Introduction to Combinational Circuit Simulation Lab: 2

Full Adder

Student : Chandani Lapasia

User ID : fpga0522-chan50

Code snippet for Full adder DATA flow modelling.

Design:

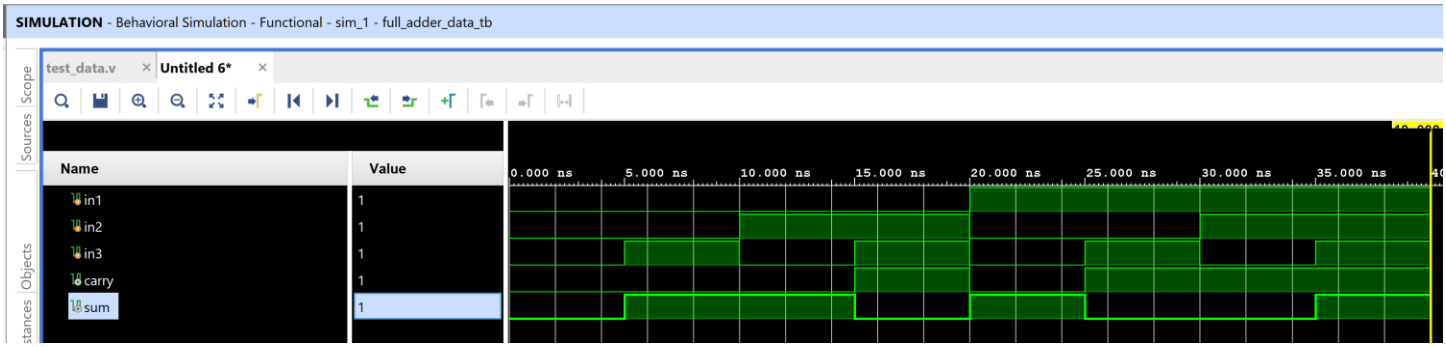
```
module full_adder_data(sum, carry,in1,in2,in3);
input in1, in2,in3;
output sum,carry;
assign sum= in1^in2^in3;
assign carry= (in1 && in2) || (in2 && in3) || (in3 && in1);
endmodule
```

Testbench:

```
module full_adder_data_tb();
wire sum,carry;
reg in1,in2,in3;
full_adder_data u0(sum,carry,in1,in2,in3);
initial begin
in1=0; in2=0; in3=0;
#5 in1=0; in2=0; in3=1;
#5 in1=0; in2=1; in3=0;
#5 in1=0; in2=1; in3=1;
#5 in1=1; in2=0; in3=0;
#5 in1=1; in2=0; in3=1;
#5 in1=1; in2=1; in3=0;
#5 in1=1; in2=1; in3=1;
#5 $finish;
```

```
end

initial begin

$monitor ($time, "\t" , "in1=%d in2=%d in3=%d carry=%d sum=%d", in1,in2,in3,carry,sum);

end

endmodule
```

Output waveform:



Output console :

```
 0   in1=0 in2=0 in3=0 carry=0 sum=0
 5   in1=0 in2=0 in3=1 carry=0 sum=1
10  in1=0 in2=1 in3=0 carry=0 sum=1
15   in1=0 in2=1 in3=1 carry=1 sum=0
20   in1=1 in2=0 in3=0 carry=0 sum=1
25   in1=1 in2=0 in3=1 carry=1 sum=0
30   in1=1 in2=1 in3=0 carry=1 sum=0
35   in1=1 in2=1 in3=1 carry=1 sum=1
```

Code snippet for Full adder BEHAVORAL flow modelling.

