Quantization: Accuracy vs. Efficiency

CPRE 482X - Dr. Duwe

Dawood Ghauri
Jacob Larimore
Ryan Hunt

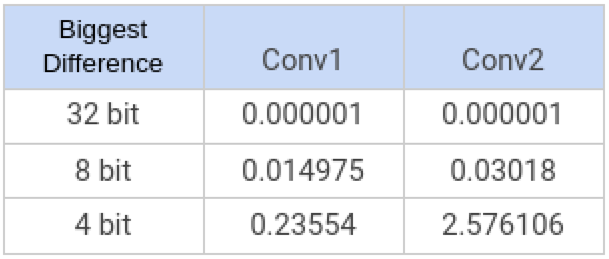
Purpose:
The purpose of our project is to compare the accuracy of a quantized deep neural network with the effects on the design area, power and timing.

Code Implementation:
We will reuse our C++ implementation from the previous labs for the 32-bit DNN and the 8-bit DNN, but we will take it one step further and also implement a 4-bit DNN. We will compare the values from these DNNs with what we expect and find a max difference between them. This will help us in determining how accurate each implementation is. As our values become more quantized, we expect our values to become less accurate when compared to the outputs we received from performing inference using TensorFlow.

Hardware Implementation:
This project will go even further than our labs by creating simple, 32-bit, 8-bit and 4-bit quantized MAC models in verilog, synthesizing them in Genus, and collecting timing, power, and area reports for the synthesized circuits. We expect the hardware metrics to improve as the lower bits are implemented. We will compare our results with the code implementation and also observe how results align with what we expect.

Code Design:
As stated in the Code Implementation, we converted our lab 2 code to suit our needs for calculating the accuracy of our 32-bit, 8-bit, and 4-bit MACs. We decided to go through the first and second layers as both of these were convolutional layers that were able to be quantized, and we felt that these two layers would be sufficient in showing the differences between different quantized results. Both layers accepted a float input, quantized, performed convolution, then dequantized for comparison/analysis and for the input to the next layer. The 32-bit used the float data type, while 8-bit and 4-bit data used the uint8_t datatype as we are not able to use a 4 bit data type in C++. Using uin8_t to store our 4-bit quantization would have been a problem if we were measuring timing in C++, however we are comparing timing using our hdl MAC units. Using the uint8_t data type should not affect the results we receive for accuracy.

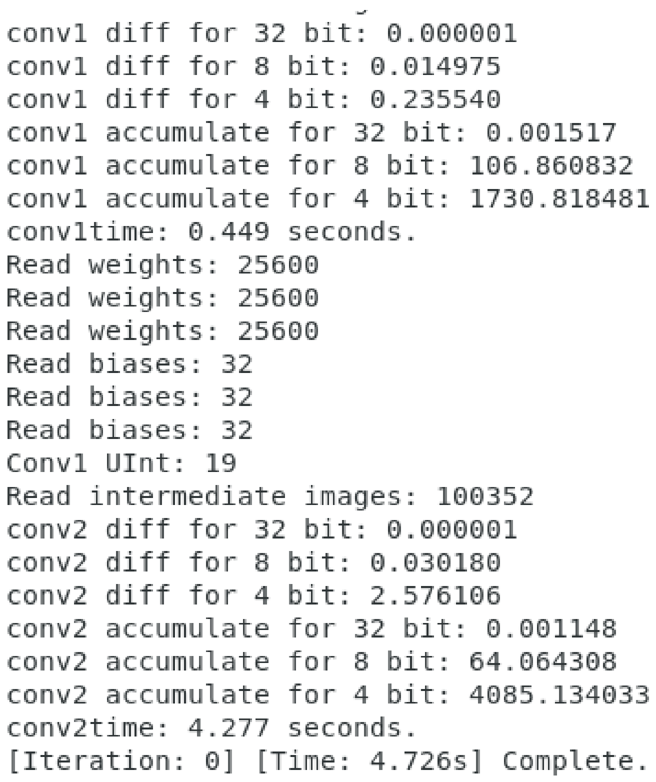
Code Simulation:



Biggest Difference	Conv1	Conv2
32 bit	0.000001	0.000001
8 bit	0.014975	0.03018
4 bit	0.23554	2.576106

Table 1: Max Difference from Tensorflow Outputs

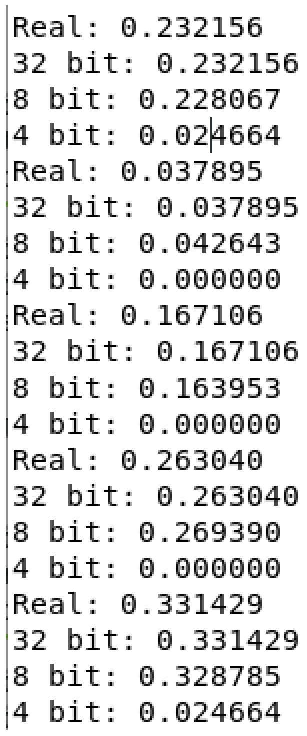
As we can see in Table 1, the difference in accuracy from the 32 bit to 8 bit is different by quite a few magnitudes while the difference between 8-bit and 4-bit is only different by about 1 magnitude for the first layer but got exponentially worse for the second layer. We believe this is because there were lots of values that got rounded down to 0 for the 4-bit implementation.



```
conv1 diff for 32 bit: 0.000001
conv1 diff for 8 bit: 0.014975
conv1 diff for 4 bit: 0.235540
conv1 accumulate for 32 bit: 0.001517
conv1 accumulate for 8 bit: 106.860832
conv1 accumulate for 4 bit: 1730.818481
conv1time: 0.449 seconds.
Read weights: 25600
Read weights: 25600
Read weights: 25600
Read biases: 32
Read biases: 32
Read biases: 32
Conv1 UInt: 19
Read intermediate images: 100352
conv2 diff for 32 bit: 0.000001
conv2 diff for 8 bit: 0.030180
conv2 diff for 4 bit: 2.576106
conv2 accumulate for 32 bit: 0.001148
conv2 accumulate for 8 bit: 64.064308
conv2 accumulate for 4 bit: 4085.134033
conv2time: 4.277 seconds.
[Iteration: 0] [Time: 4.726s] Complete.
```

Figure 1: Console Printed Results for Code Simulations

Figure 1 is showing the raw output information that we collected from simulating our code. Both `"conv1 diff.."` and `"conv2 diff.."` show the max difference for each of the implementations. The `"conv1 accumulate.."` and `"conv2 accumulate.."` give an added benchmark to show what accumulated differences look like for each implementation.



