FFT on XMT: Case Study of a Bandwidth-Intensive Regular Algorithm on a Highly-Parallel Many Core

James Edwards, Uzi Vishkin

University of Maryland (UMD)

Current state of parallel computing

- The parallel computing community has increasingly shifted its attention to communication avoidance (CA) as a way to address the end of Dennard scaling and the attendant difficulty in scaling down power consumption.
  - The National Academies “Game Over” report (2011)
  - Recent comp. arch. books published by Morgan and Claypool (2014)
  - Focus of many meetings (e.g., session on Wednesday)
- However, there are limits to the performance improvements that can be attained by focusing on reducing data movement.
  - Strength of current parallel architectures: regular algorithms requiring only limited communication
    - Ex.: dense-matrix multiplication
  - Limited speedups for other algorithms on current platforms
- Furthermore, the challenges of communication avoidance have arguably harmed programmers’ productivity (2014 CACM article).

A “what-if” question

- Despite these difficulties, the default mode for parallel programming research is reliance on off-the-shelf hardware.
- But what if alternative machines, or hardware features, are feasible, and can offer significant advantages?
- Clearly, such out-of-the-box hardware and the enabling technologies it may require are unlikely to ever be developed before their advantages are sufficiently understood.
- In contrast to work that seeks to avoid data movement, here we examine the problem from an alternate angle:
  - Assuming that is it possible to reduce the energy cost of data movement, is it possible to obtain strong speedups on problems for which such speedups have proven elusive?
- *Power consumption* of XMT ICN is ≤ 18% of total for ≥16k TCUs and decreasing

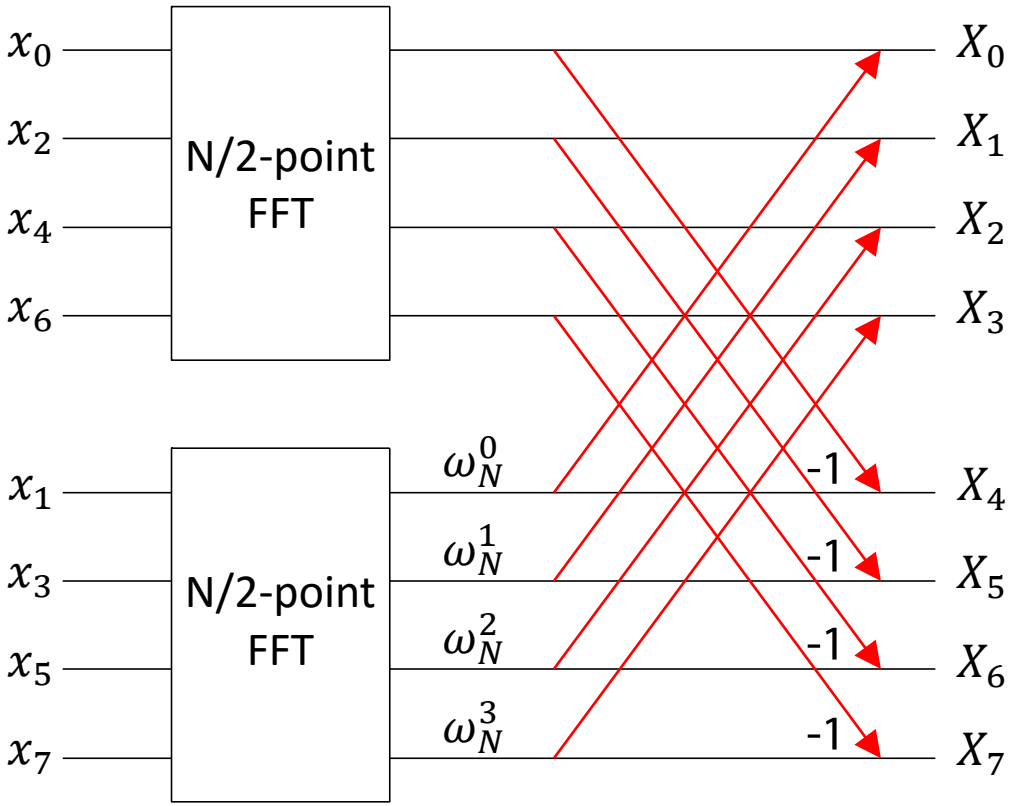
Case study in permissive communication: XMT @ UMD

