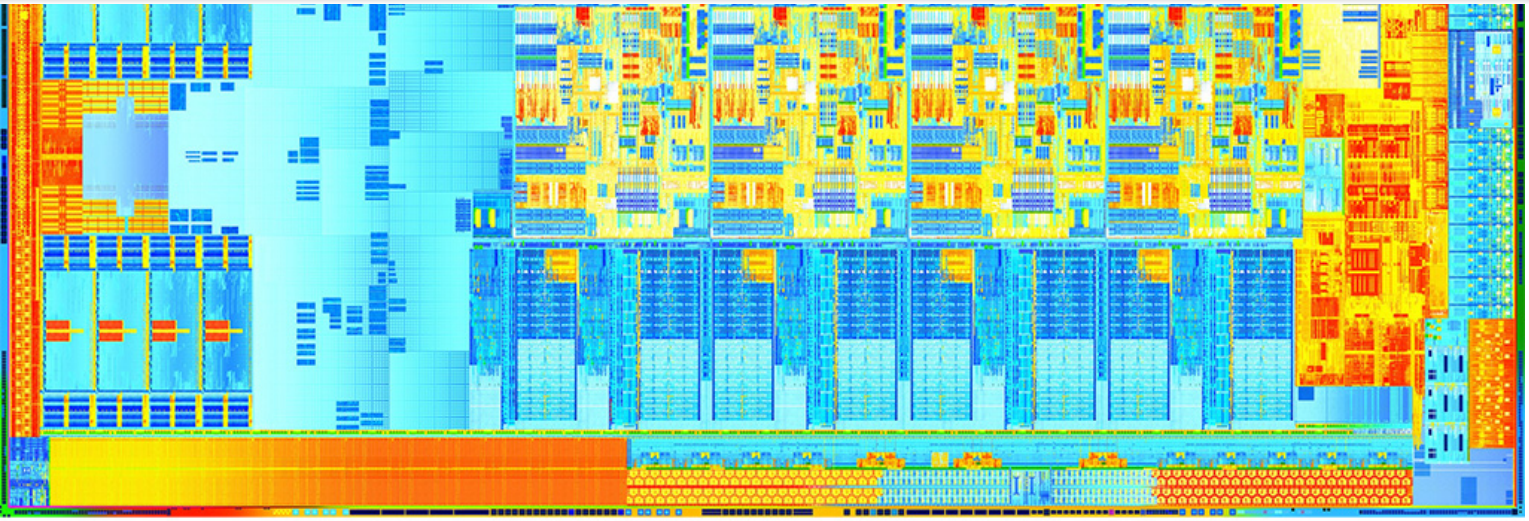
the x86 is dead. long live the x86!

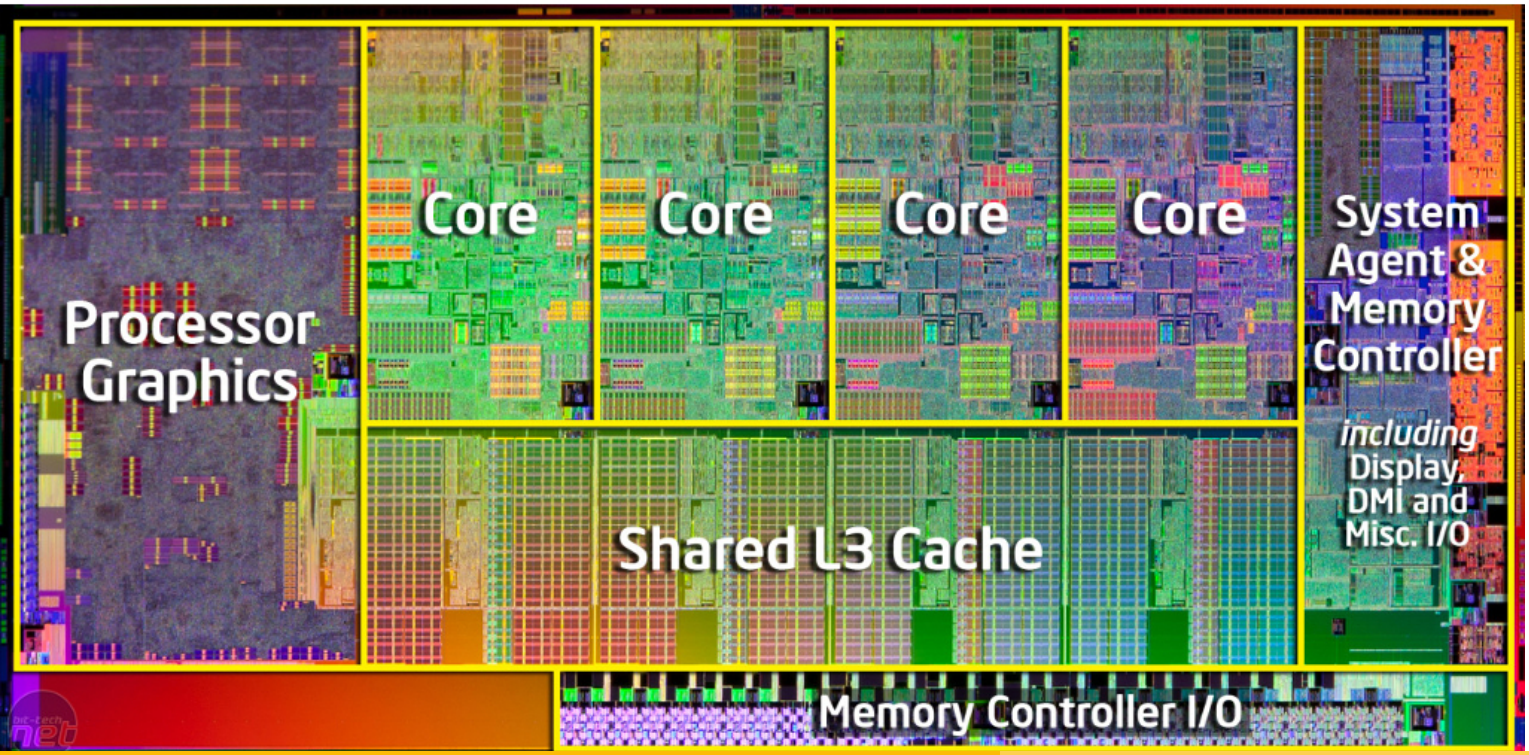
unix weapons school

with diagrams by david kanter of http://realworldtech.com

“Upon first looking into Intel’s x86”



Third generation ("Ivy Bridge") Intel Core processors pack 1.4B transistors into 160mm², with a MSRP of about $300.



that upon which we gaze is mankind’s triumph,

and we are its stewards.

use it well.

Why study the x86?

- Used in a majority of servers, workstations, and laptops
- Receives the most focus in the kernel/toolchain
- Very complex processor, thus large optimization space
- Excellent documentation and literature
- Fascinating, revealing, lengthy history

Do not think that x86 is all that's gone on over the past 30 years[1].

That said, those who've chased peak on x86 can chase it anywhere.

[1] Commonly expressed as "All the world's an x86."

In the grim future of computing there are 10,000 ISAs

- Alpha + BWX/FIX/CIX/MVI
- AVR32 + JVM
- CMS
- PA-RISC + MAX-2
- SuperH
- i960
- IA64 (Itanium)
- MIPS + MDMX/MIPS-3D
- IBMHLA (s390 + z)
- m68k
- VAX + VAXVA
- z80 / MOS6502
- MIX
- SPARC V9 + VIS3[a]
- JVM[b]
- PTX/SASS[c]
- TILE-Gx[d]
- ARM + NEON[e]
- Blackfin
- PowerISA + AltiVec/VSX[f]
- MMIX

---

[a] Most recently the "Oracle SPARC Architecture 2011".
[b] Most recently the Java SE 7 spec, 2013-02-28.
[c] Most recently the PTX ISA 3.1 spec, 2012-09-13.
[d] TILE-Gx ISA 1.2, 2013-02-26.
[e] ARMv8: A64, A32, and T32, 2011-10-27.
[f] PowerISA v.2.06B, 2010-11-03.

IA-16 CISC (MS-DOS, 640K–1M RAM)

8086/80186 (NEC, AMD, Fujitsu, OKI, Kvazar-Mikro, . . .)

- IBM PC/XT. Real mode. 16-bit regs and buses
- 1MB RAM in 16 64K segments
- 8086-2 ISA on 80186 (`ENTER/LEAVE, PUSHA/POPA, INS/OUTS`)

80286 (IBM, AMD, Fujitsu, . . .)

