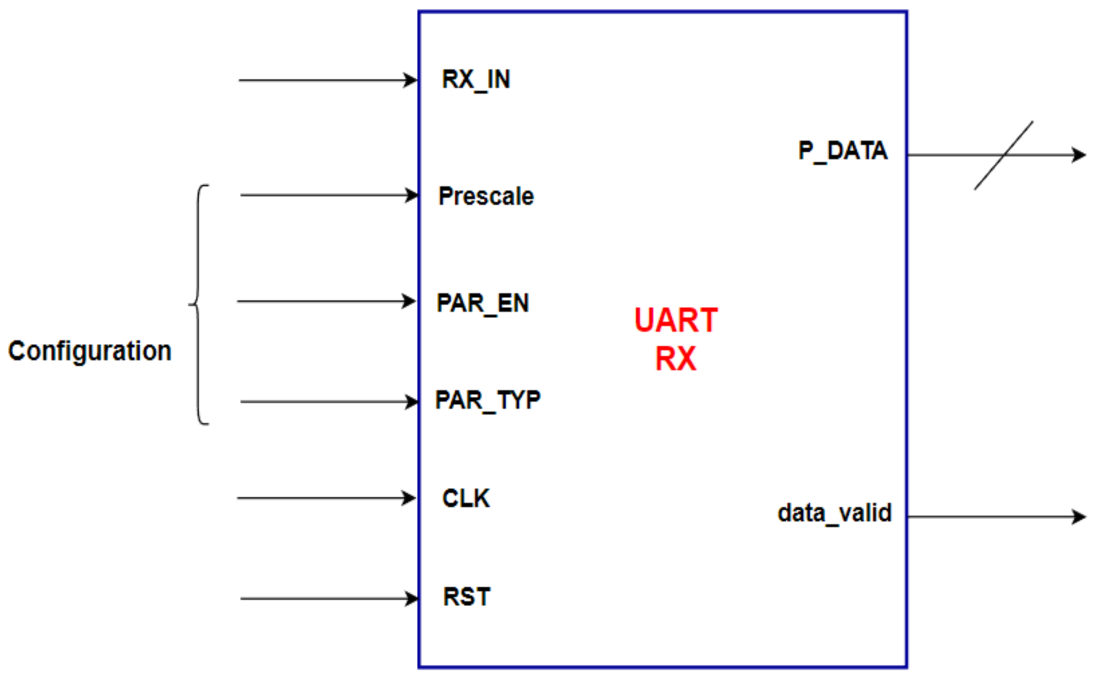
- There are many serial communication protocol as I2C, UART and SPI.
- A **U**niversal **A**synchronous **R**eceiver/**T**ransmitter (UART) is a block of circuitry responsible for implementing serial communication**.**
- UART is Full Duplex protocol (data transmission in both directions simultaneously)



- **Transmitting UART** converts parallel data from the master device (eg. CPU) into serial form and transmit in serial to receiving UART.
- **Receiving UART** will then convert the serial data back into parallel data for the receiving device.



Block Interface: -



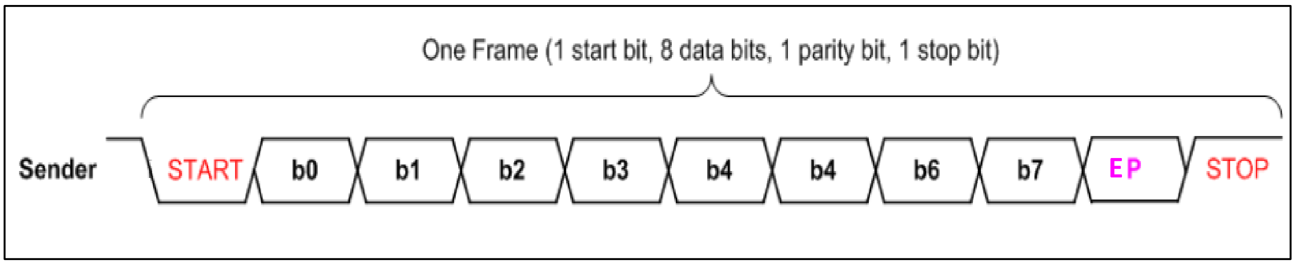
Port	Width	Description
CLK	1	UART RX Clock Signal
RST	1	Synchronized reset signal
PAR_TYP	1	Parity Type
PAR_EN	1	Parity_Enable
Prescale	6	Oversampling Prescale
RX_IN	1	Serial Data IN
P_DATA	8	Frame Data Byte
Data_valid	1	Data Byte Valid signal

