UCD School of Electrical and
Electronic Engineering

EEEN30190 Digital System Design

Calculator Design Report

Name: Jingyi Hu **Student Number: 17200558**

Working with: Philip Snell **Team: 3**

I certify that ALL of the following are true:

1. I have read the *UCD Plagiarism Policy* and the *College of Engineering and Architecture Plagiarism Protocol*. (These documents are available on Blackboard, under Assessment.)
2. I understand fully the definition of plagiarism and the consequences of plagiarism as discussed in the documents above.
3. I recognise that any work that has been plagiarised (in whole or in part) may be subject to penalties, as outlined in the documents above.
4. I have not previously submitted this work, or any version of it, for assessment in any other module in this University, or any other institution.
5. I have not plagiarised any part of this report. The work described was done by the team, but this report is all my own original work, except where otherwise acknowledged in the report.

Signed: _____Jingyi Hu________________ **Date:** 4 December 2017

Calculator Functionality

An algebraic calculator has been designed, that can do addition, subtraction, multiplication using positive integer numbers and have the function of store and recall the value and clear function as well. At the same time, four digits are in the display, but negative numbers cannot be displayed on the interface.

Addition, subtraction and multiplication can be done by this calculator. Once one of these keys (+, -, x) are pressed, the four digits in the display are changed to 0000. Furthermore, after the equal key is pressed, the results will illustrate in the display and can be also calculated as the first operand in the next calculation.

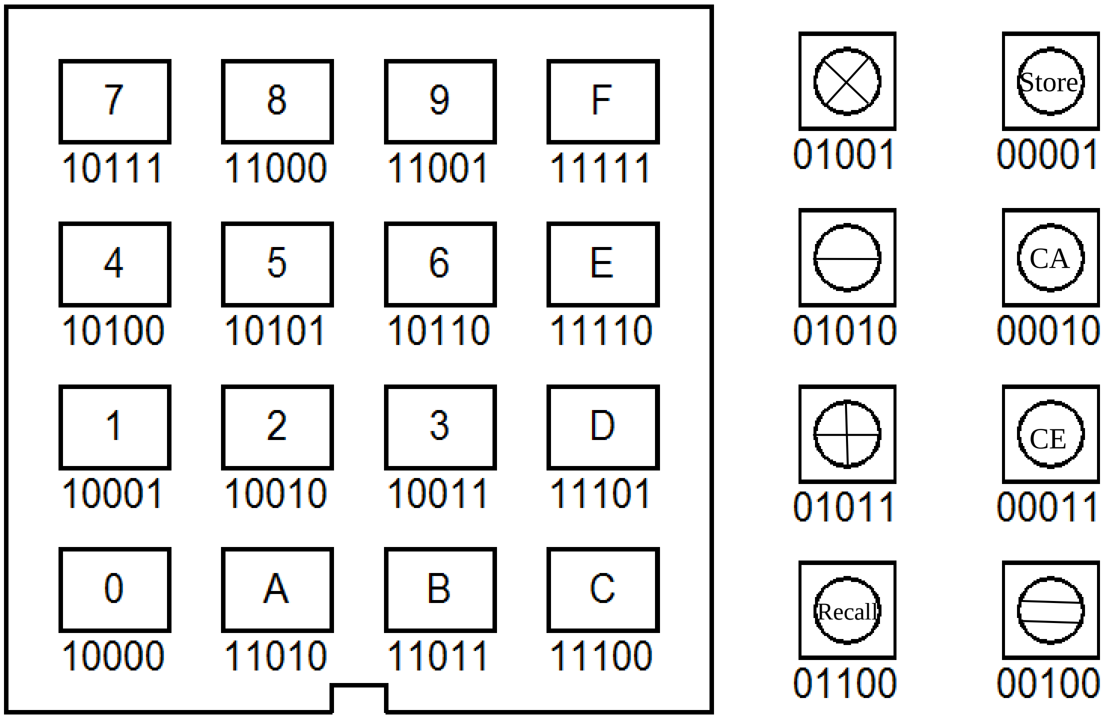
Store and Recall functions are implemented in this calculator by keys. The store key is used to copy the four digits in the display to the memory while clear the display. And the Recall key let the value in the memory displayed on the LEDs lights.

Two clear keys, besides, are provided to achieve clear functions. One is CA (clear all) similar to reset, but the memory of calculator is not affected. Once the CA button was pressed, four digits in the display are 0000. And then if the Recall button is pressed, the four digits which stored into the memory previously will be displayed. Another clear key is CE (clear entry) which only applied to clear the four digits in the display.

