**Signal Processing Systems (521279S), Fall 2025**
**Part 1 : Binary number representations**
**Design tasks, deadline for return Thu 6.11.2025 23:59**

Necessary background for the problems is provided during the lectures and in the file **intro1.pdf**. Return your report as a **pdf** file in Moodle, together with the codes requested in Subtask T3. Provide explanations in your answers to get the maximum points!

**T1**. (1p) For this problem, take the values of $r$ and $u$ assigned to your group number in columns 2-3 of the table provided on the last page of this handout.

Your task is to represent the real numbers in the range from $-r$ to $+r$ with a precision that is at least $u$ (ulp), that is, the maximum absolute error is not allowed to be greater than $u/2$. Two kinds of solution to this problem are considered in the following subtasks (a-c) and (d-f).

a) Determine the minimum number of bits (word length) in a two's complement fixed-point format which uses binary point scaling.

b) What is the precision (ulp) of that format?

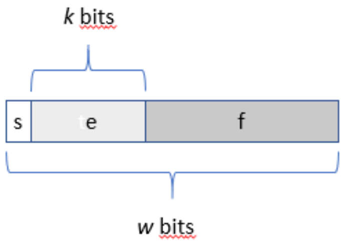
c) Represent the decimal value -6.188 in that format, that is, determine the corresponding bit string. For approximation, use rounding towards zero.

d) Determine the minimum number of bits (word length) in a two's complement fixed-point format that uses slope-bias scaling (bias is set to zero).

e) For that word length, obtain the optimal scaling i.e. the one that gives the best precision. What is that precision (ulp)?

f) Provide the bit string that represents the decimal value +2.611 in that best-precision format. For an approximation, use rounding to nearest.

**T2**. (1.2p) A floating-point arithmetic unit uses a custom format, whose word length is $w$. As usual, the first bit is the sign bit $s$, the next $k$ bits represent an unsigned integer $e$ that encodes the exponent (the range of $e$ is $0 - (2^k - 1)$), and the remaining $w - k - 1$ bits encode the significand $f$. The range of $f$ is $0 \leq f \leq 1 - 2^{-(w-k-1)}$.



Support for two modes is implemented in the arithmetic unit:

- zero mode: when $e = 0$ and $f = 0$, the bit string represents zero ($V = 0$);
- normal mode: when $1 \leq e \leq 2^k - 1$, the bit string corresponds to the real number

  $$V = (-1)^s \times (1 + f) \times 2^{e - e_b},$$

  where $e_b$ is the exponent bias.

Take your group's parameters $(w, k, e_b, S, V)$ from the table on the last page and solve the following subproblems.

a) What is the range and dynamic range for the specified format?

b) In a fixed-point format, how many bits would be needed to achieve the same dynamic range?

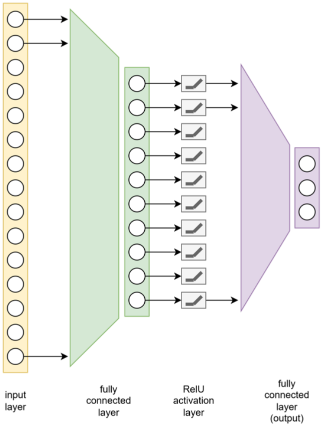
c) What would be the dynamic range, if support for the subnormal mode were added to the system? In the subnormal mode, $e = 0$, $f \neq 0$, and the bit string represents the value

$$V = (-1)^s \times f \times 2^{1 - e_b}.$$

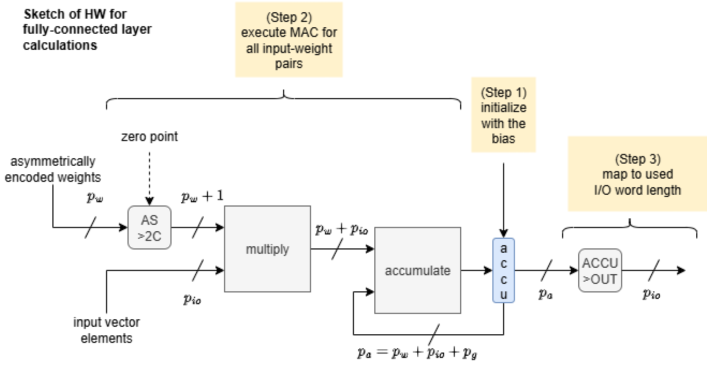
d) Determine the decimal number represented by the bit string $S$.

e) Determine the bit string that provides the best approximation of the value $V$.

