MTAAP'07 Keynote

Michael Merrill

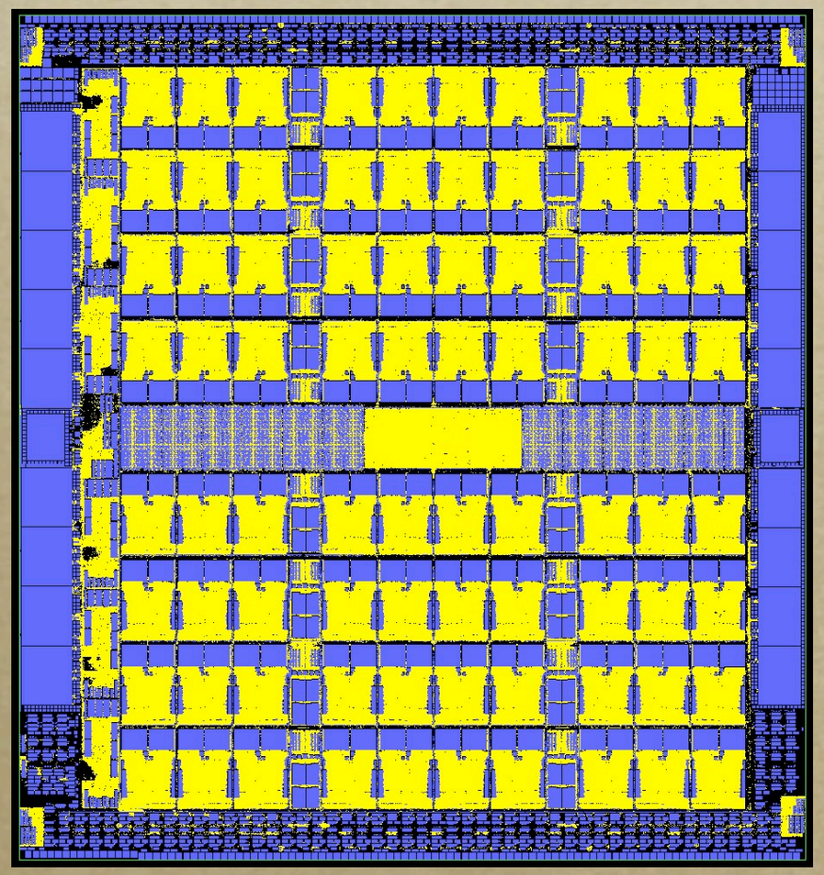
Outline

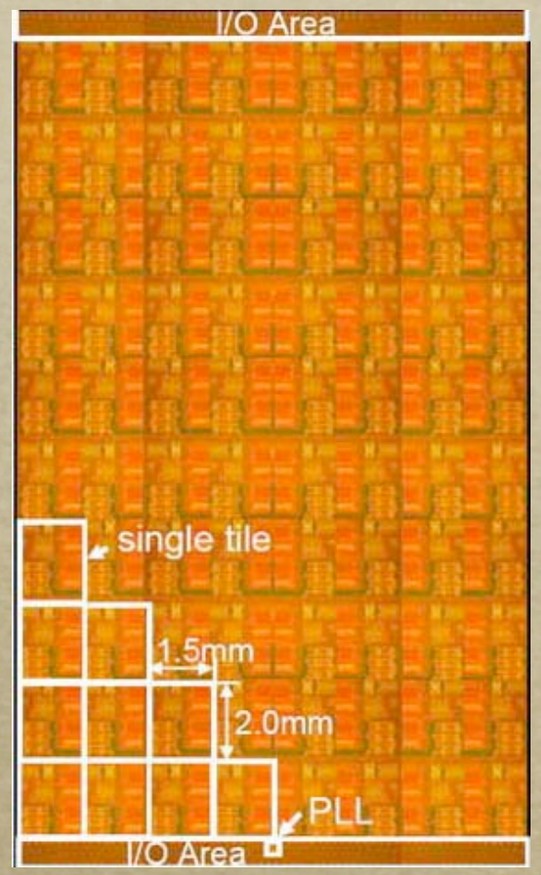
- *What's important... applications*
- *Making sense... of all this stuff*
- *What's necessary... I think*
- *What's possible... maybe*