Additionally, before starting a new calculation, the value displayed is not used in the next calculation and the first operand is not four digits number, the user have to press CA first. Besides, this calculator only can calculate two operands each time. Therefore, the user need to press the key obey the operator precedence and equal key to get a results. For example, sequence 1 + 2 * 3 = , the user would press 2 * 3 = + 1 = ; and sequence 2 + 3 + 4 = , the keys should be pressed are 2 + 3 = + 4 = . When the result more than FFFF or less than 0000, there is no overflow warning and no negative number would be displayed as well.

Key Assignment

Explain which key you used for each function. Modify the diagram below if you wish, otherwise delete it.



Calculator Hardware

The calculator core logic module is composed with four registers with synchronous reset and combination logics:

X register: x signal is 16 bits wide, which always connected to the display to show the digits entered. The input port of X register is connected to the output port of 5-to-1 multiplexer. And the previous digits are pushed to the left by the latest digit in order to change their value.

1) If the any digits key is pressed while new_key = 1, the rightmost 4 bits of keycode signal will be stored in the rightmost 4 bits of next_x, and the leftmost 12 bits of next_x will be made up of the rightmost 12 bits of x signal. Therefore, the new value entered can be shifted onto the right of the display;
2) If the = key is pressed while the new_key = 1, the result will be input to the X register;
3) If the Recall key is pressed and new_key = 1, the value in the Memory register will be copied into X register;
4) If any other key is pressed and new_key = 1, 16 bits 0 will be input to the X register;
5) Otherwise, X register will hold the value.

Y register: used to store the first operand of the operation. The input port of Y register is connected to the output port of multiplexer.

1) If one of operation buttons is pressed and new_key = 1, the value stored in the X register will be copies to Y register;
2) If CA is pressed and new_key = 1, 16 bits 0 will be input to the Y register;
3) Otherwise, Y register will hold the value.

Op register: used to store the rightmost 2 bits of keycode of operation buttons. The input port of Op register is connected to the output port of multiplexer.

1) If one of the operation key is pressed and new_key = 1,the rightmost 2 bits of keycode of the operation will be given to the op signal;
2) But if CA is pressed and new_key=1, Op register will be cleared;
3) Otherwise, Op register will hold the value.

Memory register: used to store the value in the X register. The input port of Memory register is connected to the output port of multiplexer.

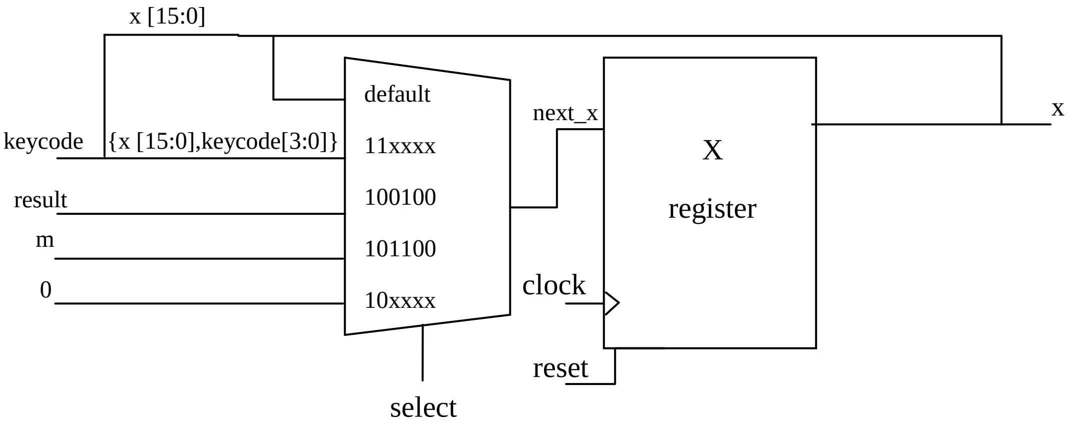
1) If the Store key is pressed and new_key=1, the value in the X register is copied to Memory register;
2) Otherwise, memory register will hold the value.

