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

July 30, 2020

Fundación
FULGOR

Outline - Table of Contents

Table of Contents

Section 1

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[a] multipliers.
- Test the design with a mixture of two sinusoids using MATLAB.
- Generate power-area-timing report with different optimizations.

[a] Canonic Signed Digit

Introduction

Introduction:

Workflow

1. Obtain the equations that correspond to Butterfly structure of radix-2^3 FFT for 8 points.
2. Apply this idea to design a 2-parallel pipelined architecture radix-2^3 16-points FFT 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.

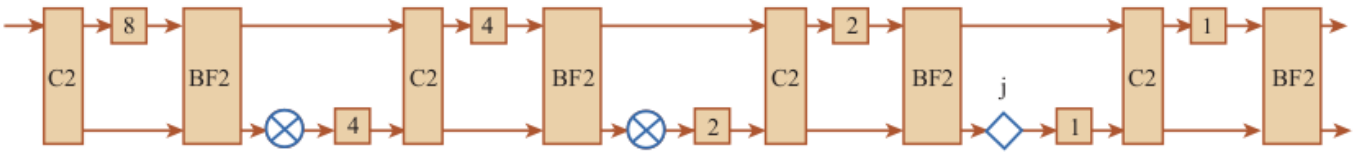
Introduction

Introduction:

Types of pipelined Radix-2 FFT architectures

1. Feedfoward 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 feedfoward MDC asrchitecture.



(1) . R2MDC(N-16)

(2) . R25DF(N-16)

Table of Contents

Section 2

The Radix-2^3 FFT Algorithm

The Radix-2^3 FFT Algorithm

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

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

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 $O(Nlog_2 N)$ for the FFT.

The Radix-2^3 FFT Algorithm

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*.
- This is based on the decomposition of an N point DFT into successively smaller DFTs.
- In our case, the numeber of stages is:

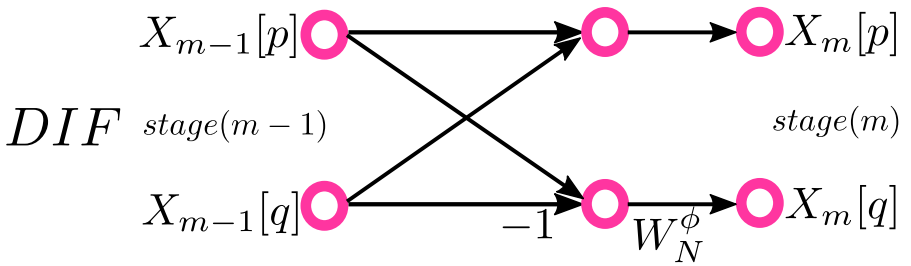
$$s = log_2(128) = 7 \tag{2}$$

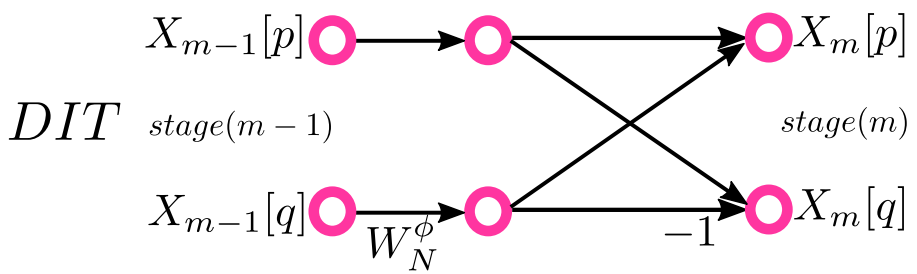
Methods to design FFT algorithms

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

DIT and DIF Butterflies

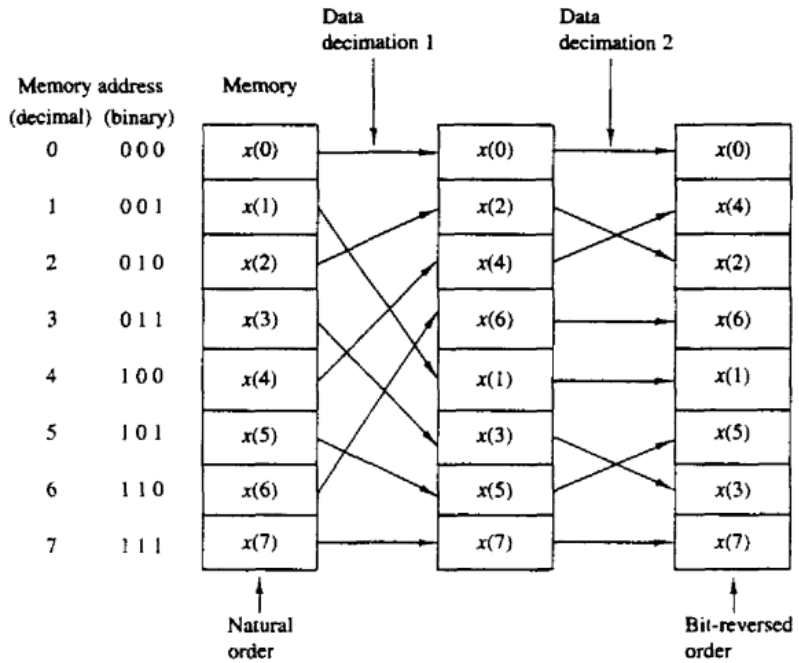
The difference is the instant in which the multiplication by W_N^{ϕ} is accomplished.