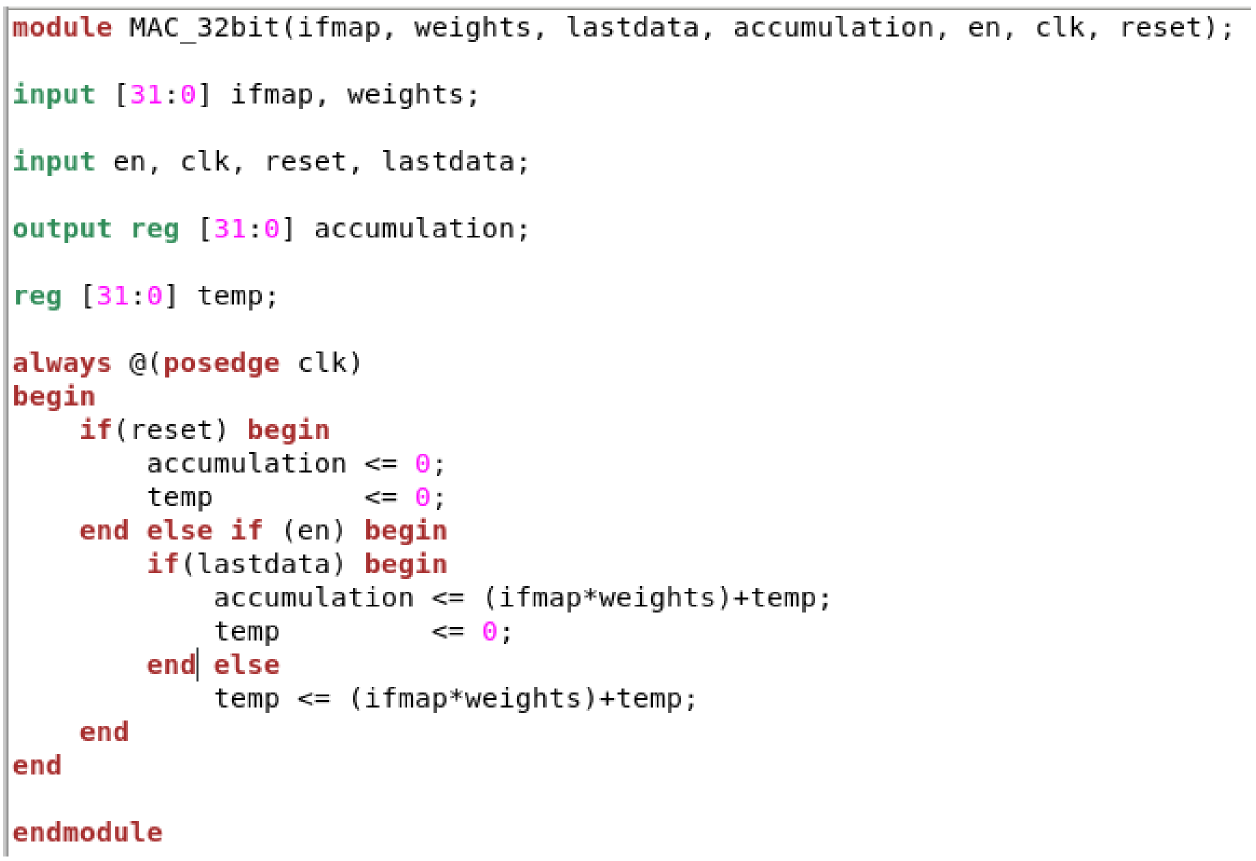
```
Real: 0.232156
32 bit: 0.232156
8 bit: 0.228067
4 bit: 0.024664
Real: 0.037895
32 bit: 0.037895
8 bit: 0.042643
4 bit: 0.000000
Real: 0.167106
32 bit: 0.167106
8 bit: 0.163953
4 bit: 0.000000
Real: 0.263040
32 bit: 0.263040
8 bit: 0.269390
4 bit: 0.000000
Real: 0.331429
32 bit: 0.331429
8 bit: 0.328785
4 bit: 0.024664
```

Figure 2: Console Output for Real Values (Test Inputs), Float-32, 8-bit, and 4-bit Convolution.

Figure 2 is showing an output stream of results when looping through convolution for every quantization MAC. Real represents the values we received from Tensorflow while all the other values correspond to their respective MAC unit.

<u>Hardware Design:</u>

Figure 3 shows the code implementation of our simplified MAC design in verilog. We wanted to keep the MAC unit fairly simple as we are only doing a side-by-side comparison of quantization and not actually applying the model to anything that would need handling. We basically have the exact same code for 32-bit, 8-bit, and 4-bit quantization as seen in Figure 3, only the input signal and output reg change BUS size accordingly. Figure 4 is a simulated test bench to confirm our MAC model works.



```verilog
module MAC_32bit(ifmap, weights, lastdata, accumulation, en, clk, reset);

input [31:0] ifmap, weights;

input en, clk, reset, lastdata;

output reg [31:0] accumulation;

reg [31:0] temp;

always @(posedge clk)
begin
    if(reset) begin
        accumulation <= 0;
        temp         <= 0;
    end else if (en) begin
        if(lastdata) begin
            accumulation <= (ifmap*weights)+temp;
            temp         <= 0;
        end else
            temp <= (ifmap*weights)+temp;
    end
end

endmodule
```

Figure 3: MAC Model Verilog Code

Figure 3 above shows the verilog code of the 32-bit implementation of the MAC unit.

