SAN JOSÉ STATE UNIVERSITY

CharlesW.DavidsonCollege of Engineering

DEPARTMENT OF ELECTRICAL ENGINEERING

**EE271 – Advanced Digital System Design and Synthesis**

Fall 2017 Final Project Report

**Topic: Implementation and Analyzing 64-bit Conditional Sum and Carry Select Adders**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Deekshith Krishnegowda | SJSU ID | 012417080 |
| Email | deekshithkrishnegowda@yahoo.com | Phone | (669)281-9059 |

**Executive Summary**

The purpose of this project is to implement, optimize and analyze the performance of two different kind of 64-bit adders(which can perform both addition and subtraction): 1. Carry Select Adder 2. Conditional Sum Adder. The implementation is done with 8-bit groups and at RTL level. Analysis is performed for three metrics: 1. Timing performance (RTL and netlist simulation) 2. Circuit area 3. Power consumption.

## I. GeneralProject Information

## Table I.1: List of EDA Tools Used

|  |  |  |
| --- | --- | --- |
| **EDA Tool Name** | **Company** | **You Used it for** |
| VCS | Synopsys | Compilation & simulation |
| VCS | Synopsys | Post-simulation |
| Design Vision | Synopsys | Synthesis & optimization |

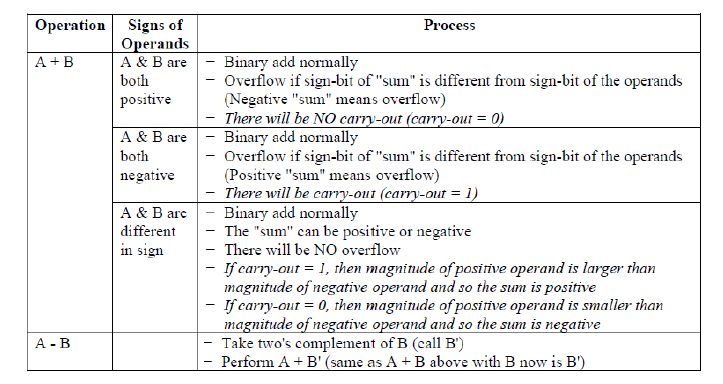
## Table I.2: List of Libraries Used

|  |  |  |
| --- | --- | --- |
| **Library file name** | **Used with**  (EDA tool name) | **The library is at**  (directories on eecad systems) |
| tc240c | Synopsys design compiler | /apps/Toshiba/sjsu/synopsys/tc240c/ |

## Table I.3: List of Verilog Modules (both design and test modules)

|  |  |  |
| --- | --- | --- |
| **Module Name** | **Ports** | **Short Description** |
| Top\_CSA | ope1[63:0],  ope2[63:0],  add\_sub,  clock,  reset,  i,j,  sum[63:0],  cout,  overf,  complete,  start | This is the topmodule for the carry select adder.  'i' and' j' is tied to logic 0 and 1 always.  Circuit is initially idle and starts at posedge of start signal. Circuit restarts whenever start changes from 0 to 1.  Complete signal indicates valid sum along with overf .  Complete is reset at posedge of clock signal. |
| Ripple\_carry\_adder | a[7:0],  b[7:0],  add\_sub,  clock,  start,  reset,  sum[7:0],  cout,  complete | This is 8-bit ripple carry adder where 'Cin' of first full adder is tied to 'add\_sum'. |
| Ripple\_carry\_adder\_zero | a[7:0],  b[7:0],  add\_sub,  clock,  start,  reset,  d, sum[7:0],  cout,  complete | This is 8-bit ripple carry adder where 'Cin' of first full adder is tied to 'd' . Logic zero is passed to d from testbench. |
| Ripple\_carry\_adder\_one | a[7:0],  b[7:0],  add\_sub,  clock,  start,  reset,d,  sum[7:0], cout,  complete | This is 8-bit ripple carry adder where 'Cin' of first full adder is tied to 'd'. Logic one is passed to d from testbench. |
| Mux\_8 | [7:0]in0\_8, [7:0]in1\_8,  sel\_8,  [7:0]out\_8 | 2:1 mux with 8-bit input and output. |
| Mux\_1 | in0\_1,  in1\_1,  sel\_1,  out\_1 | 2:1 mux with 1-bit input and output. |
| Fulladder | A,  B,  Cin,  sum,  carry | Normal full adder circuit. |
| Top\_CSA\_tb | ope1[63:0],  ope2[63:0],  add\_sub,  clock,  reset,i,j,  sum[63:0],  cout, overf,  complete, start | Test bench for top module. |
| Conditional\_sum\_adder\_64 | [63:0] ope1,[63:0]ope2, Cin,  add\_sub,  reset,  clock,  start,  [63:0] sum,  output cout,  complete,  overf | 64-bit conditional sum adder |
| conditional\_sum\_adder\_8 | [7:0] A,  [7:0]B, Cin,add\_sub,  reset,  clock,  start,  [7:0] sum, cout,  complete | 8-bit conditional sum adder |
| conditional\_sum\_adder\_8\_1 | [7:0] A,  [7:0]B, Cin,add\_sub,  reset,  clock,  start,  [7:0] sum, cout,  complete | 8 bit conditional sum adder |
| Conditional\_sum\_adder | [3:0] A,  [3:0]B, Cin,add\_sub,  reset,  clock,  start,  [3:0] sum, cout,  complete | 4-bit conditional sum adder |
| Mux1 | One,  Zero,sel,  Out | 1-bit MUX |
| Mux2 | [1:0]one,  [1:0]zero,  Sel,[1:0]out. | 2-bit input output MUX |

## II. Implementation Overview



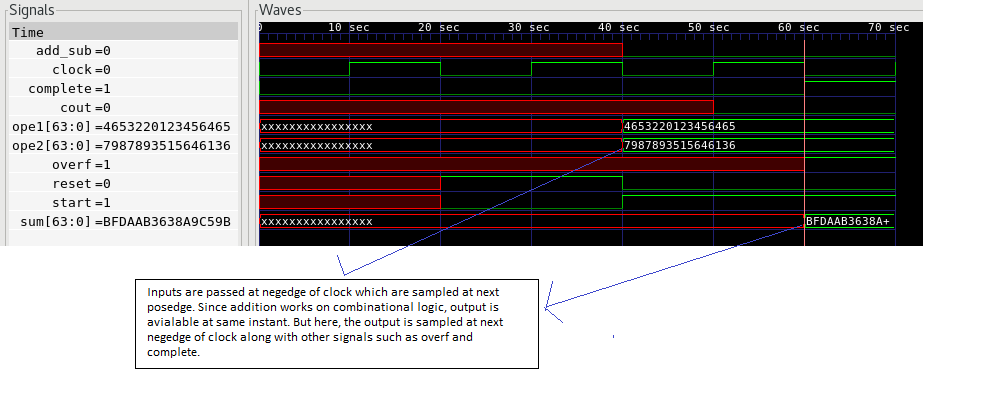
**III. RTL-Level(Pre-synthesis) Simulations/Tests**

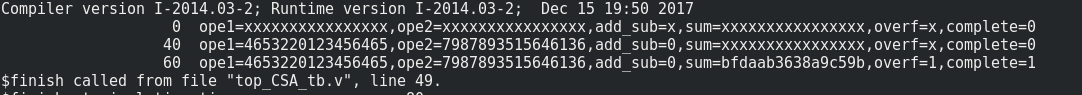
The circuit starts working when the start is high and reset is low. Complete signal is low and so overf signal has don't care initially. The inputs to the circuit is passed at negedge of the clock in order to satisfy the setup time constraint of the register. The inputs are sampled at posedge of clock and is given to adder circuit at same edge. Output is obtained at the same edge as adder circuit is combinational logic. Sum is sampled at next negedge of clock. Overf & complete is updated at same negedge of clock.

* **If 'overf' is high then the corresponding sum value is wrong.**
* **If the MSB of 'sum' is high then the number is negative and 2's complement has to be taken in order to get the actual number.**

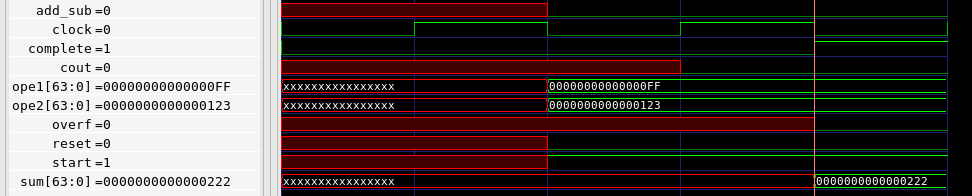
**Table III.1** – Four Selected Test Data for Carry select adder circuit

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case** | **ope1**(hex) | **ope2 (**hex) | **result (hex)** |
| **1** | 4653220123456465 | 7987893515646136 | bfdaab3638a9c59b |
| **2** | 00000000000000ff | 0000000000000123 | 0000000000000222 |
| **3** | 4895130123456465 | 7987898888846136 | Cf0d89789ac1032f |
| **4** | 0000000000000009 | 0000000000000006 | 0000000000000003 |

****

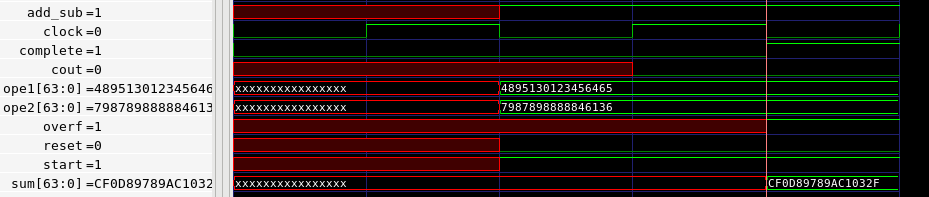


**Figure III.1a**:RTL simulation waveform that contains test case #1



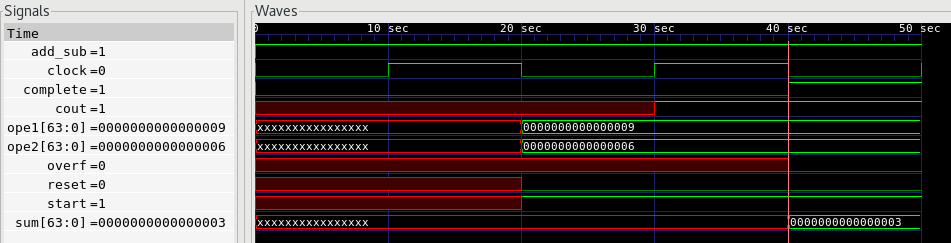
2.1 (2).PNG

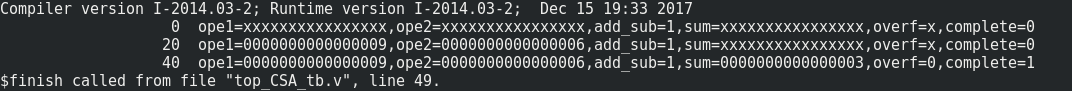
**Figure III.1b**: RTL simulation waveform that contains test case #2



3.1 (2).PNG

**Figure III.1c**: RTL simulation waveform that contains test case #3

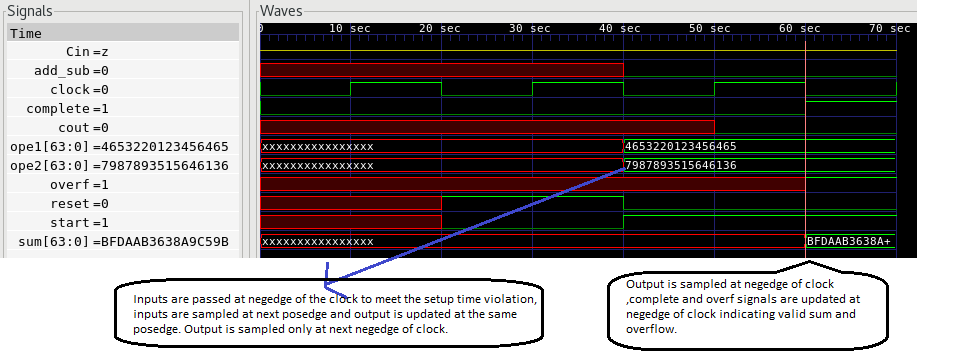


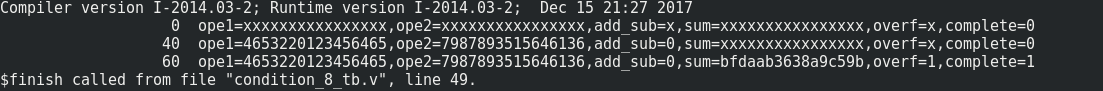


**Figure III.1d**: RTL simulation waveform that contains test case #4

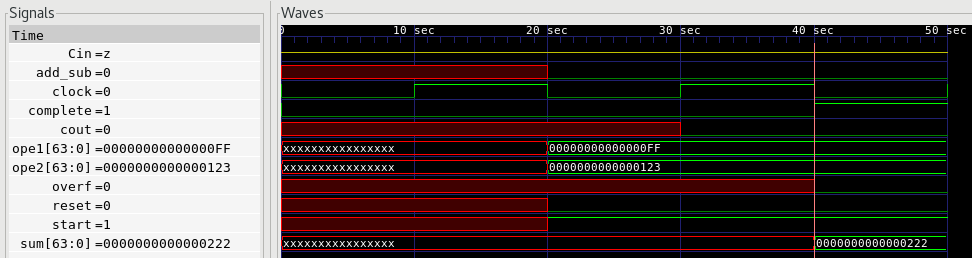
**Table III.2** – Four Selected Test Data for Conditional sum adder circuit