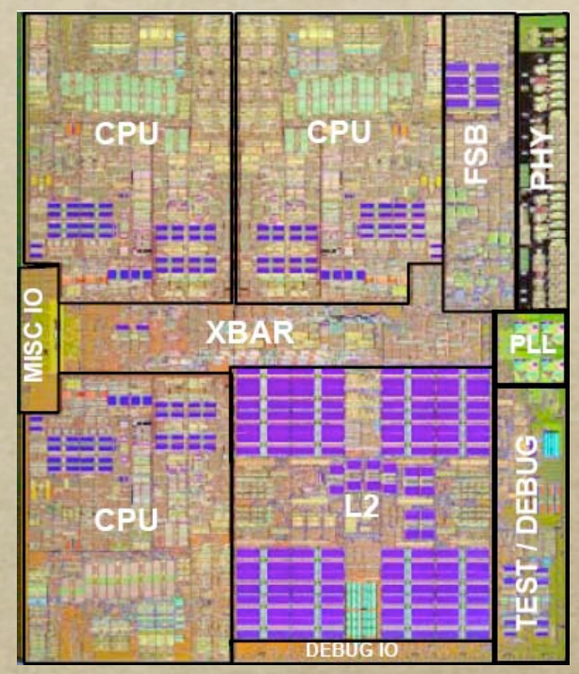
What's important... applications

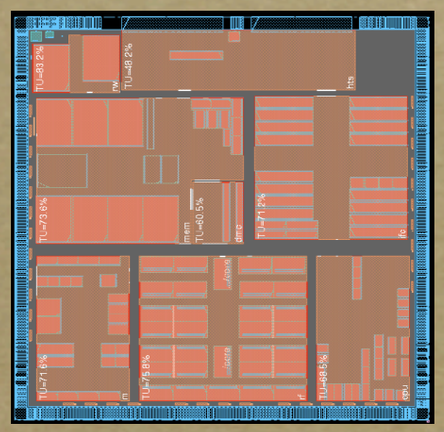
- *Data Structures are important... more support for linked data structures*
- *Ignored algorithm areas are coming back to bite us*
- *Sparse methods on unstructured data*
- *Adaptive methods are better aligned with nature but not with current architecture*
- *Helping humans deal with information overload*

