

# VLSI Implementation of a Pipelined 128 points 4-Parallel radix-2<sup>3</sup> FFT Architecture via Folding Transformation

James J. W. Kunst [jjwk89@gmail.com](mailto:jjwk89@gmail.com)

Kevin H. Viglianco [kevinviglianco@gmail.com](mailto:kevinviglianco@gmail.com)

Daniel R. Garcia [dani6rg@gmail.com](mailto:dani6rg@gmail.com)

June 19, 2020

# Tabla de Contenidos

- 1 Introduction
- 2 The Radix-2<sup>3</sup> FFT Algorithm
- 3 Design of FFT architecture via folding transformation
  - Parallel radix-2<sup>3</sup> 16-Points
- 4 Implementation
  - Parallel radix-2<sup>3</sup> 16-Points

# 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<sup>3</sup> algorithm with 128 points using folding transformation and register minimization techniques based on ... .
- Frequency of implementation: 500MHz.
- Optimization with CSD<sup>a</sup> multipliers.
- Test the design with a mixture of two sinusoids using MATLAB.
- Generate power-area-timing report with different optimizations.

---

<sup>a</sup>Canonic Signed Digit

# Introduction

## Introduction:

### Workflow

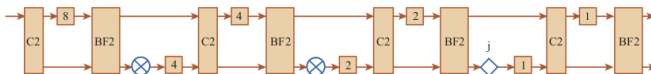
- 1 Obtain the equations that correspond to Butterfly structure of radix-2<sup>3</sup> FFT for 8 points.
- 2 Apply this idea to design a 2-parallel pipelined architecture radix-2<sup>3</sup> 16-points FFT via folding transformation.
- 3 Translate this to a 128-points model.
- 4 Elaborate a float-point 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 Radix-2 FFT architectures

- 1 Feedforward Multi-Path Delay Commutator (MDC).
  - 2 Single-Path Delay Feedback architectures (SDF).
- Both butterflies and multipliers are in 50% utilization ... .
  - The SDF use registers more efficiently.
  - We will focus on the feedforward MDC architecture.



(1) . R2MDC(N-16)



(2) . R2SDF(N-16)

## Sección 2

# The Radix-2<sup>3</sup> FFT Algorithm

# The Radix-2<sup>3</sup> FFT Algorithm

The Radix-2<sup>3</sup> FFT Algorithm:

## N-point DFT of an input sequence $x[n]$

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot W_N^{nk}, \quad k = 0, 1, \dots, N-1 \quad (1)$$

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

- Direct computation of the DFT is inefficient because it does not exploit the properties of:
  - ① Symmetry:  $W_N^{k+N/2} = -W_N^k$
  - ② Periodicity:  $W_N^{k+N} = W_N^k$
- The FFT based on Cooley-Tukey algorithm reduce the number of operations from  $O(N^2)$  for the DFT to  $O(N\log_2 N)$  for the FFT.



# The Radix-2<sup>3</sup> FFT Algorithm

The Radix-2<sup>3</sup> FFT Algorithm:

## Divide and Conquer approach

- We can calculate the DFT in series of  $s = \log_{\rho} N$  stages, where  $\rho$  is the base of the *radix*.
- This is based on the decomposition of an  $N$  point DFT into successively smaller DFTs.
- In our case, the number of stages is:

$$s = \log_2(128) = 7 \quad (2)$$

## Methods to design FFT algorithms

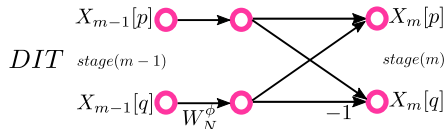
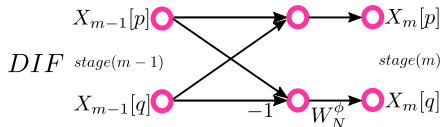
- 1 **Decimation In Time (DIT):** Splitting successively data sequence  $x[n]$  by a factor of 2.
- 2 **Decimation In Frequency (DIF):** Splitting successively the data sequence  $X[k]$  by a factor of 2.

