

# **Thread Level Parallelism**

E. Sanchez

**Politecnico di Torino**  
**Dipartimento di Automatica e Informatica**

# Introduction

**It is unlikely that further evolutions in processor or compiler architecture could achieve significant improvements in the exploitation of the available ILP.**

**On the other side, in many applications there is a significant amount of parallelism available at a higher level (*Thread-Level Parallelism*, or *TLP*).**

# Threads

**A *thread* is a separate process with its own instructions and data.**

**A thread may correspond to**

- **A process within a parallel program consisting of multiple processes, or**
- **An independent process.**

# Exploiting TLP

**There are applications that naturally expose a high degree of TLP (e.g., many server applications, OSs with multiple applications in parallel).**

**In other cases, the software is written in such a way to explicitly express the parallelism (e.g., some embedded applications).**

**In some existing applications it can be relatively difficult to identify and exploit TLP.**

# Multithreading

**In principle, ILP and TLP can be combined:**

- an ILP-oriented processor could use TLP as a source of independent instructions to feed the available functional units as much as possible.**

**Multithreading allows multiple threads to share the functional units of a single processor in an overlapping fashion, duplicating only threads private state.**

# Multithreading (II)

**It requires:**

- **Independently storing the state of each thread (register file, PC, page table, etc.)**
- **Independent access to memory (e.g., through the virtual memory mechanism)**
- **Effective mechanism for switching from one thread to another.**

# Multithreading types

- ***Fine-grained*** multithreading: the switch from one thread to another happens at every clock cycle
- ***Coarse-grained*** multithreading: the switch happens only when an expensive stall arises
- ***Simultaneous multithreading*** (SMT): it mixes fine-grained multithreading and superscalar ideas.

# **Fine-grained multithreading**

**The execution of multiple threads is interleaved (e.g., in a round-robin fashion, skipping any thread which is stalled at a given time).**

**The CPU must be able to efficiently switch between tasks at any clock cycle.**

**-Advantage:**

**When an instruction is stalled, other instructions can be found in other threads, avoiding any performance loss.**

**-Disadvantage:**

**Individual threads could be slowed down by the concurrent execution of other threads.**

# Coarse-grained multithreading

The switch to a new thread is performed only on costly stalls.

**-Advantage**

Thread switch can be less efficient.

**-Disadvantage**

Performance losses from short stalls can not be avoided.

