SoftMax Report

I will discuss three different implementations of softmax and make a comparison.

1. Design 1: softmax module
2. Design 2: softmax\_1 module
3. Design 3: softmax\_2 module.

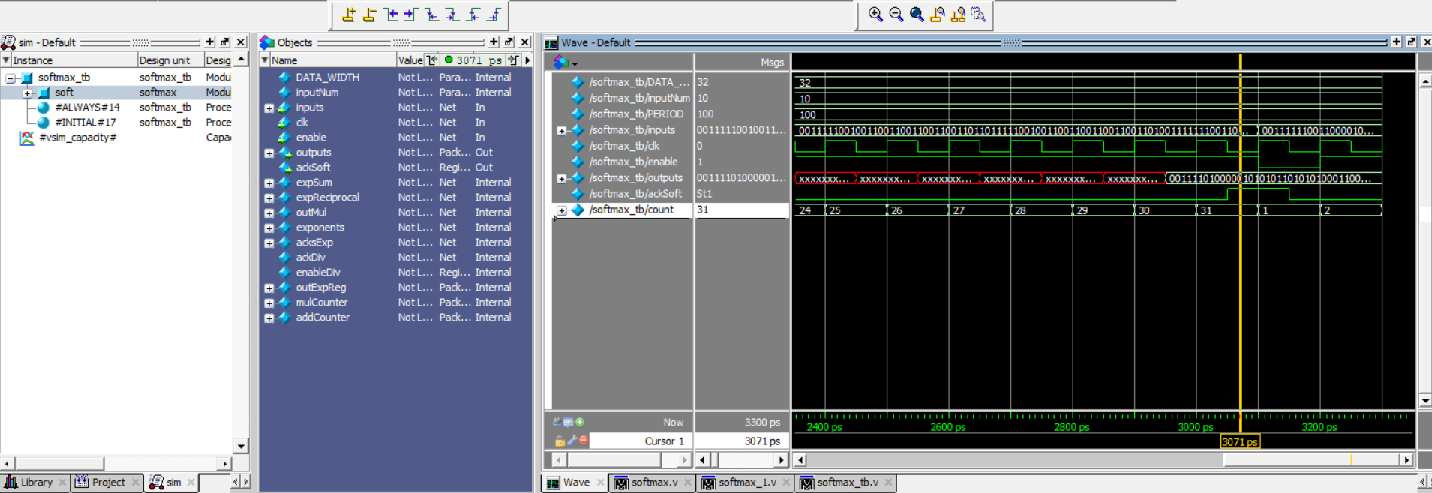
Note: The exponent uses 1356 LUTs , Reciprocal is 2600 LUTs , Multiplication is 340, Addition is 540.

All Answers were double checked with a python script and are correct.