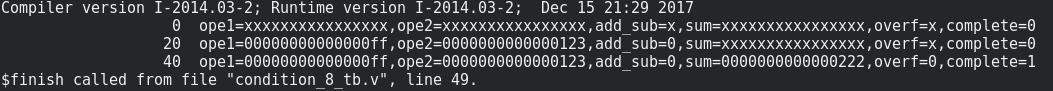
|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case** | **ope1**(hex) | **ope2 (**hex) | **result (hex)** |
| **1** | 4653220123456465 | 7987893515646136 | bfdaab3638a9c59b |
| **2** | 00000000000000ff | 0000000000000123 | 0000000000000222 |
| **3** | 4895130123456465 | 7987898888846136 | Cf0d89789ac1032f |
| **4** | 0000000000000009 | 0000000000000006 | 0000000000000003 |



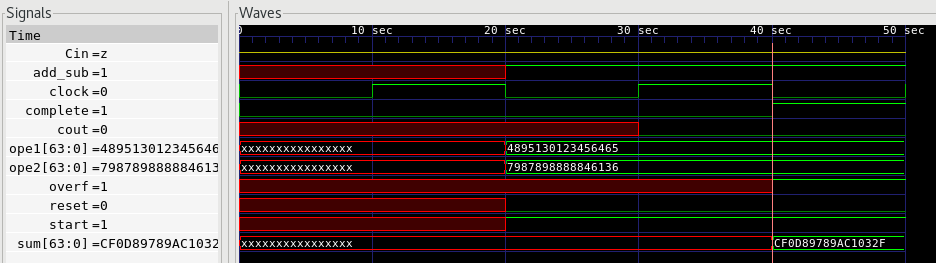


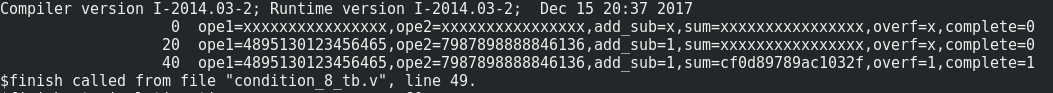
**Figure III.2a**: RTL simulation waveform that contains test case #1



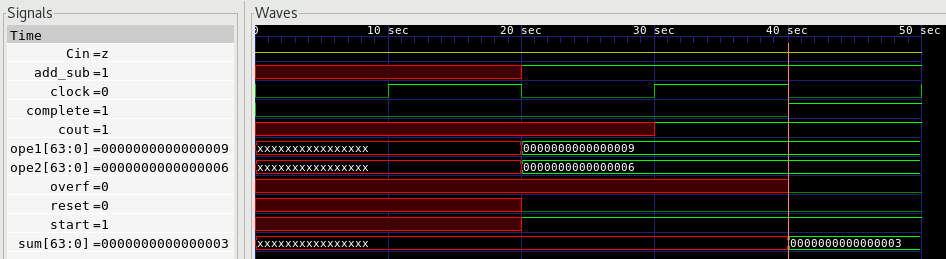


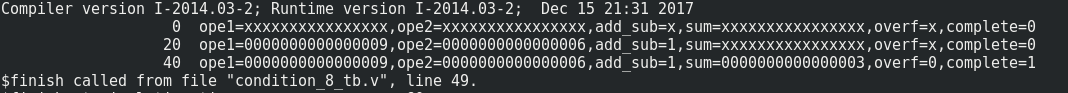
**Figure III.2b**: RTL simulation waveform that contains test case #2





**Figure III.2c**: RTL simulation waveform that contains test case #3



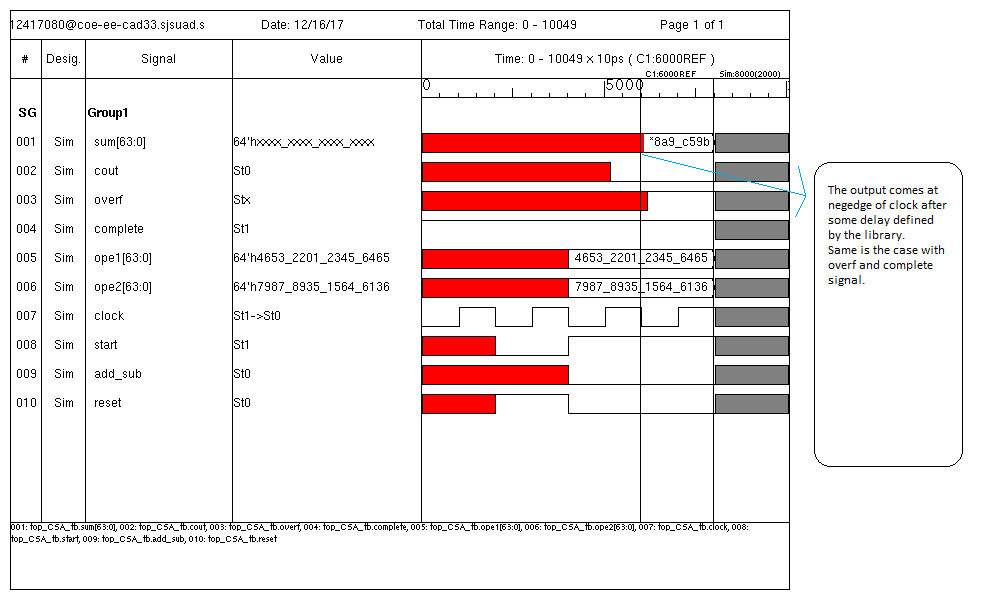


**Figure III.2d**: RTL simulation waveform that contains test case #4

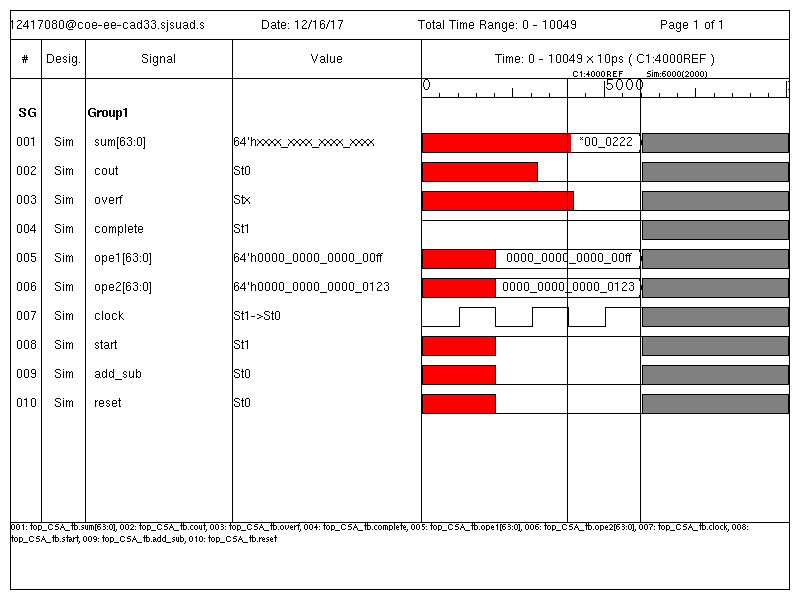
**IV. Gate-Level (Post-synthesis) Dynamic Simulations/Tests**

**Table V.1** – Four Selected Test Data for Carry Select adder circuit

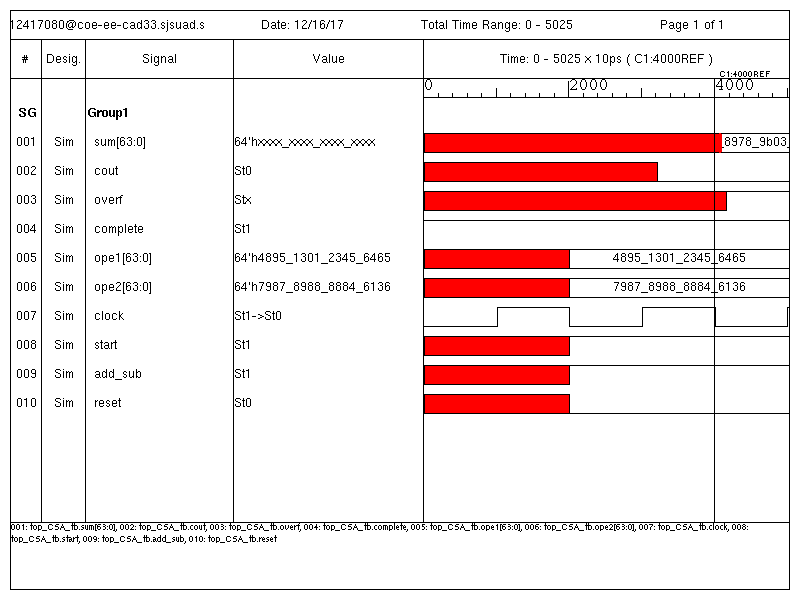
|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case** | **opera1**(hex) | **opra2 (**hex) | **Output** |
| **1** | 4653220123456465 | 7987893515646136 | bfdaab3638a9c59b |
| **2** | 00000000000000ff | 0000000000000123 | 0000000000000222 |
| **3** | 4895130123456465 | 7987898888846136 | Cf0d89789ac1032f |
| **4** | 0000000000000009 | 0000000000000006 | 0000000000000003 |

****

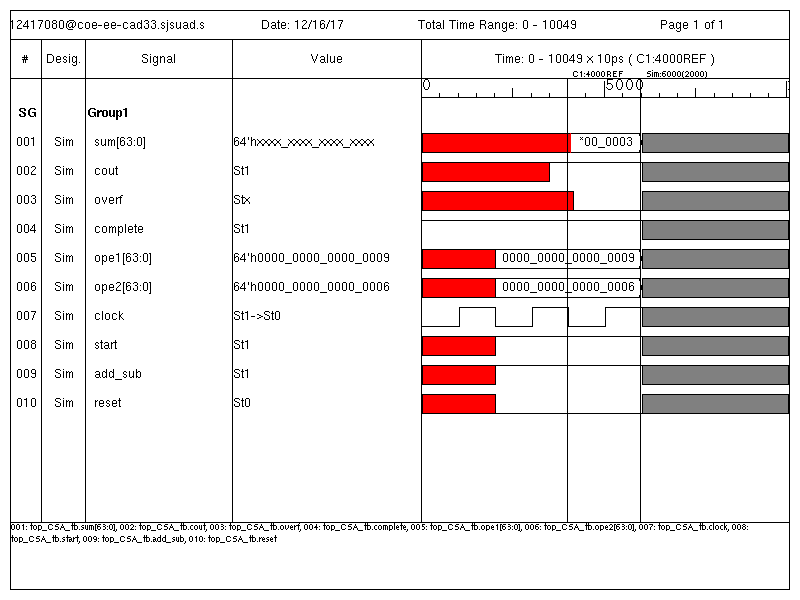
**Figure V.1a**: Gate-level simulation waveform that contains test case #1



**Figure V.1b**: Gate-level simulation waveform that contains test case #2



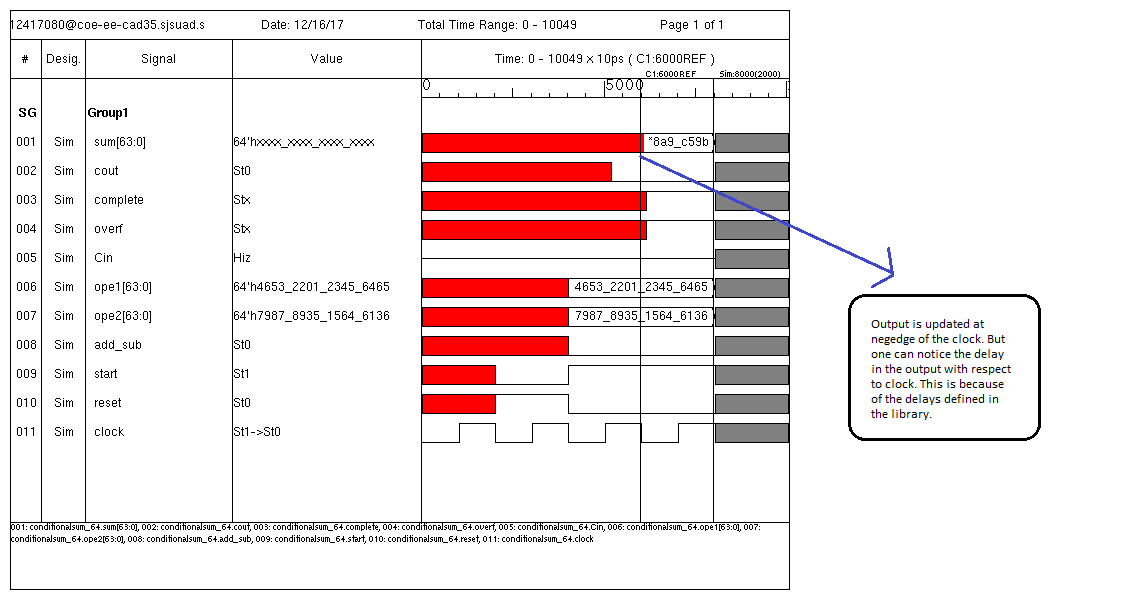
**Figure V.1c**: Gate-level simulation waveform that contains test case #3



