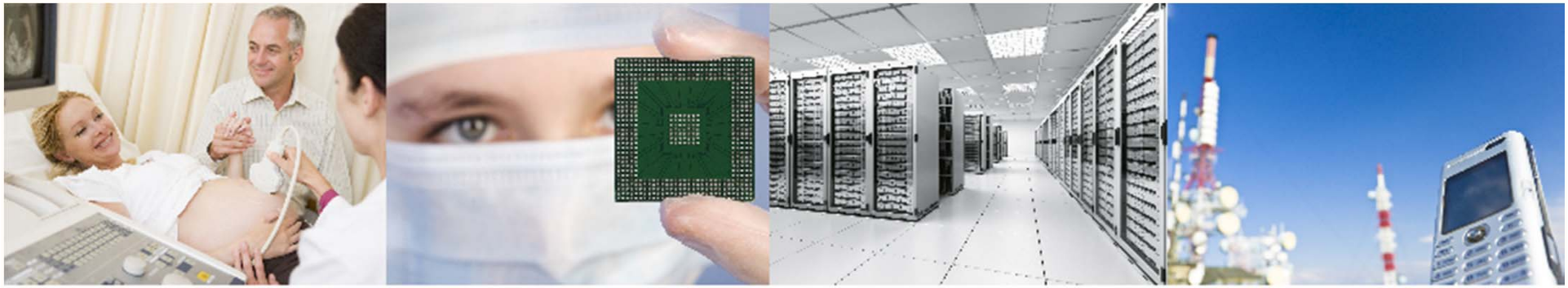
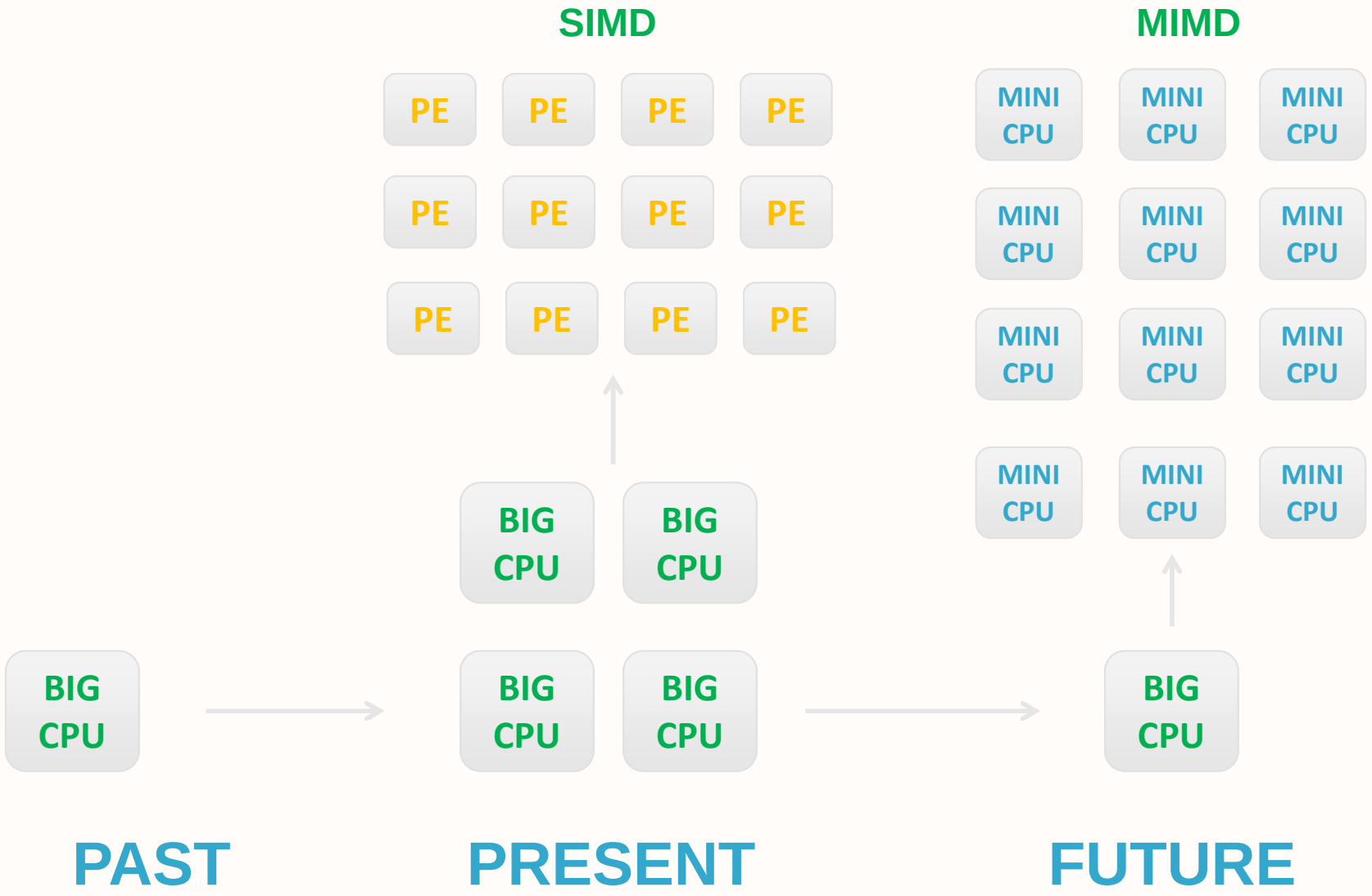


