

EHB436E

DIGITAL SYSTEM DESIGN APPLICATIONS

Muhammed Erkmen

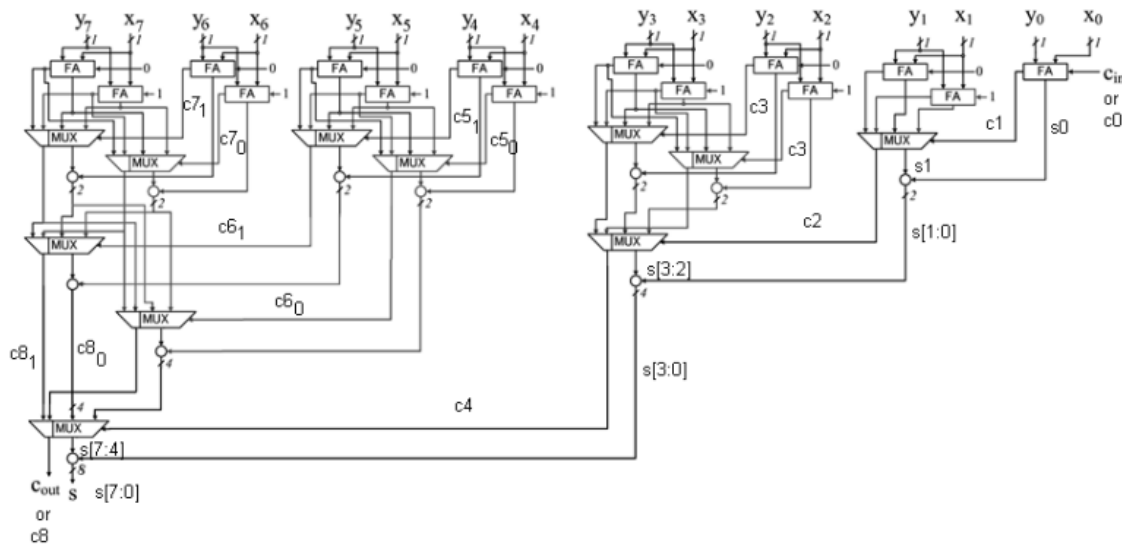
040170049

32-BIT CONDITIONAL SUM ADDER

PROJECT-1 REPORT

Conditional Sum Adder:

Idea of conditional sum adder is, calculating every possible carry and sum outputs for each one of 2-bits, then operating a selection with the previous bits' output.



First step is calculating every possible outputs for every 2-bit addition. To implement this operation, we use full adders. After that, for example for the x_3 and y_3 addition, we need to select which carry will come, so we use 2 multiplexers and give to these multiplexers as a selection bit x_2 and y_2 output carry. Then, we'll need to select which carry will come to x_2 and y_2 adding operation. And that is the third multiplexer's selection input is x_1 and y_1 adding operation's carry out which selected by x_0 and y_0 .

My designing steps:

So i designed this 32-bit conditional sum adder, first i designed an 8-bit conditional sum adder. If i designed it just 32-bit conditional without parsing the process, i would need more multiplexers. Because first 8 bit stage has 7 multiplexers, but if i designed it with connecting more than 8 bits, i would need more than $7*4=28$ multiplexers to give output. This design has less utilization, also i think the same amount of delay because if it was 32 bit connected, the next stage of 3 already needs to wait first 8 bits cout. So I think this design is better.

First stage is calculating the every possible output available, i did that by using full adders.

I put $2*k$ 'th bits outputs to the muxes, then i selected the value i will need with k 'th bits couts and added k th bits sum to the selected sum.

I did same thing in 3rd stage mux too. So i get 8 bit sum and 1 bit carry out.

After that I used 4 8-bit conditional sum adder by connected. Then connected the previous carry out to the next one. Included their sum values in a 32 bit wire respectively. And carry out of 4th conditional sum adder is the carry out of the system. **But because of i am designing this signed, this carry out has no meanings i guess.**

Adding overflow output is kind of changing the design. Because overflow output equals to last 2 carry's xor result. But in our design 2nd last carry comes from selections and its source is too above of the circuit.

Because of that reason i made this part behavioral, not structural.

```
always @(*) begin
if( (x+y > 2147483646) || (x+y<-2147483647))
v<=1;
else
v<=0;
end
```

Preparing a testbench:

I designed a testbench with 100 random binary pairs by using \$readmemb() function in testbench. I get random binary numbers from: <https://www.browserling.com/tools/random-bin>. I can write a python or c code or try to use \$random command in verilog testbench, but this would cost more time than using this website. But in assignment files a python file is needed so i created a python function to create binary numbers too.

Python Code:

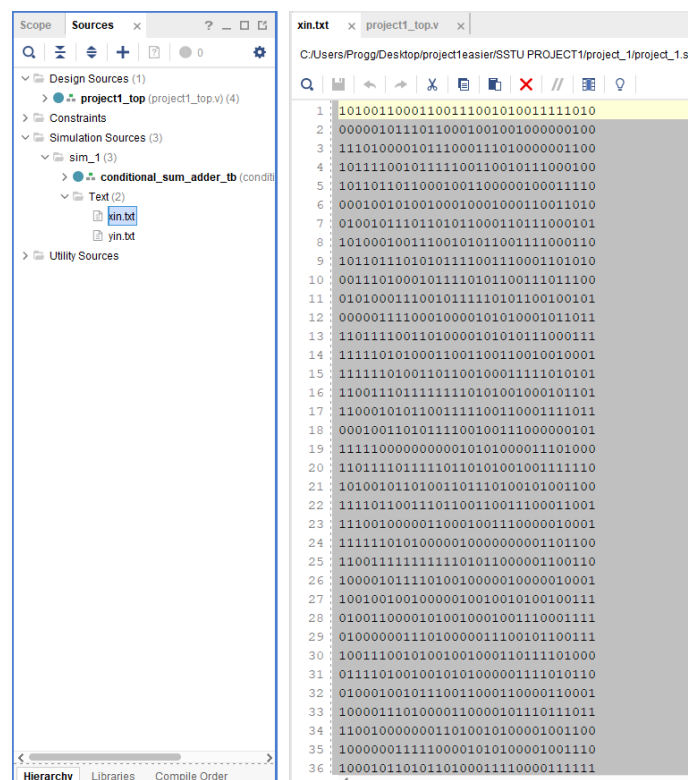
```
import random

def randombinary(b):
    reg = ""
    for i in range(b):
        temp = str(random.randint(0, 1))
        reg += temp
    return reg

file = open("output.txt", "w")

# Driver Code
n = 32
for i in range(100):
    randombinarynumber = randombinary(n)
    file.write(randombinarynumber + "\n")
    print(randombinarynumber)
```

After getting these numbers, made copy-paste to .txt files named xin,yin. Then added these files to simulation sources in Vivado.



I read these datas and assigned values to [31:0] x [0:99] and [31:0] y [0:99] signed registers.

