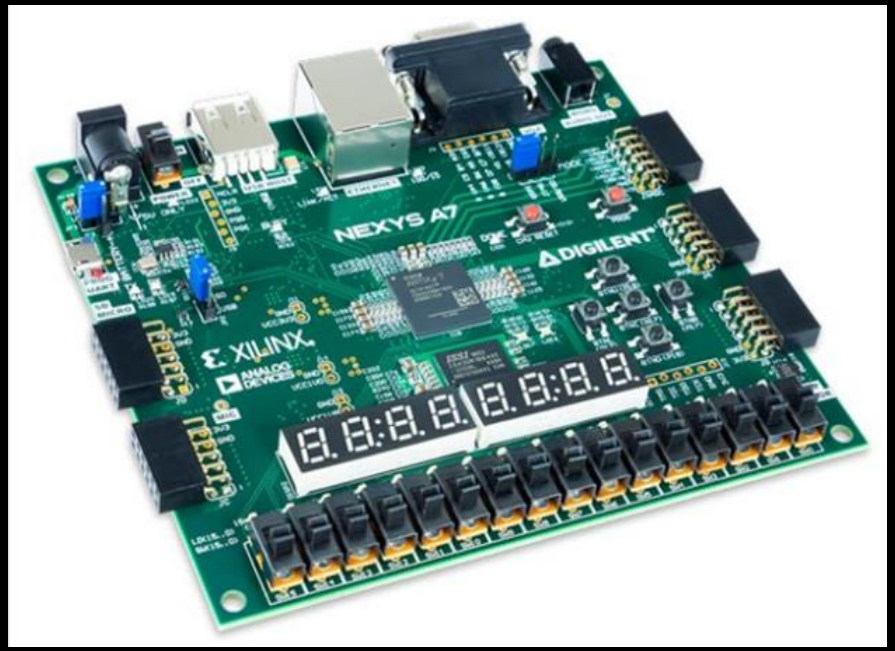
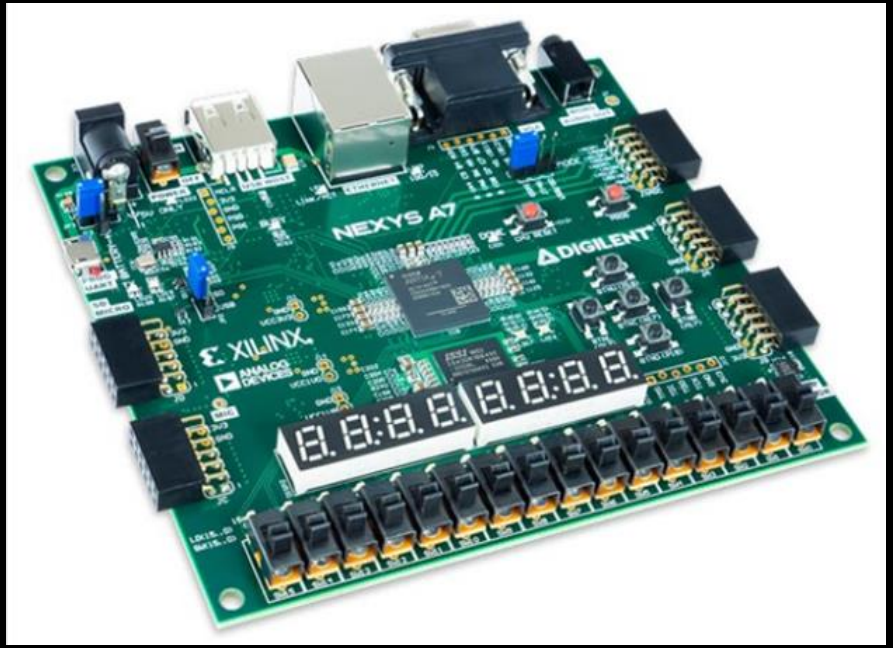


The architecture 3x3 Matrix Multiplication for unsigned numbers.

Team Members: Nour Alnounou, Genbela Lifo, Maryam Yaqo, Maryam Zadoyan
e-mails: nalnounou@oakland.edu, lifo@oakland.edu , maryamyaqo@oakland.edu, maryamzadoyan@oakland.edu.

