Digital Design and Computer Organization Lab

WEEK 3

22 Sept 2021

Vanshika Goel	**PES1UG20CS484**	H Section	Roll No: 40

Arithmetic and Logic Unit(ALU)

lib.v

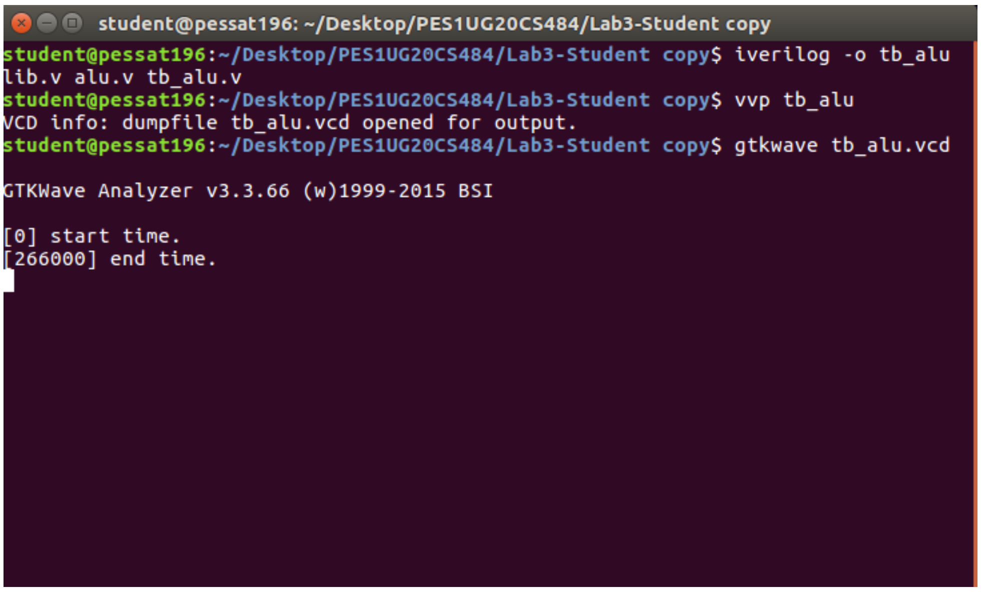
```
module invert (input wire i, output wire o);
   assign o = !i;
endmodule

module and2 (input wire i0, i1, output wire o);
  assign o = i0 & i1;
endmodule

module or2 (input wire i0, i1, output wire o);
  assign o = i0 | i1;
endmodule

module xor2 (input wire i0, i1, output wire o);
  assign o = i0 ^ i1;
endmodule

module nand2 (input wire i0, i1, output wire o);
   wire t;
   and2 and2_0 (i0, i1, t);
   invert invert_0 (t, o);
endmodule

module nor2 (input wire i0, i1, output wire o);
   wire t;
   or2 or2_0 (i0, i1, t);
   invert invert_0 (t, o);
endmodule

module xnor2 (input wire i0, i1, output wire o);
   wire t;
   xor2 xor2_0 (i0, i1, t);
   invert invert_0 (t, o);
endmodule

module and3 (input wire i0, i1, i2, output wire o);
   wire t;
   and2 and2_0 (i0, i1, t);
   and2 and2_1 (i2, t, o);
endmodule

module or3 (input wire i0, i1, i2, output wire o);
   wire t;
   or2 or2_0 (i0, i1, t);
   or2 or2_1 (i2, t, o);
endmodule

module nor3 (input wire i0, i1, i2, output wire o);
   wire t;
   or2 or2_0 (i0, i1, t);
   nor2 nor2_0 (i2, t, o);
endmodule

module nand3 (input wire i0, i1, i2, output wire o);
   wire t;
   and2 and2_0 (i0, i1, t);
   nand2 nand2_1 (i2, t, o);
endmodule

module xor3 (input wire i0, i1, i2, output wire o);
   wire t;
   xor2 xor2_0 (i0, i1, t);
   xor2 xor2_1 (i2, t, o);
endmodule

module xnor3 (input wire i0, i1, i2, output wire o);
   wire t;
   xor2 xor2_0 (i0, i1, t);
   xnor2 xnor2_0 (i2, t, o);
endmodule

module mux2 (input wire i0, i1, j, output wire o);
  assign o = (j==0)?i0:i1;
endmodule

module mux4 (input wire [0:3] i, input wire j1, j0, output wire o);
  wire  t0, t1;
  mux2 mux2_0 (i[0], i[1], j1, t0);
  mux2 mux2_1 (i[2], i[3], j1, t1);
  mux2 mux2_2 (t0, t1, j0, o);
endmodule

module mux8 (input wire [0:7] i, input wire j2, j1, j0, output wire o);
  wire  t0, t1;
  mux4 mux4_0 (i[0:3], j2, j1, t0);
  mux4 mux4_1 (i[4:7], j2, j1, t1);
  mux2 mux2_0 (t0, t1, j0, o);
endmodule
```

