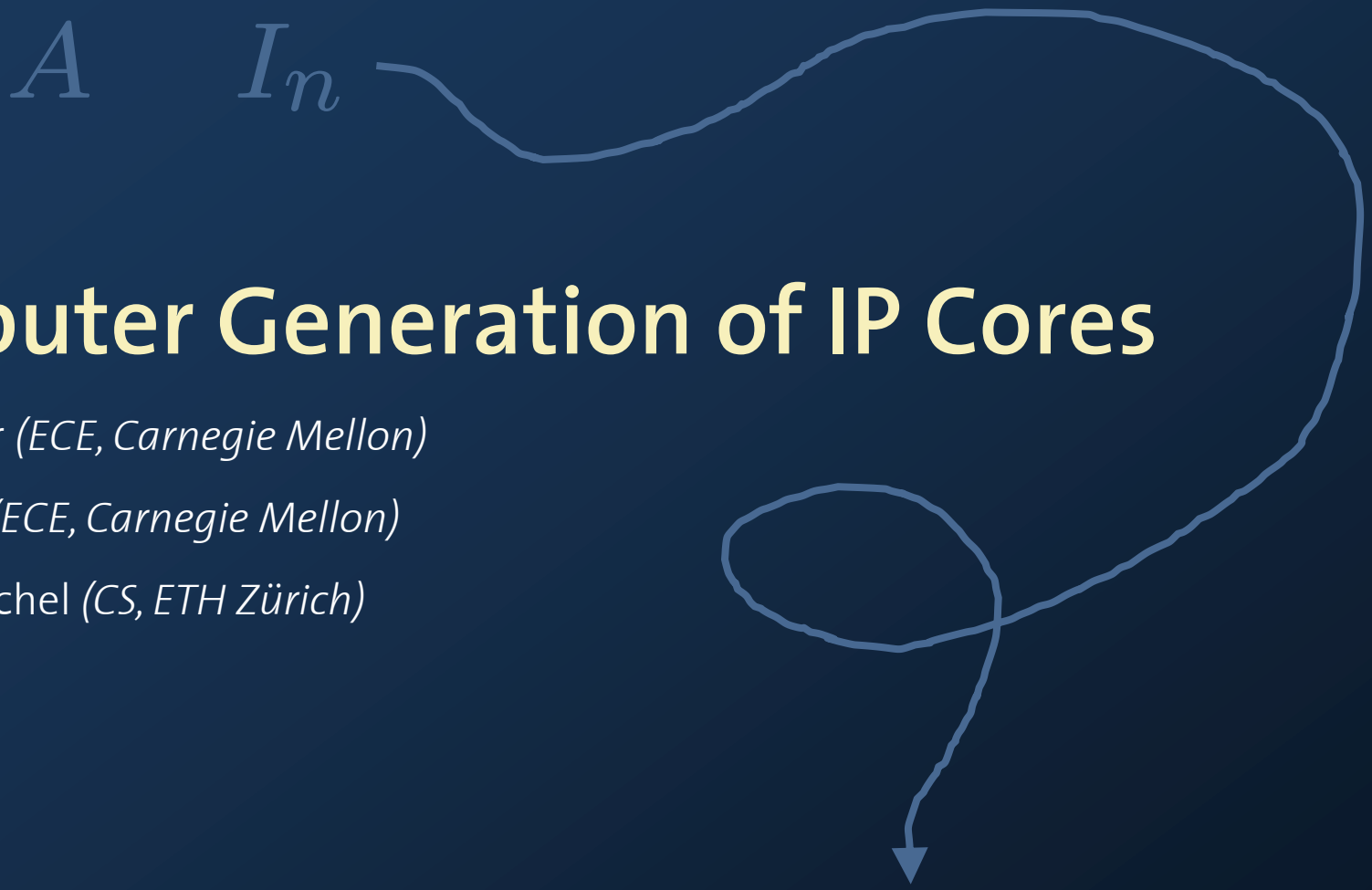


Computer Generation of IP Cores

Peter Milder *(ECE, Carnegie Mellon)*

James Hoe *(ECE, Carnegie Mellon)*

Markus Püschel *(CS, ETH Zürich)*

SPIRAL
www.spiral.net

Software

Libraries

C/C++/Fortran

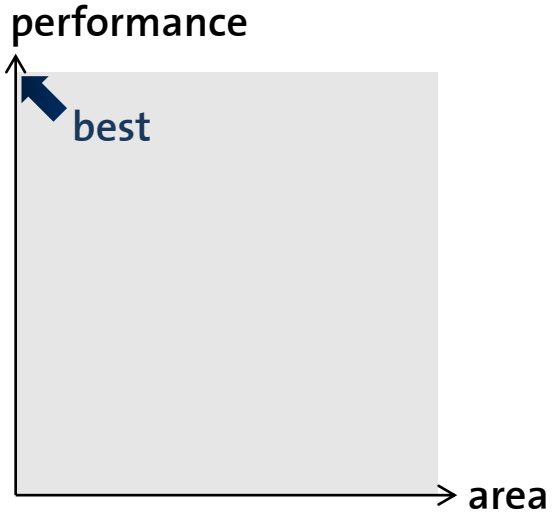
Compilation

Performance

Software	Hardware (FPGA, ASIC)
Libraries	IP Cores
C/C++/Fortran	Verilog/VHDL
Compilation	Synthesis
Performance	Area/Performance

Example:
Discrete Fourier transform (DFT), size 64

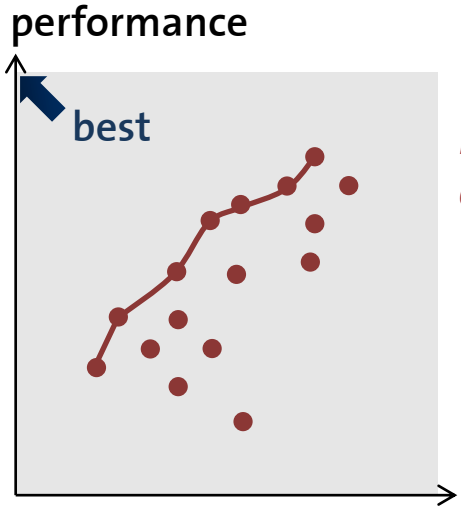
6.12 Gigasamples/second



Software	Hardware (FPGA, ASIC)
Libraries	IP Cores
C/C++/Fortran	Verilog/VHDL
Compilation	Synthesis
Performance	Area/Performance

Example:
Discrete Fourier transform (DFT), size 64

6.12 Gigasamples/second



Pareto-optimal designs

Problem: *For a given function, how to efficiently obtain the Pareto-optimal designs?*

Solution: *IP core generator to enumerate design space*

- DSL to represent algorithms
- Rewriting on DSL for architectural decisions
- Compiler: DSL to RTL Verilog

Current IP Core Generators (www.spiral.net)

DFT, other transforms, sorting

parameter	value	range	explanation
Problem specification			
transform size	64	4–32768	Number of samples (?)
direction	forward		forward or inverse DFT (?)
data type	fixed point		fixed or floating point (?)
	16 bits	4–32 bits	fixed point precision (?)
	unscaled		scaling mode (?)
Parameters controlling implementation			
architecture	fully streaming		iterative or fully streaming (?)
radix	2	2, 4, 8, 16, 32, 64	size of DFT basic block (?)
streaming width	2	2–64	number of complex words per cycle (?)
data ordering	natural in / natural out		natural or digit-reversed data order (?)
BRAM budget	1000		maximum # of BRAMs to utilize (-1 for no limit) (?)

