Theory vs. Practice

Until it’s in hardware, you’re just kidding yourself

Kung Factors

H.T. Kung, then of CMU, after building two generations of the Warp systolic array, concluded that simulations are always optimistic. He went on to list the many factors contributing to loss of theoretical performance when an architecture transitions to hardware implementation. These are remembered in the community, not so much for their specifics, but in general terms, as the Kung Factors. Those who have built hardware know them well. Those who have not are usually deluding themselves.

Today we look at two academic projects that took on the Kung Factors.

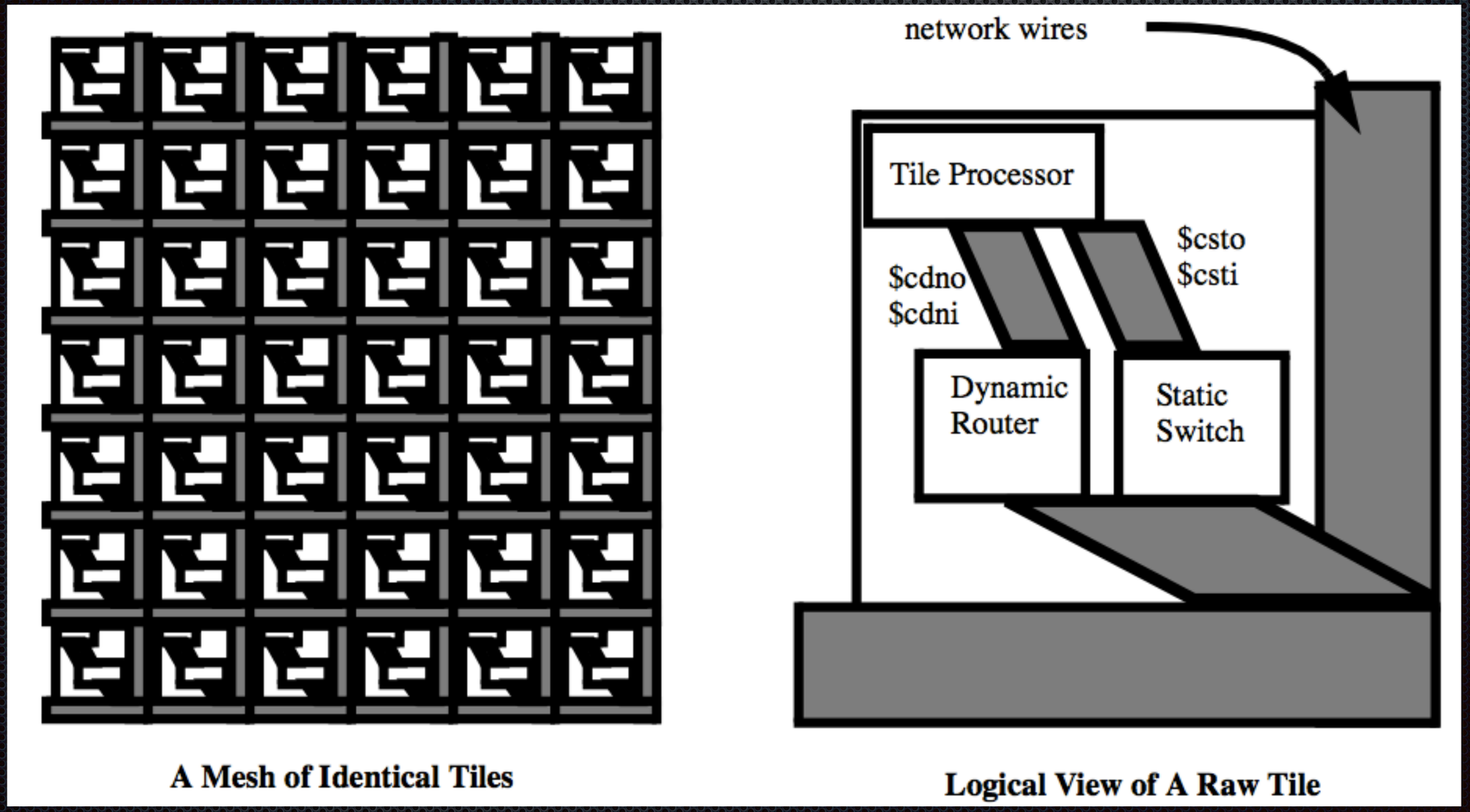
Anant Agarwal ISCA 2004

Evaluation of the Raw Microprocessor

The Early Vision (1995)

- From Michael Taylor M.S. Thesis (1999)
  - Fabric of FPGAs
  - FPGA’s support synchronization of parallel elements
  - Best for bit and byte values
  - In contrast, processors are optimized for datapath
  - Processors deal well with cold code (90%)
  - Compiling hardware is unacceptably slow

A New Vision



A Mesh of Identical Tiles

Logical View of A Raw Tile

Each Tile

- 32-bit MIPS ISA
- 32 KB I-mem, 32 KB D-mem
- FIFO queue register interface to switch processor