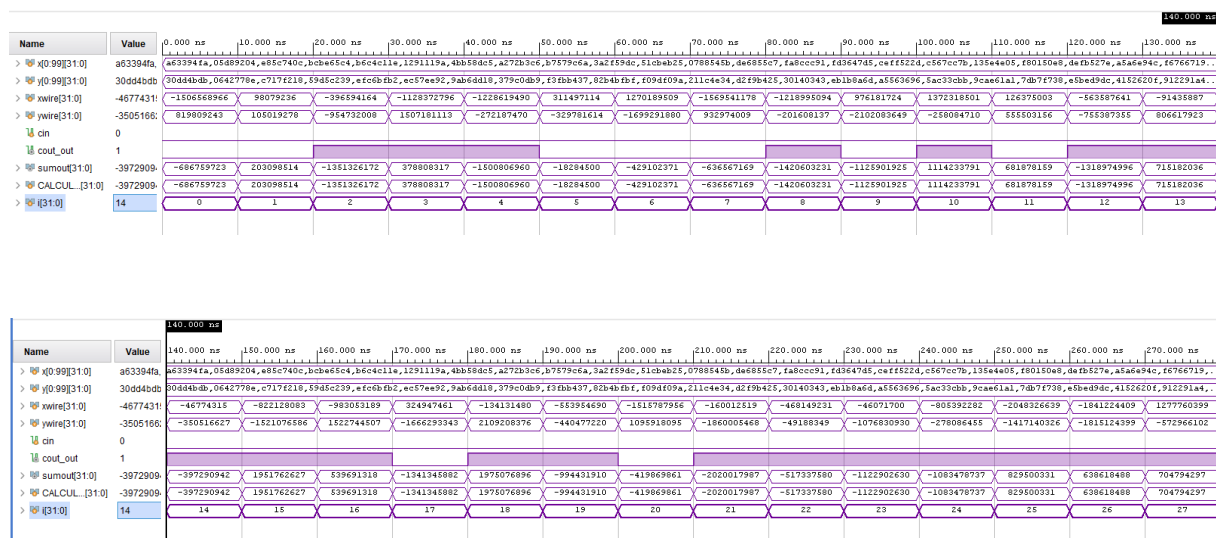
```
initial $readmemb("xin.txt",x);
initial $readmemb("yin.txt",y);
```

Just initial statements are left. I set a CALCULATOR register which gives directly $x+y+cin$ value in testbench. In every for loop, x and y values goes to my top module and calculator calculates the true result. Then an if block controls that is the output of module (sumout) equal to CALCULATOR value. If these values are equal, there will be a line in TCL console as:

x: *xvalue*, y: *yvalue*, result: *sumout*

```
for (i=0;i<=99;i=i+1)
begin
  xwire = x[i];
  ywire = y[i];
  CALCULATOR = x[i]+y[i]+cin;
  #5;
  if(CALCULATOR == sumout)
    $display ("x:%d , y: %d, result: %d", xwire,ywire,sumout);
  else
    $display("operation is wrong");
  #5;
end
end
```

Waveform outputs:



040170049 Muhammed Erkmen

| Name | Value | 280.000 ns | 290.000 ns | 300.000 ns | 310.000 ns | 320.000 ns | 330.000 ns | 340.000 ns | 350.000 ns | 360.000 ns | 370.000 ns | 380.000 ns | 390.000 ns | 400.000 ns | 410.000 ns |
|------------------|-----------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|
| > @0_99[31:0] | 336394da | | | | | | | | | | | | | | |
| > @0_99[31:0] | 306d44db | | | | | | | | | | | | | | |
| > @wire[31:0] | -10899608 | | | | | | | | | | | | | | |
| > @wire[31:0] | -3803837f | | | | | | | | | | | | | | |
| > cin | 0 | | | | | | | | | | | | | | |
| > cout | 1 | | | | | | | | | | | | | | |
| > sumout[31:0] | 10509224 | | | | | | | | | | | | | | |
| > CALCU...[31:0] | 10509224 | | | | | | | | | | | | | | |
| > @[31:0] | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 |

| | | 420,000 ms | | | | | | | | | | | | | | | |
|--------------------|-----------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|--|--|
| Name | Value | 420,000 ms | 430,000 ms | 440,000 ms | 450,000 ms | 460,000 ms | 470,000 ms | 480,000 ms | 490,000 ms | 500,000 ms | 510,000 ms | 520,000 ms | 530,000 ms | 540,000 ms | 550,000 ms | | |
| > 0 90931[3.0] | a63394da | | | | | | | | | | | | | | | | |
| > 0 90931[3.0] | 30540da8 | | | | | | | | | | | | | | | | |
| > ntwire[3.0] | -4377311 | | | | | | | | | | | | | | | | |
| > ywire[3.0] | 250801721 | | | | | | | | | | | | | | | | |
| ↳ con | 0 | | | | | | | | | | | | | | | | |
| ↳ cut_out | 0 | | | | | | | | | | | | | | | | |
| ↳ sumout[3.0] | -1888894 | | | | | | | | | | | | | | | | |
| ↳ CALCUL [3.0] | -1888894 | | | | | | | | | | | | | | | | |
| > [3.0] | 42 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | | |

[illegible][illegible]

| | | 840.000 ns | | | | | | | | | | | | | | | |
|--------------------|-----------|---|-------------|-------------|-------------|-------------|------------|-------------|-------------|-------------|------------|------------|------------|------------|-------------|--|--|
| Name | Value | 840.000 ns | 850.000 ns | 860.000 ns | 870.000 ns | 880.000 ns | 890.000 ns | 900.000 ns | 910.000 ns | 920.000 ns | 930.000 ns | 940.000 ns | 950.000 ns | 960.000 ns | 970.000 ns | | |
| > x[0:99][31:0] | a83394fa | a63394fa,05d89204,a8e740d0,bc4e65c4,bc4c411e,1291117a,4bb5856e,a27b3c67,b7579c6a,3a2f59ac,51cb4ab5,078854db,68d55c7f,1a8cc931,d364763e,ceff522a,c567c07b,135e4e05,600150e9,debf527e,af5a6e9a,f6769719,... | | | | | | | | | | | | | | | |
| > y[0:99][31:0] | 300d40bd | 00044eb,0642779e,f7171128,594dc139,afcc0b2d,c57ea92,3abed19b,379cd0d9,13b4b477,92b4bf8f,0094093a,211cc4e3,4df9b642,301d0343,abb1ba5a,a56e369e,8a393cb3,9caee111,74b7f738,eb5bd9dc,41526201,9122914a,... | | | | | | | | | | | | | | | |
| > xwre[31:0] | 76125782 | 76125782 | 1304567506 | -1836893933 | -1824291877 | 17454793764 | 480491974 | 695483409 | 1067893576 | -1195318132 | 682259773 | 2144831951 | 656714549 | 1284483389 | -1824072786 | | |
| > ywre[31:0] | 752d41459 | 752d41459 | -1393117509 | -810156867 | 1480284511 | 789380441 | 373997082 | 1703746120 | 2005626242 | 37489133 | -152740033 | -507961065 | 1221078900 | -562578979 | -207129920 | | |
| cin | 0 | | | | | | | | | | | | | | | | |
| cout_out | 0 | | | | | | | | | | | | | | | | |
| > sumout[31:0] | 15136723 | 1513672380 | 1111449997 | 1647916496 | -44007366 | -1760113091 | 854489056 | -1895737767 | -1221441478 | -1157828999 | 529519740 | 1636870886 | 1677793449 | 721904410 | -2031502714 | | |
| > calcul_out[31:0] | 15136723 | 1513672380 | 1111449997 | 1647916496 | -44007366 | -1760113091 | 854489056 | -1895737767 | -1221441478 | -1157828999 | 529519740 | 1636870886 | 1677793449 | 721904410 | -2031502714 | | |
| > 84 | 84 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | | |