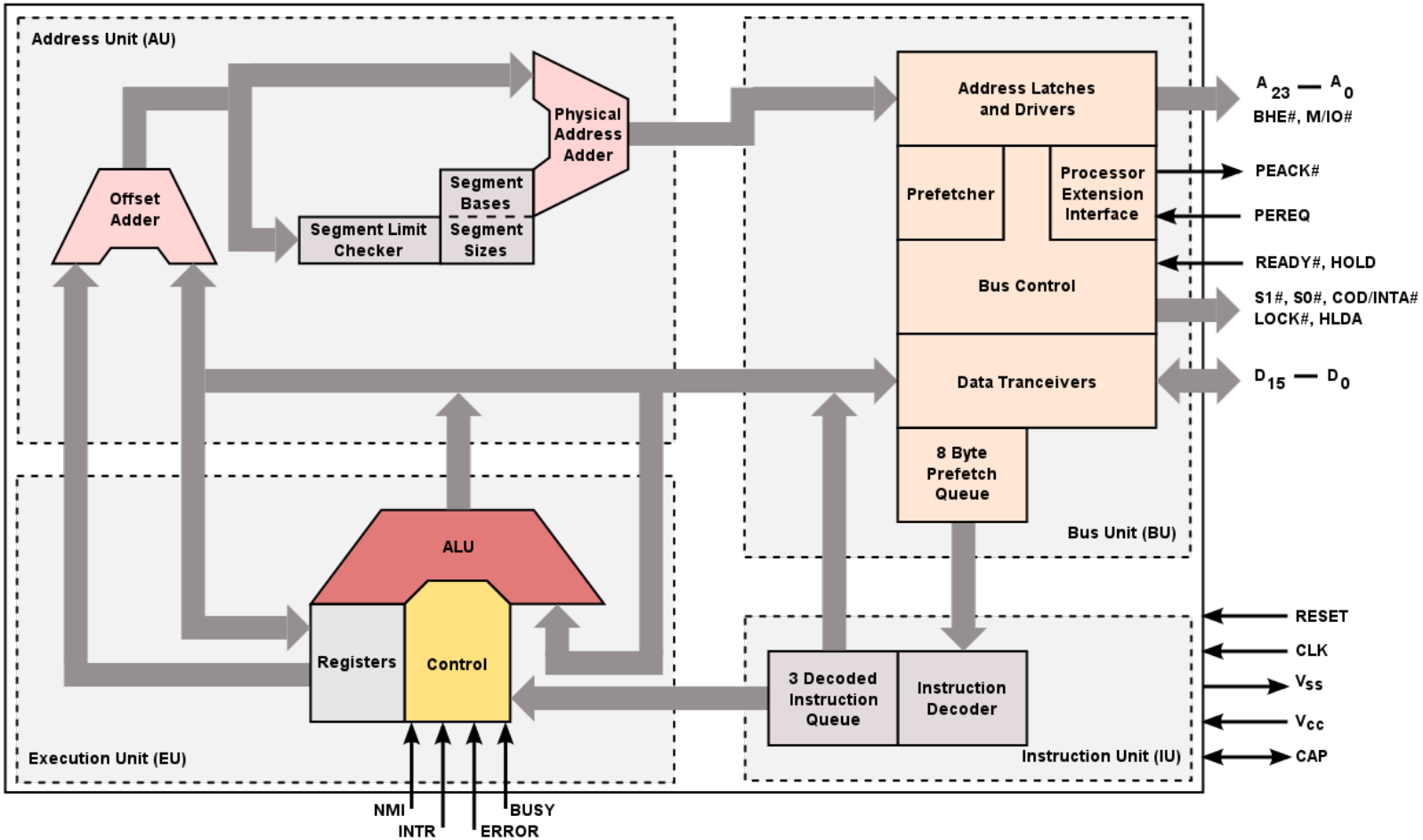
- IBM PC/AT (ISA). Protected mode w/ MMU
- 16MB RAM with privilege levels on segments

8086	1978	29K@3.2µm	16 / 16 / 20	4–10MHz
8088	1979	29K@3.2µm	16 / 8 / 20	5–8MHz
80186	1980	55K@2.6µm	16 / 16 / 20	6–25MHz
80188	1982	55K@2.6µm	16 / 8 / 20	6–25MHz
80286	1982	134K@1.5µm	16 / 16 / 24	4–12.5MHz

Intel 80286

image source: Wikimedia Commons

Intel 80286 architecture



IA-32 CISC (Win 3.11, OS/2, Linux, 386BSD)

80386 (Am386, TI, Chips, IBM, . . .)

- IBM PS/2 (MCA). 32-bit segments, regs, buses
- NatSemi 16550A UART. 80386SL: first (external) cache
- 2**c** per simple instruction

80486 (Am486, Cx486DLC, TI486SXL, IBM486DLC,. . .)

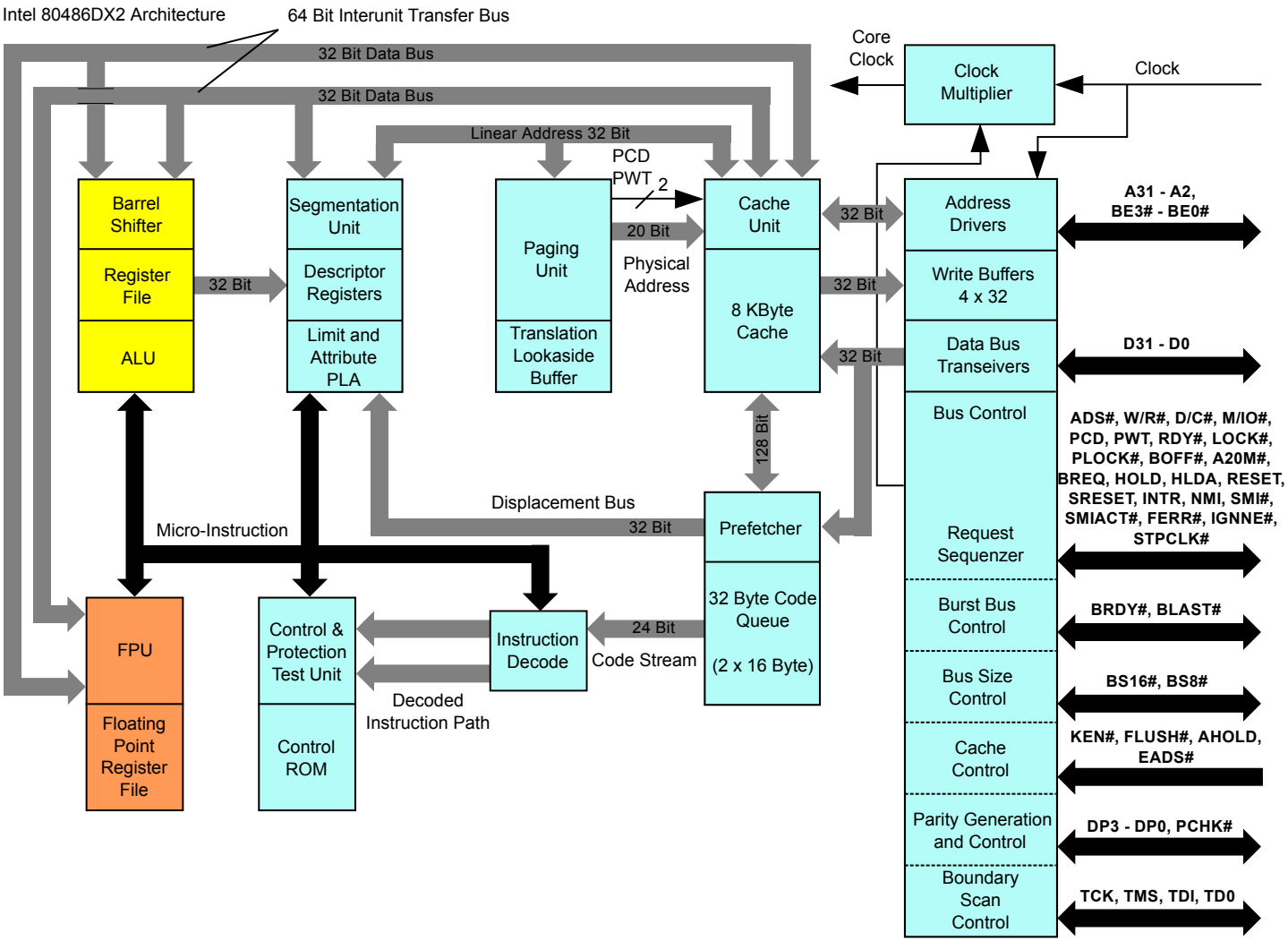
- EISA/VLB. On-die 8KB U$, `XADD`, `CMPXCHG`
- 1**c** per simple instruction, integrated FPU

80386-DX	1985	275K@1.5–1µm	32 / 32 / 32	12–33MHz
80386-SX	1988	275K@1.5–1µm	32 / 16 / 24	16–33MHz
80486-DX	1989	1.2M@1–.8µm	32 / 32 / 32	20–50MHz
80486-SX	1991	1.2M@1–.8µm	32 / 32 / 32	16–33MHz
80486-DX2	1993	1.2M@.8µm	32 / 32 / 32	40–66MHz
Am486-DX4	1995	1.25M@.44µm	32 / 32 / 32	75–120MHz

Intel 80486DX2

image source: Wikimedia Commons

http://upload.wikimedia.org/wikipedia/commons/4/42/80486DX2_arch.svg



Pipelined superscalar (Win 95, BeOS, FreeBSD)

Intel P5 (Pentium, Pentium MMX)

- U/V pipes, 8KB each I$/D$, better FPU/MUL, debug regs
- PMMX adds MMX + perf counters (`RDPMC`), 2x16KB L1

AMD K5 (SSA/5, 5k86)

