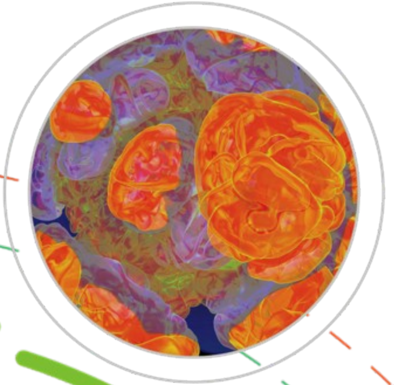
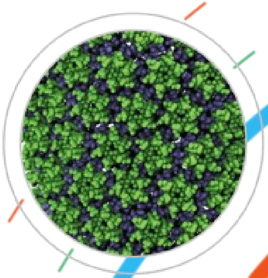
FPGAs for Supercomputing: Progress and Challenges

Hal Finkel[2] (hfinkel@anl.gov), Zheming Jin[2], Kazutomo Yoshii[1], and Franck Cappello[1]

[1]Mathematics and Computer Science (MCS)
[2]Leadership Computing Facility (ALCF)
Argonne National Laboratory





H2RC: Third International Workshop on
Heterogeneous Computing with Reconfigurable Logic
Friday, November 18, 2017
Denver, CO

Argonne **Leadership Computing** Facility

Argonne NATIONAL LABORATORY

Outline

- Why are FPGAs interesting? Where in HPC systems do they work best?
- Can FPGAs competitively accelerate traditional HPC workloads?
- Challenges and potential solutions to FPGA programming.

For some things, FPGAs are **really** good!

bioinformatics

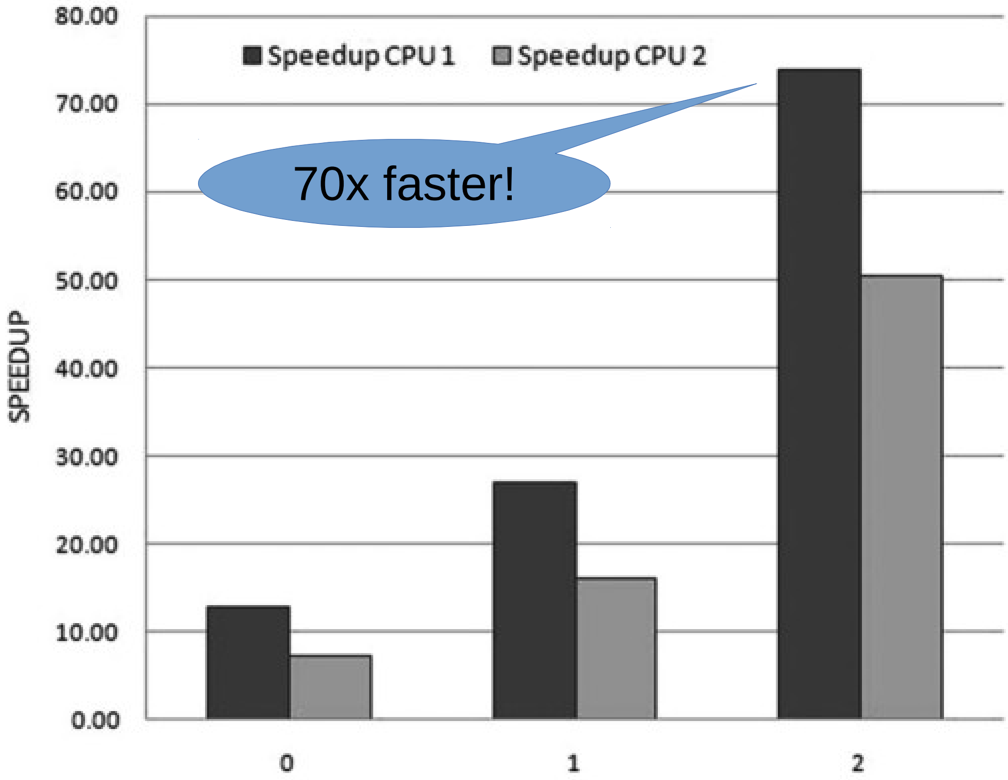


Fig. 9. Speed up of FHAST compared to BOWTIE for exact matches, one and two mismatches.

http://escholarship.org/uc/item/35x310n6

For some things, FPGAs are **really** good!

machine learning and neural networks

FPGA is faster than both the CPU and GPU, 10x more power efficient, and a much higher percentage of peak!

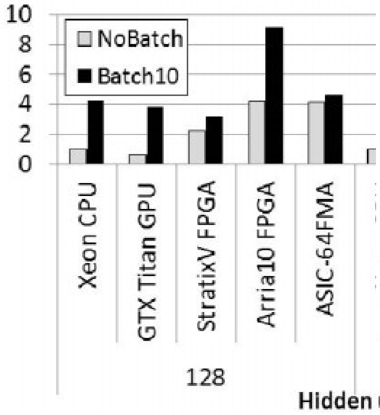


Fig. 5. Performance for all the accelerators under study, relative to CPU performance with no batching.

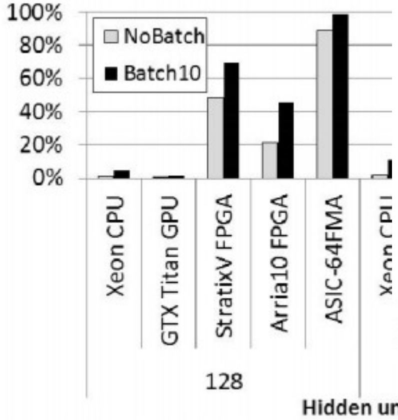


Fig. 6. Achieved performance relative to peak performance. E.g., 10% means the system is underutilized, where the achieved GFLOP/s is only at 10% of the available peak GFLOP/s. On the other hand, 100% means full utilization.

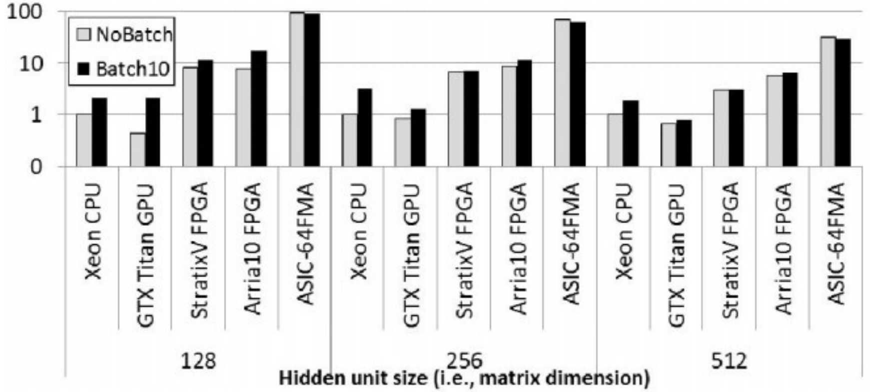
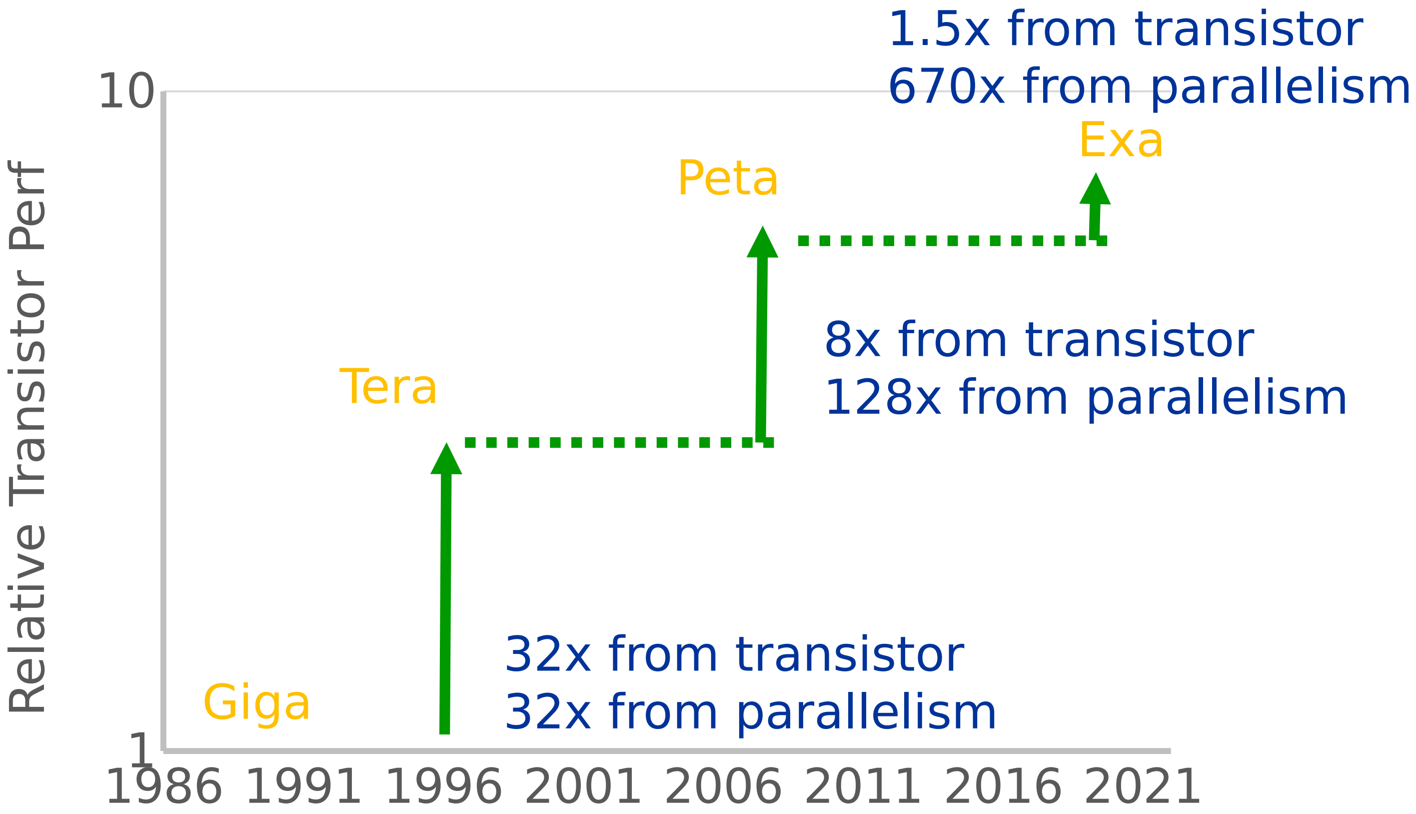


Fig. 7. Performance/Watt for all the accelerators under study, relative to the CPU performance with no batching. The y-axis is in log-scale.

http://ieeexplore.ieee.org/abstract/document/7577314/

Parallelism Triumphs As We Head Toward Exascale



System performance from parallelism

http://www.socforhpc.org/wp-content/uploads/2015/06/SBorkar-SoC-WS-DAC-June-7-2015-v1.pptx

(Maybe) It's All About the Power...

Operation	Energy (pJ)
64-bit integer operation	1
64-bit floating-point operation	20
256 bit on-die SRAM access	50
256 bit bus transfer (short)	26
256 bit bus transfer (1/2 die)	256
Off-die link (efficient)	500
256 bit bus transfer (across die)	1,000
DRAM read/write (512 bits)	16,000
HDD read/write	$O(10^6)$

Courtesy Greg Asfalk (HPE) and Bill Dally (NVIDIA)

Do FPGA's perform less data movement per computation?

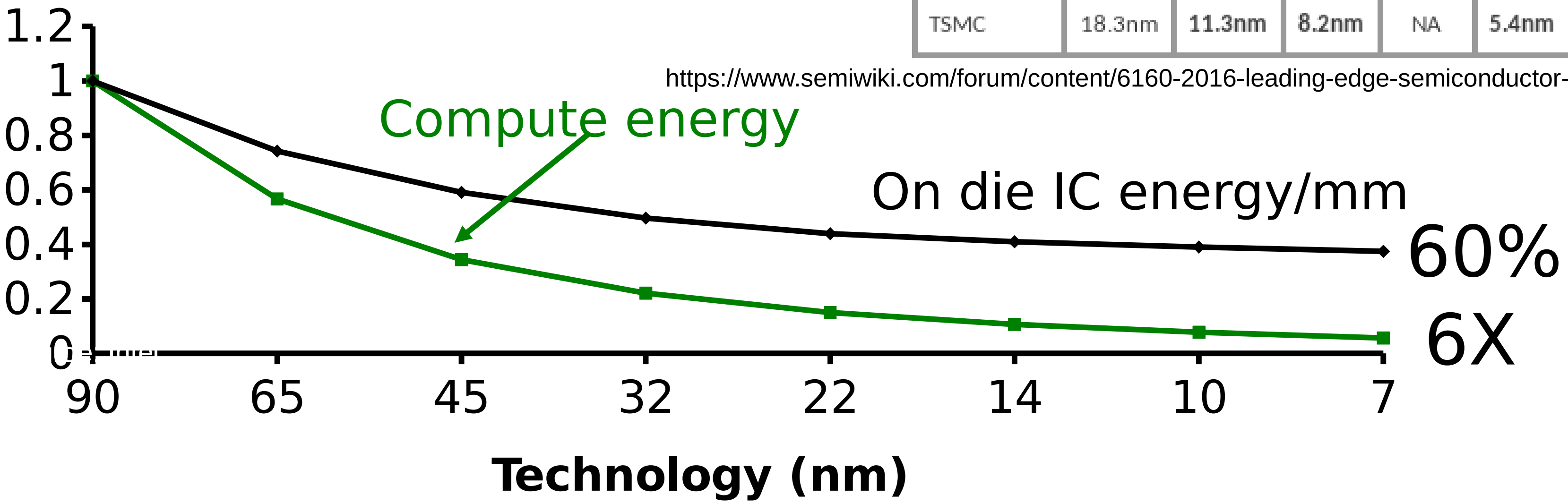
http://science.energy.gov/~/media/ascr/ascac/pdf/meetings/201604/McCormick-ASCAC.pdf

To Decrease Energy, Move Data Less!

On-die Data Movement vs Compute

Company	Current	2016	2017	2018	2019	2020
Global Foundries	16.6nm	NA	NA	8.2nm	NA	NA
Intel	**13.4nm**	NA	9.5nm	NA	NA	6.7nm
Samsung	16.6nm	12.0nm	NA	8.4nm	NA	NA
TSMC	18.3nm	**11.3nm**	**8.2nm**	NA	**5.4nm**	NA

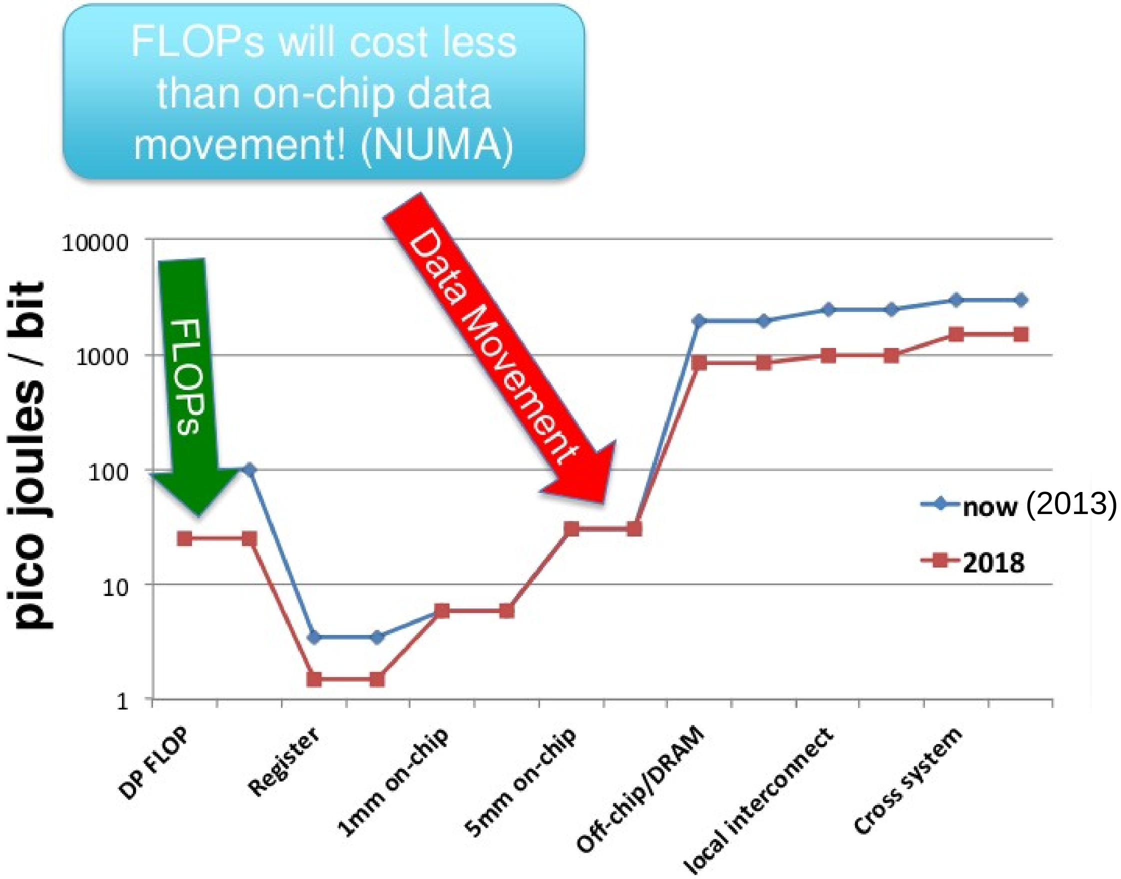
https://www.semiwiki.com/forum/content/6160-2016-leading-edge-semiconductor-landscape.html