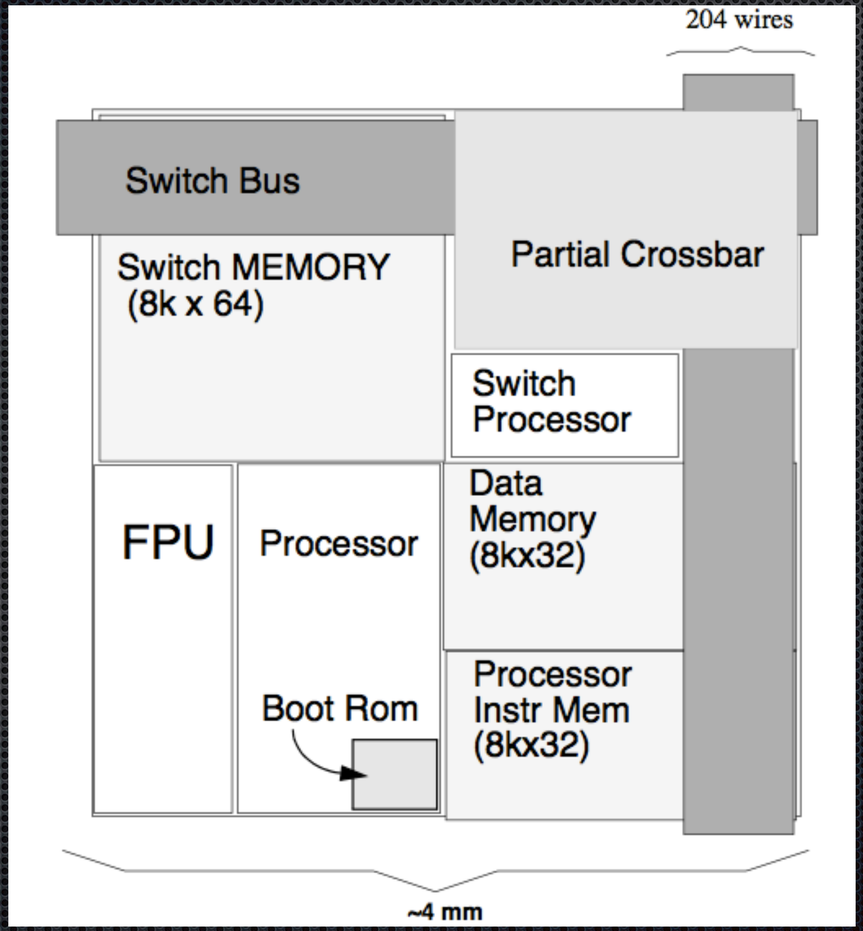
Each Switch

- Simplified processor with data movement capability
- 8K instruction I-mem
- Small set of registers
- Moves are from ports to registers

Chip Process

- 160 nm 6-metal CMOS
- Up to 24m gates, but far fewer due to wires
- Scale back total area to 16m to be safe
- For 16 tiles, 1m gates per tile available
- Dedicate half to SRAM (32k words) and half to CPU
- But Kung Factor says fewer will be usable

BotE Floorplan for a Tile



Two Networks

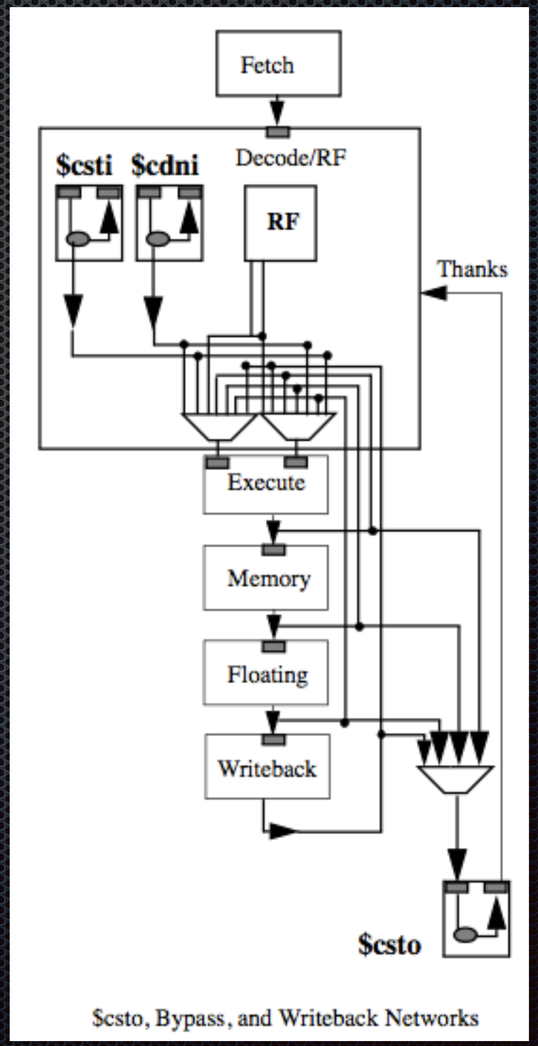
- Static -- configured at compile time, sends individual words with no header overhead
  - Initially assumed programmed I/O where compiled code assumed cycle counts between tiles would be identical
  - Due to IF statement variability and cache misses, needed to add handshaking and FIFO buffers
  - Explored multiple alternatives to switch processor

Two Networks

- Static network has to be configured and managed
- Dynamic -- for sending messages that control the setup of the static network, interrupts, I/O, sync, etc.
  - Wormhole routed, packet-based router
  - Interface is similar to static, but different scheduler

