Microarchitecture: Today and Tomorrow (2016)

* ***Today***

-- Pentium 4, M, Montecito
-- Power 5, Cell
-- Niagra

* ***The future***

-- what will technology provide
-- others' comments and my responses
-- the microprocessor of the year 2016

Pentium 4

∗ **Geneology:**

-- Pentium: Pentium, Pentium + MMX
-- Pentium Pro: Pentium-Pro, Pentium 2, 3
-- Pentium 4

∗ **Emphasis on Exploiting ALL the Latest Stuff**

∗ **Details**

-- Trace Cache

-Post decode
-12K µops, 512 entry BTB
-6 µops/trace_segment
-3 µops/cycle
-NO L1 Instruction cache

-- The Fireball

-pipelined adds, with back/back

-- Fetch/Decode

-8 bytes at a time from L2
-ONE x86 instruction
-up to 4 µops or entry into ROM

Pentium 4 Details (continued)

-- Rename/Allocate

-Data does not move
-Front end RAT, Retirement RAT

-- Reorder Buffer

-128 physical registers
-128 ROB status information
-maintain free list

-- Out-of-order execution
(dispatch 6 µops/cycle)

-- SMT

-- Store Buffer (24 entries)

-subsequent loads check store buffer

-- Branch Predictor

-4K entries
-16 entry RAS

-- Retirement (up to 3 uops/cycle)

Pentium 4 (continued)

* ***First chip***

- ***-- 1.5 GHz (4.1 GHz)***
- ***-- 55 watts***
- ***-- 42M transistors (65M)***
- ***-- 0.18μ CMOS (0.13μ)***
- ***-- 217 mm² die (131 mm²)***
- ***-- 144 new SSE2 instructions***

* ***Execution Units***

- ***-- 2 fast ALUs for simple stuff***
- ***-- 1 slow ALU for complex***
- ***-- 2 AGUs (one for LDs, one for STs)***
- ***-- 1 FPU for FP, MMX, SSE, SSE2***
- ***-- 1 FP Move***

* ***Memory Structure***

- ***-- Memory Order buffer (48 LDs, 24 STs)***
- ***-- Store buffer (24 entries)***
- ***-- L1D: 8KB, 4-way, 2 cycle***
- ***-- L2: 256KB, 8 way, 128 line***

Pentium M

- * ***NEW Design Point***

  -- Keep Performance Up
  -- BUT pay attention to energy consumption

- * ***Don't go NUTSO on saving energy: <10% from µproc***

- * ***Objectives***

  -- Execute fewer instructions/task

  - Better branch prediction
  - SSE (SIMD)

  -- Execute fewer µops/instruction

  - µop fusion
  - dedicated stack

  -- Fewer switches/µop

  - Power-managed L2 cache

  -- Less energy/transistor switch

  - speed-stepping

Pentium M (continued)

* ***Some numbers***

-- 77 million transistors
-- 0.13 μ CMOS
-- 84 mm^2
-- 1.6 GHz (24.5 watts, peak)
-- 32 GB/sec bus
-- X86 + SSE2

* ***Cache numbers***

-- L1 I,D: 32KB, 64B line, 8-way, 3 cycle
-- L2: 1MB (50 M of the 77M transistors)
8-way, 64B line, 5 cycle

Montecito

* **The numbers**

-- 1.72 billion transistors
-- 600 mm² die
-- 100 watts, 1.8 GHz, 0.09μ
-- three levels of cache: total 27MB

* **Interesting features**

-- 2 core CMP, 2 threads/core SMT
-- Thread control: switch on priority (0-7)
-- Foxtron technology
- adjusts freq, based on temperature, power
- requires synch between cores

* **Cache System**

-- L1
- separate I,D, 4-way, 16KB, in-order
- non-blocking, single cycle
- L1D: write through, dual ported

-- L2
- L2I: 1MB, 8-way, 128B line, 7 cycle out of order, pipelined
- L2D: 256KB, 5 cycle integer, out of order, pipelined, 16 misses handled, 32 requests outstanding

