CSUN | CALIFORNIA STATE UNIVERSITY NORTHRIDGE

COMP 122

DR JEFF SOFTWARE
INDIE APP DEVELOPER
© Jeff Drobman
2016-2022

COMP122

Spring 2022 Rev 1-24-22

Lectures on

Part 1

Computer Architecture

& ASSEMBLY Programming

By

Dr Jeff Drobman

website → drjeffsoftware.com/classroom.html

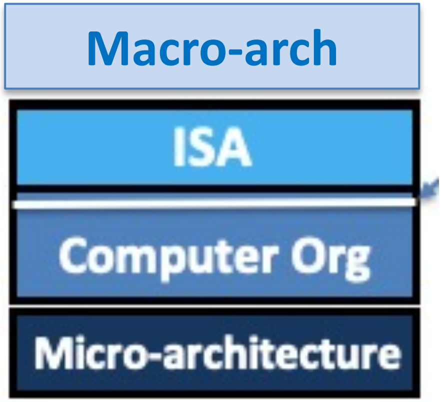
email → jeffrey.drobman@csun.edu

Index

Part 1

Course Description (122)

- ❖ **Computer Architecture/Organization (COMP122/ 222)**
  - ❑ **CPU, FPU, GPU org** (ALU, registers, addressing)
  - ❑ **ISA's: MIPS, ARM, x86**
  - ❑ **Memory models**
    - ▪ MLM- caches
    - ▪ Virtual memory
  - ❑ CPU status (PSW) & clock sync
  - ❑ Interrupts, Exceptions, Syscall
  - ❑ Cores & Threads
  - ❑ Pipelines (ICU)
  - ❑ *Microprogramming (Am2900)*
  - ❑ Logic & State Machines (FSM)
  - ❑ CPU performance/benchmarks
- ❖ **Computer Arithmetic (COMP222)**
  - ❑ **ALU**: Full adder
  - ❑ Mult/Div (Booth's algorithm)
  - ❑ Error codes (ECC, CRC, parity)
- ❖ **Parallel & *Micro* Architecture (COMP222)**
  - ❑ Multi-core, Multi-threading, *superscalar*
  - ❑ SIMD/MIMD/SPMD



System arch – *Cores*

Instructions (Primitives)
Software *Interface*

Execution Units
- ❖ ALU, ICU, Reg

Low-level execution
- ❖ Pipelines, threads
- ❖ scheduling
- ❖ branch prediction

(COMP122)

- ❖ Software Tools
  - ❑ IDE's/Assemblers
  - ❑ OS, RTOS, ***Monitors***
  - ❑ ***Simulators***
- ❖ Debug Tools
  - ❑ ICE/Logic Analyzers
  - ❑ *Disassemblers*

Section

Intro
Models

Realms of Software

~70% of all software

- Applications
  - ❑ Desktop
  - ❑ Mobile (Apps)
  - ❑ Web
- Web
  - ❑ Markup
  - ❑ Applications
  - ❑ SQL databases
- Embedded Control
  - ❑ Small (8-bit)
  - ❑ Medium (16-bit)
  - ❑ Large (32/64-bit)

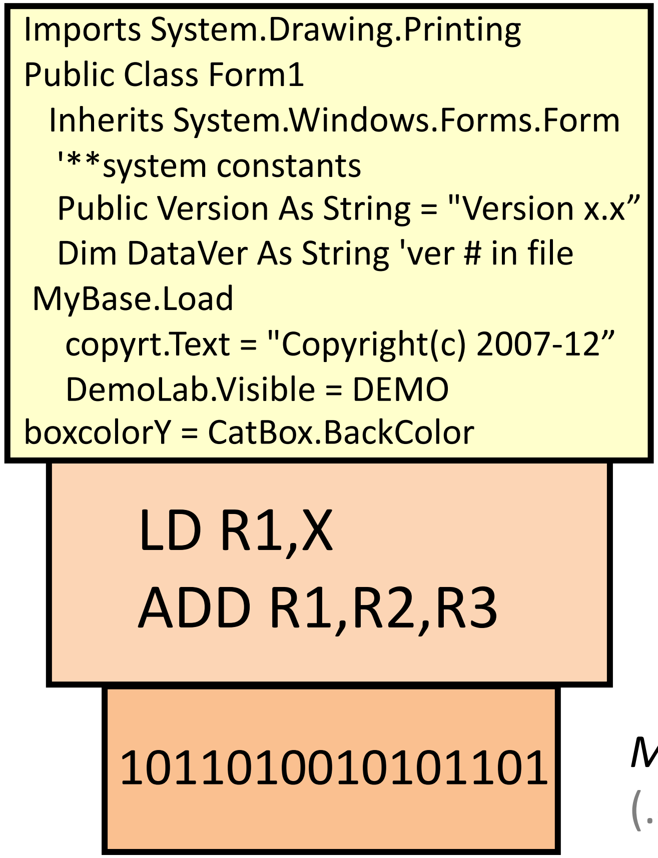
- APIs (Frameworks)

- Client-Server model
- Language "stacks" (e.g., LAMP)

- From TV remotes to
- Autonomous cars and
- Robots

- Common required properties
  - Performance
  - Reliability (bug free)
  - *Security*

Software *Levels*



High-Level

```
Imports System.Drawing.Printing
Public Class Form1
   Inherits System.Windows.Forms.Form
    '**system constants
    Public Version As String = "Version x.x”
    Dim DataVer As String 'ver # in file
 MyBase.Load
     copyrt.Text = "Copyright(c) 2007-12”
     DemoLab.Visible = DEMO
boxcolorY = CatBox.BackColor
```

Human readable
(.htm, .js, .php, .vb files)

Assembly

```
LD R1,X
ADD R1,R2,R3
```

hybrid
(.asm files)

Machine
(Binary)

```
1011010010101101
```

Machine readable
(.exe files)

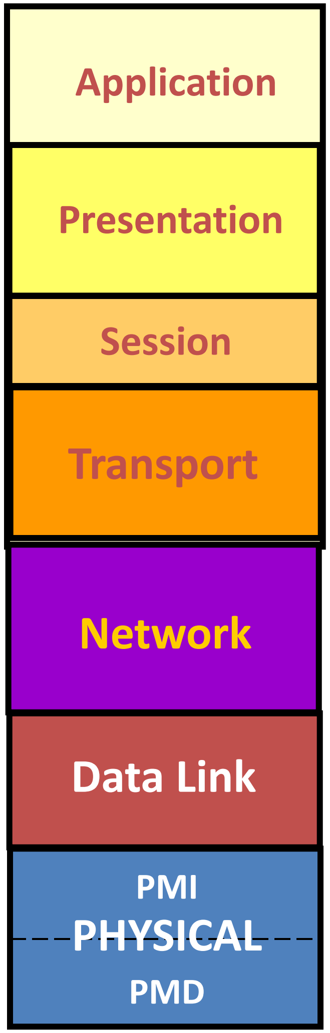
CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE
COMP122

Com Protocol Layers

DR JEFF SOFTWARE
INDIE APP DEVELOPER
© Jeff Drobman
2016-2022

7-Level
OSI MODEL
of **Protocol Layers**
IN COMMUNICATIONS SYSTEMS

Protocol
STACK
Model



LEVEL		***EXAMPLES***
7	**Application**	***Office applications suite, Adobe Acrobat*** ***Internet applications: HTTP, FTP, POP, SMTP***
6	**Presentation**	***SSL, encryption, compression***
5	**Session**	***connections***
4	**Transport**	***TCP, UDP, TLS***
3	**Network**	***IP addressing & routing***
2	**Data Link**	***MAC (Media Access Control)***
1	PMI **PHYSICAL** PMD	***PHY–*** ***CDR*** ***Transceiver*** ***Optical–*** ***Laser diode/LED, Photodetector, TIA, PA***

DIGITAL

ANALOG

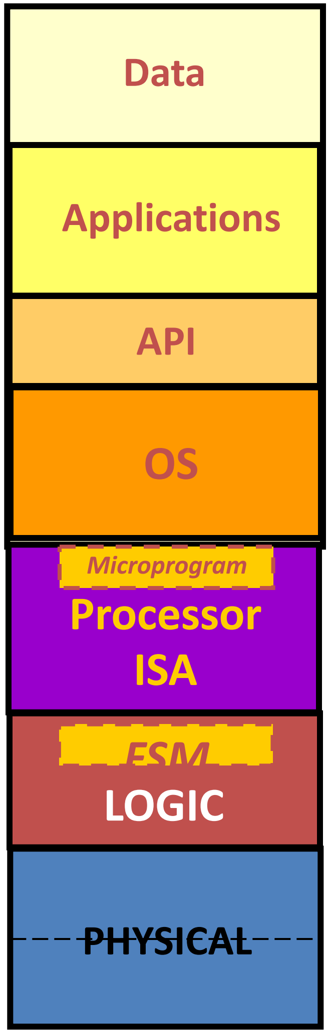
CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE
COMP122

Hardware-Software *Layers*

DR JEFF SOFTWARE
INDIE APP DEVELOPER
© Jeff Drobman
2016-2022

7-Level
STACK HIERARCHICAL MODEL OF
LEVELS OF DESIGN
OF DIGITAL SYSTEMS

STACK Model

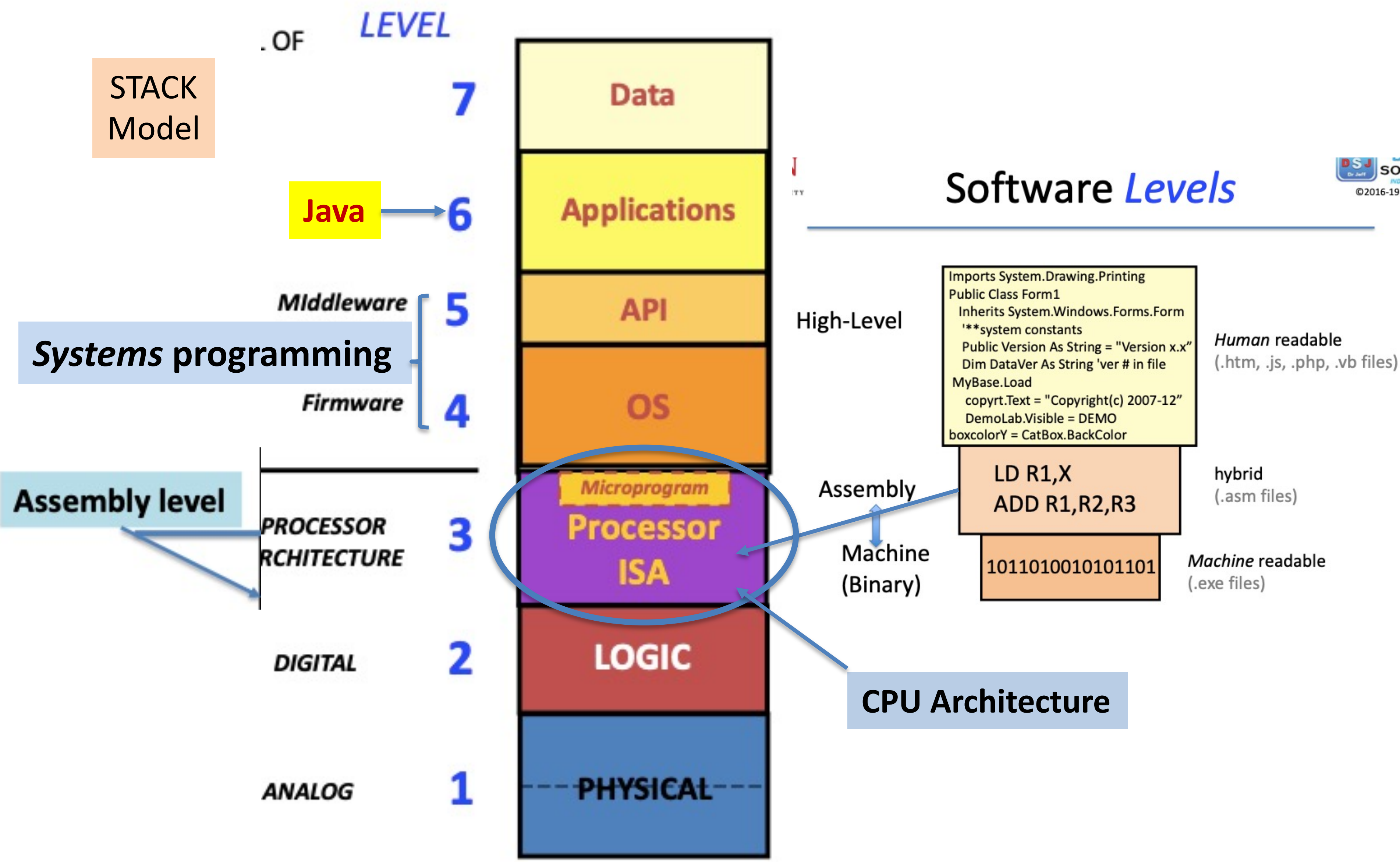


		LEVEL		***EXAMPLES***
		7	**Data**	***.doc, .xls, .sql, .csv, .txt files***
		6	**Applications**	***Microsoft WORD, Excel & "apps"***
SOFTWARE	***MIddleware***	**5**	**API**	***Microsoft .NET, Java Lib Apple Cocoa, Android API***
	Firmware	**4**	**OS**	***Win/Mac OS/Unix iOS, Android***
	PROCESSOR ARCHITECTURE	**3**	***Microprogram*** **Processor ISA**	***Intel/AMD x86 ARM, MIPS, Sparc***
HARDWARE	***DIGITAL***	**2**	***FSM*** **LOGIC**	***FSM= Finite State Machine Logic Design***
	ANALOG	**1**	**PHYSICAL**	***Communications***

ICU

(NOTE: *FIRMWARE* is any embedded software, such as microprograms, monitors, real-time executives, etc.)

Levels of System Architecture



```
Imports System.Drawing.Printing
Public Class Form1
  Inherits System.Windows.Forms.Form
   '**system constants
   Public Version As String = "Version x.x"
   Dim DataVer As String 'ver # in file
 MyBase.Load
    copyrt.Text = "Copyright(c) 2007-12"
    DemoLab.Visible = DEMO
boxcolorY = CatBox.BackColor
```

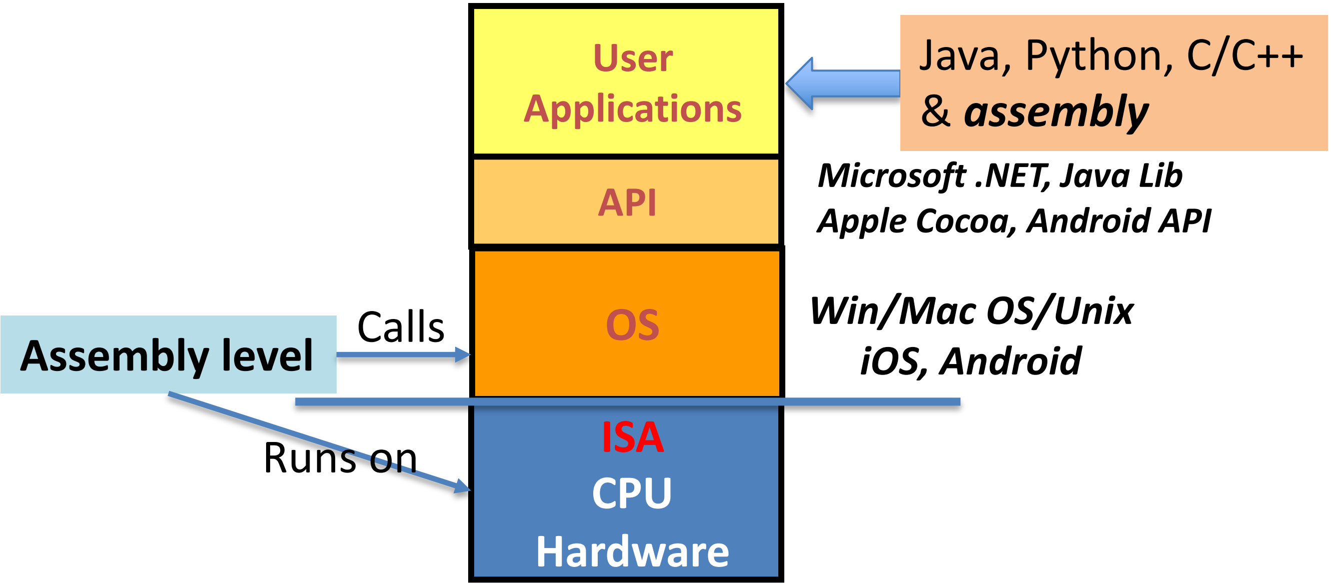
```
LD R1,X
ADD R1,R2,R3
```

```
1011010010101101
```

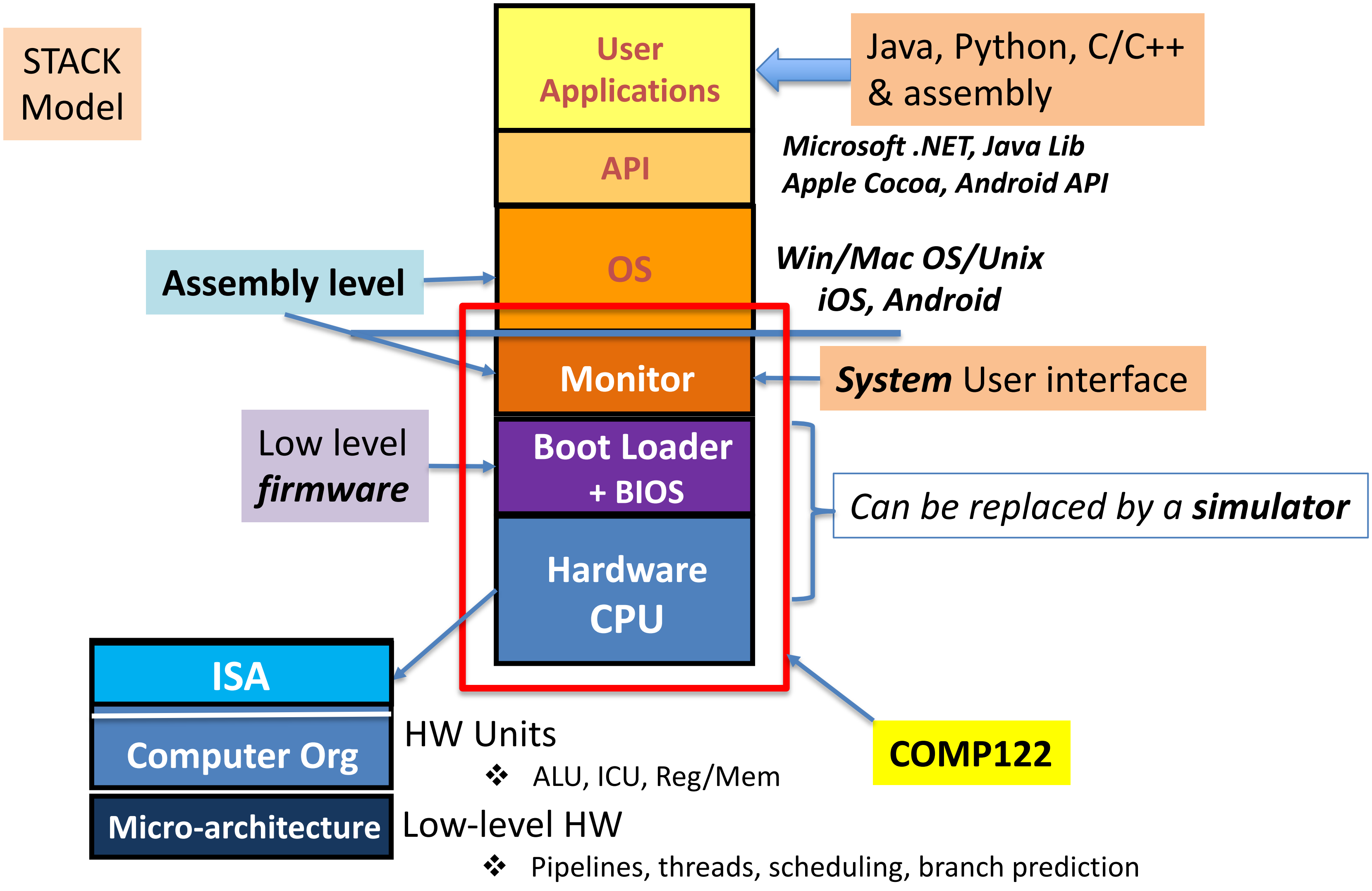
Software *Layers*

STACK Model

Simple View



Hardware/Software *Low Level*



STACK Model

User Applications

API

OS

Monitor

Boot Loader + BIOS

Hardware CPU

Java, Python, C/C++ & assembly

Microsoft .NET, Java Lib
Apple Cocoa, Android API

Win/Mac OS/Unix
iOS, Android

Assembly level

System User interface

Low level ***firmware***

*Can be replaced by a **simulator***

COMP122

ISA

Computer Org

Micro-architecture

HW Units

- ALU, ICU, Reg/Mem

Low-level HW

- Pipelines, threads, scheduling, branch prediction

Chip Specs

- Architectural
- Functional
- Mechanical
- Electrical (DC)
- Timing (AC)
- Thermal (theta JA, JC, CA)

Computer Org

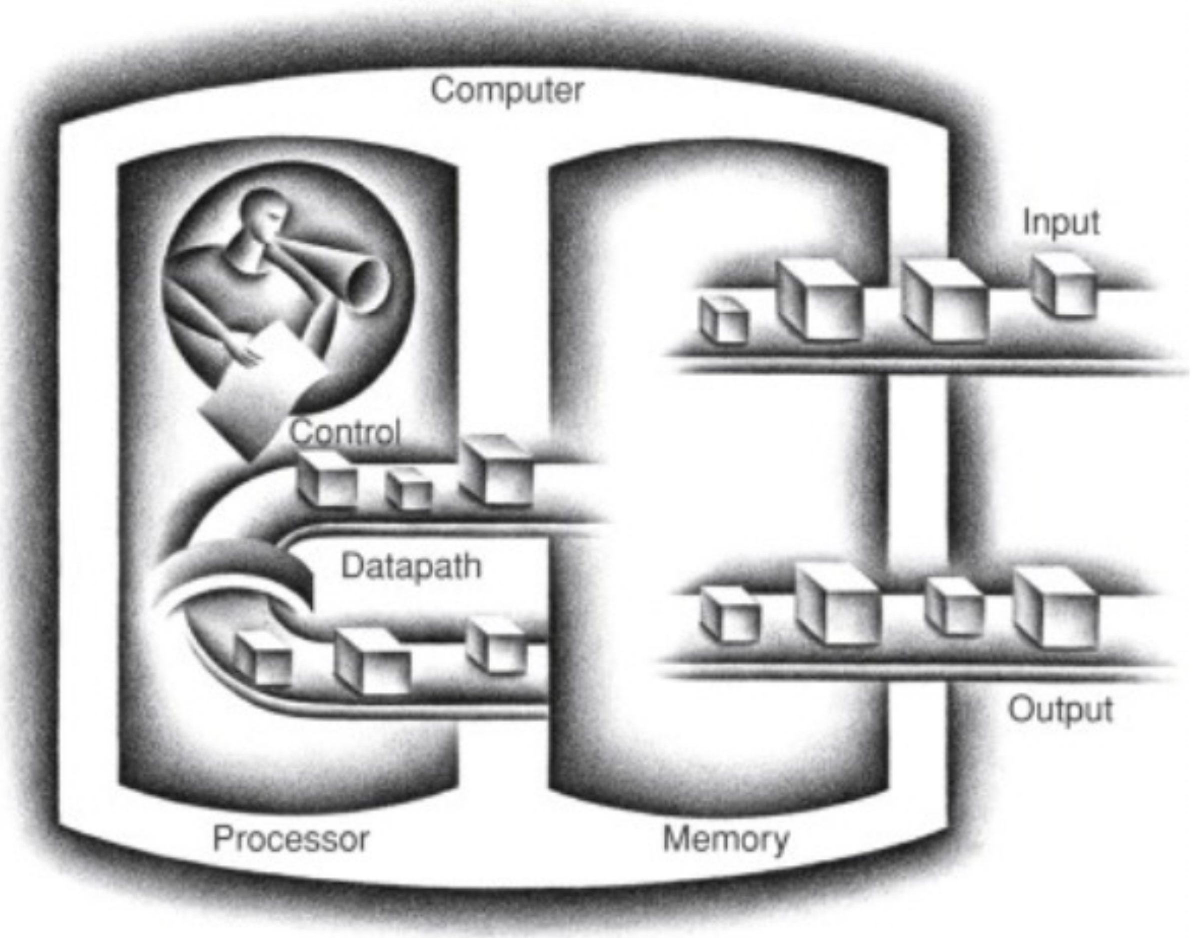
P&H Ch 1

Figure 1.4.1: The organization of a computer, showing the five classic components (COD Figure 1.5).

The processor gets instructions and data from memory. Input writes data to memory, and output reads data from memory. Control sends the signals that determine the operations of the datapath, memory, input, and output.

- ❖CPU
  - ❑ Processor
  - ❑ Memory
  - ❑ I/O

- ❖ Datapath
- ❖ Control

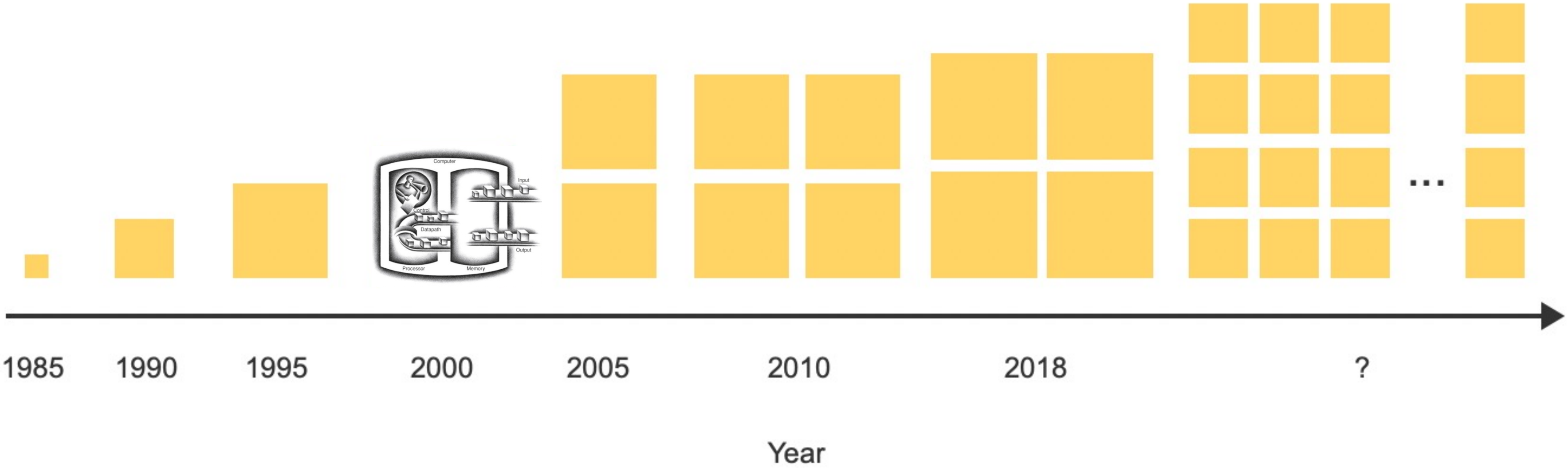


CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE
COMP122

Computer Org: Multi-Core

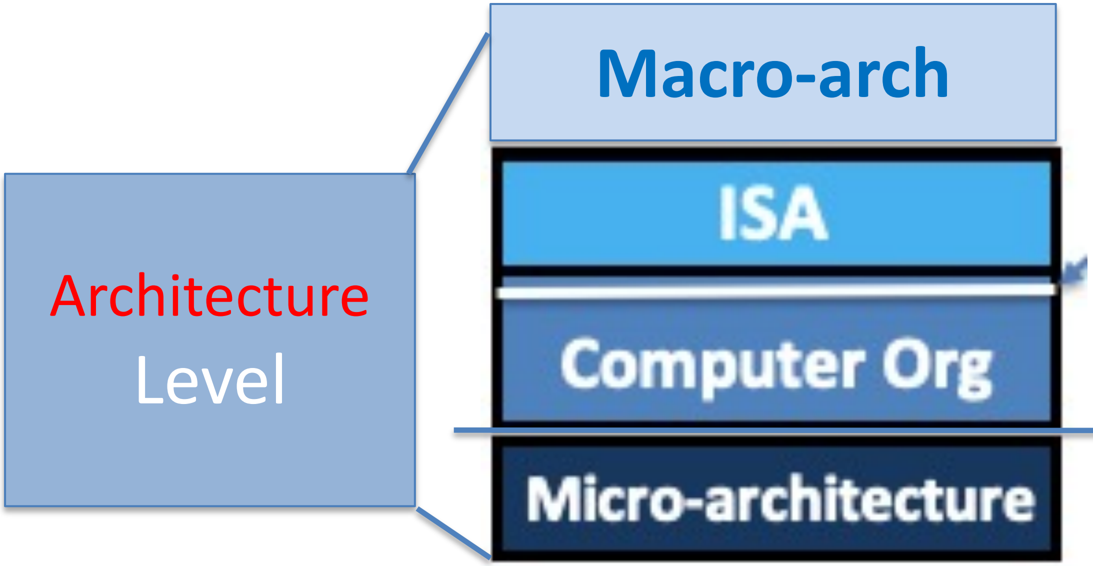
DR JEFF SOFTWARE
INDIE APP DEVELOPER

P&H Ch 1



Computer Architecture

4-Layer Stack Model



Architecture
Level

Macro-arch

ISA

Computer Org

Micro-architecture

System arch – *Cores, caches*

Instructions (Primitives)
Software *Interface*

Execution Units

- ALU, ICU, Reg

122

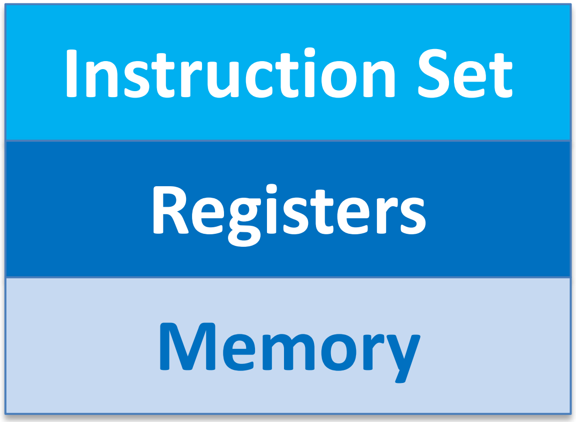
Below “see” level

Low-level execution

- Pipelines, threads
- scheduling
- branch prediction

222

ISA



Instruction Set

Registers

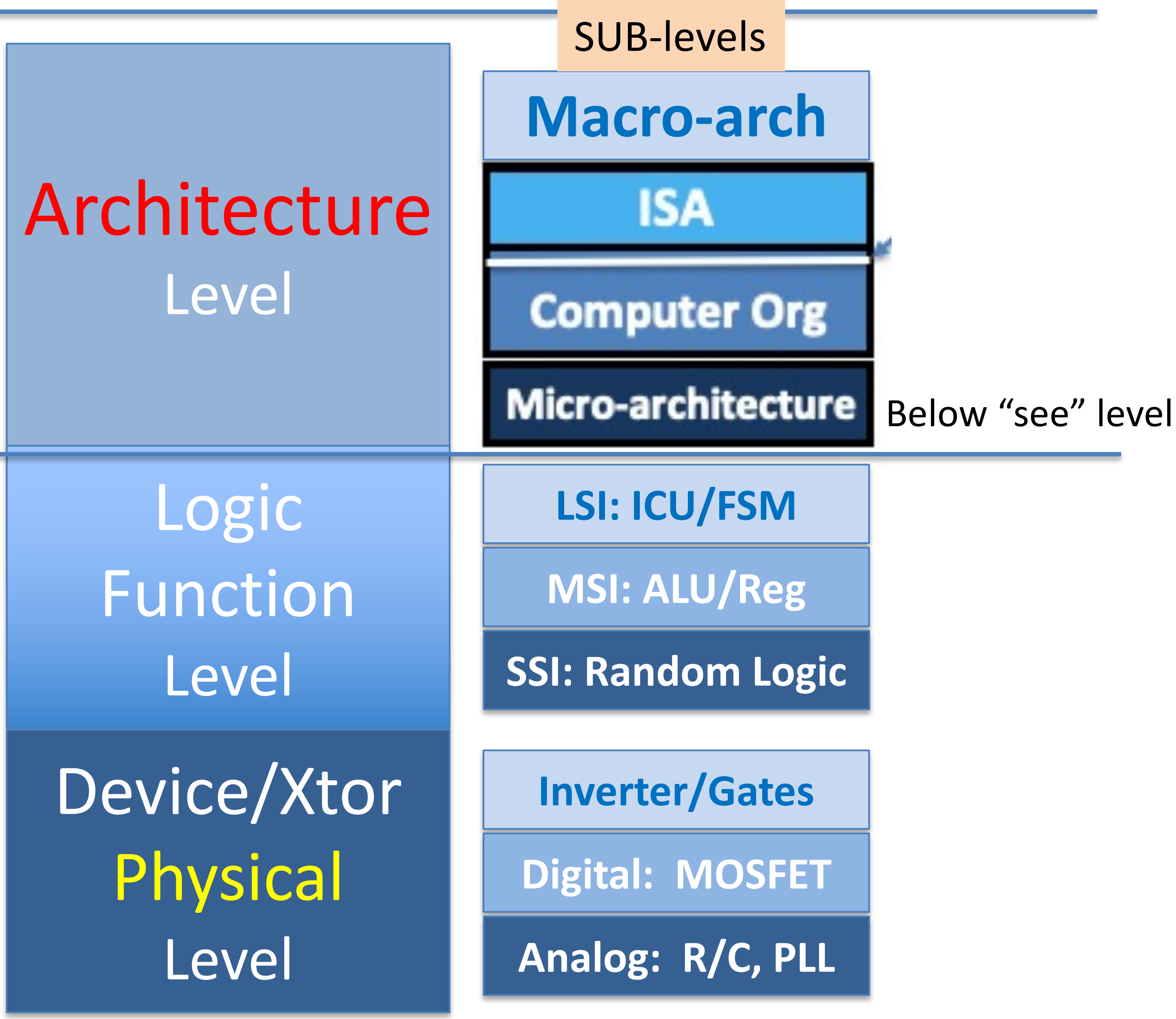
Memory

Instructions (Primitives, pseudos)

GR/Dedicated

Segmentation
Virtual ← →Physical

Transistors to Chips: Levels



SUB-levels

Architecture
Level

Macro-arch

ISA

Computer Org

Micro-architecture

Below “see” level

Logic
Function
Level

LSI: ICU/FSM

MSI: ALU/Reg

SSI: Random Logic

Device/Xtor
Physical
Level

Inverter/Gates

Digital: MOSFET

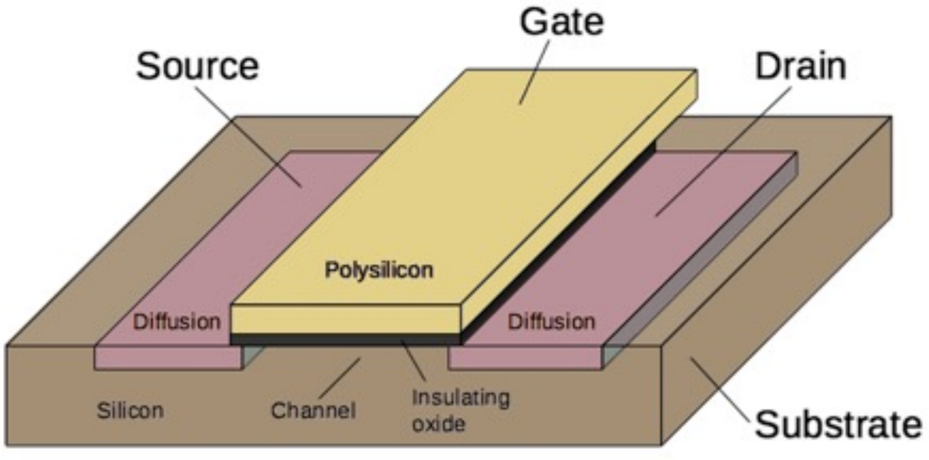
Analog: R/C, PLL

Physical Level: MOSFET

Device/Xtor
Physical
Level

Inverter/Gates

Digital: MOSFET



Structure of a MOSFET in the integrated circuit.

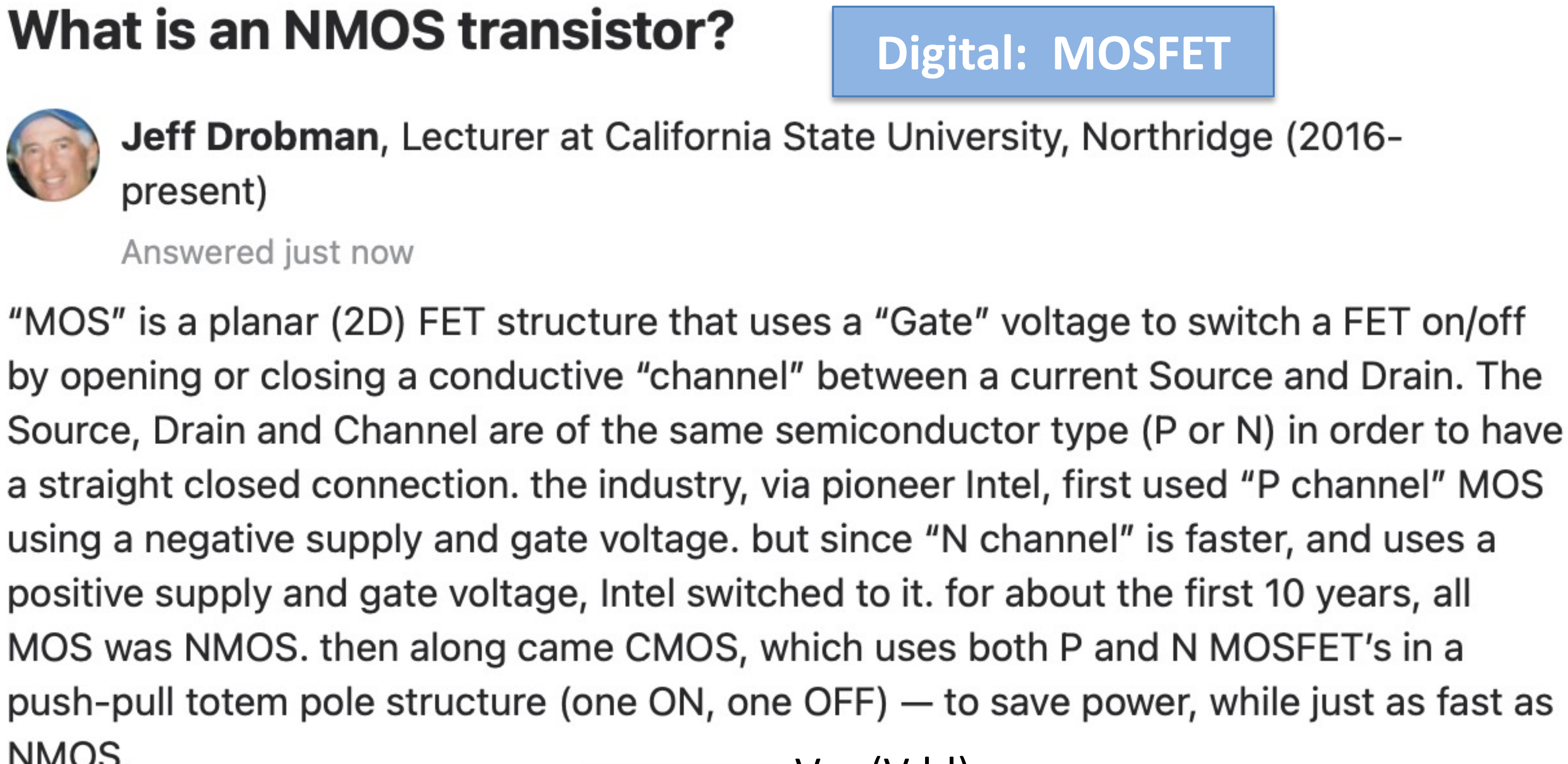
(see separate slide set ***Transistors***)

CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE
COMP122

P→N→C MOS

DR JEFF SOFTWARE
INDIE APP DEVELOPER
© Jeff Drobman
2016-2022

Digital: MOSFET



What is an NMOS transistor?

Jeff Drobman, Lecturer at California State University, Northridge (2016-present)

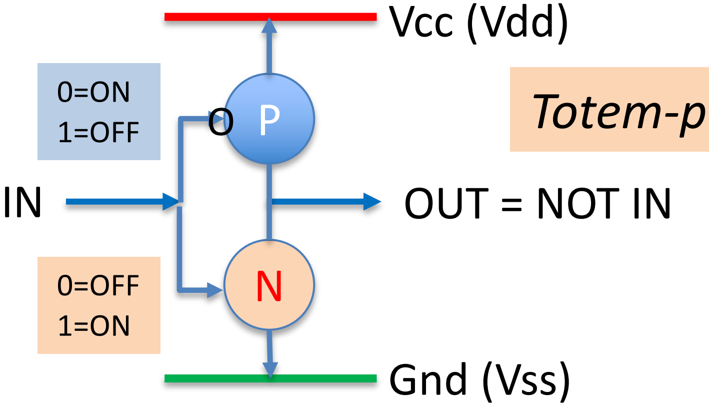
Answered just now

“MOS” is a planar (2D) FET structure that uses a “Gate” voltage to switch a FET on/off by opening or closing a conductive “channel” between a current Source and Drain. The Source, Drain and Channel are of the same semiconductor type (P or N) in order to have a straight closed connection. the industry, via pioneer Intel, first used “P channel” MOS using a negative supply and gate voltage. but since “N channel” is faster, and uses a positive supply and gate voltage, Intel switched to it. for about the first 10 years, all MOS was NMOS. then along came CMOS, which uses both P and N MOSFET’s in a push-pull totem pole structure (one ON, one OFF) — to save power, while just as fast as NMOS.

Complementary

CMOS

INVERTER



Totem-pole

Inverter/Gates

P→N→C MOS

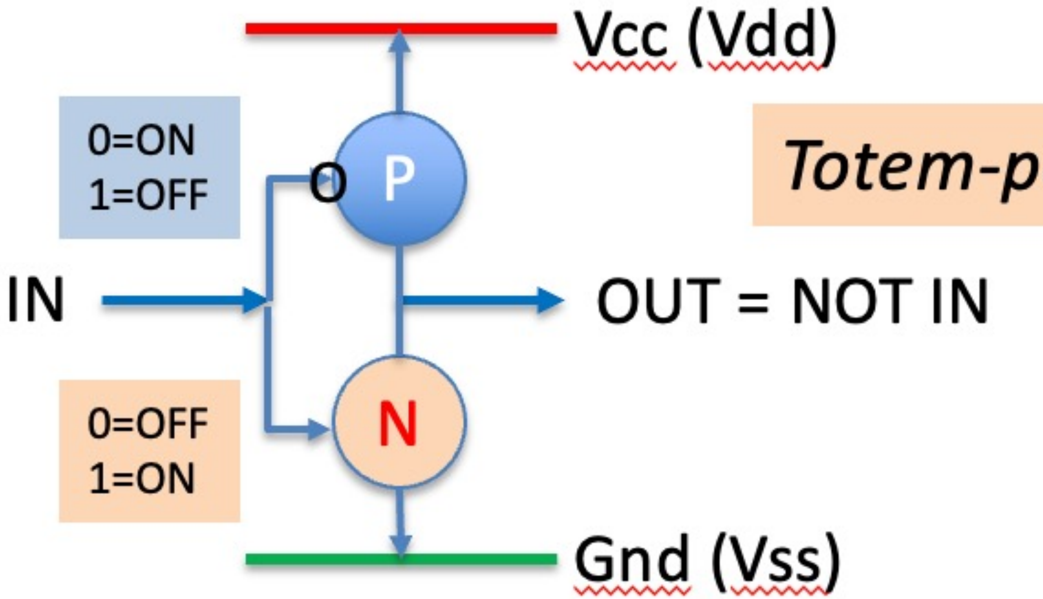
Device/Xtor
Physical
Level

Inverter/Gates

Complementary

CMOS

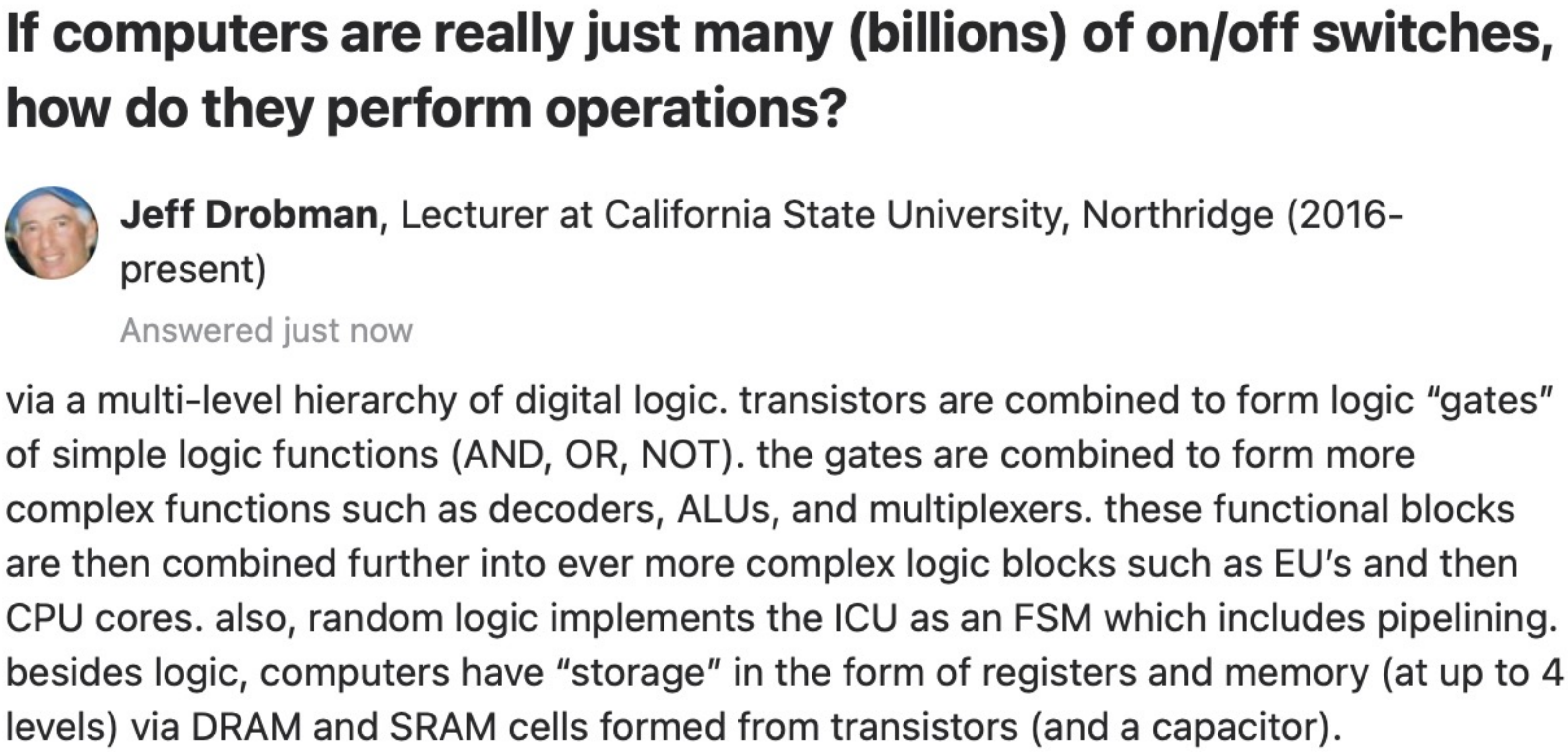
INVERTER



Totem-pole

Transistors to Computers

Quora



If computers are really just many (billions) of on/off switches, how do they perform operations?

