Computer Software:
The 'Trojan Horse' of HPC

Steve Wallach

Convey <= Convex++

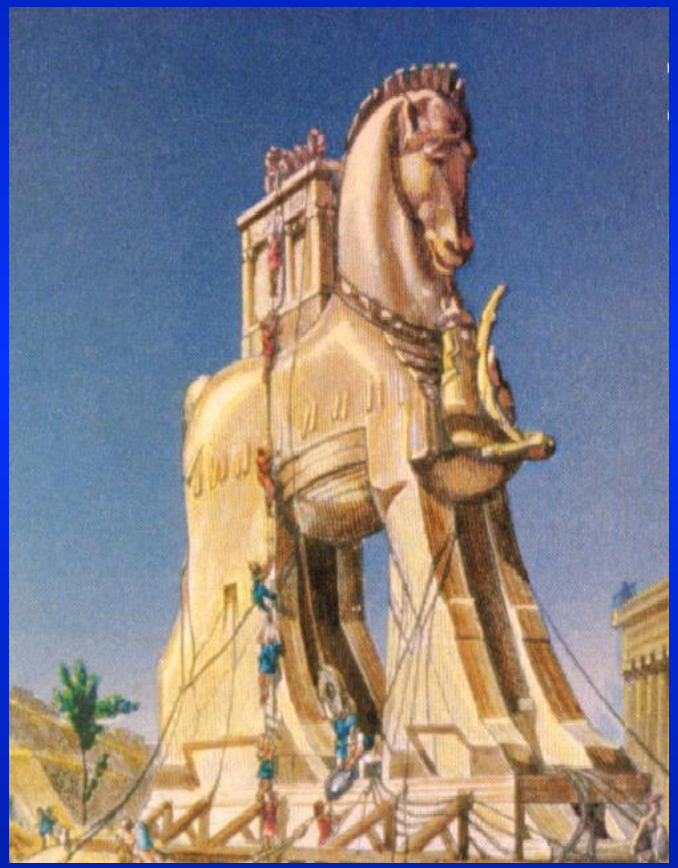
swallach”at”conveycomputer “dot”com

Discussion

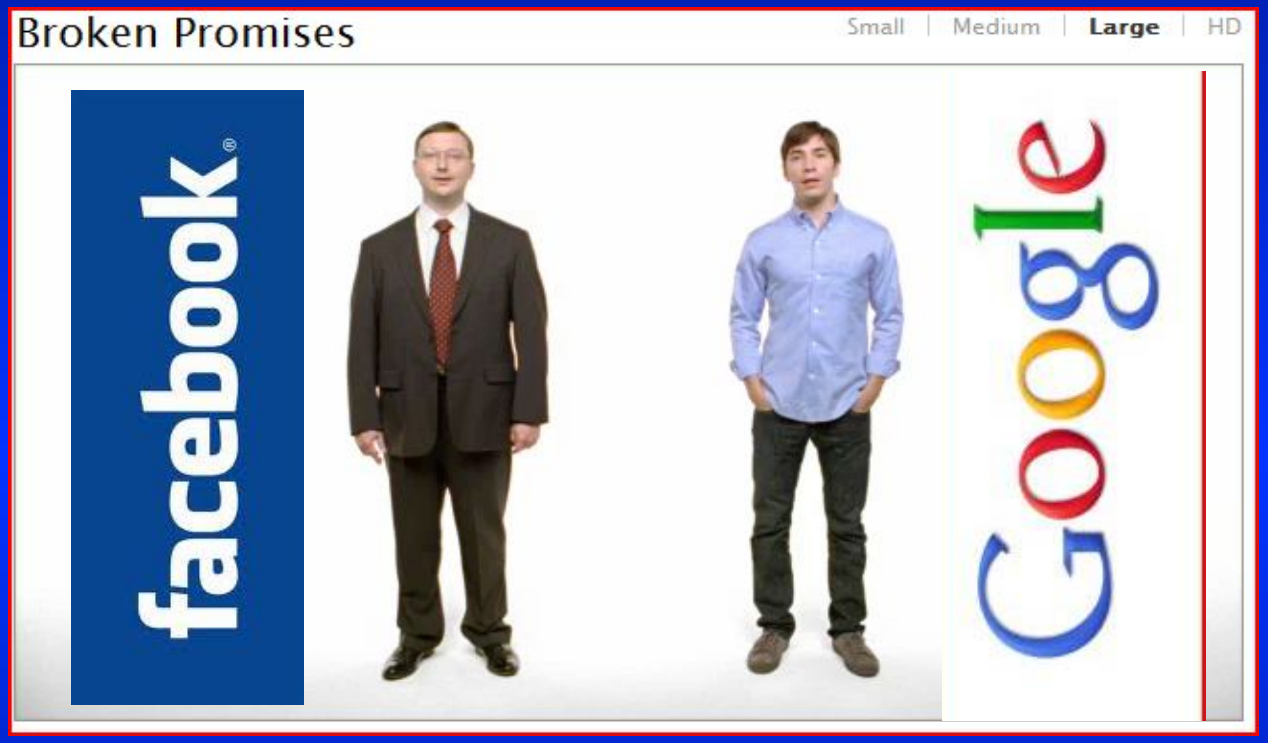
- The state of ***Languages and Compilers for Parallel Computing***
- Searching for the SOFTRON
- What Convey Computer is doing
- The path to Exascale Computing

Mythology

- In the old days, we were told "Beware of Greeks bearing gifts"



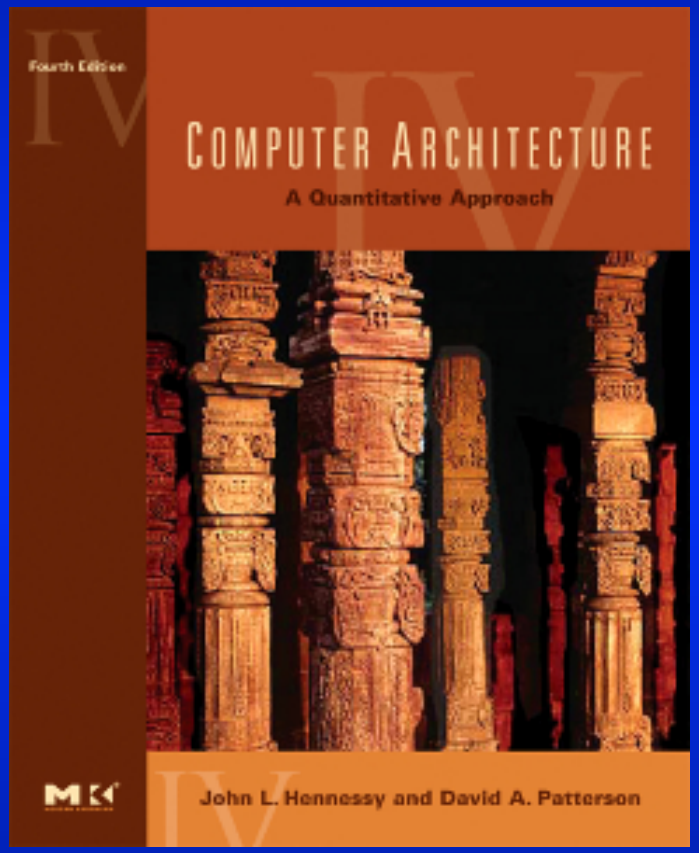
Today



- "Beware of Geeks bearing Gifts"

What problems are we solving

- New Hardware Paradigms
- Uniprocessor Performance leveling
- MPP and multi-threading for the masses

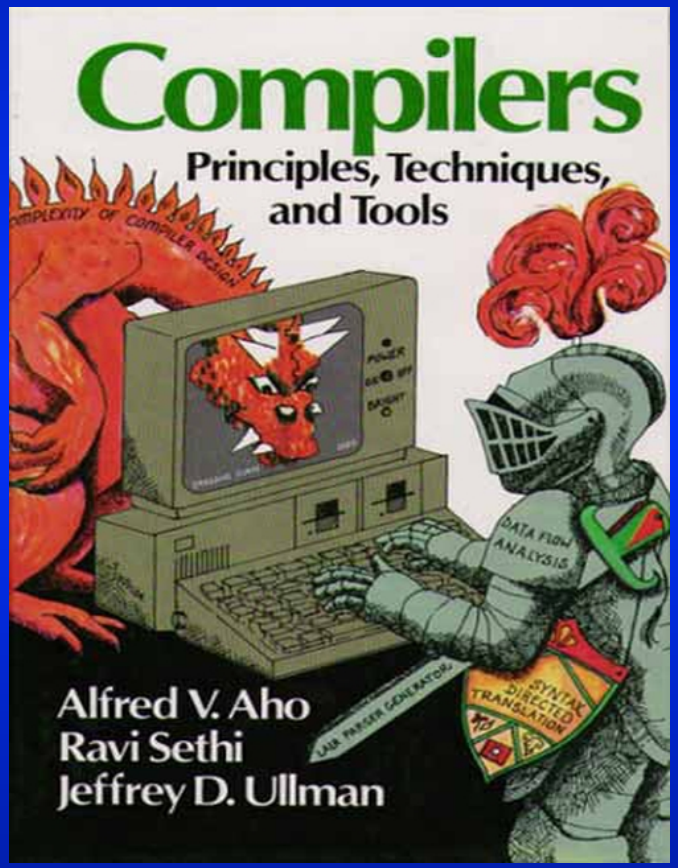


Deja Vu

- Multi-Core Evolves
  - Many Core
  - ILP fizzles
- x86 extended with sse2, sse3, and sse4
  - application specific enhancements
  - Co-processor within x86 micro-architecture
- Basically performance enhancements by
  - On chip parallel
  - Instructions for specific application acceleration
    - One application instruction replaces MANY generic instructions
- Déjà vu – all over again – 1980's
  - Need more performance than micro
  - GPU, CELL, and FPGA's
    - Different software environment
- Heterogeneous Computing AGAIN