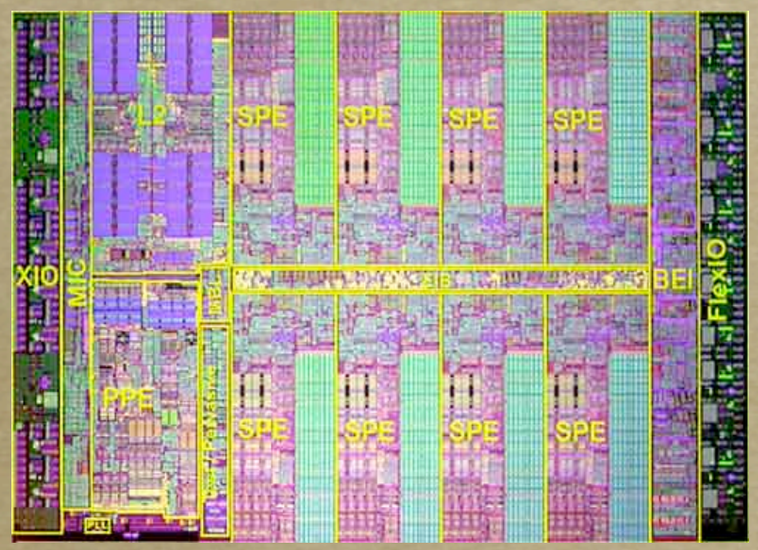
Making sense... of all this stuff

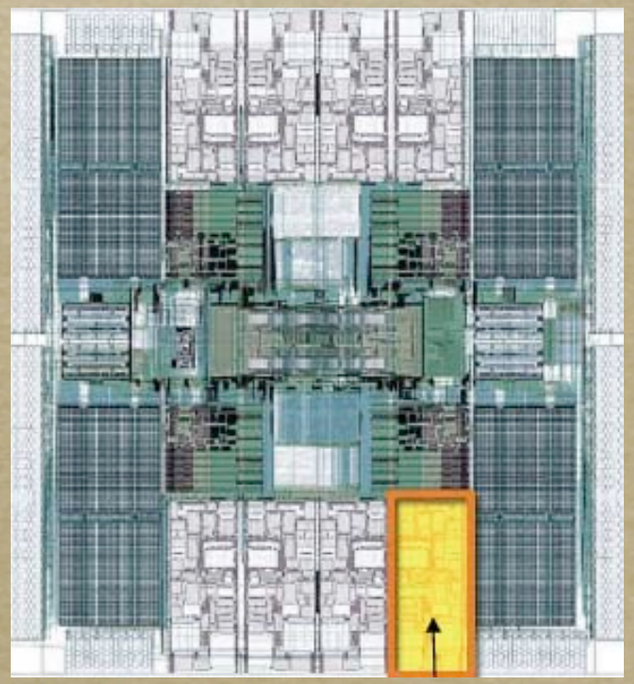












Spectrum

Hardware contexts per set of functional units

all contexts to one set of functional units	*one context to one set of functional units*