# **Simultaneous Multithreading (SMT)**

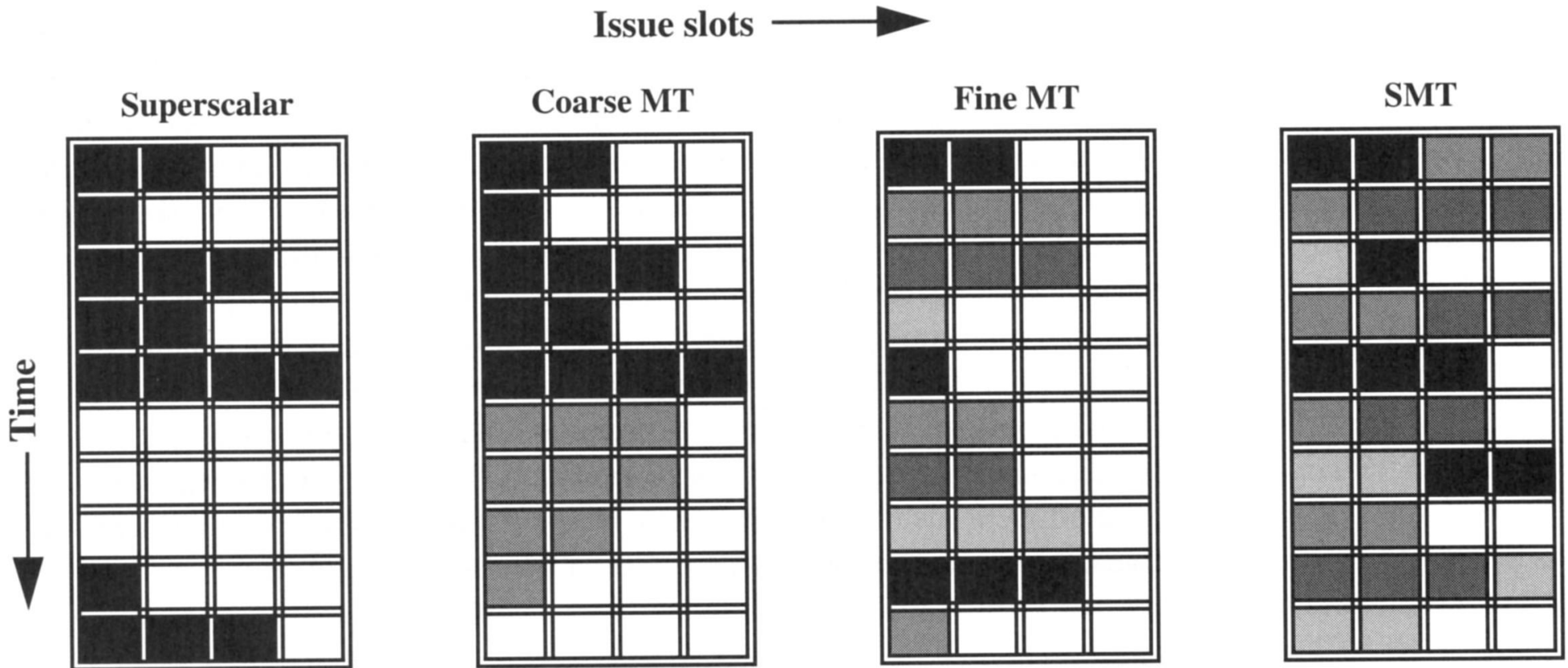
**It is a type of fine-grained multithreading enabling a superscalar processor to exploit ILP and multithreading at the same time.**

**Multiple instructions from independent threads can be issued using dynamic scheduling capabilities to resolve dependencies among them.**