Generate Verilog eset

Current IP Core Generators (www.spiral.net)

DFT, other transforms, sorting

parameter	value	range	explanation
Problem specification			
transform size	64	4–32768	Number of sample
direction	forward		forward or inverse
data type	fixed point		fixed or floating po
	16 bits	4–32 bits	fixed point precisio
	unscaled		scaling mode (?)
Parameters controlling implementation			
architecture	fully streaming		iterative or fully st
radix	2	2, 4, 8, 16, 32, 64	size of DFT basic b
streaming width	2	2–64	number of complex
data ordering	natural in / natural out		natural or digit-rev
BRAM budget	1000		maximum # of BRA limit) (?)

Generate Verilog Reset

Result

Selected parameters:

DFT Size = 64
direction = forward
data type = 16 bit fixed point, unscaled
architecture = fully streaming
radix = 2
streaming width = 2
data ordering = natural input / natural output
BRAM budget = 1000

Output:

Download Verilog

Performance information:

Input/output stream: 2 complex words per cycle
Throughput: one transform every 32 cycles
Latency: 145 cycles

Resources required:

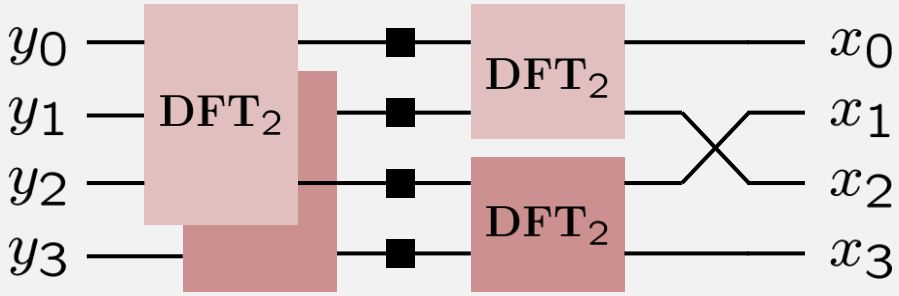
- 20 multipliers (16 x 16 bit)
- 34 adders (16 x 16 bit)
- 2 RAMs (8 words, 32 bits per word)
- 2 RAMs (16 words, 32 bits per word)
- 6 RAMs (64 words, 32 bits per word)
- 2 RAMs (32 words, 32 bits per word)
- 2 ROMs (16 words, 16 bits per word)
- 2 ROMs (8 words, 16 bits per word)
- 2 ROMs (32 words, 16 bits per word)

DFT and FFT: Example n = 4

DFT: *Fast Fourier transform (FFT):*

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{bmatrix} x = \begin{bmatrix} 1 & \cdot & 1 & \cdot \\ \cdot & 1 & \cdot & 1 \\ 1 & \cdot & -1 & \cdot \\ \cdot & 1 & \cdot & -1 \end{bmatrix} \begin{bmatrix} 1 & \cdot & \cdot & \cdot \\ \cdot & 1 & \cdot & \cdot \\ \cdot & \cdot & 1 & \cdot \\ \cdot & \cdot & \cdot & i \end{bmatrix} \begin{bmatrix} 1 & 1 & \cdot & \cdot \\ 1 & -1 & \cdot & \cdot \\ \cdot & \cdot & 1 & 1 \\ \cdot & \cdot & 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & \cdot & \cdot & \cdot \\ \cdot & \cdot & 1 & \cdot \\ \cdot & 1 & \cdot & \cdot \\ \cdot & \cdot & \cdot & 1 \end{bmatrix} x$$

FFT data flow graph



Description with matrix algebra (SPL)

$$\mathrm{DFT}_4 = (\mathrm{DFT}_2 \otimes \mathrm{I}_2)\,\mathrm{T}^4_2(\mathrm{I}_2 \otimes \mathrm{DFT}_2)\,\mathrm{L}^4_2$$

Algorithms

$$\mathrm{DFT}_{r^t} \to \left(\prod_{\ell=0}^{t-1} L_r^{r^t} \cdot (I_{r^{t-1}} \otimes \mathrm{DFT}_r) \cdot C_\ell^{r^t} \right) \cdot R_r^{r^t}$$

$$\mathrm{DFT}_{r^t} \to L_r^{r^t} \left(\prod_{\ell=0}^{t-1} (I_{r^{t-1}} \otimes \mathrm{DFT}_r) \left(I_{r^\ell} \otimes \left(E_\ell^{r^{t-\ell}} \cdot Q_\ell^{r^{t-\ell}} \right) \right) \right) R_r^{r^t}$$

$$\mathrm{DFT}_{r^k s^\ell} \to L_{r^k}^{r^k s^\ell} \cdot (I_{s^\ell} \otimes \mathrm{DFT}_{r^k}) \cdot L_{s^\ell}^{r^k s^\ell} \cdot T_{s^\ell}^{r^k s^\ell} \cdot (I_{r^k} \otimes \mathrm{DFT}_{s^\ell}) \cdot L_{r^k}^{r^k s^\ell}$$

DFT_n

$\mathrm{DFT\text{-}2D}_{n \times n}$

RDFT_n

$\mathrm{RDFT}_n \to (F_2 \oplus I_{n-2}) \cdot \mathrm{rDFT}_n(u)$

$\mathrm{rDFT}_{2km}(u)$

Each one describes a data flow graph and thus could be directly mapped to hardware.