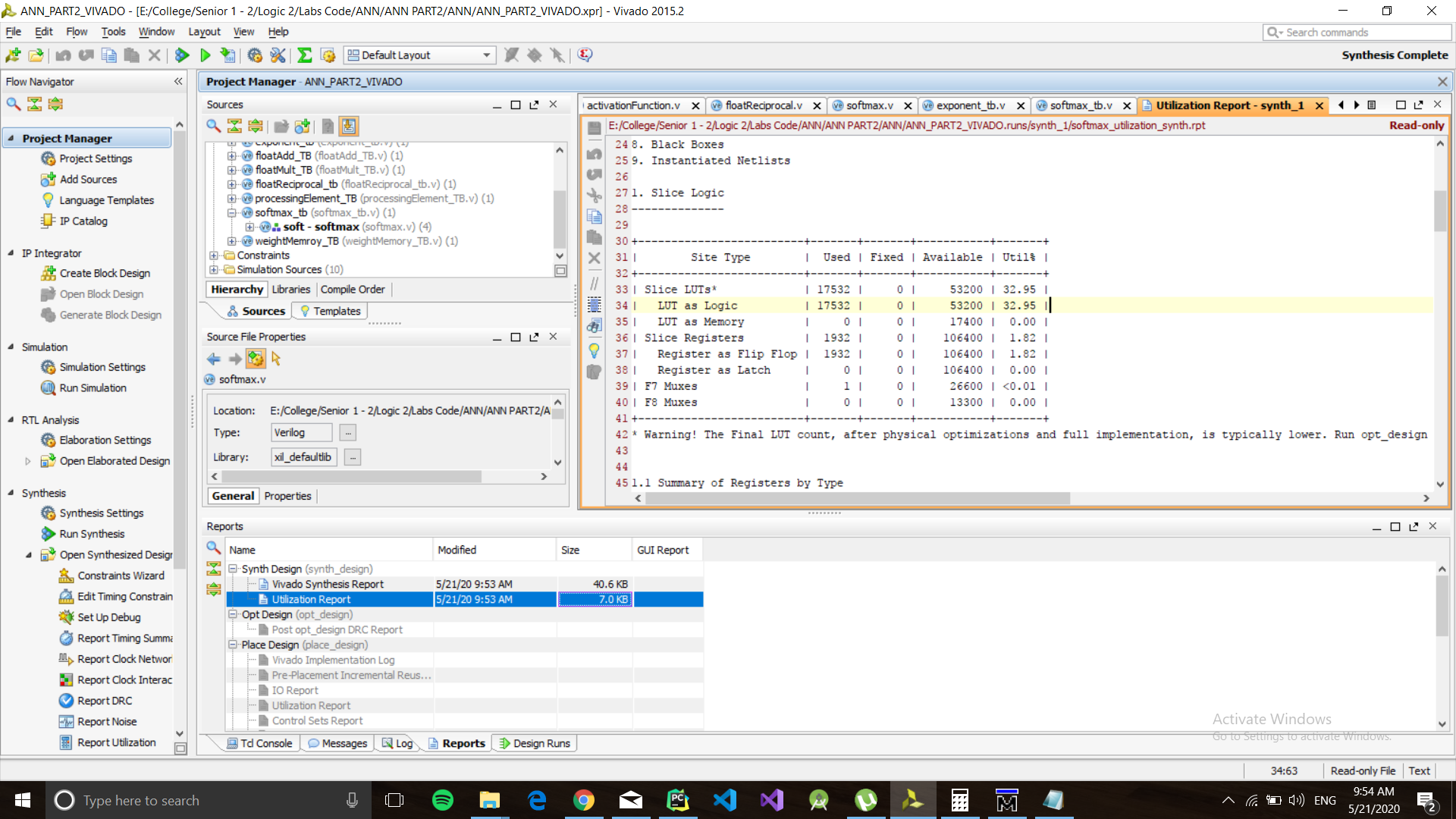
**Design 1**

Design Description: 10 exponent units, 1 floating adder , 1 multiplier, 1 division unit.

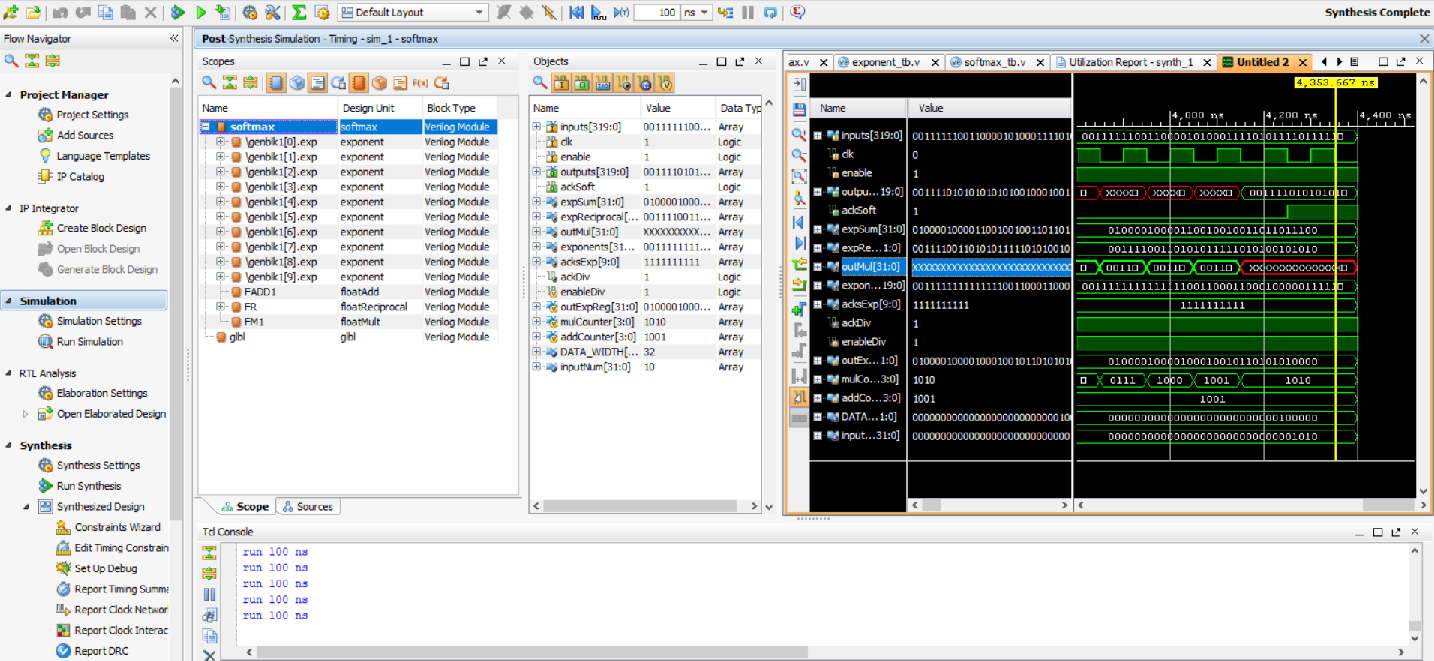
Clock Cycles: 31 clock cycles as shown below. ackSoft is triggered



