ELEC4320 Homework 1 Fall 2019

Student: LEE Lok Yin (20450512)

1a)

Clock period: 1/(200\*106)=5e-9 s

Total execution cycles: [1\*10+ (2\*1023) \*1]\*5=2057

Total execution time: 5e-9\*2057=1.0285e-5 s

Overall throughput: 16\*2\*1024/(1.0285\*10-5)=3.186e9 bits/second.

Maximum throughput: 16/5e-9=3.2e9 bits/second.

1b)

Final requirement = 2000 ICs of 50k gates

For FPGA,

Salary: 2\*10k = $20k

Direct material cost: 2000 \* $100 = $200k

Total cost: $220k

For 6’ Std Cell,

Salary: 20\*10k = $200k

Wafer Area: 5 mm2 (10k/mm2)

Overhead: 5\*0.3 = 1.5 mm2

No. of ICs per wafer: 15,000/6.5 = ~2307

Material cost: $3k

Mask cost: $200k

Total cost: $403k

For 8’ Custom,

Salary: 30\*10k = $300k

Wafer Area: 2.5 mm2

Overhead: 0.75 mm2

No. of ICs per wafer: 24,000/3.25 = ~7384

Material cost: $6k

Mask cost: $400k

Total cost: $706k

Thus, gate array is the best choice.

2a)

`timescale 1ns / 1ps

module GrayCode

(

clk,

rst,

grayout

);

input clk;

input rst;

output reg [3:0] grayout;

reg [3:0] current\_state;

reg [3:0] next\_state;

initial

begin

current\_state = 0;

next\_state = 0;

grayout = 0;

end

always @(\*)

begin

if (rst == 1'b1)

begin

grayout = 0;

end

else

begin

case (current\_state)

'D0:begin

grayout = 4'b0000;

next\_state = 'D1;

end

'D1:begin

grayout = 4'b0001;

next\_state = 'D2;

end

'D2:begin

grayout = 4'b0011;

next\_state = 'D3;

end

'D3:begin

grayout = 4'b0010;

next\_state = 'D4;

end

'D4:begin

grayout = 4'b0110;

next\_state = 'D5;

end

'D5:begin

grayout = 4'b0111;

next\_state = 'D6;

end

'D6:begin

grayout = 4'b0101;

next\_state = 'D7;

end

'D7:begin

grayout = 4'b0100;

next\_state = 'D8;

end

'D8:begin

grayout = 4'b1100;

next\_state = 'D9;

end

'D9:begin

grayout = 4'b1101;

next\_state = 'D10;

end

'D10:begin

grayout = 4'b1111;

next\_state = 'D11;

end

'D11:begin

grayout = 4'b1110;

next\_state = 'D12;

end

'D12:begin

grayout = 4'b1010;

next\_state = 'D13;

end

'D13:begin

grayout = 4'b1011;

next\_state = 'D14;

end

'D14:begin

grayout = 4'b1001;

next\_state = 'D15;

end

'D15:begin

grayout = 4'b1000;

next\_state = 'D0;

end

endcase

end

end

always@(posedge clk)

begin

if(rst)

begin

current\_state <= 4'b0;

end

else

begin

current\_state <= next\_state;

end

end

endmodule

2b)

Full Adder:

`timescale 1ns / 1ps

module fulladder( A,B,Cin,Sum,Cout);

input A,B,Cin;

output Sum,Cout;

assign Sum = A ^ B ^ Cin;

assign Cout = A & B | B & Cin | A & Cin;

endmodule

Carry Ripple Adder:

`timescale 1ns / 1ps

module CRA( A,B,Cin,Sum,Cout);

input [3:0] A,B;

input Cin;

output [3:0] Sum;

output Cout;

wire Cin1,Cin2,Cin3;

fulladder add0( .A(A[0]),.B(B[0]),.Cin(Cin),.Sum(Sum[0]),.Cout(Cin1) );

fulladder add1( .A(A[1]),.B(B[1]),.Cin(Cin1),.Sum(Sum[1]),.Cout(Cin2) );

fulladder add2( .A(A[2]),.B(B[2]),.Cin(Cin2),.Sum(Sum[2]),.Cout(Cin3) );

fulladder add3( .A(A[3]),.B(B[3]),.Cin(Cin3),.Sum(Sum[3]),.Cout(Cout) );

endmodule

3a)

