Lab 2 – Digital Systems Design

In contrast to a combinational circuit, the output of a sequential circuit can depend on the current as well as previously applied inputs. Consequently, to design sequential circuits, we need memory elements that retain the information about previous inputs. Sequential systems can be implemented using synchronous and asynchronous circuits. However, synchronous circuits are preferred due to the ease of their implementation as meeting timing requirements become easier compared to meeting them in asynchronous circuits. In this section, we will study synchronous sequential circuit design and their verification.

Basic Memory Elements and their SystemVerilog Description

All sequential circuits require some sort of memory element to store information about previous inputs. There are two basic memory elements:

1. Latch
2. Flip-Flop

Each of these elements can store 1-bit of data. We can combine multiple flip-flops to store a word (data of width more than 1) and the resulting storage element is called a register.

In this session, we will learn about the behavior of these basic memory elements and their SV descriptions.

Basic Building Blocks in Synchronous Circuits and their SystemVerilog Description

In this section of the training, we will learn different building blocks for synchronous circuits such as:

1. Counters
2. Shift Registers
3. Register Files

alongwith their SV description.

Basic Testbench Design for Synchronous Circuits

The testbench design for a synchronous circuit is different from a combinational circuit as all the outputs are changed wrt the synchronizing clock's edge. As a result, the test signals that we need to apply to synchronous circuits should also be synchronized to the clock. In this section, we will learn how to design stimulus for synchronous circuits.Design and SV Description of Finite-State Machines

We start from the design of synchronous circuits using finite-state machine (FSM) and will discuss its two types: Mealy and Moore. We will further discuss how the two state machines differ in their timing and finally learn their SV description.

Datapaths and their Controllers

Finite-state machines alone are not enough to design high-complexity sequential circuit design. As a result, we move to another design paradigm known as datapath-plus-controller paradigm in

which we will break down our circuit into two parts: Datapath and a controller. We will learn how to methodologically develop the two parts of the design and verify it using SV.

Design Examples

A universal asynchronous receiver-transmitter (UART) is a device that transmits data from a transmitter to a receiver serially one bit at a time. The clock is not transmitted and therefore the receiver is asynchronous to the transmitter clock. The standard offers multiple data widths, configurable data speeds also known as baud rate and incorporation of error detection using parity bits. However, in this design example we will be transmitting a byte data at 9600 bps baud rate without any parity.

An example of the UART frame is given below:

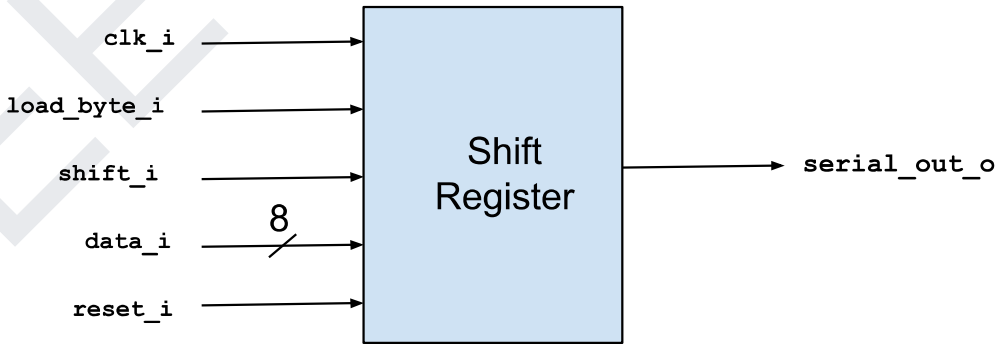
UART Data Packet [10:0]									
0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Start Bit	Data [7:0]								Stop Bit

Table 2.1. UART data packet.