Jeff Drobman, Lecturer at California State University, Northridge (2016-present)

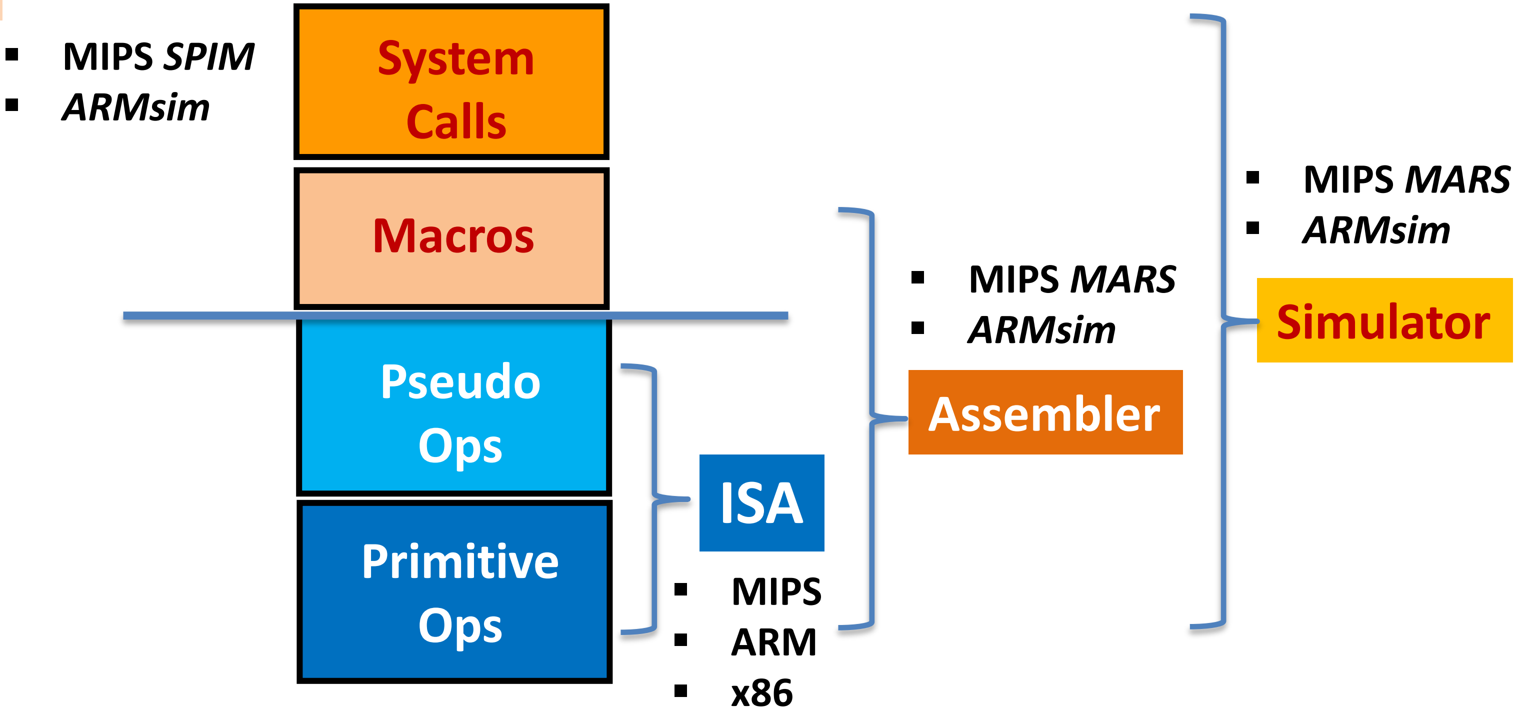
Answered just now

via a multi-level hierarchy of digital logic. transistors are combined to form logic “gates” of simple logic functions (AND, OR, NOT). the gates are combined to form more complex functions such as decoders, ALUs, and multiplexers. these functional blocks are then combined further into ever more complex logic blocks such as EU’s and then CPU cores. also, random logic implements the ICU as an FSM which includes pipelining. besides logic, computers have “storage” in the form of registers and memory (at up to 4 levels) via DRAM and SRAM cells formed from transistors (and a capacitor).

Assembly Level *Layers*

Assembly level

STACK Model



- MIPS *SPIM*
- *ARMsim*

System Calls

Macros

Pseudo Ops

Primitive Ops

ISA

- MIPS
- ARM
- x86

- MIPS *MARS*
- *ARMsim*

Assembler

- MIPS *MARS*
- *ARMsim*

Simulator

Levels of Instructions

Assembly Level Software — Building Blocks

Instruction Set

- Subroutines
  - Block of code that can be "called"
- ***Macros***
  - Block of code that will be substituted *in situ*
- ***Pseudo*** instructions
  - Group of 1 or more ***primitives*** abstracted to higher level
- ***Primitives***
  - Native ***machine instructions*** (in the ISA set)
- *Micro* instructions
  - Complete set of all *control bits* per clock cycle
  - Now = *primitive* (both execute in 1 clock cycle, per *RISC*)
  - Old CISC: Each *primitive* assigned a micro-coded subroutine
  - Can be "horizontal" = Long Instruction Word (VLIW) for *parallel* ISA's

ISA/SoC Landscape

CPU & GPU Cores

Mobile

- ❖ MIPS Focus
- ❖ ARM
  - ❑ **Apple**
  - ❑ Qualcomm
  - ❑ Samsung*
  - ❑ *Google*

Desktop

Server

- ❖ x86
  - ❑ Intel*
  - ❑ AMD

- ❖ ARM
  - ❑ Apple
  - ❑ *Nvidia*

*has own fab

- ❖ iOT
  - ❑ *Amazon*
  - ❑ *Google*

- ❖ Foundry (mfr)
  - ❑ TSMC
  - ❑ Samsung
  - ❑ GlobalFoundries (AMD)

CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE
COMP122

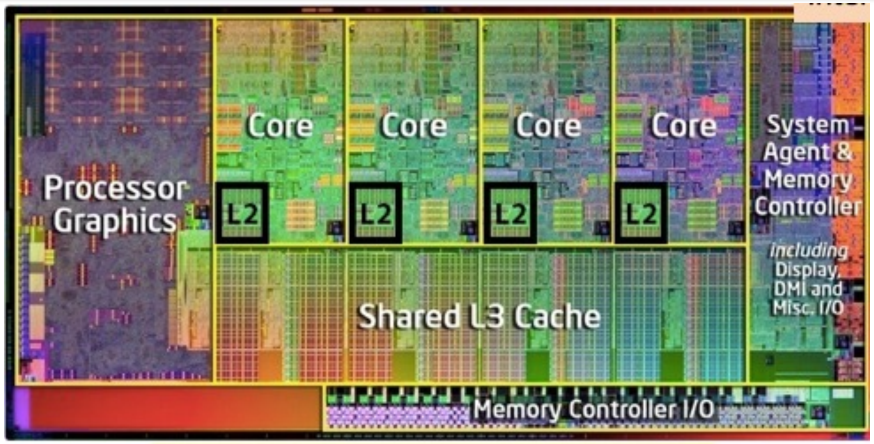
4 Levels of CPU *Architecture*

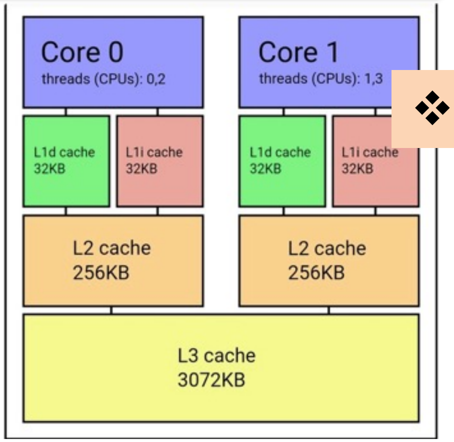
DR JEFF SOFTWARE
INDIE APP DEVELOPER
© Jeff Drobman 2016-2022

COMP122/222

❖ Macro

***System**= Multi-core SoC*





❖ Floorplan

COMP122

❖ **Org + ISA**

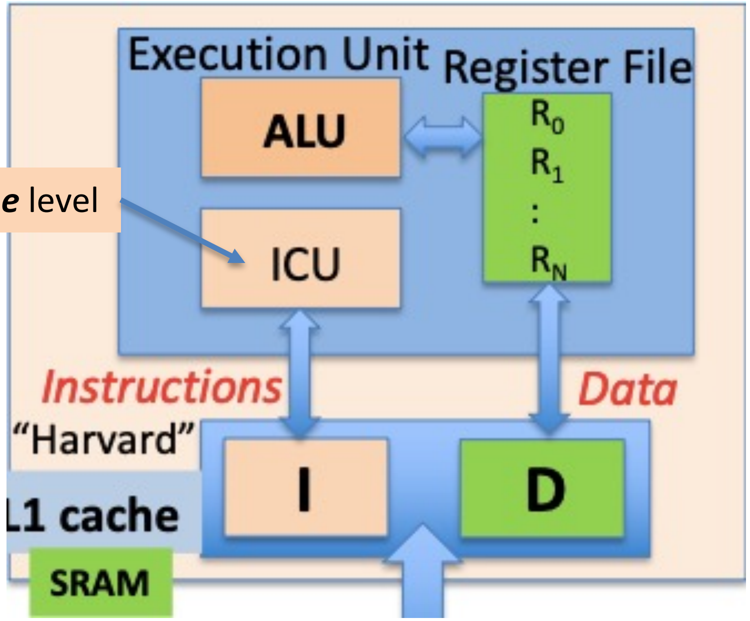
***CPU Core** internals*

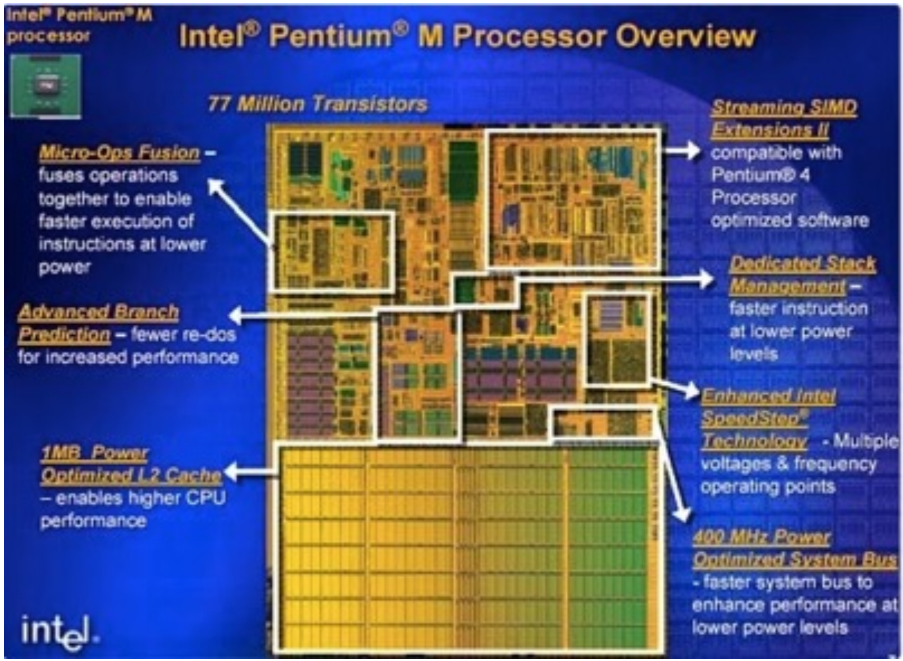
COMP222

❖ Micro

***Pipeline** level*

***Pipeline** level*





Intel® Pentium® M processor

Intel® Pentium® M Processor Overview

77 Million Transistors

Micro-Ops Fusion – fuses operations together to enable faster execution of instructions at lower power

Advanced Branch Prediction – fewer re-dos for increased performance

1MB Power Optimized L2 Cache – enables higher CPU performance

Streaming SIMD Extensions II compatible with Pentium® 4 Processor optimized software

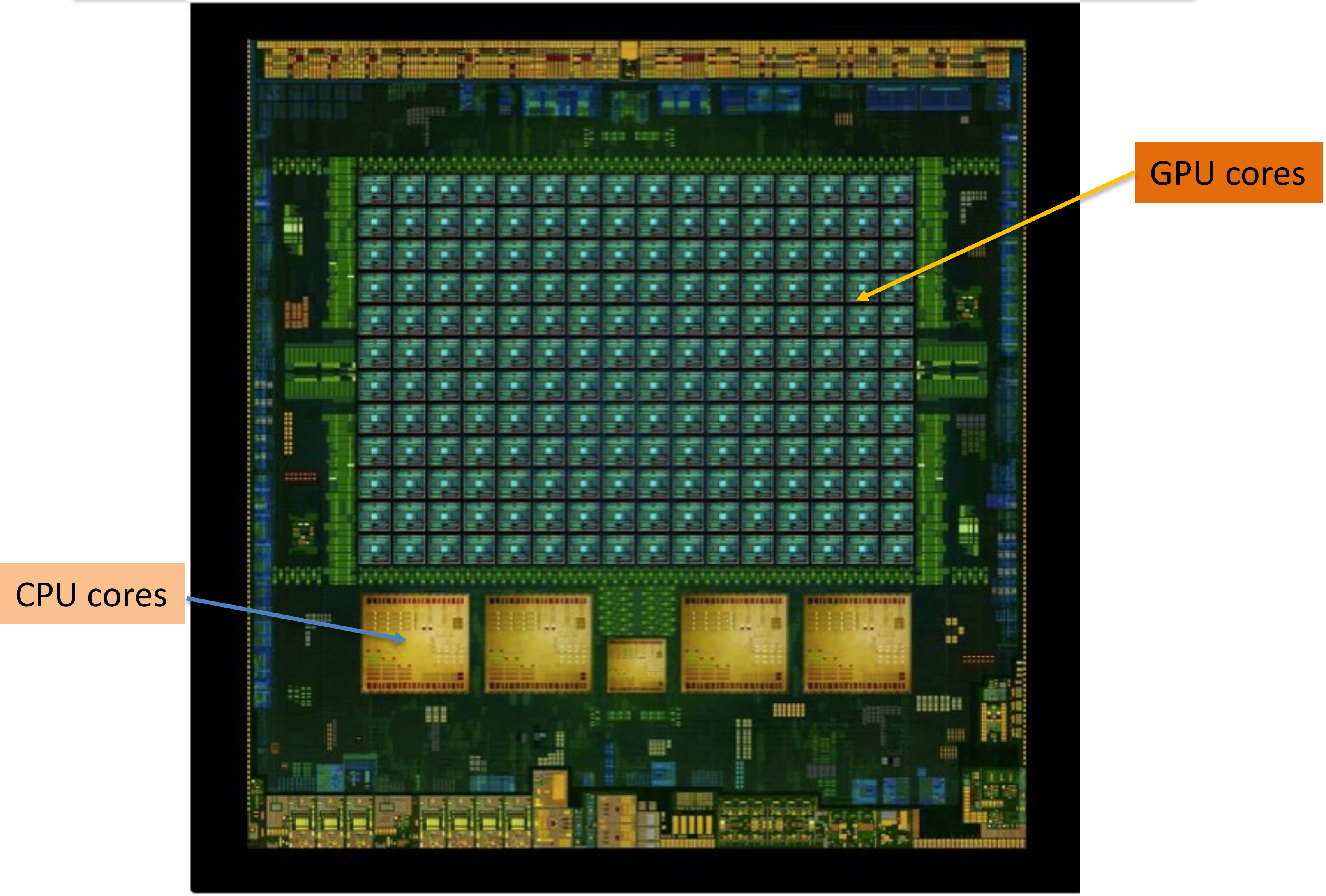
Dedicated Stack Management – faster instruction at lower power levels

Enhanced Intel SpeedStep® Technology – Multiple voltages & frequency operating points

400 MHz Power Optimized System Bus – faster system bus to enhance performance at lower power levels

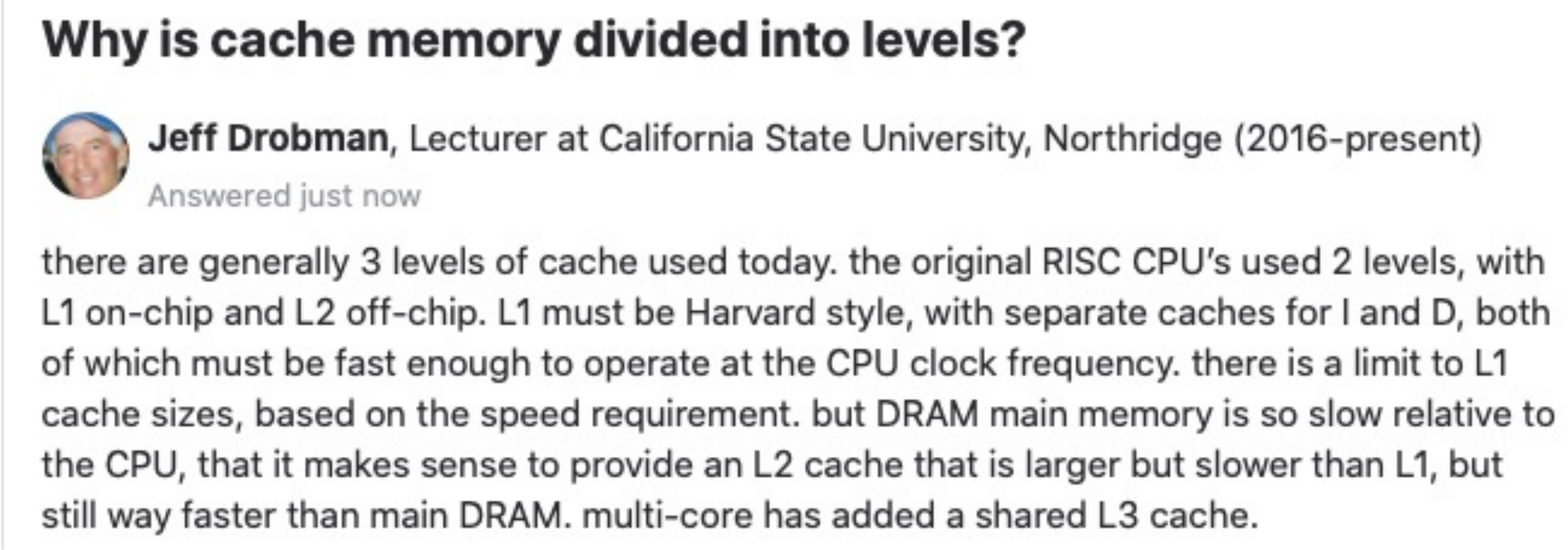
intel.

SoC = CPU + GPU



This SOC has four CPU cores (ARM Cortex) and 192 GPU cores (Kepler).

Cache Levels



Why is cache memory divided into levels?

Jeff Drobman, Lecturer at California State University, Northridge (2016-present)

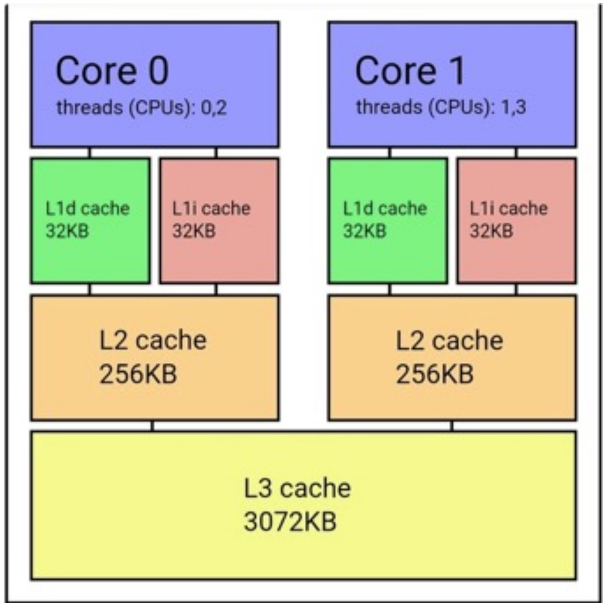
Answered just now

there are generally 3 levels of cache used today. the original RISC CPU's used 2 levels, with L1 on-chip and L2 off-chip. L1 must be Harvard style, with separate caches for I and D, both of which must be fast enough to operate at the CPU clock frequency. there is a limit to L1 cache sizes, based on the speed requirement. but DRAM main memory is so slow relative to the CPU, that it makes sense to provide an L2 cache that is larger but slower than L1, but still way faster than main DRAM. multi-core has added a shared L3 cache.

L1

L2

L3



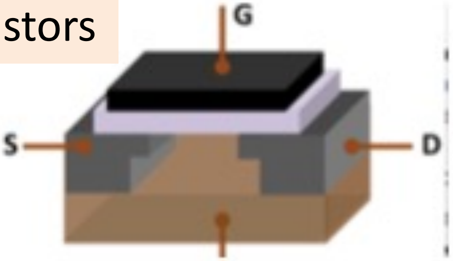
CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE
COMP122

3 Levels of Integration

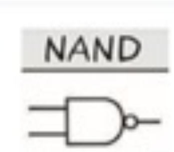
DR JEFF SOFTWARE
INDIE APP DEVELOPER
© Jeff Drobman
2016-2022

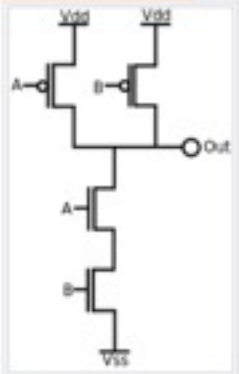
Hardware Building Blocks

0- Transistors

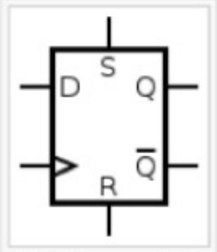


1- Gates & FF's



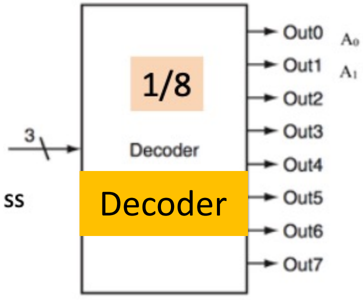


NAND gate in CMOS logic



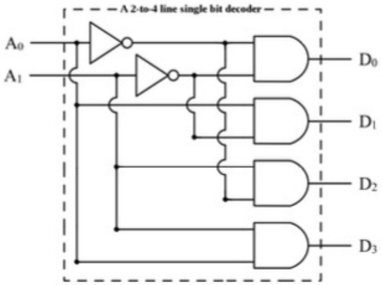
D flip-flop symbol

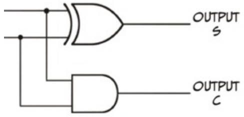
2-Single Functions

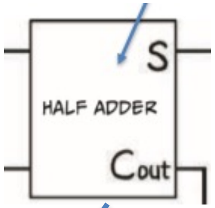


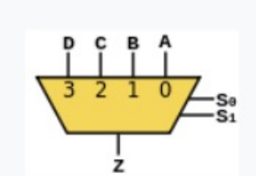
Decoder

a. A 3-bit decoder



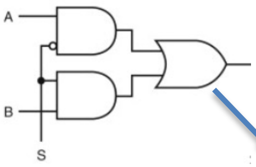


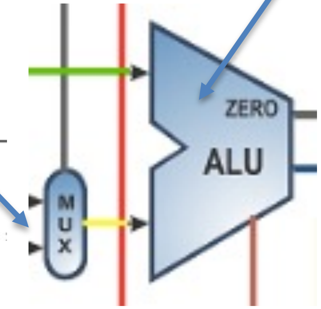




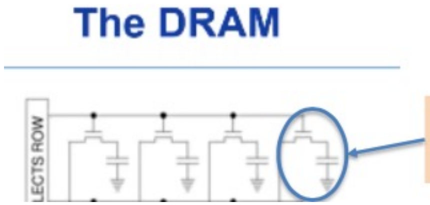
Mux

4-to-1 mux



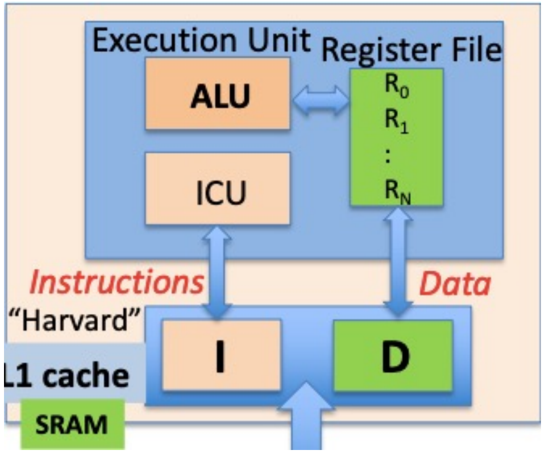


Memory Cells

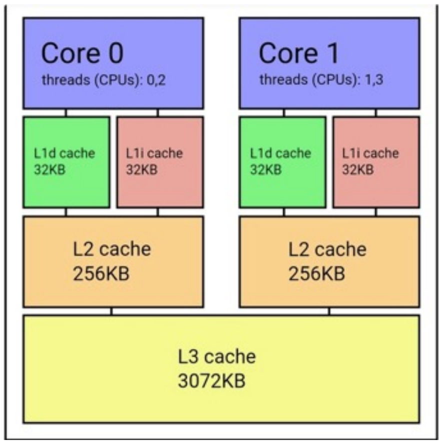


- ❖ 1 transistor
- ❖ 1 cap (parasitic)

3-CPU

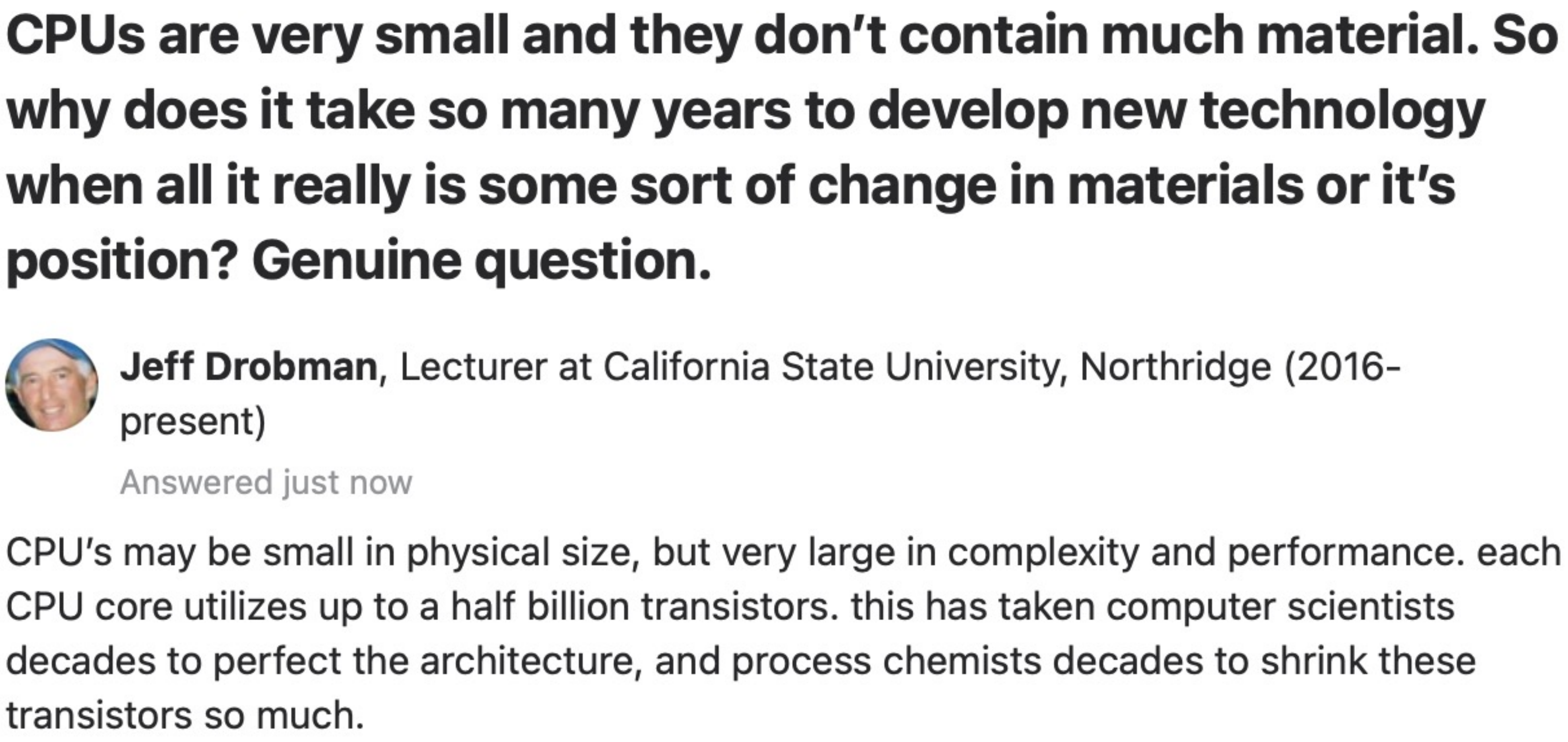


3-Multi-core CPU



CPU's Are Small?

Quora



CPUs are very small and they don't contain much material. So why does it take so many years to develop new technology when all it really is some sort of change in materials or it's position? Genuine question.

Jeff Drobman, Lecturer at California State University, Northridge (2016-present)

Answered just now

CPU's may be small in physical size, but very large in complexity and performance. each CPU core utilizes up to a half billion transistors. this has taken computer scientists decades to perfect the architecture, and process chemists decades to shrink these transistors so much.

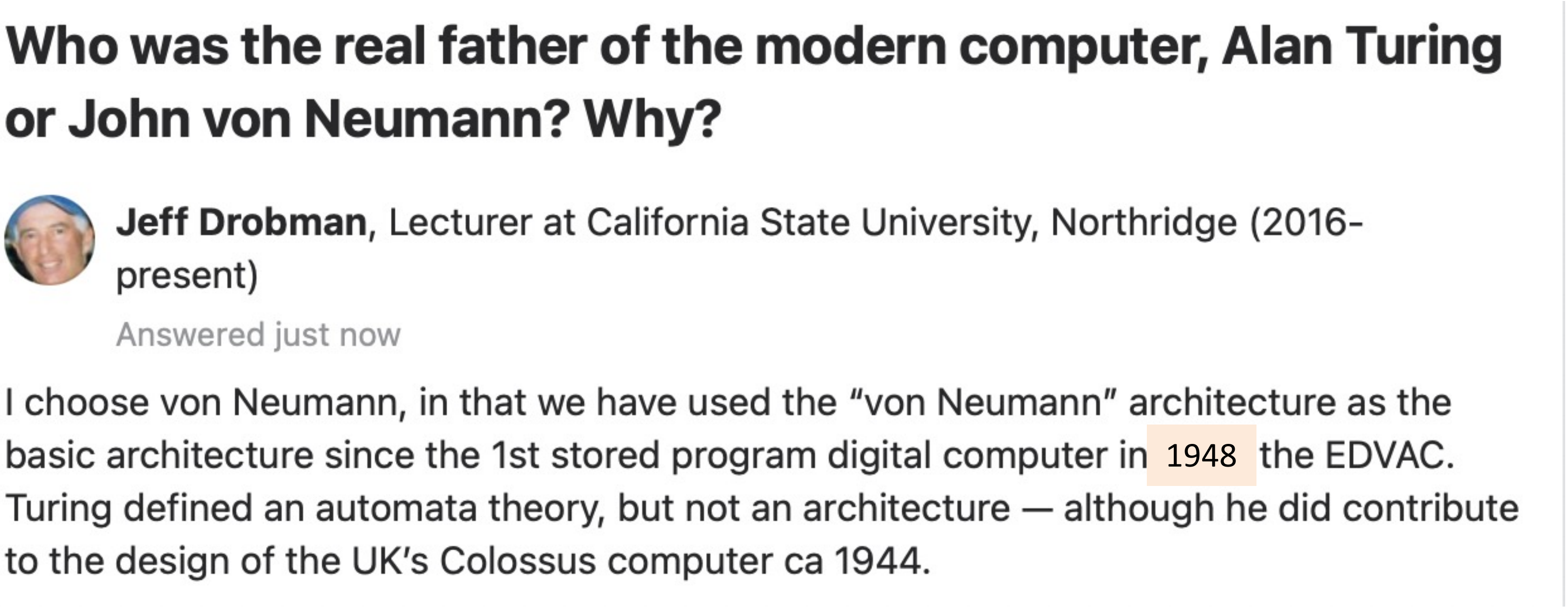
Computer Architecture

Computer History

- ➢ See separate slide set on ***History of Tech*** Vol 1

Father of Computers

Quora



Who was the real father of the modern computer, Alan Turing or John von Neumann? Why?

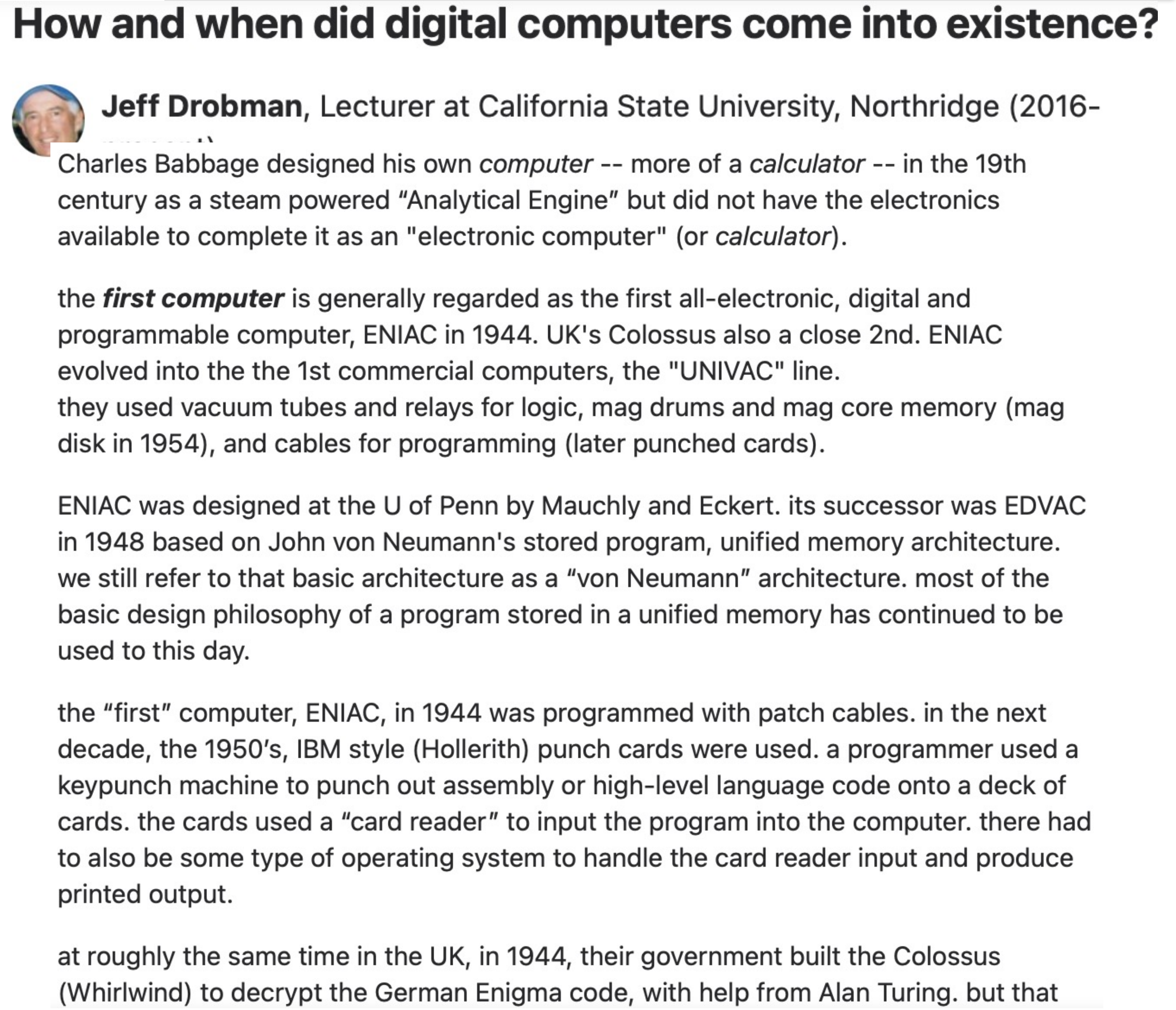
Jeff Drobman, Lecturer at California State University, Northridge (2016-present)

Answered just now

I choose von Neumann, in that we have used the "von Neumann" architecture as the basic architecture since the 1st stored program digital computer in 1948 the EDVAC. Turing defined an automata theory, but not an architecture — although he did contribute to the design of the UK's Colossus computer ca 1944.

1st Computer

Quora



How and when did digital computers come into existence?

Jeff Drobman, Lecturer at California State University, Northridge (2016-

Charles Babbage designed his own *computer* -- more of a *calculator* -- in the 19th century as a steam powered “Analytical Engine” but did not have the electronics available to complete it as an "electronic computer" (or *calculator*).

the ***first computer*** is generally regarded as the first all-electronic, digital and programmable computer, ENIAC in 1944. UK's Colossus also a close 2nd. ENIAC evolved into the the 1st commercial computers, the "UNIVAC" line.
they used vacuum tubes and relays for logic, mag drums and mag core memory (mag disk in 1954), and cables for programming (later punched cards).

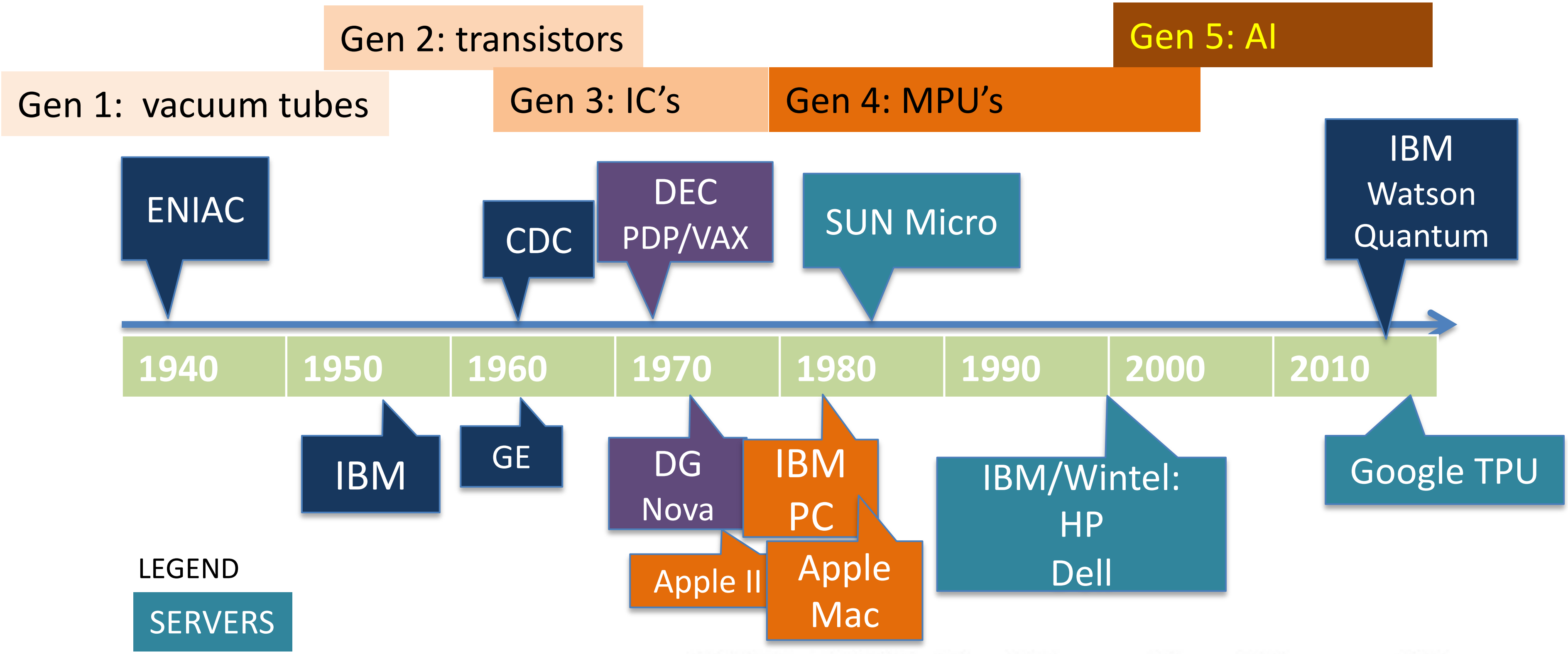
ENIAC was designed at the U of Penn by Mauchly and Eckert. its successor was EDVAC in 1948 based on John von Neumann's stored program, unified memory architecture. we still refer to that basic architecture as a “von Neumann” architecture. most of the basic design philosophy of a program stored in a unified memory has continued to be used to this day.

the “first” computer, ENIAC, in 1944 was programmed with patch cables. in the next decade, the 1950’s, IBM style (Hollerith) punch cards were used. a programmer used a keypunch machine to punch out assembly or high-level language code onto a deck of cards. the cards used a “card reader” to input the program into the computer. there had to also be some type of operating system to handle the card reader input and produce printed output.

at roughly the same time in the UK, in 1944, their government built the Colossus (Whirlwind) to decrypt the German Enigma code, with help from Alan Turing. but that

Computer Generations

TIMELINE



Gen 1: vacuum tubes

Gen 2: transistors

Gen 3: IC's

Gen 4: MPU's

Gen 5: AI

ENIAC

CDC

DEC PDP/VAX

SUN Micro

IBM Watson Quantum

1940 1950 1960 1970 1980 1990 2000 2010

IBM

GE

DG Nova

IBM PC

Apple II

Apple Mac

IBM/Wintel: HP Dell

Google TPU

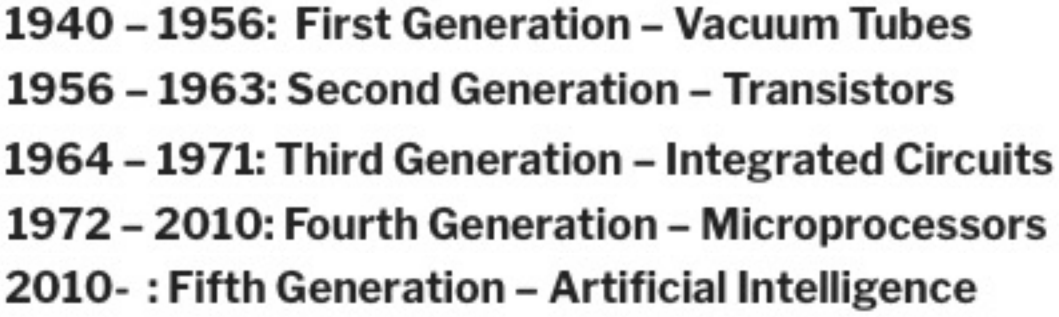
LEGEND

SERVERS

PCS

MINIS

MAINFRAMES



1940 – 1956: First Generation – Vacuum Tubes

1956 – 1963: Second Generation – Transistors

1964 – 1971: Third Generation – Integrated Circuits

1972 – 2010: Fourth Generation – Microprocessors

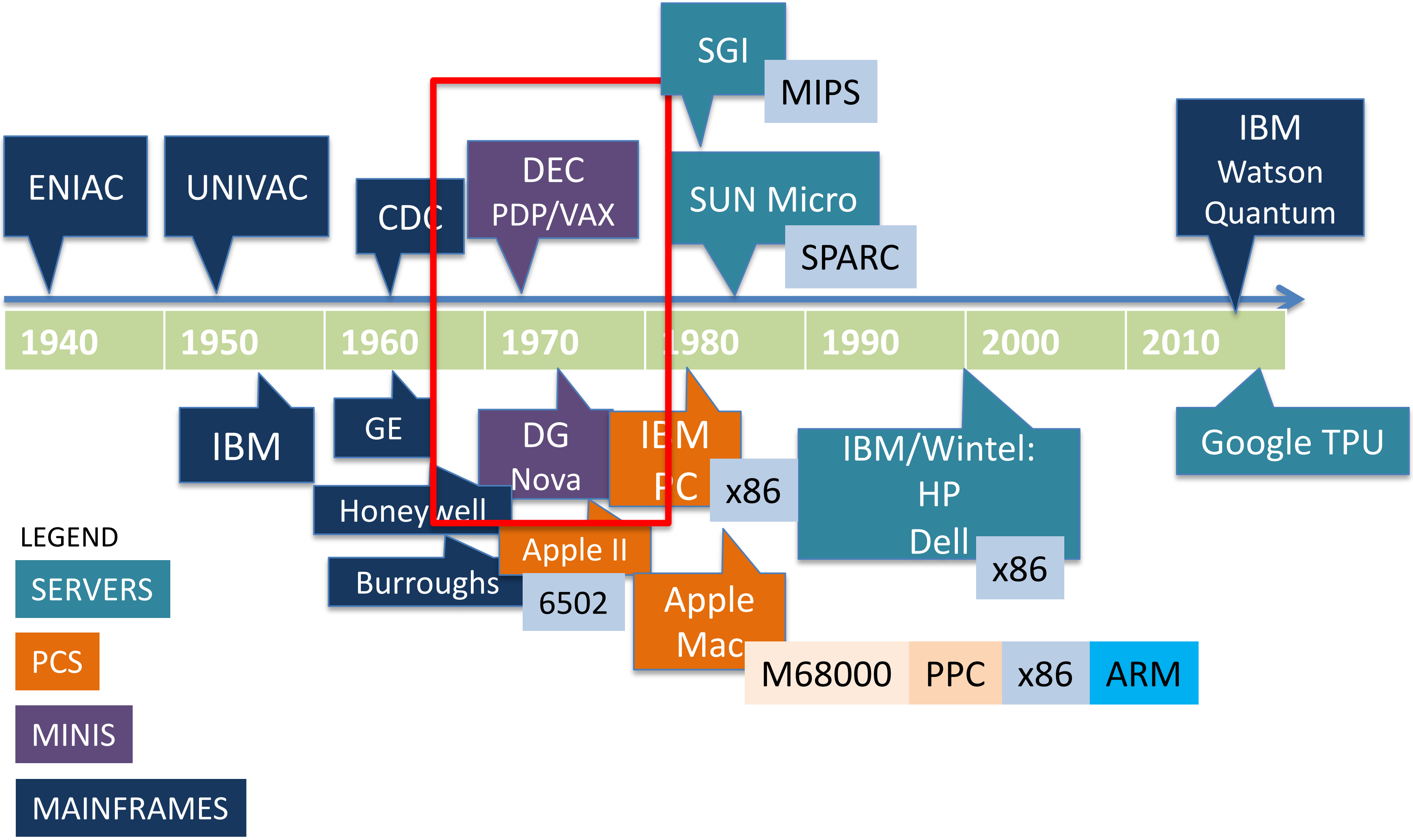
2010- : Fifth Generation – Artificial Intelligence

Computers

Minicomputers of the 1970's
Used Am2900 *bit-slices*

TIMELINE

Computers of the 1990's
Used RISC CPU's



SGI

MIPS

ENIAC

UNIVAC

CDC

DEC
PDP/VAX

SUN Micro

SPARC

IBM
Watson
Quantum

1940 | 1950 | 1960 | 1970 | 1980 | 1990 | 2000 | 2010

IBM

GE

Honeywell

Burroughs

DG
Nova

IBM
PC

x86

Apple II

6502

Apple
Mac

M68000 | PPC | x86 | ARM

IBM/Wintel:
HP
Dell

x86

Google TPU

LEGEND

SERVERS

PCS

MINIS

MAINFRAMES

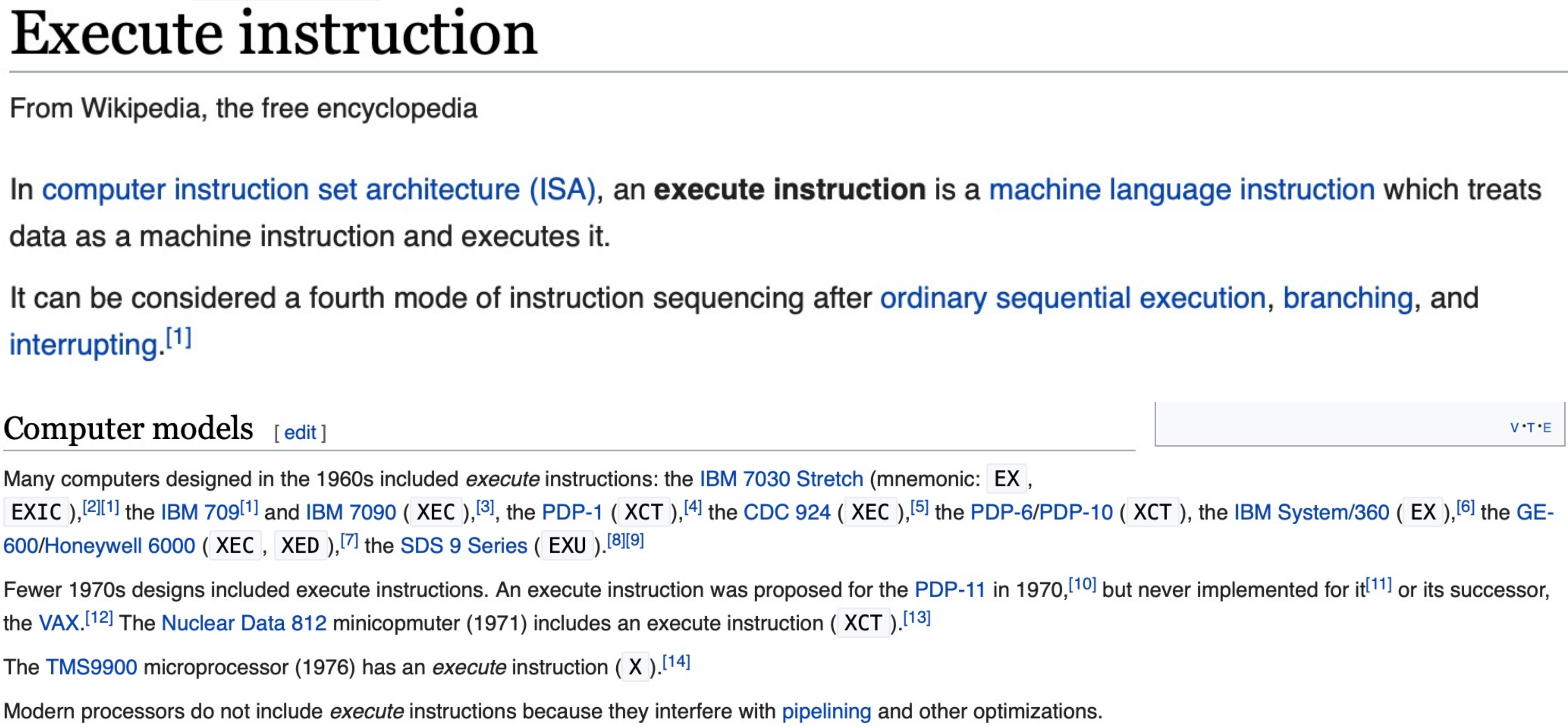
CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE

COMP122

Old Computer ISA's

DR JEFF SOFTWARE
INDIE APP DEVELOPER
© Jeff Drobman
2016-2022

WIKIPEDIA
The Free Encyclopedia



Execute instruction

From Wikipedia, the free encyclopedia

In computer instruction set architecture (ISA), an **execute instruction** is a machine language instruction which treats data as a machine instruction and executes it.

