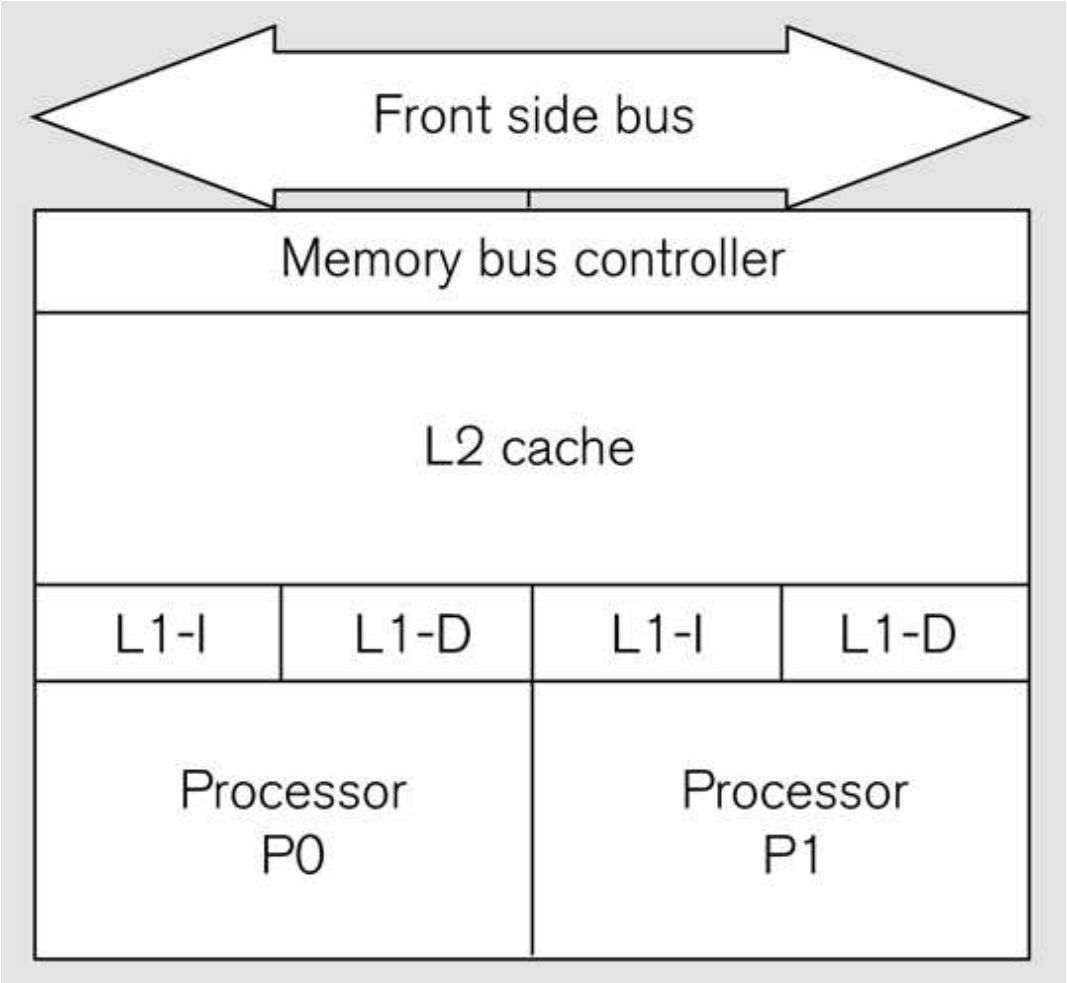
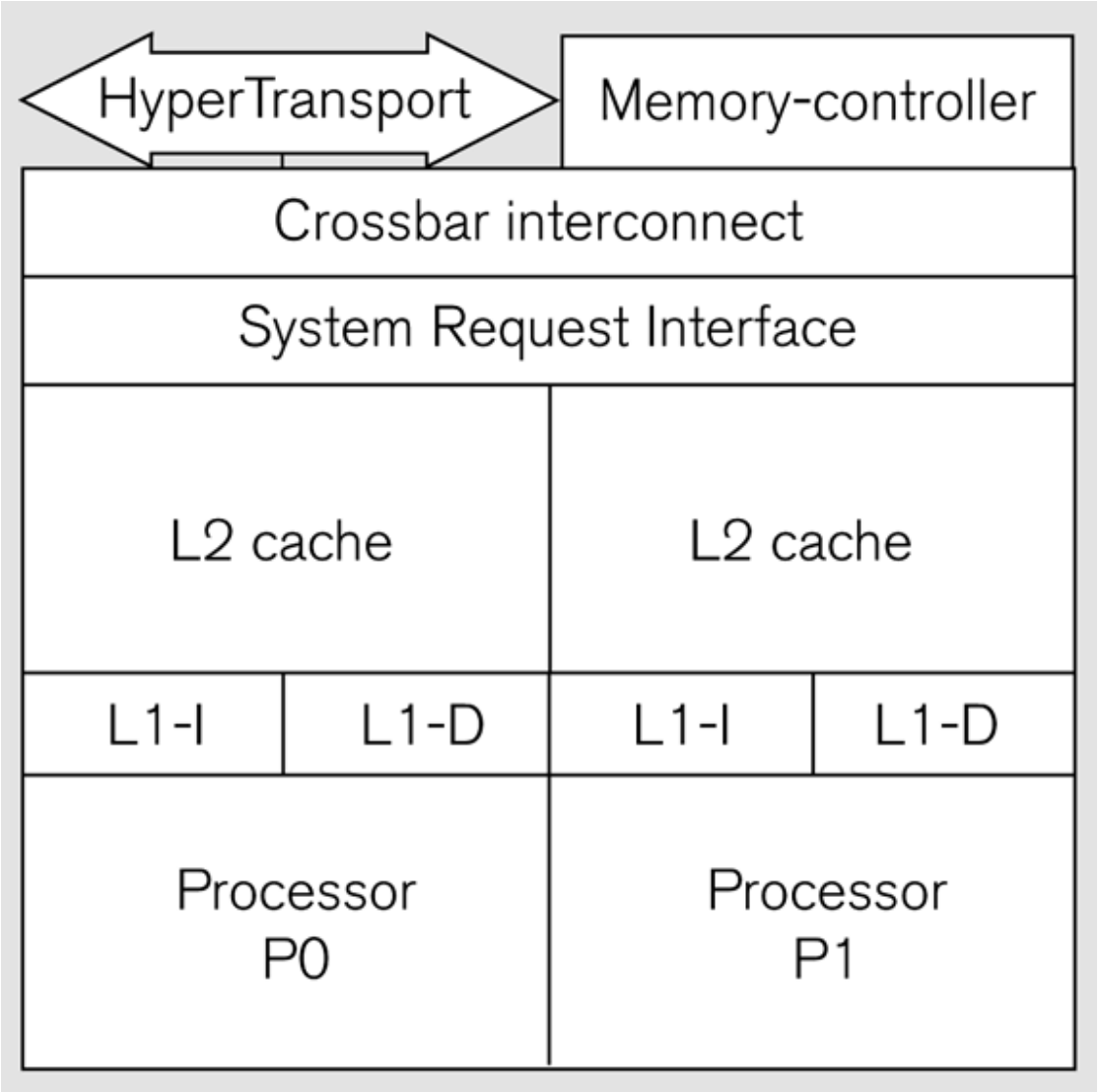
- Examples of parallel computers
- CTA: a model for predicting performance

