VLSI Implementation of a Pipelined 128 points 4-Parallel radix-2^3 FFT Architecture via Folding Transformation

James J. W. Kunst jjwk89@gmail.com
Kevin H. Viglianco kevinviglianco@gmail.com
Daniel R. Garcia dani6rg@gmail.com

June 11, 2020

Tabla de Contenidos

Table of Contents

Sección 1

Introduction

Introduction

Introduction:

Objetives

- Design and implement a 4-parallel pipelined architecture for the Complex Fast Fourier Transform (CFFT) based on the radix-2^3 algorithm with 128 points using folding transformation and register minimization techniques based on ... .
- Frecuency of implementation: 500MHz.
- Optimization with CSD multipliers.
- Test the design with a mixture of two sinusoids using MATLAB.
- Generate power-area-timing report with different optimizations.

Introduction

Workflow

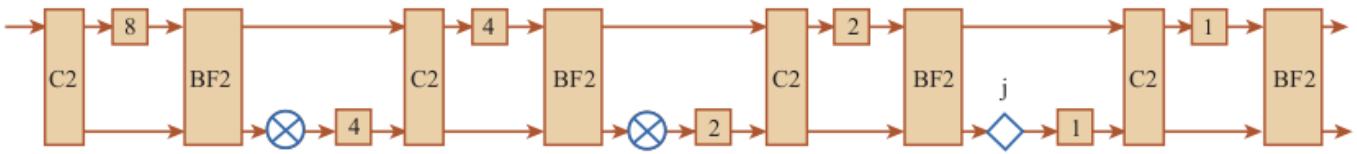
1. Obtain the equations that correspond to Butterfly structure of radix-2^3 FFT-DIF for 8 points.
2. Apply this idea to design a 2-parallel pipelined architecture radix-2^3 16-points FFT feedfoward via folding transformation.
3. Traslate this to a 128-points model.
4. Elaborate a float-point simulator in Matlab of the 128-points.
5. Elaborate a synthesizable verilog code HDL and verify the DFT functionality.
6. Generates power-area-timing report with different optimizations.

Introduction

Introduction:

Types of pipelined FFT architectures

- Feedfoward architectures, wich can be divided into:
  - Single-Path Delay Commutator (SDC).
  - Multi-Path Delay Commutator (MDC).
- Feedback architectures, wich can be divided into:
  - Single-Path Delay Feedback architectures (SDF).
  - Multi-path Delay Feedback architectures (MDF).



(1) . R2MDC(N-16)

(2) . R25DF(N-16)

Table of Contents

Sección 2

The Radix-2^3 FFT Algorithm

The Radix-2^3 FFT Algorithm

The Radix-2^3 FFT Algorithm:

- The N-point DFT of an input sequence $x[n]$ is defined as:

$$X[k] = \sum_{n=0}^{N-1} x[n] \dot{W}_N^{nk}, \quad k = 0, 1, ..., N-1 \tag{1}$$

  where $W_N^{nk} = e^{-j\frac{2\pi}{N}nk}$.

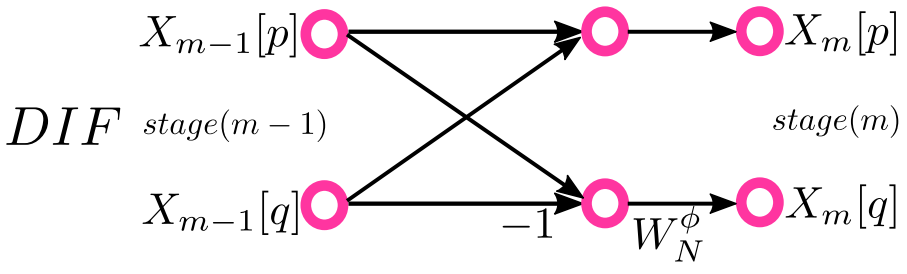
- Appling a *Divide and Conquer* approach we can calculate the DFT in series of $s = log_\rho N$ stages, where ρ is the base of the *radix*.
- In our case, the numeber of stages is:

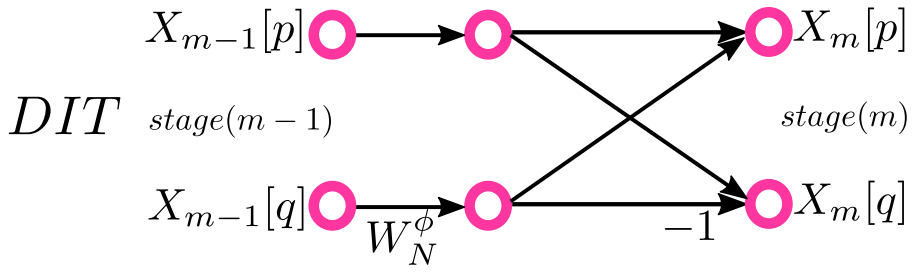
$$s = log_2(128) = 7 \tag{2}$$

The Radix-2^3 FFT Algorithm

The Radix-2^3 FFT Algorithm:

- There are two methods to design FFT algorithms:
  1. Decimation In Time (DIT).
  2. Decimation In Frcuency (DIF).
- The difference is the instant in which the multiplication by W_N^{ϕ} is accomplished.



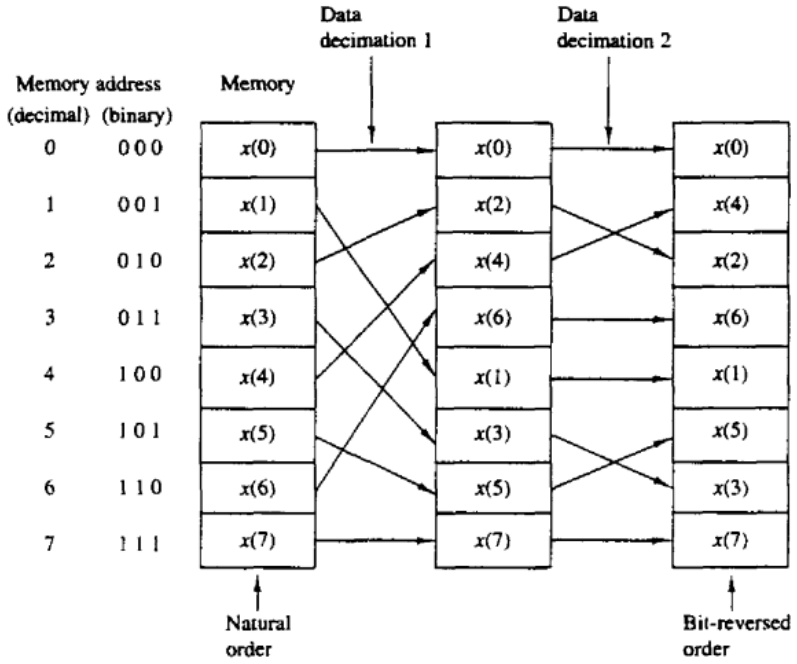


Basic butterflies computation in the decimation in time and frequency.

The Radix-2^3 FFT Algorithm

The Radix-2^3 FFT Algorithm:

- The input samples in FFT algorithms DIF are organized in natural order but its output has not in order.
- The opposite is for DIT.
- Is necessary a circuit for reordering the output data.