Specifications: -

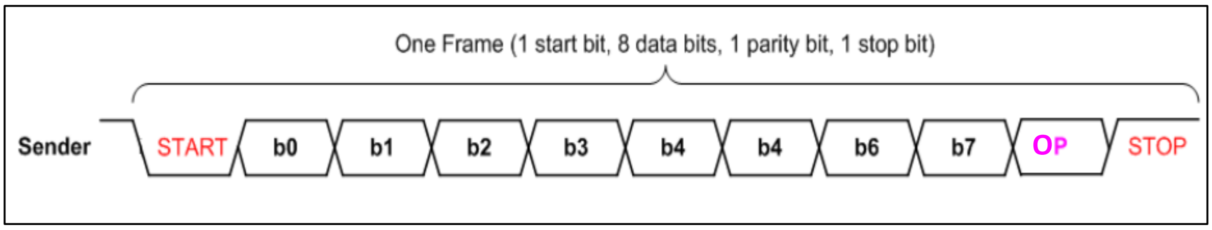
- UART RX receive a UART frame on **RX_IN**.
- UART_RX support oversampling by 8, 16, 32
- **RX_IN** is high in the **IDLE** case (No transmission).
- **PAR_ERR** signal is **high** when the calculated parity bit not equal the received frame parity bit as this mean that the frame is corrupted.
- **STP_ERR** signal is **high** when the received stop bit not equal 1 as this mean that the frame is corrupted.
- DATA is extracted from the received frame and then sent through **P_DATA** bus associated with **DATA_VLD** signal only after checking that the frame is received correctly and not corrupted. (PAR_ERR = 0 && STP_ERR = 0).
- UART_RX can accept consequent frames without any gap.
- Registers are cleared using asynchronous active low reset
- **PAR_EN** (Configuration)
  0: To disable frame parity bit
  1: To enable frame parity bit
- **PAR_TYP** (Configuration)
  0: Even parity bit
  1: Odd parity bit

All Expected Received Frames: -

1. **Data Frame (in case of Parity is enabled & Parity Type is even)**
   – One start bit (1'b0)
   – Data (LSB first or MSB, 8 bits)
   – Even Parity bit
   – One stop bit

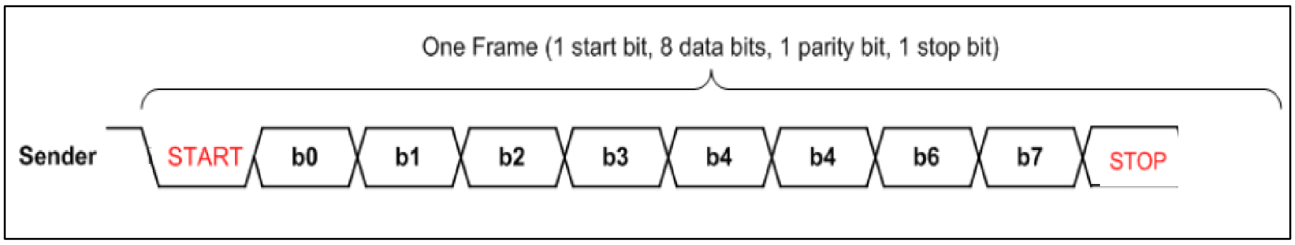

One Frame (1 start bit, 8 data bits, 1 parity bit, 1 stop bit)
Sender
START
b0
b1
b2
b3
b4
b4
b6
b7
EP
STOP

2. **Data Frame (in case of Parity is enabled & Parity Type is odd)**
   – One start bit (1'b0)
   – Data (LSB first or MSB, 8 bits)
   – Odd Parity bit
   – One stop bit



One Frame (1 start bit, 8 data bits, 1 parity bit, 1 stop bit)
Sender
START
b0
b1
b2
b3
b4
b4
b6
b7
OP
STOP

3. **Data Frame (in case of Parity is not Enabled)**
   – One start bit (1'b0)
   – Data (LSB first or MSB, 8 bits)
   – One stop bit

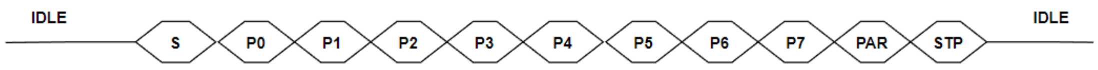


One Frame (1 start bit, 8 data bits, 1 parity bit, 1 stop bit)
Sender
START
b0
b1
b2
b3
b4
b4
b6
b7
STOP

