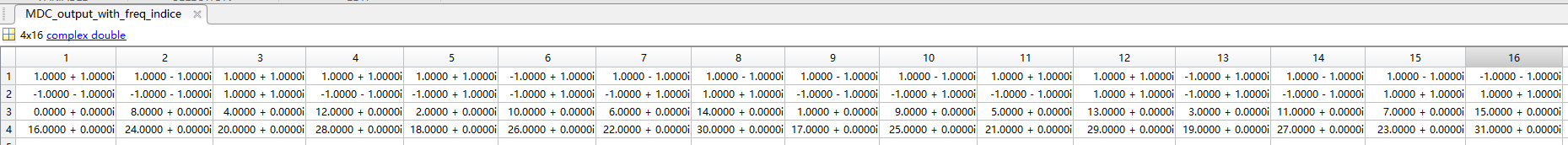
DSP in VLSI

HW5

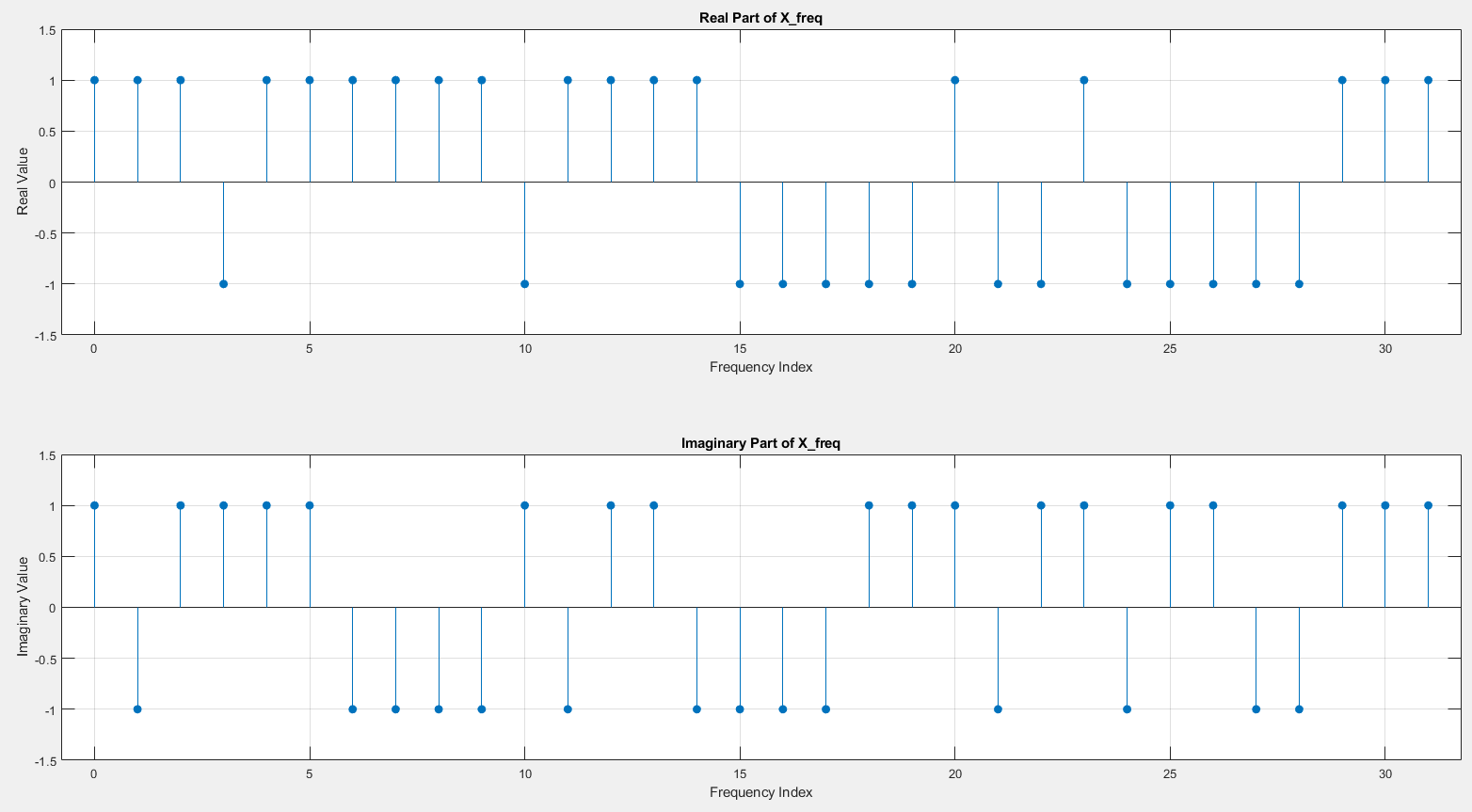
電子所ICS組, R13943015, 張根齊

1. (Step 1) Please use FFTInput32.mat as the inputs. Use your MDC FFT Matlab/Python program to generate the 32 FFT outputs. The program outputs should be in a 2 × 16 array. Because they are in bit-reversed order, write a program to generate their associated frequency domain indices. Now, you have an 4 × 16 array containing the frequency domain indices and the MDC outputs. List the array. (Maybe you can copy the results in Work Space and paste onto your report.) (10%)  
row 1 and row 2 are 32 FFT outputs (complexed number). And row3, row4 are corresponding frequency domain indices. (integer number)