A 1024-core 70GFLOPS/W Floating Point Manycore Microprocessor

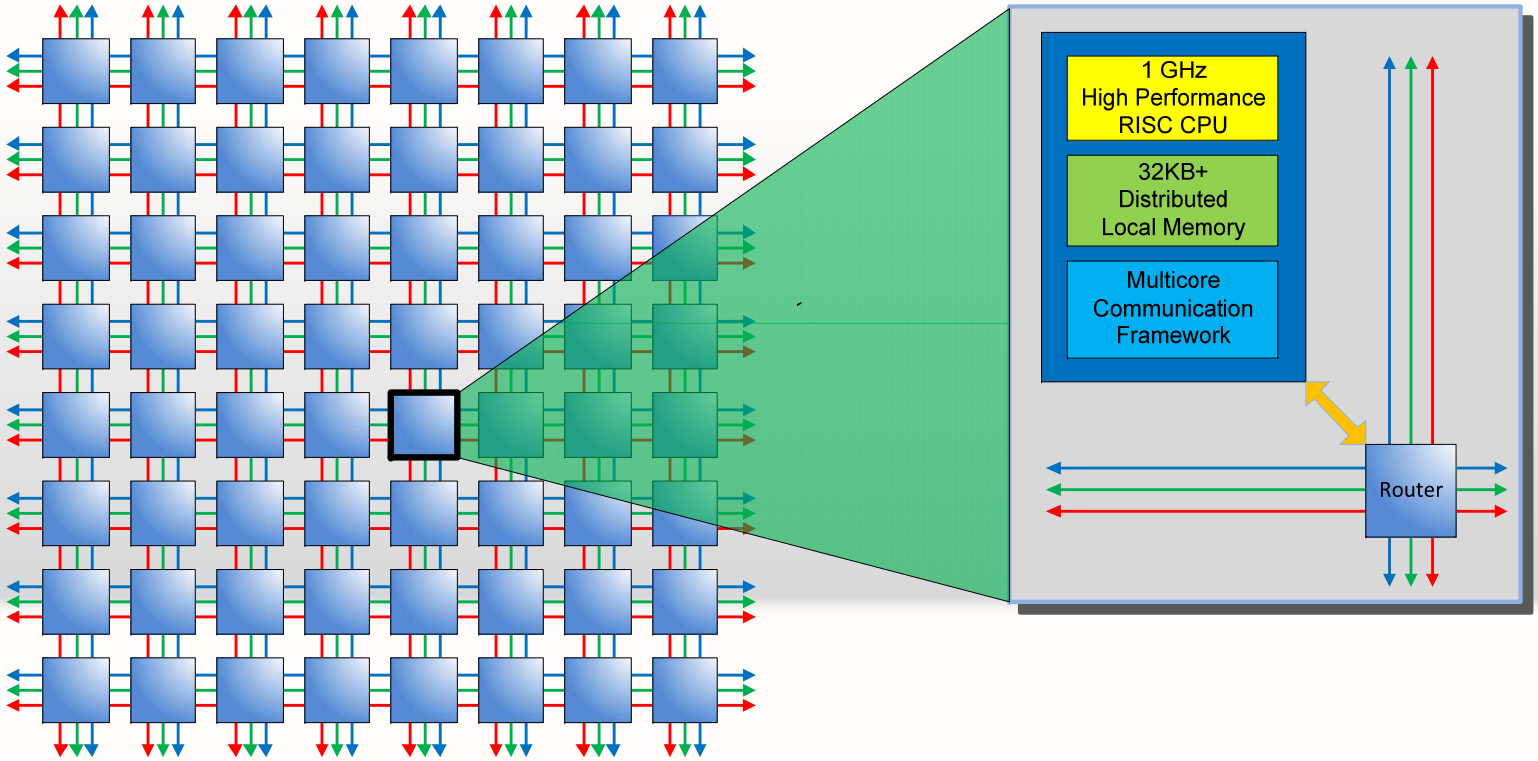
Andreas Olofsson, Roman Trogan, Oleg Raikhman
Adapteva, Lexington, MA