Interconnect energy (per mm) reduces slower than compute
On-die data movement energy will start to dominate

http://www.socforhpc.org/wp-content/uploads/2015/06/SBorkar-SoC-WS-DAC-June-7-2015-v1.pptx

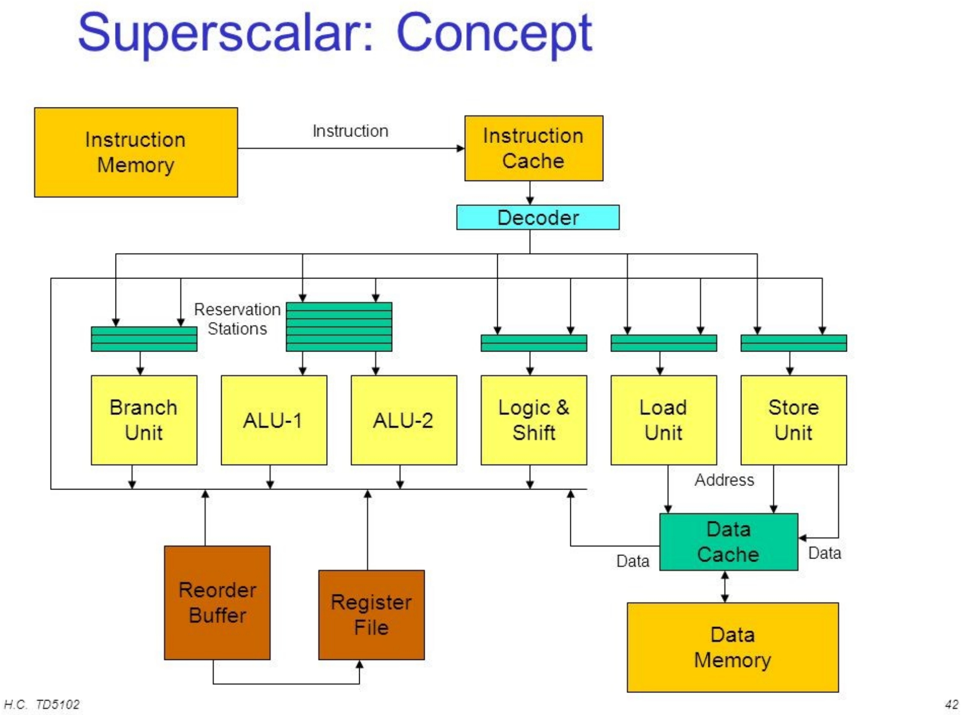
Compute vs. Movement – Changes Afoot



http://iwcse.phys.ntu.edu.tw/plenary/HorstSimon_IWCSE2013.pdf

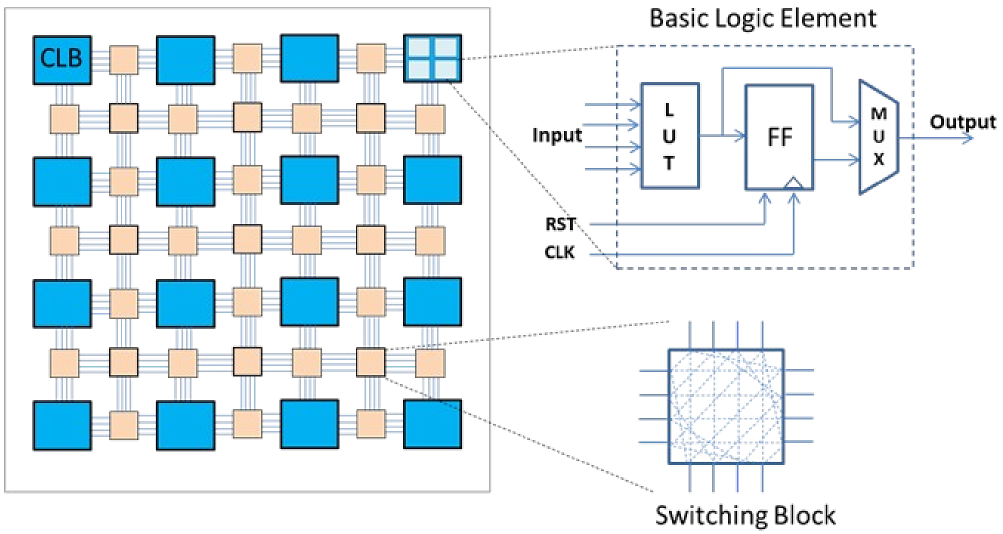
FPGAs vs. CPUs

CPU



http://www.ics.ele.tue.nl/~heco/courses/EmbSystems/adv-architectures.ppt

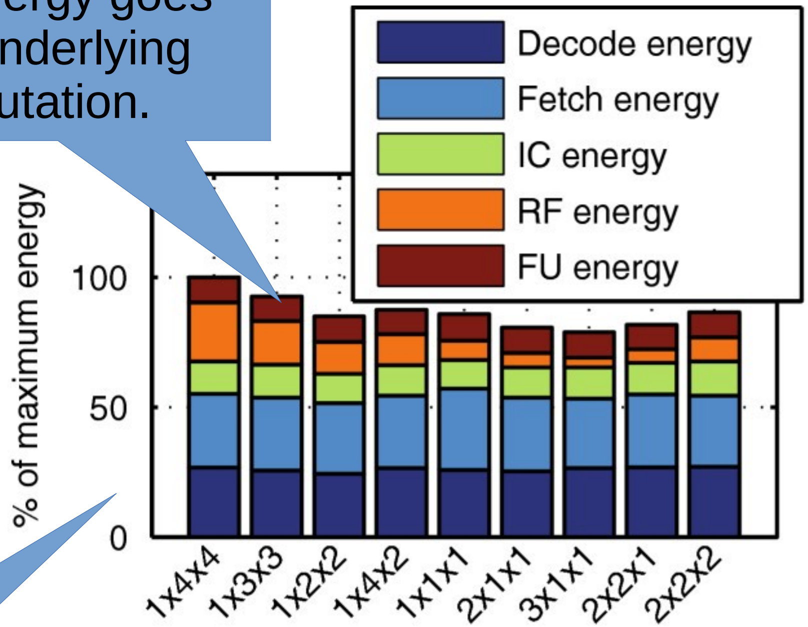
FPGA



http://evergreen.loyola.edu/dhhoe/www/HoeResearchFPGA.htm

Where Does the Power Go (CPU)?

Only a small portion of the energy goes to the underlying computation.



More centralized register files means more data movement which takes more power.

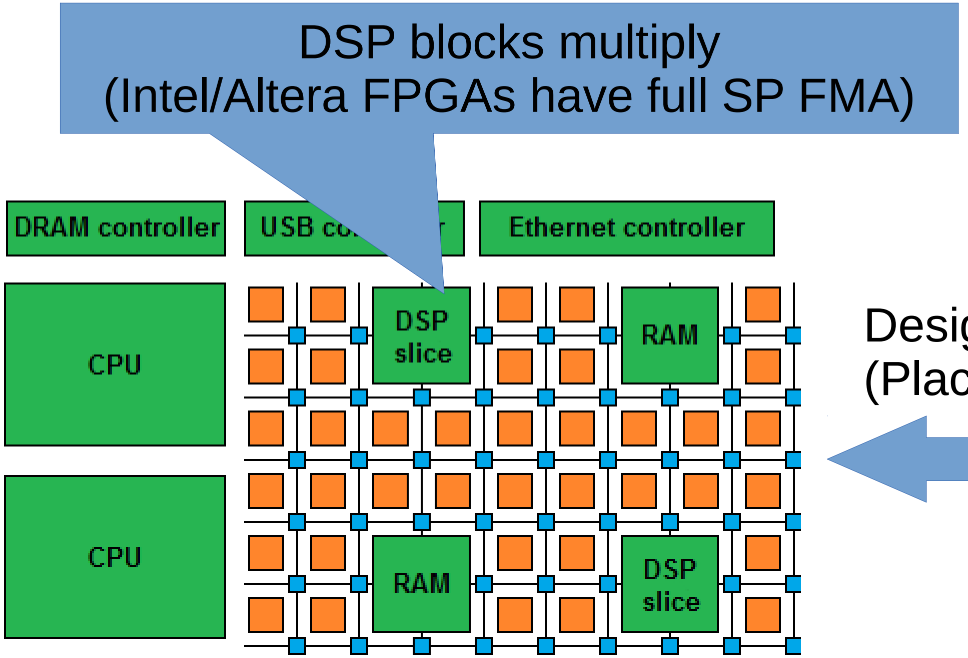
Fetch and decode take most of the energy!

(Model with (# register files) x (read ports) x (write ports))

http://link.springer.com/article/10.1186/1687-3963-2013-9

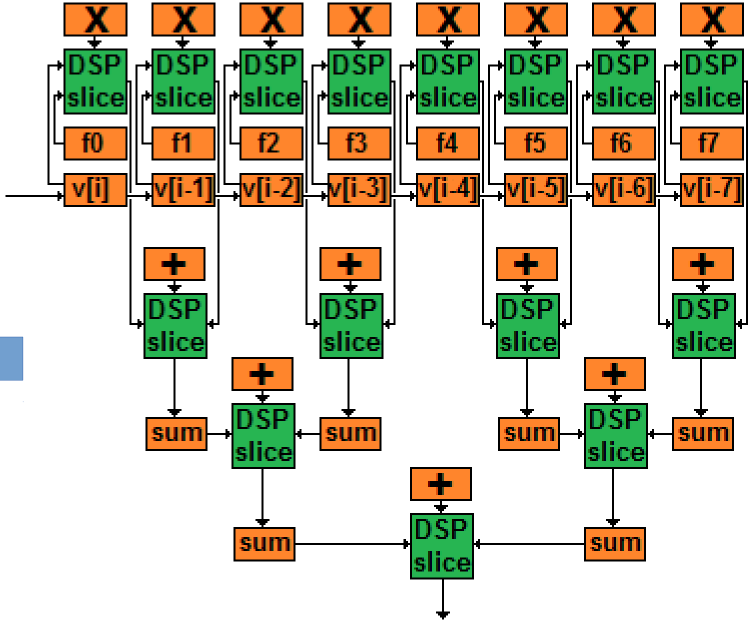
See also: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/tr-2008-130.pdf

Modern FPGAs: DSP Blocks and Block RAM



Modern FPGA: lots of hard, not-field-programmable gates

Design mapped (Place & Route)



A tree-like FPGA pipeline for N=8: v[i] is fed from left, previous elements shifted to the right, 8 values multiplied by f0 ... f7 simultaneously, summation done in a tree of depth log(N)

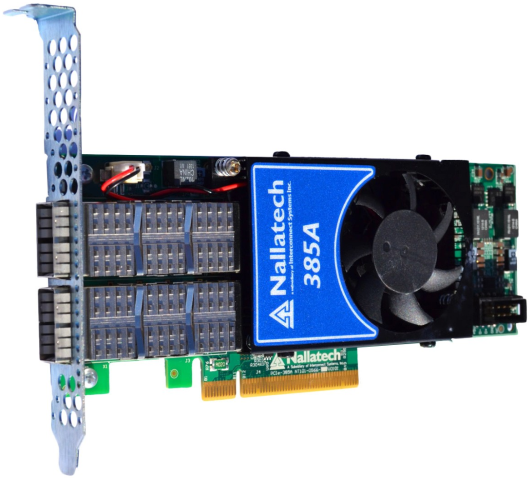
Intel Stratix 10 will have up to:

- 5760 DSP Blocks = 9.2 SP TFLOPS
- 11721 20Kb Block RAMs = 28MB
- 64-bit 4-core ARM @ 1.5 GHz

https://www.altera.com/products/fpga/stratix-series/stratix-10/features.html

http://yosefk.com/blog/category/hardware

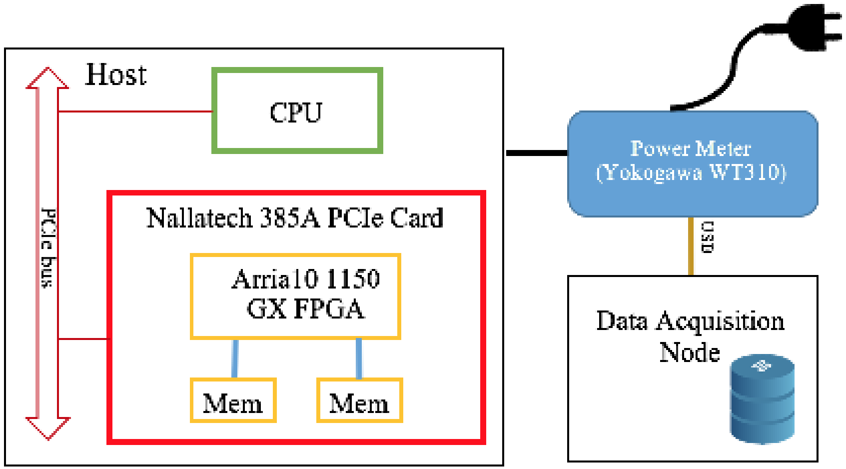
An experiment...

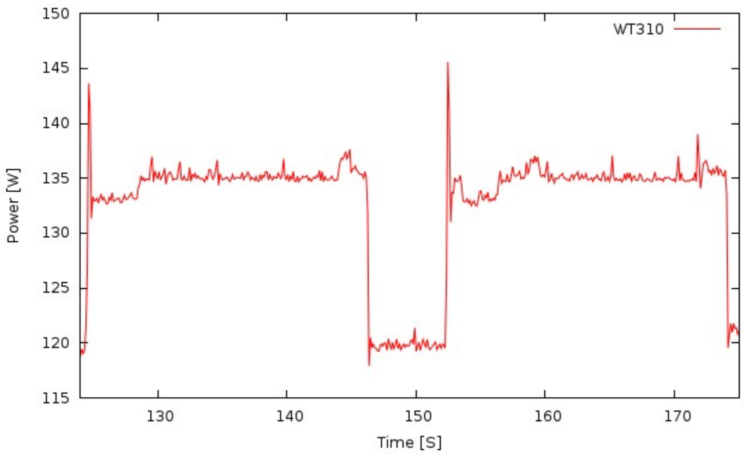


- Sandy Bridge E5-2670
- 2.6 GHz (3.3 GHz w/ turbo)
- 32 nm
- four DRAM channels. **51.2 GB/s peak**

- Nallatech 385A Arria10 board
- 200 – 300 MHz (depend on a design)
- 20 nm
- two DRAM channels. **34.1 GB/s peak**

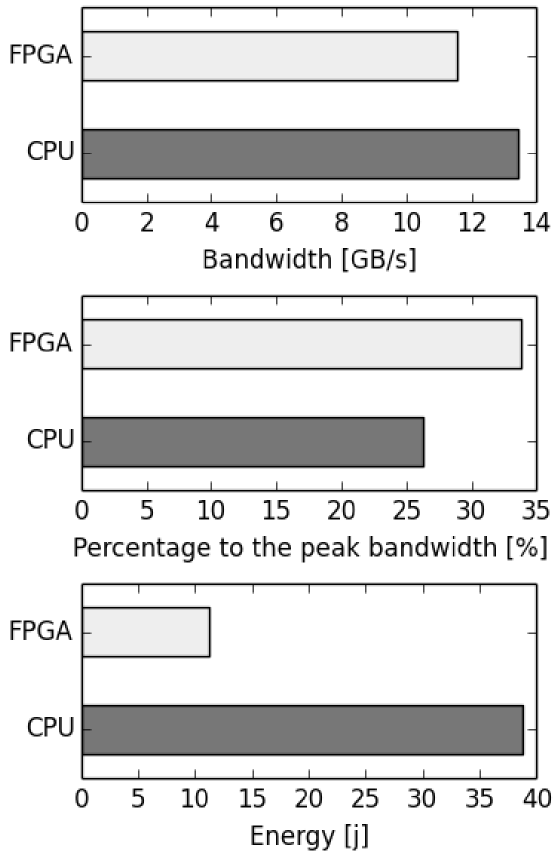
An experiment: Power is Measured...