Current Languages

- Fortran 66 ➔ Fortran 77 ➔ Fortran 95 ➔ 2003
  - HPC Fortran
  - Co-Array Fortran
- C ➔ C++
  - UPC
  - Stream C
  - C# (Microsoft)
  - Ct (Intel)



Another Bump in the Road

- GPGPU's are very cost effective for many applications.
- Matrix Multiply
  - Fortran

```
do i = 1,n1
do k = 1,n3
c(i,k) = 0.0
do j = 1,n2
c(i,k) = c(i,k) + a(i,j) * b(j,k)
Enddo
Enddo
Enddo
```

PGI Fortran to CUDA

Simplified Matrix Multiplication in CUDA, Using Tiled Algorithm

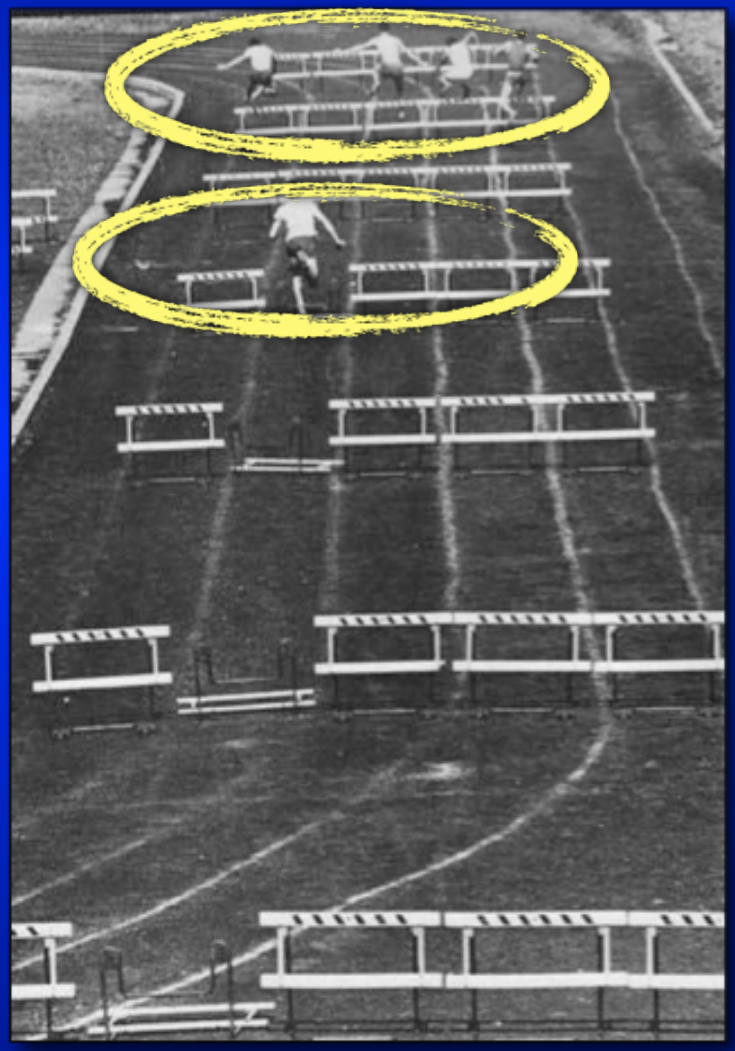
```
__global__ void
  matmulKernel( float* C, float* A, float* B, int N2, int N3 )
{  int bx = blockIdx.x,  by = blockIdx.y;
 int tx = threadIdx.x, ty = threadIdx.y;
int aFirst = 16 * by * N2;
 int bFirst = 16 * bx;
float Csub = 0;
              for( int j = 0; j < N2; j += 16 ) {    __shared__ float Atile[16][16], Btile[16][16];
              Atile[ty][tx] = A[aFirst + j + N2 * ty + tx];
              Btile[ty][tx] = B[bFirst + j*N3 + b + N3 * ty + tx];    __syncthreads();
               for( int k = 0; k < 16; ++k )
              Csub += Atile[ty][k] * Btile[k][tx];      __syncthreads();  }
              int c = N3 * 16 * by + 16 * bx;
              C[c + N3 * ty + tx] = Csub;}
void matmul( float* A, float* B, float* C,            size_t N1, size_t N2, size_t N3
{  void *devA, *devB, *devC;  cudaSetDevice(0);
cudaMalloc( &devA, N1*N2*sizeof(float) );
  cudaMalloc( &devB, N2*N3*sizeof(float) );
cudaMalloc( &devC, N1*N3*sizeof(float) );
 cudaMemcpy( devA, A, N1*N2*sizeof(float),
cudaMemcpyHostToDevice );
cudaMemcpy( devB, B, N2*N3*sizeof(float),
cudaMemcpyHostToDevice );
 dim3 threads( 16, 16 );
 dim3 grid( N1 / threads.x, N3 / threads.y);
 matmulKernel<<< grid, threads >>>( devC, devA, devB, N2, N3 );
cudaMemcpy( C, devC, N1*N3*sizeof(float),
 cudaMemcpyDeviceToHost );
 cudaFree( devA );
 cudaFree( devB );
cudaFree( devC );
```

Pornographic Programming:
Can't define it, but you know
When you see it.

http://www.linuxjournal.com/article/10216
Michael Wolfe – Portland Group
How We Should Program GPGPUs November 1st, 2008

Programmer

~~Find the~~ Accelerator

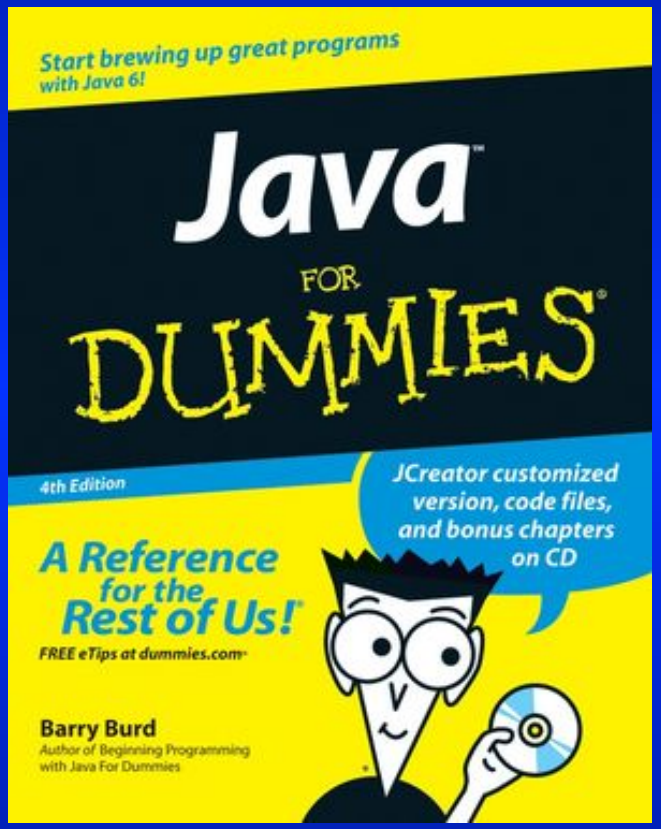


- Accelerators can be beneficial. It isn't "free" (like waiting for the next clock speed boost)
  - Worst case - you will have to completely rethink your algorithms and/or data structures
  - Performance tuning is still time consuming
  - Don't forget our long history of parallel computing...

