Thread-Level Speculation on a CMP Can Be Energy Efficient

Jose Renau

University of California at Santa Cruz

http://masc.soe.ucsc.edu

Karin Strauss, Luis Ceze, Wei Liu, Smruti Sarangi, James Tuck, and Josep Torrellas

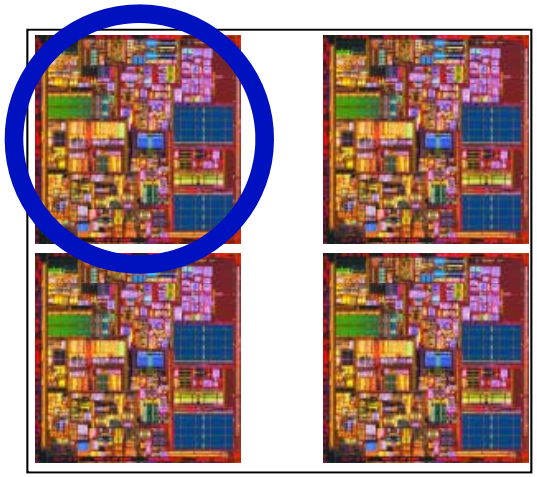
University of Illinois at Urbana-Champaign

http://iacoma.cs.uiuc.edu

Challenges

- Wire delay:
  - Not a single monolithic processor
- Power:
  - Energy-efficient design (simple cores are efficient)
- Complexity:
  - Very large block reuse

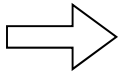
Clock Reach



Chip Multiprocessor with Thread Level Speculation?

Thread Level Speculation (TLS)

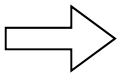
Sequential



```
for(i=0;i<n;i++) {
  X[Y[i]] = X[Z[i]]...
}
```

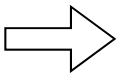
- Compilers cannot parallelize
- TLS: Assume no dependences, hardware verifies

TLS Task A



```
for(i=0;i<n/2;i++) {
  X[Y[i]] = X[Z[i]]...
}
```

TLS Task B

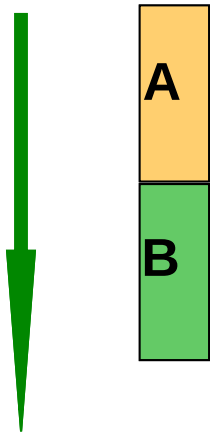


```
for(i=n/2;i<n;i++) {
  X[Y[i]] = X[Z[i]]...
}
```

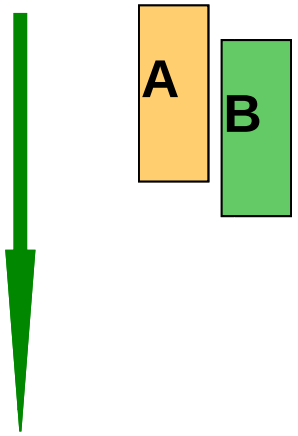
Thread Level Speculation (TLS)

- TLS Hardware:
  - Tracks data accesses at run-time
  - Detects dependence violations
  - Kills and restarts tasks

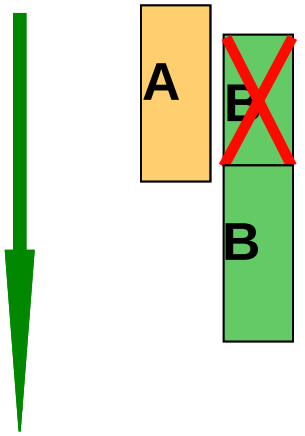
Sequential



TLS (no dep violations)



TLS (dep violation)



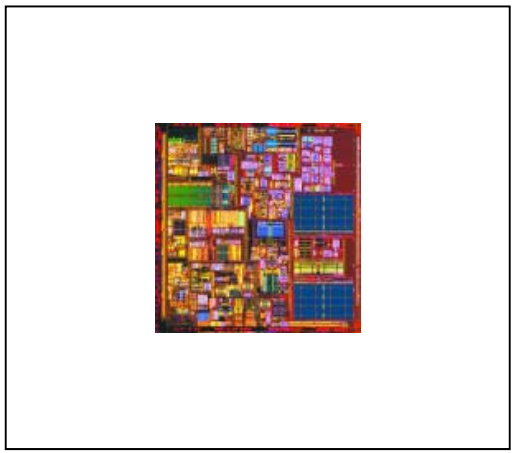
Contributions

Contrary to common wisdom, TLS CMP can be energy-efficient

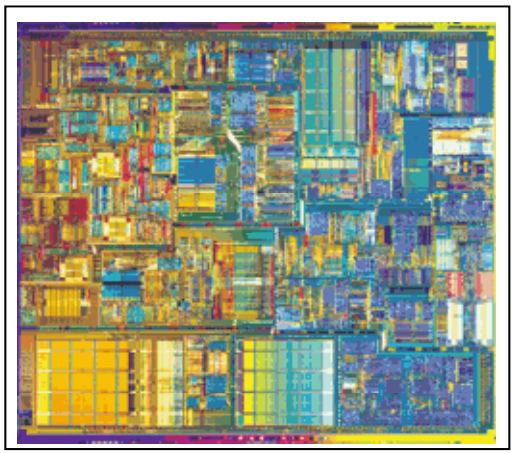
- Identify the sources of energy waste in TLS
- Propose novel energy-centric optimizations
- Design energy-efficient memory hierarchy for TLS CMP