- Intel RAPL is used to measure CPU energy
  - CPU and memory
- Yokogawa WT310, an external power meter, is used to measure the FPGA power
  - FPGA_pwr = meter_pwr - host_idle_pwr + FPGA_idle_pwr (~17 W)
  - Note that meter_pwr includes both CPU and FPGA

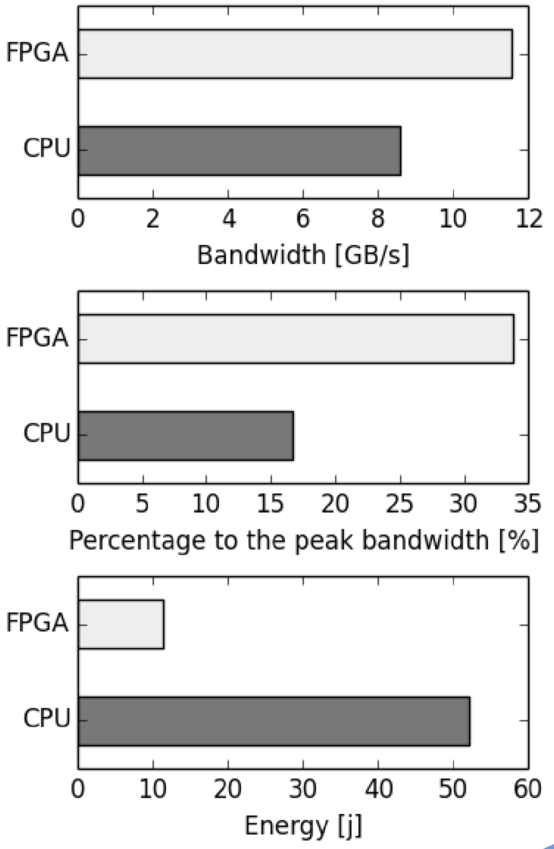
An experiment: Random Access with Computation using OpenCL




```
for (int i = 0; i < M; i++) {
  double8 tmp;
  index = rand() % len;
  tmp = array[index];
  sum += (tmp.s0 + tmp.s1) / 2.0;
  sum += (tmp.s2 + tmp.s3) / 2.0;
  sum += (tmp.s4 + tmp.s5) / 2.0;
  sum += (tmp.s6 + tmp.s7) / 2.0;
}
```

- # work-units is 256
- CPU: Sandy Bridge (4ch memory)
- FPGA: Arria 10 (2ch memory)

An experiment: Random Access with Computation using OpenCL



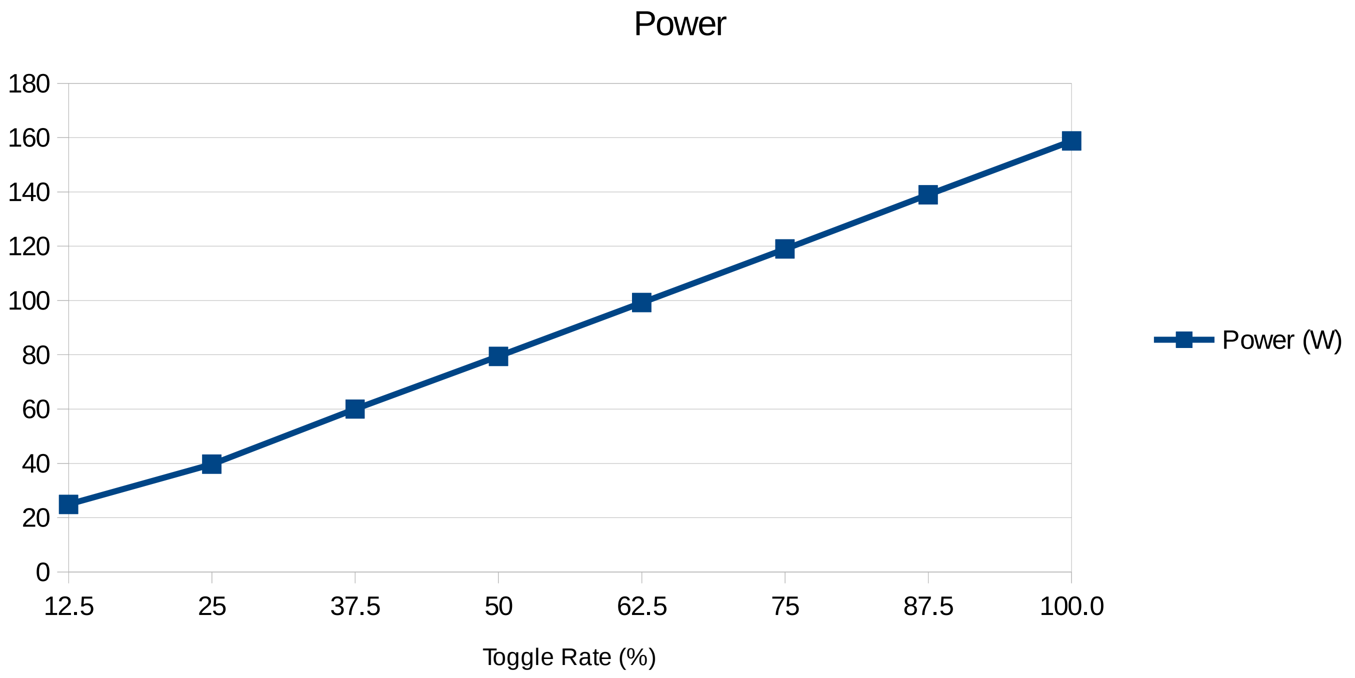

```
for (int i = 0; i < M; i++) {
   double8 tmp;
   index = rand() % len;
   tmp = array[index];
   sum += (tmp.s0 + tmp.s1) / 2.0;
   sum += (tmp.s2 + tmp.s3) / 2.0;
   sum += (tmp.s4 + tmp.s5) / 2.0;
   sum += (tmp.s6 + tmp.s7) / 2.0;
}
```

- # work-units is 256
- CPU: Sandy Bridge (2ch memory)
- FPGA: Arria 10 (2ch memory)

Make the comparison more fair...

FPGAs – Power Estimates at Peak (Compute) Performance

On an Arria 10 (GX1150), if you instantiate all of the DSPs doing floating-point operations (1518 DSPs) and then estimate the power consumption...



What Happens for a “Real” Compute Task

The earth's shape is modeled as an ellipsoid. The shortest distance along the surface of an ellipsoid between two points on the surface is along the geodesic. Computing the geodesic distance (in OpenCL):

```
__kernel void geodesic_distance (__global double * restrict lat1,
                                 __global double * restrict lon1,
                                 __global double * restrict lat2,
                                 __global double * restrict lon2,
                                 __global double * restrict out)
{
   rad_lon1 = lon1 * TO_RADIAN ;
   rad_lat1 = lat1 * TO_RADIAN ;
   rad_lon2 = lon2 * TO_RADIAN ;
   rad_lat2 = lat2 * TO_RADIAN ;

   tu1 = COMPRESSION_FACTOR * sin ( rad_lat1 ) /
         cos ( rad_lat1 ) ;
   tu2 = COMPRESSION_FACTOR * sin ( rad_lat2 ) /
         cos ( rad_lat2 ) ;

   cu1 = 1.0 / sqrt ( tu1 * tu1 + 1.0 ) ;
   su1 = cu1 * tu1 ;
   cu2 = 1.0 / sqrt ( tu2 * tu2 + 1.0 ) ;
   s = cu1 * cu2 ;
   baz = s * tu2 ;
   faz = baz * tu1 ;
   x = rad_lon2 - rad_lon1;
```

BB0

```
do {
   sx = sin ( x ) ;
   cx = cos ( x ) ;
   tu1 = cu2 * sx ;
   tu2 = baz - su1 * cu2 * cx ;
   sy = sqrt ( tu1 * tu1 + tu2 * tu2 ) ;
   cy = s * cx + faz ;
   y = atan2 ( sy, cy ) ;
   sa = s * sx / sy ;
   c2a = - sa * sa + 1.0;
   cz = faz + faz ;
   if ( c2a > 0.0 ) cz = -cz / c2a + cy ;
   e = cz * cz * 2.0 - 1.0 ;
   c = ( ( -3.0 * c2a + 4.0 ) * FLATTENING + 4.0 ) * c2a *
        FLATTENING / 16.0 ;
   d = x ;
   x = ( ( e * cy * c + cz ) * sy * c + y ) * sa ;
   x = ( 1.0 - c ) * x * FLATTENING + rad_lon2 - rad_lon1 ;
} while ( fabs ( d - x ) > EPS ) ;
```

BB1

```
x = sqrt ( ELLIPSOIDAL * c2a + 1.0 ) + 1.0 ;
x = ( x - 2.0 ) / x ;
c = 1.0 - x ;
c = ( x * x / 4.0 + 1.0 ) / c ;
d = ( 0.375 * x * x - 1.0 ) * x ;
x = e * cy ;
s = 1.0 - e - e ;
s = ( ( ( ( sy * sy * 4.0 - 3.0 ) * s * cz * d / 6.0 - x ) *
    d / 4.0 + cz ) * sy * d + y ) * c * POLAR_RADIUS ;
out[i] = s;
}
```

BB2

What Happens for a "Real" Compute Task

On an Arria 10 GX1150 FPGA (Nallatech 385A), for single precision:

	cu1	cu4	cu9
Logic utilization	15%	28%	49%
Memory bits	8%	12%	17%
RAM blocks	18%	35%	63%
#DSPs	160	640	1440
Fmax (MHz)	280	255	212

For double precision:

(fpc) == --fp-relaxed

	cu1	cu1 (fpc)	cu2	cu2 (fpc)
Logic utilization	36%	28%	61%	45%
Memory bits	14%	14%	22%	21%
RAM blocks	25%	25%	44%	38%
#DSPs	515	515	1030	1030
Fmax (MHz)	230	233	227	221

What Happens for a “Real” Compute Task

Power and Time...

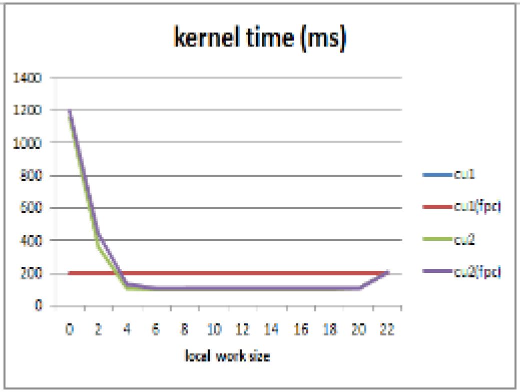


Fig 2. Kernel execution time of the double-precision implementations. The local work size in the x axis indicates $2^{\text{local work size}}$

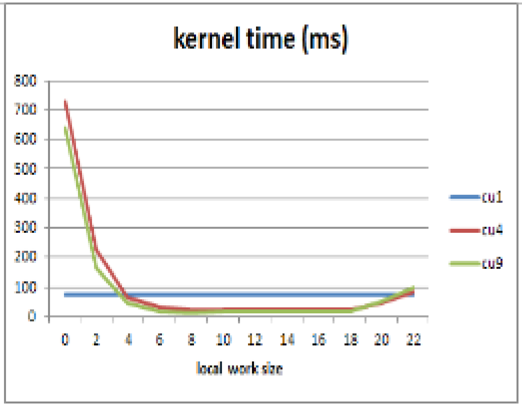


Fig 3. Kernel execution time of the single-precision implementations. The local work size in the x axis indicates $2^{\text{local work size}}$

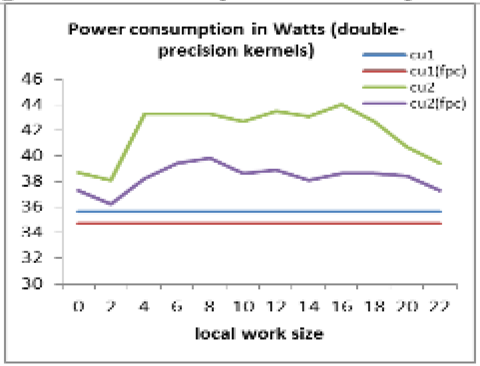


Fig 4. Power consumption of the double-precision kernel implementations. The local work size in the x axis indicates $2^{\text{local work size}}$

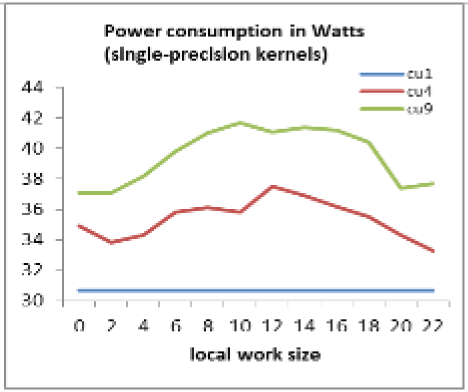


Fig 5. Power consumption of the single-precision kernel implementations. The local work size in the x axis indicates $2^{\text{local work size}}$

Optimal time vs. optimal power can differ a lot.

What Happens for a "Real" Compute Task

And so…

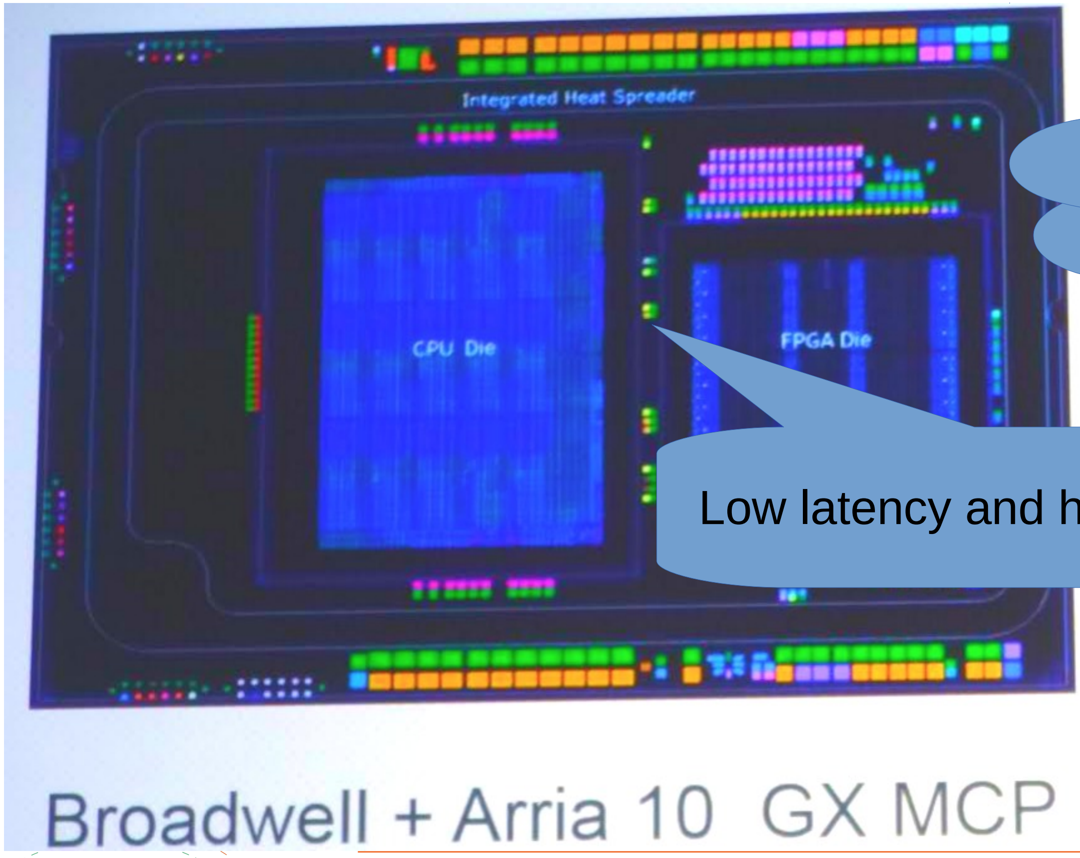
Comparing the Arria 10, an Intel Xeon Phi Knights Landing (KNL) 7210 processor with 64 cores and four threads per core, and an NVIDIA K80 with 2496 cores.

	CPU DP	CPU SP	GPU DP	GPU SP	FPGA DP	FPGA SP
Execution time (ms)	18.3	4	17.7	5.4	100.5	13
Maximum power (W)	190	190	145.5	136.7	44	42

The power efficiency of the single-precision kernel on FPGA is 1.35X better than K80 and KNL7210 while the power efficiency of the double-precision kernel on FPGA 1.36X and 1.72X worse than CPU and GPU respectively.

High-End CPU + FPGA Systems Are Coming...

- Intel/Altera are starting to produce Xeon + FPGA systems
- Xilinx are producing ARM + FPGA systems



These are not just embedded cores, but state-of-the-art multicore CPUs

Low latency and high bandwidth

A cache!

CPU + FPGA systems fit nicely into the HPC accelerator model! (“#pragma omp target” can work for FPGAs too)

https://www.nextplatform.com/2016/03/14/intel-marrying-fpga-beefy-broadwell-open-compute-future/

Challenges Remain...

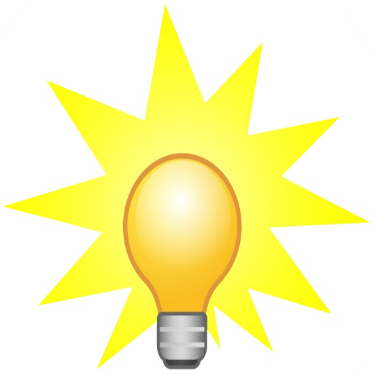
- OpenMP 4 technology for FPGAs is in its infancy (even less mature than the GPU implementations).
- High-level synthesis technology has come a long way, but is just now starting to give competitive performance to hand-programmed HDL designs.
- CPU + FPGA systems with cache-coherent interconnects are very new.
- High-performance overlay architectures have been created in academia, but none targeting HPC workloads. High-performance on-chip networks are tricky.
- No one has yet created a complete HPC-practical toolchain.

Theoretical maximum performance on many algorithms on GPUs is 50-70%.
This is lower than CPU systems, but CPU systems have higher overhead.

In theory, FPGAs offer high percentage of peak and low overhead,
but can that be realized in practice?

Conclusions

- FPGA technology offers the most-promising direction toward higher FLOPS/Watt.
- FPGAs, soon combined with powerful CPUs, will naturally fit into our accelerator-infused HPC ecosystem.
- FPGAs can compete with CPUs/GPUs on traditional workloads while excelling at bioinformatics, machine learning, and more!
- Combining high-level synthesis with overlay architectures can address FPGA programming challenges.
- Even so, pulling all of the pieces together will be challenging!



- ALCF is supported by DOE/SC under contract DE-AC02-06CH11357

Extra Slides

FPGAs – Molecular Dynamics – Strong Scaling Again!

