**VISVESVARAYA TECHNOLOGICAL UNIVERSITY,**

**BELAGAVI, KARNATAKA, INDIA**

**NITTE EDUCATION TRUST | NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY**

**An Autonomous Institution with A$^+$ Grade UGC by NAAC UGC, Approved by UGC, AICTE, Government of Karnataka, Yelahanka, Bengaluru-560064, Karnatka, India.**

**Internship Report On**

# "*Design and Verification of 4-bit serial adder using Verilog*"

A internship report submitted in partial fulfillment of the requirement for the award of

**BACHELOR OF ENGINEERING**

**In**

**ELECTRONICS AND COMMUNICATION ENGINEERING**

**2021-2022**

Submitted By:

**R.MEGHANA** **1NT18EC118**

Under the Guidance of:
**Dr.Sowmya Madhavan**
Associate Professor
Dept. of Electronics and Communication Engineering
Nitte Meenakshi Institute of Technology
Yelahanka, Bangalore-560064

**Name of person under whom the internship is carried out**
Mr. Damodara M S
**(Duration : May 5,2021 to june 24,2021)**

NITTE EDUCATION TRUST | NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY

**An Autonomous Institution with A+ Grade UGC by NAAC UGC, Approved by UGC, AICTE, Government of Karnataka, Yelahanka, Bengaluru-560064, Karnatka, India.**

## Department of Electronics and Communication Engineering

# Certificate

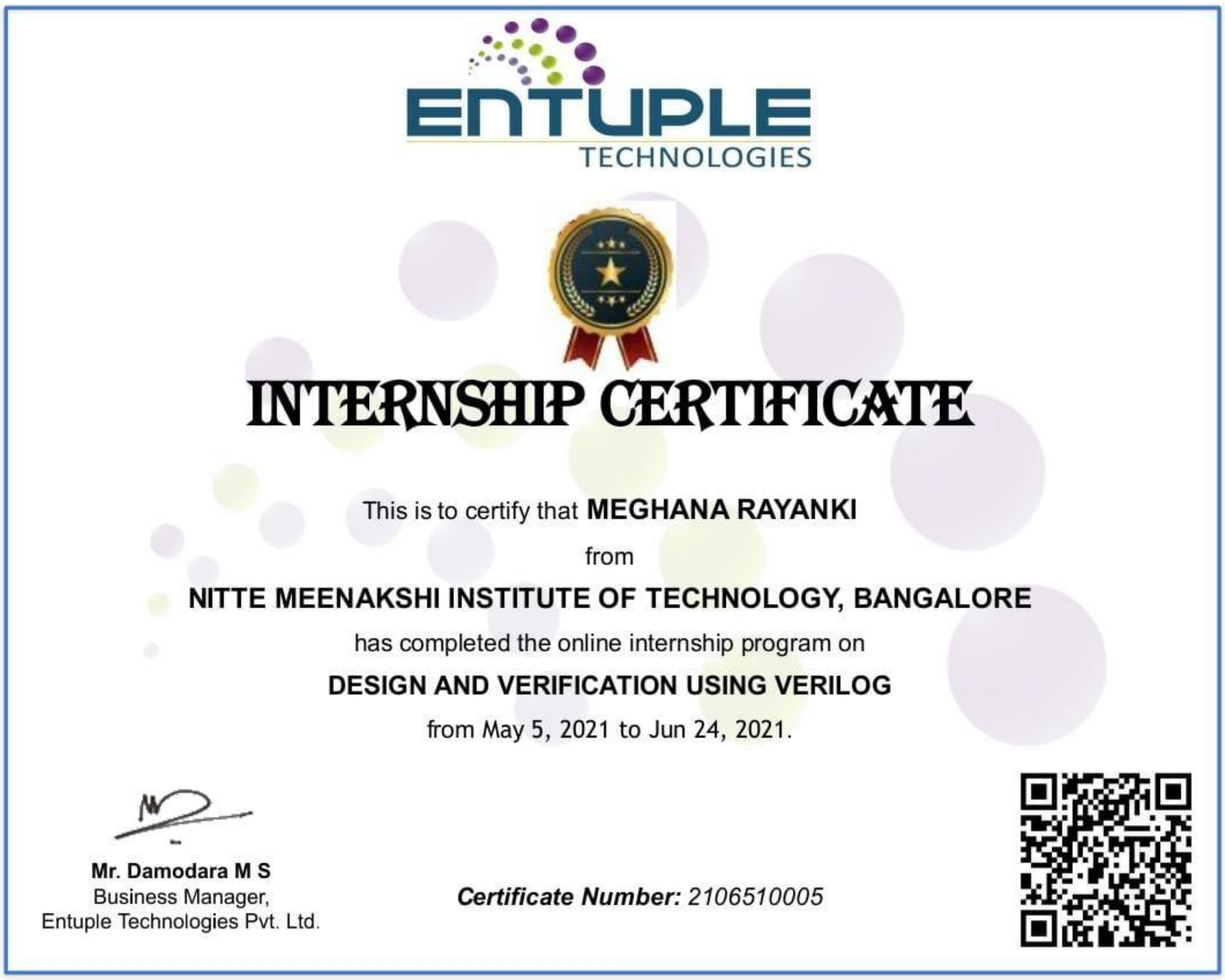
*This is to certify that **R.MEGHANA(1NT18EC118)**has submitted the internship report entitled **"Design and Verification of 4-bit serial adder using Verilog"** in fulfillment for the award of Bachelor of Engineering in Electronics and Communication Engineering from Visvesvaraya Technological University, Belagavi during the year 2021-2022. It is certified that all the corrections, suggestions indicated for internal assessment have been incorporated in the report. The internship report has been approved as it satisfies the academic requirements in respect of work prescribed for the aforesaid degree.*

| | | |
|---|---|---|
| Prof. _______________ | Dr. Ramachandra A C | Dr. H. C. Nagaraj |
| **Guide** | **HOD** | **Principal** |

## External Viva Voce

| **Name of Examiners** | **Signature with Date** |
|---|---|
| **1) ____________________** | |
| **2) ____________________** | |

**Internship certificate provided by the InternshipInstitution/Company**
(Colored Xerox Copy)



ENTUPLE TECHNOLOGIES

INTERNSHIP CERTIFICATE

This is to certify that **MEGHANA RAYANKI**

from

**NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY, BANGALORE**

has completed the online internship program on

**DESIGN AND VERIFICATION USING VERILOG**

from May 5, 2021 to Jun 24, 2021.