Intel Core Duo



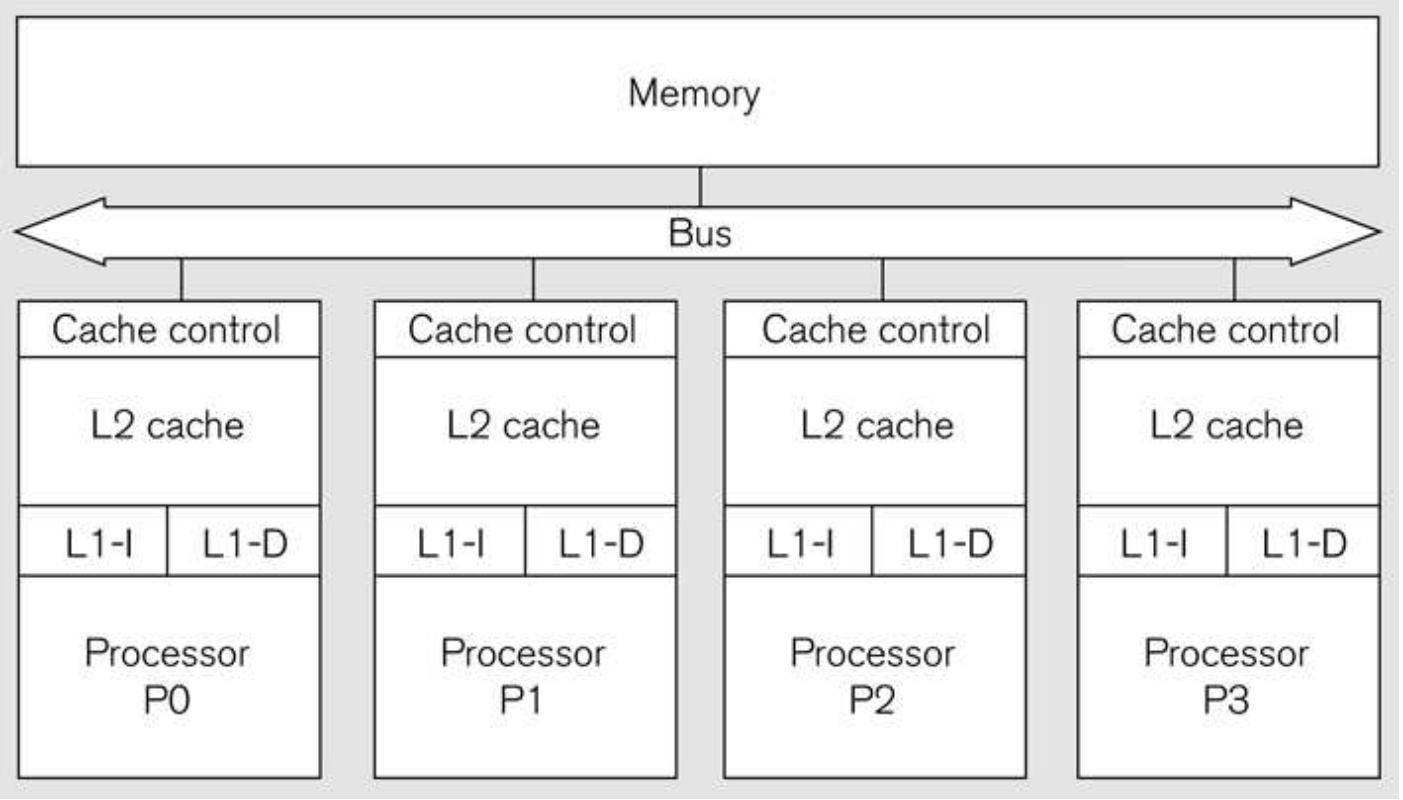
Use: Typical PCs

AMD Dual Core Opteron



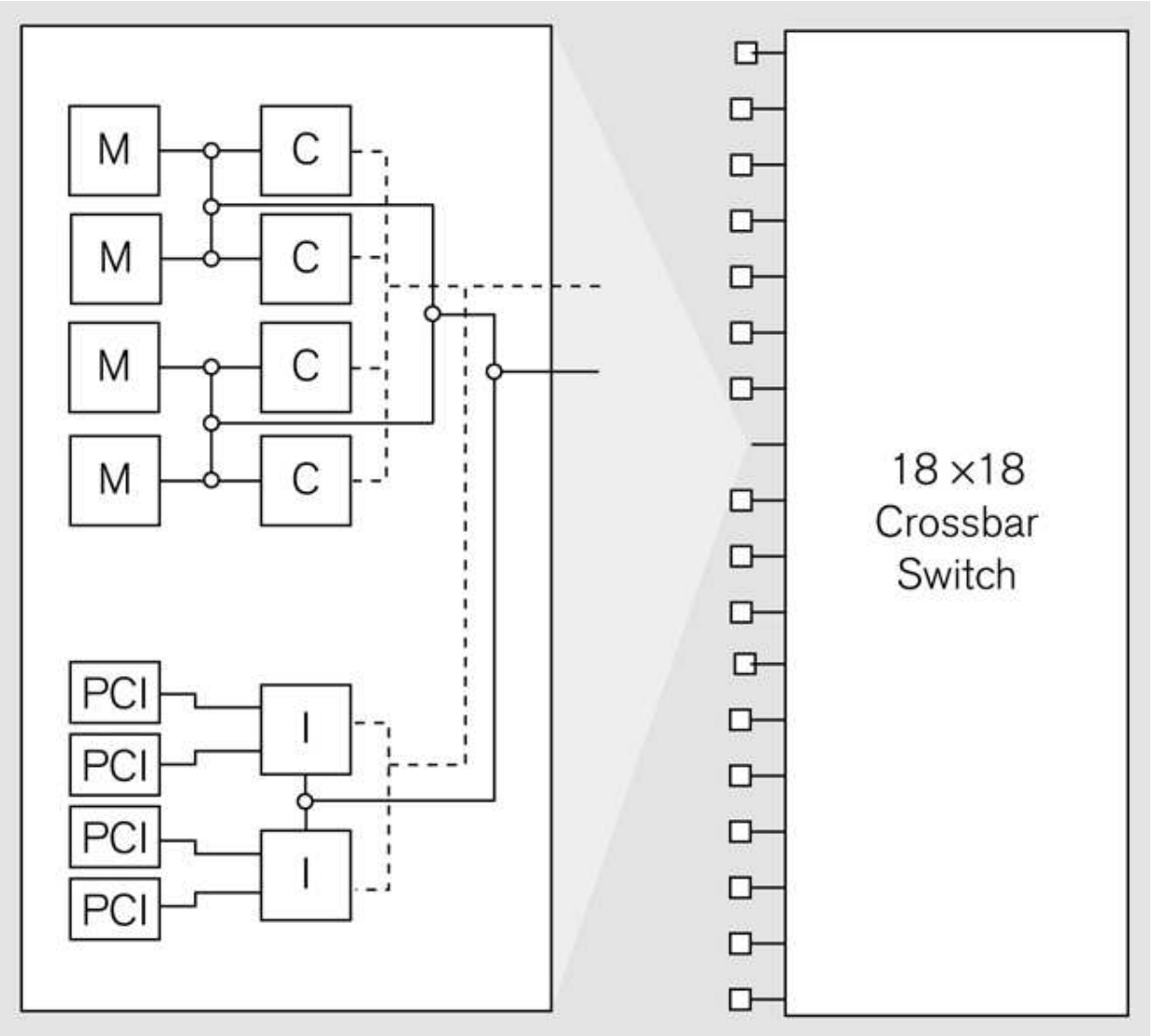
