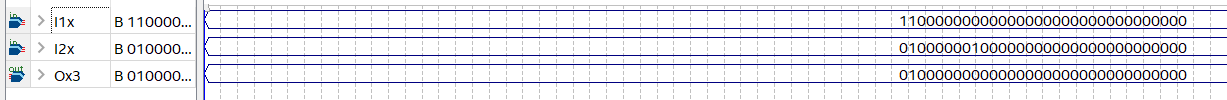
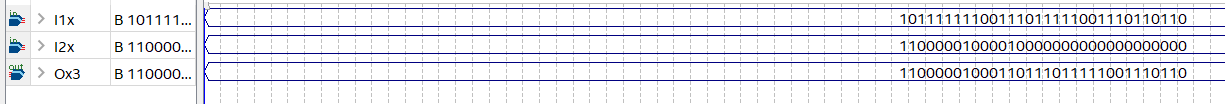
**Floating point adder and subtractor**

I1x and I2x are the inputs and 03x is the output.

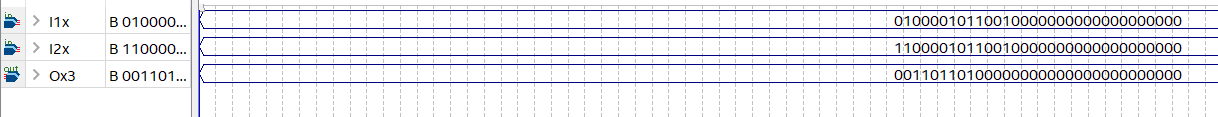
-2+4=2 (testing case when I1x is negative and I2x is positive)

****

-1.234-8.5= -9.73399925232 (both negative case)

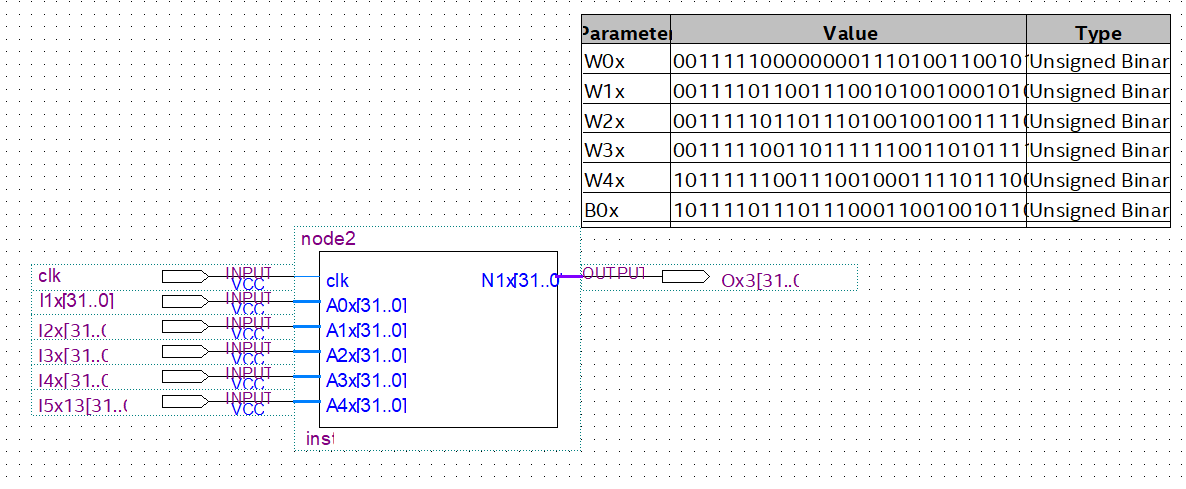


100-100= 3.81469726562e-06 (second input is negative case)

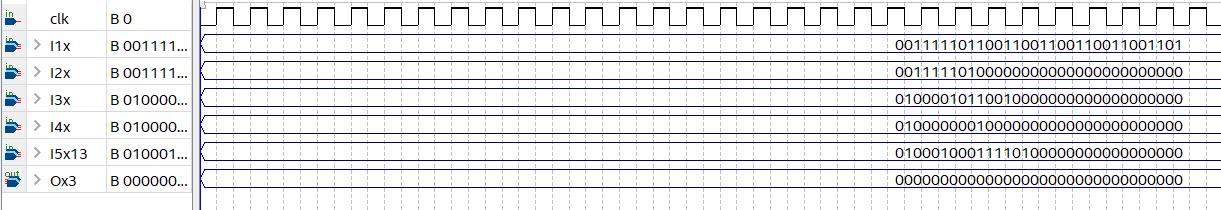


This is proof that the adder and two subtractors inside float\_adder work correctly.

