


Current HPC architectures and a little bit beyond

Aad van der Steen
NCF/HPC Research
The Netherlands

- 
1. A tiny bit of history.
 2. The speed development cycle.
 3. Where we are now.
 4. Work in progress ...
 5. ... and a bit of future.

A tiny bit of history — 1

What to do in 1980 when

- you needed a significant speedup;
- had no access to a Cray;
- had no budget to buy one?

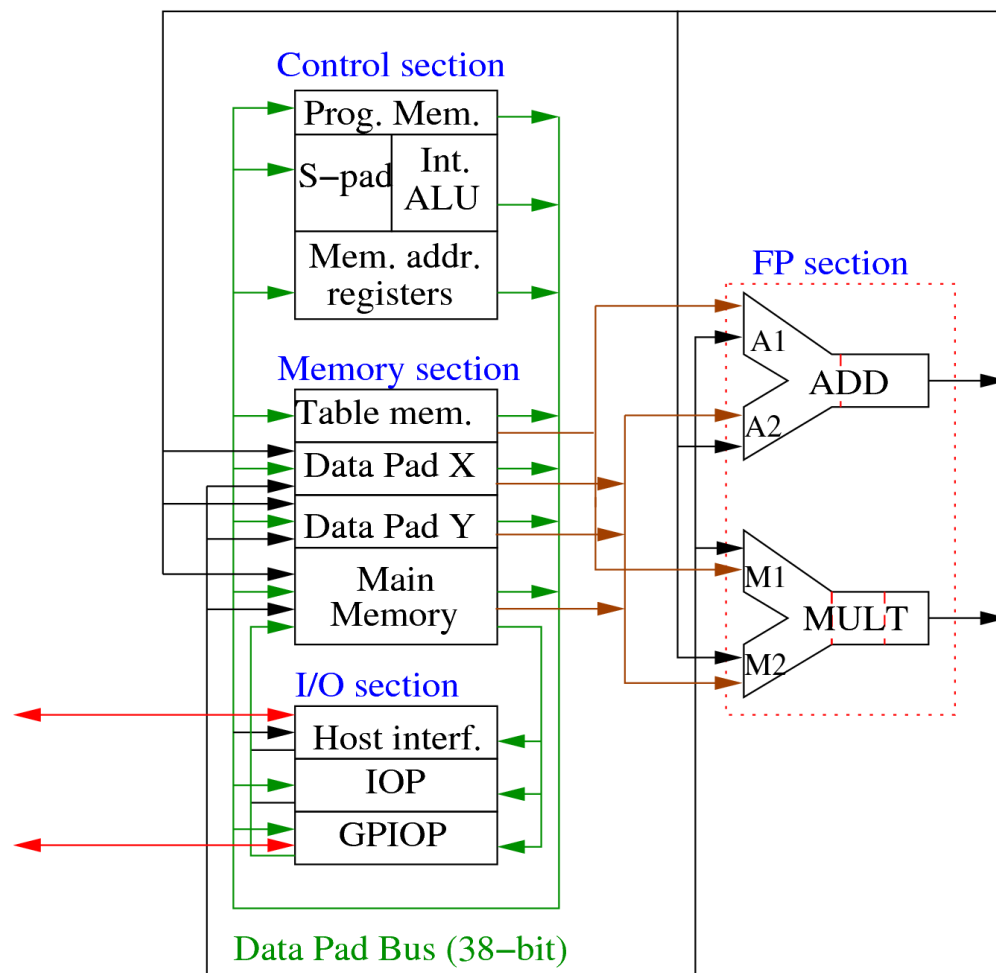
Possible solution:

You bought an accelerator, e.g., a Floating Point Systems AP-120B.

A tiny bit of history — 2

The FPS AP-120B was an attached processor with:

- a 16-bit ALU;
- 38-bit floating-point ADD and MULT pipelines, (resp. 2 and 3 stages);
- Peak performance 12 Mflop/s;
- Program memory 6× faster than data memory.



A tiny bit of history — 3

Mid 1980s Weitek produced its x167 floating-point co-processors ($x = 1, \dots, 4$).

They complemented processors from

- Intel (I286, I386);
- SPARC;
- Early MIPS (XL).

A case of half-integration: no software involvement from the user anymore in contrast to FPS accelerator.

It illustrates a development cycle in computer architecture.

The speed development cycle

From the earliest use of computers on (Collosus, 1943) users think their computers too slow (and they are right!). So, what happens:

1. Devise hardware add-on to accelerate.
(Dearth of software, cumbersome: for hero-programming).
2. Accelerator is absorbed in systems/processors.
(Maybe half-integrated: either transparent or complete software control).
3. User is not satisfied with speed of current machines.
(So ...):



Where we are now – 1

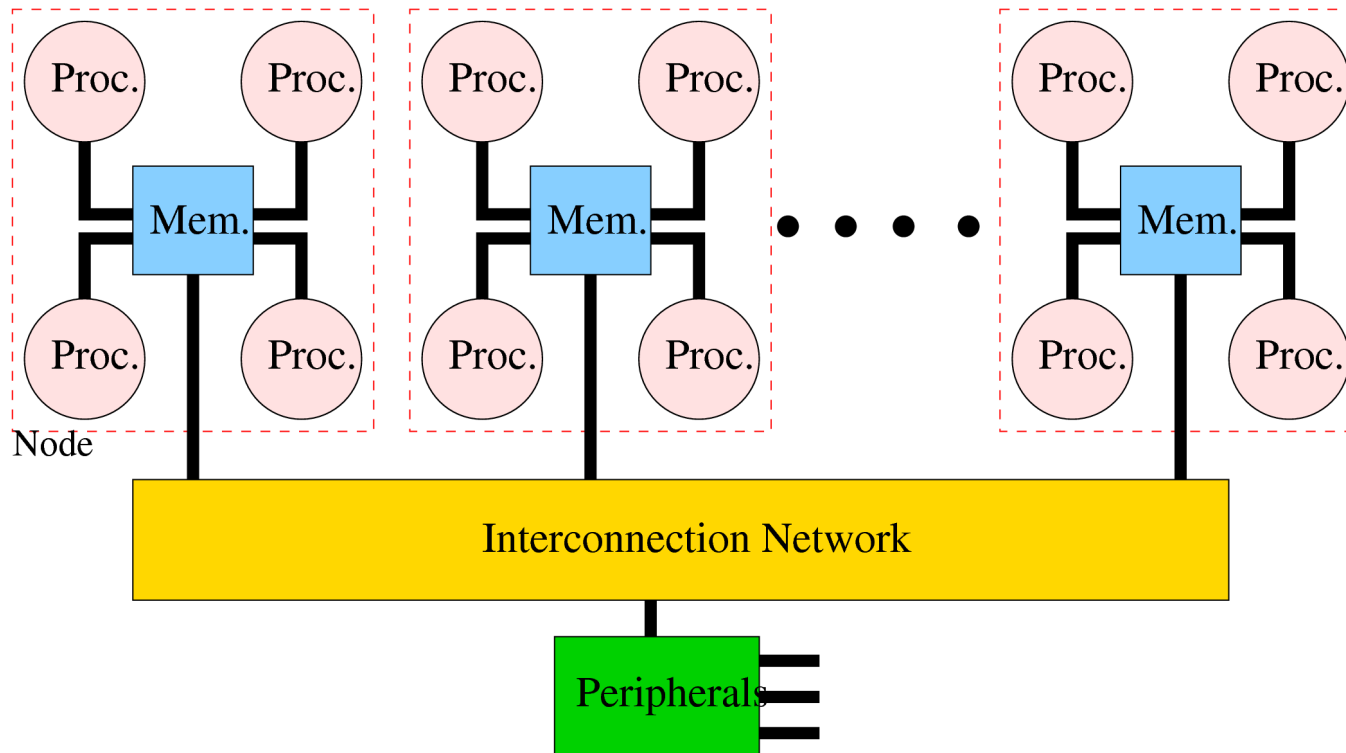
Simplest solution for processor speedup seems to be to increase about everything on chip, but:

1. Clock frequency. ↔ Widens memory-CPU gap. (-)
2. Cache size. ↔ Tends to be slower. (-)
3. Process threads. ↔ Can take advantage of stalling memory requests. (+)
4. Number of processor cores. ↔ Have to share memory bandwidth. (-)
5. Network on chip. ↔ Topology important. (+/-)

Net result: Efficiency likely to decrease, peak performance likely to increase.