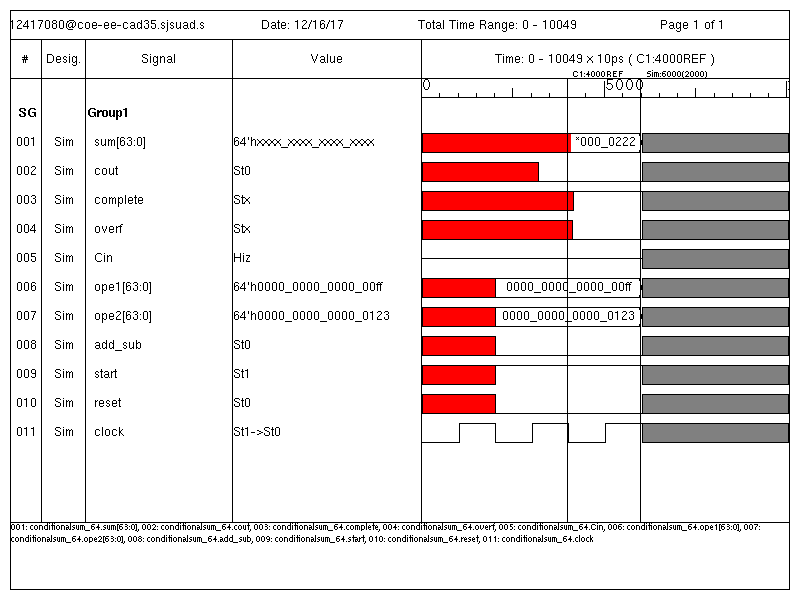
**Figure V.1d**: Gate-level simulation waveform that contains test case #4

**Table V.2** – Four Selected Test Data for Conditional sum adder circuit

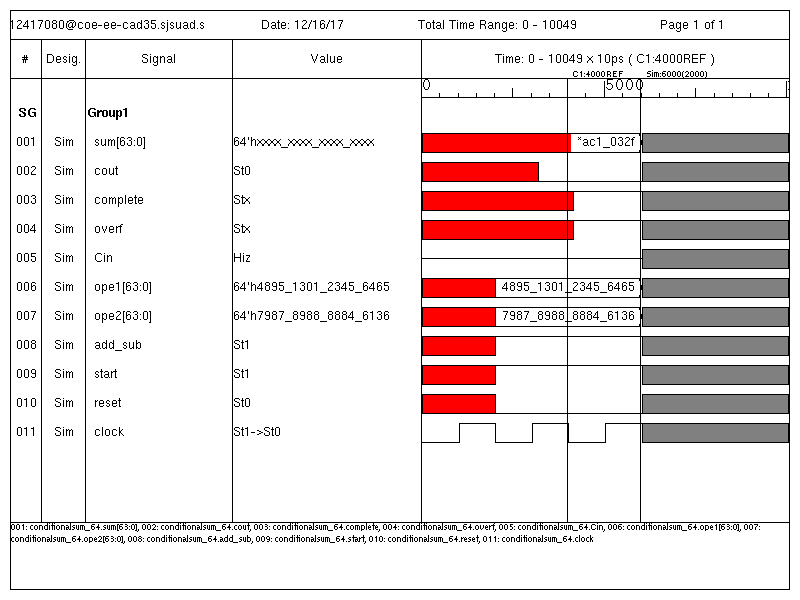
|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case** | **Ope1**(hex) | **Ope2(**hex) | **Output** |
| **1** | 4653220123456465 | 7987893515646136 | bfdaab3638a9c59b |
| **2** | 00000000000000ff | 0000000000000123 | 0000000000000222 |
| **3** | 4895130123456465 | 7987898888846136 | Cf0d89789ac1032f |
| **4** | 0000000000000009 | 0000000000000006 | 0000000000000003 |



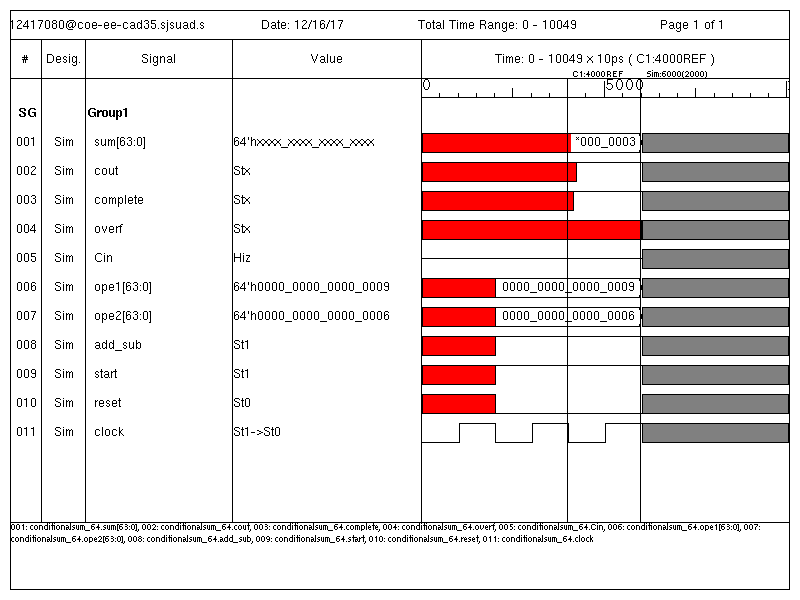
**Figure V.2a**: Gate-level simulation waveform that contains test case #1



**Figure V.2b**: Gate-level simulation waveform that contains test case #2



**Figure V.2c**: Gate-level simulation waveform that contains test case #3



**Figure V.2d**: Gate-level simulation waveform that contains test case #4

It can clearly be noted that there is library(toshiba tc240c) defined delay between the expected output and the actual output value.

**VI. Conclusion**

The analysis of two adder circuits shows that carry select adder circuit has slack of 0.41, dynamic power consumption of 8.5908mW, 562 combinational cells and 376 sequential cells. While the conditional sum has slack of 3.22, dynamic power consumption of 1.2454mW, 686 combinational cells and 241 sequential cells. So it can be concluded that Carry select adder has better area while Conditional sum adder has better power and timing performance.

**Appendix A**

**A.1 Contents from EDA Tool Configurations and Setup Files**

set link\_library {/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25}

set target\_library {/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25}

set synthetic\_library {dw\_foundation.sldb standard.sldb}

set\_min\_library /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25 -min\_version /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_BCCOM25

**A.2 Commands and/or Scripts Used for Simulation and Synthesis**

**Compilation: vcs +v2k design.v testbench.v**

**Simulation: ./simv**

**Synthesis: dc\_shell -xg -f synthesis.script | tee synres.txt**

**Gate-level simulation: vcs +v2k -debug\_all -gui -y /apps/toshiba/sjsu/verilog/tc240c +libext+.tsbvlibp +access+r top\_CSA\_tb.v top\_netlist.v**

**Script file for Carry select adder**

set link\_library {/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25}

set target\_library {/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25}

set synthetic\_library {dw\_foundation.sldb standard.sldb}

set\_min\_library /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25 -min\_version /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_BCCOM25

read\_verilog {top\_CSA.v}

current\_design top\_CSA

link

check\_design

create\_clock clock -name clock -period 10

set\_propagated\_clock clock

set\_clock\_uncertainty .05 clock

set\_fix\_hold clock

compile -map\_effort high

report\_cell

report\_net

update\_timing

report\_timing -max\_paths 8

report\_timing >> report\_time.txt

report\_area >> report\_area.txt

report\_power >> report\_power.txt

write -hierarchy -format verilog -output top\_netlist.v

quit

**Script file for Conditional sum adder**

set link\_library {/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25}

set target\_library {/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25}

set synthetic\_library {dw\_foundation.sldb standard.sldb}

set\_min\_library /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25 -min\_version /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_BCCOM25

read\_verilog {condition\_64.v}

current\_design conditional\_sum\_adder\_64

link

check\_design

create\_clock clock -name clock -period 30

set\_propagated\_clock clock

set\_clock\_uncertainty .05 clock

set\_fix\_hold clock

compile -map\_effort high

report\_cell

report\_net

update\_timing

report\_timing -max\_paths 1

report\_timing >> report\_time.txt

report\_area >> report\_area.txt

report\_power >> report\_power.txt

write -hierarchy -format verilog -output top\_netlist.v

quit

**Appendix B**

**Reports**

*1.Carry\_select\_adder Area report*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Report : area

Design : top\_CSA

Version: K-2015.06-SP5-1

Date : Sat Dec 16 11:00:16 2017

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Library(s) Used:

tc240c (File: /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25)

Number of ports: 1468

Number of nets: 2294

Number of cells: 1087

Number of combinational cells: 562

Number of sequential cells: 376

Number of macros/black boxes: 0

Number of buf/inv: 15

Number of references: 34

Combinational area: 1198.500000

Buf/Inv area: 15.000000

Noncombinational area: 1260.000000

Macro/Black Box area: 0.000000

Net Interconnect area: undefined (No wire load specified)

Total cell area: 2458.500000

Total area: undefined

1

*2. Carry\_select\_adder Power report*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Report : power

-analysis\_effort low

Design : top\_CSA

Version: K-2015.06-SP5-1

Date : Sat Dec 16 11:00:17 2017

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Library(s) Used:

tc240c (File: /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25)

Operating Conditions: WCCOM25 Library: tc240c

Wire Load Model Mode: top

Global Operating Voltage = 2.3

Power-specific unit information :

Voltage Units = 1V

Capacitance Units = 1.000000ff

Time Units = 1ns

Dynamic Power Units = 1uW (derived from V,C,T units)

Leakage Power Units = Unitless

Cell Internal Power = 7.8210 mW (91%)

Net Switching Power = 769.8483 uW (9%)

---------

Total Dynamic Power = 8.5908 mW (100%)

Cell Leakage Power = 0.0000

Error: Either dynamic power or leakage power, in the library, is unitless. Unable to display complete power group summary. (PWR-799)

Internal Switching Leakage Total

Power Group Power Power Power Power ( % ) Attrs

--------------------------------------------------------------------------------------------------

io\_pad 0.0000 0.0000 0.0000 NA ( N/A)

memory 0.0000 0.0000 0.0000 NA ( N/A)

black\_box 0.0000 0.0000 0.0000 NA ( N/A)

clock\_network 0.0000 0.0000 0.0000 NA ( N/A)

register 5.8138e+03 189.5154 0.0000 NA ( N/A)

sequential 0.0000 0.0000 0.0000 NA ( N/A)

combinational 2.0072e+03 580.3328 0.0000 NA ( N/A)

--------------------------------------------------------------------------------------------------

Total 7.8210e+03 uW 769.8482 uW 0.0000 NA

1

*3. Carry\_select\_adder timing report*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Report : timing

-path full

-delay max

-max\_paths 1

Design : top\_CSA

Version: K-2015.06-SP5-1

Date : Sat Dec 16 11:00:15 2017

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Operating Conditions: WCCOM25 Library: tc240c

Wire Load Model Mode: top

Startpoint: RCA0/B\_temp\_reg[0]

(rising edge-triggered flip-flop clocked by clock)

Endpoint: RCA0/sum\_reg[7]

(falling edge-triggered flip-flop clocked by clock)

Path Group: clock

Path Type: max

Point Incr Path

-----------------------------------------------------------

clock clock (rise edge) 0.00 0.00

clock network delay (propagated) 0.00 0.00

RCA0/B\_temp\_reg[0]/CP (CFD1QXL) 0.00 0.00 r

RCA0/B\_temp\_reg[0]/Q (CFD1QXL) 0.67 0.67 f

RCA0/U4/Z (CEOX1) 0.34 1.01 f

RCA0/FA0/B (fulladder\_0) 0.00 1.01 f

RCA0/FA0/U3/Z (CEOX1) 0.32 1.33 f

RCA0/FA0/U5/Z (CAOR2X1) 0.32 1.65 f

RCA0/FA0/carry (fulladder\_0) 0.00 1.65 f

RCA0/FA1/Cin (fulladder\_119) 0.00 1.65 f

RCA0/FA1/U5/Z (CAOR2X1) 0.35 2.00 f

RCA0/FA1/carry (fulladder\_119) 0.00 2.00 f

RCA0/FA2/Cin (fulladder\_118) 0.00 2.00 f

RCA0/FA2/U5/Z (CAOR2X1) 0.35 2.35 f

RCA0/FA2/carry (fulladder\_118) 0.00 2.35 f

RCA0/FA3/Cin (fulladder\_117) 0.00 2.35 f

RCA0/FA3/U5/Z (CAOR2X1) 0.35 2.70 f

RCA0/FA3/carry (fulladder\_117) 0.00 2.70 f

RCA0/FA4/Cin (fulladder\_116) 0.00 2.70 f

RCA0/FA4/U5/Z (CAOR2X1) 0.35 3.06 f

RCA0/FA4/carry (fulladder\_116) 0.00 3.06 f

RCA0/FA5/Cin (fulladder\_115) 0.00 3.06 f

RCA0/FA5/U5/Z (CAOR2X1) 0.35 3.41 f

RCA0/FA5/carry (fulladder\_115) 0.00 3.41 f

RCA0/FA6/Cin (fulladder\_114) 0.00 3.41 f

RCA0/FA6/U5/Z (CAOR2X1) 0.35 3.76 f

RCA0/FA6/carry (fulladder\_114) 0.00 3.76 f

RCA0/FA7/Cin (fulladder\_113) 0.00 3.76 f

RCA0/FA7/U3/Z (CEOX1) 0.28 4.04 f

RCA0/FA7/sum (fulladder\_113) 0.00 4.04 f

RCA0/sum\_reg[7]/D (CFDN1QXL) 0.00 4.04 f

data arrival time 4.04

clock clock (fall edge) 5.00 5.00

clock network delay (propagated) 0.00 5.00

clock uncertainty -0.05 4.95

RCA0/sum\_reg[7]/CPN (CFDN1QXL) 0.00 4.95 f

library setup time -0.50 4.45

data required time 4.45

-----------------------------------------------------------

data required time 4.45

data arrival time -4.04

-----------------------------------------------------------

slack (MET) 0.41

1

*4. Conditional sum adder area report*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Report : area