Main Results

TLS: 27% faster and 28% more energy

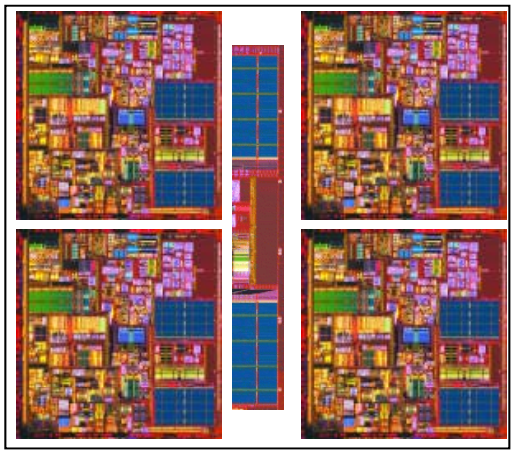


1 CPU 3-issue



1 CPU 6-issue

TLS CMP



4 CPUs 3-issue with TLS

6-issue: 23% faster and 52% more energy

Energy Cost of TLS

Source of energy waste	Why?

Energy Cost of TLS

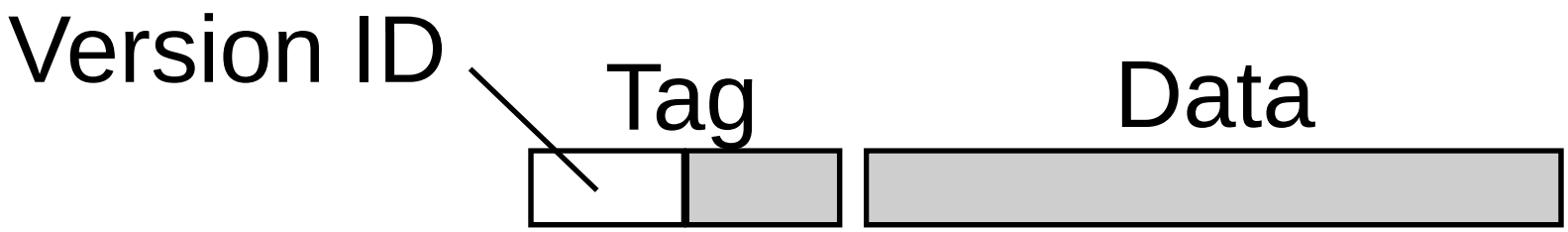
Source of energy waste	Why?
Task squash	Dependence violation

Energy Cost of TLS

Source of energy waste	Why?
Task squash	Dependence violation
Additional storage & logic in memory system	Need to version data

Additional Storage & Logic

- Cache lines are associated to tasks
  - Cache line tags are extended with Version ID



- Messages between caches compare Version IDs

Energy Cost of TLS

Source of energy waste	Why?
Task squash	Dependence violation
Additional storage & logic in memory system	Need to version data
Additional traffic in memory system	TLS aware cache coherence protocol

Additional Traffic

- More cache misses:
  - Cannot displace speculative data
- Handling multiple versions:
  - Example: Find the correct version on cache miss
- Detect data dependence violations across tasks
  - Need extra checks

Energy-Centric Optimizations

- Substantial energy reduction
- Overlooked in performance-centric designs

Source of energy waste	Energy-centric optimization
Task squash	Stall after second restart Energy-aware profiling
Additional storage & logic	Avoid walking the cache
Additional traffic	---
Additional instructions	Energy-aware profiling

Additional Instructions

- Spawn & commit instructions inserted by compiler
- Conventional compiler optimizations not so effective due to code partitioning into tasks
- Live-ins spilling

~15% additional instructions

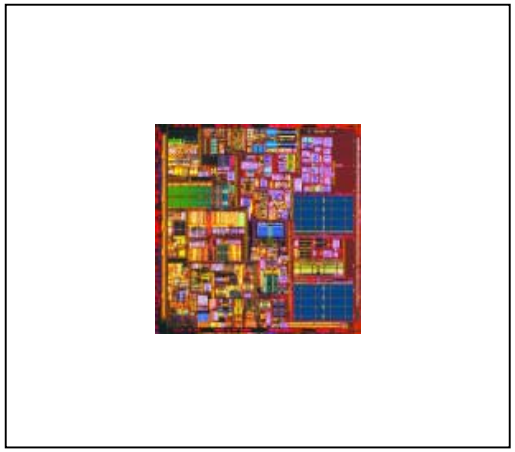
Design Philosophy

- Reduce number of checks
- Reduce cost of each check
- Eliminate low-return work

- Example: Energy-aware profiling
  - Prune tasks that are expected to give minor speedups at high energy cost