Where we are now – 2

Let's look at a generic HPC system:



Defining characteristics:

- Multi-processor/multi-core nodes
(not necessarily SMP nodes anymore!)
- Network interconnecting nodes

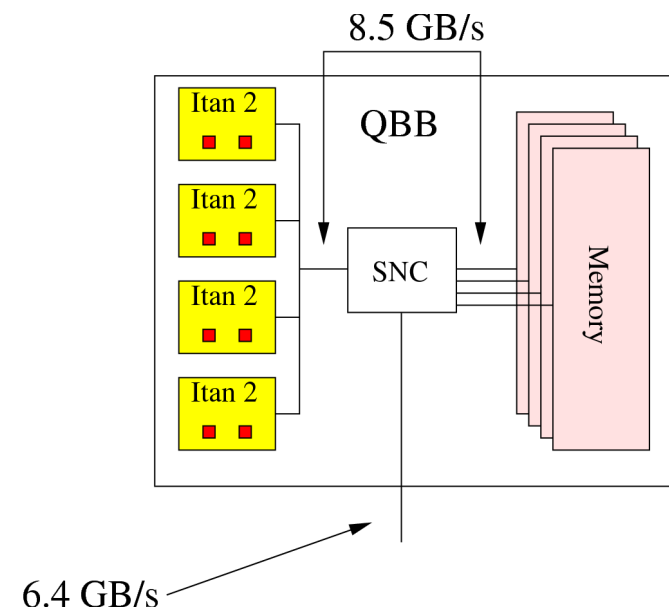
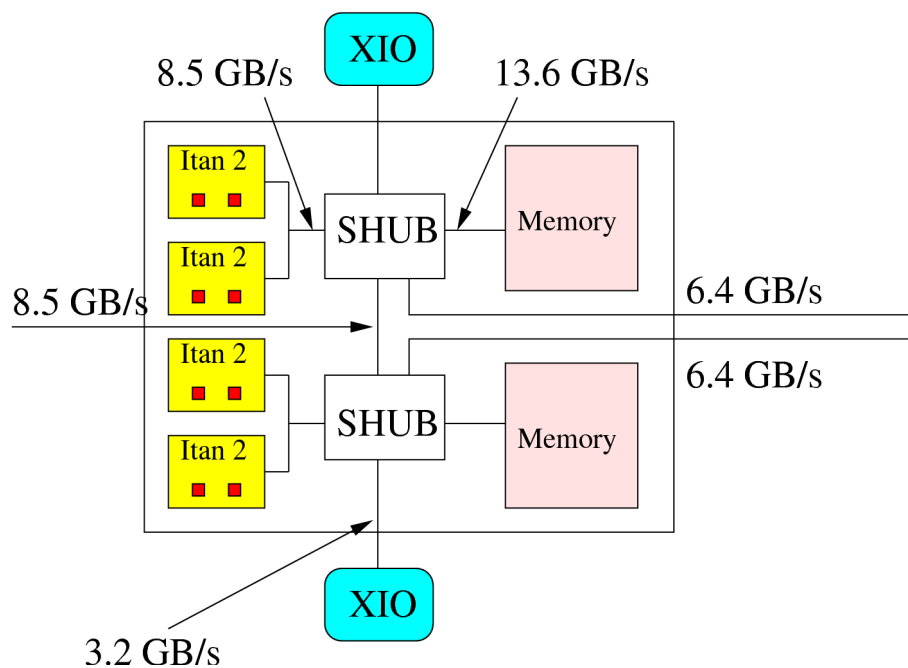
Where we are now – 3

This trend is induced by cost and physical factors:

In **integrated parallel** systems 4—32 processors/node.

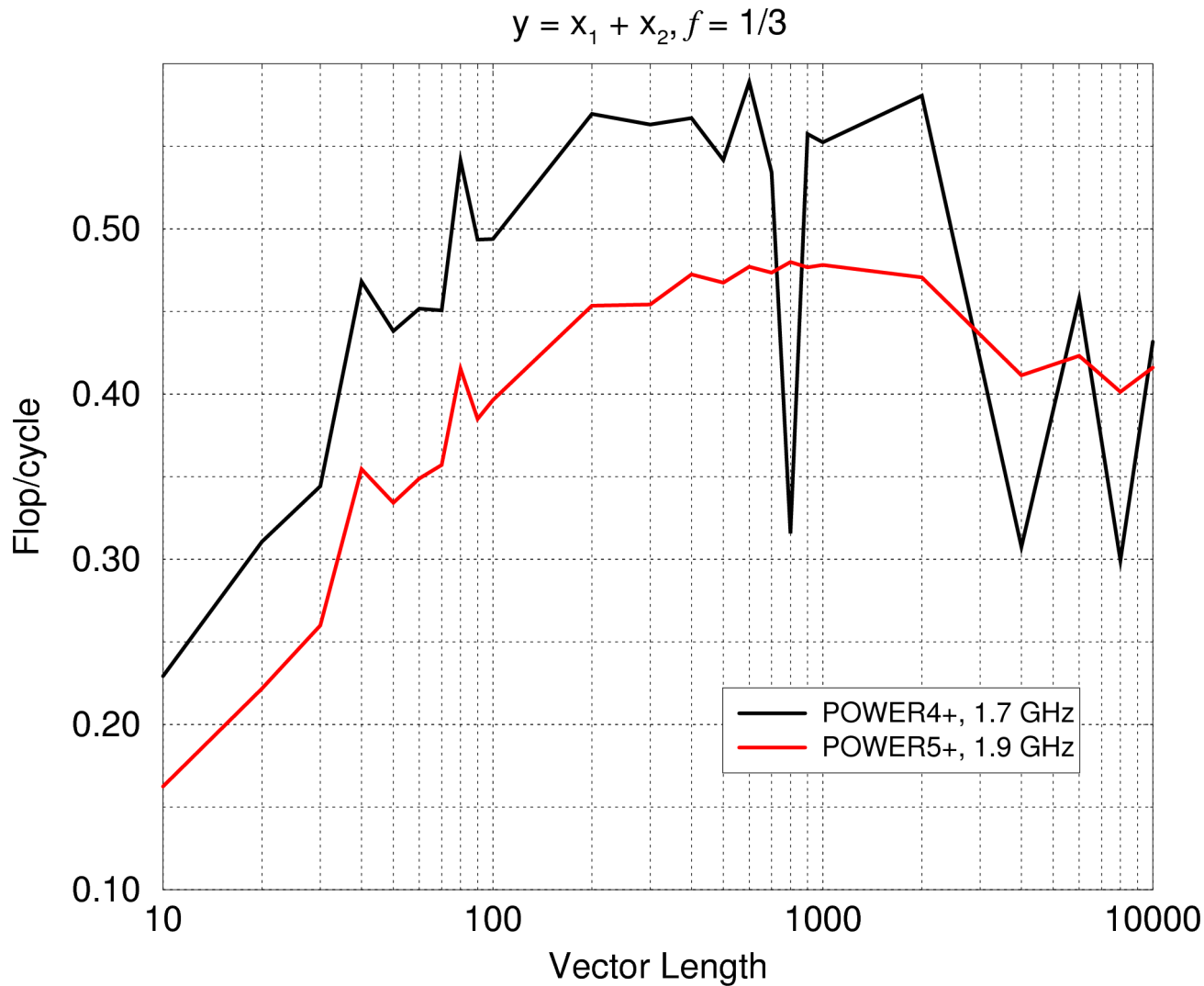
In **clusters** 4 cores/node (has gone up to 8—16 recently).

