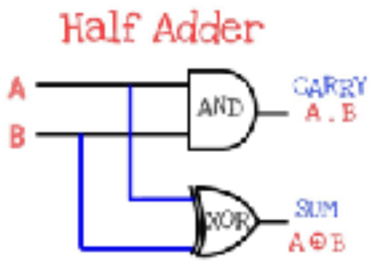
Amrita Vishwa Vidyapeetham
Amrita School of Computing, Bangalore
Department of Computer Science and Engineering

19CSE211 Computer Organization and Architecture
Lab Handout - 3
Combinational Circuits
Adder/Subtractor and Multiplexer/De-Multiplexer

Exercise Problems

1. Write a verilog code to implement Half Adder using gate level modeling



```
module half_adder_gl(a, b, sum,
carry);
input a, b;
output sum, carry;
and g1(carry, a, b);
xor g2(sum, a, b);
endmodule
```

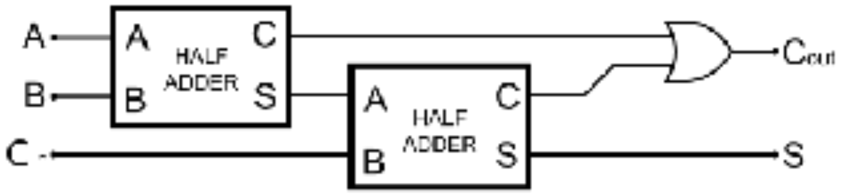
```
module half_adder_gl_tb;

reg a, b;
wire sum, carry;

half_adder_gl i(a, b, sum,
carry);

initial
begin
    a = 1'b0;
    b = 1'b0;
    $monitor("Time:%0t a=%b
b=%b sum=%b carry=%b", $time,
a, b, sum, carry);
    #5 a=1'b0; b=1'b0;
    #5 a=1'b0; b=1'b1;
    #5 a=1'b1; b=1'b0;
    #5 a=1'b1; b=1'b1;
end
endmodule
```

2. Write a verilog code to implement Full Adder using Half Added. Use the half adder module in Q.No 1



```
module half_adder_gl(a, b, sum,
carry);
input a, b;
output sum, carry;
and g1(carry, a, b);
xor g2(sum, a, b);
endmodule

module
full_adder_using_half_adder(a,
b, c, sum, carry);
input a, b, c;
output sum, carry;
wire w1, w2, w3;

half_adder_gl i1(a, b, w1, w2);
half_adder_gl i2(w1, c, sum,
w3);
or g1(carry, w2, w3);

endmodule
```

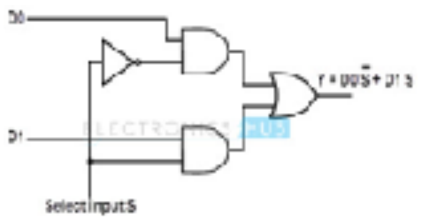
```
module
full_adder_using_half_adder_tb;

reg a, b, c;
wire sum, carry;

full_adder_using_half_adder
i(a, b, c, sum, carry);

initial
begin
    a = 1'b0;
    b = 1'b0;
    c = 1'b0;
    $monitor("Time:%0t a=%b
b=%b c=%b sum=%b carry=%b",
$time, a, b, c, sum, carry);
    #5 a=1'b0; b=1'b0; c=1'b0;
    #5 a=1'b0; b=1'b0; c=1'b1;
    #5 a=1'b0; b=1'b1; c=1'b0;
    #5 a=1'b0; b=1'b1; c=1'b1;
    #5 a=1'b1; b=1'b0; c=1'b0;
    #5 a=1'b1; b=1'b0; c=1'b1;
    #5 a=1'b1; b=1'b1; c=1'b0;
    #5 a=1'b1; b=1'b1; c=1'b1;
end
endmodule
```