Martin Herbordt (Boston University)

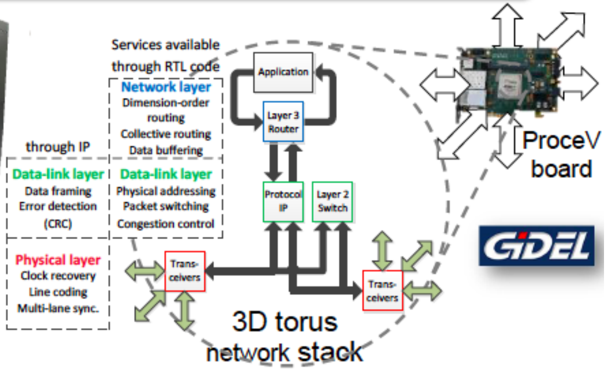
Goal: Enable large-scale app acceleration with a reconfigurable 3D-torus network

Motivation: Large-scale RSC apps are communication-bound

Turn communication-bound problems into computation-bound problems

Approach

- Novo-G# network design to support multi-FPGA apps efficiently
  - ✓ 40 Gbps link support, <10% FPGA util.
- Modeling & simulation of novel topologies, architectures & protocols
  - ✓ Scalable, accurate VisualSim model avail.
- OpenCL support for productive multi-FPGA development
  - ✓ BSP* with inter-FPGA channel support avail.
- **Case study: 3D FFT**



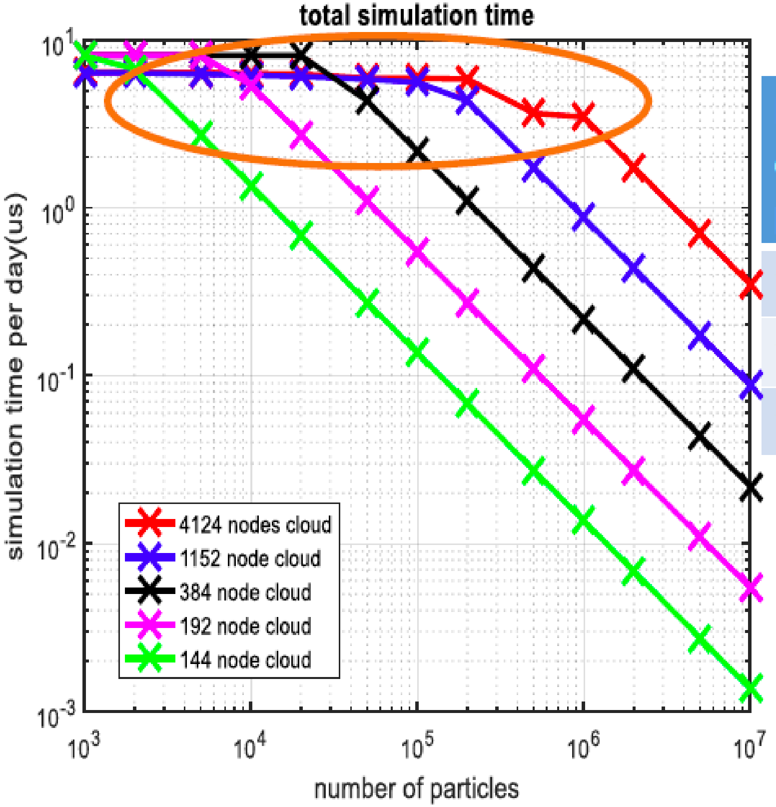
Novo-G#
- 128 Gidel ProceV (Stratix V D8)
- 3D torus or 6D hypercube
- 6 Rx-Tx links per FPGA
- <10KW

*BSP: Board-support package

FPGAs – Molecular Dynamics – Strong Scaling Again!

Martin Herbordt (Boston University)

Limited by FFT performance



Simulation time/day=2fs*86400/time per iter

Higher is better!

Compare with state-of-the-art (unit: us/day)

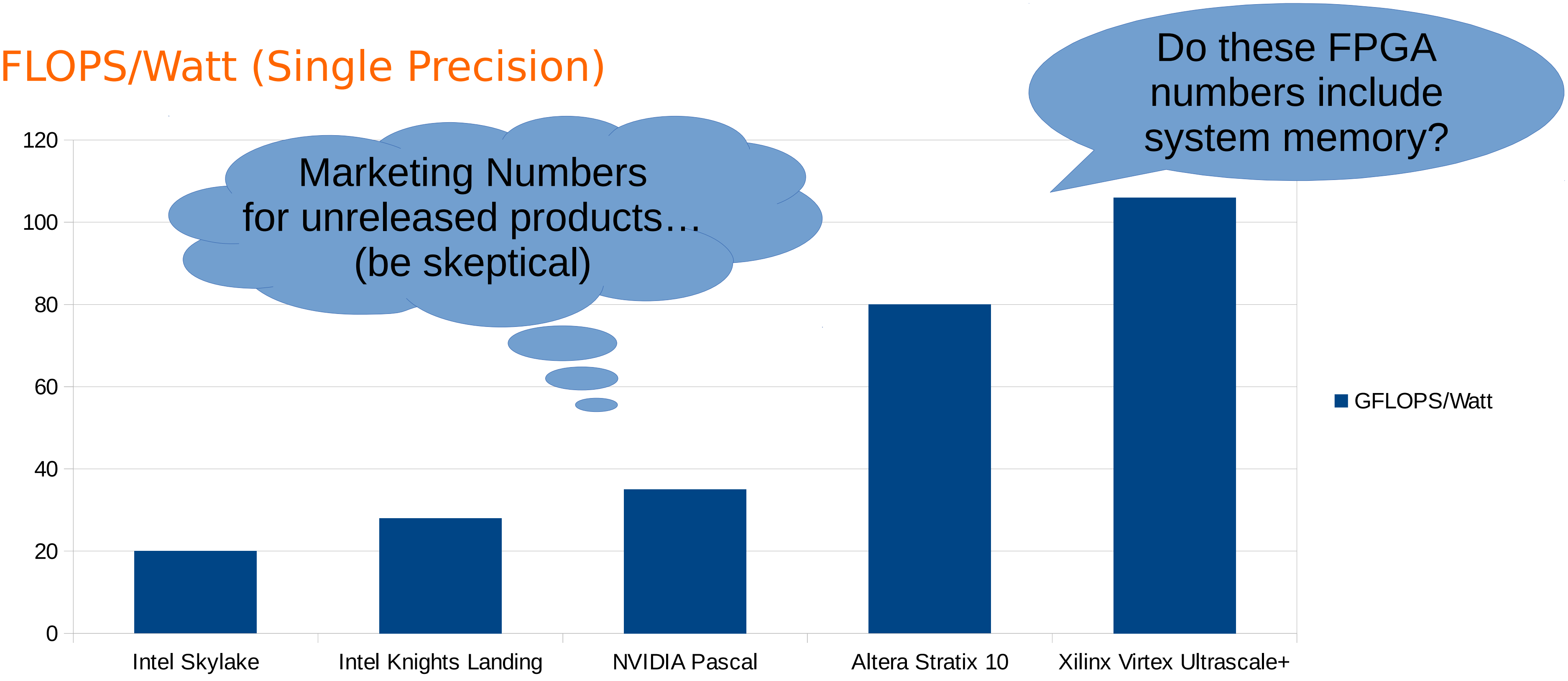
Number of particles	Cloud	Anton2 [3]	Anton1[4]	CPU cluster or GPU
13K	8.05	85.8	19.7	1.1(a)
100K	5.89	59.4	7.5	0.29(b)
1M	3.46	9.5	Not avail.	0.035(c)

(a) GROMACS on a Xeon E5-2690 processor with an NVIDIA GTX TITAN GPU[5]

(b) Desmond on 1,024 cores of a Xeon E5430 cluster[6]

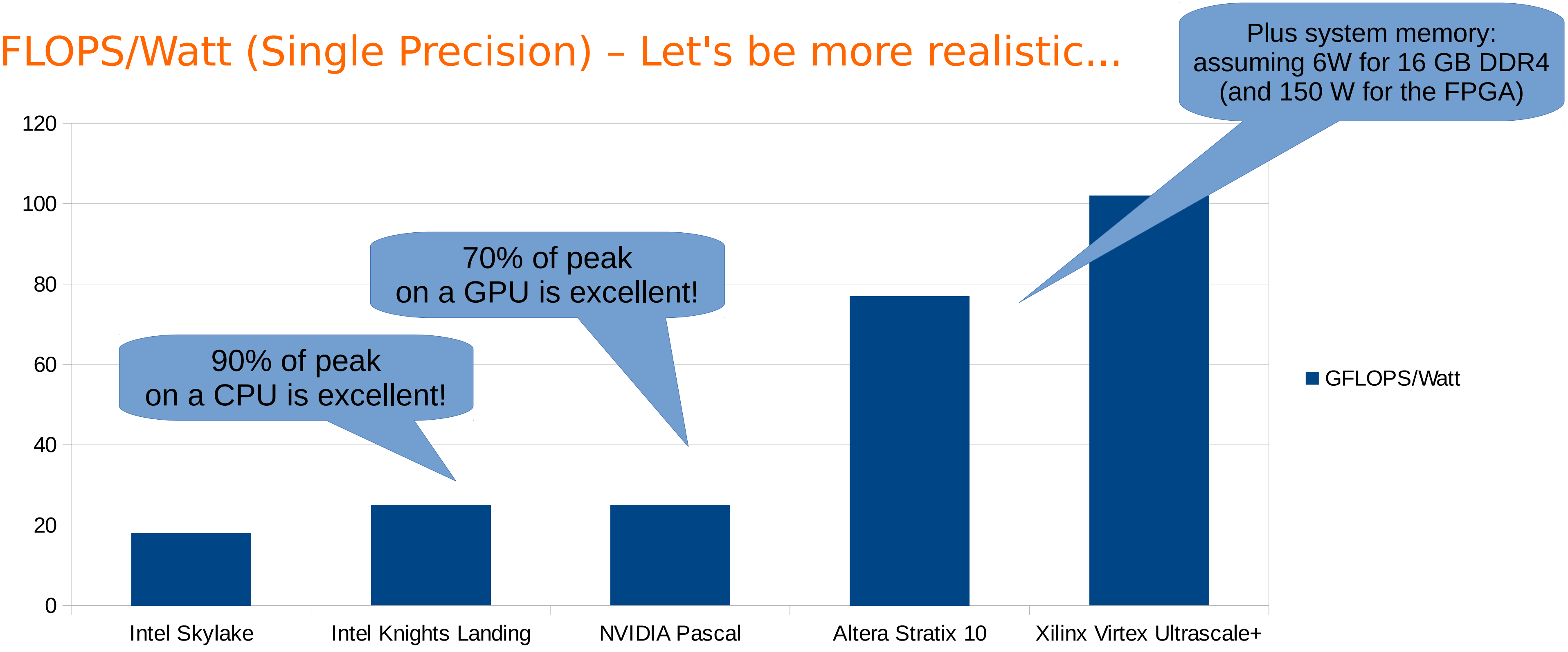
(c) NAMD on 16,384 cores of Cray Jaguar XK6[7]

GFLOPS/Watt (Single Precision)



- http://wccftech.com/massive-intel-xeon-e5-xeon-e7-skylake-purley-biggest-advancement-nehalem/ - Taking 165 W max range
- http://cgo.org/cgo2016/wp-content/uploads/2016/04/sodani-slides.pdf
- http://www.xilinx.com/applications/high-performance-computing.html - Ultrascale+ figure inferred by a 33% performance increase (from Hotchips presentation)
- https://devblogs.nvidia.com/parallelforall/inside-pascal/
- https://www.altera.com/products/fpga/stratix-series/stratix-10/features.html

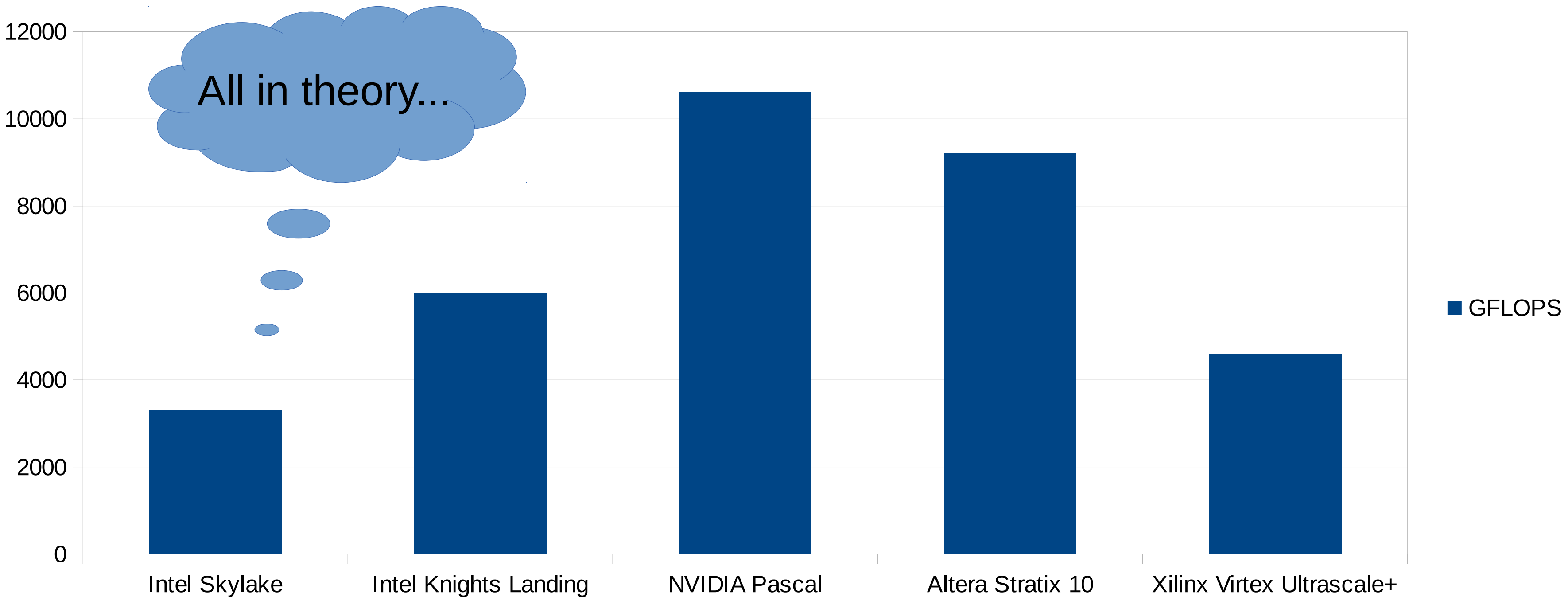
GFLOPS/Watt (Single Precision) – Let's be more realistic...



- http://www.tomshardware.com/reviews/intel-core-i7-5960x-haswell-e-cpu,3918-13.html
- https://hal.inria.fr/hal-00686006v2/document
- http://www.eecg.toronto.edu/~davor/papers/capalija_fpl2014_slides.pdf - Tile approach yields 75% of peak clock rate on full device

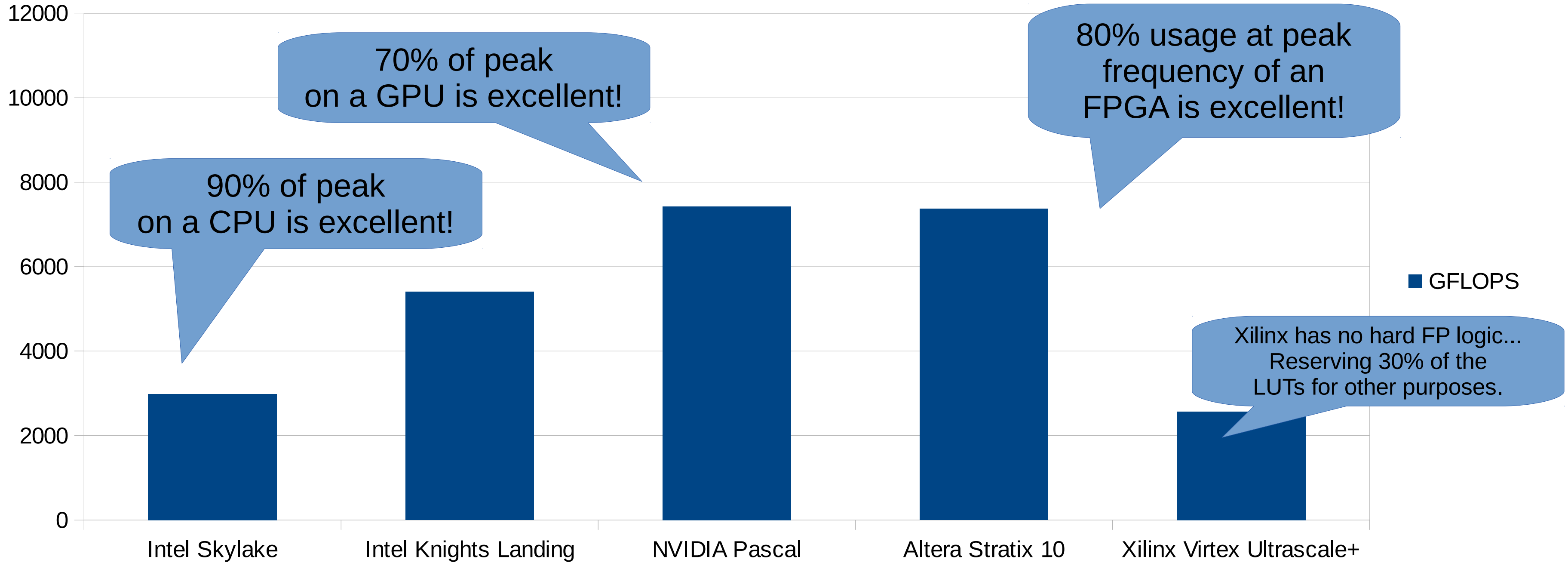
Conclusion: FPGAs are a competitive HPC accelerator technology by 2017!

GFLOPS/device (Single Precision)



- https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/pt/stratix-10-product-table.pdf - Largest variant with all DSPs doing FMAs @ the 800 MHz max
- http://www.xilinx.com/support/documentation/ip_documentation/ru/floating-point.html
- http://www.xilinx.com/support/documentation/selection-guides/ultrascale-plus-fpga-product-selection-guide.pdf - LUTs, not DSPs, are the limiting resource – filling device with FMAs @ 1 GHz
- https://devblogs.nvidia.com/parallelforall/inside-pascal/
- http://wccftech.com/massive-intel-xeon-e5-xeon-e7-skylake-purley-biggest-advancement-nehalem/ - 28 cores @ 3.7 GHz * 16 FP ops per cycle * 2 for FMA (assuming same clock rate as the E5-1660 v2)
- http://cgo.org/cgo2016/wp-content/uploads/2016/04/sodani-slides.pdf

GFLOPS/device (Single Precision) – Let's be more realistic...