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 intpus and outputs for DIF FFT.

The Radix-2^3 FFT Algorithm

The Radix-2^3 FFT Algorithm:

Mathematic expression of radix-8 butterfly element

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

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

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

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

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

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

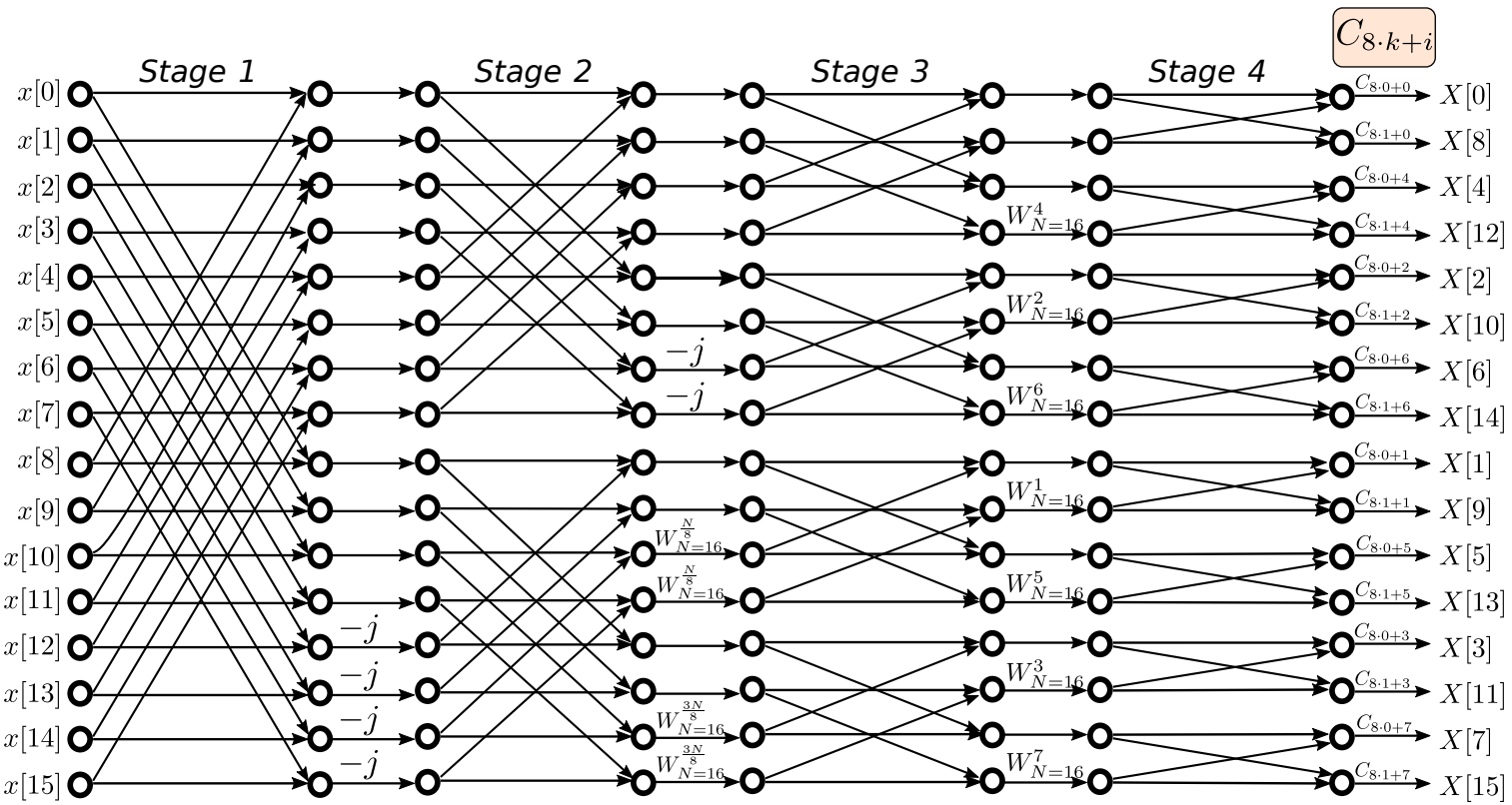
$$C_{8k+3} = \sum_{n=0}^{N/8-1} \left\{ [(x_n - x_{n+\frac{N}{2}}) + j(x_{n+\frac{N}{4}} - x_{n+\frac{3N}{4}})] + W_N^{3N/8}[(x_{n+\frac{N}{8}} - x_{n+\frac{5N}{8}}) + j(x_{n+\frac{3N}{8}} - x_{n+\frac{7N}{8}})] \right\} W_N^{3n} W_{N/8}^{nk}$$