- RISC Am29000 with x86 frontend emitting µcode
- Variable register window sizes

Cyrix 6x86

- Scratchpad ("L0") cache, great ZOPS, crap FPU

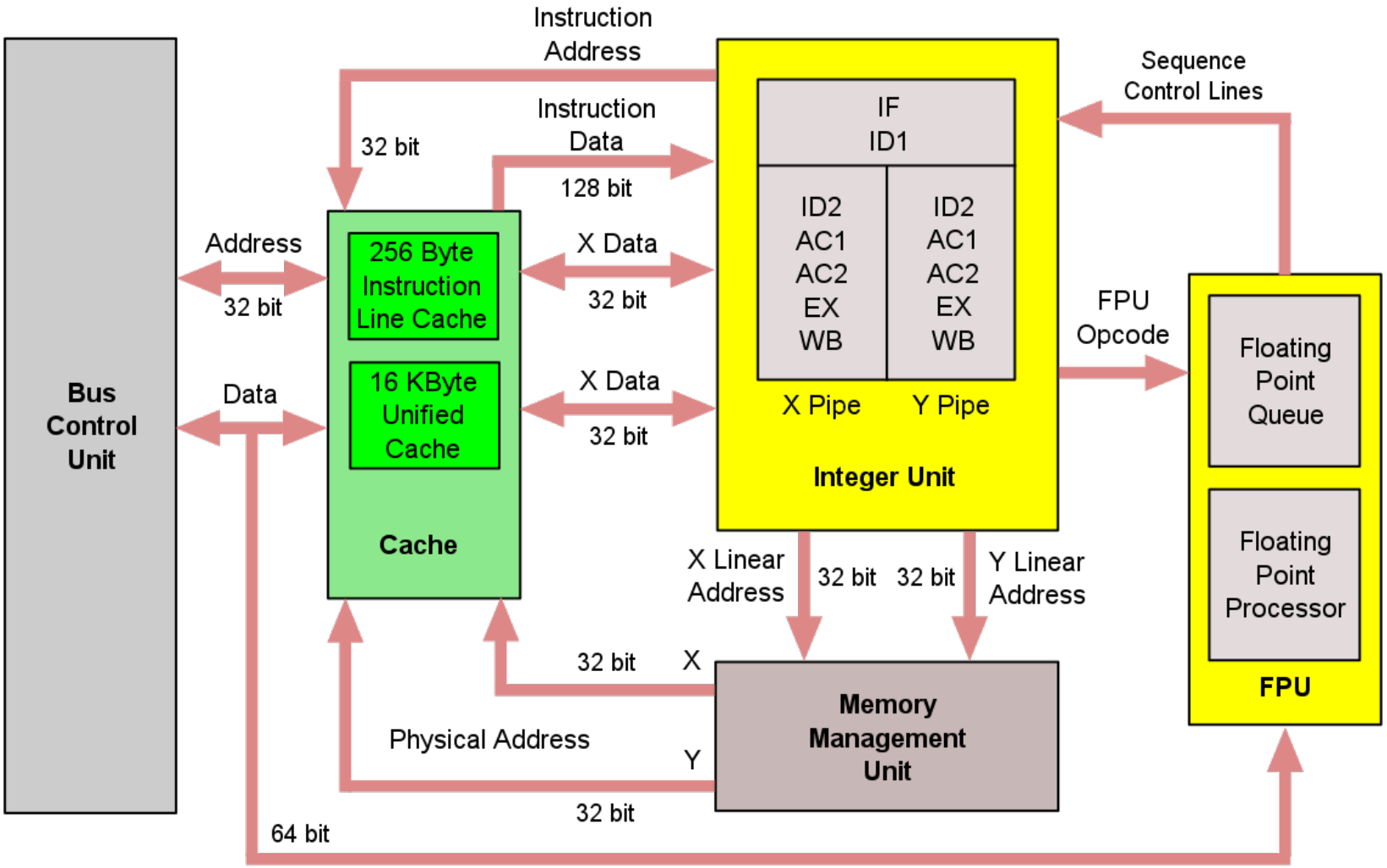
Pentium[2]	1993	3.1M@.8–.35µm	5	60–300MHz
AMD SSA/5	1996	4.3M@.5–.35µm	5	75–100MHz
AMD 5k86	1996	4.3M@.5–.35µm	6	90–133MHz
Cyrix 6x86	1996	3.3M@.65–.35µm	7	80–150MHz

[2] The faster .35µm Pentiums (P54CS, P55C, Tillamook) didn't emerge until 1995 (P54CS) or 1997.

Cyrix 6x86

image source: Wikimedia Commons

Cyrix 6x86 architecture



IF: Prefetch Stage
AC: Address Calculation Stages
EX: Execution Stages
ID: Decode Stages
WB: Write Back Stages

SSE + OOO CRISC (Win 98, FreeBSD 3, Linux 2)

Intel P6 (PPro / PII / PIII)

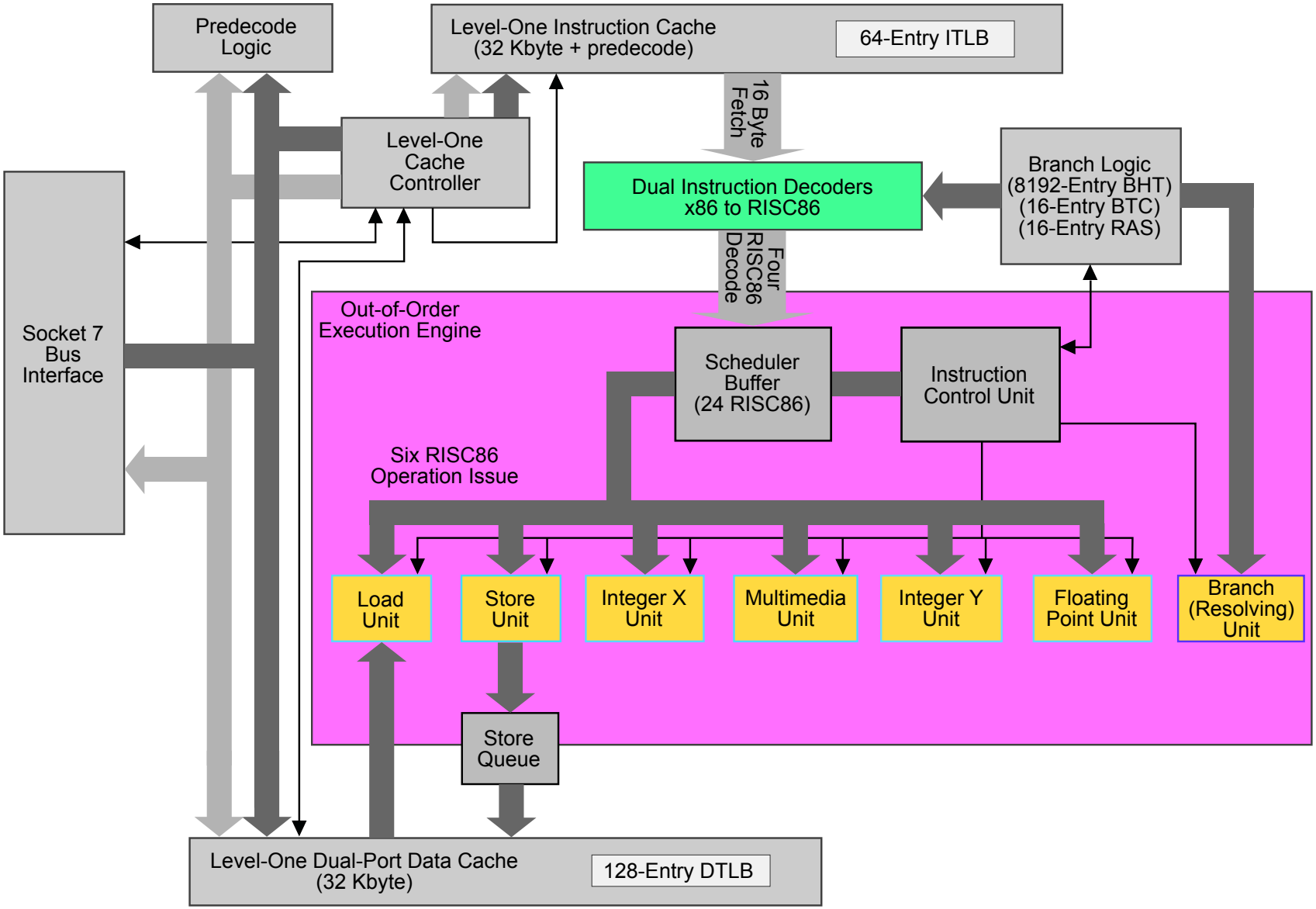
- PPro adds `SYSENTER`, `SYSEXIT`, big "on-package" L2
- PIII adds 8 128-bit regs, SSE1 and `CMOVcc`
- Low-latency 8W L2 on Coppermine, HW prefetch on Tualatin

AMD (NexGen) K6 (K6 / K6-2 / K6-III)

- Adds `SYSCALL`, `SYSRET`
- K6-2 adds 3DNow! atop MMX regs
- K6-III just sticks an L2 on K6-2

Pentium Pro	1995	5.5M@.6–.35µm	14	150–333MHz / 60–66MHz
K6	1997	8.8M@.35–.25µm	6	166–300MHz / 66MHz
K6-2	1998	9.3M@.25µm	6	300–550MHz / 66–100MHz
Pentium III	1999	9.5M@.25–.13µm	10	400–1400MHz / 100–130MHz
K6-III	1999	21.4M@.25–.18µm	10	350–500MHz / 66–100MHz