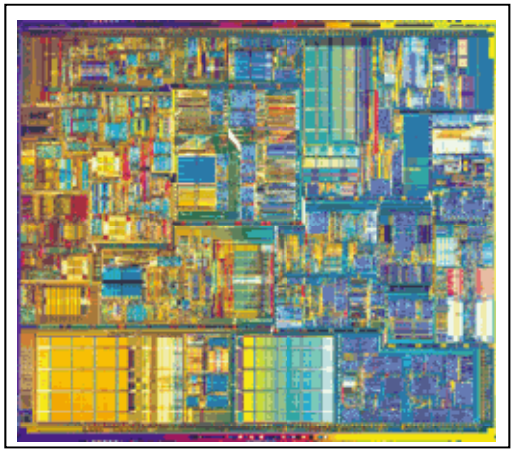
Simulation Environment

Uni-4i



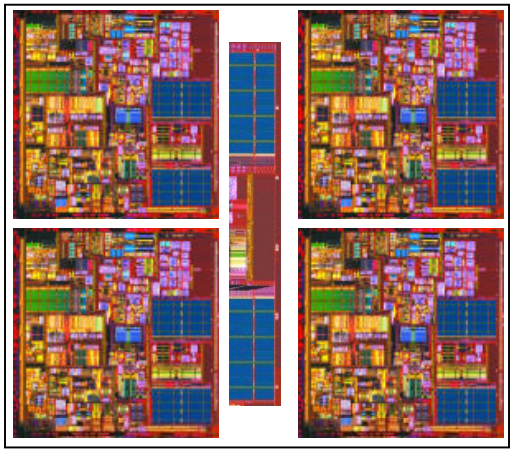
1 CPU 3-issue

Uni-6i



1 CPU 6-issue

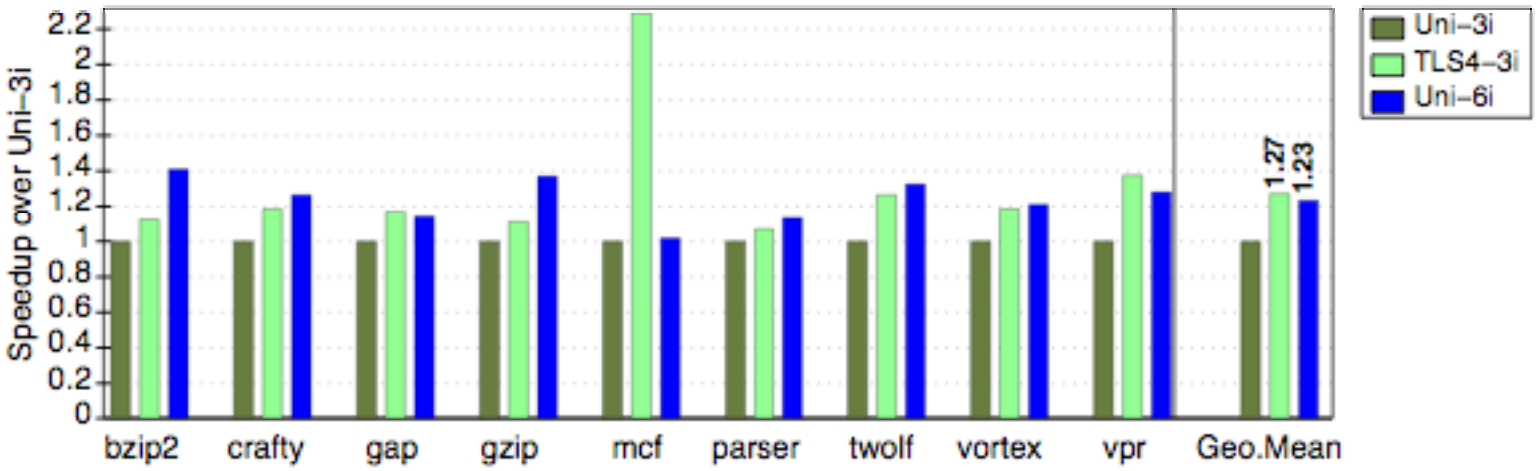
TLS4-3i



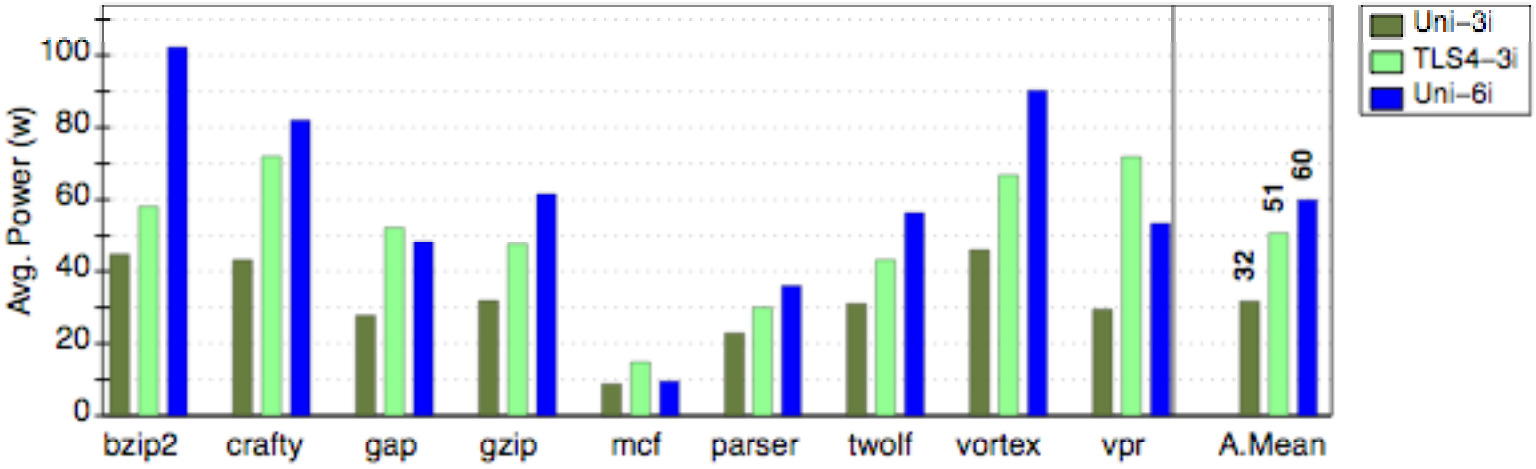
4 CPUs 3-issue with TLS

- 70nm @ 5GHz (same area approx.)
- All processors have same pipeline depth
- 16KB L1 cache (1 cycle slower in TLS due to versioning)
- 1MB L2 cache on-chip