adapteva

The Past, Present, & Future of Computing



Adapteva's Manycore Architecture



C/C++ Programmable

Incredibly Scalable

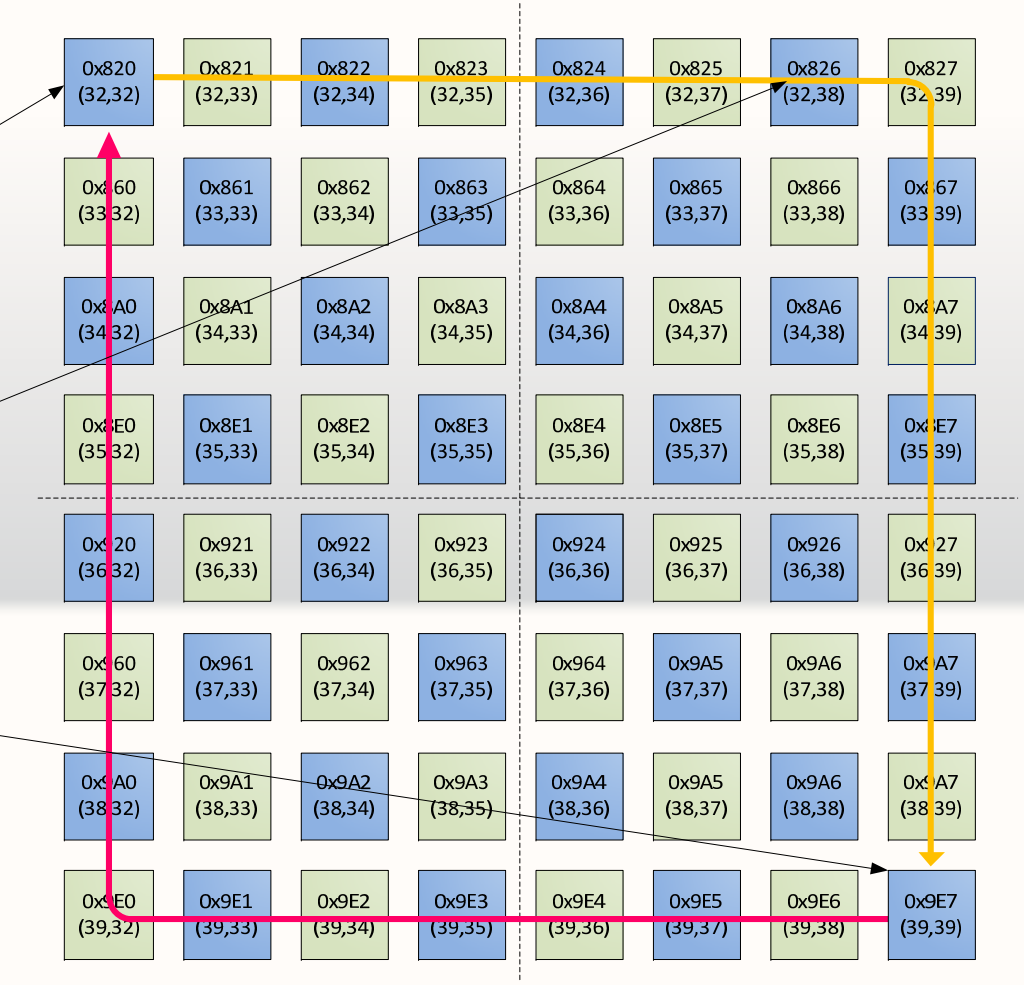
70 GFLOPS/W

Routing Architecture

1. Core 32,32 sends a read request to core 39,39 and specifies