AND plane: 8x8=64 programmable elements

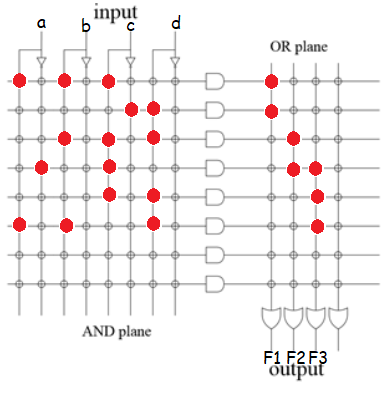
OR plane: 4x8=32 programmable elements

Total number of programmable elements: 96

F1 = abc + c’d

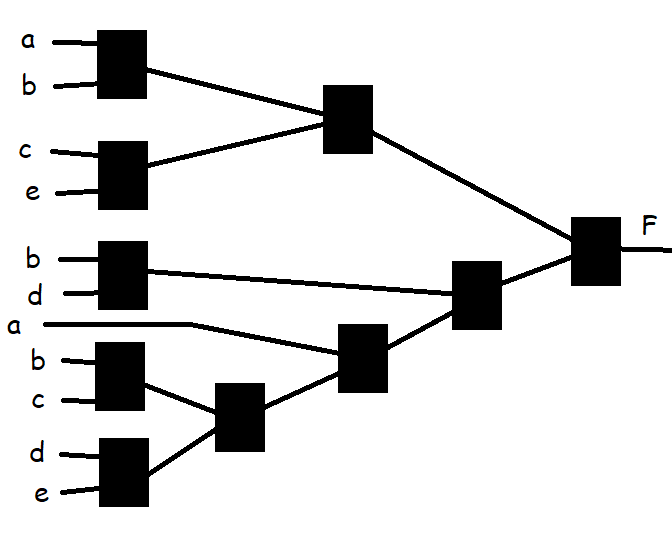
F2 = bcd + a’c

F3 = a’c + cd + abd



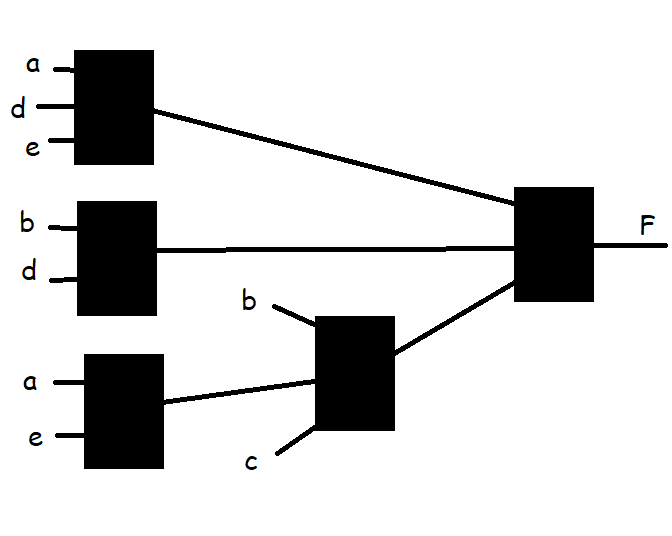
3b)

F = abc + ade + b’d + a’bce.

2-input LUTs: F = a(bc+de) + b’d + (a’b)(ce)

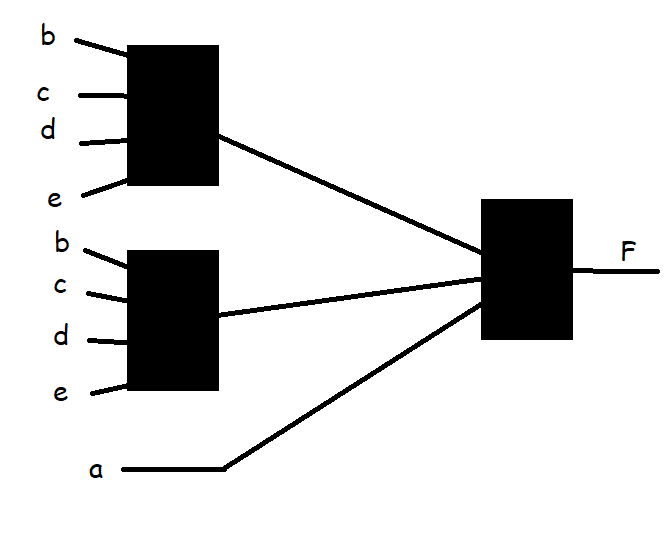
Need 10 cells.

3-input LUTs: F = (a+a’e)bc + ade + b’d



Need 5 cells.

4-input LUTs: F = a(bc + de + b’d) + a’(b’d + bce)



Need 3 cells.