Design code :

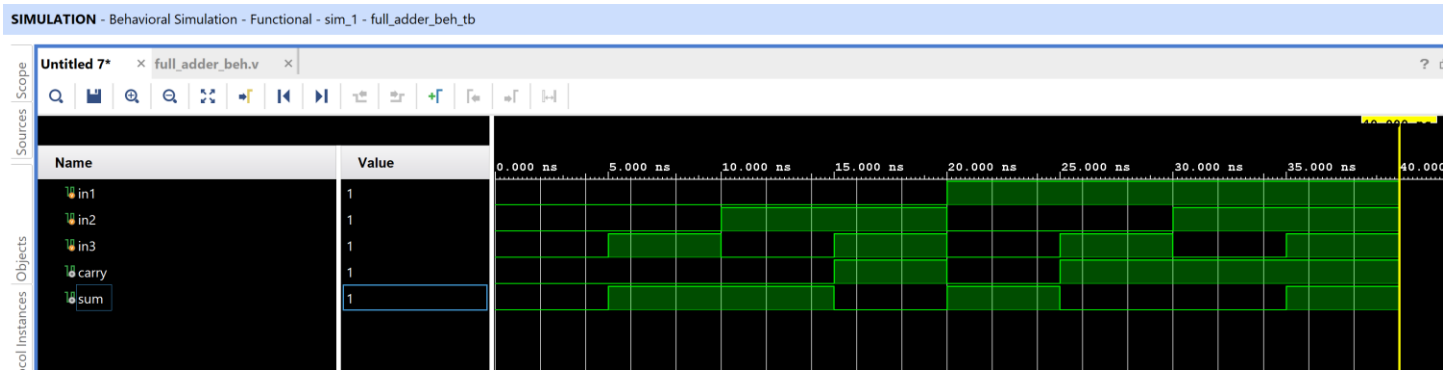
```
module full_adder_beh(sum,carry,in1,in2,in3);
input in1, in2,in3;
output sum,carry;
reg sum,carry;
always @(in1 or in2 or in3)
 begin
 sum= in1^in2^in3;
 carry = (in1 && in2) || (in2 && in3) || (in3 && in1);
 end
endmodule
```

Testbench code :

```
module full_adder_beh_tb();
wire sum,carry;
reg in1,in2,in3;

full_adder_beh u0(sum,carry,in1,in2,in3);

initial begin
in1=0; in2=0; in3=0;
#5 in1=0; in2=0; in3=1;
#5 in1=0; in2=1; in3=0;
#5 in1=0; in2=1; in3=1;
#5 in1=1; in2=0; in3=0;
#5 in1=1; in2=0; in3=1;
#5 in1=1; in2=1; in3=0;
#5 in1=1; in2=1; in3=1;
#5 $finish;
```

```
end

initial begin

$monitor ($time, "\t" , "in1=%d in2=%d in3=%d carry=%d sum=%d", in1,in2,in3,carry,sum);

end

endmodule
```

Output waveform :



Output console :

```
0  in1=0 in2=0 in3=0 carry=0 sum=0

5     in1=0 in2=0 in3=1 carry=0 sum=1

10  in1=0 in2=1 in3=0 carry=0 sum=1

15     in1=0 in2=1 in3=1 carry=1 sum=0

20     in1=1 in2=0 in3=0 carry=0 sum=1

25     in1=1 in2=0 in3=1 carry=1 sum=0

30     in1=1 in2=1 in3=0 carry=1 sum=0

35     in1=1 in2=1 in3=1 carry=1 sum=1
```

Code snippet for Full adder STRUCTURAL flow modelling.

Design code :