It can be considered a fourth mode of instruction sequencing after ordinary sequential execution, branching, and interrupting.[1]

Computer models [edit]

Many computers designed in the 1960s included *execute* instructions: the IBM 7030 Stretch (mnemonic: `EX`, `EXIC`),[2][1] the IBM 709[1] and IBM 7090 (`XEC`),[3], the PDP-1 (`XCT`),[4] the CDC 924 (`XEC`),[5] the PDP-6/PDP-10 (`XCT`), the IBM System/360 (`EX`),[6] the GE-600/Honeywell 6000 (`XEC`, `XED`),[7] the SDS 9 Series (`EXU`).[8][9]

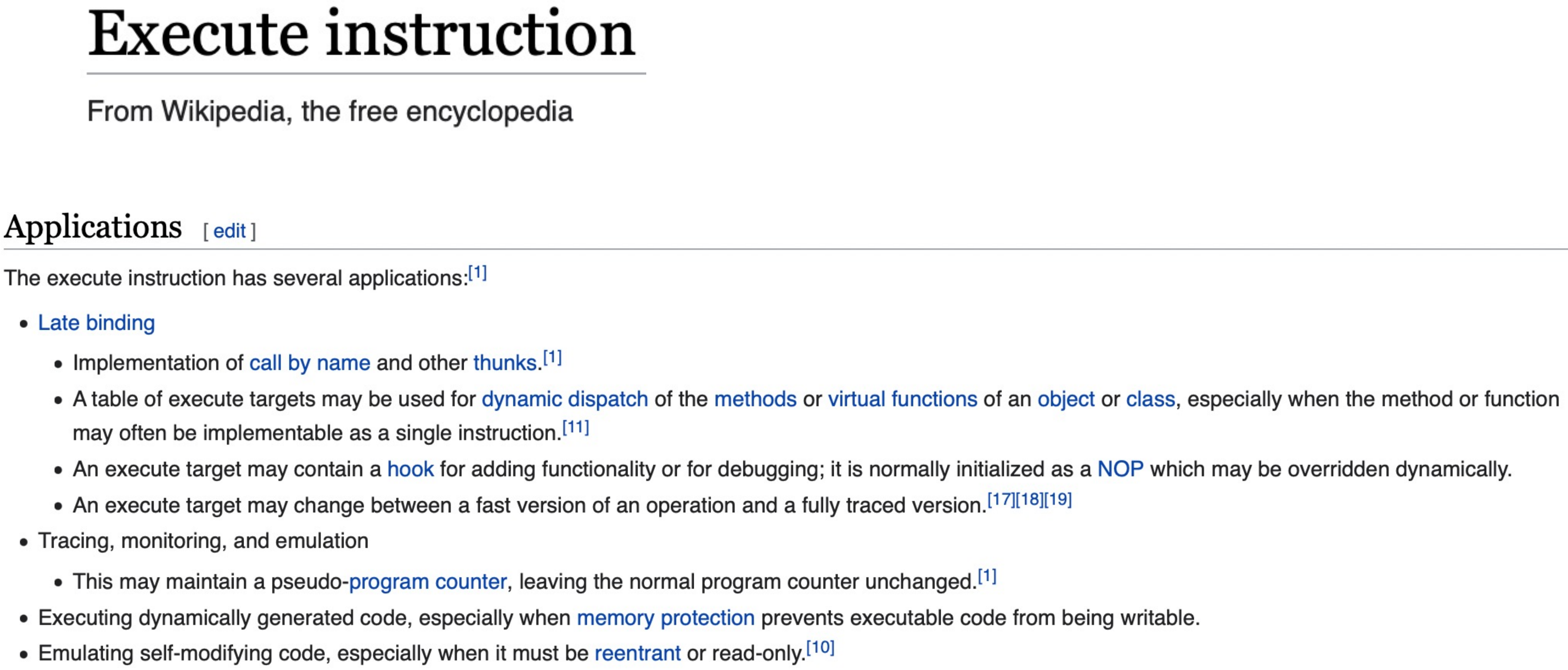
Fewer 1970s designs included execute instructions. An execute instruction was proposed for the PDP-11 in 1970,[10] but never implemented for it[11] or its successor, the VAX.[12] The Nuclear Data 812 minicopmuter (1971) includes an execute instruction (`XCT`).[13]

The TMS9900 microprocessor (1976) has an *execute* instruction (`X`).[14]

Modern processors do not include *execute* instructions because they interfere with pipelining and other optimizations.

WIKIPEDIA
The Free Encyclopedia

Old Computer ISA’s



Execute instruction

From Wikipedia, the free encyclopedia

Applications [edit]

The execute instruction has several applications:[1]

- Late binding
  - Implementation of call by name and other thunks.[1]
  - A table of execute targets may be used for dynamic dispatch of the methods or virtual functions of an object or class, especially when the method or function may often be implementable as a single instruction.[11]
  - An execute target may contain a hook for adding functionality or for debugging; it is normally initialized as a NOP which may be overridden dynamically.
  - An execute target may change between a fast version of an operation and a fully traced version.[17][18][19]
- Tracing, monitoring, and emulation
  - This may maintain a pseudo-program counter, leaving the normal program counter unchanged.[1]
- Executing dynamically generated code, especially when memory protection prevents executable code from being writable.
- Emulating self-modifying code, especially when it must be reentrant or read-only.[10]

CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE

COMP122

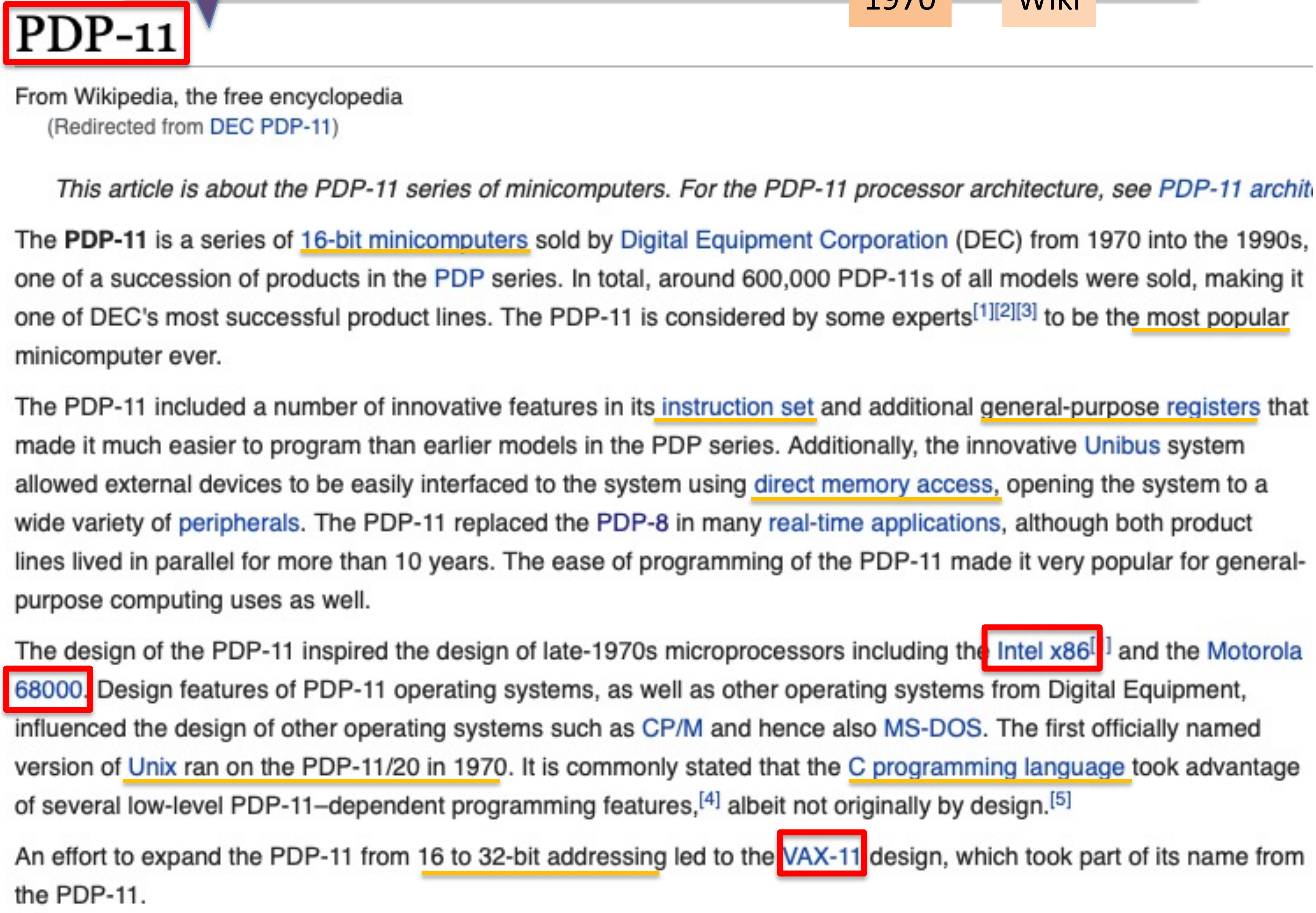
DEC PDP/VAX

DEC PDP-11

DR JEFF SOFTWARE
INDIE APP DEVELOPER
© Jeff Drobman
2016-2022

1970

Wiki



PDP-11

From Wikipedia, the free encyclopedia
(Redirected from DEC PDP-11)

This article is about the PDP-11 series of minicomputers. For the PDP-11 processor architecture, see PDP-11 archit

The **PDP-11** is a series of 16-bit minicomputers sold by Digital Equipment Corporation (DEC) from 1970 into the 1990s, one of a succession of products in the PDP series. In total, around 600,000 PDP-11s of all models were sold, making it one of DEC's most successful product lines. The PDP-11 is considered by some experts[1][2][3] to be the most popular minicomputer ever.

The PDP-11 included a number of innovative features in its instruction set and additional general-purpose registers that made it much easier to program than earlier models in the PDP series. Additionally, the innovative Unibus system allowed external devices to be easily interfaced to the system using direct memory access, opening the system to a wide variety of peripherals. The PDP-11 replaced the PDP-8 in many real-time applications, although both product lines lived in parallel for more than 10 years. The ease of programming of the PDP-11 made it very popular for general-purpose computing uses as well.

The design of the PDP-11 inspired the design of late-1970s microprocessors including the Intel x86[ ] and the Motorola 68000. Design features of PDP-11 operating systems, as well as other operating systems from Digital Equipment, influenced the design of other operating systems such as CP/M and hence also MS-DOS. The first officially named version of Unix ran on the PDP-11/20 in 1970. It is commonly stated that the C programming language took advantage of several low-level PDP-11–dependent programming features,[4] albeit not originally by design.[5]

An effort to expand the PDP-11 from 16 to 32-bit addressing led to the VAX-11 design, which took part of its name from the PDP-11.

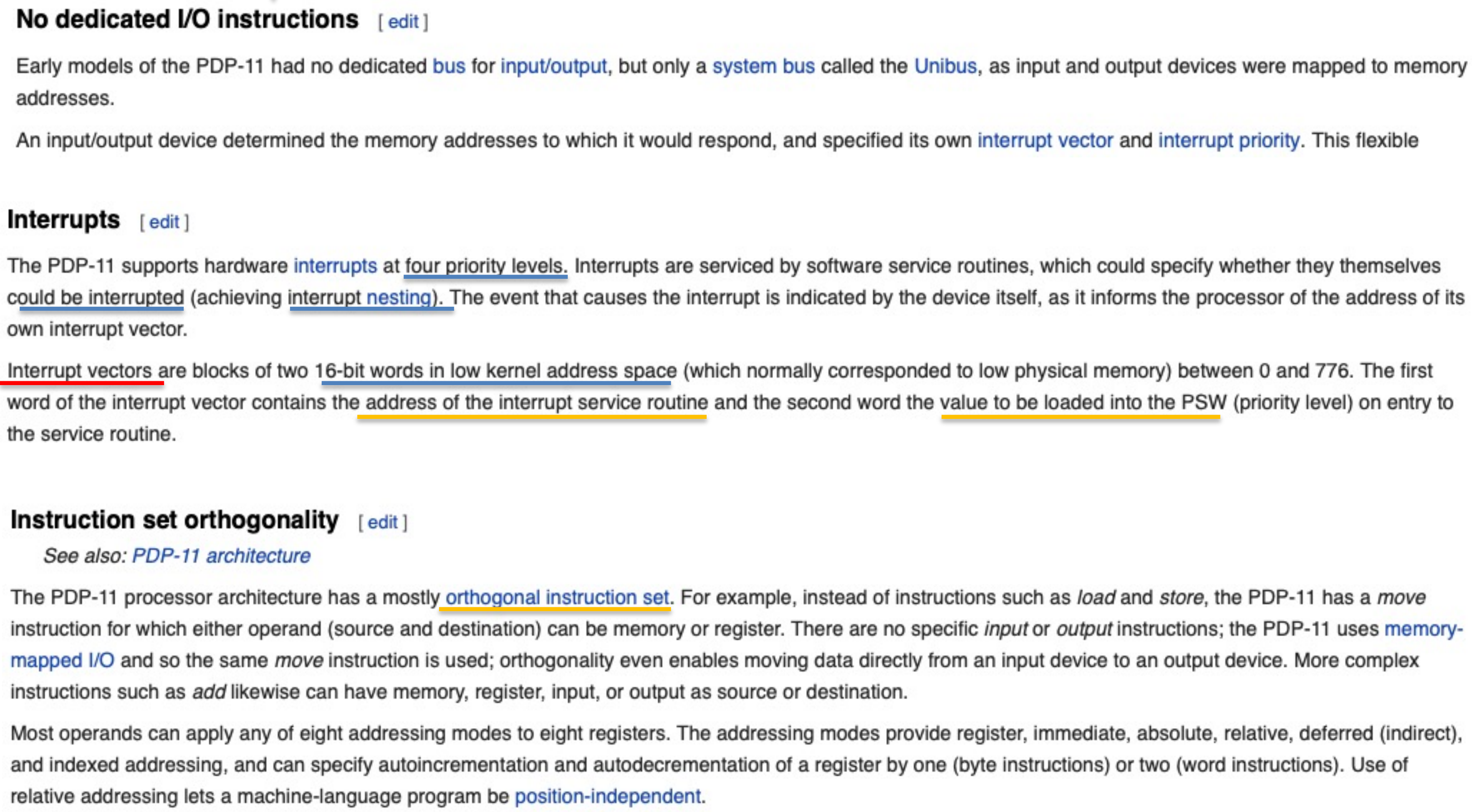
CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE
COMP122

DEC PDP/VAX

DEC PDP-11

1970 Wiki

DR JEFF SOFTWARE
INDIE APP DEVELOPER



No dedicated I/O instructions [edit]

Early models of the PDP-11 had no dedicated bus for input/output, but only a system bus called the Unibus, as input and output devices were mapped to memory addresses.

An input/output device determined the memory addresses to which it would respond, and specified its own interrupt vector and interrupt priority. This flexible

Interrupts [edit]

The PDP-11 supports hardware interrupts at four priority levels. Interrupts are serviced by software service routines, which could specify whether they themselves could be interrupted (achieving interrupt nesting). The event that causes the interrupt is indicated by the device itself, as it informs the processor of the address of its own interrupt vector.

Interrupt vectors are blocks of two 16-bit words in low kernel address space (which normally corresponded to low physical memory) between 0 and 776. The first word of the interrupt vector contains the address of the interrupt service routine and the second word the value to be loaded into the PSW (priority level) on entry to the service routine.

Instruction set orthogonality [edit]

See also: PDP-11 architecture

The PDP-11 processor architecture has a mostly orthogonal instruction set. For example, instead of instructions such as *load* and *store*, the PDP-11 has a *move* instruction for which either operand (source and destination) can be memory or register. There are no specific *input* or *output* instructions; the PDP-11 uses memory-mapped I/O and so the same *move* instruction is used; orthogonality even enables moving data directly from an input device to an output device. More complex instructions such as *add* likewise can have memory, register, input, or output as source or destination.

Most operands can apply any of eight addressing modes to eight registers. The addressing modes provide register, immediate, absolute, relative, deferred (indirect), and indexed addressing, and can specify autoincrementation and autodecrementation of a register by one (byte instructions) or two (word instructions). Use of relative addressing lets a machine-language program be position-independent.

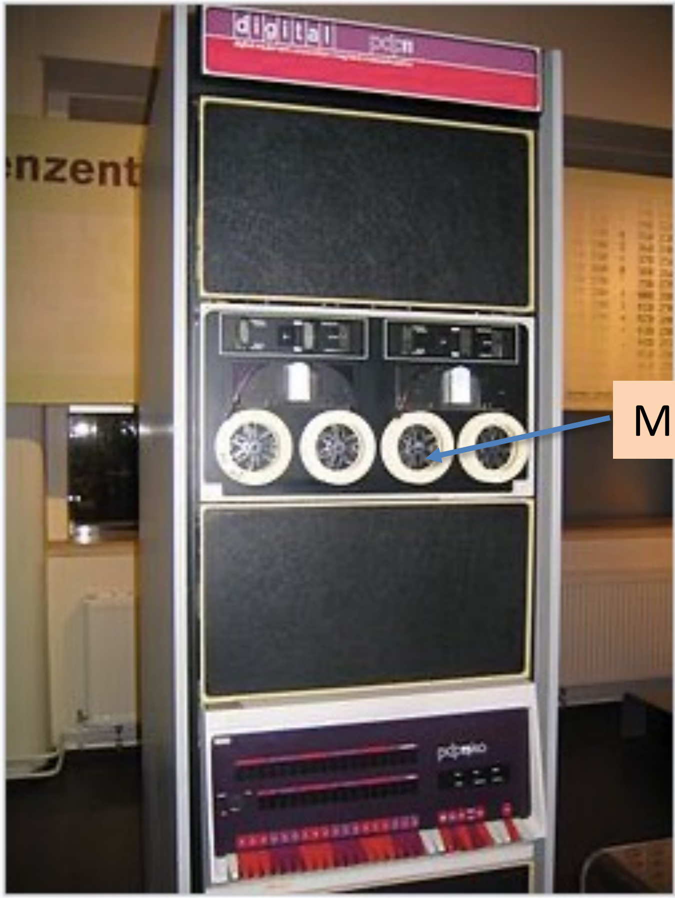
DEC PDP/VAX

DEC PDP-11

1st **LSI**-chip Computer

1970

Wiki



Mag tape

PDP-11/40. The processor is at the bottom. A TU56 dual DECtape drive is installed above it.

CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE
COMP122

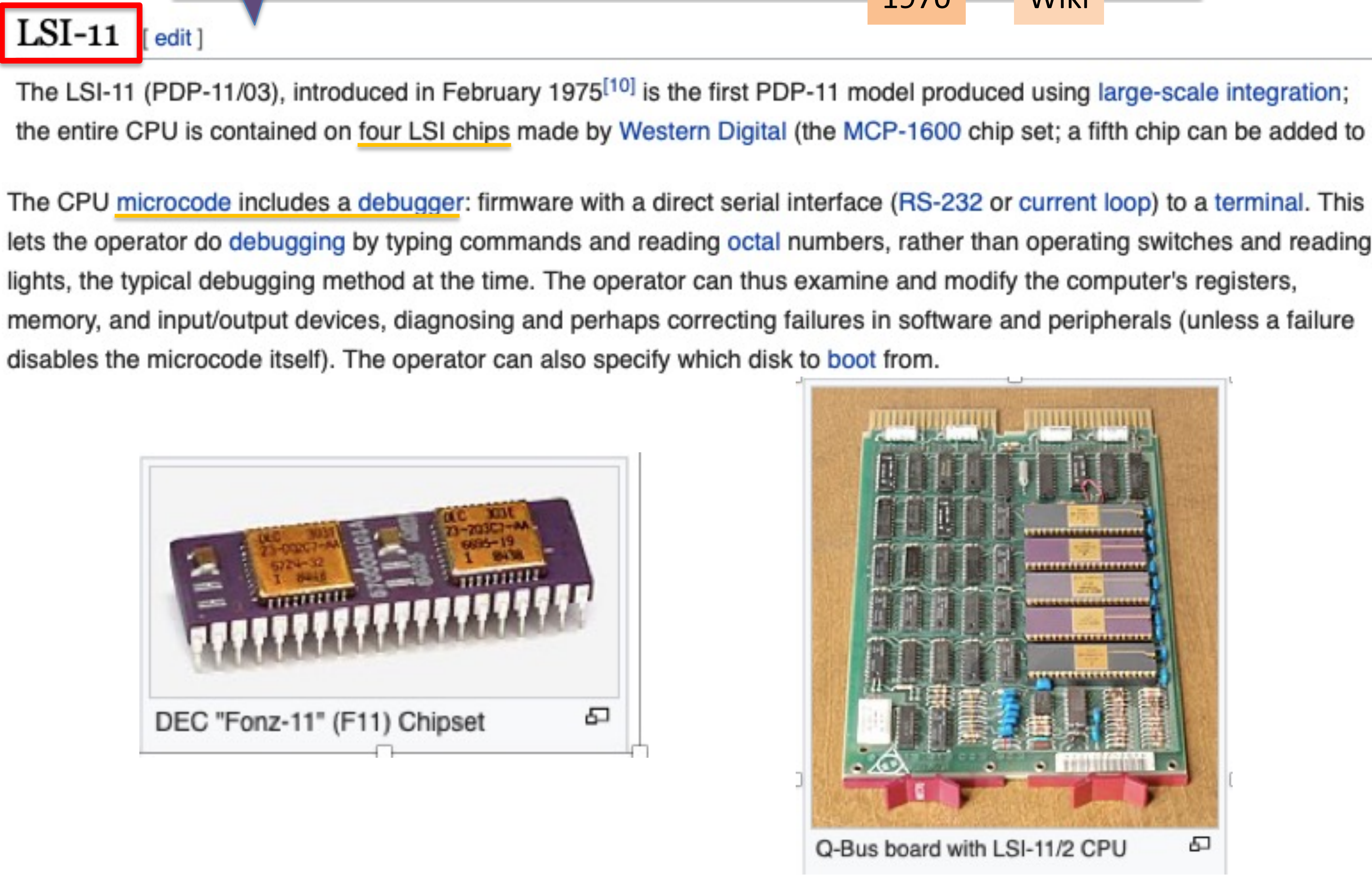
DEC PDP/VAX

DEC PDP-11

DR JEFF SOFTWARE
INDIE APP DEVELOPER
© Jeff Drobman
2016-2022

1970

Wiki



LSI-11

[edit]

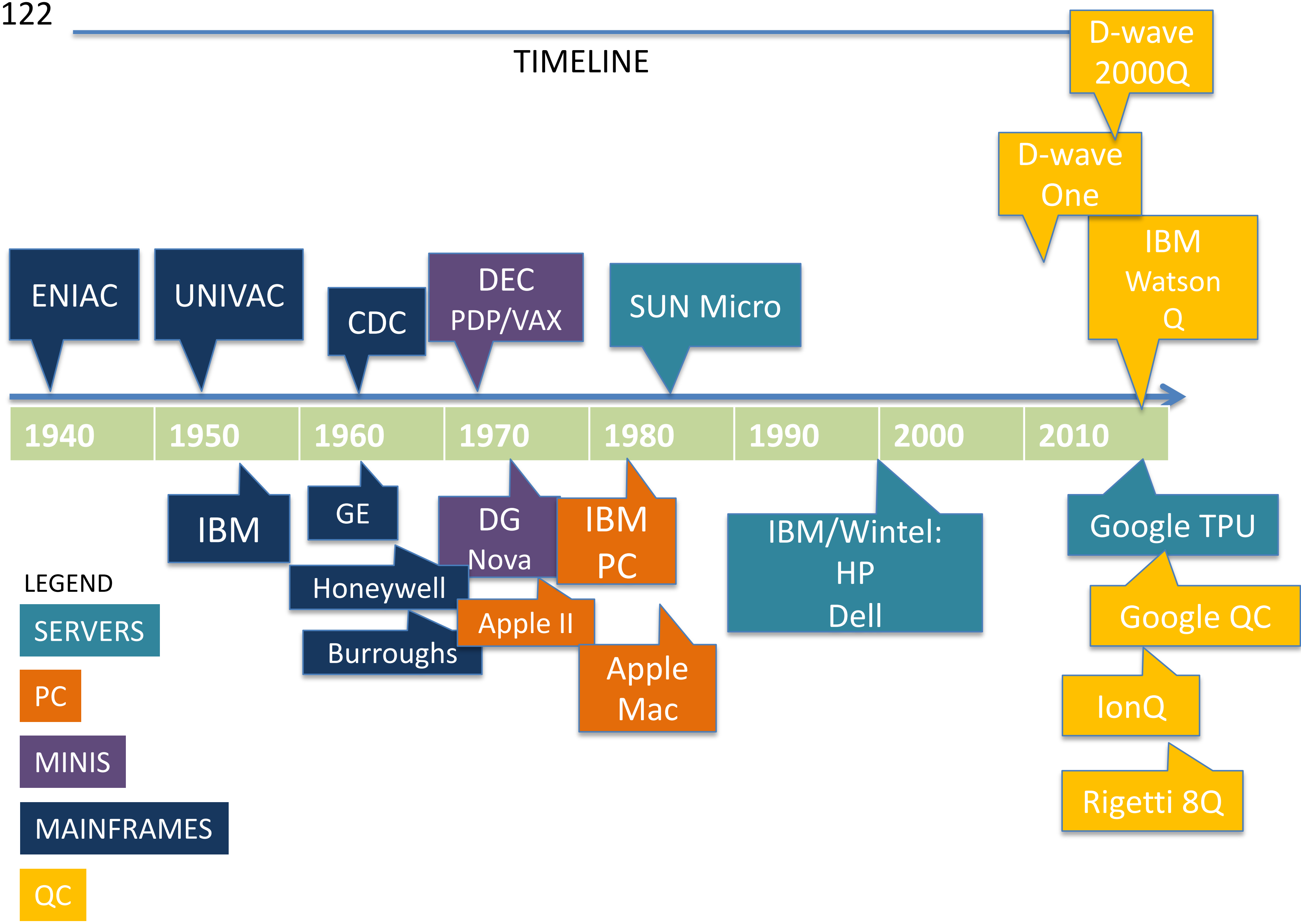
The LSI-11 (PDP-11/03), introduced in February 1975[10] is the first PDP-11 model produced using large-scale integration; the entire CPU is contained on four LSI chips made by Western Digital (the MCP-1600 chip set; a fifth chip can be added to

The CPU microcode includes a debugger: firmware with a direct serial interface (RS-232 or current loop) to a terminal. This lets the operator do debugging by typing commands and reading octal numbers, rather than operating switches and reading lights, the typical debugging method at the time. The operator can thus examine and modify the computer's registers, memory, and input/output devices, diagnosing and perhaps correcting failures in software and peripherals (unless a failure disables the microcode itself). The operator can also specify which disk to boot from.

DEC "Fonz-11" (F11) Chipset

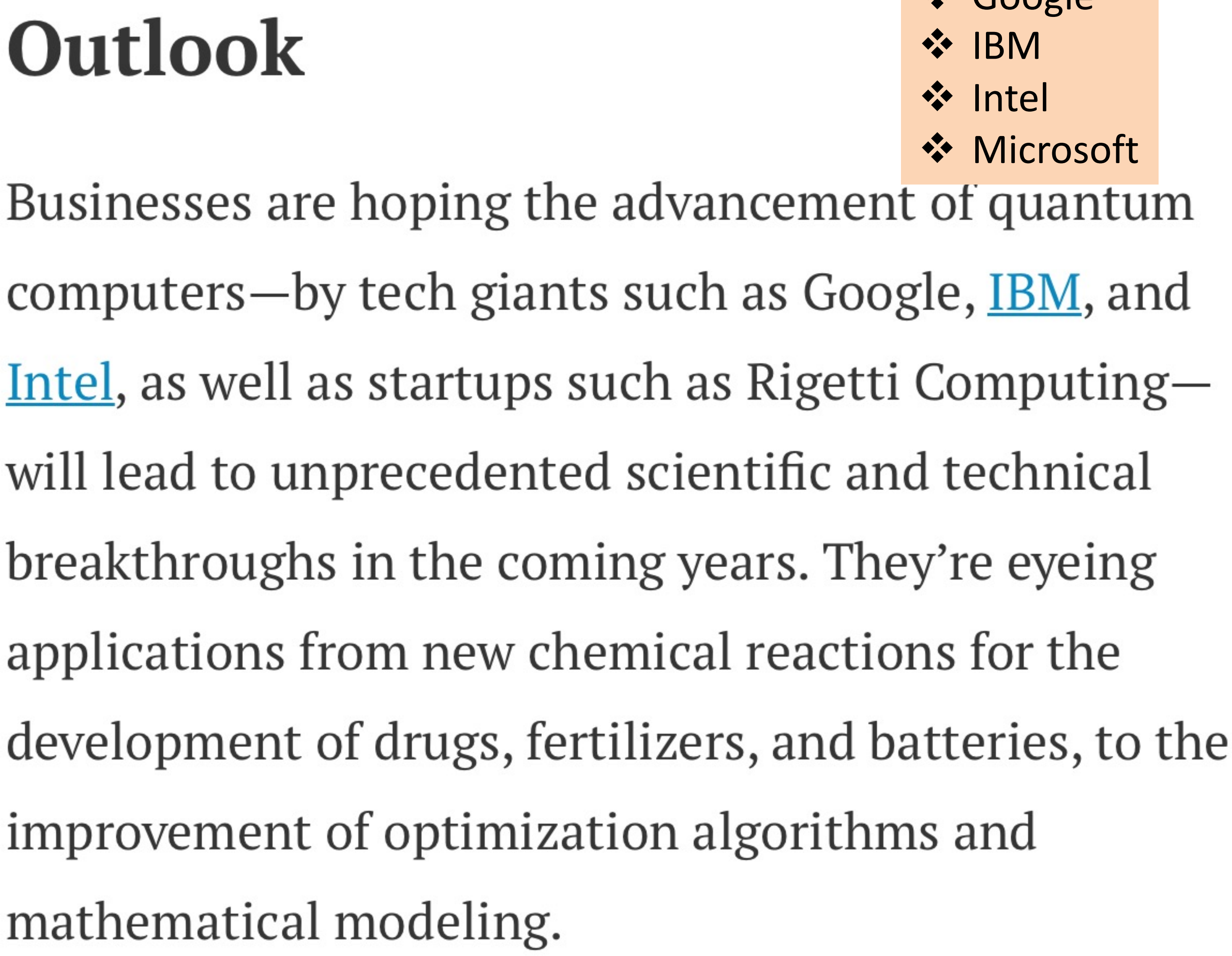
Q-Bus board with LSI-11/2 CPU

Computers & QC's



Quantum Computers (QC)

- ❖ Google
- ❖ IBM
- ❖ Intel
- ❖ Microsoft



Outlook

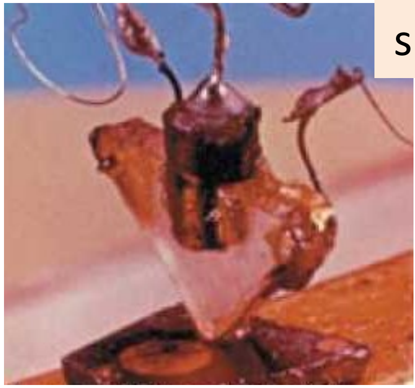
Businesses are hoping the advancement of quantum computers—by tech giants such as Google, IBM, and Intel, as well as startups such as Rigetti Computing—will lead to unprecedented scientific and technical breakthroughs in the coming years. They're eyeing applications from new chemical reactions for the development of drugs, fertilizers, and batteries, to the improvement of optimization algorithms and mathematical modeling.

Computer Architecture

IC
History

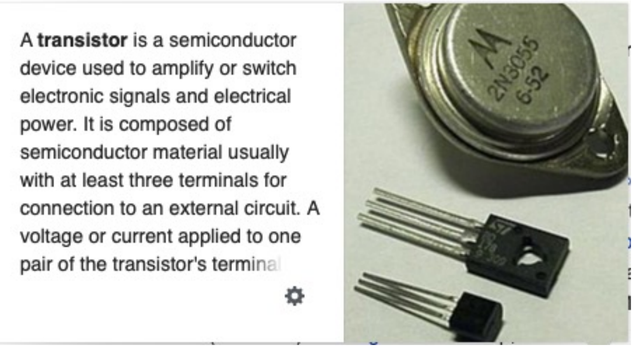
- See separate slide set on ***Transistors***

The Transistor



size = ~1 inch

1947 ushered in the era of ***Microelectronics***

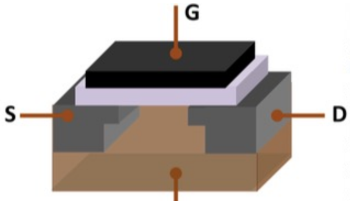


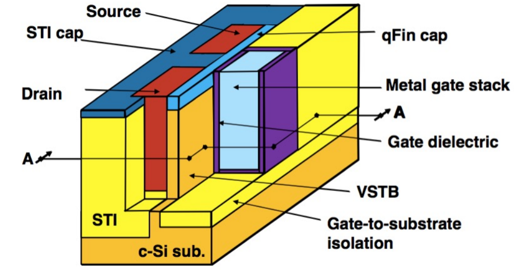
A **transistor** is a semiconductor device used to amplify or switch electronic signals and electrical power. It is composed of semiconductor material usually with at least three terminals for connection to an external circuit. A voltage or current applied to one pair of the transistor's termina

- 1947- Bipolar point/junction
- 1959- Planar bipolar [10]*
- 1964- MOS (P-channel) [100]
- 1972- MOS (N-channel) [1,000]
- 1978- CMOS [4,000]
- 1990- sub-micron [10,000]
- 2000- 100 nm [100,000]
- 2011- FinFET [1,000,000]
- 2019- 7nm [10,000,000,000]

*no. of transistors

Transistors have been shrunk every 2 years according to ***Moore's Law***

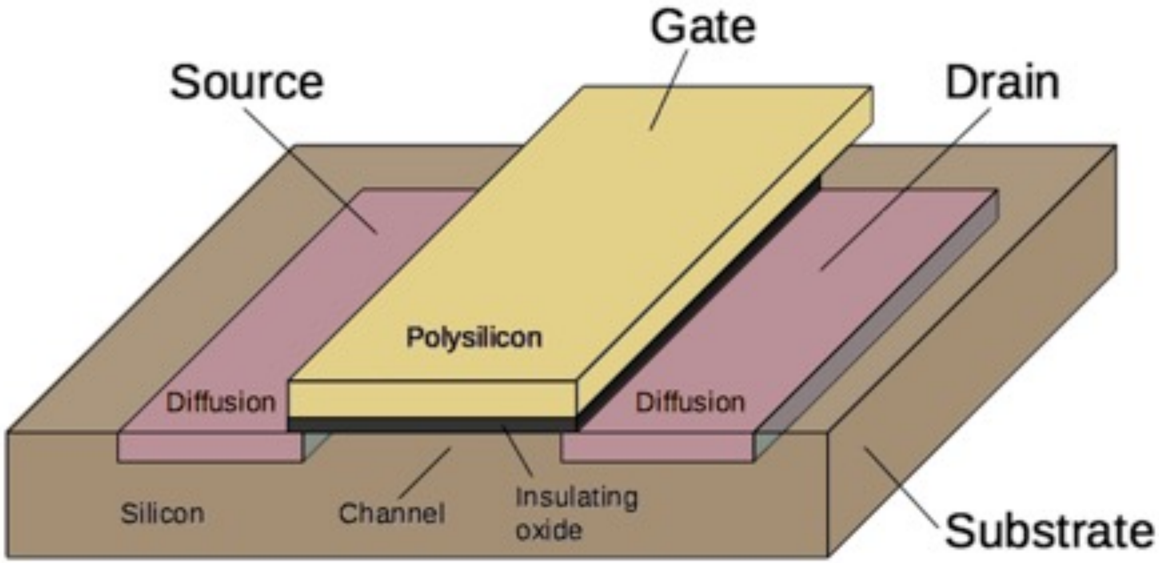




- size = 10 nm = $4\text{x}10^{-7}$ inches
- yields → ~1M devices per cm^2

MOS Transistor

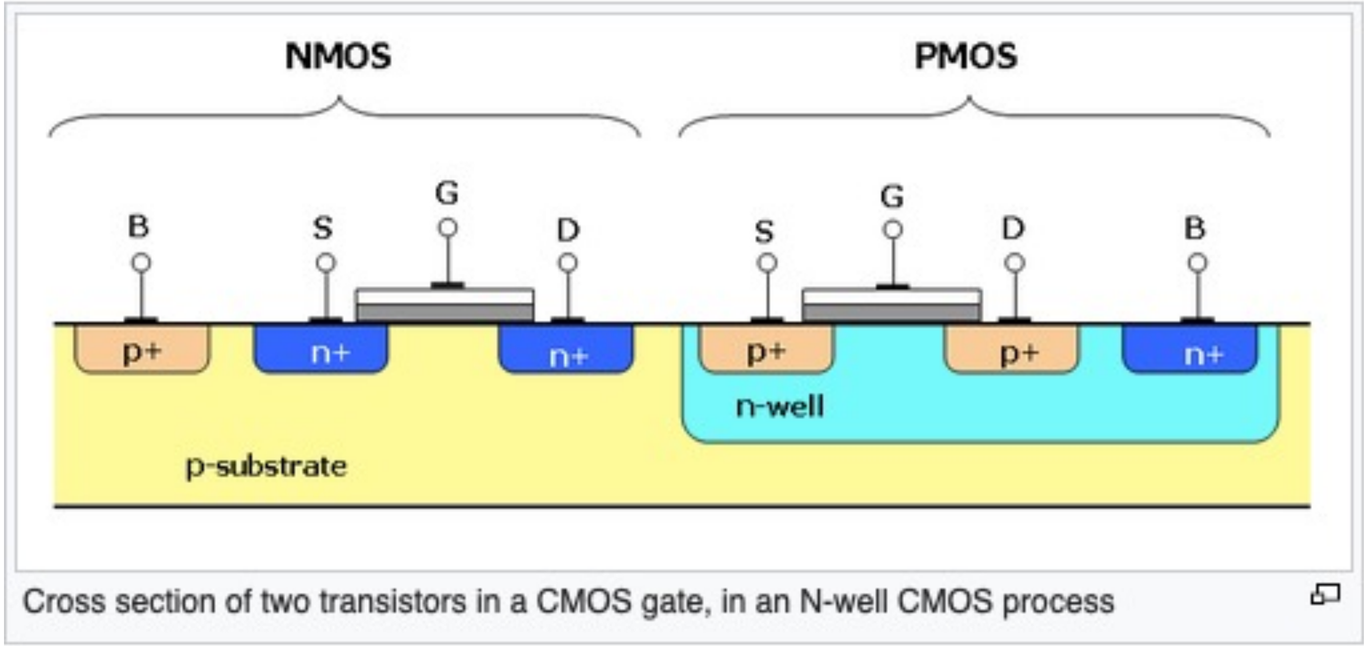
WikiSemi



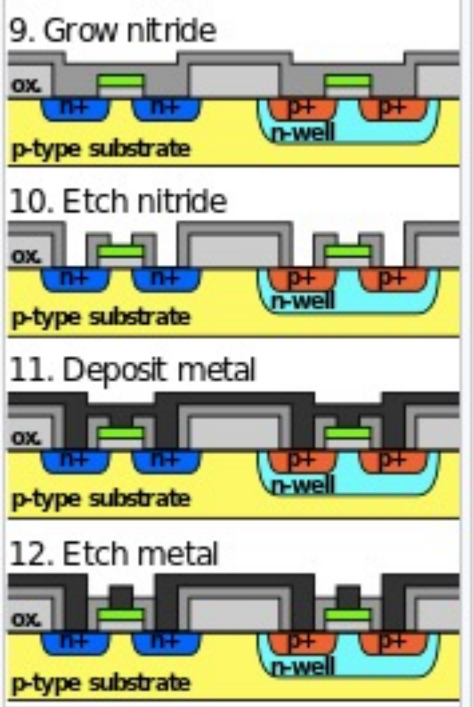
Structure of a MOSFET in the integrated circuit.

MOS Transistors

MOSFET



Cross section of two transistors in a CMOS gate, in an N-well CMOS process



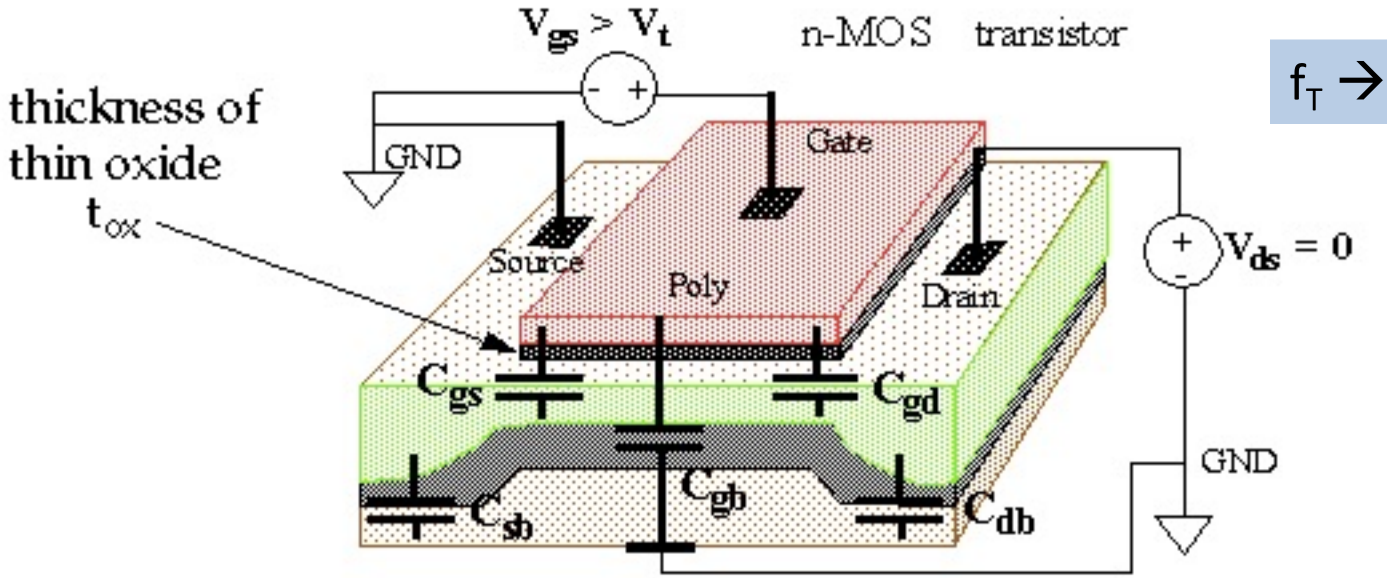
Last 4 steps

MOS Transistors

The channel will have a length (distance from one electrode to the other) and a width (imagine this diagram coming out of the screen). The electrodes have geometries. The gate doesn't always span the full width of the channel and is its own critical dimension.

The sizes of each of these things are critical dimensions. They all have an effect on the performance of the device because they will contribute parasitic capacitance and resistance.