Node architecture may differ, even with similar components (see <http://www.euroben.nl/reports/>):



Where we are now – 4

Does a larger cache help? — Not always:



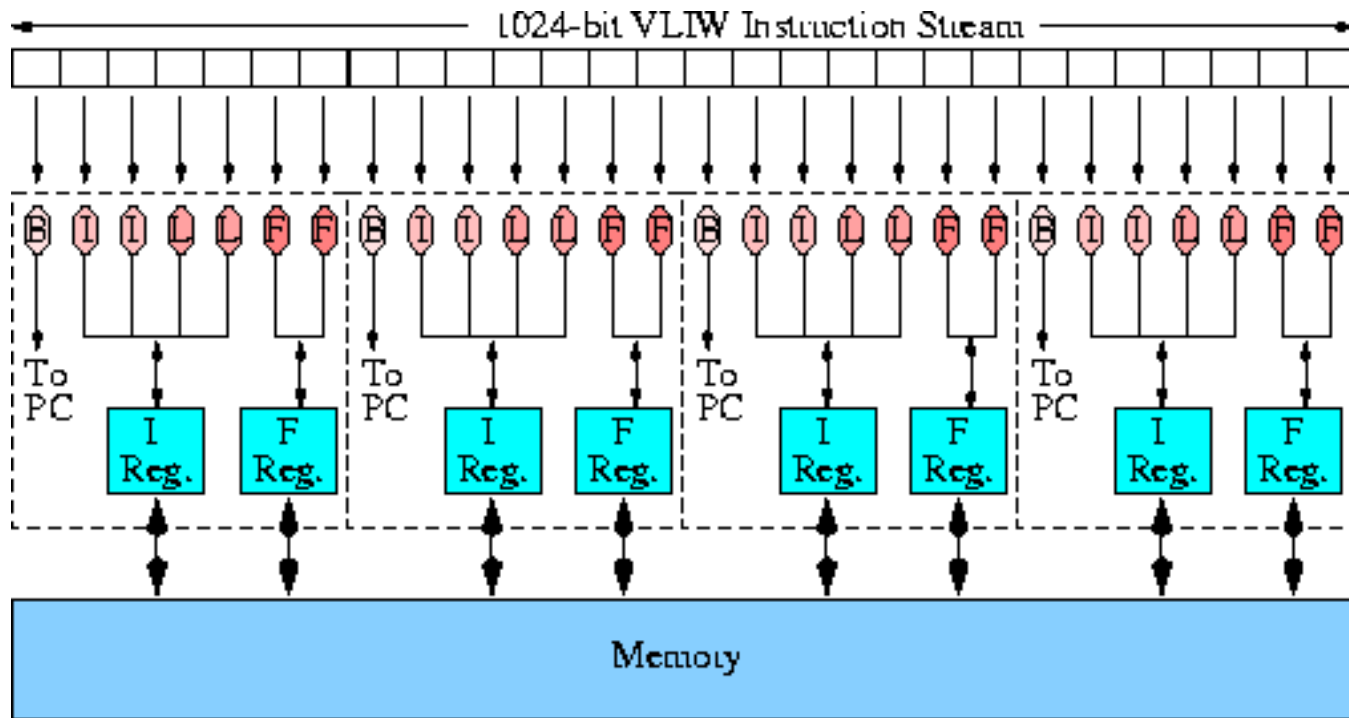
Where we are now – 5

What else can we do to fight the memory wall?

1. – Static scheduling.
2. – Vector processing.
3. – Latency hiding.

Where we are now – 6

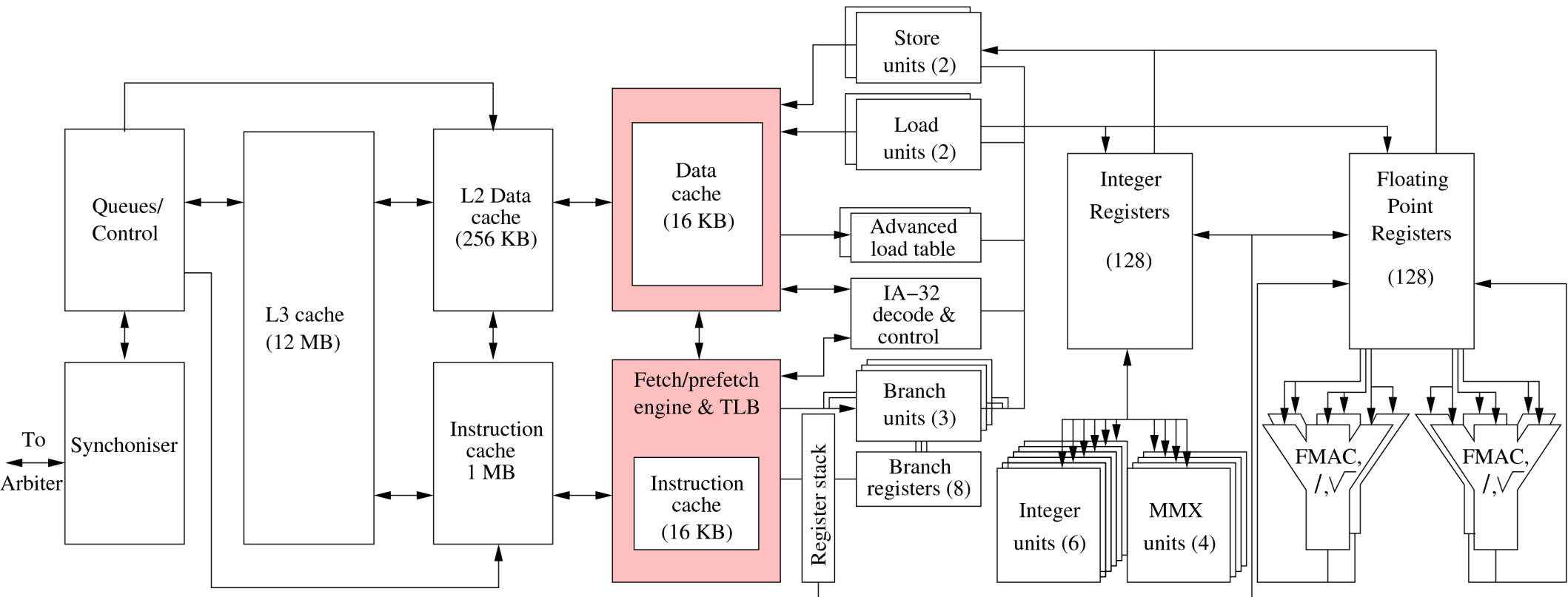
Static scheduling stems from Very Long Instruction Word architectures (VLIW), example: **Multiflow 300**. **Advantage:** less complex instruction scheduling and supporting hardware.



- B: Branch Unit
- I: Integer ALU
- L: Load/Store Unit
- F: Floating-point Unit
- I Reg.: Integer Register
- F Reg.: Floating-point register
- PC: Program Counter

Where we are now – 7

Present-day representative: Intel's Itanium EPIC architecture



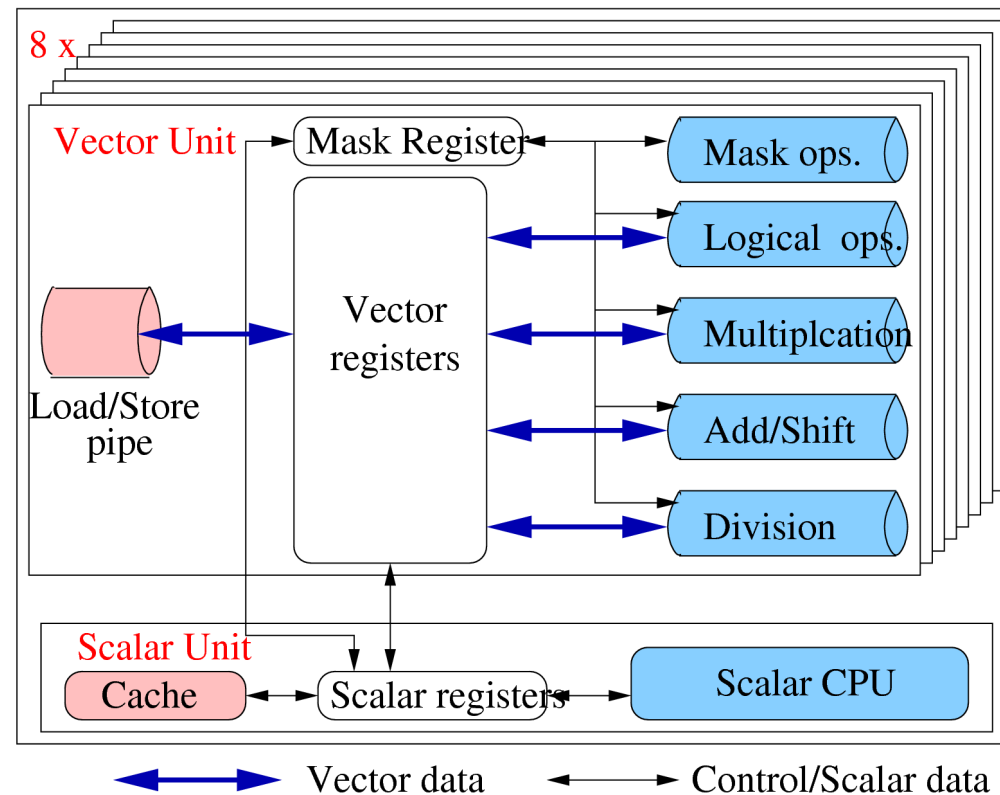
Less extreme instruction word length: 128 bits comprising 3 41-bit instructions, a **bundle**, and 5 predication bits.