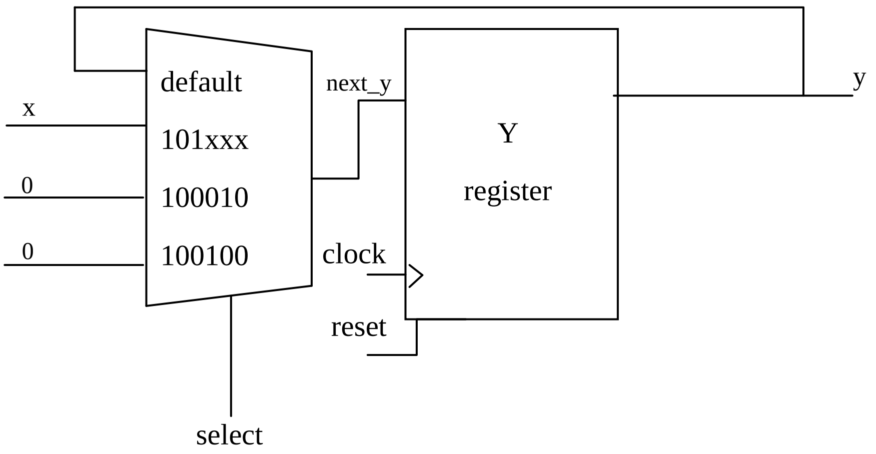
Operation combination logics: used to get result by multiplexer. First get the results of addition, subtraction and multiplication, and then give one of result to result signal selected by the op signal.

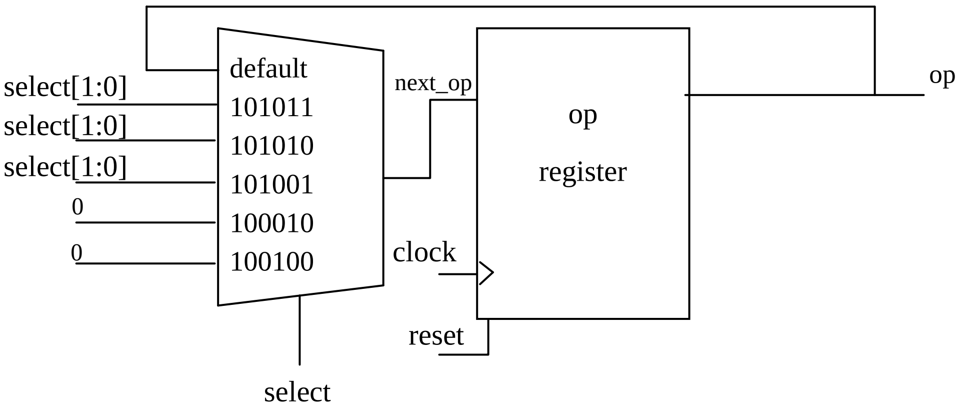
1) op = 11, the result is from addition;
2) op = 01, the result is from multiplication;
3) op = 10, the result is from subtraction;
4) Otherwise, the result is the value in the X register.

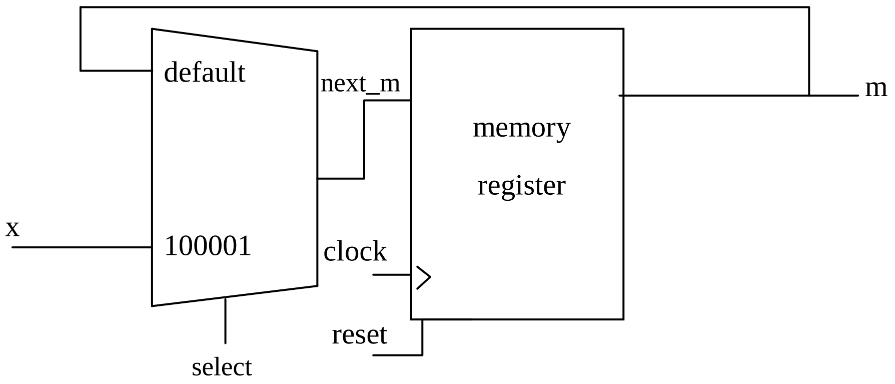
In addition, if more than one key is pressed, the system will ignore those signal and all the registers will hold the current value.

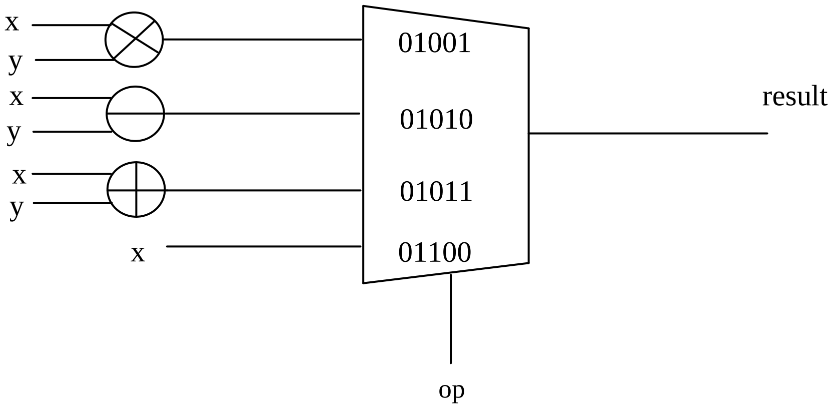
RTL Diagram


x [15:0]
keycode
{x [15:0],keycode[3:0]}
result
m
0
default
11xxxx
100100
101100
10xxxx
select
next_x
clock
reset
X
register
x


x
0
0
default
101xxx
100010
100100
select
next_y
clock
reset
Y
register
y


select[1:0]
select[1:0]
select[1:0]
0
0
default
101011
101010
101001
100010
100100
select
next_op
clock
reset
op
register
op





(All the RTL diagrams are created by Jingyi Hu and Philip Snell.)