Design : conditional\_sum\_adder\_64

Version: K-2015.06-SP5-1

Date : Sat Dec 16 11:15:29 2017

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Library(s) Used:

tc240c (File: /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25)

Number of ports: 1871

Number of nets: 2713

Number of cells: 1191

Number of combinational cells: 686

Number of sequential cells: 241

Number of macros/black boxes: 0

Number of buf/inv: 129

Number of references: 14

Combinational area: 1367.000000

Buf/Inv area: 129.000000

Noncombinational area: 728.000000

Macro/Black Box area: 0.000000

Net Interconnect area: undefined (No wire load specified)

Total cell area: 2095.000000

Total area: undefined

1

*5. Conditional sum adder power report*

Loading db file '/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25'

Warning: Main library 'tc240c' does not specify the following unit required for power: 'Leakage Power'. (PWR-424)

Information: Propagating switching activity (low effort zero delay simulation). (PWR-6)

Warning: Design has unannotated primary inputs. (PWR-414)

Warning: Design has unannotated sequential cell outputs. (PWR-415)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Report : power

-analysis\_effort low

Design : conditional\_sum\_adder\_64

Version: K-2015.06-SP5-1

Date : Sat Dec 16 11:15:29 2017

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Library(s) Used:

tc240c (File: /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25)

Operating Conditions: WCCOM25 Library: tc240c

Wire Load Model Mode: top

Global Operating Voltage = 2.3

Power-specific unit information :

Voltage Units = 1V

Capacitance Units = 1.000000ff

Time Units = 1ns

Dynamic Power Units = 1uW (derived from V,C,T units)

Leakage Power Units = Unitless

Cell Internal Power = 1.0924 mW (88%)

Net Switching Power = 152.9883 uW (12%)

---------

Total Dynamic Power = 1.2454 mW (100%)

Cell Leakage Power = 0.0000

Error: Either dynamic power or leakage power, in the library, is unitless. Unable to display complete power group summary. (PWR-799)

Internal Switching Leakage Total

Power Group Power Power Power Power ( % ) Attrs

--------------------------------------------------------------------------------------------------

io\_pad 0.0000 0.0000 0.0000 NA ( N/A)

memory 0.0000 0.0000 0.0000 NA ( N/A)

black\_box 0.0000 0.0000 0.0000 NA ( N/A)

clock\_network 0.0000 0.0000 0.0000 NA ( N/A)

register 683.0954 20.4825 0.0000 NA ( N/A)

sequential 0.0000 0.0000 0.0000 NA ( N/A)

combinational 409.3056 132.5057 0.0000 NA ( N/A)

--------------------------------------------------------------------------------------------------

Total 1.0924e+03 uW 152.9883 uW 0.0000 NA

1

*6. Conditional sum adder timing report*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Report : timing

-path full

-delay max

-max\_paths 1

Design : conditional\_sum\_adder\_64

Version: K-2015.06-SP5-1

Date : Sat Dec 16 11:15:29 2017

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Operating Conditions: WCCOM25 Library: tc240c

Wire Load Model Mode: top

Startpoint: CSA\_0/CSA\_4\_1/B\_temp\_reg[2]

(rising edge-triggered flip-flop clocked by clock)

Endpoint: CSA\_7/CSA\_4\_2/sum\_reg[2]

(falling edge-triggered flip-flop clocked by clock)