its own core id(39,39) as a return (source) address

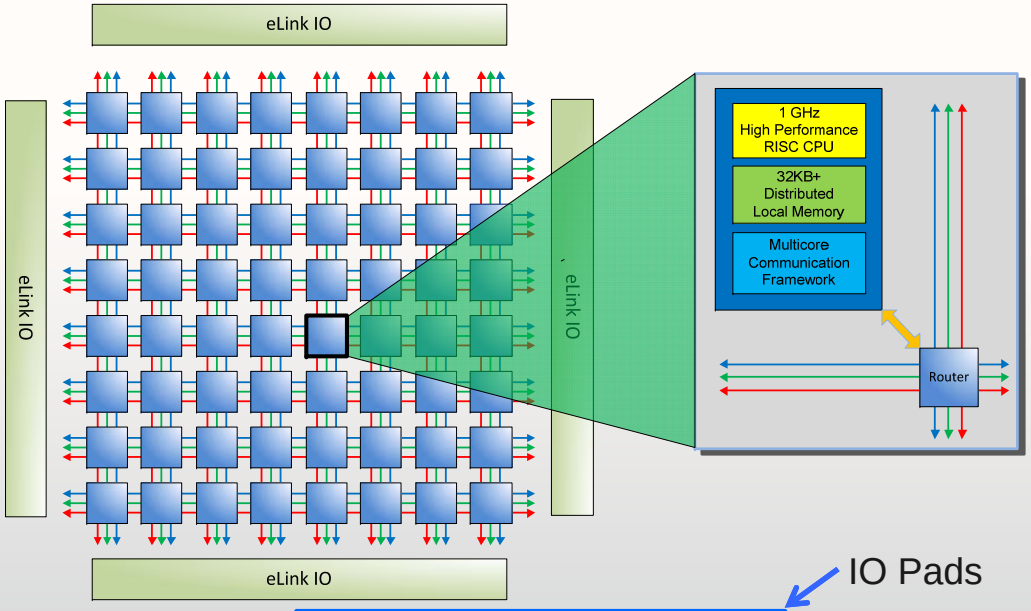
2.Along the path, routing nodes compare the transaction address to the routers hardcoded ID and decide if the transaction should be sent north, east, south, or west.