# The Radix-2<sup>3</sup> FFT Algorithm

The Radix-2<sup>3</sup> FFT Algorithm:

## DIT and DIF Butterflies

The difference is the instant in which the multiplication by  $W_N^\phi$  is accomplished.

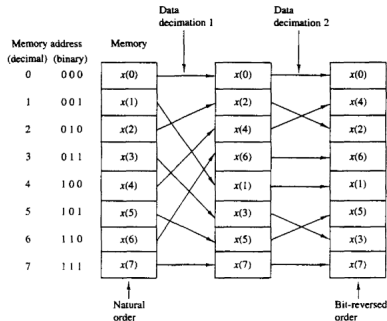


# The Radix-2<sup>3</sup> FFT Algorithm

The Radix-2<sup>3</sup> FFT Algorithm:

## Order of samples in DIT and DIF

- The input samples in FFT algorithms DIF are organized in natural order but its output has not in order.
- The opposite is for DIT.



Order of inputs and outputs for DIF FFT.

# The Radix-2<sup>3</sup> FFT Algorithm

The Radix-2<sup>3</sup> FFT Algorithm:

## Mathematic expression of radix-8 butterfly element

$$C_{8k+0} = \sum_{n=0}^{N/8-1} \left\{ [(x_n + x_{n+\frac{N}{2}}) + (x_{n+\frac{N}{4}} + x_{n+\frac{3N}{4}})] + [(x_{n+\frac{N}{8}} + x_{n+\frac{5N}{8}}) + (x_{n+\frac{3N}{8}} + x_{n+\frac{7N}{8}})] \right\} W_N^{0n} W_{N/8}^{nk}$$

$$C_{8k+4} = \sum_{n=0}^{N/8-1} \left\{ [(x_n + x_{n+\frac{N}{2}}) + (x_{n+\frac{N}{4}} + x_{n+\frac{3N}{4}})] - [(x_{n+\frac{N}{8}} + x_{n+\frac{5N}{8}}) + (x_{n+\frac{3N}{8}} + x_{n+\frac{7N}{8}})] \right\} W_N^{4n} W_{N/8}^{nk}$$

$$C_{8k+2} = \sum_{n=0}^{N/8-1} \left\{ [(x_n + x_{n+\frac{N}{2}}) - (x_{n+\frac{N}{4}} + x_{n+\frac{3N}{4}})] - j[(x_{n+\frac{N}{8}} + x_{n+\frac{5N}{8}}) - (x_{n+\frac{3N}{8}} + x_{n+\frac{7N}{8}})] \right\} W_N^{2n} W_{N/8}^{nk}$$

$$C_{8k+6} = \sum_{n=0}^{N/8-1} \left\{ [(x_n + x_{n+\frac{N}{2}}) - (x_{n+\frac{N}{4}} + x_{n+\frac{3N}{4}})] + j[(x_{n+\frac{N}{8}} + x_{n+\frac{5N}{8}}) - (x_{n+\frac{3N}{8}} + x_{n+\frac{7N}{8}})] \right\} W_N^{6n} W_{N/8}^{nk}$$

$$C_{8k+1} = \sum_{n=0}^{N/8-1} \left\{ [(x_n - x_{n+\frac{N}{2}}) - j(x_{n+\frac{N}{4}} - x_{n+\frac{3N}{4}})] + W_N^{N/8} [(x_{n+\frac{N}{8}} - x_{n+\frac{5N}{8}}) - j(x_{n+\frac{3N}{8}} - x_{n+\frac{7N}{8}})] \right\} W_N^{n} W_{N/8}^{nk}$$

$$C_{8k+5} = \sum_{n=0}^{N/8-1} \left\{ [(x_n - x_{n+\frac{N}{2}}) - j(x_{n+\frac{N}{4}} - x_{n+\frac{3N}{4}})] - W_N^{N/8} [(x_{n+\frac{N}{8}} - x_{n+\frac{5N}{8}}) - j(x_{n+\frac{3N}{8}} - x_{n+\frac{7N}{8}})] \right\} W_N^{5n} W_{N/8}^{nk}$$