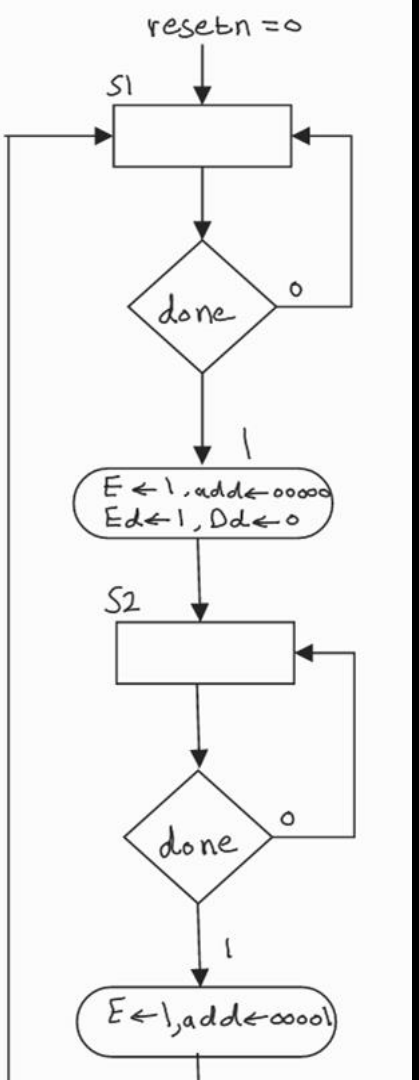
Introduction

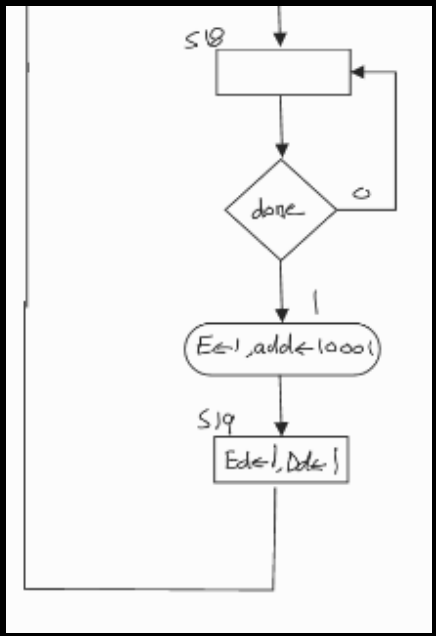
The project shows step-by-step the planning of the architectural design, the implementation, and the final results.
The architecture of the matrix multiplexer project was implemented through VHDL in the Xilinx Vivado Design Suite. For this project, an external keyboard was utilized as the system's input, which is connected through a USB port to the Nexys A7-100T board. The board will be programmed to turn the 8-bit input from the keyboard into a 4-bit input and to complete the 3x3 matrix. After the signal goes through the matrix process it will be converted from Hex to a 7-segment decoder which after will display the results on the board.



