EX.NO: 1 DESIGN AND SIMULATION OF HALF AND FULL ADDER/SUBTRACTORS

DATE: 24/01/2022

AIM:

To verify the functionality and timing of your design or portion of your design. To interpret Verilog code into circuit functionality and displays logical results of the described HDL to determine correct circuit operation and to create and verify complex functions in a relatively small amount of time.

TOOLS REQUIRED:

1. Xilinx Vivado Design Suite: WebEdition

PROCEDURE:

1. Start by clicking **Vivado Design Suite: WebEdition**

2. Go to **File → new project → Enter project name**, select the top level source as **HDL** & click **next**.

3. Enter **device properties** as

Product Category : General Purpose
Family : Kintex 7
Device : xc7k70t
Package :fbg484
Speed : -1
Top Level Source Type : HDL
Synthesis Tool : XST (VHDL/Verilog)
Simulator : ISim (VHDL/Verilog)
Preferred Language : Verilog

& **Click next**.

4. **Right click** the device name (XCS3540) in the source window

to create **new source**.

5. Select **Verilog module** and enter **file name** in the new source window & click **next**.

6. Write the **Verilog code** in the **Verilog editor window**.

7. Write the **testbench** by create **new source simulation source.**

8. Run Check syntax through **Process window→ synthesize→** double click **Behavioral Check Syntax→** and removes error if present, with proper syntax & coding.

9. Click on the symbol of FPGA device and then right click→ click on **new source**.

10. Select the desired parameters for simulating the design. In this case **combinational circuit** and **simulation time** click **finish**.

11. Assign **all input signal** (**high or low**) using just **click** on this and save file.

12. From the **source process window**. Click **Behavioral simulation**

from drop-down menu.

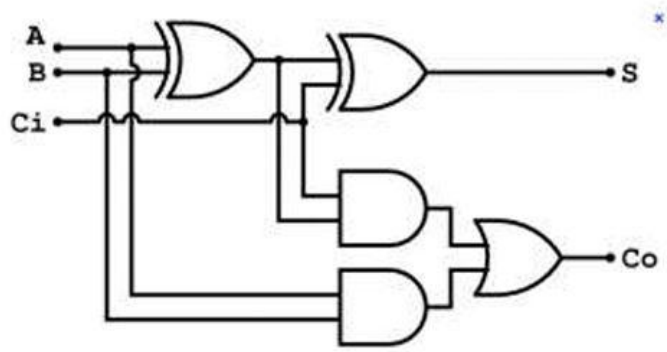
13. Double click the **Simulation Behavioral Model**.

14. Verify your design in **wave window** by seeing behavior of output signal with respect to input signal.

Full Adder:

b) FULL ADDER:

Logic Circuit



Truth Table

Input			Output	
A	B	Cin	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

Expression

$$\text{Sum} = A \oplus B \oplus Cin$$

$$Cout = AB + BCin + CinA$$

PROGRAM

Gate Level

```
//1960628 Prem Full Adder Gate Level
module fulladder(
    input a,
    input b,
    input ci,
    output s,
    output co
    );
wire w0,w1,w2;
xor xort(w0,a,b);
xor xort1(s,w0,ci);
and and1(w1,ci,w0);
and and2(w2,a,b);
or ort(co,w1,w2);

endmodule
```

Data Flow

```
//1960628 Prem Full adder Dataflow
module fa_dataflow(
    input a,
    input b,
    input ci,
    output s,
    output co
    );
assign s = a^b^ci;
assign co = (a&b) | (b&ci) | (ci&a);
endmodule
```

Behavioural

```
//1960628 Prem full adder behav
module fa_behav(
    input a,
    input b,
    input ci,
    output reg s,
    output reg co
    );
always @(a,b,ci)
begin
if (a == 1'b0 & b == 1'b0 & ci == 1'b0)
begin
s = 1'b0;
co = 1'b0;
end
if (a == 1'b0 & b == 1'b0 & ci == 1'b1)
begin
s = 1'b1;
co = 1'b0;
end
if (a == 1'b0 & b == 1'b1 & ci == 1'b0)
begin
s = 1'b1;
co = 1'b0;
end
if (a == 1'b0 & b == 1'b0 & ci == 1'b1)
begin
s = 1'b0;
co = 1'b1;
end
if (a == 1'b1 & b == 1'b0 & ci == 1'b0)
begin
```

```
s = 1'b1;
co = 1'b0;
end
if (a == 1'b1 & b == 1'b0 & ci == 1'b1)
begin
s = 1'b0;
co = 1'b1;
end
if (a == 1'b1 & b == 1'b1 & ci == 1'b0)
begin
s = 1'b0;
co = 1'b1;
end
if (a == 1'b1 & b == 1'b1 & ci == 1'b1)
begin
s = 1'b1;
co = 1'b1;
end
end
endmodule
```

