EE342 - Digital System Design

Laboratory Experiment - 4
2-Line Serial Transmitter/Receiver - Part 1

Preliminary Work

1. Serial Transmitter

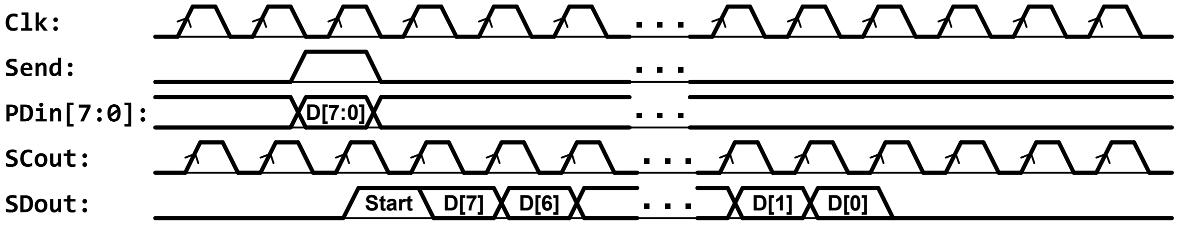
Write the Verilog module for a 2-line serial data transmitter that has the following inputs and outputs:

`Clk:` 20 MHz clock input. Other inputs, **`Send`** and **`PDin[7:0]`**, are valid at the rising clock edge.

`Send:` One-clock-cycle input pulse that indicates **`PDin[7:0]`** are valid and triggers serial transmission.

`PDin[7:0]:` 8-bit parallel data input.
`SCout:` 20 MHz clock output, **`SDout`** is valid at the rising clock edge.
`SDout:` Serial data output. **`PDin[7]`** is the first bit sent out.

Serial data output is 0 when it is idle. Every data packet begins with a start bit (the first bit where **`SDout`** is 1 after the idle period). 8 data bits follow the start bit as shown in the following timing diagram.

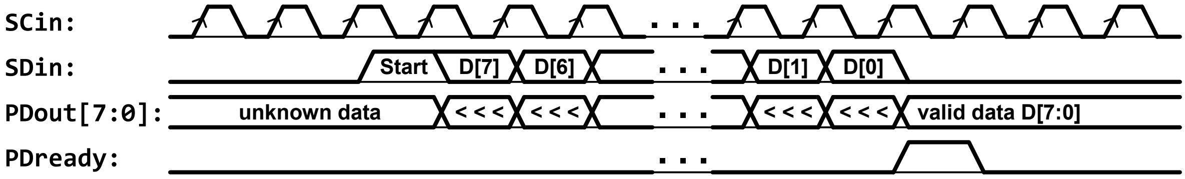


2. Serial Receiver

Write the Verilog module for a 2-line serial data receiver that has the following inputs and outputs.

`SCin:` Clock input, **`SDin`** is valid at the rising clock edge.
`SDin:` Serial data input, set to 0 when it is idle.
`PDout[7:0]:` 8-bit parallel data output, **`PDout[7]`** is the first bit received.
`PDready:` One-clock-cycle output pulse generated after the last serial data bit is received.

The receiver module should be able to receive the data sent by the transmitter described above. The timing diagram given below shows the input and output signals during data reception.



Procedure

1. Create a project directory and set up a new project in Quartus II using the **File->New Project Wizard...** menu item following the instructions given in EE342_Lab_QuartusIntro.pdf.

2. Write a top-level Verilog module that instantiates both of the transmitter and receiver modules you wrote in the preliminary work. Internally connect **`SoClk`** to **`SiClk`** and **`SDout`** to **`SDin`**. Connect **`SoClk`** and **`SDout`** to top-level module outputs to monitor serial lines easily in simulation.

3. Save the top-level module source code. Add the source code of the instantiated modules into project file list. Save your project.

4. Compile and debug the project.

5. Simulate your code to see the serial data and clock output waveforms. Check if the parallel data sent by the transmitter is received correctly at the receiver end.