- https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/wp/wp-01222-understanding-peak-floating-point-performance-claims.pdf
- https://www.altera.com/en_US/pdfs/literature/wp/wp-01028.pdf (old but still useful)

Common Algorithm Classes in HPC

Algorithm / Science areas	Dense linear algebra	Sparse linear algebra	Spectral Methods (FFTs)	Particle Methods	Structured Grids	Unstructured or AMR Grids	Data Intensive
Accelerator Science		X	X	X	X	X	
Astrophysics	X	X	X	X	X	X	X
Chemistry	X	X	X	X			X
Climate			X		X	X	X
Combustion					X	X	X
Fusion	X	X		X	X	X	X
Lattice Gauge		X	X	X	X		
Material Science	X		X	X	X		

http://crd.lbl.gov/assets/pubs_presos/CDS/ATG/WassermanSOTON.pdf

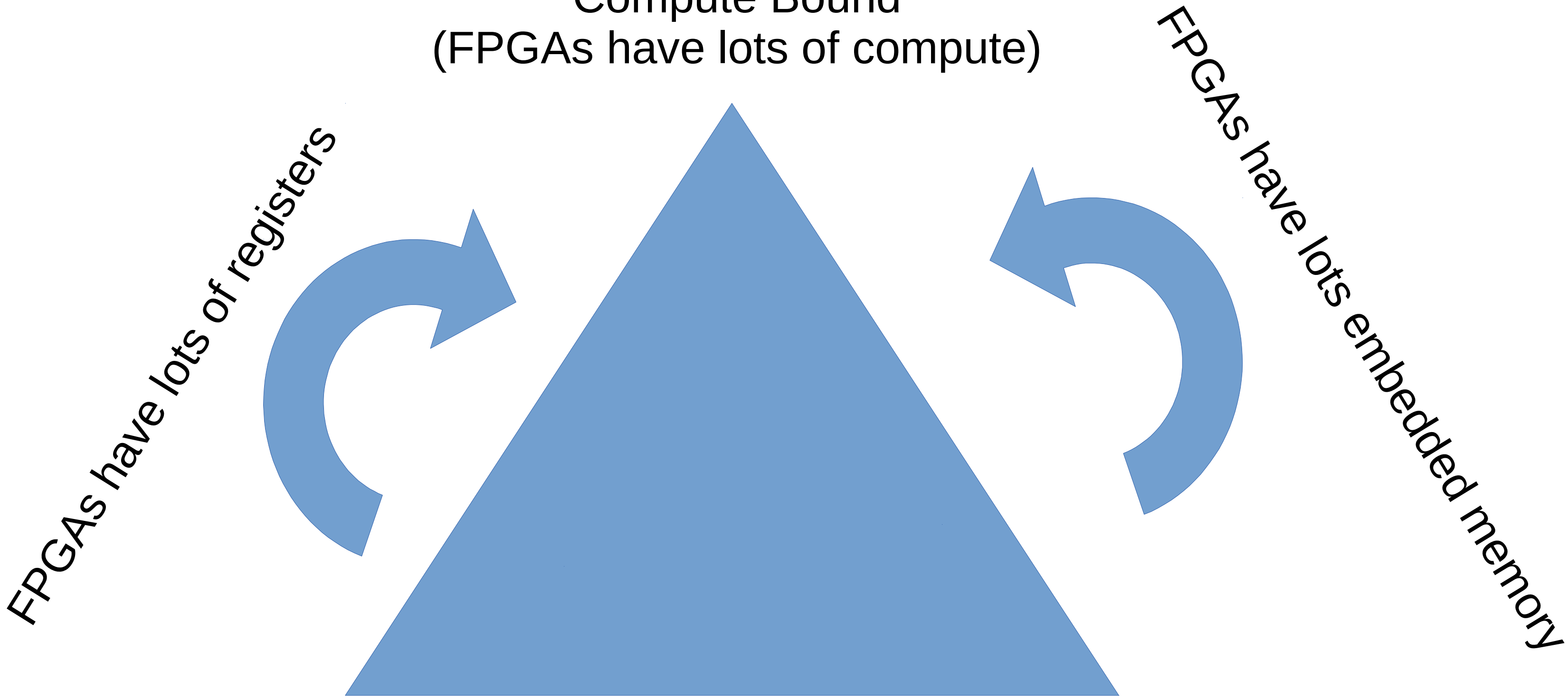
Common Algorithm Classes in HPC – What do they need?

Algorithm / Science areas	Dense linear algebra	Sparse linear algebra	Spectral Methods (FFT)s	Particle Methods	Structured Grids	Unstructured or AMR Grids	Data Intensive
	High Flop/s rate	High performance memory system	High bisection bandwidth	High performance memory system	High flop/s rate	Low latency, efficient gather /scatter	Storage, Network Infrastructure
Accelerator Science							
Astrophysics							
Chemistry							
Climate							
Combustion							
Fusion							
Lattice Gauge							
Material Science							

http://crd.lbl.gov/assets/pubs_presos/CDS/ATG/WassermanSOTON.pdf

FPGAs Can Help Everyone!

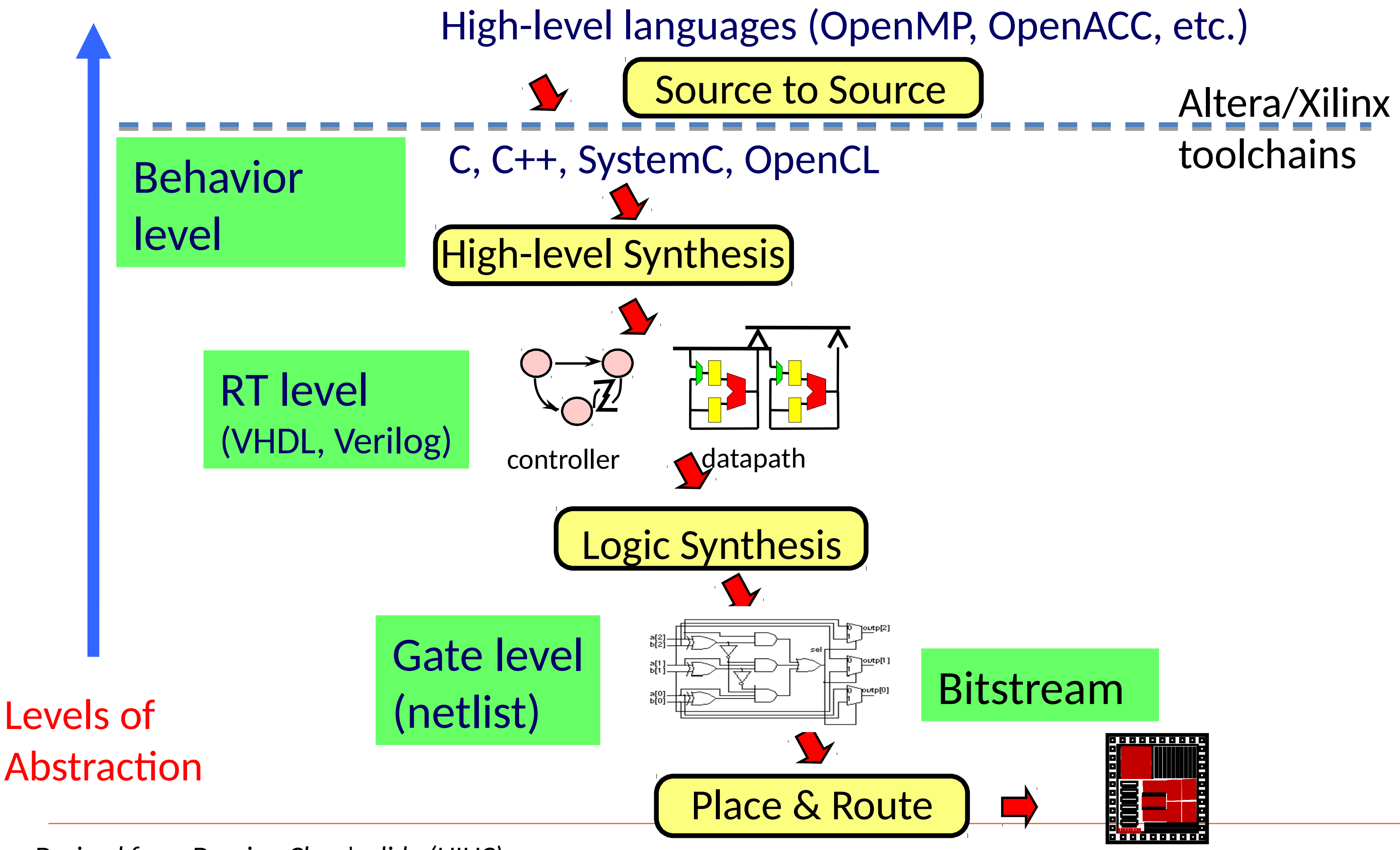
Compute Bound
(FPGAs have lots of compute)



Memory-Latency Bound
(FPGAs can pipeline deeply)

Memory-Bandwidth Bound
(FPGAs can do on-the-fly compression)

FPGA Programming: Levels of Abstraction



Derived from Deming Chen's slide (UIUC).

FPGA Programming Techniques

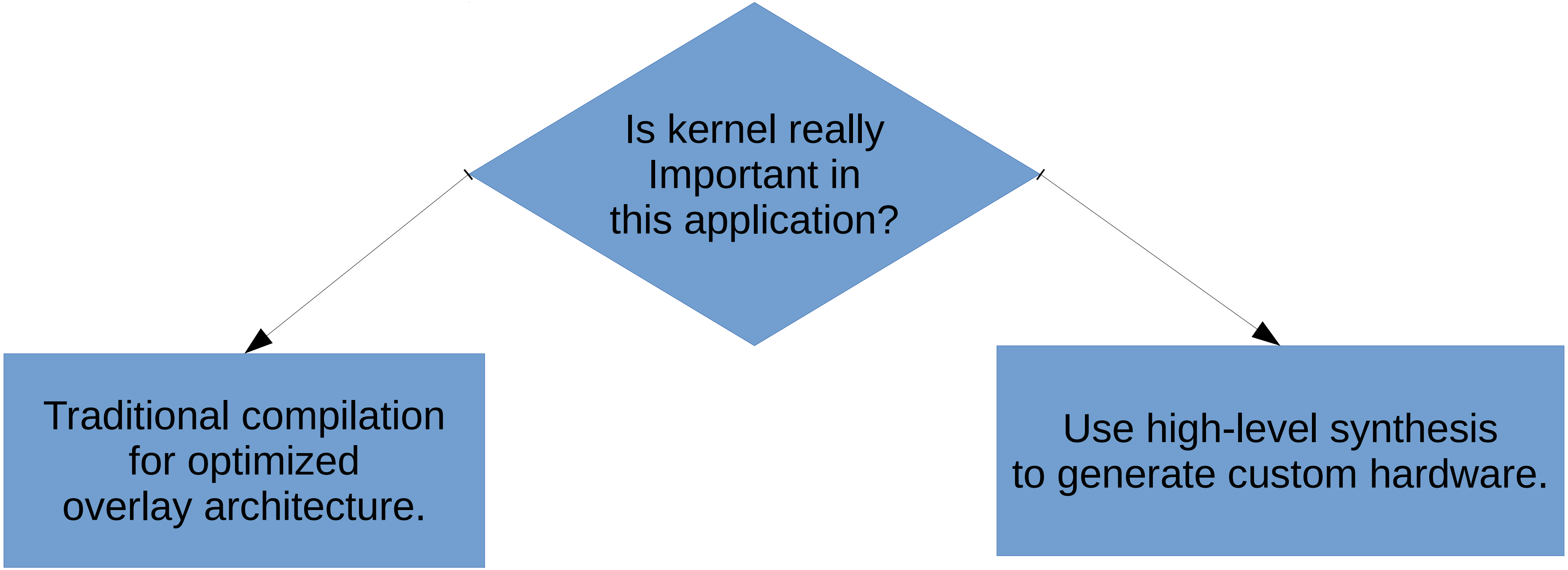
- Lowest Risk
- Lowest User Difficulty

- Use FPGAs as accelerators through (vendor-)optimized libraries
- Use of FPGAs through overlay architectures (pre-compiled custom processors)
- Use of FPGAs through high-level synthesis (e.g. via OpenMP)
- Use of FPGAs through programming in Verilog/VHDL (the FPGA “assembly language”)

- Highest Risk
- Highest User Difficulty

Beware of Compile Time...

- Compiling a full design for a large FPGA (synthesis + place & route) can take many hours!
- Tile-based designs can help, but can still take tens of minutes!
- Overlay architectures (pre-compiled custom processors and on-chip networks) can help...



Overlay (iDEA)

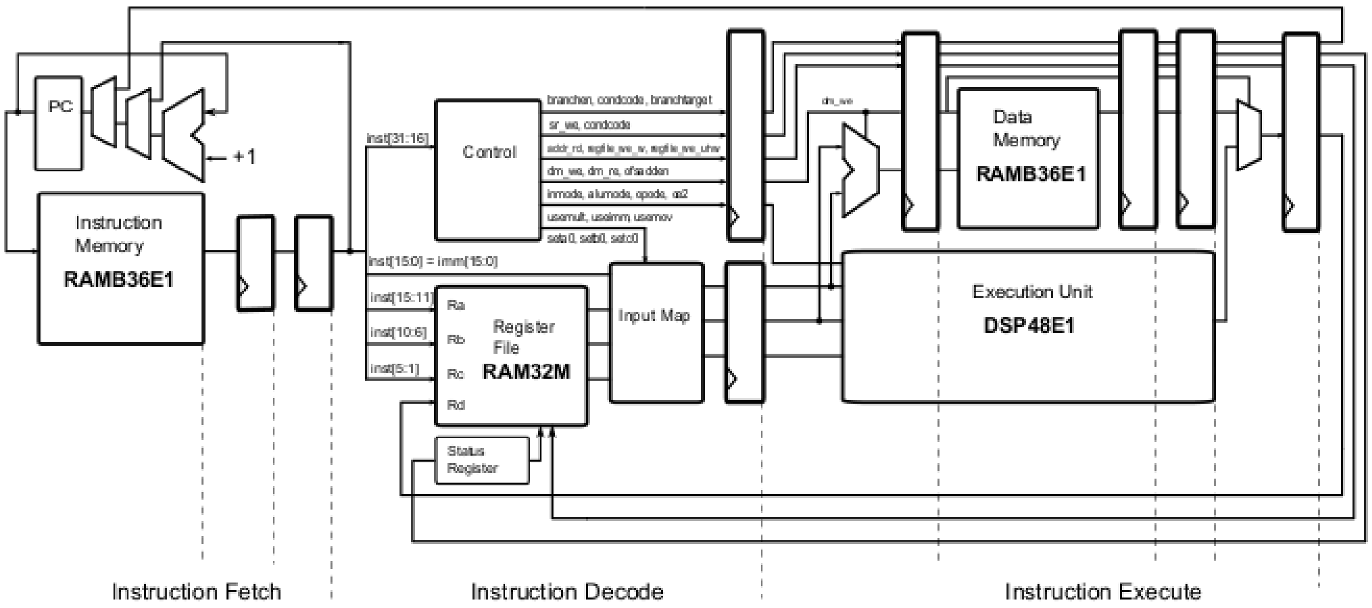


Fig. 2. Processor Block Diagram.

https://www2.warwick.ac.uk/fac/sci/eng/staff/saf/publications/fpt2012-cheah.pdf

- A very-small CPU.
- Runs near peak clock rate of the block RAM / DSP block!
- Makes use of dynamic configuration of the DSP block.

Overlay (DeCO)

Each of these is a small soft CPU.

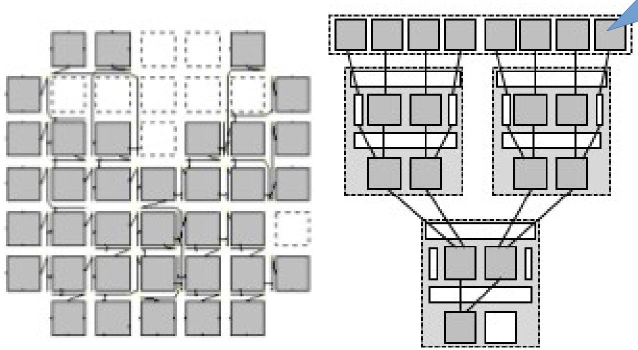


Fig. 8: Mapping of *kmeans* on Overlay-II vs. DeCO.