TESTBENCH

```
//1960628 Prem Testbench Full adder
module tb_fa();
wire s,co;
reg a,b,ci;

fulladder I1(a,b,ci,s,co);
initial
begin
a = 1'b0;
b = 1'b0;
ci = 1'b0;
#10
a = 1'b0;
b = 1'b1;
ci = 1'b0;
#10
a = 1'b1;
b = 1'b0;
ci = 1'b0;
#10
a = 1'b1;
b = 1'b1;
ci = 1'b0;
#10
a = 1'b0;
b = 1'b0;
ci = 1'b1;
#10
a = 1'b0;
b = 1'b1;
ci = 1'b1;
#10
a = 1'b1;
b = 1'b0;
ci = 1'b1;
#10
```

```
a = 1'b1;
b = 1'b1;
ci = 1'b1;
$finish();
end
endmodule
```

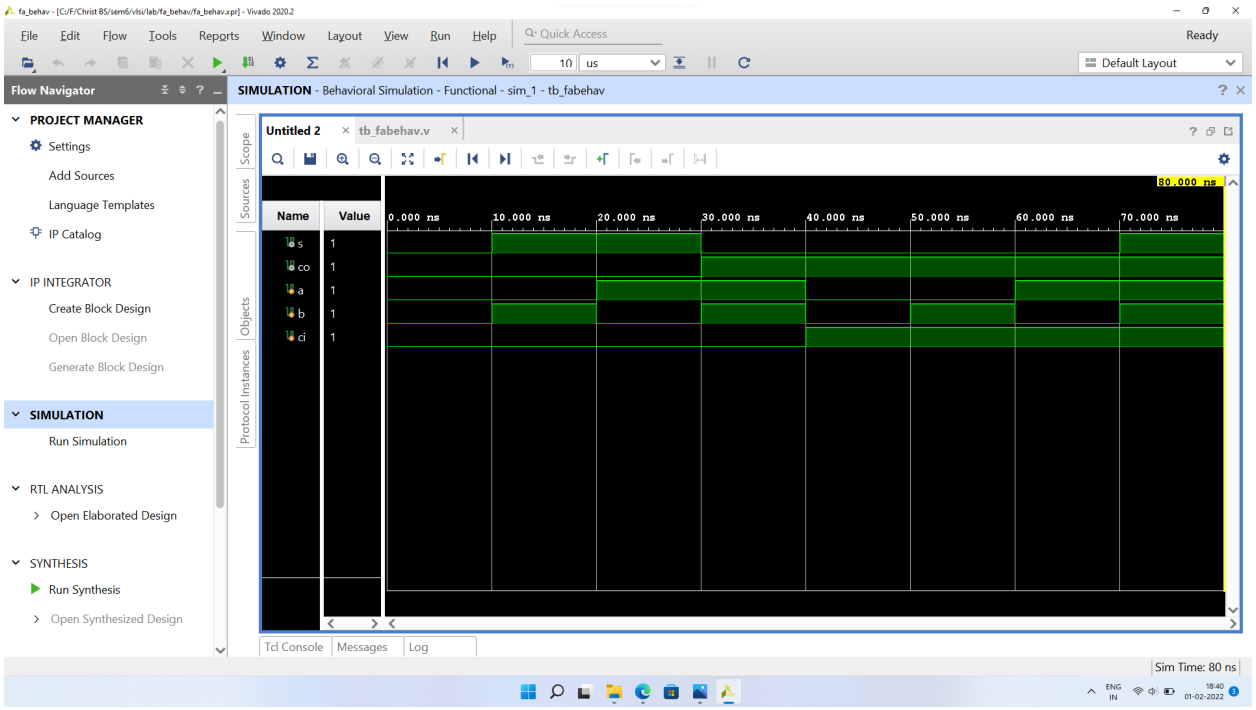
Behavioural testbench

```
//1960628 Prem Testbench Full adder
module tb_fabehav();
wire s,co;
reg a,b,ci;

fa_behav I1(a,b,ci,s,co);
initial
begin
a = 1'b0;
b = 1'b0;
ci = 1'b0;
#10
a = 1'b0;
b = 1'b1;
ci = 1'b0;
#10
a = 1'b1;
b = 1'b0;
ci = 1'b0;
#10
a = 1'b1;
b = 1'b1;
ci = 1'b0;
#10
a = 1'b0;
b = 1'b0;
ci = 1'b1;
#10
a = 1'b0;
b = 1'b1;
ci = 1'b1;
```

```
#10
a = 1'b1;
b = 1'b0;
ci = 1'b1;
#10
a = 1'b1;
b = 1'b1;
ci = 1'b1;
#10
$finish();
end
endmodule
```

