Final value of counter beast?

We added a counter to the multicore code. What will the

Review Problem 56
Intel Micro-Architecture (Ice Lake/Sunny Cove)

A bit of x86 instruction set
14-deep pipeline
Register renaming
Speculative execution
Superscalar

Highlights:

Goal: Fast overview of one of Intel's main processors
Backwards compatible...

2019: Ice Lake/Sunny Cove

2013: Haswell (7th), 1.6 GHz, 4 cores, on-die GPU, 2 branch units
2012: Ivy Bridge, 2.6 GHz, 4 cores, 2.5 GHz, 2M transistors, 14-stage pipeline, multi-core
2008: Nehalem / Intel Core 2 Duo, 64-bit, 1.0-2.3 GHz, 291 transistors, 14-stage pipeline, uncore

2006: Core 2 Duo, 64-bit, 1.0-3.0 GHz, 42 transistors, 20 deep pipeline, symmetric

2000: Pentium 4, 1.3-3.0 GHz, 42 transistors, 20 deep pipeline, symmetric

1996: Pentium M, 150-2.33 GHz, 4.5M transistors, SIMD (single instruction multiple data) instructions

1995: Pentium Pro, 32-bit, 200 MHz, 55M transistors, CP1=1/3, 24-staging pipeline, out-of-order execution, predicted instructions, 4-bit branch history

1992: Pentium M, 32-bit, 60-66 MHz, 3.3M transistors, 1.6KB L1, 1.2KB L2 branch predict, 1.2KB L1 data cache, 8KB L2 cache, 3-staging pipe

1989: 80486, 32-bit, 25 MHz, 1.2M transistors, 8KB L1, 6-staging pipe

1985: 80386, 32-bit, 16-33 MHz, 256K transistors, 256KB code cache

1982: 80286, 16-bit, 8-12 MHz, 134K transistors, 4-bit address space

1980: 8087 floating point coprocessor

1978: 8086, 8-bit, 2.0 MHz, 6K transistors

1974: 8080, 8-bit, 2.0 MHz, 6K transistors

Some relevant steps (not all):

X86 Milestones — Evolution of the Instruction Set
<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mem[Reg(id)+Reg(id2)*2scale+displace]</td>
<td>Base + scaled + 8/32/64 displacen</td>
<td>Multiple data memory addressing modes:</td>
</tr>
<tr>
<td>Mem[Reg(id)+Reg(id2)*2scale]</td>
<td>Base + scaled index</td>
<td>Immediate</td>
</tr>
<tr>
<td>Mem[Reg(id)]</td>
<td>Base + 8/32/64-bit displacen</td>
<td>Memory</td>
</tr>
<tr>
<td>Mem[Reg(id)]]</td>
<td>Register</td>
<td>Memory</td>
</tr>
<tr>
<td></td>
<td>Immediate</td>
<td>Memory</td>
</tr>
<tr>
<td></td>
<td>Register</td>
<td>Memory</td>
</tr>
<tr>
<td></td>
<td>Immediate</td>
<td>Memory</td>
</tr>
<tr>
<td></td>
<td>Source/destination operand</td>
<td>Memory</td>
</tr>
<tr>
<td></td>
<td>Second source operand</td>
<td>Memory</td>
</tr>
</tbody>
</table>

**X86 Operands**

A = A + B

**2-operand instructions:**

16x64-bit registers plus special-purpose registers (Flag, segments, etc.)
Single instruction, multiple data (i.e., 4x8-bit adds simultaneously, SIMD (MMX, SSE))

Control flow: conditional & unconditional jumps, calls, returns

Arithmetic & logic: test, integer, decimal math, etc.

Data movement: Move, push, pop

Binary code details

Instructions
Extra byte for scaled index mode.
May have extra byte to indicate addressing mode
Opcode says bitwidth of 8-bit/32-bit.
Range from 1-byte to 17-bytes!

Instruction Encoding
Picture from David Schor, WickChipPort.

To multiplex RISC-like micro-ops, Pseudo-instructions converted to X86 Instructions are essentially.

CPU decodes X86 into micro-ops at runtime.
On-chip GPU

Symmetric Multithreading (2-way per core)

Each chip has 2-4 cores

Register Renaming to 180 registers.

Superscalar picks from 352-instruction window.

CPUs is ~10-way superscalar, ~14 pipeline stages (P4 had 20!)
~36-cycle latency

13-cycle latency

5-byte loads / stores. 128B/cycle latency.

4-byte loads / stores. 16B/cycle latency.

Can fetch 16B/cycle.

All 64 byte blocks, write-back.

Cache Organization