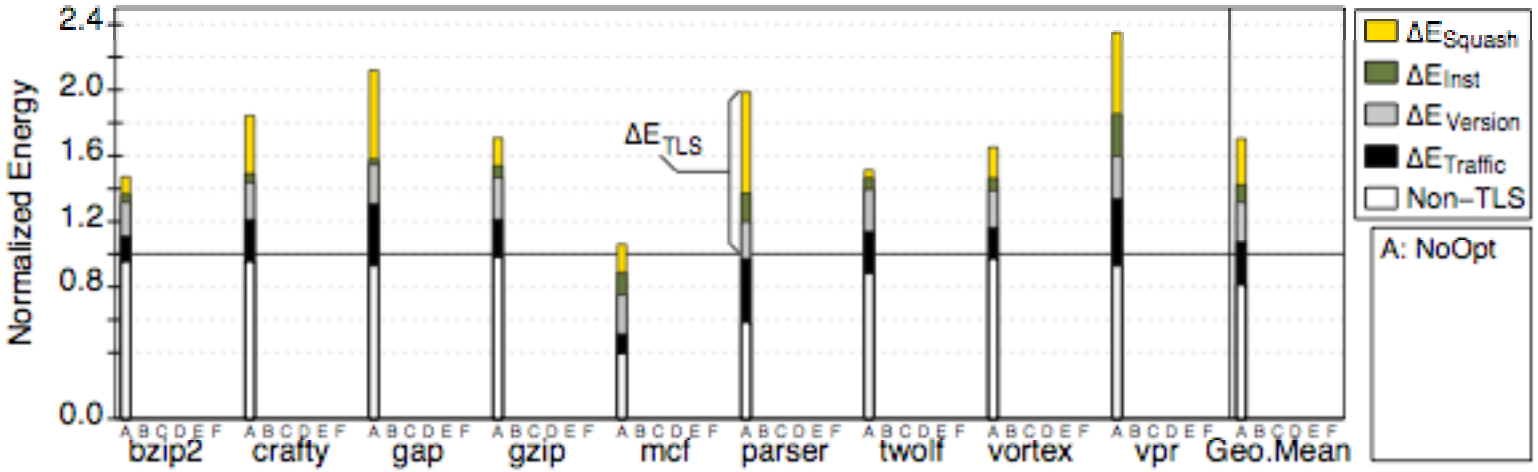
Performance



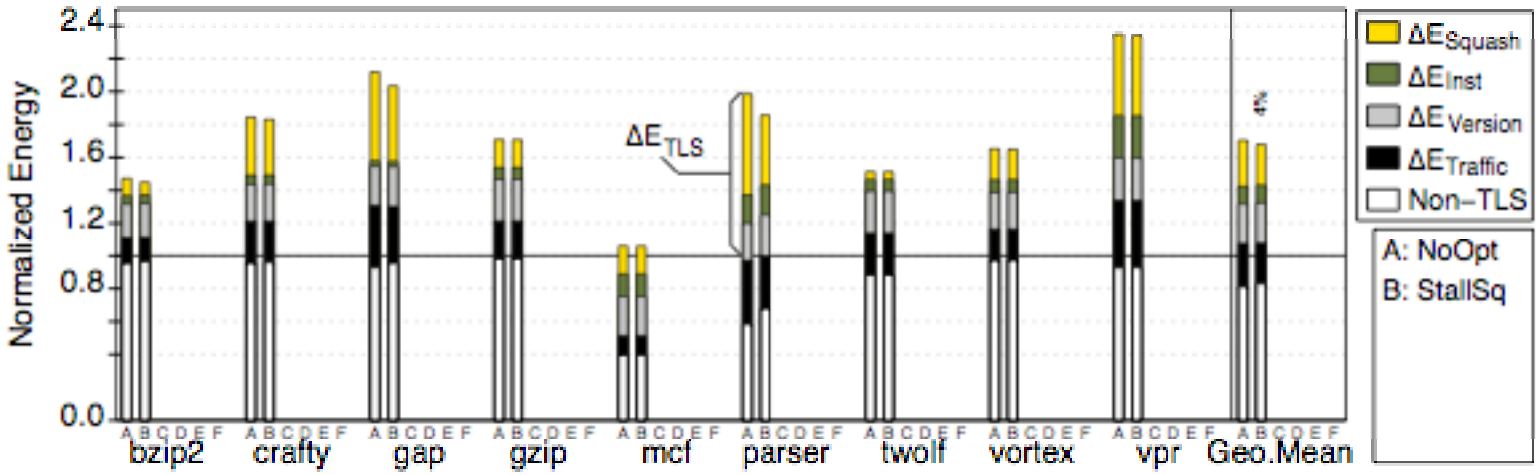
Power


Avg. Power (w)
100
80
60
40
20
0
bzip2
crafty
gap
gzip
mcf
parser
twolf
vortex
vpr
A.Mean
32
51
60
Uni-3i
TLS4-3i
Uni-6i

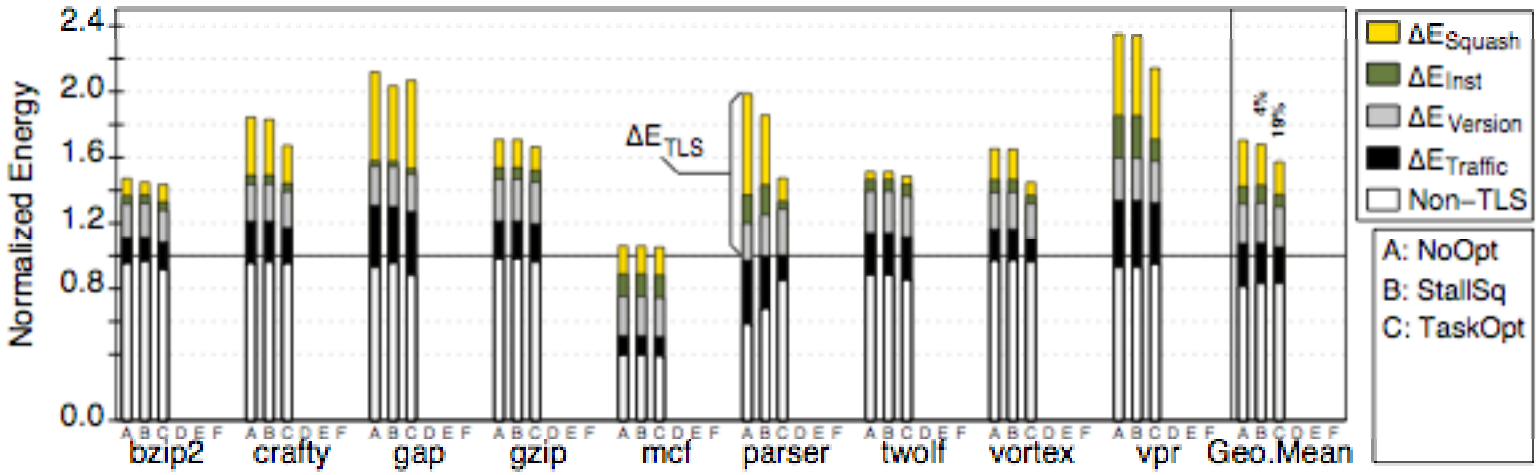
Cost of TLS


Normalized Energy
2.4
2.0
1.6
1.2
0.8
0.0
ΔE_TLS
A B C D E F
bzip2
crafty
gap
gzip
mcf
parser
twolf
vortex
vpr
Geo.Mean
ΔE_Squash
ΔE_Inst
ΔE_Version
ΔE_Traffic
Non–TLS
A: NoOpt