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

The Radix-2^3 FFT Algorithm

Radix-2^k Implementation

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



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

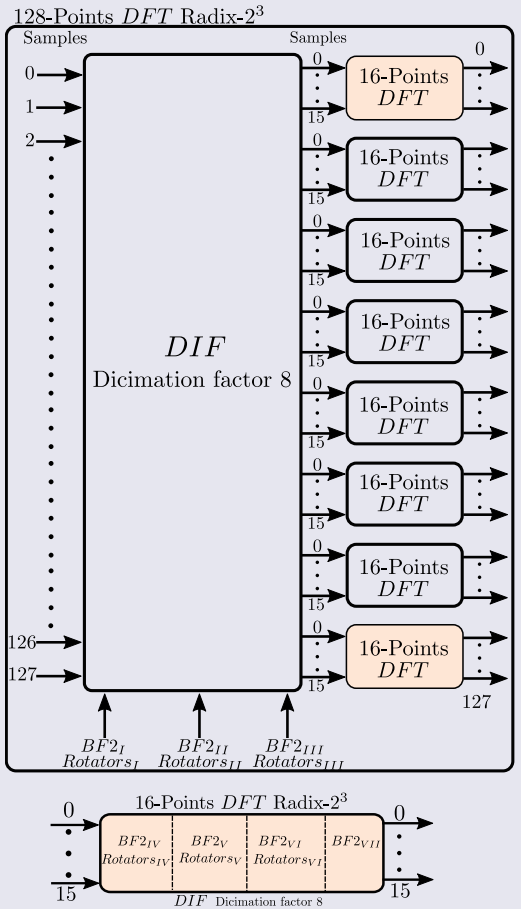
The Radix-2^3 FFT Algorithm

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

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

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.



Section 3

Design of FFT architecture via folding transformation

Design of FFT architecture via folding transformation

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 jth position (where 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 lth 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:

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

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-2^3 16-Points

Folding equations without retiming/pipeline

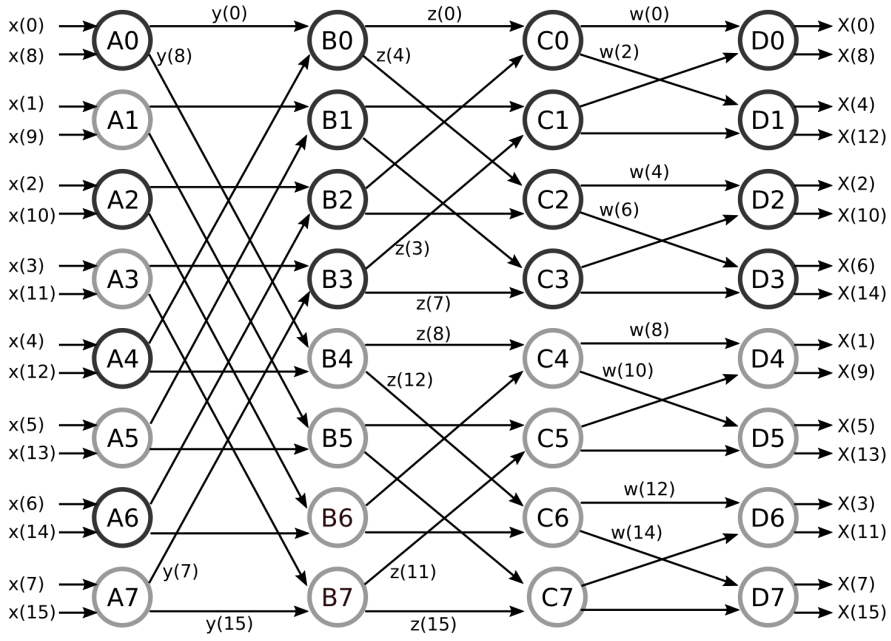
- Consider the folding sets:

$$A = \{A0, A2, A4, A6\} \quad A' = \{A1, A3, A5, A7\}$$
$$B = \{B1, B3, B0, B2\} \quad B' = \{B5, B7, B4, B6\}$$
$$C = \{C2, C1, C3, C0\} \quad C' = \{C6, C5, C7, C4\}$$
$$D = \{D3, D0, D2, D1\} \quad D' = \{D7, D4, D6, D5\}$$

- For example:

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

- The folding equations can be derived for all edges.



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

Design of FFT architecture via folding transformation

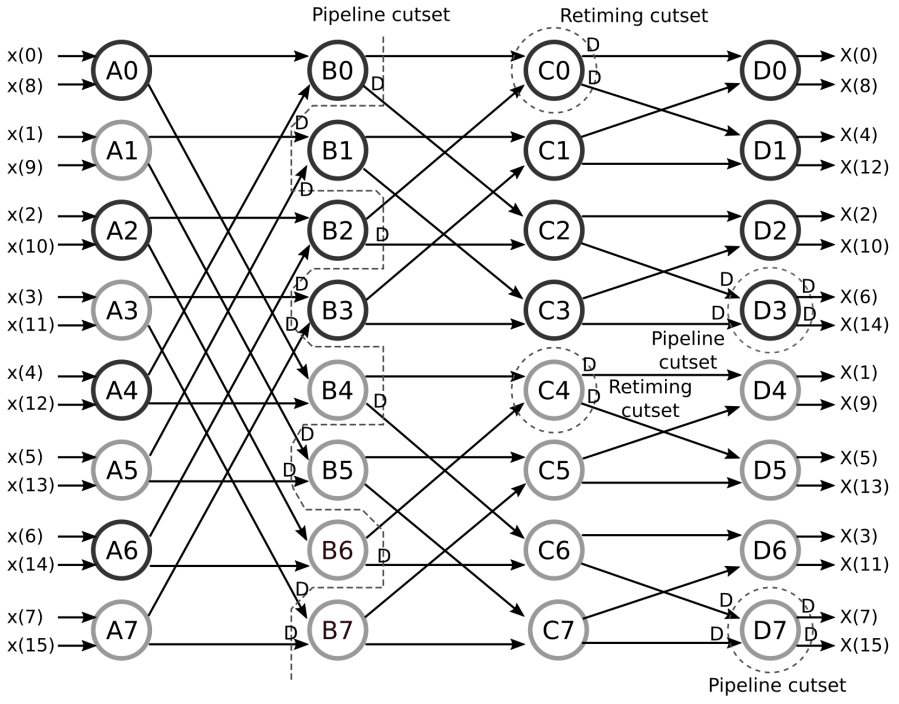
Design of FFT architecture via folding transformation: Parallel radix-2^3 16-Points

Folding equations with retimming/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-2^3 16-point DIF DFT applying pipeline and retiming.