Path Group: clock

Path Type: max

Point Incr Path

--------------------------------------------------------------------------

clock clock (rise edge) 0.00 0.00

clock network delay (propagated) 0.00 0.00

CSA\_0/CSA\_4\_1/B\_temp\_reg[2]/CP (CFD1QXL) 0.00 0.00 r

CSA\_0/CSA\_4\_1/B\_temp\_reg[2]/Q (CFD1QXL) 0.59 0.59 r

CSA\_0/CSA\_4\_1/U3/Z (CEOX1) 0.49 1.08 r

CSA\_0/CSA\_4\_1/FA3\_1/B (fulladder\_108) 0.00 1.08 r

CSA\_0/CSA\_4\_1/FA3\_1/U3/Z (CEOX1) 0.38 1.47 r

CSA\_0/CSA\_4\_1/FA3\_1/U5/Z (CAOR2X1) 0.35 1.82 r

CSA\_0/CSA\_4\_1/FA3\_1/carry (fulladder\_108) 0.00 1.82 r

CSA\_0/CSA\_4\_1/MUX3\_CARRY\_1/sel (mux1\_107) 0.00 1.82 r

CSA\_0/CSA\_4\_1/MUX3\_CARRY\_1/U1/Z (CIVX2) 0.08 1.90 f

CSA\_0/CSA\_4\_1/MUX3\_CARRY\_1/U2/Z (CAOR2X1) 0.33 2.23 f

CSA\_0/CSA\_4\_1/MUX3\_CARRY\_1/out (mux1\_107) 0.00 2.23 f

CSA\_0/CSA\_4\_1/MUX3\_CARRY\_2/one (mux1\_106) 0.00 2.23 f

CSA\_0/CSA\_4\_1/MUX3\_CARRY\_2/U2/Z (CAOR2X1) 0.35 2.58 f

CSA\_0/CSA\_4\_1/MUX3\_CARRY\_2/out (mux1\_106) 0.00 2.58 f

CSA\_0/CSA\_4\_1/cout (conditional\_sum\_adder) 0.00 2.58 f

CSA\_0/CSA\_4\_2/Cin (conditional\_sum\_adder\_1\_0) 0.00 2.58 f

CSA\_0/CSA\_4\_2/FA1/Cin (fulladder\_105) 0.00 2.58 f

CSA\_0/CSA\_4\_2/FA1/U5/Z (CAOR2X1) 0.40 2.98 f

CSA\_0/CSA\_4\_2/FA1/carry (fulladder\_105) 0.00 2.98 f

CSA\_0/CSA\_4\_2/MUX2\_CARRY/sel (mux1\_104) 0.00 2.98 f

CSA\_0/CSA\_4\_2/MUX2\_CARRY/U1/Z (CIVX2) 0.06 3.04 r

CSA\_0/CSA\_4\_2/MUX2\_CARRY/U2/Z (CAOR2X1) 0.33 3.37 r

CSA\_0/CSA\_4\_2/MUX2\_CARRY/out (mux1\_104) 0.00 3.37 r

CSA\_0/CSA\_4\_2/MUX3\_CARRY\_2/sel (mux1\_99) 0.00 3.37 r

CSA\_0/CSA\_4\_2/MUX3\_CARRY\_2/U1/Z (CIVX2) 0.09 3.46 f

CSA\_0/CSA\_4\_2/MUX3\_CARRY\_2/U2/Z (CAOR2X1) 0.37 3.82 f

CSA\_0/CSA\_4\_2/MUX3\_CARRY\_2/out (mux1\_99) 0.00 3.82 f

CSA\_0/CSA\_4\_2/cout (conditional\_sum\_adder\_1\_0) 0.00 3.82 f

CSA\_0/cout (conditional\_sum\_adder\_8) 0.00 3.82 f

CSA\_1/Cin (conditional\_sum\_adder\_8\_1\_0) 0.00 3.82 f

CSA\_1/CSA\_4\_1/Cin (conditional\_sum\_adder\_1\_14) 0.00 3.82 f

CSA\_1/CSA\_4\_1/FA1/Cin (fulladder\_98) 0.00 3.82 f

CSA\_1/CSA\_4\_1/FA1/U5/Z (CAOR2X1) 0.40 4.22 f

CSA\_1/CSA\_4\_1/FA1/carry (fulladder\_98) 0.00 4.22 f

CSA\_1/CSA\_4\_1/MUX2\_CARRY/sel (mux1\_97) 0.00 4.22 f

CSA\_1/CSA\_4\_1/MUX2\_CARRY/U1/Z (CIVX2) 0.06 4.28 r

CSA\_1/CSA\_4\_1/MUX2\_CARRY/U2/Z (CAOR2X1) 0.33 4.61 r

CSA\_1/CSA\_4\_1/MUX2\_CARRY/out (mux1\_97) 0.00 4.61 r

CSA\_1/CSA\_4\_1/MUX3\_CARRY\_2/sel (mux1\_92) 0.00 4.61 r

CSA\_1/CSA\_4\_1/MUX3\_CARRY\_2/U1/Z (CIVX2) 0.09 4.70 f

CSA\_1/CSA\_4\_1/MUX3\_CARRY\_2/U2/Z (CAOR2X1) 0.37 5.07 f

CSA\_1/CSA\_4\_1/MUX3\_CARRY\_2/out (mux1\_92) 0.00 5.07 f

CSA\_1/CSA\_4\_1/cout (conditional\_sum\_adder\_1\_14) 0.00 5.07 f

CSA\_1/CSA\_4\_2/Cin (conditional\_sum\_adder\_1\_13) 0.00 5.07 f

CSA\_1/CSA\_4\_2/FA1/Cin (fulladder\_91) 0.00 5.07 f

CSA\_1/CSA\_4\_2/FA1/U5/Z (CAOR2X1) 0.40 5.47 f

CSA\_1/CSA\_4\_2/FA1/carry (fulladder\_91) 0.00 5.47 f

CSA\_1/CSA\_4\_2/MUX2\_CARRY/sel (mux1\_90) 0.00 5.47 f

CSA\_1/CSA\_4\_2/MUX2\_CARRY/U1/Z (CIVX2) 0.06 5.53 r

CSA\_1/CSA\_4\_2/MUX2\_CARRY/U2/Z (CAOR2X1) 0.33 5.86 r

CSA\_1/CSA\_4\_2/MUX2\_CARRY/out (mux1\_90) 0.00 5.86 r

CSA\_1/CSA\_4\_2/MUX3\_CARRY\_2/sel (mux1\_85) 0.00 5.86 r

CSA\_1/CSA\_4\_2/MUX3\_CARRY\_2/U1/Z (CIVX2) 0.09 5.95 f

CSA\_1/CSA\_4\_2/MUX3\_CARRY\_2/U2/Z (CAOR2X1) 0.37 6.31 f

CSA\_1/CSA\_4\_2/MUX3\_CARRY\_2/out (mux1\_85) 0.00 6.31 f

CSA\_1/CSA\_4\_2/cout (conditional\_sum\_adder\_1\_13) 0.00 6.31 f

CSA\_1/cout (conditional\_sum\_adder\_8\_1\_0) 0.00 6.31 f

CSA\_2/Cin (conditional\_sum\_adder\_8\_1\_6) 0.00 6.31 f

CSA\_2/CSA\_4\_1/Cin (conditional\_sum\_adder\_1\_12) 0.00 6.31 f

CSA\_2/CSA\_4\_1/FA1/Cin (fulladder\_84) 0.00 6.31 f

CSA\_2/CSA\_4\_1/FA1/U5/Z (CAOR2X1) 0.40 6.71 f

CSA\_2/CSA\_4\_1/FA1/carry (fulladder\_84) 0.00 6.71 f

CSA\_2/CSA\_4\_1/MUX2\_CARRY/sel (mux1\_83) 0.00 6.71 f

CSA\_2/CSA\_4\_1/MUX2\_CARRY/U1/Z (CIVX2) 0.06 6.77 r

CSA\_2/CSA\_4\_1/MUX2\_CARRY/U2/Z (CAOR2X1) 0.33 7.11 r

CSA\_2/CSA\_4\_1/MUX2\_CARRY/out (mux1\_83) 0.00 7.11 r

CSA\_2/CSA\_4\_1/MUX3\_CARRY\_2/sel (mux1\_78) 0.00 7.11 r

CSA\_2/CSA\_4\_1/MUX3\_CARRY\_2/U1/Z (CIVX2) 0.09 7.19 f

CSA\_2/CSA\_4\_1/MUX3\_CARRY\_2/U2/Z (CAOR2X1) 0.37 7.56 f

CSA\_2/CSA\_4\_1/MUX3\_CARRY\_2/out (mux1\_78) 0.00 7.56 f

CSA\_2/CSA\_4\_1/cout (conditional\_sum\_adder\_1\_12) 0.00 7.56 f

CSA\_2/CSA\_4\_2/Cin (conditional\_sum\_adder\_1\_11) 0.00 7.56 f

CSA\_2/CSA\_4\_2/FA1/Cin (fulladder\_77) 0.00 7.56 f

CSA\_2/CSA\_4\_2/FA1/U5/Z (CAOR2X1) 0.40 7.96 f

CSA\_2/CSA\_4\_2/FA1/carry (fulladder\_77) 0.00 7.96 f

CSA\_2/CSA\_4\_2/MUX2\_CARRY/sel (mux1\_76) 0.00 7.96 f

CSA\_2/CSA\_4\_2/MUX2\_CARRY/U1/Z (CIVX2) 0.06 8.02 r

CSA\_2/CSA\_4\_2/MUX2\_CARRY/U2/Z (CAOR2X1) 0.33 8.35 r

CSA\_2/CSA\_4\_2/MUX2\_CARRY/out (mux1\_76) 0.00 8.35 r

CSA\_2/CSA\_4\_2/MUX3\_CARRY\_2/sel (mux1\_71) 0.00 8.35 r

CSA\_2/CSA\_4\_2/MUX3\_CARRY\_2/U1/Z (CIVX2) 0.09 8.44 f

CSA\_2/CSA\_4\_2/MUX3\_CARRY\_2/U2/Z (CAOR2X1) 0.37 8.81 f

CSA\_2/CSA\_4\_2/MUX3\_CARRY\_2/out (mux1\_71) 0.00 8.81 f

CSA\_2/CSA\_4\_2/cout (conditional\_sum\_adder\_1\_11) 0.00 8.81 f

CSA\_2/cout (conditional\_sum\_adder\_8\_1\_6) 0.00 8.81 f

CSA\_3/Cin (conditional\_sum\_adder\_8\_1\_5) 0.00 8.81 f

CSA\_3/CSA\_4\_1/Cin (conditional\_sum\_adder\_1\_10) 0.00 8.81 f

CSA\_3/CSA\_4\_1/FA1/Cin (fulladder\_70) 0.00 8.81 f

CSA\_3/CSA\_4\_1/FA1/U5/Z (CAOR2X1) 0.40 9.20 f

CSA\_3/CSA\_4\_1/FA1/carry (fulladder\_70) 0.00 9.20 f

CSA\_3/CSA\_4\_1/MUX2\_CARRY/sel (mux1\_69) 0.00 9.20 f

CSA\_3/CSA\_4\_1/MUX2\_CARRY/U1/Z (CIVX2) 0.06 9.26 r

CSA\_3/CSA\_4\_1/MUX2\_CARRY/U2/Z (CAOR2X1) 0.33 9.60 r

CSA\_3/CSA\_4\_1/MUX2\_CARRY/out (mux1\_69) 0.00 9.60 r

CSA\_3/CSA\_4\_1/MUX3\_CARRY\_2/sel (mux1\_64) 0.00 9.60 r

CSA\_3/CSA\_4\_1/MUX3\_CARRY\_2/U1/Z (CIVX2) 0.09 9.68 f

CSA\_3/CSA\_4\_1/MUX3\_CARRY\_2/U2/Z (CAOR2X1) 0.37 10.05 f

CSA\_3/CSA\_4\_1/MUX3\_CARRY\_2/out (mux1\_64) 0.00 10.05 f

CSA\_3/CSA\_4\_1/cout (conditional\_sum\_adder\_1\_10) 0.00 10.05 f

CSA\_3/CSA\_4\_2/Cin (conditional\_sum\_adder\_1\_9) 0.00 10.05 f

CSA\_3/CSA\_4\_2/FA1/Cin (fulladder\_63) 0.00 10.05 f

CSA\_3/CSA\_4\_2/FA1/U5/Z (CAOR2X1) 0.40 10.45 f

CSA\_3/CSA\_4\_2/FA1/carry (fulladder\_63) 0.00 10.45 f

CSA\_3/CSA\_4\_2/MUX2\_CARRY/sel (mux1\_62) 0.00 10.45 f

CSA\_3/CSA\_4\_2/MUX2\_CARRY/U1/Z (CIVX2) 0.06 10.51 r

CSA\_3/CSA\_4\_2/MUX2\_CARRY/U2/Z (CAOR2X1) 0.33 10.84 r

CSA\_3/CSA\_4\_2/MUX2\_CARRY/out (mux1\_62) 0.00 10.84 r

CSA\_3/CSA\_4\_2/MUX3\_CARRY\_2/sel (mux1\_57) 0.00 10.84 r

CSA\_3/CSA\_4\_2/MUX3\_CARRY\_2/U1/Z (CIVX2) 0.09 10.93 f

CSA\_3/CSA\_4\_2/MUX3\_CARRY\_2/U2/Z (CAOR2X1) 0.37 11.30 f

CSA\_3/CSA\_4\_2/MUX3\_CARRY\_2/out (mux1\_57) 0.00 11.30 f

CSA\_3/CSA\_4\_2/cout (conditional\_sum\_adder\_1\_9) 0.00 11.30 f

CSA\_3/cout (conditional\_sum\_adder\_8\_1\_5) 0.00 11.30 f

CSA\_4/Cin (conditional\_sum\_adder\_8\_1\_4) 0.00 11.30 f

CSA\_4/CSA\_4\_1/Cin (conditional\_sum\_adder\_1\_8) 0.00 11.30 f

CSA\_4/CSA\_4\_1/FA1/Cin (fulladder\_56) 0.00 11.30 f

CSA\_4/CSA\_4\_1/FA1/U5/Z (CAOR2X1) 0.40 11.70 f

CSA\_4/CSA\_4\_1/FA1/carry (fulladder\_56) 0.00 11.70 f

CSA\_4/CSA\_4\_1/MUX2\_CARRY/sel (mux1\_55) 0.00 11.70 f

CSA\_4/CSA\_4\_1/MUX2\_CARRY/U1/Z (CIVX2) 0.06 11.76 r

CSA\_4/CSA\_4\_1/MUX2\_CARRY/U2/Z (CAOR2X1) 0.33 12.09 r

CSA\_4/CSA\_4\_1/MUX2\_CARRY/out (mux1\_55) 0.00 12.09 r

CSA\_4/CSA\_4\_1/MUX3\_CARRY\_2/sel (mux1\_50) 0.00 12.09 r

CSA\_4/CSA\_4\_1/MUX3\_CARRY\_2/U1/Z (CIVX2) 0.09 12.18 f

CSA\_4/CSA\_4\_1/MUX3\_CARRY\_2/U2/Z (CAOR2X1) 0.37 12.54 f

CSA\_4/CSA\_4\_1/MUX3\_CARRY\_2/out (mux1\_50) 0.00 12.54 f

CSA\_4/CSA\_4\_1/cout (conditional\_sum\_adder\_1\_8) 0.00 12.54 f

CSA\_4/CSA\_4\_2/Cin (conditional\_sum\_adder\_1\_7) 0.00 12.54 f

CSA\_4/CSA\_4\_2/FA1/Cin (fulladder\_49) 0.00 12.54 f

CSA\_4/CSA\_4\_2/FA1/U5/Z (CAOR2X1) 0.40 12.94 f

CSA\_4/CSA\_4\_2/FA1/carry (fulladder\_49) 0.00 12.94 f

CSA\_4/CSA\_4\_2/MUX2\_CARRY/sel (mux1\_48) 0.00 12.94 f

CSA\_4/CSA\_4\_2/MUX2\_CARRY/U1/Z (CIVX2) 0.06 13.00 r

CSA\_4/CSA\_4\_2/MUX2\_CARRY/U2/Z (CAOR2X1) 0.33 13.34 r

CSA\_4/CSA\_4\_2/MUX2\_CARRY/out (mux1\_48) 0.00 13.34 r

CSA\_4/CSA\_4\_2/MUX3\_CARRY\_2/sel (mux1\_43) 0.00 13.34 r

CSA\_4/CSA\_4\_2/MUX3\_CARRY\_2/U1/Z (CIVX2) 0.09 13.42 f

CSA\_4/CSA\_4\_2/MUX3\_CARRY\_2/U2/Z (CAOR2X1) 0.37 13.79 f

CSA\_4/CSA\_4\_2/MUX3\_CARRY\_2/out (mux1\_43) 0.00 13.79 f

CSA\_4/CSA\_4\_2/cout (conditional\_sum\_adder\_1\_7) 0.00 13.79 f

CSA\_4/cout (conditional\_sum\_adder\_8\_1\_4) 0.00 13.79 f

CSA\_5/Cin (conditional\_sum\_adder\_8\_1\_3) 0.00 13.79 f

CSA\_5/CSA\_4\_1/Cin (conditional\_sum\_adder\_1\_6) 0.00 13.79 f

CSA\_5/CSA\_4\_1/FA1/Cin (fulladder\_42) 0.00 13.79 f

CSA\_5/CSA\_4\_1/FA1/U5/Z (CAOR2X1) 0.40 14.19 f

CSA\_5/CSA\_4\_1/FA1/carry (fulladder\_42) 0.00 14.19 f

CSA\_5/CSA\_4\_1/MUX2\_CARRY/sel (mux1\_41) 0.00 14.19 f

CSA\_5/CSA\_4\_1/MUX2\_CARRY/U1/Z (CIVX2) 0.06 14.25 r

CSA\_5/CSA\_4\_1/MUX2\_CARRY/U2/Z (CAOR2X1) 0.33 14.58 r

CSA\_5/CSA\_4\_1/MUX2\_CARRY/out (mux1\_41) 0.00 14.58 r

CSA\_5/CSA\_4\_1/MUX3\_CARRY\_2/sel (mux1\_36) 0.00 14.58 r

CSA\_5/CSA\_4\_1/MUX3\_CARRY\_2/U1/Z (CIVX2) 0.09 14.67 f

CSA\_5/CSA\_4\_1/MUX3\_CARRY\_2/U2/Z (CAOR2X1) 0.37 15.03 f

CSA\_5/CSA\_4\_1/MUX3\_CARRY\_2/out (mux1\_36) 0.00 15.03 f

CSA\_5/CSA\_4\_1/cout (conditional\_sum\_adder\_1\_6) 0.00 15.03 f

CSA\_5/CSA\_4\_2/Cin (conditional\_sum\_adder\_1\_5) 0.00 15.03 f

CSA\_5/CSA\_4\_2/FA1/Cin (fulladder\_35) 0.00 15.03 f

CSA\_5/CSA\_4\_2/FA1/U5/Z (CAOR2X1) 0.40 15.43 f

CSA\_5/CSA\_4\_2/FA1/carry (fulladder\_35) 0.00 15.43 f

CSA\_5/CSA\_4\_2/MUX2\_CARRY/sel (mux1\_34) 0.00 15.43 f

CSA\_5/CSA\_4\_2/MUX2\_CARRY/U1/Z (CIVX2) 0.06 15.49 r

CSA\_5/CSA\_4\_2/MUX2\_CARRY/U2/Z (CAOR2X1) 0.33 15.83 r

CSA\_5/CSA\_4\_2/MUX2\_CARRY/out (mux1\_34) 0.00 15.83 r