Use: Typical PCs

Generic SMP



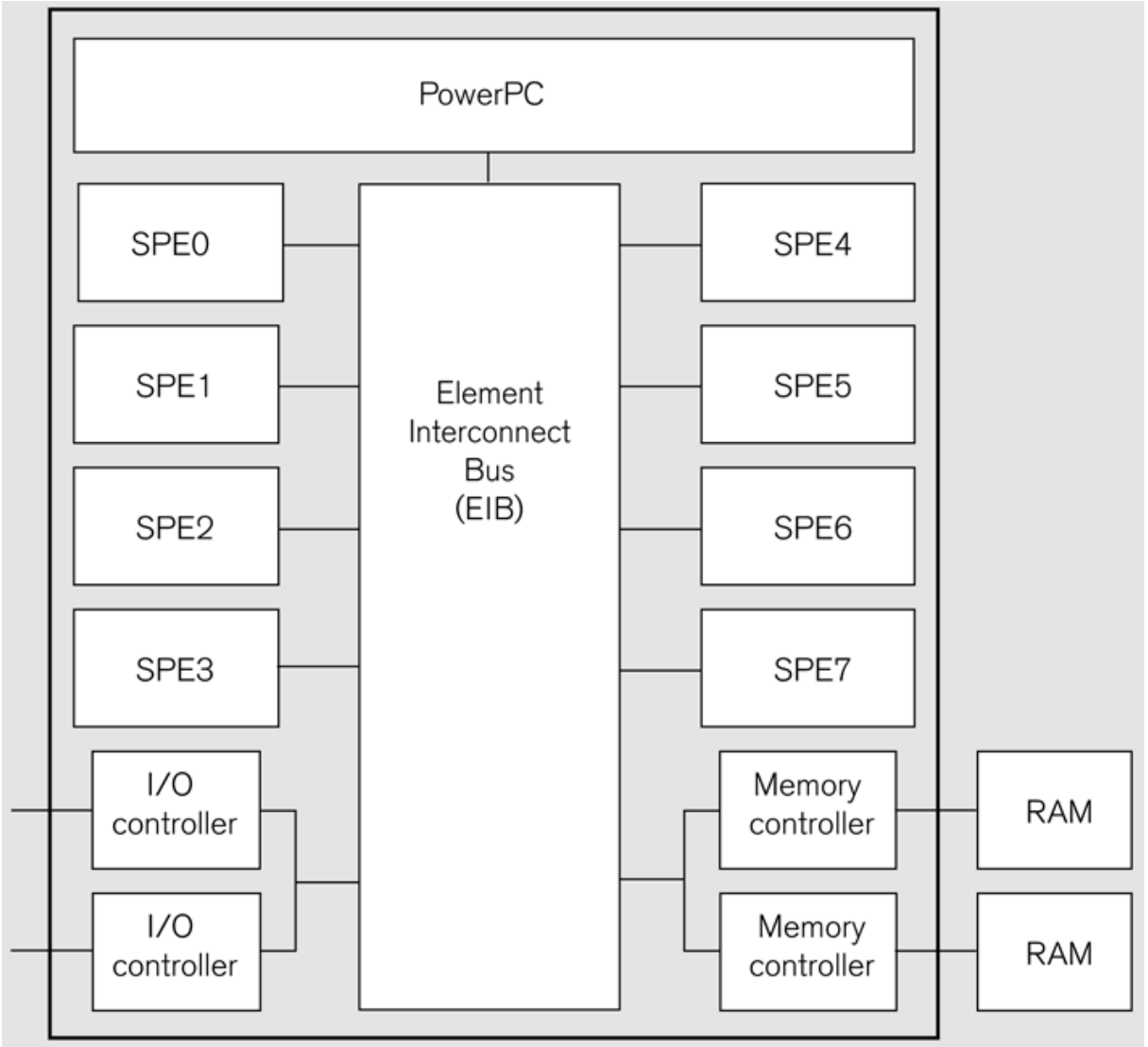
Use: Both multi-core and multi-CPU PCs

Sun Fire E25K