$$C_{8k+3} = \sum_{n=0}^{N/8-1} \left\{ [(x_n - x_{n+\frac{N}{2}}) + j(x_{n+\frac{N}{4}} - x_{n+\frac{3N}{4}})] + W_N^{3N/8} [(x_{n+\frac{N}{8}} - x_{n+\frac{5N}{8}}) + j(x_{n+\frac{3N}{8}} - x_{n+\frac{7N}{8}})] \right\} W_N^{3n} W_{N/8}^{nk}$$

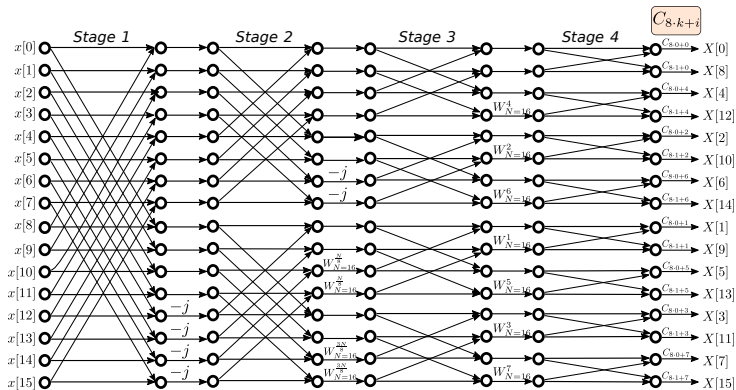
$$C_{8k+7} = \sum_{n=0}^{N/8-1} \left\{ [(x_n - x_{n+\frac{N}{2}}) + j(x_{n+\frac{N}{4}} - x_{n+\frac{3N}{4}})] - W_N^{3N/8} [(x_{n+\frac{N}{8}} - x_{n+\frac{5N}{8}}) + j(x_{n+\frac{3N}{8}} - x_{n+\frac{7N}{8}})] \right\} W_N^{7n} W_{N/8}^{nk}$$

# The Radix-2<sup>3</sup> FFT Algorithm

The Radix-2<sup>3</sup> FFT Algorithm:

## Radix-2<sup>k</sup> Implementation

The quantity of rotators of an architecture radix-2<sup>k</sup> (with  $k > 1$ ) is less than the radix-2.



Flow graph of a radix-2<sup>3</sup> 16-point DIF DFT.

# The Radix-2<sup>3</sup> FFT Algorithm

The Radix-2<sup>3</sup> FFT Algorithm:

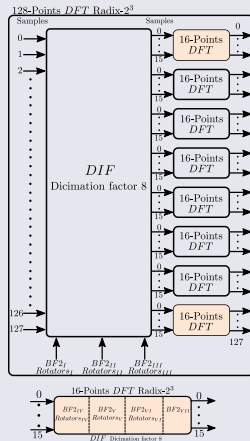
## The Radix-2<sup>3</sup> FFT Algorithm

- Applying (12) for the 128 point DFT and calculating each coefficient for  $k = 0, 1, \dots, (128/8) - 1$ :

$$C_{8k+i} = \sum_{n=0}^{128/8-1} \{ \cdot \}$$

We get a sequence in chain of butterflies with its corresponding rotation factor.

- The 128 point DFT goes through a processes that involve tree stages of butterflies to arrive finally to a set of eight DFT where each of one is a 16 point DFT.



## Sección 3

### Design of FFT architecture via folding transformation

# Design of FFT architecture via folding transformation

Design of FFT architecture via folding transformation: Parallel radix-2<sup>3</sup> 16-Points

## Folding Set

- Is an ordered set of operations executed by the same functional unit.
- Each folding set contains  $K$  entries, where  $K$  is called the folding factor.
- The operation in the  $j$ th position (where  $j$  goes from 0 to  $K-1$ ) is called the folding order.