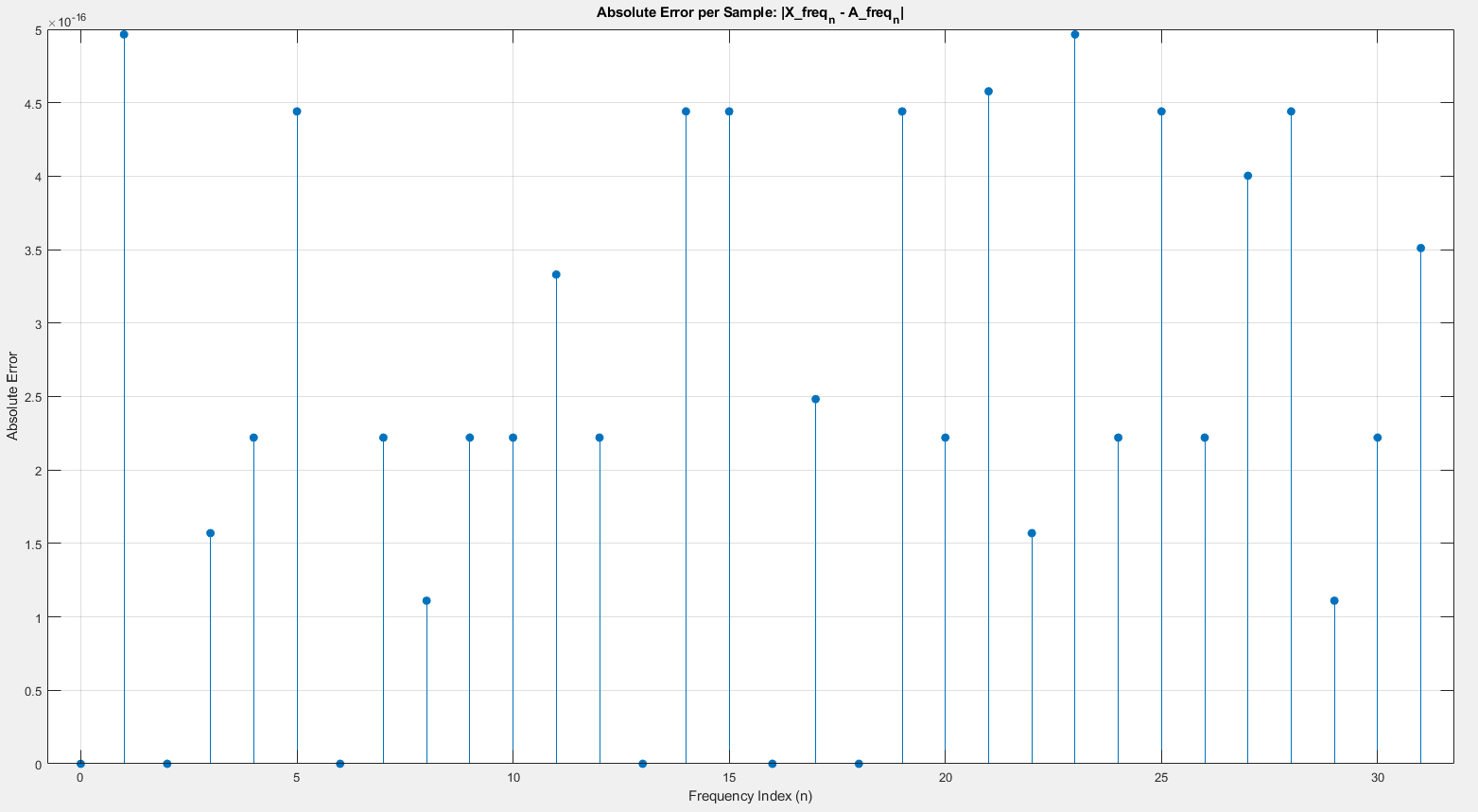


2. (Step 2) Please use FFTInput32.mat as the inputs. Show that your program for output re-ordering is correct. Draw the real part and imaginary part of 𝑋0~𝑋31. (10%) Compare the results of 𝑋0~𝑋31 to A0~A31 . Draw the absolute error of each sample. The error should be small than 10^-10 because of double-precision floating-point operations. If the error is not acceptable, the first 20 points will be deducted, too. (10%)