CSA\_5/CSA\_4\_2/MUX3\_CARRY\_2/sel (mux1\_29) 0.00 15.83 r

CSA\_5/CSA\_4\_2/MUX3\_CARRY\_2/U1/Z (CIVX2) 0.09 15.91 f

CSA\_5/CSA\_4\_2/MUX3\_CARRY\_2/U2/Z (CAOR2X1) 0.37 16.28 f

CSA\_5/CSA\_4\_2/MUX3\_CARRY\_2/out (mux1\_29) 0.00 16.28 f

CSA\_5/CSA\_4\_2/cout (conditional\_sum\_adder\_1\_5) 0.00 16.28 f

CSA\_5/cout (conditional\_sum\_adder\_8\_1\_3) 0.00 16.28 f

CSA\_6/Cin (conditional\_sum\_adder\_8\_1\_2) 0.00 16.28 f

CSA\_6/CSA\_4\_1/Cin (conditional\_sum\_adder\_1\_4) 0.00 16.28 f

CSA\_6/CSA\_4\_1/FA1/Cin (fulladder\_28) 0.00 16.28 f

CSA\_6/CSA\_4\_1/FA1/U5/Z (CAOR2X1) 0.40 16.68 f

CSA\_6/CSA\_4\_1/FA1/carry (fulladder\_28) 0.00 16.68 f

CSA\_6/CSA\_4\_1/MUX2\_CARRY/sel (mux1\_27) 0.00 16.68 f

CSA\_6/CSA\_4\_1/MUX2\_CARRY/U1/Z (CIVX2) 0.06 16.74 r

CSA\_6/CSA\_4\_1/MUX2\_CARRY/U2/Z (CAOR2X1) 0.33 17.07 r

CSA\_6/CSA\_4\_1/MUX2\_CARRY/out (mux1\_27) 0.00 17.07 r

CSA\_6/CSA\_4\_1/MUX3\_CARRY\_2/sel (mux1\_22) 0.00 17.07 r

CSA\_6/CSA\_4\_1/MUX3\_CARRY\_2/U1/Z (CIVX2) 0.09 17.16 f

CSA\_6/CSA\_4\_1/MUX3\_CARRY\_2/U2/Z (CAOR2X1) 0.37 17.53 f

CSA\_6/CSA\_4\_1/MUX3\_CARRY\_2/out (mux1\_22) 0.00 17.53 f

CSA\_6/CSA\_4\_1/cout (conditional\_sum\_adder\_1\_4) 0.00 17.53 f

CSA\_6/CSA\_4\_2/Cin (conditional\_sum\_adder\_1\_3) 0.00 17.53 f

CSA\_6/CSA\_4\_2/FA1/Cin (fulladder\_21) 0.00 17.53 f

CSA\_6/CSA\_4\_2/FA1/U5/Z (CAOR2X1) 0.40 17.92 f

CSA\_6/CSA\_4\_2/FA1/carry (fulladder\_21) 0.00 17.92 f

CSA\_6/CSA\_4\_2/MUX2\_CARRY/sel (mux1\_20) 0.00 17.92 f

CSA\_6/CSA\_4\_2/MUX2\_CARRY/U1/Z (CIVX2) 0.06 17.98 r

CSA\_6/CSA\_4\_2/MUX2\_CARRY/U2/Z (CAOR2X1) 0.33 18.32 r

CSA\_6/CSA\_4\_2/MUX2\_CARRY/out (mux1\_20) 0.00 18.32 r

CSA\_6/CSA\_4\_2/MUX3\_CARRY\_2/sel (mux1\_15) 0.00 18.32 r

CSA\_6/CSA\_4\_2/MUX3\_CARRY\_2/U1/Z (CIVX2) 0.09 18.40 f

CSA\_6/CSA\_4\_2/MUX3\_CARRY\_2/U2/Z (CAOR2X1) 0.37 18.77 f

CSA\_6/CSA\_4\_2/MUX3\_CARRY\_2/out (mux1\_15) 0.00 18.77 f

CSA\_6/CSA\_4\_2/cout (conditional\_sum\_adder\_1\_3) 0.00 18.77 f

CSA\_6/cout (conditional\_sum\_adder\_8\_1\_2) 0.00 18.77 f

CSA\_7/Cin (conditional\_sum\_adder\_8\_1\_1) 0.00 18.77 f

CSA\_7/CSA\_4\_1/Cin (conditional\_sum\_adder\_1\_2) 0.00 18.77 f

CSA\_7/CSA\_4\_1/FA1/Cin (fulladder\_14) 0.00 18.77 f

CSA\_7/CSA\_4\_1/FA1/U5/Z (CAOR2X1) 0.40 19.17 f

CSA\_7/CSA\_4\_1/FA1/carry (fulladder\_14) 0.00 19.17 f

CSA\_7/CSA\_4\_1/MUX2\_CARRY/sel (mux1\_13) 0.00 19.17 f

CSA\_7/CSA\_4\_1/MUX2\_CARRY/U1/Z (CIVX2) 0.06 19.23 r

CSA\_7/CSA\_4\_1/MUX2\_CARRY/U2/Z (CAOR2X1) 0.33 19.56 r

CSA\_7/CSA\_4\_1/MUX2\_CARRY/out (mux1\_13) 0.00 19.56 r

CSA\_7/CSA\_4\_1/MUX3\_CARRY\_2/sel (mux1\_8) 0.00 19.56 r

CSA\_7/CSA\_4\_1/MUX3\_CARRY\_2/U1/Z (CIVX2) 0.09 19.65 f

CSA\_7/CSA\_4\_1/MUX3\_CARRY\_2/U2/Z (CAOR2X1) 0.37 20.02 f

CSA\_7/CSA\_4\_1/MUX3\_CARRY\_2/out (mux1\_8) 0.00 20.02 f

CSA\_7/CSA\_4\_1/cout (conditional\_sum\_adder\_1\_2) 0.00 20.02 f

CSA\_7/CSA\_4\_2/Cin (conditional\_sum\_adder\_1\_1) 0.00 20.02 f

CSA\_7/CSA\_4\_2/FA1/Cin (fulladder\_7) 0.00 20.02 f

CSA\_7/CSA\_4\_2/FA1/U5/Z (CAOR2X1) 0.40 20.42 f

CSA\_7/CSA\_4\_2/FA1/carry (fulladder\_7) 0.00 20.42 f

CSA\_7/CSA\_4\_2/MUX2\_CARRY/sel (mux1\_6) 0.00 20.42 f

CSA\_7/CSA\_4\_2/MUX2\_CARRY/U1/Z (CIVX2) 0.06 20.48 r

CSA\_7/CSA\_4\_2/MUX2\_CARRY/U2/Z (CAOR2X1) 0.33 20.81 r

CSA\_7/CSA\_4\_2/MUX2\_CARRY/out (mux1\_6) 0.00 20.81 r

CSA\_7/CSA\_4\_2/MUX8/sel (mux2\_1) 0.00 20.81 r

CSA\_7/CSA\_4\_2/MUX8/U1/Z (CIVX2) 0.10 20.91 f

CSA\_7/CSA\_4\_2/MUX8/U3/Z (CAOR2X1) 0.32 21.23 f

CSA\_7/CSA\_4\_2/MUX8/out[0] (mux2\_1) 0.00 21.23 f

CSA\_7/CSA\_4\_2/sum\_reg[2]/D (CFDN1QXL) 0.00 21.23 f

data arrival time 21.23

clock clock (fall edge) 25.00 25.00

clock network delay (propagated) 0.00 25.00

clock uncertainty -0.05 24.95

CSA\_7/CSA\_4\_2/sum\_reg[2]/CPN (CFDN1QXL) 0.00 24.95 f

library setup time -0.50 24.45

data required time 24.45

--------------------------------------------------------------------------

data required time 24.45

data arrival time -21.23

--------------------------------------------------------------------------

slack (MET) 3.22

1

**Appendix B**

**Completed Verilog Source Codes and Testbenches**

**1. Conditional\_sum\_adder**

module conditional\_sum\_adder\_64(input [63:0] ope1,ope2,input Cin,add\_sub,reset,clock,start,

output [63:0] sum,output cout,complete,output reg overf);

wire c0,c1,c2,c3,c4,c5,c6;

wire com0,com1,com2,com3,com4,com5,com6,com7;

wire i=0;

always@(\*)

begin

if(complete)

begin

if(ope1[63]==ope2[63])

begin

if(ope1[63]==sum[63])

overf<=0;

else

overf<=1;

end

else

begin

if((cout && (!sum[63])) || ((!cout) && sum[63]))

overf<=0;

else overf<=1;

end

end

else

overf<=1'bx;

end

assign complete=(com0 &&com1 &&com2 &&com3 &&com4 &&com5 &&com6 &&com7 )? 1:0;

conditional\_sum\_adder\_8 CSA\_0( .A(ope1[7:0]),.B(ope2[7:0]),.Cin(i),.add\_sub(add\_sub),.reset(reset),.clock(clock),.start(start),

.sum(sum[7:0]),.cout(c0),.complete(com0));

conditional\_sum\_adder\_8\_1 CSA\_1( .A(ope1[15:8]),.B(ope2[15:8]),.Cin(c0),.add\_sub(add\_sub),.reset(reset),.clock(clock),.start(start),

.sum(sum[15:8]),.cout(c1),.complete(com1));

conditional\_sum\_adder\_8\_1 CSA\_2( .A(ope1[23:16]),.B(ope2[23:16]),.Cin(c1),.add\_sub(add\_sub),.reset(reset),.clock(clock),.start(start),

.sum(sum[23:16]),.cout(c2),.complete(com2));

conditional\_sum\_adder\_8\_1 CSA\_3( .A(ope1[31:24]),.B(ope2[31:24]),.Cin(c2),.add\_sub(add\_sub),.reset(reset),.clock(clock),.start(start),

.sum(sum[31:24]),.cout(c3),.complete(com3));

conditional\_sum\_adder\_8\_1 CSA\_4( .A(ope1[39:32]),.B(ope2[39:32]),.Cin(c3),.add\_sub(add\_sub),.reset(reset),.clock(clock),.start(start),

.sum(sum[39:32]),.cout(c4),.complete(com4));

conditional\_sum\_adder\_8\_1 CSA\_5( .A(ope1[47:40]),.B(ope2[47:40]),.Cin(c4),.add\_sub(add\_sub),.reset(reset),.clock(clock),.start(start),

.sum(sum[47:40]),.cout(c5),.complete(com5));

conditional\_sum\_adder\_8\_1 CSA\_6( .A(ope1[55:48]),.B(ope2[55:48]),.Cin(c5),.add\_sub(add\_sub),.reset(reset),.clock(clock),.start(start),

.sum(sum[55:48]),.cout(c6),.complete(com6));

conditional\_sum\_adder\_8\_1 CSA\_7( .A(ope1[63:56]),.B(ope2[63:56]),.Cin(c6),.add\_sub(add\_sub),.reset(reset),.clock(clock),.start(start),

.sum(sum[63:56]),.cout(cout),.complete(com7));

endmodule

module conditional\_sum\_adder(input [3:0] A,B,input add\_sub,reset,clock,start,

output reg [3:0] sum,output cout,output reg complete);

reg [3:0]A\_temp,B\_temp;

wire [3:0]sum\_temp;

wire temp;

always@(posedge clock)

begin

if(start && !reset)

begin

A\_temp<=A;

B\_temp<=B;

end

else

begin

A\_temp<=4'bx;

B\_temp<=4'bx;

end

end

always@(negedge clock)

begin

if(!reset && start)

sum<=sum\_temp;

else

sum<=4'bx;

end

always@(negedge clock)

begin

if(start)

complete<=0;

else

complete<=1'bx;

if(start&&((|sum\_temp) | (!sum\_temp)))

complete<=1;

else

complete<=0;

end

wire d0,d1,d2,d3;

wire c0,c1;

wire s1\_0,s2\_0,s3\_0,

s1\_1,s2\_1,s3\_1;

wire c1\_0,c2\_0,c3\_0,

c1\_1,c2\_1,c3\_1;

wire t2\_0,t2\_1;

wire z2\_0,z2\_1;

wire [1:0] y2\_0,y2\_1;

assign d0=B\_temp[0]^add\_sub;

assign d1=B\_temp[1]^add\_sub;

assign d2=B\_temp[2]^add\_sub;

assign d3=B\_temp[3]^add\_sub;

wire i=0;

wire j=1;

fulladder FA1( .A(A\_temp[0]),.B(d0),.Cin(add\_sub),.sum(sum\_temp[0]),.carry(c0));

//////////////////////1//////////////////////////////////////////////////

fulladder FA2\_0( .A(A\_temp[1]),.B(d1),.Cin(i),.sum(s1\_0),.carry(c1\_0));

fulladder FA2\_1( .A(A\_temp[1]),.B(d1),.Cin(j),.sum(s1\_1),.carry(c1\_1));

mux1 MUX2\_SUM(.one(s1\_1),.zero(s1\_0),.sel(c0),.out(sum\_temp[1]));

mux1 MUX2\_CARRY(.one(c1\_1),.zero(c1\_0),.sel(c0),.out(c1));

///////////////////////2/////////////////////////////////////////////////

fulladder FA3\_0( .A(A\_temp[2]),.B(d2),.Cin(i),.sum(s2\_0),.carry(c2\_0));

fulladder FA3\_1( .A(A\_temp[2]),.B(d2),.Cin(j),.sum(s2\_1),.carry(c2\_1));

////////////////////////3////////////////////////////////////////////////

fulladder FA4\_0( .A(A\_temp[3]),.B(d3),.Cin(i),.sum(s3\_0),.carry(c3\_0));

fulladder FA4\_1( .A(A\_temp[3]),.B(d3),.Cin(j),.sum(s3\_1),.carry(c3\_1));

mux1 MUX3\_SUM\_0(.one(s3\_1),.zero(s3\_0),.sel(c2\_0),.out(t2\_0));

mux1 MUX3\_SUM\_1(.one(s3\_1),.zero(s3\_0),.sel(c2\_1),.out(t2\_1));

assign y2\_0={t2\_0,s2\_0};

assign y2\_1={t2\_1,s2\_1};

mux2 MUX8(.one(y2\_1),.zero(y2\_0),.sel(c1),.out(sum\_temp[3:2]));

mux1 MUX3\_CARRY\_0(.one(c3\_1),.zero(c3\_0),.sel(c2\_0),.out(z2\_0));

mux1 MUX3\_CARRY\_1(.one(c3\_1),.zero(c3\_0),.sel(c2\_1),.out(z2\_1));

mux1 MUX3\_CARRY\_2(.one(z2\_1),.zero(z2\_0),.sel(c1),.out(cout));

endmodule

module mux1(input one,zero,sel,output out);

assign out=(sel)?one:zero;

endmodule

module mux2(input [1:0]one,zero,input sel,output [1:0]out);

assign out=(sel)?one:zero;

endmodule

module conditional\_sum\_adder\_1(input [3:0] A,B,input Cin, add\_sub,reset,clock,start,

output reg [3:0] sum,output cout,output reg complete);

reg [3:0]A\_temp,B\_temp;

wire [3:0]sum\_temp;

wire temp;

always@(posedge clock)

begin

if(start && !reset)

begin

A\_temp<=A;

B\_temp<=B;

end

else

begin

A\_temp<=4'bx;

B\_temp<=4'bx;

end

end

always@(negedge clock)

begin

if(!reset && start)

sum<=sum\_temp;

else

sum<=4'bx;

end

always@(negedge clock)

begin

if(start)

complete<=0;

else

complete<=1'bx;

if(start&&((|sum\_temp) | (!sum\_temp)))

complete<=1;

else

complete<=0;

end

wire d0,d1,d2,d3;

wire c0,c1;

wire s1\_0,s2\_0,s3\_0,

s1\_1,s2\_1,s3\_1;

wire c1\_0,c2\_0,

c1\_1,c2\_1;

wire t2\_0,t2\_1;

wire z2\_0,z2\_1;

wire [1:0] y2\_0,y2\_1;

assign d0=B\_temp[0]^add\_sub;

assign d1=B\_temp[1]^add\_sub;

assign d2=B\_temp[2]^add\_sub;

assign d3=B\_temp[3]^add\_sub;

wire i=0;

wire j=1;

fulladder FA1( .A(A\_temp[0]),.B(d0),.Cin(Cin),.sum(sum\_temp[0]),.carry(c0));