System State Diagram

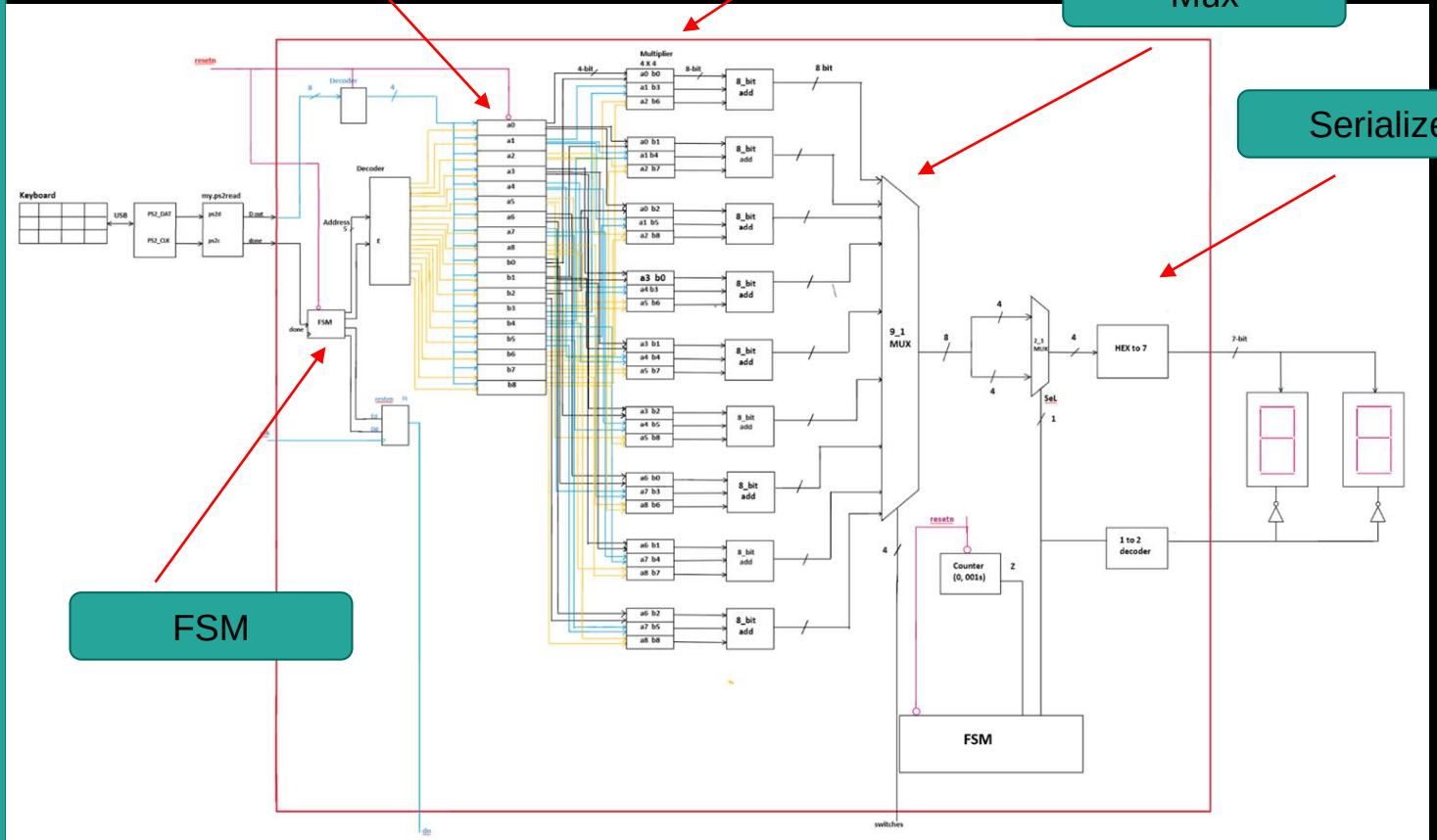
- Starting from the Restet to the first state.
- After getting the data from the keyboard the done is received.
- The enable for the decoder will be one and the address 00000 is sent.
- The second enable remains 1 and Dd done for the flip flop will be 0.
- This process repeats until state 18. After the last data from the keyboard will receive the done. Then both enables and the dd will be 1 which concludes all the data and the process goes back to state 1.