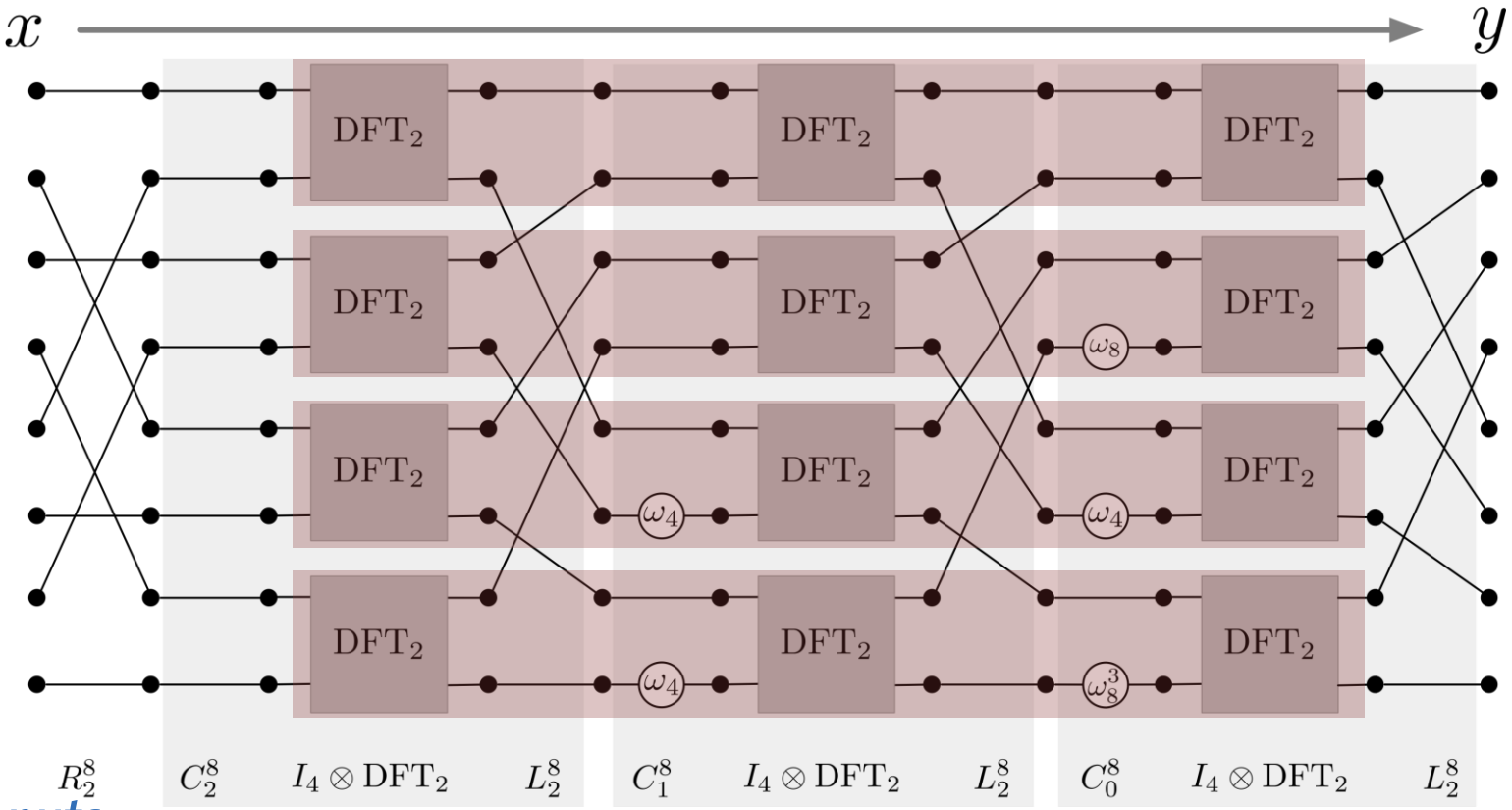
But how to obtain a design space?

$$\mathrm{rDFT}_{2km}(0) \to V_{k,m}^{2km} \left(\prod_{i=0}^{\log_k m} (I_m \otimes_\ell \mathrm{rDFT}_{2k}(s_{k,q}(0,0,f_{k,m}(i,\ell)))\, L_{2m}^{2km} \right) L_{km}^{2km}$$

$$\mathrm{rDFT}_{2km}(0) \to V_{k,m}^{2km} \cdot \left(\prod_{i=0}^{q} \left(I_{m/k^i} \otimes W_i^{2 \cdot k^{i+1}} \right) \left(I_m \otimes_\ell \mathrm{rDFT}_{2k}(s_{k,q-i}(0,0,\lfloor \ell/k^i \rfloor)) \right) \right) \cdot L_m^{2km}$$

$$\mathrm{DCT\text{-}2}_{2^k} \to \sqrt{\frac{2}{2^k}} \cdot U_{2^k} \cdot \left(\prod_{s=k-1}^{0} S_{2^k}^{(s)} (I_{2^{k-s-1}} \otimes L_{2^s}^{2^{s+1}}) \right) P_{2^k}^H.$$

