Dillon Engineering

5021 Vernon Avenue, Suite 285, Edina, 55424
Tel.: 952-836-2413 Fax: 952-927-6514
www.dilloneng.com

Dillon Engineering

Specification Document

Load-Unload Fft IP

Table of Contents

1 Overview

This document describes a the Load-Unload Fft IP used to perform radix-2 length FFTs.

The ASIC logic is delivered in Verilog source and can also be sythesized for operation in any FPGA technology. The FFT engine is licensed via the Dillon Engineering Site License Agreement.

The Load-Unload Fft IP engine would contain a single FFT processing unit with 1, 2 or 4 butterflies. A single clock would run the I/O and processing. An input data enable throttles the incoming data down to the input rate.

There is an optional input buffer, so that loading can overlap with processing. If the input buffer is not used, then the processor buffer is loaded directly from in the input.

The processing time is measure from the time the last point is input till the last point is output. The Processing time consists of loading data to processing buffer, fft processing, and unload data from processing buffer.

The latency measured in processor clock cycles for the B butterfly version of length N would be N/2/B*log2(N) (FFT computation)+ N (unload time). B is the number of butterflies and N is the FFT length.

Symbol length and scaling are selected at run time on a symbol by symbol basis.

Data and Twiddle Factor width are selected at compile time and are independent of each other.

All FFT IP code will be delivered in Verilog and suitable for synthesis to either FPGA or ASIC.

2 Design

The structure of the Load-Unload Fft IP is describe in this section.

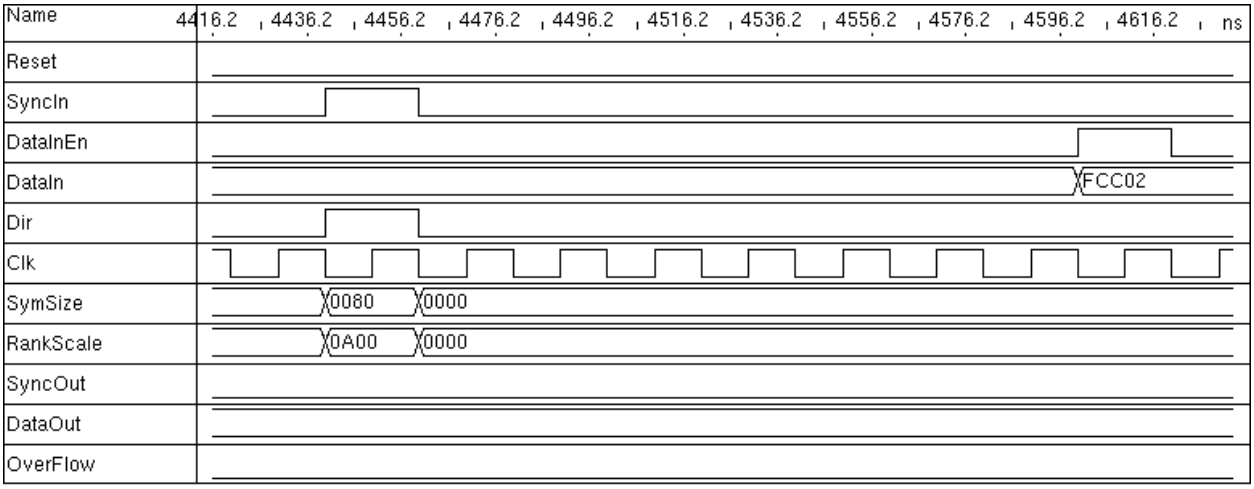
2.1 I/O

This section define the proposed I/O to the FFT processing engines.

Port	*In/Out*	*Width*	*Function*
Clk	In	1	Processing clock
SyncIn	In	1	New symbol sync, proceeds data by a clock cycle.
Dir	In	1	IFFT/FFT select, 0 selects IFFT 1 for FFT. Sampled on SyncIn cycle.
SymSize	In	14	Symbol size, FFT is valid from 2**5 to 2**10. Maximum length can be changed at compile time, will require larger memories.
RankScale	In	13	Rank scale control, 1 bit per rank for rank 0 to 9. A 1 indicates to scale bufferfly results by 2 for that rank. Sampled on Sync cycle. For lengths shorter than 2**10, the rankscale is left justified, with RankScale[9] always the scale select for rank 0.
DataIn	In	nBits	Input data. Real only input data.
DataInEn	In	1	Input data qualification,must be at least one clock cycle after SyncIn.
SyncOut	Out	1	Output symbol sync, proceeds output data by one clock cycle.
OverFlow	Out	1	Processor overflow. Valid on SyncOut cycle.
DataOut	Out	nBits*2	Processor results data, valid one clock cycle after SyncOut for N clock cycles. Real is on DataOut[nBits*2 – 1:nBits] and imaginary DataOut[nBits-1:0].