Design of FFT architecture via folding transformation

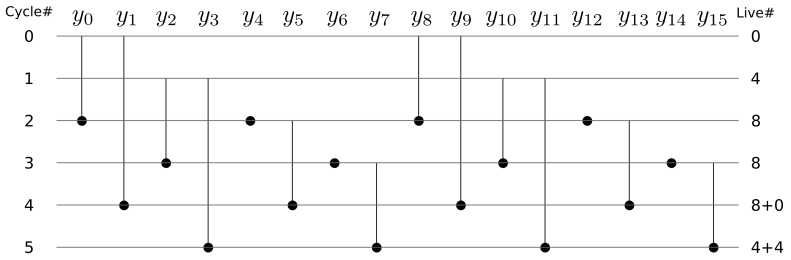
Design of FFT architecture via folding transformation: Parallel radix-2^3 16-Points

Lifetime analysis

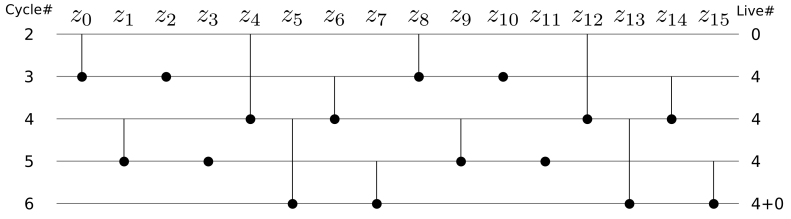
- Is a procedure used to compute the minimun number of registers.
- For example, the variable y_1 be live during time units $n \in \{1, 2, 3, 4\}$.
- The number of live variables y_i during the time units $\{1, 2, 3, 4, 5\}$ is $\{0, 4, 8, 8, 8, 8\}$. So the number of register for this stage is:

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

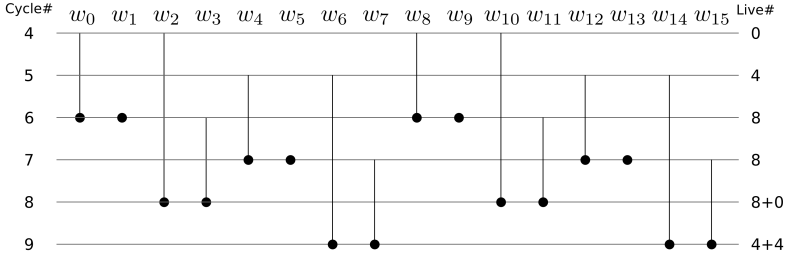
- The total number of registers is reduced from 80 to 20.



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-2^3 16-Points

Forward Register Allcation

- This dictates how the variables are assigned to the minimum numbers of registers.
- If R_i holds 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
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
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.

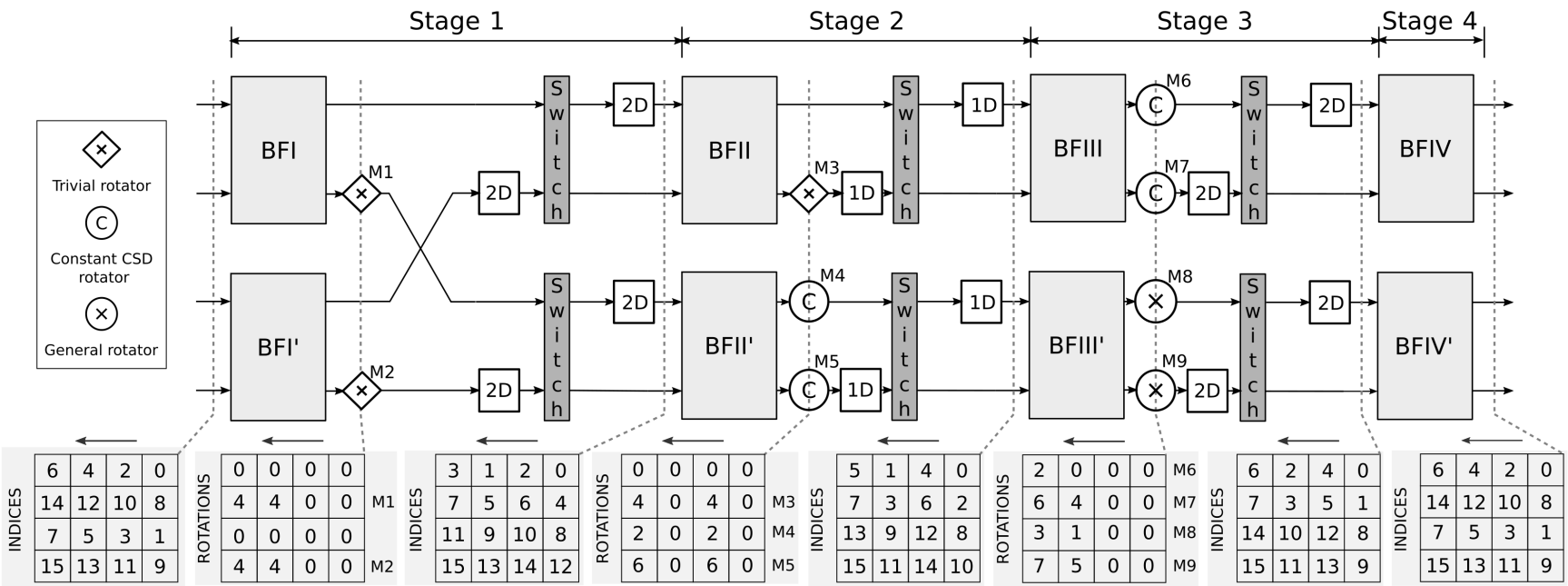
	I/P	R1	R2	R3	R4	R5	R6	R7	R8
4	w0,w2,w8,w10								
5	w4,w6,w12,w14	w2		w0		w10		w8	
6	w1,w3,w9,w11	w6	w2	w4	w0	w14	w10	w12	w8
7	w5,w7,w13,w15	w3	w6	w2	w4	w11	w14	w10	w12
8		w7	w3	w6	w2	w15	w11	w14	w10
9			w7		w6		w15		w14