-- L3
- 12MB, 14 cycle, 128B line
- critical byte first
- priority to L2I
- no clock, r/w valid signal

Power 5

* Changes to Power 4

-- Adds SMT to CMP (2 threads/core, 2 cores/chip)
-- Both processors still share common L2
-- L3 removed from hierarchy, is now huge victim cache (L3 still off chip)
-- Memory controller now on chip

* SMT

-- one thread each cycle – alternates (up to 8 instructions/core each cycle)
-- Separate PCs, RAS
-- 120 shared physical registers

* Thread Control

-- Dynamic Resource Balancing
-- Software directed adjustable thread priority
-- If threads are both priority 0 or 1, enter power save mode

* Branch Predictor

-- Two BHT use bimodal and path-correlated hybrid
-- RAS for each thread

Power 5 (continued)

* **Group Commit**

-- up to 5 instructions/group
-- all instructions from same thread
-- commit group as a unit
-- group occupies one of 20 entry
Group Completion Table

* **Execution Core**

-- group dispatch to in-order issue queues
-- out-of-order from issue queue to execute
units
-- issue 8 each cycle
-- 8 execution units
(2 FPU, 2 FXU, 2 LSU, 1 LXU, 1 BRU)

* **Memory Structure**

-- L1: 64KB I, 32 KB D, 4-way
-- L2: 1.875MB in 3 parts, 10-way
(three simultaneous accesses)
-- L3: 36MB off chip (directory on-chip)
(huge victim cache)

* **Chip Details**

-- 0.13μ, 389 mm^2, 276M transistors

Niagra

** Soul of Niagra:*

-- Conserve Power
-- Leave hard stuff to the software
-- 8 processors: in-order, 4-way SMT
-- NO speculation
-- Single issue
-- Short (6 stage) pipeline

** Thread select: LRU Augmented*

** Chip structure*

-- 8 processors
-- non-blocking crossbar (oldest first)
-- 4 banks, L2 cache
-- memory controller

** Numbers*

-- 1.0 GHz, 0.09μ, 200 mm^2, <60 watts

** Memory system*

-- L1: 16KB I + 64 ITLB
8KB D + 64 DTLB (write through)
-- L2: 3MB + directory for L1 coherency

The Challenges:

* ***Ten billion transistors***

-- ***power requirements (leakage current)***

-- ***design complexity***

* ***Frequency in excess of 10GHz***

-- ***on-chip/off-chip disparity (the memory wall)***

-- ***on-chip propagation time***

Naysayers

- ***Go for the low-hanging fruit***

  --Easy to get great improvement
  (The hard problems still remain)

- ***CAD can't verify***

  --Make simpler machines
  (Make better CAD tools)

- ***Law of Diminishing Returns***

  Performance does not track resources applied
  (Let's look more closely)

- ***We don't need anything faster***

  1986: MIPS R2000, 1996: Pentium Pro, 2006: ??
  (Check out the marketplace)

- ***Processor can't keep up***

  --Memory Wall
  (Algorithm, compiler, microcode)

- ***Our experiment proves it can't be done***

  1.85 IPC, for example
  (Let's look more closely)

Moore's Law:

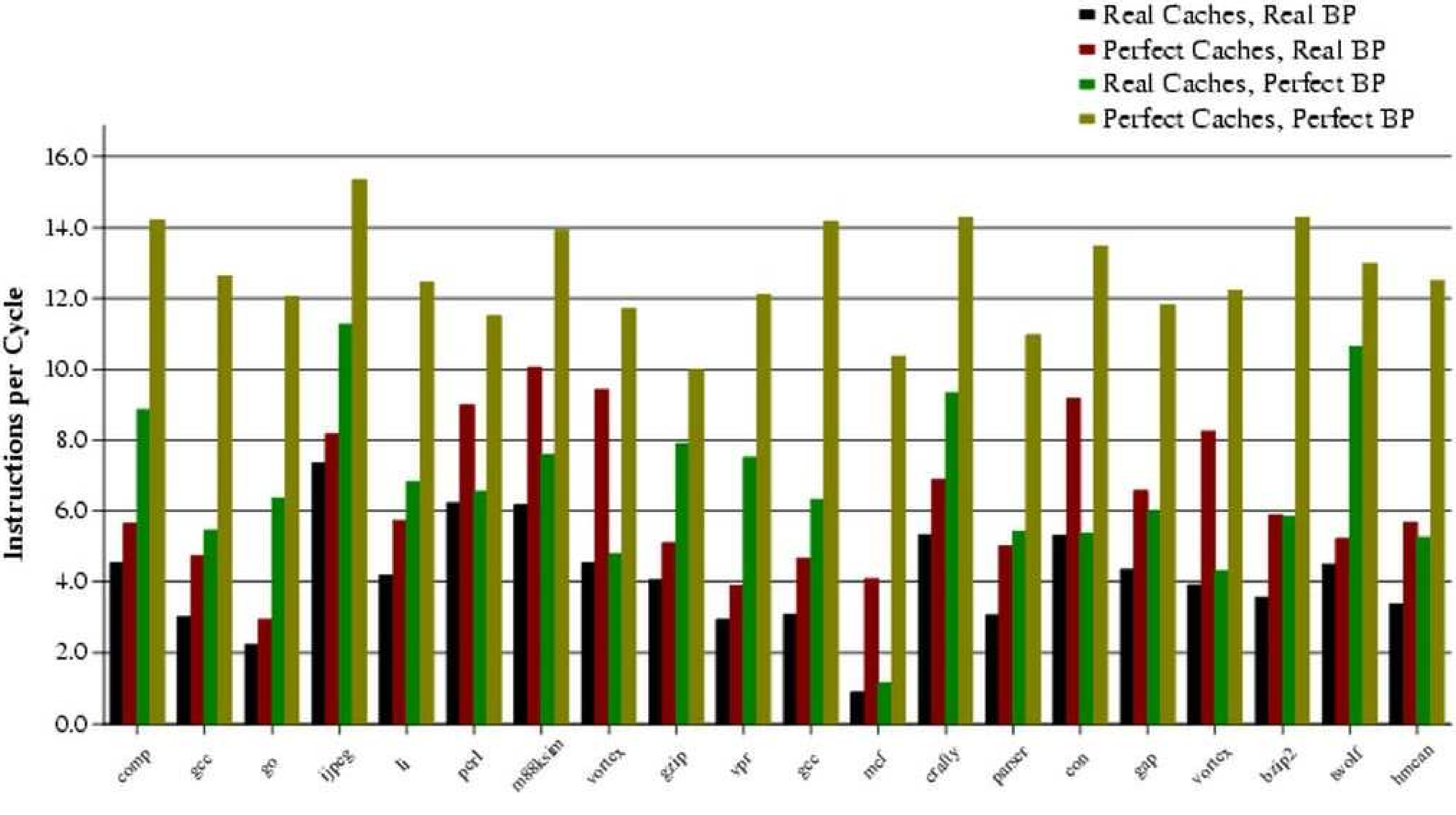
(a) Physics?

(b) Process Technology?

(c) Microarchitecture?

(d) Psychology?

Potential of Improving Caches and BP



"Tomorrow's" Microprocessor

* Wholistic approach
--The transformation hierarchy revisited

* X + Superscalar

* Compiler/μarch symbiosis
--multiple levels of cache
--Block-structured ISA
--part by compiler, part by μarch
--fast track/slow track

* Power Awareness

* Multiple cycle times
--asynch/synch together

* SSMT (aka helper threads)