Position Jun 2010	Position Jun 2009	Delta in Position	Programming Language	Ratings Jun 2010	Delta Jun 2009	Statu
1	1	=	Java	18.033%	-2.11%	A
2	2	=	C	17.809%	+1.03%	A
3	3	=	C++	10.757%	+0.16%	A
4	4	=	PHP	8.934%	-0.74%	A
5	5	=	(Visual) Basic	5.868%	-2.07%	A
6	7	↑	C#	5.196%	+0.66%	A
7	6	↓	Python	4.266%	-0.49%	A
8	9	↑	Perl	3.200%	-0.71%	A
9	45	↑↑↑↑↑↑↑↑↑	Objective-C	2.469%	+2.35%	A
10	11	↑	Delphi	2.394%	+0.21%	A
11	8	↓↓↓	JavaScript	2.191%	-1.83%	A
12	10	↓↓	Ruby	2.070%	-0.56%	A
13	12	↓	PL/SQL	0.787%	-0.09%	A
14	14	=	SAS	0.703%	-0.06%	A
15	15	=	Pascal	0.702%	-0.06%	A-
16	18	↑↑	Lisp/Scheme/Clojure	0.654%	+0.05%	B
17	19	↑↑	Lua	0.592%	+0.04%	B
18	20	↑↑	MATLAB	0.589%	+0.06%	B
19	16	↓↓↓	ABAP	0.577%	-0.15%	B
20	27	↑↑↑↑↑↑	PowerShell	0.529%	+0.23%	B
21			Go	0.519%		
22			ActionScript	0.501%		
23			Transact-SQL	0.486%		
24			RPG (OS/400)	0.443%		
25			Bourne shell	0.426%		
26			Ada	0.416%		
27			D	0.398%		
28			JavaFX Script	0.393%		
29			FoxPro/xBase	0.388%		
30			COBOL	0.380%		
31			Fortran	0.376%		
32			Haskell	0.365%		
33			S-lang	0.347%		
34			Alice	0.340%		
35			LabVIEW	0.333%		
36			Logo	0.330%		
37			Scratch	0.329%		
38			Tcl/Tk	0.321%		
39			J	0.305%		
40			NXT-G	0.301%		
41			Forth	0.294%		
42			Prolog	0.254%		
43			Scala	0.253%		
44			Groovy	0.251%		

Interpretive Languages

- User cycles are more important than computer cycles
  - We need ISA's that model and directly support these interpretive languages and applications
  - FPGA's cores to support extensions to the standard ISA
- Language Directed Design
  - *ACM-IEEE Symp. on High-Level-Language Computer Architecture, 7-8 November 1973, College Park, MD*
  - FAP – Fortran Assembly Program (IBM 7090)
  - B1700 – S-languages (Burroughs)
    - Architecture defined by compiler writers



Cautionary Tale: A Brief History of Languages

- **When vector machines were king**
  - Parallel "languages" were loop annotations (IVDEP)
  - Performance was fragile, but there was good user support

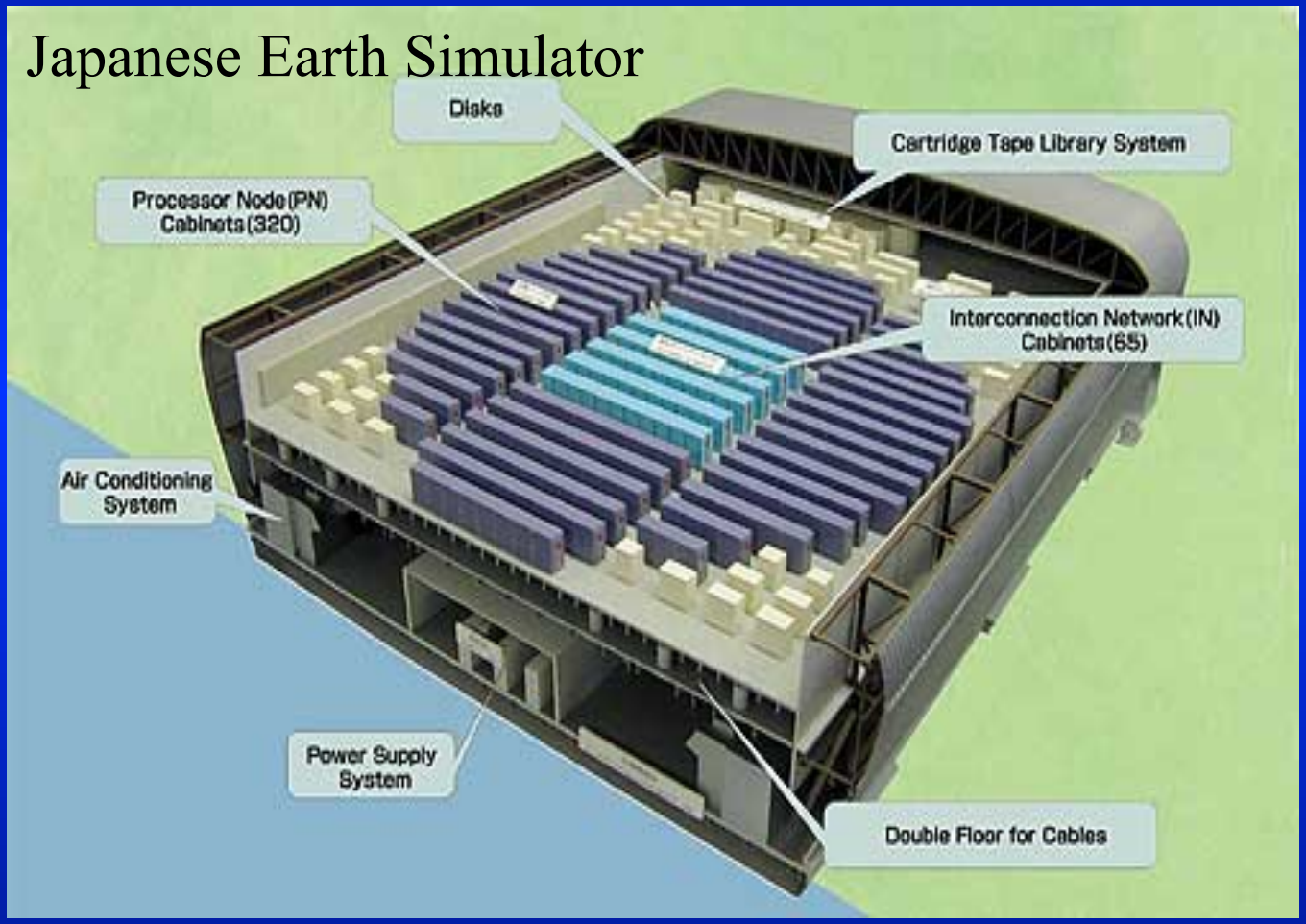
Kathy Yelick, 2008 Keynote, Salishan Conference

Hardware Designers

- We follow Moore's Law
- We continue to development new architectures and topologies
- Software is left to those other "guys"
- Ken Kennedy
  - "Architects have it great --- they build new and complex machines with high peak performances and then get to blame the compiler writers when performance does not meet expectations!"
- To further make life difficult the physicists have created:

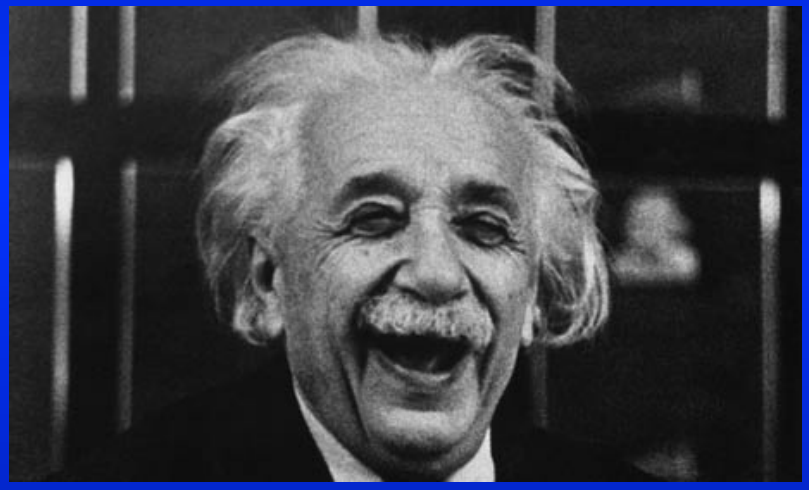
The C2K problem

C2K Problem


Japanese Earth Simulator
Disks
Cartridge Tape Library System
Processor Node (PN) Cabinets (320)
Interconnection Network (IN) Cabinets (65)
Air Conditioning System
Power Supply System
Double Floor for Cables