Allocation table for stage 3.

Design of FFT architecture via folding transformation

Folding architecture for radix-2^3 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.



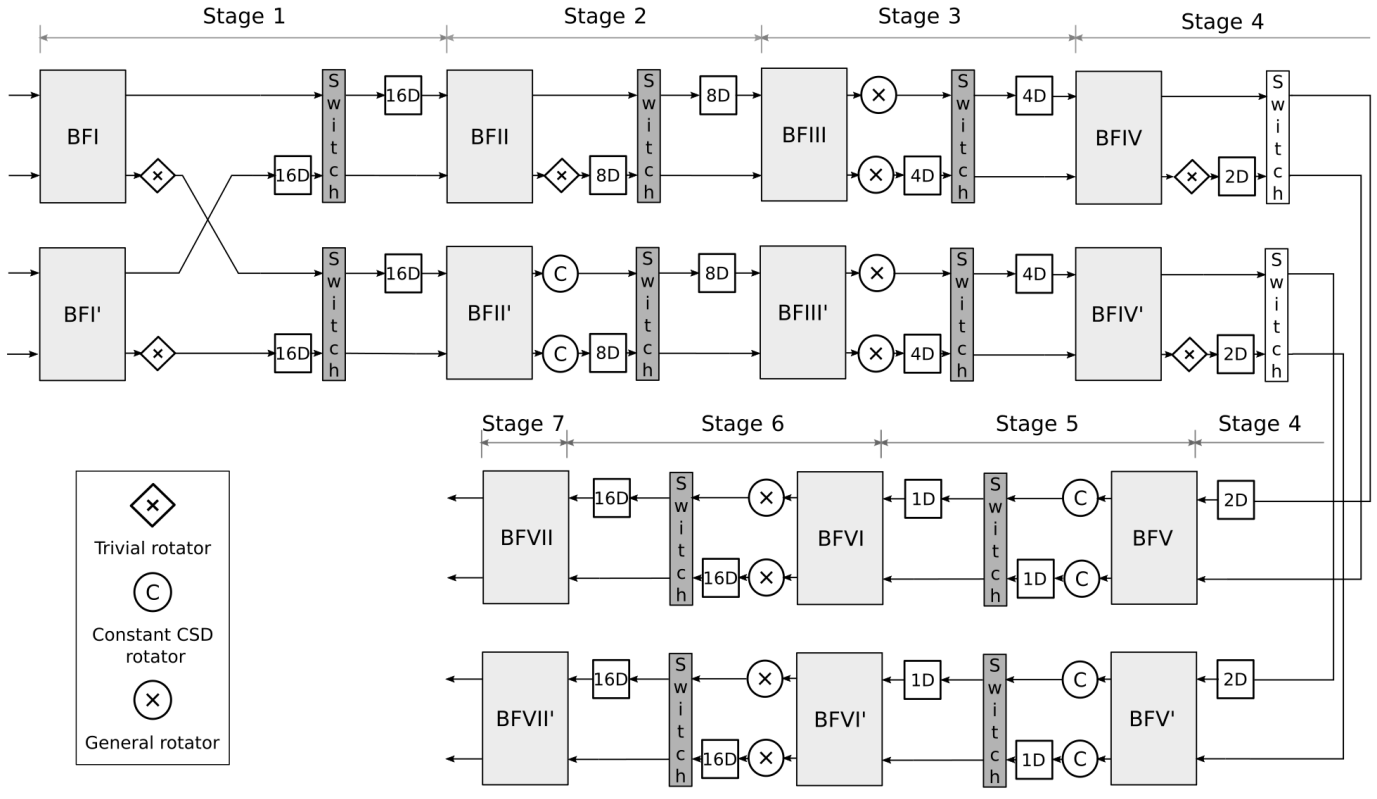
Folding architecture for radix-2^3 16 points FFT.

Design of FFT architecture via folding transformation

Design of FFT architecture via folding transformation: Parallel radix-2^3 16-Points

Folding architecture for radix-2^3 128 points FFT

We can deduce the folding architecture for 128 points radix-2^3 following the same method.