Processor Design

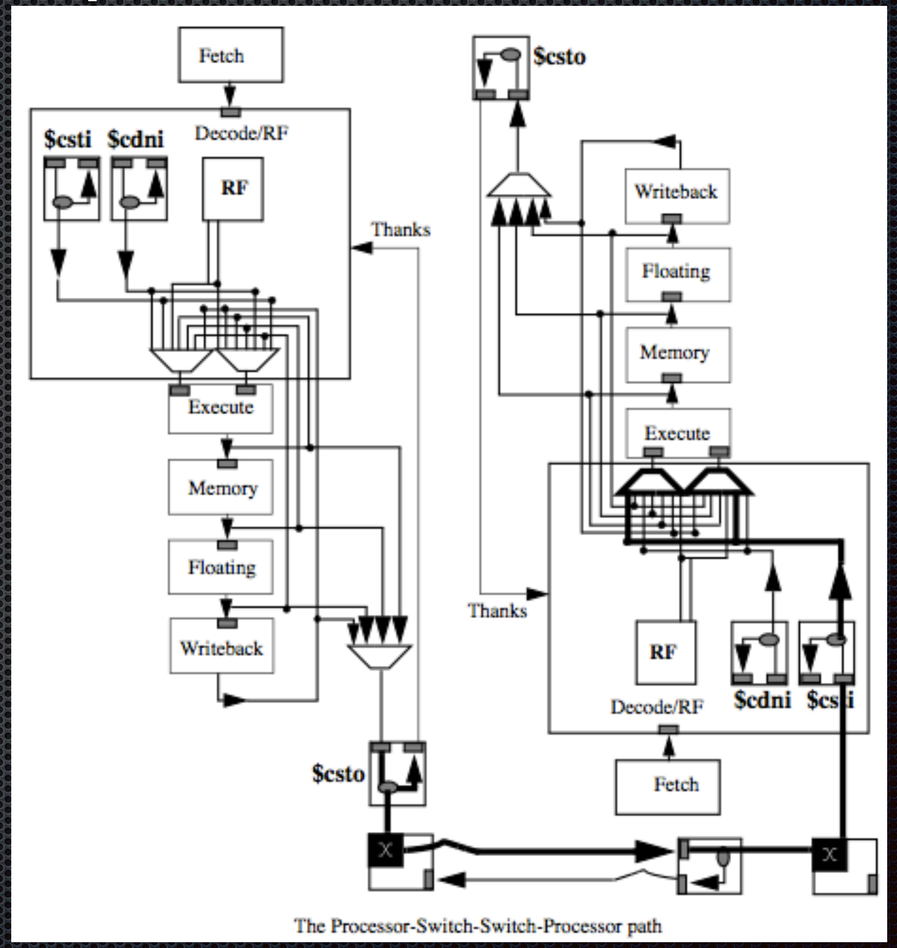
- MIPS R2000, expanded to 6 stages to accommodate floating point
- Static network ports mapped to registers 24 & 25



$csto, Bypass, and Writeback Networks

Switch Interlock Complication

- Because switch interface is in registers, data goes to switch early in pipe
- But this may be wrong for an exception
- Can deadlock switch, so need to ensure data has enough buffer space to allow a pipe restart



The Processor-Switch-Switch-Processor path

Floating Point

- Subset of IEEE 754, only in 32-bit
- Needs to interface to static network with same timing
- Adds a stage to the main pipe

Configurable Logic?

- Few applications beyond basic parallelism
- Hard to implement in ASIC
- Messes up pipeline and network timing
- Need a lot of it to do anything interesting
- Greatly complicates software support

No!

I/O Via Networks

- Simple interface to each port to allow chips to talk
- At edges, connect to FPGA translators, to enable communication with other devices
- Scales with chip pin count
- Doesn’t scale linearly with processor count, but OK for modest numbers of processor chips
- Unfortunately, not enough pins even for baseline chip

Predicted Performance

- Assuming pinout problem can be solved
- Assuming memory interface can be worked out
- Assuming unlimited SRAM available
- Assuming simulator within 10% of hardware

TABLE 7. Preliminary Results - 16 tiles

Benchmark	Speedup versus one tile
Cholesky	10.30
Matrix Mul	12.20
Tomcatv	9.91
Vpenta	10.59
Adpcm-encode	1.26
SHA	1.44
MPEG-kernel	4.48
Moldyn	4.48
Unstructured	5.34

Five Years Later...

- Double the number of networks
  - 2 static nets, each with 32k program memory
  - 2 dynamic nets, one for memory, the other for user messaging
  - Pinout growth hasn’t dodged the prior I/O problem
    - Expected 1124/1657 signal pins, but got 1080
- Expected 290 MHz clock, got 425 MHz (simulated)