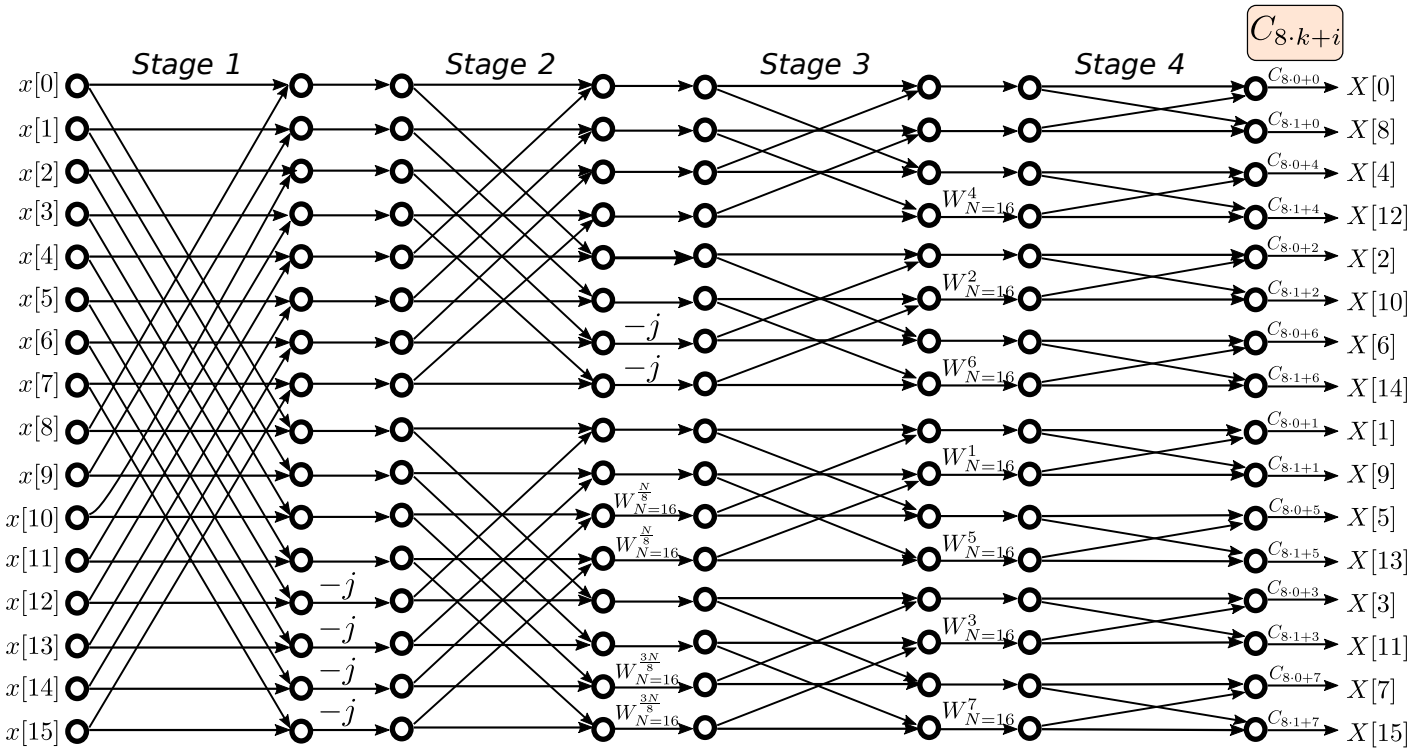


Order of intpus and outputs for DIF FFT.

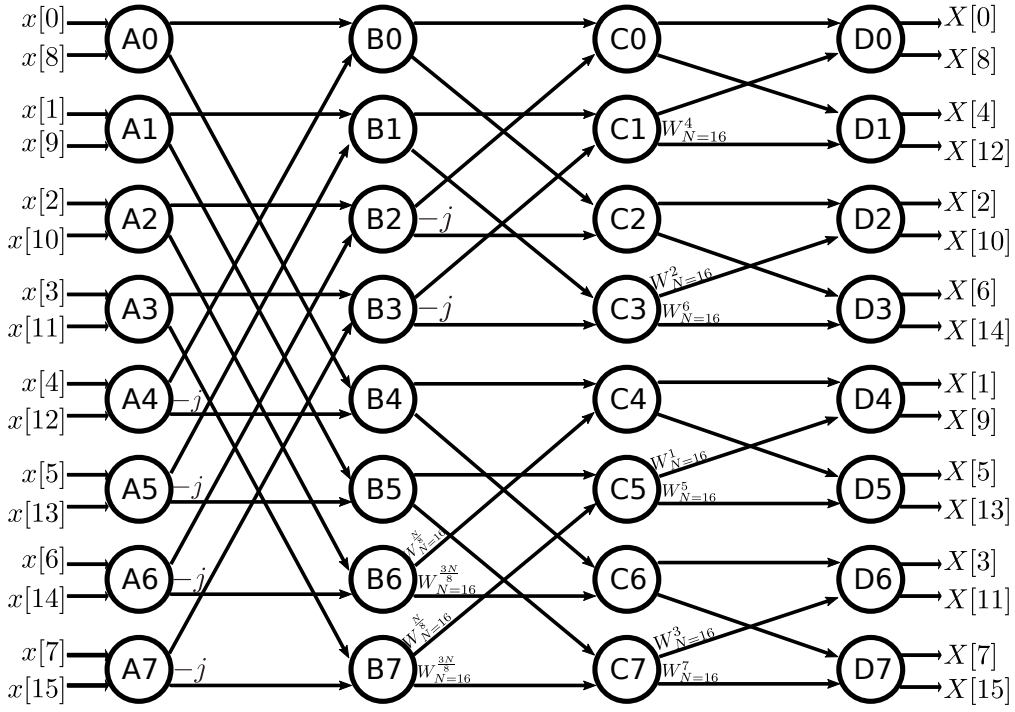
The Radix-2^3 FFT Algorithm

The Radix-2^3 FFT Algorithm:

- The fundamental equations for DIF are:



Flow graph of a radix-2^3 16-point DIF DFT.



Data flow graph (DFG) of a radix-2^3 16-point DIF DFT.

Preguntas

Muchas Gracias!