| | | | | | | | | | | 1,000,000 ns |
|----------------------------|----------|--|--|--|--|--|--|--|--|--------------|
| Name | Value | | | | | | | | | |
| <code>%[0]9[31]0</code> | 963394fa | | | | | | | | | |
| <code>%[0]9[31]0</code> | 304da4db | | | | | | | | | |
| <code>%w[16]e</code> | 14378006 | | | | | | | | | |
| <code>%w[16]e[31:0]</code> | 18859577 | | | | | | | | | |
| <code>in</code> | 0 | | | | | | | | | |
| <code>cout</code> | 0 | | | | | | | | | |
| <code>sum</code> | 16263964 | | | | | | | | | |
| <code>sum[31:0]</code> | 16263964 | | | | | | | | | |
| <code>%[31:0]</code> | 100 | | | | | | | | | |

TCL Console output when simulation applied:

```

Tcl Console x Messages Log
Q 三 中 II 田 画 窗

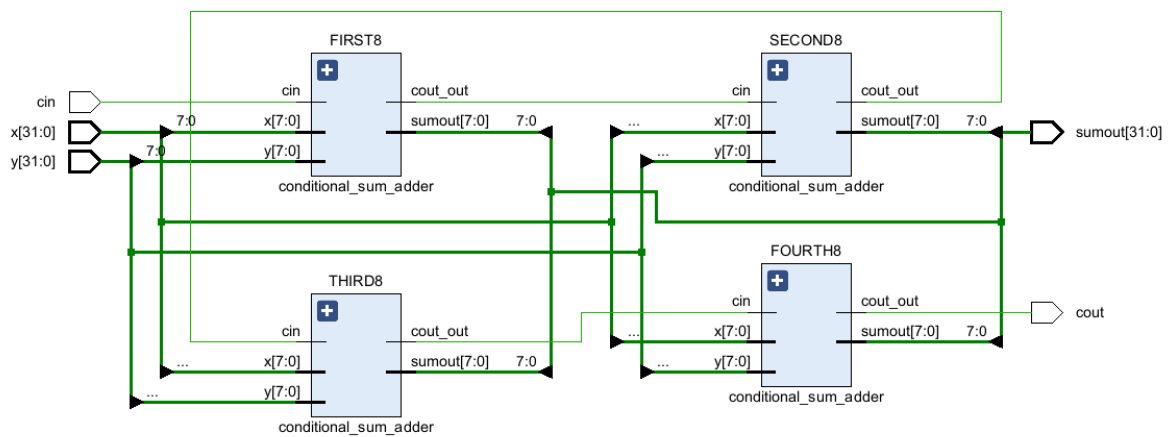
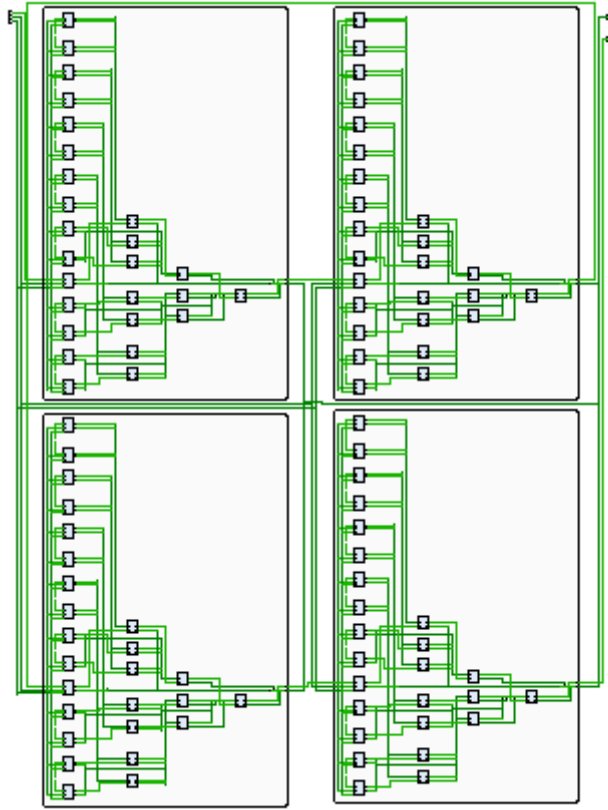
# }
# run 1000ns
x='bin=1010011000010001100010011111010 dec=2788398330','y='bin=001100001011101010010111101011 dec=819809243', result='bin=1010111000100001110000011010101 dec=-666759723', overflow=0 status=TRUE
x='bin=000001011101000100100000001000 dec= 98079236','y='bin=000001010000000101111010001110 dec= 105019278', result='bin=000010000010110000010110000101100010 dec= 203098514', overflow=0 status=TRUE
x='bin=111010000010110001101000000100 dec=3989373132','y='bin=100011000101001111101000000000 dec=3340235289', result='bin=10101110110100001010000100000100 dec=-1351326172', overflow=0 status=TRUE
x='bin=1110100101100110011001011000100 dec=3166594500','y='bin=01001101100110100000000110001 dec=1507181113', result='bin=0001011000101000000001111110101 dec= 378089317', overflow=0 status=TRUE
x='bin=1010100001010000010000010000110 dec=3066347806','y='bin=11011101100010101111101100010 dec=4022779826', result='bin=1001010100010111000000001010000 dec=-1500806960', overflow=0 status=TRUE
x='bin=0001001001000000000001000101010 dec= 311497114','y='bin=110110001001111011010000000001000 dec=396518562', result='bin=1111101101001000000000000101000 dec=-18284500', overflow=0 status=TRUE
x='bin=1001010101000101000101000101010 dec=1270189599','y='bin=100101010101010101010100000000 dec=2595675416', result='bin=1010100101010000101010101011101 dec=-429102373', overflow=0 status=TRUE
x='bin=10100000010000010111000110001010 dec=2725426118','y='bin=00110111001100000001011011001 dec= 932974009', result='bin=1011010000010110100000010111111 dec= -636567169', overflow=0 status=TRUE
x='bin=101010101001100110001010101010 dec=3075972202','y='bin=11100111011011010100000100111 dec=4093359159', result='bin=1010101010010101000001000001 dec=-1420603231', overflow=0 status=TRUE
x='bin=0011000001010101010101101100 dec= 976181724','y='bin=100000010101000101111011111 dec=212883647', result='bin=1011001100100000010011001011 dec=-1135901925', overflow=0 status=TRUE
x='bin=010000101001101101010010010101 dec=1372318501','y='bin=1110000101111000001001010 dec=4038682586', result='bin=01000001010010101101011011111 dec=-114233791', overflow=0 status=TRUE
x='bin=0000010100000000010100000101010 dec= 126375003','y='bin=00100001000010000100010100010100 dec= 555503156', result='bin=0010000100010001000010000101111 dec= 681878159', overflow=0 status=TRUE
x='bin=111001100010100000101011000111 dec=3731379555','y='bin=10010011110001010100000001001 dec= 3359579941', result='bin=101000101000100000000101110100 dec=-1318974996', overflow=0 status=TRUE
x='bin=1111101000010010010010010001000 dec=4203531409','y='bin=00110000000101000000000101000011 dec= 806617923', result='bin=001001010100000010011101010100 dec= 715182036', overflow=0 status=TRUE
x='bin=111101010101000100011101010101 dec=4248192981','y='bin=101010100001011000010001010101 dec=394450069', result='bin=1011000001000010110001000100000 dec=-397290942', overflow=0 status=TRUE
x='bin=100101111101010001000100010101 dec=347239213','y='bin=1001010101010100010101000101010 dec=2773890710', result='bin=00110000010010101000010001100001 dec= 1951764627', overflow=0 status=TRUE
x='bin=100010010101010101010001010101 dec=3511914107','y='bin=0010101010000001001110001010101 dec=1527454077', result='bin=0001000000000101000001000101010 dec= 339691318', overflow=0 status=TRUE
x='bin=000101111001010101010100000101 dec= 324947461','y='bin=100110010101010100010000010100001 dec=4626873953', result='bin=1011000000000100101111010010 dec= 1341345893', overflow=0 status=TRUE
x='bin=11100000000000010000011000001000 dec=4160385916','y='bin=0011110101010111101101001000 dec=2019206793', result='bin=0010101010101001010000001000000 dec=-1972076956', overflow=0 status=TRUE
x='bin=100110110101010101010001010100 dec=3740102606','y='bin=11001010101010100101010100 dec=3854490076', result='bin=1000010000101010000101000101010 dec= -994431910', overflow=0 status=TRUE
x='bin=100101010101000101010001000100 dec=2775179340','y='bin=100000010101000101000100000001111 dec=1095918095', result='bin=10010101111001010001010101011 dec= -419698961', overflow=0 status=TRUE
x='bin=111010010101000101010001000101 dec=4134954777','y='bin=1000010010000100010001000100100 dec=2434961828', result='bin=1000011000100010011100001011011 dec=-2020017987', overflow=0 status=TRUE
x='bin=1110010000010000010101000000010 dec=3826818065','y='bin=1111010000100001011000000000011 dec=4275778947', result='bin=1100000100101000000110000001000 dec=-517337580', overflow=0 status=TRUE
x='bin=1110100000000000000000000001010 dec=4248989556','y='bin=101111000000010101000001010 dec=3218136366', result='bin=10111000000000010101010101010 dec=-1123902630', overflow=0 status=TRUE
x='bin=1001100111110101000001000101010 dec=3489575014','y='bin=101101010101000101010001001 dec= 4016808041', result='bin=10111010101010101010100011111 dec=-803478737', overflow=0 status=TRUE
x='bin=1000001011010010000001000000000 dec=2246940657','y='bin=100101010100010000000101101010 dec=2877826970', result='bin=0011000001010000100101010101011 dec= 1289500331', overflow=0 status=TRUE
x='bin=1001000000000001000101000101011 dec=2453742897','y='bin=1001001100110101010001000001 dec=2479842397', result='bin=000100000100000100000100010111000 dec= 638614489', overflow=0 status=TRUE
x='bin=1001010000010000000001010101011 dec=1277763999','y='bin=101101010101000101010101010 dec=3720011594', result='bin=000101000000000100010101010101 dec= 704794297', overflow=0 status=TRUE
x='bin=100000000101000000100010101011 dec=1089490971','y='bin=1101010101010101000000000001010 dec=4254928921', result='bin=0011010001010001010110000000000 dec= 1056224549', overflow=0 status=TRUE
x='bin=100100100001000101010101010000 dec=2628029929','y='bin=0100001011100010101011001110 dec=1660535566', result='bin=1111101001100010001001000101010 dec= -6403801', overflow=0 status=TRUE
x='bin=0111001001001000000101000001010 dec=0501703766','y='bin=1001000101000001011100010000 dec=3697045288', result='bin=01010100010010101110001011110 dec= 1453781759', overflow=0 status=TRUE
x='bin=0000010010010100010000010000010 dec=1153010737','y='bin=10010101010001000001000101001 dec=2640778835', result='bin=1100010000000000010000010000010 dec=-501177724', overflow=0 status=TRUE
x='bin=10000101000000010000010001010101 dec=2269318075','y='bin=1110000101010100010101000101000 dec=4067376024', result='bin=00110010010100100100000101010101 dec= 2047326803', overflow=0 status=TRUE
x='bin=1001000000001000010000000000010 dec=3356305484','y='bin=10010000001000010101010001010 dec=3428925624', result='bin=100100000011101000000101010101 dec=-1740733475', overflow=0 status=TRUE
x='bin=1000000001110000010100000000010 dec=2180032590','y='bin=11000001010000000001000000000 dec=3807413280', result='bin=00100101010000010001000101010 dec= 1692478574', overflow=0 status=TRUE
x='bin=100001010010000111000001110000111 dec=2337946697','y='bin=00101010000100001010011011011 dec=1515300711', result='bin=1001010101010101110000101001010 dec=-441719989', overflow=0 status=TRUE
x='bin=10001010010100010101010100000 dec=2268692064','y='bin=10101000000000000001010000011 dec=3892381507', result='bin=00101110010111001010000010100001 dec= 1866104278', overflow=0 status=TRUE
x='bin=000000010010010101010101010101 dec=120919245','y='bin=00101010010010010010101101010 dec=2057645595', result='bin=1011010101010000010001010100001 dec=-116402493', overflow=0 status=TRUE
x='bin=10101111000100010000010001010101 dec=1754598977','y='bin=10010101010101000000000001010 dec=4291272130', result='bin=0011010000010000010000000001011 dec= 1809755487', overflow=0 status=TRUE
