COMP311-1: Logic Circuit Design

Fall 2019, Prof. Taigon Song

Project 2. Due: Dec. 2, 8:59am [Total: 90 + 10 points]

# [2017114559]

# 박선우 Park Sun Woo

|  |  |  |
| --- | --- | --- |
| No. | Checksheet item | Done? [Y/N] |
| 1. | RCA adder design (no delay, 10 points) | Y |
| 2. | CLA adder design (no delay, 20 points) | Y |
| 3. | CSA adder design 1 (no delay, 15 points) | Y |
| 4. | CSA adder design 2 (no delay, 5 points) | Y |
| 5. | Four adder design with delay (10 points) | Y |
| 6. | Tradeoff between area and speed (Synthesis report, 10 points) | Y |
| 7. | Delay result/analysis (10 points) | Y |
| 8. | Any other discussion (10 points) | Y |
| 9. | Bonus: Submission date (+ 10 points) | Y |

1. **RCA adder design**

rca16b.v , bit16adder.v , bit4adder.v , bit1fulladder.v

module rca16b(input[15:0] a1,b1, input cin, output[15:0] sum, output cout);

bit16adder u1(a1, b1, cin, sum, cout);

endmodule

module bit16adder(input[15:0] a, b, input c\_in, output[15:0] sum, output c\_out);

wire c1,c2,c3;

bit4adder a0(a[3:0],b[3:0],c\_in,sum[3:0],c1);

bit4adder a1(a[7:4],b[7:4],c1,sum[7:4],c2);

bit4adder a2(a[11:8],b[11:8],c2,sum[11:8],c3);

bit4adder a3(a[15:12],b[15:12],c3,sum[15:12],c\_out);

endmodule

module bit4adder(input[3:0] a, b,input c\_in, output[3:0] sum, output c\_out);

wire c1,c2,c3;

bit1fulladder a0(a[0],b[0],c\_in,sum[0],c1);

bit1fulladder a1(a[1],b[1],c1,sum[1],c2);

bit1fulladder a2(a[2],b[2],c2,sum[2],c3);

bit1fulladder a3(a[3],b[3],c3,sum[3],c\_out);

endmodule

`timescale 1ns/1ns

module bit1fulladder(input a,b,c\_in, output sum, c\_out);

wire s1,s2,c1;

and #7 a1(c1,a,b), a2(s2,s1,c\_in);

xor #6 x1(s1,a,b), x2(sum,s1,c\_in), x3(c\_out,s2,c1);

endmodule

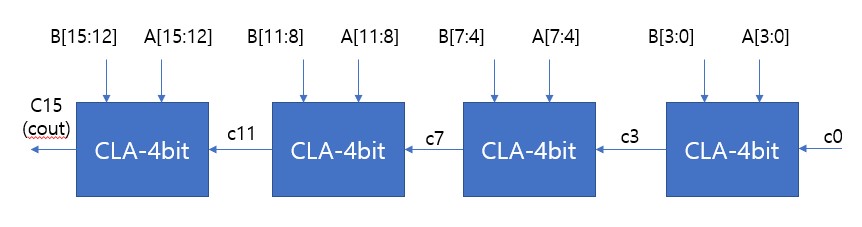
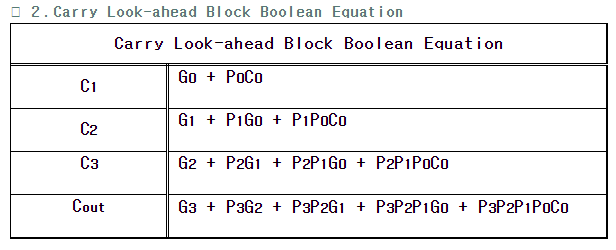
I reuse bit16adder in project1 and set delay in bit1fulladder.