//////////////////////1//////////////////////////////////////////////////

fulladder FA2\_0( .A(A\_temp[1]),.B(d1),.Cin(i),.sum(s1\_0),.carry(c1\_0));

fulladder FA2\_1( .A(A\_temp[1]),.B(d1),.Cin(j),.sum(s1\_1),.carry(c1\_1));

mux1 MUX2\_SUM(.one(s1\_1),.zero(s1\_0),.sel(c0),.out(sum\_temp[1]));

mux1 MUX2\_CARRY(.one(c1\_1),.zero(c1\_0),.sel(c0),.out(c1));

///////////////////////2/////////////////////////////////////////////////

fulladder FA3\_0( .A(A\_temp[2]),.B(d2),.Cin(i),.sum(s2\_0),.carry(c2\_0));

fulladder FA3\_1( .A(A\_temp[2]),.B(d2),.Cin(j),.sum(s2\_1),.carry(c2\_1));

////////////////////////3////////////////////////////////////////////////

fulladder FA4\_0( .A(A\_temp[3]),.B(d3),.Cin(i),.sum(s3\_0),.carry(c3\_0));

fulladder FA4\_1( .A(A\_temp[3]),.B(d3),.Cin(j),.sum(s3\_1),.carry(c3\_1));

mux1 MUX3\_SUM\_0(.one(s3\_1),.zero(s3\_0),.sel(c2\_0),.out(t2\_0));

mux1 MUX3\_SUM\_1(.one(s3\_1),.zero(s3\_0),.sel(c2\_1),.out(t2\_1));

assign y2\_0={t2\_0,s2\_0};

assign y2\_1={t2\_1,s2\_1};

mux2 MUX8(.one(y2\_1),.zero(y2\_0),.sel(c1),.out(sum\_temp[3:2]));

mux1 MUX3\_CARRY\_0(.one(c3\_1),.zero(c3\_0),.sel(c2\_0),.out(z2\_0));

mux1 MUX3\_CARRY\_1(.one(c3\_1),.zero(c3\_0),.sel(c2\_1),.out(z2\_1));

mux1 MUX3\_CARRY\_2(.one(z2\_1),.zero(z2\_0),.sel(c1),.out(cout));

endmodule

module conditional\_sum\_adder\_8(input [7:0] A,B,input Cin,add\_sub,reset,clock,start,

output [7:0] sum,output cout,complete);

wire c1;

wire com1,com2;

conditional\_sum\_adder CSA\_4\_1( .A(A[3:0]),.B(B[3:0]),.add\_sub(add\_sub),.reset(reset),.clock(clock),.start(start),

.sum(sum[3:0]), .cout(c1),.complete(com1));

conditional\_sum\_adder\_1 CSA\_4\_2( .A(A[7:4]),.B(B[7:4]),.add\_sub(add\_sub),.Cin(c1),.reset(reset),.clock(clock),.start(start),

.sum(sum[7:4]), .cout(cout),.complete(com2));

assign complete=(com1 && com2)?1:0;

endmodule

module conditional\_sum\_adder\_8\_1(input [7:0] A,B,input Cin,add\_sub,reset,clock,start,

output [7:0] sum,output cout,complete);

wire c1;

wire com1,com2;

conditional\_sum\_adder\_1 CSA\_4\_1( .A(A[3:0]),.B(B[3:0]),.add\_sub(add\_sub),.Cin(Cin),.reset(reset),.clock(clock),.start(start),

.sum(sum[3:0]), .cout(c1),.complete(com1));

conditional\_sum\_adder\_1 CSA\_4\_2( .A(A[7:4]),.B(B[7:4]),.add\_sub(add\_sub),.Cin(c1),.reset(reset),.clock(clock),.start(start),

.sum(sum[7:4]), .cout(cout),.complete(com2));

assign complete=(com1 && com2)?1:0;

endmodule

module fulladder(input A,B,Cin,output reg sum,carry);

always@(\*)

begin

case({A,B,Cin})

3'b000:begin

sum=0;carry=0;

end

3'b001:begin

sum=1;carry=0;

end

3'b010:begin

sum=1;carry=0;

end

3'b011:begin

sum=0;carry=1;

end

3'b100:begin

sum=1;carry=0;

end

3'b101:begin

sum=0;carry=1;

end

3'b110:begin

sum=0;carry=1;

end

3'b111:begin

sum=1;carry=1;

end

endcase

end

endmodule

TESTBENCH

`timescale 1ns / 10ps

module conditionalsum\_64();

wire [63:0]sum;

wire cout,complete,overf;

reg [63:0]ope1,ope2;

reg add\_sub,start,reset,clock;

conditional\_sum\_adder\_64 DUT (ope1,ope2,Cin,add\_sub,reset,clock,start,

sum,cout,complete,overf);

always

begin

clock=1'b0;

#10;clock=1'b1;

#10;

end

initial

$monitor($time,,,"ope1=%h,ope2=%h,add\_sub=%h,sum=%h,overf=%h,complete=%h",ope1,ope2,add\_sub,sum,overf,complete);

initial

begin

$dumpfile("CSA\_64.vcd");

$dumpvars(0);

@(negedge clock);reset=1;start=0;

@(negedge clock);reset=0;start=1;

ope1=64'h4653220123456465;

ope2=64'h7987893515646136;

add\_sub=0;

/\* @(negedge clock);reset=0;start=1;

ope1=64'h00000000000000ff;

ope2=64'h0000000000000123;

add\_sub=0;

\*/

/\* @(negedge clock);reset=0;start=1;

ope1=64'h4895130123456465;

ope2=64'h7987898888846136;

add\_sub=1;

\*/

/\* @(negedge clock);reset=0;start=1;

ope1=64'h9;

ope2=64'h6;

add\_sub=1;

\*/

#40;$finish;

