Digital Design Verification

Lab Manual # 09 – Sequential Circuits using System Verilog (Counters / Digital clocks)

Release: 1.3

Date: 13-Aug-2024

NUST Chip Design Centre (NCDC), Islamabad, Pakistan

Revision History

Revision Number	Revision Date	Revision By	Nature of Revision	Approved By
1.0	22/01/2024	Hira Sohail, Muhammad Bilal	Complete manual	Deputy PD
1.1	28/02/2024	Hira Sohail, Muhammad Bilal	Revision in manual	Deputy PD
1.2	15/05/2024	Hira Sohail, Muhammad Bilal	Revision in manual	Deputy PD
1.3	13/08/2024	Saad Khan	Revision in manual	-

Contents

Objective

The objective of this lab is to:

- Understand and practice basic concepts of System Verilog for sequential circuits.
- Implementing shift register, sequence counter and digital clock used in digital circuits.
- Practice writing good RTL using SV with comprehensive testbench.

Tools

- Xilinx Vivado
- Cadence Xcelium

Introduction

Shift Register

- The sequential circuit that loads the data present on its inputs and then moves or “shifts” it to its output once every clock cycle, hence the name Shift Register.
- Shift Registers are used for data storage or for the movement of data and are therefore commonly used inside calculators or computers to store data or to convert the data from either a serial to parallel or parallel to serial format.
- A shift register basically consists of several single bit registers, one for each data bit, either a logic “0” or a “1”, connected together in a serial type daisy-chain arrangement so that the output from one D-FF becomes the input of the next D-FF and so on.

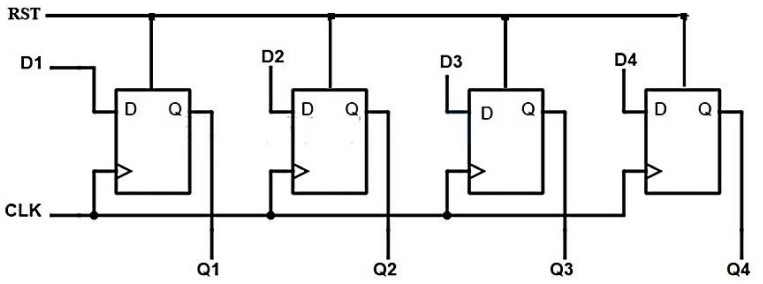


Fig 1: Block diagram of a 4-bit shift register

Design Methodology

The clock, load and shift of each flip flop will be connected to a 4-bit bus.

sh = shift

ld = load

When sh = ld = 0 no state change occurs (the register is in a hold state.) when sh = 0 ld = 1 the register loads the inputs with data. When sh = 1, ld = X the register right shifts. The next state action only occurs at rising edges of the clock. The same logic applies to left shift.

Timing Diagram

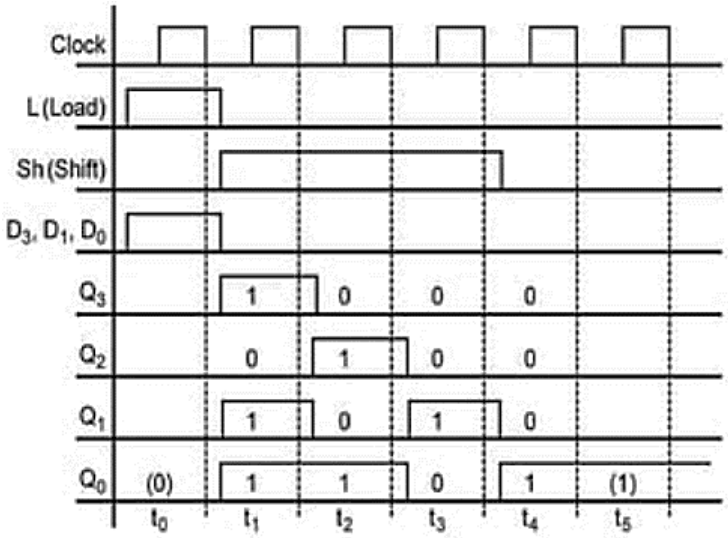


Fig 3: Timing diagram of 4-bit right shift register operating on falling edge of clock

From the diagram, when load = shift = 0, the outputs hold. When shift = 0 and load = 1, the outputs get loaded with the inputs, (Q3 = D3, Q2 = D2, Q1 = D1, Q0 = D0). Finally, when shift = 1 and load = X you can see that right shift occurs, (Q3 gets the serial input, Q2 gets the current value of Q1, Q1 gets the value of Q2 and Q0 gets the value of Q1)

LAB TASK 1: Serial inn, Serial out Shift Register

Design an 8 bit shift Register. With a 2 bit control signal(CTRL), it should do the following.