3. Write a verilog code to implement 2:1 multiplexer using gate level modeling



```
module mux2to1 (d0, d1, s, y);
input d0, d1, s;
output y;
wire sbar, w1, w2;

not g1 (sbar, s);
and g2 (w1, d0, sbar);
and g3 (w2, d1, s);
or g4(y, w1, w2);

endmodule
```

```
module mux2to1_tb;
reg d0, d1, s;
wire y;

mux2to1 i(d0, d1, s, y);

initial
begin
    s=1'b0; d0 = 1'b0; d1 =
1'b0;
    $monitor("Time:%f, s:%b
d0:%b d1:%b y:%b", $time, s,
d0, d1, y);
    #5 s=1'b0; d0 = 1'b1; d1 =
1'b0;
    #5 s=1'b0; d0 = 1'b0; d1 =
1'b1;
    #5 s=1'b1; d0 = 1'b1; d1 =
1'b0;
    #5 s=1'b1; d0 = 1'b0; d1 =
1'b1;
end

endmodule
```

4. Write a verilog code to implement 4:1 multiplexer using data flow modeling.

$$Y = S0'.S1'.I0 + S0'.S1.I1 + S0.S1'.I2 + S0.S1.I3$$

```
module mux4to1_dl(s0, s1,
i0, i1, i2, i3, y);
input s0, s1, i0, i1, i2,
i3;
output y;

assign y =
((~s0)&(~s1)&i0)|
((~s0)&(s1)&i1)|
((~0)&(~s1)&i2)|
((s0)&(s1)&i3);

endmodule
```

```
module mux4to1_dl_tb;

reg s0, s1, i0, i1, i2, i3;
wire y;

mux4to1_dl i(s0, s1, i0, i1, i2, i3,
y);

initial
begin
    s0 = 1'b0; s1 = 1'b0; i0 = 1'b0;
i1 = 1'b0; i2 = 1'b0; i3 = 1'b0;
    $monitor ("Time:%f, s0:%b s1:%b
i0:%b i1:%b i2:%b i3:%b y:%b",$time,
s0, s1, i0, i1, i2, i3, y);
    #5 s0 = 1'b0; s1 = 1'b0; i0 =
1'b0; i1 = 1'b0; i2 = 1'b0; i3 = 1'b0;
    #5 s0 = 1'b0; s1 = 1'b0; i0 =
1'b1; i1 = 1'b0; i2 = 1'b0; i3 = 1'b0;
    #5 s0 = 1'b0; s1 = 1'b1; i0 =
1'b0; i1 = 1'b0; i2 = 1'b0; i3 = 1'b0;
    #5 s0 = 1'b0; s1 = 1'b1; i0 =
1'b0; i1 = 1'b1; i2 = 1'b0; i3 = 1'b0;
    #5 s0 = 1'b1; s1 = 1'b0; i0 =
1'b0; i1 = 1'b0; i2 = 1'b0; i3 = 1'b0;
    #5 s0 = 1'b1; s1 = 1'b0; i0 =
1'b0; i1 = 1'b0; i2 = 1'b1; i3 = 1'b0;
    #5 s0 = 1'b1; s1 = 1'b1; i0 =
1'b0; i1 = 1'b0; i2 = 1'b0; i3 = 1'b0;
    #5 s0 = 1'b1; s1 = 1'b1; i0 =
1'b0; i1 = 1'b0; i2 = 1'b0; i3 = 1'b1;
end
endmodule
```

Assignment

1. Write a verilog code to implement half adder using data flow modeling.
2. Write a verilog code to implement half subtractor using gate level modeling
3. Write a verilog code to implement full subtractor using data flow modeling
4. Write a verilog code to implement 16:1 multiplexer using 4:1 multiplexer. Implement 4:1 multiplexer using gate level modeling.
5. Write a verilog code to implement 1:4 de-multiplexer using data flow modeling.
6. Implement the below expression using 2:1 multiplexer.

$$F = a'b' + a'b$$