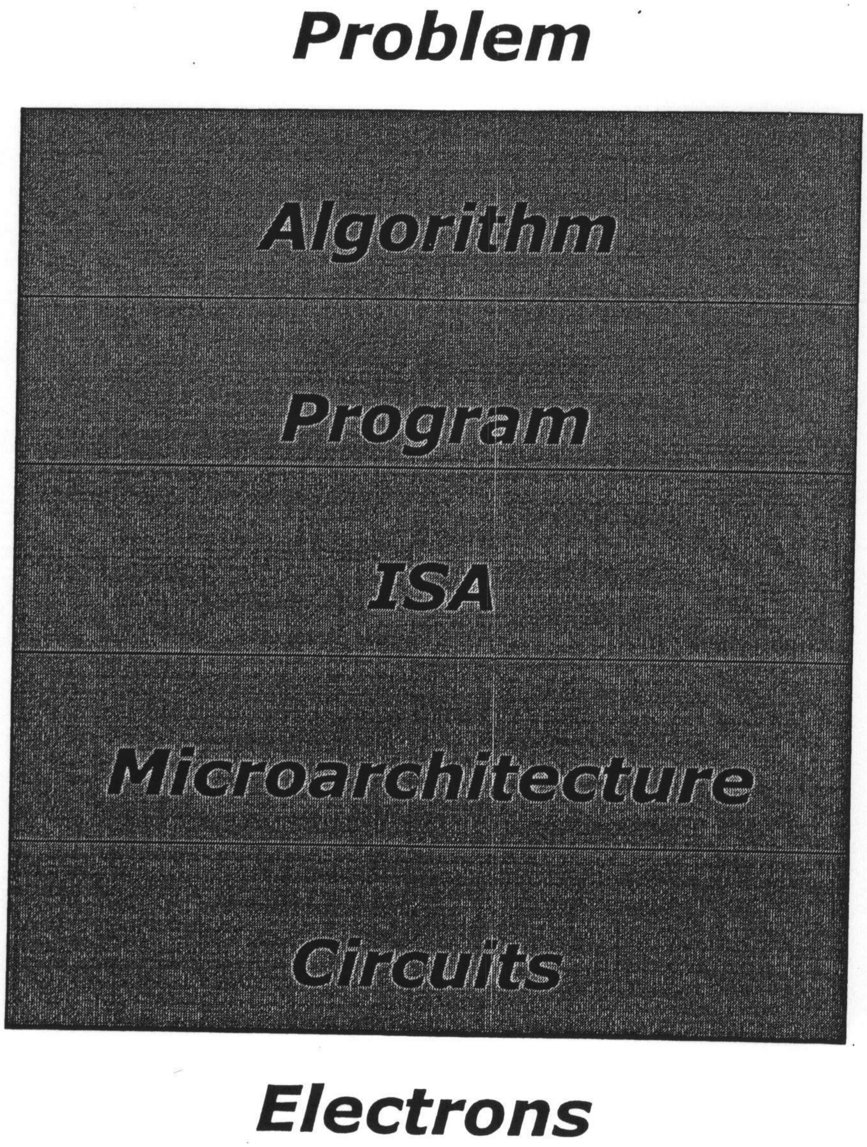
* More Microcode

* Verification Hooks

* Internal fault tolerance

* Niagra X / Pentium Y

* Security



Problem

Algorithm

Program

ISA

Microarchitecture

Circuits

Electrons

x + superscalar

- *DSP + superscalar*
- *graphics + superscalar*
- *data base + superscalar*
- *networks + superscalar*
- *vectors + superscalar*

Hardware vs. Microarchitecture/Compiler

Rather than ask the Hardware to do the whole thing,

1. ***Partition the problem into part compiler/ part μarch***

2. ***Augment ISA to deliver compiler-generated information to the μarch***

3. ***μarch manages the optimization, combining compile-time part and hardware part.***

Fast track / Slow track
(Another compiler/μarch symbiosis)

1. Implement the ISA in two parts:

--Fast track: those things that can go fast

--Slow track: those things that can't

2. Compiler knows what is done where, and compiles accordingly

<u>Power</u>

Virtual Allocate, Physical Store (UPC)

Retire Does Not Necessarily Mean Update

Partition The Cache, Don't Replicate

Demand Only Broadcast

The Refrigerator

No More Registers (Only Buffers)

Block-Structured ISA

An observation:

> ***Computer Architecture will always be alive and healthy as long as people can dream***

Reason:

> ***Computer architecture is about the interface between what technology can provide and what the market demands***

i.e.:

> ***As process technology continues to advance, and dreamers find new uses for computers, Computer Architecture's future should be very, very bright***