OUTPUT

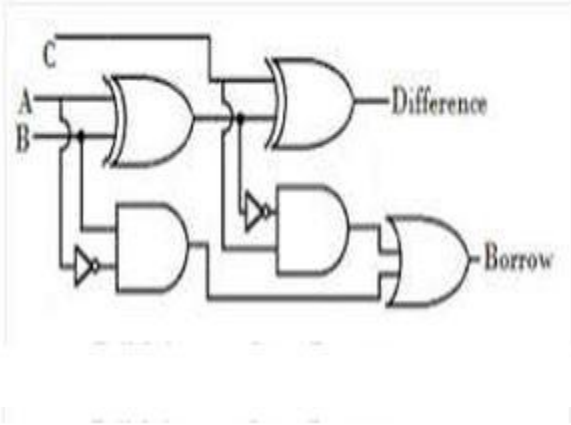


Result:

Thus the Verilog code was scripted, simulated and waveforms were verified using Xilinx Vivado Webpack.

Full Subtractor:

Logic Circuit



Truth Table

Full Subtractor-Truth Table				
Input			Output	
A	B	C	Difference	Borrow
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Expression

$$\textbf{Difference} = A \oplus B \oplus C$$

$$\textbf{Borrow} = A'B + BC + A'C$$

PROGRAM

Gate Level

```
//1960628 Prem full subtractor gate level
module fsgate(
    input a,
    input b,
    input c,
    output d,
    output bor
    );
wire w0,w1,w2,w3,w4;
xor xort(w0,a,b);
xor xort2(d,c,w0);
not nort(w1,w0);
not nort2(w2,a);
and and1(w3,w2,b);
and and2(w4,c,w1);
or ort(bor,w3,w4);
endmodule
```

Data Flow

```
//1960628 Prem full subtractor dataflow
module fs_dataflow(
    input a,
    input b,
    input c,
    output d,
    output bor
    );
assign d = a^b^c;
assign bor = ((~a)&b)|(b&c)|((~a)&c);
endmodule
```

Behavioural

```
//1960628 Prem Full Subtractor Behavioural
module fsbehav(
    input [2:0]a,
    output reg [1:0]d
    );
always@ (a,d)
    begin
    case(a)
    3'b000: begin d = 2'b00; end
    3'b001: begin d = 2'b11; end
    3'b010: begin d = 2'b11; end
    3'b011: begin d = 2'b01; end
    3'b100: begin d = 2'b10; end
    3'b101: begin d = 2'b00; end
    3'b110: begin d = 2'b00; end
    3'b111: begin d = 2'b11; end
    endcase
    end
endmodule
```