C2K Problem

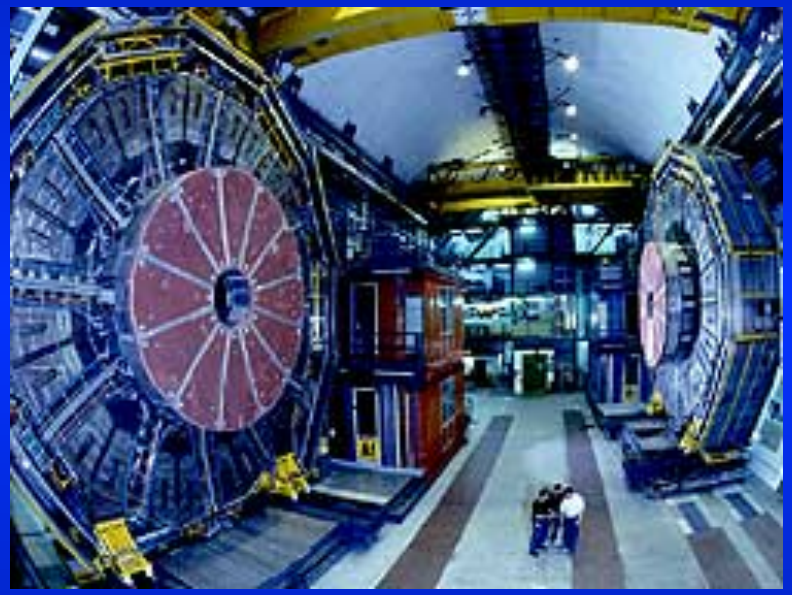
- 40 Meter Radius (80 Meter Round Trip)
- Lower Bound of 400 ns Latency (Hardware - General Case)
- Thus over time, Latency gets proportionally worse
  - Same Problem exists in Processor Clock Vs. Memory Access Time.
- Solutions
  - Better Spatial locality (Bigger SMP's)
  - Spatially/Topologically Systems
    - Remember 250 Watts/Sq. Ft Cooling (air)
    - LANL 40,000 Sq. Ft => 10 Megawatts
  - Better Latency Tolerant Algorithms



SOFTRON Particle Physics

- Hardware has
  - Electrons, photons, etc.
  - Well behaved and understood rules
- Physicists search for the predicted

  "HIGGS BOSON"
- The Computer Scientists must search for the SOFTRON



GeDanken Experiment

- Black Box Experiments. What is the nature of the Softron
  - Norm Augustin
  - Nathan Myrhold
  - Steve Wallach

The Experiment

SOFTWARE

Control-Alt-Delete

Norm Augustine

- Software is like entropy, it weighs nothing, it always increases, and no one understands it“

- PHYSICS: **measure of unavailable energy:** a measure of the energy in a system or process that is unavailable to do work. In a reversible thermodynamic process, entropy is expressed as the heat absorbed or emitted divided by the absolute temperature.
  - **Microsoft® Encarta® Reference Library 2004**

Steve Wallach - Observations

- Half life about 10 years
- Fixing a bug is inversely proportional to the time it takes to find it
- Debugging the first 95% is easy, the second 95% is more difficult
- Optimizing compilers make hardware run faster

Steve Wallach - Observations

- No Mass
- No Charge
  - Actually accumulative over a 5 year period (license and support fees)
- High Energy Research need to be applied to determine its fundamental structure
  - Long Term Basic Research
  - World-Wide Effort

The Next Fifty Years of Software

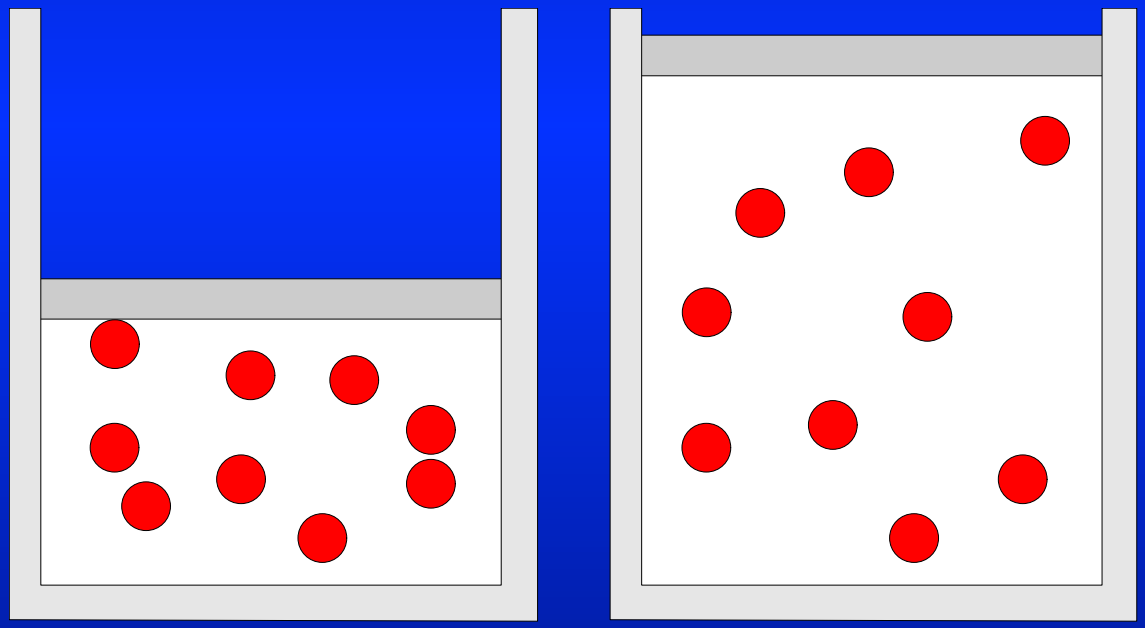
Nathan P. Myhrvold
Chief Technology Officer
Microsoft Corporation

http://research.microsoft.com/acm97/

Nathan's 1st Law of Software

Software is a gas!

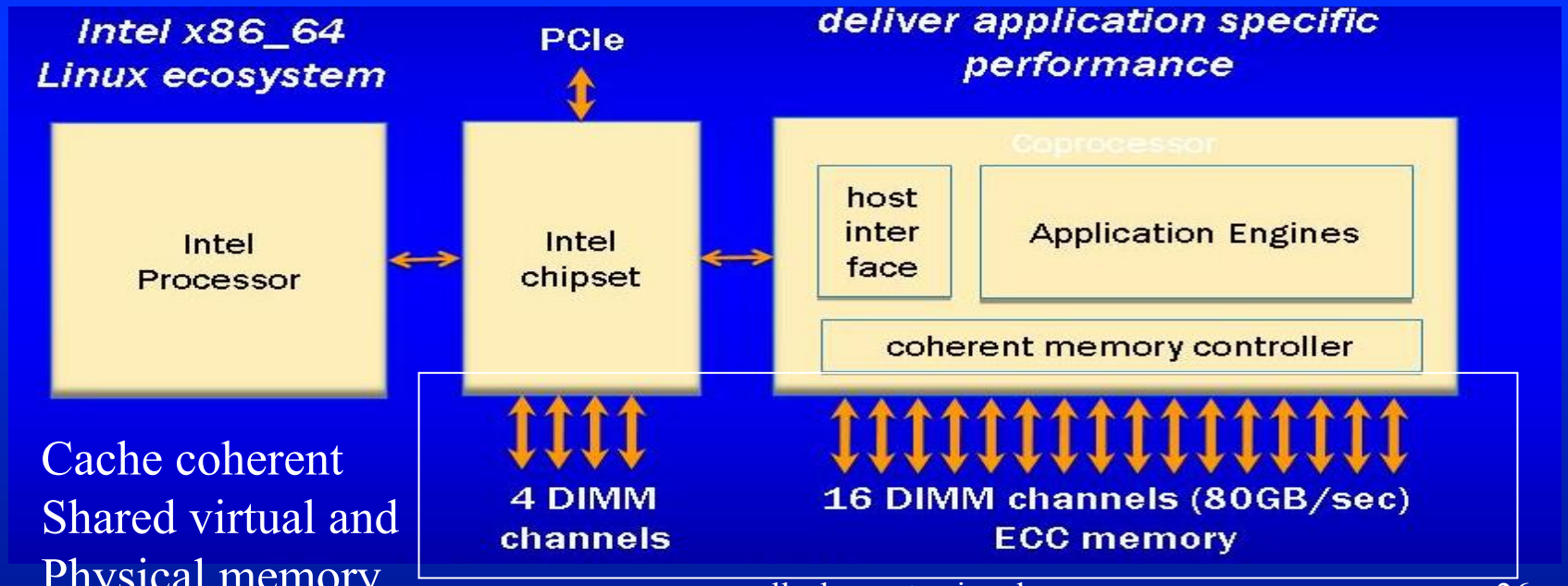
It expands to fit the container it is in!