1. **CLA adder design**

cla16b.v , cla4bit.v , clb.v

|  |
| --- |
| module cla16b(input[15:0] a1,b1, input cin, output[15:0] sum, output cout);  wire[2:0] c;  cla4bit u0(a1[3:0],b1[3:0],cin,sum[3:0],c[0]);  cla4bit u1(a1[7:4],b1[7:4],c[0],sum[7:4],c[1]);  cla4bit u2(a1[11:8],b1[11:8],c[1],sum[11:8],c[2]);  cla4bit u3(a1[15:12],b1[15:12],c[2],sum[15:12],cout);  endmodule  module cla4bit(input[3:0] a,b, input c\_in, output[3:0] sum, output c\_out);  wire[3:0] c;  clb c1(a,b,c\_in,c);  bit1fulladder a0(a[0],b[0],c\_in,sum[0],);  bit1fulladder a1(a[1],b[1],c[0],sum[1],);  bit1fulladder a2(a[2],b[2],c[1],sum[2],);  bit1fulladder a3(a[3],b[3],c[2],sum[3],);  assign c\_out = c[3];  endmodule  `timescale 1ns/1ns  module clb(input[3:0] a,b, input c, output[3:0] out);  wire[3:0] g,p;  wire s00, s10,s11, s20,s21,s22, s30,s31,s32,s33;    and #7 n0(g[0],a[0],b[0]), n1(g[1],a[1],b[1]), n2(g[2],a[2],b[2]), n3(g[3],a[3],b[3]);  xor #6 x0(p[0],a[0],b[0]), x1(p[1],a[1],b[1]), x2(p[2],a[2],b[2]), x3(p[3],a[3],b[3]);  or #7 o00(out[0],g[0],s00);  and #7 n00(s00,p[0],c);  or #7 o10(out[1],g[1],s10,s11);  and #7 n10(s10,p[1],g[0]), n11(s11,p[1],p[0],c);  or #7 o20(out[2],g[2],s20,s21,s22);  and #7 n20(s20,p[2],g[1]), n21(s21,p[2],p[1],g[0]), n22(s22,p[2],p[1],p[0],c);  or #7 o30(out[3],g[3],s30,s31,s32,s33);  and #7 n30(s30,p[3],g[2]), n31(s31,p[3],p[2],g[1]), n32(s32,p[3],p[2],p[1],g[0]), n33(s33,p[3],p[2],p[1],p[0],c);  endmodule |

First I make clb , In clb I make 4bit g, p by doing (a and b), (a xor b) and then, I make out[3:0] by carry look ahead block Boolean equation. After clb I make 4bit cla using clb and bit1fulladder. Finally, I make 16bit clb by connecting four 4bit cla.



1. **CSA adder design 1**

csa16b1.v , mux1b.v , mux16b.v

|  |
| --- |
| module csa16b1(input[15:0] a1,b1, input cin, output[15:0] sum, output cout);  wire[1:0] c;  wire[15:0] sum0,sum1;  cla16b c0(a1,b1,0,sum0,c[0]);  cla16b c1(a1,b1,1,sum1,c[1]);  mux1b m1(c[0],c[1],cin,cout);  mux16b m2(sum0,sum1,cin,sum);  endmodule  `timescale 1ns/1ns  module mux1b(input a1,b1,cin, output reg cout);  always@(\*)begin  if(cin==0)  #3 assign cout = a1;  else  #3 assign cout = b1;  end  endmodule  `timescale 1ns/1ns  module mux16b(input[15:0] a1,b1, input cin, output reg[15:0] cout);  always@(\*)begin  if(cin==0)  #3 assign cout = a1;  else  #3 assign cout = b1;  end  endmodule |

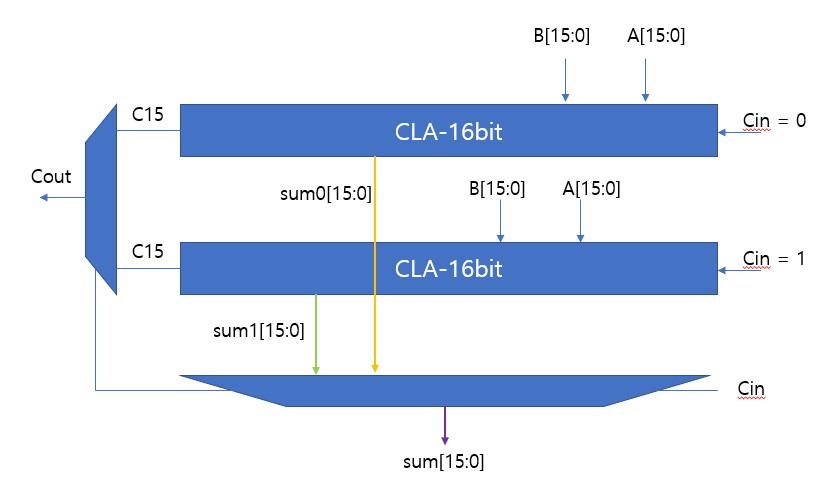
I make 1bit mux and 16bit mux then, I reuse cla16b 2 times one is cin = 0 , the other is cin = 1, and then I select the result using mux by cin signal, and cout also select by cin.