1. Ctrl == 2'b00: shift right – default state
2. Ctrl == 2'b01: shift left
3. Ctrl == 2'b10: rotate right
4. Ctrl == 2'b11: rotate left.

Timing diagram and truth table hold marks in your report.

Universal Shift Register

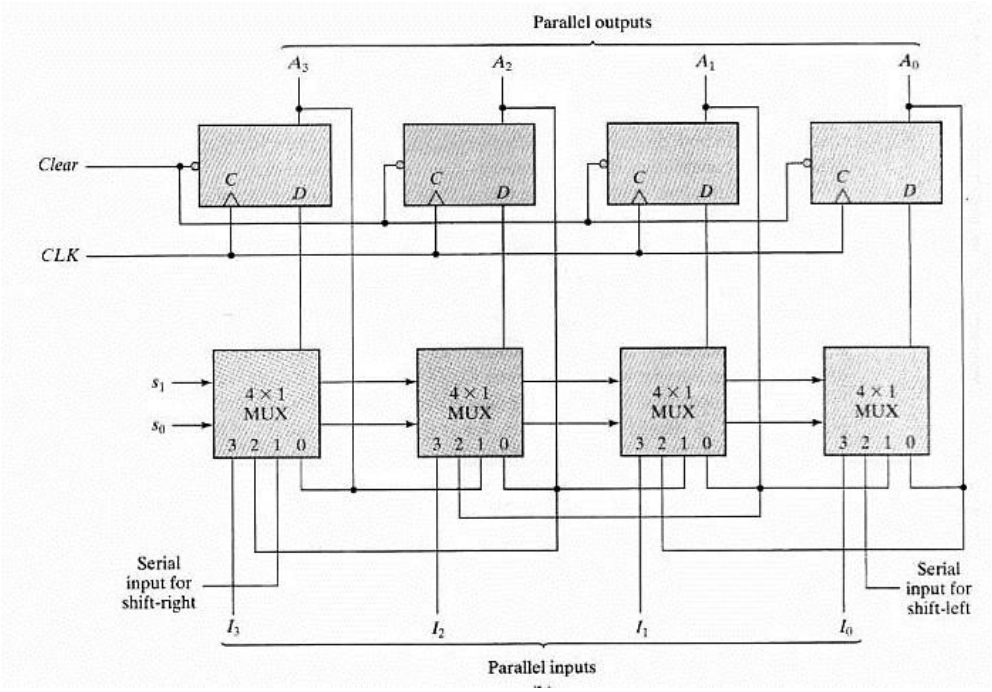
A universal shift register is a combination of sequential and combination logic that can store data within, and on every clock pulse it transfers data to the output port.

The universal shift register can be used as

1. Parallel In Parallel Out shift register
2. Parallel In Serial Out shift register.
3. Serial In Parallel Out shift register
4. Serial In Serial Out shift register.

Mode Control (Select)		Operation
0	0	No change
0	1	Shift right
1	0	Shift left
1	1	Parallel load

Block Diagram



LAB TASK 2: Universal Shift Register

Design a 4-bit universal shift register using system Verilog. Write a test bench to test functionality by passing different test cases. The report should include a clear screenshot of the waveform and code files.

Counter

Counter is basically used to count the number of clock pulses applied to a flip-flop. It can also be used for Frequency divider, time measurement, frequency measurement, distance measurement, and for generating square waveforms. There are many types of counters depending on the application and requirement. Ripple counter, decade counter, up counter, down counter, ring counter, sequence counter etc.

