FFT Assignment (pt.1)

Tasks:

- Required to deliver an RTL of the butterfly unit
- The twiddle factor ROM
- The implementation in fixed point representation format

1) The Butterfly unit

As can be seen from the below graph (figure 1), FFT butterfly signal flow consists of multiple instantiations of parallel and sequential butterfly unit with 2 input and 2 outputs as can be seen from (figure 2).

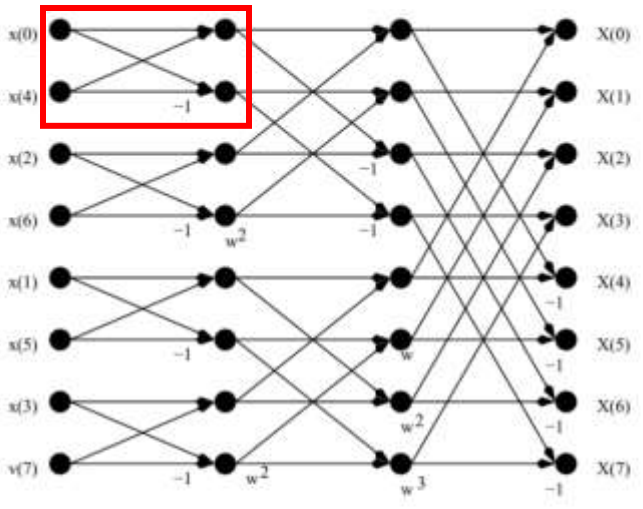


Figure 1: signal flow of an 8-pt radix 2 fft DIT Showing that butterfly unit is the building block.

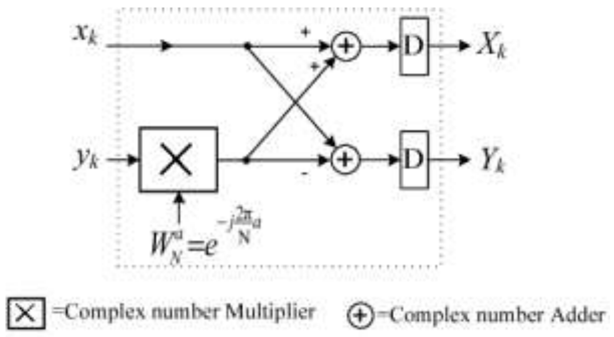
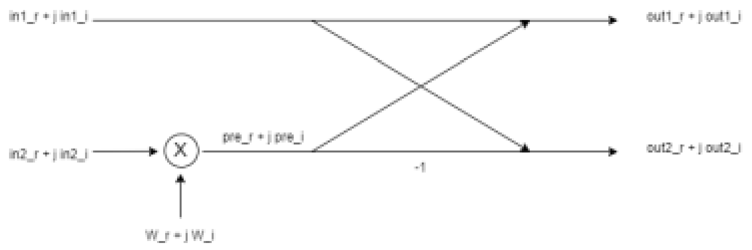


Figure 2: the butterfly unit

So, the first task was to implement this butterfly unit on Verilog to be the building block for the actual fft block (with any N-pt fft implementation).

The implementation of the butterfly was simple as it’s just a number of addition and multiplication done on complex number. The complex numbers were represented in two registers (variables) one for the real part and the other for the imaginary part. The following set of equations describes the operations of the butterfly unit.



$$pre_r + jpre_i = (in2_r + jin2_i)(W_r + jW_i) = (in2_rW_r - in2_iW_i) + j(in2_rW_i + in2_iW_r)$$

$$pre_r = (in2_rW_r - in2_iW_i) \qquad pre_i = (in2_rW_i + in2_iW_r)$$

$$out1_r + jout1_i = (in1_r + pre_r) + j(in1_i + pre_i)$$

$$out2_r + jout2_i = (in1_r - pre_r) + j(in1_i - pre_i)$$

2) Twiddle factor ROM

Twiddle factor is simple a complex exponent that is multiplied by the input to achieve the DFT equation. It's notated by $W_N^i = e^{\frac{-j2\pi i}{N}}$. When we represent the twiddle factor we represent it as a complex number using Euler formula $e^{\frac{-j2\pi i}{N}} = cos\frac{2\pi i}{N} - \mathrm{j}\sin\frac{2\pi i}{N}$. To be able to generate the values of the twiddle factor we use Matlab script to generate the values and store it in a text file where we can extract the data from the file in the ROM using $readmemb command.

3) Fixed point representation

Since we are dealing with real numbers we need to find a way to represent those numbers in binary representation. There's mainly two ways, the floating point which is very sophisticated and high accurate but it's more complex and uses huge resources to store and decode. The other is the fixed point where we simply represent the number in binary as we do in decimal with integral bits and fractional bits as shown in (figure 4). The advantage of this method is that we can do addition and multiplication simply as we do in normal integer representation.

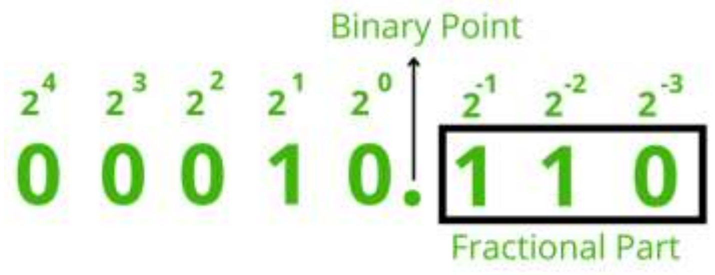


Figure 4: the weights of bits in the fixed-point representation.

4) Deliverables

1. RTL Verilog code executes the butterfly unit diagram
2. ROM implementation for the twiddle factor
3. Simple implementation of 16-point DIT FFT in Matlab and Verilog
4. Matlab script generating the twiddle factor, the input vectors and the output references.
5. Testbench for the butterfly unit
6. Testbench for the 16-point fft module
7. Matlab scripts implementing 2^n input DIT & DIF fft.

5) Notes

1. I was wondering about how I will use the twiddle factor rom in the fft module while it contains only one output port so it will deliver a single output at a time at the time we need more outputs.
2. To execute the butterfly_tb module you need first to run the script Reference/butterfly_unit.m and generate the text files that will be used by TB.
3. To execute the FFT_tb module you need first to run the script Reference/FFT_16_DIT.m for the same purpose
4. the accuracy in implementing the butterfly unit is 4 bits but in the FFT implementation it reaches 6 bits due to the accumulation of error along the stages.
5. The fft implementation doesn't include registers or clock dealing with it as a long combinational chain.
6. I've made simple generic matlab code to implement both DIT and DIF fft in any number of 2^n inputs which will be the base for building a generic fft module.

6) Future works

1. Trying to increase the accuracy of the implementation by modifying the precision of the fixed-point representation.
2. Implement a sophisticated pipelined algorithm of fft.
3. Solve the ROM problem.