MTA/XMT *UltraSparc T1* *Cyclops64*

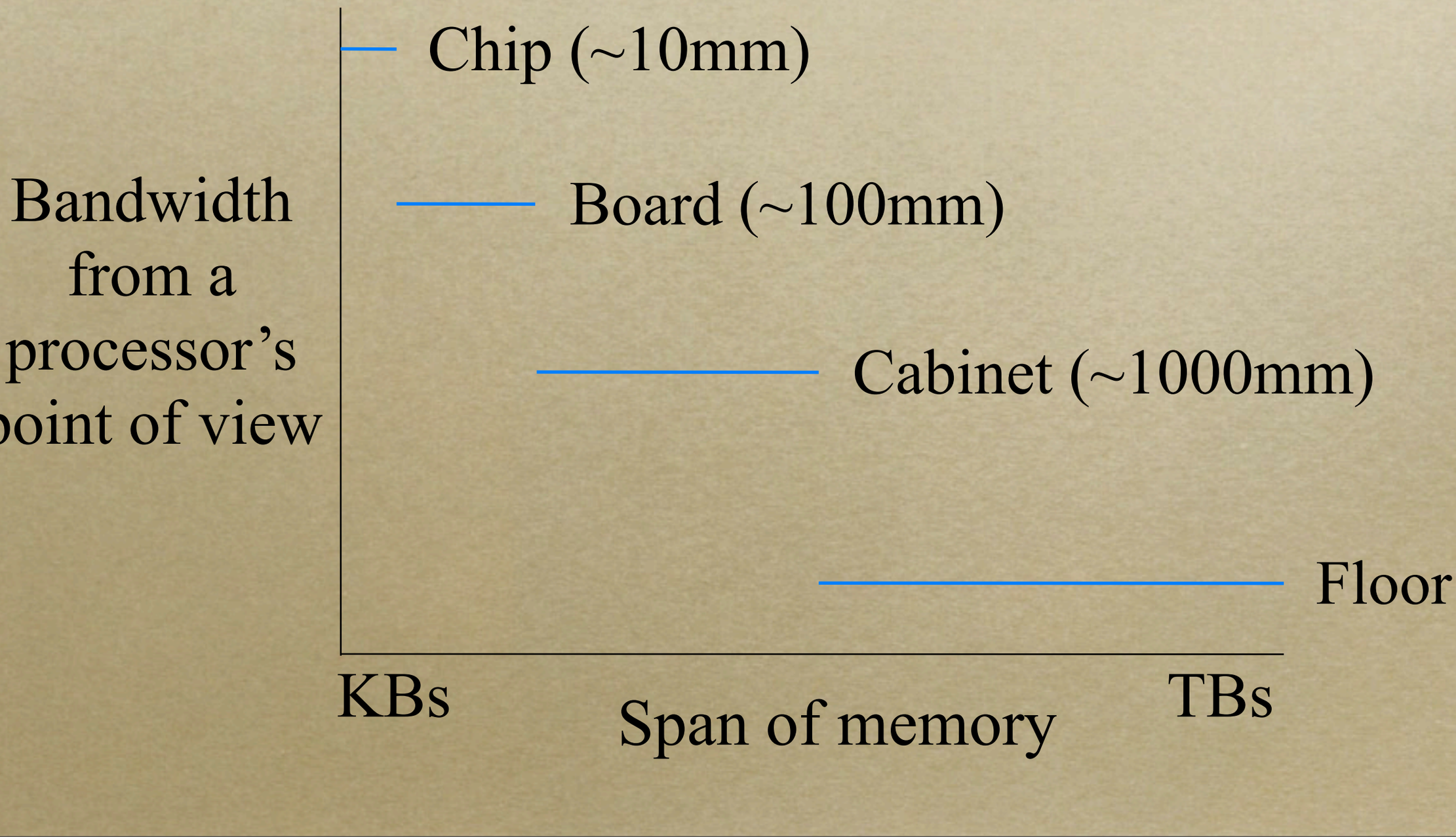
Stuff

- *Virtualization*
  - *How much is enough?*
    - *Fault tolerance*
- *How much baggage does a context have? Probably affects virtualization*
- *Synchronization*