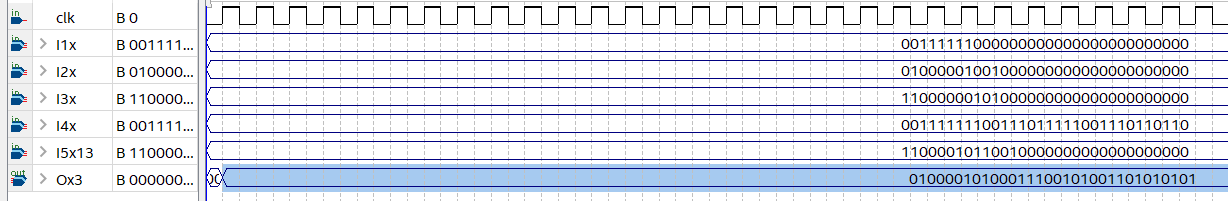
**Node2\_1 test**



|  |  |
| --- | --- |
| **Input** | **Expected output** |
| 0.4-00111110110011001100110011001101 | -679.27-11000100001010011101000101001000  After Relu should be all zero |
| 0.25-00111110100000000000000000000000 |  |
| 100-01000010110010000000000000000000 |  |
| 3-01000000010000000000000000000000 |  |
| 1000-01000100011110100000000000000000 |  |



|  |  |
| --- | --- |
| **Input** | **Expected output** |
| 0.5-00111111000000000000000000000000 | 71.09  01000010100011100010111000010100 |
| 10-01000001001000000000000000000000 |  |
| -5 |  |
| 1.234 |  |
| -100 |  |



**Layer 1 test with input file neg with 000.00000 and 8 bit input/16 bit overflow**

|  |
| --- |
| **Expected outputs** |
|  |
| **Quartus outputs** |
| Node\_1= 0 as expected for a negative number  Node\_2=0.09375 (this one is quite off?)  Node\_3=0.625  Node\_4= 0  Node\_5= 0.0625 |

**Layer 2 test with input file neg with 000.00000 and 8 bit input/16 bit overflow**

|  |
| --- |
| **Expected outputs** |
|  |
| **Quartus outputs** |
| Node\_1= 0.4375  Node\_2=0.5  Node\_3=0  Node\_4=0  Node\_5=0  Node\_6 = 0.46875  Node\_7=0.34375  Node\_8= 0.03125  Node\_9=0  Node\_10=0 |

The design is giving the correct sign but the magnitudes seem to be a little bit off. Perhaps we need to change the placement of the decimal point and the number of overflow bits.

**Layer 1 with q6.26 with 16 bit input (forgot to retrain for 16 bit input, this is the 8 bit model)**

|  |
| --- |
|  |
| Node\_1=0  Node\_2= 0.1875  Node\_3=0.75  Node\_4=0  Node\_5=0.117 |

**Layer 1 with q6.26 with 16 bit input (16 bit trained)**

|  |
| --- |
|  |
| Node\_1=0  Node\_2= 0.1953  Node\_3=0.75  Node\_4=0  Node\_5=0.122 |

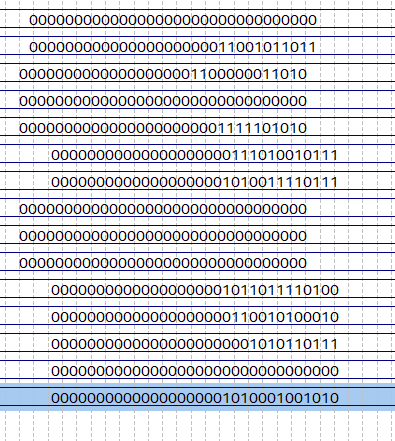
**Layer 2 with q6.26 16 bit input**

|  |
| --- |
|  |
| 0.455  0.655  0  0  0  0.717  0.39  0.08  0  0 wrong |

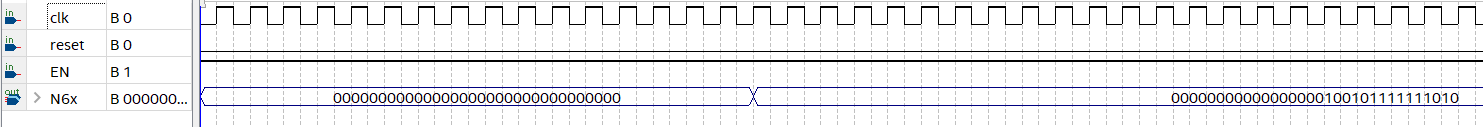
**Final results**

|  |  |  |
| --- | --- | --- |
| **Layer** | **Expected** | **Result** |
| 1 | 0  0.1691306406  0.7398626  0  0.05069419 | 0  0.1953  0.75  0  0.122 |
| 2 | 0.6181573670;  0.6051208924;  0.0000000000;  0.0000000000;  0.0000000000;  0.4855168283;  0.4722739135;  0.1755495747;  0.0000000000;  0.7082911282; | 0.455  0.655  0  0  0  0.717  0.39  0.08  0  0.634 |

**Final waveform for q6.26 16 bit input**



**Healthy ECG output**



This is 2.34 and we were expecting 1.34, for our purposes of only detecting a non-zero number this will still able to classify, the difference in the numbers might cause a slight drop in accuracy when testing with lots of data.

**Unhealthy ECG output**

Gave us 0 as expected