[illegible]

**Mr. Damodara M S**
Business Manager,
Entuple Technologies Pvt. Ltd.

***Certificate Number:*** *2106510005*

# ***ACKNOWLEDGEMENT***

It is my proud privilege and duty to acknowledge the kind help and guidance received from several people in preparation of this report. It would not have been possible to prepare this report in this form without valuable suggestions, cooperation and guidance.

First, I would like to thank **Mr. Damodara M S** for giving me the opportunity to do the internship.

I wish to record my sincere gratitude to Management, and **Dr. H. C. Nagaraj,** Principal, Nitte Meenakshi Institute of Technology, Bengalurufor the permission provided to accomplish this internship.

My sincere thanks to **Dr. Ramachandra A.C ,** Professor and Head, Department of Electronics and Communication Engineering for his valuable suggestions and guidance.

My sincere gratitude to **Dr. Rajesh N, Dr. Sunil, S. Harakannanavar** and **Dr. Thimmaraja Yadava G,** Internship Coordinators**,** Department of Electronics and Communication Engineering, for their valuable suggestions in preparing this report.

I express my sincere gratitude to our beloved guide**, Dr. Sowmya Madhavan,** Asst. Professor, Department of Electronics and Communication and Engineering for his/her support and guidance.

I am extremely great full to faculty members of Department of Electronics and Communication Engineering, and friends for their support and encouragement in successful completion of this internship.

| **Name of Student** | **USN** |
|---|---|
| R. Meghana | 1NT18EC118 |

# INDEX

## ABSTRACT

Verilog, standardized as IEEE 1364, is a hardware description language (HDL) used to model electronic systems. It is most commonly used in the design and verification of digital circuits at the register-transfer level of abstraction. It is also used in the verification of analog circuits and mixed-signal circuits, as well as in the design of genetic circuits.

Hardware description languages such as Verilog are similar to software programming languages because they include ways of describing the propagation time and signal strengths (sensitivity). There are two types of assignment operators; a blocking assignment (=), and a non-blocking (<=) assignment.

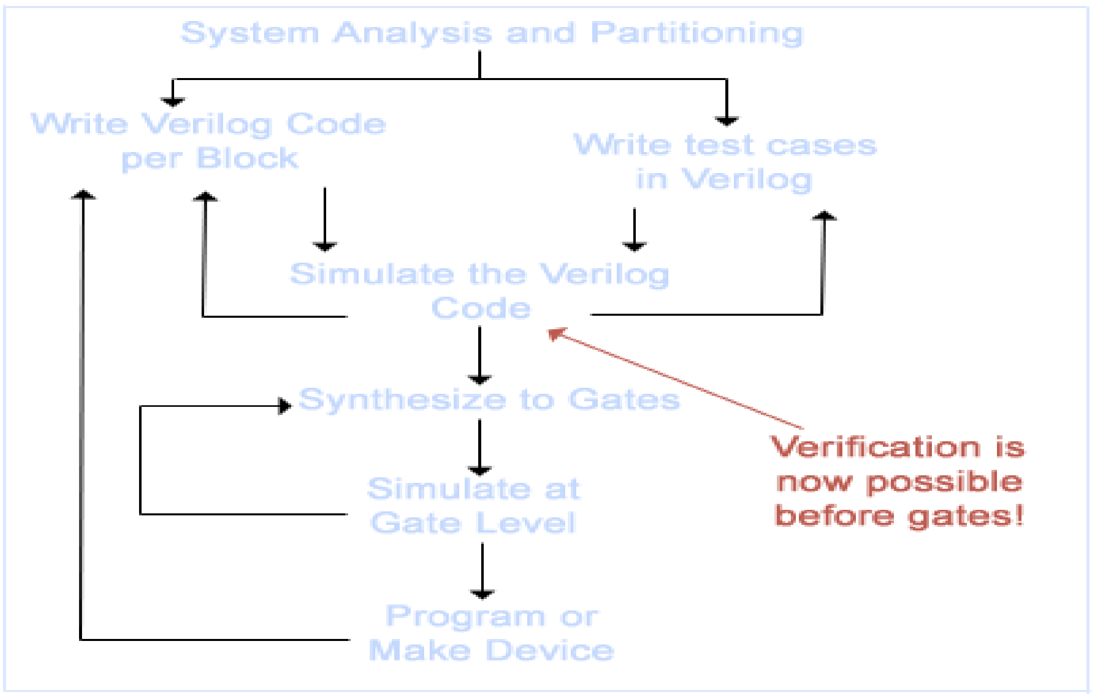
A Verilog design consists of a hierarchy of modules. Modules encapsulate *design hierarchy*, and communicate with other modules through a set of declared input, output, and bidirectional ports. Internally, a module can contain any combination of the following: net/variable declarations (wire, reg, integer, etc.), concurrent and sequential statement blocks, and instances of other modules (sub-hierarchies). Sequential statements are placed inside a begin/end block and executed in sequential order within the block. However, the blocks themselves are executed concurrently.

A subset of statements in the Verilog language are synthesizable. Verilog modules that conform to a synthesizable coding style, known as RTL (register-transfer level), can be physically realized by synthesis software. Synthesis software algorithmically transforms the Verilog source into a netlist, a logically equivalent description consisting only of elementary logic primitives that are available in a specific FPGA or VLSI technology. Further manipulations to the netlist ultimately lead to a circuit fabrication blueprint.

The serial binary adder or bit-serial adder is a digital circuit that performs binary addition bit by bit. The serial full adder has three single-bit inputs for the numbers to be added and the carry in. There are two single-bit outputs for the sum and carry out. The carry-in signal is the previously calculated carry-out signal. The addition is performed by adding each bit, lowest to highest, one per clock cycle. A shift register (serial-in parallel-out type) consists of a group of flip-flops arranged such that the output of one feeds the input of the next so that the binary numbers stored shift from one flip-flop to the next controlled by a clock pulse. This implementation is a 4-bit shift register utilising d-type flip-flops.

# INTRODUCTION

## ➢ Design flow using verilog


System Analysis and Partitioning
Write Verilog Code per Block
Write test cases in Verilog
Simulate the Verilog Code
Synthesize to Gates
Simulate at Gate Level
Program or Make Device
Verification is now possible before gates!

## Adders

An adder is a digital logic circuit in electronics that is extensively used for the addition of numbers. In many computers and other types of processors, adders are even used to calculate addresses and related activities and calculate table indices in the ALU and even utilized in other parts of the processors. Adders are basically classified into two types: Half Adder and Full Adder.