Prototype Design

- Note changes in pipelines, register assignments
- Off chip ports

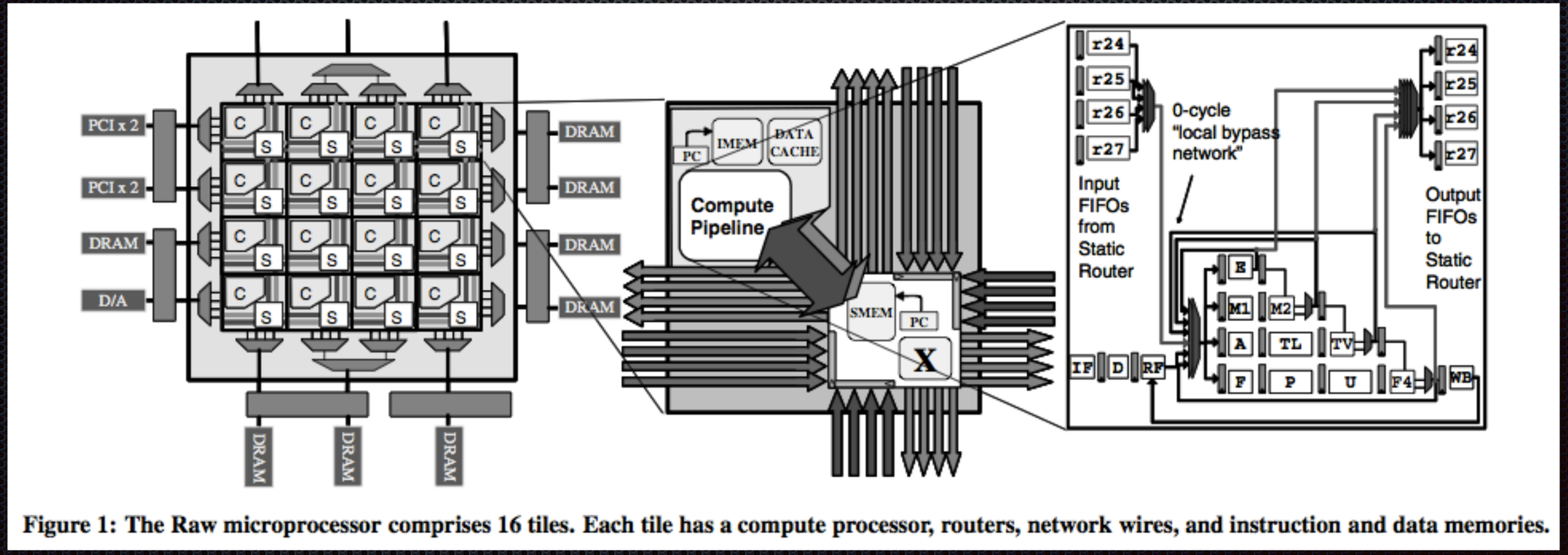


Figure 1: The Raw microprocessor comprises 16 tiles. Each tile has a compute processor, routers, network wires, and instruction and data memories.

Compared to a Carefully Chosen Equivalent P-III

Benchmark	Source	# Raw Tiles	Cycles on Raw	Speedup vs P3	
				Cycles	Time
Dense-Matrix Scientific Applications					
Swim	Spec95	16	14.5M	4.0	2.9
Tomcatv	Nasa7:Spec92	16	2.05M	1.9	1.3
Btrix	Nasa7:Spec92	16	516K	6.1	4.3
Cholesky	Nasa7:Spec92	16	3.09M	2.4	1.7
Mxm	Nasa7:Spec92	16	247K	2.0	1.4
Vpenta	Nasa7:Spec92	16	272K	9.1	6.4
Jacobi	Raw bench. suite	16	40.6K	6.9	4.9
Life	Raw bench. suite	16	332K	4.1	2.9
Sparse-Matrix/Integer/Irregular Applications					
SHA	Perl Oasis	16	768K	1.8	1.3
AES Decode	FIPS-197	16	292K	1.3	0.96
Fpppp-kernel	Nasa7:Spec92	16	169K	4.8	3.4
Unstructured	CHAOS	16	5.81M	1.4	1.0

Note that results are still from simulation

Predictions vs. Estimates

- Tomcatv: 9.91
- Cholesky: 10.30
- Mxm: 12.20
- Vpenta: 10.59
- SHA: 1.44
- Unstructured: 5.34

Benchmark	Number of tiles				
	1	2	4	8	16
Dense-Matrix Scientific Applications					
Swim	1.0	1.1	2.4	4.7	9.0
Tomcatv	1.0	1.3	3.0	5.3	8.2
Btrix	1.0	1.7	5.5	15.1	33.4
Cholesky	1.0	1.8	4.8	9.0	10.3
Mxm	1.0	1.4	4.6	6.6	8.3
Vpenta	1.0	2.1	7.6	20.8	41.8
Jacobi	1.0	2.6	6.1	13.2	22.6
Life	1.0	1.0	2.4	5.9	12.6
Sparse-Matrix/Integer/Irregular Applications					
SHA	1.0	1.5	1.2	1.6	2.1
AES Decode	1.0	1.5	2.5	3.2	3.4
Fpppp-kernel	1.0	0.9	1.8	3.7	6.9
Unstructured	1.0	1.8	3.2	3.5	3.1

Single Core

- Parallel nodes usually suffer a slowdown
- Fewer resources per core, but more cores

Benchmark	Source	# Raw Tiles	Cycles on Raw	Speedup vs P3	
				Cycles	Time
172.mgrid	SPECfp	1	.240B	0.97	0.69
173.applu	SPECfp	1	.324B	0.92	0.65
177.mesa	SPECfp	1	2.40B	0.74	0.53
183.equake	SPECfp	1	.866B	0.97	0.69
188.ammp	SPECfp	1	7.16B	0.65	0.46
301.apsi	SPECfp	1	1.05B	0.55	0.39
175.vpr	SPECint	1	2.52B	0.69	0.49
181.mcf	SPECint	1	4.31B	0.46	0.33
197.parser	SPECint	1	6.23B	0.68	0.48
256.bzip2	SPECint	1	3.10B	0.66	0.47
300.twolf	SPECint	1	1.96B	0.57	0.41

Stream and Dense Matrix Performance vs. P-III

Benchmark	Cycles Per Output on Raw	Speedup vs P3	
		Cycles	Time
Beamformer	2074.5	7.3	5.2
Bitonic Sort	11.6	4.9	3.5
FFT	16.4	6.7	4.8
Filterbank	305.6	15.4	10.9
FIR	51.0	11.6	8.2
FMRadio	2614.0	9.0	6.4

Table 11: StreamIt performance results.

Benchmark	Problem Size	MFlops on Raw	Speedup vs P3	
			Cycles	Time
Matrix Multiplication	256 x 256	6310	8.6	6.3
LU factorization	256 x 256	4300	12.9	9.2
Triangular solver	256 x 256	4910	12.2	8.6
QR factorization	256 x 256	5170	18.0	12.8
Convolution	256 x 16	4610	9.1	6.5

Table 13: Performance of linear algebra routines.