Parasitic capacitances



f_T → lengths

Parasitics depicted by University of New Mexico

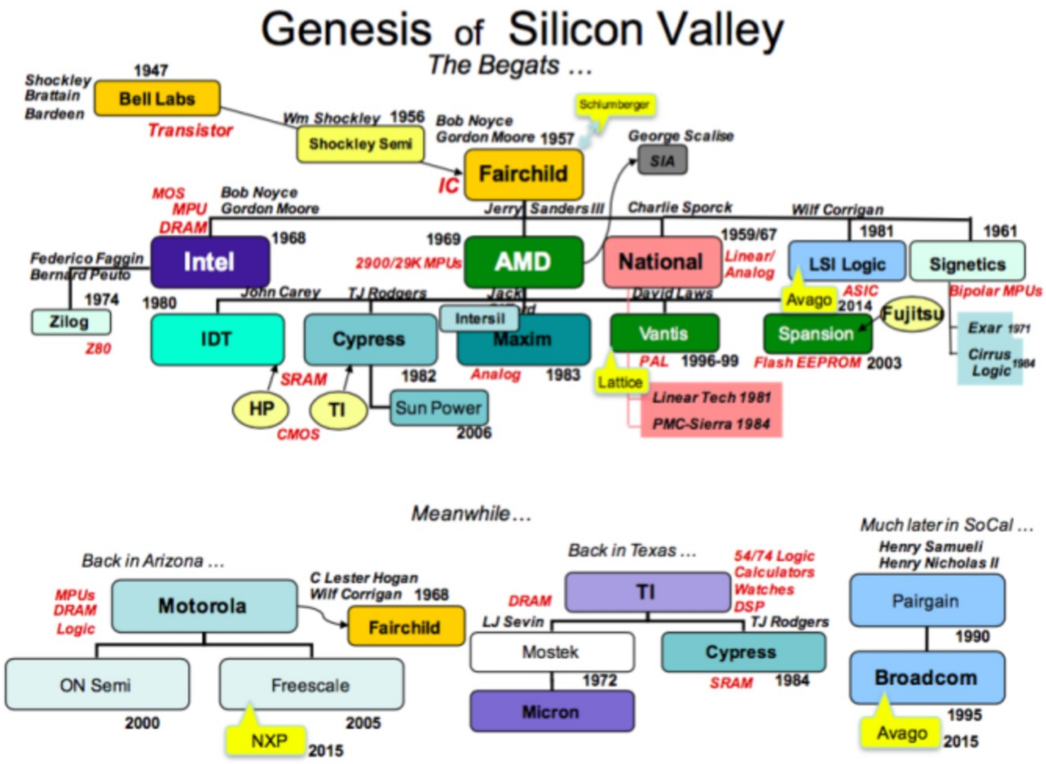
Si Valley History

Genesis: A Silicon Valley Tale

TECH HISTORY ARTICLE BY DR JEFF DROBMAN

Highlights

- ❖ Fairchild founding
- ❖ Intel founding
- ❖ AMD history
- ❖ AMD – Intel rivalry
- ❖ Search for CMOS
- ❖ RISC CPU Architecture
- ❖ Legendary Parties & Conferences
- ❖ Anecdotes
- ❖ Valley Significant Others
- ❖ Genesis org-chart
- ❖ Process Technology Evolution
- ❖ Anniversaries of Technologies



Shockley Labs

Beginning of ***Silicon Valley***

SHOCKLEY SEMICONDUCTOR LABORATORY
BECKMAN INSTRUMENTS INC.
MOUNTAIN VIEW, CALIFORNIA
Shockley Semiconductor Laboratory

Mountain View, CA 1956

+

Stanford U

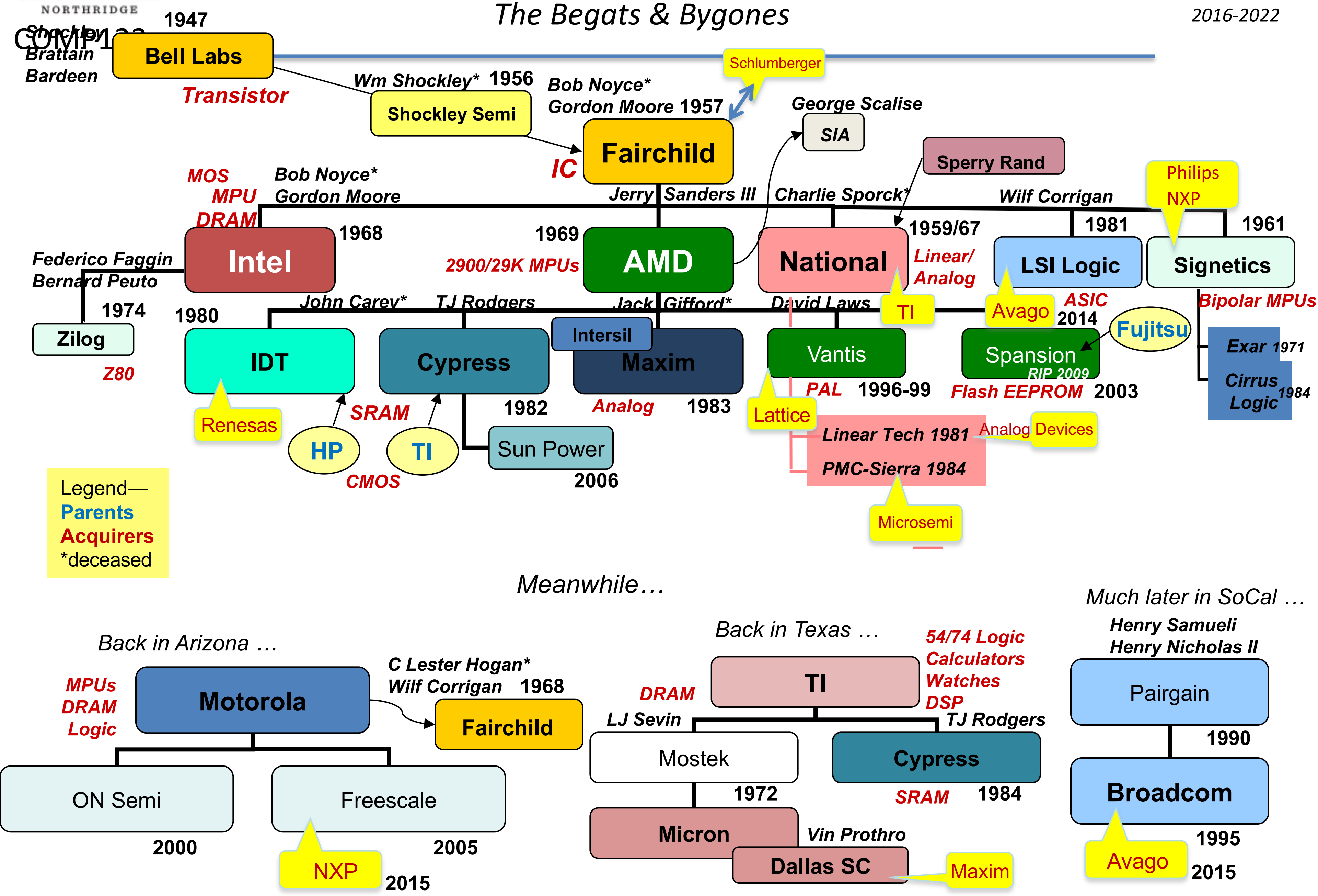
CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE

COMP122

Genesis of Silicon Valley

The Begats & Bygones

DSJ Dr Jeff
DR JEFF SOFTWARE
INDIE APP DEVELOPER
© Jeff Drobman
2016-2022



1947
Shockley
Brattain
Bardeen
Bell Labs
Transistor

*Wm Shockley** 1956
Shockley Semi

*Bob Noyce**
Gordon Moore 1957
Fairchild
IC
Schlumberger

George Scalise
SIA

Sperry Rand

MOS
MPU
DRAM
*Bob Noyce**
Gordon Moore
Intel
1968

Jerry Sanders III
1969
2900/29K MPUs
AMD

*Charlie Sporck**
1959/67
National
Linear/
Analog

Wilf Corrigan
1981
LSI Logic

Philips
NXP
1961
Signetics
Bipolar MPUs

Federico Faggin
Bernard Peuto
1974
Zilog
Z80

1980
*John Carey**
IDT
Renesas
HP

TJ Rodgers
Cypress
SRAM
TI
CMOS
1982
Sun Power
2006

*Jack Gifford**
Intersil
Maxim
Analog
1983

David Laws
TI
Vantis
PAL
1996-99
Lattice

Avago
ASIC
2014
Fujitsu
Spansion
RIP 2009
Flash EEPROM
2003

Exar 1971
Cirrus Logic 1984

Linear Tech 1981
Analog Devices
PMC-Sierra 1984
Microsemi

Legend—
Parents
Acquirers
*deceased

Meanwhile…

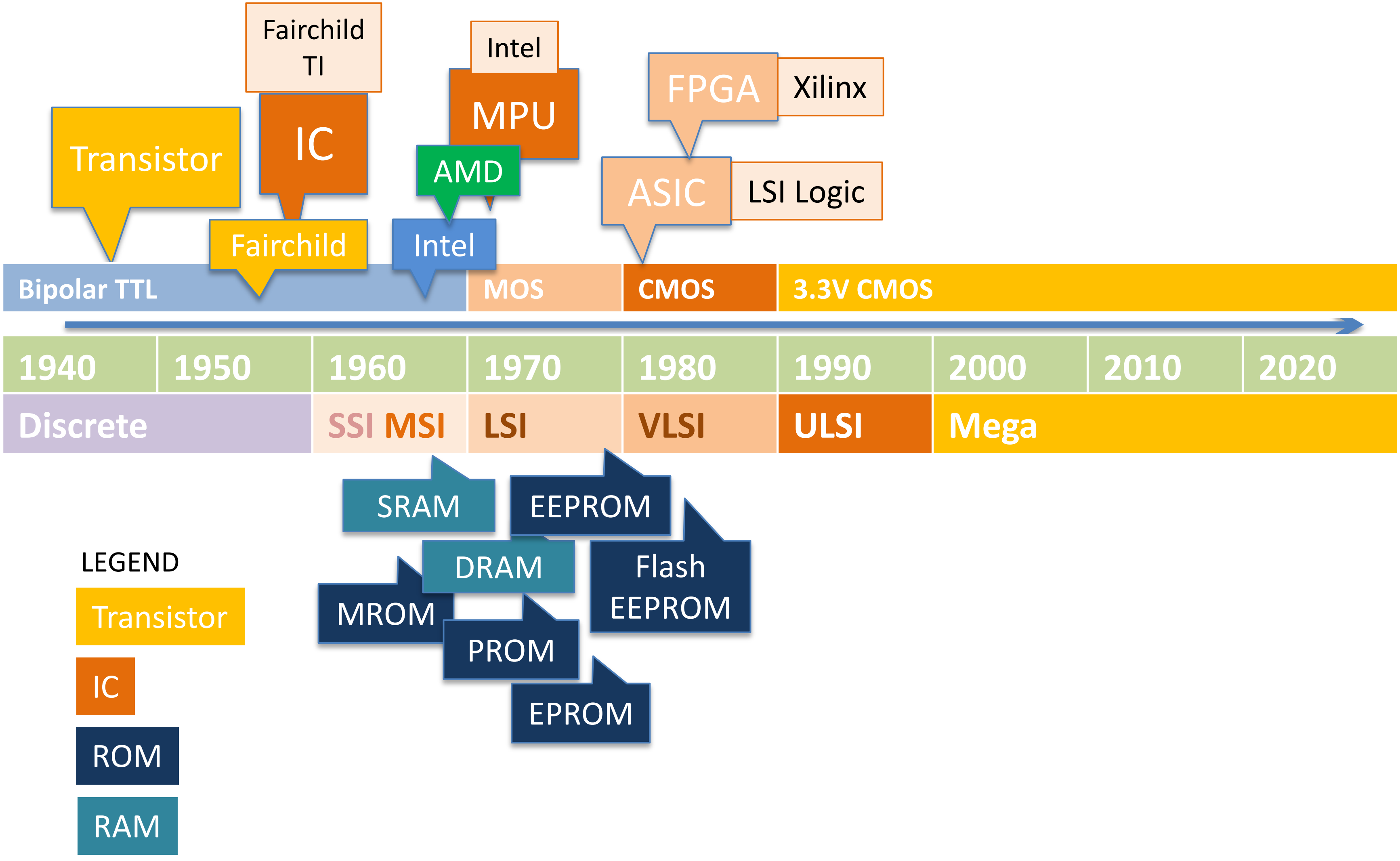
Back in Arizona …
MPUs
DRAM
Logic
Motorola
*C Lester Hogan**
Wilf Corrigan 1968
Fairchild
ON Semi
2000
Freescale
2005
NXP
2015

Back in Texas …
DRAM
TI
54/74 Logic
Calculators
Watches
DSP
LJ Sevin
Mostek
1972
Micron
TJ Rodgers
Cypress
SRAM
1984
Vin Prothro
Dallas SC
Maxim

Much later in SoCal …
Henry Samueli
Henry Nicholas II
Pairgain
1990
Broadcom
1995
Avago
2015

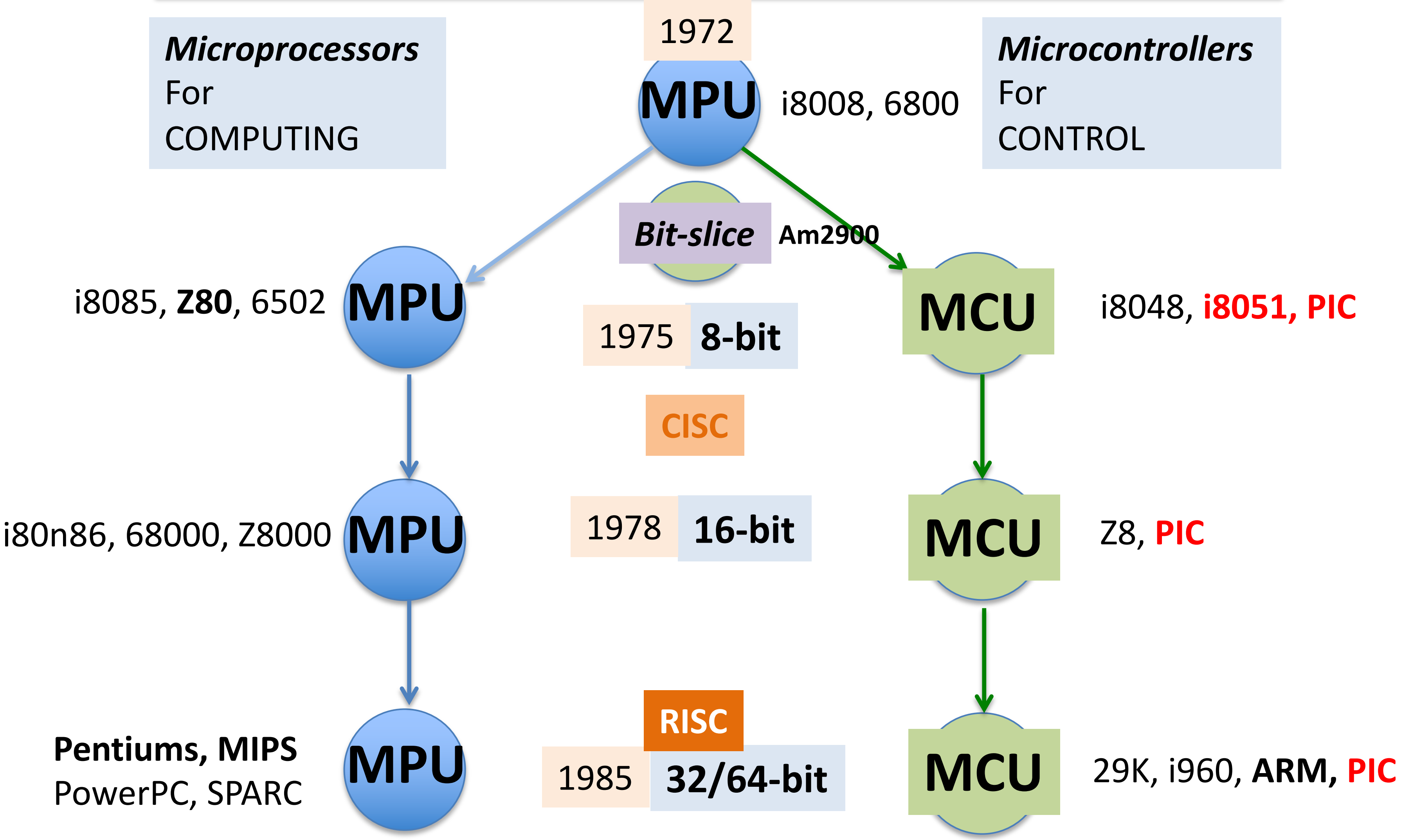
IC Technology

TIMELINE



Microprocessor History

MPU/MCU Generations



Microprocessors
For
COMPUTING

Microcontrollers
For
CONTROL

1972

MPU i8008, 6800

Bit-slice **Am2900**

i8085, **Z80**, 6502 MPU

1975 **8-bit**

MCU i8048, **i8051, PIC**

CISC

i80n86, 68000, Z8000 MPU

1978 **16-bit**

MCU Z8, **PIC**

Pentiums, MIPS
PowerPC, SPARC MPU

RISC

1985 **32/64-bit**

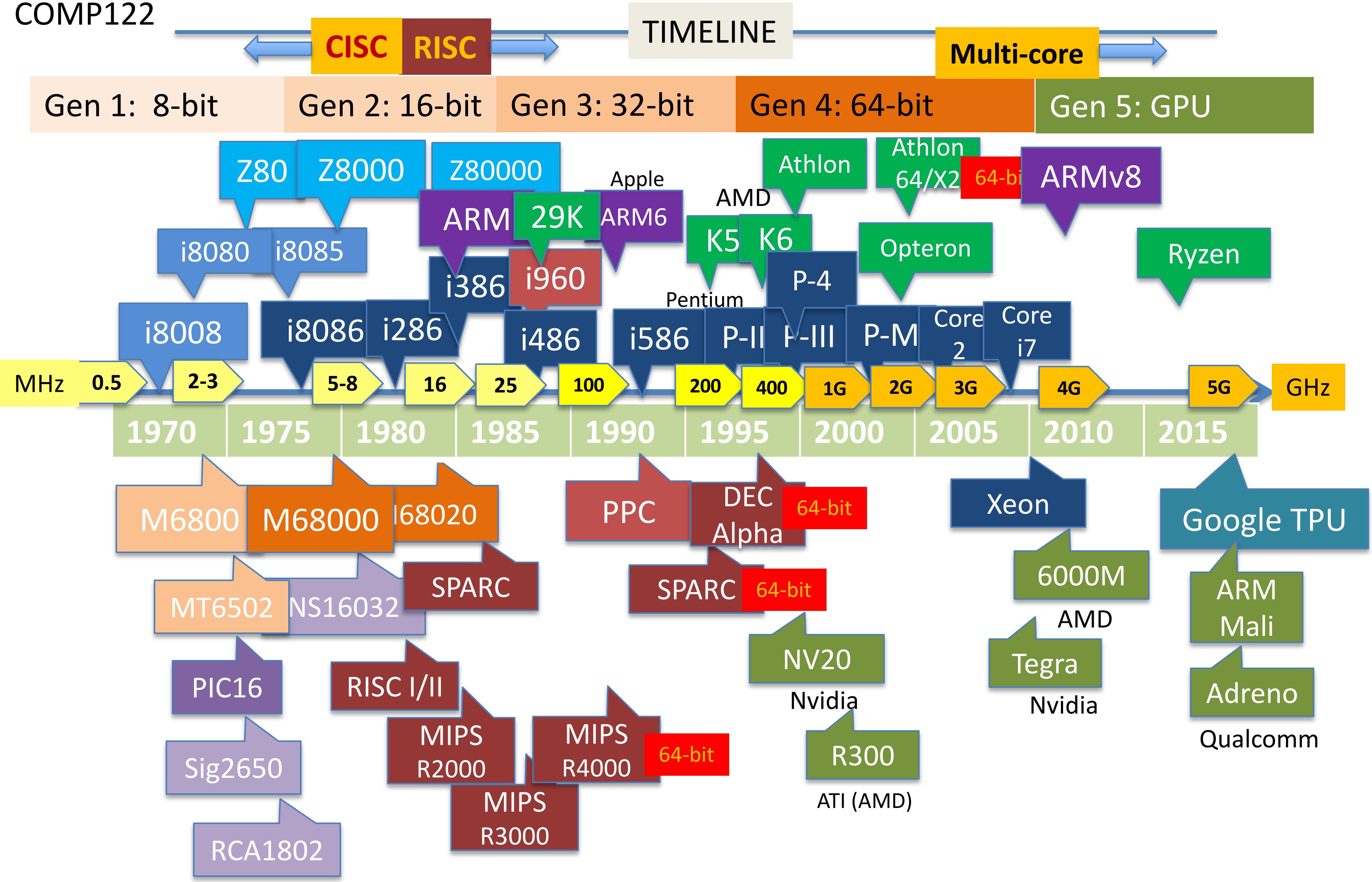
MCU 29K, i960, **ARM**, **PIC**

CSUN
CALIFORNIA
STATE UNIVERSITY
NORTHRIDGE

COMP122

MPU Generations

DSJ Dr Jeff
DR JEFF SOFTWARE
INDIE APP DEVELOPER
© Jeff Drobman
2016-2022



CISC RISC

TIMELINE

Multi-core

Gen 1: 8-bit
Gen 2: 16-bit
Gen 3: 32-bit
Gen 4: 64-bit
Gen 5: GPU

Z80
Z8000
Z80000
Apple
ARM
29K
ARM6
AMD
Athlon
Athlon 64/X2
64-bi
ARMv8
i8080
i8085
K5
K6
Opteron
Ryzen
i386
i960
P-4
Pentium
i8008
i8086
i286
i486
i586
P-II
P-III
P-M
Core 2
Core i7

MHz 0.5 2-3 5-8 16 25 100 200 400 1G 2G 3G 4G 5G GHz

1970 1975 1980 1985 1990 1995 2000 2005 2010 2015

M6800
M68000
68020
PPC
DEC Alpha
64-bit
Xeon
Google TPU
MT6502
NS16032
SPARC
SPARC
64-bit
6000M
AMD
ARM Mali
Tegra
Nvidia
Adreno
Qualcomm
PIC16
RISC I/II
NV20
Nvidia
Sig2650
MIPS R2000
MIPS R4000
64-bit
R300
ATI (AMD)
MIPS R3000
RCA1802

CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE
COMP122

Embedded Control

DR JEFF SOFTWARE
INDIE APP DEVELOPER
© Jeff Drobman
2016-2022

Microprocessors
For
COMPUTING

- All 32/64-bit CPUs
- Large *data processing* applications
  - Employee records
  - Accounting
  - Payroll
- Operating systems (OS)
- "Apps" (applications)
  - PC/Mac
  - Mobile (phones, tablets)
  - Web apps
  - Cloud apps (SaaS)

Focus is ***Memory*** for large Data Files

Large DRAM, Disk, Flash

Microcontrollers
For
CONTROL

✧ *Real-time*
✧ *All-in-one*

- Small *embedded control* applications (8-bit MCU)
  - Appliances
  - Disk controllers
  - Remote controllers
  - Garage/gate openers

✧ Tiny
✧ Low power
✧ Low cost

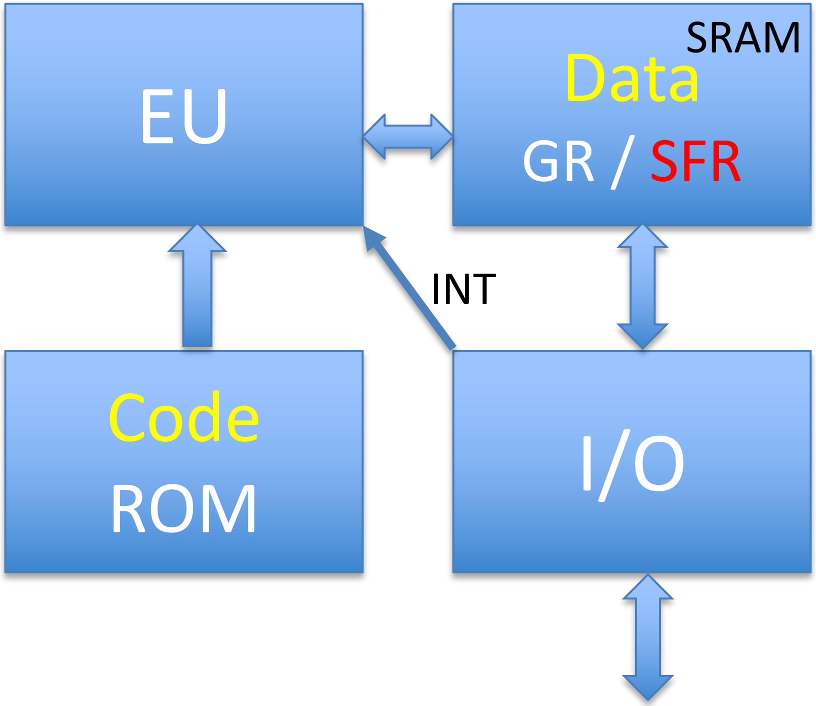
- Medium *embedded control* (16-bit MCU)
  - User devices (iPods, phones, etc.)
  - Car/Airplane engine control
  - Car/Airplane braking & safety
  - Car transmission control
  - Home Automation (HAN)
- Large *embedded control* (32/64-bit MCU)
  - Car/Airplane entertainment
  - Car/Airplane navigation, systems management
  - Printers (MF)
  - Communications gear (WiFi, cable TV boxes)

Focus is ***I/O*** – *Interrupts*

MCU Block Diagram

8/16/32-bit

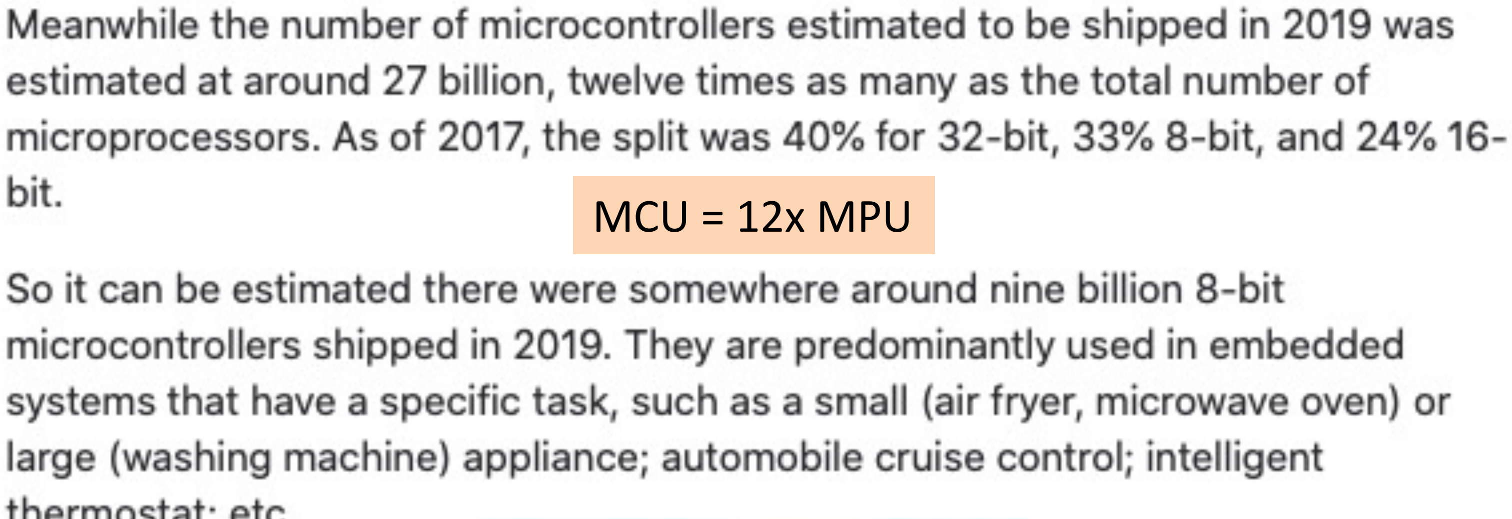
BASIC MODEL



All on one cheap chip

- No Cache
- No External RAM

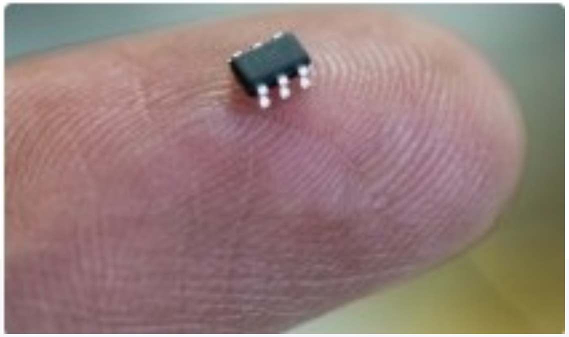
MCU

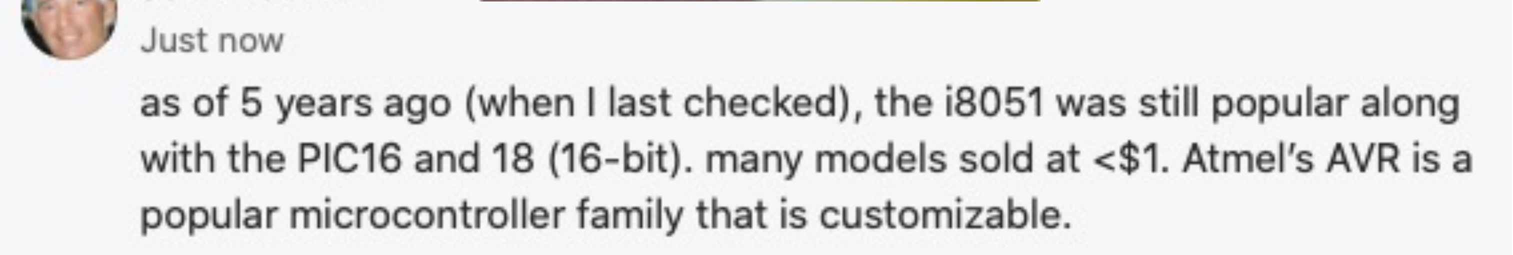


Meanwhile the number of microcontrollers estimated to be shipped in 2019 was estimated at around 27 billion, twelve times as many as the total number of microprocessors. As of 2017, the split was 40% for 32-bit, 33% 8-bit, and 24% 16-bit.

MCU = 12x MPU

So it can be estimated there were somewhere around nine billion 8-bit microcontrollers shipped in 2019. They are predominantly used in embedded systems that have a specific task, such as a small (air fryer, microwave oven) or large (washing machine) appliance; automobile cruise control; intelligent thermostat; etc.



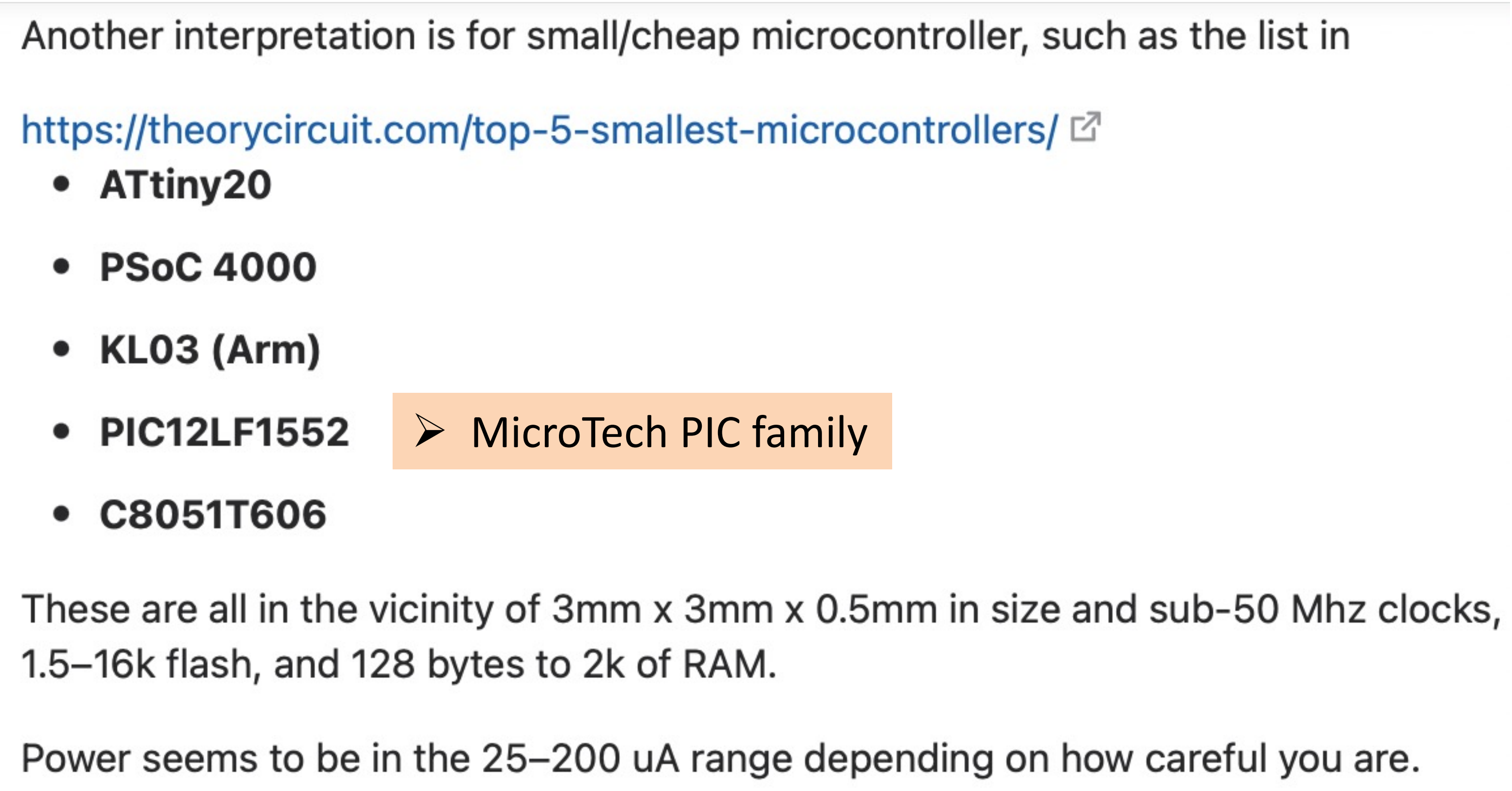


Jeff Drobman
Just now

as of 5 years ago (when I last checked), the i8051 was still popular along with the PIC16 and 18 (16-bit). many models sold at <$1. Atmel's AVR is a popular microcontroller family that is customizable.

Small/Cheap MCU's

Quora



Another interpretation is for small/cheap microcontroller, such as the list in

https://theorycircuit.com/top-5-smallest-microcontrollers/

- **ATtiny20**
- **PSoC 4000**
- **KL03 (Arm)**
- **PIC12LF1552**
- **C8051T606**

➢ MicroTech PIC family

These are all in the vicinity of 3mm x 3mm x 0.5mm in size and sub-50 Mhz clocks, 1.5–16k flash, and 128 bytes to 2k of RAM.

Power seems to be in the 25–200 uA range depending on how careful you are.

CISC vs RISC:

Complex/Reduced Instruction Set Architecture

❖Microprocessor History

- 1971-85: **CISC** (8/16-bit)
  - ✧ Intel i4004 (4-bit)
  - ✧ Intel i8008 (8-bit) → i8080 → i8085, Z80 → i8086 (16-bit) → "x86"
  - ✧ Motorola 6800 (8-bit) → 6502 → 68000 (16-bit)
  - ✧ IBM PC used i8088 (8/16-bit) in 1981 → i80n86 ("x86") → *Pentiums* (now RISC)
- 1985-2000: **RISC** – (32/64-bit)
  - ✧ SPARC* (UC Berkeley→ Sun/Oracle)
  - ✧ MIPS* (Stanford)
  - ✧ PowerPC (Motorola/IBM)
  - ✧ AMD 29K
  - ✧ Intel i960
  - ✧ ARM*

*still exist

Register Arch/Org

Hennessy & Patterson

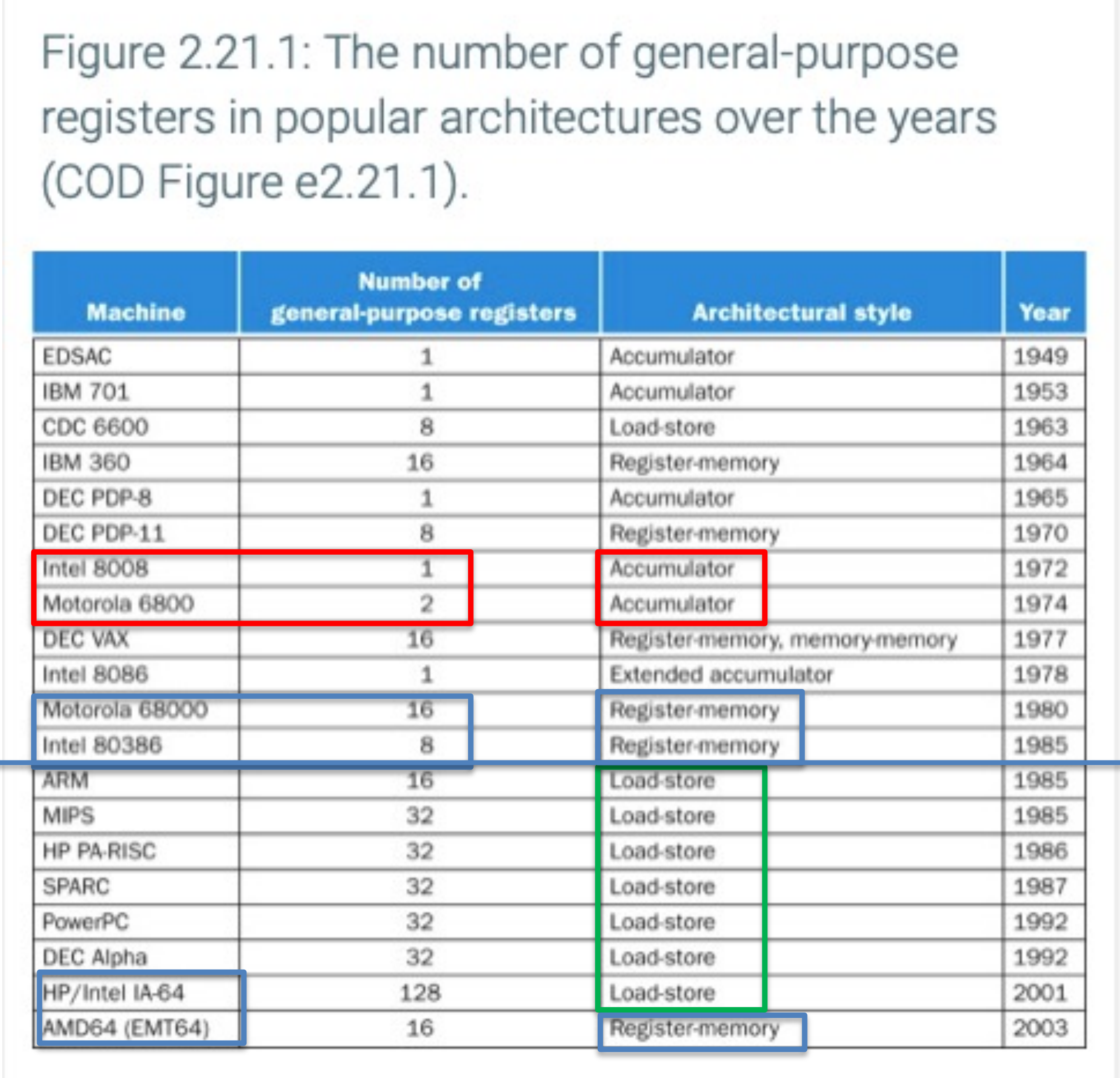


Figure 2.21.1: The number of general-purpose registers in popular architectures over the years (COD Figure e2.21.1).

Machine	Number of general-purpose registers	Architectural style	Year
EDSAC	1	Accumulator	1949
IBM 701	1	Accumulator	1953
CDC 6600	8	Load-store	1963
IBM 360	16	Register-memory	1964
DEC PDP-8	1	Accumulator	1965
DEC PDP-11	8	Register-memory	1970
Intel 8008	1	Accumulator	1972
Motorola 6800	2	Accumulator	1974
DEC VAX	16	Register-memory, memory-memory	1977
Intel 8086	1	Extended accumulator	1978
Motorola 68000	16	Register-memory	1980
Intel 80386	8	Register-memory	1985
ARM	16	Load-store	1985
MIPS	32	Load-store	1985
HP PA-RISC	32	Load-store	1986
SPARC	32	Load-store	1987
PowerPC	32	Load-store	1992
DEC Alpha	32	Load-store	1992
HP/Intel IA-64	128	Load-store	2001
AMD64 (EMT64)	16	Register-memory	2003

CISC

RISC

CISC vs RISC Performance

❖CISC → *CPI* = ~5-9 (typ)

❖RISC → *CPI* = ~1.4 (typ) → 5X faster

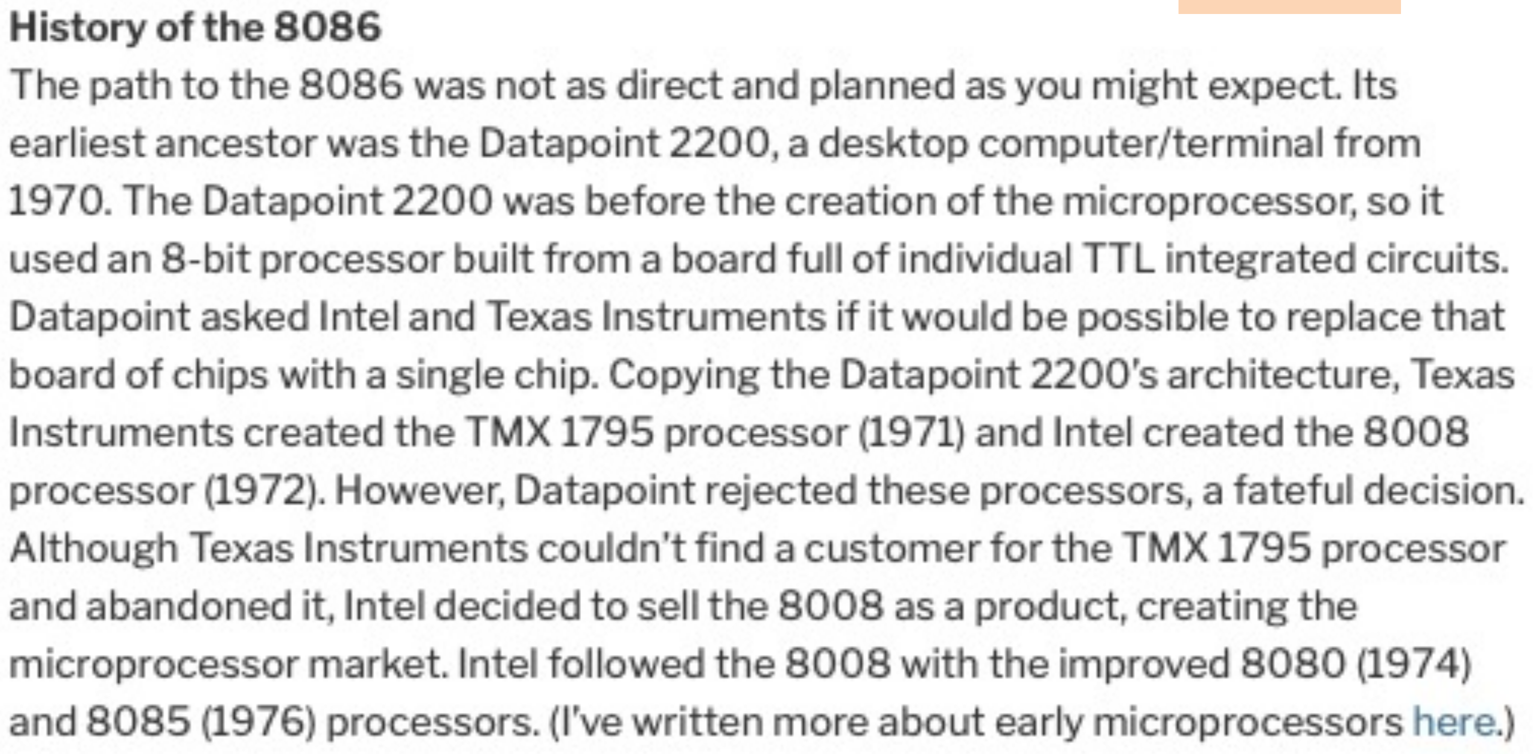
Single core, single pipeline
(no instruction level parallelism)

Single-cycle execution → +Delays for Load, Branch

- Pipeline architecture
- Memory access limited (Load-Store)

i8086 History

WikiSemi



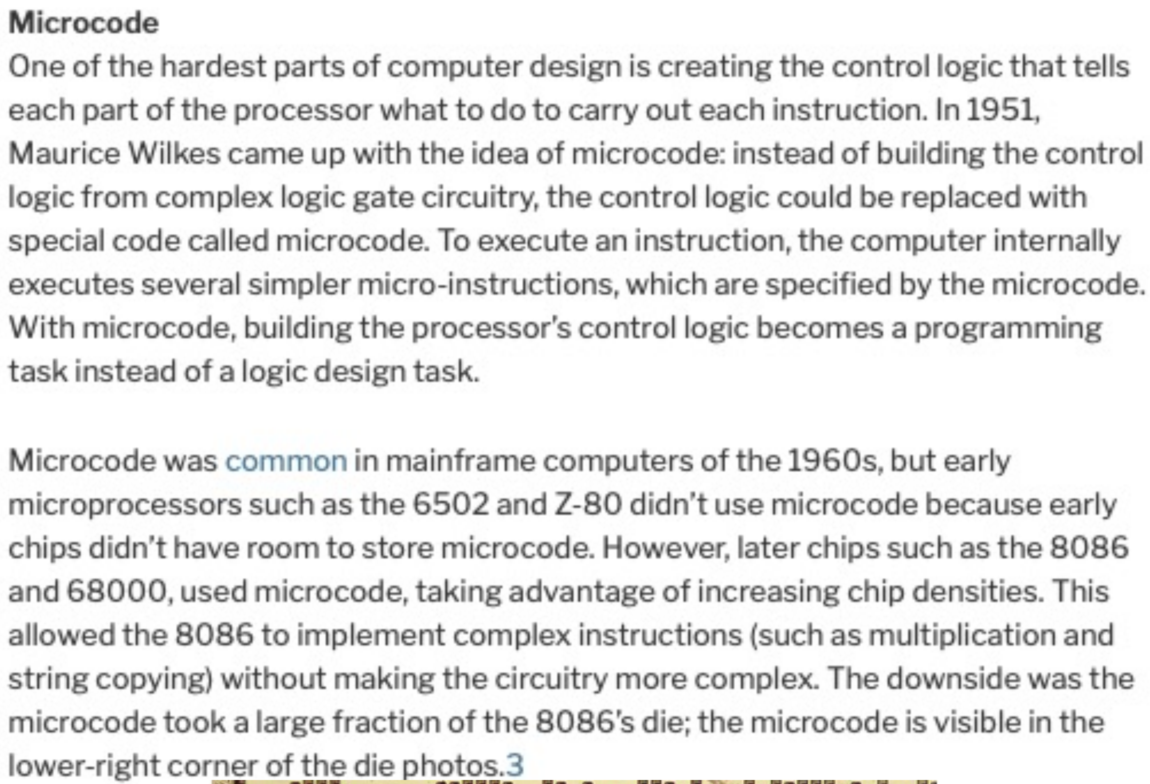
History of the 8086

The path to the 8086 was not as direct and planned as you might expect. Its earliest ancestor was the Datapoint 2200, a desktop computer/terminal from 1970. The Datapoint 2200 was before the creation of the microprocessor, so it used an 8-bit processor built from a board full of individual TTL integrated circuits. Datapoint asked Intel and Texas Instruments if it would be possible to replace that board of chips with a single chip. Copying the Datapoint 2200's architecture, Texas Instruments created the TMX 1795 processor (1971) and Intel created the 8008 processor (1972). However, Datapoint rejected these processors, a fateful decision. Although Texas Instruments couldn't find a customer for the TMX 1795 processor and abandoned it, Intel decided to sell the 8008 as a product, creating the microprocessor market. Intel followed the 8008 with the improved 8080 (1974) and 8085 (1976) processors. (I've written more about early microprocessors here.)

Datapoint 2200 computer. Photo courtesy of Austin Roche.

i8086 History

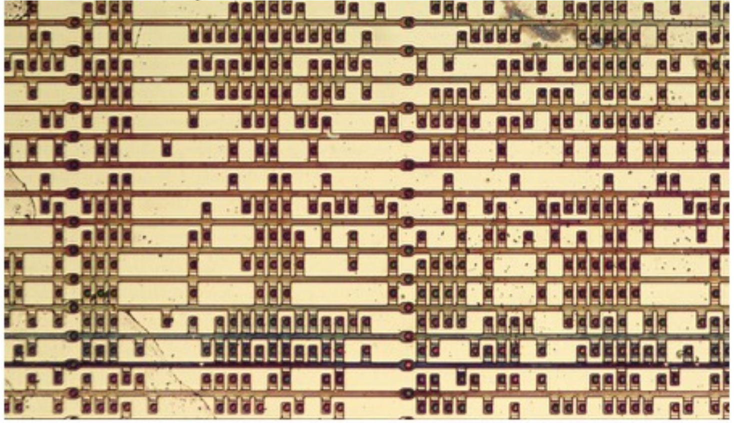
WikiSemi



Microcode

One of the hardest parts of computer design is creating the control logic that tells each part of the processor what to do to carry out each instruction. In 1951, Maurice Wilkes came up with the idea of microcode: instead of building the control logic from complex logic gate circuitry, the control logic could be replaced with special code called microcode. To execute an instruction, the computer internally executes several simpler micro-instructions, which are specified by the microcode. With microcode, building the processor's control logic becomes a programming task instead of a logic design task.

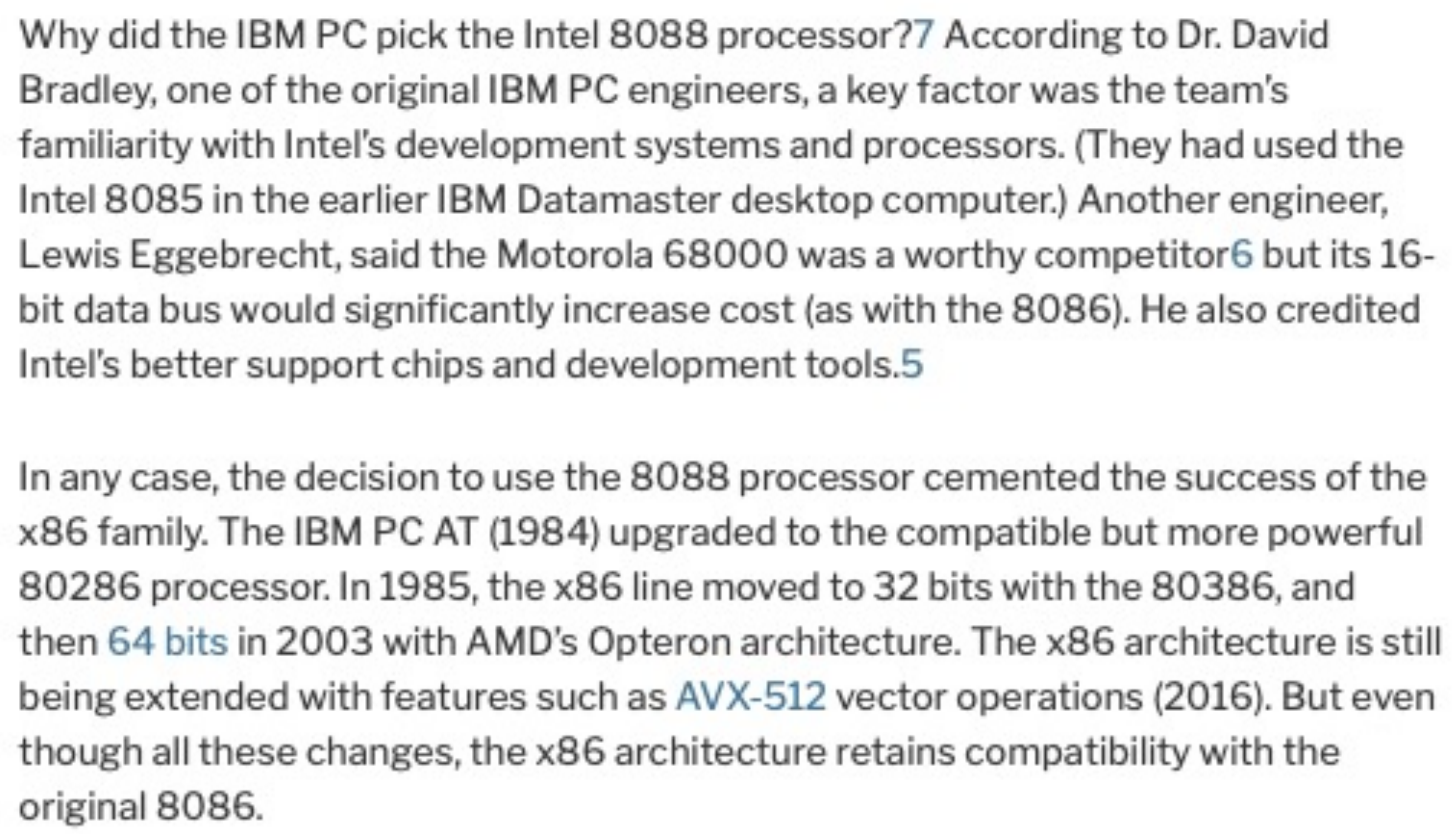
Microcode was common in mainframe computers of the 1960s, but early microprocessors such as the 6502 and Z-80 didn't use microcode because early chips didn't have room to store microcode. However, later chips such as the 8086 and 68000, used microcode, taking advantage of increasing chip densities. This allowed the 8086 to implement complex instructions (such as multiplication and string copying) without making the circuitry more complex. The downside was the microcode took a large fraction of the 8086's die; the microcode is visible in the lower-right corner of the die photos.3



A section of the microcode ROM.

i8086 History

WikiSemi

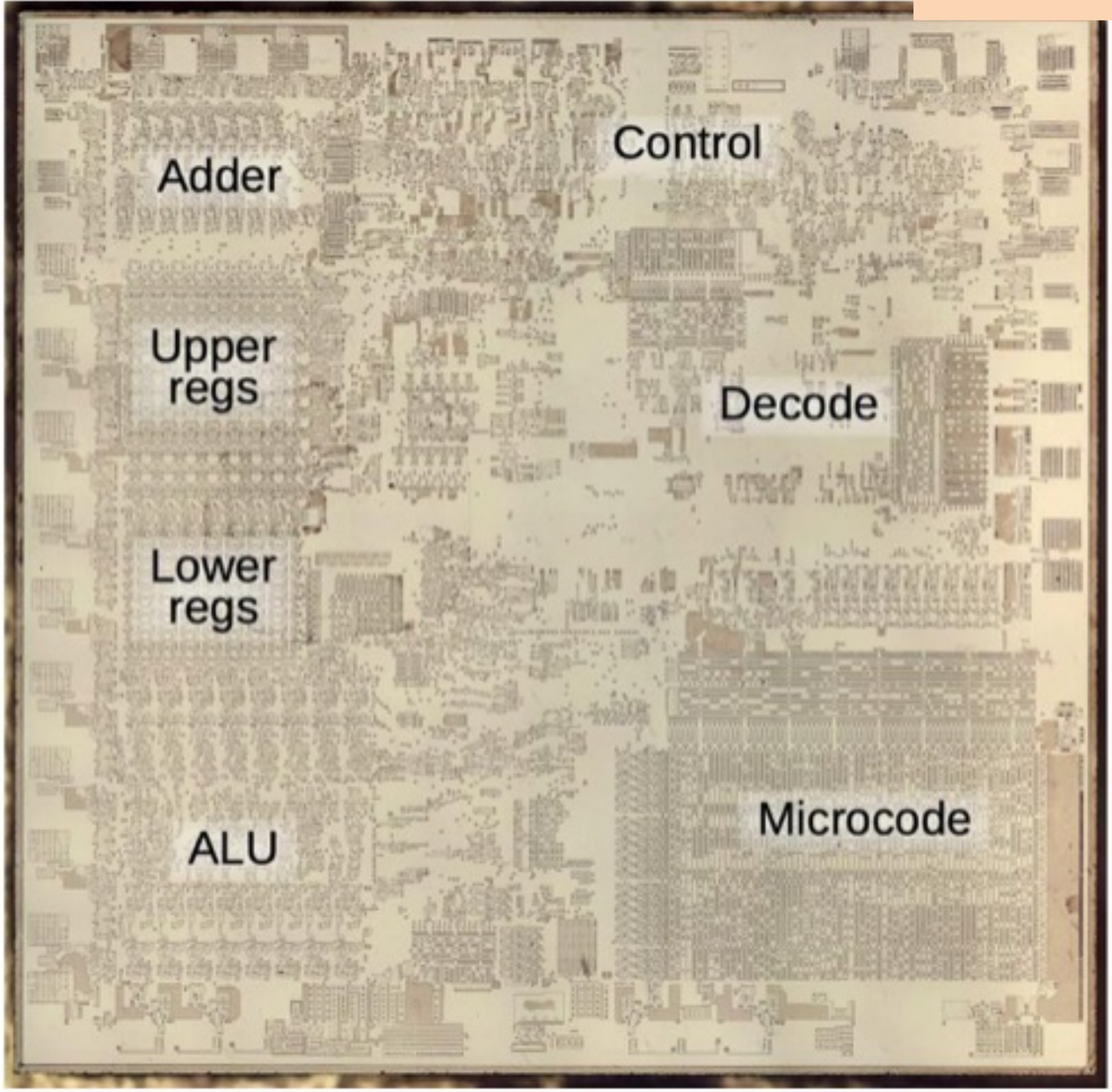


Why did the IBM PC pick the Intel 8088 processor?[7] According to Dr. David Bradley, one of the original IBM PC engineers, a key factor was the team's familiarity with Intel's development systems and processors. (They had used the Intel 8085 in the earlier IBM Datamaster desktop computer.) Another engineer, Lewis Eggebrecht, said the Motorola 68000 was a worthy competitor[6] but its 16-bit data bus would significantly increase cost (as with the 8086). He also credited Intel's better support chips and development tools.[5]

In any case, the decision to use the 8088 processor cemented the success of the x86 family. The IBM PC AT (1984) upgraded to the compatible but more powerful 80286 processor. In 1985, the x86 line moved to 32 bits with the 80386, and then 64 bits in 2003 with AMD's Opteron architecture. The x86 architecture is still being extended with features such as AVX-512 vector operations (2016). But even though all these changes, the x86 architecture retains compatibility with the original 8086.