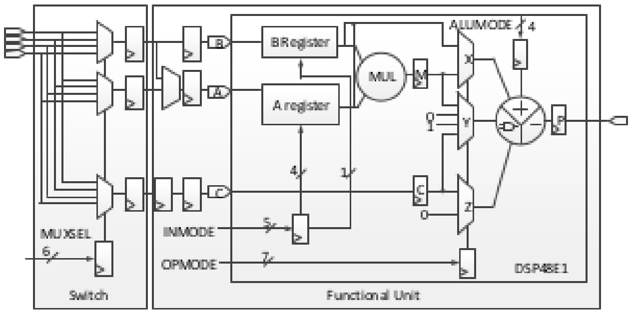
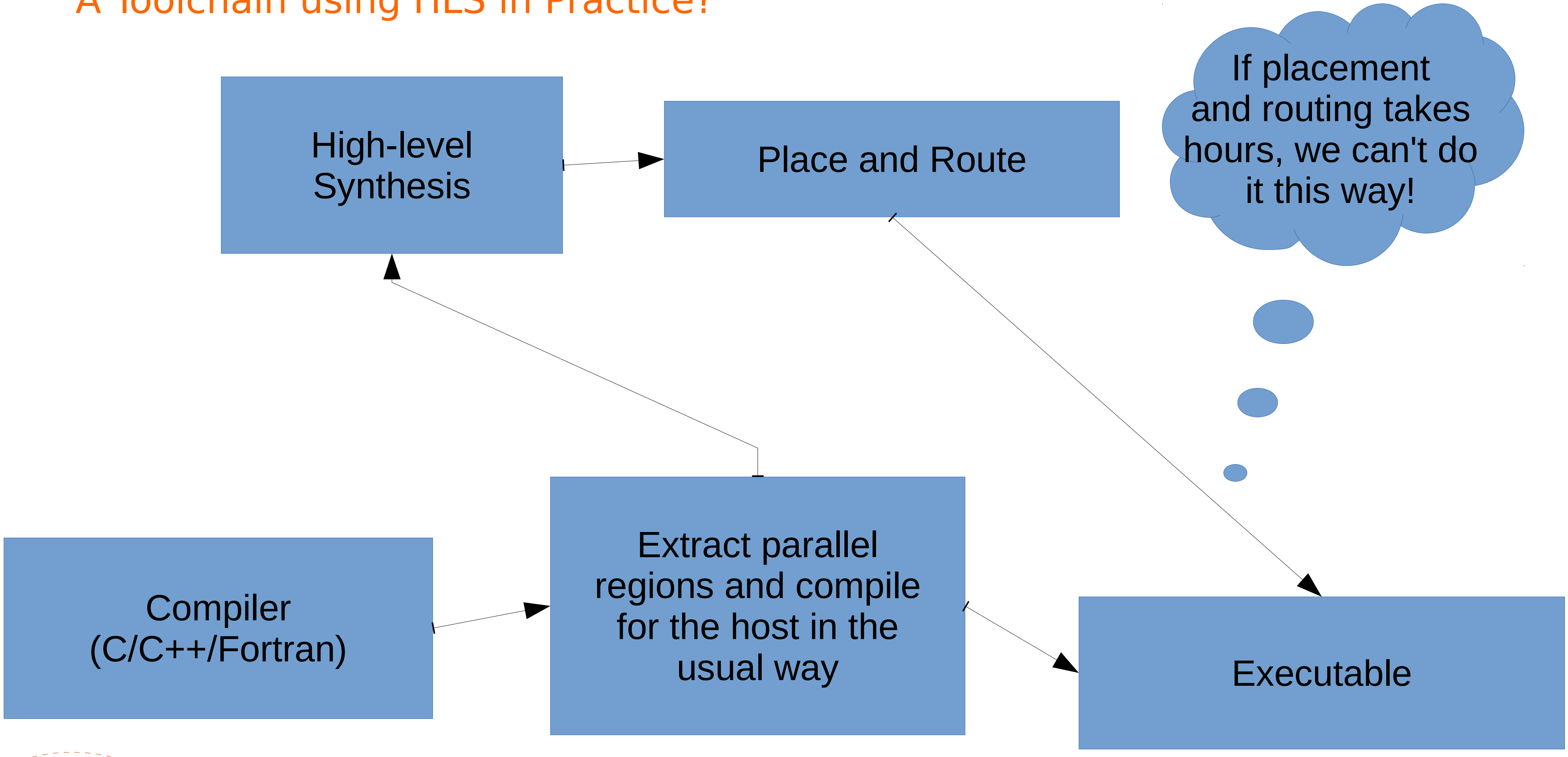


Fig. 4: The 32-bit functional unit and interconnect switch.

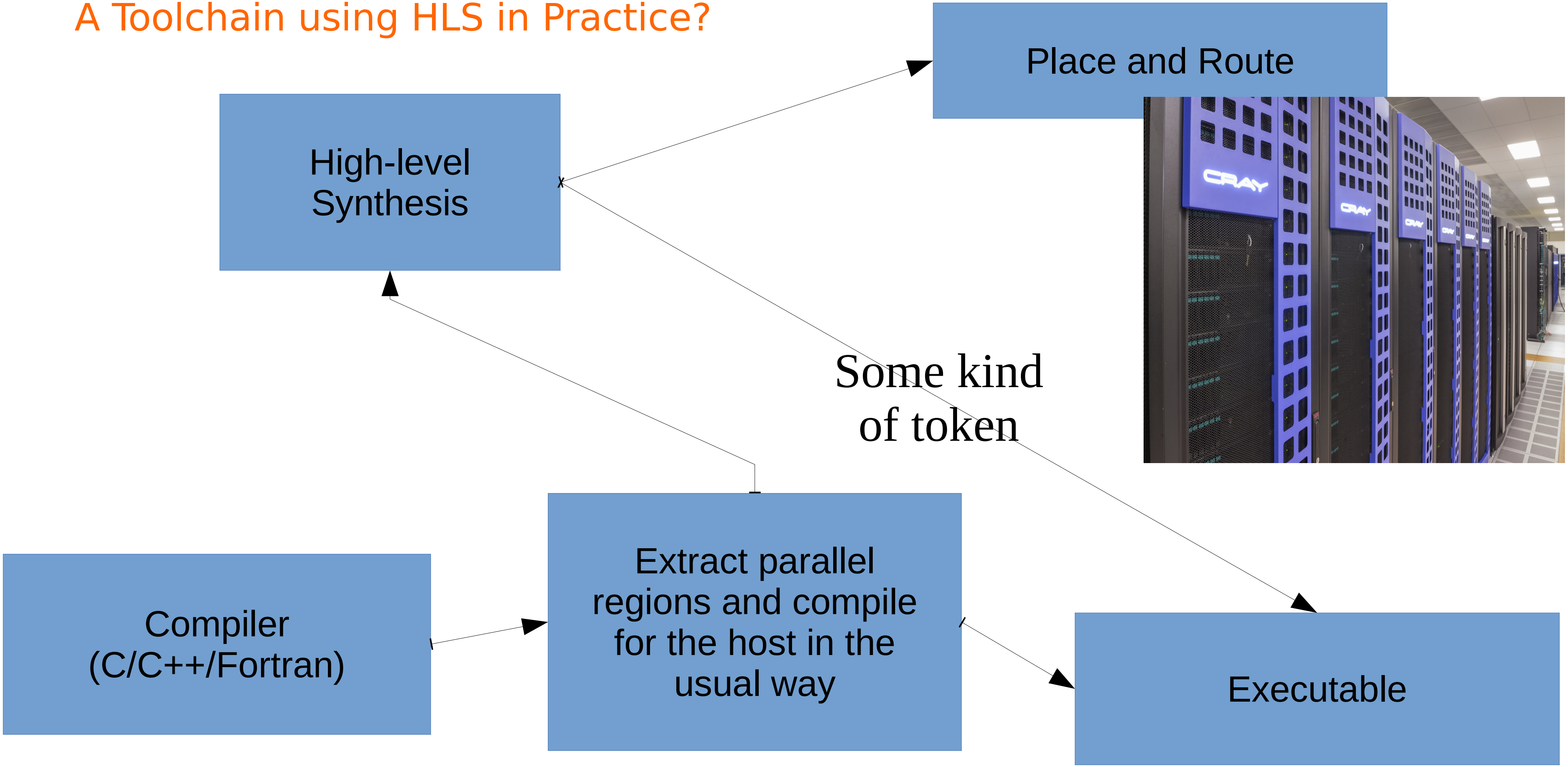
https://www2.warwick.ac.uk/fac/sci/eng/staff/saf/publications/fccm2016-jain.pdf

- Also spatial computing, but with much coarser resources.
- Place & Route is **much** faster!
- Performance is very good.

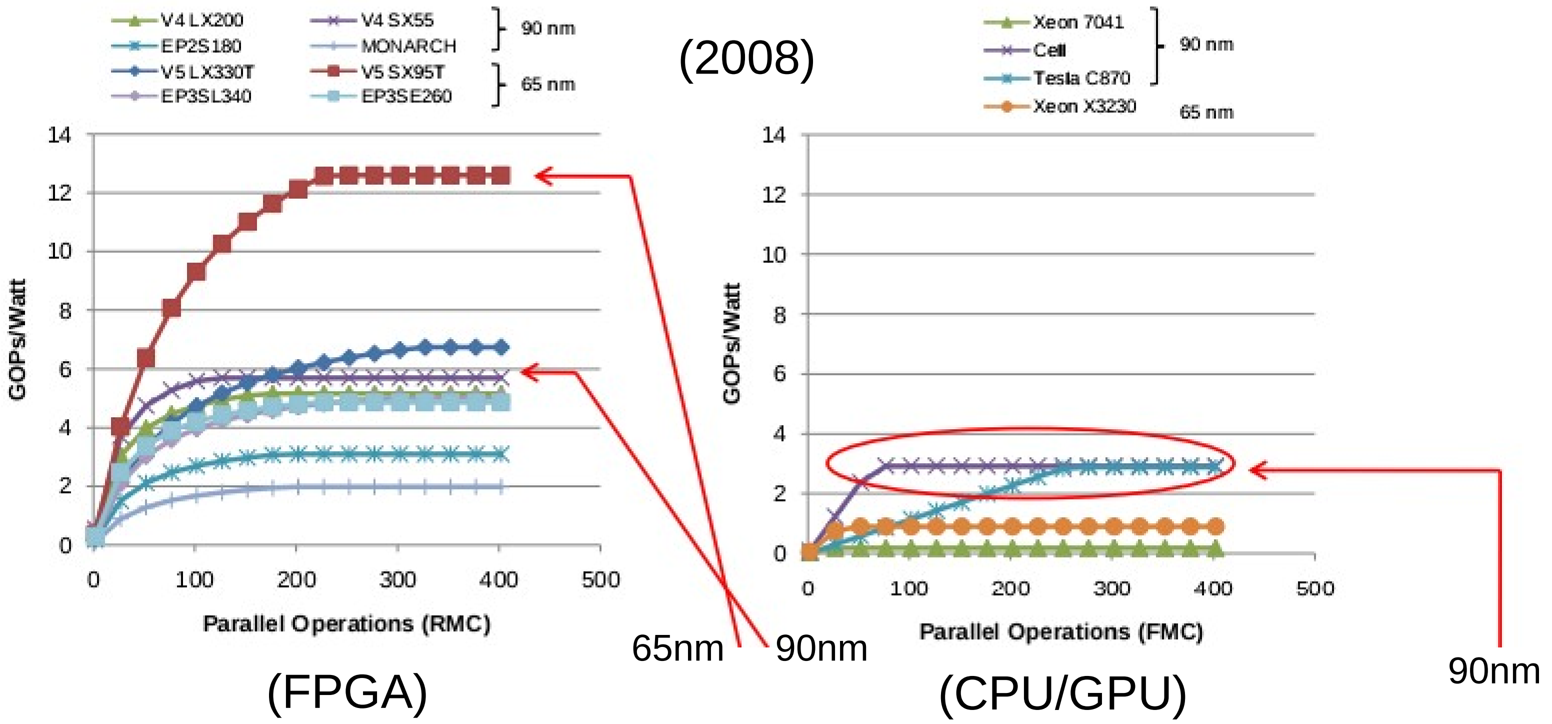
A Toolchain using HLS in Practice?



A Toolchain using HLS in Practice?

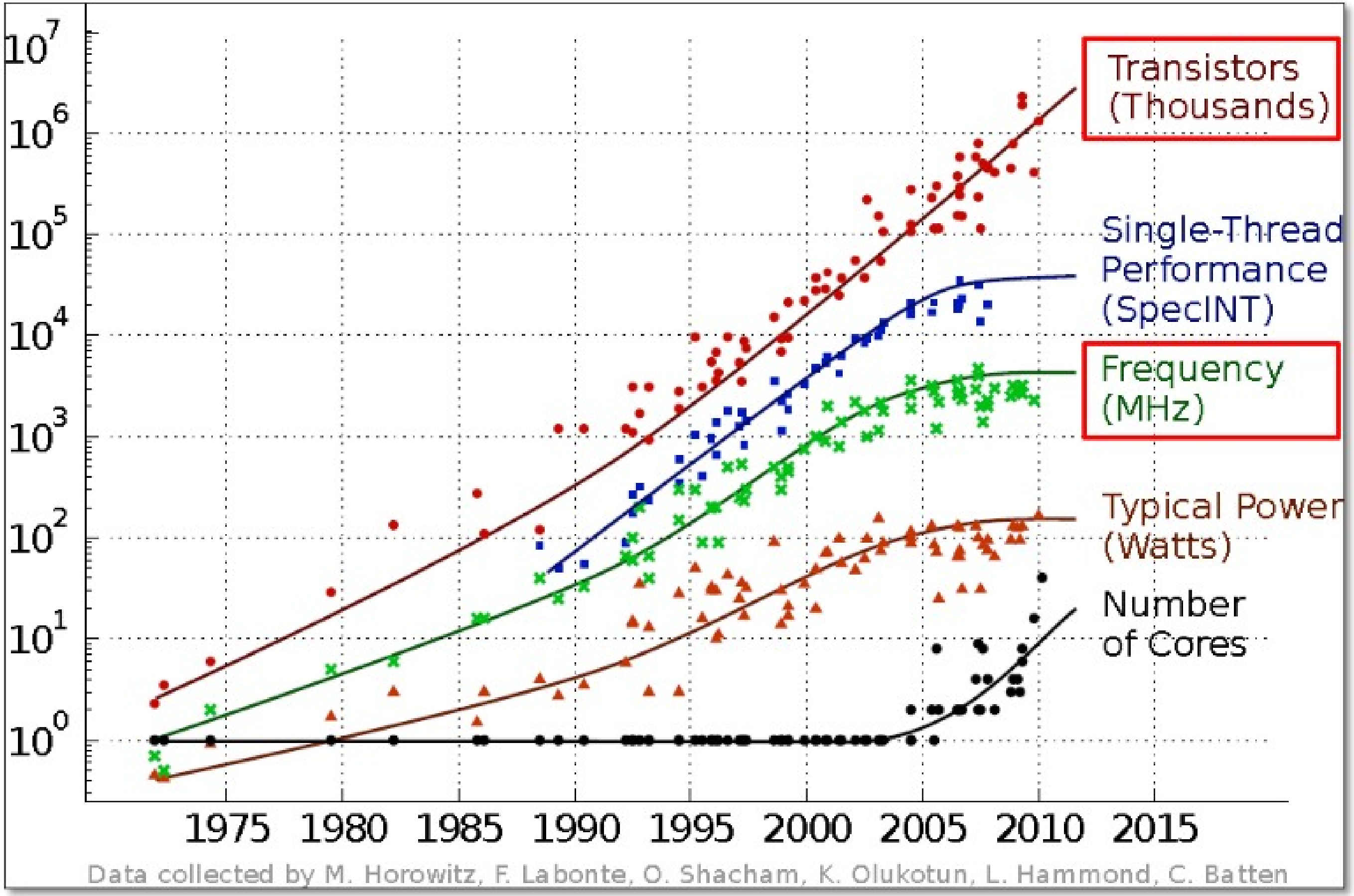


For FPGAs, Parallelism is Essential



http://rssi.ncsa.illinois.edu/proceedings/academic/Williams.pdf

Progress in CMOS CPU Technology



Moore's Law continues

- Transistor count still doubles every 24 months

Dennard scaling stalls

- Voltage
- Clock Speed
- Power
- Performance/clock

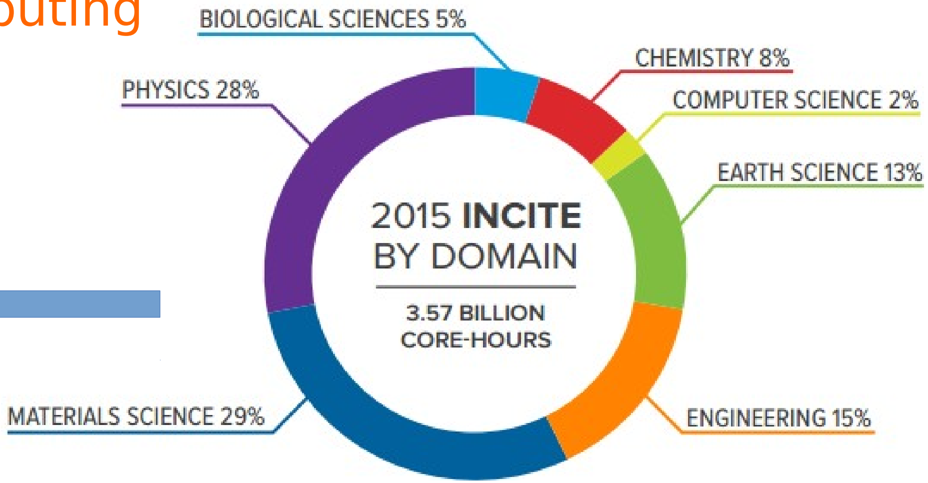
http://fire.pppl.gov/FESAC_AdvComput_Binkley_041014.pdf