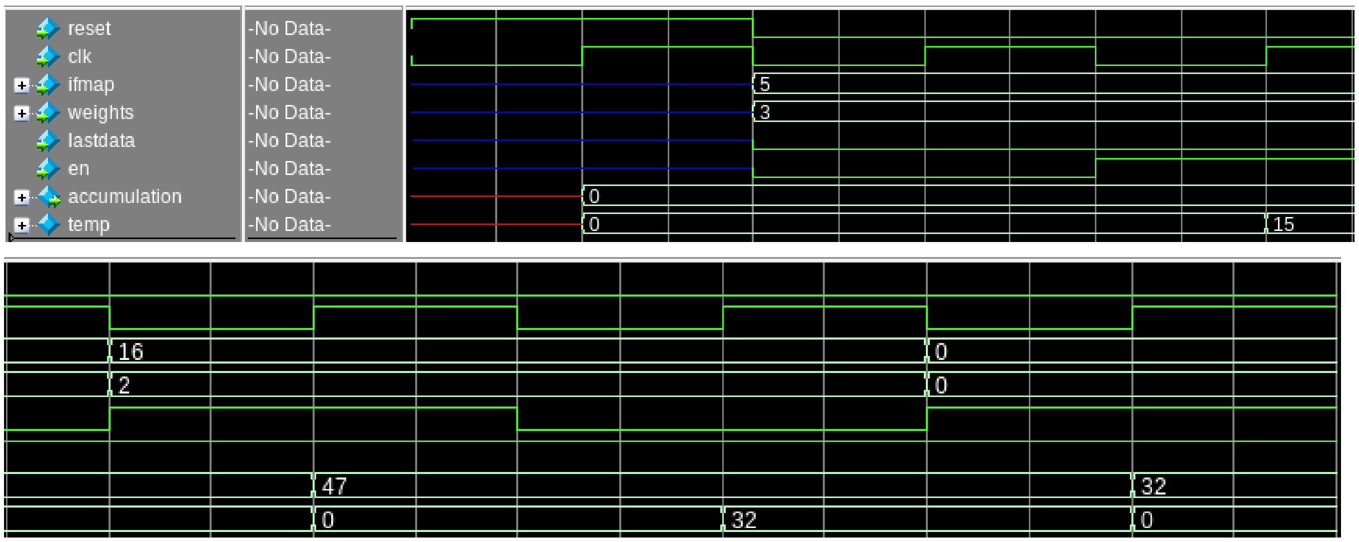


Figure 4: MAC Model Simulation

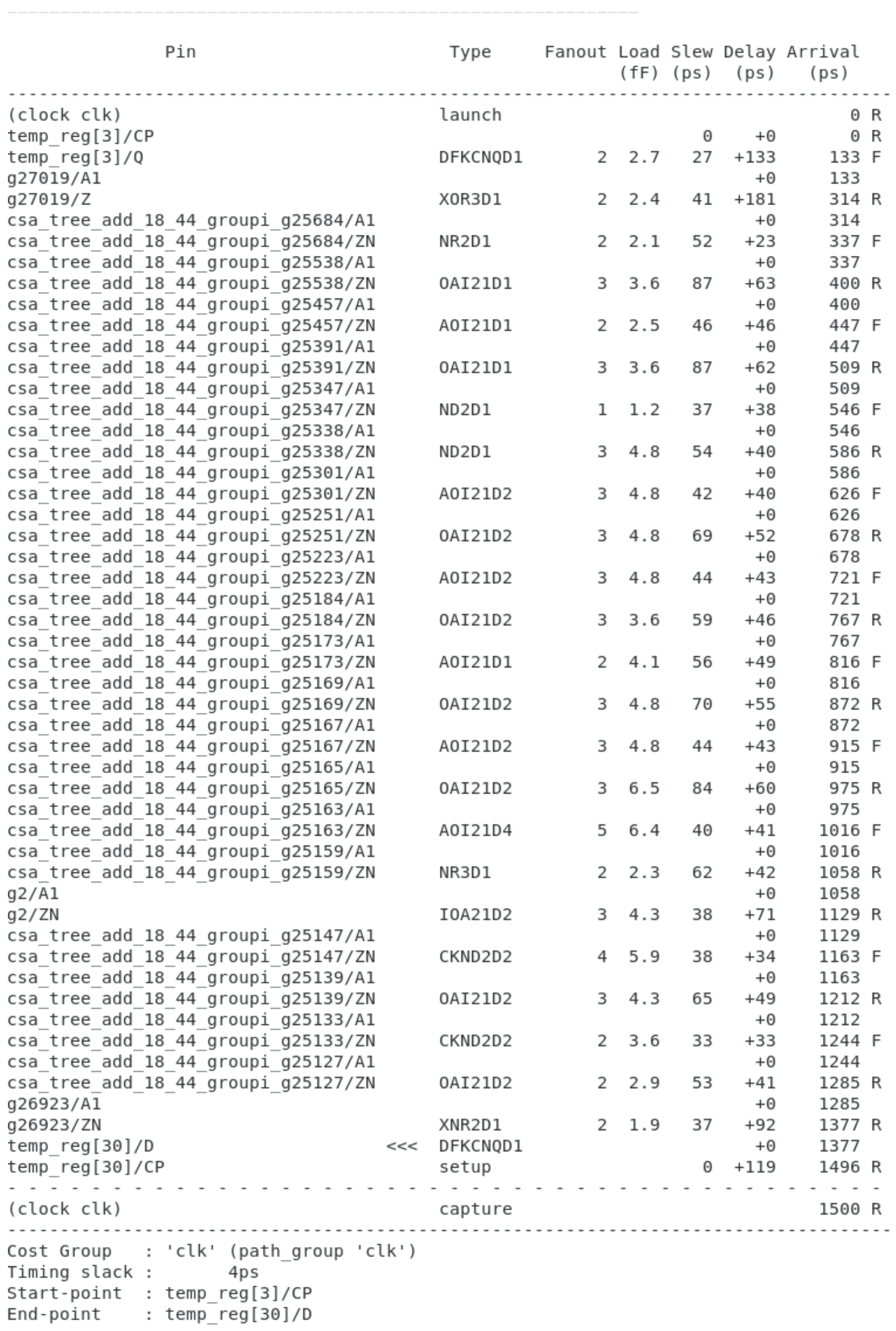
Figure 4 above shows a simple testbench demonstrating that the MAC is working as expected.

Hardware Synthesis Reports:

After we created our circuits in Verilog, we used Genus to synthesize the circuits and run timing, power, and area reports for each. The following pictures and figures show the results we got.

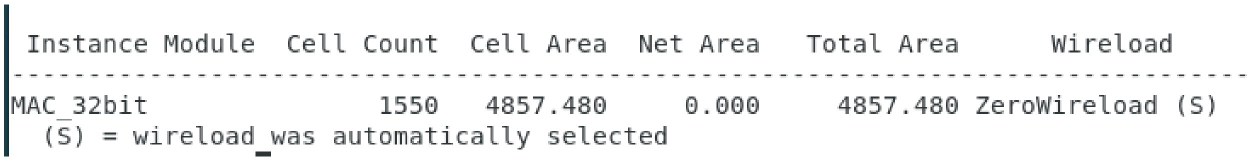
32-bit Implementation:

Clock period: ***1.5 ns***



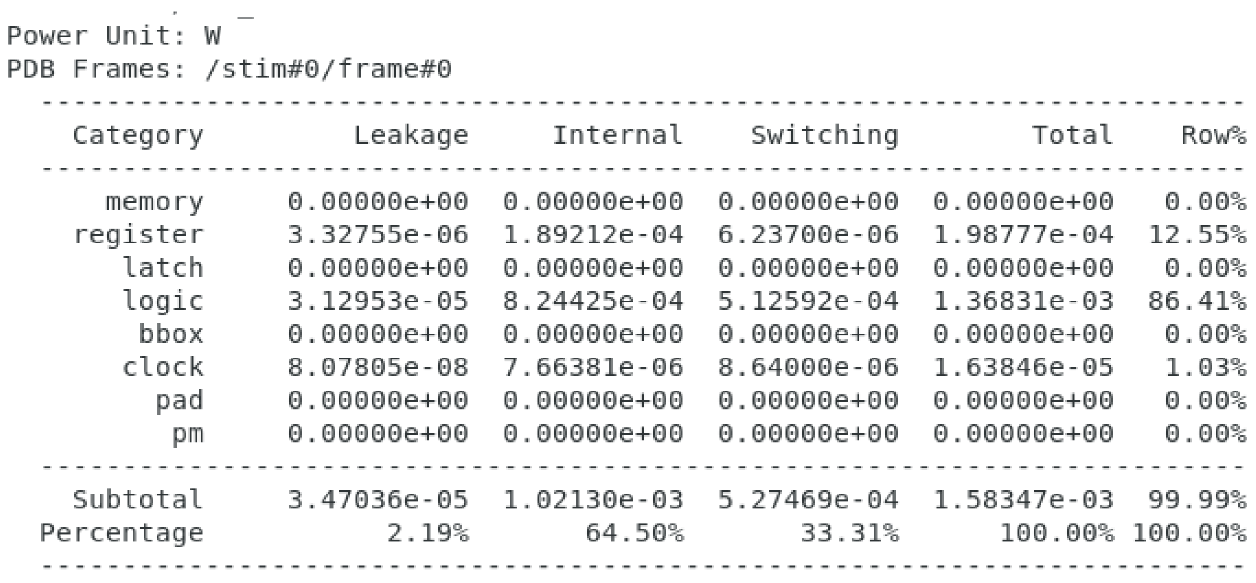
```
              Pin                         Type     Fanout Load Slew Delay Arrival
                                                          (fF) (ps)  (ps)   (ps)
------------------------------------------------------------------------------------
(clock clk)                              launch                                0 R
temp_reg[3]/CP                                                    0    +0      0 R
temp_reg[3]/Q                            DFKCNQD1       2  2.7   27  +133    133 F
g27019/A1                                                              +0    133
g27019/Z                                 XOR3D1         2  2.4   41  +181    314 R
csa_tree_add_18_44_groupi_g25684/A1                                    +0    314
csa_tree_add_18_44_groupi_g25684/ZN      NR2D1          2  2.1   52   +23    337 F
csa_tree_add_18_44_groupi_g25538/A1                                    +0    337
csa_tree_add_18_44_groupi_g25538/ZN      OAI21D1        3  3.6   87   +63    400 R
csa_tree_add_18_44_groupi_g25457/A1                                    +0    400
csa_tree_add_18_44_groupi_g25457/ZN      AOI21D1        2  2.5   46   +46    447 F
csa_tree_add_18_44_groupi_g25391/A1                                    +0    447
csa_tree_add_18_44_groupi_g25391/ZN      OAI21D1        3  3.6   87   +62    509 R
csa_tree_add_18_44_groupi_g25347/A1                                    +0    509
csa_tree_add_18_44_groupi_g25347/ZN      ND2D1          1  1.2   37   +38    546 F
csa_tree_add_18_44_groupi_g25338/A1                                    +0    546
csa_tree_add_18_44_groupi_g25338/ZN      ND2D1          3  4.8   54   +40    586 R
csa_tree_add_18_44_groupi_g25301/A1                                    +0    586
csa_tree_add_18_44_groupi_g25301/ZN      AOI21D2        3  4.8   42   +40    626 F
csa_tree_add_18_44_groupi_g25251/A1                                    +0    626
csa_tree_add_18_44_groupi_g25251/ZN      OAI21D2        3  4.8   69   +52    678 R
csa_tree_add_18_44_groupi_g25223/A1                                    +0    678
csa_tree_add_18_44_groupi_g25223/ZN      AOI21D2        3  4.8   44   +43    721 F
csa_tree_add_18_44_groupi_g25184/A1                                    +0    721
csa_tree_add_18_44_groupi_g25184/ZN      OAI21D2        3  3.6   59   +46    767 R
csa_tree_add_18_44_groupi_g25173/A1                                    +0    767
csa_tree_add_18_44_groupi_g25173/ZN      AOI21D1        2  4.1   56   +49    816 F
csa_tree_add_18_44_groupi_g25169/A1                                    +0    816
csa_tree_add_18_44_groupi_g25169/ZN      OAI21D2        3  4.8   70   +55    872 R
csa_tree_add_18_44_groupi_g25167/A1                                    +0    872
csa_tree_add_18_44_groupi_g25167/ZN      AOI21D2        3  4.8   44   +43    915 F
csa_tree_add_18_44_groupi_g25165/A1                                    +0    915
csa_tree_add_18_44_groupi_g25165/ZN      OAI21D2        3  6.5   84   +60    975 R
csa_tree_add_18_44_groupi_g25163/A1                                    +0    975
csa_tree_add_18_44_groupi_g25163/ZN      AOI21D4        5  6.4   40   +41   1016 F
csa_tree_add_18_44_groupi_g25159/A1                                    +0   1016
csa_tree_add_18_44_groupi_g25159/ZN      NR3D1          2  2.3   62   +42   1058 R
g2/A1                                                                  +0   1058
g2/ZN                                    IOA21D2        3  4.3   38   +71   1129 R
csa_tree_add_18_44_groupi_g25147/A1                                    +0   1129
csa_tree_add_18_44_groupi_g25147/ZN      CKND2D2        4  5.9   38   +34   1163 F
csa_tree_add_18_44_groupi_g25139/A1                                    +0   1163
csa_tree_add_18_44_groupi_g25139/ZN      OAI21D2        3  4.3   65   +49   1212 R
csa_tree_add_18_44_groupi_g25133/A1                                    +0   1212
csa_tree_add_18_44_groupi_g25133/ZN      CKND2D2        2  3.6   33   +33   1244 F
csa_tree_add_18_44_groupi_g25127/A1                                    +0   1244
csa_tree_add_18_44_groupi_g25127/ZN      OAI21D2        2  2.9   53   +41   1285 R
g26923/A1                                                              +0   1285
g26923/ZN                                XNR2D1         2  1.9   37   +92   1377 R
temp_reg[30]/D                      <<<  DFKCNQD1                          +0   1377
temp_reg[30]/CP                          setup                    0  +119   1496 R
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
(clock clk)                              capture                            1500 R
------------------------------------------------------------------------------------
Cost Group   : 'clk' (path_group 'clk')
Timing slack :      4ps
Start-point  : temp_reg[3]/CP
End-point    : temp_reg[30]/D
```

Figure 5: Timing Report for 32-bit Implementation



```
 Instance Module  Cell Count  Cell Area  Net Area   Total Area      Wireload
------------------------------------------------------------------------------
MAC_32bit              1550   4857.480     0.000     4857.480 ZeroWireload (S)
  (S) = wireload was automatically selected
```