2.1 Draw the real part and imaginary part of 𝑋0~𝑋31



2.2 Draw the absolute error of each sample



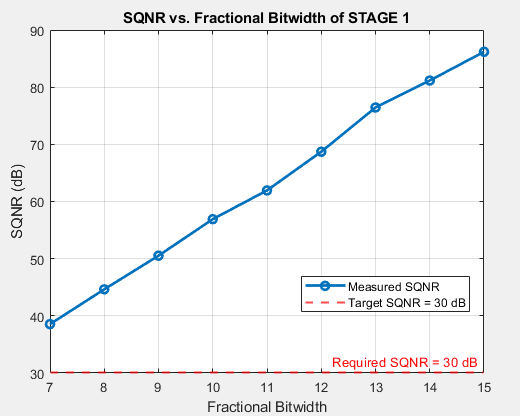
Average error (< 10^-10):



3. (Step 3) Generate your own inputs of 96 samples. Draw the SQNR versus fractional part word-length N stage by stage. (30%)

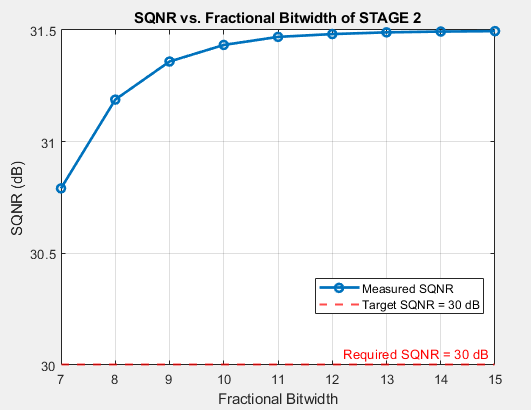
(a) Quantize the first stage. Evaluate the FFT output SQNR versus fractional part word-length N for N=7, 8, 9, 10, …, 13, 14,15 for stage 1. Draw the figure.

Choose stage 1 fractional bit-width = 10 (reserved margin).



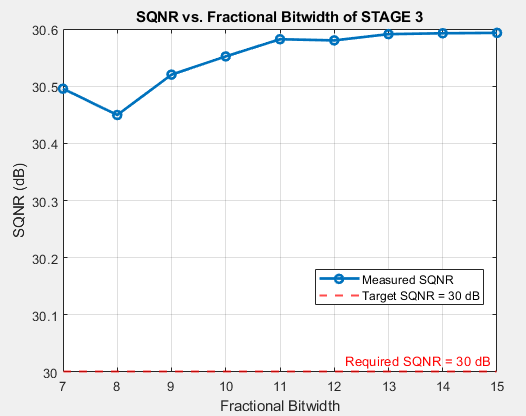
(b) Fix the setting for stage 1. Quantize the second stage. Evaluate the FFT output SQNR versus fractional part word-length N for N=7, 8, 9, 10,…, 13, 14,15 for stage 2. Draw the figure.

Choose stage 2 fractional bit-width = 10 (reserved margin):



(c) Fix the setting for stage 1 and 2. Quantize the third stage. Evaluate the FFT output SQNR versus fractional part word-length N for N=7, 8, 9, 10,…, 13,14,15 for stage 3 and so on. Draw the figure

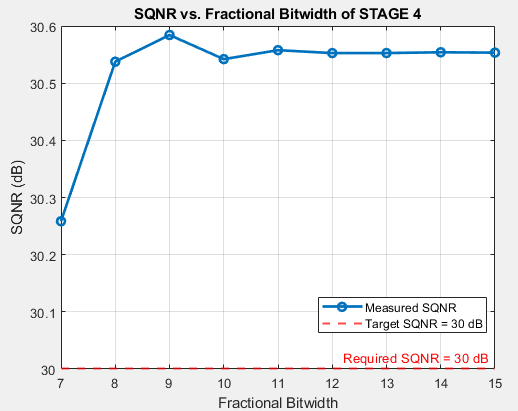
Choose stage 3 fractional bit-width = 10 (reserved margin):



(d) Repeat until stage 5. So, you have five figures.

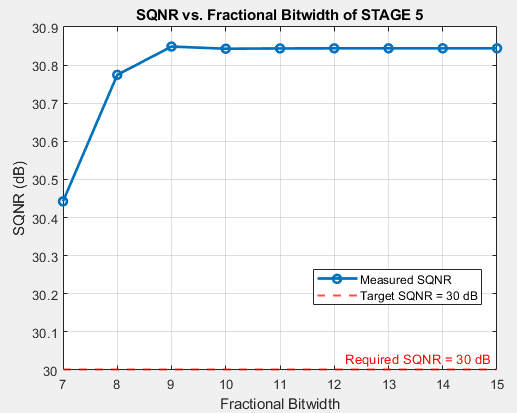
d-1: stage 4:

Choose stage 4 fractional bit-width = 10 (reserved margin):