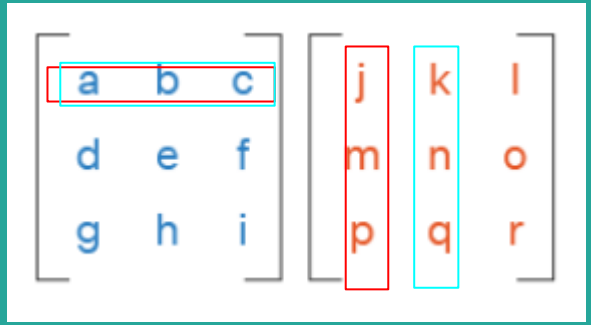




Components used for the circuit

Register
Mux
FSM
Decoder



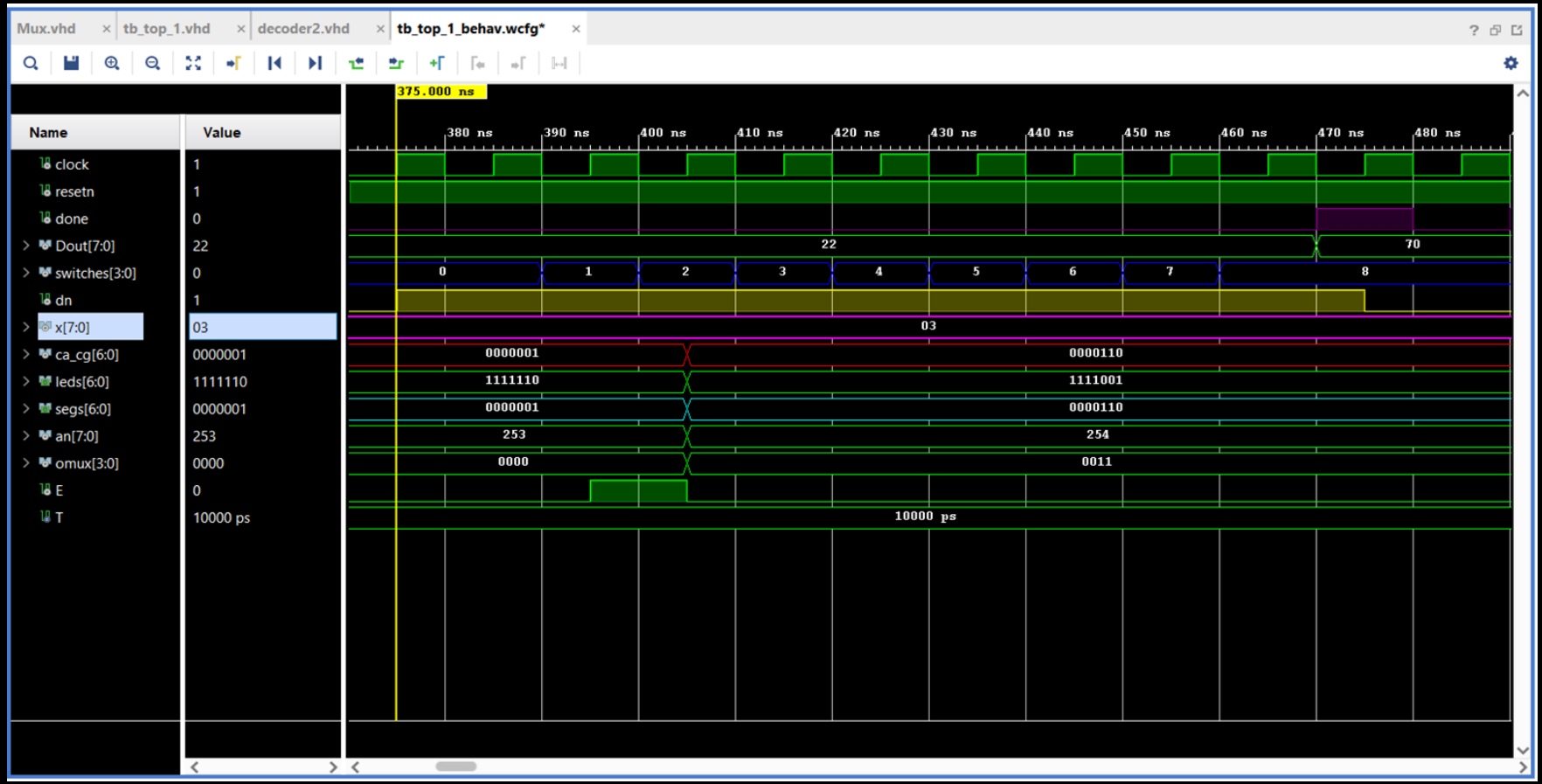


$$= \begin{bmatrix} (aj + bm + cp) & (ak + bn + cq) & (al + bo + cr) \\ (dj + em + fp) & (dk + en + fq) & (dl + eo + fr) \\ (gj + hm + ip) & (gk + hn + iq) & (gl + ho + ir) \end{bmatrix}$$