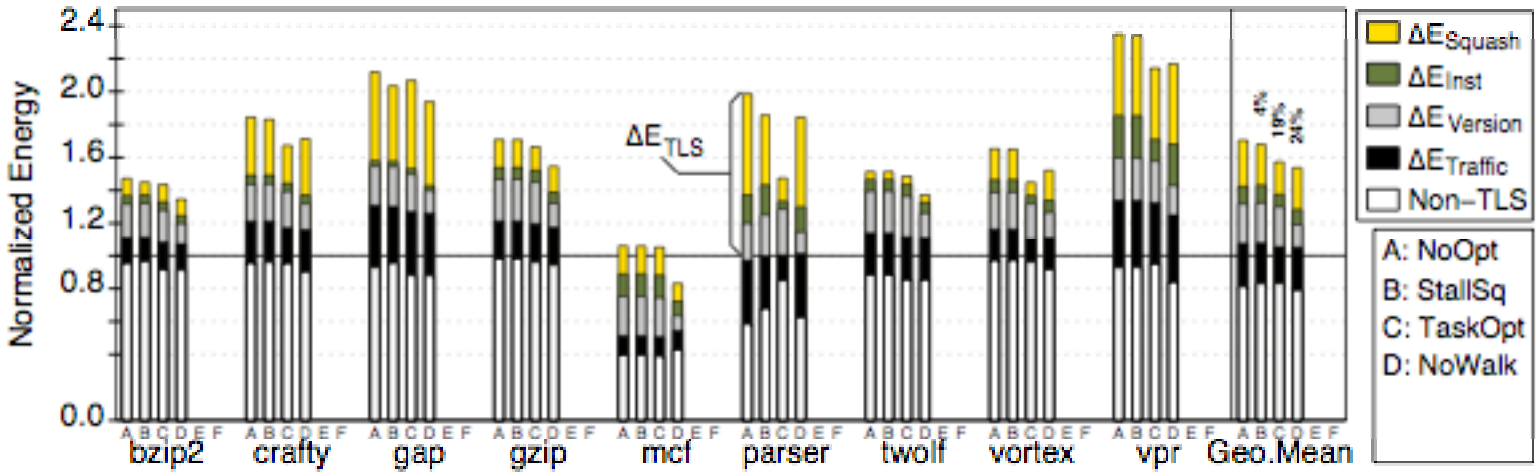
Cost of TLS


Normalized Energy
2.4
2.0
1.6
1.2
0.8
0.0
ΔE_TLS
ΔE_Squash
ΔE_Inst
ΔE_Version
ΔE_Traffic
Non−TLS
A: NoOpt
B: StallSq
A B C D E F
bzip2
crafty
gap
gzip
mcf
parser
twolf
vortex
vpr
Geo.Mean

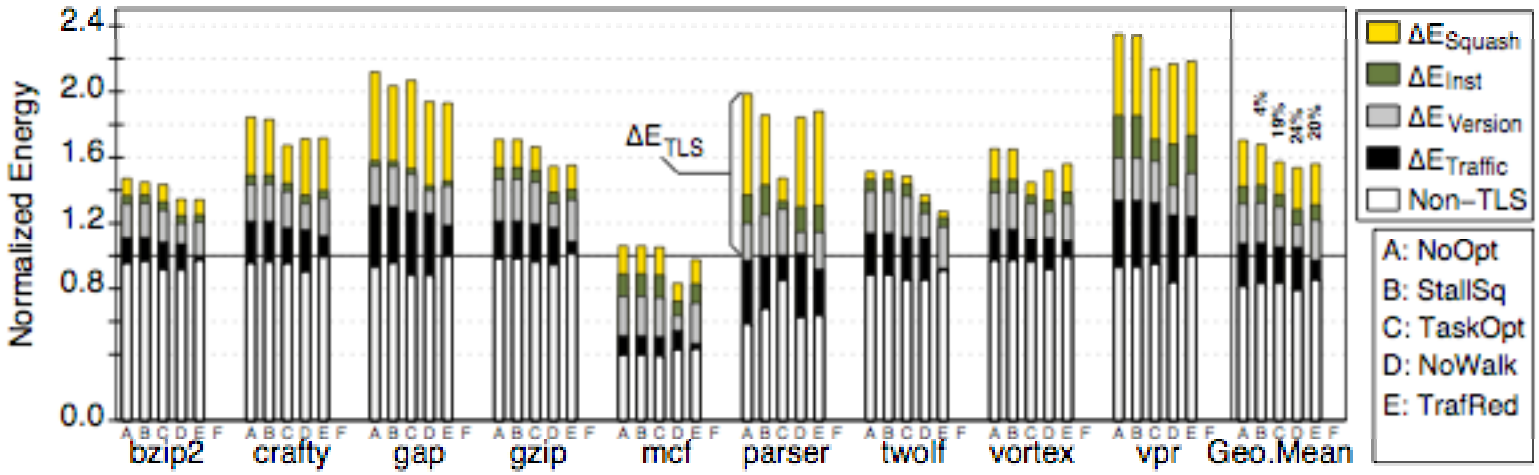
Cost of TLS


Normalized Energy
2.4
2.0
1.6
1.2
0.8
0.0
ΔE_TLS
A B C D E F
bzip2
crafty
gap
gzip
mcf
parser
twolf
vortex
vpr
Geo.Mean
ΔE_Squash
ΔE_Inst
ΔE_Version
ΔE_Traffic
Non–TLS
A: NoOpt
B: StallSq
C: TaskOpt