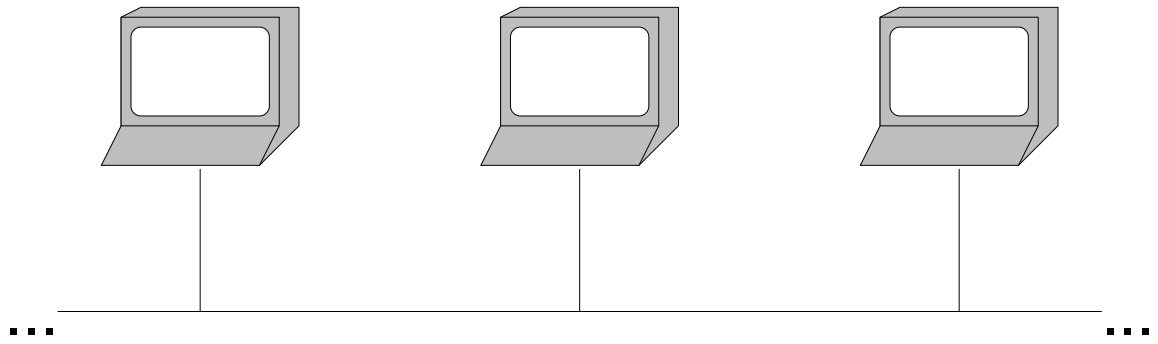
Use: High-end servers from Sun

Cell



Use: PlayStation 3, Roadrunner supercomputer (w/Opterons)