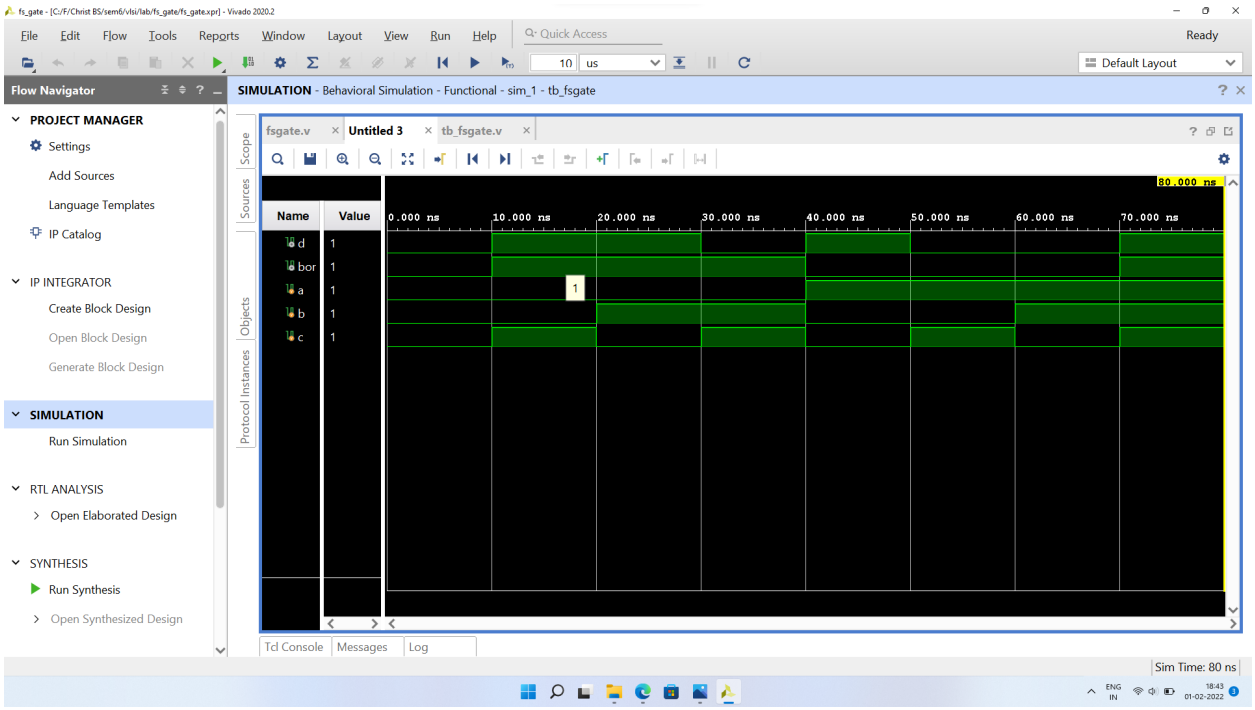
TESTBENCH

```
//1960628 Prem full adder testbemch
module tb_fsbehav();
wire [1:0]d;
reg [2:0]a;

fsbehav I1(a,d);
initial
begin
a = 3'b000;
#10
a = 3'b001;
#10
a = 3'b010;
#10
a = 3'b011;
#10
a = 3'b100;
#10
a = 3'b101;
#10
a = 3'b110;
#10
a = 3'b111;
#10
$finish();
end

endmodule
```

OUTPUT



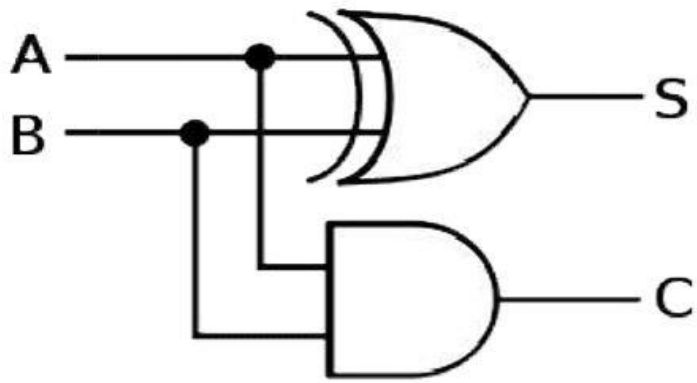
Result:

Thus the Verilog code was scripted, simulated and waveforms were verified using Xilinx Vivado Webpack.

Half Adder:

b) HALF ADDER:

Logic Circuit



Truth Table

Input		Output	
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Expression

$$\text{Sum} = A \oplus B$$

$$\text{Carry} = A.B$$

PROGRAM

Gate Level

```
//1960628 Prem half adder gate level
module hagate(
    input a,
    input b,
    output s,
    output c
    );
xor xort(s,a,b);
and and1(c,a,b);
endmodule
```

Data Flow

```
//1960628 Prem Half adder dataflow
module hadataflow(
    input a,
    input b,
    output s,
    output c
    );
assign s = a^b;
assign c = a&b;

endmodule
```

Behavioural

```
//1960628 prem half adder behavioural
module hsbehav(
    input [1:0]a,
    output reg [1:0]d
    );
always @(d,a)
    begin
        case(a)
        2'b00: begin d = 2'b00; end
        2'b01: begin d = 2'b11; end
        2'b10: begin d = 2'b10; end
        2'b11: begin d = 2'b00; end
        endcase
    end
endmodule
```