Discussion

Mark Gebhart ASPLOS 09

An Evaluation of the TRIPS Computer System

TRIPS Concept

- Dataflow meets control flow

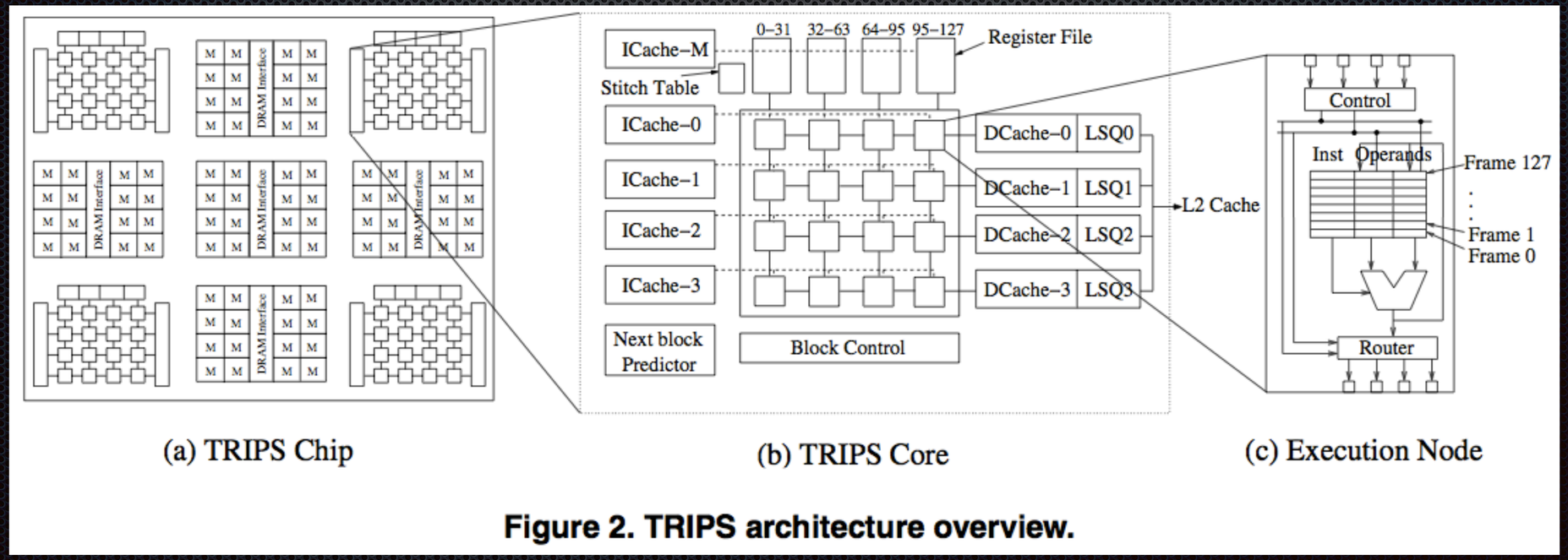


Figure 2. TRIPS architecture overview.

Block Oriented

- Programs partitioned into blocks
  - Single entry point
  - No internal loops
  - Potentially multiple exits
- Interrupts are block-precise

2003 Projections

- 4 16-wide cores
- Array of 32KB memory tiles on a routed network
- Distributed memory controllers
- 100 nm process, with 2005 target date
- Polymorphous -- supporting multiple operation modes
  - Some elements fixed, some variable or can be disabled

Polymorphous Resources

- Frames: Reservation stations with same index
- Register banks: More than in the ISA spec
- Block sequencing: Can be chained, etc., for modes
- Memory tiles: Can be L2 NUCA cache or specialized

Morphs: Desktop

- High Instruction Level Parallelism
  - Large distributed issue window
- Hyperblocks encoded VLIW style
- Memory is NUCA cache

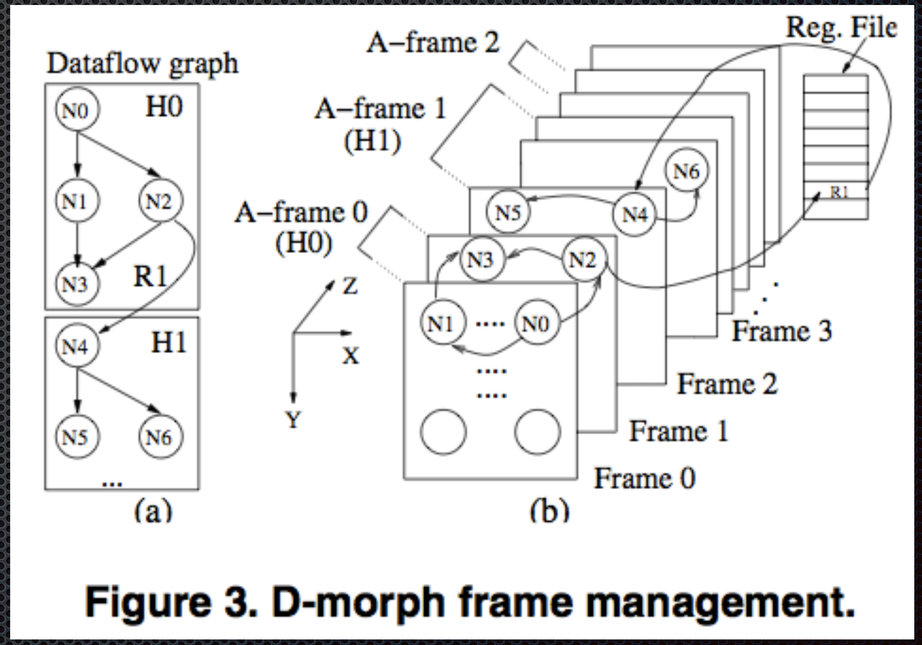


Figure 3. D-morph frame management.