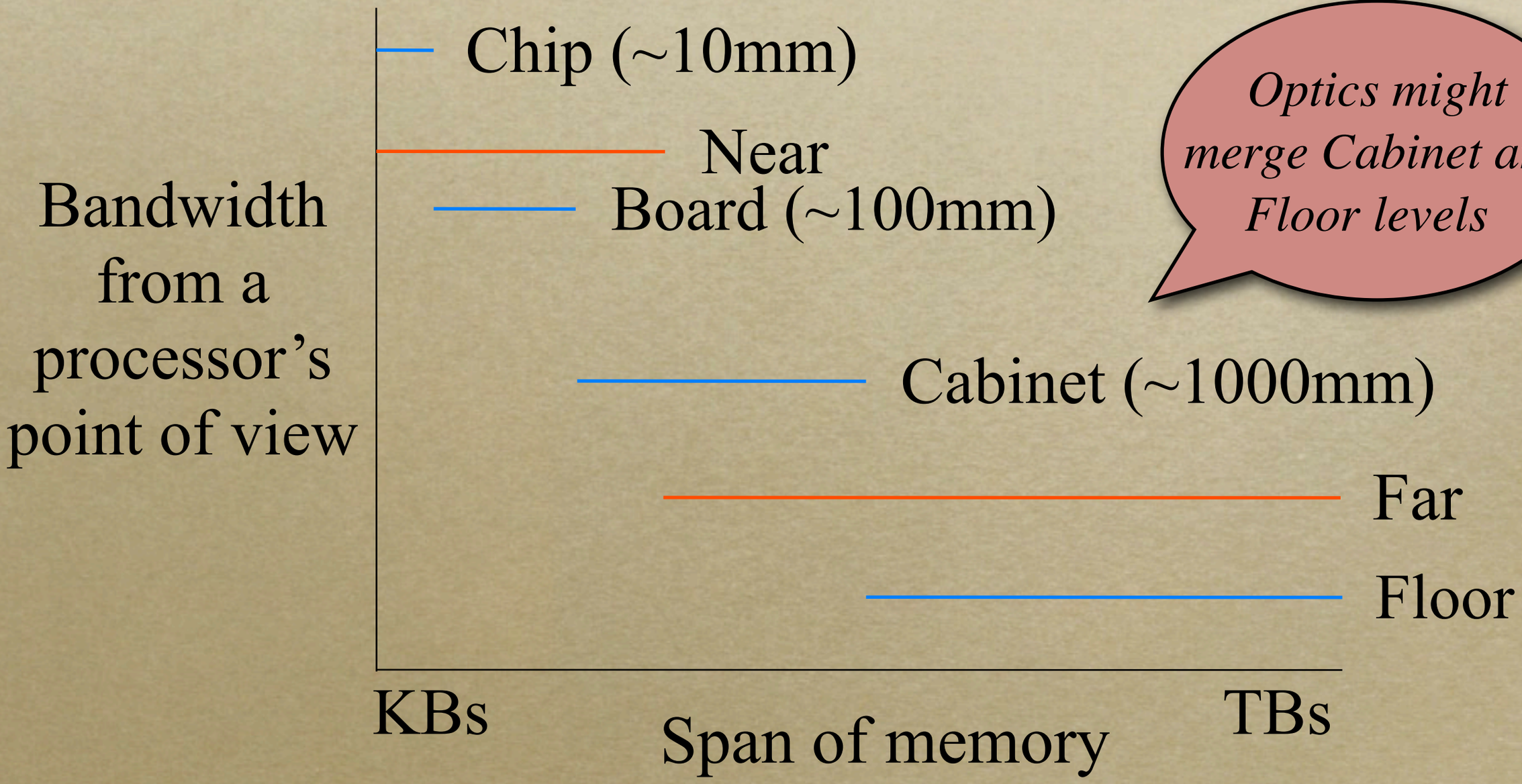
More Stuff

- *Explicit memory hierarchy?*
- *I-cache! Don't make the programmer worry about code size!?!*
- *Commercial use vs. scientific use... vs. something else*