Summarizing

Convey Product

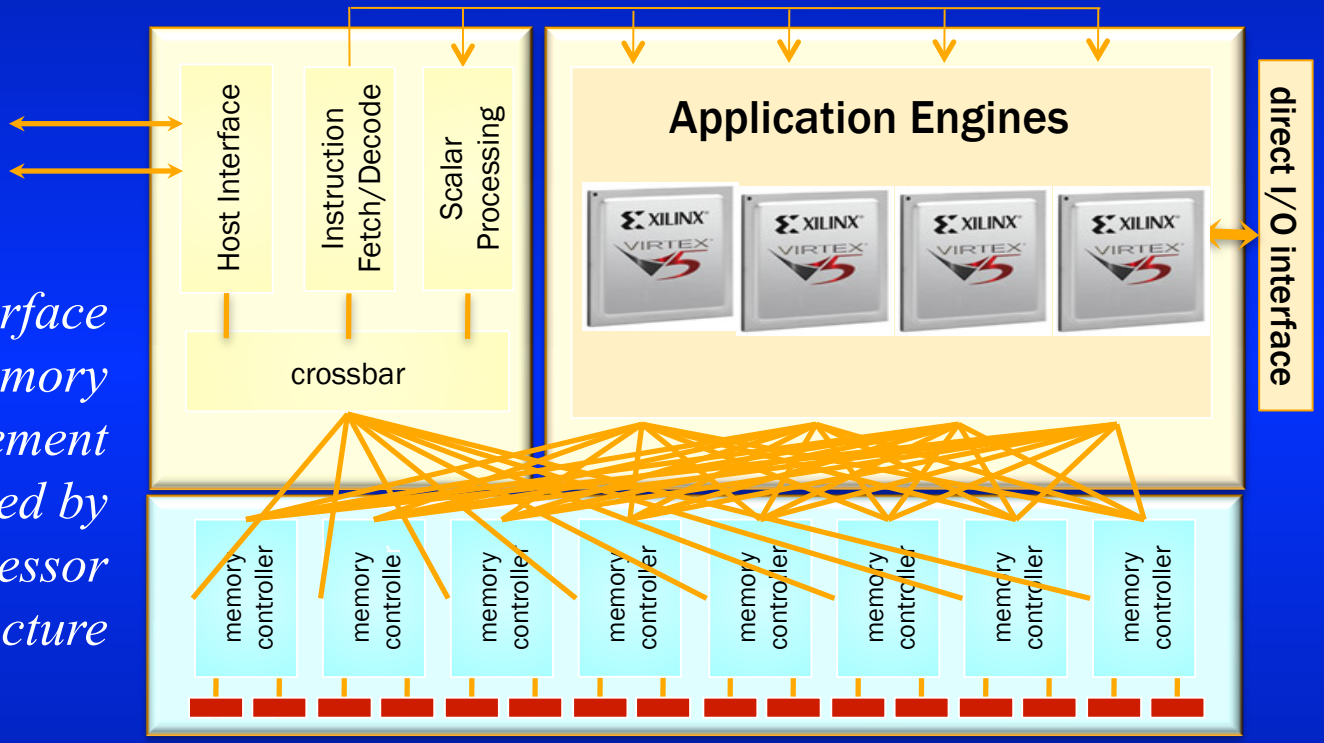
- Reconfigurable Co-Processor to Intel x86-64
- Shared 64_bit Virtual and Physical Memory (cache coherent)
- Coprocessor executes instructions that are viewed as extensions to the x86 ISA
- Convey Developed Compilers: C,C++, & Fortran based on open 64)
  - Automatic Vectorization/Parallelization
    - SIMD Multi-threading
  - Generates both x86 and coprocessor instructions



Cache coherent
Shared virtual and
Physical memory

Inside the Coprocessor

Personalities dynamically loaded into AEs implement application specific instructions



System interface and memory management implemented by coprocessor infrastructure

16 DDR2 memory channels
Standard or Scatter-Gather DIMMs
80GB/sec throughput

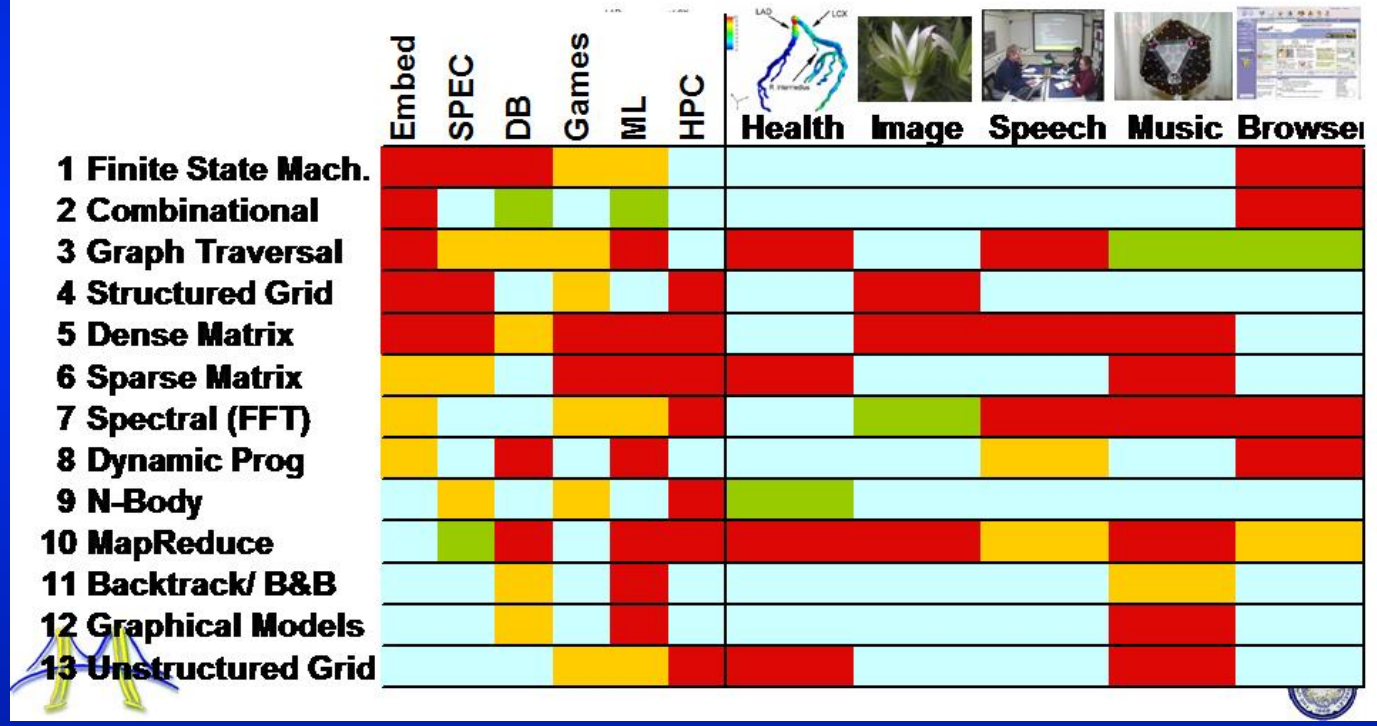
Non-blocking
Virtual output queuing
Round-robin arbitration

Berkeley's 13 Motifs

"Motif/Dwarf" Popularity

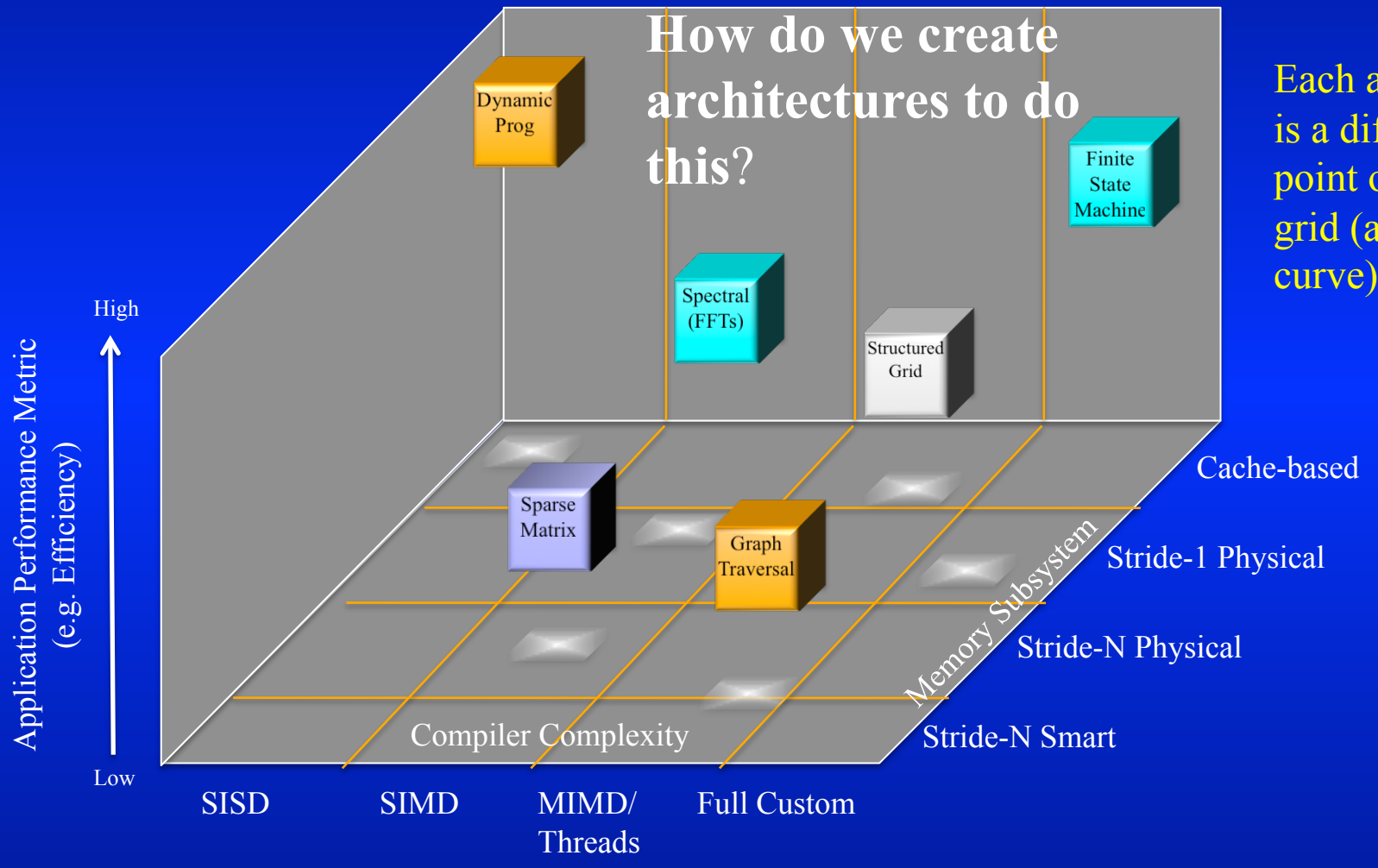
(Red Hot → Blue Cool)