Morphs: Thread Parallel

- Similar to SMT
- Statically partitions reservation stations
- Eliminates reorder buffer
- Frames are partitioned in advance; assigned to threads
- Multiple PCs are provided
- Cache must avoid cross-thread accesses

Morphs: Streaming

- Fuses frames into a super frame
- Blocks are recycled in reservation stations
- Imagine-like stream register file

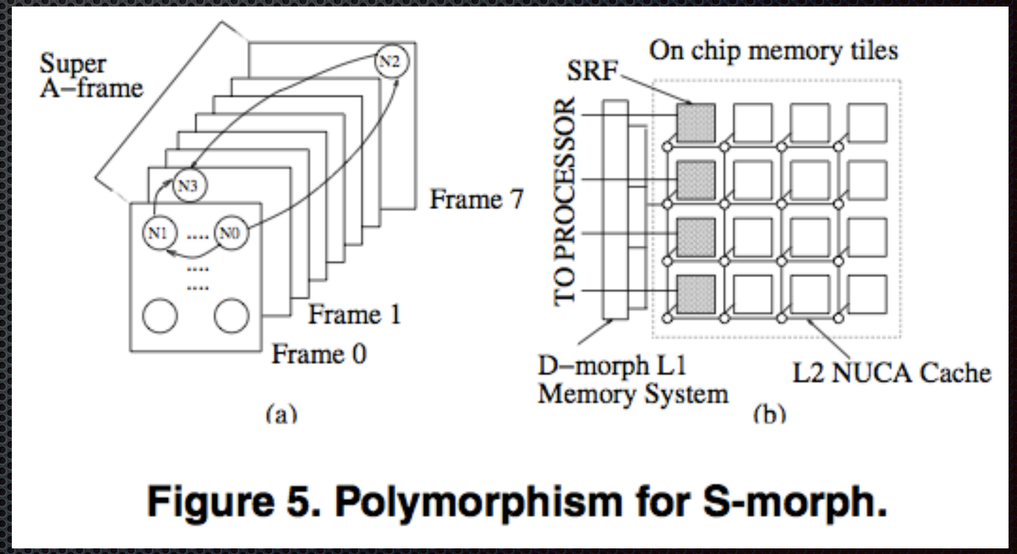


Figure 5. Polymorphism for S-morph.

Six Years Later...

- TRIPS is an EDGE (Explicit Data Graph Execution) ISA
- Block atomic dataflow
- 170M transistor ASIC at 366 MHz, 30 W

Die Photo

- Notice how instruction and data caches are on the same side now
- Two processors rather than four
- All memory in one area

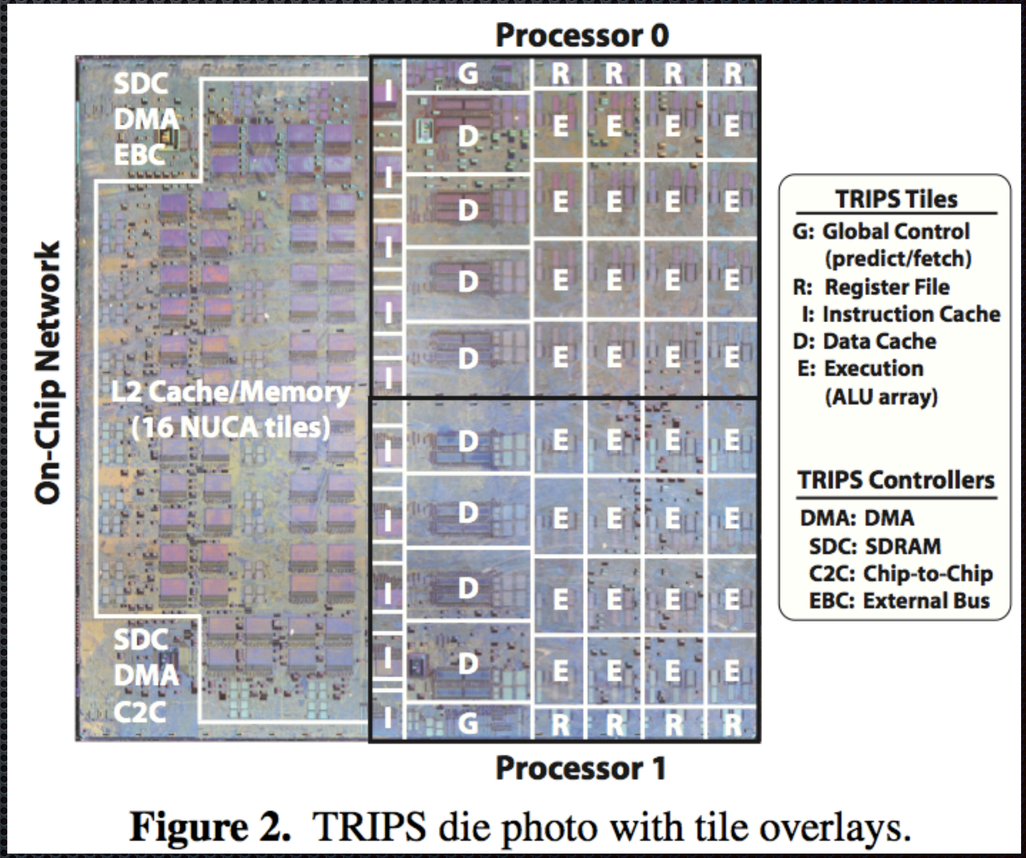


Figure 2. TRIPS die photo with tile overlays.

Comparison

- Proc/Mem Ratio is speed of processor vs. memory

System	Issue Width	Proc Speed (MHz)	Mem Speed (MHz)	Proc/Mem Ratio	L1 Cap. (D/I) (KB)	L2 Cap. (MB)	Mem Cap. (GB)
TRIPS	16	366	200	1.83	32 / 80	1	2
Core 2	4	1600	800	2.00	32 / 32	2	2
Pentium 4	4	3600	533	6.75	16 / 150	2	2
Pentium III	3	450	100	4.50	16 / 16	0.5	0.256

Table 1. Reference platforms.