Figure 6: Area Report for 32-bit Implementation

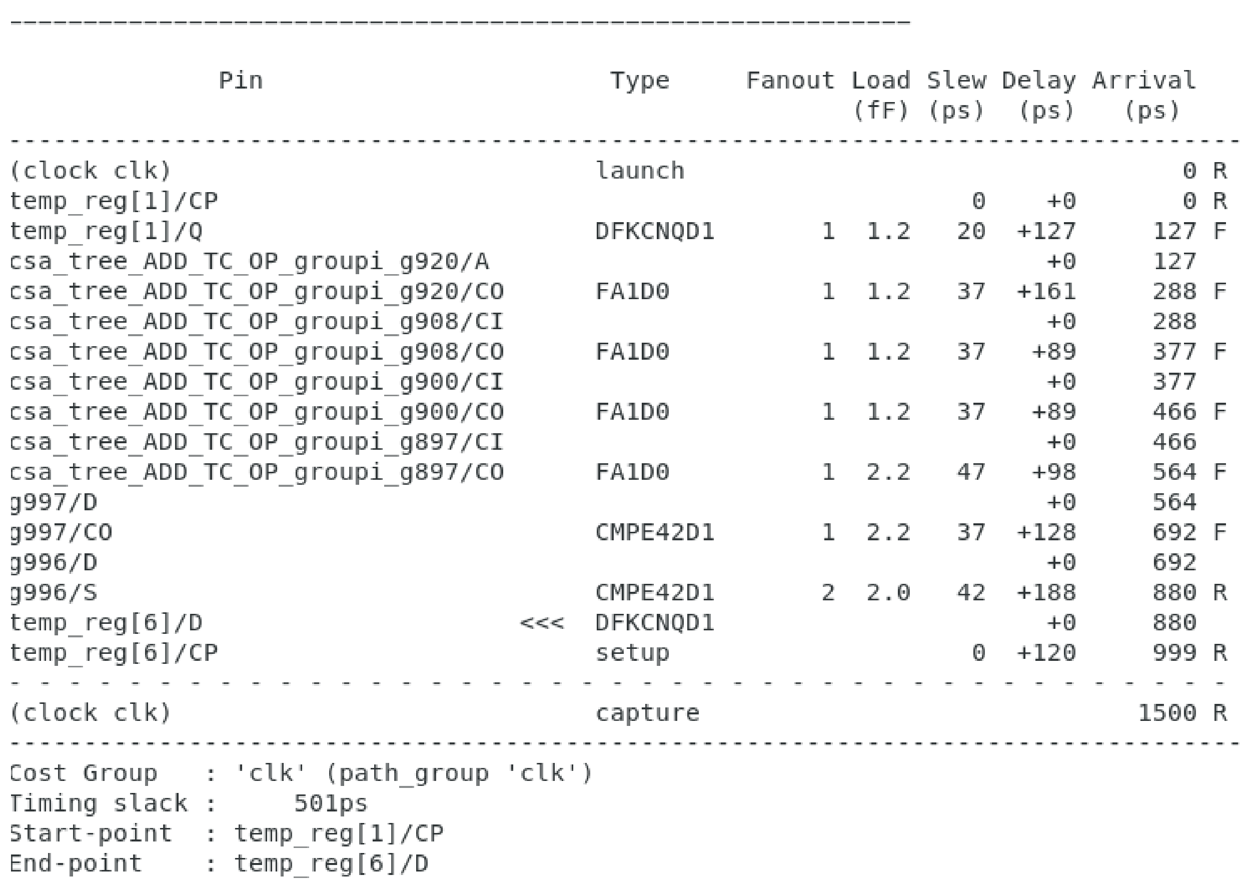


```
Power Unit: W
PDB Frames: /stim#0/frame#0
  -------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  -------------------------------------------------------------------------
      memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
    register     3.32755e-06  1.89212e-04  6.23700e-06  1.98777e-04  12.55%
       latch     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       logic     3.12953e-05  8.24425e-04  5.12592e-04  1.36831e-03  86.41%
        bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       clock     8.07805e-08  7.66381e-06  8.64000e-06  1.63846e-05   1.03%
         pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
          pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  -------------------------------------------------------------------------
    Subtotal     3.47036e-05  1.02130e-03  5.27469e-04  1.58347e-03  99.99%
  Percentage           2.19%       64.50%       33.31%      100.00% 100.00%
  -------------------------------------------------------------------------
```

Figure 7: Power Report for 32-bit Implementation

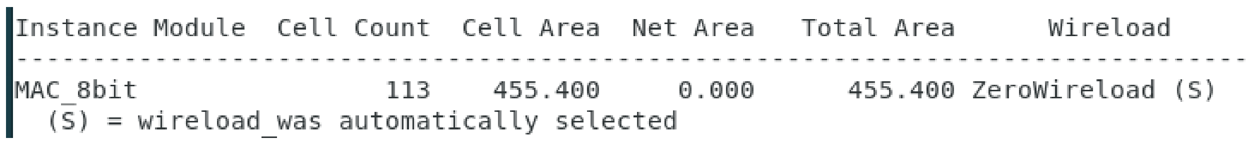
8-bit Implementation:

Clock period: ***1.5 ns***



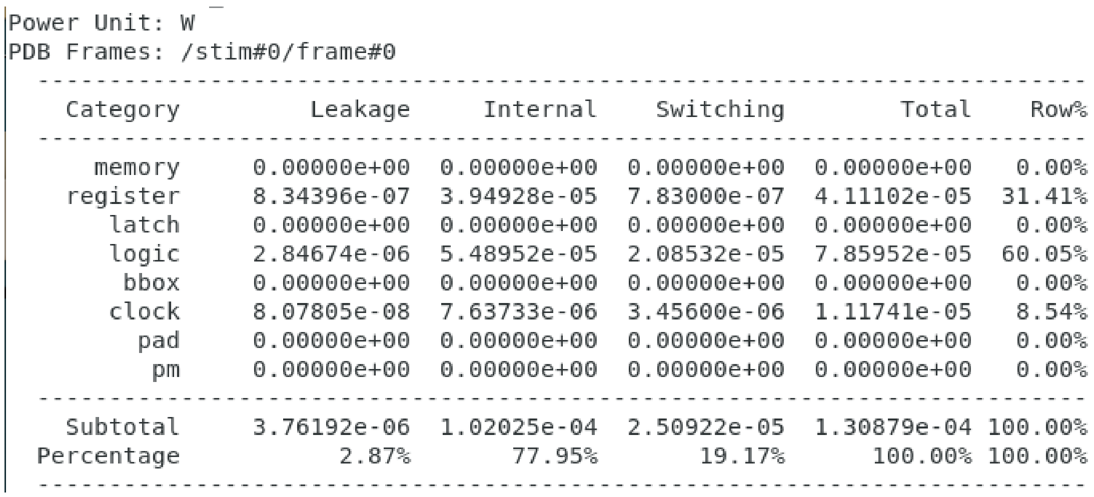
```
------------------------------------------------------------

             Pin                       Type     Fanout Load Slew Delay Arrival
                                                       (fF) (ps)  (ps)   (ps)
--------------------------------------------------------------------------------
(clock clk)                           launch                                 0 R
temp_reg[1]/CP                                                 0    +0       0 R
temp_reg[1]/Q                         DFKCNQD1       1  1.2   20  +127     127 F
csa_tree_ADD_TC_OP_groupi_g920/A                                    +0     127
csa_tree_ADD_TC_OP_groupi_g920/CO     FA1D0          1  1.2   37  +161     288 F
csa_tree_ADD_TC_OP_groupi_g908/CI                                   +0     288
csa_tree_ADD_TC_OP_groupi_g908/CO     FA1D0          1  1.2   37   +89     377 F
csa_tree_ADD_TC_OP_groupi_g900/CI                                   +0     377
csa_tree_ADD_TC_OP_groupi_g900/CO     FA1D0          1  1.2   37   +89     466 F
csa_tree_ADD_TC_OP_groupi_g897/CI                                   +0     466
csa_tree_ADD_TC_OP_groupi_g897/CO     FA1D0          1  2.2   47   +98     564 F
g997/D                                                              +0     564
g997/CO                               CMPE42D1       1  2.2   37  +128     692 F
g996/D                                                              +0     692
g996/S                                CMPE42D1       2  2.0   42  +188     880 R
temp_reg[6]/D                    <<<  DFKCNQD1                      +0     880
temp_reg[6]/CP                        setup                    0  +120     999 R
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
(clock clk)                           capture                             1500 R
--------------------------------------------------------------------------------
Cost Group   : 'clk' (path_group 'clk')
Timing slack :     501ps
Start-point  : temp_reg[1]/CP
End-point    : temp_reg[6]/D
```