- The preceding question has been partially answered in the affirmative by prior work on the Explicit Multi-Threading (XMT) general-purpose architecture.
- Goals of XMT:
  - Faster single-task completion time
  - Improved ease-of-programming
  - Efficient support for Parallel Random Access Model (PRAM) programming.
- Published speedups of 8-129× vs. GPU/CPU on basic and advanced irregular problems such as graph algorithms & data compression (backup slide)
- This paper: regular but communication intensive algorithms

Fast Fourier transform (FFT)

- The FFT is an important numerical algorithm used in fields such as signal processing and scientific computing.
- What sets the FFT apart from other regular numerical algorithms is its high communication needs;
  - $O(S)$ local memory → $O\left(\frac{n \log n}{\log S}\right)$ I/O operations
  - Suggests that caches are of limited use in reducing the bandwidth required by the FFT.
- Indeed, prior work using existing platforms obtained modest speedups relative to the hardware invested
  - Example a few slides later

Communication in the FFT



Arrows = communication

Fourier transform

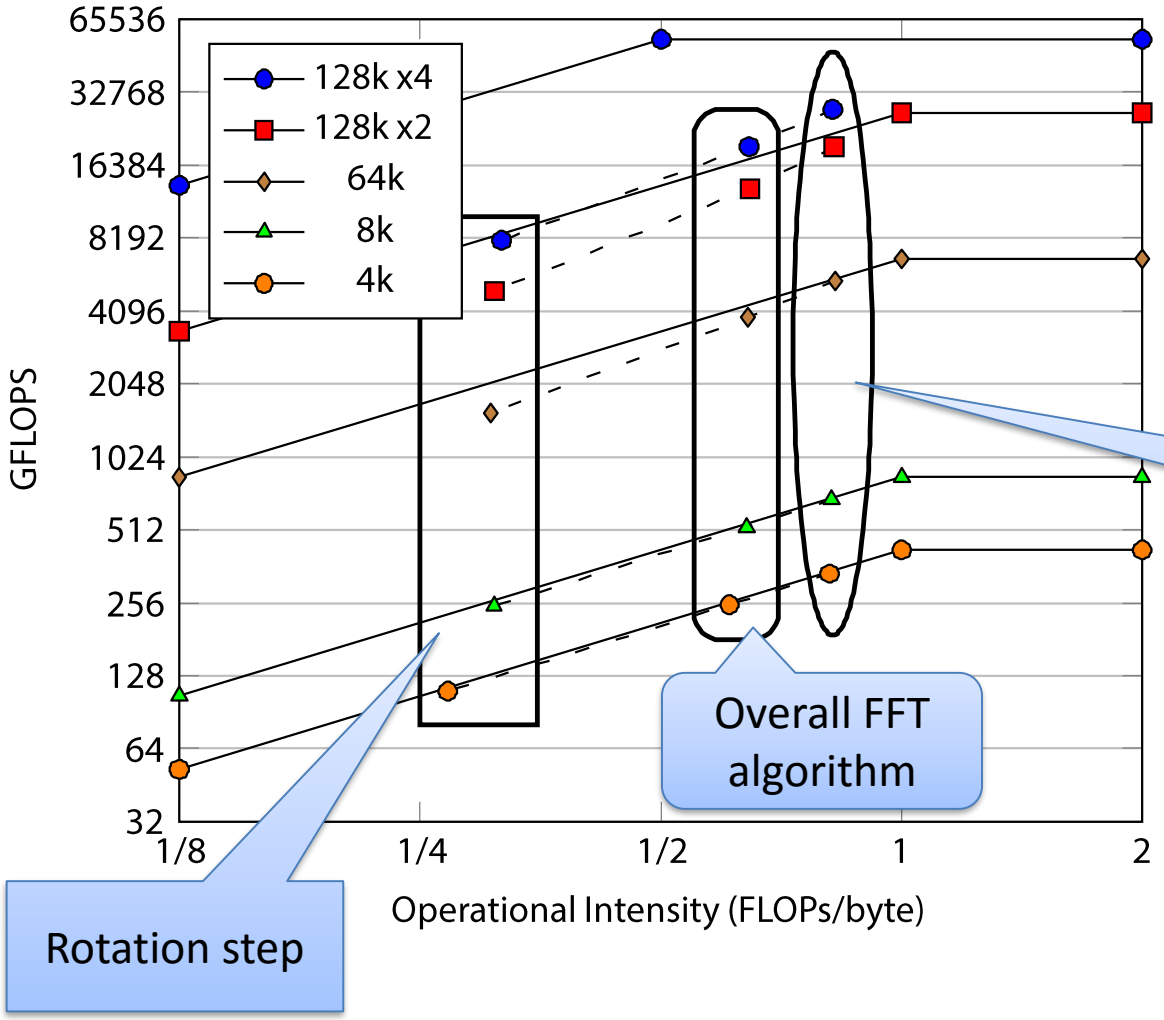
$$X_k = \sum_{n=0}^{N-1} \left(x_n \cdot \omega_N{}^{-kn}\right)$$