## Few applications:

- Adders are widely used in computer's ALU (Arithmetic logic unit) to compute addition as well as CPU (Central Processing unit) and GPU (Graphics Processing unit) for graphics applications to reduce the circuit complexity.
- Adder are basically used for performing arithmetical functions like addition, subtraction, multiplication and division in electronic calculators and digital instruments.

- Adders are used in digital calculators for arithmetic addition and devises that uses some kind of increment or arithmetic process
- They are also used in microcontrollers for arithmetic additions, PC (program counter) and timers.
- It is also used in networking and DSP (Digital signal processor) oriented system.

**4-Bit serial adder with parallel load**

The assigned problem statement circuit of 4-bit shift register consists of two 4-bit shift registers with parallel load, a full adder, and a D-type flip-flop for storing carry-out. Starting with LSB (least significant bit), at each cycle one bit of first number and one bit of second number are being added. So total 4 clock cycles are required to complete the addition of two 4-bit numbers. D-type flip-flop is used as a temporary storage element, which stores Cout of present clock cycle and feeds it as Cin for the next clock cycle. Two output registers sum and  Cout are 4-bit each which contains final required  sum and carryout of two 4-bit numbers which were fed as inputs.

## COMPANY PROFILE

ENTUPLE TECHNOLOGIES

### Overview of the company:

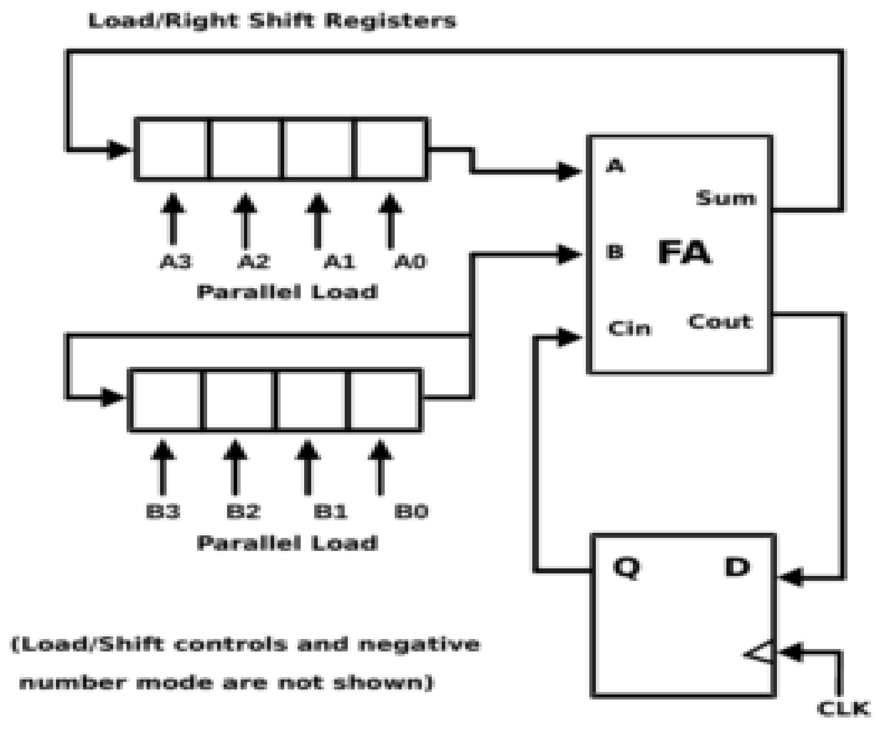
- Head Quartered at Bangalore, India, Entuple Technologies was Founded on 1st January 2010 by professionals with a combined experience of over 80 years in the Electronics Industry.
- Combined from the words "Enable" and "n-tuple", Entuple is suggestive of enabling multi-dimensional possibilities and growth for all their stakeholders.
- The objective of the company is to provide precise quality solutions to their customers and win loyalty by leveraging their technical capability. Satisfaction of the customer is valued and is of paramount importance right from the beginning of the engagement with the customer.
- The management team with its experience in different sectors such as Aerospace & Defence, Small & Medium Business, Research & Academia has joined together to build a world class team of Next Generation Solution Enablers in system design technologies. Partnering with technology leaders in such areas they also bring together a dynamic eco-system for their customers.
- In the academic sector, Entuple is committed to bridge the growing gap between curriculum and advancements in the industry by providing effective tools, technologies and enablement to the campuses.

**Few products made by the company are:**

- Automated Test Equipment (ATE) : It plays a crucial role as it enables more vigorous testing at faster rates and in a more controlled manner than was previously possible using manual procedures. ATE can involve a single measurement made continuously at very high rates or multiple measurements made by a host of different instruments.
- Wavect - Control and Measurements : Wavect real–time control prototyping system is modular, scalable and supported by a powerful model based design environment. It is ideal for leading edge research and development in power conversion systems. They provide example designs and hardware setups for applications such as Motor drives, Renewable Energy systems, Power Quality, Electric Vehicles and Microgrid etc.
- Antenna Feed : An antenna feed may be insignificant in size but plays a very important role in attaining the required performance of the intended RF beam. At Entuple, they specialise in making custom feeds for particular applications.

## WORK CARRIED OUT

**Using Verilog, Design and Verify a 4-bit serial adder as per the circuit below.**

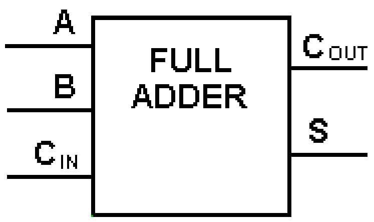


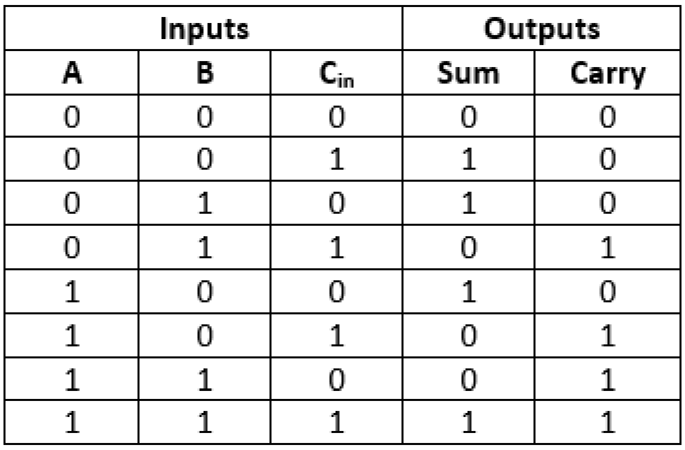
Load/Right Shift Registers
A
Sum
B
FA
Cin
Cout
A3
A2
A1
A0
Parallel Load
B3
B2
B1
B0
Parallel Load
Q
D
CLK
(Load/Shift controls and negative number mode are not shown)