image source: Wikimedia Commons

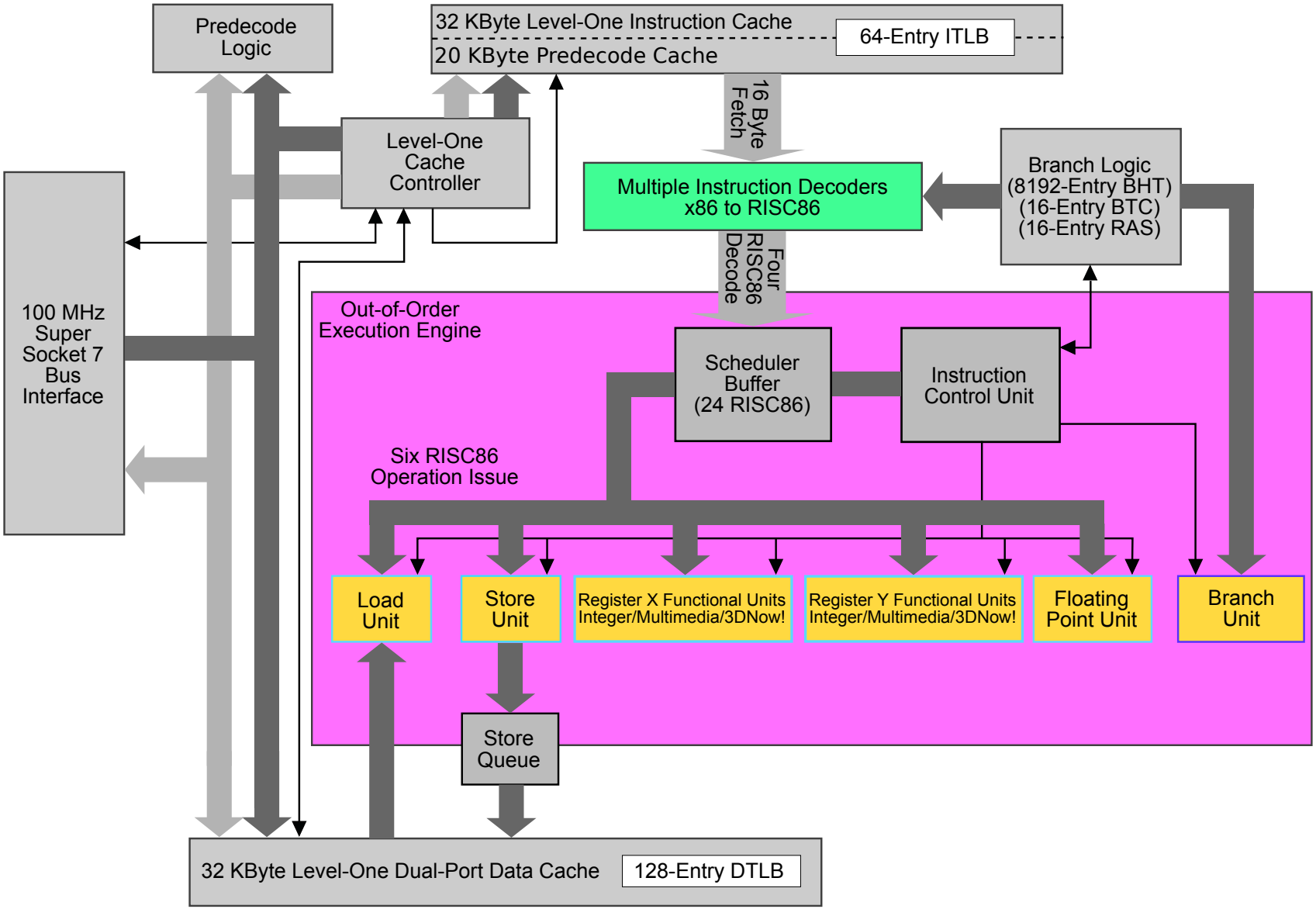


The x86 is dead. Long live the x86!

K6-2

image source: Wikimedia Commons

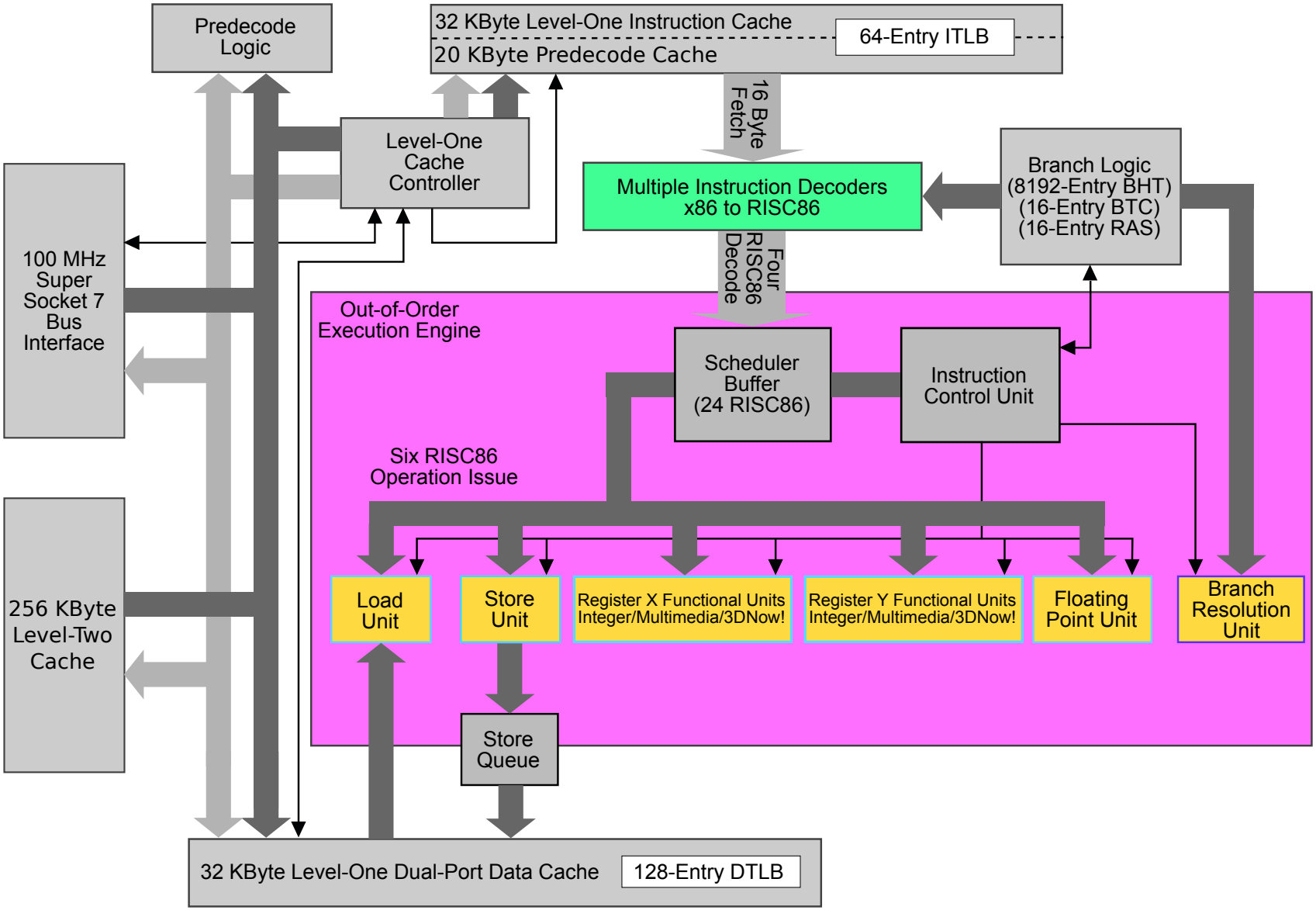
AMD K6-2 Architecture



K6-III

image source: Wikimedia Commons

AMD K6-III Architecture



The x86 is dead. Long live the x86!

SSE2 + long pipelines (FreeBSD 4, Linux 2.4)

Intel NetBurst (P4, PD)

- P4 adds SSE2, HyperThreading, quad-pumped FSB

AMD K7 (Athlon)

- Alpha's EV6 bus, 3-way FPU fixes AMD FP
- Enhanced 3DNow! Palomino gets SSE2
- T-Bird adds L2 cache on-die

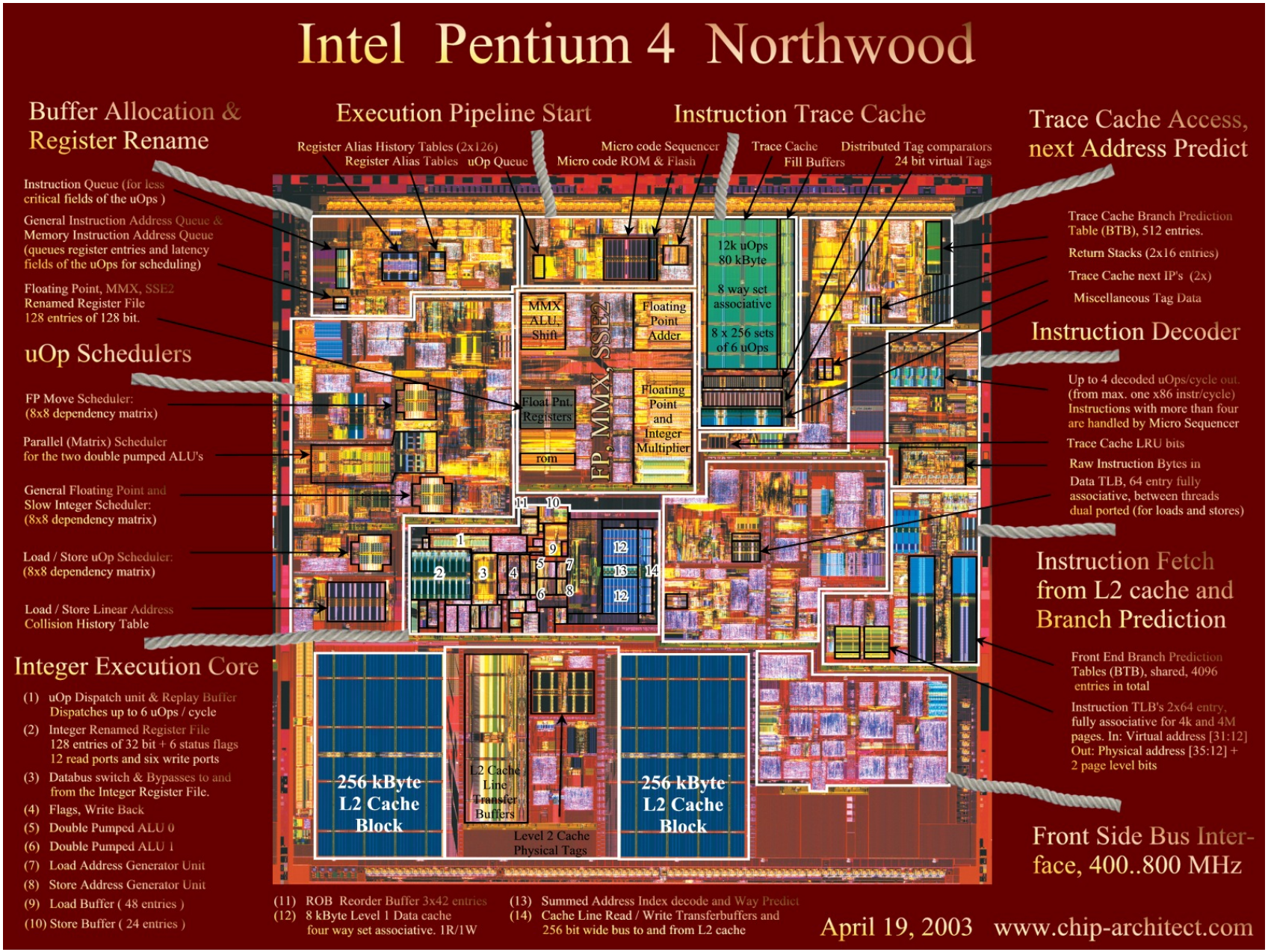
VIA C3+C7 / Transmeta Crusoe+Efficeon

- Who cares? x86 officially becomes a two-party system.