$$= X_k^{even} + \omega_N^{-k}$$

where $\omega_N = e^{i2\pi/N}$

Not shown: communication implied by rotation (transpose) step for 3D FFT

FFT: Roofline model



Lines

Solid line = Roofline model

Dashed line = Empirical results

3D Fast Fourier Transform (FFT) on XMT

[Edwards, Vishkin, in press]

- With enabling technologies, XMT can outperform a much larger supercomputer (Edison) on 3D FFT
- XMT speedups vs. best serial (FFTW):
  - Without enabling technologies (8k TCUs): 31X (0.24 TFLOPS)
  - With enabling technologies (128k TCUs): 2,494X (19.0 TFLOPS)

Comparison of Edison machine (Cray XC30) to XMT

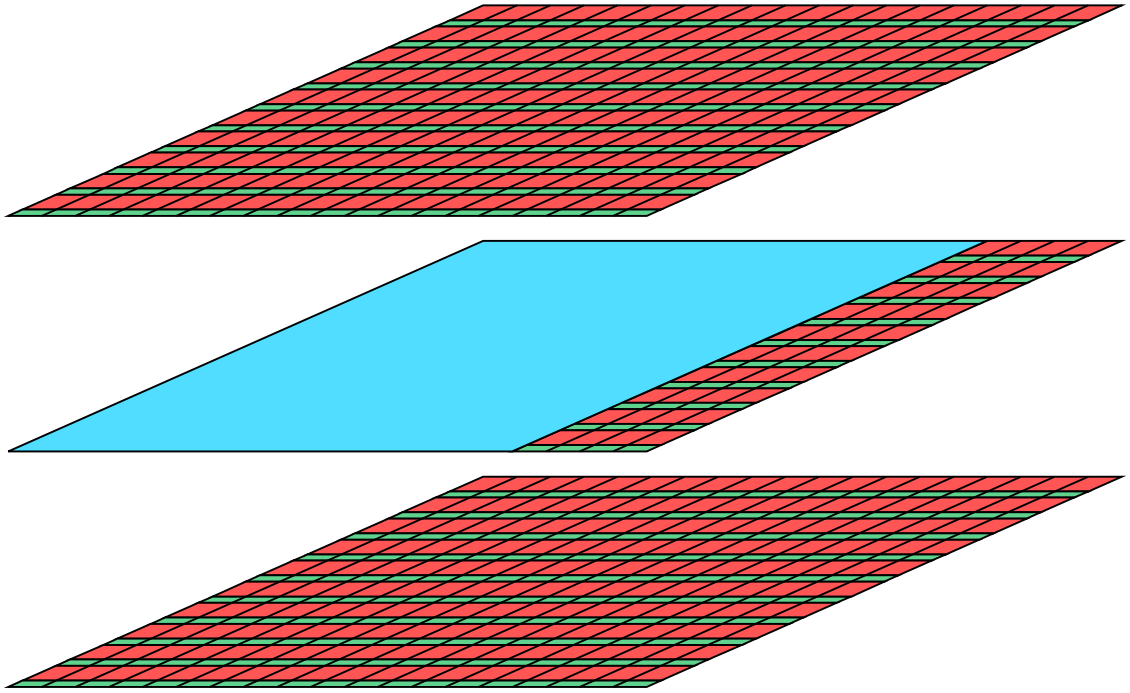
	XMT (128k x4)	Edison (Cray XC30)	Factor
# processing elements	131,072 TCUs	124,608 cores	
# processor groups	4,096 clusters	5,192 nodes	
Total cache memory	128 MB	311,520 MB	2433
# chips	1	10,384 CPU + 1,298 router	11682
Total silicon area (process)	35.4 cm^2 (14 nm)	56,177 cm^2 (22 nm) + 4,072 cm^2 (40 nm)	
Normalized silicon area (22 nm)	66 cm^2	57,409 cm^2	871
Peak power consumption	7.0 KW	2,500 KW	357
Peak teraFLOPS	54	2,390	44
TeraFLOPS for FFT (size)	19.0 (512^3)	13.6 (1024^3)	/1.4
% of peak FLOPS	35%	0.57%	/61