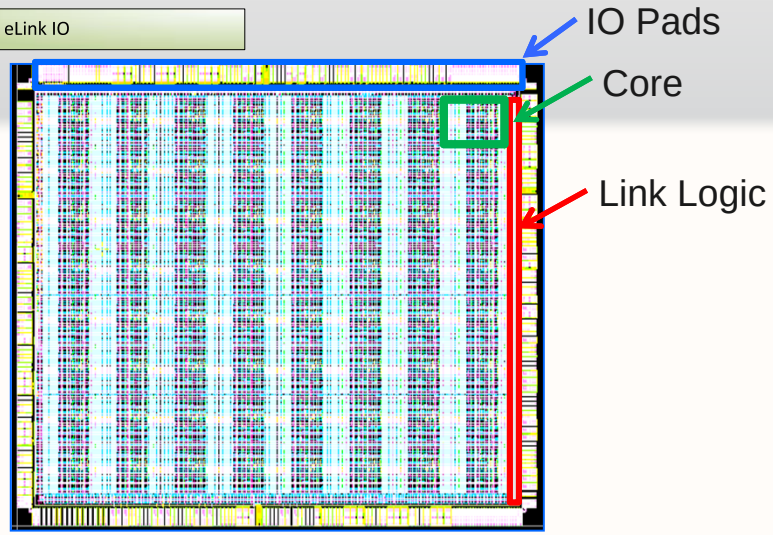
3. Core 39,39 receives read transaction, fetches data from memory and returns the data to core 32,32 (source) using the write network.



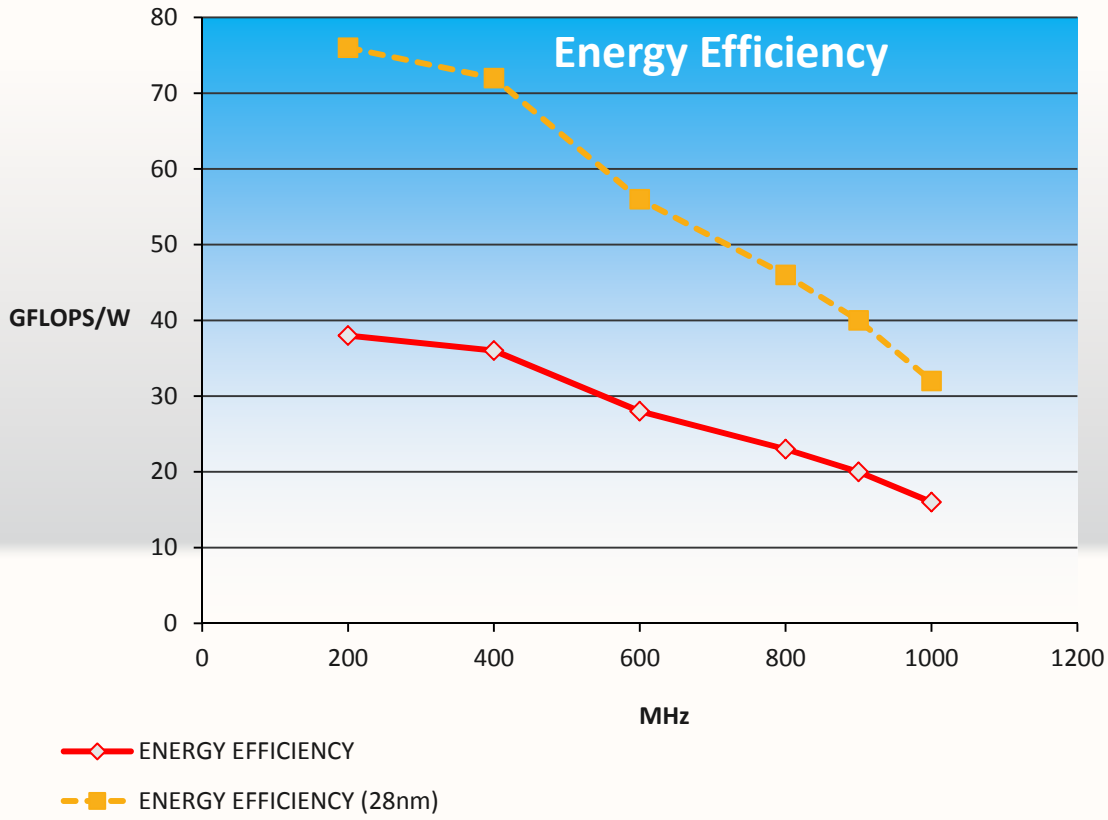
E64G400 Specifications (Jan-2012)