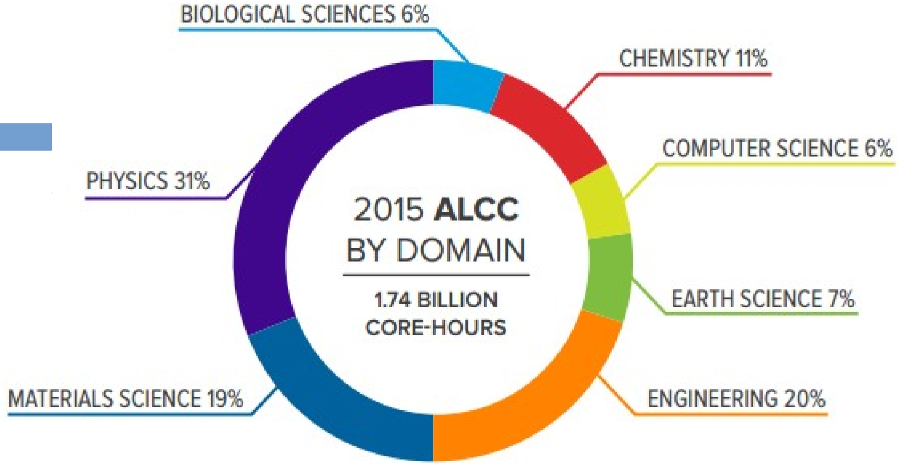
ALCF Systems

How They Compare	Mira	Theta	Aurora
Peak Performance	10 PF	>8.5 PF	180 PF
Compute Nodes	49,152	>2,500	>50,000
Processor	PowerPC A2 1600 MHz	2nd Generation Intel Xeon Phi	3rd Generation Intel Xeon Phi
System Memory	768 TB	>480 TB	>7 PB
File System Capacity	26 PB	10 PB	>150 PB
File System Throughput	300 GB/s	200 GB/s	>1 TB/s
Intel Architecture (x86-64) Compatibility	No	Yes	Yes
Peak Power Consumption	4.8 MW	1.7 MW	>13 MW
GFLOPS/watt	2.1	>5	>13

https://www.alcf.anl.gov/files/alcfscibro2015.pdf

Current Large-Scale Scientific Computing





https://www.alcf.anl.gov/files/alcfscibro2015.pdf

ASCR Computing Upgrades At a Glance

System attributes	NERSC Now	OLCF Now	ALCF Now	NERSC Upgrade	OLCF Upgrade	ALCF Upgrades	
Name Planned Installation	**Edison**	**TITAN**	**MIRA**	**Cori** **2016**	**Summit** **2017-2018**	**Theta** **2016**	**Aurora** **2018-2019**
System peak (PF)	2.6	27	10	> 30	**200**	>8.5	180
Peak Power (MW)	2	9	4.8	< 3.7	**13.3**	1.7	13
Total system memory	357 TB	710TB	768TB	~1 PB DDR4 + High Bandwidth Memory (HBM)+1.5PB persistent memory	**> 2.4 PB DDR4 + HBM + 3.7 PB persistent memory**	>480 TB DDR4 + High Bandwidth Memory (HBM)	> 7 PB High Bandwidth On-Package Memory Local Memory and Persistent Memory
Node performance (TF)	0.460	1.452	0.204	> 3	**> 40**	> 3	> 17 times Mira
Node processors	Intel Ivy Bridge	AMD Opteron Nvidia Kepler	64-bit PowerPC A2	Intel Knights Landing many core CPUs Intel Haswell CPU in data partition	**Multiple IBM Power9 CPUs & multiple Nvidia Voltas GPUS**	Intel Knights Landing Xeon Phi many core CPUs	Knights Hill Xeon Phi many core CPUs
System size (nodes)	5,600 nodes	18,688 nodes	49,152	9,300 nodes 1,900 nodes in data partition	**~4,600 nodes**	>2,500 nodes	>50,000 nodes
System Interconnect	Aries	Gemini	5D Torus	Aries	**Dual Rail EDR-IB**	Aries	2nd Generation Intel Omni-Path Architecture
File System	7.6 PB 168 GB/s, Lustre®	32 PB 1 TB/s, Lustre®	26 PB 300 GB/s GPFS™	28 PB 744 GB/s Lustre®	**120 PB** **1 TB/s** **GPFS™**	10PB, 210 GB/s Lustre initial	150 PB 1 TB/s Lustre®

http://science.energy.gov/~/media/ascr/ascac/pdf/meetings/201604/2016-0404-ascac-01.pdf

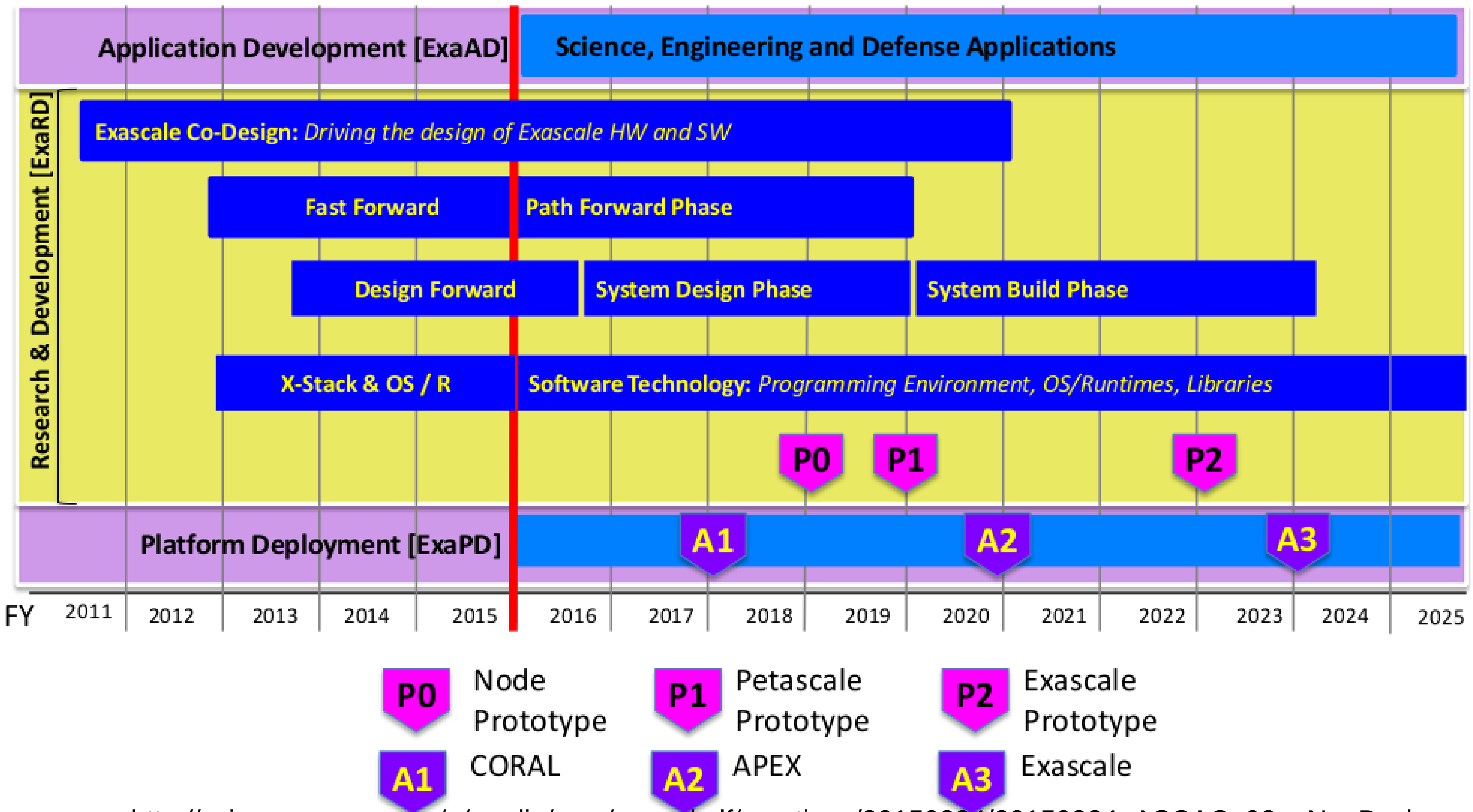
CORAL Node/Rack Layout – ORNL Summit Computer

CORAL rack layout
- 18 nodes
- **779 TF**
- 11 TB RAM
- **55 KW**

CORAL System
- ~200 racks

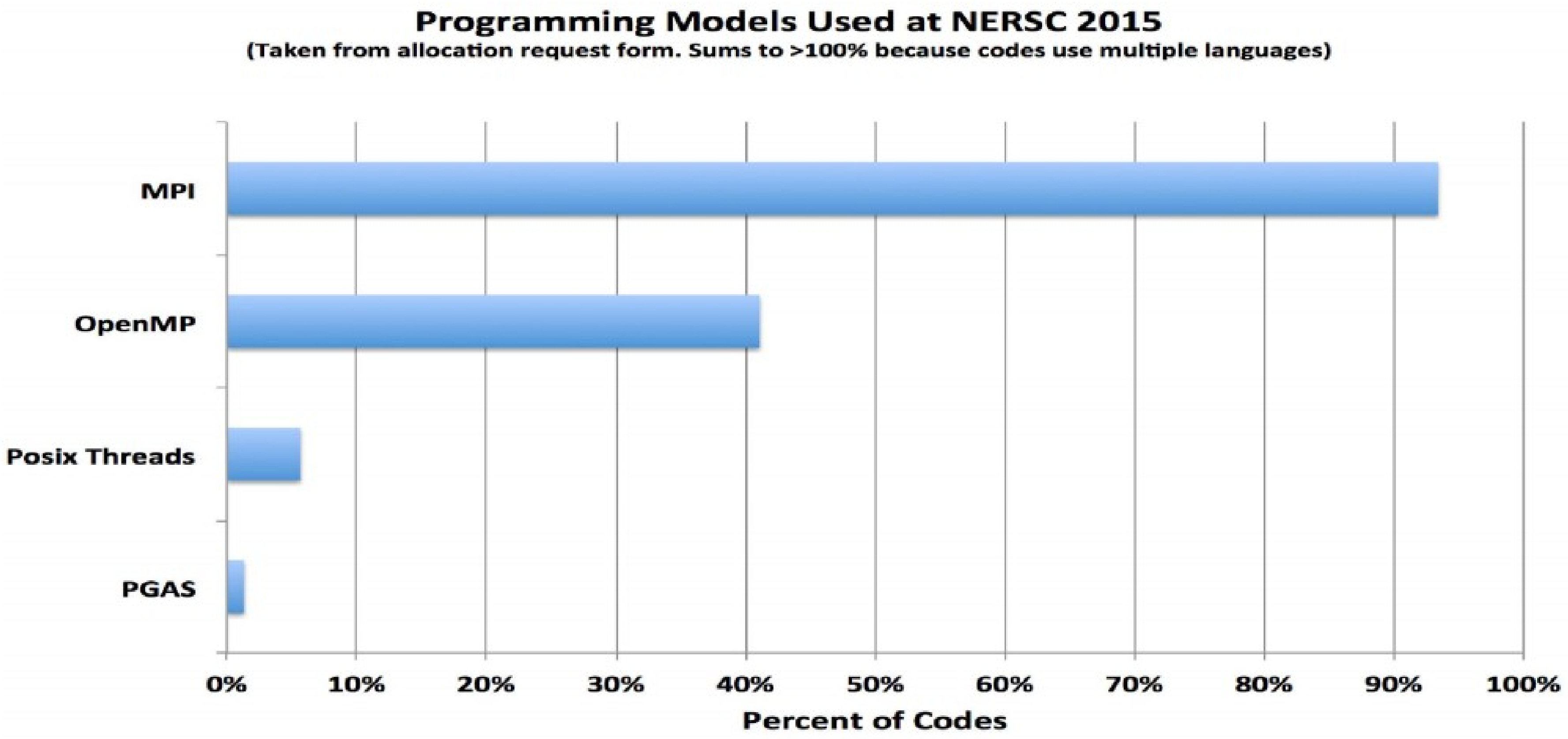
http://science.energy.gov/~/media/ascr/ascac/pdf/meetings/20150324/20150324_ASCAC_02a_No_Backups.pdf

Exascale Computing Initiative (ECI) Timeline



http://science.energy.gov/~/media/ascr/ascac/pdf/meetings/20150324/20150324_ASCAC_02a_No_Backups.pdf

How do we express parallelism?

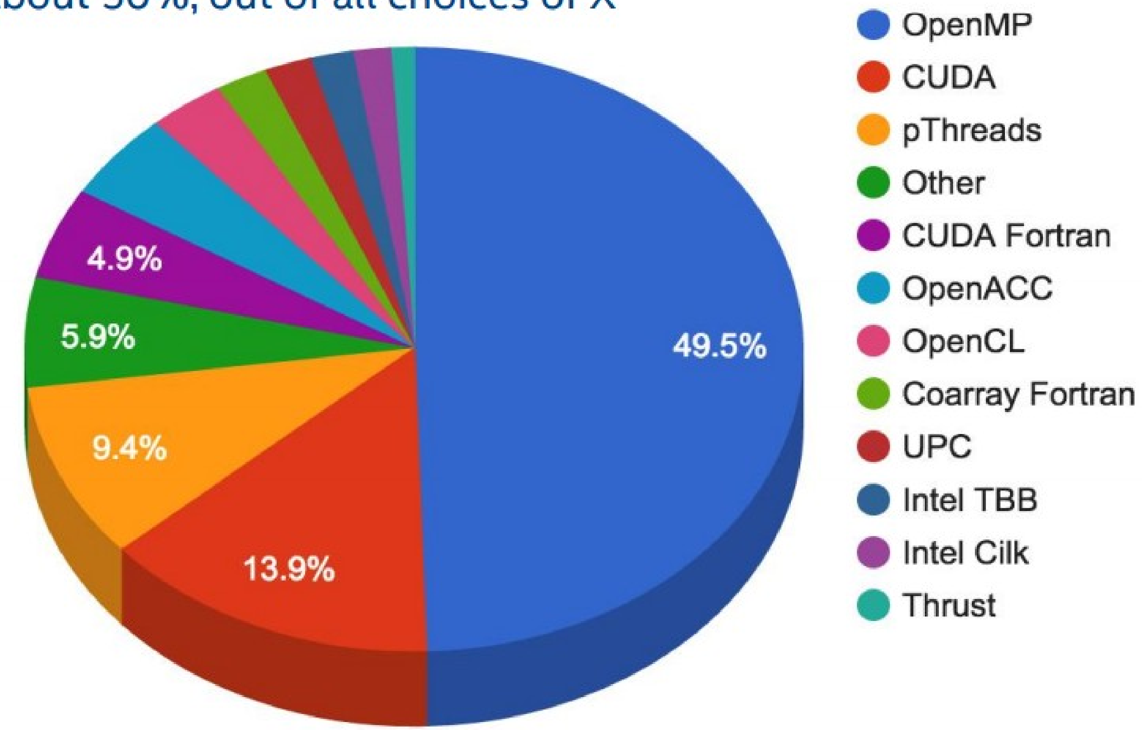


Courtesy of Yun (Helen) He, Alice Koniges, et. al., (NERSC) at OpenMPCon'2015

http://llvm-hpc2-workshop.github.io/slides/Tian.pdf

How do we express parallelism - MPI+X?

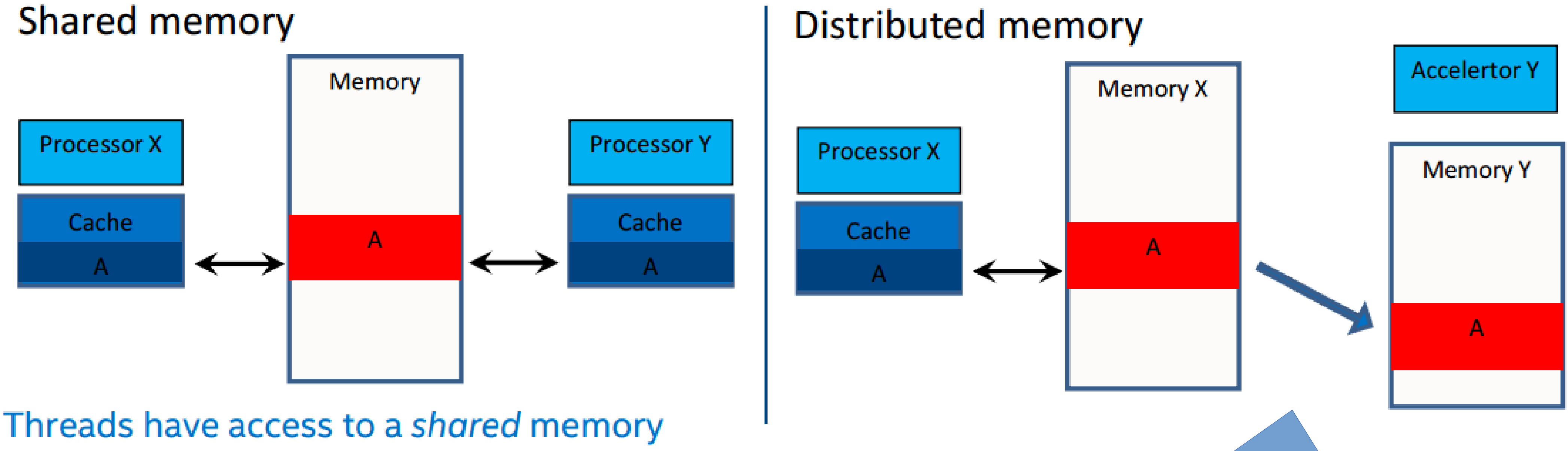
✓ OpenMP is about 50%, out of all choices of X



Courtesy of Yun (Helen) He, Alice Koniges, et. al., (NERSC) at OpenMPCon'2015

http://llvm-hpc2-workshop.github.io/slides/Tian.pdf

OpenMP Evolving Toward Accelerators



http://llvm-hpc2-workshop.github.io/slides/Tian.pdf

New in OpenMP 4

OpenMP Accelerator Support – An Example (SAXPY)

```
int main(int argc, const char* argv[]) {
  float *x = (float*) malloc(n * sizeof(float));
  float *y = (float*) malloc(n * sizeof(float));
  // Define scalars n, a, b & initialize x, y









  for (int i = 0; i < n; ++i){
        y[i] = a*x[i] + y[i];
  }

  free(x); free(y); return 0;
}
```

http://llvm-hpc2-workshop.github.io/slides/Wong.pdf

OpenMP Accelerator Support – An Example (SAXPY)

```
int main(int argc, const char* argv[]) {
  float *x = (float*) malloc(n * sizeof(float));
  float *y = (float*) malloc(n * sizeof(float));
  // Define scalars n, a, b & initialize x, y

#pragma omp target data map(to:x[0:n])
{
#pragma omp target map(tofrom:y)
#pragma omp teams num_teams(num_blocks) num_threads(bsize)
```


```
#pragma omp distribute
  for (int i = 0; i < n; i += num_blocks){
```


```
#pragma omp parallel for
     for (int j = i; j < i + num_blocks; j++)
```


```
       y[j] = a*x[j] + y[j];
  } }
} free(x); free(y); return 0; }
```

Memory transfer if necessary.