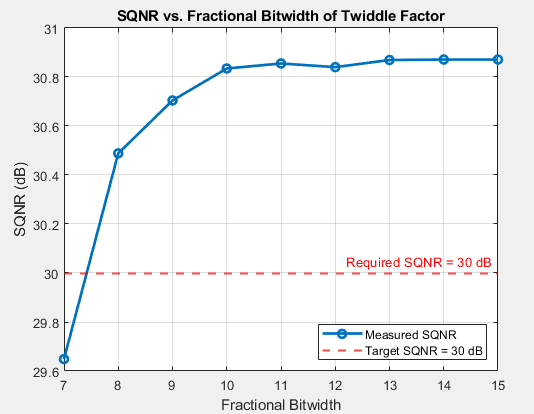
d-2: stage 5:

Choose stage 5 fractional bit-width = 10 (reserved margin):



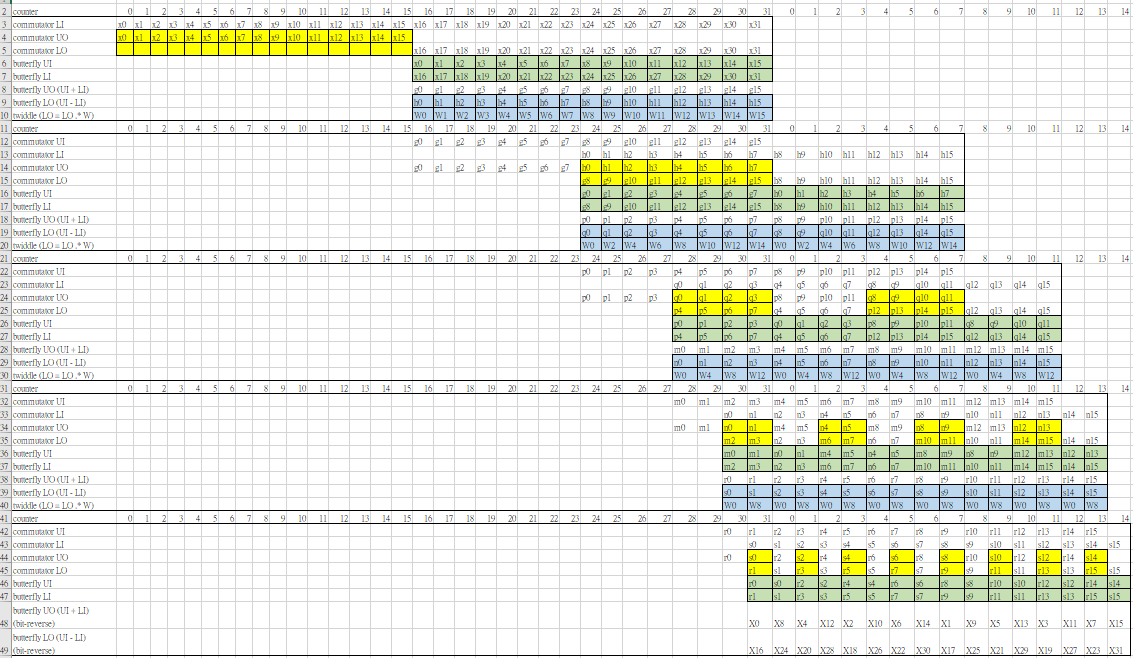
(e) Then, decide the fractional part word-length for twiddle factors of all the stages. Draw the FFT output SQNR versus fractional part word-length of twiddle factors.

Choose twiddle factors fractional bit-width = 10:



4. (Step 4) Now, use a 5-bit counter counting from 0 to 31, which is synchronized to the input index. Generate the control signals of commutator and butterfly units for each stage. Show how you generate them and why? (20%)

The overall schedule is shown below (The file is included in the submitted assignment package). The yellow sections indicate that the control signal for the commutator should be set to switch mode; otherwise, it should be in bypass mode. The green sections indicate that the control signal for the butterfly should be set to computation mode; otherwise, it should be in bypass mode.