- 64-Core Microprocessor
- 100 GFLOPS performance
- 800 MHz Operation
- 8GB/sec IO bandwidth
- 1.6 TB/sec on chip memory BW
- 0.8 TB/sec network on chip BW
- 64 Billion Messages/sec
- 2 Watt total chip power
- 2MB on chip memory
- 10 mm^2 total silicon area
- 324 ball 15x15mm plastic BGA

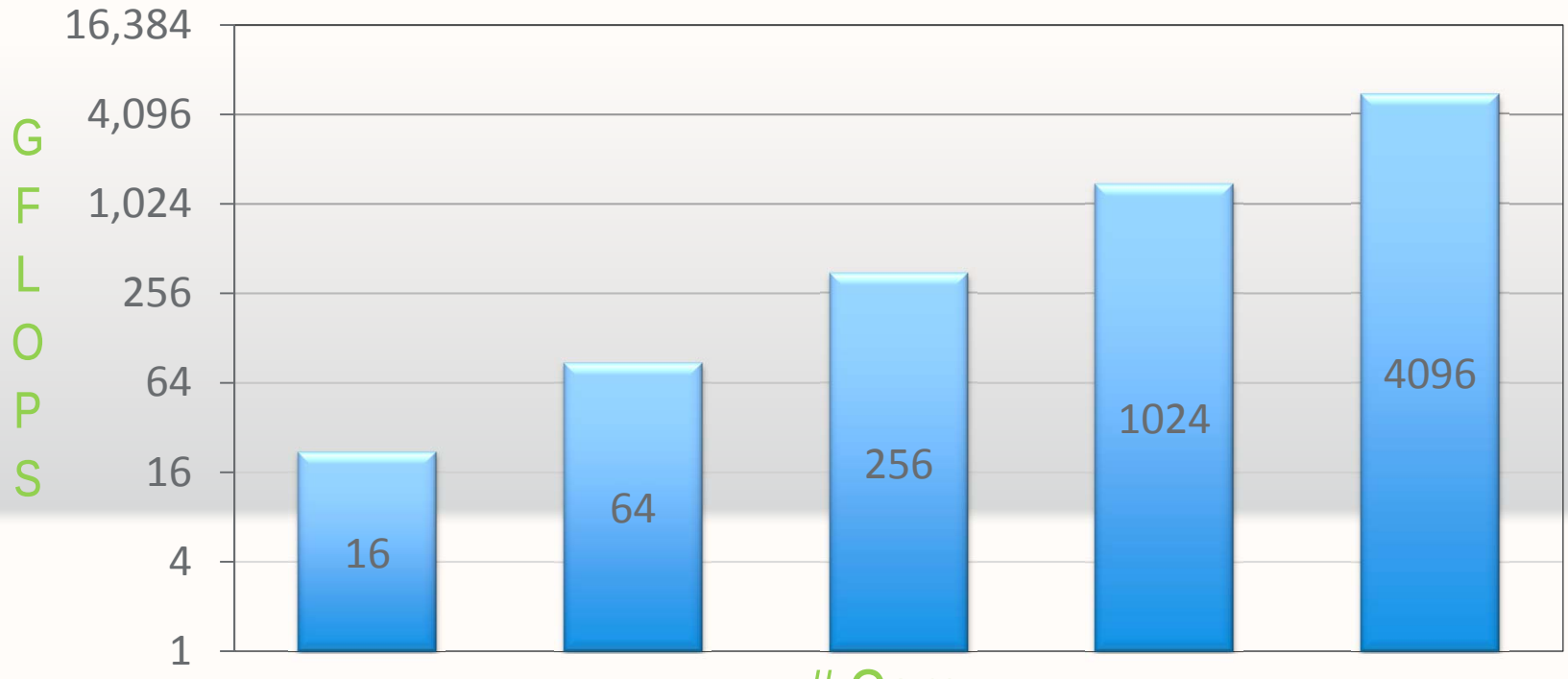




Lab Measurements



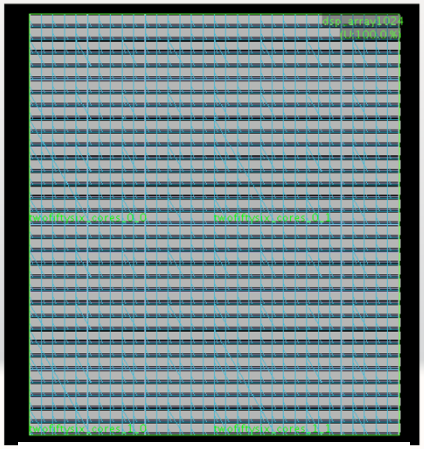
Epiphany Performance Scaling



