Fast Fourier Transform on FPGA: Design Choices and Evaluation

Peter A. Milder, Franz Franchetti, James C. Hoe, Markus Püschel

SPIRAL www.spiral.net

Carnegie Mellon Electrical & Computer ENGINEERING

Problem Statement

- **The discrete Fourier transform (DFT) is among the most important tools in signal processing**
- **DFT has many algorithms (FFTs) and design choices**
- **How to represent, generate, and evaluate the design space for given user constraints?**
- **Results: 1) FFT IP core generator: "point and click"**

2) FFT implementation guidelines

Abstraction Level	Options	Objectives / Constraints	Suggestions
Algorithmic	algorithm	-minimize latency or cost -maximize throughput	Pease FFT Iterative FFT
	radix	-reduce cost	find by exploration (typically 2, 4, or 8)
Architectural	horizontal-reuse	-minimize latency or cost -maximize throughput	yes no, fully-streamed instead
	streaming width	-balance cost/performance	set to desired tradeoff
FPGA-mapping	complex multiply	-mult. blocks plentiful -otherwise	4 mults, 2 adds 3 mults, 5 adds
	permutation	-BRAM plentiful -otherwise	memory-based method FIFO-based method

Algorithmic Level

Discrete Fourier Transform (DFT)

$$y = \mathrm{DFT}_n\, x, \quad \mathrm{DFT}_n = [(\exp(2\pi\sqrt{-1}/n)^{k\ell}]_{0\le k,\ell<n}$$

Fast Fourier Transform (FFT) Algorithms

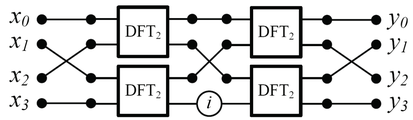
- Matrix factorization

$$\mathrm{DFT}_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{bmatrix} = \begin{bmatrix} 1 & . & 1 & . \\ . & 1 & . & 1 \\ 1 & . & -1 & . \\ . & 1 & . & -1 \end{bmatrix} \begin{bmatrix} 1 & . & . & . \\ . & 1 & . & . \\ . & . & 1 & . \\ . & . & . & i \end{bmatrix} \begin{bmatrix} 1 & 1 & . & . \\ 1 & -1 & . & . \\ . & . & 1 & 1 \\ . & . & 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & . & . & . \\ . & . & 1 & . \\ . & 1 & . & . \\ . & . & . & 1 \end{bmatrix}$$

- Representation as matrix formula

$$\mathrm{DFT}_4 = (\mathrm{DFT}_2 \otimes I_2) T_2^4 (I_2 \otimes \mathrm{DFT}_2) L_{4,2}$$

- Formula describes combinational datapath



Pease FFT [2]:

$$\mathrm{DFT}_{r^t} = \left[\prod_{k=0}^{t-1} L_{r^t,r}\,(I_{r^{t-1}} \otimes \mathrm{DFT}_r)\,D_{n,k}\right] R_{r^t,r}$$

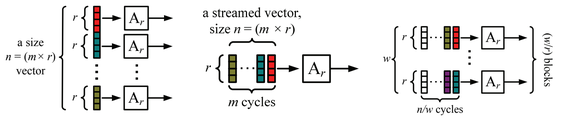
explicit parallelism; scaling; stride permutation; basic block of radix r; bit reversal

Iterative FFT [3]:

$$\mathrm{DFT}_{r^t} = L_{r^t,r}\Big(\prod_{k=0}^{t-2} (I_{r^{t-1}} \otimes \mathrm{DFT}_r)\,D_{n,k}(I_{r^k} \otimes L_{r^{t-k},r^{t-k-1}})$$
$$(I_{r^{k+1}} \otimes L_{r^{t-k-1},r})\Big)\,(I_{r^{t-1}} \otimes \mathrm{DFT}_r)\,R_{r^t,r}$$

Architectural Level

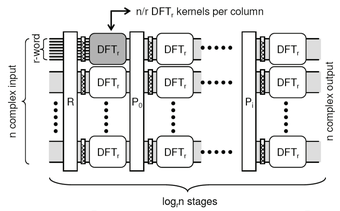
Formal View of Streaming



width = n | width = r | r < width < n

$$I_m \otimes A_r \qquad I_m \otimes^{\mathrm{sr}} A_r \qquad I_{n/w} \otimes^{\mathrm{sr}} (I_{w/r} \otimes A_r)$$

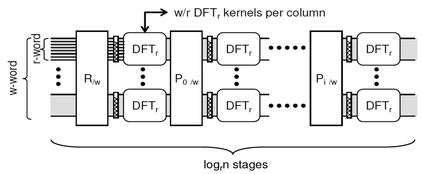
Combinational Datapath



$$\mathrm{DFT}_{r^t} = \left[\prod_{k=0}^{t-1} L_{r^t,r}\,(I_{r^{t-1}} \otimes \mathrm{DFT}_r)\,D_{n,k}\right] R_{r^t,r}$$

Fold vertically

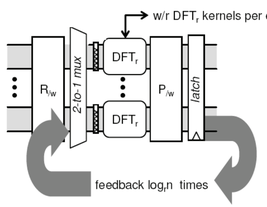
Streaming: Throughput Optimized (Iterative FFT)



$$\mathrm{DFT}_{r^t} = \underbrace{L_{r^t,r}}_{\mathrm{stream}(w)} \Big(\prod_{k=0}^{t-2} (I_{n/w} \otimes^{\mathrm{sr}} (I_{w/r} \otimes \mathrm{DFT}_r))\ \underbrace{D_{n,k}}_{\mathrm{stream}(w)}\ \underbrace{P_i}_{\mathrm{stream}(w)}\Big)$$
$$\cdot (I_{n/w} \otimes^{\mathrm{sr}} (I_{w/r} \otimes \mathrm{DFT}_r))\ \underbrace{R_{r^t,r}}_{\mathrm{stream}(w)}$$