4.1. Stage 1

4.1.1 Commutator

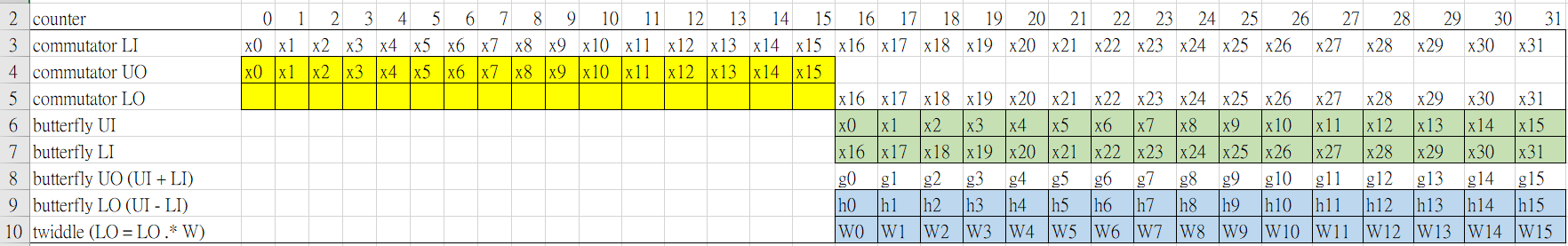
Counter = 0 ~ 15: switch mode

else: bypass mode

4.1.2 Butterfly

Counter = 16 ~ 31: computation mode

else: bypass mode



4.2. Stage 2

4.2.1 Commutator

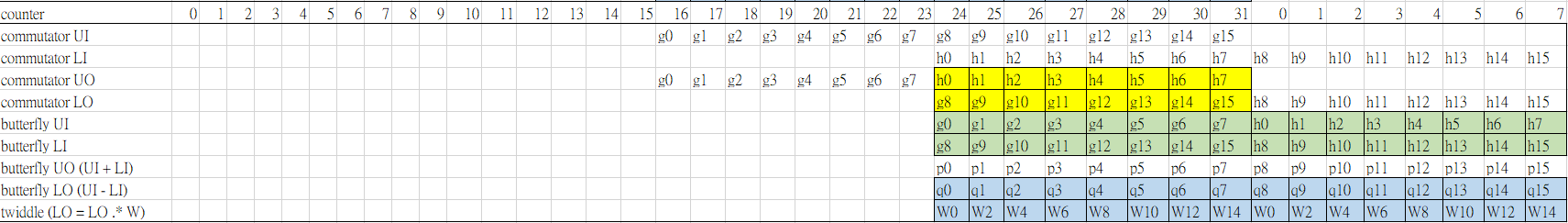
Counter = 24 ~ 31: switch mode

else: bypass mode

4.2.2 Butterfly

Counter = 24 ~ 31, 0 ~ 7: computation mode

else: bypass mode



4.3. Stage 3

4.3.1 Commutator

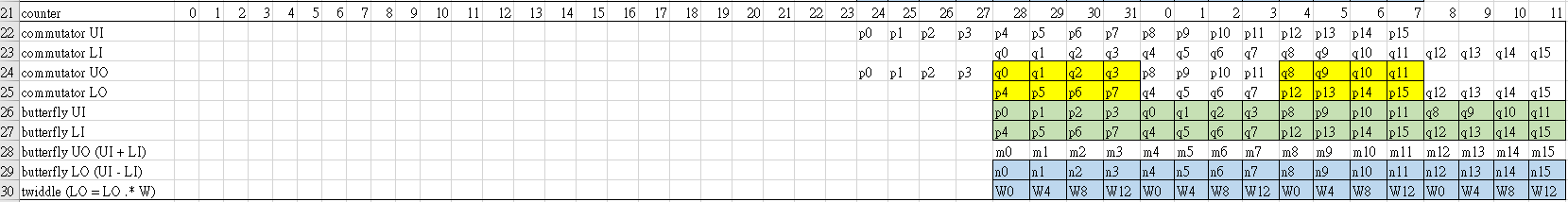
Counter = 28 ~ 31, 4 ~ 7: switch mode

else: bypass mode

4.3.2 Butterfly

Counter = 28 ~ 31, 0 ~ 11: computation mode

else: bypass mode



4.4. Stage 4

4.4.1 Commutator

Counter = 30 ~ 31, 2 ~ 3, 6 ~ 7, 10 ~ 11: switch mode

else: bypass mode

4.4.2 Butterfly

Counter = 30 ~ 31, 0 ~ 13: computation mode

else: bypass mode



4.5. Stage 5

4.5.1 Commutator

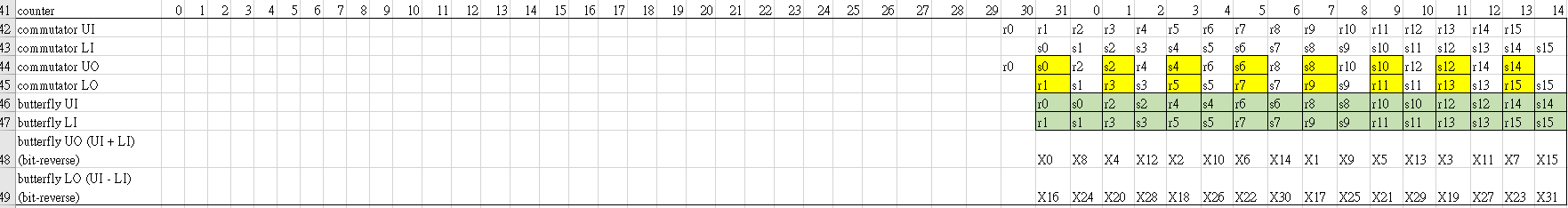
Counter = 31, 1, 3, 5, 7, 9, 11, 13: switch mode