x='bin=000000101111000001001010100000 dec= 117204336','y='bin=100101110001010101010101010 dec=2945431454', result='bin=101010100001000001000000010000010 dec=-1232331506', overflow=0 status=TRUE
x='bin=1110000001000001010101010010101 dec=4028820651','y='bin=00100001010100010100000101100 dec= 565801276', result='bin=00010000101010000101010101011 dec= -59456431', overflow=0 status=TRUE
x='bin=000010010000010000010001010101 dec= 342859515','y='bin=00001001000001011110000000010 dec= 4491827932', result='bin=00010010100101000101000000010 dec= 792042247', overflow=0 status=TRUE
x='bin=1100101010100001000000000000000 dec=3957326112','y='bin=000010101100010101000001000 dec= 250861721', result='bin=00010001010100010001010001010 dec=-186698463', overflow=0 status=TRUE
x='bin=1001000001000000100000010000010 dec=3502392133','y='bin=0001010101010100000101010001 dec=1003949425', result='bin=000010000100010010010100000101010 dec= 21374262', overflow=0 status=TRUE

x='bin=10010000010000010000010000010001 dec=3502392133','y='bin=000101010101010000010001010001 dec=1003949425', result='bin=0000100001000100100100000101010 dec= 21374262', overflow=0 status=TRUE
x='bin=00000100000100000100000100000001 dec= 7426115','y='bin=0110101000010000010001000000010101 dec=1097640765', result='bin=00101010100000010101010101000000 dec= 2070060680', overflow=0 status=TRUE
x='bin=10010101010101010001000001000101 dec=3430462499','y='bin=1000000101000000000100000100100 dec=3250610516', result='bin=00010010001011010101010101011 dec=1295869719', overflow=0 status=TRUE
x='bin=1001010001000000010101010101010 dec=3058419394','y='bin=1001000100000100010101011000000 dec=3429178304', result='bin=1000000100100000010001010101011 dec=-210234354', overflow=0 status=TRUE
x='bin=0010000100001000010000000000010 dec=1372480424','y='bin=01000000000001010101010000001 dec=1348526217', result='bin=100000010100000100010101010001 dec=-177606655', overflow=0 status=TRUE
x='bin=1001010000000100010101010101010 dec=2894452397','y='bin=00000100000001101010100010010 dec= 383393922', result='bin=10100101000100000100010111111 dec=18262174977', overflow=0 status=TRUE
x='bin=0010010001000001010000010001010 dec=1771139282','y='bin=10001001000101000001101010101010 dec=383384762', result='bin=0010000101000001010101000101000 dec= 860056748', overflow=0 status=TRUE
x='bin=001001000000010000010001010101 dec=2038929589','y='bin=1000010001000100100101010101110 dec=2392145752', result='bin=000000000111000000010111010101011 dec= 65109875', overflow=0 status=TRUE
x='bin=0010010000010000010000010000000 dec=1318204256','y='bin=0010010001000000000100000001010 dec= 702557208', result='bin=0011000001000000010101010101010 dec= 2020761462', overflow=0 status=TRUE
x='bin=1000000000010000000000000000000 dec=2153005232','y='bin=0010000001011110000000000000000 dec=1525627470', result='bin=10000000000000000001010101010 dec= -489421361', overflow=0 status=TRUE
x='bin=1110101010101000000100000000010 dec=429867564','y='bin=0011010101010101010000000001010 dec=2101272130', result='bin=0011000000010000010000000001010 dec= 2016162389', overflow=0 status=TRUE
x='bin=0010110100000101110100000000010 dec=1867245630','y='bin=00101010010001000100010001000 dec= 995732274', result='bin=1010010101000101000001000100000 dec=-1431959592', overflow=0 status=TRUE
x='bin=0000000100000100000000000000000 dec=1101150288','y='bin=10101010101010000000000000010 dec= 3606906097', result='bin=0000010000010111000101010000000 dec= 4013090985', overflow=0 status=TRUE
x='bin=10100100000000010001000001000101 dec=3509209098','y='bin=1101010100000000000000000100101 dec=3934798651', result='bin=10000101000001010101010100011 dec= 7450349593', overflow=0 status=TRUE
x='bin=10000110010100010001000101011110 dec=1009311486','y='bin=10001010000100000100010000010101 dec=2857599069', result='bin=000100101000000001000000010101 dec= -64043259', overflow=0 status=TRUE
x='bin=00000001001000000101010101010101 dec= 40090045','y='bin=00101100000101101101010001010 dec=1880170534', result='bin=0100000001000001101010101011011 dec= 1620179579', overflow=0 status=TRUE
x='bin=00101011000000000001000101000110 dec=1874972046','y='bin=0010010001000000000000000000011 dec=1955064495', result='bin=10000000000000000101010101011 dec=-464390755', overflow=0 status=TRUE
x='bin=10010001000000010100010100010101 dec=2773766865','y='bin=001100000100000100000100010001010 dec=1011934814', result='bin=1100000101000100010001000101010 dec=-509265617', overflow=0 status=TRUE
x='bin=100100000101010001000000000101010 dec=343831882','y='bin=111000010000000100000100000101000 dec=4171293666', result='bin=1001001010101010101100010010010 dec= -874709044', overflow=0 status=TRUE
x='bin=0000000100010100010001010000010 dec=103975842','y='bin=100010000111010101010101010 dec=3389914546', result='bin=0000010101000000000000000000000 dec= 197923092', overflow=0 status=TRUE
x='bin=11101000000000000001000101010101 dec=4129462993','y='bin=0010101001010000000100000101010 dec= 860903686', result='bin=0010000000000101000001010000011 dec= 315895283', overflow=0 status=TRUE
x='bin=0000000001000001000000000000000 dec=259250959','y='bin=0011010000000000000000000000000 dec=1050957455', result='bin=1010101011111000011000000000010 dec= -704759761', overflow=0 status=TRUE
x='bin=1111000000000001000101010001010 dec=4265522470','y='bin=001111000001011110000100000001000 dec=1043325460', result='bin=00110000010000010000010101010 dec=1014206364', overflow=0 status=TRUE
x='bin=00000001010100010001000100010001 dec= 426413657','y='bin=1000100100000100000100000100010 dec=2509131050', result='bin=1010101101110000010101000000010 dec=-1359422589', overflow=0 status=TRUE
x='bin=0000000001010001000100010001000 dec= 30762324','y='bin=0001010001000100010001000100000 dec=1045590728', result='bin=000000000000000101101000000001000 dec= 1076353052', overflow=0 status=TRUE
x='bin=00100101010100010001000101000101 dec=1732070989','y='bin=0001000100010001000101000101000 dec= 610087202', result='bin=1000010101100010100010001010111 dec=-1947809105', overflow=0 status=TRUE
x='bin=00000100000100010001000100010001 dec= 172975410','y='bin=1101010000100100010001000100001 dec=4119704545', result='bin=1111101010000100010000000001011 dec= -2287341', overflow=0 status=TRUE
x='bin=00010010000000000100010001000100 dec=2584169320','y='bin=0010010001000001000001000000010 dec=1732815426', result='bin=0000000000010001111101010001000 dec= 22017450', overflow=0 status=TRUE
x='bin=10010001011011010101010001000100 dec=2814311196','y='bin=1000010001010001000000000000001 dec=2346238361', result='bin=000100010001010101010100010101 dec= 865582261', overflow=0 status=TRUE
x='bin=00000000010100010000000000000101 dec= 64434221','y='bin=100100000110010000000100010010111 dec=341979827', result='bin=1001000001000000000000000100000 dec=-173938316', overflow=0 status=TRUE
x='bin=1110000010001001000100000000010 dec=4039877902','y='bin=000100000100000000000001000100011 dec= 341979827', result='bin=000000010000010001000101010100000 dec= 86904933', overflow=0 status=TRUE
x='bin=10000100010001000001000001010100 dec=2394277749','y='bin=0001001001010100090000000000000 dec= 762717820', result='bin=101011000001000001010101010100001 dec=-1137917728', overflow=0 status=TRUE
x='bin=1000000001010001000100010111111 dec=3260834687','y='bin=000100100100000000000000000001011 dec= 763883637', result='bin=1101011110000010000001000101010 dec= -270417442', overflow=0 status=TRUE
x='bin=00000000010001010110000000000101 dec= 47119540','y='bin=0001010000000000000000000000000 dec= 513646010', result='bin=00000000000101010001000101010 dec= 360815550', overflow=0 status=TRUE
x='bin=00100100010001000100000000010001 dec=406762537','y='bin=1000000001011000000101010101000 dec=2197596040', result='bin=00010001000100010101010100000 dec=-690567719', overflow=0 status=TRUE
x='bin=10000001000100010001000100010001 dec=2208898146','y='bin=1000000000000000010001000100000 dec=3238311121', result='bin=0000010001000101010101000000000 dec= 1152331971', overflow=0 status=TRUE
x='bin=00000001010100010001000100000101 dec=1198161039','y='bin=000100000000010100010001000101011 dec= 183540511', result='bin=000100011100010001010101000000010 dec= 1743206150', overflow=0 status=TRUE
x='bin=00010001000100010001000100010101 dec= 562443511','y='bin=00100000010000100000010000010101010 dec=1849046494', result='bin=1010100100010000010001
```