Two bundles can be scheduled at the same time.

Where we are now – 8

Another way out of scheduling complexity is vector processing:

NEC SX-9



Because of lower instruction count and clear instruction separation high efficiency in vector units. But there are drawbacks...

Where we are now – 9

	MIPS R14K 500 MHz Mflop/s	Itanium 2 1.6 GHz Mflop/s	Cray X1E 1.125 GHz Mflop/s
daxpy	328	2768	9709
1 st order recursion	244	397	93

Both wide instruction word machines **and** vector processors cannot deal well with recursions:

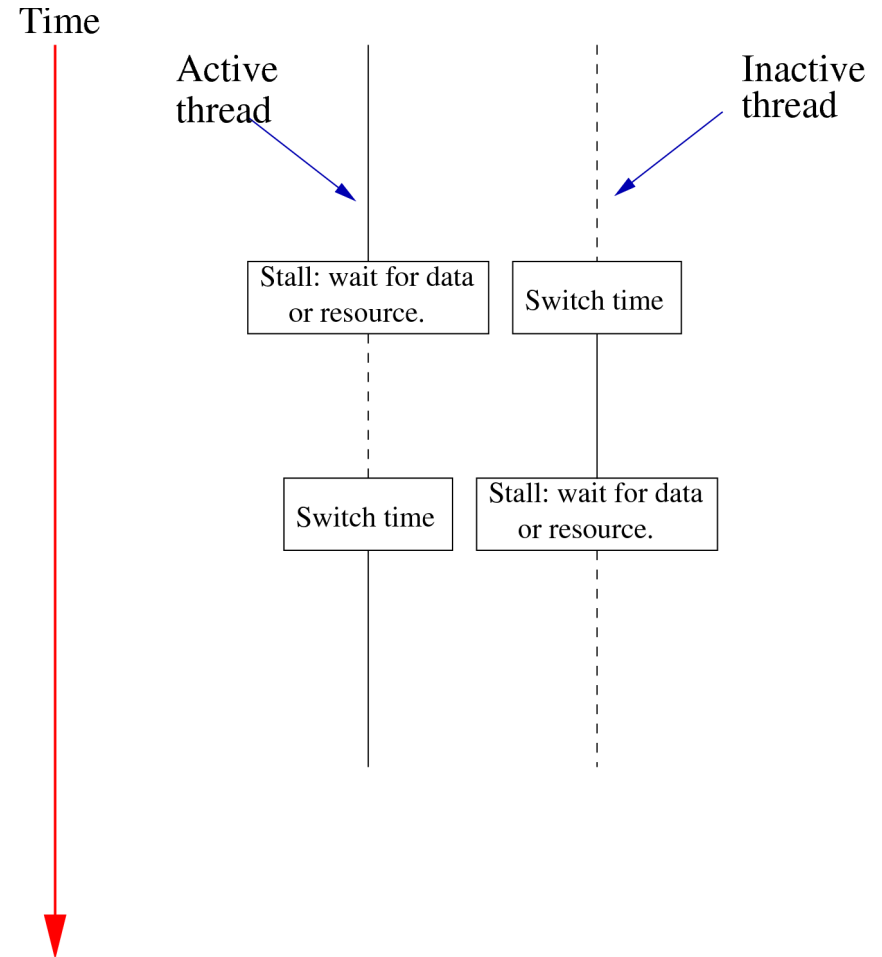
- Cannot be vectorised.
- Cannot be scheduled ahead. So, pipelining advantage is lost.

Where we are now – 10

Multi-threading:

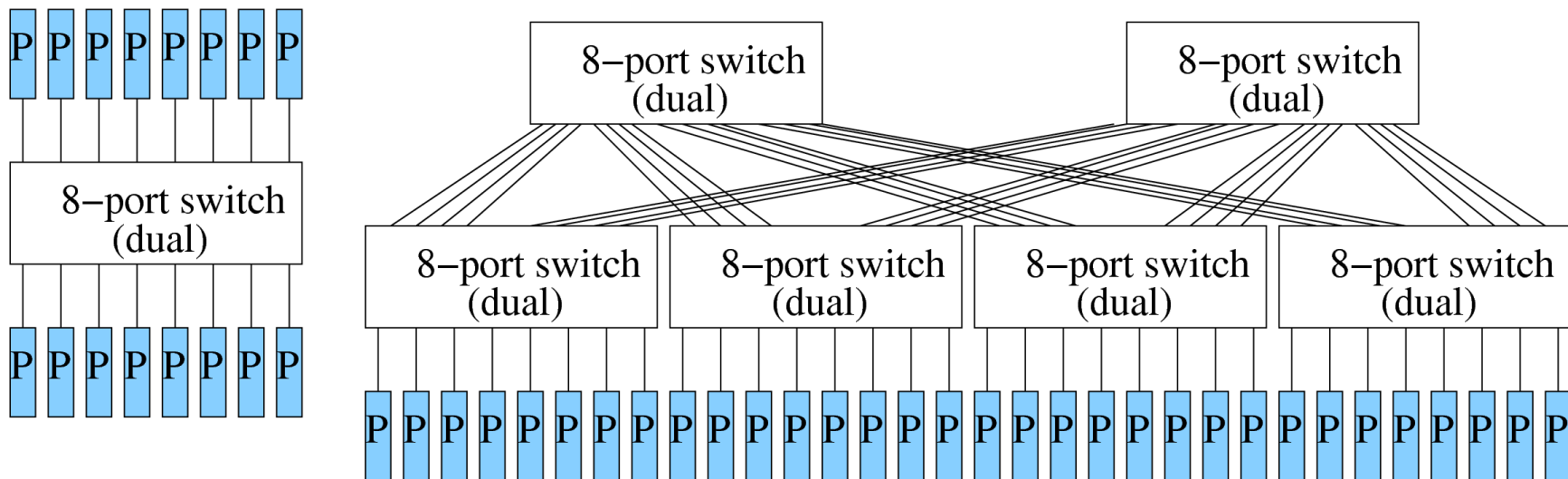
Available in almost all main (scalar) processor families and extremely so in the Cray XMT.

However, care must be taken in using automatic multi-threading: can actually **slow down** applications.



Where we are now – 11

Complexity of networks grows superlinearly (when good performance is required).



Obviously this becomes quickly very expensive, so a network with a bandwidth that is reduced for processors that are more remote is popular: a **fat tree**.

Where we are now – 12

Networks in clusters also have greatly improved lately
Which makes the clusters more versatile (and costly):

	Bandwidth MB/s	Latency μ s
Gbit Ethernet	120	± 40
Infiniband SDR (4 \times)	850	7
Myrinet MX-10G	1000 [‡]	2.2
QsNet	980 [†]	2.0
SCI	500	1.5