## Folding Equations

- Consider an edge  $e$  connecting the nodes  $U$  and  $V$  with  $w(e)$  delays.
- The executions of the  $l$ th iteration of  $U$  and  $V$  are scheduled at the time units  $Kl + u$  and  $Kl + v$  respectively, where  $u$  and  $v$  are the folding orders of the nodes  $U$  and  $V$ .
- The folding equation for the edge  $e$  is:

$$D_F(U \rightarrow V) = Kw(e) - P_U + v - u \quad (3)$$

where  $P_U$  is the number of pipeline stages in the node  $U$ .



# Design of FFT architecture via folding transformation

Design of FFT architecture via folding transformation: Parallel radix-2<sup>3</sup> 16-Points

## Folding equations without retiming/pipeline

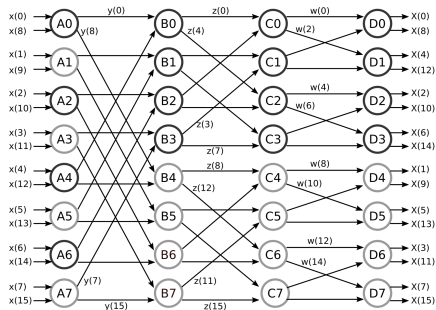
- Consider the folding sets:

$$\begin{aligned} A &= \{A_0, A_2, A_4, A_6\} & A' &= \{A_1, A_3, A_5, A_7\} \\ B &= \{B_1, B_3, B_5, B_7\} & B' &= \{B_0, B_2, B_4, B_6\} \\ C &= \{C_2, C_1, C_3, C_0\} & C' &= \{C_6, C_5, C_7, C_4\} \\ D &= \{D_3, D_0, D_2, D_1\} & D' &= \{D_7, D_4, D_6, D_5\} \end{aligned}$$

- For example:

$$\begin{aligned} D_F(D_3 \rightarrow B_3) &= v - u \\ &= 0 - 1 \\ &= -1 \end{aligned}$$

- The folding equations can be derived for all edges.



DFG of a radix-2<sup>3</sup> 16-point DIF DFT.

# Design of FFT architecture via folding transformation

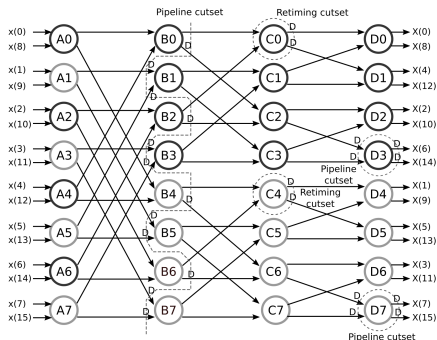
Design of FFT architecture via folding transformation: Parallel radix-2<sup>3</sup> 16-Points

## Folding equations with retiming/pipeline

- For the folded system to be realizable,  $D_F(U \rightarrow V) \geq 0$  must hold for all the edges.
- For example:

$$\begin{aligned} D_F(D3 \rightarrow B3) &= Kw(e) + v - u \\ &= 4(1) + 0 - 1 \\ &= 3 \end{aligned}$$

- This result  $D_F(U \rightarrow V) \geq 0$  for all the edges.
- Applying the folding equations for all the edges, the number of registers required is 80.



DFG of a radix-2<sup>3</sup> 16-point DIF DFT applying pipeline and retiming.

# Design of FFT architecture via folding transformation

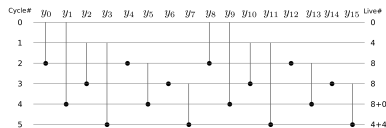
Design of FFT architecture via folding transformation: Parallel radix-2<sup>3</sup> 16-Points

## Lifetime analysis

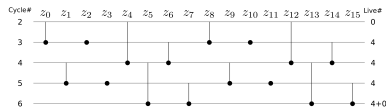
- Is a procedure used to compute the minimum number of registers.
- For example, the variable  $y_1$  be live during time units  $n \in \{1, 2, 3, 4\}$ .
- The number of live variables  $y_i$  during the time units  $\{1, 2, 3, 4, 5\}$  is  $\{4, 8, 8, 8, 4\}$ . So the number of register for this stage is:

$$\max\{4, 8, 8, 8, 4\} = 8$$

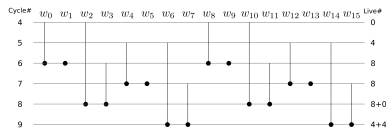
- The total number of registers is reduced from 80 to 20.