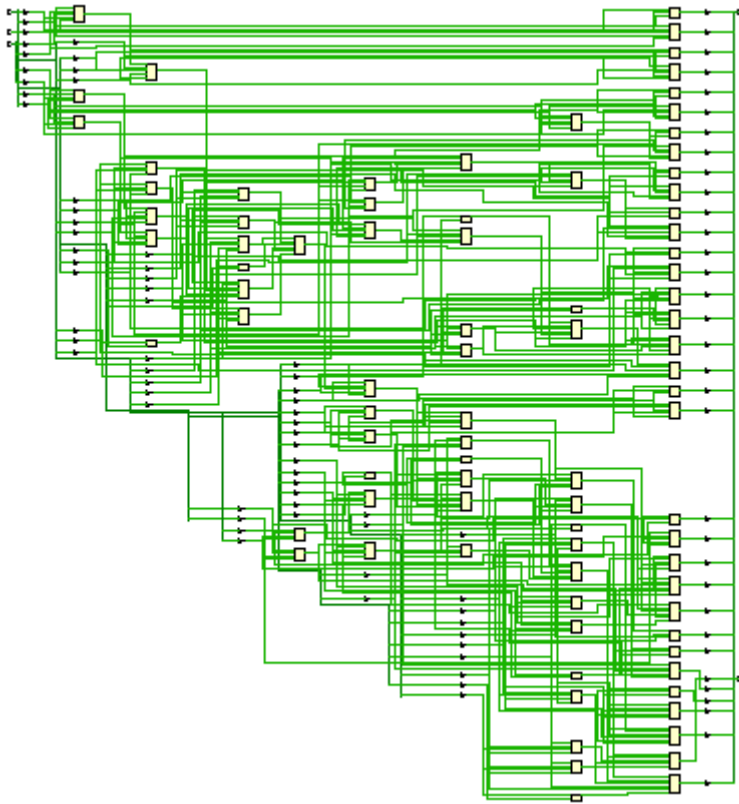
Implementation Reports:

So i put top of my module (* DONT_TOUCH = "TRUE" *) statement then i started implementation

RTL Schematic:

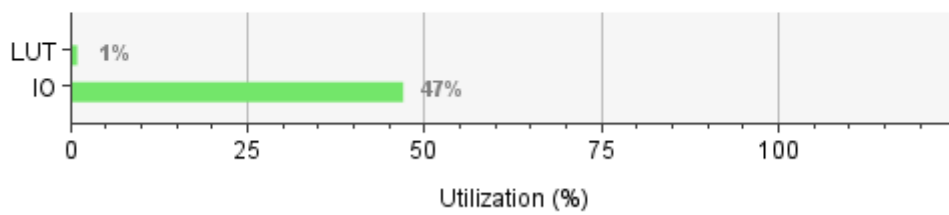


Technology Schematic:



Utilization Report:

| Resource | Utilization | Available | Utilization % |
|----------|-------------|-----------|---------------|
| LUT | 59 | 32600 | 0.18 |
| IO | 98 | 210 | 46.67 |



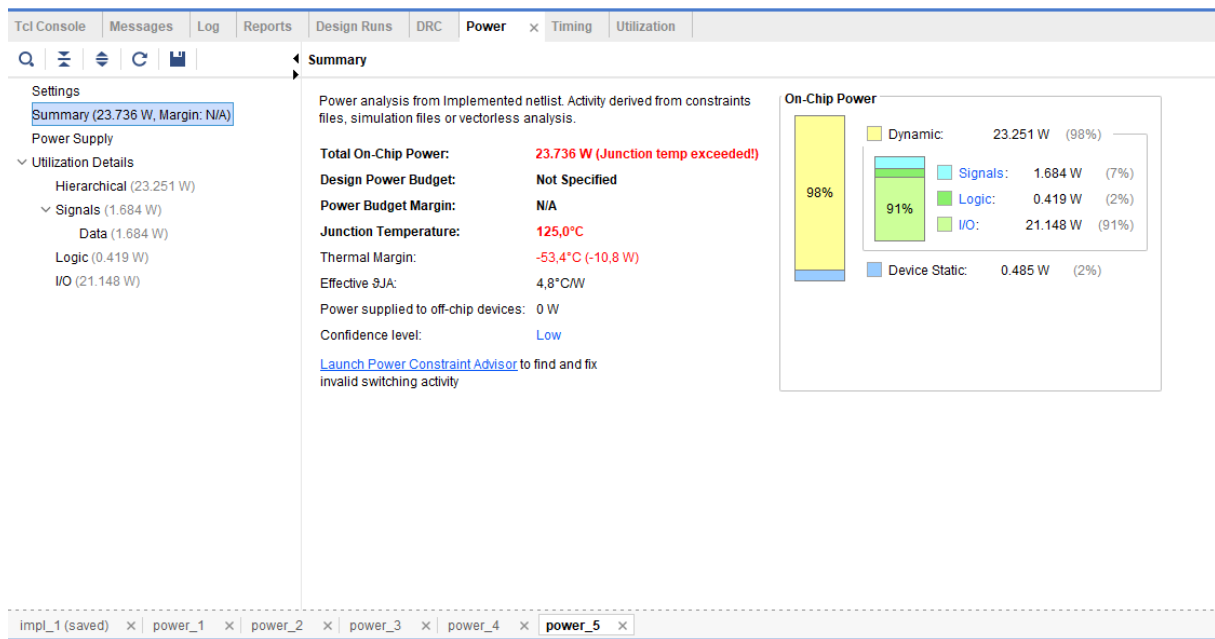
59 LUT used in this design

Timing Report:

| Combinational Delays | | | | | |
|----------------------|------------|--------------------|-----------|--------------------|------|
| From Port | To Port | Max Process Corner | Min Delay | Min Process Corner | |
| x[1] | sumout[28] | 18.519 | SLOW | 6.894 | FAST |
| x[0] | sumout[28] | 18.366 | SLOW | 6.857 | FAST |
| x[1] | sumout[29] | 18.270 | SLOW | 6.793 | FAST |
| x[5] | sumout[28] | 18.175 | SLOW | 6.559 | FAST |
| x[0] | sumout[29] | 18.117 | SLOW | 6.756 | FAST |
| x[1] | sumout[31] | 18.001 | SLOW | 6.719 | FAST |
| x[4] | sumout[28] | 17.950 | SLOW | 6.474 | FAST |
| x[5] | sumout[29] | 17.926 | SLOW | 6.458 | FAST |
| x[1] | sumout[30] | 17.895 | SLOW | 6.675 | FAST |
| x[7] | sumout[28] | 17.888 | SLOW | 6.328 | FAST |
| x[0] | sumout[31] | 17.848 | SLOW | 6.682 | FAST |
| x[0] | sumout[30] | 17.743 | SLOW | 6.637 | FAST |
| x[4] | sumout[29] | 17.701 | SLOW | 6.373 | FAST |
| x[5] | sumout[31] | 17.657 | SLOW | 6.384 | FAST |
| x[12] | sumout[28] | 17.655 | SLOW | 6.229 | FAST |
| x[7] | sumout[29] | 17.639 | SLOW | 6.227 | FAST |
| x[5] | sumout[30] | 17.552 | SLOW | 6.340 | FAST |
| cin | sumout[28] | 17.524 | SLOW | 6.523 | FAST |
| x[4] | sumout[31] | 17.432 | SLOW | 6.299 | FAST |
| x[1] | sumout[27] | 17.418 | SLOW | 6.484 | FAST |

So maximum delay is 18.519 nanosecond. That means our period must be bigger than 18.519 nS. Lets assume it 18.52 nS, maximum clock frequency is $1/((18.52)*(10e-9))$ almost 54 MHz.

Power Report:



When we use don't touch, power consumption is too much.

| Utilization | Name | I/O Type | I/O Standard | Drive Strength | Input Pins | Output Pins | Bidir Pins | IO LOGIC SERDES | IO DELAY | IBUF LOW PWR | Input Term |
|-------------------------|--------------|----------|--------------|----------------|------------|-------------|------------|-----------------|----------|--------------|------------|
| 21.148 W (89% of total) | project1_top | | | | | | | | | | |
| 20.403 W (86% of total) | sumout | HR | LVC MOS18 | 12.000 | 0 | 32 | 0 | No | Off | No | NONE |
| 0.485 W (2% of total) | cout | HR | LVC MOS18 | 12.000 | 0 | 1 | 0 | No | Off | No | NONE |
| 0.128 W (1% of total) | x | HR | LVC MOS18 | N/A | 32 | 0 | 0 | No | Off | No | RTT_NONE |
| 0.128 W (1% of total) | y | HR | LVC MOS18 | N/A | 32 | 0 | 0 | No | Off | No | RTT_NONE |
| 0.004 W (<1% of total) | cin | HR | LVC MOS18 | N/A | 1 | 0 | 0 | No | Off | No | RTT_NONE |

Average power consumption for Leaf Cells is $23.251/185 = 0.125$ watt

Average power consumption for LUTs is $23.251/59 = 0.394$ watt

| Utilization | Name | Signals (W) | Data (W) | Logic (W) | I/O (W) |
|-------------------------|------------------|-------------|----------|-----------|---------|
| 23.251 W (98% of total) | project1_top | | | | |
| 23.251 W (98% of total) | Leaf Cells (185) | | | | |

Source Codes:

Full Adder:

```
// -----
module HA (
input x,
input y,
output cout,
output sum
);
assign sum = ((~x)&&y) || (x&&(~y));
assign cout = x&&y;
endmodule
//-----

module FA (
input x,
input y,
input cin,
output cout,
output sum);
wire sum1;
wire cout1;
wire sum2;
wire cout2;
HA HA1 (.x(x),.y(y),.sum(sum1),.cout(cout1));
HA HA2 (.x(sum1),.y(cin),.sum(sum2),.cout(cout2));
assign sum = sum2;
assign cout = cout2 || cout1;
endmodule
```

Level1 Mux:

```
module level1MUX(
input [3:0] D,
input S,
output reg selected_sum,
output reg selected_cout
);

always @(S,D) begin
case(S)
1'b0 :
begin
selected_sum<= D[0];
selected_cout<= D[1];
end
1'b1:
begin
selected_sum <= D[2];
selected_cout <= D[3];
end
endcase
end
endmodule
```

Level2 Mux:

```
// -----
module level2MUX(
input [5:0] D,
input S,
output reg [1:0] selectedsum,
output reg selectedcout
);
always @(S,D) begin
case(S)
1'b1 :
begin
selectedsum<= D[1:0];
selectedcout<= D[2];
end
1'b0:
begin
selectedsum <= D[4:3];
selectedcout<= D[5];
end
endcase
end
endmodule
```

Level3 Mux:

```
`timescale 1ns / 1ps
module level3MUX(
input [9:0] D,
input S,
output reg [3:0] selectedsum,
output reg selectedcout
);
always @(S,D) begin
case(S)
1'b1 :
begin
selectedsum<= D[3:0];
selectedcout<= D[4];
end
1'b0:
begin
selectedsum <= D[8:5];
selectedcout<= D[9];
end
endcase
end
endmodule
```