Folding architecture for radix-2^3 128 points FFT.

Table of Contents

Section 4

Implementation

Design of FFT architecture via folding transformation

Implementation: Floating and Fixed point Simulator

Signal Quantization

- The signals in each stage are quantized in order to obtain a high SQNR[a] ($\approx 50\,dB$).

$$SQNR_{dB} = 10log_{10}\left(\frac{Var\{Signal_{FloatPoint}\}}{Var\{Signal_{FloatPoint} - Signal_{FixedPoint}\}}\right)$$

- The quantization of the signals is:
  - Input: $S(10, 9)$, $SQNR \approx 57\,dB$.
  - Twiddle factors: $S(11, 9)$.
  - Output: $S(22, 15)$, $SQNR \approx 47\,dB$.

[a] Signal to Quantization Noise Ratio

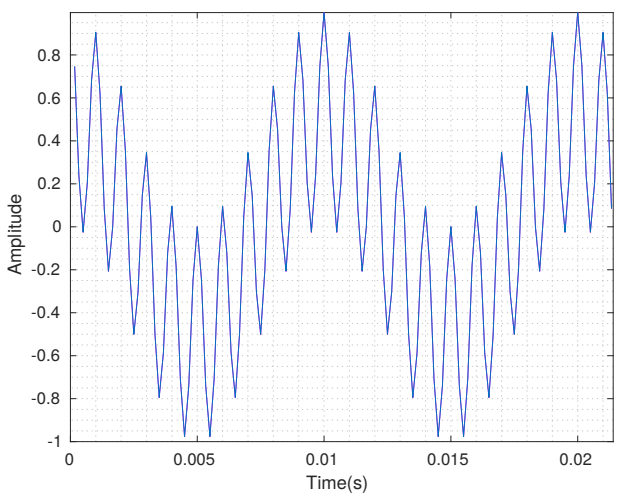
Design of FFT architecture via folding transformation

Implementation: Floating and Fixed point Simulator

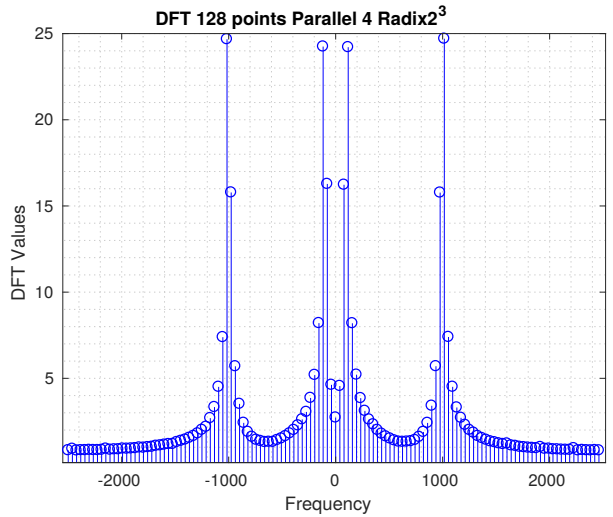
Testing signals

- The input signal will be a mixture of two sinusoid signals with frecuencies $f_1 = 100Hz$, $f_2 = 1000Hz$.

$$x'[n] = cos(2\pi f_1 n T_s) + cos(2\pi f_2 n T_s) \quad (4)$$
$$x[n] = x'[n]/max\{x'[n]\}$$



Input signal $x[n]$ in time domain.



Output samples, absolute value vs frequency $|X[k]|$.