Athlon T-Bird	2000	37M@180nm	10/15	.6–1.4GHz / 333MT/s
Intel Northwood	2003	55M@130nm	20	1.6–3.4GHz / 800MT/s
Athlon Barton	2003	54M@130nm	10/15	1.833–2.333MHz / 400MT/s
Intel Prescott	2004	125M@90nm	31	3–3.8GHz / 800MT/s

Pentium 4 Northwood

image source: www.chip-architect.com



x86-64 multicore (FreeBSD 7, Linux 2.6)

Intel Core

- 45nm Penryn replaces SiO_2 with HfSiON.
- Woodcrest Xeon adds SSSE3

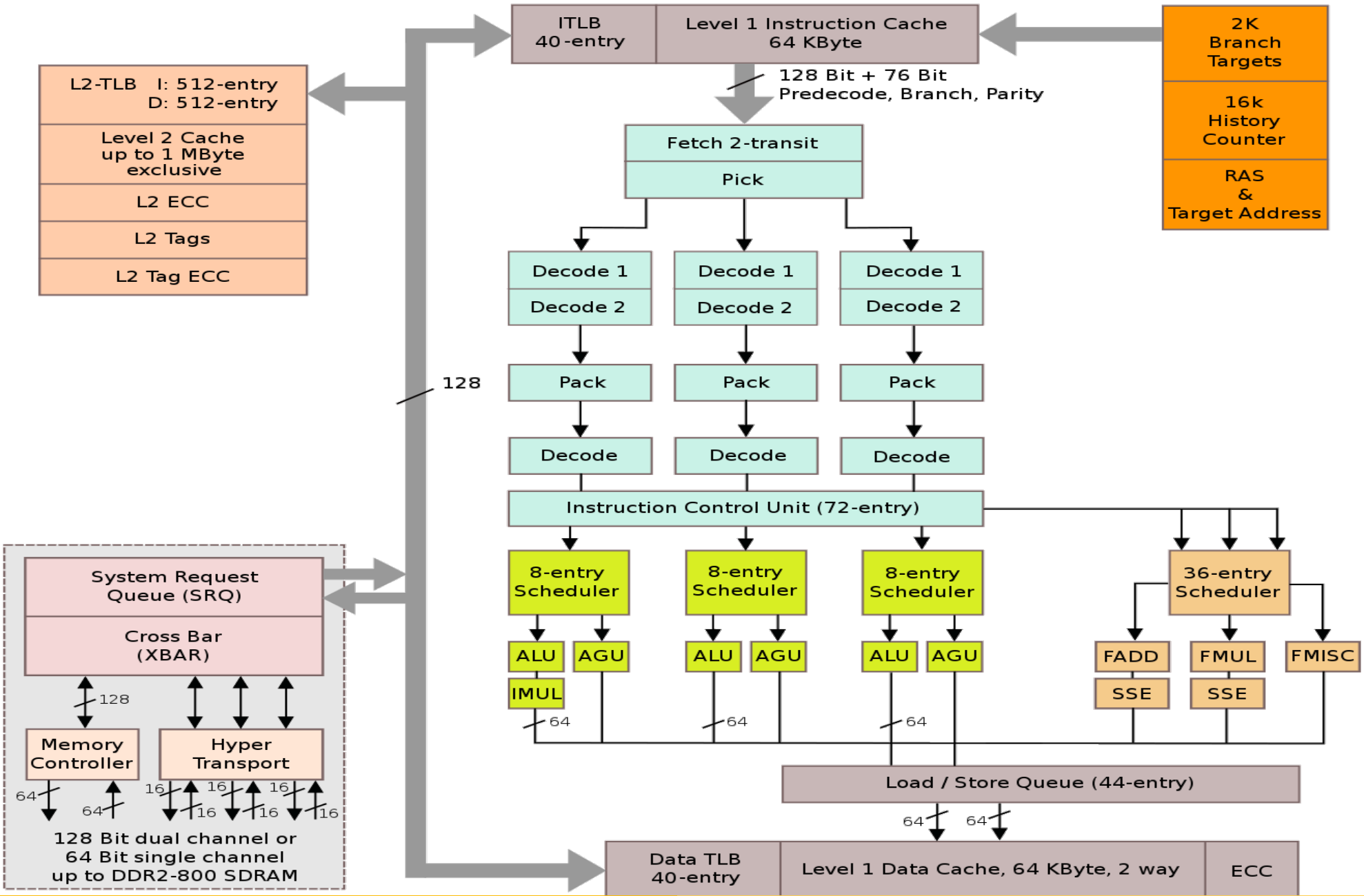
AMD K8

- HyperTransport @ 1GHz on Socket 939
- San Diego adds SSE3, IOMMU
- Quad FX (Windsor) goes NUMA on HT2

AMD San Diego	2005	114M@90nm	12/17	2.2–2.6GHz / 1GHz
Intel Conroe	2006	291M@65nm	14	1.6–2.6GHZ / 533–800MHz
AMD Windsor	2007	154M@90nm	12/17	2–3.2GHz / 2GHz
Intel Penryn	2008	410M@45nm	14	1.2–3.06GHz / 800–1333MHz

AMD K8 Opteron

image source: Wikimedia Commons



The x86 is dead. Long live the x86!

AVX + scaling + heterogeneity (FreeBSD 8, Linux 3)

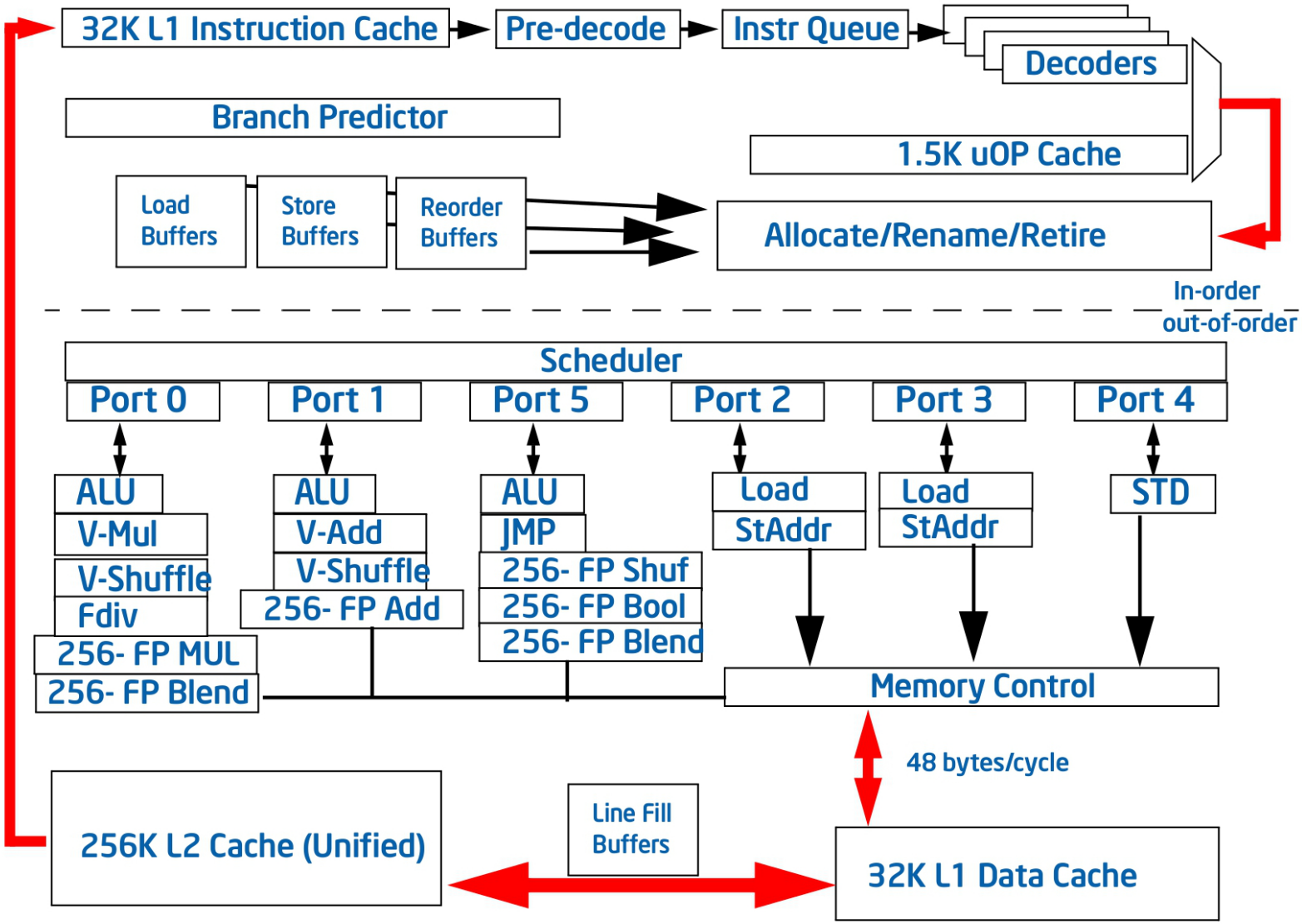
Recent Intel

- Return of the barrel shifter (and HyperThreading)!
- Nehalem adds SSE4.2, 2TLB, VT-d, integrates northbridge
- Quick Path Interconnect on Nehalem-E
- Sandy Bridge integrates GPU, GMCH, L3, 128-bit AVX
- Ivy Bridge adds PCIe 3.0, FinFET trigates
- Haswell adds transactional memory, 256-bit AVX2, FMA3

