1 Introduction

Fast Fourier Transform (FFT) is an efficient version of Discrete Fourier Transform (DFT), and the history of this algorithm's development can be traced through the 1800s and 1900s. Fourier analysis converts a signal from its original domain to representation in the frequency domain or vice-versa.

Decimation in time denotes reducing samples in the time domain, while decimation in frequency refers to reducing samples in the frequency domain. A **radix-2 decimation in time,** partitions a signal into two half-length signals of even and odd indexed input samples. This decimation is done repeatedly to convert a signal in the time domain with *N* points to *N* signals in the time domain each of a single point. As shown in Figure 1, these single points are combined into an *N* point signal in the frequency domain using butterfly calculations, where weighting factors are multiplied per stage to obtain the new values. A **radix-2 decimation in frequency**, on the other hand, splits the signal into the first half and the second half of input samples repeatedly. A similar process is applied to convert the single point signals to one combined signal in the time domain, as shown in Figure 2.

For this lab, you are expected to complete the following tasks:

a) Run the behavioral simulation of the radix-2 decimation in time FFT design. (Pre-lab)

b) Implement the radix-2 decimation in frequency FFT design and run the behavioral simulation. (Post-lab, highly recommended to finish this step before your lab session)

c) Run the radix-2 decimation in time FFT and the radix-2 decimation in frequency FFT design on the FPGA. (Post-lab)

d) Implement the pipelined radix-2 decimation in frequency FFT design and run the behavioral simulation (Post-lab, GR only).

The design structure for tasks a) and b) are shown in Figure 1 and Figure 2, respectively. In both designs, we have 8 8-bit inputs and 8 8-bit outputs. To handle such data, we need 3 stages, as shown in Figure 1 and Figure 2.

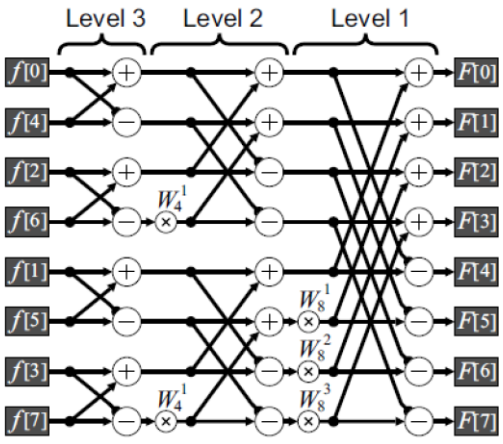

Figure 1: Radix-2 Decimation in Time FFT

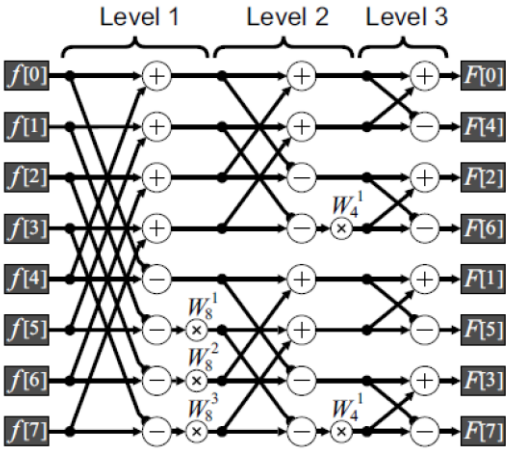

Figure 2: Radix-2 Decimation in Frequency FFT

Here are several helpful resources that introduces DFT, FFT and Radix-2 FFT. It's recommended to go over them before you proceed.

a) https://www.youtube.com/watch?v=nl9TZanwbBk&t=3s&ab_channel=SteveBrunton (DFT)

b) https://www.youtube.com/watch?v=E8HeD-MUrjY&ab_channel=SteveBrunton (FFT)

c) https://riptutorial.com/algorithm/example/27088/radix-2-fft (Radix-2 FFT)

d) https://cnx.org/contents/8D0YvnW1@7.1:XaYDVUAS@6/Decimation-in-Frequency-DIF-Radix-2-FFT (Radix-2 FFT)

e) https://en.wikipedia.org/wiki/Cooley%E2%80%93Tukey_FFT_algorithm (Radix-2 FFT)

2 Lab Design on Radix-2 Decimation in Time

In this section, we need to implement the hardware designs of the modules in Vivado. Before you proceed, please download "**Lab4_student_code.zip**" from Piazza and **extract it**.
After extraction, you will get a folder named "**Lab4_student_code/**".

a) Copy the folder "**lab3_vivado**" from lab 3 and rename it as "**lab4_vivado**".

b) Open the project by double-click on "**lab4_vivado/base/base.xpr**".

c) Add "**fft_time.v**", "**butterfly.v**", "**multiply.v**", and "**fft_time_tb.v**" (please do not modify the test bench) to the project from "**Lab4_student_code/**" and implement your design according to Section 1.

d) Please use an 8-bit signed fixed-point number with 4 bits for the fraction.