### Brief explanation of the circuit components:

### Full Adder:

Full Adder is the adder which adds three inputs and produces two outputs. The first two inputs are A and B and the third input is an input carry as C-IN. The output carry is designated as C-OUT and the normal output is designated as S which is SUM.





| Inputs | | | Outputs | |
|---|---|---|---|---|
| A | B | $C_{in}$ | Sum | Carry |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

## Bit-Serial Adder:

The serial binary adder or bit-serial adder is a digital circuit that performs binary addition bit by bit. The serial full adder has three single-bit inputs for the numbers to be added and the carry in. There are two single-bit outputs for the sum and carry out. The carry-in signal is the previously calculated carry-out signal. The addition is performed by adding each bit, lowest to highest, one per clock cycle.

## Example of operation:

## Decimal:

5+9=14

## Binary:

0101+1001=1110

Addition of each step

| **Inputs** | | | **Outputs** | |
|---|---|---|---|---|
| **Cin** | **X** | **Y** | **Sum** | **Cout** |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 |

*addition starts from lowest bit

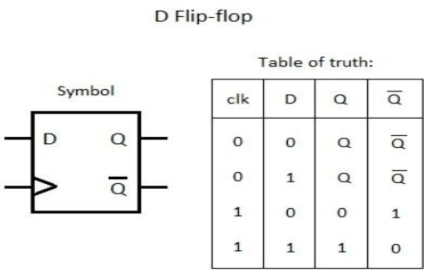
Result=1110 or 14

## Flipflop:

A flip-flop or latch is a circuit that has two stable states and can be used to store state information – a bistable multivibrator. The circuit can be made to change state by signals applied to one or more control inputs and will have one or two outputs. A flip-flop is a device which stores a single bit (binary digit) of data; one of its two states represents a "one" and the other represents a "zero". Such data storage can be used for storage of state, and such a circuit is described as sequential logic in electronics. Flip-flops and latches are fundamental building blocks of digital electronics systems used in computers, communications, and many other types of systems.

## D-Flipflop:

A D(or Delay) Flip Flop is a digital electronic circuit used to delay the change of state of its output signal until the next rising edge of a clock timing input signal occurs.



| clk | D | Q | $\overline{Q}$ |
|---|---|---|---|
| 0 | 0 | Q | $\overline{Q}$ |
| 0 | 1 | Q | $\overline{Q}$ |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |

### 4-bit shift register

- A shift register (serial-in parallel-out type) consists of a group of flip-flops arranged such that the output of one feeds the input of the next so that the binary numbers stored shift from one flip-flop to the next controlled by a clock pulse. This implementation is a 4-bit shift register utilising d-type flip-flops. In this type of circuit, the clock inputs of all the flip-flops connect to a common line, so they receive clock inputs simultaneously. With a d-type flip-flop, the value at the input D transfers to the output Q on the rising edge of every clock pulse. Since they all receive the clock pulse simultaneously, they all do this operation together on the rising edge.

- This type of shift register is also known as serial in parallel out register because we load it through its serial input; however, we read it from its parallel output. This type of register is also the basis of a serial to parallel converter and is therefore sometimes utilised in serial communication systems.
- In these types of circuits, we reset the flip-flops simultaneously and therefore their control pins connects to a common line.

Shift register truth table:

| Outputs | $Q_0$ | $Q_1$ | $Q_2$ | $Q_3$ |
|---|---|---|---|---|
| Reset | 0 | 0 | 0 | 0 |
| CK Pulse 1 | 1 | 0 | 0 | 0 |
| CK Pulse 2 | 0 | 1 | 0 | 0 |
| CK Pulse 3 | 0 | 0 | 1 | 0 |
| CK Pulse 4 | 0 | 0 | 0 | 1 |

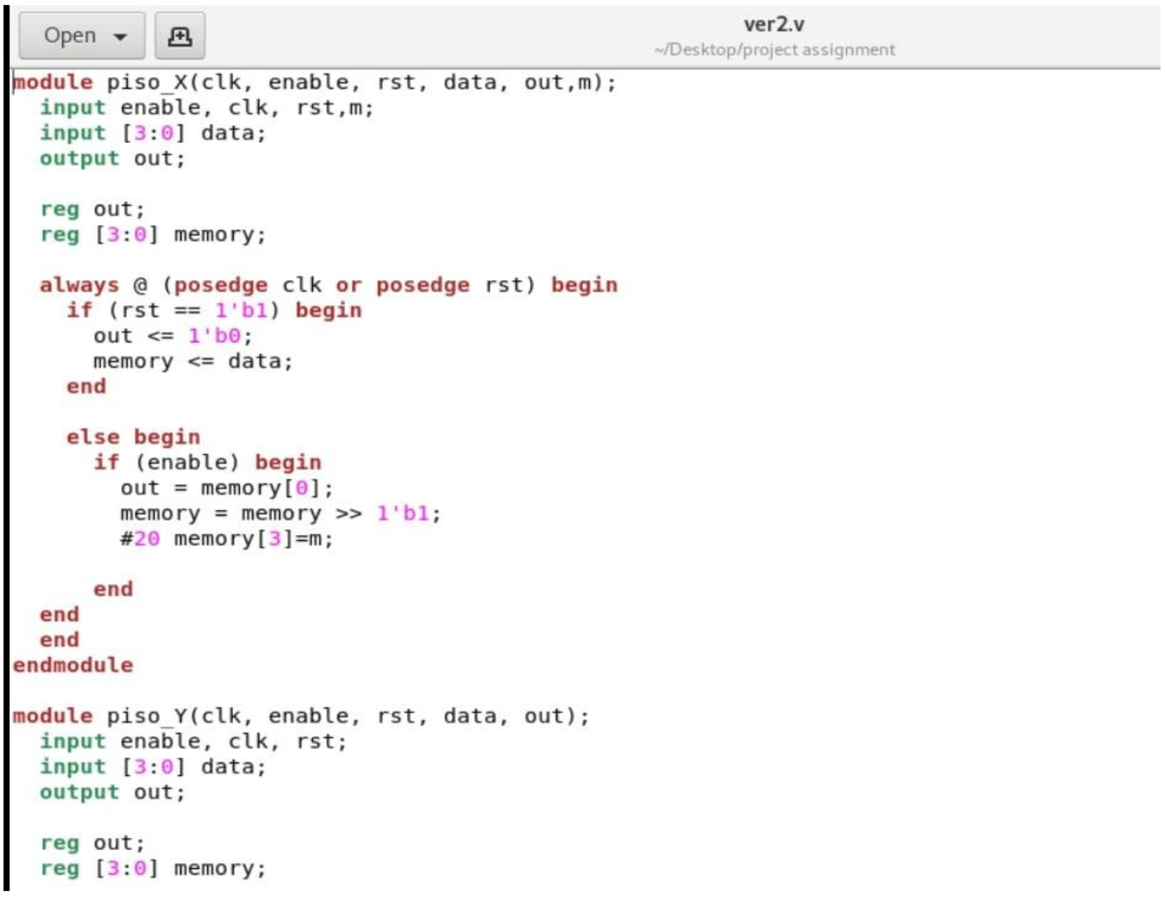
This truth table shows the outputs at successive clock pulses if we were to load the register with binary 0 0 0 1 through the serial input. If you click on the header image at the top of the page, we have an animation of this circuit that might be useful.