Synthesis Report: 17532/53200 LUTs 32.95% utilization.



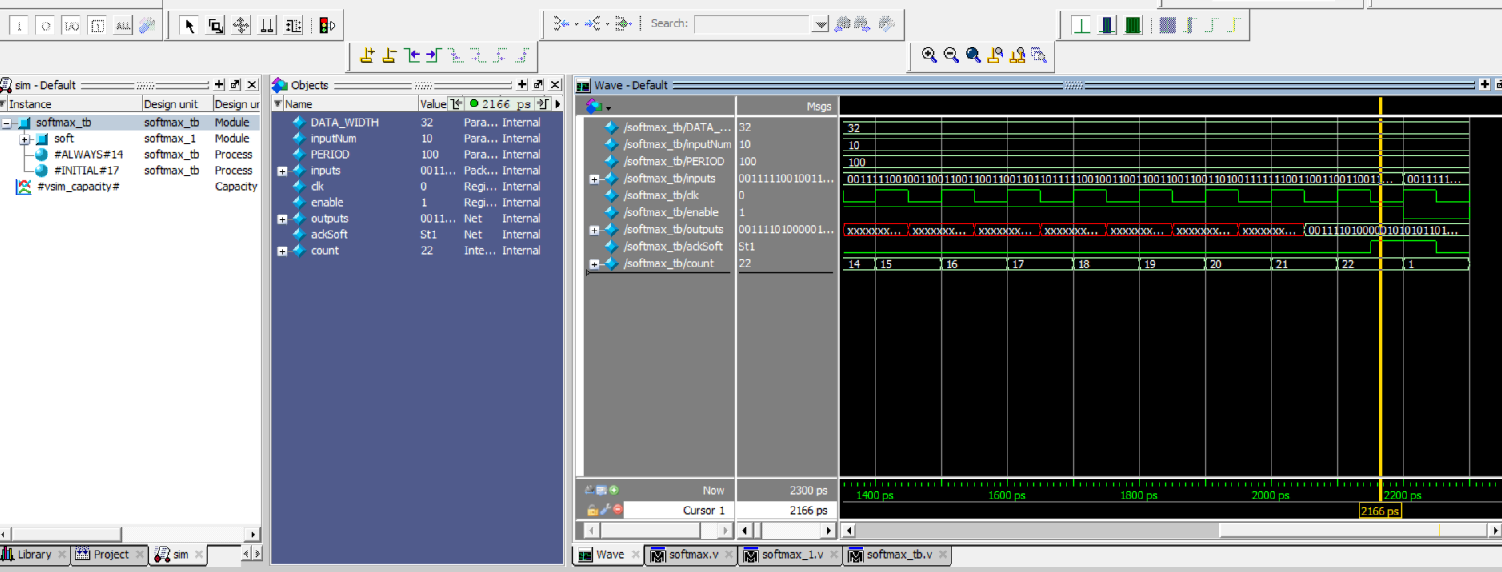
Timing Diagram



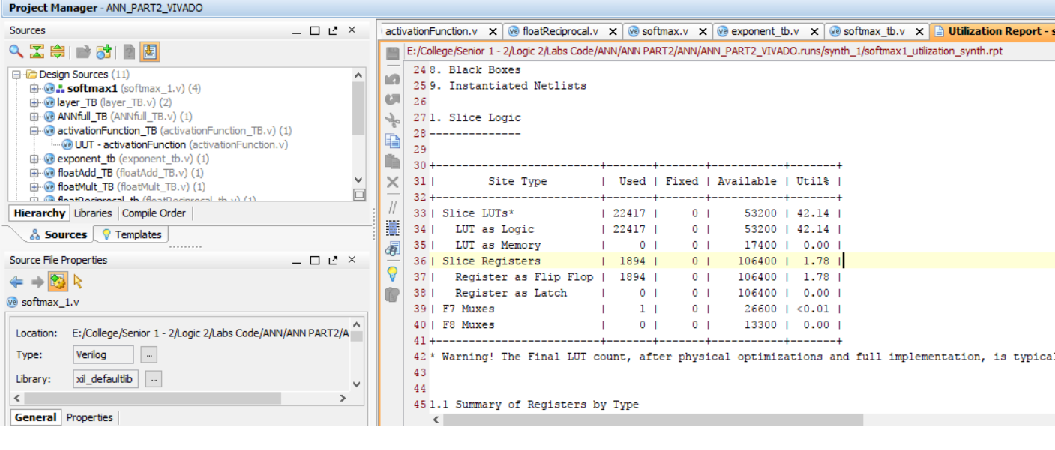
**Design 2**

Design Description: 