**T3**. (1.8p) The task is to consider quantized neural network (NN) implementation.[1] As shown below, NN has three computation layers: (1) fully-connected layer[2] (FC0), (2) a RelU activation[3] layer, and (3) fully-connected output layer (FC1). The dimension of the NN input varies between groups, FC0 and RelU outputs have dimension 10, and FC1 output has dimension 3.[4]



NN is to be implemented on a platform, where fully-connected layer outputs are computed using an 2's complement integer multiply-accumulate (MAC) based hardware shown below. The MAC unit supports asymmetric $p_w$-bit unsigned encoding of the layer weights. The inputs and outputs of NN layers are encoded using $p_{io}$-bit integers and the third parameter defining the word lengths is the number of guard bits for ACCU, $p_g$. To produce layer outputs, the computation is activated 10 times for FC0 layer and 3 times for FC1 layer.


Sketch of HW for fully-connected layer calculations
(Step 2) execute MAC for all input-weight pairs
(Step 1) initialize with the bias
(Step 3) map to used I/O word length
zero point
asymmetrically encoded weights
$p_w$
AS >2C
$p_w + 1$
multiply
$p_w + p_{io}$
accumulate
accu
$p_a$
ACCU >OUT
$p_{io}$
$p_{io}$
input vector elements
$p_a = p_w + p_{io} + p_g$

[1] The problem will be introduced in Lecture 3 (3.11.).

[2] A fully-connected layer performs operation $\mathbf{y} = \mathbf{W}\mathbf{x} + \mathbf{b}$, where $\mathbf{x}$ is the input vector, $\mathbf{W}$ is a weight matrix, and $\mathbf{b}$ is a vector of bias values.

[3] The output of RelU operation is $y = \max(0, x)$, where $x$ denotes the input.

[4] The size of $\mathbf{W}$ is $10 \times d$, where $d$ is the dimension of the input data, for FC0 and $3 \times 10$ for FC1. The size of $\mathbf{b}$ is the same as for layer output $\mathbf{y}$, $10 \times 1$ for FC0 and $3 \times 1$ for FC1.

This computation is demonstrated in a Matlab live script **QNN_comparison.mlx**. The script runs a NN quantization simulator **NN_FCRELUFC_SIM.m** with varying word lengths for the NN weight parameters.

Use your group number as the parameter for the hyperparameters function. This function call can be found on line 3 in the live script.

a) Your task is to implement the NN weight quantization. The function is named **compute_qweights** and it can be found at the end of the **search_asymmetric_parameters**. Remove the call to function **quantizew**, and replace it with your own code. You must also saturate the values if they exceed the limits of the data type used.

*Hint: Before you proceed to the next task, make sure that your quantization works. You can get the correct result by calling the quantizew function on the Matlab command line. Your implementation should produce identical results. See the documentation of the* **compute_qweights** *if you need help calling the quantizew function.*

b) When you have implemented the function in a), run the live script. You should see a warning. Investigate the warning. In your report, explain what the warning is and why it is raised. It is possible to change the parameters of the simulation from the live script file **QNN_comparison.mlx** so that the warning disappears. Find a way to do this. Report what parameters you changed and what the parameters are.

c) The simulation is run with varying bit width for the fully connected layer inputs. This includes the dataset as it is the input for the first fully connected layer. Find the Figure at the end of the Matlab live script where the average difference and standard deviation between the floating point outputs and quantized outputs are illustrated. What is the least amount of bits that will result in a standard deviation of less than $T$ (you can find the $T$ for your group from the Table at the end of this document)?