Lifetime chart for stage 1.



Lifetime chart for stage 2.



Lifetime chart for stage 3.

# Design of FFT architecture via folding transformation

Design of FFT architecture via folding transformation: Parallel radix-2<sup>3</sup> 16-Points

## Forward Register Allocation

- This dictates how the variables are assigned to the minimum numbers of registers.
- If  $R_i$  holds a variable in the current cycle, then  $R_{i+1}$  hold the same variable on the next cycle.

	I/P	R1	R2	R3	R4	R5	R6	R7	R8
0	y0,y8,y1,y9								
1	y2,y10,y3,y11	y1		y0		y9		y8	
2	(y4)y12,y5,y13	y3	y1	y2	(y0)	y11	y9	y10	(y8)
3	(y6)y14,y7,y15	y5	y3	y1	(y2)	y13	y11	y9	(y10)
4		y7	(y5)	y3	(y1)	y15	y13	y11	(y9)
5			(y7)		(y3)	(y15)		(y11)	

Allocation table for stage 1.

	I/P	R1	R2	R3	R4
2	z0,z4,z8,z12				
3	(z2)z6(z10)z14	z4	(z0)	z12	(z8)
4	z1,z5,z9,z13	(z6)	(z4)	(z14)	(z12)
5	(z3)z7(z11)z15	z5	(z1)	z13	(z9)
6		(z7)	(z5)	(z15)	(z13)

Allocation table for stage 2.

	I/P	R1	R2	R3	R4	R5	R6	R7	R8
4	w0,w2,w8,w10								
5	w4,w6,w12,w14	w2		w0		w10		w8	
6	(w1)w3(w9)w11	w6	w2	w4	(w0)	w14	w10	w12	(w8)
7	(w5)w7(w13)w15	w3	w6	w2	(w4)	w11	w14	w10	(w12)
8		w7	(w3)	w6	(w2)	w15	(w11)	w14	(w10)
9			(w7)		(w6)	(w15)		(w14)	

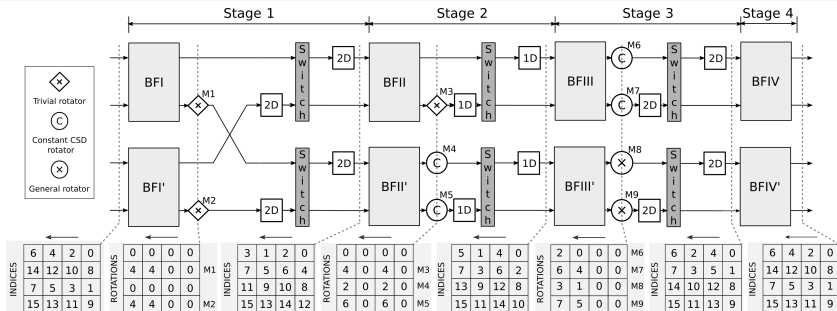
Allocation table for stage 3.

# Design of FFT architecture via folding transformation

Design of FFT architecture via folding transformation: Parallel radix-2<sup>3</sup> 16-Points

## Folding architecture for radix-2<sup>3</sup> 16 points FFT

- The values in the same column are data that flow in parallel and values in the same row flow through the same path in consecutive clock cycles.
- For the rotator matrix, each number  $k$  of the matrix represent a multiplication by  $W_N^k$ .