2.2 I/O Waveforms

2.2.1 Input

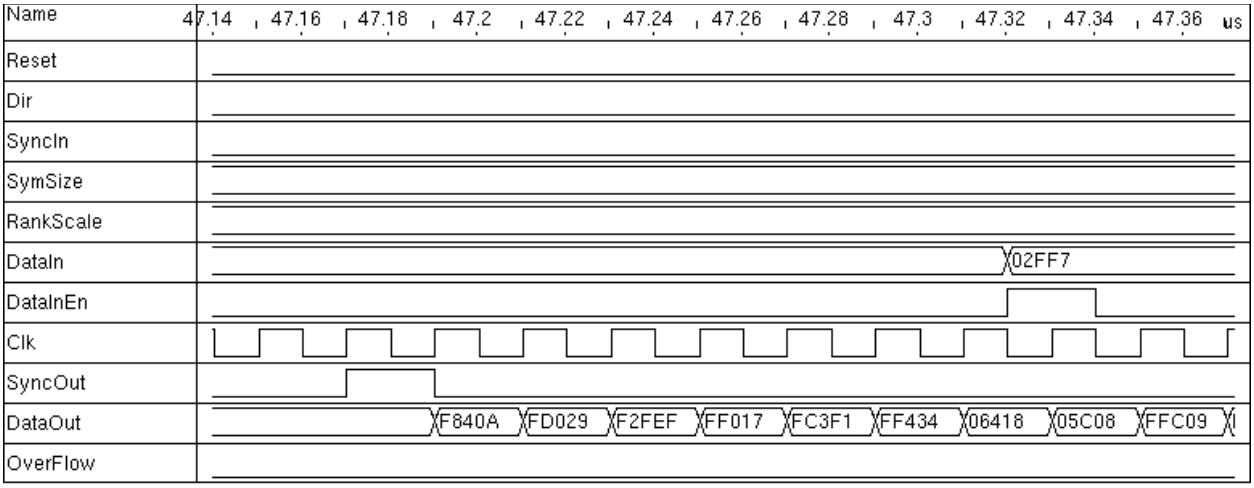


All I/O sampled/driven on rising edge of clock.

First input valid data is FCC02.

Note, SymSize, RankScale, Dir are sample on SyncIn cycle.

2.2.2 Output



OverFlow was sampled 0 cn the SyncOut cycle.

Data follows SyncOut, first valid DataOut is F840A.