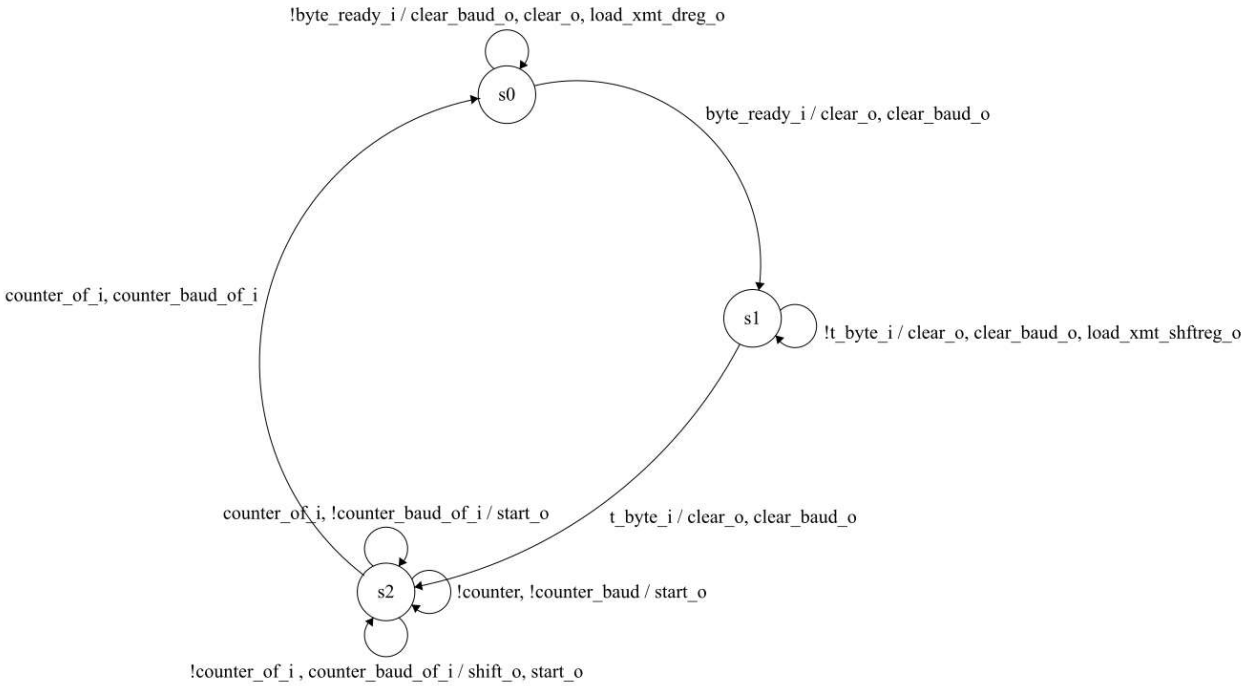
The UART that is going to transmit data receives the data from a data bus. UART gets the parallel data from the data bus, it adds a start bit, a parity bit (optional), and a stop bit, creating the data packet. In this session, we are going to implement the UART module without parity bit. The data will be 8 bits. The UART module will transfer the data at a fixed rate which is called the ***baud rate***. Baud rate is the number of bits transferred per second. The baud rate is set at a fixed value, before the data transfer can be initiated between the sender and the receiver.

Tasks

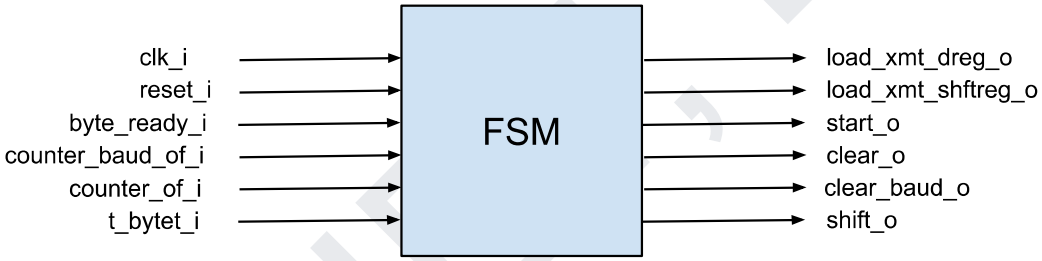
1. A 8-bit parallel-input serial-output shift register takes in 8-bit parallel values as input and takes out one bit at a time starting from the LSB. A block diagram with input/output specifications is given below:



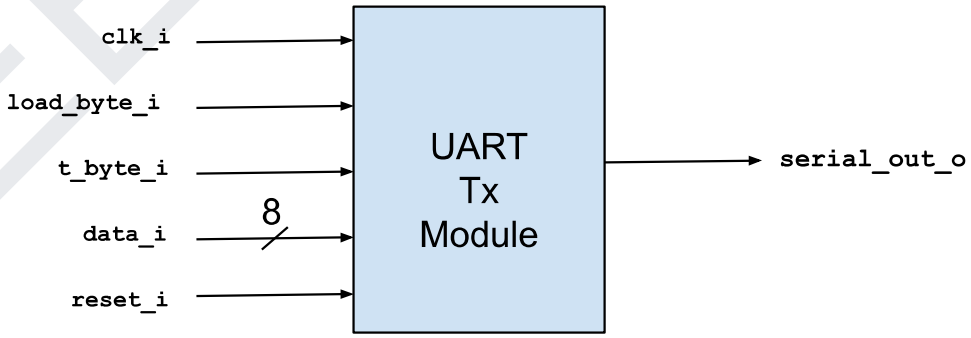
   The input **`data_i`** is loaded into the shift register and when **`shift_i`** is asserted, one bit comes out of the **`serial_out_o`** pin with LSB first.
2. Write SystemVerilog description of a 8-bit parallel-input serial-output shift register.
3. Verify the 8-bit shift register by applying random inputs and observing the output.
4. Write a SystemVerilog description of the following state-transition graph (STG).



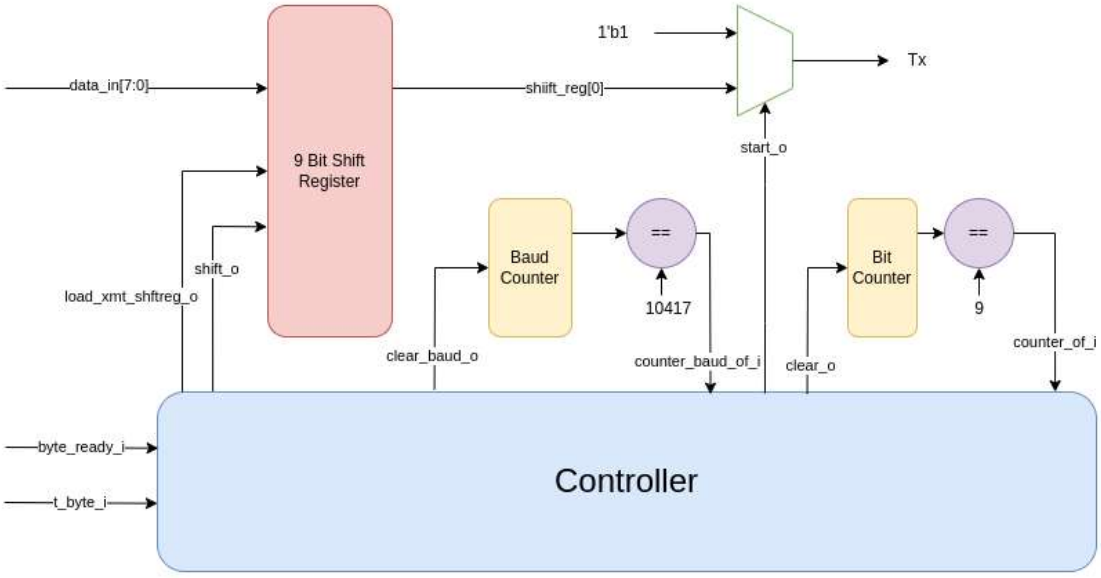
The FSM has the following block diagram.



5. Design and implement a UART module using datapath + controller paradigm. The user will set the 8 bit data using switches. After that a push button will be used for the load operation. After loading the data another push button will be used to transmit the data. Data will be transferred at the baud rate of 9600. The output of the UART module will be a single data transfer (Tx pin). The UART module should be tested on an FPGA using the UART to USB converter and the data can be observed on the computer terminal using putty



Design the UART Tx module using the FSM created in the above task as its controller and the following datapath.



References

[1] Harris, Sarah L., and David Harris. *Digital Design and Computer Architecture, RISC-V Edition*. Morgan Kaufmann, 2021.