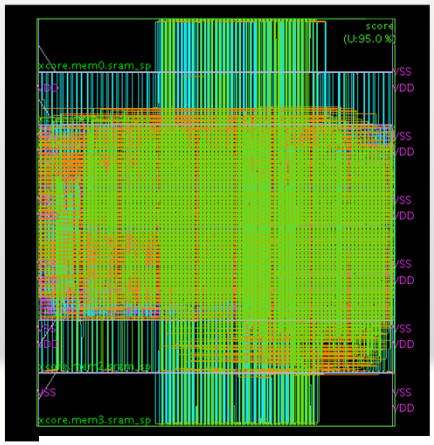
On-demand scaling from 0.25W to 64 Watt

Hold on...the title said 1024 cores!

- We can build it any time!
- Waiting for customer
- LEGO approach to design
- No global timing paths
- Guaranteed by design
- Generate any array in 1 day
- ~130 mm^2 silicon area



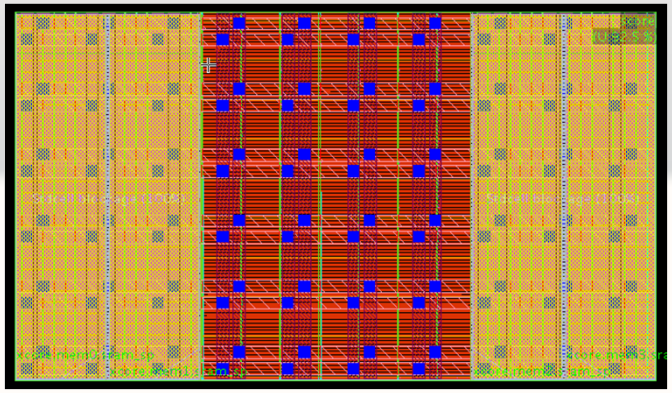
1024 Cores



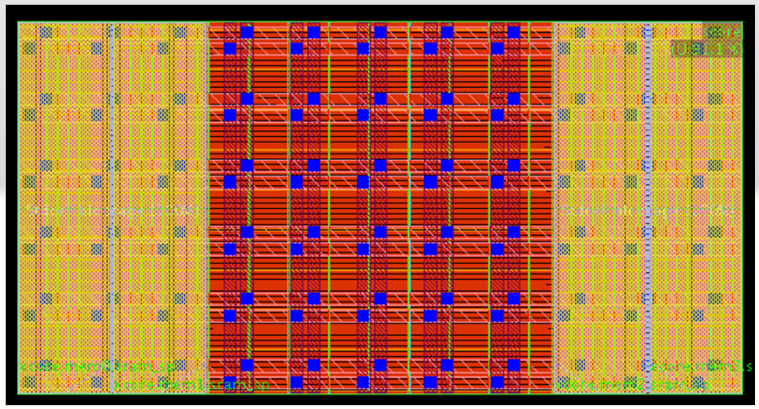
1 Core

What about 64-bit Floating Point?