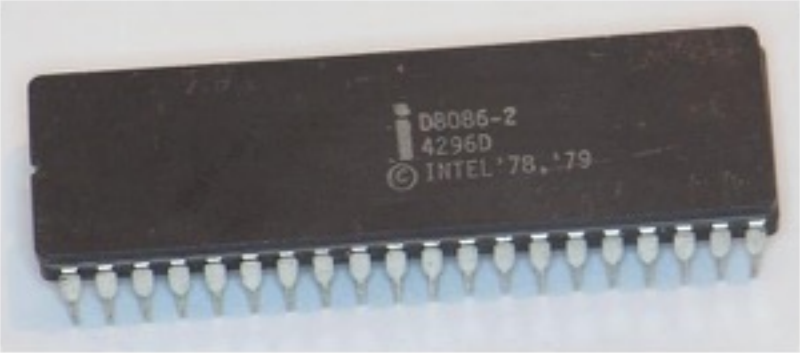
i8086 Die (etched)

WikiSemi

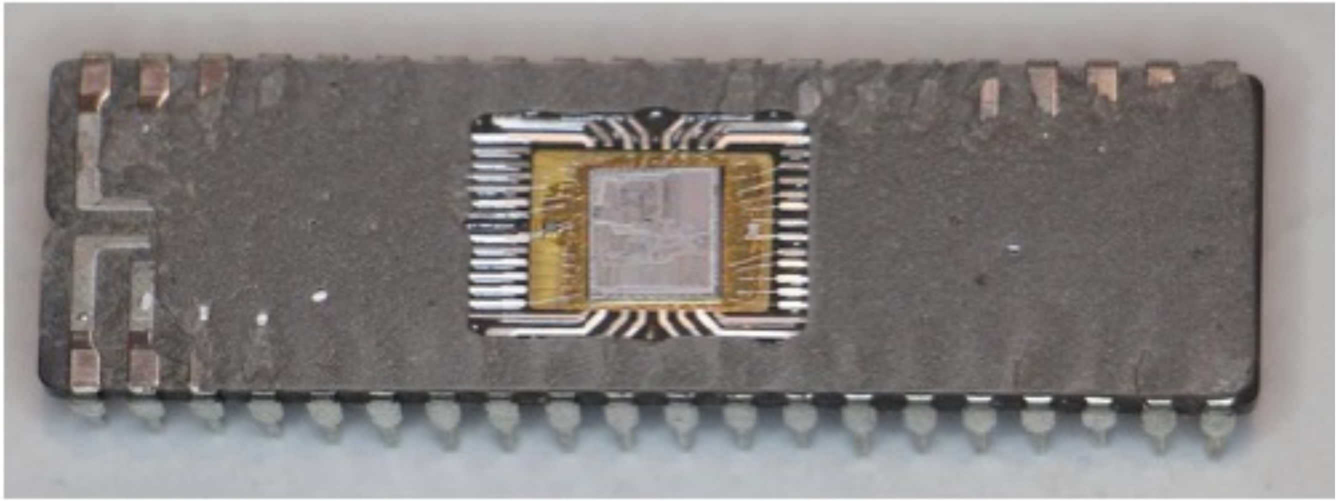


i8086 Packaged

WikiSemi



The 8086 chip, in 40-pin ceramic DIP package.

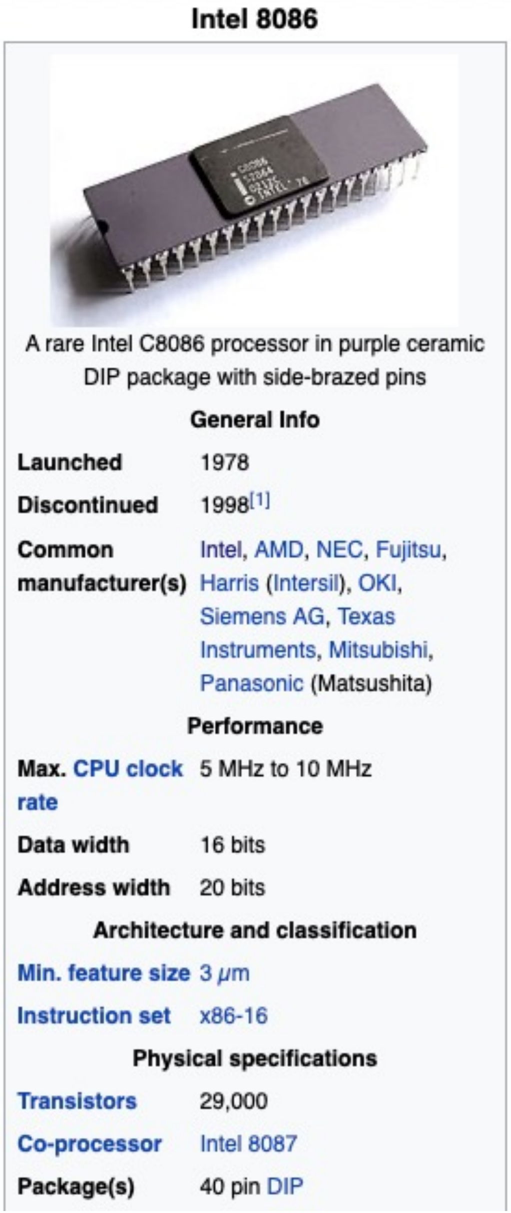


The 8086 die is visible in the middle of the integrated circuit package.

i8086 16-bit MPU

1st 16-bit MPU

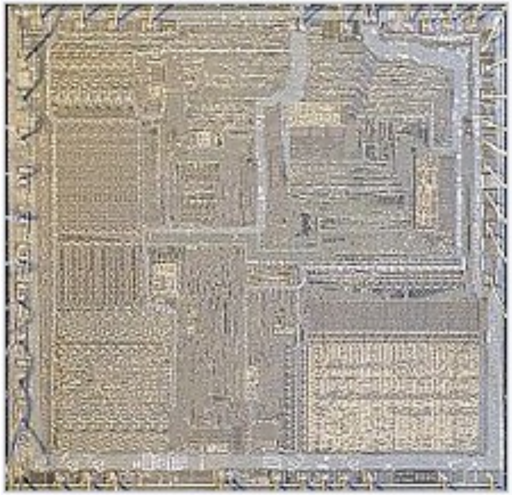
1978



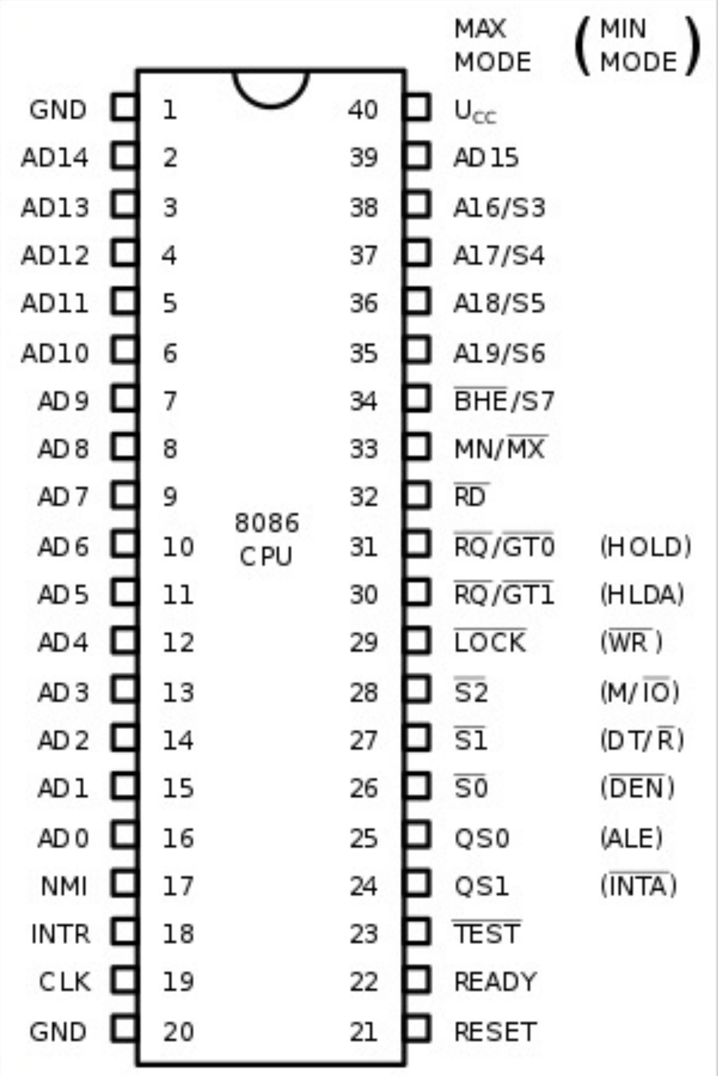
Intel 8086

A rare Intel C8086 processor in purple ceramic DIP package with side-brazed pins

General Info	
Launched	1978
Discontinued	1998[1]
Common manufacturer(s)	Intel, AMD, NEC, Fujitsu, Harris (Intersil), OKI, Siemens AG, Texas Instruments, Mitsubishi, Panasonic (Matsushita)
Performance	
Max. CPU clock rate	5 MHz to 10 MHz
Data width	16 bits
Address width	20 bits
Architecture and classification	
Min. feature size	3 μm
Instruction set	x86-16
Physical specifications	
Transistors	29,000
Co-processor	Intel 8087
Package(s)	40 pin DIP



Intel 8086 CPU die image



			MAX MODE	(MIN MODE)
GND	1	40	U_{CC}	
AD14	2	39	AD15	
AD13	3	38	A16/S3	
AD12	4	37	A17/S4	
AD11	5	36	A18/S5	
AD10	6	35	A19/S6	
AD9	7	34	$\overline{BHE}$/S7	
AD8	8	33	MN/$\overline{MX}$	
AD7	9	32	$\overline{RD}$	
AD6	10	31	$\overline{RQ}/\overline{GT0}$	(HOLD)
AD5	11	30	$\overline{RQ}/\overline{GT1}$	(HLDA)
AD4	12	29	$\overline{LOCK}$	($\overline{WR}$)
AD3	13	28	$\overline{S2}$	(M/$\overline{IO}$)
AD2	14	27	$\overline{S1}$	(DT/$\overline{R}$)
AD1	15	26	$\overline{S0}$	($\overline{DEN}$)
AD0	16	25	QS0	(ALE)
NMI	17	24	QS1	($\overline{INTA}$)
INTR	18	23	$\overline{TEST}$	
CLK	19	22	READY	
GND	20	21	RESET	

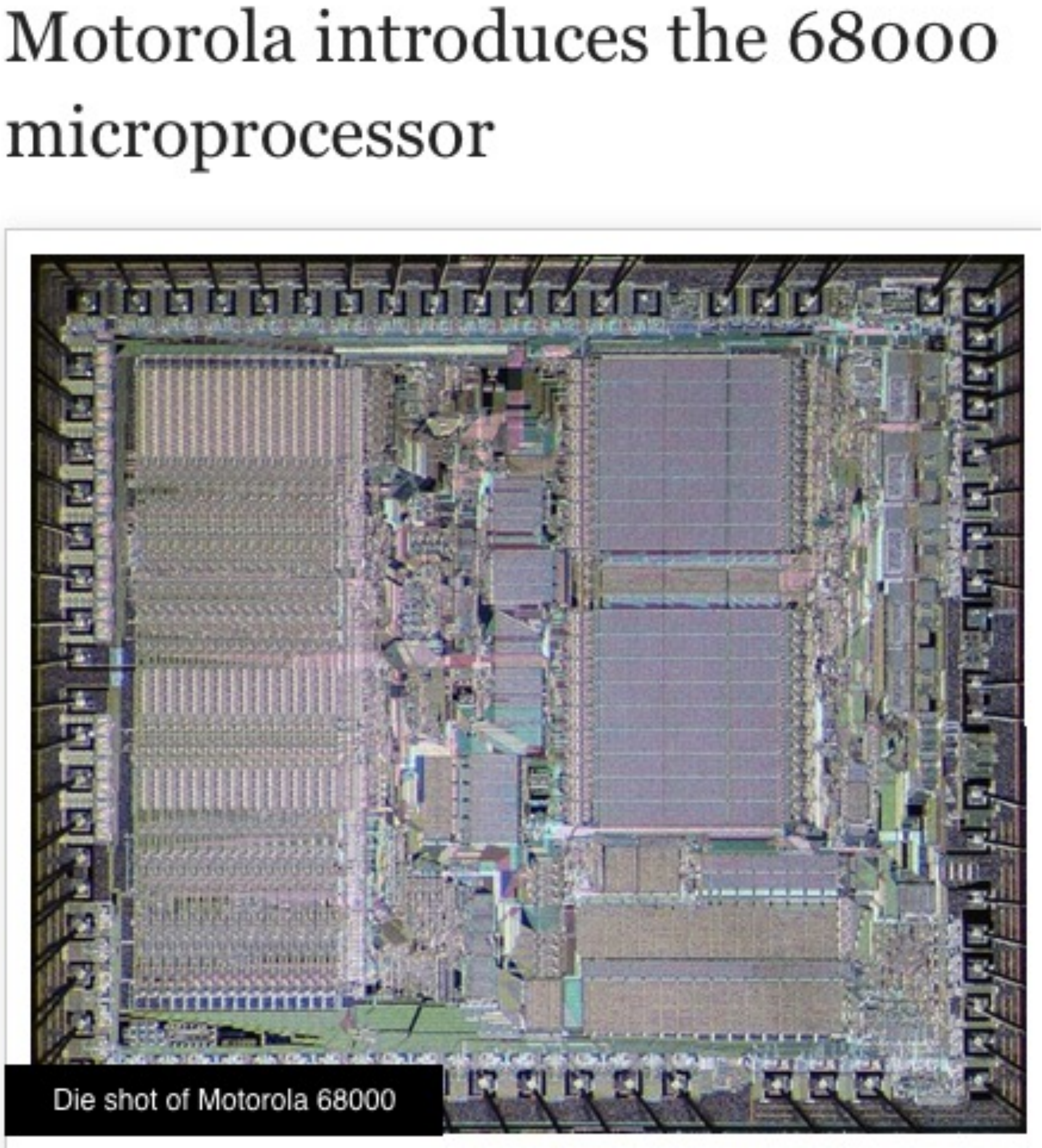
The 8086 pin assignments in min and max mode

CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE
COMP122

DR JEFF SOFTWARE
INDIE APP DEVELOPER
© Jeff Drobman
2016-2022

M68000 16-bit MPU

1980



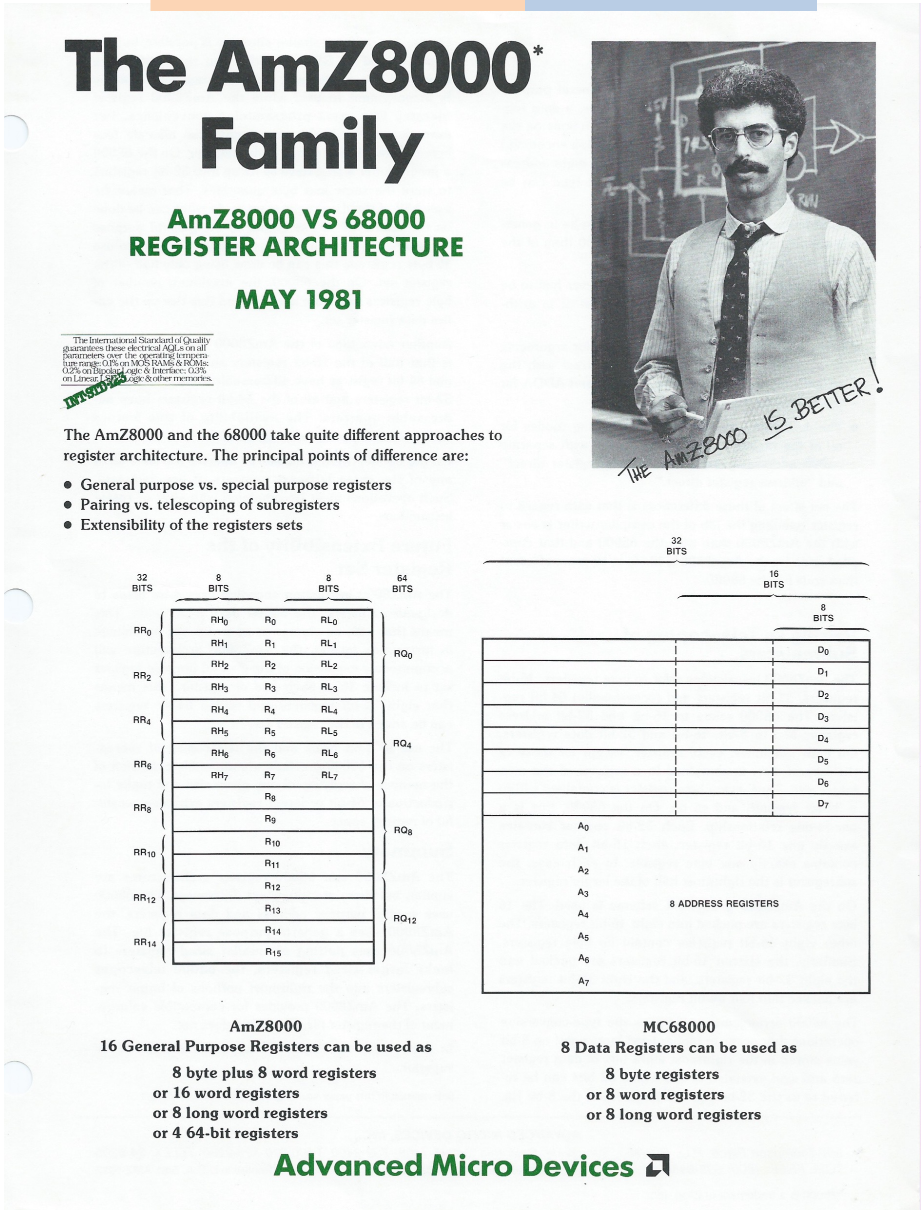
Motorola introduces the 68000 microprocessor

Die shot of Motorola 68000

CPU ISA's

Z8000 vs. M6800 | 16-bit MPU's

- x86
  - ❑ i8088
  - ❑ Pentium
    - Intel P, M
    - AMD K5-8
- MIPS
  - ❑ R3000/4000
  - ❑ MIPS32/64
- ARM
  - ❑ Cortex (A, M)
  - ❑ ARMv7/8
- RISC-V



The AmZ8000* Family

**AmZ8000 VS 68000
REGISTER ARCHITECTURE**

MAY 1981

The International Standard of Quality guarantees these electrical AQLs on all parameters over the operating temperature range: 0.1% on MOS RAMs & ROMs: 0.2% on Bipolar Logic & Interface: 0.3% on Linear, LSI, Logic & other memories.

INT-STD-123

THE AmZ8000 IS BETTER!

The AmZ8000 and the 68000 take quite different approaches to register architecture. The principal points of difference are:

- General purpose vs. special purpose registers
- Pairing vs. telescoping of subregisters
- Extensibility of the registers sets

**AmZ8000
16 General Purpose Registers can be used as**

8 byte plus 8 word registers
or 16 word registers
or 8 long word registers
or 4 64-bit registers

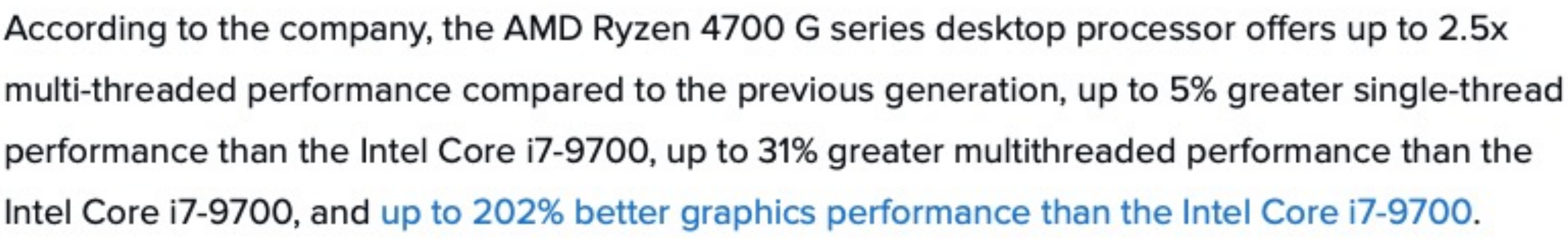
**MC68000
8 Data Registers can be used as**

8 byte registers
or 8 word registers
or 8 long word registers

Advanced Micro Devices

AMD vs Intel: CPU Families

Market Segment	AMD	Intel
Desktop	Ryzen 4K/Athlon 3K	Core i7/i9 (10th gen)
Laptop	Ryzen 4000	Ice Lake
Gaming	Ryzen Threadripper +Radeon	Core Extreme
Server/Workstn	Epyc	Xeon



According to the company, the AMD Ryzen 4700 G series desktop processor offers up to 2.5x multi-threaded performance compared to the previous generation, up to 5% greater single-thread performance than the Intel Core i7-9700, up to 31% greater multithreaded performance than the Intel Core i7-9700, and up to 202% better graphics performance than the Intel Core i7-9700.

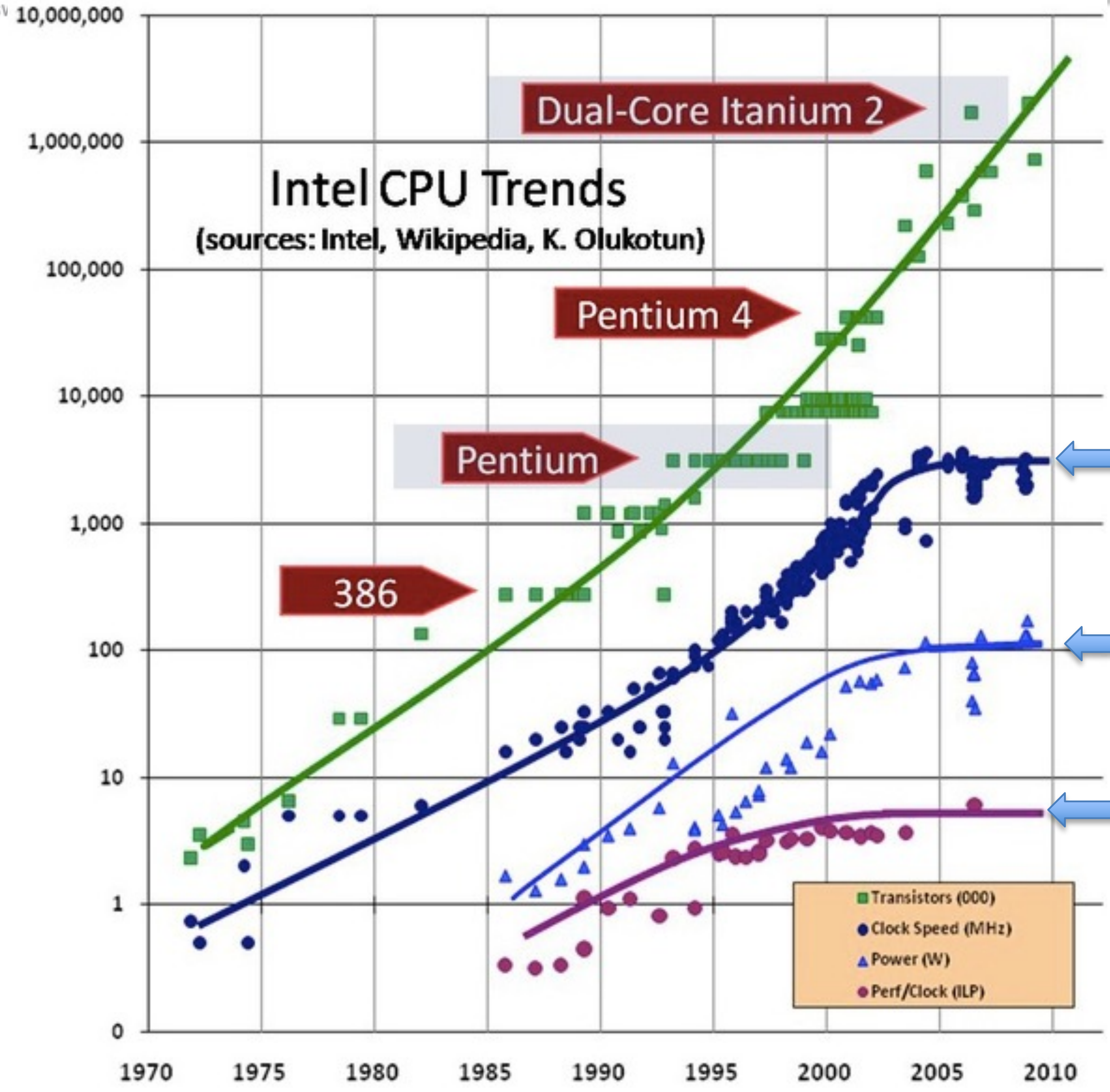
IC & Microprocessor Trends

Quora

CPU Trends

Steve Baker, Blogger at LetsRunWithIt.com (2013-present)

Answ



5 GHz

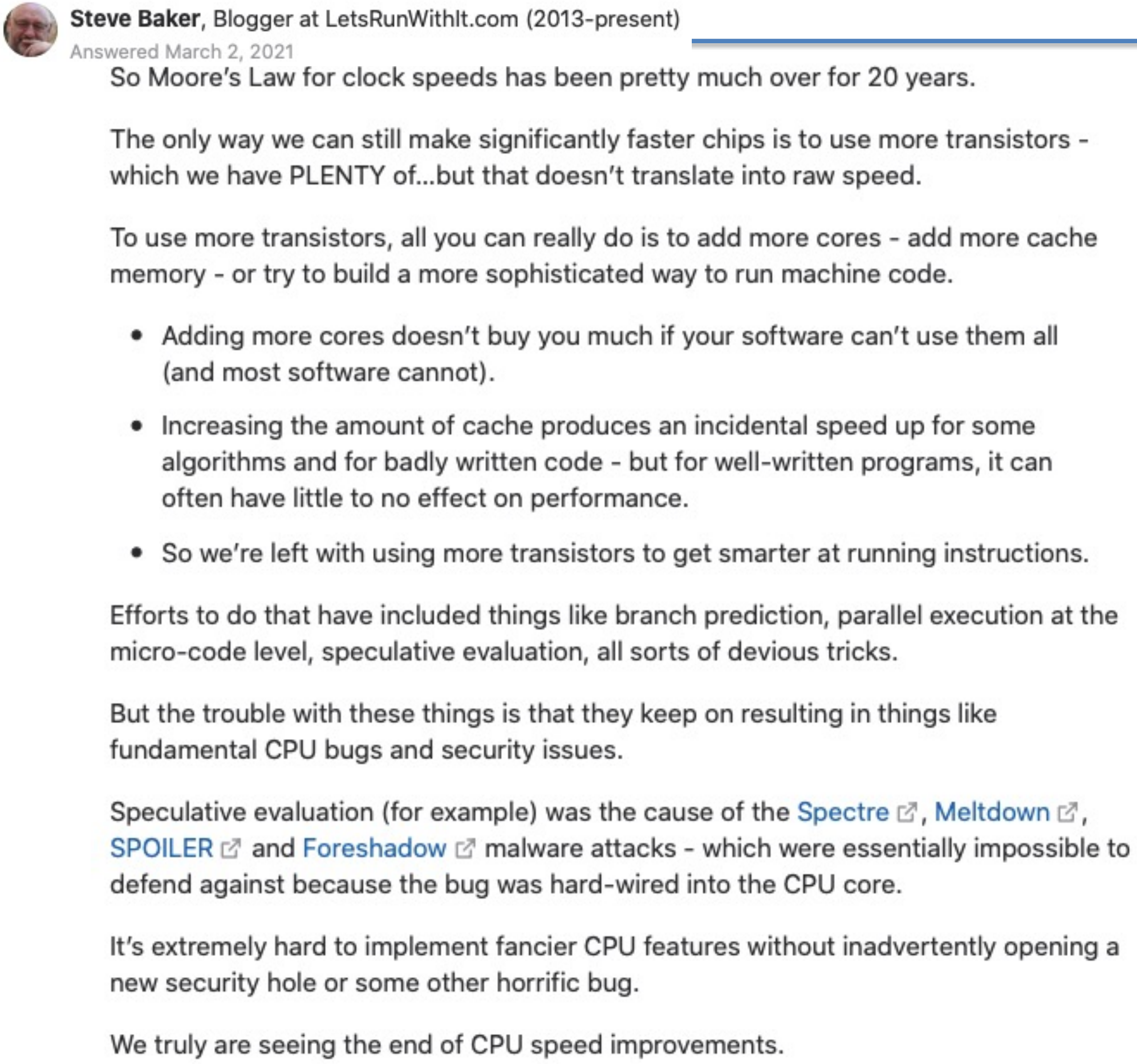
100 W

CPI/IPC

Micro-arch

Quora

CPU Trends



Steve Baker, Blogger at LetsRunWithIt.com (2013-present)
Answered March 2, 2021

So Moore's Law for clock speeds has been pretty much over for 20 years.

The only way we can still make significantly faster chips is to use more transistors - which we have PLENTY of...but that doesn't translate into raw speed.

To use more transistors, all you can really do is to add more cores - add more cache memory - or try to build a more sophisticated way to run machine code.

- Adding more cores doesn't buy you much if your software can't use them all (and most software cannot).
- Increasing the amount of cache produces an incidental speed up for some algorithms and for badly written code - but for well-written programs, it can often have little to no effect on performance.
- So we're left with using more transistors to get smarter at running instructions.

Efforts to do that have included things like branch prediction, parallel execution at the micro-code level, speculative evaluation, all sorts of devious tricks.

But the trouble with these things is that they keep on resulting in things like fundamental CPU bugs and security issues.

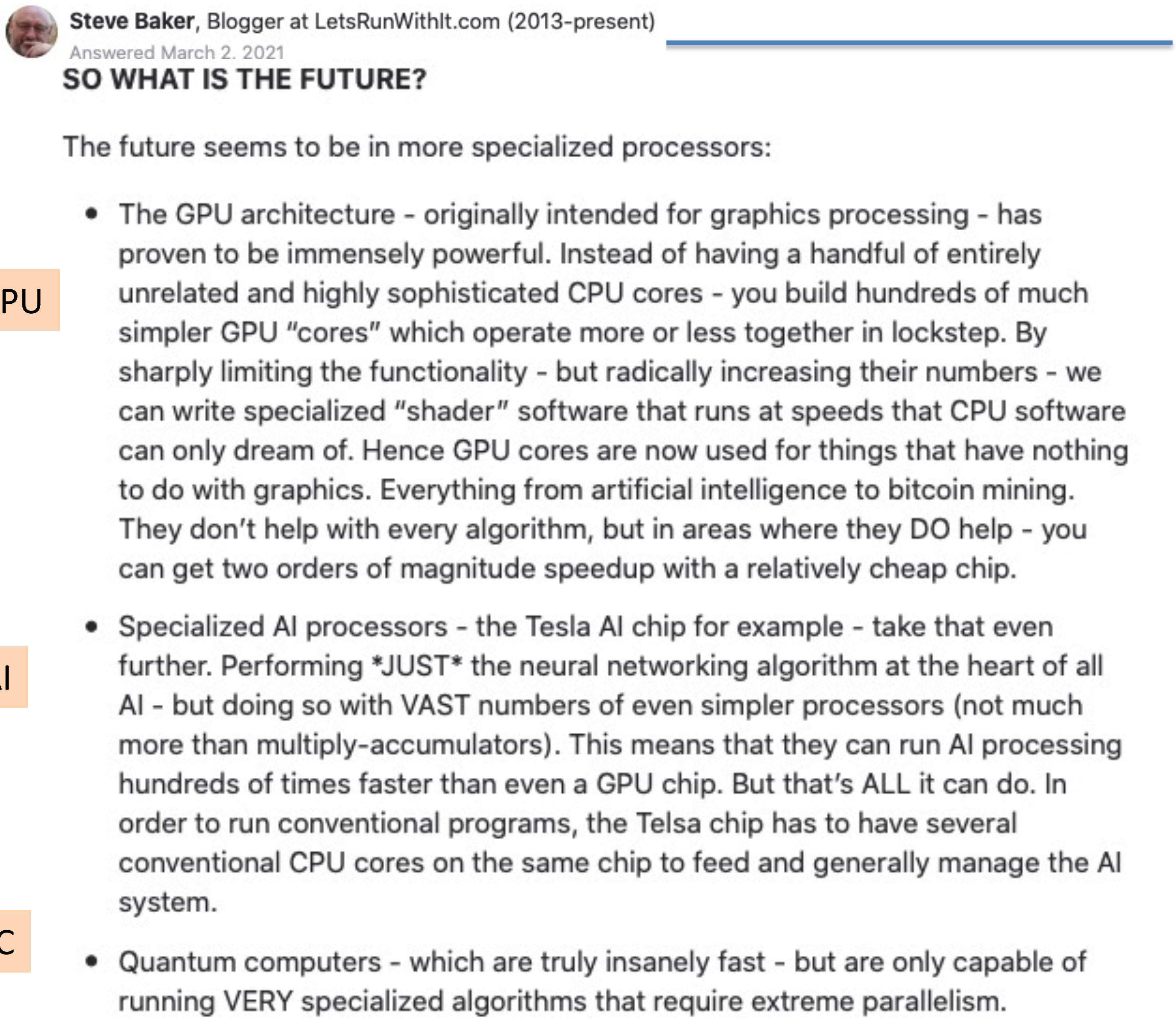
Speculative evaluation (for example) was the cause of the Spectre, Meltdown, SPOILER and Foreshadow malware attacks - which were essentially impossible to defend against because the bug was hard-wired into the CPU core.

It's extremely hard to implement fancier CPU features without inadvertently opening a new security hole or some other horrific bug.

We truly are seeing the end of CPU speed improvements.

Quora

CPU Trends



Steve Baker, Blogger at LetsRunWithIt.com (2013-present)

Answered March 2. 2021

SO WHAT IS THE FUTURE?

The future seems to be in more specialized processors:

GPU

- The GPU architecture - originally intended for graphics processing - has proven to be immensely powerful. Instead of having a handful of entirely unrelated and highly sophisticated CPU cores - you build hundreds of much simpler GPU "cores" which operate more or less together in lockstep. By sharply limiting the functionality - but radically increasing their numbers - we can write specialized "shader" software that runs at speeds that CPU software can only dream of. Hence GPU cores are now used for things that have nothing to do with graphics. Everything from artificial intelligence to bitcoin mining. They don't help with every algorithm, but in areas where they DO help - you can get two orders of magnitude speedup with a relatively cheap chip.

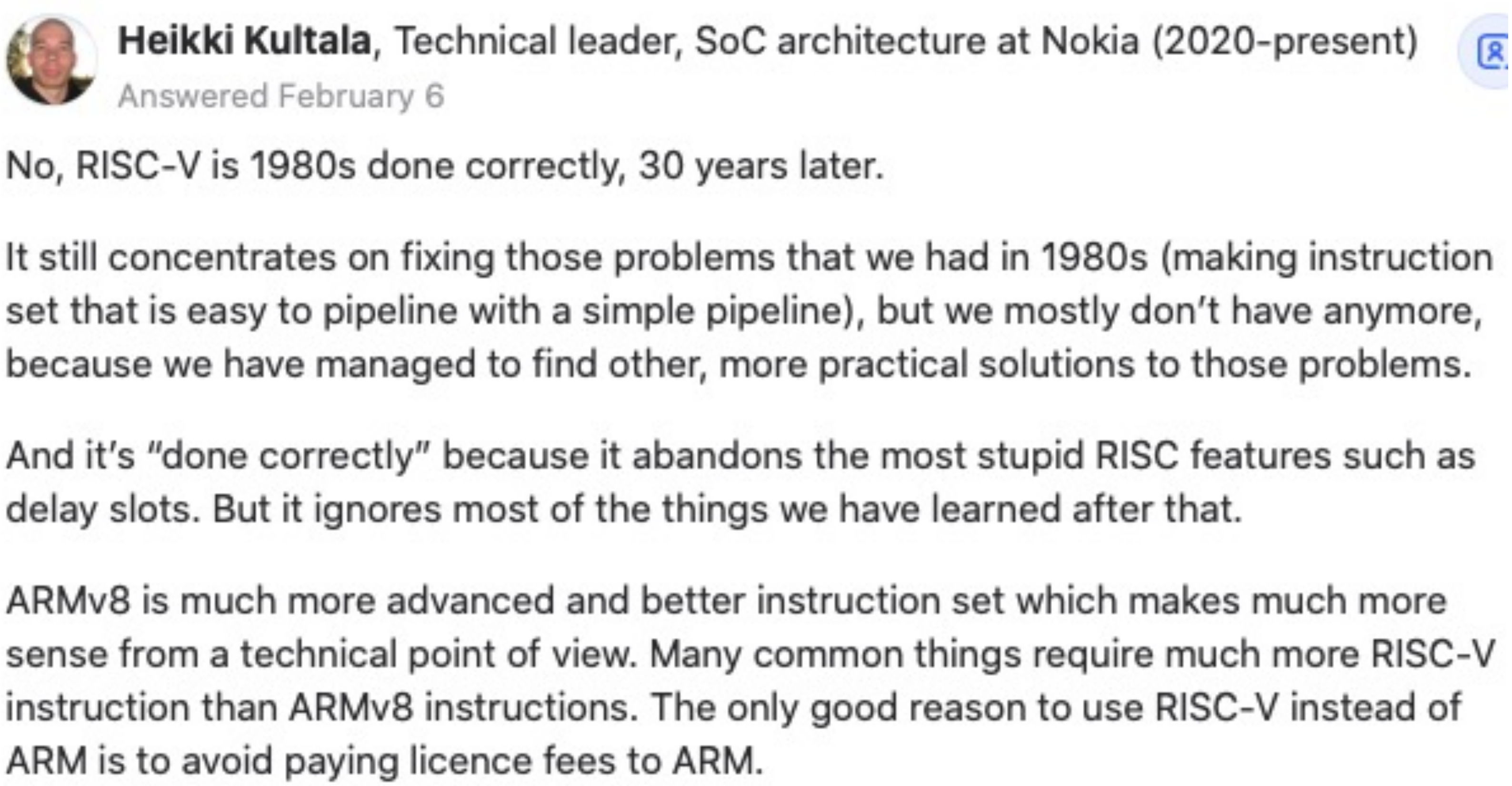
AI

- Specialized AI processors - the Tesla AI chip for example - take that even further. Performing *JUST* the neural networking algorithm at the heart of all AI - but doing so with VAST numbers of even simpler processors (not much more than multiply-accumulators). This means that they can run AI processing hundreds of times faster than even a GPU chip. But that's ALL it can do. In order to run conventional programs, the Telsa chip has to have several conventional CPU cores on the same chip to feed and generally manage the AI system.

QC

- Quantum computers - which are truly insanely fast - but are only capable of running VERY specialized algorithms that require extreme parallelism.

RISC-V



Heikki Kultala, Technical leader, SoC architecture at Nokia (2020-present)

Answered February 6

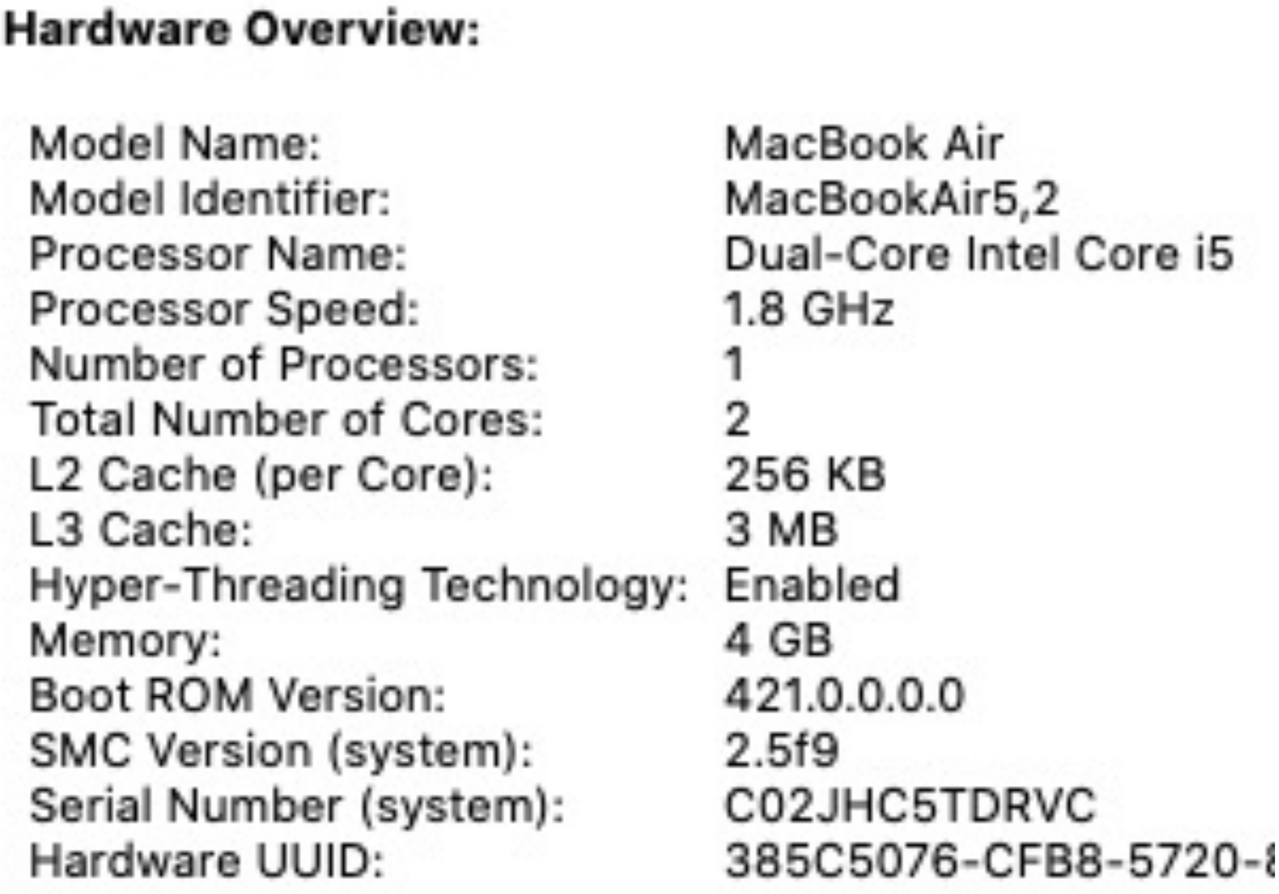
No, RISC-V is 1980s done correctly, 30 years later.

It still concentrates on fixing those problems that we had in 1980s (making instruction set that is easy to pipeline with a simple pipeline), but we mostly don’t have anymore, because we have managed to find other, more practical solutions to those problems.

And it’s “done correctly” because it abandons the most stupid RISC features such as delay slots. But it ignores most of the things we have learned after that.

ARMv8 is much more advanced and better instruction set which makes much more sense from a technical point of view. Many common things require much more RISC-V instruction than ARMv8 instructions. The only good reason to use RISC-V instead of ARM is to avoid paying licence fees to ARM.