1. **CSA adder design 2**

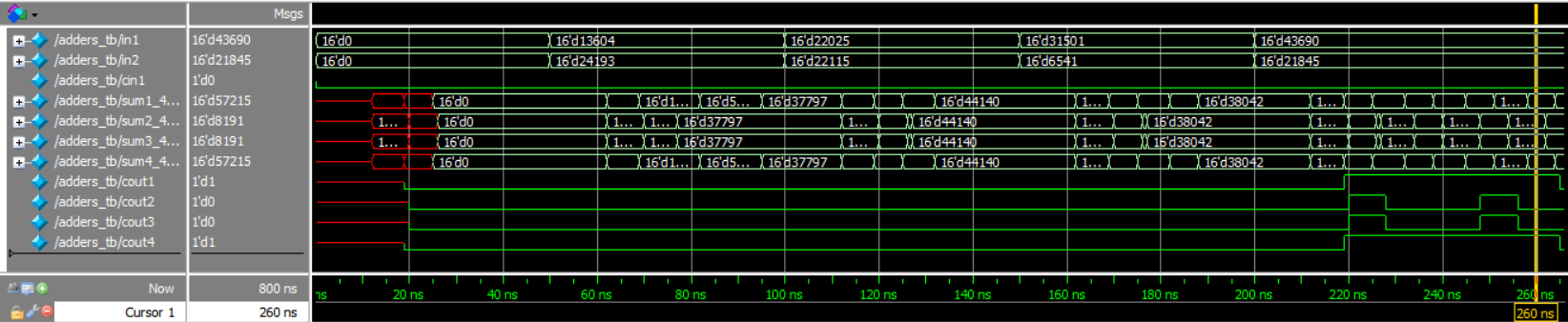
csa16b2.v

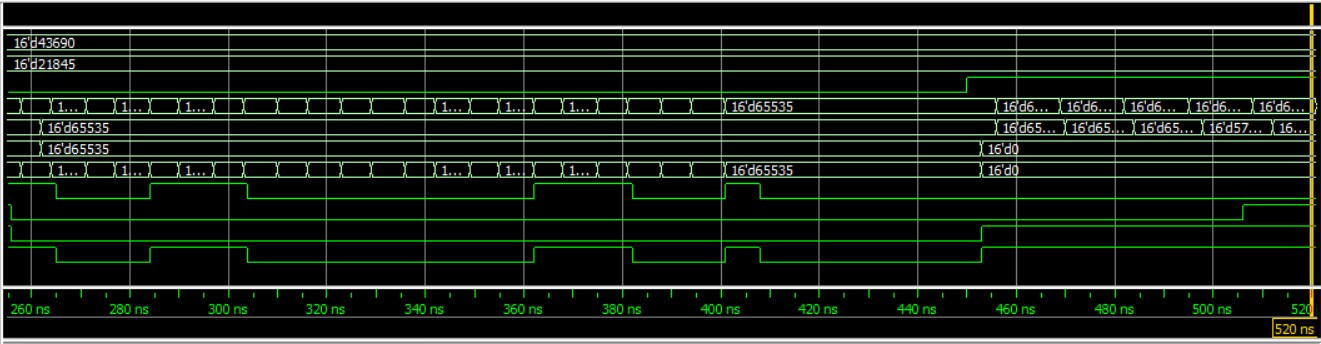
|  |
| --- |
| module csa16b2(input[15:0] a1,b1, input cin, output[15:0] sum, output cout);  wire[1:0] c;  wire[15:0] sum0,sum1;  bit16adder u0(a1, b1, 0, sum0, c[0]);  bit16adder u1(a1, b1, 1, sum1, c[1]);  mux1b m1(c[0],c[1],cin,cout);  mux16b m2(sum0,sum1,cin,sum);  endmodule |