Intel Nehalem	2008	731M@45nm	16	2–3.33GHz
Intel Sandy Bridge	2011	1.2B@32nm	14/19	2.3–4GHz
AMD Bulldozer	2011	1.2B@32nm	15/20	3–4.6GHz
Intel Haswell	2013	1.5B@22nm	14/19	2.3–3.9GHz

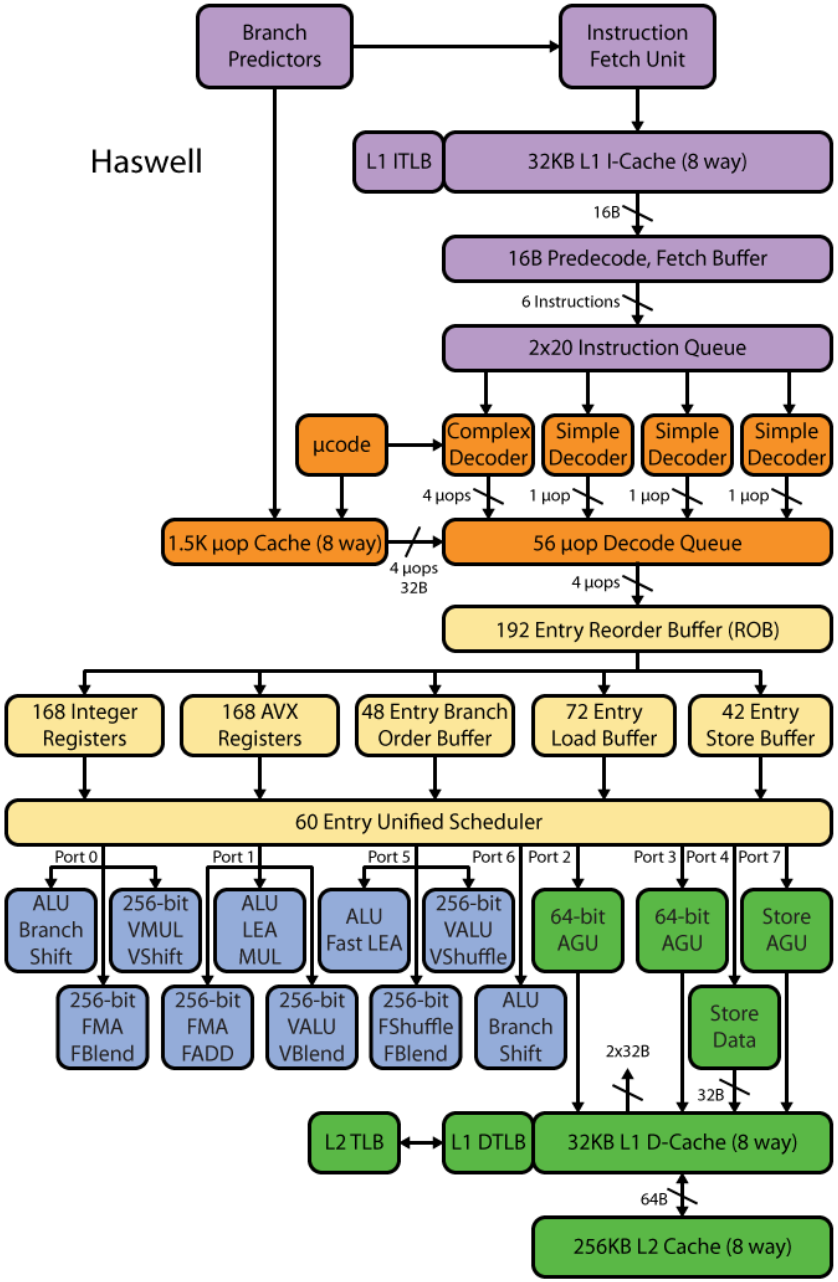
Sandy Bridge (2011)

image source: *Intel Optimization Manual*



Haswell (2013)

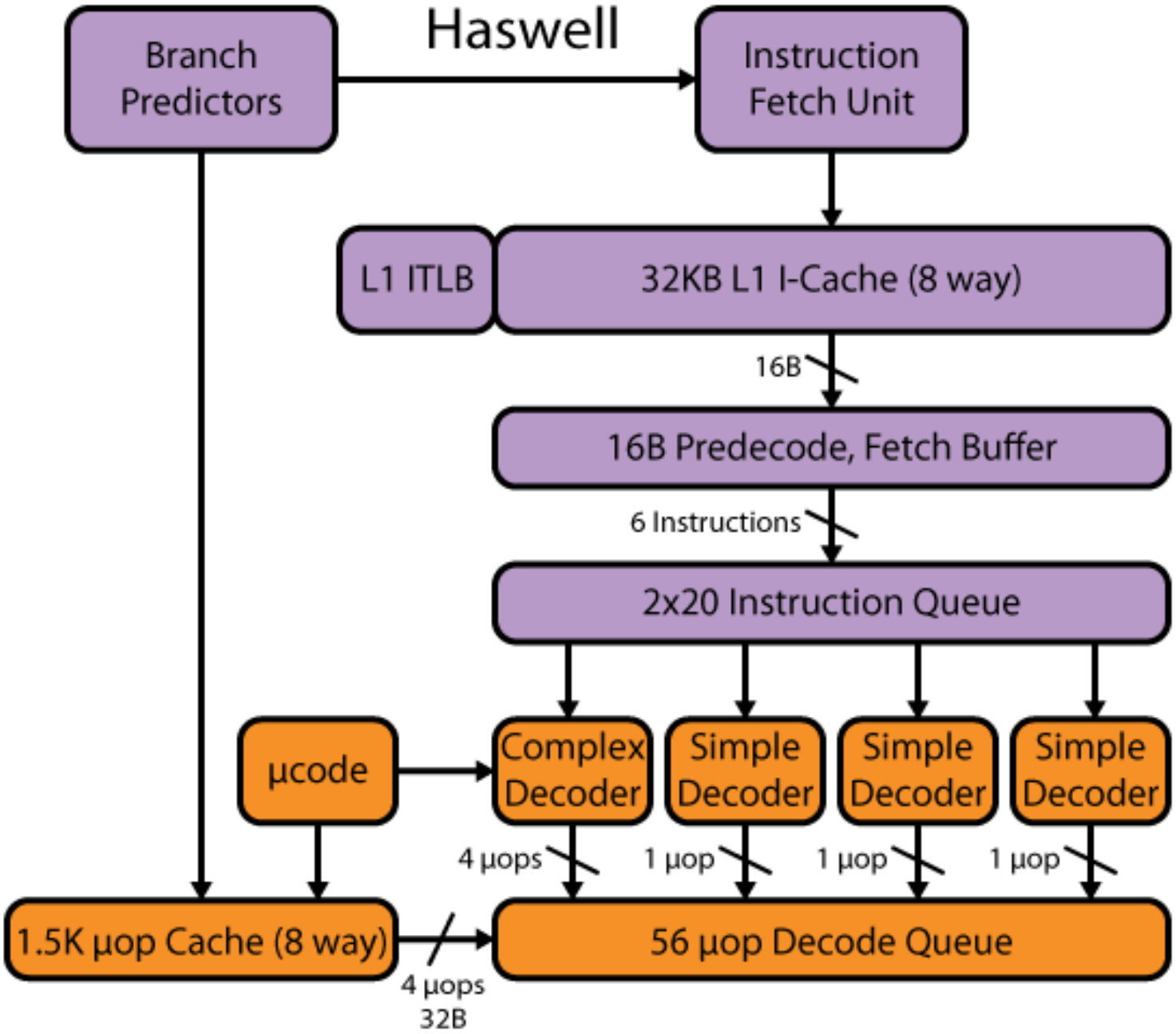
image source: Real World Technologies



The x86 is dead. Long live the x86!