Figure 8: Timing Report for 8-bit Implementation



```
Instance Module  Cell Count  Cell Area  Net Area   Total Area      Wireload
-------------------------------------------------------------------------------
MAC_8bit                113    455.400     0.000      455.400 ZeroWireload (S)
  (S) = wireload_was automatically selected
```

Figure 9: Area Report for 8-bit Implementation

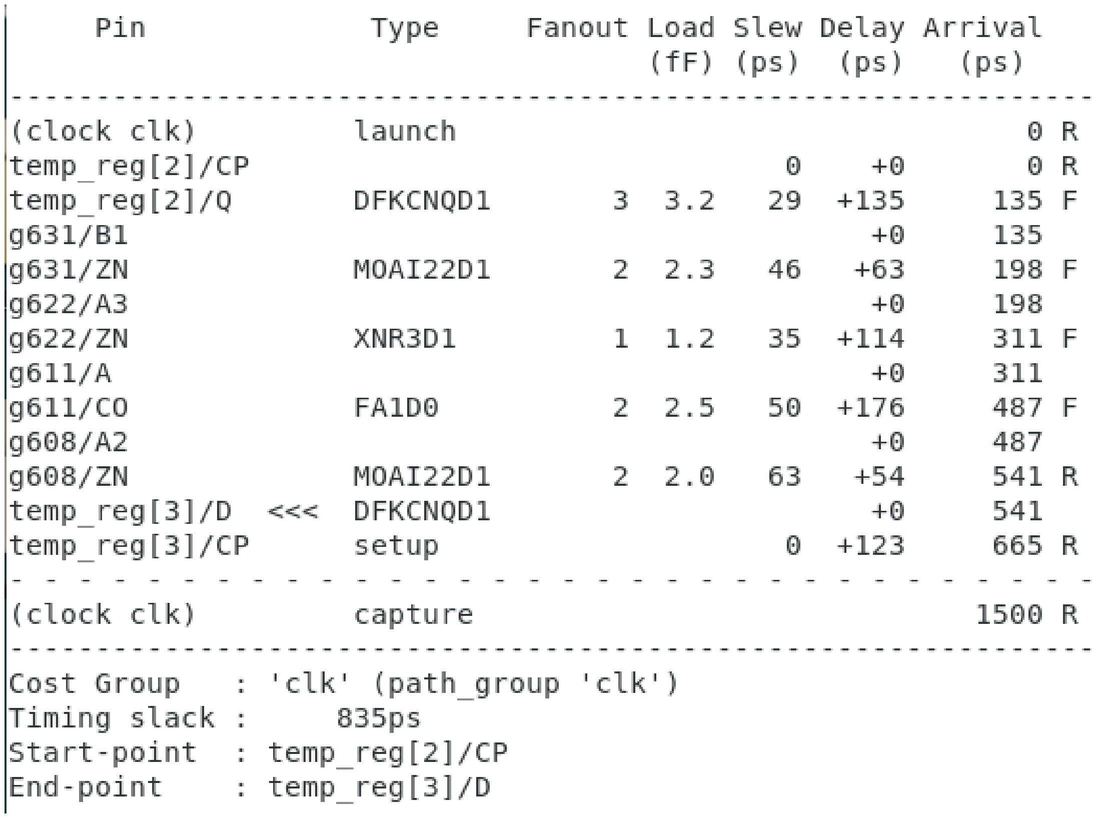


```
Power Unit: W
PDB Frames: /stim#0/frame#0
  -------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  -------------------------------------------------------------------------
      memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
    register     8.34396e-07  3.94928e-05  7.83000e-07  4.11102e-05  31.41%
       latch     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       logic     2.84674e-06  5.48952e-05  2.08532e-05  7.85952e-05  60.05%
        bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       clock     8.07805e-08  7.63733e-06  3.45600e-06  1.11741e-05   8.54%
         pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
          pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  -------------------------------------------------------------------------
    Subtotal     3.76192e-06  1.02025e-04  2.50922e-05  1.30879e-04 100.00%
  Percentage           2.87%       77.95%       19.17%      100.00% 100.00%
  -------------------------------------------------------------------------
```