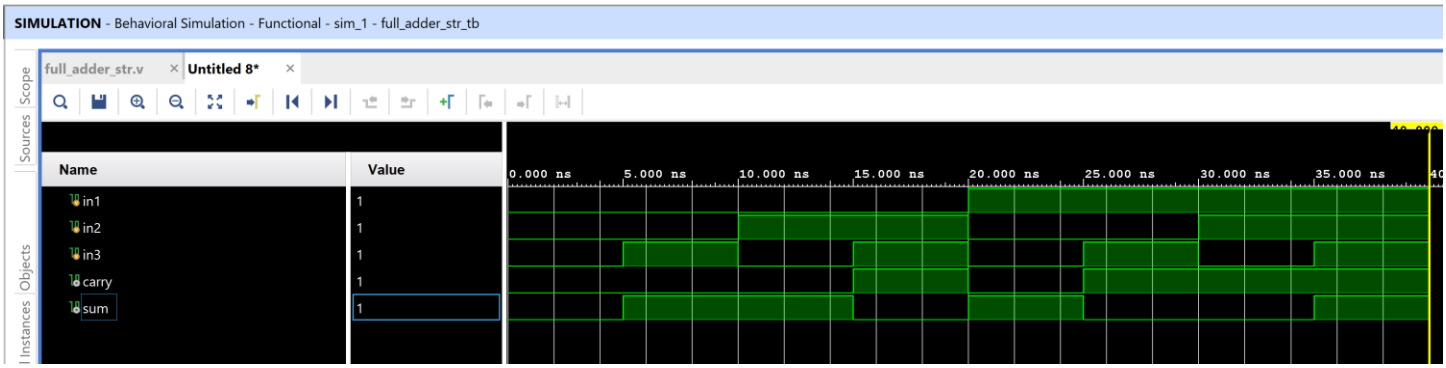
```
module full_adder_str(sum,carry,in1,in2,in3);
input in1, in2,in3;
output sum,carry;
wire sum,carry;
wire temp1,temp2,temp3,temp4;
xor(sum,temp1,in3);
xor(temp1, in1,in2);
or(carry,temp2,temp3,temp4);
and(temp4,in1,in3);
and(temp3,in2,in3);
and(temp2,in1,in2);
endmodule
```

Testbench code :

```
module full_adder_str_tb();
wire sum,carry;
reg in1,in2,in3;
full_adder_str u0(sum,carry,in1,in2,in3);
initial begin
in1=0; in2=0; in3=0;
#5 in1=0; in2=0; in3=1;
#5 in1=0; in2=1; in3=0;
#5 in1=0; in2=1; in3=1;
#5 in1=1; in2=0; in3=0;
#5 in1=1; in2=0; in3=1;
#5 in1=1; in2=1; in3=0;
#5 in1=1; in2=1; in3=1;
#5 $finish;
```

```
end

initial begin

$monitor ($time, "\t" , "in1=%d in2=%d in3=%d carry=%d sum=%d", in1,in2,in3,carry,sum);

end

endmodule
```

Output waveform :



Output console :

```
 0  in1=0 in2=0 in3=0 carry=0 sum=0
 5  in1=0 in2=0 in3=1 carry=0 sum=1
10  in1=0 in2=1 in3=0 carry=0 sum=1
15  in1=0 in2=1 in3=1 carry=1 sum=0
20  in1=1 in2=0 in3=0 carry=0 sum=1
25  in1=1 in2=0 in3=1 carry=1 sum=0
30  in1=1 in2=1 in3=0 carry=1 sum=0
35  in1=1 in2=1 in3=1 carry=1 sum=1
```

Question and Answers:

1. What is meant by combinational circuits?
   Combinational Logic Circuits are memoryless digital logic circuits whose output at any instant in time depends only on the combination of its inputs. The outputs of Combinational Logic Circuits are only determined by the logical function of their current input state, logic “0” or logic “1”, at any given instant in time. The result is that combinational logic circuits have no feedback, and any changes to the signals being applied to their inputs will immediately have an effect at the output.

2. Write the sum and carry expression for half and full adder.

Half adder :
Sum = A ^ B ; (read as A xor B)
Carry = AB (read as A and B)

Full Adder :
Sum= A^B^C; (read as A xor B xor C)
Carry =AB + BC + CA (read as (A and B) or (B and C) or (A and C))

3. What is signal? How it is declared?

A digital signal is a signal that represents data as a sequence of discrete values. The values can be in form of logic 0 and logic 1.

Signals can be declared as variables. The datatype of the signal can be reg, integer, int etc.

Example :

```
reg signal1 = 1'b0;
integer signal2=1'b1;
```

========== END==========