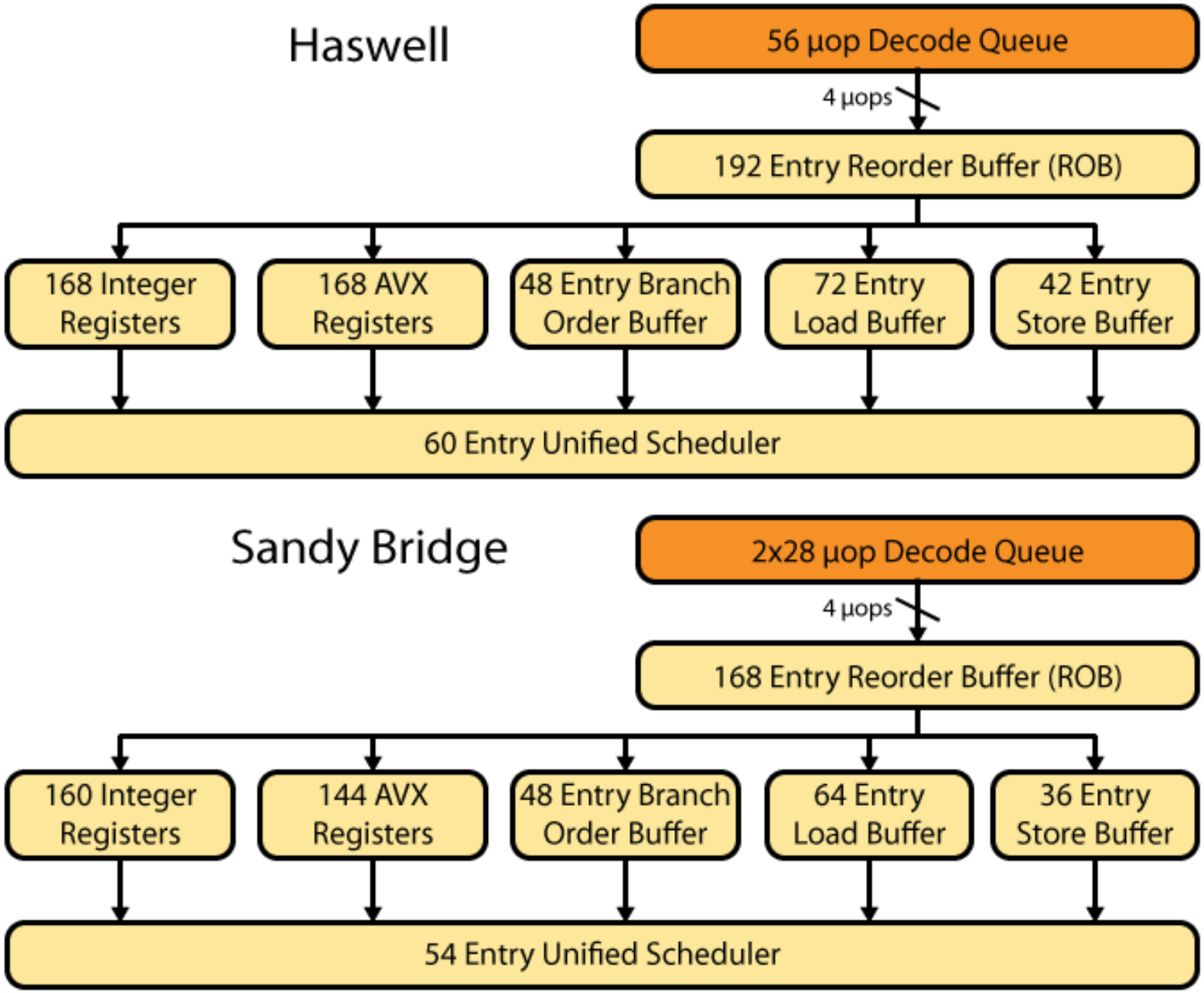
Decoding

image source: Real World Technologies



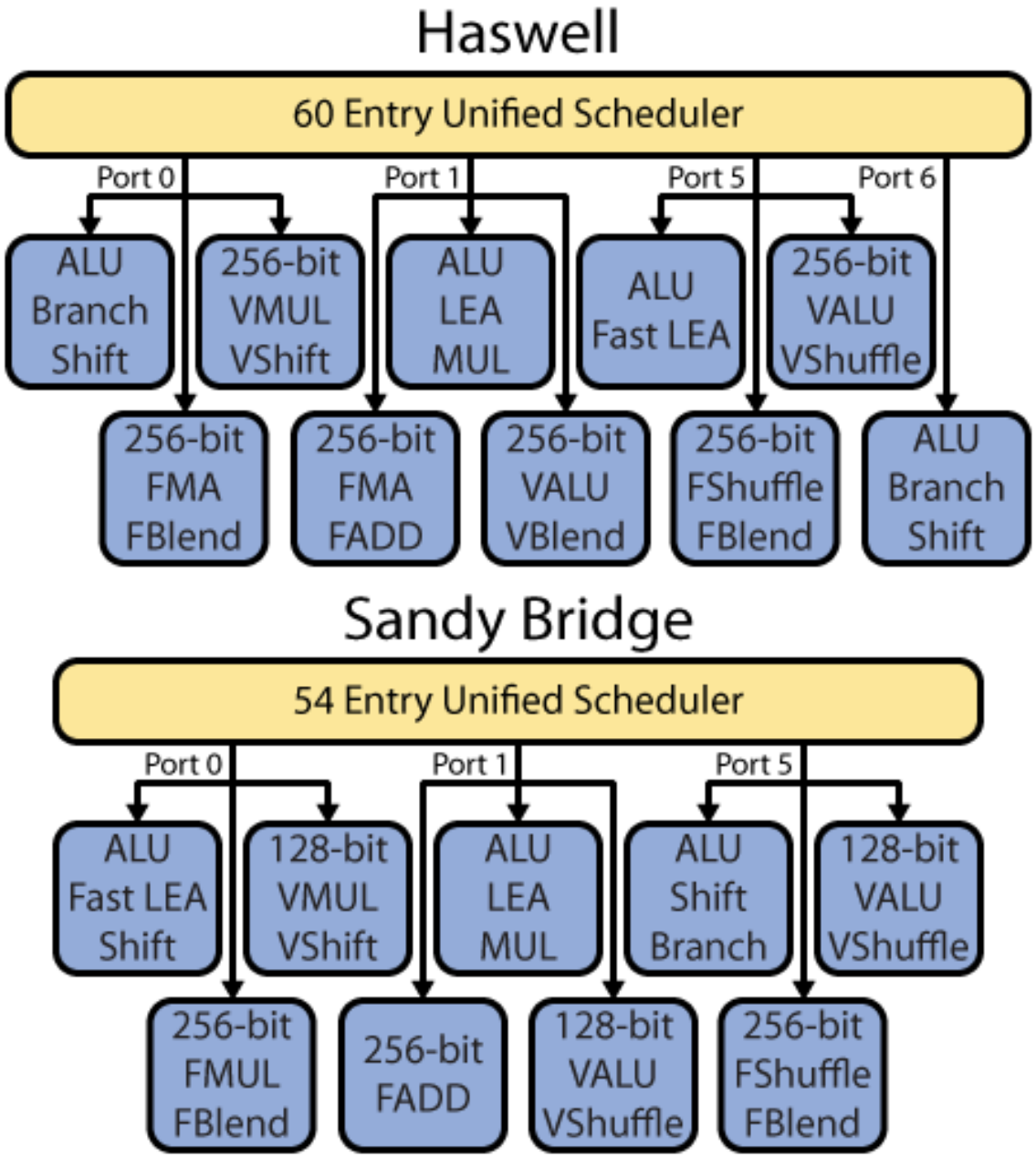
Schedulers

image source: Real World Technologies



Execution

image source: Real World Technologies



Write order with store-buffer forwarding

Very rigid memory model for all but non-temporal accesses[3]:

- Atomics are never reordered
- The pipeline and I$ are coherent
- Stores can be reordered after loads, but not before
- Accesses to uncacheable memory are never reordered
- Loads can be reordered after stores to other locations
- String ops, streaming stores, non-temporal moves and CLFLUSH can reorder writes among themselves

Among multiple (logical or physical) processors:

- Individual processors maintain this model
- Writes by a single processor are seen in the same order by all processors
- Writes among processors are not totally ordered, unless locked
- Memory ordering obeys causality
- Stores are externally consistent

Implication: concurrent stores by multiple processors will result in each processor seeing its own write occur first!

[3] Centaur Winchip implemented "x86 oostore", with less strongly ordered memory.

Task model

Hardware-recognized task state is stored in Task State Segments:

- Segment selectors of segment registers and LDTR
- GP registers, EFLAGS, EIP, CR3, TR
- I/O map base address and I/O map
- Stack pointers for CPL0, 1, 2 stacks
- Link to previously executed task

Share among tasks via three methods:

- Through the GDT segment descriptors
  (share among all processes)
- Through a shared LDT
  (share among some cooperating processes)
- Through shared descriptors in distinct LDTs mapped to a common address in linear address space
  (share among some non-cooperating processes)

Simultaneous multithreading

HyperThreading, when activated by BIOS and OS, results in:

Replication Registers, renamed RSB, large page ITLB

Partition Load/store buffers, ROB, small page ITLB

Competition Reservation stations, cache[4], DTLB, STLB

[4] Actually, L1 D$ can be non-competitively shared between logical cores in *adaptive mode*, so long as their CR3 registers match, their paging modes match, and bit 24 of the IA32_MISC_ENABLE MSR isn't set.