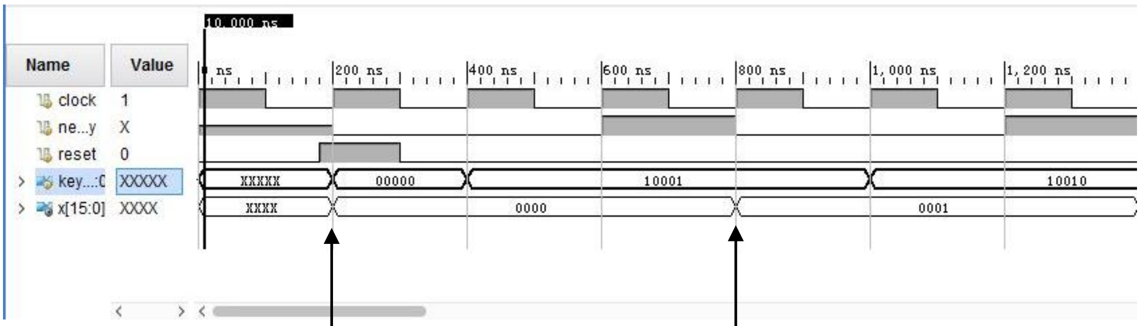
Verification

In order to verify this calculator's behaviour is correct, the whole functions of the calculator have been verified.

1) The x signal whether changes to 0000_{HEX} after reset;
2) The new digit whether pushes the previous digits to the left;
3) The x signal whether changes to 0000, once the + key is pressed;
4) Check the result correct or not, whether is in the display, after entre another operand and press the = key;
5) Check the result whether is available for next calculation;
6) Repeat 3) – 5) twice, but pressing the – and X key each time;
7) Check whether the x signal changes to 0000_{HEX} after the Store key is pressed;
8) Check the value in X register whether stored into the Memory register successfully by pressing the recall button, once Recall is pressed, the value stored in the Memory register should be in the display;
9) Check whether CA key is working correctly. Pressing the CA key after the second operand input to the X register, and then input another new value, press the = key, check the result.

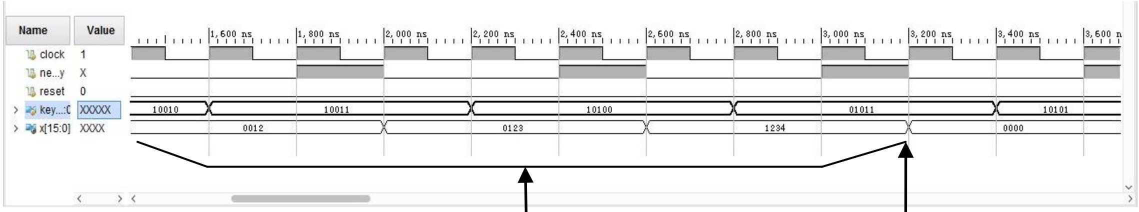
10) Check whether the CE key is working, but it cannot be checked only look at the x signal, the internal signal should be looked at as well.

Timing Diagram



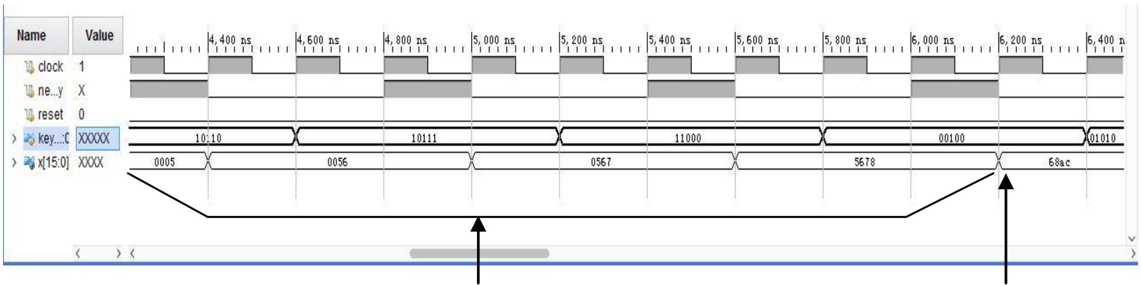
After reset, x changes to 0000_{HEX} at the following positive edge of clock

When the 1 key is pressed, the keycode stored in the rightmost 4 bits of x signal, and other bits are pushed to the left. At this moment, 0001 are in the display.