In the test bench, **fr** and **fi** are the real and the imaginary parts of the inputs. **Fr** and **Fi** are the real part and the imaginary part of the outputs. The inputs are $\sin(7) + 0i, \sin(6) + 0i, \ldots, \sin(0) + 0i$. Each input and output consist of 8 groups of 8-bit fix-point numbers. The wires **fr_split**, **fi_split, Fr_split,** and **Fr_split** help you read the inputs and outputs' values more easily. Please set the radix of them as float point number in the simulation.

To do behavior simulation, click on "**Simulation Settings**" on the Flow Navigator panel on the left. Modify the simulation top module name as **fft_time_tb**.

a) Click on "**Simulation**," you can modify the simulation time here. **50ns** is sufficient for Lab 4. Click "**ok**" to save the settings.

b) Click on "run simulation->run behavioral simulation" to launch the simulation.

If your design and testbench are correct, you should see the outputs at Fr and Fi. Explain to TA how you can infer the FFT results from Fr and Fi.

3 Implementing the design on the FPGA

In this section, we will implement the design of radix-2 decimation in time FFT on the FPGA.

a) Add "**top_fft_time.v**" into your project, right click its file name in the "**source**" panel and click on "**Set as top**" to set it as the top module.

b) Add a dual port RAM in the same way as in the previous lab. Make sure the memory block is set as "**True Dual Port Ram**." The write width should be **8,** and the write depth should be **65536**. The operation mode should be "**Read First,**" and the enable port type should be "**Always Enabled**".

c) Please launch SDK and generate the boot image (BOOT.bin) as in the previous lab with one exception: Use the bitstream file base/base.sdk/top_fft_time_hw_platform_0/top_fft_time.bit.

d) Copy the updated **BOOT.bin** and **lab4_fft_time_test** into your SD card, boot the FPGA and run the test **lab4_fft_time_test**.

e) Take a screenshot of the terminal when the result shows.

f) Unmount the SD card, exit the serial communication and turn off your FPGA.

Some commonly used commands:

- **mount /dev/mmcblk0p1 /mnt/**
- **cd /mnt/**
- **insmod transfpga.ko**
- **mknod /dev/transfpga c 245 0**
- **./lab4_fft_time_test**
- **cd /**
- **umount /mnt/**

4 Radix-2 Decimation in Frequency FFT

Please repeat Section 2 and 3 to design a **radix-2 decimation in frequency FFT**. Complete this with the use of these files: "**fft_freq.v**", "**fft_freq_tb.v**", "**top_fft_freq.v**", "**lab4_fft_freq _test**". If you get everything correct, you should see roughly the same outputs as in fft_time with some minor differences.

5 Pipelined Design (Graduates Only)

a) Please add "fft_freq_pipe.v", "fft_freq_pipe_tb.v", and "top_fft_freq_pipe.v" to your project.

b) Implement the pipelining of radix-2 decimation in frequency FFT and separate it into 3 pipelined

stages.

c) Please run the behavioral simulation of the pipelined design and include it in the report.

d) (Optional) We also provided "**lab4_fft_freq_pipe_test**" to test your design in the FPGA.

6 Pre-lab Submission

1. Please only submit 1 PID file, containing the following items:
    a. Screenshots of the behavioral simulation of the radix-2 decimation in time FFT design.
    b. Justifications of the correctness of your screenshots.
2. Please name the PDF file as "Lab#_PreLab_Section#_LastName_FirstName.pdf".
3. Please submit the PDF file on Canvas before February 21 (Monday) 11:59 pm.

7 Post-lab Submission

1. Please only submit 1 PID file, containing the following items:
    a. Screenshots of the terminal output of the radix-2 decimation in time FFT design.
    b. Screenshots of the behavioral simulation of the radix-2 decimation in freq FFT design.
    c. Screenshots of the terminal output of the radix-2 decimation in freq FFT design.
    d. Copy of your codes on the radix-2 decimation in freq FFT design.
    e. Justifications of the correctness of your screenshots.
2. Please name the PDF file as "Lab#_PostLab_Section#_LastName_FirstName.pdf".
3. Please submit the PDF file on Canvas before February 25 (Friday) 11:59 pm.