else: bypass mode

4.5.2 Butterfly

Counter = 31, 0 ~ 14: computation mode

else: bypass mode



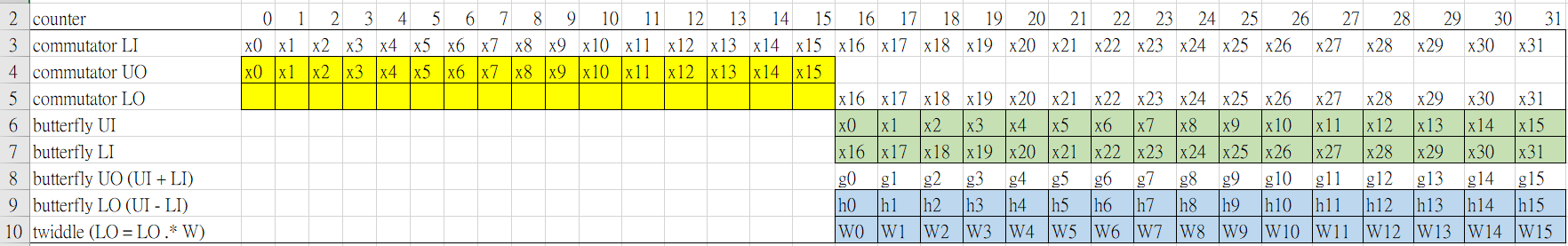
5. (Step 5) Generate the control signals of complex multipliers for each stage. Show how you generate them and why? (10%)

5.1. Stage 1

5.1.1 Multipliers

Counter = 16 ~ 31: Multiplication mode

else: bypass mode

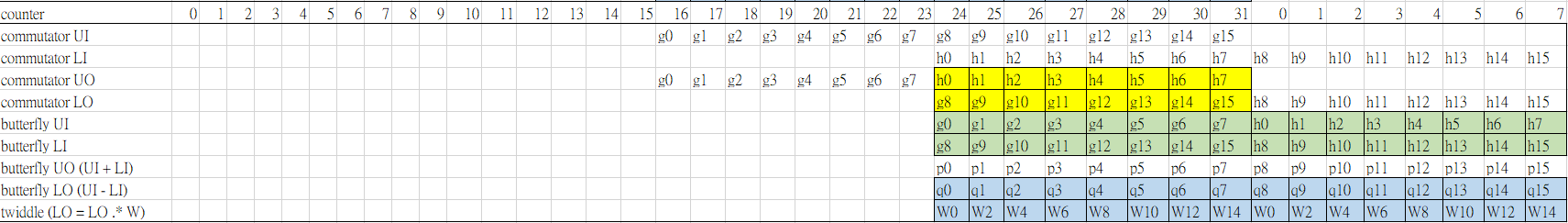


5.2. Stage 2

5.2.1 Multipliers

Counter = 24 ~ 31, 0 ~ 7: Multiplication mode

else: bypass mode

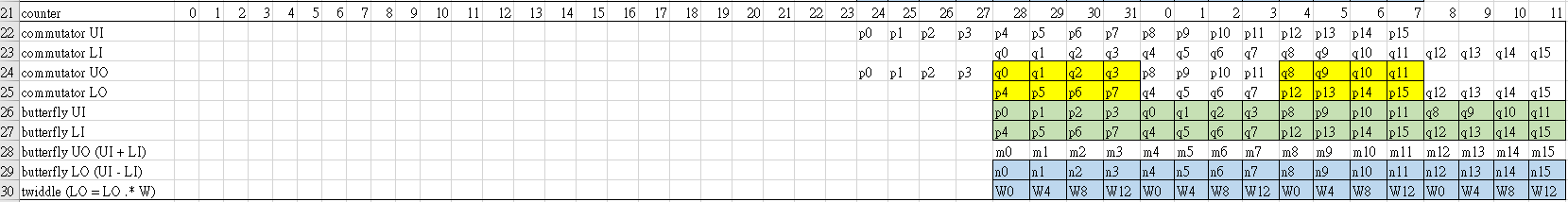


5.3. Stage 3

5.3.1 Multipliers

Counter = 28 ~ 31, 0 ~ 11: Multiplication mode

else: bypass mode



5.4. Stage 4

5.4.1 Multipliers

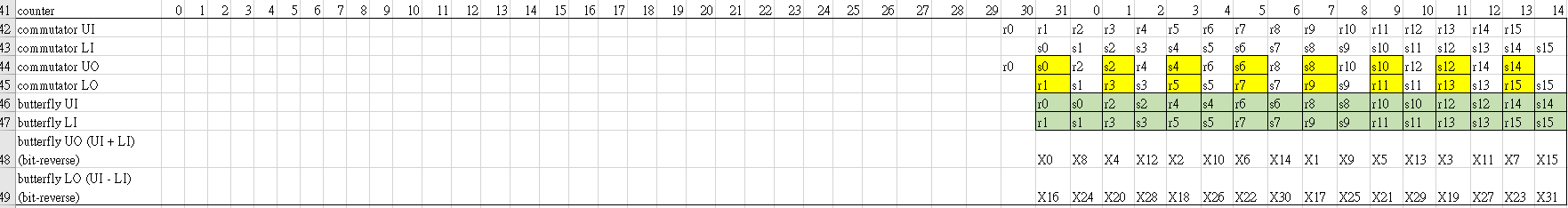
Counter = 30 ~ 31, 0 ~ 13: Multiplication mode