Dynamic Instructions

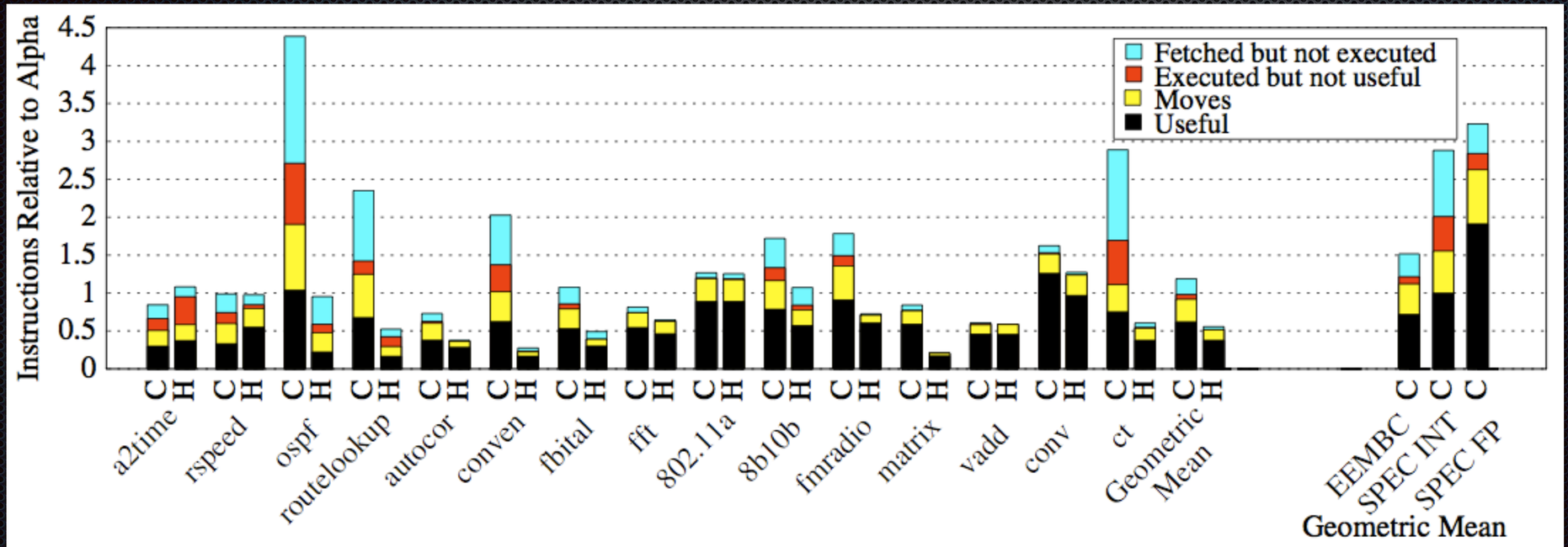


Figure 4. TRIPS instructions normalized to Alpha for compiled (C) and hand-optimized (H) benchmarks.

IPC

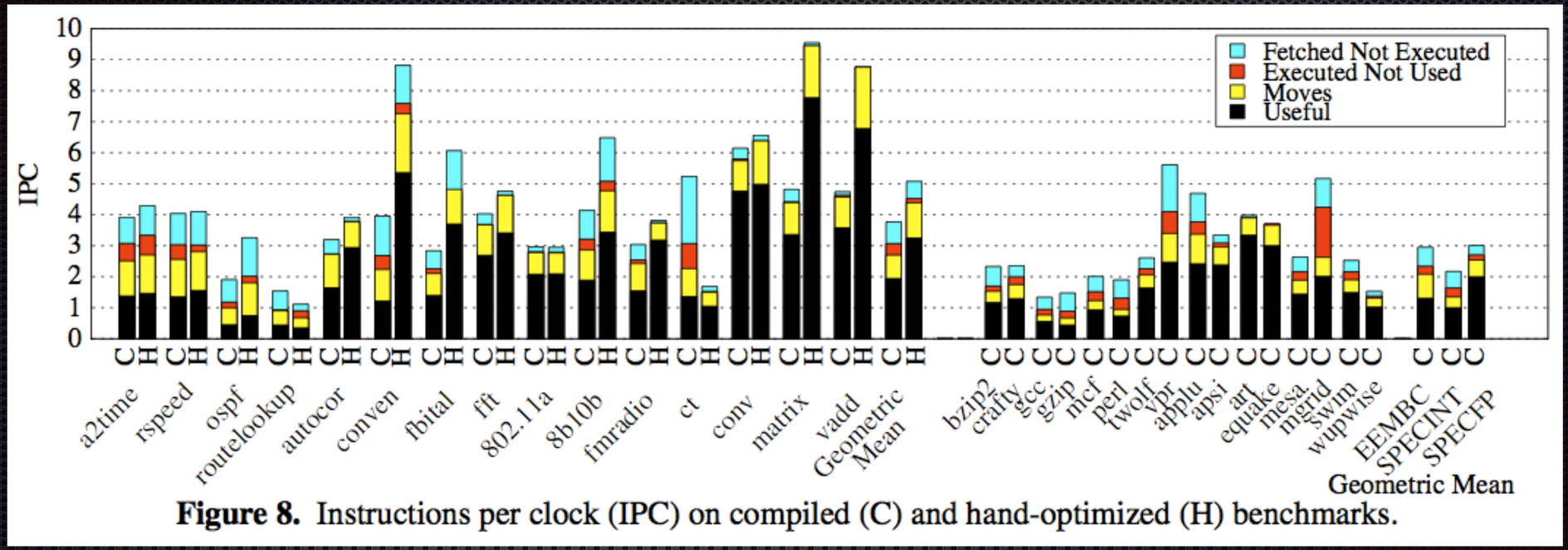


Figure 8. Instructions per clock (IPC) on compiled (C) and hand-optimized (H) benchmarks.