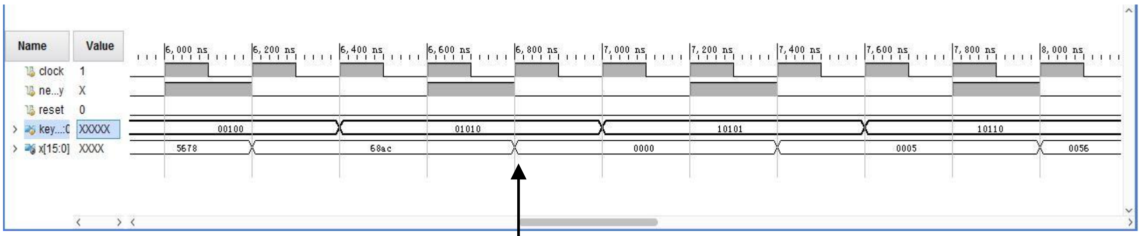
Operand 1234 is stored in the X register, it can be seen that the previous digits are shifting onto left when new digit entered.

The + key is pressed, x signal changes to 0000_{HEX} as we hoped.

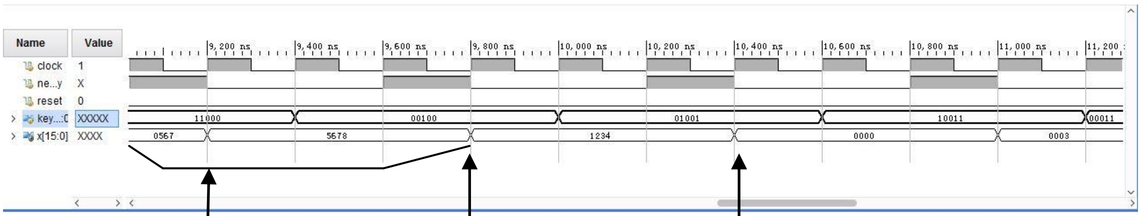


The second operand 5678 is entered and displayed.

The = key is pressed, the result of 1234 + 5678 is copied to the x register, and the result is correct.



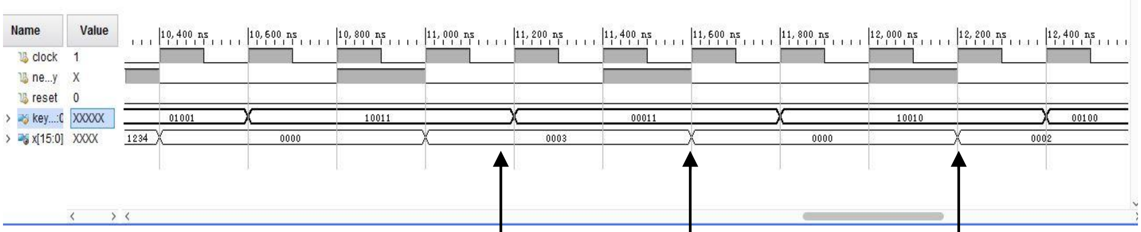
Using the result as the first operand of next subtract calculation,



The second operand of subtraction is 5678.

The = key is pressed, the result of 68ac-5678 is correct.

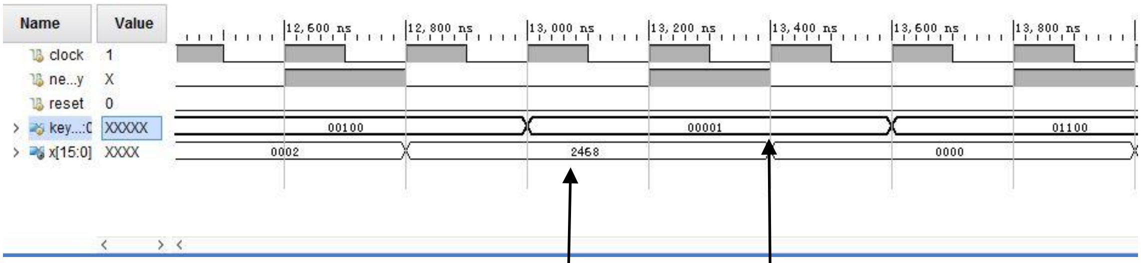
The × key is pressed, the result is used as the first operand of next multiply calculation.