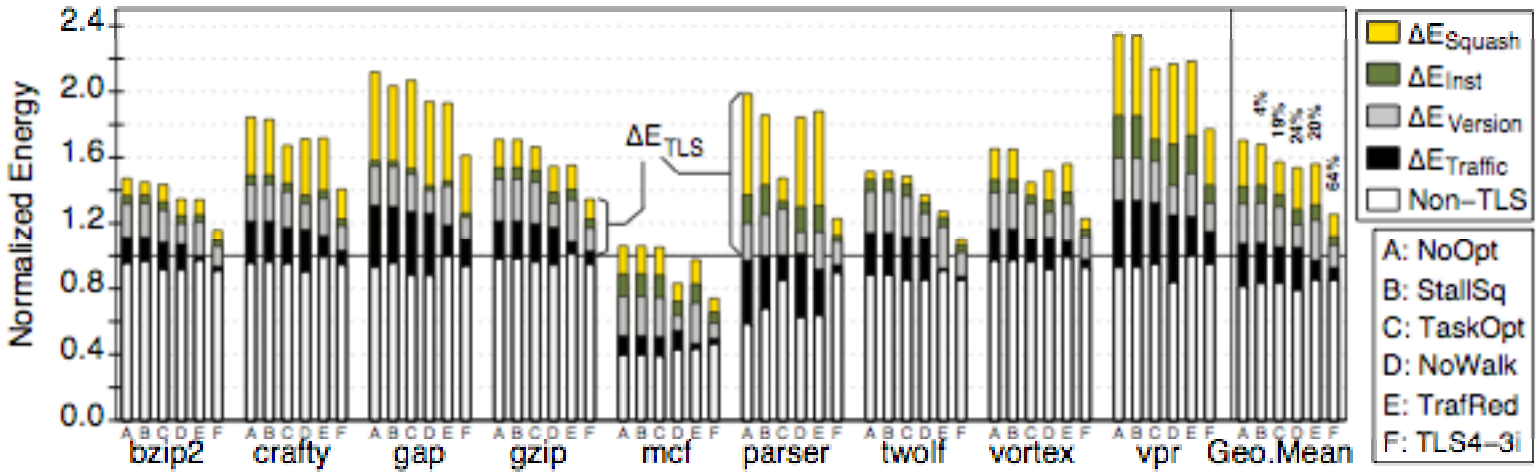
Cost of TLS


Normalized Energy
2.4
2.0
1.6
1.2
0.8
0.0
ΔE_TLS
4%
19%
24%
ΔE_Squash
ΔE_Inst
ΔE_Version
ΔE_Traffic
Non−TLS
A: NoOpt
B: StallSq
C: TaskOpt
D: NoWalk
A B C D E F
bzip2
crafty
gap
gzip
mcf
parser
twolf
vortex
vpr
Geo.Mean