Note: TCU = Thread Control Unit (lightweight processor)

Enabling technologies

- For increased on-chip bandwidth to shared cache:
  - 3D VLSI
  - Microfluidic cooling
- The above technologies enable XMT to scale up to 8x larger than would be possible without them
  - Temperature and power results reported as part of a new software spiral proposal [Intel CATC 2015]
- Extension to off-chip bandwidth to DRAM:
  - Silicon photonics
- While it is expected that increased bandwidth enabled by such technologies would lead to improved performance the (high) rate of improvement shows great promise.

Starting with what we figured out. Schematic floorplan for active silicon layers

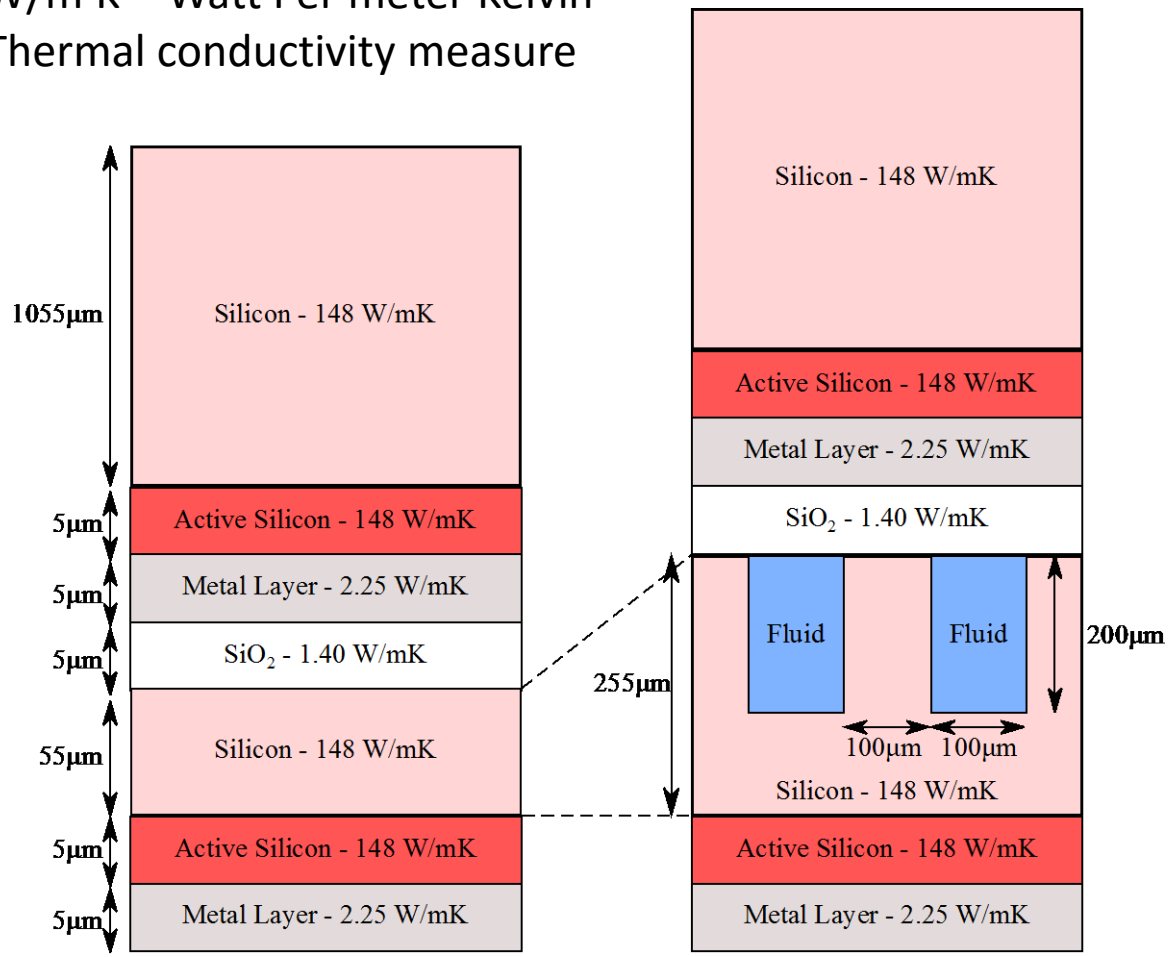


Red blocks - TCU clusters. Green blocks - memory modules. Blue block - the ICN.

Main points of this slide: ICN is confined to one layer. It is a middle layer. Can share with other functions. Memory ports in external layer.

Cross-section view of uncooled vs cooled 3D IC stack

W/m·K – Watt Per meter Kelvin
Thermal conductivity measure



*Figures are not to scale

Bigger picture
20 mm wide chip ➔
100 fluid microchannels/layer
100 µm wide, 100 µm apart

For our largest configuration
Pump supplying 200 kPa head at 1 L/min. Commercial product*: under 16cm long

*Also: Datasheet for pump meeting specs Drawn ~1.5W. Fits in 3x3x3 inch^3 cube. Additional: mounting piece.

Silicon (Si) photonics

- Promise (since we do not have results): can scale XMT to 128k TCUs with high off-chip bandwidth
- Potential for even greater scaling with multi-chip design
- But, need to sort out many approaches & tradeoffs to silicon photonics
  - How “monolithic” can integration be?
  - To what extent can electronics and photonics
    1. share the same wafer improving density and lowering parasitic effects, versus
    2. be fabricated on different platforms that favor each better; then brought together, e.g., by short wire bonds and flip chip
  - How can temperature fluctuation be managed?