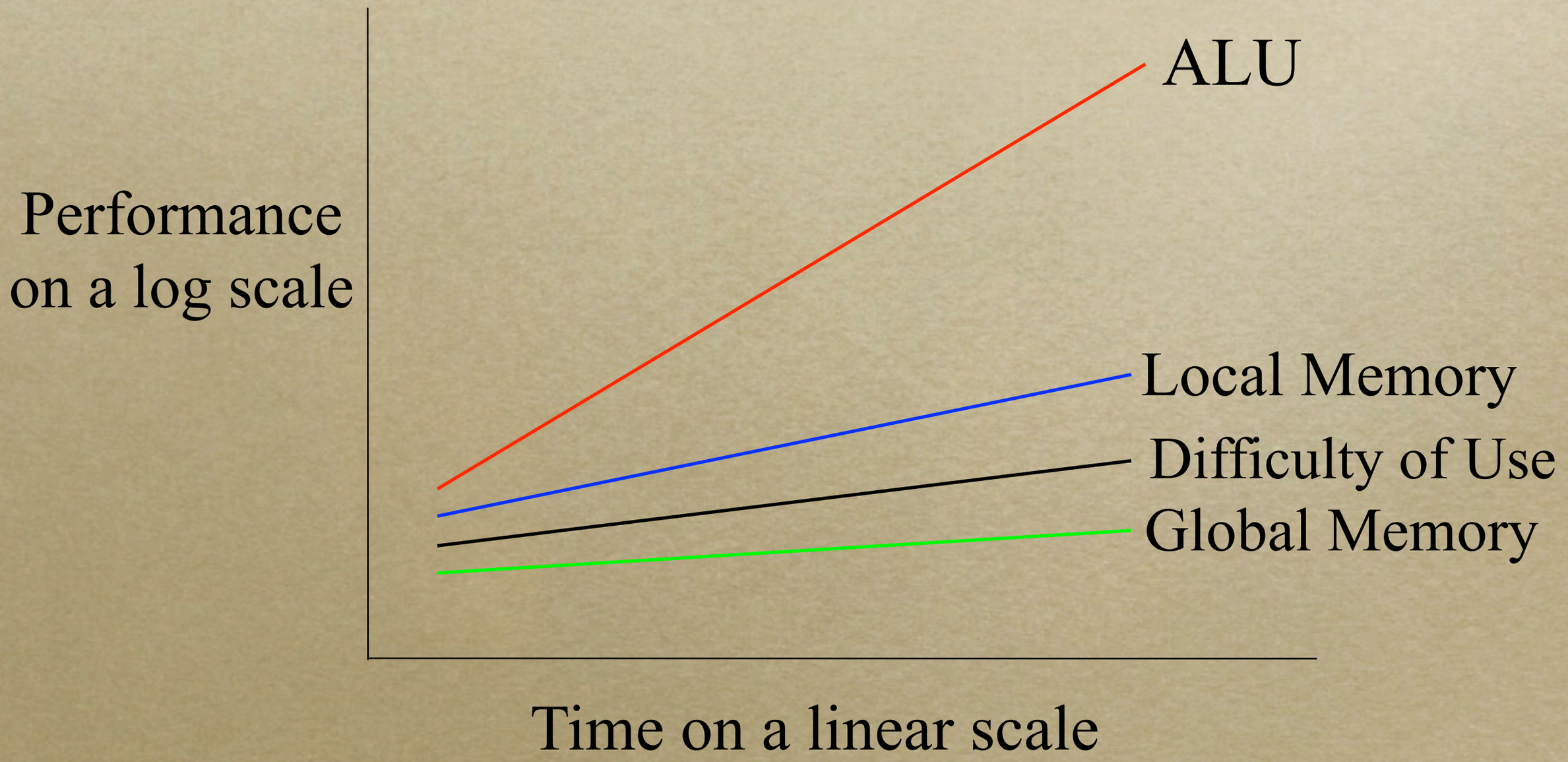
Natural Bandwidth Boundaries



Natural Bandwidth Boundaries



Trends we live with...



Different Balance

- *Costs have changed drastically*
- *Transistors are cheap... Wires are expensive*
- *Processor complexity vs. power is an issue*
- *Balance costs... apply transistors to use wires more effectively... not just for cache*
- *This is why you see architecture changing*

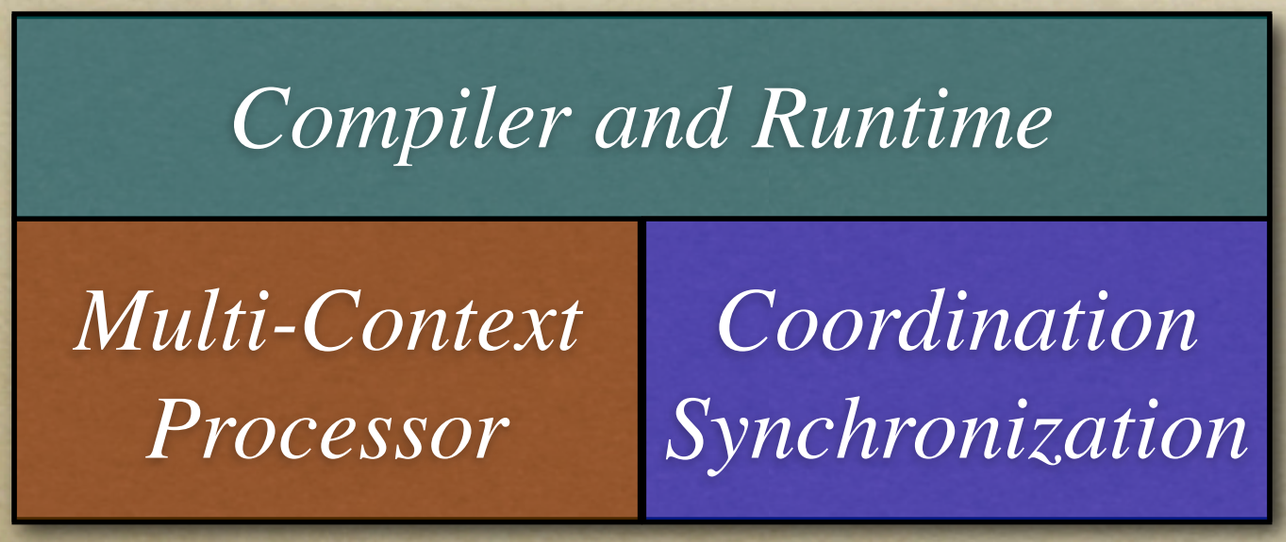
What's necessary... I think

- *Need to provide an effective system solution HW and SW!*
- *Why? Days of coarse grained scaling are at an end... so threads/contexts will necessarily work together to perform a task.*