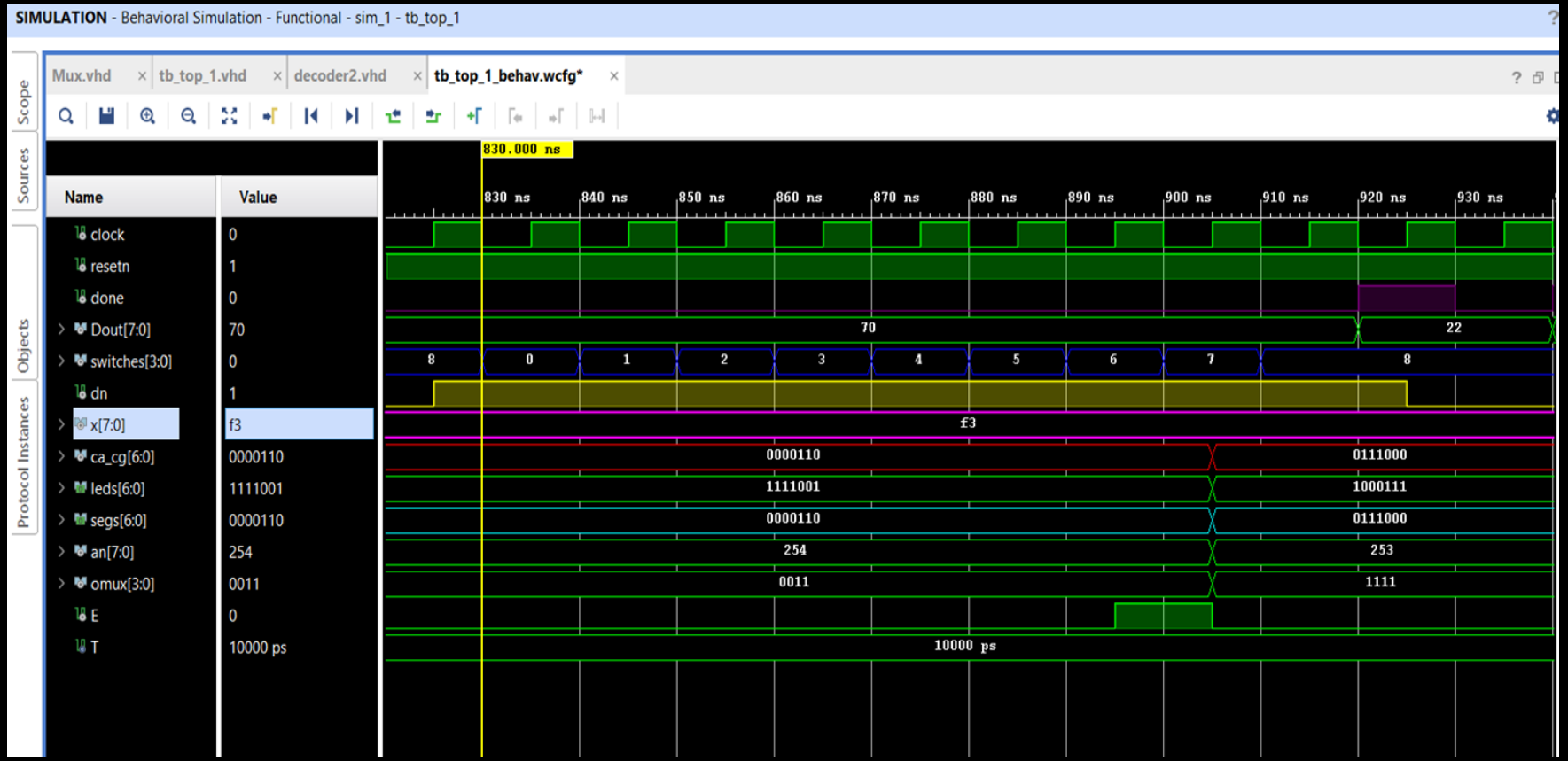
Methodology

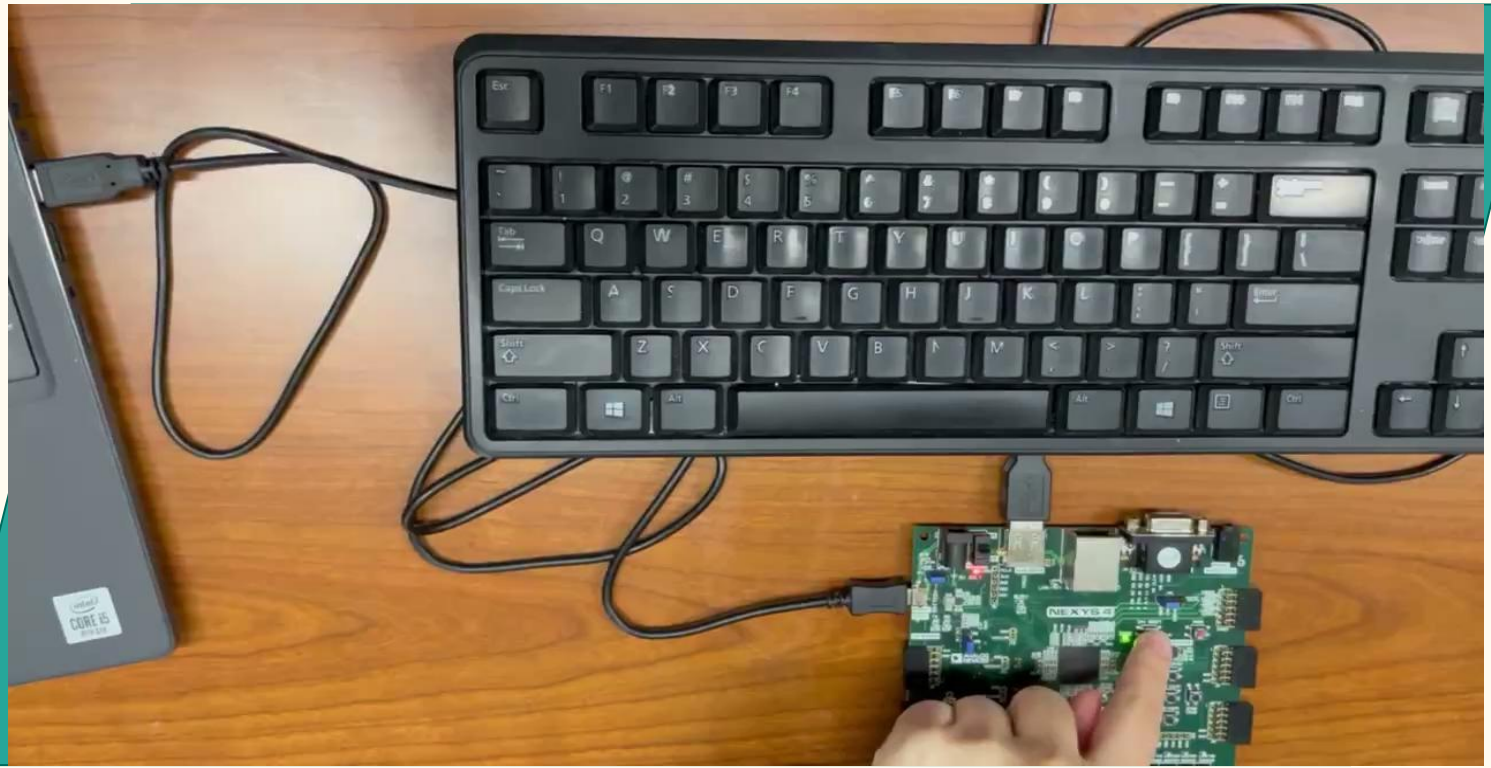
The process of building the circuit consists of one somewhat challenging circuit. The circuits not only requires knowledge on how to wire and code the components together but also understand how metsics work. The picture on the side show the math behind the multiplication of two 3 by 3 matrices

Simulation


Mux.vhd
tb_top_1.vhd
decoder2.vhd
tb_top_1_behav.wcfg*
375.000 ns
Name
Value
clock
1
resetn
1
done
0
Dout[7:0]
22
switches[3:0]
0
dn
1
x[7:0]
03
ca_cg[6:0]
0000001
leds[6:0]
1111110
segs[6:0]
0000001
an[7:0]
253
omux[3:0]
0000
E
0
T
10000 ps
380 ns
390 ns
400 ns
410 ns
420 ns
430 ns
440 ns
450 ns
460 ns
470 ns
480 ns
22
70
0
1
2
3
4
5
6
7
8
03
0000001
0000110