10 exponent units, 10 floating adder , 1 multiplier, 1 division unit.

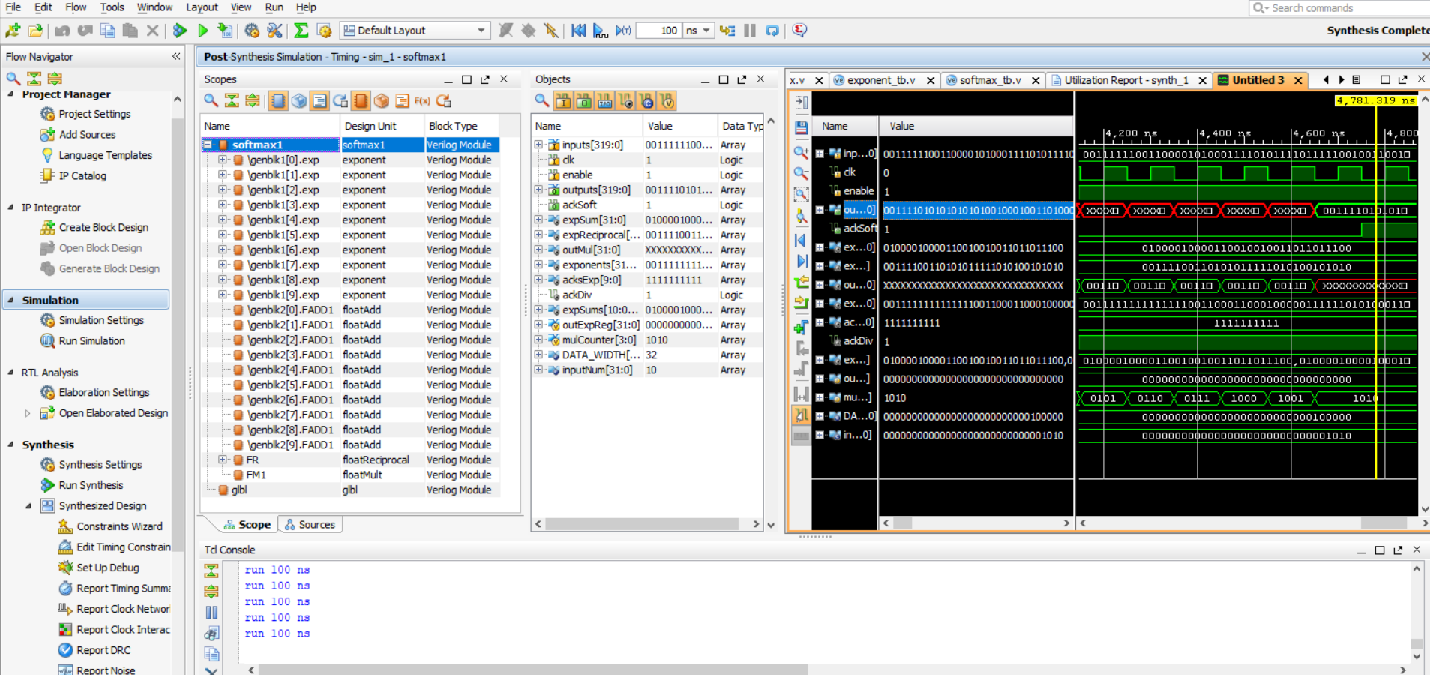
Clock Cycles: 22 clock cycles as shown below, ackSoft is triggered