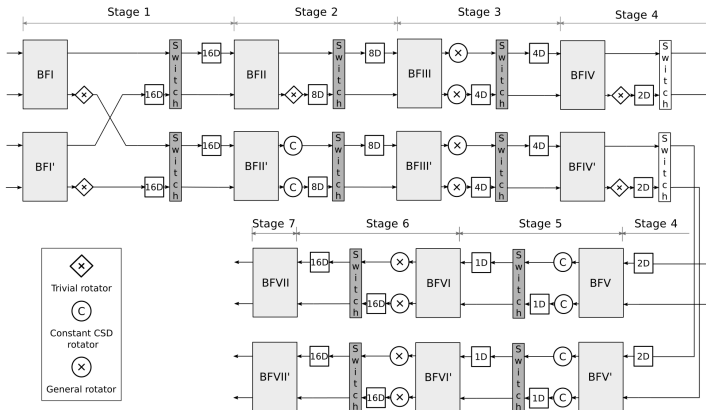


Folding architecture for radix-2<sup>3</sup> 16 points FFT.

# Design of FFT architecture via folding transformation

Design of FFT architecture via folding transformation: Parallel radix-2<sup>3</sup> 16-Points

We can deduce the folding architecture for 128 points radix-2<sup>3</sup> following the same method.



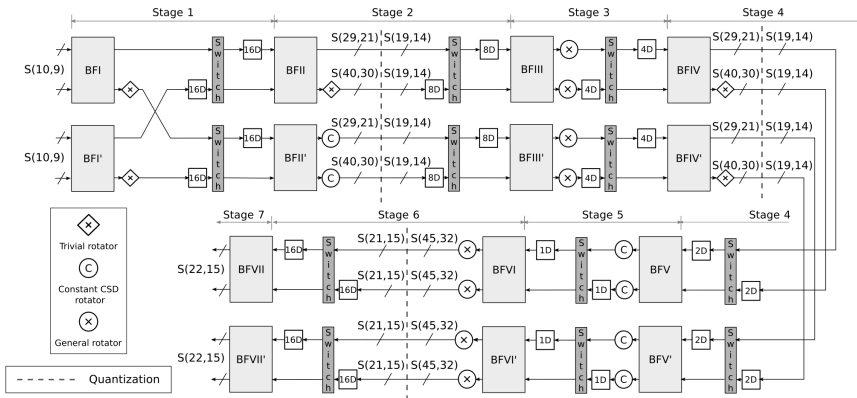
Folding architecture for radix-2<sup>3</sup> 128 points FFT.

## Sección 4

# Implementation

# Design of FFT architecture via folding transformation

Implementation: Parallel radix-2<sup>3</sup> 16-Points

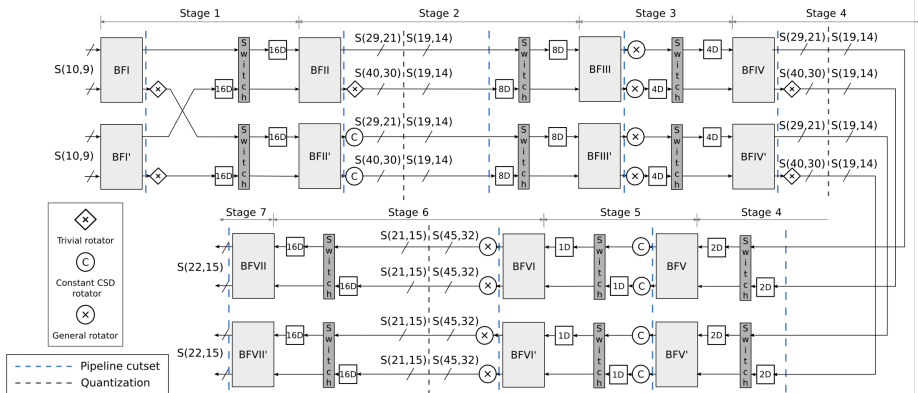


Folding architecture for radix-2<sup>3</sup> 128 points FFT with quantization.



# Design of FFT architecture via folding transformation

Implementation: Parallel radix-2<sup>3</sup> 16-Points



Folding architecture for radix-2<sup>3</sup> 128 points FFT with quantization and pipeline.

**Thanks!**