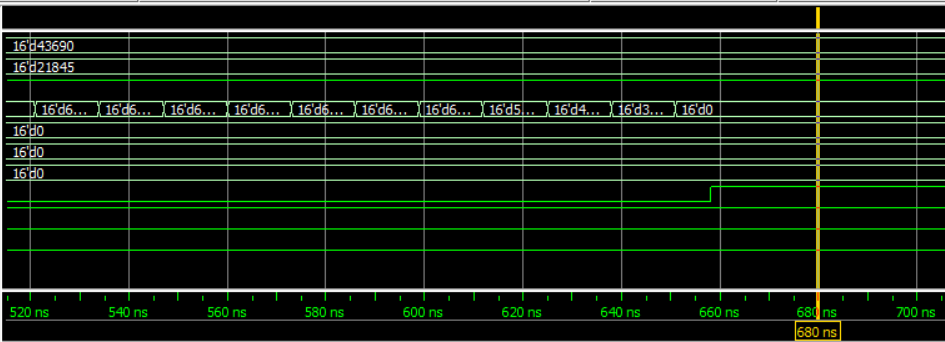
In csa 1 I replace 16bit cla to 16bit rca



**Figure 1 - 16-bit CLA for Project 2**

1. **Four adder with delay result**





1. **Tradeoff between area and speed**

RCA \_ 1bitfulladder : 3 xor , 2 and , 4bitadder = 4 \* 1bitfulladder ,

**RCA = 16bitadder = 4 \* 4bitadder = 4 \* 4 \* (3 xor , 2 and) = 48xor , 32and**

CLA \_ CLB = 14and, 4xor, 4or , CLA4bit = CLB+4\*1bitfulladder = 22and, 16xor, 4or

**CLA16bit = 4\* CLA4bit = 88and, 64xor, 16or**

**CSA16bit1** = 2\*CLA16bit + mux1bit (2and , 1or , 1not) + mux16bit(32and ,16or, 16not)

**= 210and , 128xor , 49or , 17not**

**CSA16bit2** = 2\*RCA16bit + mux1bit + mux16bit **= 98and , 96xor , 17or , 17not**

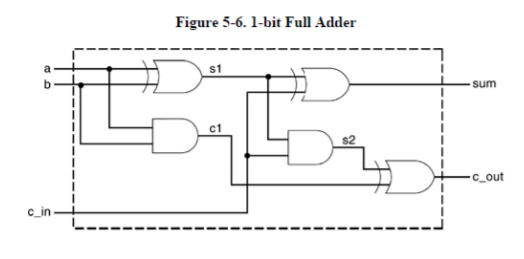
When running vivado I exclude all mux because there are error in assign

And I add two synthesis report csa16b1.vds , csa16b2.vds because in CSA1,2 there are included RCA, CLA

**Speed compare**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| delay(ns) | 0->37797 | -> 44140 | -> 38042 | -> 65535 | -> 0 (overflow) |
| RCA | 45 | 32 | 38 | 201 | 201 |
| CLA | 27 | 27 | 27 | 62 | 62 |
| CSA1 | 27 | 27 | 27 | 62 | 3 |
| CSA2 | 45 | 32 | 38 | 201 | 3 |

CSA is fastest in change cin (only 3ns) , and CLA is faster than RCA



1. **Delay result/analysis**

RCA\_ 1bitfulladder : sum = 12ns , cout = 19ns

4bitadder : sum[0] =12ns, sum[1] =19+6 =25, sum[2] = 32+6=38, sum[3]=45+6=52

Cout[0] = 19, cout[1] = 19+7+6=32, cout[2] = 32+7+6=45, count[3] = 45+7+6=58

16bitadder = RCA : sum = 19+13\*14+6=207ns , cout = 19+13\*15=214ns

CLA : 4bit = (CLB: 7+7+7=21ns) + (1bitfulladder 6ns) = 27ns

CLA16bit = 21+21+21+6=69ns

CSA1 delay is same as CLA because in CSA1 I reuse CLA

Also CSA2 dalay is same as RCA.

And there are 3ns delay in CSA when change cin. It`s right result because there are mux delay 3ns in CSA.

1. **Any other discussion**

The simulation result and my calculation are different. Why?

In RCA16bit my calculation is 207ns but simulation is 201ns and

CLA16bit calculation is 69ns but simulation is 62ns.

I calculate many times but I cant find any answer.

So I think there are some parallel operation in this sequence.

And maybe I miss it in my calculation.