**Brief Explanation Of 4-Bit Serial Adder With Parallel Load:**

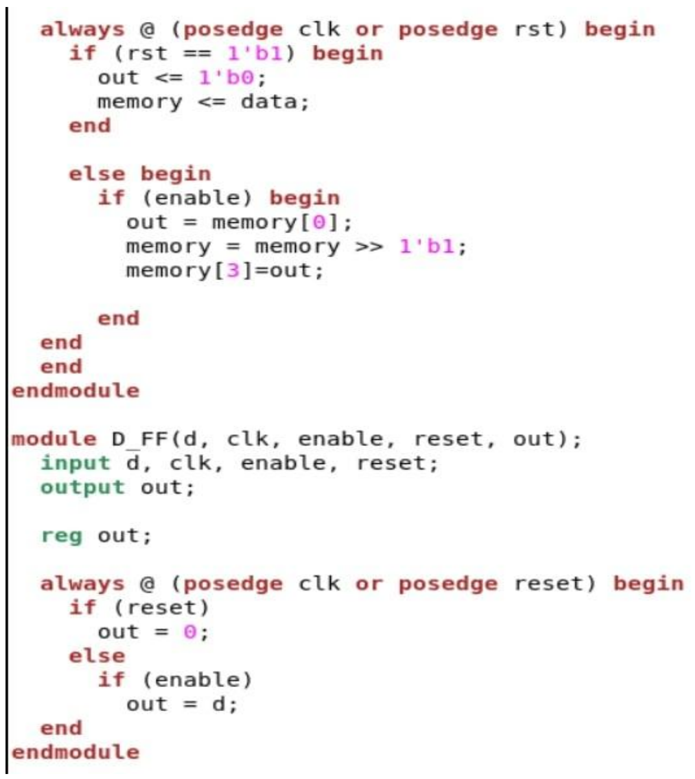
- It consists of two 4-bit shift registers with parallel load, a full adder, and a D-type flip-flop for storing carry-out. In order to load data (4-bit) to these shift registers initially, shift capability of the registers should be disabled and loading mode should be enabled. Loading of numbers to these shift registers will occur in one clock cycle. After this, shifting mode should be enabled to perform the arithmetic operation. The addition of numbers stored in these shift registers requires 4 clock cycles. Starting with LSB (least significant bit), at each cycle one bit of first number and one bit of second number are being added. The sum is stored at the MSB (most significant bit) of first shift register as in circuit above. Carry-out output produced after each cycle is fed-back to the full adder as a carry-in of the next significant bit. For this purpose one D-type flip-flop is used as a temporary storage element. The LSB of the data in second shift register is fed as the input to the MSB of the same data. Hence rotation operation of this second shift register happens.

- Here initially the two shift registers are loaded with data 1011 and 1000 respectively. Initially Cin is set as 0. When reset=1 data is fed parallely to these shift registers and output is 0. When reset=0, shift operation is performed be these registers. A variable count is declared because we just have to shift data 4 times in order to calculate sum of a 4-bit number. Therefore when count < 4, enable=1 the circuit adds the LSB bits. When count > 0 and enable=0 the same output value is retained.