Design Space: Example 8 Point FFT

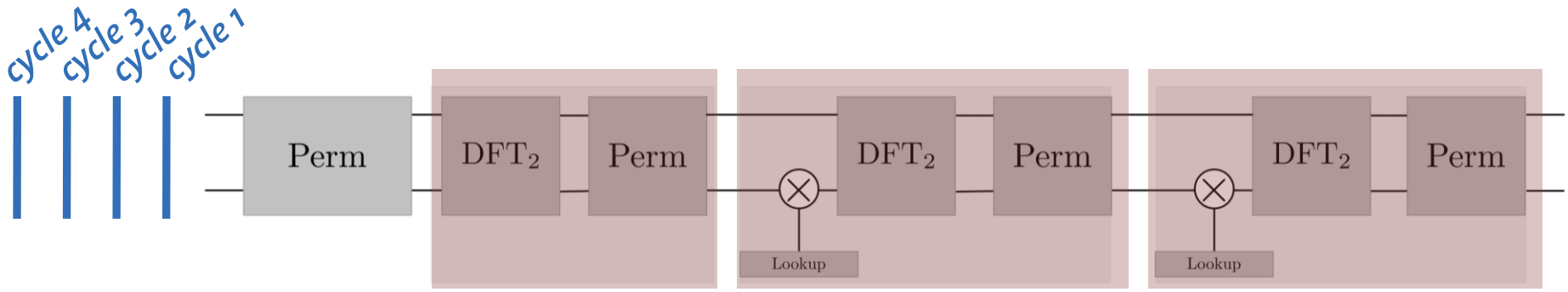


All 8 inputs in parallel

Parallelism allows folding

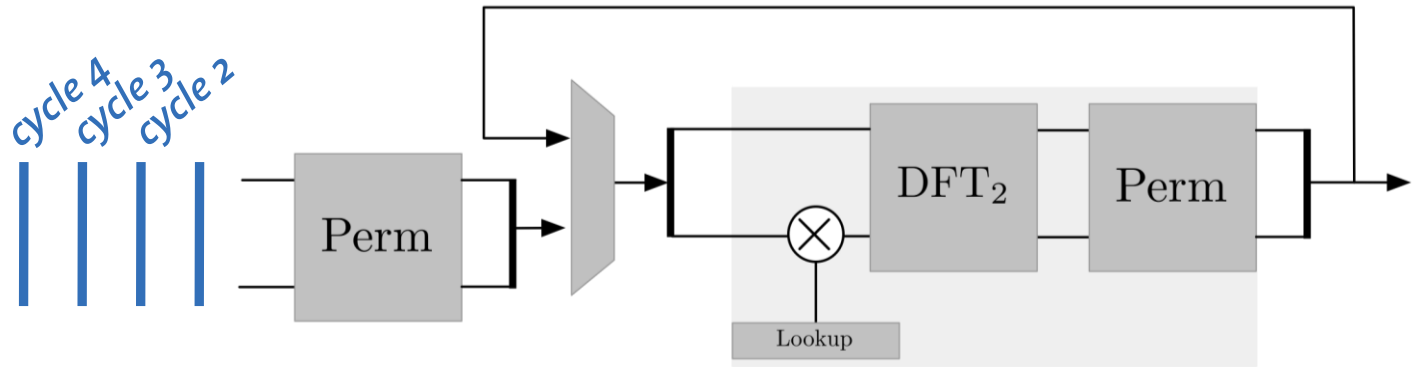
Design Space: Example 8 Point FFT

Two inputs in parallel, streamed over four cycles



Repeated stages allow folding

Design Space: Example 8 Point FFT



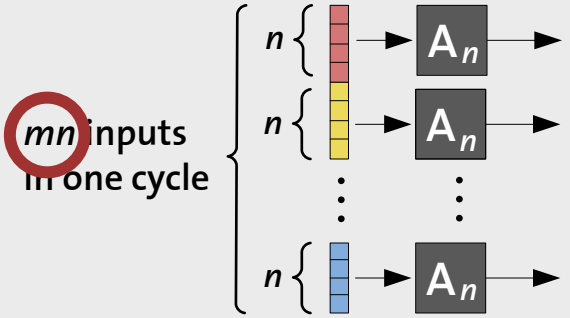
- *Vertical and horizontal folding:*
  Tradeoff between area cost and performance
- How to do formally and systematically?