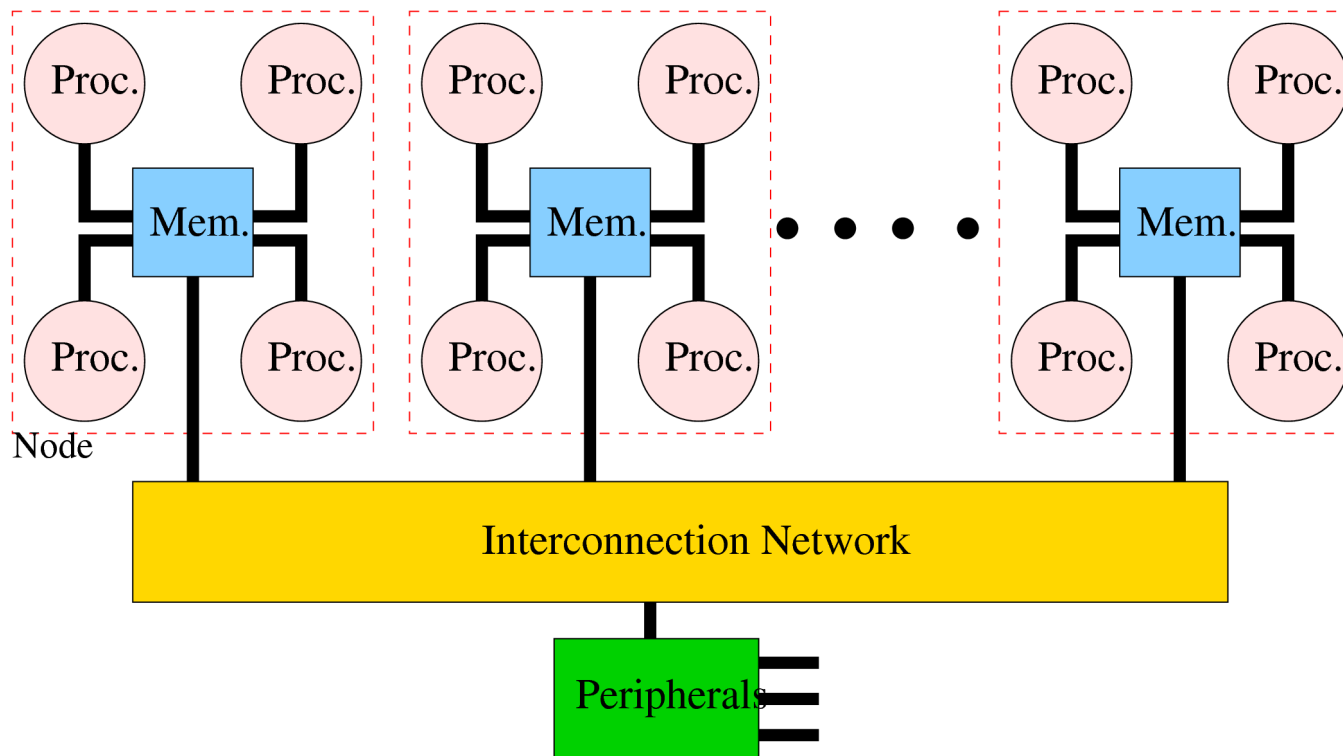
[‡]Constrained by PCI bus, ± 1200 MB/s

[†]Constrained by PCI bus, > 1000 MB/s

Recently: Woven Inc.'s EFX 1000 10 GbE:
Bandwidth ± 1 GB/s, latency ± 4 μ s.

Where we are now – 13

The macro-architecture of HPC systems is presently remarkably uniform:



Will it stay that way? — **Certainly not.**

Work in progress ... – 1

There are several factors that have an impact on the system architectures in the present:

- 1 Power consumption has become a primary headache.
- 2 Processor speed is never enough.
- 3 Network complexity/latency is a main hindrance.
- 4 There is **still** the memory wall.

Interestingly, solutions for point **1** and **2** can often be combined.

Work in progress ... – 2

Since a few years computational accelerators of various kinds are offered that may address **speed and power consumption**.

Example: ClearSpeed CSX600.

- Theoretical Peak: 192 Gflop/s/chip.
- Two chips/board.
- Power consumption: 25 Watt/board.

Power concern

Work in progress ... – 3

Another example is the new PowerXCell 8i from IBM:

Rather than increasing the clock speed on going from 90 nm to 65 nm feature size IBM chose to:

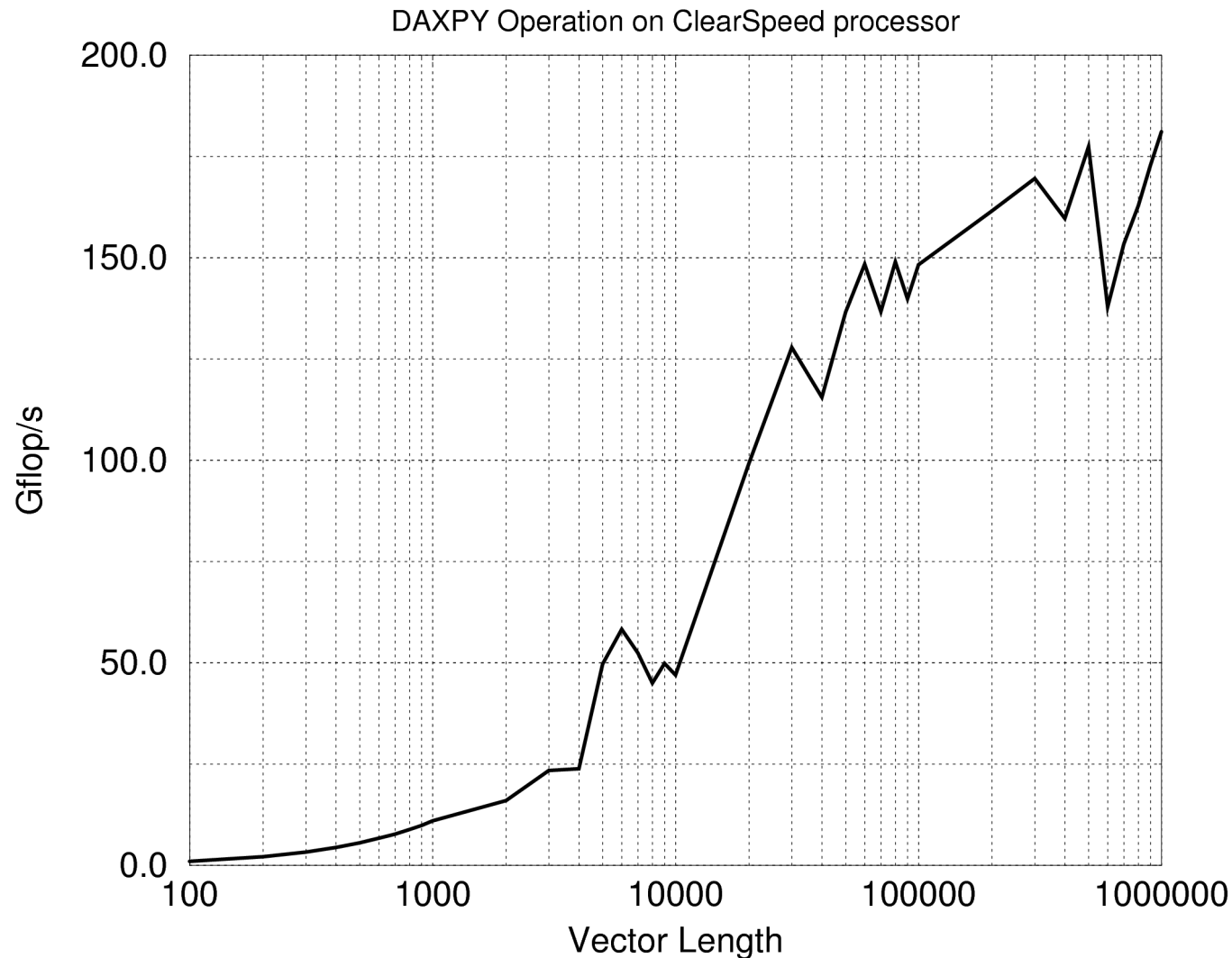
- 1 Keep the clock speed constant.
- 2 Add few additional devices (primarily to support 64-bit computation).
- 3 Use DDR-2 memory instead of RAMBUS memory.

Result: About twice performance/Watt compared with old Cell processor.