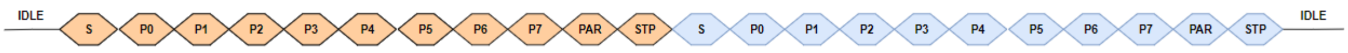
Waveforms: -

Expected Input (RX_IN): -

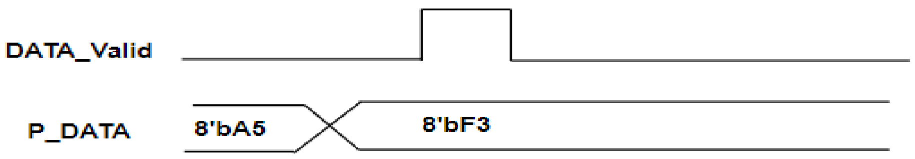
1. In case of one frame: -



2. In case of consequent frames: -

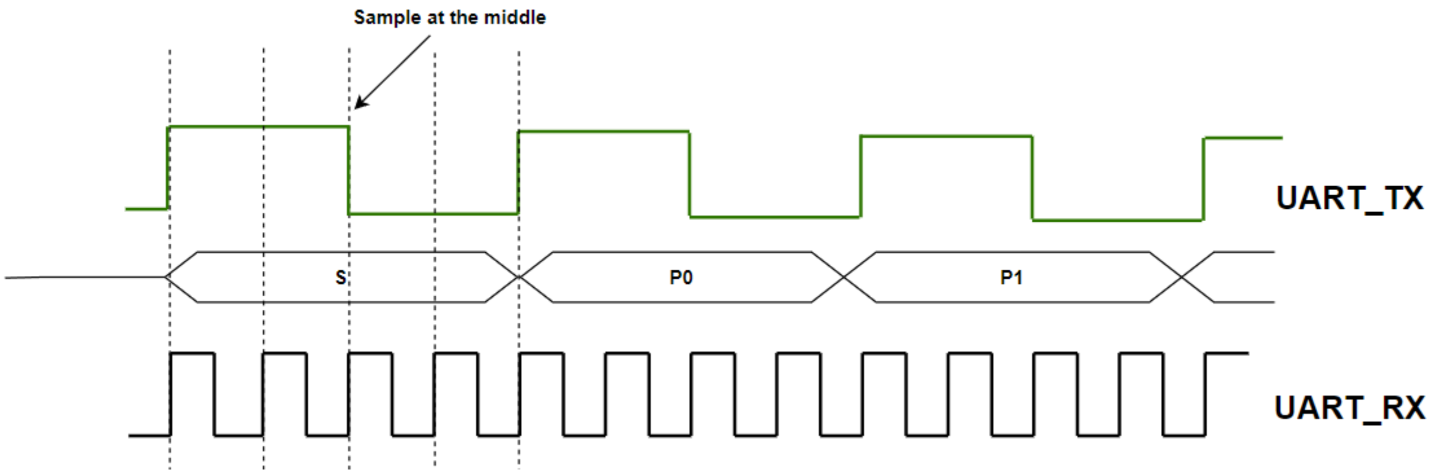


Expected Output: -

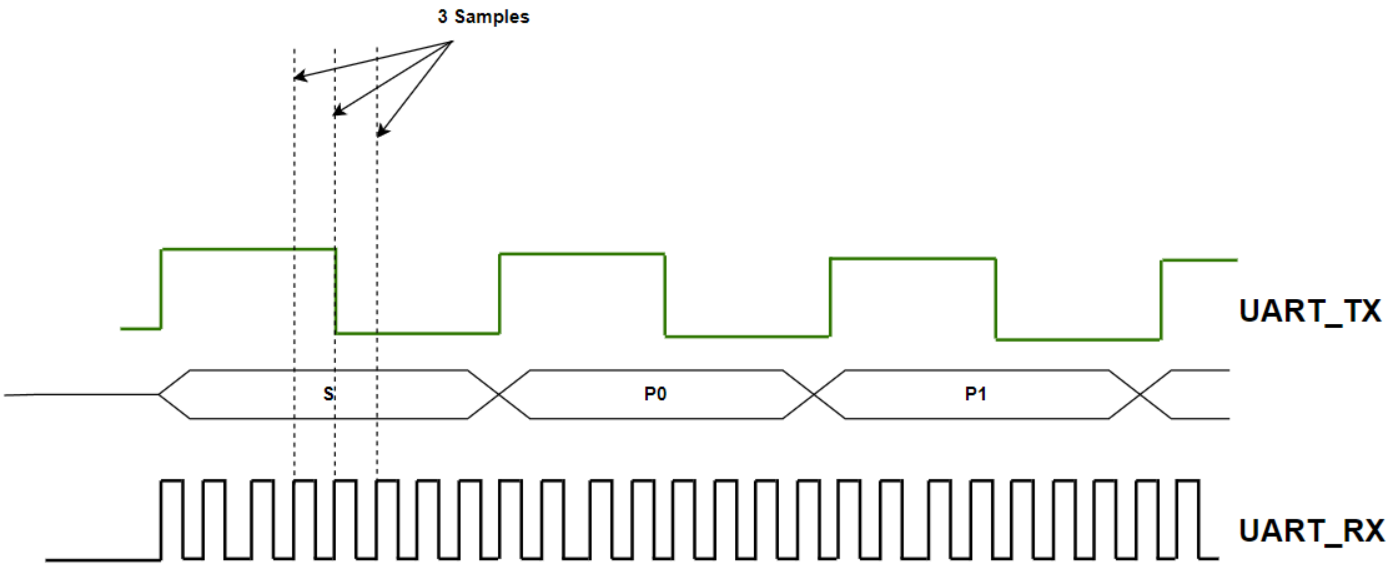


<u>Oversampling: -</u>

1. **Oversampling by 4: This means that the clock speed of UART_RX is 4 times the speed of UART_TX.**