Immediate state

Architectural component	Count
64-bit x86-64 GP / segment registers	16 / 6
80-bit x87/MMX registers	8
AVX/XMM registers	16
Special-purpose	2 (EFLAGS, EIP)

μarchitectural component	Haswell	Sandy Bridge	Nehalem
Load buffers	72	64	48
Store buffers	42	36	32
PRF entries (x86-64+AVX)	168+168	160+144	N/A
Line fill buffers	10	10	10
Reservation stations	60	54	34
Reorder buffer entries	192	168	128

Instruction store

- 32KB instruction cache, 16B/**c** $\Longrightarrow$ 48GB/s @ 3GHz
- 8-way on Sandy Bridge, 4-way on Nehalem
- 128 4K ITLB entries, 8 large ITLB entries[5]
- MSROM 4μop/**c**
- 1.5Kμop trace cache[6]

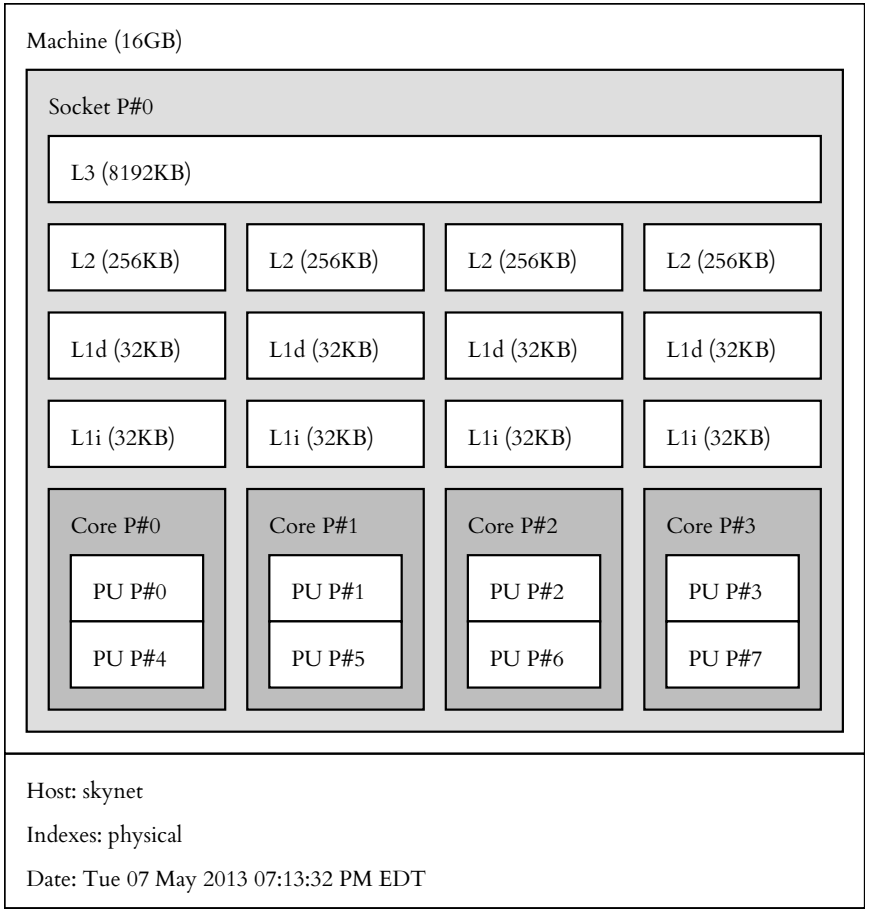
[5] 7 on Nehalem

[6] The Pentium 4 trace cache was a whopping 12Kμops.

On-die data store

The latencies below are best cases (simple address mode, no stalls, no eviction, etc.). MESI details are stored in L1/L2, while directory details are kept in the LLC (hence the latter's inclusivity).

- 32KB 8-way 64B/line data cache, 4**c** latency, 2x16B/**c**
- Sandy Bridge provides 32B/**c** L1 load, Haswell 64B/**c** L1/L2 load and 32B/**c** L1 store
- 256KB 8-way 64B/line unified non-inclusive L2, 12**c** (10**c** on Nehalem), 32B/**c**
- Variable 64B/line unified inclusive L3, 26–31**c** (35–40**c** on Nehalem), 32B/**c**
- 43**c**/60**c** for clean/dirty forward



`lstopo(1)` output

TLBs and Memory interface

- 4-way DTLB translates up to 2 loads/**c** and 1 store/**c**
- DTLB: 64 4KB entries, 32 2MB/4MB entries, 4 1GB entries
- 4-way unified STLB with 512[7] 4KB entries, 7**c** latency
- L1: IP-based stride prefetcher, DCU streaming prefetcher
- L2: Spatial prefetcher, streaming prefetcher
- Nehalem: 3x 8B DDR3 @ 1.333GT/s (31.992GB/s)
- Sandy Bridge-E: 4x 8B DDR3 @ 1.6GT/s (51.2GB/s)
- Speculative reads allowed from WB, WC, and WT memory

[7] 1024 on Haswell

Instruction set I—Encoding

x86-64 instructions can be up to 15 bytes.

- Up to four *legacy prefixes*:
  - LOCK/REPNE/REPNZ/REP/REPE/REPZ
  - Segment overrides and branch hinting
  - Operand-size override
  - Address-size override
- Opcode with possible *family prefix* (REX, VEX, XOP)
- Possible 1-byte ModR/M (operand encoding, register/memory)
- Possible 1-byte SIB (scale-index-base)
- 0, 1, 2, 4, or 8-byte displacement
- 0, 1, 2, 4, or 8-byte immediate

Recall that 4 instructions per 16B/**c** fetch is necessary for peak decoder throughput (but this is not requisite for peak issue rate).

Instruction set II—Memory

Non-temporal accesses bypass cache and do not require RFOs; they are write-combining, write-collapsing, and weakly ordered. They do not take PATs into account. When properly bunched into cacheline-length chunks, they can deliver the full bus bandwidth. *Prefetches* (as opposed to preloading) do not consume a register, do not stall retirement, do not split cache lines, and cannot fault[8]. *Fence* instructions ensure coherence of data.

MOVNT*[9]	_mm_stream_*	Streaming store to mem
MOVNTDQA	_mm_stream_load_*	Streaming loads from USWC MMIO
PREFETCHNTA	_mm_prefetch(NTA)	Prefetch into second-level cache
PREFETCHNT0	_mm_prefetch(T0)	Prefetch into all cache levels
MASKMOVD*	_mm_maskmove_*	Bytemasked store to mem
SFENCE/LFENCE	_mm_sfence/_lfence	Order point for stores/loads
MFENCE	_mm_mfence	Order point for stores and loads[10]
WBINVD	"wbinvd":::"memory"	Write back, and invalidate cache
INVD	"invd"	Screw it, just invalidate the caches
INVLPG	"invlpg"	Invalidate TLB entries containing address
INVPCID	"invpcid"	Invalidate TLB entries based on PCID
CLFLUSH	"clflush %0":"+m"	Cacheline invalidation

[8] CPUID.01H.EBX holds the stride length (D$/U$ only).

[9] See also AMD's SSE4a's MOVNTSD/MOVNTSS.

[10] Not equivalent to SFENCE+LFENCE, which would be mutually unordered! Only MFENCE orders CLFLUSH.

Instruction set III—Some control instructions

HALT	Halt the processor
MWAIT (SSE3)	Monitor a cache line (_mm_mwait)
CCMOVxx	Conditional move (control dep to data dep)
JMP	Unconditional branch
Jxx / Jx	Conditional branches
CALL	Task switch
IRET	Return from far call

Instruction set IV—Some arithmetics

VFMADD*[PS][DS]	_mm_dp_p*	FP fused multiply-add
VFNMADD*[PS][DS]	_mm_dp_p*	FP fused multiply-negate-add
VFMSUB*[PS][DS]	_mm_dp_p*	FP fused multiply-sub
VFNMSUB*[PS][DS]	_mm_dp_p*	FP fused multiply-negate-sub
VFMADDSUB*[PS][DS]	_mm_dp_p*	FP fused multiply-sub/add
VGATHER[DQ]D	_m256i_mm256_add_epixx	Gather ints from memory
VGATHER[DQ]P[SD]	_m256i_mm256_add_epixx	Gather FPs from memory
VPADD[BWDQ]	_m256i_mm256_add_epixx	Packed integer add
VPADDS[BW]	_m256i_mm256_adds_epixx	Packed sinteger add, signed saturation
VPADDUS[BW]	_m256i_mm256_adds_epuxx	Packed unsinteger add, unsigned saturation
VPAND/VPANDN	_m256i_mm256_and[not]_si256	Logical AND / NOT+AND[11]
VPAVG[BW]	_m256i_mm256_avg_epuXX	Integer average
VPBLEND[BW]	_m256i_mm256_blend_epiXX	Blend
VPERM[DQ]	_m256i_mm256_perm_epuXX	Permute integers
VPERMP[SD]	_m256i_mm256_perm_epuXX	Permute FPs
VPSLLV[DQ]	_m256i_mm256_perm_epuXX	Logical shift left
VPSRAV[DQ]	_m256i_mm256_perm_epuXX	Arithmetic shift right
VPSRLV[DQ]	_m256i_mm256_perm_epuXX	Logical shift right
PCMPcc[BWDQ]	_m256i_mm256_cmpCC_epiXX	Compare packed integers
PMULDQ/LD	_mm_mul*_epi32	Packed 32-bit integer multiply

[11] "Material nonimplication" aka $\nrightarrow$.

Recommended reading

- Mel Gorman. "Huge Pages in Linux" LWN, in five parts (2010).
- Owens et al. "x86-TSO: A Rigorous and Usable Programmer's Model for x86 Multiprocessors" (2010).
- Bhandarkar and Clark. "Performance from Architecture: Comparing a RISC and a CISC with Similar Hardware Organization" (1991).
- Blem et al. "Power Struggles: Revisiting the RISC vs. CISC Debate on Contemporary ARM and x86 Architectures" (2013).
- Michael Thomadakis. "The Architecture of the Nehalem Processor and Nehalem-EP SMP Platforms" (2011).
- Neil Dickson. "Simple, Linear-Time x86 Jump Encoding" (2011).
- http://www.sandpile.org, http://www.realworldtech.com, and http://www.cpu-world.com
- *Intel 64 and IA-32 Architectures Instruction Reference*.
- *CUDA Parallel Thread Execution ISA Reference*.

Hack on!

"We still have judgment here, that we but teach
Bloody instructions, which, being taught, return
To plague th' inventor"

—Shakespeare, *Macbeth*

"Each day the mythical return Enzian dreamed of seems less possible. Once it was necessary to know uniforms, insignia, airplane markings, to observe boundaries. But by now too many choices have been made. The single root lost, way back there in the May desolation. Each bird has his branch now, and each one is the Zone."

—Thomas Pynchon, *Gravity's Rainbow*