Power concern

Work in progress ... – 4

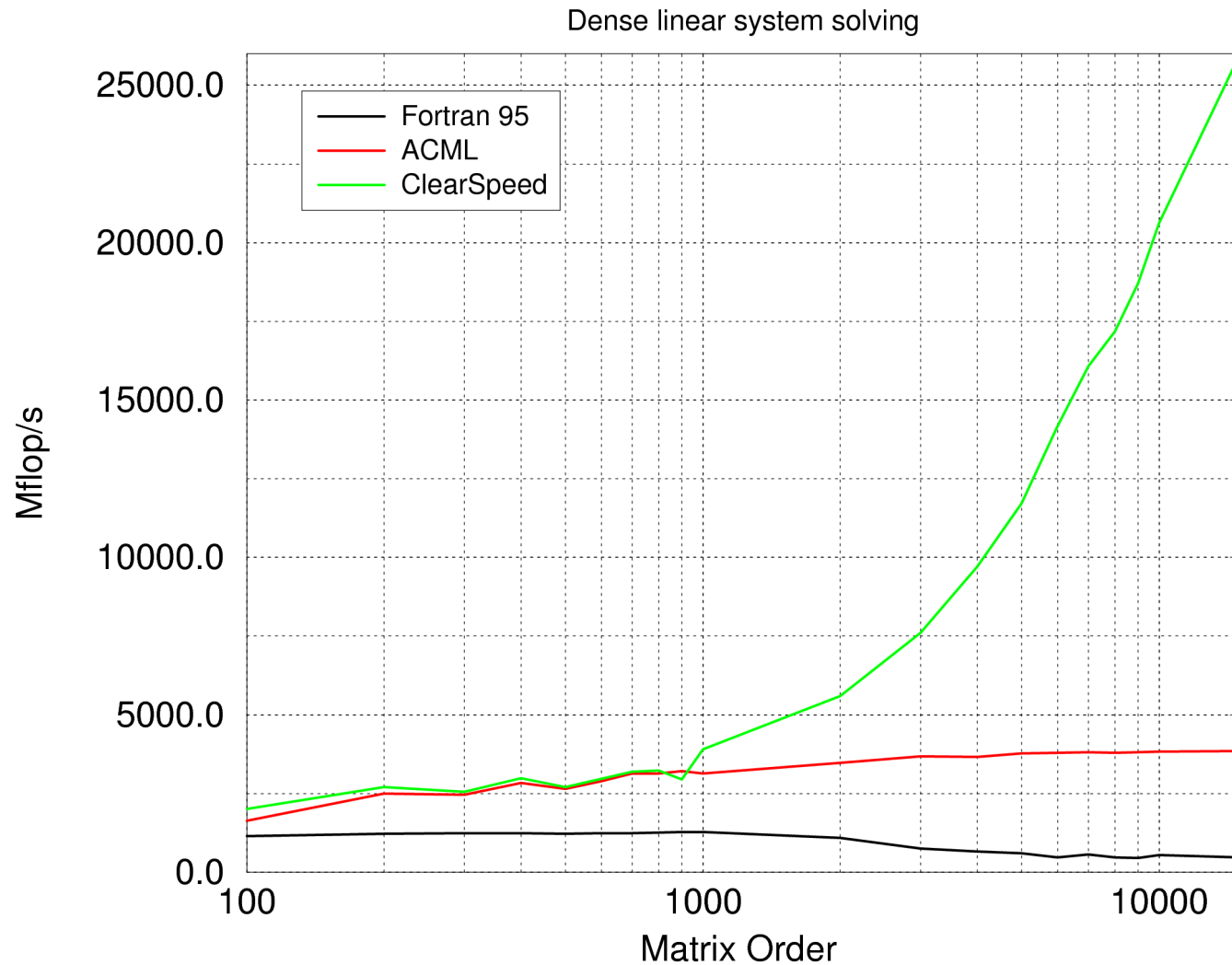
Possible to attain a high fraction of the Theoretical Peak Performance **on the chip proper.**



However...

Work in progress ... – 5

One has to ship data to/from the ClearSpeed board, restricting the overall speed:



Work in progress ... – 6

This is a general problem that is met by at least AMD and Intel:

- AMD through its Torrenza initiative: use HyperTransport connections as provided by AMD.
- Intel by opening its socket specifications (will be the QuickPath Interface from 2nd half of 2008 on).

Goal: Let accelerator communicate directly with the system's memory and General Purpose Computational Cores.

Work in progress ... – 7

Other types of accelerators:

- Field Programmable Gate Arrays (FPGAs), already mostly embedded on a card (CPU Tech, DRC, Pico, ...) or integrated in a system (Cray XTs, SGI RASC, SRC, ...).
Can also be very power effective.
- High-end Graphics boards (AMD/ATI, NVIDIA).
High power usage but Gflop/Watt ratio still good.

Drawback of **all** accelerators:

- Software Development Kits far from uniform (but improving rapidly, OpenFPGA initiative, ...). → **No standard (yet).**
- Programming is hard work.

Work in progress ... – 8

Tools range from a library set (for FPGAs almost always in the signal processing area) to C(-like) languages to develop accelerated routines.

– Examples of C(-like) languages:

* C^n – ClearSpeed

* Impulse C – Pico

* Handel C – Celoxica

* Mitrion C – Mitrionics

* CUDA – NVIDIA

– Graphical Developer:

* Viva – Nallatech, Starbridge (Windows Interface only)

– Libraries:

* Celoxica, ClearSpeed, Mitrionics, Nallatech, Starbridge, ...

Work in progress ... – 9

Some new vendors have been looking at the **total systems design** to get high performance with low power consumption:

– Liquid Computing LiquidIQ:

- * 4 AMD dual/quad-core Opterons/compute module.
- * Up to 20 modules/chassis.
- * Integrated point-to-point network via midplane, bandwidth ≤ 2 GB/s, latency $2.5 \mu\text{s}$.
- * Low power consumption: (≤ 14 kW/chassis).
- * No single point of failure.

Work in progress ... – 10a

Second and extreme example:

- SiCortex 5832:
 - * 5832 MIPS64 processors in 972 6-CPU nodes @ 500 Mhz (1 Gflop/s peak).
 - * New type of network (Kautz graph), bandwidth 2 GB/s, latency $\pm 2.0 \mu\text{s}$.
 - * Revolutionary design with respect to power consumption: 18 kW.

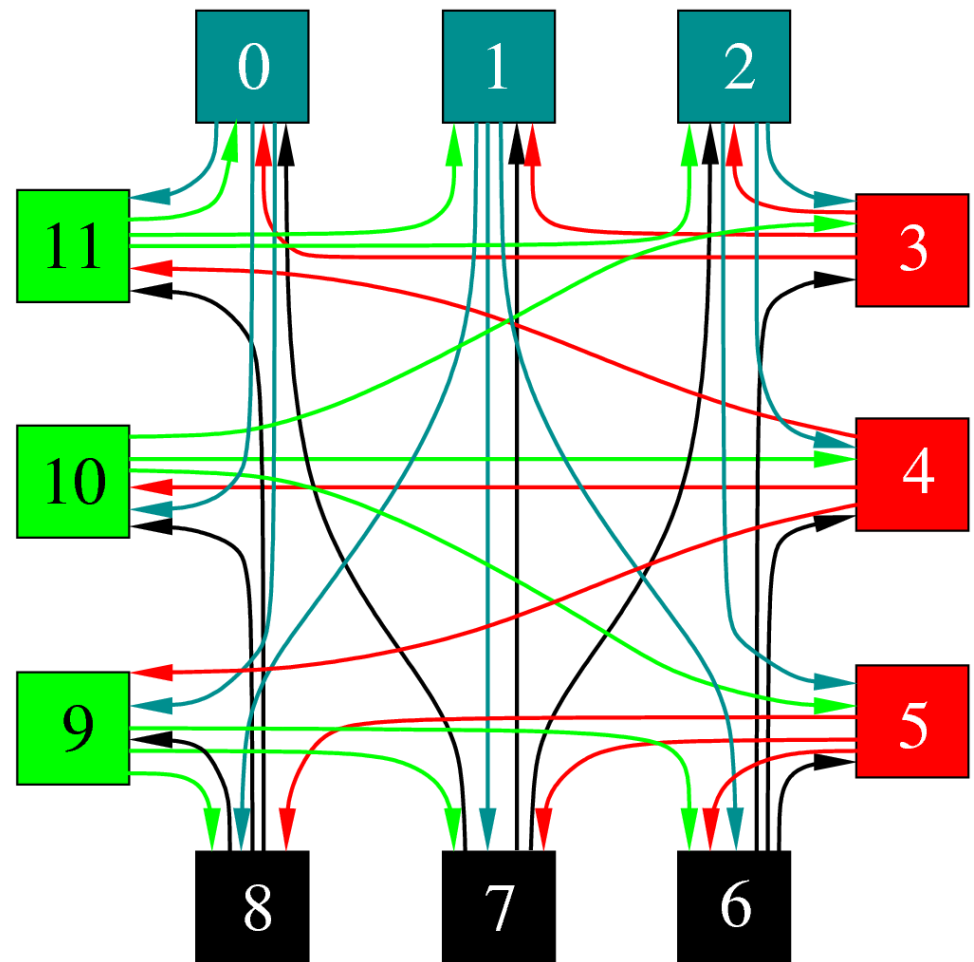


Work in progress ... – 10b

An aside: The Kautz graph:

The diameter of this simplest non-trivial Kautz graph is 2:

At most 2 hops are required to reach any node.



Work in progress ... – 11

Recent developments w.r.t. networks/components:

- Quad Data Rate InfiniBand: Mellanox just turned out the first Host Bus Adapters. Capable of:
 - * 1, 4, or 12 GB/s (1×, 4×, 12×) bandwidth.
 - * Dynamic Routing.

N.B. Only HBAs are available at this point, no switches and the like yet.

- Quadrics QsNet^{III} developed in conjunction with HP and European partners (e.g., Forschungszentrum Jülich). Might be (partly) photonic.

... a bit of the future – 1: Networks

Addressing network speed/complexity, **photonics**:

- IBM, Intel, Luxtera have demonstrated integrated photonic communication:
 - * Bandwidth 10—20 GB/s.
 - * Latency in ps—ns range.
- Lightfleet will market first 32×32 all-optical crossbar this spring.