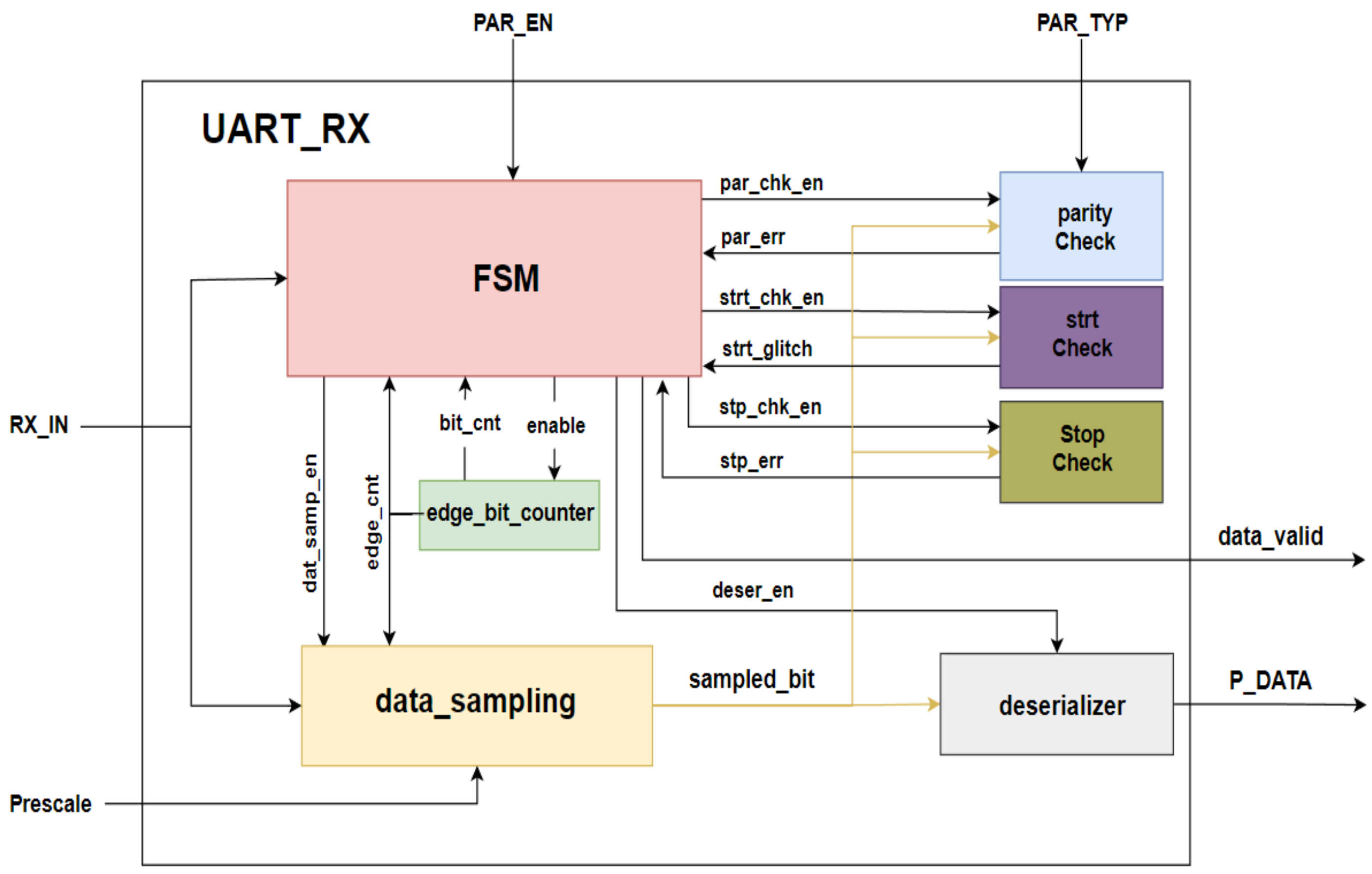


2. **Oversampling by 8: This means that the clock speed of UART_RX is 8 times the speed of UART_TX.**



we will mainly use prescaler 16

Recommended Block Diagram: -



Requirements: -

1- Implement the above Specifications for UART RX using Verilog language.

2- Write a testbench to validate your design using 200 MHz clock frequency.