Asynchronous Counter

In asynchronous counter we don't use universal clock, only first flip flop is driven by main clock and the clock input of rest of the following flip flop is driven by output of previous flip flops. We can understand it by following diagram-

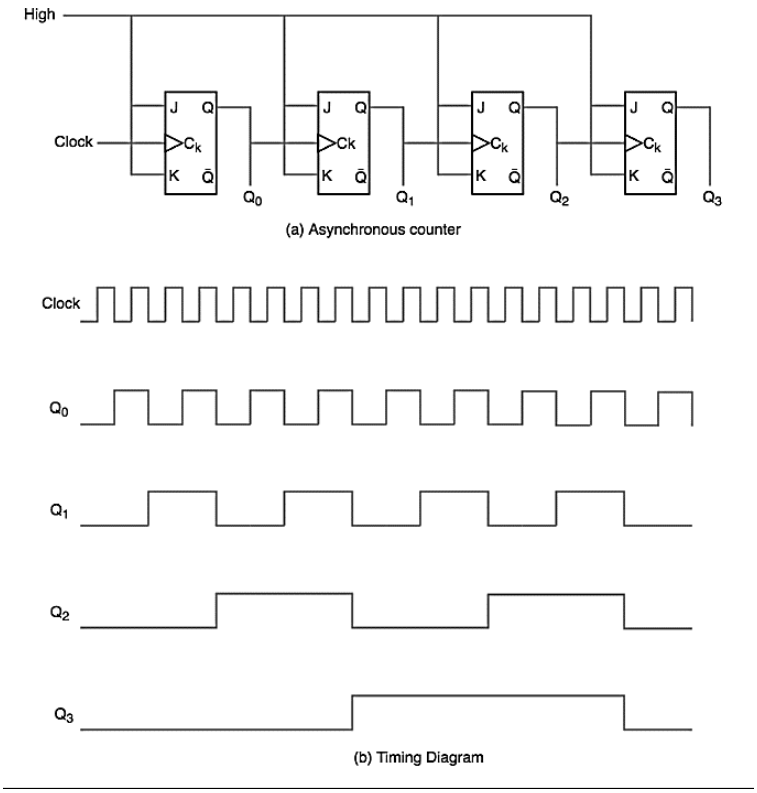


Fig 4: Asynchronous counter

It is evident from the timing diagram that Q0 is changing as soon as the rising edge of clock pulse is encountered, Q1 is changing when rising edge of Q0 is encountered (because Q0 is like clock pulse for second flip flop) and so on. In this way ripples are generated through Q0, Q1, Q2, Q3 hence it is also called **RIPPLE counter and serial counter.** A ripple counter is a cascaded arrangement of flip flops where the output of one flip flop drives the clock input of the following flip flop.

Synchronous Counter

Unlike the asynchronous counter, synchronous counter has one global clock which drives each flip flop so output changes in parallel. The one advantage of synchronous counter over asynchronous counter is, it can operate on higher frequency than asynchronous counter as it does not have cumulative delay because of same clock is given to each flip flop. It is also called as parallel counter.

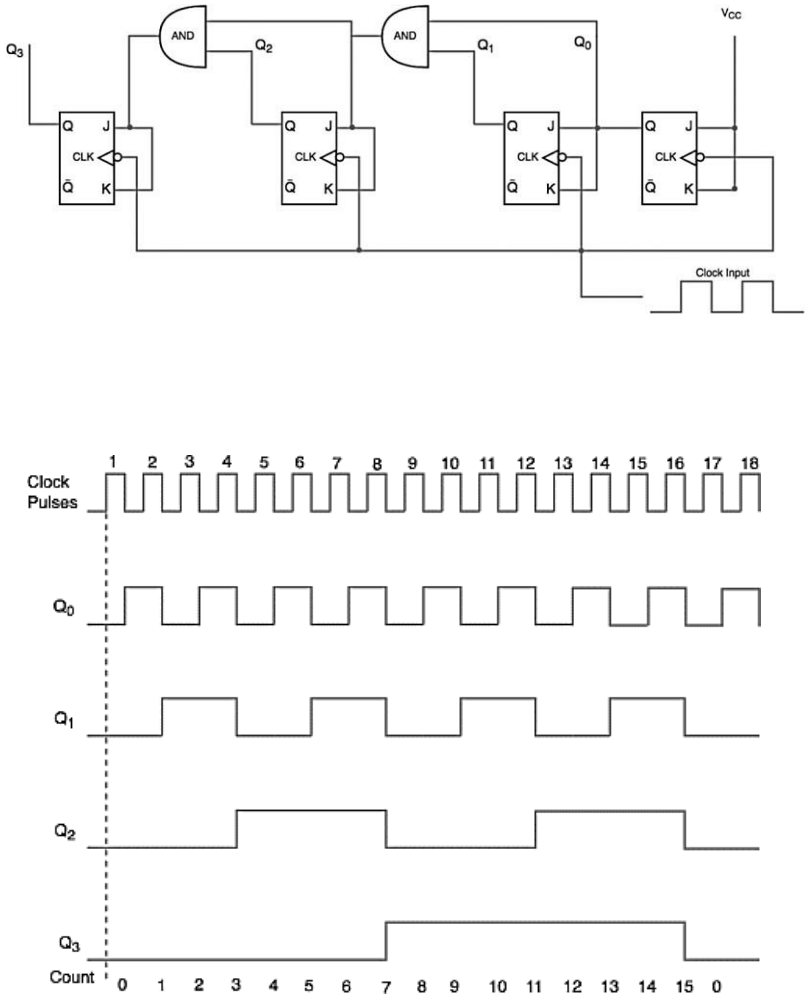
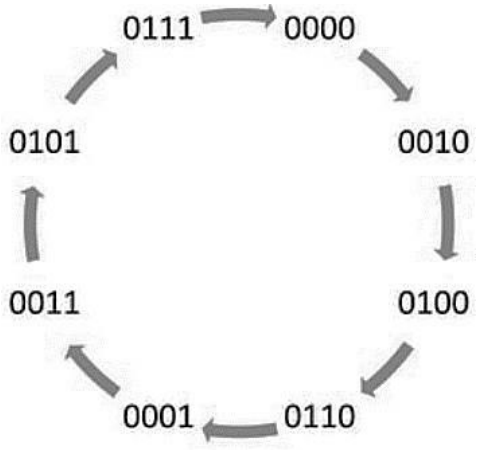