... a bit of the future – 2a: Memory

A quick overview:

Attribute	DRAM	SRAM	NAND	EDRAM	FCRAM	MRAM	Z-RAM
Fast	B+	A+	C-	A	B-	A(+)	A+
Inexpensive	A+	C-	A+	C-	A	C	A+
Durable	A+	A+	C	A+	C	A+	A?
Reliable	A+	B-	B	A-	A	A+	A?
Low Power	B	C	A	B-	C	C	A
Non-Volatile	F	F	A+	F	A+	A+	F

(A+ best, F worst)

... a bit of the future – 2b: Memory

Remarks:

A near future contenders: **1.** Magnetic RAM (MRAM)

Based on difference in resistivity of aligned or non-aligned spins of free electrons in a magnetic medium.

Highly desirable properties:

- Permanent: no current required to maintain its information.
- Fast, presently about as fast as SRAM (few nanosec.) with potential for even faster switching times.

To improve:

- Density still not as high as normal DRAM.
- Production costs still high.

... a bit of the future – 2c: Memory

Remarks:

A near future contenders: **2. Z-RAM, TTRAM**
Based on “floating body effect” encountered in Silicon-On-Insulator technology.

Desirable properties:

- Very dense: density up to twice as high as that of DRAM.
- Speed comparable to SRAM.

Drawbacks:

- No permanent storage as in MRAM.
- No volume production, not stabilised (reliability, durability not yet confirmed).

... a bit of the future – 3a: Processors

At the end of the day it are the processors that must drive the performance. So, we may expect new processor architectures that in some way try to improve on current ones.

There may be a large variety but they will have some properties in common:

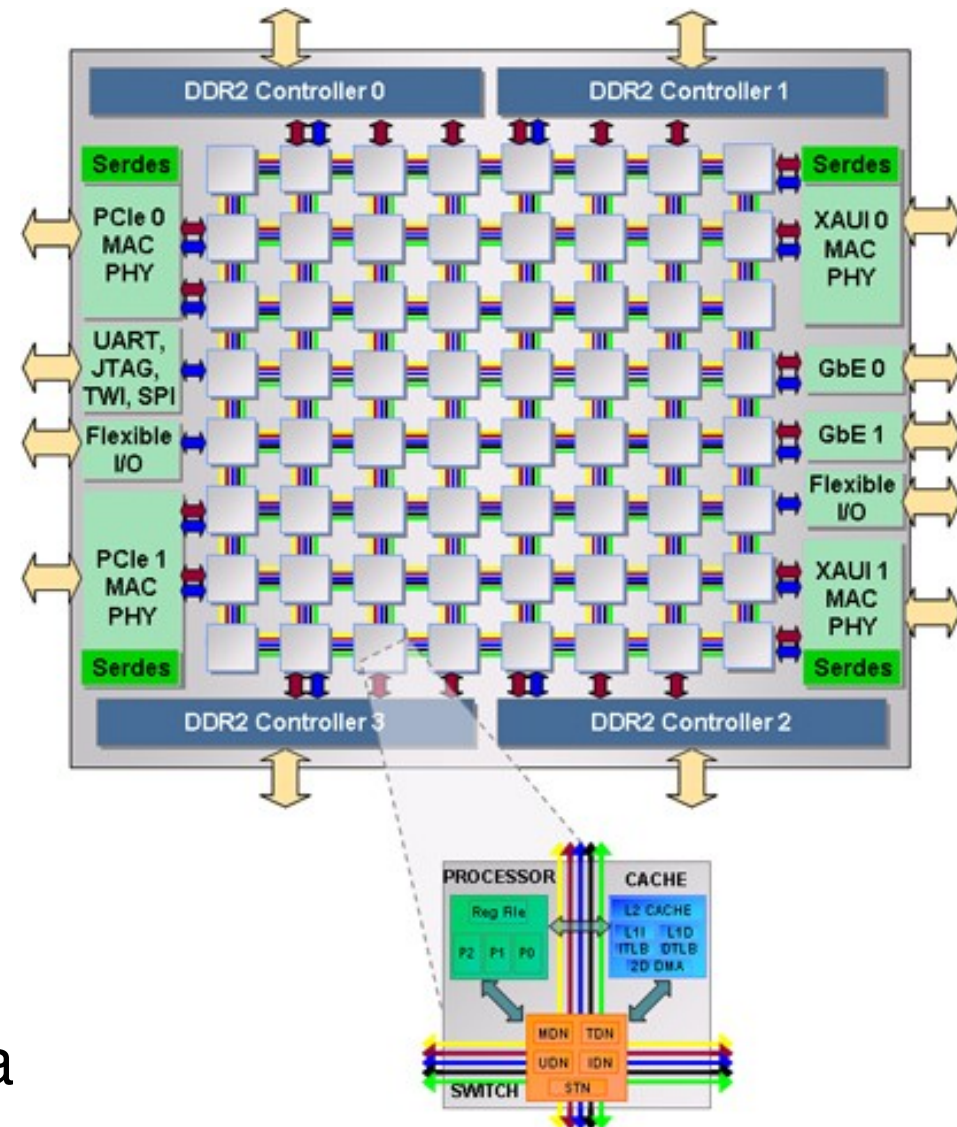
- Many cores: 64 — ≥ 100 .
- Fast and tightly coupled inter-core interconnect.

Some recent examples ...

... a bit of the future – 3b: Processors

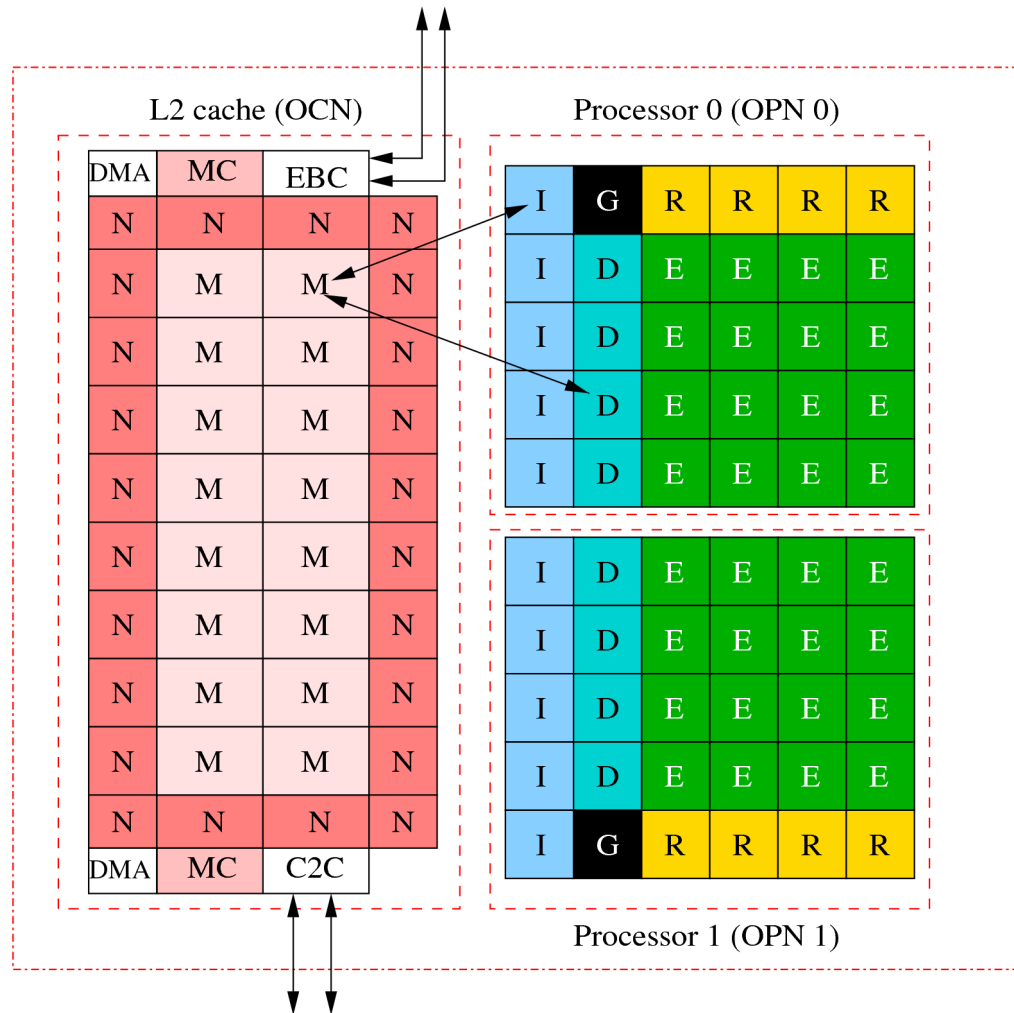
The Tileria Tile64 processor:

- 866 MHz clock
- 1 GB/s bandwidth/link
- 75 kB cache/tile (L1I, L1D, L2)
- Efficient communication library (ILIB)
- Presently targeted for communication and multimedia (32 and 16-bit processing)



... a bit of the future – 3c: Processors

The TRIPS processor:



OPN tiles

- E: Execution tile
- D: L1 Data cache tile
- I: L1 Instruction cache tile
- G: Global control tile
- R: Register tile

OCN tiles

- M: Memory tile
- MC: Memory controller
- N: Network tile
- C2C: Chip-to-chip interface
- DMA: DMA controller
- EBC: External Bus Controller

... a bit of the future – 3d: Processors

TRIPS processor: Execution model, ISA.