Synthesis Report: 22417/53200 LUTs 42.14% utilization



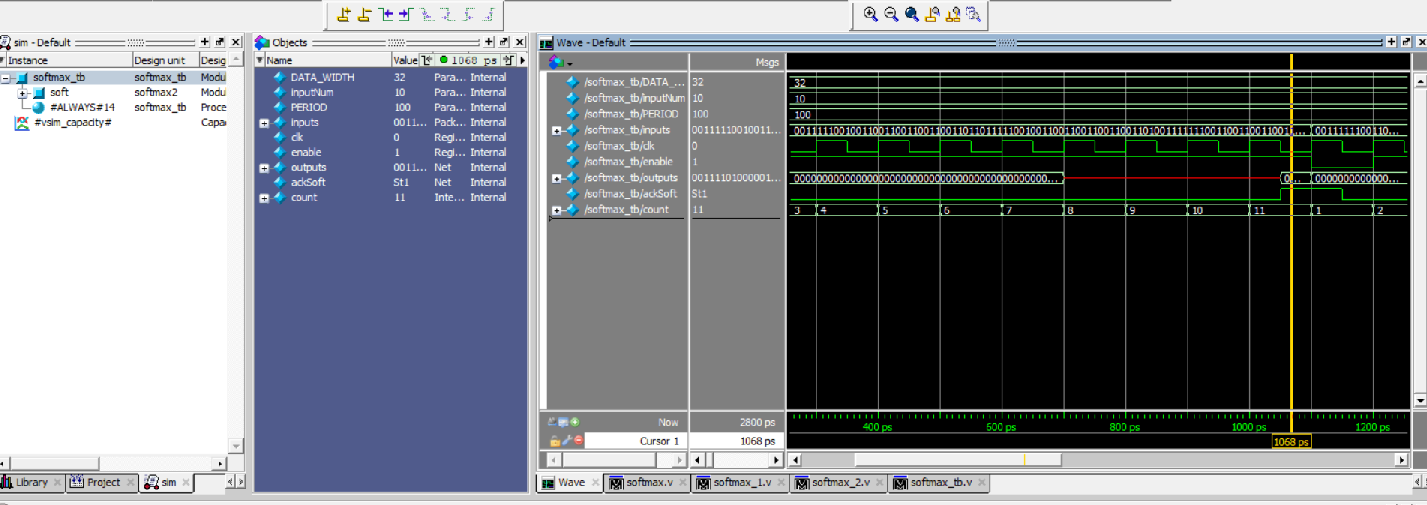
Timing Diagram:



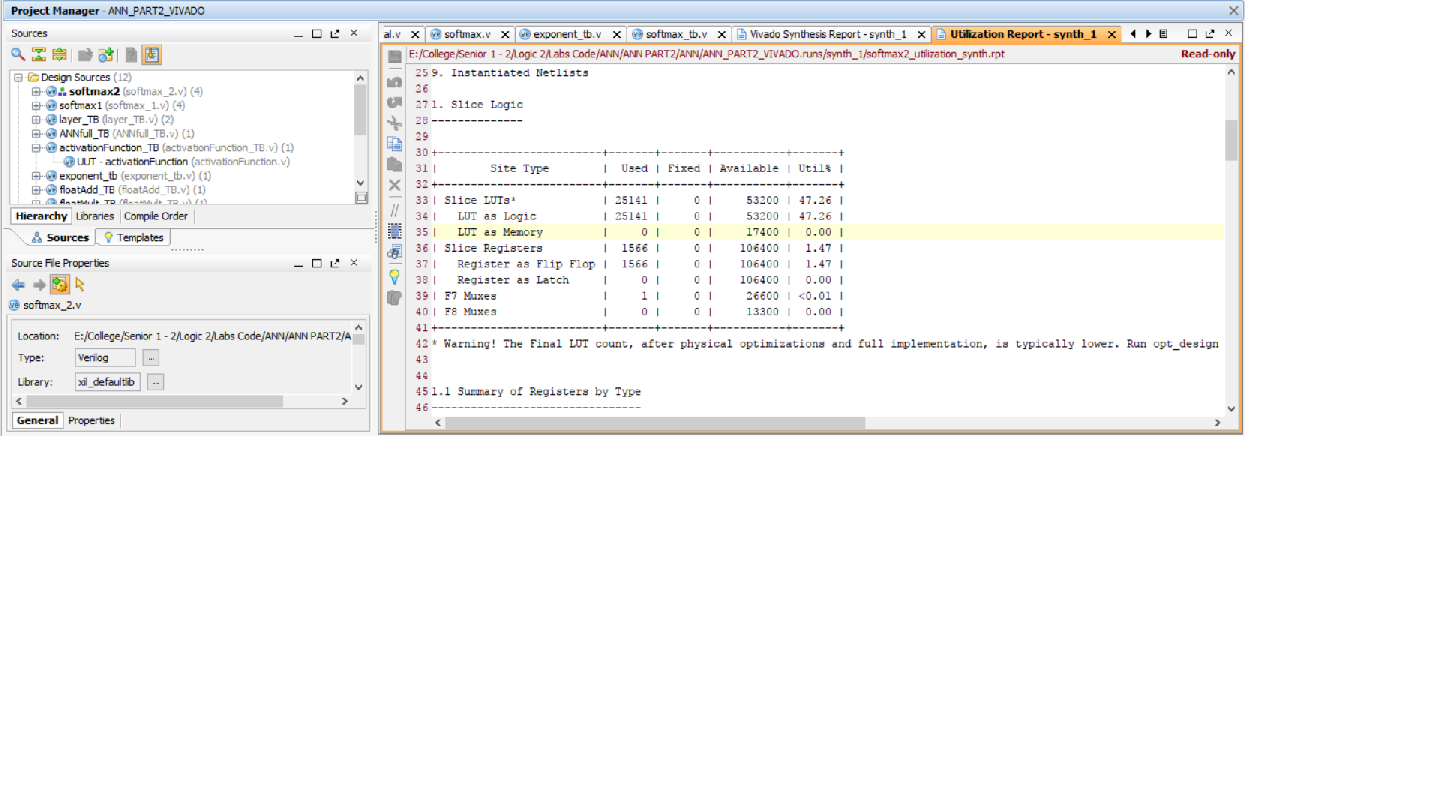
**Design 3**

Design Description: Design Description: 10 exponent units, 10 floating adder , 10 multiplier, 1 division unit.