Traditional CPU-targeted OpenMP might only need this directive!

HPC-relevant Parallelism is Coming to C++17!

Almost as concise as OpenMP, but in many ways more powerful!

```
using namespace std::execution::parallel;
int a[] = {0,1};
for_each(par, std::begin(a), std::end(a), [&](int i) {
  do_something(i);
});
```

http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2014/n4071.htm

```
void f(float* a, float*b) {
    ...
    for_each(par_unseq, begin, end, [&](int i)
    {
        a[i] = b[i] + c;
    });
}
```

The “par_unseq” execution policy allows for vectorization as well.

HPC-relevant Parallelism is Coming to C++17!

Table 1 — Table of parallel algorithms

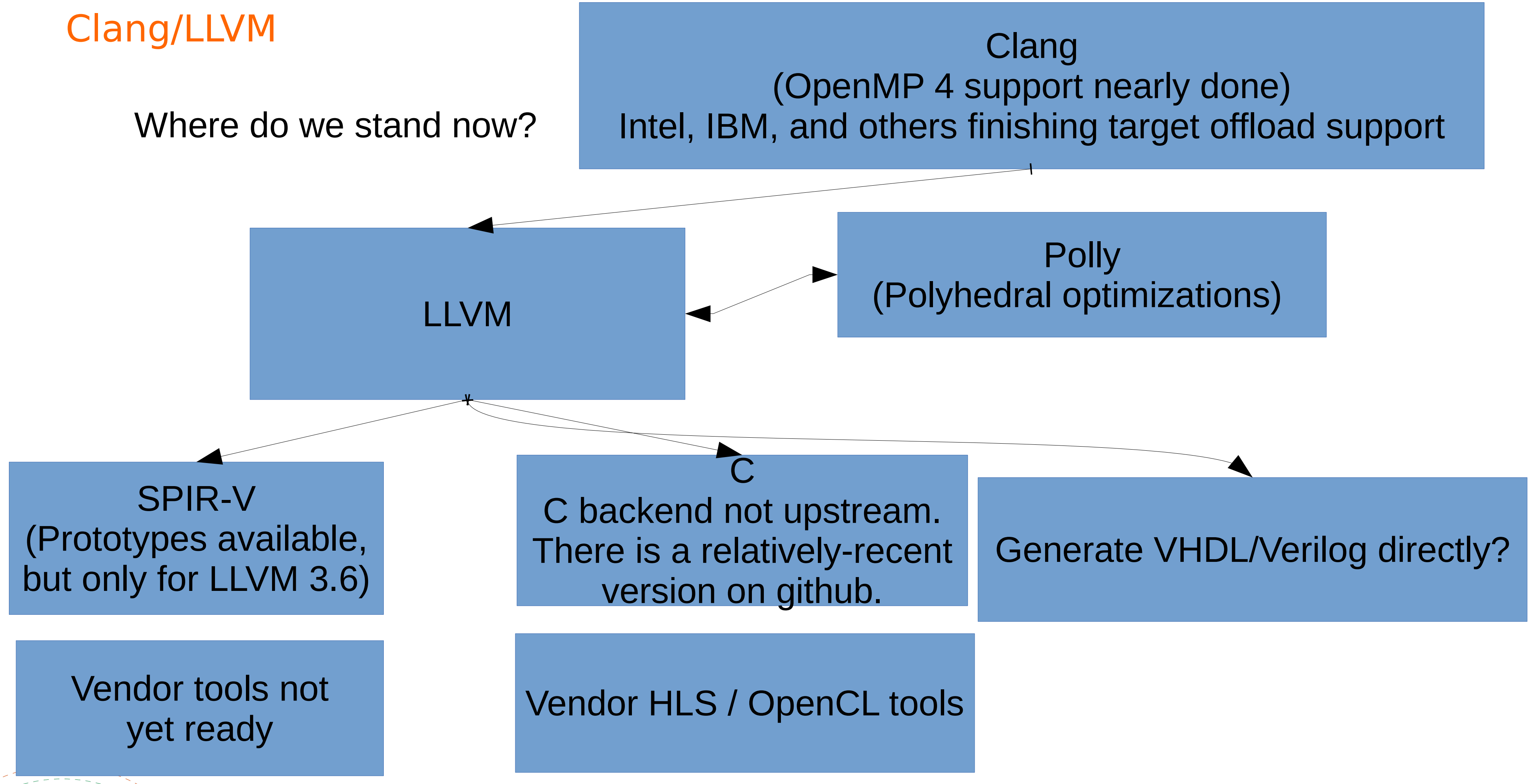
adjacent_difference	adjacent_find	all_of	any_of
copy	copy_if	copy_n	count
count_if	equal	exclusive_scan	fill
fill_n	find	find_end	find_first_of
find_if	find_if_not	for_each	for_each_n
generate	generate_n	includes	inclusive_scan
inner_product	inplace_merge	is_heap	is_heap_until
is_partitioned	is_sorted	is_sorted_until	lexicographical_compare
max_element	merge	min_element	minmax_element
mismatch	move	none_of	nth_element
partial_sort	partial_sort_copy	partition	partition_copy
reduce	remove	remove_copy	remove_copy_if
remove_if	replace	replace_copy	replace_copy_if
replace_if	reverse	reverse_copy	rotate
rotate_copy	search	search_n	set_difference
set_intersection	set_symmetric_difference	set_union	sort
stable_partition	stable_sort	swap_ranges	transform
uninitialized_copy	uninitialized_copy_n	uninitialized_fill	uninitialized_fill_n
unique	unique_copy		

[*Note:* Not all algorithms in the Standard Library have counterparts in Table 1. — *end note*]

http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2014/n4071.htm

Clang/LLVM

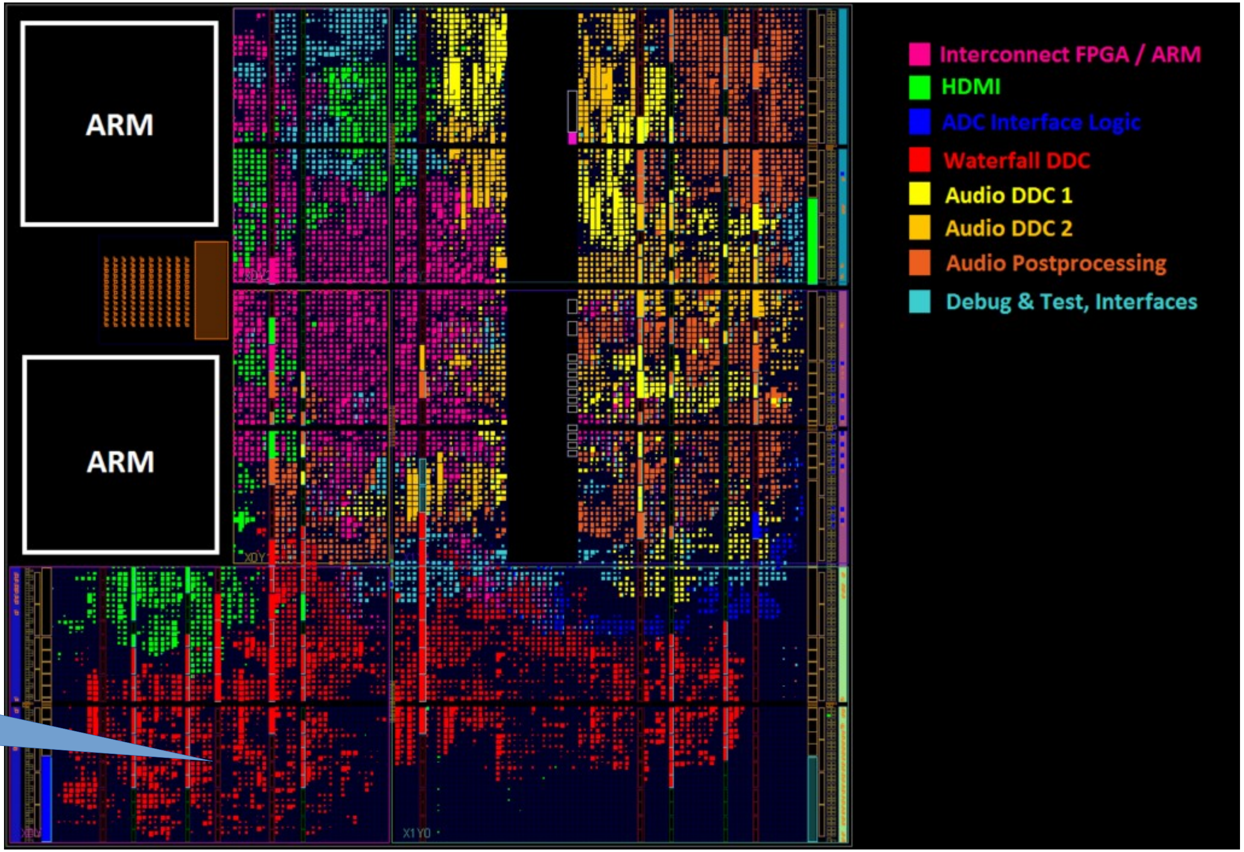
Where do we stand now?



Current FPGA + CPU System

Xilinx Zynq 7020 has two ARM Cortex A9 cores.

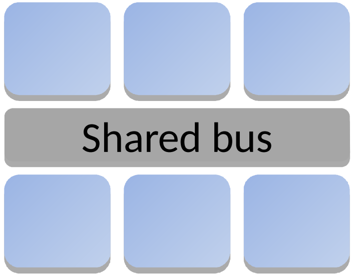
53,200 LUTs
560 KB SRAM
220 DSP slices



http://www.panoradio-sdr.de/sdr-implementation/fpga-software-design/

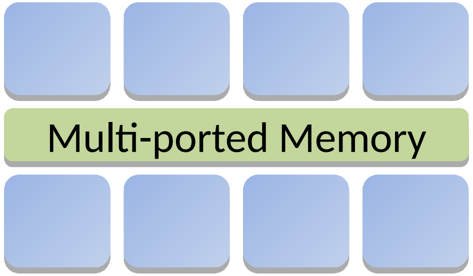
Interconnect Energy

Buses over short distance



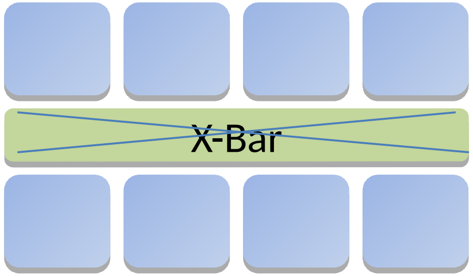
1 to 10 fJ/bit
0 to 5mm
Limited scalability

Shared memory



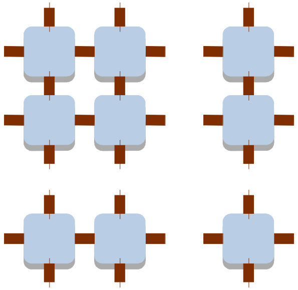
10 to 100 fJ/bit
1 to 5mm
Limited scalability

Cross Bar Switch



0.1 to 1pJ/bit
2 to 10mm
Moderate scalability

Packet Switched Network

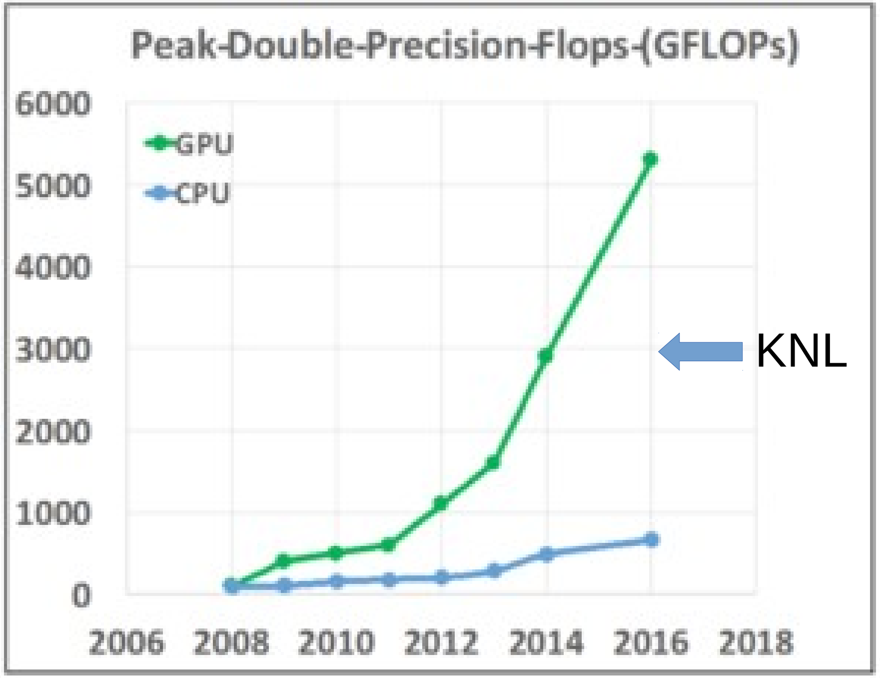


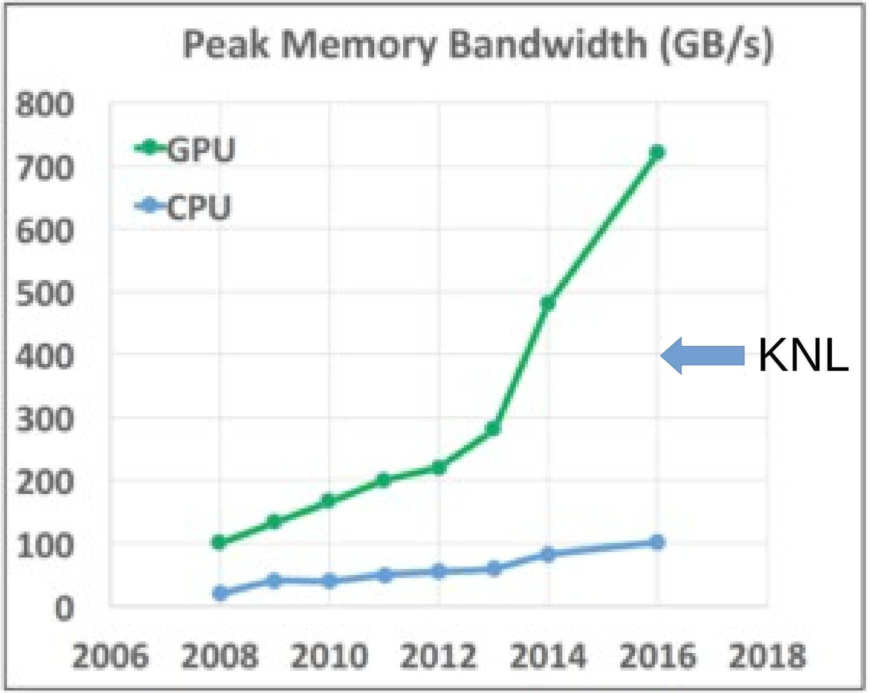
1 to 3pJ/bit
>5 mm, scalable

Interconnect Structures

http://www.socforhpc.org/wp-content/uploads/2015/06/SBorkar-SoC-WS-DAC-June-7-2015-v1.pptx

CPU and GPU Trends





https://www.hpcwire.com/2016/08/23/2016-important-year-hpc-two-decades/

CPU vs. FGPA Efficiency

CPU and FPGA achieve maximum algorithmic efficiency at polar opposite sides of the parameter space!

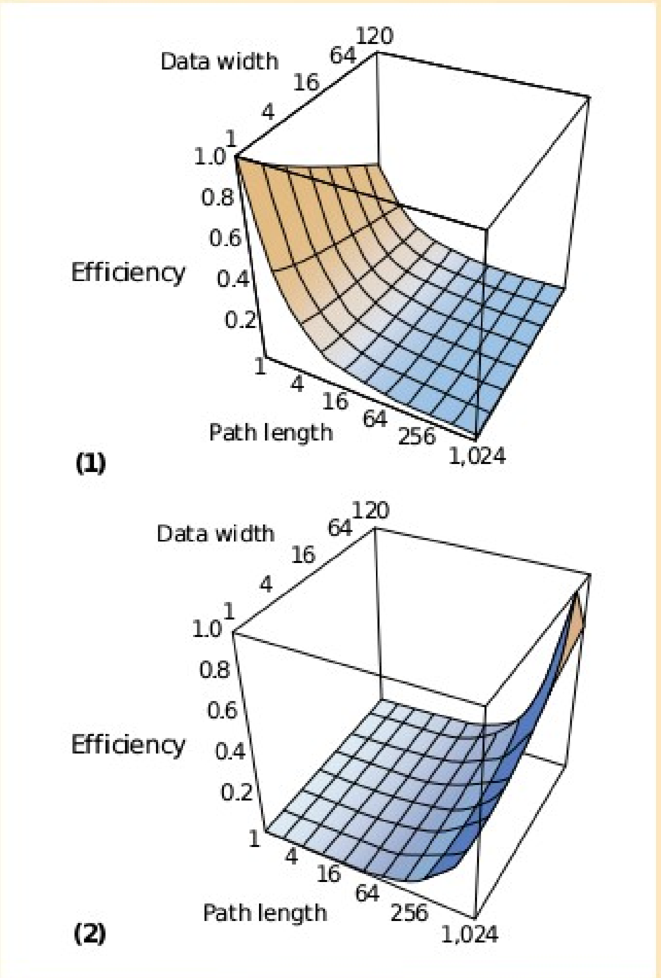


Figure C. Design efficiency at varying application data widths and path lengths of (1) an FPGA and (2) a processor.

http://authors.library.caltech.edu/1629/1/DEHcomputer00.pdf