The second operand 3 is entered.

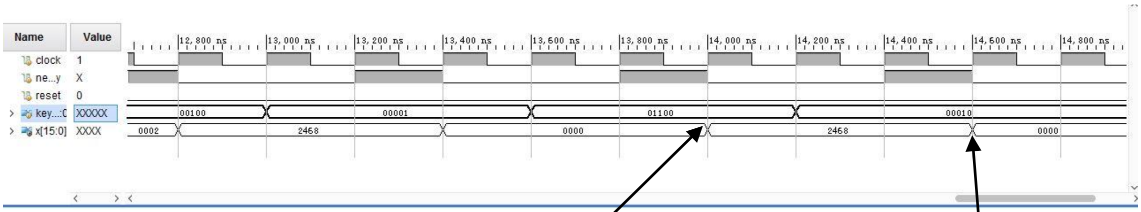
Pressing the CE key to change the second operand.

The latest second operand is 2



1234×2=2468, which is correct.

The Store key is pressed, x signal changes to $0000_{HEX.}$

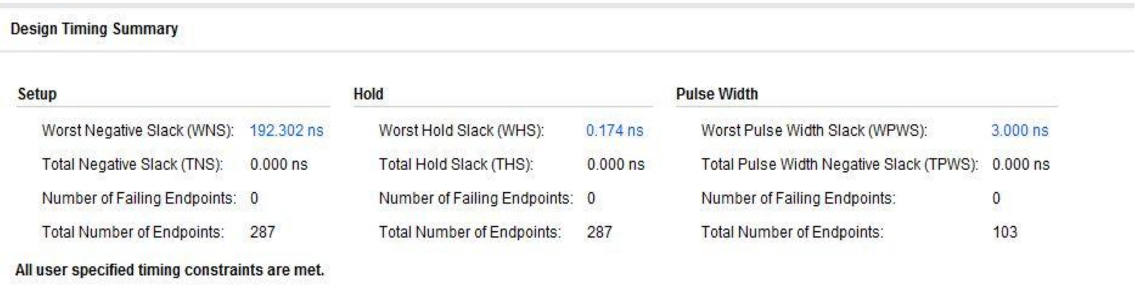


Press the Recall, the value stored in the memory is copied to the X register. Therefore, the value 2468 indeed stored in the memory.

X register is cleared after the CA button is pressed, while other register should be cleared as well.

Synthesis and Implementation

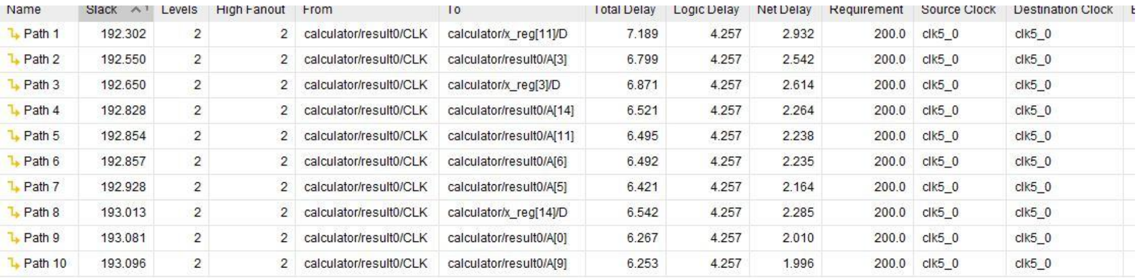
The following figures are timing analysis and utilization summary:



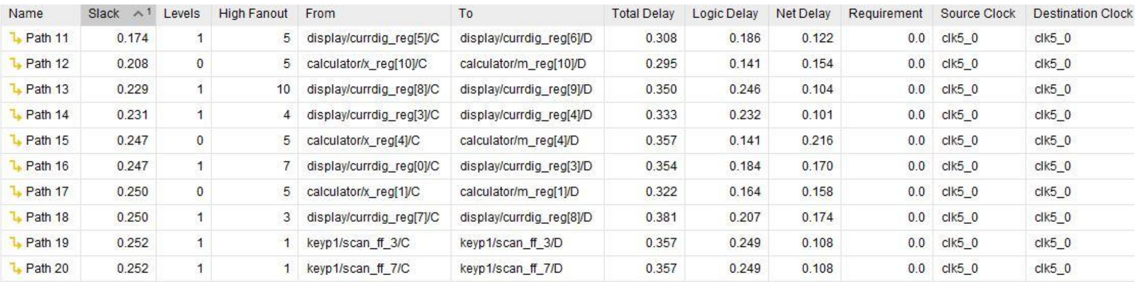
Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	192.302 ns	Worst Hold Slack (WHS):	0.174 ns	Worst Pulse Width Slack (WPWS):	3.000 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	287	Total Number of Endpoints:	287	Total Number of Endpoints:	103