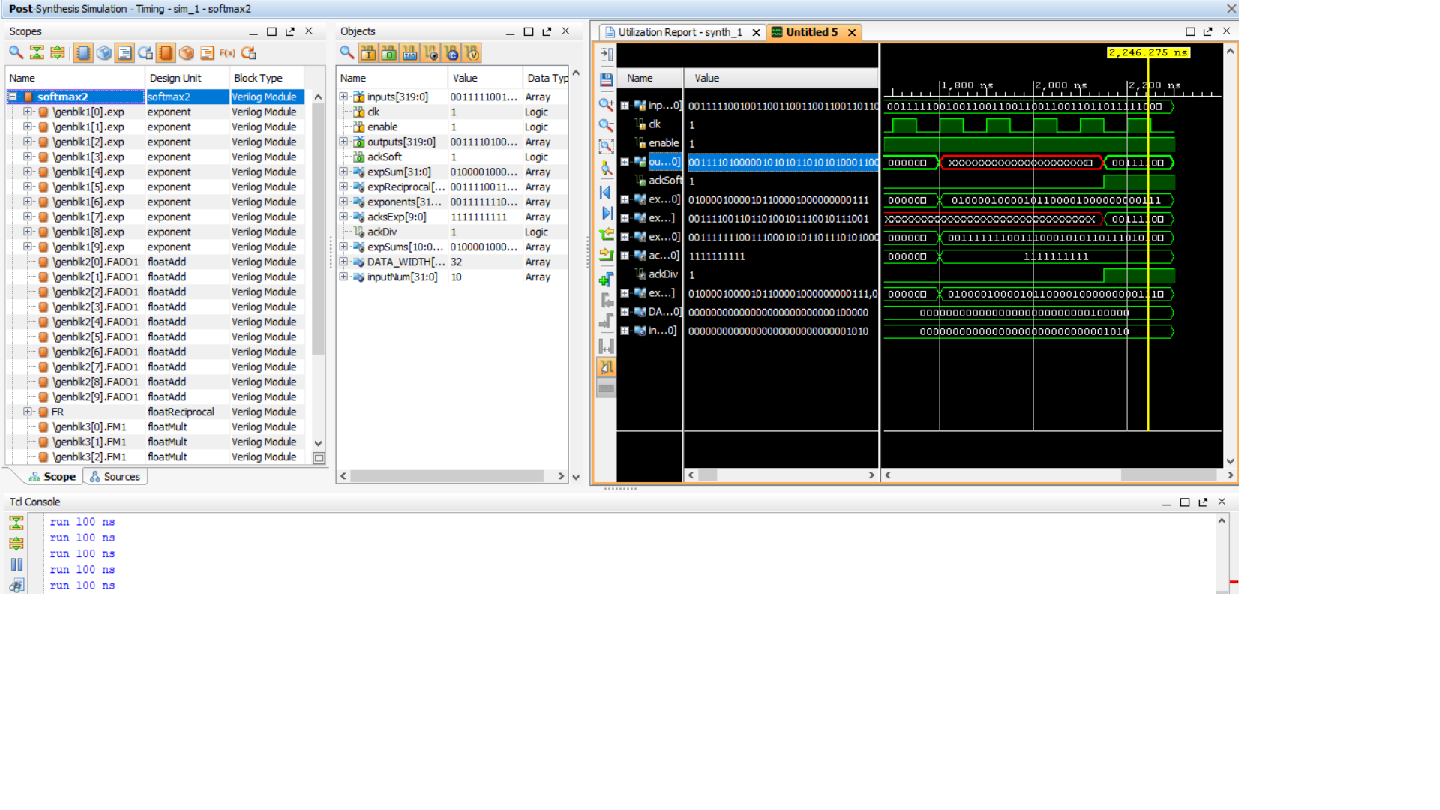
Clock Cycles : 11 cycles as shown. ackSoft is triggered



Synthesis Report: 25141/53200 47.26% utlization



Timing Diagram



Verification that the design is working

In this section I will discuss how I verified that the design is working using python scripts and Floating Point to IEEE converter. <https://www.h-schmidt.net/FloatConverter/IEEE754.html>