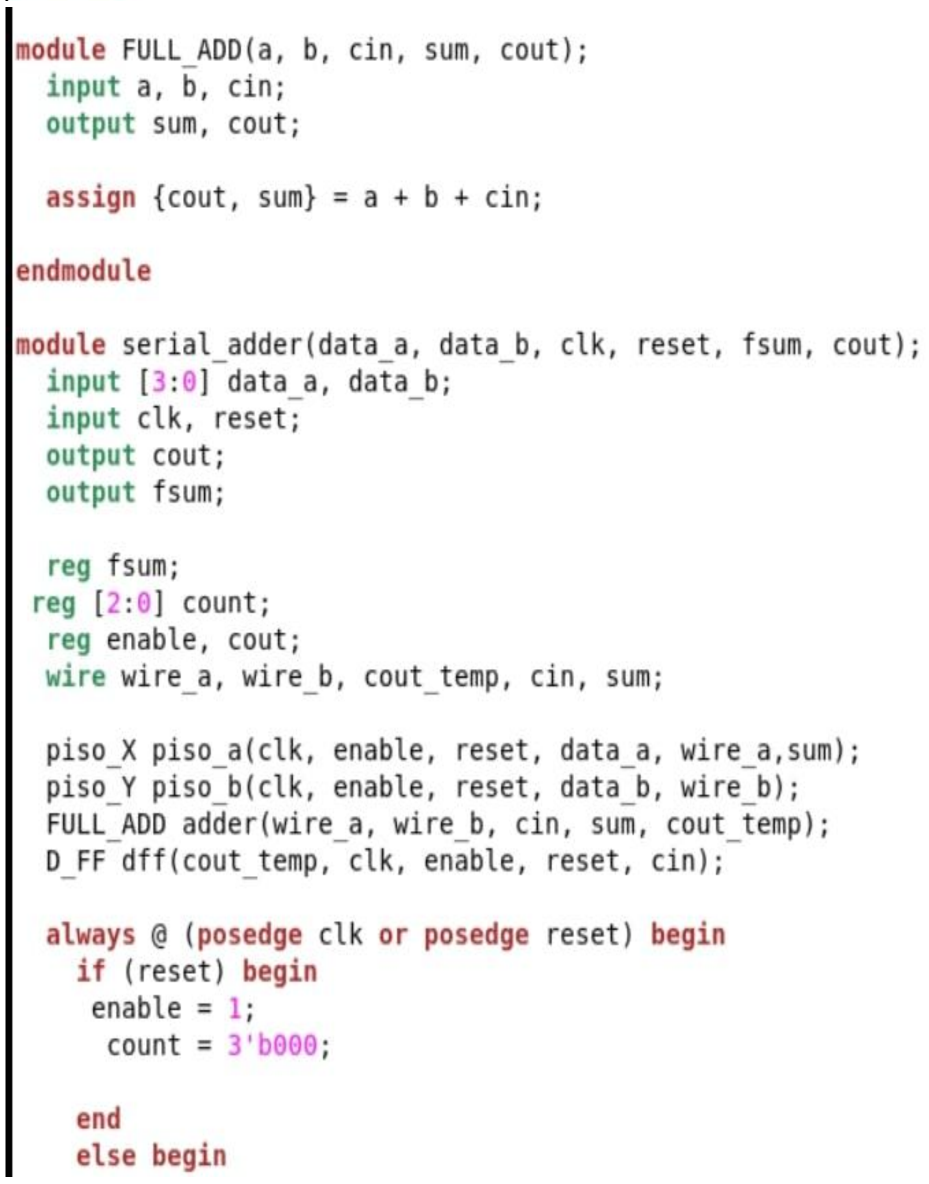
- **Code**



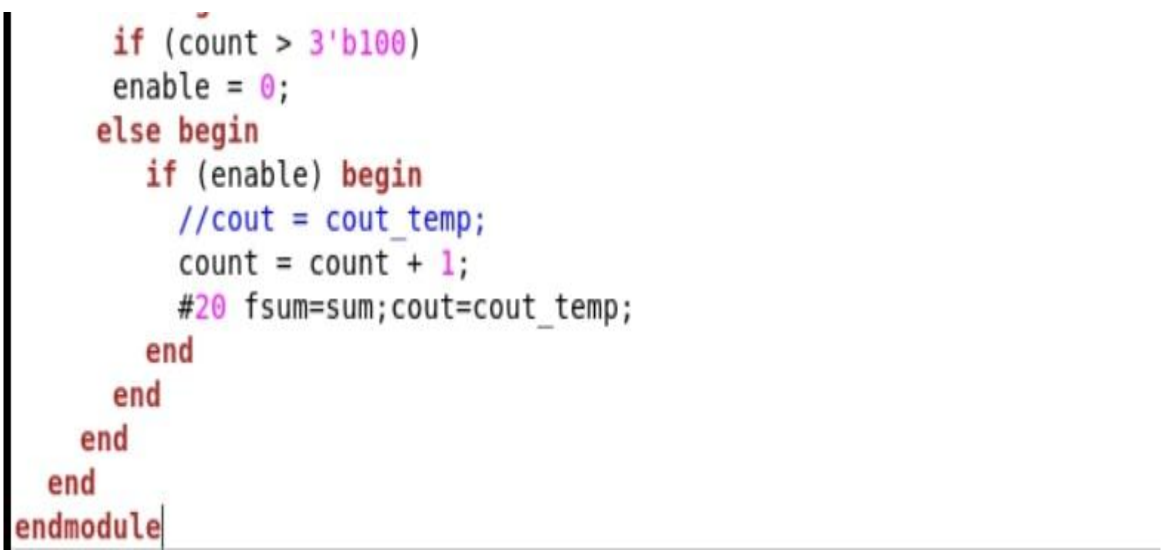
```
module piso_X(clk, enable, rst, data, out,m);
  input enable, clk, rst,m;
  input [3:0] data;
  output out;

  reg out;
  reg [3:0] memory;

  always @ (posedge clk or posedge rst) begin
    if (rst == 1'b1) begin
      out <= 1'b0;
      memory <= data;
    end

    else begin
      if (enable) begin
        out = memory[0];
        memory = memory >> 1'b1;
        #20 memory[3]=m;

      end
  end
  end
endmodule

module piso_Y(clk, enable, rst, data, out);
  input enable, clk, rst;
  input [3:0] data;
  output out;

  reg out;
  reg [3:0] memory;
```



```
  always @ (posedge clk or posedge rst) begin
    if (rst == 1'b1) begin
      out <= 1'b0;
      memory <= data;
    end

    else begin
      if (enable) begin
        out = memory[0];
        memory = memory >> 1'b1;
        memory[3]=out;

      end
  end
  end
endmodule

module D_FF(d, clk, enable, reset, out);
  input d, clk, enable, reset;
  output out;

  reg out;

  always @ (posedge clk or posedge reset) begin
    if (reset)
      out = 0;
    else
      if (enable)
        out = d;
  end
endmodule
```