In a Perfect World...

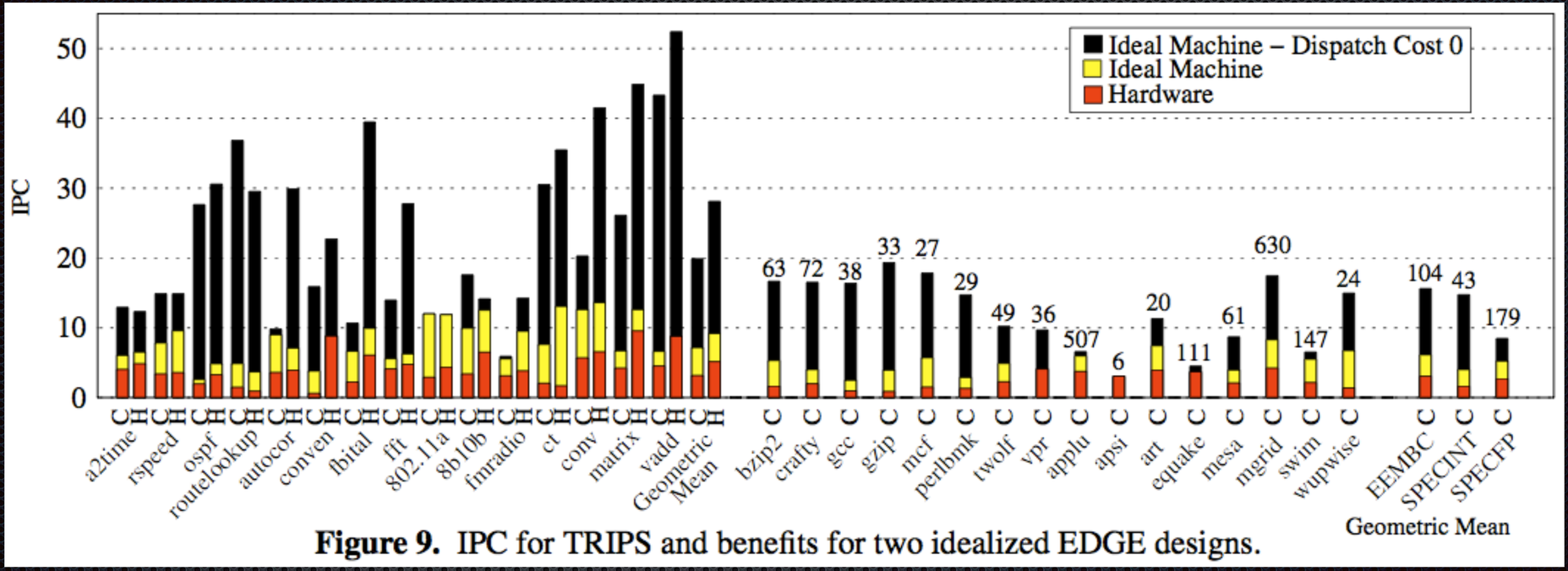


Figure 9. IPC for TRIPS and benefits for two idealized EDGE designs.

If Clocked at Same Rate as Core-2, Using gcc

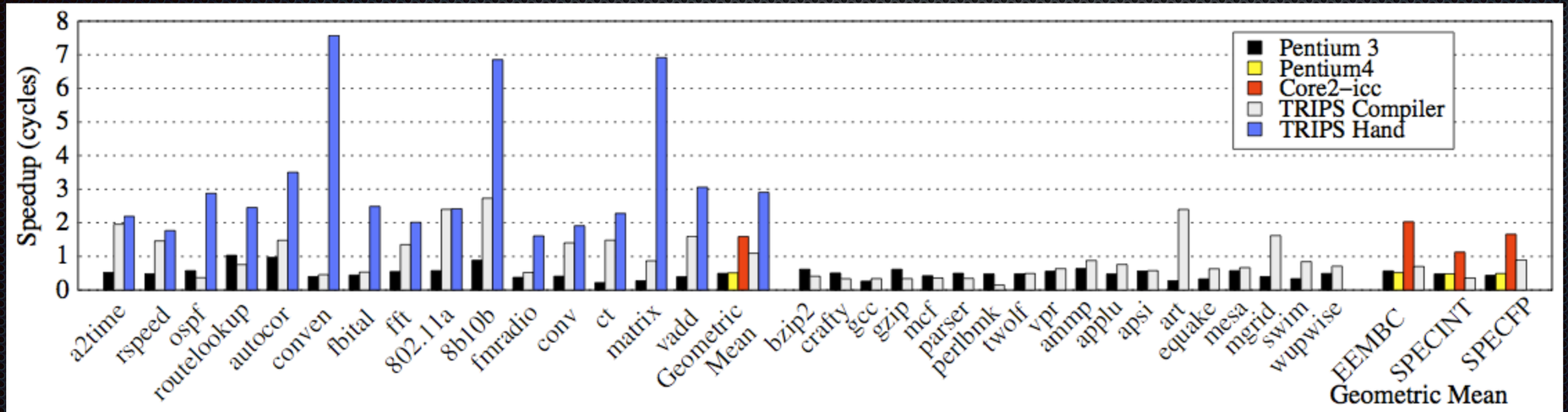


Figure 10. Speedup of TRIPS relative to the Core 2 on simple, EEMBC, and SPEC benchmarks.

Lessons

- Need operand broadcast
- Instruction block header overhead too high
- Need to map so instruction to instruction communication stays on the same tile
- Need predicate prediction
- Better memory distribution

Future?

- Even if clock matches modern processors, speedup will still be a small factor
- Benefits not enough to warrant a radical shift in ISA
- Limitations for supporting code not in C or Fortran (e.g., object-oriented, functional)

Discussion