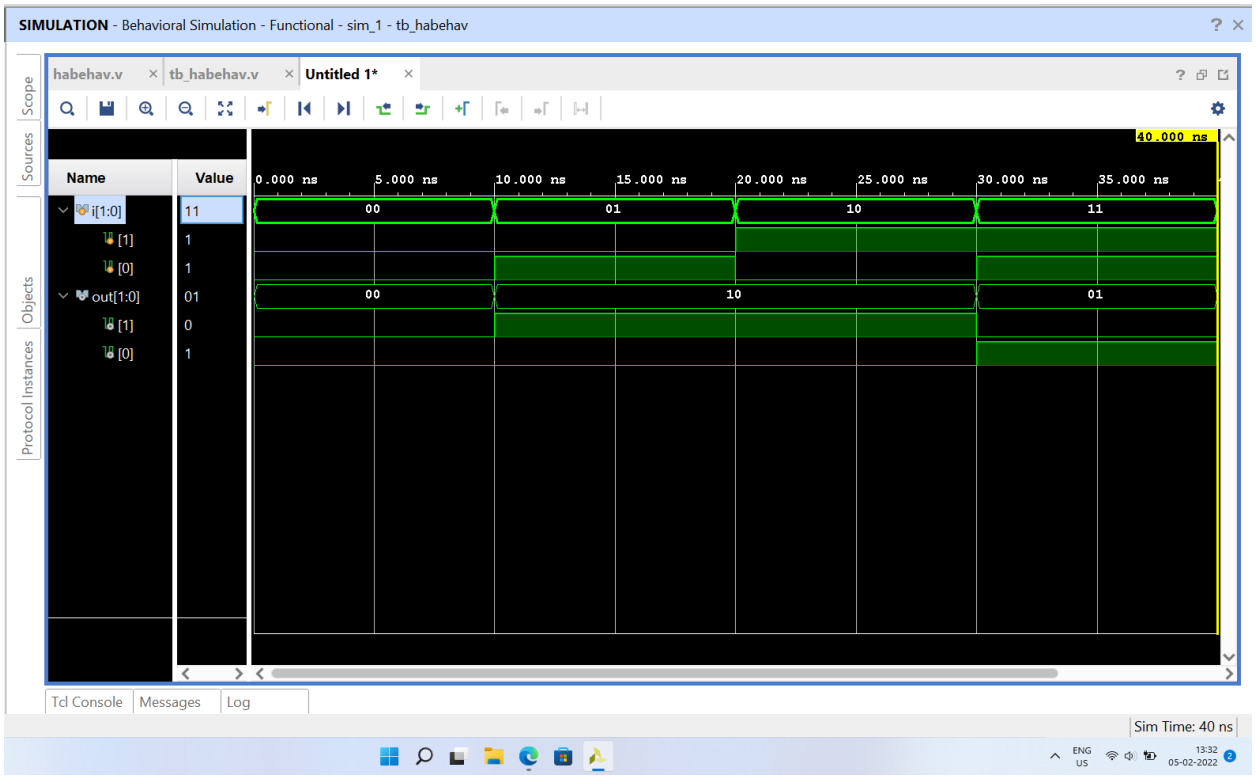
TESTBENCH

```
//1960628 Prem Half Adder testbench
module tb_hsbehav();
wire [1:0]d;
reg [1:0]a;

hsbehav I1(a,d);
initial
begin
a = 2'b00;
#10
a = 2'b01;
#10
a = 2'b10;
#10
a = 2'b11;
#10
$finish();
end

endmodule
```

OUTPUT



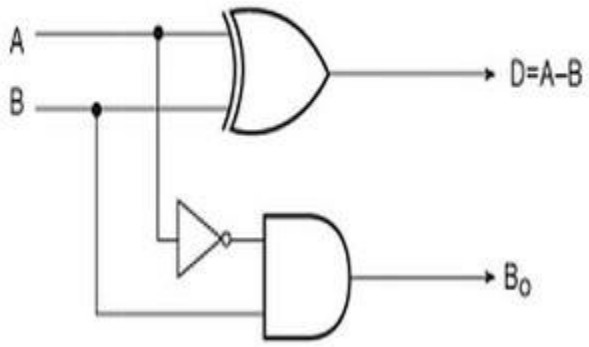
Result:

Thus the Verilog code was scripted, simulated and waveforms were verified using Xilinx Vivado Webpack.