**Instructions from multiple threads can be mixed in the data path thanks to**

- the high number of available registers**
- the renaming mechanism.**

# SMT behavior

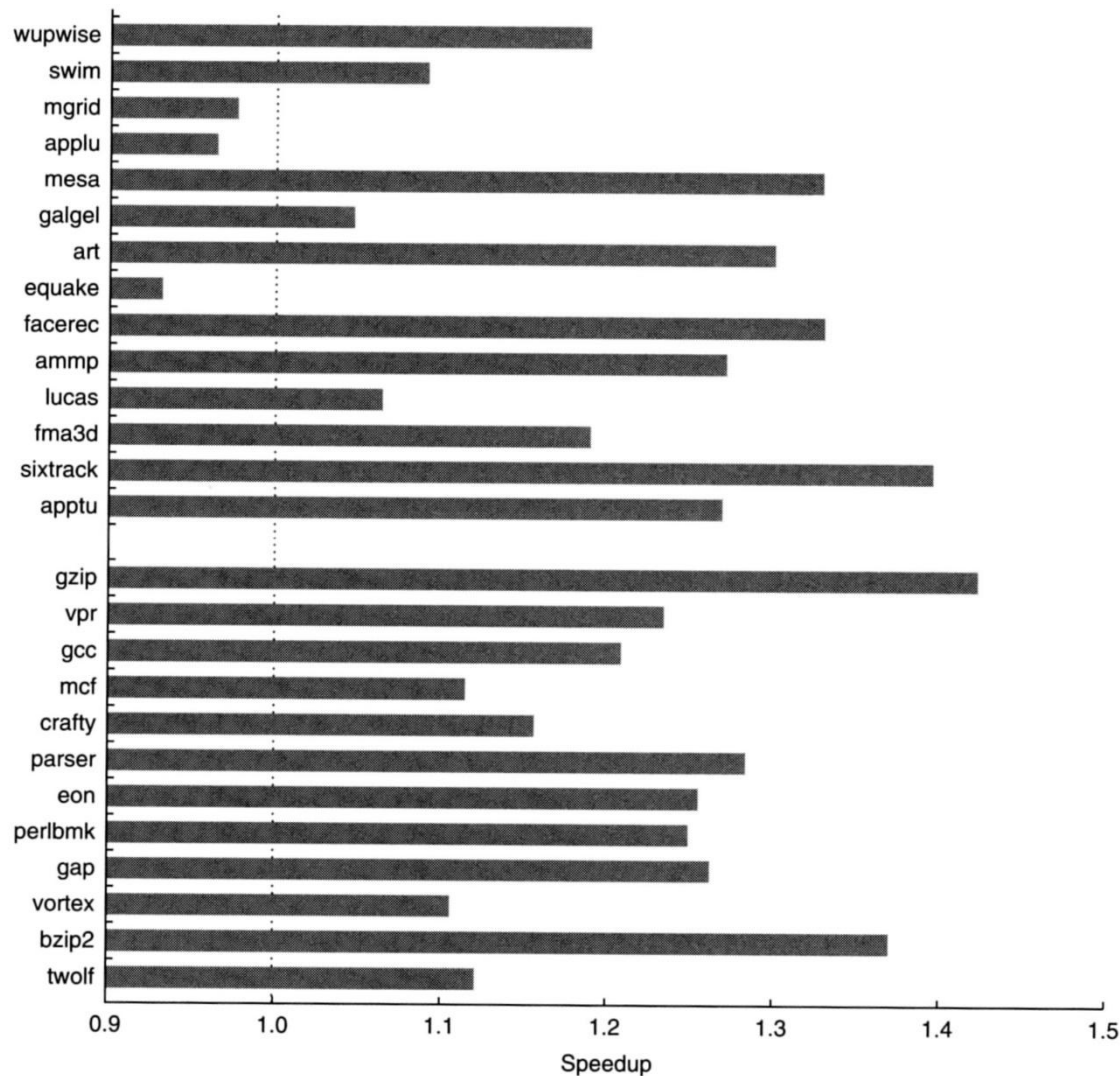


# SMT implementation

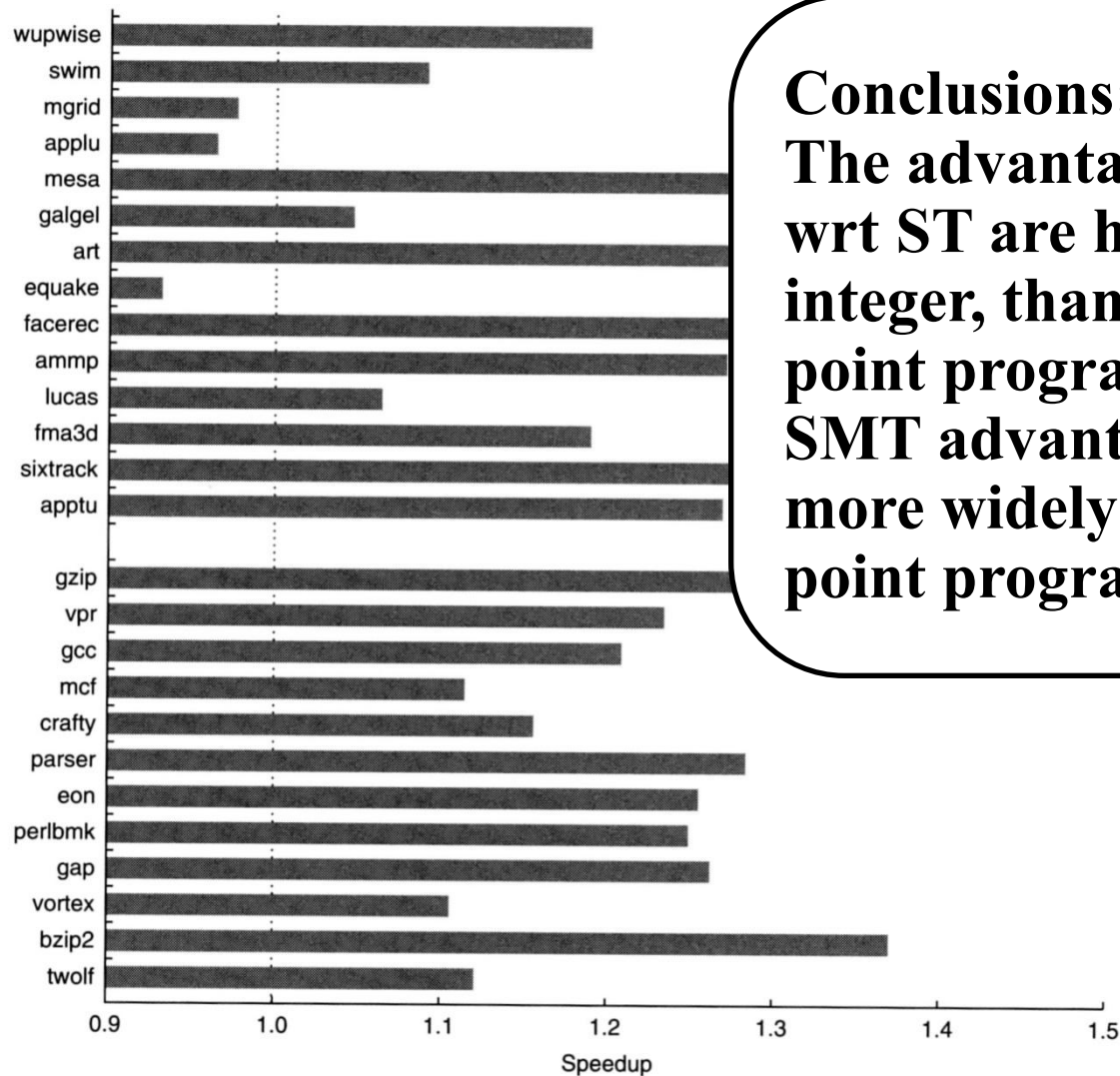
**SMT can be implemented on the top of an out-of-order processor by**