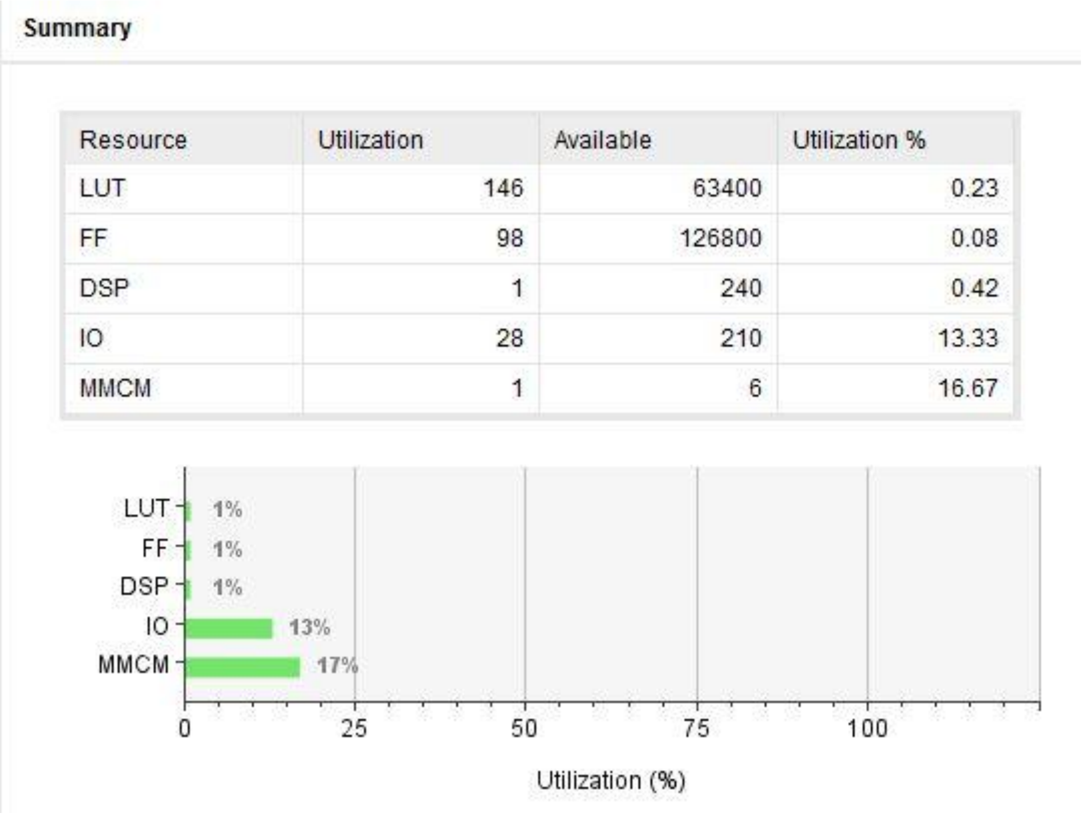
All user specified timing constraints are met.



Name	Slack	Levels	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Clock
Path 1	192.302	2	2	calculator/result0/CLK	calculator/x_reg[11]/D	7.189	4.257	2.932	200.0	clk5_0	clk5_0
Path 2	192.550	2	2	calculator/result0/CLK	calculator/result0/A[3]	6.799	4.257	2.542	200.0	clk5_0	clk5_0
Path 3	192.650	2	2	calculator/result0/CLK	calculator/x_reg[3]/D	6.871	4.257	2.614	200.0	clk5_0	clk5_0
Path 4	192.828	2	2	calculator/result0/CLK	calculator/result0/A[14]	6.521	4.257	2.264	200.0	clk5_0	clk5_0
Path 5	192.854	2	2	calculator/result0/CLK	calculator/result0/A[11]	6.495	4.257	2.238	200.0	clk5_0	clk5_0
Path 6	192.857	2	2	calculator/result0/CLK	calculator/result0/A[6]	6.492	4.257	2.235	200.0	clk5_0	clk5_0
Path 7	192.928	2	2	calculator/result0/CLK	calculator/result0/A[5]	6.421	4.257	2.164	200.0	clk5_0	clk5_0
Path 8	193.013	2	2	calculator/result0/CLK	calculator/x_reg[14]/D	6.542	4.257	2.285	200.0	clk5_0	clk5_0
Path 9	193.081	2	2	calculator/result0/CLK	calculator/result0/A[0]	6.267	4.257	2.010	200.0	clk5_0	clk5_0
Path 10	193.096	2	2	calculator/result0/CLK	calculator/result0/A[9]	6.253	4.257	1.996	200.0	clk5_0	clk5_0