Fig 5: Synchronous counter

From circuit diagram we see that Q0 bit gives response to each falling edge of clock while Q1 is dependent on Q0, Q2 is dependent on Q1 and Q0 , Q3 is dependent on Q2,Q1 and Q0.

LAB TASK 3: Counter



Given sequence counts 0, 2, 4, 6, 1, 3, 5, 7…. Repeat

Design Methodology

Note: 'A+' donates A_{t+1} and so on

DCBA	D+C+B+A+
0000	0010
0001	0011
0010	0100
0011	0101
0100	0110
0101	0111
0110	0001
0111	0000
1000	XXXX
1001	XXXX
1010	XXXX
1011	XXXX
1100	XXXX
1101	XXXX
1110	XXXX
1111	XXXX

Design above given sequence counter using System Verilog. Write a comprehensive test bench to test its functionality. Include RTL code, its testbench, simulation waveform and brief conclusion of your design methodology in the report.32ws

LAB TASK 4: Freq Divider

Task 4a:

1. **Objective:** Design a frequency divider that reduces the input clock frequency from 100 MHz to 25 MHz

2. **Instructions:**

- Create a System Verilog module that takes a 100 MHz clock input.
- Implement the frequency divider using registers or flip-flops to divide the clock frequency by 4.
- The output should be a 25 MHz clock signal.
- Simulate your design to verify its functionality.

Task 4b:

3. **Objective:** Design a standalone clock divider that generates a 75 kHz clock signal from a 100 MHz input clock.

4. **Instructions:**

- Create a separate System Verilog module that takes a 100 MHz clock input.
- Implement the clock divider using a counter to reduce the frequency to 75 kHz.
- Focus on understanding how counters work to achieve this frequency reduction.
- Simulate the module to ensure it correctly outputs the 75 kHz clock.

Digital Clock

LAB TASK 5: Digital Clock

Design a digital clock using system Verilog. You will use a 50 MHz clock signal as the input and derive the necessary timing for seconds, minutes, and hours.

- Generate a 1 Hz signal from the 50 MHz input clock. This will involve creating a clock divider module that accurately produces a 1 Hz enabled signal. There should be three outputs to tell the time - seconds, minutes, and hours.
- The time units should undergo incrementation within an always block utilizing Behavioral modeling. With each clock cycle, the 'seconds' unit should undergo incrementation. Upon reaching '59', the 'minutes' unit should increment by 1. Similarly, when 'minutes' reaches '59', 'hours' should increase by 1. If 'hours' reaches '23', the digital clock should reset.
- Write a testbench to test the functionality of the clock.
- Report should include clear screenshot of the waveforms.

Note: The Clock should display the time in 24 hours format.

i.e.: HH:MM:SS