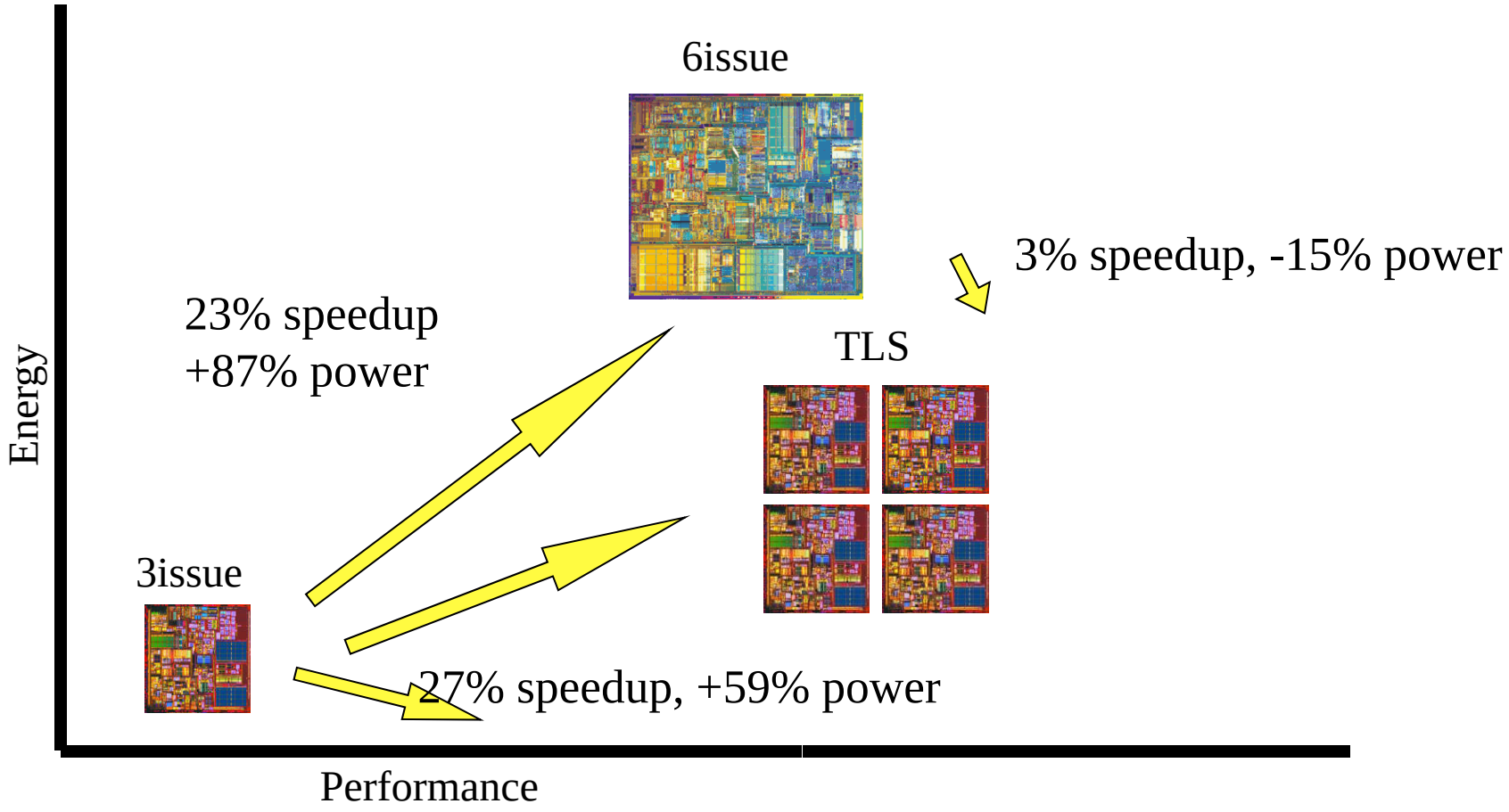
Cost of TLS


Normalized Energy
2.4
2.0
1.6
1.2
0.8
0.0
ΔE_TLS
4%
19%
24%
20%
A B C D E F
bzip2
crafty
gap
gzip
mcf
parser
twolf
vortex
vpr
Geo.Mean
ΔE_Squash
ΔE_Inst
ΔE_Version
ΔE_Traffic
Non-TLS
A: NoOpt
B: StallSq
C: TaskOpt
D: NoWalk
E: TrafRed

Cost of TLS


Normalized Energy
2.4
2.0
1.6
1.2
0.8
0.4
0.0
ΔE_TLS
4%
19%
24%
20%
64%
ΔE_Squash
ΔE_Inst
ΔE_Version
ΔE_Traffic
Non−TLS
A: NoOpt
B: StallSq
C: TaskOpt
D: NoWalk
E: TrafRed
F: TLS4−3i
A B C D E F
bzip2
crafty
gap
gzip
mcf
parser
twolf
vortex
vpr
Geo.Mean

Conclusions: TLS Power is Promising



Results for single thread applications (SPECint2000)

Quesions?

Jose Renau

University of California at Santa Cruz

http://masc.soe.ucsc.edu

Karin Strauss, Luis Ceze, Wei Liu, Smruti Sarangi, James Tuck, and Josep Torrellas

University of Illinois at Urbana-Champaign

http://iacoma.cs.uiuc.edu

Backup Slides

Processor

TLS CMP with four 3-issue cores (*TLS4-3i*)						
Processor						
Frequency: **5.0 GHz** @ 70 nm				Fetch/issue/comm width: 6/**3**/3		
Branch penalty: 13 cyc (min)				I-window/ROB size: 68/126		
RAS: 32 entries				Int/FP registers: 90/68		
BTB: 2K entries, 2-way assoc.				LdSt/Int/FP units: 1/2/1		
Branch predictor (spec. update):				Ld/St queue entries: 48/42		