- How do compelling apps relate to 13 motif/dwarfs?



http://www.eecs.berkeley.edu/Pubs/TechRpts/2008/EECS-2008-23.html

What does this all mean?

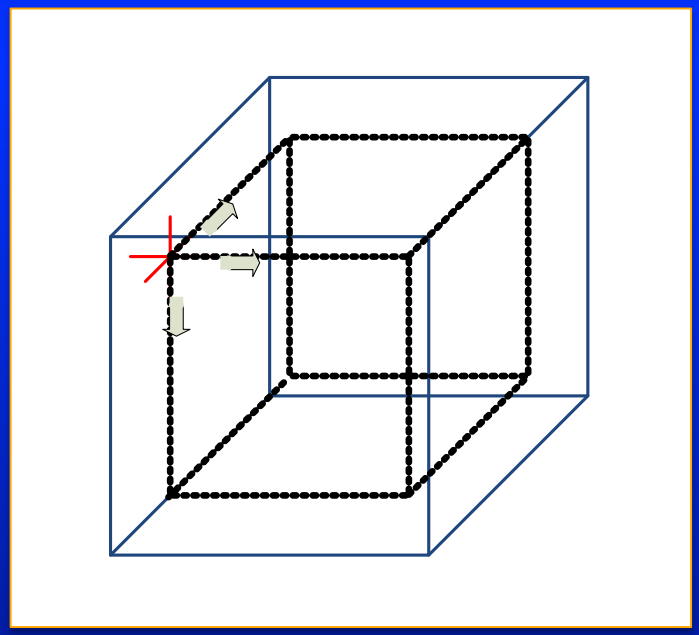


Each application is a different point on this 3D grid (actually a curve)

3D Finite Difference (3DFD) Personality

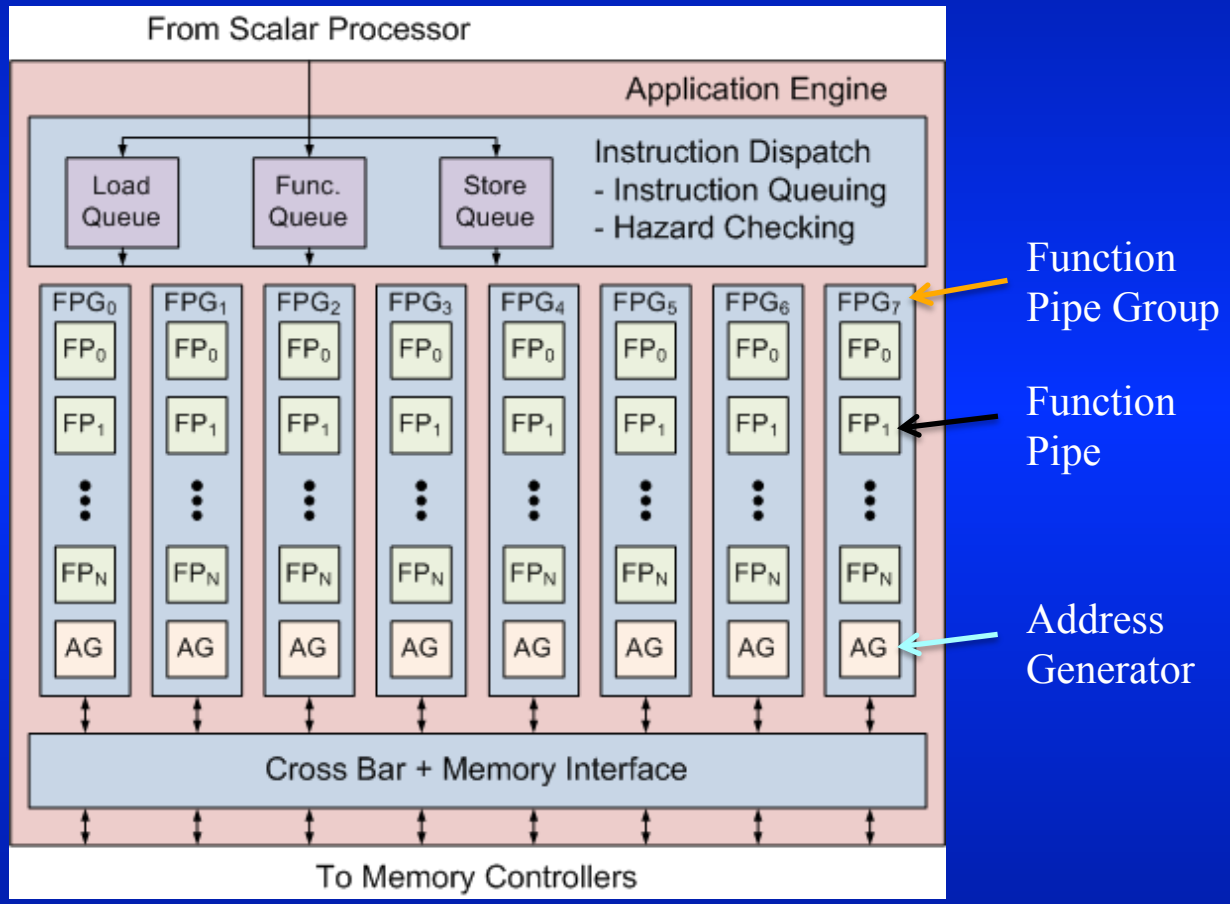
- designed for nearest neighbor operations on structured grids
  - maximizes data reuse
- reconfigurable "registers"
  - 1D (vector), 2D, and 3D modes
  - 8192 elements in a register
- operations on entire cubes
  - "add points to their neighbor to the left times a scalar" is a single instruction
  - up to 7 points away in any direction
- finite difference method for post-stack reverse-time migration

```
X(I,J,K) = S0*Y(I  ,J  ,K  )
         + S1*Y(I-1,J  ,K  )
         + S2*Y(I+1,J  ,K  )
         + S3*Y(I  ,J-1,K  )
         + S4*Y(I  ,J+1,K  )
         + S5*Y(I  ,J  ,K-1)
         + S6*Y(I  ,J  ,K+1)
```



