**Computer Hardware Experiments**

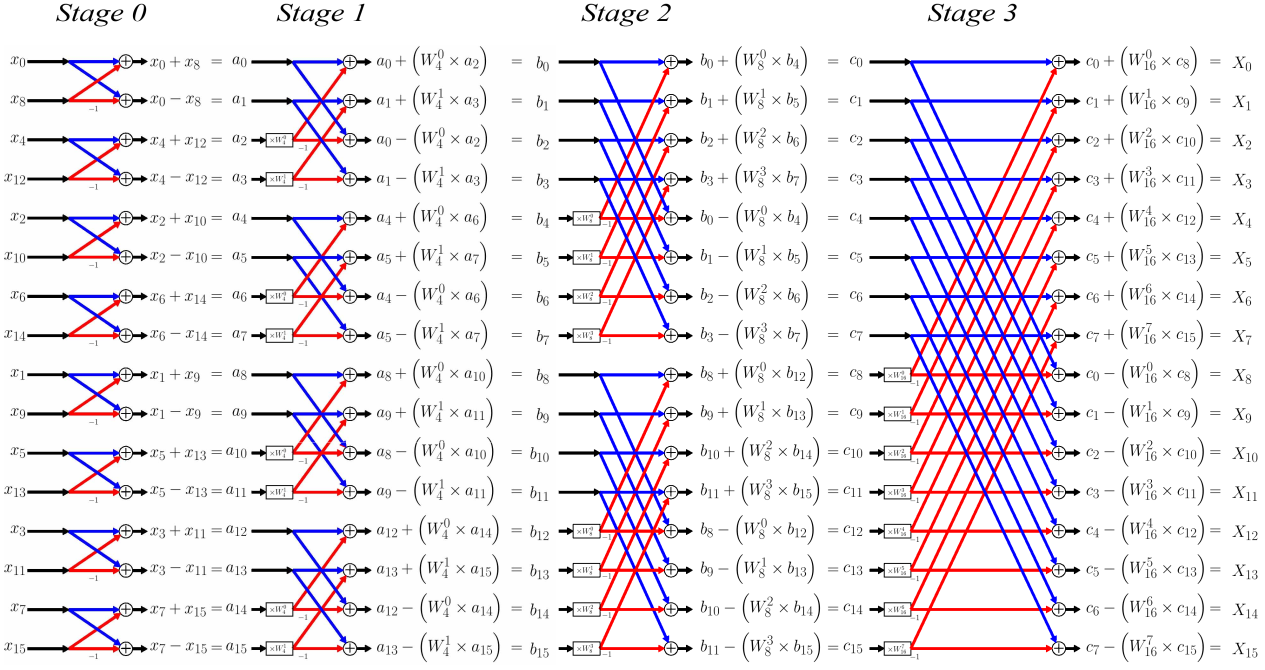
Lab\_09: FFT computing unit for 1D (N=16) FFT Algorithm

**Name : Student\_ID:**

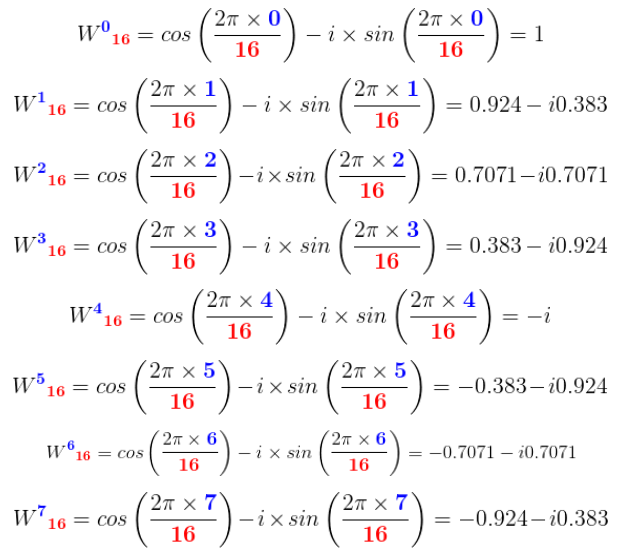
Purpose:

To understand and implement the 1D FFT Algorithm.

You are going to implement a FFT computing unit that can perform FFT algorithm for 1D (N=16) signal FFT computation. The following is the 1D FFT computation flow charts:



Here, the weight matrixes are given as below formulas:



W\_N = e^(- i\*2\*pi\*n/N)  and N=2,4,8,16 here.

W\_N^n = cos(2\*pi\*n/N) - i\*sin(2\*pi\*n/N);  and n has range from 0 to 1/3/7/15.

According to the above figures, all the input and output signals are given, you MUST represent each signal in you VHDL. “*reset*” is an asynchronous reset. When “*reset*” is 1, all the output signals are reset to 0.

You MUST use the following entity to implement you 1D (N=16) FFT computing unit:

entity fft16 is

port(

clk : IN std\_logic;

reset : IN std\_logic;

s : in comp\_array; --input signals in time domain

      y : out comp\_array  --output signals in frequency domain

    );

end fft16;

You are going to use the test input as following:

   stim\_proc: process

   begin

    --sample inputs in time domain.

s(0) <= (1,0);

        s(1) <= (2,0);

        s(2) <= (3,0);

        s(3) <= (4,0);

        s(4) <= (5,0);

        s(5) <= (6,0);

        s(6) <= (7,0);

        s(7) <= (8,0);

s(8) <= (8,0);

        s(9) <= (7,0);

        s(10) <= (6,0);

        s(11) <= (5,0);

        s(12) <= (4,0);

        s(13) <= (3,0);

        s(14) <= (2,0);

        s(15) <= (1,0);

      wait;

   end process;

Furthermore, you are going to implement a 2D FFT computing unit using 1D FFT computing unit according to the steps below:

Step1. do 1D FFT on each row (real/complex to complex)

Step 2. do 1D FFT on each column resulting from (1) (complex to complex)

For example, we can perform the 2D FFT computation of 16×16 matrix by performing 16×1D (horizontal) FFTs followed by 16×1D (vertical) FFTs, for a total of 32×1D FFTs.

You MUST use the following entity to implement you 2D (16×16) FFT computing unit:

entity fft16\_2D is

port(

clk : IN std\_logic;

reset : IN std\_logic;

s : in comp\_array2D; --input signals in time domain

      y : out comp\_array2D --output signals in frequency domain

    );

end fft16\_2D;

You are going to use the test input as following:

   stim\_proc: process

   begin

    --sample inputs in time domain.

s(0) <= (1,0);

        s(1) <= (2,0);

        s(2) <= (3,0);

        s(3) <= (4,0);

        s(4) <= (5,0);

        s(5) <= (6,0);

        s(6) <= (7,0);

        s(7) <= (8,0);

s(8) <= (8,0);

        s(9) <= (7,0);

        s(10) <= (6,0);

        s(11) <= (5,0);

        s(12) <= (4,0);

        s(13) <= (3,0);

        s(14) <= (2,0);

        s(15) <= (1,0);

….

s(127) <= (1,0);

….

s(255) <= (1,0);

      wait;

   end process;

You can test your FFT computing results in Matlab by using “fft()” and “fft2()” function.