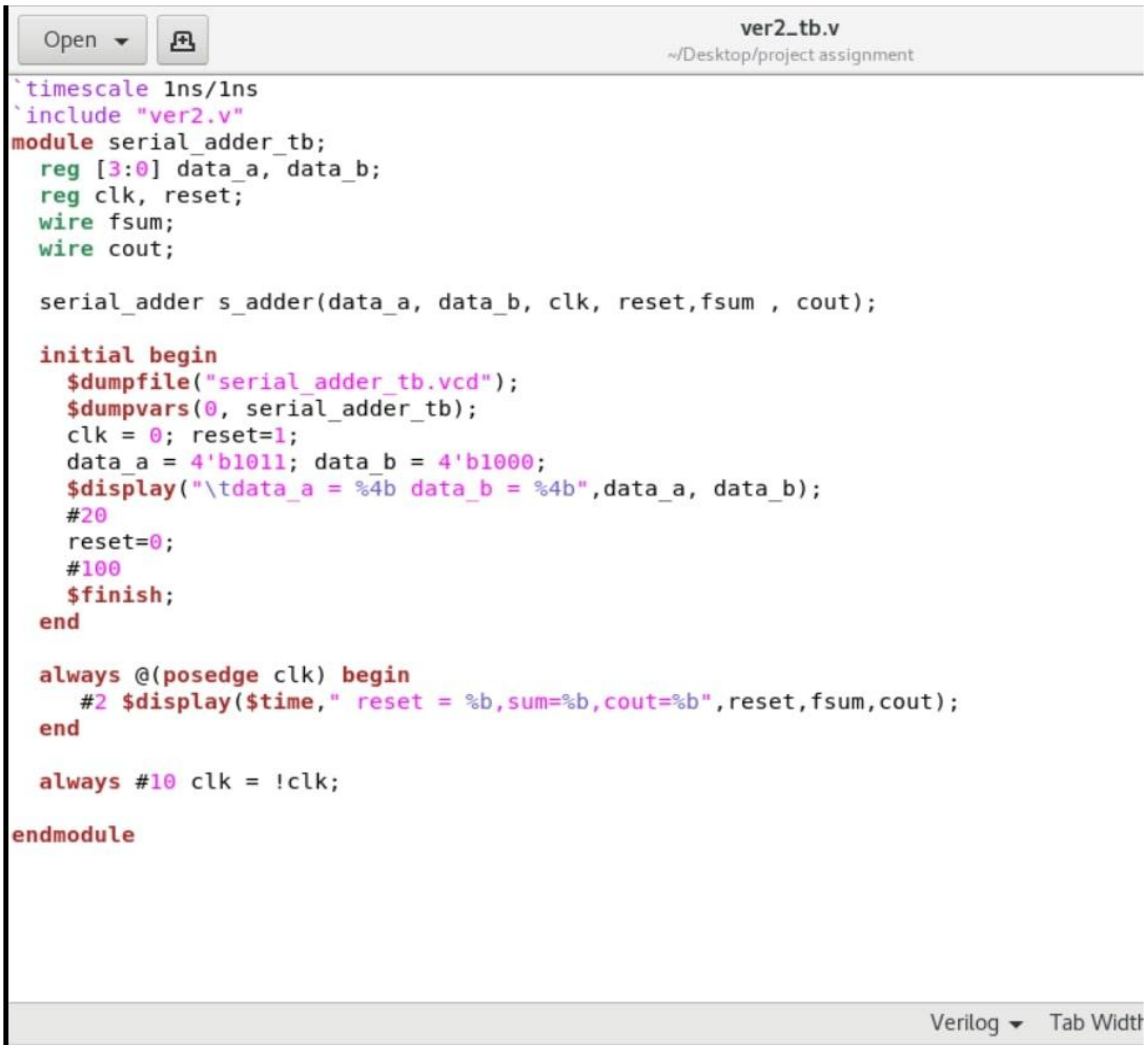


```
module FULL_ADD(a, b, cin, sum, cout);
  input a, b, cin;
  output sum, cout;

  assign {cout, sum} = a + b + cin;

endmodule

module serial_adder(data_a, data_b, clk, reset, fsum, cout);
  input [3:0] data_a, data_b;
  input clk, reset;
  output cout;
  output fsum;

  reg fsum;
 reg [2:0] count;
  reg enable, cout;
  wire wire_a, wire_b, cout_temp, cin, sum;

  piso_X piso_a(clk, enable, reset, data_a, wire_a,sum);
  piso_Y piso_b(clk, enable, reset, data_b, wire_b);
  FULL_ADD adder(wire_a, wire_b, cin, sum, cout_temp);
  D_FF dff(cout_temp, clk, enable, reset, cin);

  always @ (posedge clk or posedge reset) begin
    if (reset) begin
     enable = 1;
      count = 3'b000;

    end
    else begin
```



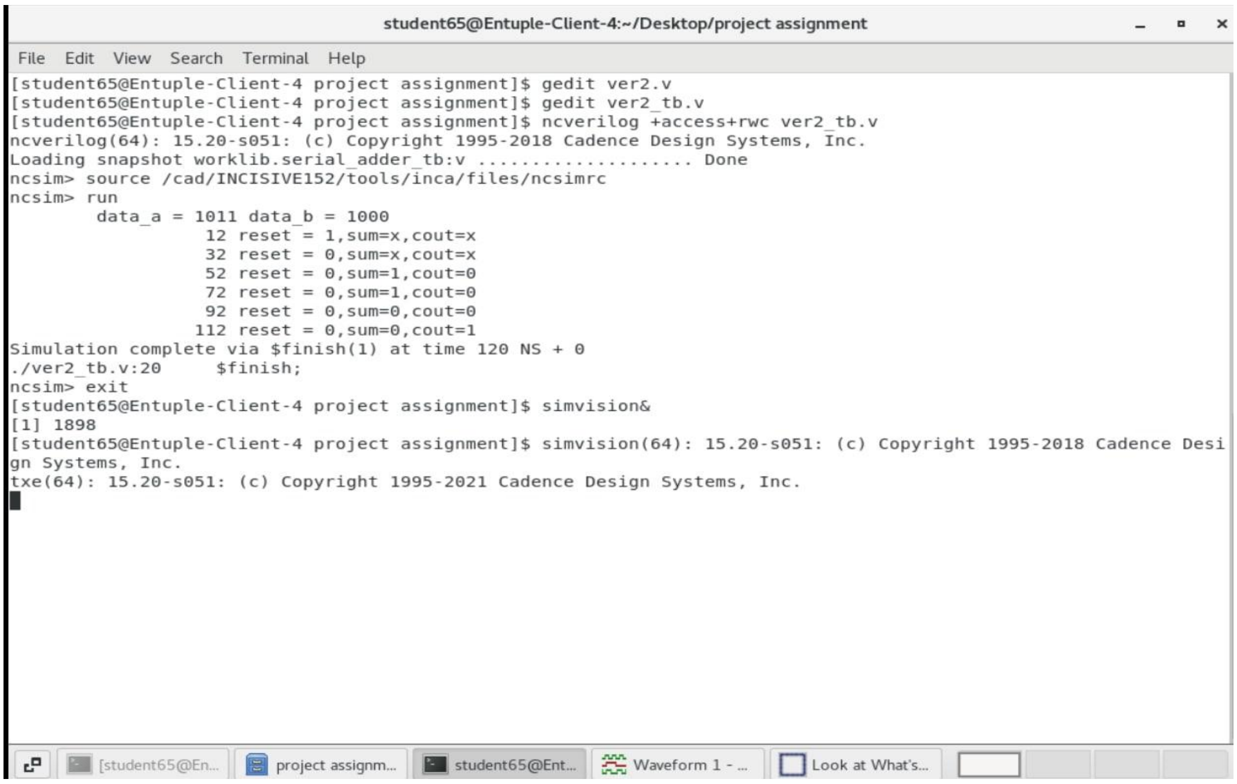
```
      if (count > 3'b100)
      enable = 0;
     else begin
        if (enable) begin
          //cout = cout_temp;
          count = count + 1;
          #20 fsum=sum;cout=cout_temp;
        end
      end
    end
  end
endmodule
```