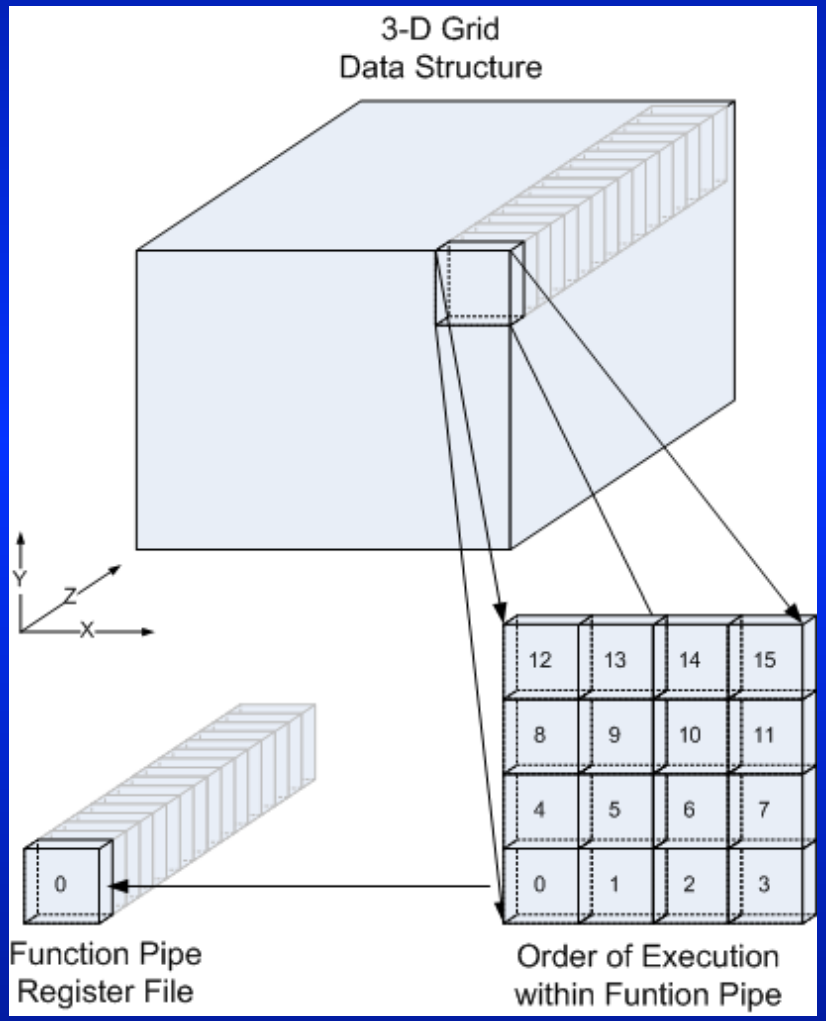
3DFD Personality Framework

- The number of Function Pipe Groups is set to match available memory bandwidth.
- The number of Function Pipes is set based on the resources available.
- 3-D HW grid structure
  - X axis, Number of FPG
  - Y axis, Number of FP per FPG
  - Z axis, Vector Elements within a FP

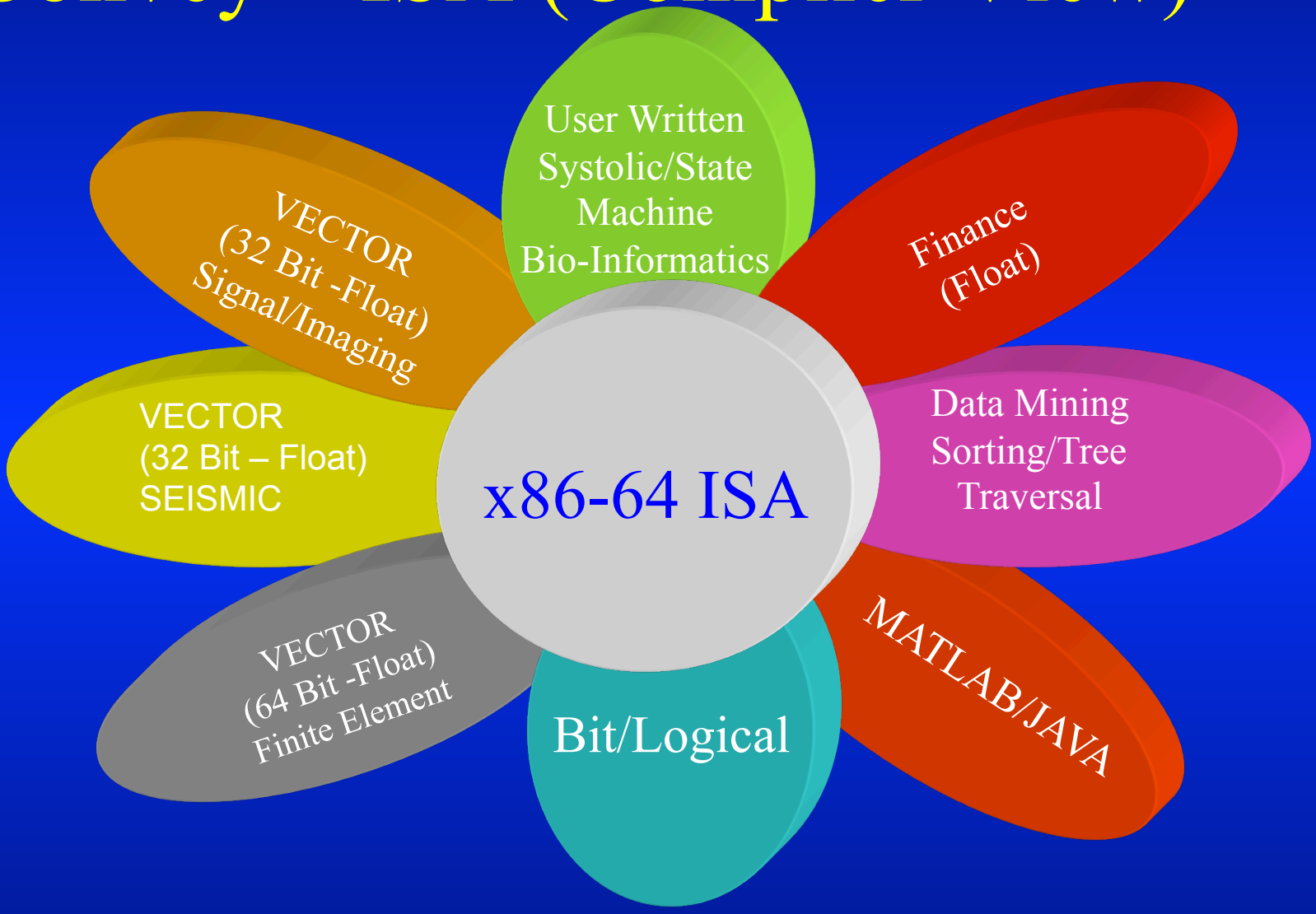


3D Grid Data Storage

- Personality maps 3-D grid data structure to register files within function units to achieve maximum data reuse.
- Dimensions of 3-D data structure can be easily changed.
- A few dozen instructions will perform a 14th order stencil operation.

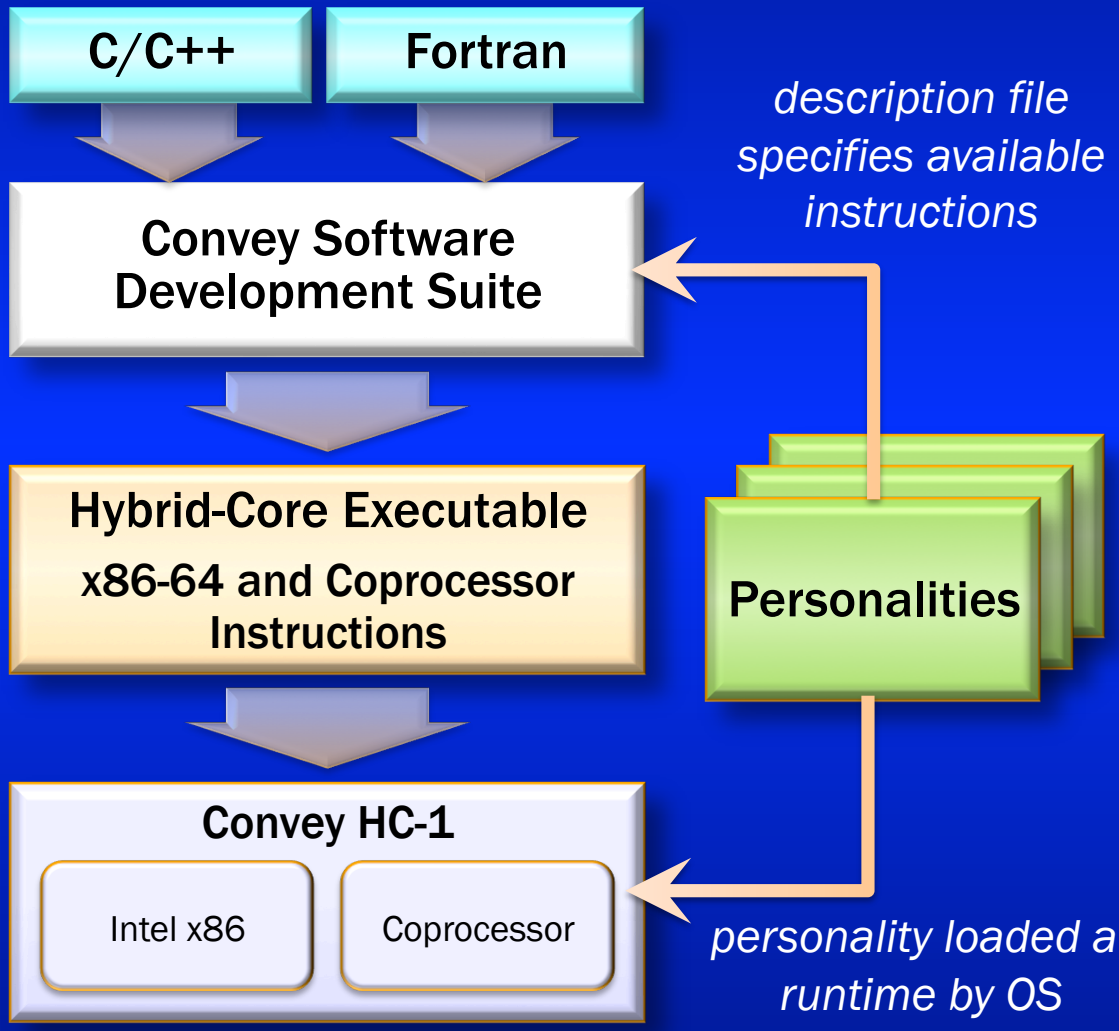


Convey – ISA (Compiler View)