1111110
1111001
0000001
0000110
253
254
0000
0011
10000 ps

Simulation


SIMULATION - Behavioral Simulation - Functional - sim_1 - tb_top_1
Mux.vhd
tb_top_1.vhd
decoder2.vhd
tb_top_1_behav.wcfg*
Scope
Sources
Objects
Protocol Instances
Name
Value
clock
0
resetn
1
done
0
Dout[7:0]
70
switches[3:0]
0
dn
1
x[7:0]
f3
ca_cg[6:0]
0000110
leds[6:0]
1111001
segs[6:0]
0000110
an[7:0]
254
omux[3:0]
0011
E
0
T
10000 ps
830.000 ns
830 ns
840 ns
850 ns
860 ns
870 ns
880 ns
890 ns
900 ns
910 ns
920 ns
930 ns
70
22
8
0
1
2
3
4
5
6
7
8
f3
0000110
0111000
1111001
1000111
0000110
0111000
254
253
0011
1111
10000 ps


Esc
F1
F2
F3
F4
Tab
Q
W
E
R
T
Y
U
I
O
P
Caps Lock
A
S
D
F
G
H
J
K
L
Enter
Shift
Z
X
C
V
B
N
M
Shift
Ctrl
Alt
Alt
Ctrl
CORE i5
NEXYS 4

Demo Video

Thank you!!