Fabrication

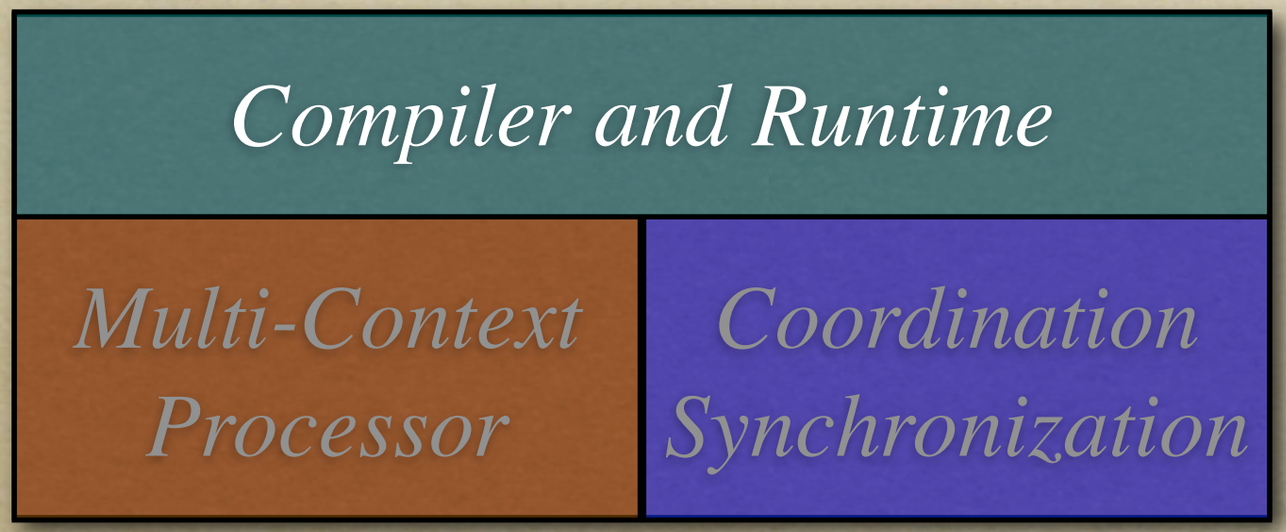
- *Tiled architecture with partial good chips for lower costs*
- *Detect failed computation*
- *Retry failed computation*
- *Move away from fixed number of threads/ contexts*

What's necessary... I think



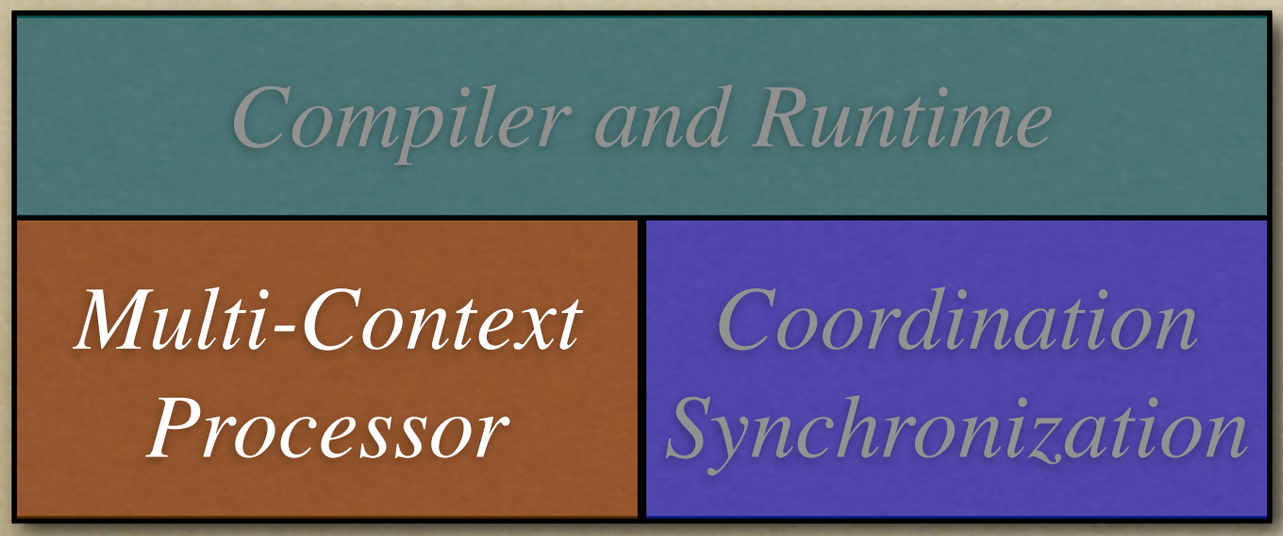
- *Need at least these three things working together to produce an effective environment for the application developer*