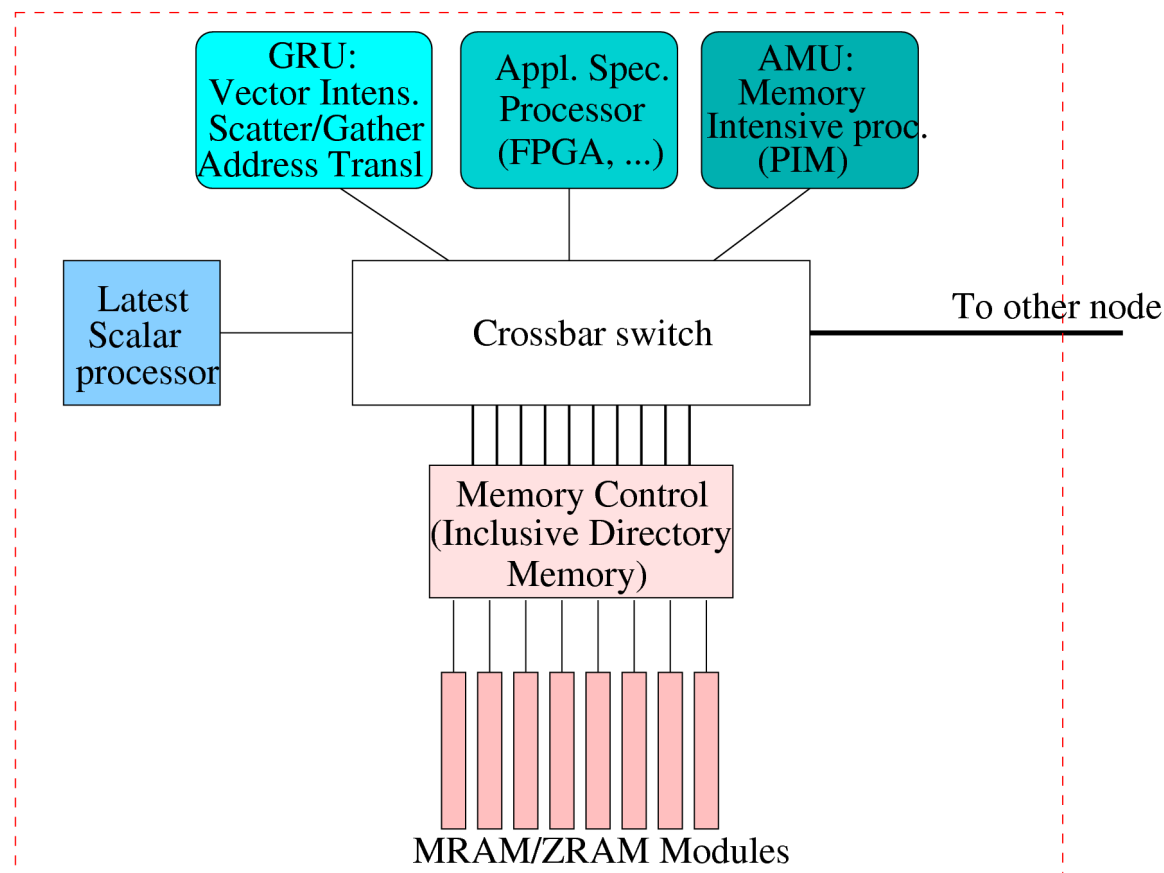
- The processor is able to schedule up to 128 instructions on the 16 execution tiles of a processor.

(2-D Large Instruction Word processor)

- Codes have typically less than 128 instructions in a basic block. So, fusion of basic blocks in **hyperblocks** is attempted.
- Dependent operations are scheduled on the same or nearby tile.

... a bit of the future – 3e: Processors

A generic picture of a near future HPC system:
The notion of “accelerator” becomes somewhat academic in hybrid systems .



Generalised Compute Node

... a bit of the future – 4a: Software

For **real** High Performance Computing we also will have to resort to new programming models/languages to cope with the new facilities that soon will come available. PGAS (**P**artitioned **G**lobal **A**ddress **S**pace) languages are believed to help in that respect.

Present-day languages are UPC and CAF (**C**o-**A**rray **F**ortran, availability is limited).

Missing features in UPC and CAF are awareness of of the **type and location** of memory and support of **asynchronous** memory access.

- Chapel (Cray);
- X10 (IBM);

support such features which can greatly help with scalability and thread control.

... a bit of the future – 4b: Software

Atomic operations are supported:

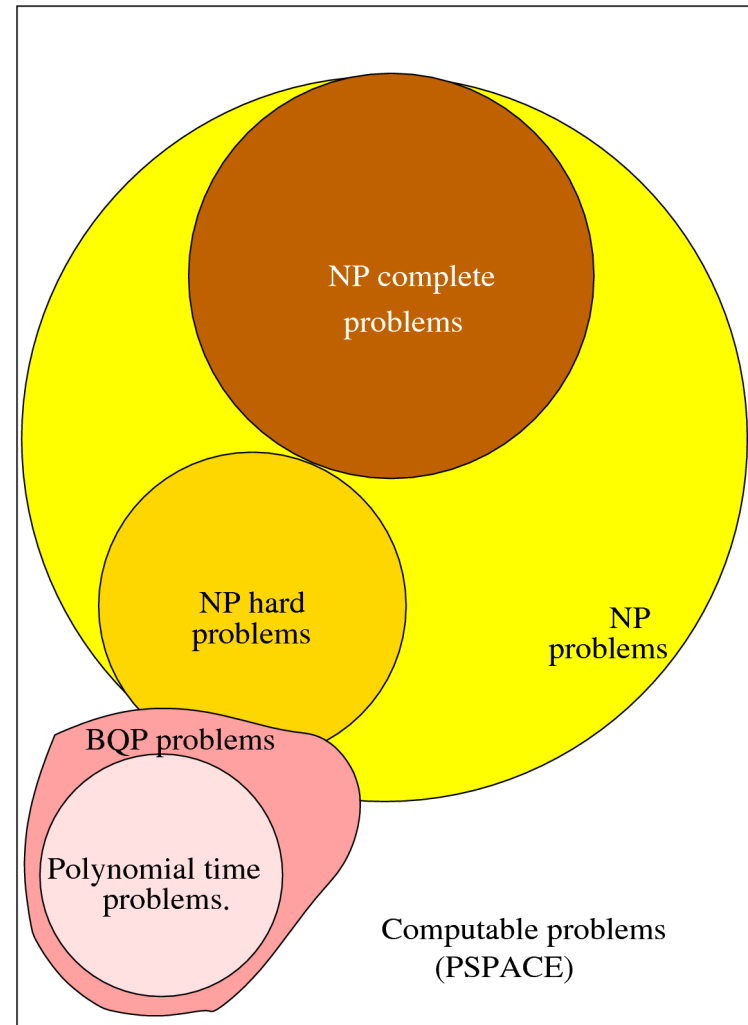
```
const problem_space: domain(1) distributed(Block) = [1..n];
var    a, b: [problem_space] elem_type;
var dot_a_b: elem_type;
    .
    .
    .
dot_a_b = 0.0;
atomic( dot_a_b += a*b );
    .
    .
    .
```

The atomic operations free the programmer from many threading headaches but the efficiency highly depends on hardware memory support. — Also an issue in Transactional Memory.

... a bit of the future – 5a: Speculative

How about quantum computing? Will it save the day with respect to computational speed? **Alas, no.**

1. There is only a limited class of problems that can benefit: some NP problems and some NP-hard problems (Factorising large numbers).
2. Known to date, for about all NP problems $S^{1/2}$ steps are required where S is the number of possible solutions.



The End

Now go forth and multiply
(add, subtract, divide, ...)

Thank you!