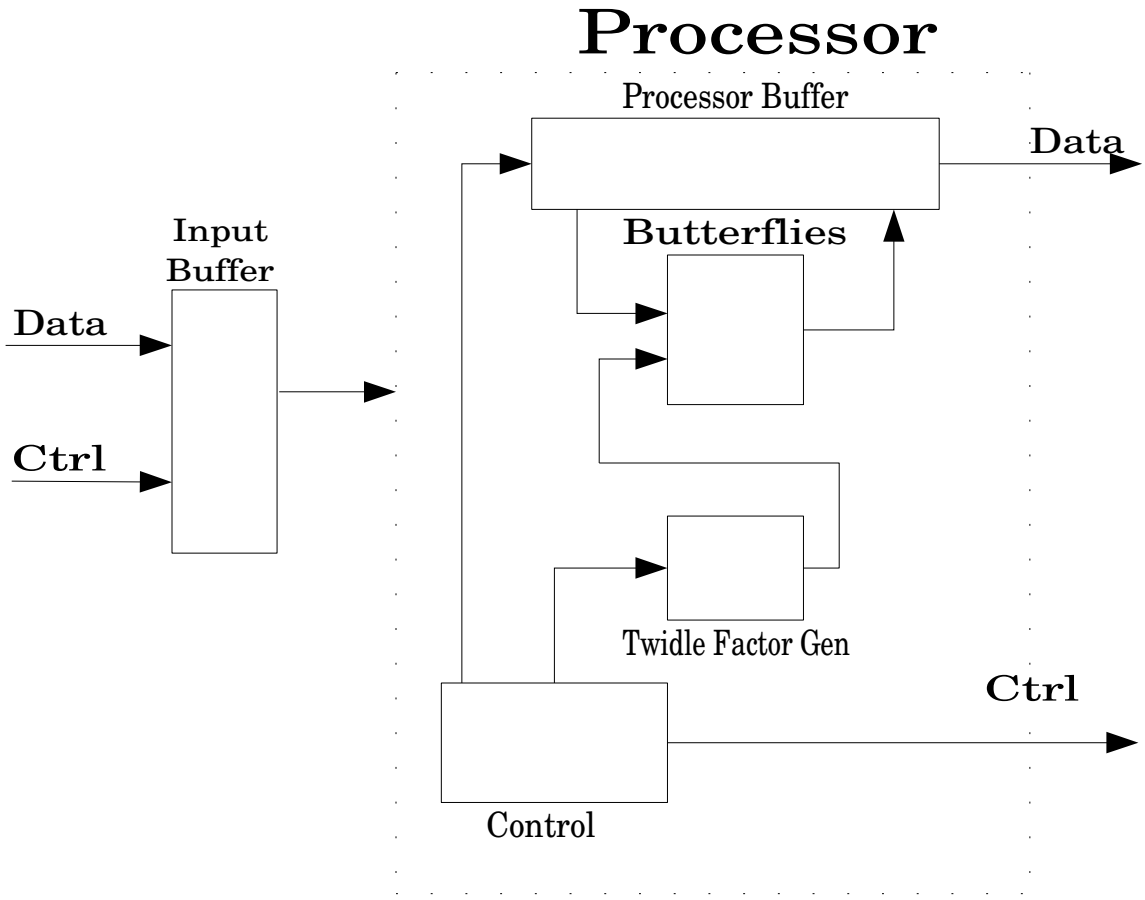
2.3 Logic Blocks

Each engine contains the following blocks:

- Input Buffer
- Processor Control
- Processor Twiddle Factor Generation
- Processor Butterfly Block
- Processor Buffer

The following diagrams illustrates the data flow in the system.

Note, Input Buffer is optional.



FFT Processor Block Diagram

2.4 Input Buffer

The input buffer is optional.

The Input Buffer will hold a single symbol and utilizes single port memory.

Note, there is no restriction on the fill rate of the FFT Input Buffer, it can accept data every clock cycle. The Input Buffer does require quiet cycles to empty into the Processor and will use any non write cycles for that purpose.

2.4.1 Function

The Input Buffer simply buffers a full symbol (length 2**5 to 2**10 depending upon SymLength), once a full symbol is stored, it outputs it as fast as possible. Both input and output occur in natural order.

The Input Buffer utilizes single port synchronous SRAM, which means there is potential for conflict reading and writing. Data into the memory will take priority over data out, in other words, when a point needs to be written into the memory, it will cause a single point gap in the output stream.

Several input signals are passed through with the symbol.

2.5 FFT Processor

The FFT Processor is a stand alone FFT/IFFT processor.

It inputs/outputs a single point per clock cycle. The input rate is controlled by a DataEn while the output in on consecutive clock cycles.

It functions with 3 distinct modes:

1. Load
2. Processing
3. Unload

The FFT Processor is made up of the following major blocks:

- Processor Control
- Twiddle Factor Generation
- Bufferfly Block
- Processor Buffer

2.5.1 Processor Control

The Processor Control provides all control for the FFT processor execution including:

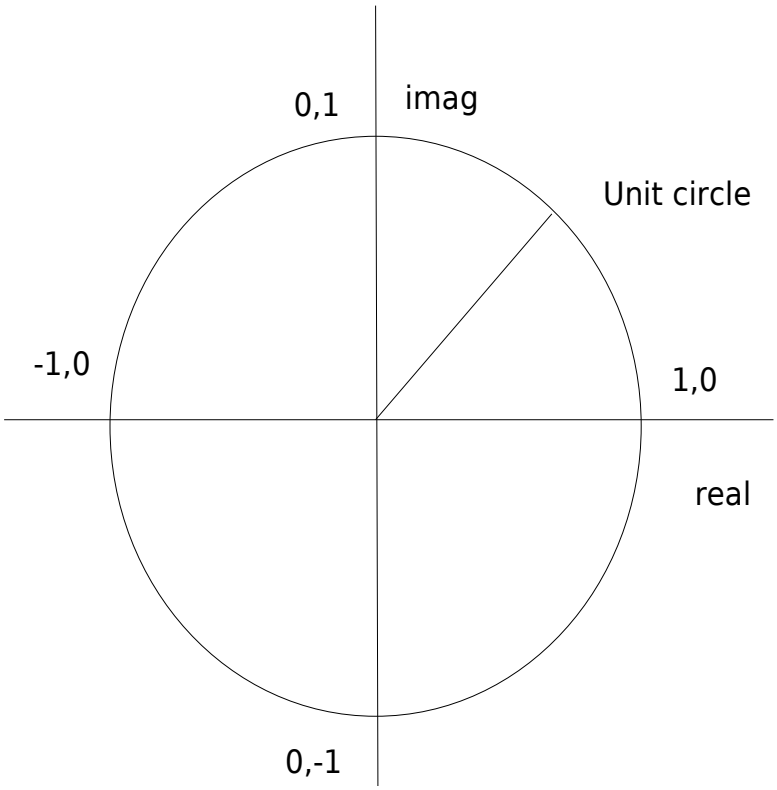
- Buffer address sequence generation
- Twiddle factor generation control
- Butterfly execution control
- Output data control