alu.v

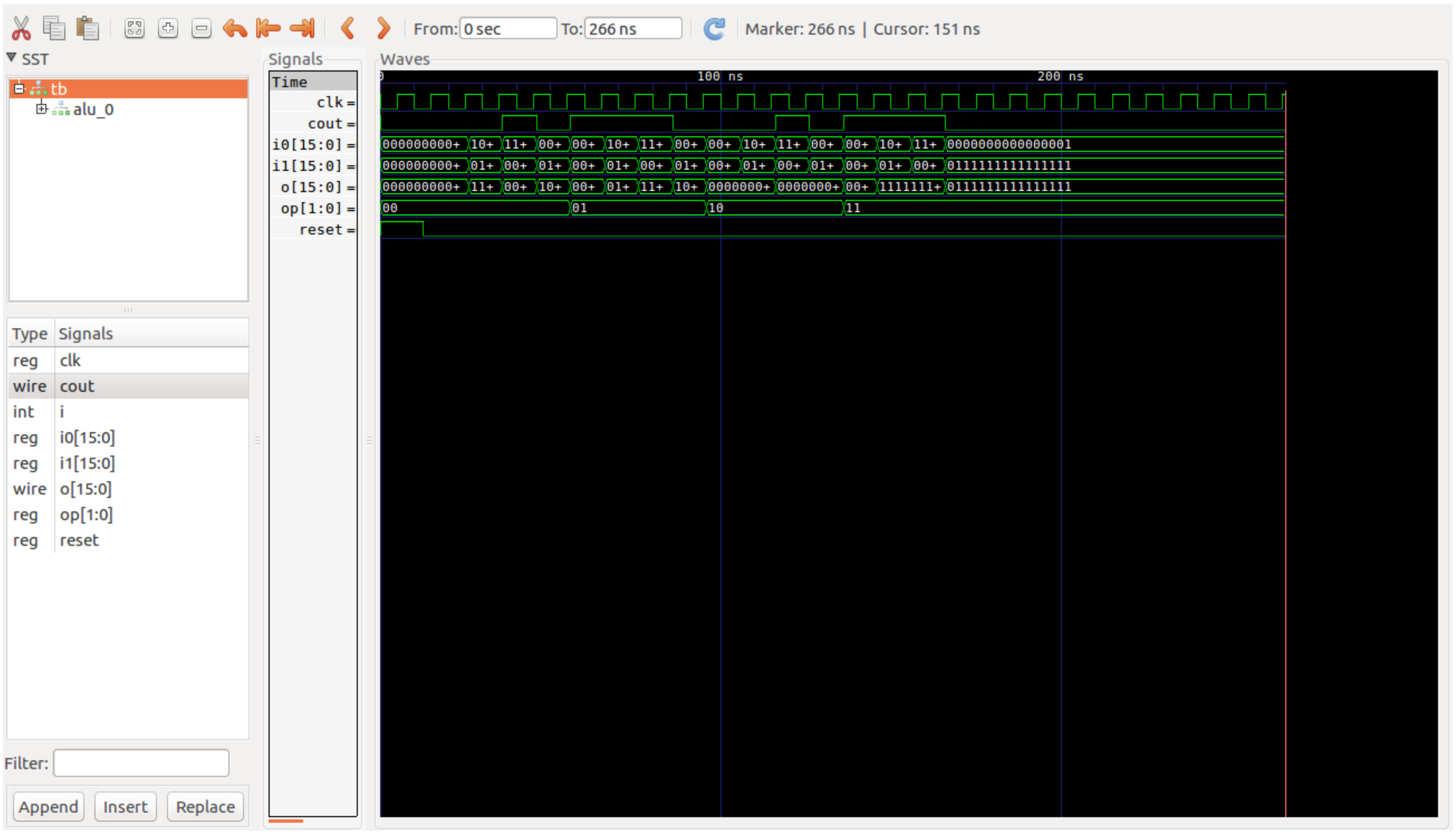
```
module fulladd(input wire a, b, cin, output wire sum, cout);
wire [4:0] t;
```

```
   xor2 x0(a, b, t[0]);
   xor2 x1(t[0], cin, sum);

   and2 a0(a, b, t[1]);
   and2 a1(a, cin, t[2]);
   and2 a2(b, cin, t[3]);

   or2 o0(t[1], t[2], t[4]);
   or2 o1(t[3], t[4], cout);
endmodule

module alu_slice(input wire [1:0]op,input wire a,b,c,output wire o,c1);
wire [4:0] inter;
          xor2 X(b,op[0],inter[0]);
          fulladd F1(a,inter[0],c,inter[1],c1);
          and2 A(a,b,inter[2]);
          or2 O(a,b,inter[3]);
          mux2 M1(inter[2],inter[3],op[0],inter[4]);
          mux2 M2(inter[1],inter[4],op[1],o);
endmodule

module alu (input wire [1:0] op, input wire [15:0] i0, i1,
   output wire [15:0] o, output wire cout);
wire [14:0] c;
alu_slice I1(op,i0[0],i1[0],op[0],o[0],c[0]);
alu_slice I2(op,i0[1],i1[1],c[0],o[1],c[1]);
alu_slice I3(op,i0[2],i1[2],c[1],o[2],c[2]);
alu_slice I4(op,i0[3],i1[3],c[2],o[3],c[3]);
alu_slice I5(op,i0[4],i1[4],c[3],o[4],c[4]);
alu_slice I6(op,i0[5],i1[5],c[4],o[5],c[5]);