Example Arch: My Mac



Hardware Overview:

Model Name:	MacBook Air
Model Identifier:	MacBookAir5,2
Processor Name:	Dual-Core Intel Core i5
Processor Speed:	1.8 GHz
Number of Processors:	1
Total Number of Cores:	2
L2 Cache (per Core):	256 KB
L3 Cache:	3 MB
Hyper-Threading Technology:	Enabled
Memory:	4 GB
Boot ROM Version:	421.0.0.0.0
SMC Version (system):	2.5f9
Serial Number (system):	C02JHC5TDRVC
Hardware UUID:	385C5076-CFB8-5720-

Other CPU Chips

Apple
A/M Series
ARM
SoC

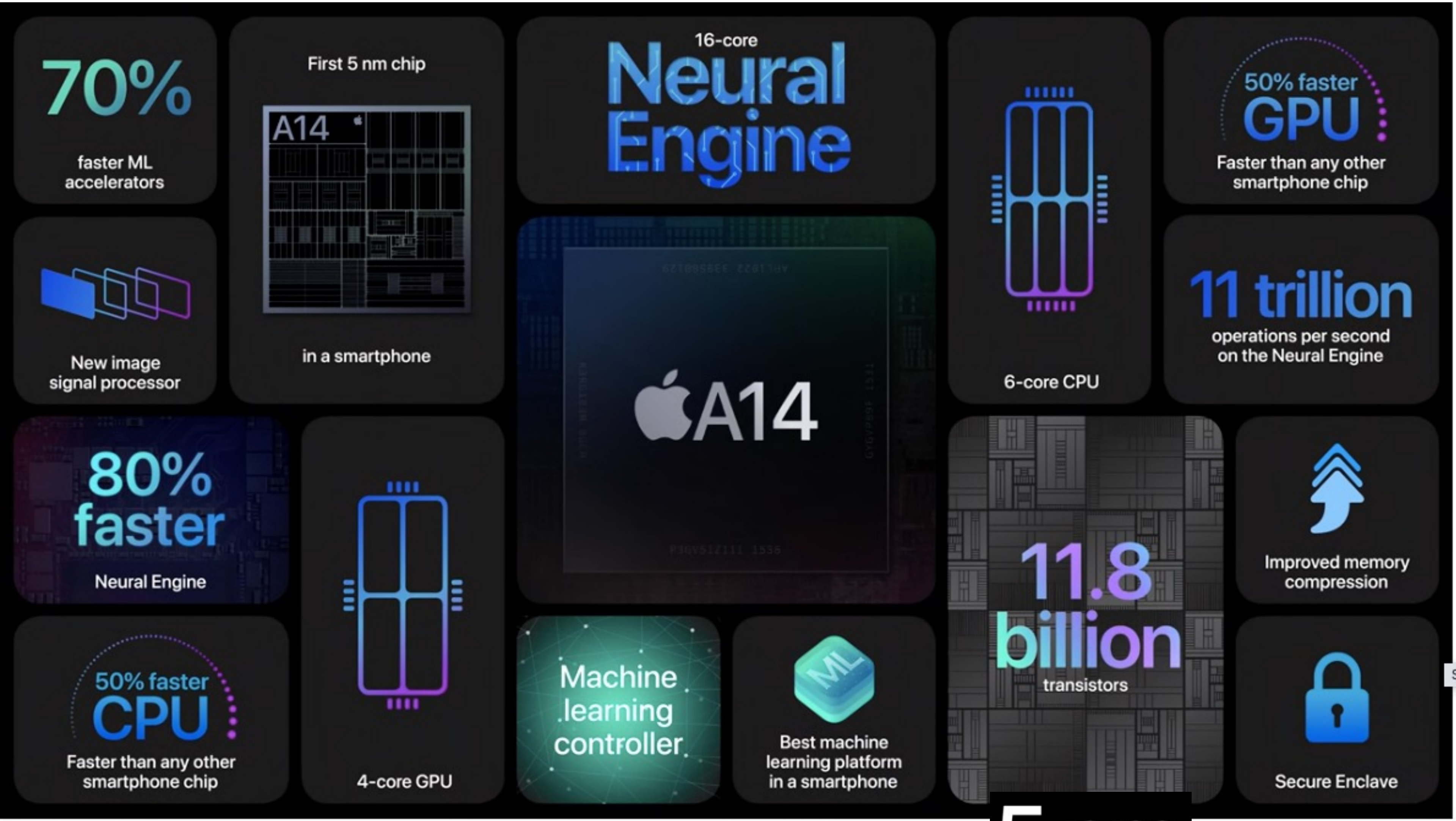
See separate slide set: ***SoC***

New Apple A14/iPads

Apple A14

October 13, 2020

A14 Bionic



70% faster ML accelerators

First 5 nm chip

A14

in a smartphone

16-core Neural Engine

6-core CPU

50% faster GPU

Faster than any other smartphone chip

New image signal processor

A14

11 trillion operations per second on the Neural Engine

80% faster Neural Engine

4-core GPU

11.8 billion transistors

Improved memory compression

50% faster CPU

Faster than any other smartphone chip

Machine learning controller

ML

Best machine learning platform in a smartphone

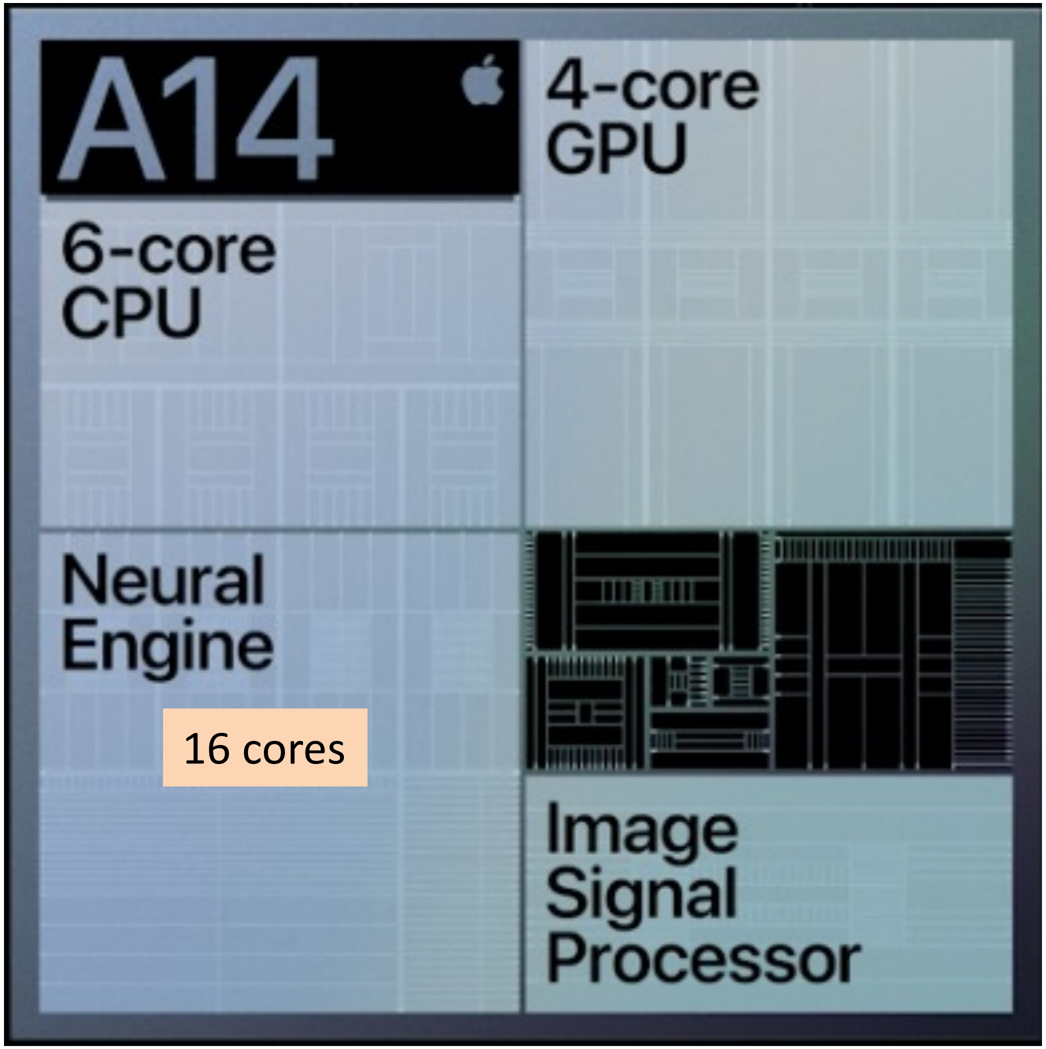
Secure Enclave

5 nm

Apple A14

October 13, 2020

A14

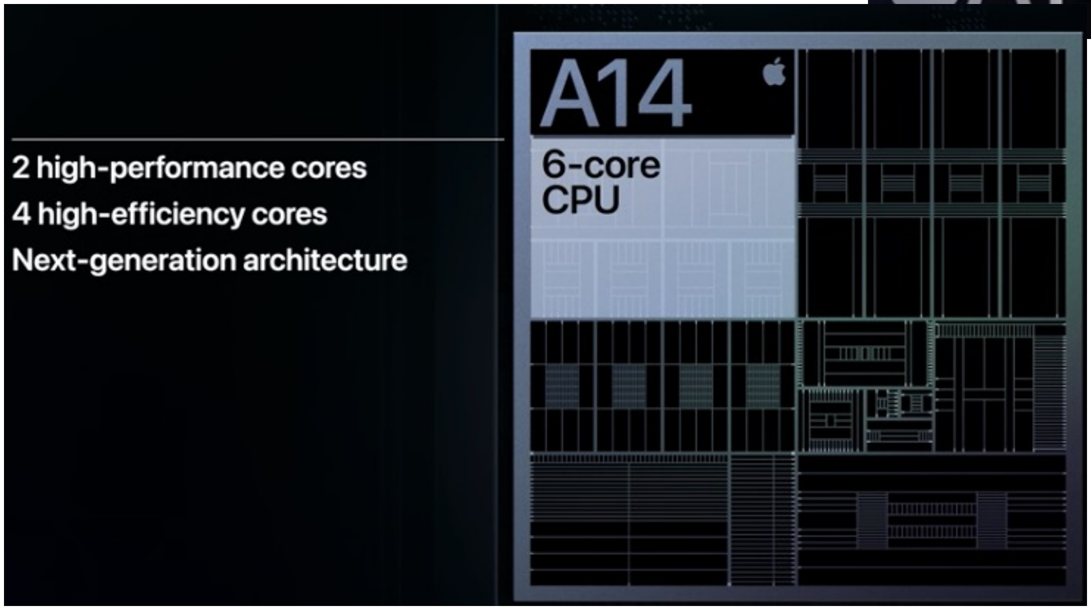


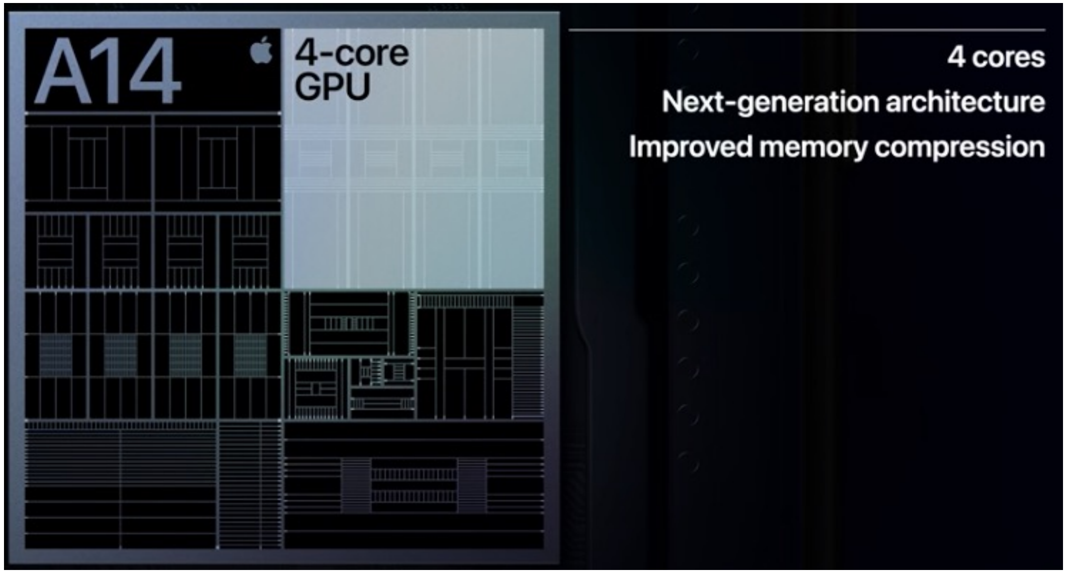
16 cores

Apple A14

October 13, 2020

A14

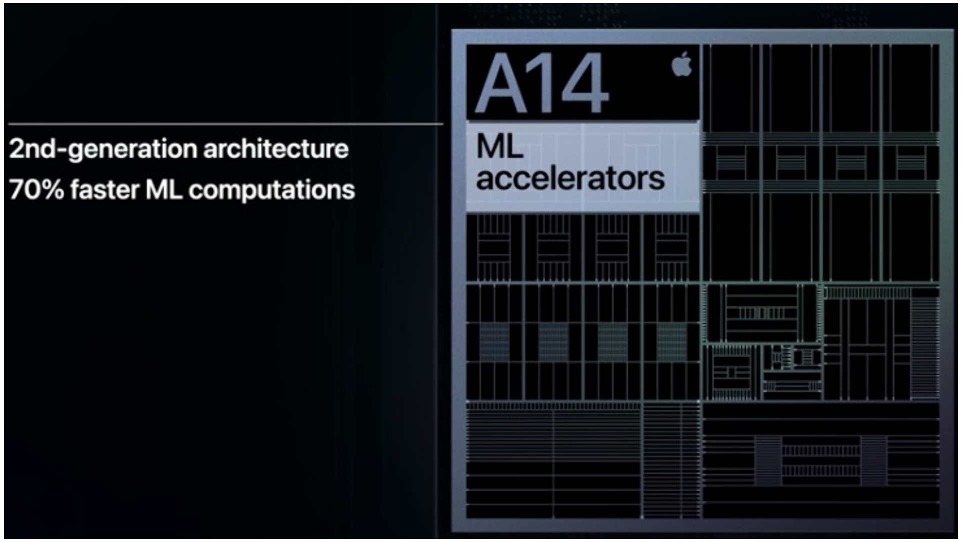

2 high-performance cores
4 high-efficiency cores
Next-generation architecture


4 cores
Next-generation architecture
Improved memory compression

Apple A14

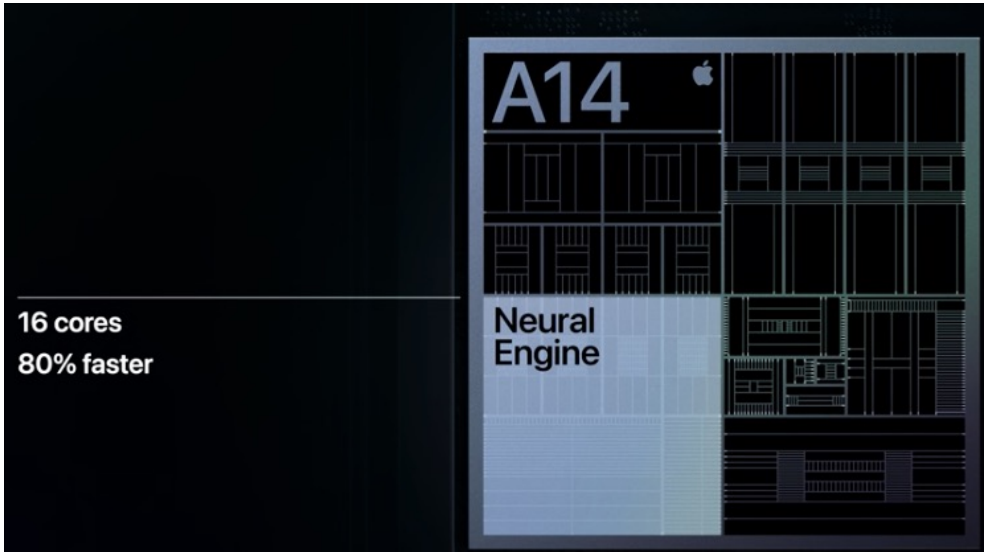
October 13, 2020

A14



2nd-generation architecture

70% faster ML computations

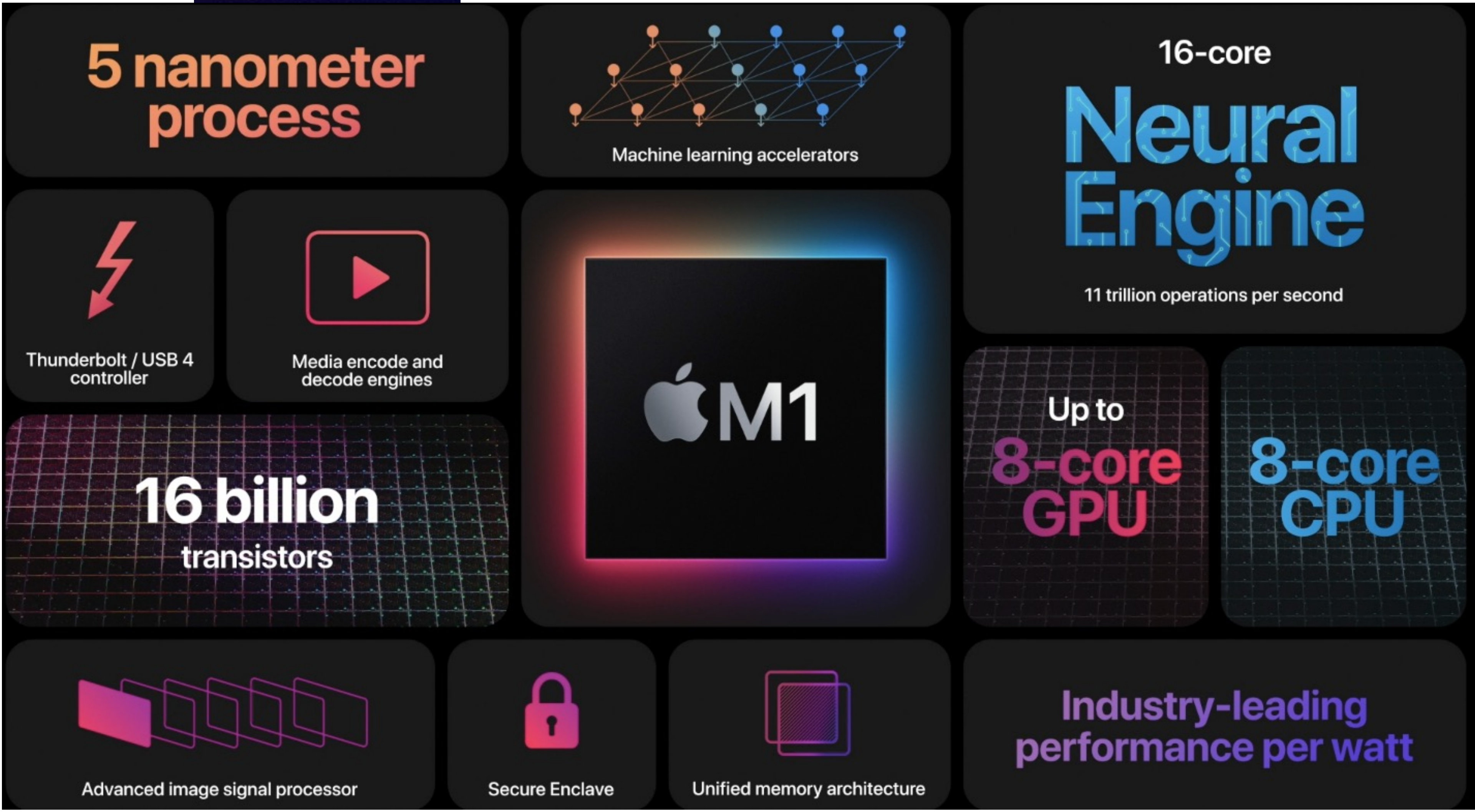


16 cores

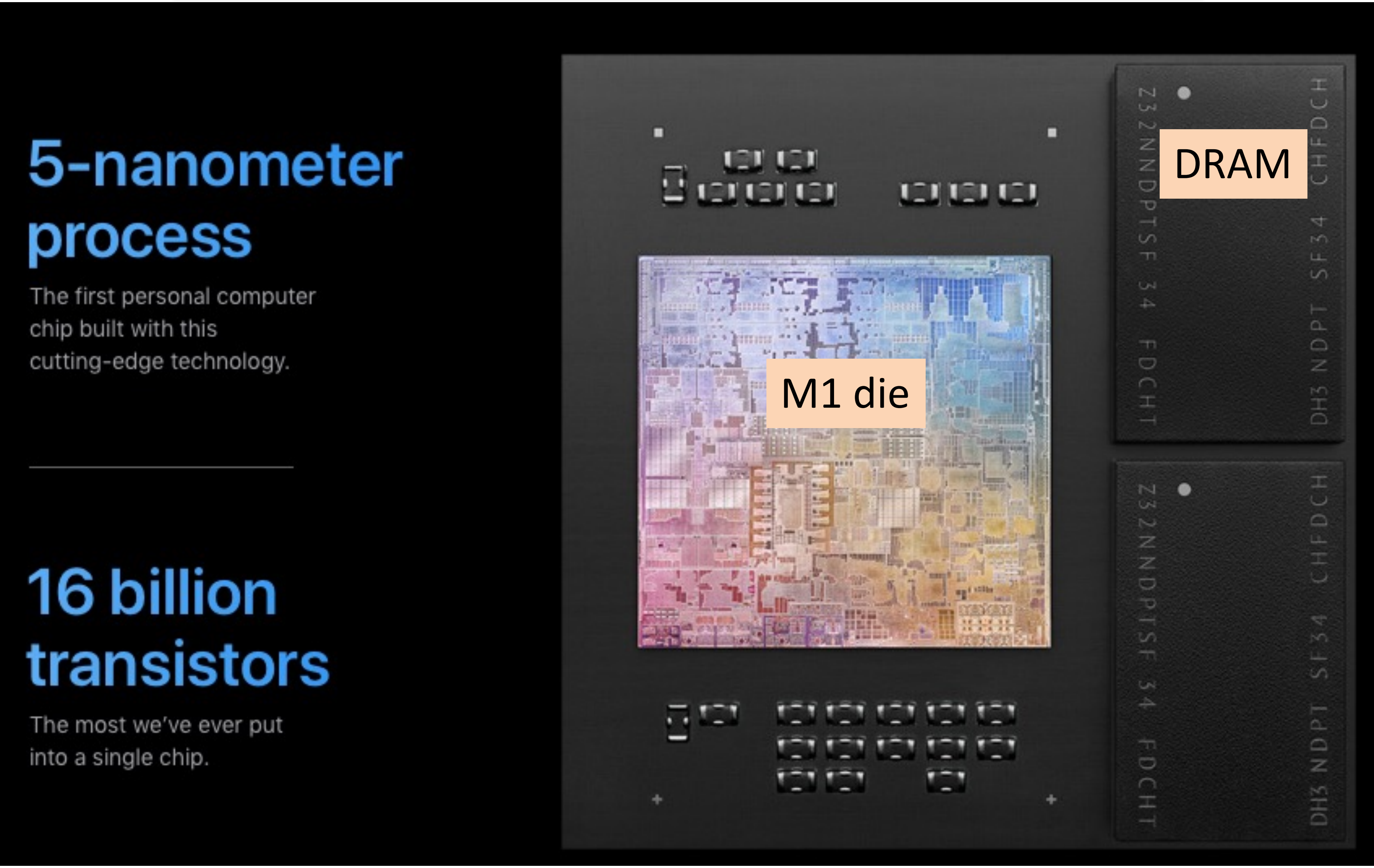
80% faster

Apple Event

November 10, 2020


5 nanometer process
Machine learning accelerators
16-core
Neural Engine
11 trillion operations per second
Thunderbolt / USB 4 controller
Media encode and decode engines
M1
16 billion transistors
Up to 8-core GPU
8-core CPU
Advanced image signal processor
Secure Enclave
Unified memory architecture
Industry-leading performance per watt

Apple M1



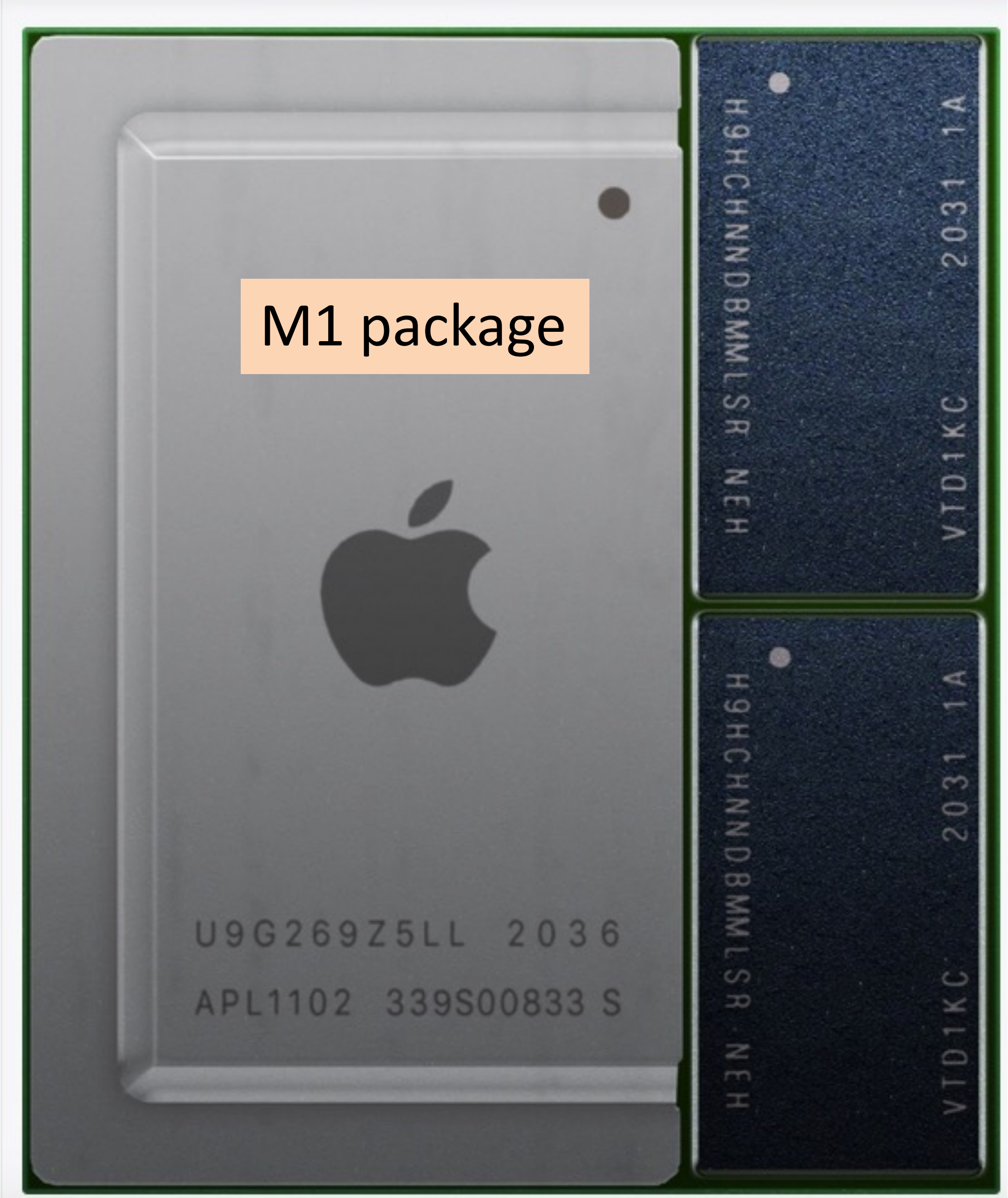
5-nanometer process

The first personal computer chip built with this cutting-edge technology.

16 billion transistors

The most we've ever put into a single chip.

Apple M1 Module



M1 package

Apple M1

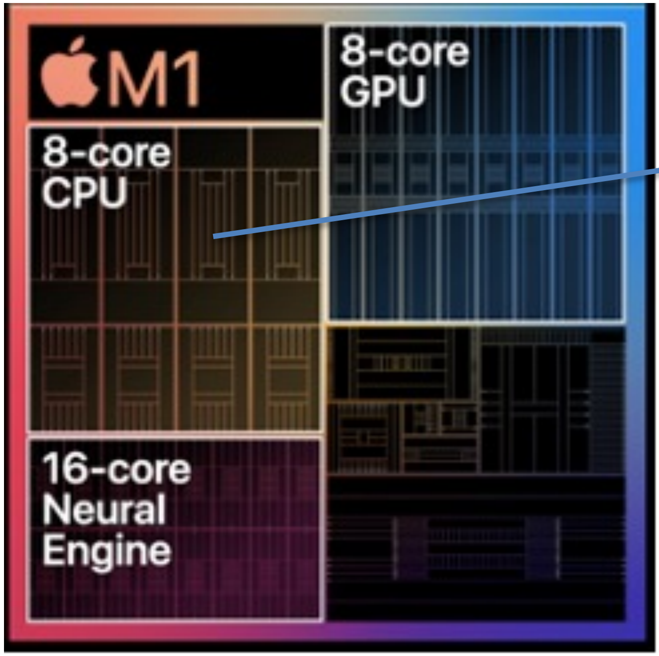
November 10, 2020

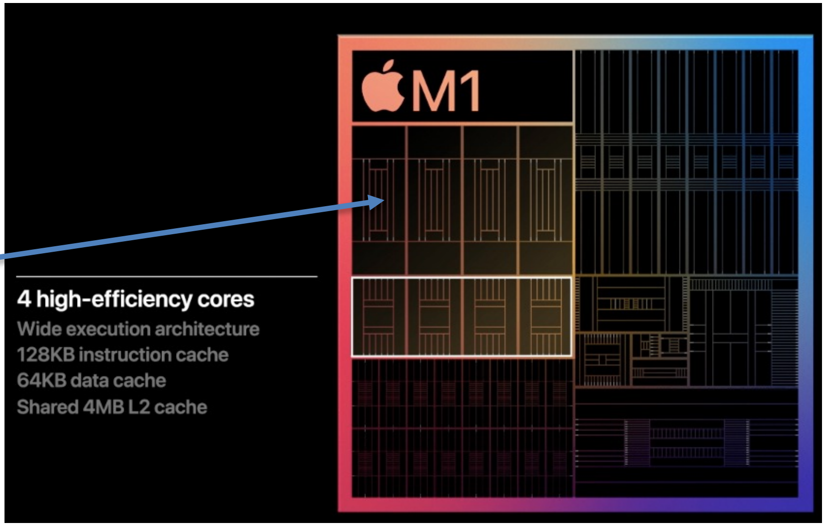
11 trillion
Operations per second

11 Tera FLOPS

❖ CPU cores
- ❑ 4 Hi Perf (20W)
- ❑ 4 Hi Efficiency
  (1.3W low power)

❖ Cores
- ❑ 8 CPU
- ❑ 8 GPU
- ❑ 16 NPU





4 high-efficiency cores
Wide execution architecture
128KB instruction cache
64KB data cache
Shared 4MB L2 cache

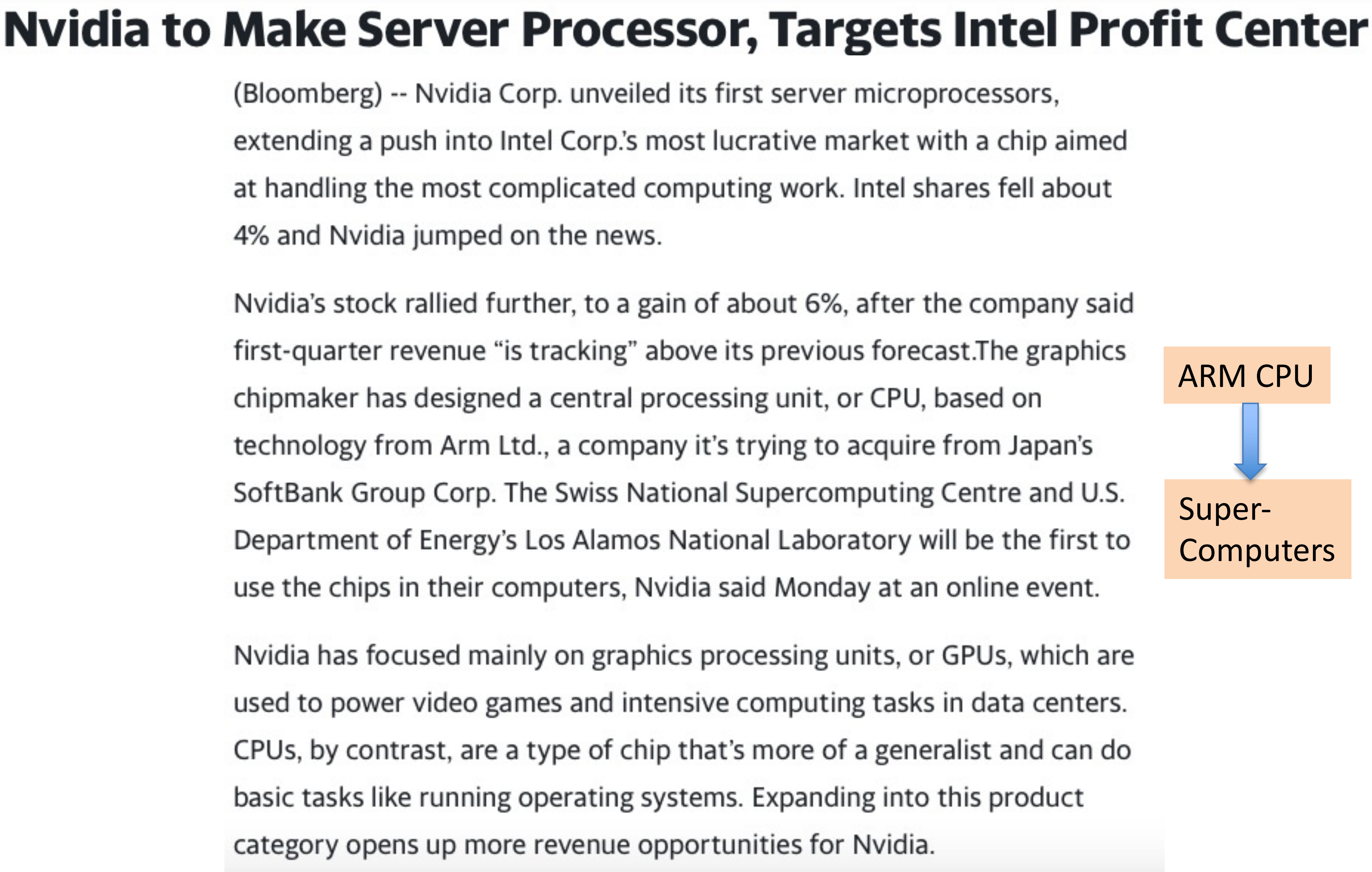
Other CPU Chips

Nvidia

See separate slide set: ***SoC***

Nvidia + ARM = HPC

Apr 9, 2021



Nvidia to Make Server Processor, Targets Intel Profit Center

(Bloomberg) -- Nvidia Corp. unveiled its first server microprocessors, extending a push into Intel Corp.'s most lucrative market with a chip aimed at handling the most complicated computing work. Intel shares fell about 4% and Nvidia jumped on the news.

Nvidia's stock rallied further, to a gain of about 6%, after the company said first-quarter revenue "is tracking" above its previous forecast.The graphics chipmaker has designed a central processing unit, or CPU, based on technology from Arm Ltd., a company it's trying to acquire from Japan's SoftBank Group Corp. The Swiss National Supercomputing Centre and U.S. Department of Energy's Los Alamos National Laboratory will be the first to use the chips in their computers, Nvidia said Monday at an online event.

Nvidia has focused mainly on graphics processing units, or GPUs, which are used to power video games and intensive computing tasks in data centers. CPUs, by contrast, are a type of chip that's more of a generalist and can do basic tasks like running operating systems. Expanding into this product category opens up more revenue opportunities for Nvidia.

ARM CPU

↓

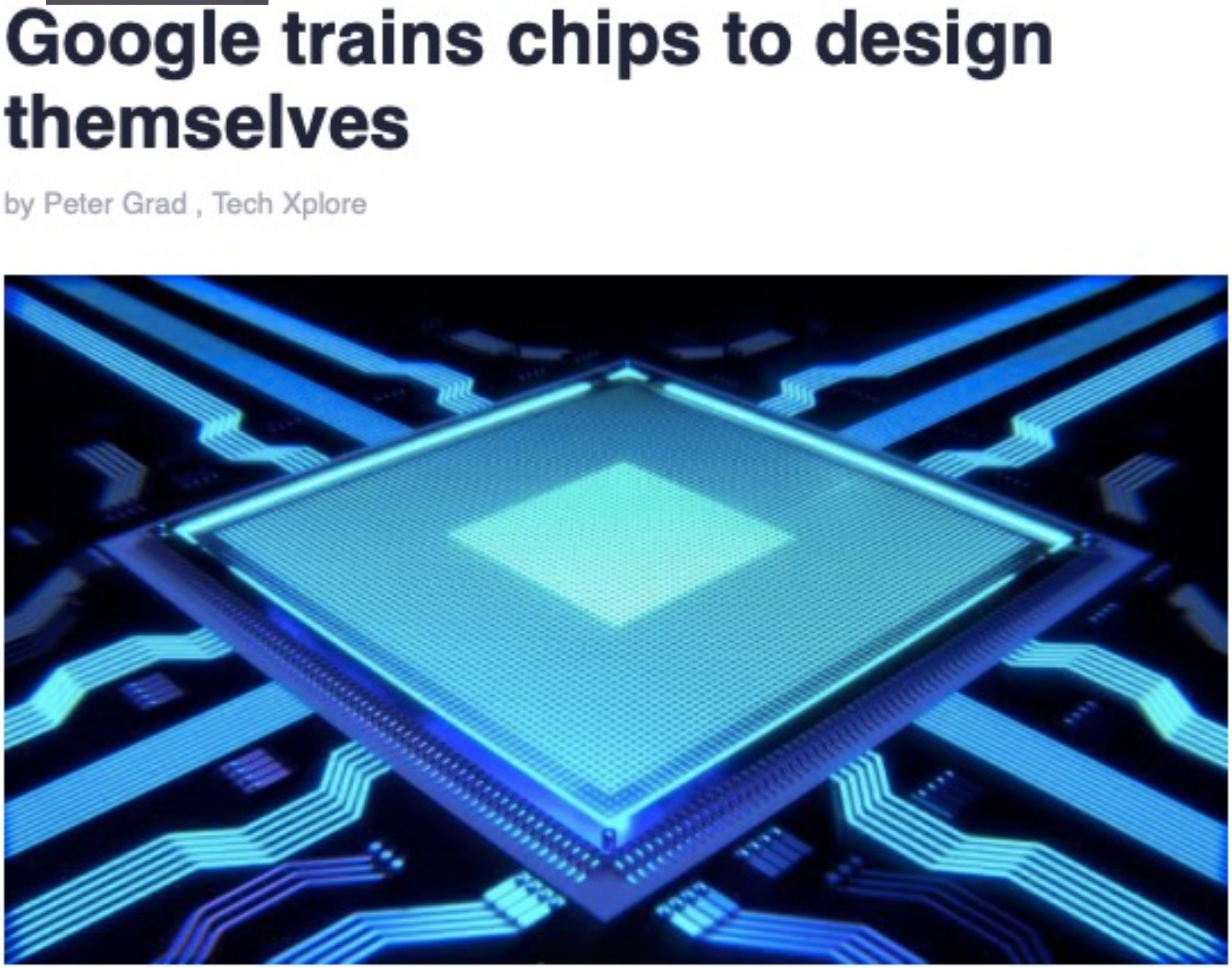
Super-Computers

Other CPU Chips

Google

See separate slide set: ***SoC***

Google Chips



Google trains chips to design themselves

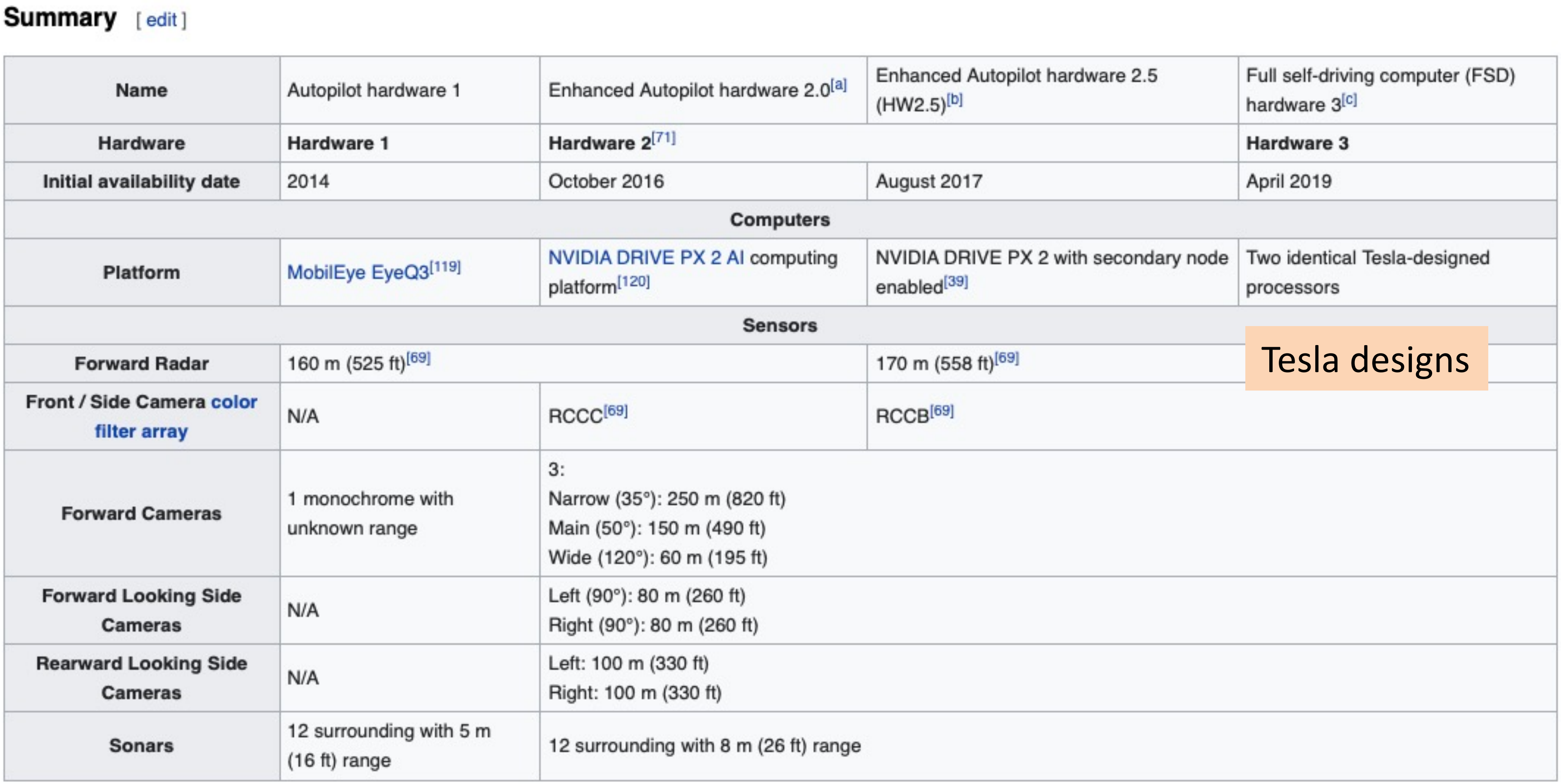
by Peter Grad , Tech Xplore

Other CPU Chips

Tesla

See separate slide set: ***SoC***

Tesla Computers



Summary [edit]

Name	Autopilot hardware 1	Enhanced Autopilot hardware 2.0[a]	Enhanced Autopilot hardware 2.5 (HW2.5)[b]	Full self-driving computer (FSD) hardware 3[c]
Hardware	**Hardware 1**	**Hardware 2**[71]		**Hardware 3**
Initial availability date	2014	October 2016	August 2017	April 2019
Computers				
Platform	MobilEye EyeQ3[119]	NVIDIA DRIVE PX 2 AI computing platform[120]	NVIDIA DRIVE PX 2 with secondary node enabled[39]	Two identical Tesla-designed processors
Sensors				
Forward Radar	160 m (525 ft)[69]		170 m (558 ft)[69]	
Front / Side Camera color filter array	N/A	RCCC[69]	RCCB[69]	
Forward Cameras	1 monochrome with unknown range	3: Narrow (35°): 250 m (820 ft) Main (50°): 150 m (490 ft) Wide (120°): 60 m (195 ft)		
Forward Looking Side Cameras	N/A	Left (90°): 80 m (260 ft) Right (90°): 80 m (260 ft)		
Rearward Looking Side Cameras	N/A	Left: 100 m (330 ft) Right: 100 m (330 ft)		
Sonars	12 surrounding with 5 m (16 ft) range	12 surrounding with 8 m (26 ft) range		

Tesla designs

Organization

Organization – ISA

INSTRUCTION SET ARCHITECTURE

Instruction Set

- ALU
- Branch/Jump (Loop)
- Call/Return (+RFI)
- Control bits/flags
- I/O ports
- Special (NOP)

Memory Addressing Modes

- Direct (Immediate)
- Register Indirect (pointer)
- Base (reg) + offset (immed)

Memory – I/O

REGISTER model

- General (GR)
- Dedicated

MEMORY model

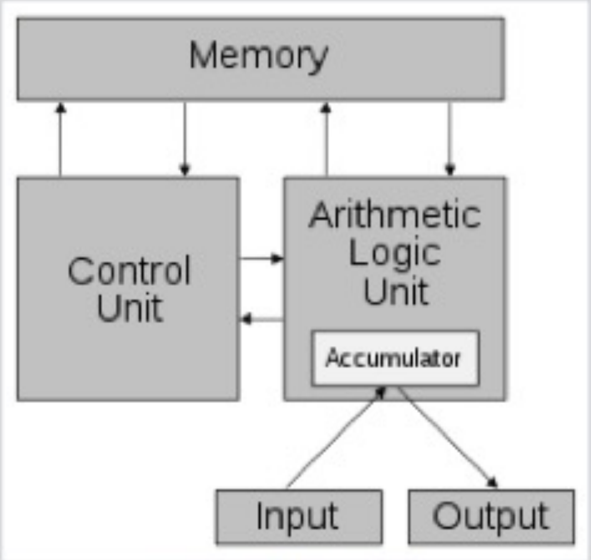
- Direct vs. Virtual (TLB)
- Segmented/paged
- Open vs. dedicated (reserved)
- Cache (I/D, L2/L3)

I/O model

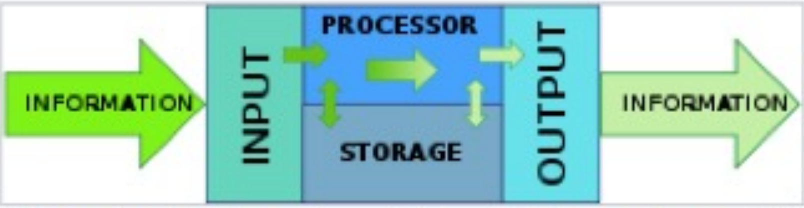
- Memory
  - Dedicated (IN/OUT)
  - **Memory mapped**
- Handling
  - Polling
  - **Interrupt**

1st Gen Architecture

von Neumann



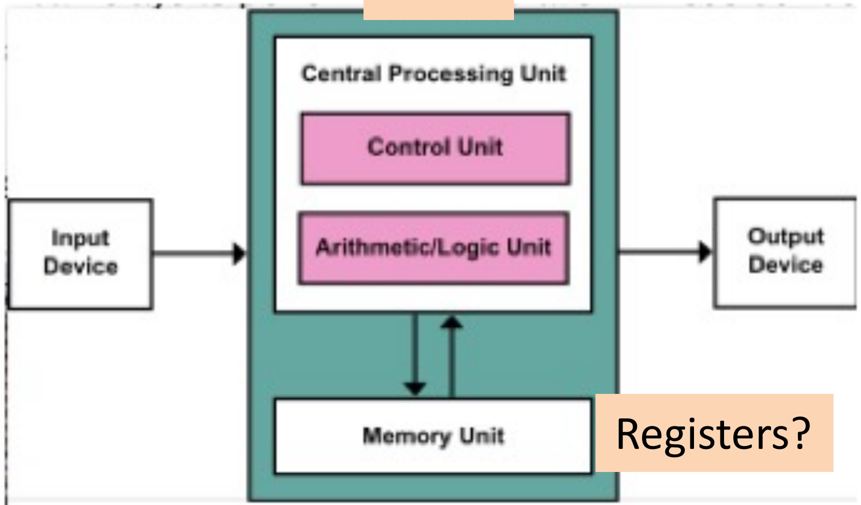
Design of the von Neumann architecture, 1947



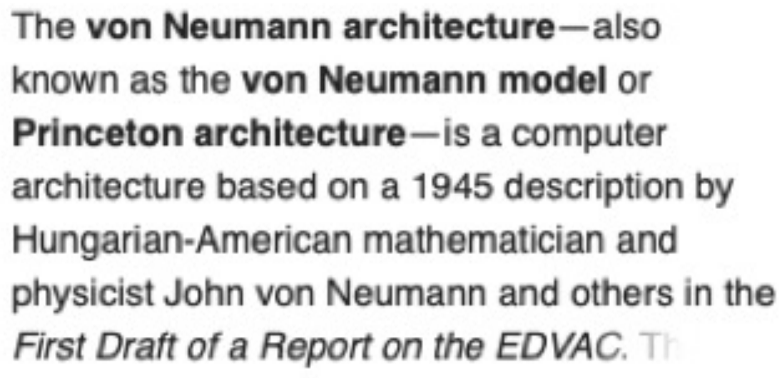
Computing hardware is a platform for information processing.

I-P-O

I-P-O

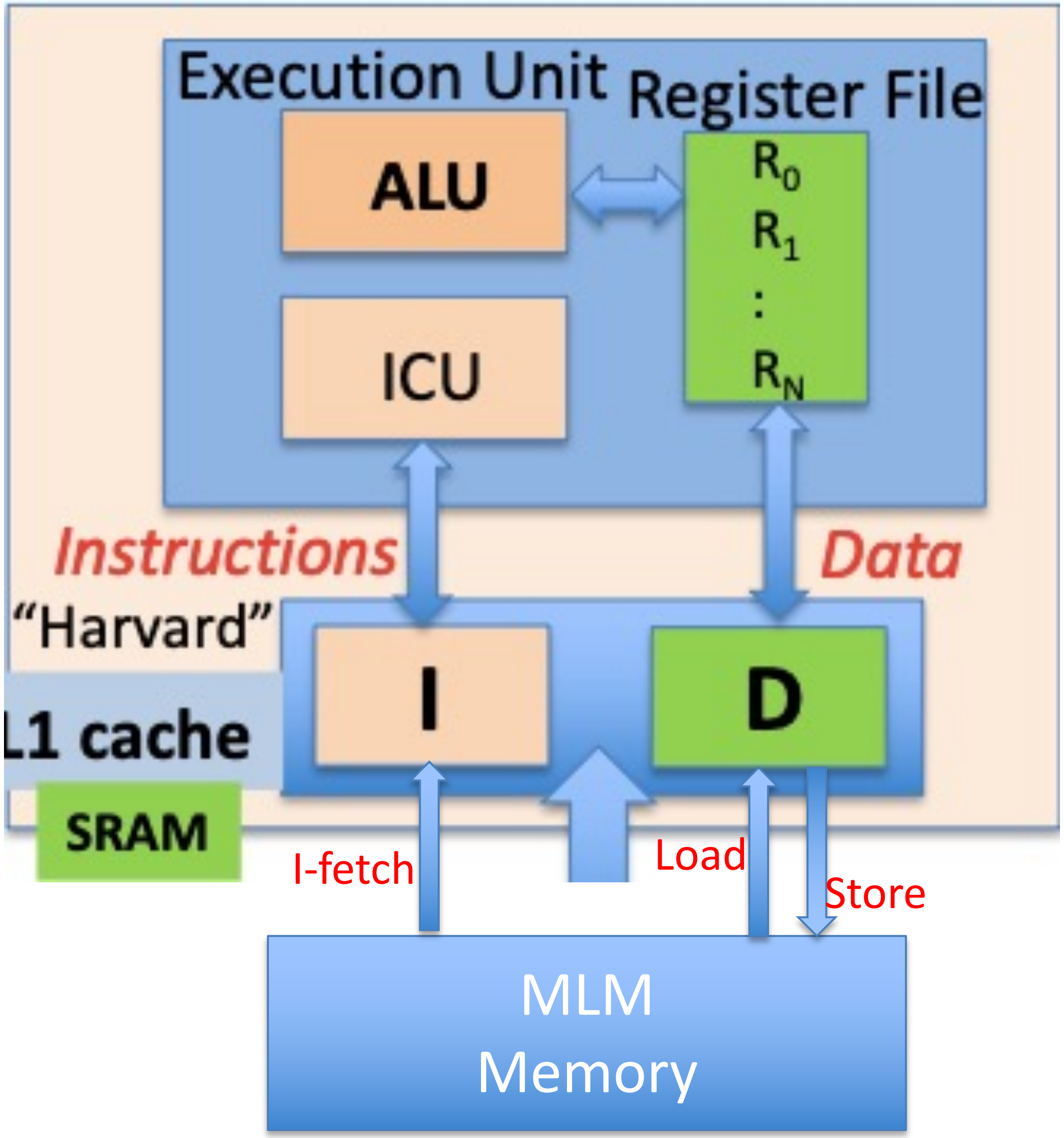


Registers?