Single Precision
2 FLOPS/CYCLE
64KB SRAM
0.215mm^2
700MHz



Double Precision
2 FLOPS/CYCLE
64KB SRAM
0.237mm^2
600MHz



Epiphany Latency Specifications

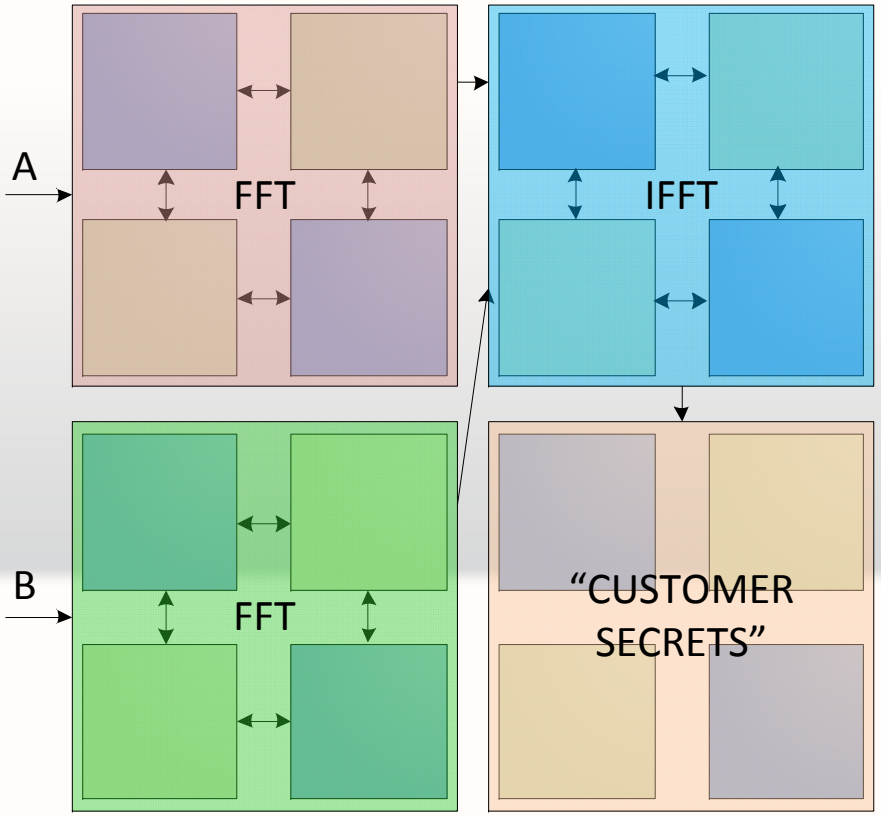
	Spec
Nearest Neighbor Core to Core Write	<4ns
Nearest Neighbor Core to Core Read	<15ns
Farthest On-chip Core to Core Write	<15ns
Farthest On-chip Core to Core Read	<30ns
Chip to Chip Write	<60ns
Smallest Message that Achieve Peak Bandwidth	8 Bytes
Maximum Messages Per Second	**64 Billion**

What about the programming model?

- Whatever fits within a XX-KB memory constraint
- Fork/join task scheduling demonstrated
- Message passing (like MPI) demonstrated
- Master/slave openCL framework in progress
- Others?

Generic Processing Example

- Algorithm:
  - Fast convolution
    - FFT done on input streams
    - Point-wise multiply, then inverse FFT
  - Multicore FFT routines provided by Adapteva
  - The magic happens at the backend
- Key enablers:
  - Embarrassment of riches (like FPGA)
  - Ease of use (like microprocessor)
  - High bandwidth, flexibility



Technology Status (Shipping)

BittWare Design Wins

Shipping Evaluation Systems Worldwide

The Big Dream For 2012

Process Technology	28nm
#CPUs/Chip	1024
#Chips/Node	16
#Nodes/System	48
#CPUs/Rack	786,432
Performance:	~1PFLOP
Max Power:	~30KW

Conclusions

- It's possible to make CPU's energy efficient.
- 70 GFLOPS/Watt is possible in 2012
- There is an alternative to SIMD GPGPUs.
- It's possible to get 60-80% peak efficiency from C-code.
- Parallel programming is still too hard!
- It's possible to put 1 Million CPU Cores in a single rack