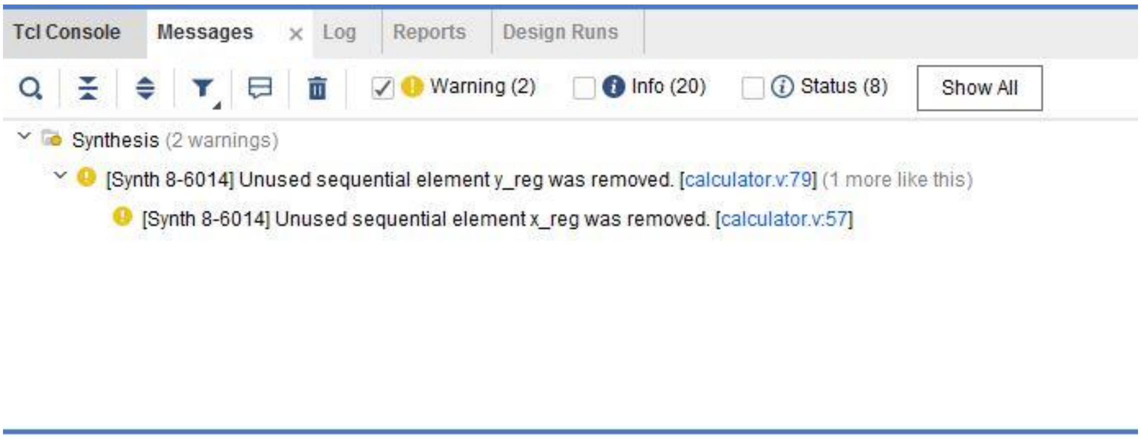
Name	Slack	Levels	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Clock
Path 11	0.174	1	5	display/currdig_reg[5]/C	display/currdig_reg[6]/D	0.308	0.186	0.122	0.0	clk5_0	clk5_0
Path 12	0.208	0	5	calculator/x_reg[10]/C	calculator/m_reg[10]/D	0.295	0.141	0.154	0.0	clk5_0	clk5_0
Path 13	0.229	1	10	display/currdig_reg[8]/C	display/currdig_reg[9]/D	0.350	0.246	0.104	0.0	clk5_0	clk5_0
Path 14	0.231	1	4	display/currdig_reg[3]/C	display/currdig_reg[4]/D	0.333	0.232	0.101	0.0	clk5_0	clk5_0
Path 15	0.247	0	5	calculator/x_reg[4]/C	calculator/m_reg[4]/D	0.357	0.141	0.216	0.0	clk5_0	clk5_0
Path 16	0.247	1	7	display/currdig_reg[0]/C	display/currdig_reg[3]/D	0.354	0.184	0.170	0.0	clk5_0	clk5_0
Path 17	0.250	0	5	calculator/x_reg[1]/C	calculator/m_reg[1]/D	0.322	0.164	0.158	0.0	clk5_0	clk5_0
Path 18	0.250	1	3	display/currdig_reg[7]/C	display/currdig_reg[8]/D	0.381	0.207	0.174	0.0	clk5_0	clk5_0
Path 19	0.252	1	1	keyp1/scan_ff_3/C	keyp1/scan_ff_3/D	0.357	0.249	0.108	0.0	clk5_0	clk5_0
Path 20	0.252	1	1	keyp1/scan_ff_7/C	keyp1/scan_ff_7/D	0.357	0.249	0.108	0.0	clk5_0	clk5_0



Summary

Resource	Utilization	Available	Utilization %
LUT	146	63400	0.23
FF	98	126800	0.08
DSP	1	240	0.42
IO	28	210	13.33
MMCM	1	6	16.67

Warning In Synthesis



Synthesis (2 warnings)

[Synth 8-6014] Unused sequential element y_reg was removed. [calculator.v:79] (1 more like this)

[Synth 8-6014] Unused sequential element x_reg was removed. [calculator.v:57]

Nothing was removed actually, the warning are here might because the optimizing way between simulator and synthesizer is different. Therefore, what the synthesizer does could not run on a simulator.

Recourses

1) calculator.v
2) CalculatorTop.v
3) clockReset.v
4) Display_interface.v
5) hex2seg.v
6) keypad-black-box.v
7) keypad.ngc

8) Nexys4_KeypadLeft.xdc
9) Nexys4_Master.xdc

10) Finaltestbench.v