else: bypass mode



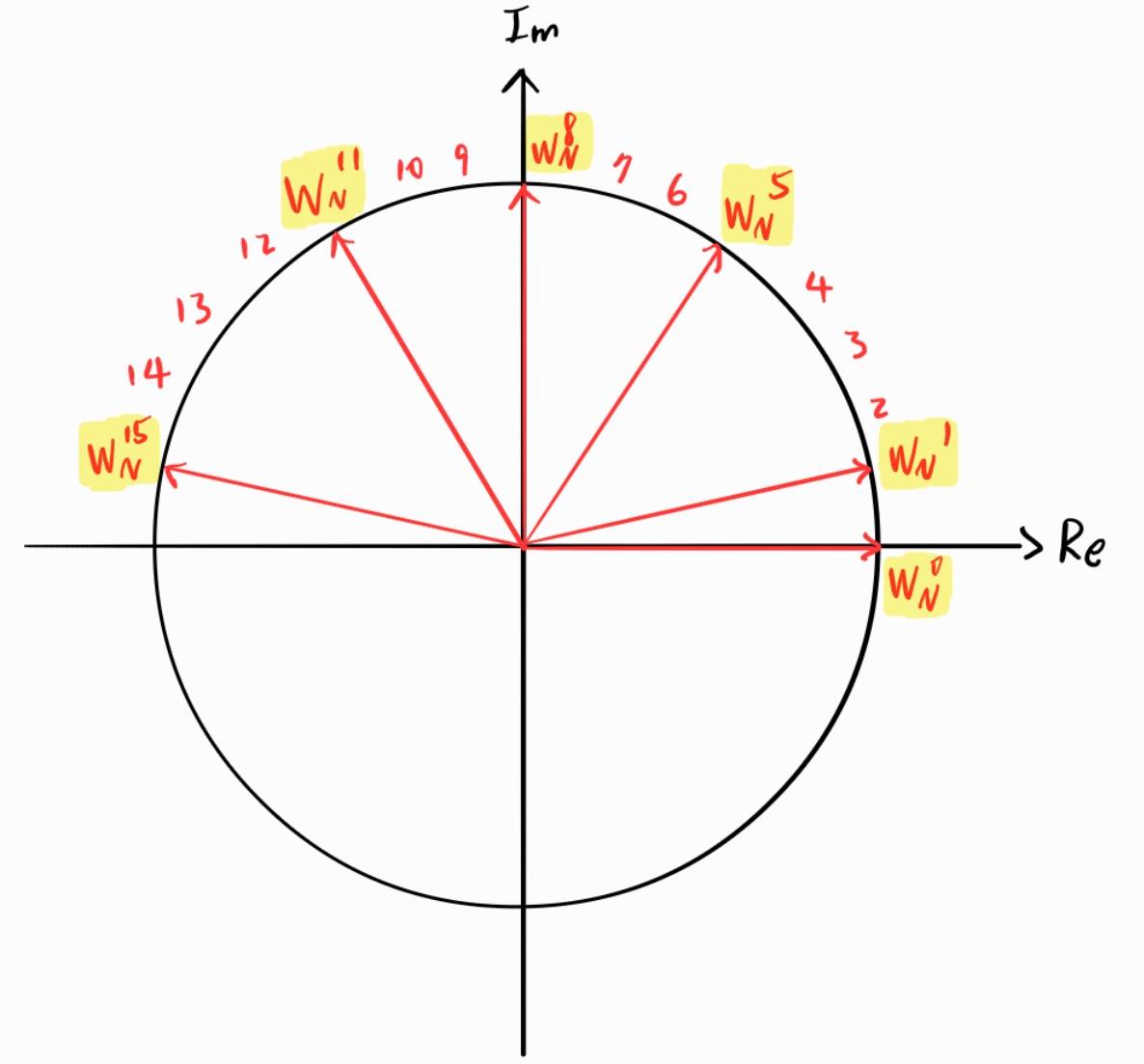
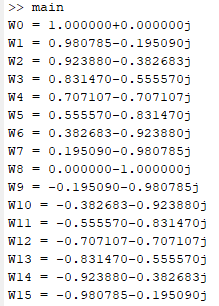
4.5. Stage 5