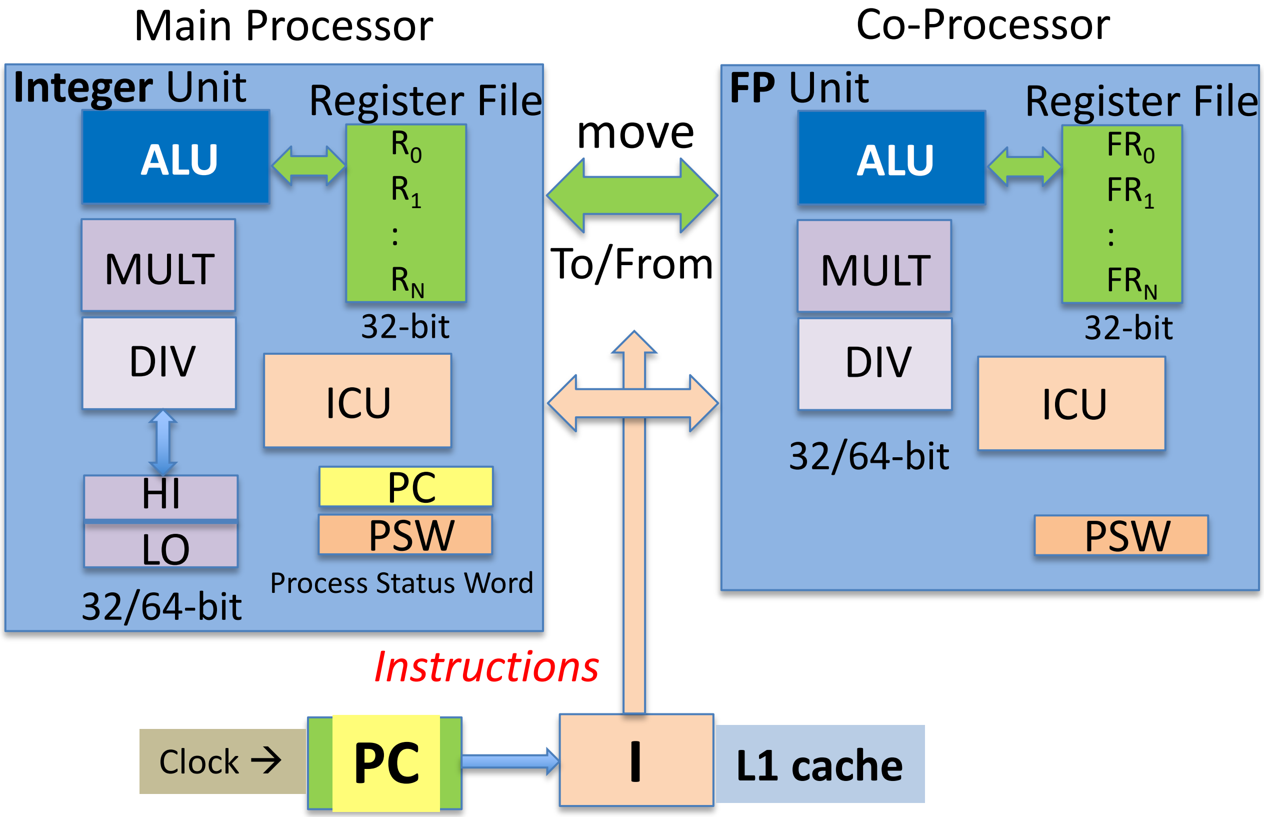
The **von Neumann architecture**—also known as the **von Neumann model** or **Princeton architecture**—is a computer architecture based on a 1945 description by Hungarian-American mathematician and physicist John von Neumann and others in the *First Draft of a Report on the EDVAC*. Th

Integer Execution Unit (EU)



Integer Unit + FPU

CPU = Integer Unit + FPU



CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE

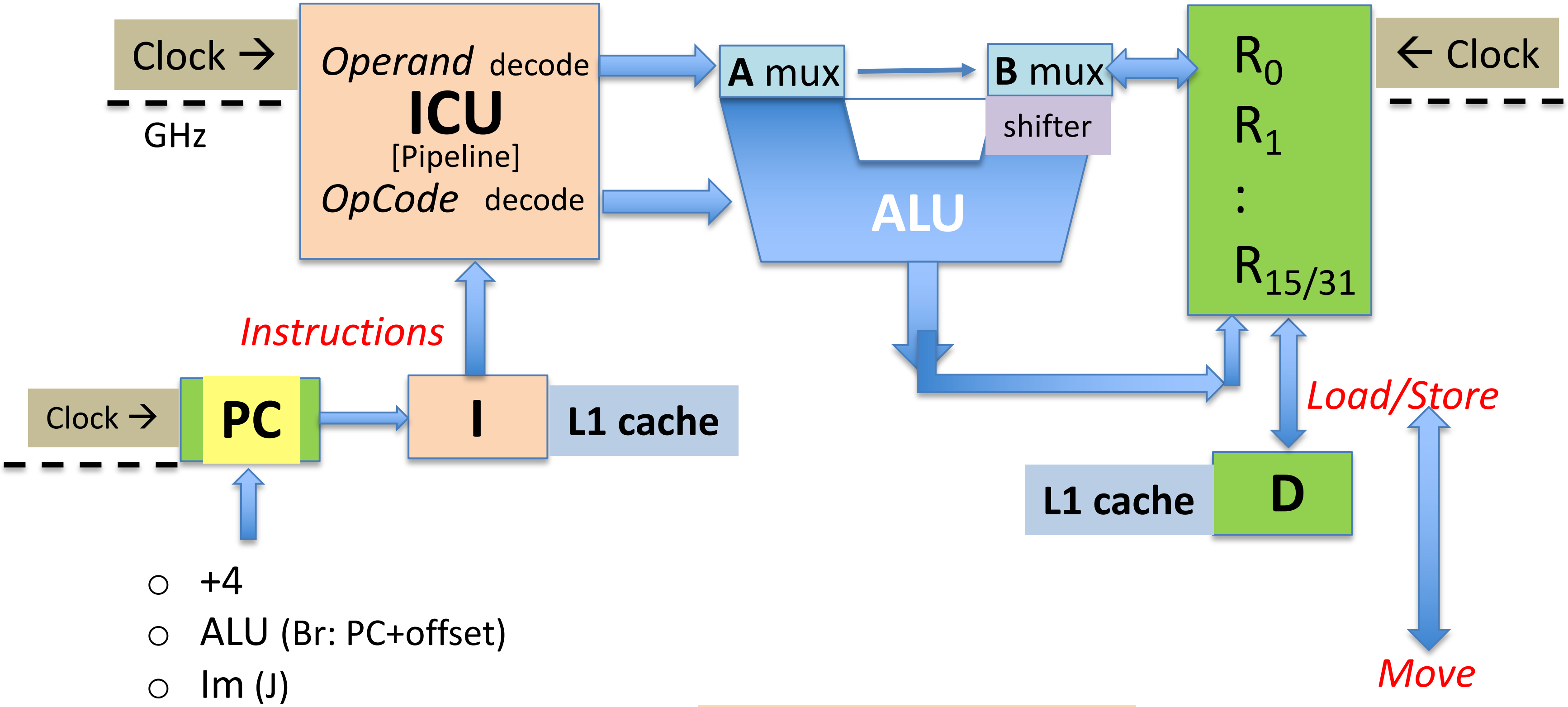
COMP122

Complete CPU Org

DR JEFF SOFTWARE
INDIE APP DEVELOPER
© Jeff Drobman
2016-2022

CPU: Integer ***Execution Unit***

Clock → *synchronized*

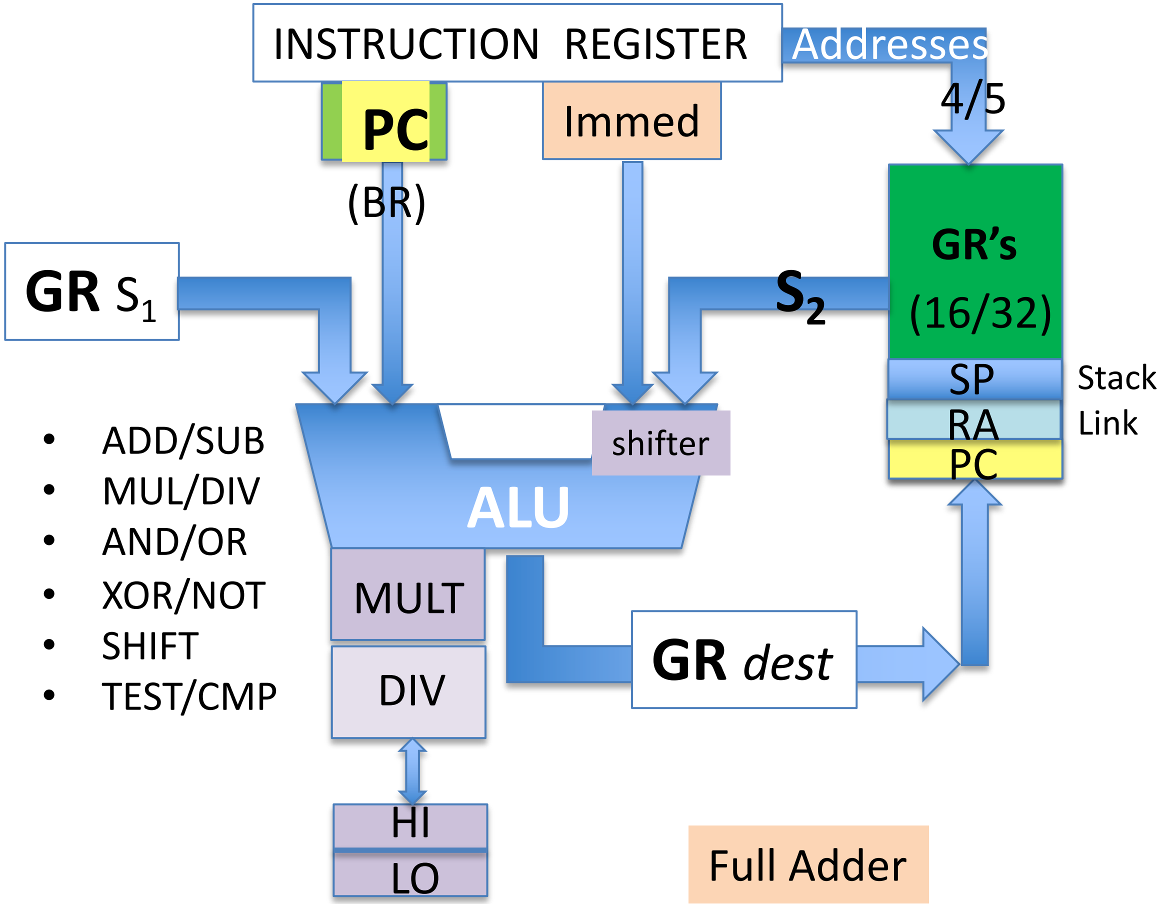


- ○ +4
- ○ ALU (Br: PC+offset)
- ○ Im (J)

❖ **FPU**

- ❑ ALU
- ❑ Mult/Div/Sqrt
- ❑ **FR's** (32)

CPU: ALU Ops/Operands



- ADD/SUB
- MUL/DIV
- AND/OR
- XOR/NOT
- SHIFT
- TEST/CMP

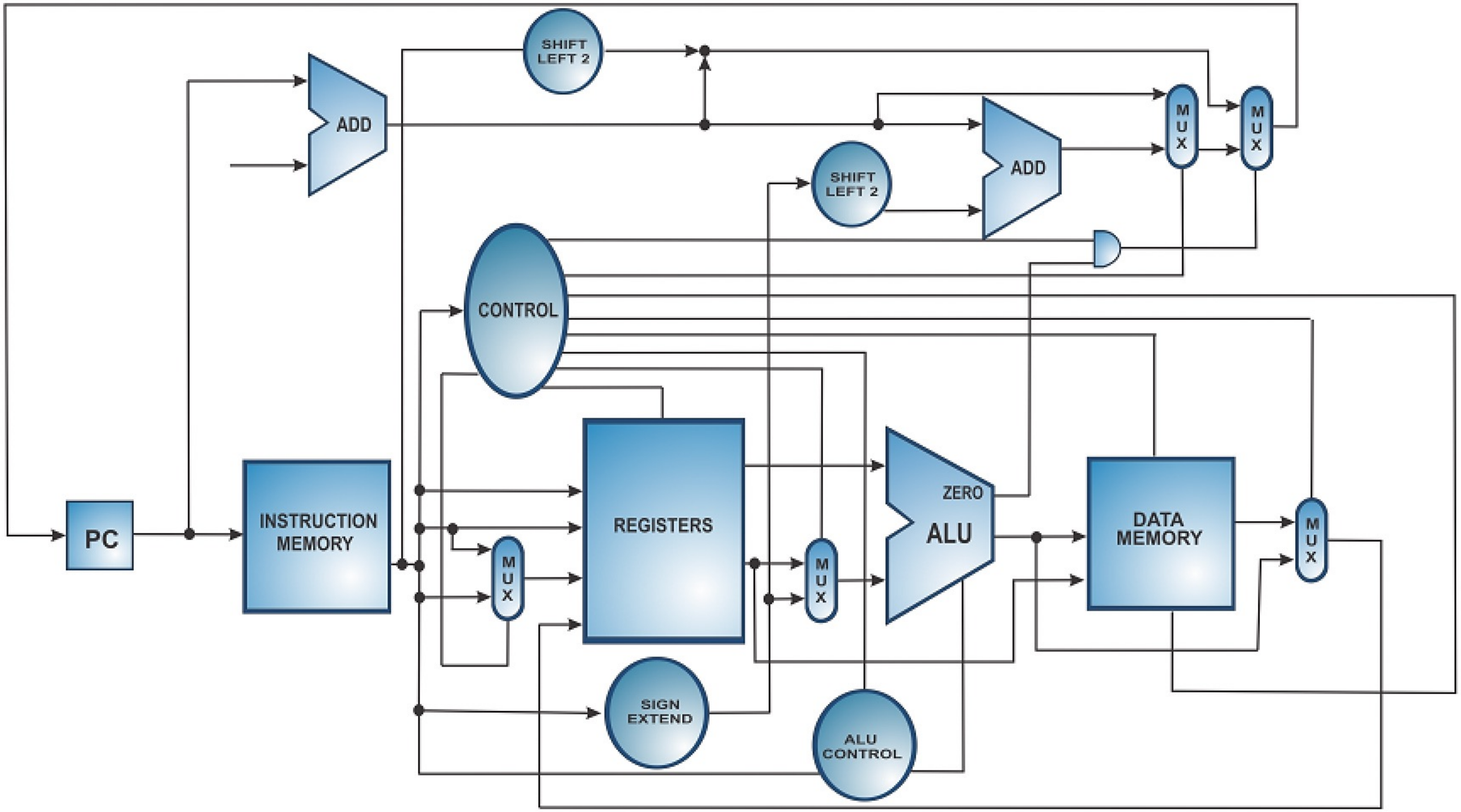
Full Adder

Si = Ai + Bi + Cin
Cout = [any 2]

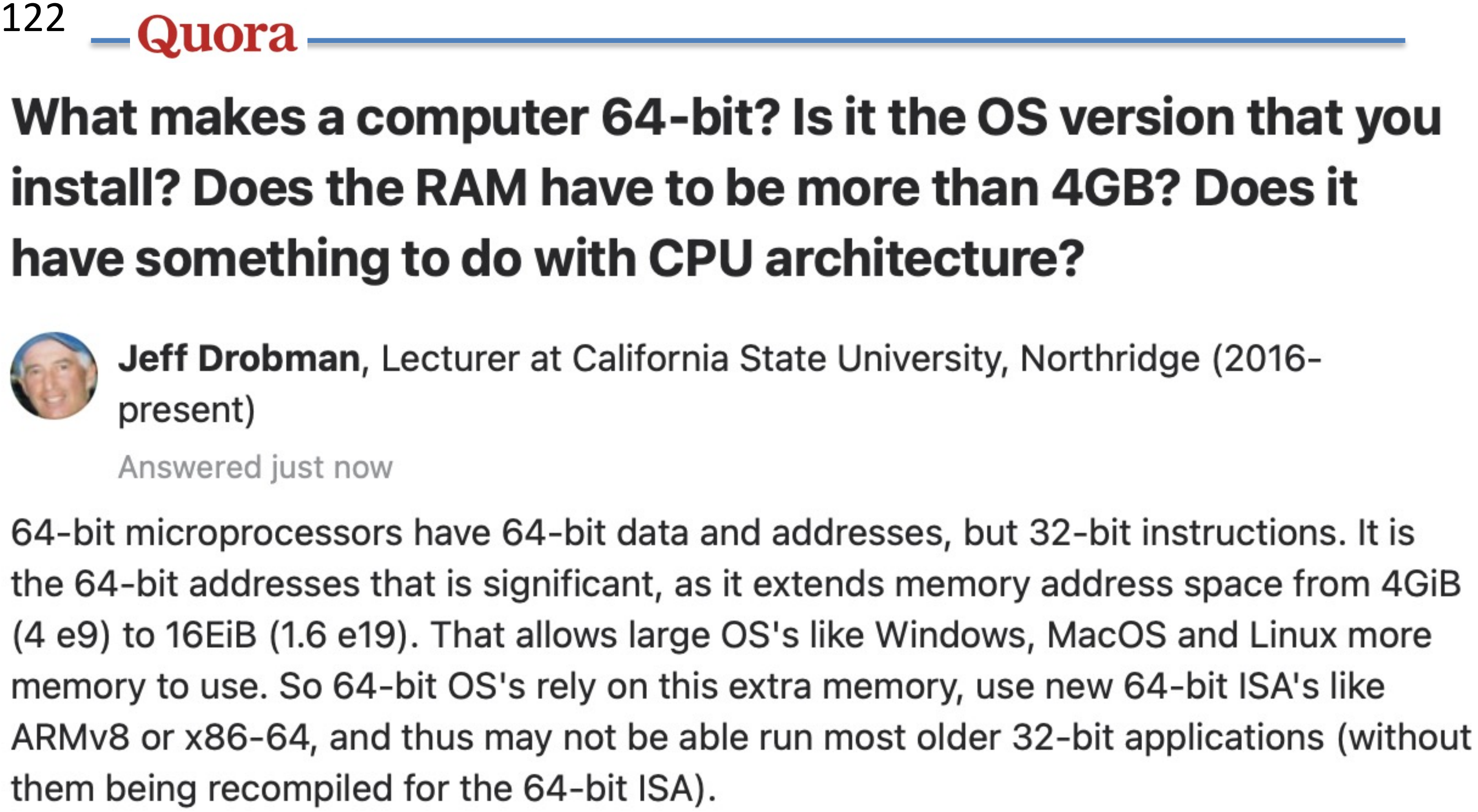
CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE
COMP122

MIPS MARS Xray

DR JEFF SOFTWARE
INDIE APP DEVELOPER
© Jeff Drobman
2016-2022



64-Bit CPU



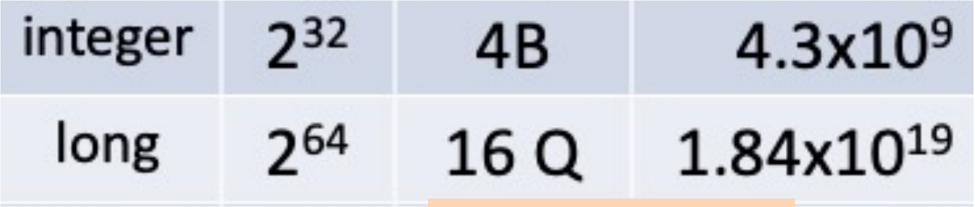
Quora

What makes a computer 64-bit? Is it the OS version that you install? Does the RAM have to be more than 4GB? Does it have something to do with CPU architecture?

Jeff Drobman, Lecturer at California State University, Northridge (2016-present)

Answered just now

64-bit microprocessors have 64-bit data and addresses, but 32-bit instructions. It is the 64-bit addresses that is significant, as it extends memory address space from 4GiB (4 e9) to 16EiB (1.6 e19). That allows large OS's like Windows, MacOS and Linux more memory to use. So 64-bit OS's rely on this extra memory, use new 64-bit ISA's like ARMv8 or x86-64, and thus may not be able run most older 32-bit applications (without them being recompiled for the 64-bit ISA).



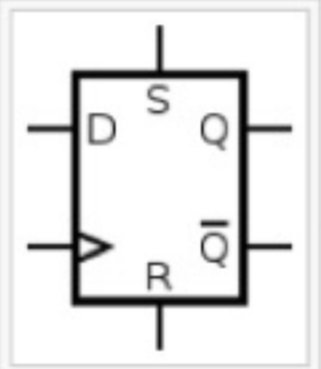
integer	2^{32}	4B	$4.3\text{x}10^{9}$
long	2^{64}	16 Q	$1.84\text{x}10^{19}$

16 EiB=18.4 EB

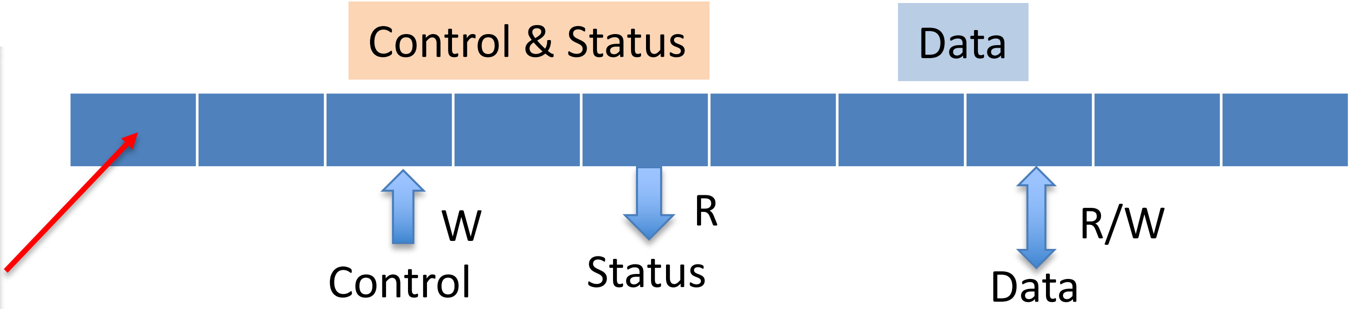
Computer Architecture

Registers

Registers



D flip-flop symbol



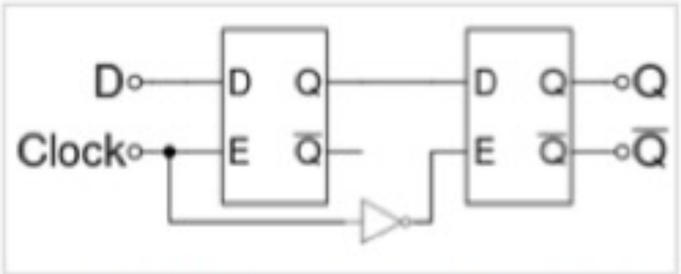
Control & Status

Data

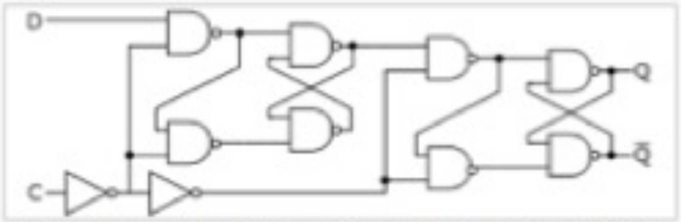
W
Control

R
Status

R/W
Data

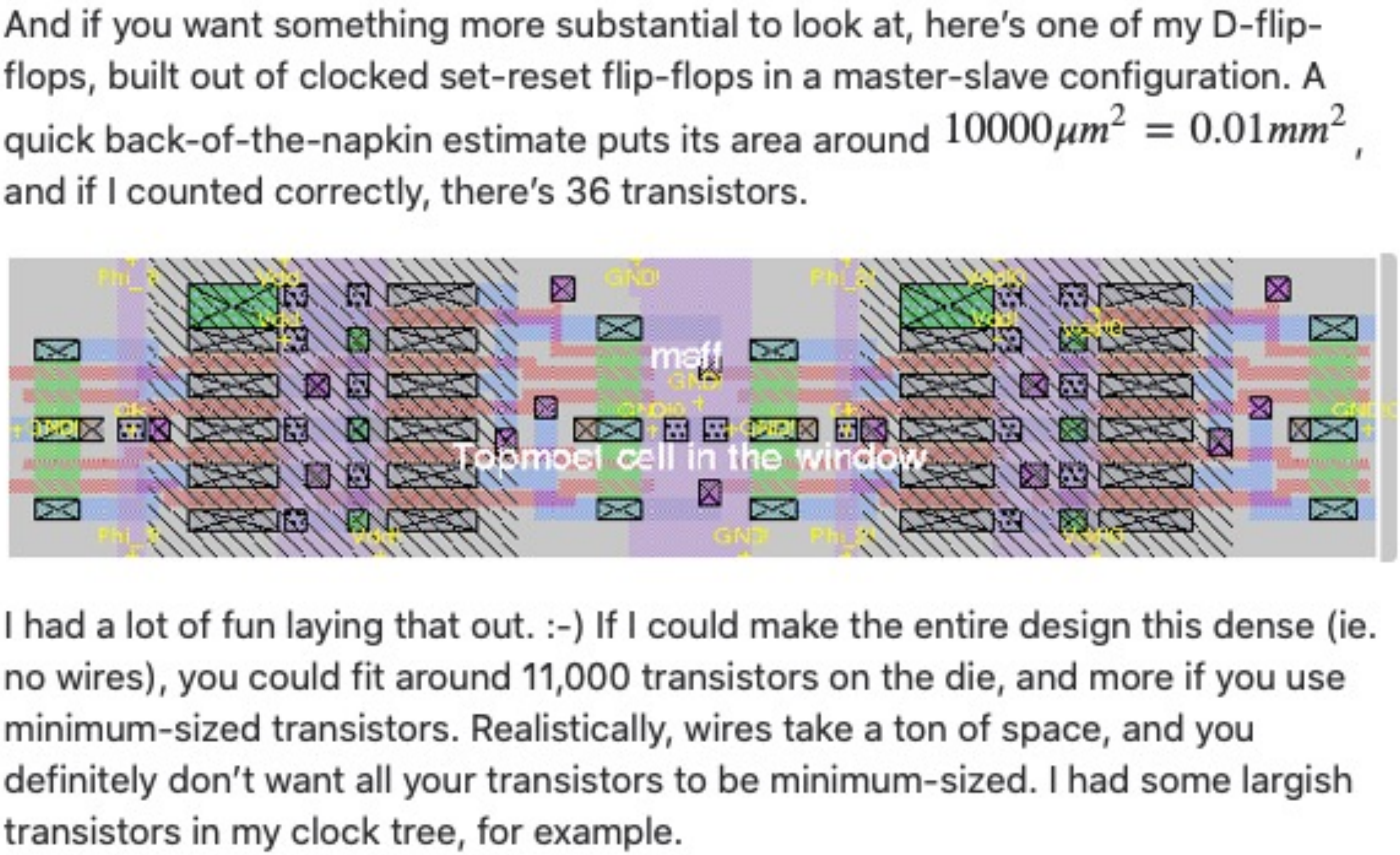


A master–slave D flip-flop. It responds on the falling edge of the *enable* input (usually a clock)



An implementation of a master–slave D flip-flop that is triggered on the rising edge of the clock

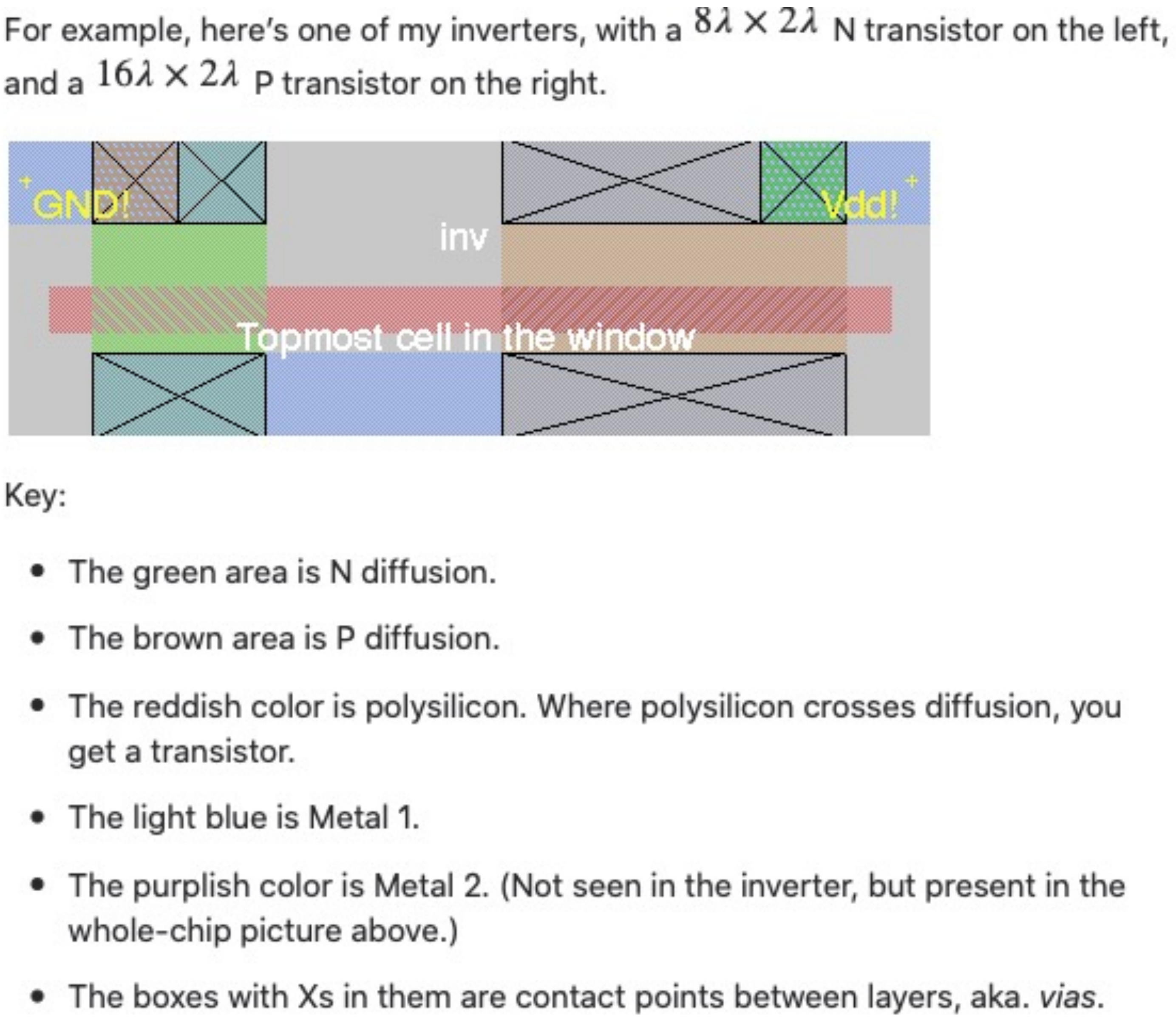
D Flip-Flops



And if you want something more substantial to look at, here's one of my D-flip-flops, built out of clocked set-reset flip-flops in a master-slave configuration. A quick back-of-the-napkin estimate puts its area around $10000\mu m^2 = 0.01mm^2$, and if I counted correctly, there's 36 transistors.

I had a lot of fun laying that out. :-) If I could make the entire design this dense (ie. no wires), you could fit around 11,000 transistors on the die, and more if you use minimum-sized transistors. Realistically, wires take a ton of space, and you definitely don't want all your transistors to be minimum-sized. I had some largish transistors in my clock tree, for example.

D Flip-Flops: Inverter



For example, here's one of my inverters, with a $8\lambda \times 2\lambda$ N transistor on the left, and a $16\lambda \times 2\lambda$ P transistor on the right.

Key:

- The green area is N diffusion.
- The brown area is P diffusion.
- The reddish color is polysilicon. Where polysilicon crosses diffusion, you get a transistor.
- The light blue is Metal 1.
- The purplish color is Metal 2. (Not seen in the inverter, but present in the whole-chip picture above.)
- The boxes with Xs in them are contact points between layers, aka. *vias*.

CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE
COMP122

Register Arch/Org

DR JEFF SOFTWARE
INDIE APP DEVELOPER
© Jeff Drobman
2016-2022

Hennessy & Patterson

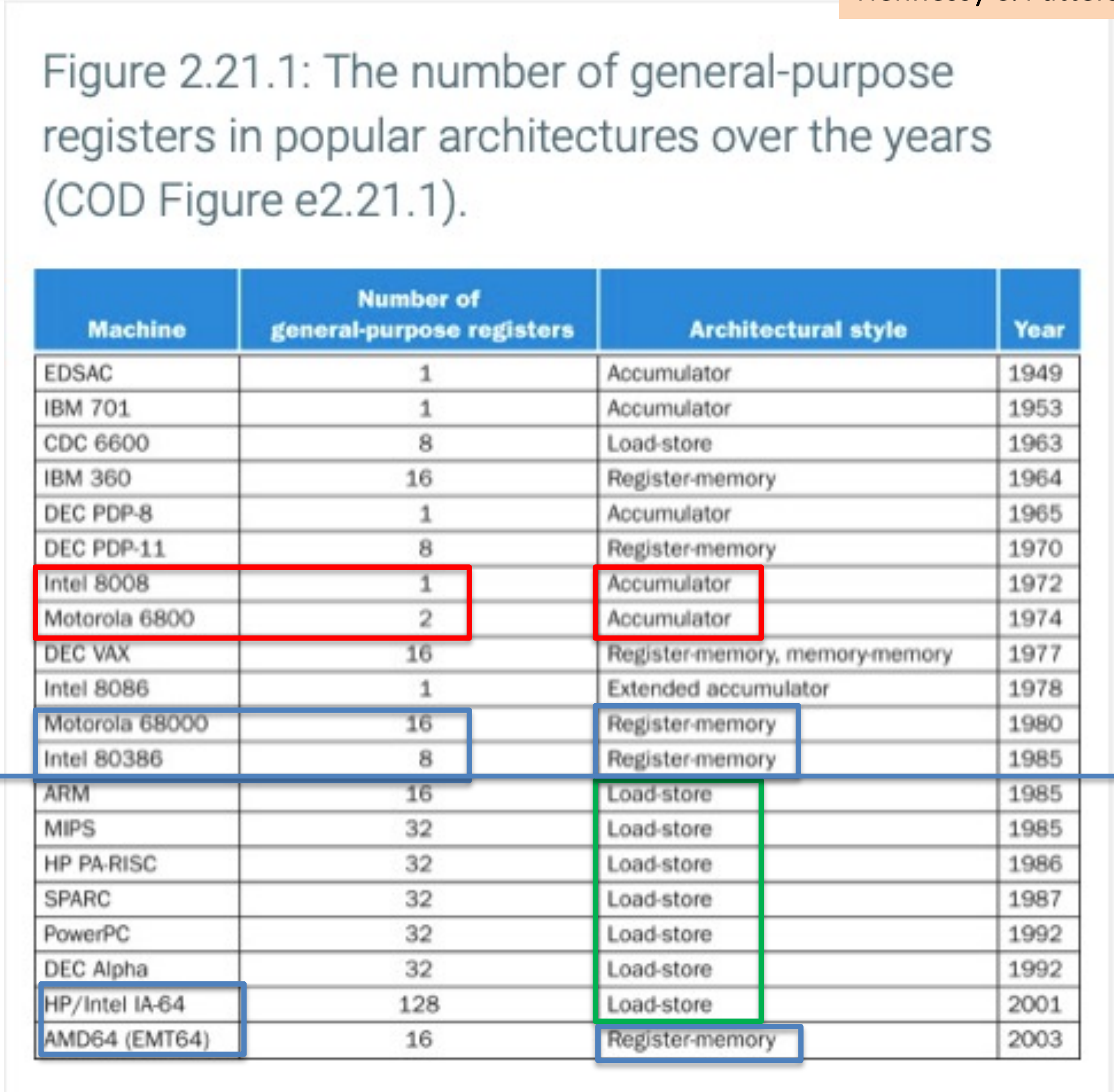


Figure 2.21.1: The number of general-purpose registers in popular architectures over the years (COD Figure e2.21.1).

Machine	Number of general-purpose registers	Architectural style	Year
EDSAC	1	Accumulator	1949
IBM 701	1	Accumulator	1953
CDC 6600	8	Load-store	1963
IBM 360	16	Register-memory	1964
DEC PDP-8	1	Accumulator	1965
DEC PDP-11	8	Register-memory	1970
Intel 8008	1	Accumulator	1972
Motorola 6800	2	Accumulator	1974
DEC VAX	16	Register-memory, memory-memory	1977
Intel 8086	1	Extended accumulator	1978
Motorola 68000	16	Register-memory	1980
Intel 80386	8	Register-memory	1985
ARM	16	Load-store	1985
MIPS	32	Load-store	1985
HP PA-RISC	32	Load-store	1986
SPARC	32	Load-store	1987
PowerPC	32	Load-store	1992
DEC Alpha	32	Load-store	1992
HP/Intel IA-64	128	Load-store	2001
AMD64 (EMT64)	16	Register-memory	2003

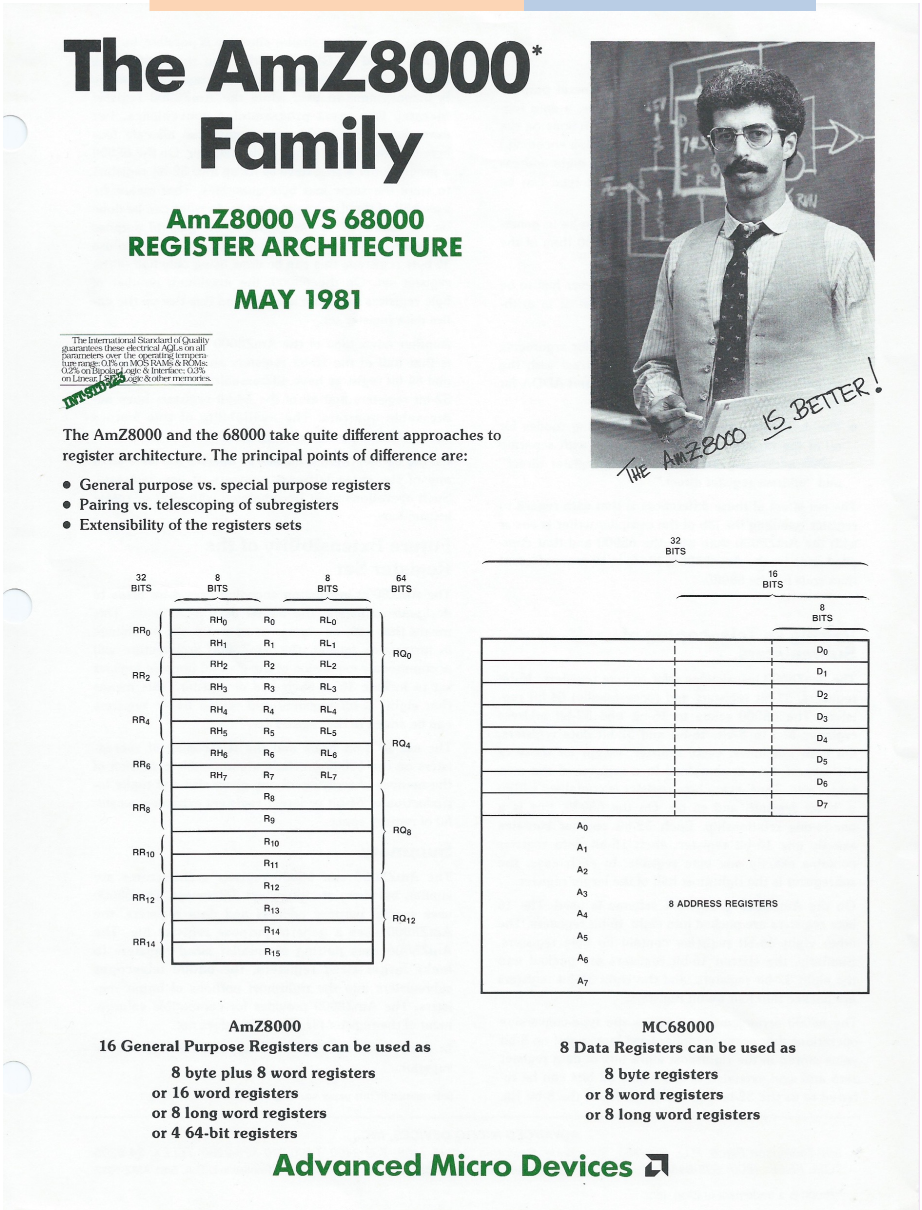
CISC

RISC

CPU ISA's

Z8000 vs. M6800 | 16-bit MPU's

- x86
  - i8088
  - Pentium
    - Intel P, M
    - AMD K5-8
- MIPS
  - R3000/4000
  - MIPS32/64
- ARM
  - Cortex (A, M)
  - ARMv7/8
- RISC-V



The AmZ8000* Family

AmZ8000 VS 68000
REGISTER ARCHITECTURE

MAY 1981

The International Standard of Quality guarantees these electrical AQLs on all parameters over the operating temperature range: 0.1% on MOS RAMs & ROMs: 0.2% on Bipolar Logic & Interface: 0.3% on Linear, LSI Logic & other memories.

INT-STD-123

THE AmZ8000 IS BETTER!

The AmZ8000 and the 68000 take quite different approaches to register architecture. The principal points of difference are:

- General purpose vs. special purpose registers
- Pairing vs. telescoping of subregisters
- Extensibility of the registers sets

AmZ8000
16 General Purpose Registers can be used as

8 byte plus 8 word registers
or 16 word registers
or 8 long word registers
or 4 64-bit registers

MC68000
8 Data Registers can be used as

8 byte registers
or 8 word registers
or 8 long word registers

Advanced Micro Devices

CPU ISA’s

- ❖MIPS
- ❖ARM

16-32 **GR**’s

- ❖ Includes specials
  - ❑ Stack: SP, FP
  - ❑ Globals: GP
  - ❑ Zero
- ❖ 64-bit
  - ❑ *Paired* 32-bit

➢ **PC** dedicated

- ❖x86
  - ❑ Intel
  - ❑ AMD

Dedicated Registers

- ❖ 4 Accumulators
  - ❑ A, B, C, D
- ❖ 4 Pointers
  - ❑ SI, DI
  - ❑ BP, SP
- ❖ 5 Memory Segments
  - ❑ CS, DS, ES, FS, GS, SS
- ❖ 64-bit
  - ❑ *Telescoping*

GP Registers

Register use convention: A

Hennessy & Patterson

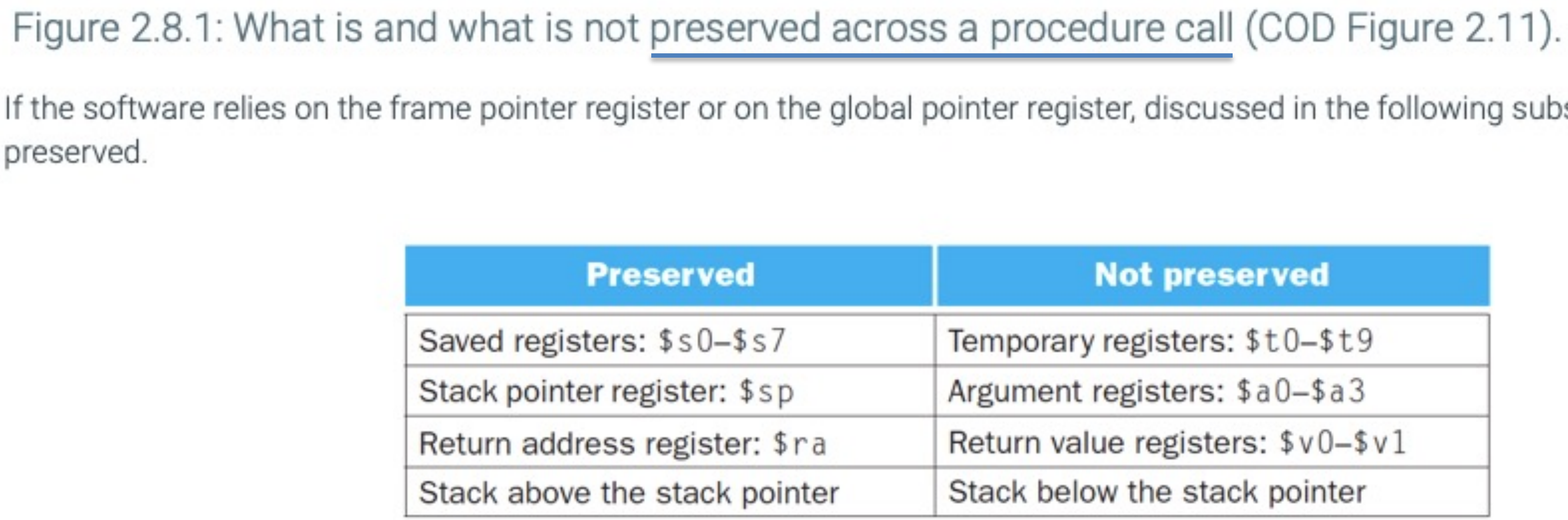


Figure 2.8.1: What is and what is not preserved across a procedure call (COD Figure 2.11).

If the software relies on the frame pointer register or on the global pointer register, discussed in the following subs
preserved.