- **Adding a per-thread renaming table**
- **Keeping separate PCs**
- **Allowing instructions from multiple threads to commit (which requires a separate ROB for every thread).**

# SMT performance wrt single –thread (ST)



# SMT performance wrt single –thread (ST)



**Conclusions:**  
The advantages of SMT wrt ST are higher for integer, than for floating point programs. SMT advantages vary more widely for floating point programs.

# Examples

Processor	Microarchitecture	Fetch/ issue/ execute	Func. units	Clock rate (GHz)	Transistors and die size	Power
Intel Pentium 4 Extreme	Speculative dynamically scheduled; deeply pipelined; SMT	3/3/4	7 int. 1 FP	3.8	125M 122 mm <sup>2</sup>	115 W
AMD Athlon 64 FX-57	Speculative dynamically scheduled	3/3/4	6 int. 3 FP	2.8	114M 115 mm <sup>2</sup>	104 W
IBM Power5 1 processor	Speculative dynamically scheduled; SMT; two CPU cores/chip	8/4/8	6 int. 2 FP	1.9	200M 300 mm <sup>2</sup> (estimated)	80 W (estimated)
Intel Itanium 2	EPIC style; primarily statically scheduled	6/5/11	9 int. 2 FP	1.6	592M 423 mm <sup>2</sup>	130 W

# Industry trend

**The current trend is not towards embedding SMT in aggressive speculative processors.**

**Rather, designers are opting to implement multiple CPU cores on a single die with less aggressive support for multiple issue and multithreading.**

# Sun T1

**It has been introduced by Sun in 2005 as a server processor.**

**It focuses on exploiting TLP more than ILP; throughput is achieved through**

- Multithreading**
- Multiple cores.**

**Each T1 processor contains 8 processor cores, each supporting 4 threads.**

# **T1 processor core**

**T1 uses fine-grained multi-threading, switching to a new thread on each clock cycle.**

**Each T1 core implements a 6-stage RISC pipeline (the usual 5 stages, plus one for thread switching).**

**Threads that are waiting due to a pipeline delay or cache miss are bypassed in the scheduling.**

**The processor is idle only when all four threads are idle or stalled.**

**Load and branch instructions incur a 3-cycle delay that can possibly be hidden by other threads.**

**A single set of FP units is shared by the eight cores.**

# T1 architecture