Parallelism: $I_m \otimes A_n$

streaming width w

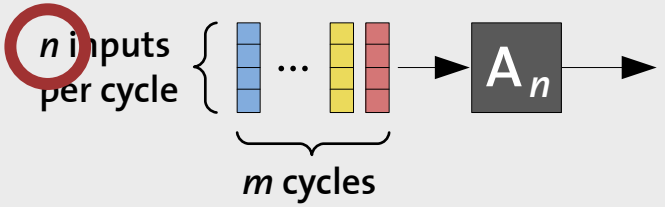
structurally parallel

$$I_m \otimes A_n$$



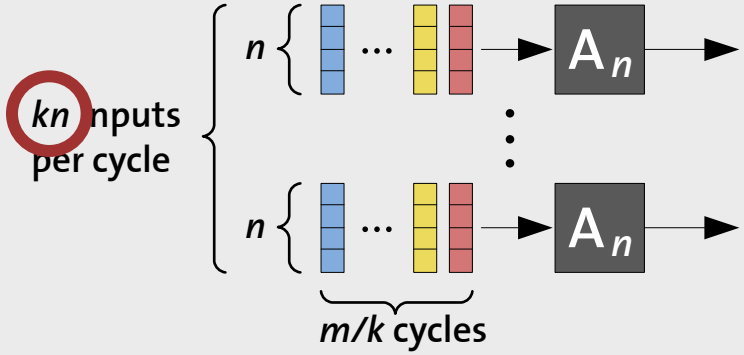
streaming: full reuse

$$I_m \otimes^{\mathrm{sr}} A_n$$



streaming: partial reuse

$$I_{m/k} \ {}^{\mathrm{sr}}\otimes \ (I_k \ A_n)$$

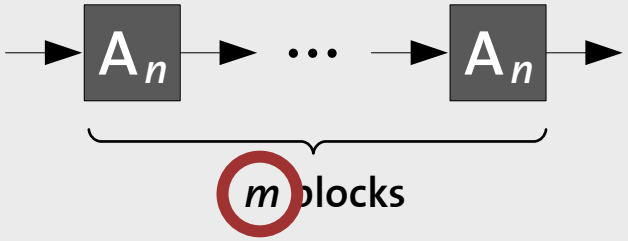


Repeated stages: $\prod_m A_n$

depth d

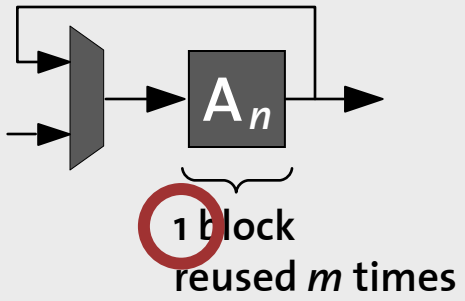
cascade

$$\prod_{\ell=0}^{m-1} A_n$$



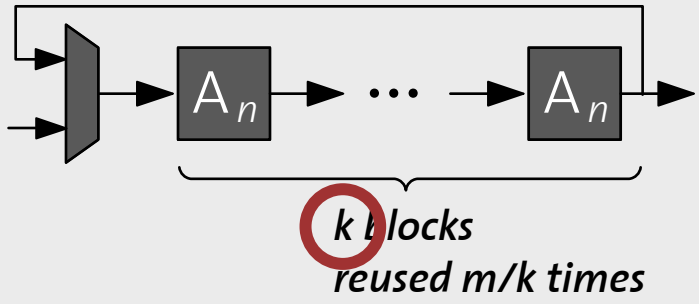
iterative: full reuse

$$\prod_{\ell=0}^{m-1}{}^{\text{ir}}\, A_n$$

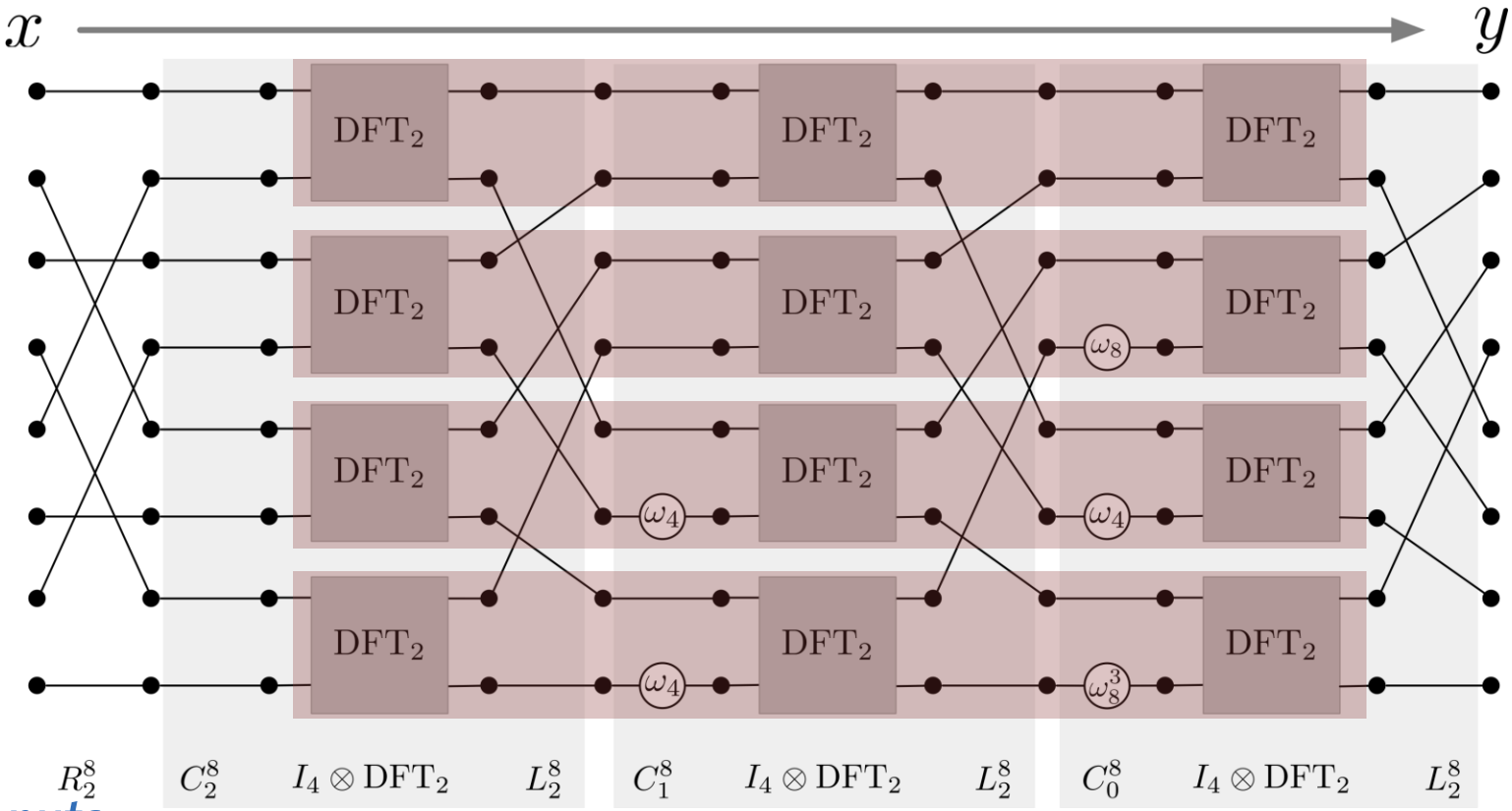


iterative: partial reuse

$$\prod_{\ell_0=0}^{m/k-1}{}^{\text{ir}} \left(\prod_{\ell_1=0}^{k-1} A_n \right)$$



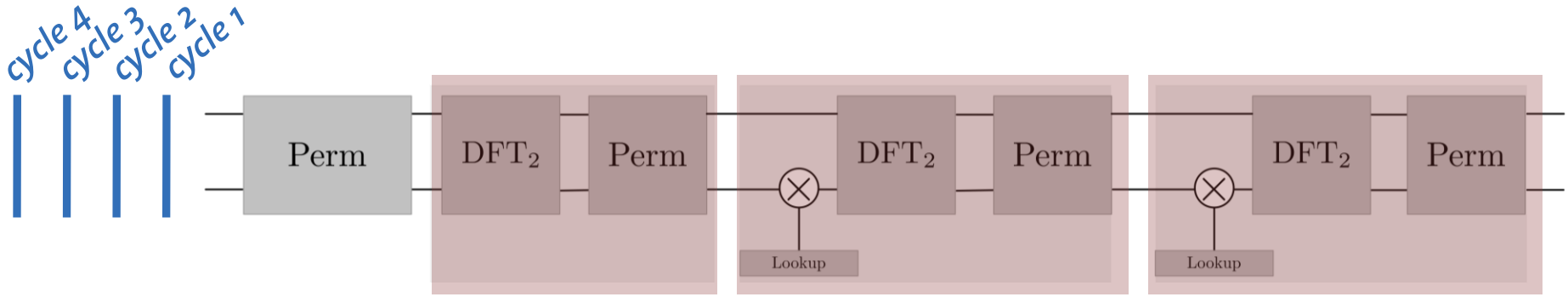
Previous Example



All 8 inputs in parallel

Parallelism allows for streaming reuse (sr)

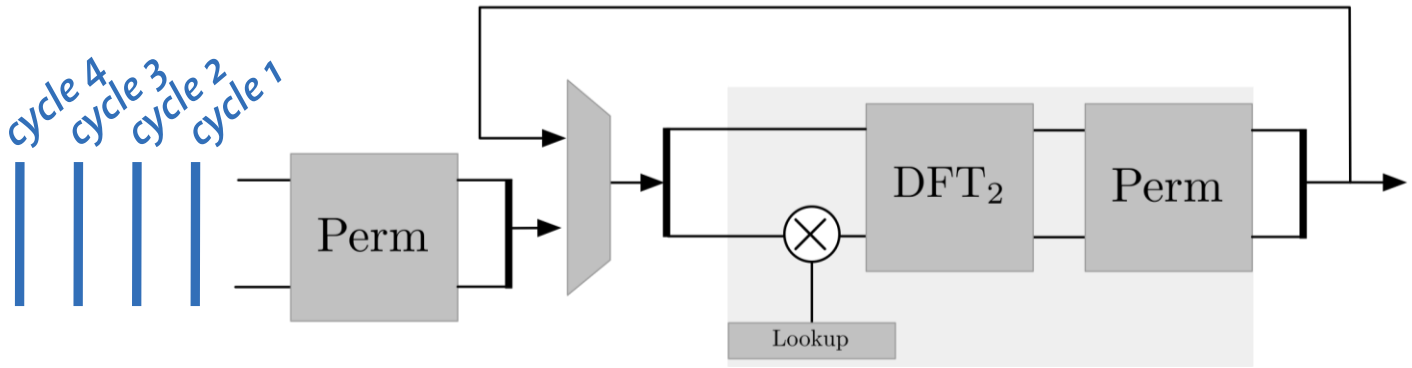
Vertical Folding By Rewriting



$$\underbrace{\mathrm{DFT}_8}_{\mathrm{stream}(2)} \longrightarrow$$

Repeated stages allow for iterative reuse (ir)