alu_slice I7(op,i0[6],i1[6],c[5],o[6],c[6]);
alu_slice I8(op,i0[7],i1[7],c[6],o[7],c[7]);
alu_slice I9(op,i0[8],i1[8],c[7],o[8],c[8]);
alu_slice I10(op,i0[9],i1[9],c[8],o[9],c[9]);
alu_slice I11(op,i0[10],i1[10],c[9],o[10],c[10]);
alu_slice I12(op,i0[11],i1[11],c[10],o[11],c[11]);
alu_slice I13(op,i0[12],i1[12],c[11],o[12],c[12]);
alu_slice I14(op,i0[13],i1[13],c[12],o[13],c[13]);
alu_slice I15(op,i0[14],i1[14],c[13],o[14],c[14]);
alu_slice I16(op,i0[15],i1[15],c[14],o[15],cout);

endmodule
```

Output:


student@pessat196: ~/Desktop/PES1UG20CS484/Lab3-Student copy

```
student@pessat196:~/Desktop/PES1UG20CS484/Lab3-Student copy$ iverilog -o tb_alu
lib.v alu.v tb_alu.v
student@pessat196:~/Desktop/PES1UG20CS484/Lab3-Student copy$ vvp tb_alu
VCD info: dumpfile tb_alu.vcd opened for output.
student@pessat196:~/Desktop/PES1UG20CS484/Lab3-Student copy$ gtkwave tb_alu.vcd

GTKWave Analyzer v3.3.66 (w)1999-2015 BSI

[0] start time.
[266000] end time.
```



Type	Signals
reg	clk
wire	cout
int	i
reg	i0[15:0]
reg	i1[15:0]
wire	o[15:0]
reg	op[1:0]
reg	reset