➢ **Testbench**



```
`timescale 1ns/1ns
`include "ver2.v"
module serial_adder_tb;
  reg [3:0] data_a, data_b;
  reg clk, reset;
  wire fsum;
  wire cout;

  serial_adder s_adder(data_a, data_b, clk, reset,fsum , cout);

  initial begin
    $dumpfile("serial_adder_tb.vcd");
    $dumpvars(0, serial_adder_tb);
    clk = 0; reset=1;
    data_a = 4'b1011; data_b = 4'b1000;
    $display("\tdata_a = %4b data_b = %4b",data_a, data_b);
    #20
    reset=0;
    #100
    $finish;
  end

  always @(posedge clk) begin
     #2 $display($time," reset = %b,sum=%b,cout=%b",reset,fsum,cout);
  end

  always #10 clk = !clk;

endmodule
```

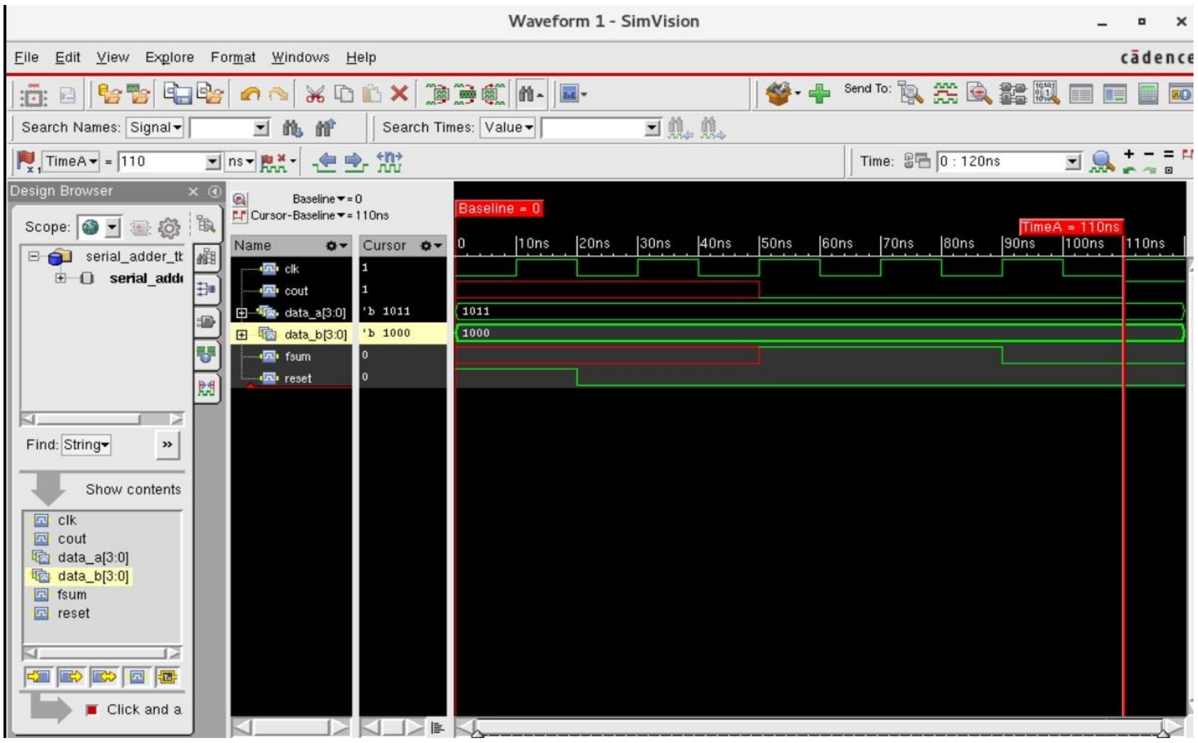
## ➢ Simulation and Implementation of the RTL code along with Testbench for the same



student65@Entuple-Client-4:~/Desktop/project assignment

File Edit View Search Terminal Help

```
[student65@Entuple-Client-4 project assignment]$ gedit ver2.v
[student65@Entuple-Client-4 project assignment]$ gedit ver2_tb.v
[student65@Entuple-Client-4 project assignment]$ ncverilog +access+rwc ver2_tb.v
ncverilog(64): 15.20-s051: (c) Copyright 1995-2018 Cadence Design Systems, Inc.
Loading snapshot worklib.serial_adder_tb:v .................... Done
ncsim> source /cad/INCISIVE152/tools/inca/files/ncsimrc
ncsim> run
        data_a = 1011 data_b = 1000
                  12 reset = 1,sum=x,cout=x
                  32 reset = 0,sum=x,cout=x
                  52 reset = 0,sum=1,cout=0
                  72 reset = 0,sum=1,cout=0
                  92 reset = 0,sum=0,cout=0
                 112 reset = 0,sum=0,cout=1
Simulation complete via $finish(1) at time 120 NS + 0
./ver2_tb.v:20     $finish;
ncsim> exit
[student65@Entuple-Client-4 project assignment]$ simvision&
[1] 1898
[student65@Entuple-Client-4 project assignment]$ simvision(64): 15.20-s051: (c) Copyright 1995-2018 Cadence Desi
gn Systems, Inc.
txe(64): 15.20-s051: (c) Copyright 1995-2021 Cadence Design Systems, Inc.
```



# CONCLUSION

As seen in the above simulated waveform, there are two 4-bit shift registers namely register a and register b which are loaded with 4-bit data each i.e,1011 and 1000 respectively. So during initial clock cycle LSB bits are added considering Cin=0.  Output of this addition is assigned to sum and Cout respectively. This repeats for four clock cycles to ensure the addition of four bits. Final sum and Cout registers each of 4-bit will have 0011 and 1000 respectively. There is a input reset signal given to 4-bit shift registers. When reset=1, data is fed parallely to these shift registers and output is 0. When reset=0, shift operation is performed by these registers. we have to shift data 4 times in order to calculate sum of a 4-bit number. So, a variable count is declared for this. When count < 4, enable=1 the circuit adds the LSB bits. When count > 0 and enable=0 the same output value is retained.

In this internship we were:

- Trained in Design and Verification using Verilog(both theory and lab session)
- Assigned project to design and verify a 4-bit serial adder as per the mentioned problem statement
- Cadence genus and EDA playground were used for simulating the developed RTL code along with the testbench for the same assigned problem statement
- Detailed report along with the simulated waveforms was prepared
- Final project results was analysed