end

endmodule

**2. Carry\_select\_adder**

**//`timescale 1ns / 10ps**

**module top\_CSA (input [63:0]ope1,ope2,input add\_sub,clock,reset,start,i,j,**

**output [63:0]sum,output complete,cout, output reg overf);**

**wire complete\_temp0,complete\_temp1,complete\_temp2,complete\_temp3,**

**complete\_temp4,complete\_temp5,complete\_temp6,complete\_temp7,**

**complete\_temp1\_0,complete\_temp1\_1, complete\_temp2\_0,complete\_temp2\_1,**

**complete\_temp3\_0,complete\_temp3\_1, complete\_temp4\_0,complete\_temp4\_1,**

**complete\_temp5\_0,complete\_temp5\_1, complete\_temp6\_0,complete\_temp6\_1,**

**complete\_temp7\_0,complete\_temp7\_1;**

**wire w0,**

**w1\_0,w1\_1,**

**w2\_0,w2\_1,**

**w3\_0,w3\_1,**

**w4\_0,w4\_1,**

**w5\_0,w5\_1,**

**w6\_0,w6\_1,**

**w7\_0,w7\_1;**

**wire d1,d2,d3,d4,d5,d6;**

**wire [7:0] sum1\_0,sum1\_1,**

**sum2\_0,sum2\_1,**

**sum3\_0,sum3\_1,**

**sum4\_0,sum4\_1,**

**sum5\_0,sum5\_1,**

**sum6\_0,sum6\_1,**

**sum7\_0,sum7\_1;**

**//wire [7:0] sum1,sum2,sum3,sum4,sum5,sum6;**

**assign complete\_temp1=complete\_temp1\_0|complete\_temp1\_1;**

**assign complete\_temp2=complete\_temp2\_0|complete\_temp2\_1;**

**assign complete\_temp3=complete\_temp3\_0|complete\_temp3\_1;**

**assign complete\_temp4=complete\_temp4\_0|complete\_temp4\_1;**

**assign complete\_temp5=complete\_temp5\_0|complete\_temp5\_1;**

**assign complete\_temp6=complete\_temp6\_0|complete\_temp6\_1;**

**assign complete\_temp7=complete\_temp7\_0|complete\_temp7\_1;**

**assign complete=(complete\_temp0 & complete\_temp1 & complete\_temp2 & complete\_temp3 &**

**complete\_temp4 & complete\_temp5 & complete\_temp6 & complete\_temp7)?1'b1:1'b0;**

**ripple\_carry\_adder RCA0(.a(ope1[7:0]),.b(ope2[7:0]),.add\_sub(add\_sub),.clock(clock),.reset(reset),**

**.start(start),.sum(sum[7:0]),.cout(w0),.complete(complete\_temp0));**

**ripple\_carry\_adder\_zero RCA1\_0(.a(ope1[15:8]),.b(ope2[15:8]),.add\_sub(add\_sub),.clock(clock),.reset(reset),**

**.start(start),.sum(sum1\_0),.cout(w1\_0),.complete(complete\_temp1\_0),.d(i));**

**ripple\_carry\_adder\_one RCA1\_1(.a(ope1[15:8]),.b(ope2[15:8]),.add\_sub(add\_sub),.clock(clock),.reset(reset),**

**.start(start),.sum(sum1\_1),.cout(w1\_1),.complete(complete\_temp1\_1),.d(j));**

**mux\_8 MUX1\_8(.in0\_8(sum1\_0),.in1\_8(sum1\_1),.sel\_8(w0),.out\_8(sum[15:8]));**

**mux\_1 MUX1\_1(.in0\_1(w1\_0),.in1\_1(w1\_1),.sel\_1(w0),.out\_1(d1));**

**/////////////////////////////////////1/////////////////////////////////////////////////////////**

**ripple\_carry\_adder\_zero RCA2\_0(.a(ope1[23:16]),.b(ope2[23:16]),.add\_sub(add\_sub),.clock(clock),.reset(reset),**

**.start(start),.sum(sum2\_0),.cout(w2\_0),.complete(complete\_temp2\_0),.d(i));**

**ripple\_carry\_adder\_one RCA2\_1(.a(ope1[15:8]),.b(ope2[15:8]),.add\_sub(add\_sub),.clock(clock),.reset(reset),**

**.start(start),.sum(sum2\_1),.cout(w2\_1),.complete(complete\_temp2\_1),.d(j));**

**mux\_8 MUX2\_8(.in0\_8(sum2\_0),.in1\_8(sum2\_1),.sel\_8(d1),.out\_8(sum[23:16]));**

**mux\_1 MUX2\_1(.in0\_1(w2\_0),.in1\_1(w2\_1),.sel\_1(d1),.out\_1(d2));**

**///////////////////////////////////2////////////////////////////////////////////////////////////**

**ripple\_carry\_adder\_zero RCA3\_0(.a(ope1[31:24]),.b(ope2[31:24]),.add\_sub(add\_sub),.clock(clock),.reset(reset),**

**.start(start),.sum(sum3\_0),.cout(w3\_0),.complete(complete\_temp3\_0),.d(i));**

**ripple\_carry\_adder\_one RCA3\_1(.a(ope1[31:24]),.b(ope2[31:24]),.add\_sub(add\_sub),.clock(clock),.reset(reset),**

**.start(start),.sum(sum3\_1),.cout(w3\_1),.complete(complete\_temp3\_1),.d(j));**

**mux\_8 MUX3\_8(.in0\_8(sum3\_0),.in1\_8(sum3\_1),.sel\_8(d2),.out\_8(sum[31:24]));**

**mux\_1 MUX3\_1(.in0\_1(w3\_0),.in1\_1(w3\_1),.sel\_1(d2),.out\_1(d3));**

**///////////////////////////////////3/////////////////////////////////////////////////////////////**

**ripple\_carry\_adder\_zero RCA4\_0(.a(ope1[39:32]),.b(ope2[39:32]),.add\_sub(add\_sub),.clock(clock),.reset(reset),**

**.start(start),.sum(sum4\_0),.cout(w4\_0),.complete(complete\_temp4\_0),.d(i));**

**ripple\_carry\_adder\_one RCA4\_1(.a(ope1[39:32]),.b(ope2[39:32]),.add\_sub(add\_sub),.clock(clock),.reset(reset),**

**.start(start),.sum(sum4\_1),.cout(w4\_1),.complete(complete\_temp4\_1),.d(j));**

**mux\_8 MUX4\_8(.in0\_8(sum4\_0),.in1\_8(sum4\_1),.sel\_8(d3),.out\_8(sum[39:32]));**

**mux\_1 MUX4\_1(.in0\_1(w4\_0),.in1\_1(w4\_1),.sel\_1(d3),.out\_1(d4));**

**//////////////////////////////////4////////////////////////////////////////////////////////////////**

**ripple\_carry\_adder\_zero RCA5\_0(.a(ope1[47:40]),.b(ope2[47:40]),.add\_sub(add\_sub),.clock(clock),.reset(reset),**

**.start(start),.sum(sum5\_0),.cout(w5\_0),.complete(complete\_temp5\_0),.d(i));**

**ripple\_carry\_adder\_one RCA5\_1(.a(ope1[47:40]),.b(ope2[47:40]),.add\_sub(add\_sub),.clock(clock),.reset(reset),**

**.start(start),.sum(sum5\_1),.cout(w5\_1),.complete(complete\_temp5\_1),.d(j));**

**mux\_8 MUX5\_8(.in0\_8(sum5\_0),.in1\_8(sum5\_1),.sel\_8(d4),.out\_8(sum[47:40]));**

**mux\_1 MUX5\_1(.in0\_1(w5\_0),.in1\_1(w5\_1),.sel\_1(d4),.out\_1(d5));**

**//////////////////////////////////////6///////////////////////////////////////////////////////////**

**ripple\_carry\_adder\_zero RCA6\_0(.a(ope1[55:48]),.b(ope2[55:48]),.add\_sub(add\_sub),.clock(clock),.reset(reset),**

**.start(start),.sum(sum6\_0),.cout(w6\_0),.complete(complete\_temp6\_0),.d(i));**

**ripple\_carry\_adder\_one RCA6\_1(.a(ope1[55:48]),.b(ope2[55:48]),.add\_sub(add\_sub),.clock(clock),.reset(reset),**

**.start(start),.sum(sum6\_1),.cout(w6\_1),.complete(complete\_temp6\_1),.d(j));**

**mux\_8 MUX6\_8(.in0\_8(sum6\_0),.in1\_8(sum6\_1),.sel\_8(d5),.out\_8(sum[55:48]));**

**mux\_1 MUX6\_1(.in0\_1(w6\_0),.in1\_1(w6\_1),.sel\_1(d5),.out\_1(d6));**

**/////////////////////////////////////7////////////////////////////////////////////////////////////**

**ripple\_carry\_adder\_zero RCA7\_0(.a(ope1[63:56]),.b(ope2[63:56]),.add\_sub(add\_sub),.clock(clock),.reset(reset),**

**.start(start),.sum(sum7\_0),.cout(w7\_0),.complete(complete\_temp7\_0),.d(i));**

**ripple\_carry\_adder\_one RCA7\_1(.a(ope1[63:56]),.b(ope2[63:56]),.add\_sub(add\_sub),.clock(clock),.reset(reset),**

**.start(start),.sum(sum7\_1),.cout(w7\_1),.complete(complete\_temp7\_1),.d(j));**

**mux\_8 MUX7\_8(.in0\_8(sum7\_0),.in1\_8(sum7\_1),.sel\_8(d6),.out\_8(sum[63:56]));**

**mux\_1 MUX7\_1(.in0\_1(w7\_0),.in1\_1(w7\_1),.sel\_1(d6),.out\_1(cout));**

**always@(\*)**

**begin**

**/\* if(start)**

**begin**

**if(sum<=((2\*\*63)-1) || sum<=(2\*\*63) )**

**overf<=0;**

**else**

**overf<=1;**

**end**

**else**

**overf<=1'bx;\*/**

**if(complete)**

**begin**

**if(ope1[63]==ope2[63])**

**begin**

**if(ope1[63]==sum[63])**

**overf<=0;**

**else**

**overf<=1;**

**end**

**else**

**begin**

**if((cout && (!sum[63])) || ((!cout) && sum[63]))**

**overf<=0;**

**else overf<=1;**

**end**

**end**

**else**

**overf<=1'bx;**

**end**

**endmodule**

**module ripple\_carry\_adder(input [7:0] a,b,input add\_sub,clock,start,reset, output reg [7:0] sum,output cout,**

**output reg complete);**

**reg [7:0] A\_temp,B\_temp;**

**wire [7:0]sum\_temp;**

**wire cout0,cout1,cout2,cout3,cout4,cout5,cout6;**

**wire d0,d1,d2,d3,d4,d5,d6,d7;**

**assign d0=B\_temp[0]^add\_sub;**

**assign d1=B\_temp[1]^add\_sub;**

**assign d2=B\_temp[2]^add\_sub;**

**assign d3=B\_temp[3]^add\_sub;**

**assign d4=B\_temp[4]^add\_sub;**

**assign d5=B\_temp[5]^add\_sub;**

**assign d6=B\_temp[6]^add\_sub;**

**assign d7=B\_temp[7]^add\_sub;**

**fulladder FA0(.A(A\_temp[0]),.B(d0),.Cin(add\_sub),.sum(sum\_temp[0]),.carry(cout0));**

**fulladder FA1(.A(A\_temp[1]),.B(d1),.Cin(cout0),.sum(sum\_temp[1]),.carry(cout1));**

**fulladder FA2(.A(A\_temp[2]),.B(d2),.Cin(cout1),.sum(sum\_temp[2]),.carry(cout2));**

**fulladder FA3(.A(A\_temp[3]),.B(d3),.Cin(cout2),.sum(sum\_temp[3]),.carry(cout3));**

**fulladder FA4(.A(A\_temp[4]),.B(d4),.Cin(cout3),.sum(sum\_temp[4]),.carry(cout4));**

**fulladder FA5(.A(A\_temp[5]),.B(d5),.Cin(cout4),.sum(sum\_temp[5]),.carry(cout5));**

**fulladder FA6(.A(A\_temp[6]),.B(d6),.Cin(cout5),.sum(sum\_temp[6]),.carry(cout6));**

**fulladder FA7(.A(A\_temp[7]),.B(d7),.Cin(cout6),.sum(sum\_temp[7]),.carry(cout));**

**always@(posedge clock) //inputs will be sampled only when start is high and reset is low**

**begin**

**if(start && !reset)**

**begin**

**A\_temp<=a;**

**B\_temp<=b;**

**// complete<=0;**

**end**

**else**

**begin**

**A\_temp<=8'bx;**

**B\_temp<=8'bx;**

**// complete<=1'bx;**

**end**

**end**

**always@(negedge clock )**

**begin**

**sum<=sum\_temp;**

**end**

**always@(negedge clock)**

**begin**

**if((|sum\_temp) | (!sum\_temp)) //complete signal is made high by reduction operation on sum**

**complete<=1'b1;**

**else**

**complete<=1'b0;**

**end**

**endmodule**

**module ripple\_carry\_adder\_zero(input [7:0] a,b,input add\_sub,clock,start,reset,d, output reg [7:0] sum,output cout,**

**output reg complete);**

**reg [7:0] A\_temp,B\_temp;**

**wire [7:0]sum\_temp;**

**wire cout0,cout1,cout2,cout3,cout4,cout5,cout6;**

**wire d0,d1,d2,d3,d4,d5,d6,d7;**

**assign d0=B\_temp[0]^add\_sub;**

**assign d1=B\_temp[1]^add\_sub;**

**assign d2=B\_temp[2]^add\_sub;**

**assign d3=B\_temp[3]^add\_sub;**

**assign d4=B\_temp[4]^add\_sub;**

**assign d5=B\_temp[5]^add\_sub;**

**assign d6=B\_temp[6]^add\_sub;**

**assign d7=B\_temp[7]^add\_sub;**

**fulladder FA0(.A(A\_temp[0]),.B(d0),.Cin(d),.sum(sum\_temp[0]),.carry(cout0));**

**fulladder FA1(.A(A\_temp[1]),.B(d1),.Cin(cout0),.sum(sum\_temp[1]),.carry(cout1));**

**fulladder FA2(.A(A\_temp[2]),.B(d2),.Cin(cout1),.sum(sum\_temp[2]),.carry(cout2));**

**fulladder FA3(.A(A\_temp[3]),.B(d3),.Cin(cout2),.sum(sum\_temp[3]),.carry(cout3));**

**fulladder FA4(.A(A\_temp[4]),.B(d4),.Cin(cout3),.sum(sum\_temp[4]),.carry(cout4));**

**fulladder FA5(.A(A\_temp[5]),.B(d5),.Cin(cout4),.sum(sum\_temp[5]),.carry(cout5));**

**fulladder FA6(.A(A\_temp[6]),.B(d6),.Cin(cout5),.sum(sum\_temp[6]),.carry(cout6));**

**fulladder FA7(.A(A\_temp[7]),.B(d7),.Cin(cout6),.sum(sum\_temp[7]),.carry(cout));**

**always@(posedge clock) //inputs will be sampled only when start is high and reset is low**

**begin**

**if(start && !reset)**

**begin**

**A\_temp<=a;**

**B\_temp<=b;**

**// complete<=0;**

**end**

**else**

**begin**

**A\_temp<=8'bx;**

**B\_temp<=8'bx;**

**// complete<=1'bx;**

**end**

**end**

**always@(negedge clock )**

**begin**

**sum<=sum\_temp;**

**end**

**always@(negedge clock)**

**begin**

**if((|sum\_temp) | (!sum\_temp)) //complete signal is made high by reduction operation on sum\_temp**

**complete<=1'b1;**

**else**

**complete<=1'b0;**

**end**

**endmodule**

**module ripple\_carry\_adder\_one(input [7:0] a,b,input add\_sub,clock,start,reset,d, output reg [7:0] sum,output cout,**

**output reg complete);**

**reg [7:0] A\_temp,B\_temp;**

**wire [7:0]sum\_temp;**

**wire cout0,cout1,cout2,cout3,cout4,cout5,cout6;**

**wire d0,d1,d2,d3,d4,d5,d6,d7;**

**assign d0=B\_temp[0]^add\_sub;**

**assign d1=B\_temp[1]^add\_sub;**

**assign d2=B\_temp[2]^add\_sub;**

**assign d3=B\_temp[3]^add\_sub;**

**assign d4=B\_temp[4]^add\_sub;**

**assign d5=B\_temp[5]^add\_sub;**

**assign d6=B\_temp[6]^add\_sub;**

**assign d7=B\_temp[7]^add\_sub;**

**fulladder FA0(.A(A\_temp[0]),.B(d0),.Cin(d),.sum(sum\_temp[0]),.carry(cout0));**

**fulladder FA1(.A(A\_temp[1]),.B(d1),.Cin(cout0),.sum(sum\_temp[1]),.carry(cout1));**

**fulladder FA2(.A(A\_temp[2]),.B(d2),.Cin(cout1),.sum(sum\_temp[2]),.carry(cout2));**

**fulladder FA3(.A(A\_temp[3]),.B(d3),.Cin(cout2),.sum(sum\_temp[3]),.carry(cout3));**

**fulladder FA4(.A(A\_temp[4]),.B(d4),.Cin(cout3),.sum(sum\_temp[4]),.carry(cout4));**

**fulladder FA5(.A(A\_temp[5]),.B(d5),.Cin(cout4),.sum(sum\_temp[5]),.carry(cout5));**

**fulladder FA6(.A(A\_temp[6]),.B(d6),.Cin(cout5),.sum(sum\_temp[6]),.carry(cout6));**

**fulladder FA7(.A(A\_temp[7]),.B(d7),.Cin(cout6),.sum(sum\_temp[7]),.carry(cout));**

**always@(posedge clock) //inputs will be sampled only when start is high and reset is low**

**begin**

**if(start && !reset)**

**begin**

**A\_temp<=a;**

**B\_temp<=b;**

**// complete<=0;**

**end**

**else**

**begin**

**A\_temp<=8'bx;**

**B\_temp<=8'bx;**

**// complete<=1'bx;**

**end**

**end**

**always@(negedge clock )**

**begin**

**sum<=sum\_temp;**

**end**

**always@(negedge clock)**

**begin**

**if((|sum\_temp) | (!sum\_temp)) //complete signal is made high by reduction operation on sum\_temp**

**complete<=1'b1;**

**else**

**complete<=1'b0;**

**end**

**endmodule**

**module mux\_8(input [7:0]in0\_8,in1\_8,input sel\_8,output [7:0]out\_8);**

**assign out\_8=(sel\_8)? in1\_8:in0\_8;**

**endmodule**

**module mux\_1(input in0\_1,in1\_1,input sel\_1,output out\_1);**

**assign out\_1=(sel\_1)?in1\_1:in0\_1;**

**endmodule**

**module fulladder(input A,B,Cin,output reg sum,carry);**

**always@(\*)**

**begin**

**case({A,B,Cin})**

**3'b000:begin**

**sum=0;carry=0;**

**end**

**3'b001:begin**

**sum=1;carry=0;**

**end**

**3'b010:begin**

**sum=1;carry=0;**

**end**

**3'b011:begin**

**sum=0;carry=1;**

**end**

**3'b100:begin**

**sum=1;carry=0;**

**end**

**3'b101:begin**

**sum=0;carry=1;**

**end**

**3'b110:begin**

**sum=0;carry=1;**

**end**

**3'b111:begin**

**sum=1;carry=1;**

**end**

**endcase**

**end**

**endmodule**

**TESTBENCH**

**`timescale 1ns/10ps**

**module top\_CSA\_tb();**

**reg [63:0] ope1,ope2;**

**reg clock,start,add\_sub,reset,i,j;**

**wire [63:0] sum;**

**wire cout,overf,complete;**

**top\_CSA DUT (ope1,ope2,add\_sub,clock,reset,start,i,j,**

**sum,complete,cout,overf);**

**always**

**begin**

**clock=1'b0;**

**#10;clock=1'b1;**

**#10;**

**end**

**initial**

**$monitor($time,,,"ope1=%h,ope2=%h,add\_sub=%h,sum=%h,overf=%h,complete=%h",ope1,ope2,add\_sub,sum,overf,complete);**

**initial**

**begin**

**$dumpfile("CSA\_64.vcd");**

**$dumpvars(0);**

**i=0;**

**j=1;**

**/\* @(negedge clock);reset=1;start=0;**

**@(negedge clock);reset=0;start=1;**

**ope1=64'h4653220123456465;**

**ope2=64'h7987893515646136;**

**add\_sub=0;**

**\*/**

**/\* @(negedge clock);reset=0;start=1;**

**ope1=64'h00000000000000ff;**

**ope2=64'h0000000000000123;**

**add\_sub=0;**

**\*/**

**/\* @(negedge clock);reset=0;start=1;**

**ope1=64'h4895130123456465;**

**ope2=64'h7987898888846136;**

**add\_sub=1;**

**i=0;**

**j=1;**

**\*/**

**@(negedge clock);reset=0;start=1;**

**ope1=64'h9;**

**ope2=64'h6;**

**add\_sub=1;**

**#40;$finish;**

**end**

**endmodule**