Figure 10: Power Report for 8-bit Implementation

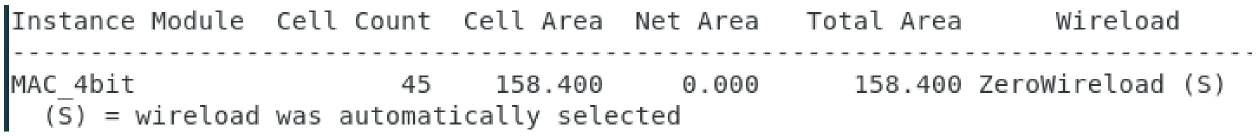
<u>4-bit Implementation:</u>

Clock period: ***1.5 ns***



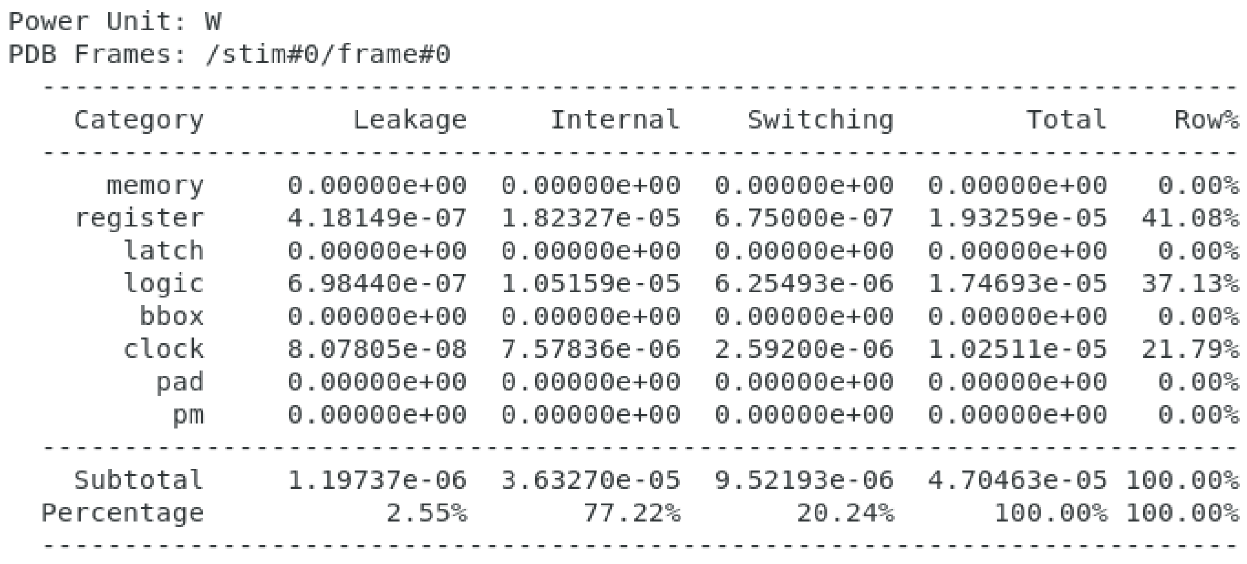
```
     Pin              Type     Fanout Load Slew Delay Arrival
                                      (fF) (ps)  (ps)   (ps)
--------------------------------------------------------------
(clock clk)        launch                                0 R
temp_reg[2]/CP                              0    +0      0 R
temp_reg[2]/Q      DFKCNQD1        3  3.2   29  +135    135 F
g631/B1                                          +0     135
g631/ZN            MOAI22D1        2  2.3   46   +63    198 F
g622/A3                                          +0     198
g622/ZN            XNR3D1          1  1.2   35  +114    311 F
g611/A                                           +0     311
g611/CO            FA1D0           2  2.5   50  +176    487 F
g608/A2                                          +0     487
g608/ZN            MOAI22D1        2  2.0   63   +54    541 R
temp_reg[3]/D  <<< DFKCNQD1                      +0     541
temp_reg[3]/CP     setup                    0   +123    665 R
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
(clock clk)        capture                             1500 R
--------------------------------------------------------------
Cost Group   : 'clk' (path_group 'clk')
Timing slack :     835ps
Start-point  : temp_reg[2]/CP
End-point    : temp_reg[3]/D
```

Figure 11: Timing Report for 4-bit Implementation



```
Instance Module  Cell Count  Cell Area  Net Area   Total Area      Wireload
-------------------------------------------------------------------------
MAC_4bit                 45    158.400     0.000      158.400 ZeroWireload (S)
  (S) = wireload was automatically selected
```

Figure 12: Area Report for 4-bit Implementation



```
Power Unit: W
PDB Frames: /stim#0/frame#0
  ------------------------------------------------------------------------
    Category         Leakage     Internal    Switching        Total    Row%
  ------------------------------------------------------------------------
      memory     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
    register     4.18149e-07  1.82327e-05  6.75000e-07  1.93259e-05  41.08%
       latch     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       logic     6.98440e-07  1.05159e-05  6.25493e-06  1.74693e-05  37.13%
        bbox     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
       clock     8.07805e-08  7.57836e-06  2.59200e-06  1.02511e-05  21.79%
         pad     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
          pm     0.00000e+00  0.00000e+00  0.00000e+00  0.00000e+00   0.00%
  ------------------------------------------------------------------------
    Subtotal     1.19737e-06  3.63270e-05  9.52193e-06  4.70463e-05 100.00%
  Percentage           2.55%       77.22%       20.24%      100.00% 100.00%
  ------------------------------------------------------------------------
```