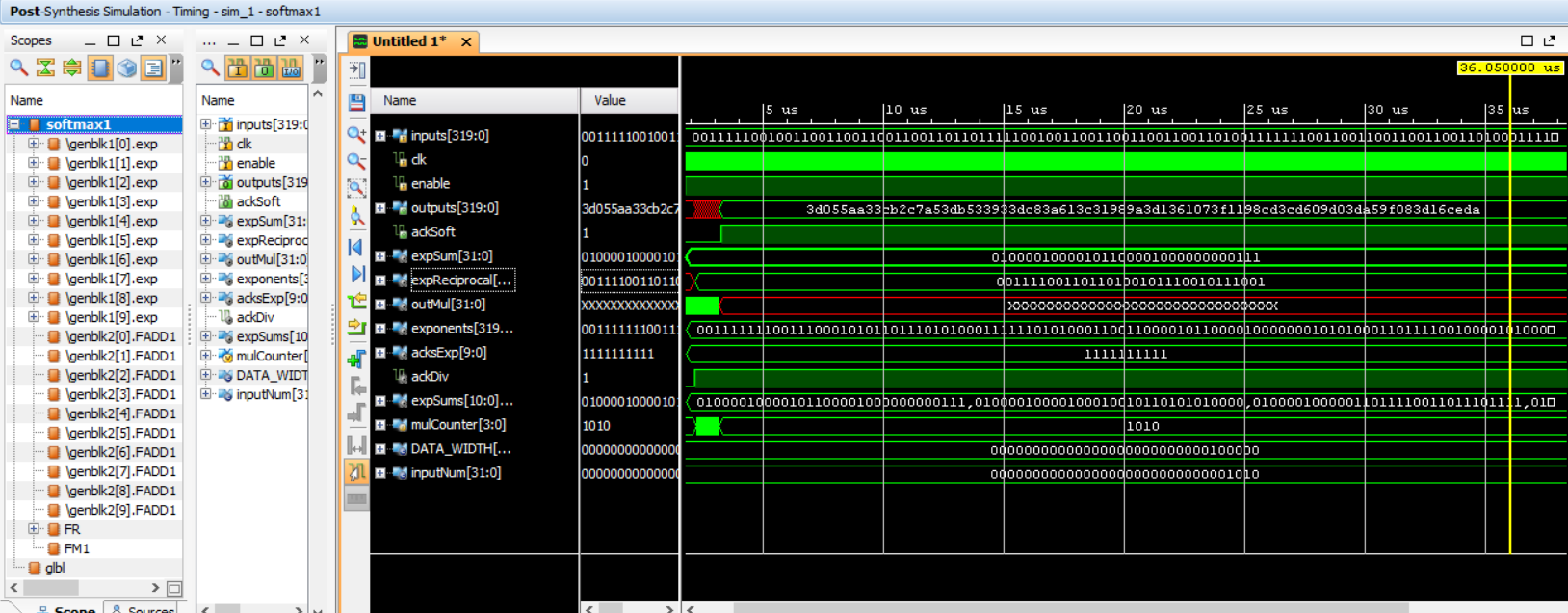
The 10 inputs we will use to verify are 0.2 -0.2 1.2 1.3 -0.9 0.3 3.1 -0.02 1.11 0.323

The input is then run through the softmax.py file and the input binary string which is the input of the module in the post synthesis simulation is written to log.txt.

In our case the input string is

00111110010011001100110011001101101111100100110011001100110011010011111110011001100110011001101000111111101001100110011001100110101111110110011001100110011001100011111010011001100110011001101001000000010001100110011001100110101111001010001111010111000010100011111110001110000101000111101100111110101001010110000001000010

In the figure below is the output of the system after the ackSoft signal is High.



Hexadecimal result: 3d055aa33cb2c7a53db533933dc83a613c31989a3d1361073f1198cd3cd609d03da59f083d16ceda

Binary result:

00111101000001010101101010100011001111001011001011000111101001010011110110110101001100111001001100111101110010000011101001100001001111000011000110011000100110100011110100010011011000010000011100111111000100011001100011001101001111001101011000001001110100000011110110100101100111110000100000111101000101101100111011011010

The binary result is then used in the split.py file to get 10 IEEE 32 bit floating point numbers.

The result of splitting this string is ['00111101000001010101101010100011', '00111100101100101100011110100101', '00111101101101010011001110010011',  
 '00111101110010000011101001100001', '00111100001100011001100010011010', '00111101000100110110000100000111',  
 '00111111000100011001100011001101', '00111100110101100000100111010000', '00111101101001011001111100001000',  
 '00111101000101101100111011011010']

Which corresponds to these numbers

[0.0325571410358, 0.0218237135559, 0.0884772762656, 0.097767598927, 0.0108396057039, 0.0359812043607, 0.56873780489, 0.0261277258396, 0.0808697342873, 0.0368183627725]

Now after running the softmax.py we get two results , first is the softmax numbers if we use math.exponent, and softmax if we use taylor.

Here is the results If we use actual exponent function.

RealSoftmax is [0.03182415 0.02133236 0.086507 0.09560502 0.01059334 0.03517112  
 0.57837666 0.02553948 0.07906144 0.03598943]

Here is the results if we use taylor exponent used in the design.

TaySoftmax is [0.03255714 0.0218237 0.08847725 0.09776757 0.0108396 0.0359812  
 0.56873776 0.02612771 0.08086971 0.03681836]

The Tay softmax results are very similar to the real softmax considering the approximation we used for the exponent function. And also the Tay softmax results are the same as the output of the softmax module in the post synthesis simulation.