Design of FFT architecture via folding transformation

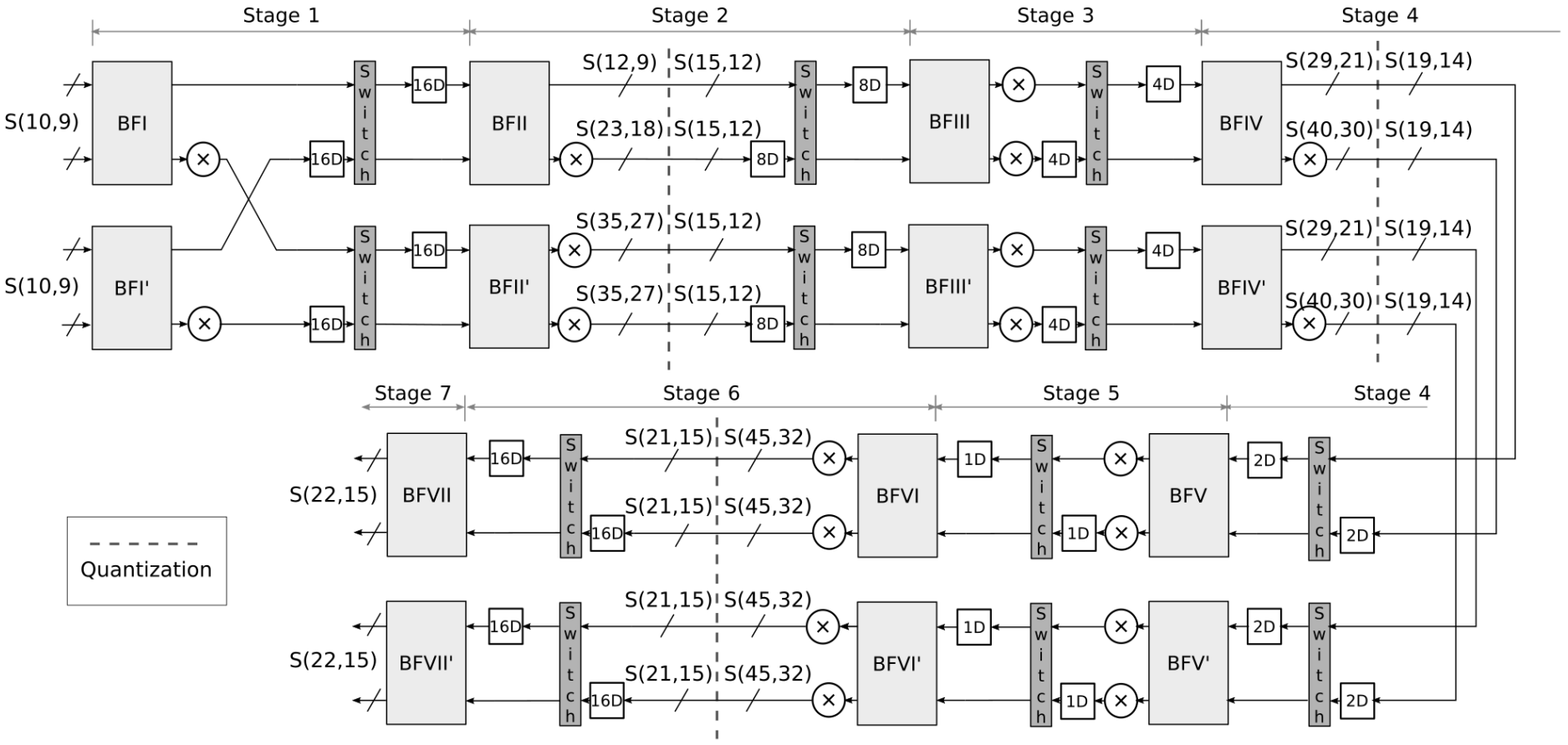
Implementation: Floating and Fixed point Simulator

Design Implementations

- Design 1:
  - Quantitions blocks placed after the multiplers in each stage.
  - Multiplicators are not optimized.
- Design 2:
  - A Pipeline cut-set is added after every quantization block.
- Design 3:
  - Implementation with Trival multiplicators for $-j$ factors.
- Design 4:
  - A Pipeline cut-set is added inside of each butterfly block.
  - Implementation of CSD multipliers.

Design of FFT architecture via folding transformation

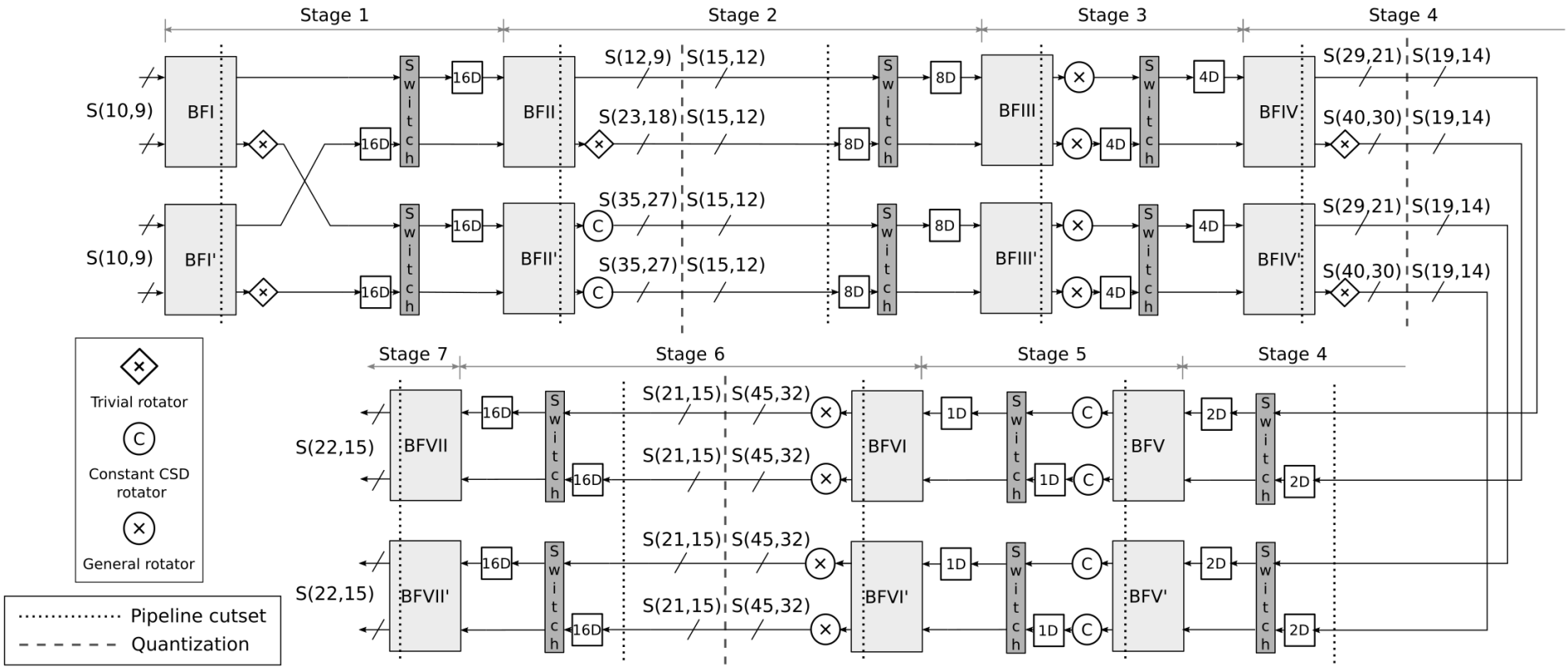
Implementation: Floating and Fixed point Simulator



Folding architecture for radix-2^3 128 points FFT with quatization (Design 1).