8-bit Conditional Sum Adder:

```

module conditional_sum_adder(
    input [7:0] x,
    input [7:0] y,
    input cin,
    output [7:0] sumout,
    output cout_out
);
wire s0,s1;
genvar i;
wire firstselect;
wire [1:0] muxselects_1 [2:0];
wire [1:0] mastersums_1 [2:0];
wire [3:0] muxinputs_1 [2:0];
wire [1:0] cselection_1 [7:1];
wire [1:0] sumwilladd_1 [7:1];
wire [1:0] cout [7:1];
wire [1:0] sum [7:1];
// LEVEL 1

FA fa0 (.x(x[0]),.y(y[0]),.cin(cin),.cout(c1),.sum(s0));
FA fa1_0 (.x(x[1]),.y(y[1]),.cin(0),.cout(cout[1][0]),.sum(sum[1][0]));
FA fa1_1 (.x(x[1]),.y(y[1]),.cin(1),.cout(cout[1][1]),.sum(sum[1][1]));
FA fa2_0
(.x(x[2]),.y(y[2]),.cin(0),.cout(cselection_1[2][0]),.sum(sumwilladd_1[2][0]
));
FA fa2_1
(.x(x[2]),.y(y[2]),.cin(1),.cout(cselection_1[2][1]),.sum(sumwilladd_1[2][1]
));
FA fa3_0 (.x(x[3]),.y(y[3]),.cin(0),.cout(cout[3][0]),.sum(sum[3][0]));
FA fa3_1 (.x(x[3]),.y(y[3]),.cin(1),.cout(cout[3][1]),.sum(sum[3][1]));
FA fa4_0
(.x(x[4]),.y(y[4]),.cin(0),.cout(cselection_1[4][0]),.sum(sumwilladd_1[4][0]
));
FA fa4_1
(.x(x[4]),.y(y[4]),.cin(1),.cout(cselection_1[4][1]),.sum(sumwilladd_1[4][1]
));
FA fa5_0 (.x(x[5]),.y(y[5]),.cin(0),.cout(cout[5][0]),.sum(sum[5][0]));
FA fa5_1 (.x(x[5]),.y(y[5]),.cin(1),.cout(cout[5][1]),.sum(sum[5][1]));
FA fa6_0
(.x(x[6]),.y(y[6]),.cin(0),.cout(cselection_1[6][0]),.sum(sumwilladd_1[6][0]
));
FA fa6_1
(.x(x[6]),.y(y[6]),.cin(1),.cout(cselection_1[6][1]),.sum(sumwilladd_1[6][1]
));
FA fa7_0 (.x(x[7]),.y(y[7]),.cin(0),.cout(cout[7][0]),.sum(sum[7][0]));
FA fa7_1 (.x(x[7]),.y(y[7]),.cin(1),.cout(cout[7][1]),.sum(sum[7][1]));

wire [3:0] mux0_in = {cout[1][1],sum[1][1],cout[1][0],sum[1][0]};
wire [3:0] mux1_in = {cout[3][1],sum[3][1],cout[3][0],sum[3][0]};
wire [3:0] mux2_in = {cout[3][1],sum[3][1],cout[3][0],sum[3][0]};
wire [3:0] mux3_in = {cout[5][1],sum[5][1],cout[5][0],sum[5][0]};
wire [3:0] mux4_in = {cout[5][1],sum[5][1],cout[5][0],sum[5][0]};
wire [3:0] mux5_in = {cout[7][1],sum[7][1],cout[7][0],sum[7][0]};
wire [3:0] mux6_in = {cout[7][1],sum[7][1],cout[7][0],sum[7][0]};

wire [1:0] selsum [7:1];
wire [1:0] selcout [7:1];
level1MUX M0 (.D(mux0_in),.S(c1),.selected_sum(s1),.selected_cout(c2));

```

```

level1MUX M1
(.D(mux1_in),.S(cselection_1[2][0]),.selected_sum(selsum[3][0]),.selected_cout(selcout[3][0]));
level1MUX M2
(.D(mux2_in),.S(cselection_1[2][1]),.selected_sum(selsum[3][1]),.selected_cout(selcout[3][1]));
level1MUX M3
(.D(mux3_in),.S(cselection_1[4][0]),.selected_sum(selsum[5][0]),.selected_cout(selcout[5][0]));
level1MUX M4
(.D(mux4_in),.S(cselection_1[4][1]),.selected_sum(selsum[5][1]),.selected_cout(selcout[5][1]));
level1MUX M5
(.D(mux5_in),.S(cselection_1[6][0]),.selected_sum(selsum[7][0]),.selected_cout(selcout[7][0]));
level1MUX M6
(.D(mux6_in),.S(cselection_1[6][1]),.selected_sum(selsum[7][1]),.selected_cout(selcout[7][1]));

wire [5:0] mux0_2_in =
{selcout[3][0],selsum[3][0],sumwilladd_1[2][0],selcout[3][1],selsum[3][1],sumwilladd_1[2][1]};
wire [5:0] mux1_2_in =
{selcout[7][0],selsum[7][0],sumwilladd_1[6][0],selcout[7][1],selsum[7][1],sumwilladd_1[6][1]};
wire [5:0] mux2_2_in =
{selcout[7][0],selsum[7][0],sumwilladd_1[6][0],selcout[7][1],selsum[7][1],sumwilladd_1[6][1]};

wire [1:0] s2;
wire c3;
wire [1:0] mux2sumoutput0,mux2sumoutput1;
wire mux2cout0,mux2cout1;
level2MUX M20 (.D(mux0_2_in),.S(c2),.selectedsum(s2),.selectedcout(c3));
level2MUX M21_0
(.D(mux1_2_in),.S(selcout[5][0]),.selectedsum(mux2sumoutput0),.selectedcout(mux2cout0));
level2MUX M21_1
(.D(mux2_2_in),.S(selcout[5][1]),.selectedsum(mux2sumoutput1),.selectedcout(mux2cout1));

wire [9:0] mux3inputs =
{mux2cout0,mux2sumoutput0,selsum[5][0],sumwilladd_1[4][0],mux2cout1,mux2sumoutput1,selsum[5][1],sumwilladd_1[4][1]};

wire clast;
wire [3:0] lastsum;
level3MUX LASTMUX
(.D(mux3inputs),.S(c3),.selectedsum(lastsum),.selectedcout(cout_out));

assign sumout = {lastsum,s2,s1,s0};
endmodule

```


32-bit Conditional Sum Adder TOP Module:

```

`timescale 1ns / 1ps

(* DONT_TOUCH = "TRUE" *)
module project1_top(
    input signed [31:0] x,
    input signed [31:0] y,
    input cin,
    output [31:0] sumout,
    output cout,
    output reg v
);

wire [7:0] sum1,sum2,sum3,sum4;
conditional_sum_adder FIRST8
(.x(x[7:0]),.y(y[7:0]),.cin(cin),.sumout(sum1),.cout_out(co1));
conditional_sum_adder SECONDD8
(.x(x[15:8]),.y(y[15:8]),.cin(co1),.sumout(sum2),.cout_out(co2));
conditional_sum_adder THIRDD8
(.x(x[23:16]),.y(y[23:16]),.cin(co2),.sumout(sum3),.cout_out(co3));
conditional_sum_adder FOURTH8
(.x(x[31:24]),.y(y[31:24]),.cin(co3),.sumout(sum4),.cout_out(cout));

assign sumout = {sum4,sum3,sum2,sum1};

always @(*) begin
if( (x+y > 2147483646) || (x+y<-2147483647))
v<=1;
else
v<=0;
end

endmodule

```

Work package of this report:

| | |
|---------------|-----------------|
| Research | Muhammed Erkmen |
| Understanding | Muhammed Erkmen |
| Design | Muhammed Erkmen |
| Code | Muhammed Erkmen |
| Test | Muhammed Erkmen |
| Report | Muhammed Erkmen |