4.5.1 Multipliers: bypass mode (no need to use a multiplier)

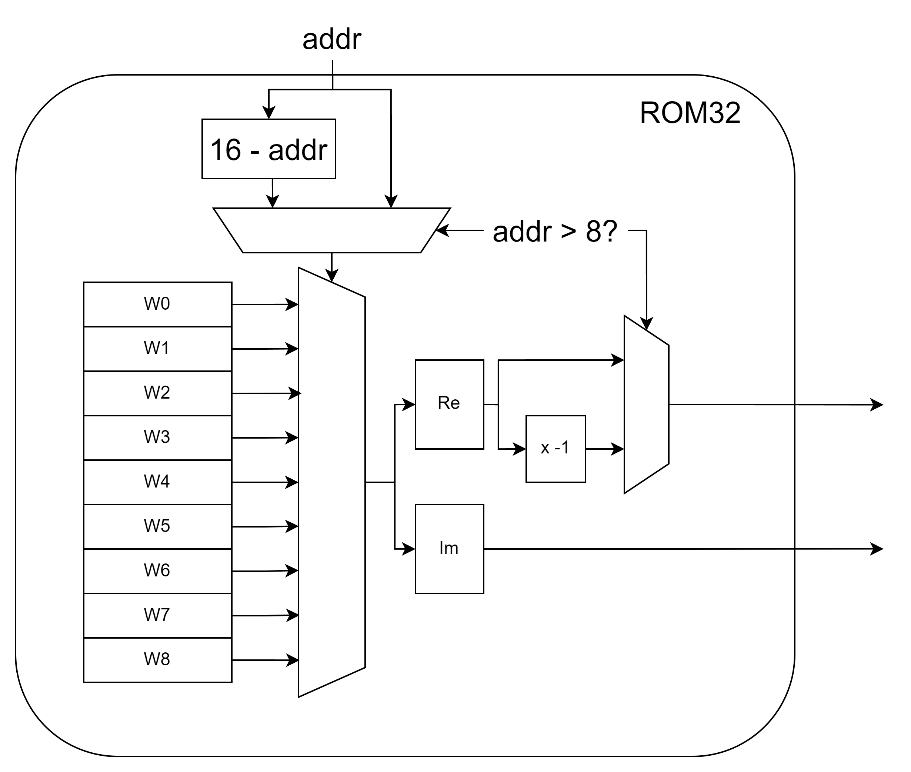


6. (Step 6) Explain the way that you use the sine/cosine values in the first quadrant to generate the required 32 phases for ROM 32. Draw the block diagram (10%)

As shown in the figure below, we can observe that the values in the first quadrant (W1 to W7) are the same as those in the second quadrant (W15 to W9), except for the sign of the real part. Ex: . So, we only need to save the values in the first quadrant.

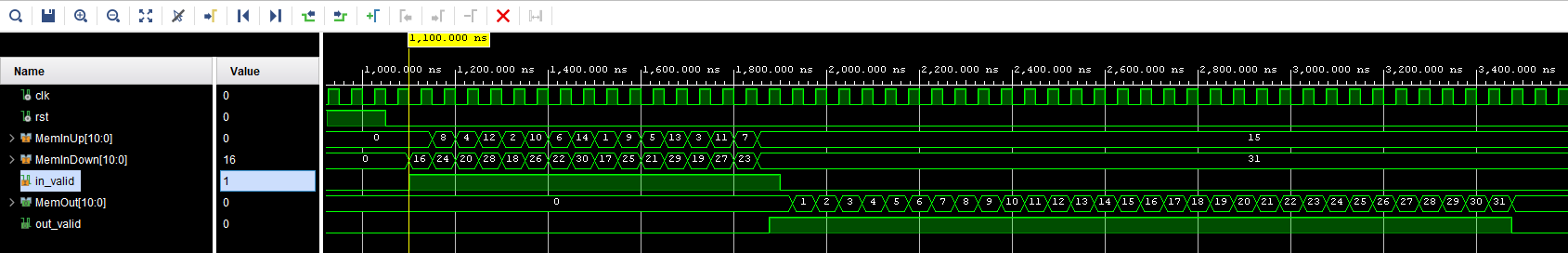
 

6.1. Block diagram

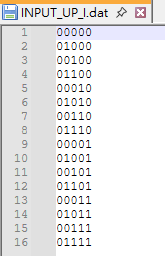
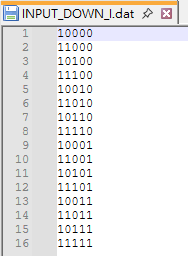


7. (Step 7) Show the timing diagram of behavior simulation for bit-reverse re-ordering with ping-pong accessed bit-reversal buffer as in Fig. 9. (15%)

7.1 timing diagram:



7.2 test pattern: MemInUp & MemInDown

8. (Step 8) Please use FFTInput32.mat as the inputs. Quantize them using the word-length selected by yourself. Show the timing diagram of behavior simulation for MDC FFT. Compared the results to your answer in Q1(Step 1). Draw the error for 32 samples of real part and imaginary part. (25%)