Design of FFT architecture via folding transformation

Implementation: Floating and Fixed point Simulator



Folding architecture for radix-2^3 128 points FFT with quantization and pipeline (Design 4).

Design of FFT architecture via folding transformation

Implementation: Results

Timing Report at 500MHz

Point	Path(ns)
data arrival time	5.60
clock CLK (rise edge)	2.00
clock network delay (ideal)	2.00
library setup time	1.95
data required time	1.95
data arrival time	-5.60
slack (VIOLATED)	-3.65

Design instance 1.

Point	Path(ns)
data arrival time	2.71
clock CLK (rise edge)	2.00
clock network delay (ideal)	2.00
library setup time	1.95
data required time	1.95
data arrival time	-2.71
slack (VIOLATED)	-0.76

Design instance 2.

Point	Path(ns)
data arrival time	2.76
clock CLK (rise edge)	2.00
clock network delay (ideal)	2.00
library setup time	1.95
data required time	1.95
data arrival time	-2.76
slack (VIOLATED)	-0.81

Design instance 3.

Point	Path(ns)
data arrival time	1.94
clock CLK (rise edge)	2.00
clock network delay (ideal)	2.00
library setup time	1.94
data required time	1.94
data arrival time	-1.94
slack (MET)	0.00

Design instance 4.

Design of FFT architecture via folding transformation

Implementation: Results

Area Report at 500MHz

Logical Elements	
Number of ports	1228
Number of nets	112376
Number of cells	102726
Number of combinational cells	95310
Number of sequential cells	7404
Number of macros/black boxes	0
Number of buf/inv	27817
Combinational area	291201.116637
Buf/Inv area	44892.767764
Noncombinational area	58830.508095
Total cell area	350031.624731

Design instance 1.

Logical Elements	
Number of ports	1826
Number of nets	114367
Number of cells	104198
Number of combinational cells	96271
Number of sequential cells	7912
Number of macros/black boxes	0
Number of buf/inv	26770
Combinational area	294733.068172
Buf/Inv area	41738.133324
Noncombinational area	62955.185703
Total cell area	357688.253875

Design instance 2.

Logical Elements	
Number of ports	1571
Number of nets	94523
Number of cells	87311
Number of combinational cells	80220
Number of sequential cells	7076
Number of macros/black boxes	0
Number of buf/inv	23823
Combinational area	233949.801414
Buf/Inv area	36323.350042
Noncombinational area	56229.647552
Total cell area	290179.448967

Design instance 3.

Logical Elements	
Number of ports	2315
Number of nets	75713
Number of cells	66284
Number of combinational cells	58017
Number of sequential cells	8240
Number of macros/black boxes	0
Number of buf/inv	14789
Combinational area	195877.841872
Buf/Inv area	24429.880240
Noncombinational area	64463.046570
Total cell area	260340.888442

Design instance 4.

Design of FFT architecture via folding transformation

Implementation: Results

Power Report at 500MHz

Power Group	Internal	Switching	Leakage	Total Power
io pad	0.0000	0.0000	0.0000	0.0000
clock network	34.8220	603.2268	1.4573e+06	639.6328
register	54.2228	0.2699	4.0540e+05	54.8980
sequential	0.0000	0.0000	0.0000	0.0000
combinational	0.2884	0.7304	1.5016e+04	1.0337
Total	89.333mW	604.227mW	1.877e+06nW	695.564mW

Design instance 1.

Power Group	Internal	Switching	Leakage	Total Power
io pad	0.0000	0.0000	0.0000	0.0000
clock network	36.5175	603.3234	1.3702e+06	641.2228
register	57.8422	0.2604	4.3383e+05	58.5363
sequential	0.0000	0.0000	0.0000	0.0000
combinational	0.5748	1.4180	1.5702e+05	2.1498
Total	94.934mW	605.001mW	1.961e+06nW	701.908mW

Design instance 2.

Power Group	Internal	Switching	Leakage	Total Power
io pad	0.0000	0.0000	0.0000	0.0000
clock network	24.2875	583.5885	1.0269e+06	608.9492
register	51.0349	0.1728	3.8748e+05	51.5952
sequential	0.0000	0.0000	0.0000	0.0000
combinational	0.9554	1.1271	1.8640e+05	2.2689
Total	76.277mW	584.888mW	1.600e+06nW	662.813mW

Design instance 3.

Power Group	Internal	Switching	Leakage	Total Power
io pad	0.0000	0.0000	0.0000	0.0000
clock network	18.1655	581.6307	7.8320e+05	600.6672
register	58.2275	0.2873	4.4422e+05	58.9591
sequential	0.0000	0.0000	0.0000	0.0000
combinational	0.7768	0.9958	1.5970e+05	1.9322
Total	77.169mW	582.913mW	1.387e+06nW	661.558mW

Design instance 4.

Questions

Thanks!