Half Subtractor:

c) HALF SUBTRACTOR:

Logic Circuit



Truth Table

A	B	D	B_0
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Expression

$$D = A \oplus B$$

$$B = A'.B$$

PROGRAM

Gate Level

```
//1960628 Prem Half Subtracotr Gate level
module hsgate(
    input a,
    input b,
    output d,
    output bo
    );
wire w0;
xor xort(d,a,b);
not not1(w0,a);
and and1(bo,b,w0);
endmodule
```

Data Flow

```
//1960628 Prem Half Subtractor Data flow
module hsdata(
    input a,
    input b,
    output d,
    output bo
    );
assign d = a^b;
assign bo = (~a)&b;
endmodule
```

Behavioural

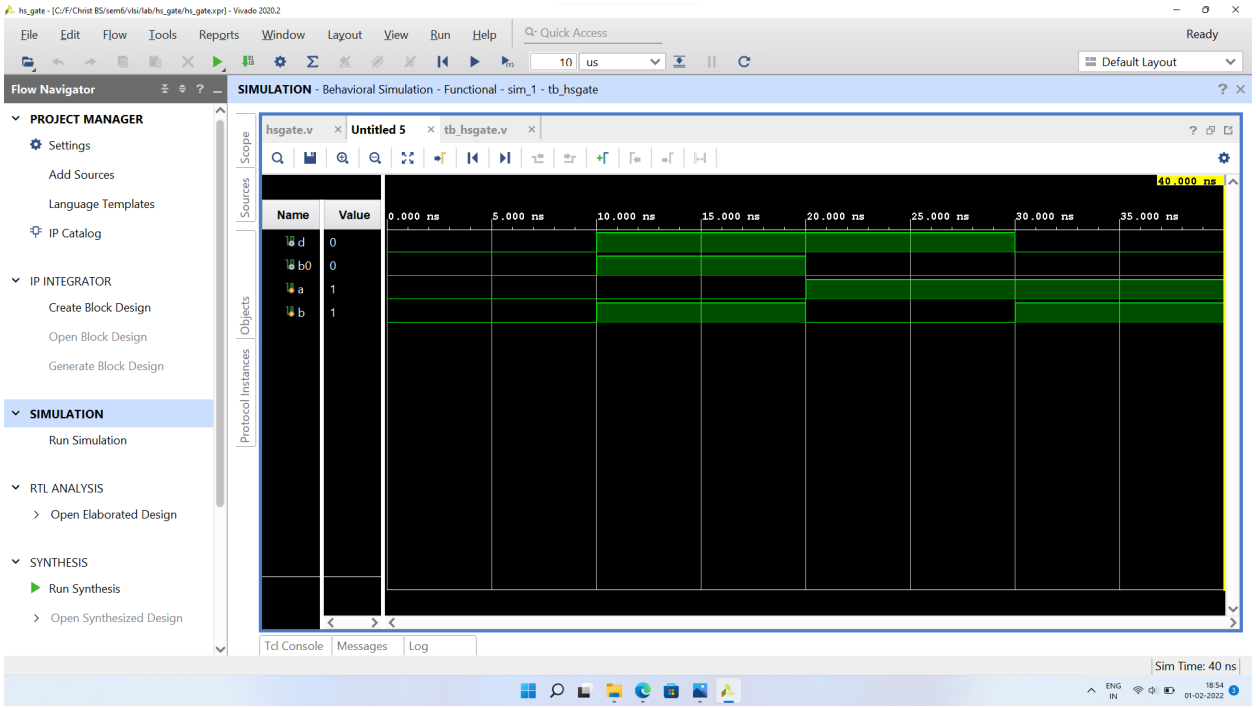
```
//1960628 prem half subtracotr behavioural
module hsbehav(
    input [1:0]a,
    output reg [1:0]d
    );
always @(d,a)
    begin
        case(a)
        2'b00: begin d = 2'b00; end
        2'b01: begin d = 2'b11; end
        2'b10: begin d = 2'b10; end
        2'b11: begin d = 2'b00; end
        endcase
    end
endmodule
```

TESTBENCH

```
//1960628 Prem Half Subtracotr testbench
module tb_hsgate();
wire d,b0;
reg a,b;

hsgate I1(a,b,d,b0);
initial
begin
a = 1'b0;
b = 1'b0;
#10
a = 1'b0;
b = 1'b1;
#10
a = 1'b1;
b = 1'b0;
#10
a = 1'b1;
b = 1'b1;
#10
$finish();
end
endmodule
```

OUTPUT



Result:

Thus the Verilog code was scripted, simulated and waveforms were verified using Xilinx Vivado Webpack.