The control block sequence generation so that only a single symbol buffer can be used is very complicated since this is a variable length FFT.

2.5.2 Twiddle Factor Generation

The Twiddle Factors are stored in ROMs. Not all the twiddle factor are store since with ¼ of the points the rest can be calculated with fairly simple math.

This module takes as input the twiddle factor number and returns the complex number representing the twiddle factor. Since only ¼ of the twiddle factors are store, some calculation are required to get the proper one.



The twiddle factor are number 0 to N/2-1, N being the symbol length.

The twiddle factor is the coordinate on the unit circle, after rotating by the appropriate angle, between 0 and N/2-1. The angle spans for 0 to 180 degrees (actually 1 tick short of 180).

The FFT angle follows the path from 1,0, through 0,-1 and to -1,0.

The IFFT angle follows the opposite path, from 1,0, through 0,1 and to -1,0.

That is the only difference between the FFT and IFFT logic.

Note in the above circle diagram, all the unique coordinates are contained in 45 degress of data, so we only need N/8 unique location in the ROM.

Bases upon the FFT/IFFT selection, and the quadrant of the angle, it is easy to calculate which ROM location should be used and and if any need to be negated.

Basically you end up with two ROMs, one for real and one for imaginary. You must manipulate the address based upon the quadrant and also the result read from the ROM.

Each butterfly has a N/8 x twBits * 2 ROM. TwBits being the real width of the twiddle factors.

2.5.3 Butterfly Block

The Butterfly Block contains the butterflies for the engine. Either 1, 2 or 4 butterflies are present depending on the implementation.

2.5.4 Processor Buffer

The Processor Buffer contains a full symbol during the processor execution.

Each butterfly has two buffers.

The length of each the memory is N/2/numButterflies (numButterflies is number of butterflies in the engine).

The width of each buffer is nBits*2 (nBits is real data width).

The memory here is dual port capable of reading and writing a single full width point per clock cycle.

If single port memory has to be used, double the number of memories required for the processor buffer.

3 Engine Simulation

The Engine Simulation task creates a test bench for the full system.

There will be an option to use input data supplied by Dillon Engineering and produce results that the client can use to verify proper operation.

3.1 Rigorous Test Cases

All FFT logic designed and delivered by DE has been verified against the following test cases, providing a very rigorous testing of the logic delivered. In the following, the value of N is the length of the FFT.

A. Single FFT tests - N inputs and N outputs
    1. Input random data
    2. Inputs are all zeros
    3. Inputs are all ones (or some other nonzero value)
    4. Inputs alternate between +1 and -1.
    5. Input is e^(8*j*2*pi*i/N) for i = 0,1,2, ...,N-1. (j = sqrt(-1))
    6. Input is cos(8*2*pi*i/N) for i = 0,1,2, ...,N-1.
    7. Input is e^((43/7)*j*2*pi*i/N) for i = 0,1,2, ...,N-1. (j = sqrt(-1))
    8. Input is cos((43/7)*2*pi*i/N) for i = 0,1,2, ...,N-1.

B. Multi FFT tests - run continuous sets of random data
    1. Data sets start at times 0, N, 2N, 3N, 4N, ...
    2. Data sets start at times 0, N+1, 2N+2, 3N+3, 4N+4, ...

These tests have been carefully chosen to expose design issues commonly found in FFT logic. The single FFT tests are designed to expose problems with the arithmetic or control used when processing a single set of FFT data. The multi FFT tests are designed to expose problems with the logic that controls how multiple sets of data are passed through the FFT logic.