  - How to scale reliably to 100K high-bandwidth devices on chip using automatic fabrication?
- Various approaches offer different tradeoffs to the above questions.
- Will microfluidic cooling (and 3D-VLSI) provide the key for the scale we seek? Not only above issues. Also: greater flexibility with the remaining pieces since intermediate steps may be power inefficient

Conclusion

- XMT shows that strong speedups are possible for problems that have proven difficult for current platforms.
- Enabling technologies allow the advantage of XMT to be extended to larger scales and more types of problems.
- The results presented here help to resolve the chicken-and-egg problem posed by enabling technologies:
  - development of enabling technologies will not advance without evidence of their benefit, while such evidence apparently cannot be obtained until these technologies have already been developed.
- Benefit is not limited to XMT
  - Main difference of XMT from other platforms is ICN
  - Enabling technologies can improve switches used by existing platforms
- Final point for consideration: Has the (exclusive) focus on communication avoidance been a strategic mistake?
  - Focus on CA → vendors will produce HW that requires CA
  - In the serial era, technology obstacles to the programming model were put on a roadmap (e.g., ITRS) and addressed. CA took the steam away from this successful model.

References

- U. Vishkin. Using simple abstraction to reinvent computing for parallelism. Communications of the ACM 54,1 (2011), 75-85. [Over 15K downloads]
- S. O'Brien, U. Vishkin, J. Edwards, E. Waks and B. Yang. Can Cooling Technology Save Many-Core Parallel Programming from Its Programming Woes? In Proc. Compiler, Architecture and Tools Conference (CATC), Intel Development Center, Haifa, Israel, Nov 2015. http://drum.lib.umd.edu/handle/1903/17153
- U. Vishkin. Is Multi-Core Hardware for General-Purpose Parallel Processing Broken? Viewpoint article. Communications of the ACM 57,4 (2014), 35-39.