Effective Software



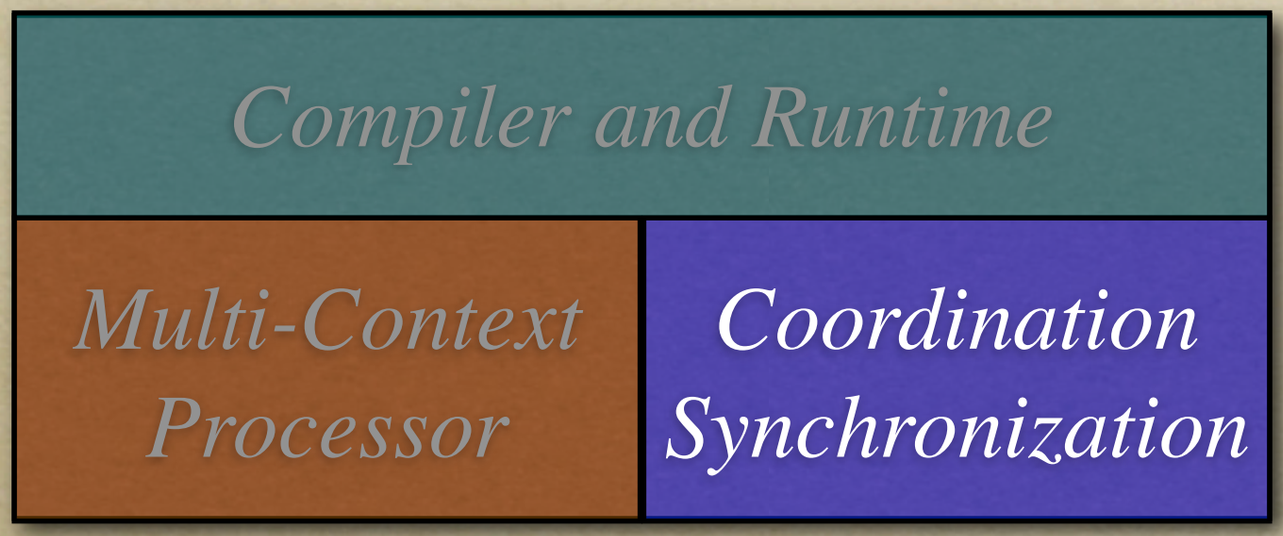
- *Runtime that provides effective dynamic work management so the unbalanced nature of the application can be mitigated.*
- *Compiler that takes advantage of such a runtime increases programmer effectiveness and productivity allowing them to concentrate on the application.*

Latency Tolerance/Management



- *Effective use of the bandwidth provided by the internal system networks through the use of latency tolerance and/or latency management techniques.*
- *Many of these techniques require the exposure of abundant fine-grained parallelism in the application.*

Low Overhead Coordination



- *Threads will necessarily work together to compute so effective coordination will be essential.*
- *Any cycles spent waiting on synchronization events are not spent computing and therefore decrease efficiency.*

What's possible... maybe

- *Don't look for any major companies to make things significantly better because it messes with the current business too much.*
- *Which direction to go?*

Straight Forward Scaling

	90nm	*65nm*	*45nm*	*32nm*
TU	*160*	*306*	*640*	*1266*
FPU	*80*	*153*	*320*	*633*
TU/XB	*2*	*3*	*4*	*6*
XBar	*80*	*102*	*160*	*211*
Clock	*500M*	*585M*	*684M*	*800M*
Perf	*80G*	*179G*	*437G*	*1.01T*
SRAM	*4.8M*	*9.2M*	*19.2M*	*37.9M*

- *start with Cyclops64*
- *22x23mm die*
- *150W to 190W*
- *3DE ?*

What About Software?

- *Need good compiler technology to exploit on chip explicit memory*
- *Much higher level of abstraction*
- *Need to separate the how-to from the what-for but express both*
- *Diagnose hot spots (resource contention)*
- *etc…*

- *Questions? ... I have a ton ;-)*