Vertical & Horizontal Folding By Rewriting



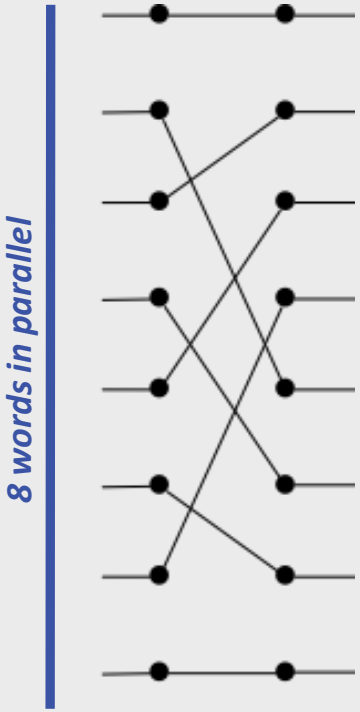
$$\underbrace{\mathrm{DFT}_8}_{\mathrm{stream}(2),\ \mathrm{depth}(1)} \longrightarrow$$

How to build streaming permutations?

$$P \xrightarrow{\ ?\ } \underbrace{P}_{\mathrm{stream}(2)}$$

Parallel permutation

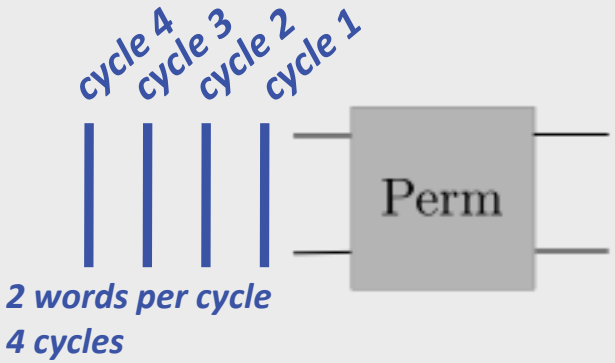
Example: 8 points



- Just wires

Streaming permutation

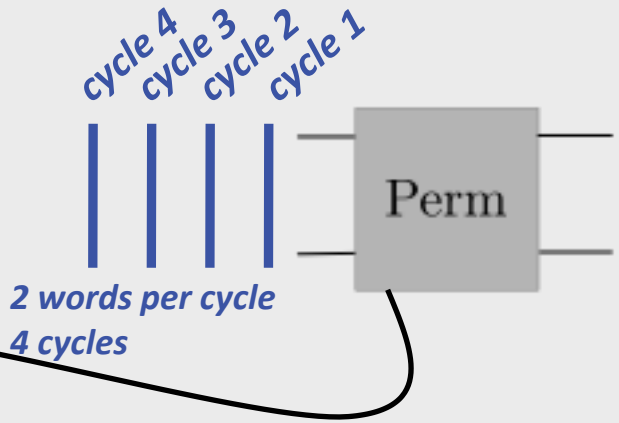
Example: 8 points, 2 points per cycle

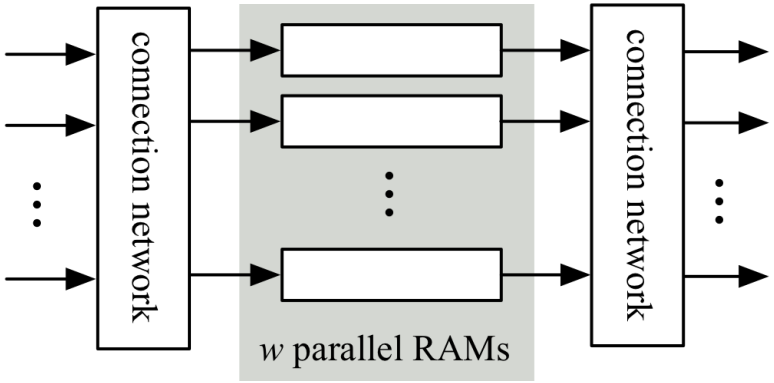


$$\underbrace{P}_{\text{stream}(2)}$$

Streaming permutation

Example: 8 points, 2 points per cycle

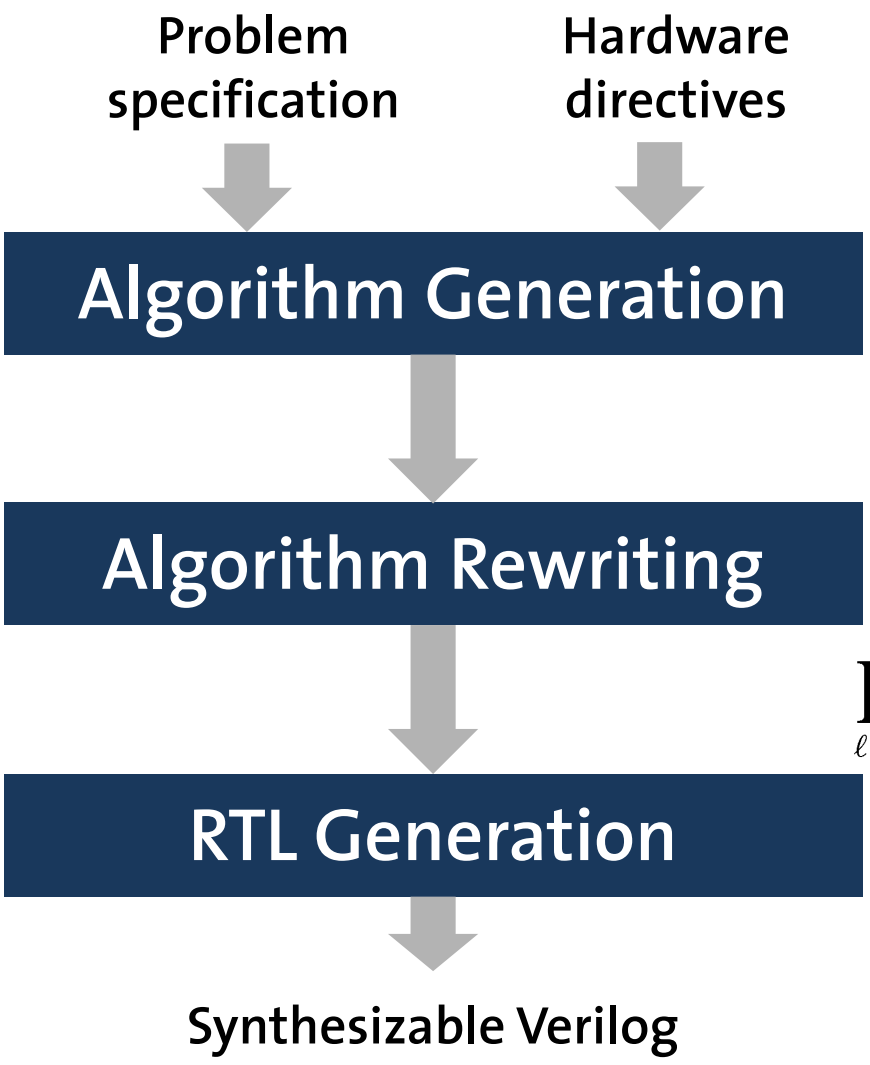




- Data stream must be buffered (using multiple banks of RAMs)
- Need to route data carefully to prevent port conflicts