Figure 13: Power Report for 4-bit Implementation

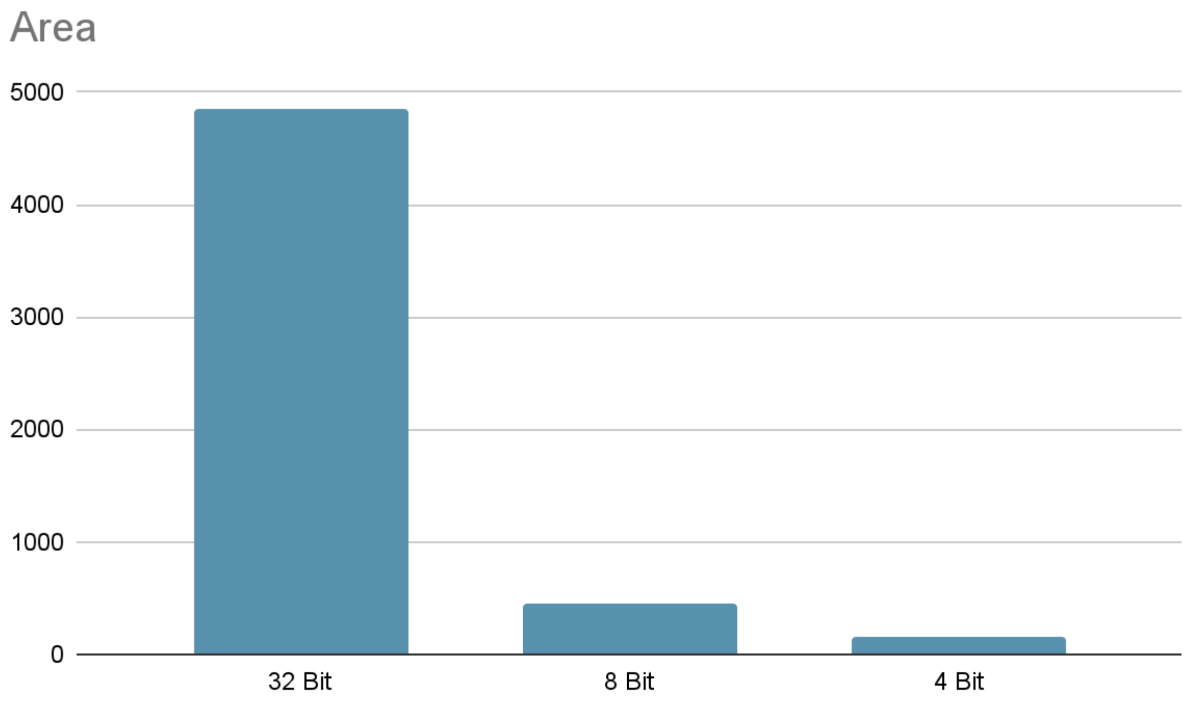


Figure 14: Quantization Area Comparison

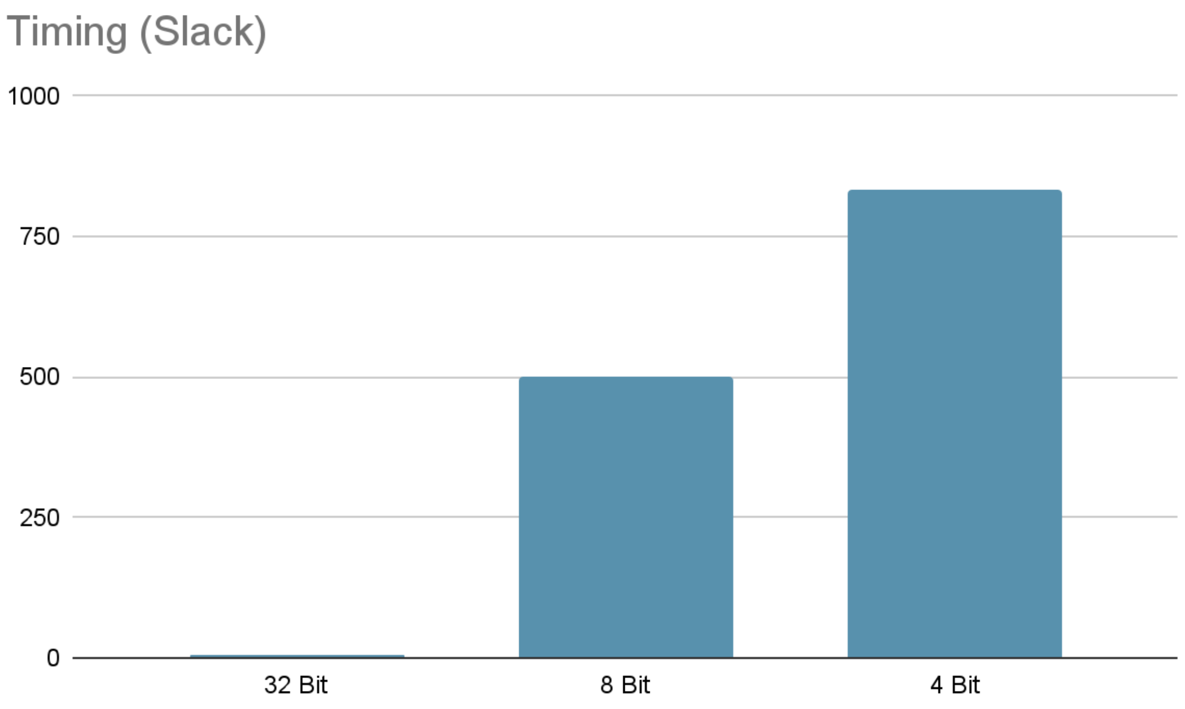


Figure 15: Quantization Timing Comparison

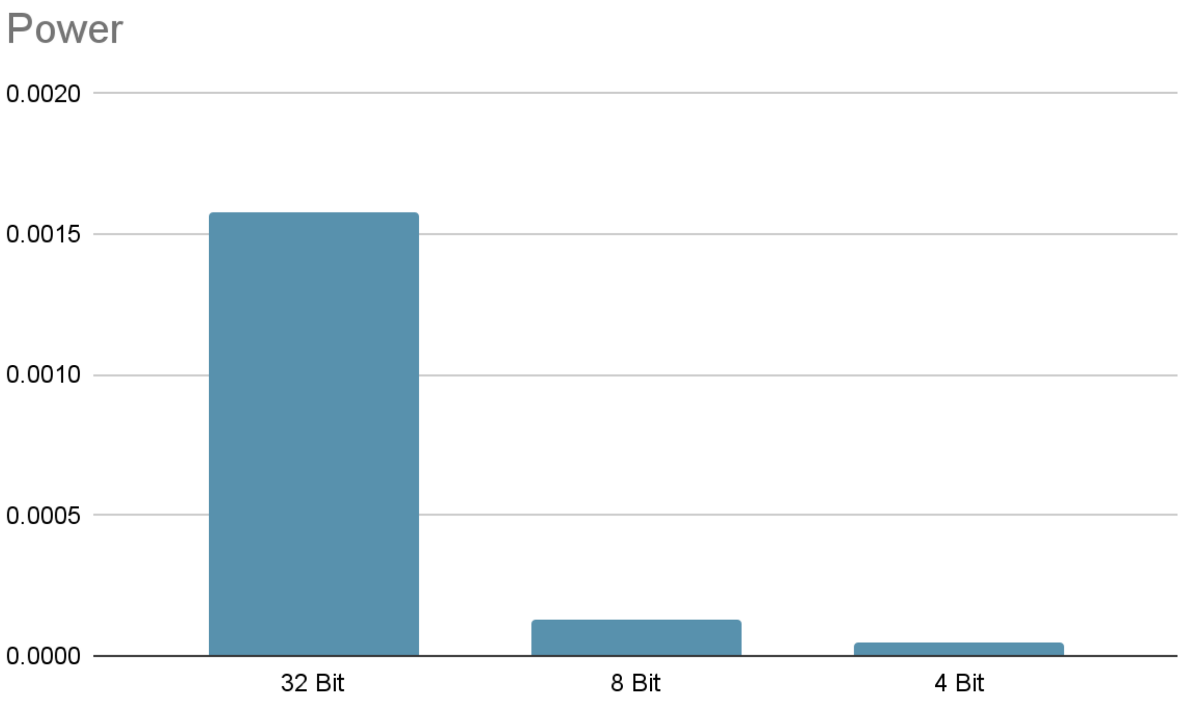


Figure 16: Quantization Power Comparison

Source of Error:
While implementing this project we realized one source of error that did occur: our implemented simple MAC only does fixed point unsigned multiplication and addition. This fits with our quantized 4 and 8 bit numbers, but with 32 bits, we used floating point. This is something that just needs to be kept in mind as this should not affect the accuracy of the project as a whole that much.

Conclusion:
After looking through the data that we collected we came to the conclusion that the 8-bit implementation would be best for a general use DNN. The 8-bit implementation is still fairly accurate while being improved greatly in the hardware implementation with area, timing and power. There is still merit to the other implementations as the 32-bit implementation would be very good if you needed extremely accurate data and were not concerned with any hardware metrics. The 4-bit implementation would be good if you did not need a very accurate system but you had very strict hardware requirements.