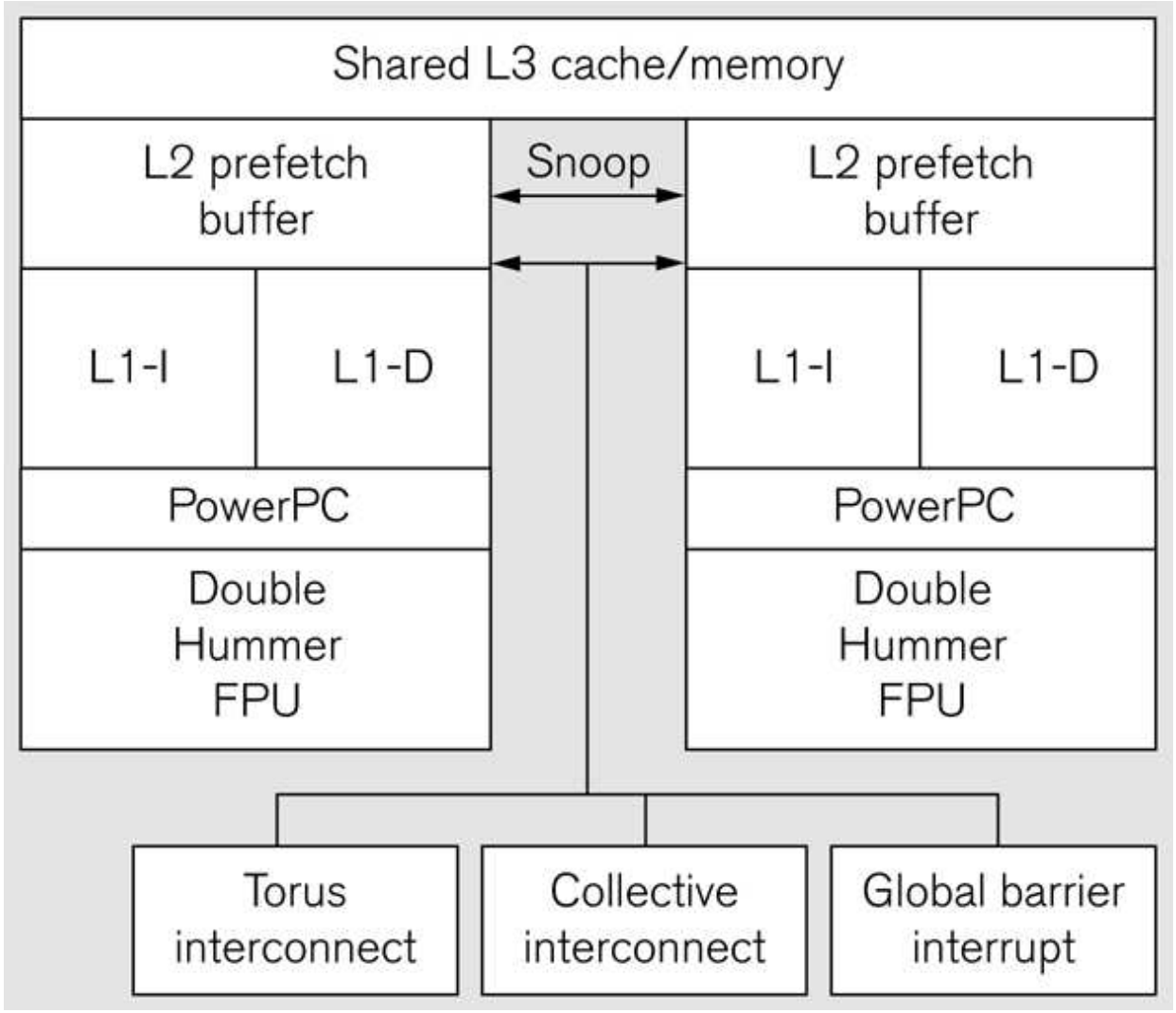
Cluster



Use: Low-cost large-scale parallelism, Emulab

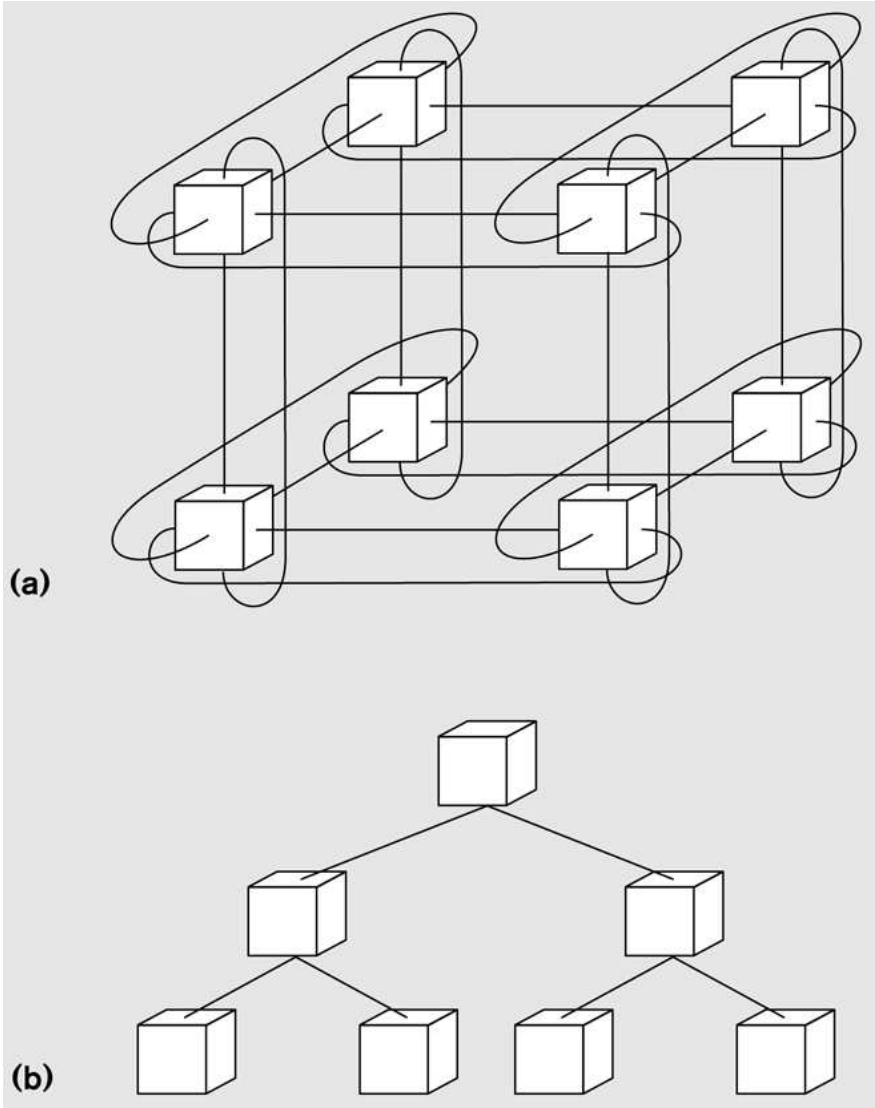
Internet as network ⇒ ***grid computing***

BlueGene/L



Use: Supercomputing

BlueGene/L Networks (2 out of 3)



Spectrum of Machines

- A few processors up to thousands
  - Product roadmaps point to more and more cores
- Shared memory versus distributed memory
  - but always a notion of “here” versus “elsewhere”

$\Rightarrow$ need scalable, portable programs

Expect a ***MIMD*** perspective, mostly ignoring ***SIMD***:

- MMX instructions (width 4 or so)
- GPU instructions (width 64 or so)
- Vector machines like Convex

Models of Computation

Most successful sequential model:

RAM, a.k.a. ***von Neumann***

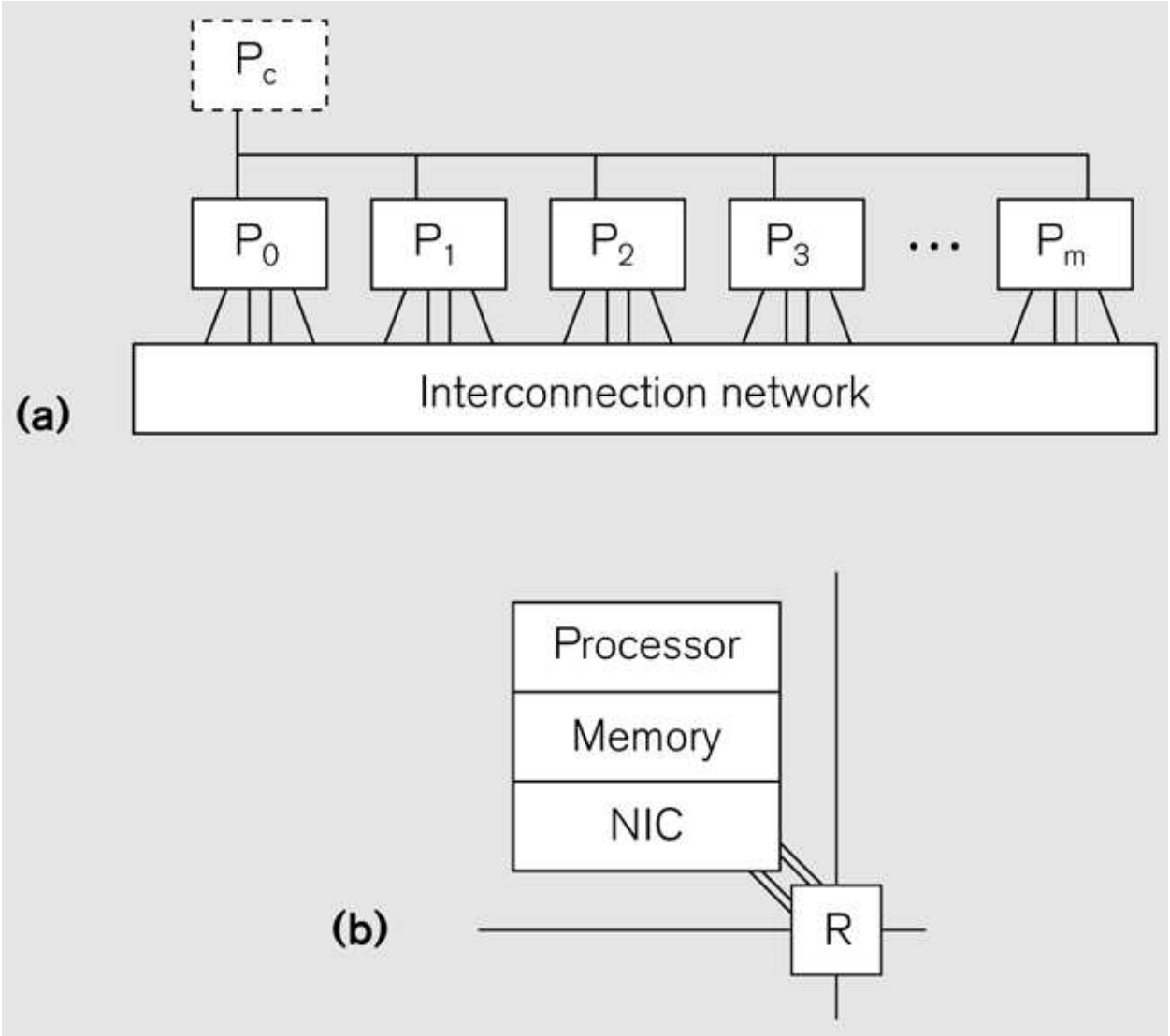
For example, predicts that binary search will be much faster than serial search

[confirm by timing C and Java programs]

Models of Parallelism

- Not a good model: ***PRAM***
  - Assumes the same cost for accessing any memory location
  - Fine for asymptotic lower bounds, misleading for pratice
- A good model: ***CTA***
  - Stands for ***Candidate Type Architecture***
  - Makes useful predictions about real performance

CTA



accessing memory “elsewhere” takes λ times as long as “here”

Measuring Approximate λ

```
static volatile int val;
...

void read_loop(int id)
{
  int j;

  for (j = 0; j < iters; j++)
    result += val;
}
```

32-bit 2 Pentium D: 100
32-bit Core Duo: 100
64-bit 2 Opteron: 40
64-bit Opteron 2-Core: 40
64-bit Athlon 2-Core: 40
64-bit 4 Xeon: 50

```
static volatile int val;
...

void read_loop(int id)
{
  int j;

  for (j = 0; j < iters; j++) {
    asm("mfence");
    result += val;
  }
}

void write_loop(int id)
{
  int j = 0;

  for (j = 0; 1; j++) {
    asm("mfence");
    val = j;
  }
}
```

Estimated λ for Various Architectures

Family	Computer	λ
Chip Multiprocessor	AMD Opteron	100
Multiprocessor	Sun Fire E25K	400-660
Co-processor	Cell	N/A
Cluster	HP BL6000 w/GbE	4160-5120
Supercomputer	BlueGene/L	8960