[J. of the ACM 2009; DATE 2009]

IP Core Generator (Sketch) [TODAES 2012; DAC 2008]

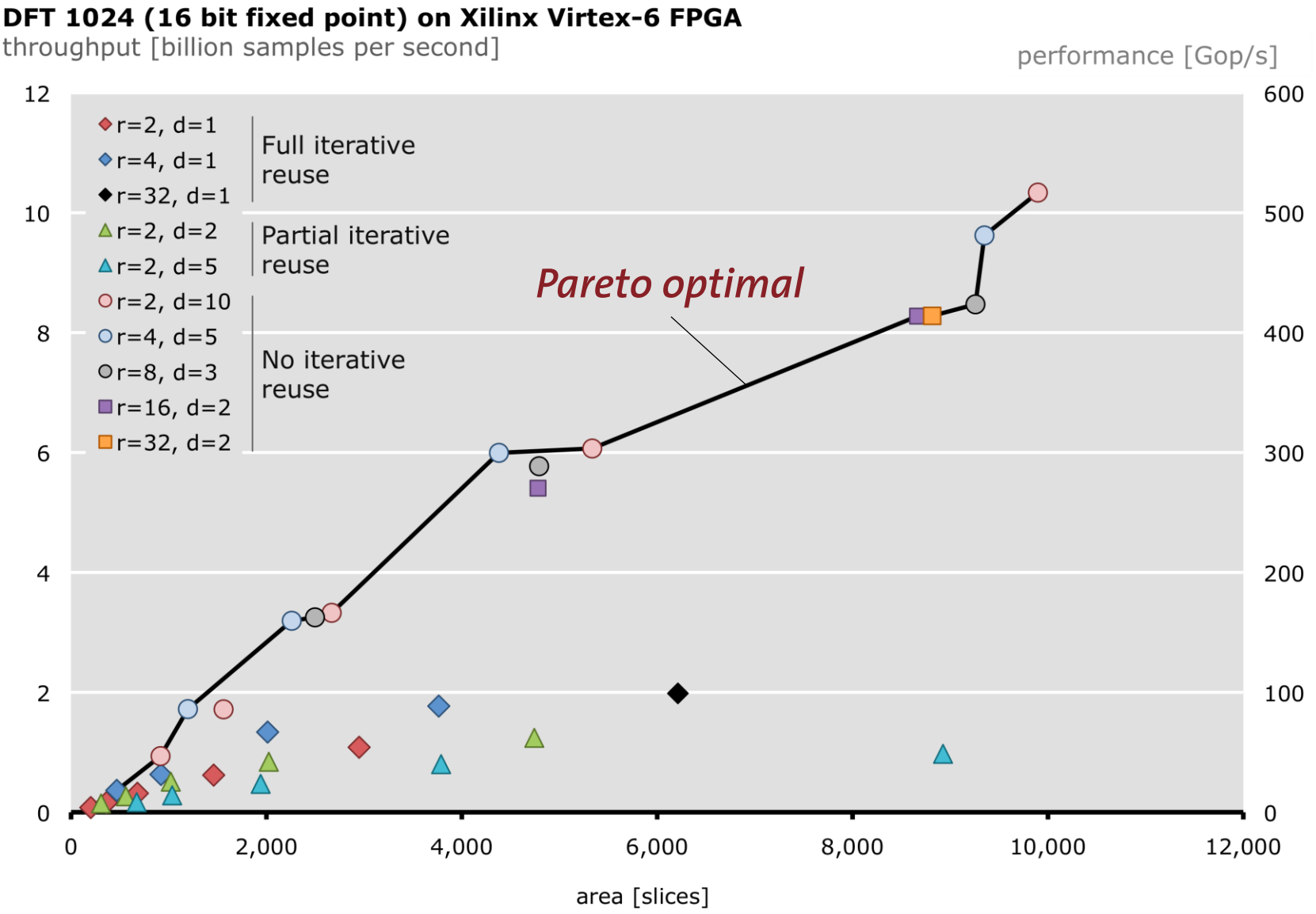


$$\underbrace{\mathrm{DFT}_{2^7}}_{\mathrm{stream}(4),\ \mathrm{depth}(1)}$$

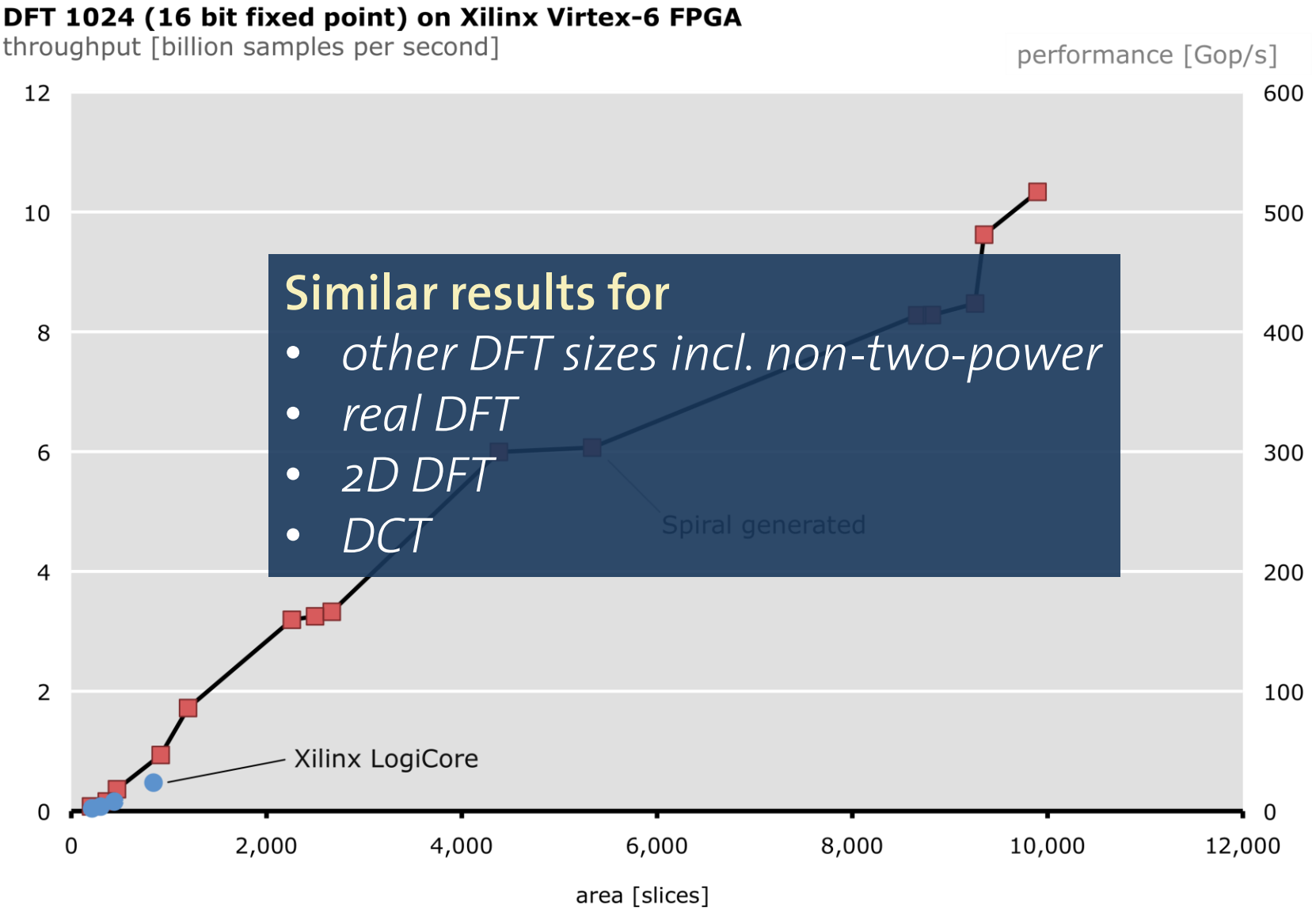
$$\underbrace{\left(\prod_{\ell=0}^{6} L_2^{128} \cdot (I_{64} \otimes \mathrm{DFT}_2) \cdot C_\ell^{128}\right) \cdot R_2^{128}}_{\mathrm{stream}(4),\ \mathrm{depth}(1)}$$