bimodal size: 16K entries				TaskHolders/processor: 8		
gshare-11 size: 16K entries				TaskHolder access time, energy: 1 cyc, 0.25nJ		
Cache	D-L1	VC	L2		D-L1	VC
Size:	16KB	4KB	1MB	LID Table:		
RT:	**3 cyc**	8 cyc	10 cyc	entries/ports:	64/2	32/1
Assoc:	4-way	4-way	8-way	acc time/energy:	1cyc/0.11nJ	1cyc/0.07nJ
Line size:	64B	64B	64B			
Ports:	1	1	1	Latency from spawn to new thread: 14 cyc		
Pend ld/st:	16	64	64			

6-issue superscalar chip (*Uni-6i*)			
Processor			
Frequency: **5.0 GHz** @ 70 nm		Fetch/issue/comm width: 6/**6**/6	
Branch penalty: 13 cyc (min)		I-window/ROB size: 104/204	
RAS: 32 entries		Int/FP registers: 132/104	
BTB: 2K entries, 2-way assoc.		LdSt/Int/FP units: 2/4/2	
Branch predictor (spec. update):		Ld/St queue entries: 66/54	
bimodal size: 16K entries			
gshare-11 size: 16K entries			
Cache	D-L1	L2	
Size:	16KB	1MB	
RT:	**2 cyc**	10 cyc	
Assoc:	4-way	8-way	
Line size:	64B	64B	
Ports:	2	1	
Pend ld/st:	16	64	

I-L1	Size: 16KB; RT: 2 cyc; assoc: 2-way; line size: 64B; ports: 1
Bus & Memory	FSB frequency: 533MHz; FSB width: 128bit; memory: DDR-2; DRAM bandwidth: 8.528GB/s; memory RT: 98ns
Profiling parameters	R_{squash}: 0.8; T_i: 1; T_m: 200 cyc; E_i: 8pJ; T_{perf}: 90 cyc; $Size_{energy}$: 45; $Size_{perf}$: 30; $Hoist_{energy}$: 120; $Hoist_{perf}$: 110