Return your modified **search_asymmetric_parameters.m** and **QNN_comparison.mlx** files to Moodle together with your pdf report.

Parameters to be used in subtasks T1, T2 and T3.

| Group number | T1 | | T2 | | | | | T3 |
|---|---|---|---|---|---|---|---|---|
| | *r* | *u* | *w* | *k* | $e_b$ | *S* | *V* | *T* |
| 1 | 10 | 0.1 | 11 | 4 | 8 | 10101111010 | 3.096 | 2.50 |
| 2 | 10 | 0.15 | 14 | 5 | 14 | 11100000110100 | 6.689 | 3.09 |
| 3 | 22 | 0.1 | 10 | 5 | 15 | 1101001101 | 7.991 | 3.01 |
| 4 | 26 | 0.15 | 12 | 4 | 7 | 000100111000 | -3.717 | 2.72 |
| 5 | 34 | 0.05 | 10 | 4 | 6 | 1110111101 | 2.404 | 2.25 |
| 6 | 30 | 0.1 | 11 | 4 | 8 | 11000110011 | 4.084 | 3.01 |
| 7 | 38 | 0.15 | 13 | 4 | 6 | 1101010010100 | 7.722 | 3.35 |
| 8 | 18 | 0.15 | 13 | 5 | 15 | 1001011111111 | 4.052 | 2.34 |
| 9 | 14 | 0.05 | 14 | 5 | 16 | 11101100101011 | 2.348 | 1.79 |
| 10 | 30 | 0.2 | 9 | 4 | 6 | 000110011 | -5.414 | 3.85 |
| 11 | 34 | 0.15 | 12 | 5 | 14 | 111100010010 | 5.337 | 3.47 |
| 12 | 42 | 0.15 | 11 | 5 | 14 | 10110101010 | 4.446 | 2.61 |
| 13 | 22 | 0.15 | 10 | 5 | 15 | 1011101010 | 6.690 | 3.11 |
| 14 | 18 | 0.05 | 13 | 4 | 7 | 0001000000101 | -3.333 | 2.64 |
| 15 | 34 | 0.15 | 12 | 4 | 6 | 110110101111 | 4.494 | 3.22 |
| 16 | 34 | 0.05 | 10 | 4 | 7 | 1111011101 | 7.691 | 1.63 |
| 17 | 26 | 0.2 | 13 | 5 | 16 | 0000100111111 | -2.970 | 2.49 |
| 18 | 14 | 0.2 | 12 | 5 | 15 | 010110111101 | -2.600 | 3.16 |
| 19 | 14 | 0.1 | 11 | 5 | 14 | 10110010010 | 7.550 | 3.80 |
| 20 | 10 | 0.05 | 9 | 4 | 8 | 111010000 | 2.297 | 2.44 |
| 21 | 42 | 0.2 | 12 | 4 | 7 | 110011110100 | 2.309 | 3.26 |
| 22 | 30 | 0.05 | 12 | 5 | 14 | 110001001100 | 3.841 | 2.85 |
| 23 | 30 | 0.15 | 12 | 5 | 14 | 101010101111 | 2.390 | 3.61 |
| 24 | 18 | 0.2 | 11 | 5 | 14 | 10100101011 | 3.183 | 1.57 |
| 25 | 10 | 0.1 | 9 | 4 | 6 | 111011010 | 2.564 | 2.09 |
| 26 | 30 | 0.1 | 11 | 4 | 8 | 11000110011 | 4.084 | 2.67 |
| 27 | 38 | 0.15 | 13 | 4 | 6 | 1101010010100 | 7.722 | 2.75 |
| 28 | 18 | 0.15 | 13 | 5 | 15 | 1001011111111 | 4.052 | 1.42 |
| 29 | 14 | 0.05 | 14 | 5 | 16 | 11101100101011 | 2.348 | 2.30 |
| 30 | 30 | 0.2 | 9 | 4 | 6 | 000110011 | -5.414 | 2.81 |