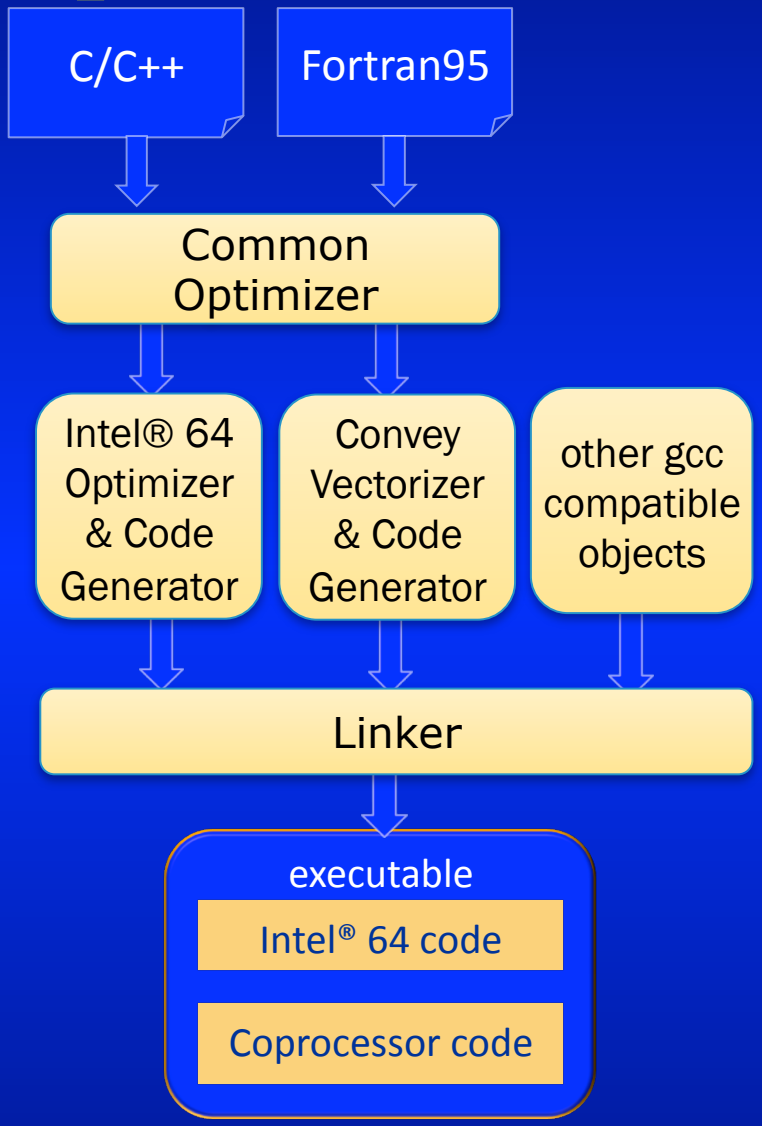
Using Personalities

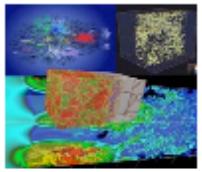
- Program using ANSI standard C/ C++ and Fortran
- User specifies personality at compile time
- OS demand loads personalities at runtime



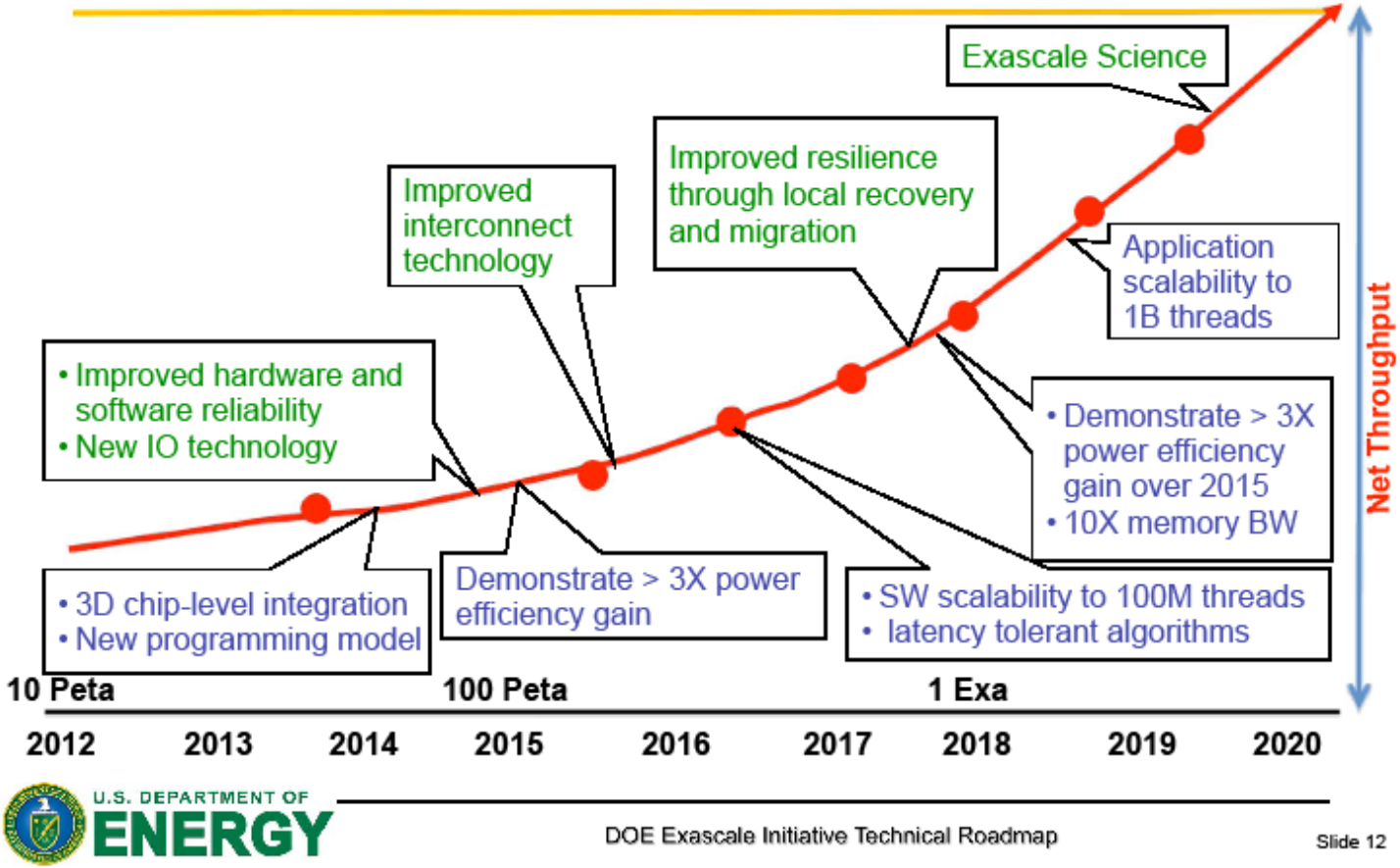
Convey Compilers

- Program in ANSI standard C/C++ and Fortran
- Unified compiler generates x86 & coprocessor instructions
- Seamless debugging environment for Intel & coprocessor code
- Executable can run on x86_64 nodes or on Convey Hybrid-Core nodes





Technology Roadmap



U.S. DEPARTMENT OF ENERGY

DOE Exascale Initiative Technical Roadmap

Slide 12

Predictions and Conclusions

- Parallelism will explode
  - Number of cores will double every ~2 years
  - Petaflop (million processor) machines will be common in HPC by 2015 (all top 500 machines will have this)
- Performance will become a software problem
  - Parallelism and locality are fundamental; can save power by pushing these to software
- Locality will continue to be important
  - On-chip to off-chip as well as node to node
  - Need to design algorithms for what counts (communication not computation)
- Massive parallelism required (including pipelining and overlap)

Kathy Yelick, 2008 Keynote, Salishan Conference

Concluding

- Uniprocessor Performance has to be increased
  - Heterogeneous here to stay
  - The easiest to program will be the correct technology
- Smarter Memory Systems (PIM)
- New HPC Software must be developed.
  - SMARTER COMPILERS
  - ARCHITECTURE TRANSPARENT
- New algorithms, not necessarily new languages

Finally


AS LEAD SOFTWARE ENGINEER, I GIVE YOU THE FIRST UNIT OF OUR TEN THOUSAND COPY PRODUCTION RUN.