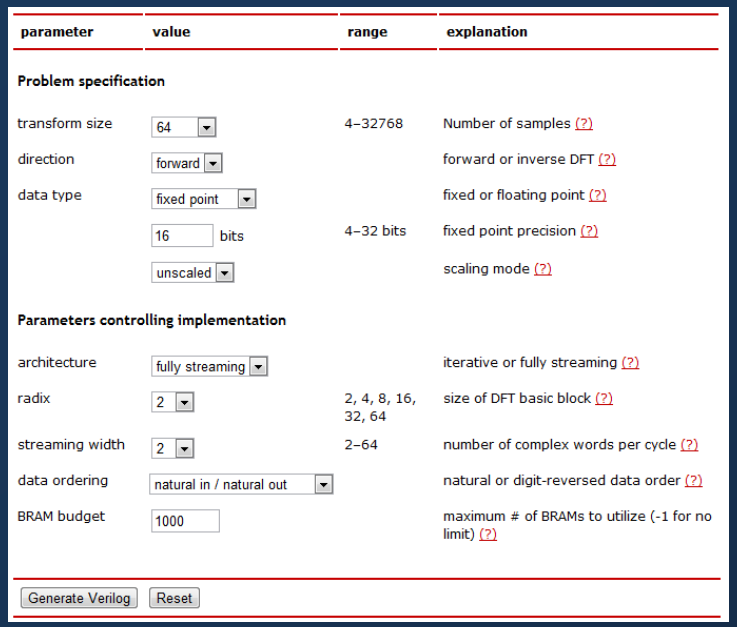
$$\prod_{\ell=0}^{6}{}^{\mathrm{ir}} \underbrace{L_2^{128}}_{\mathrm{stream}(4)} (I_{32} \otimes^{\mathrm{sr}} (I_2 \otimes \mathrm{DFT}_2)) \underbrace{C_\ell^{128}}_{\mathrm{stream}(4)} \underbrace{R_2^{128}}_{\mathrm{stream}(4)}$$

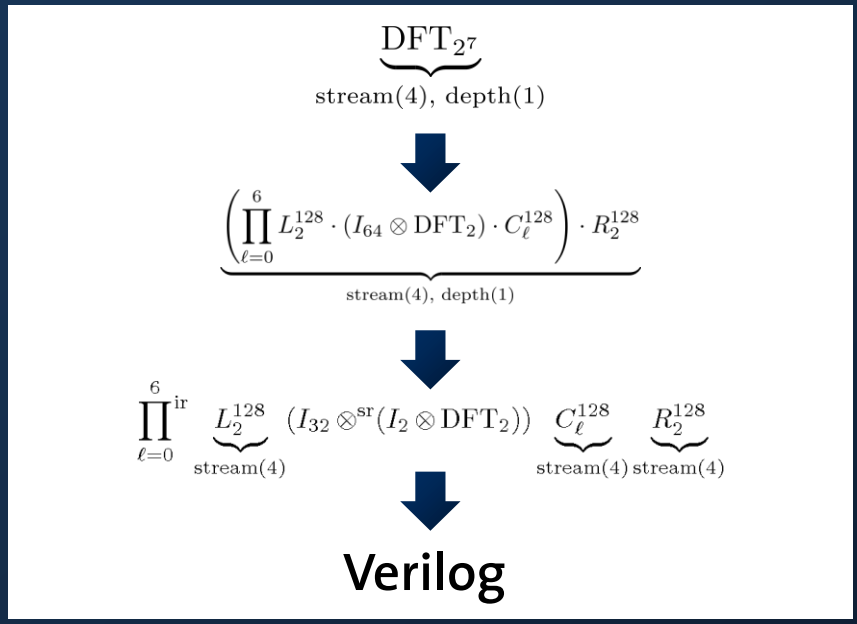
FPGA: Area vs. Throughput

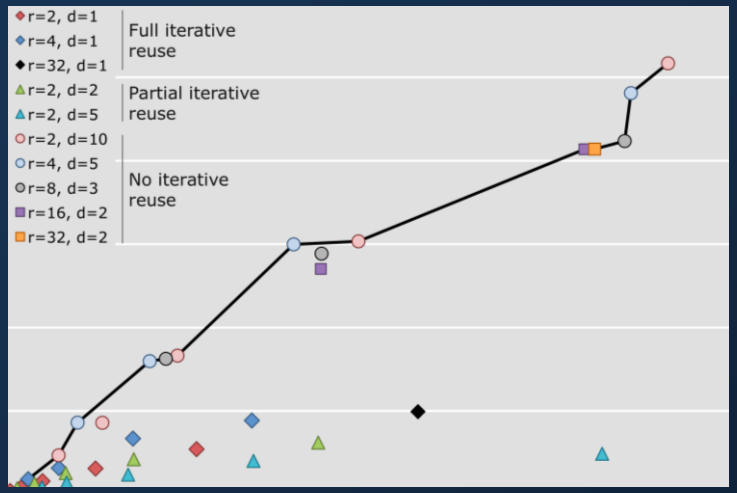


FPGA: Area vs. Throughput









Not shown:

- *ASIC results, power/perf tradeoff*
- *IP Generator for sorters*
- *Finding Pareto points without exhaustive design space enumeration*

www.spiral.net