Fold horizontally

Horizontal Reuse: Latency optimized (Pease FFT)

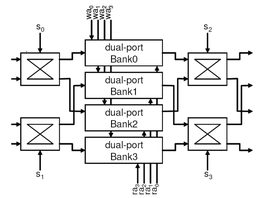


$$\underbrace{\mathrm{DFT}_{r^t}}_{\mathrm{stream}(w)} = \Big[\prod_{k=0}^{t-1}{}^{\mathrm{hr}}\ \underbrace{L_{r^t,r}}_{\mathrm{stream}(w)}\ (I_{n/w} \otimes^{\mathrm{sr}} (I_{w/r} \otimes \mathrm{DFT}_r))\ \underbrace{D_{n,k}}_{\mathrm{stream}(w)}\Big]\ \underbrace{R_{r^t,r}}_{\mathrm{stream}(w)}$$

FPGA Mapping

Stride permutation

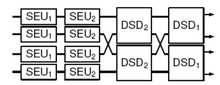
- **Method 1: RAM-Based [4]**



Example: $L_{256,2}$ with w = 4 ports

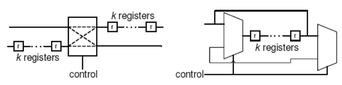
property	cost
storage	2n words
logic	low, "optimal"
control	low, "optimal"

- **Method 2: FIFO-Based [5]**



(a) $L_{32,8}$ with streaming width $w = 4$

property	cost
storage	n/2 words, "optimal"
logic	high
control	high

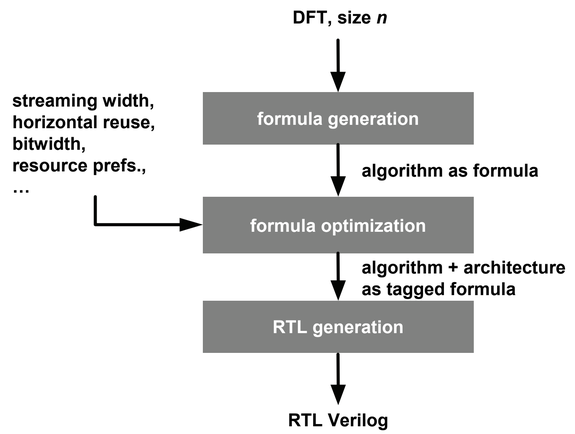


(b) DSD$_k$ block **(c) SEU$_k$ block**

Other FPGA-Mapping Options

- Complex multiplication (2 options)
- Twiddle factor storage (3 options)

FFT IP Core Generator



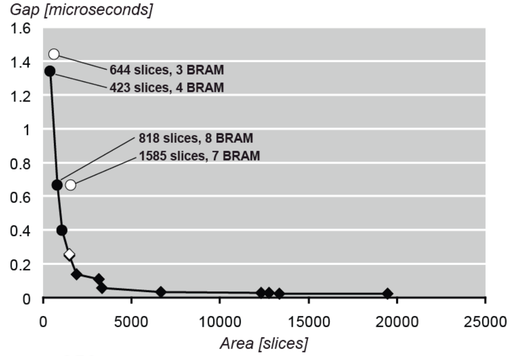
Prototype at: www.spiral.net/hardware/dftgen.html

Evaluation

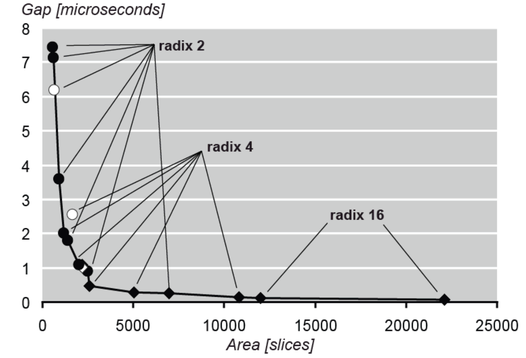
- Synthesis: Xilinx ISE version 8.1i
- Spiral generated FFT IP cores vs. Xilinx LogiCore FFT 3.2
- Gap (1 / throughput) versus area
- Pareto-optimal points

☐ Xilinx LogiCore ■ Spiral Generated ◆ (diamond): streaming only ● (circle): streaming + hor. reuse

n = 64



n = 256



Cost / performance comparable to benchmarks. High degree of control over tradeoffs.

References

1. C. Van Loan. **Computational Framework of the Fast Fourier Transform.** SIAM, 1992.
2. M. C. Pease. **An Adaptation of the fast Fourier transform for parallel processing**. *ACM*, 15(2), April 1968.
3. P. A. Milder, F. Franchetti, J. C. Hoe, and M. Püschel. **Discrete Fourier transform compiler: from mathematical representation to efficient hardware.** CSSI Technical Report #CSSI 07-01, Carnegie Mellon University, January 2007. Available at http://www.ece.cmu.edu/~pam/papers/dftcomp.pdf.
4. M. Püschel, P. A. Milder, and J. C. Hoe. **Permuting streaming data using RAMs.** Journal submission under preparation.
5. T. Järvinen, P. Salmela, H. Sorokin, and J. Takala. **Stride permutation networks for array processors**. In Proc. IEEE Intl. Conf. on Application-Specific Systems, Architectures and Processors, 2004.

This work was supported by DARPA under DOI grant NBCH-1050009 and by NSF awards ACR-0234293 and ITR/ACI-0325687.