Preserved	Not preserved
Saved registers: $s0–$s7	Temporary registers: $t0–$t9
Stack pointer register: $sp	Argument registers: $a0–$a3
Return address register: $ra	Return value registers: $v0–$v1
Stack above the stack pointer	Stack below the stack pointer

- $a(0:3) *args*
- $at, $k(0:1) *reserved*
- $v(0:1) *values*
- $t(0-9) *temp*
- $s(0:7) *saved*
- $gp *global ptr*
- $sp *stack ptr*
- $fp *frame ptr*
- $ra *return addr*

CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE

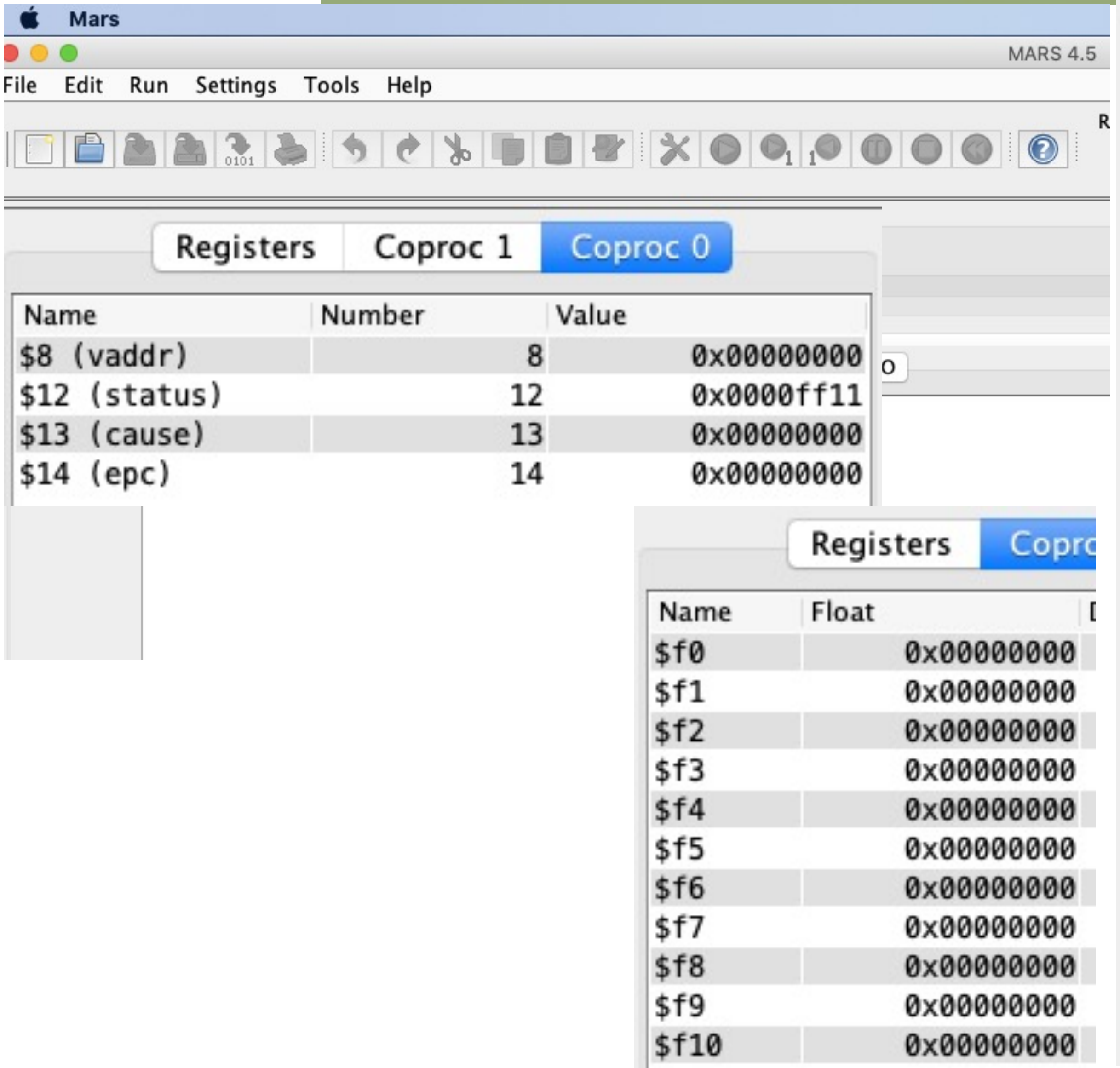
COMP122

MARS

DR JEFF SOFTWARE
INDIE APP DEVELOPER
© Jeff Drobman
2016-2022

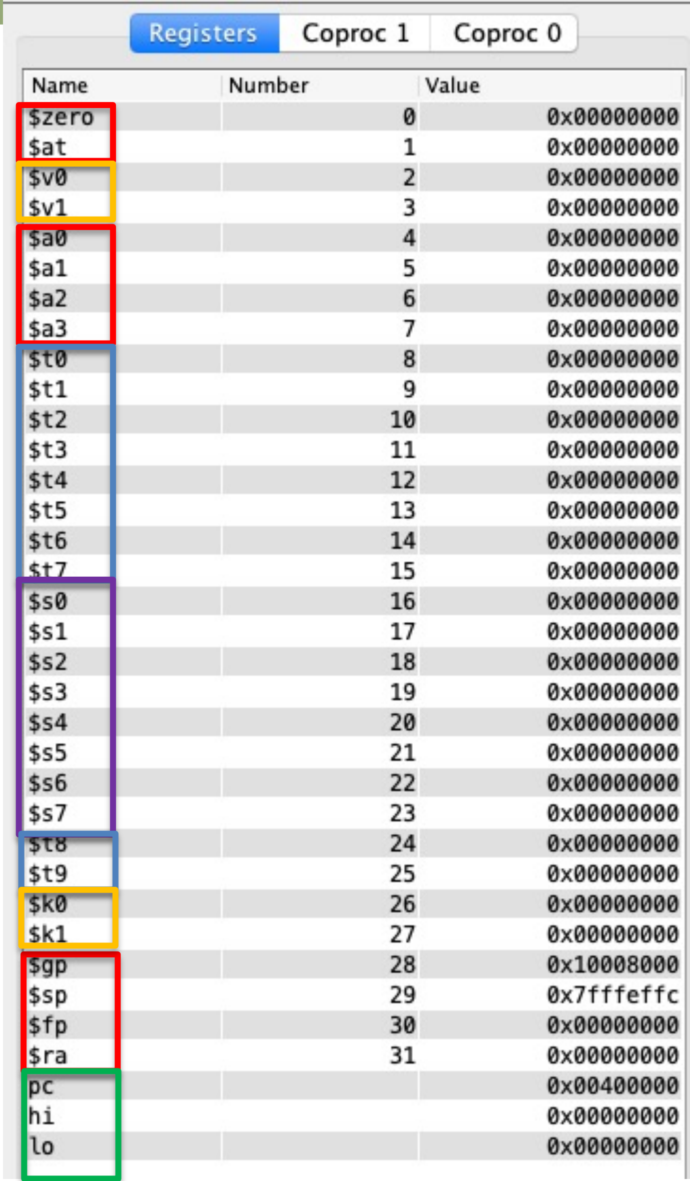
MARS (MIPS Assembler and Runtime Simulator)

Registers



Name	Number	Value
$8 (vaddr)	8	0x00000000
$12 (status)	12	0x0000ff11
$13 (cause)	13	0x00000000
$14 (epc)	14	0x00000000

Name	Float
$f0	0x00000000
$f1	0x00000000
$f2	0x00000000
$f3	0x00000000
$f4	0x00000000
$f5	0x00000000
$f6	0x00000000
$f7	0x00000000
$f8	0x00000000
$f9	0x00000000
$f10	0x00000000

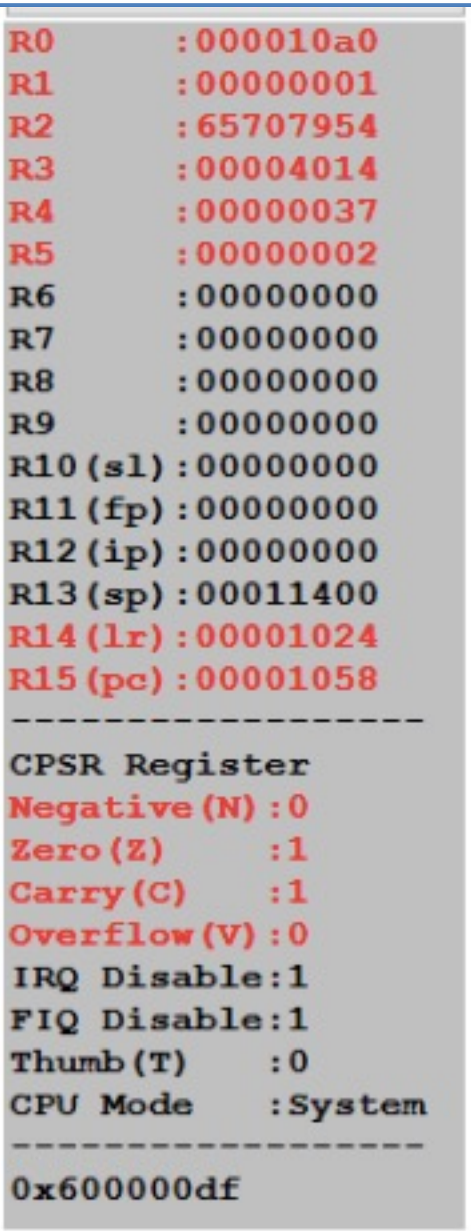


Name	Number	Value
$zero	0	0x00000000
$at	1	0x00000000
$v0	2	0x00000000
$v1	3	0x00000000
$a0	4	0x00000000
$a1	5	0x00000000
$a2	6	0x00000000
$a3	7	0x00000000
$t0	8	0x00000000
$t1	9	0x00000000
$t2	10	0x00000000
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x00000000
$s1	17	0x00000000
$s2	18	0x00000000
$s3	19	0x00000000
$s4	20	0x00000000
$s5	21	0x00000000
$s6	22	0x00000000
$s7	23	0x00000000
$t8	24	0x00000000
$t9	25	0x00000000
$k0	26	0x00000000
$k1	27	0x00000000
$gp	28	0x10008000
$sp	29	0x7fffeffc
$fp	30	0x00000000
$ra	31	0x00000000
pc		0x00400000
hi		0x00000000
lo		0x00000000

ARM(v5/7) Regs – ARMsim

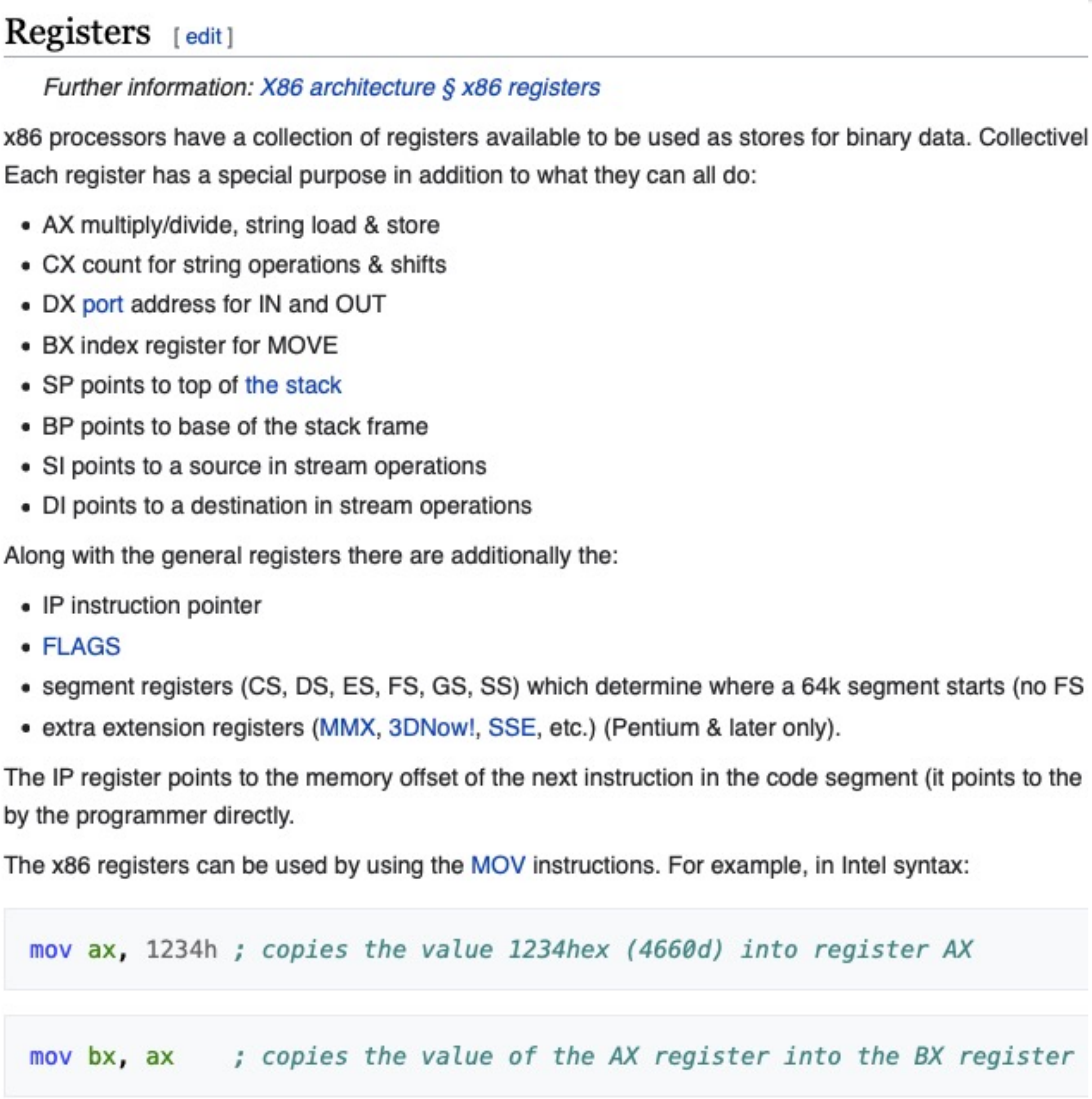
ARMv5 → 16 GR's

ARMv7 → 32 GR's



```
R0       :000010a0
R1       :00000001
R2       :65707954
R3       :00004014
R4       :00000037
R5       :00000002
R6       :00000000
R7       :00000000
R8       :00000000
R9       :00000000
R10(sl)  :00000000
R11(fp)  :00000000
R12(ip)  :00000000
R13(sp)  :00011400
R14(lr)  :00001024
R15(pc)  :00001058
-------------------
CPSR Register
Negative(N):0
Zero(Z)    :1
Carry(C)   :1
Overflow(V):0
IRQ Disable:1
FIQ Disable:1
Thumb(T)   :0
CPU Mode   :System
-------------------
0x600000df
```

x86 Registers



Registers [edit]

Further information: X86 architecture § x86 registers

x86 processors have a collection of registers available to be used as stores for binary data. Collectivel
Each register has a special purpose in addition to what they can all do:

- AX multiply/divide, string load & store
- CX count for string operations & shifts
- DX port address for IN and OUT
- BX index register for MOVE
- SP points to top of the stack
- BP points to base of the stack frame
- SI points to a source in stream operations
- DI points to a destination in stream operations

Along with the general registers there are additionally the:

- IP instruction pointer
- FLAGS
- segment registers (CS, DS, ES, FS, GS, SS) which determine where a 64k segment starts (no FS
- extra extension registers (MMX, 3DNow!, SSE, etc.) (Pentium & later only).

The IP register points to the memory offset of the next instruction in the code segment (it points to the
by the programmer directly.

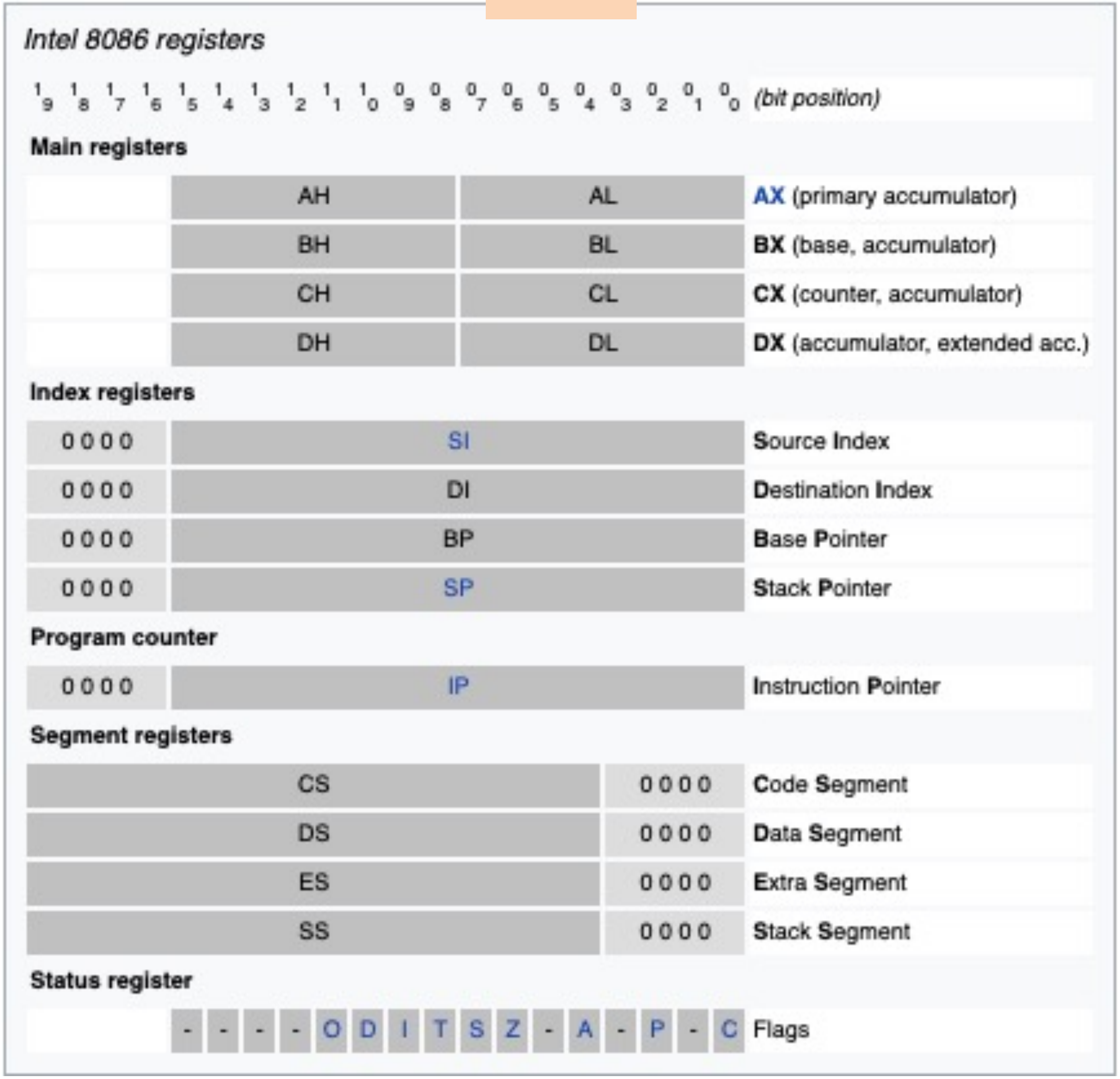
The x86 registers can be used by using the MOV instructions. For example, in Intel syntax:

```
mov ax, 1234h ; copies the value 1234hex (4660d) into register AX
```

```
mov bx, ax    ; copies the value of the AX register into the BX register
```

x86 Registers

20-bit



Intel 8086 registers

19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00	*(bit position)*
Main registers																				
				AH								AL								**AX** (primary accumulator)
				BH								BL								**BX** (base, accumulator)
				CH								CL								**CX** (counter, accumulator)
				DH								DL								**DX** (accumulator, extended acc.)
Index registers																				
0000				SI																Source Index
0000				DI																Destination Index
0000				BP																Base Pointer
0000				SP																Stack Pointer
Program counter																				
0000				IP																Instruction Pointer
Segment registers																				
CS																0000				Code Segment
DS																0000				Data Segment
ES																0000				Extra Segment
SS																0000				Stack Segment
Status register																				
				-	-	-	-	O	D	I	T	S	Z	-	A	-	P	-	C	Flags

x86 Registers

Hennessy & Patterson

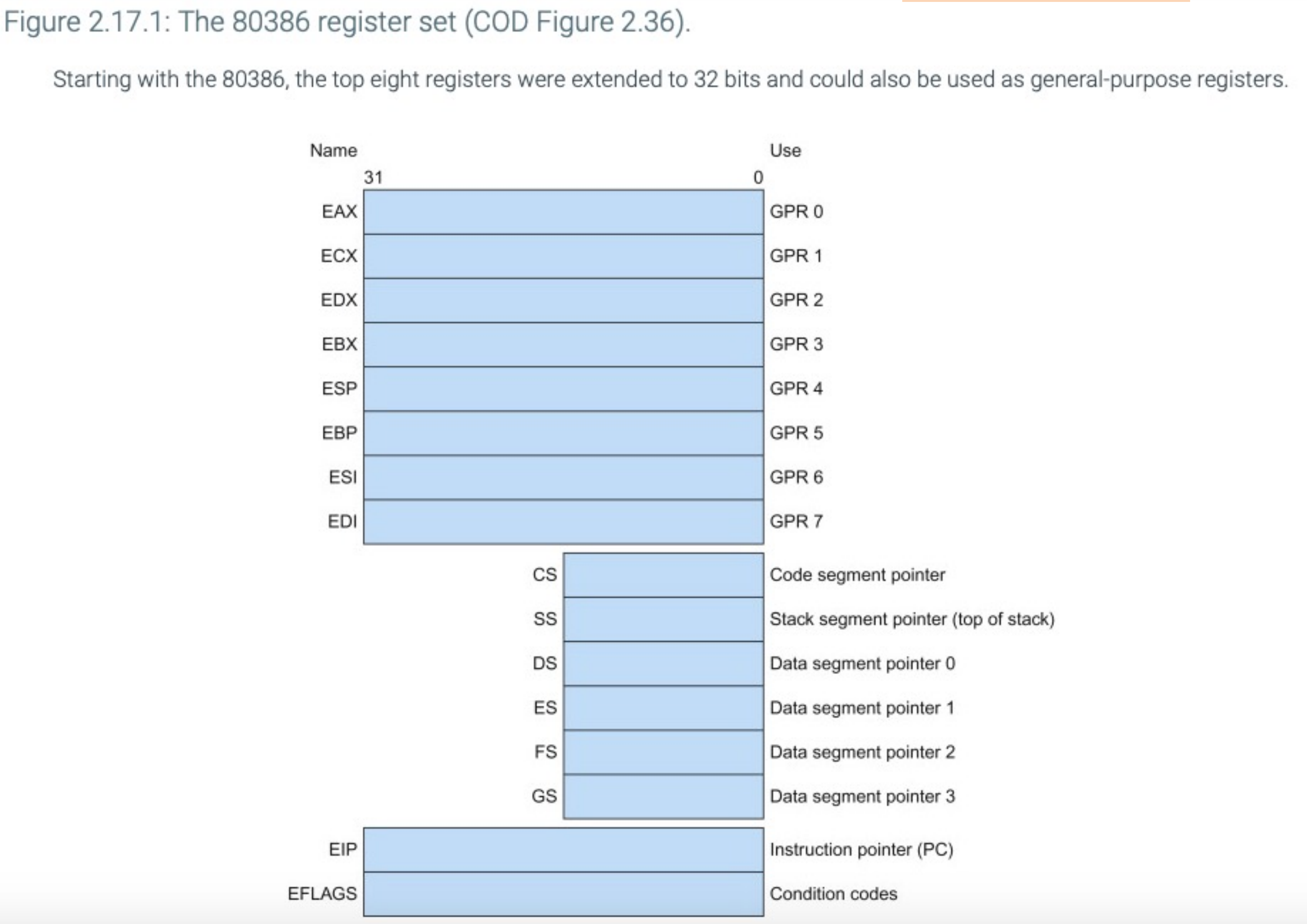
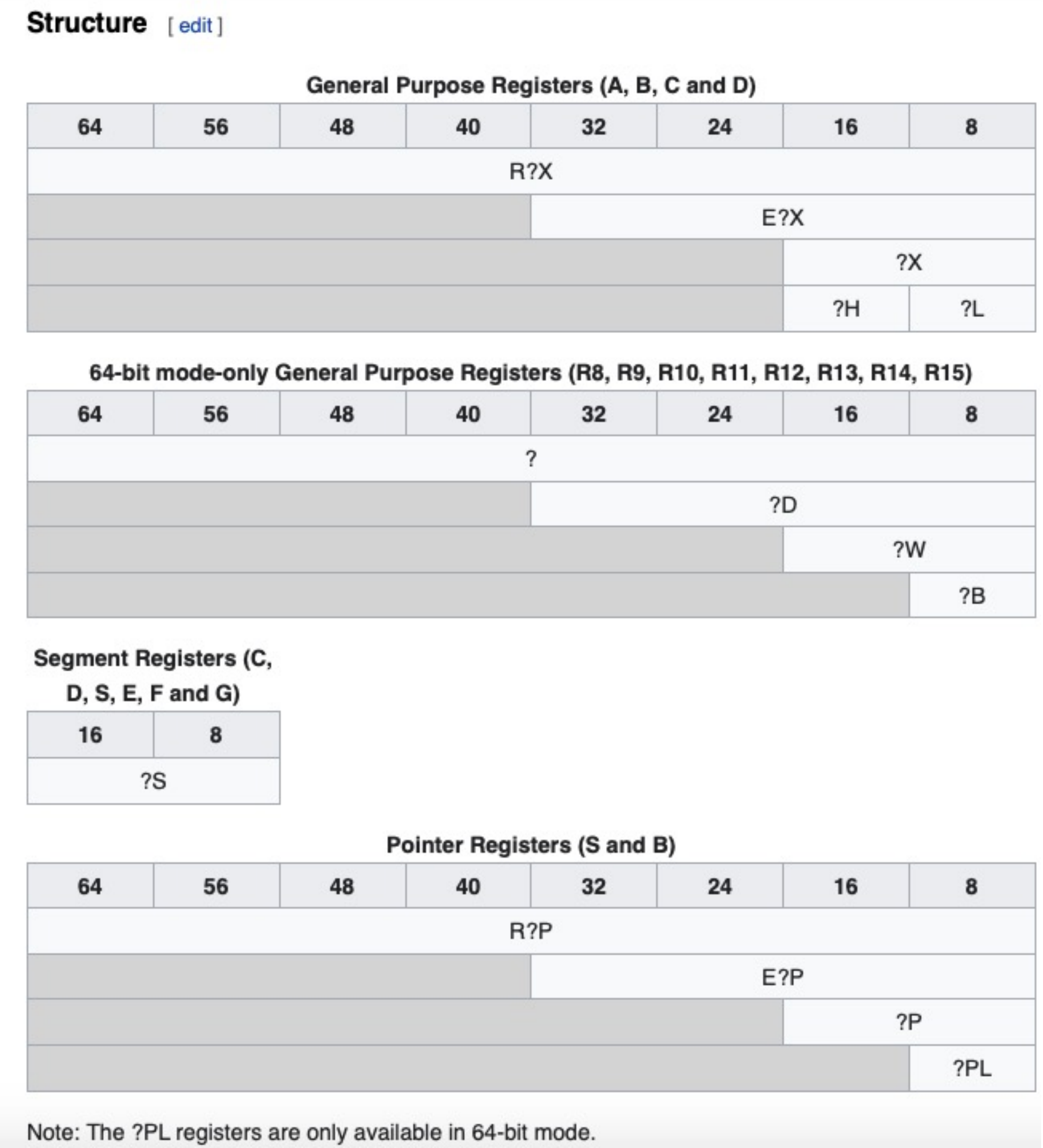


Figure 2.17.1: The 80386 register set (COD Figure 2.36).

Starting with the 80386, the top eight registers were extended to 32 bits and could also be used as general-purpose registers.

Name	31 0	Use
EAX		GPR 0
ECX		GPR 1
EDX		GPR 2
EBX		GPR 3
ESP		GPR 4
EBP		GPR 5
ESI		GPR 6
EDI		GPR 7
CS		Code segment pointer
SS		Stack segment pointer (top of stack)
DS		Data segment pointer 0
ES		Data segment pointer 1
FS		Data segment pointer 2
GS		Data segment pointer 3
EIP		Instruction pointer (PC)
EFLAGS		Condition codes

x86 Registers



Structure [edit]

General Purpose Registers (A, B, C and D)

64	56	48	40	32	24	16	8
R?X							
				E?X			
						?X	
						?H	?L

64-bit mode-only General Purpose Registers (R8, R9, R10, R11, R12, R13, R14, R15)

64	56	48	40	32	24	16	8
?							
				?D			
						?W	
							?B

Segment Registers (C, D, S, E, F and G)

16	8
?S	

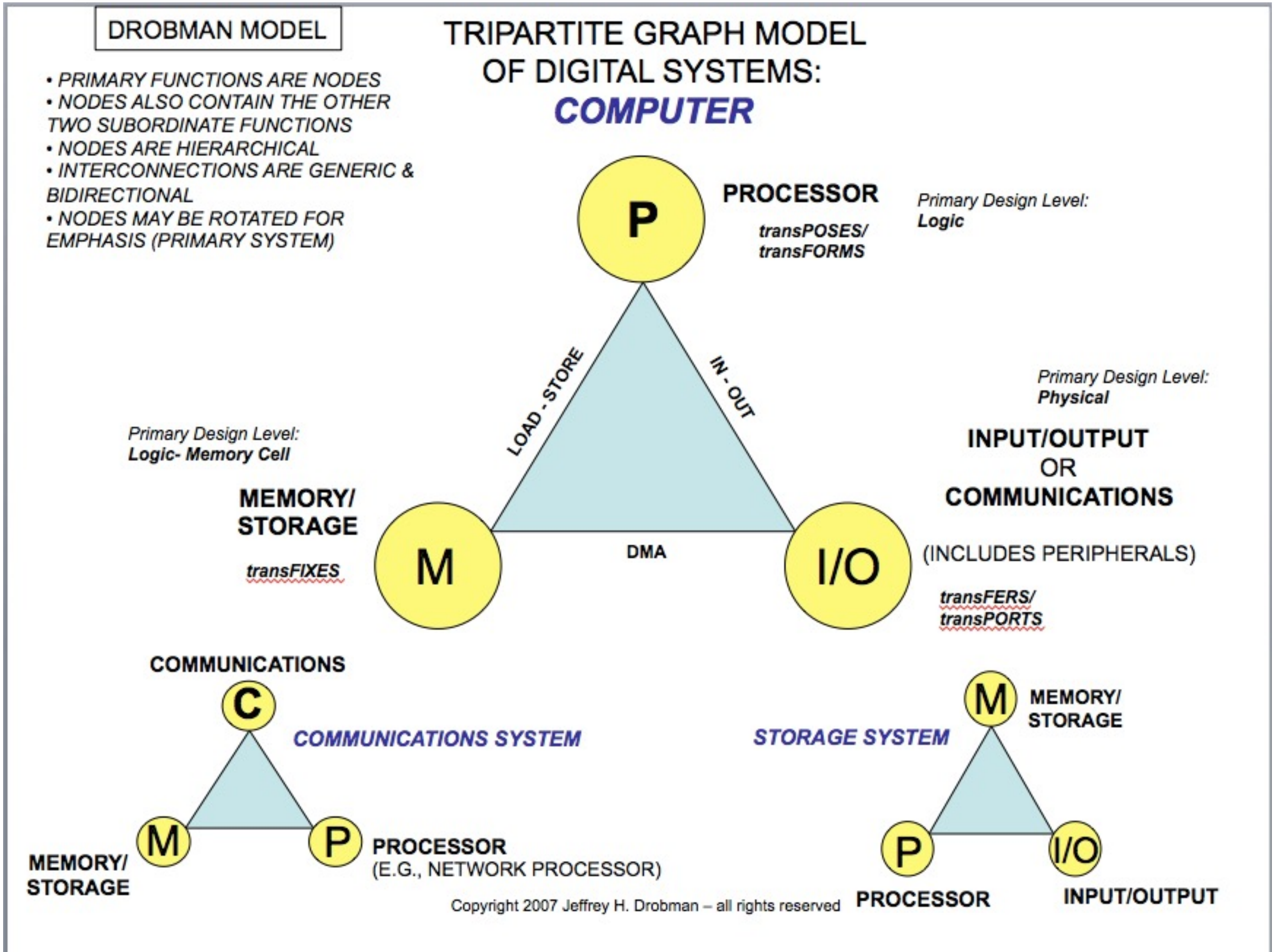
Pointer Registers (S and B)

64	56	48	40	32	24	16	8
R?P							
				E?P			
						?P	
							?PL

Note: The ?PL registers are only available in 64-bit mode.

Buses

Hardware-System Model



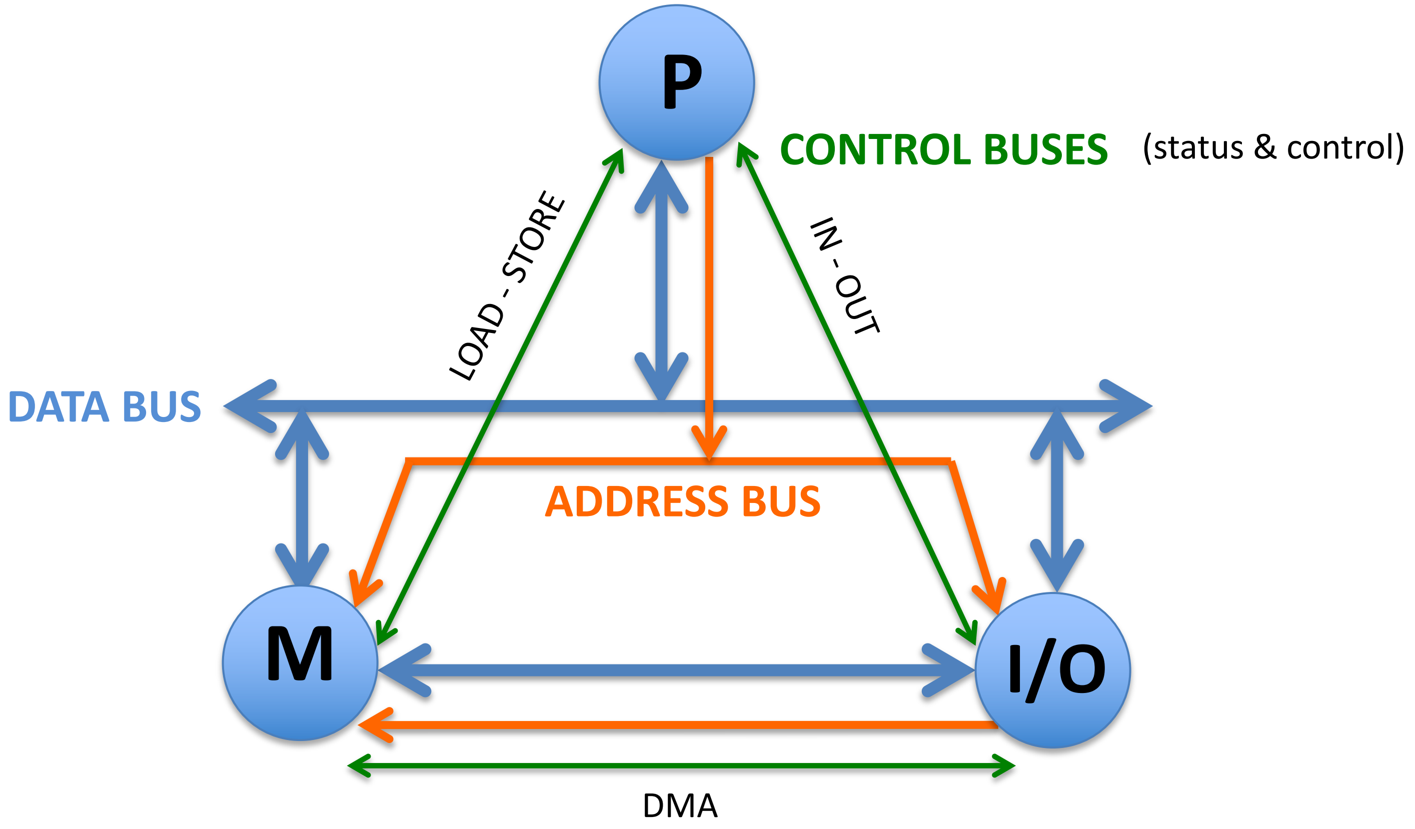
DROBMAN MODEL

- *PRIMARY FUNCTIONS ARE NODES*
- *NODES ALSO CONTAIN THE OTHER TWO SUBORDINATE FUNCTIONS*
- *NODES ARE HIERARCHICAL*
- *INTERCONNECTIONS ARE GENERIC & BIDIRECTIONAL*
- *NODES MAY BE ROTATED FOR EMPHASIS (PRIMARY SYSTEM)*

TRIPARTITE GRAPH MODEL
OF DIGITAL SYSTEMS:
COMPUTER

System Buses

NON-Multiplexed **A+D** Buses



NON-MULTIPLEXED BUSES

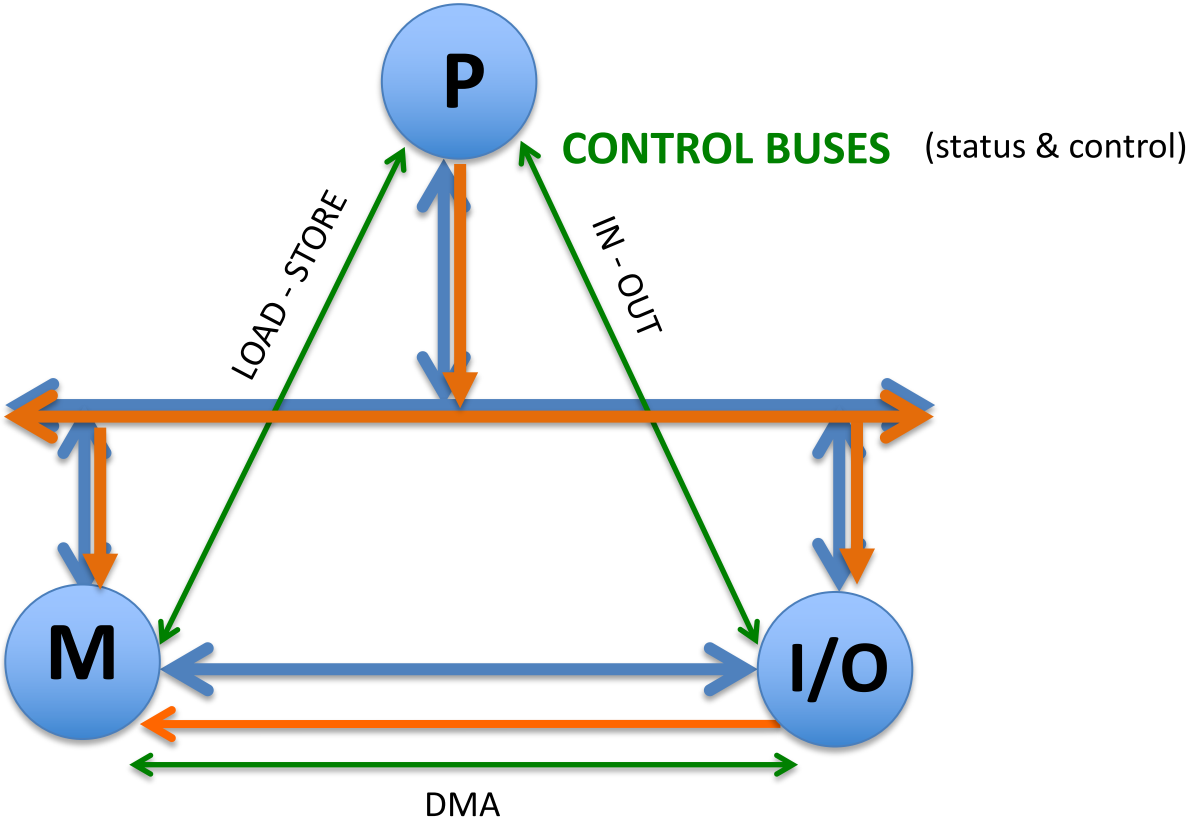
System Buses

Multiplexed **A/D** Bus

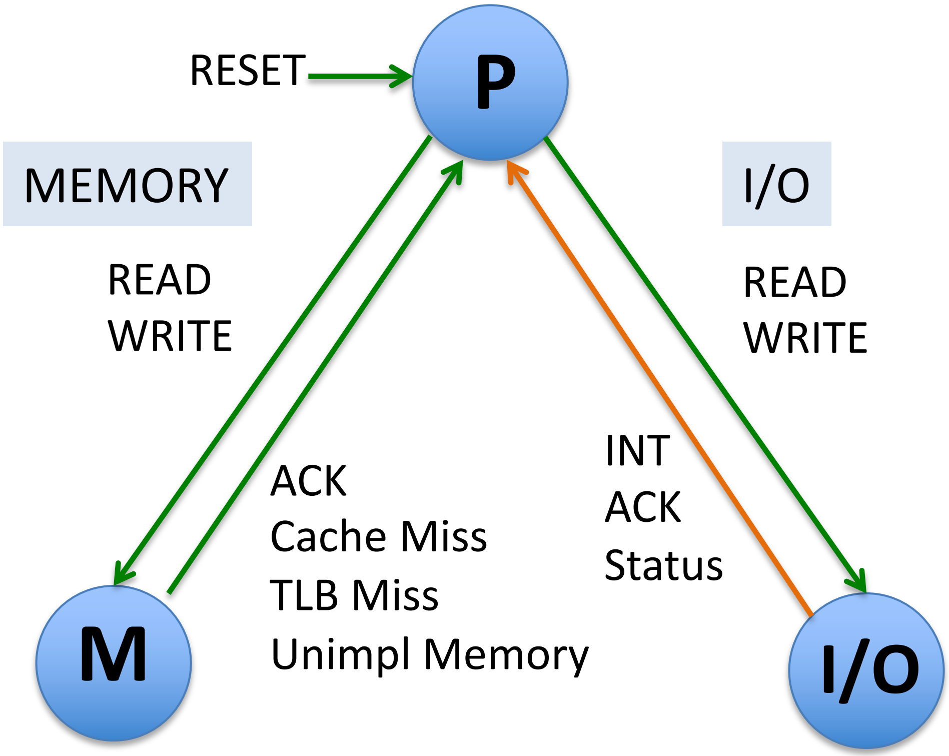
MULTIPLEXED BUS
(shared)

ADDRESS /
DATA BUS

enable A/D*



Control Signals



RESET

MEMORY

READ
WRITE

ACK
Cache Miss
TLB Miss
Unimpl Memory

I/O

READ
WRITE

INT
ACK
Status

- ✧ Intel mode → R*, W*
- ✧ Motorola mode → R/W*, DS*

- ✧ Polling
- ✧ Interrupts

Interrupts

- ○ NMI
- ○ NVI
- ○ VI

System PCI Bus & Bridges

Hennessy & Patterson

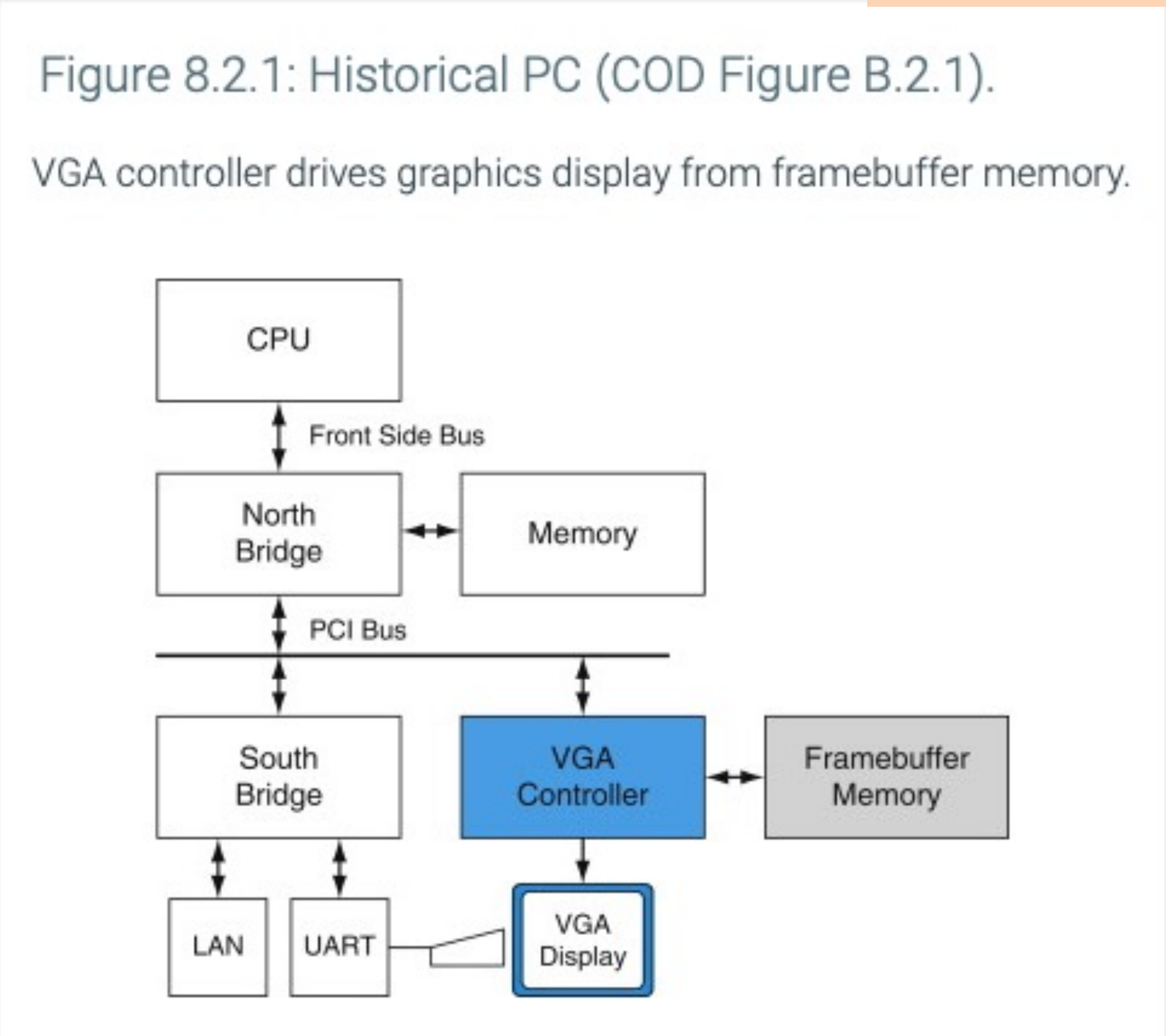


Figure 8.2.1: Historical PC (COD Figure B.2.1).

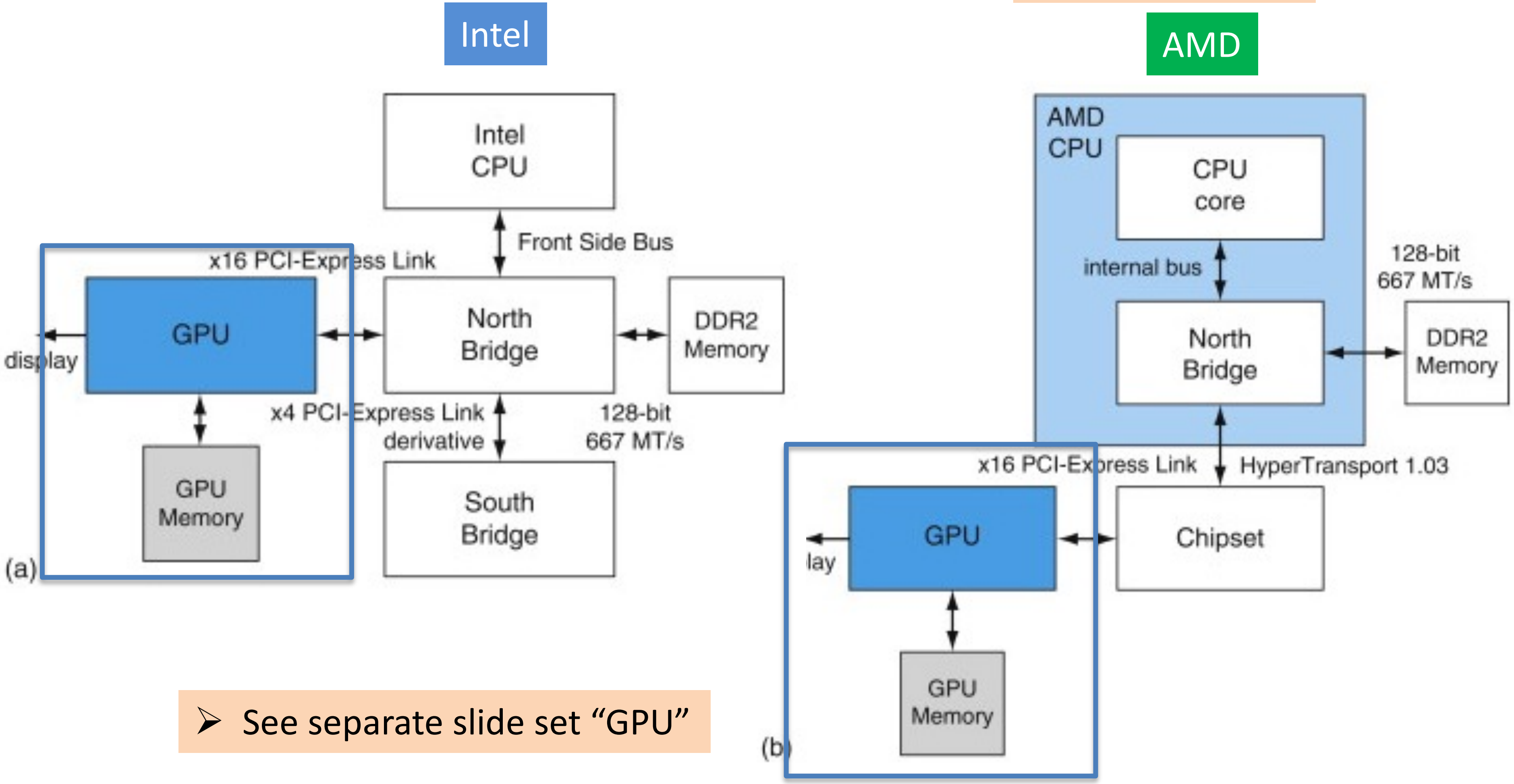
VGA controller drives graphics display from framebuffer memory.

CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE
COMP122

APU = CPU + GPU

DR JEFF SOFTWARE
INDIE APP DEVELOPER
© Jeff Drobman
2016-2022

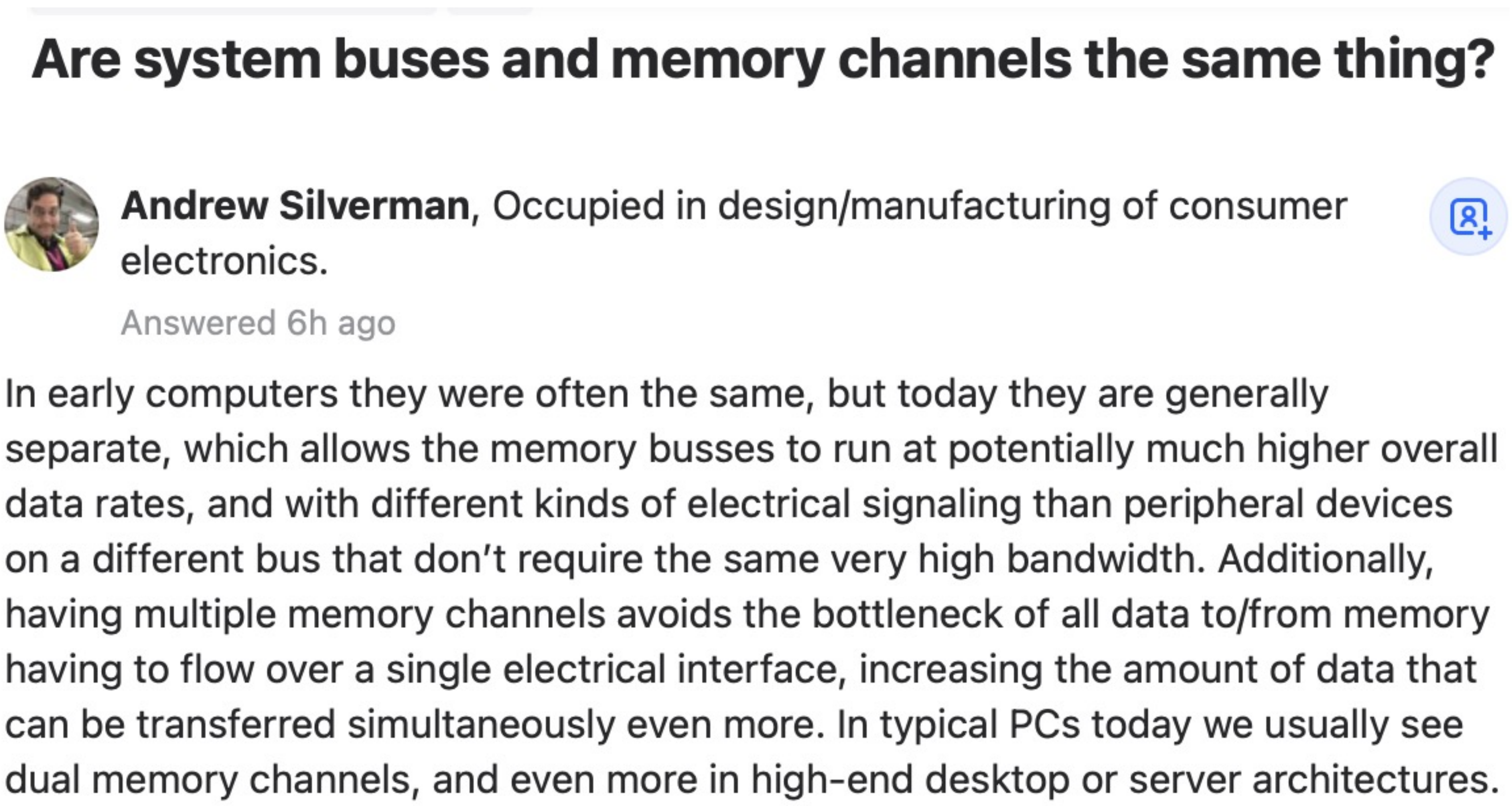
Hennessy & Patterson



- ➢ See separate slide set "GPU"

Memory Buses

Quora



Are system buses and memory channels the same thing?

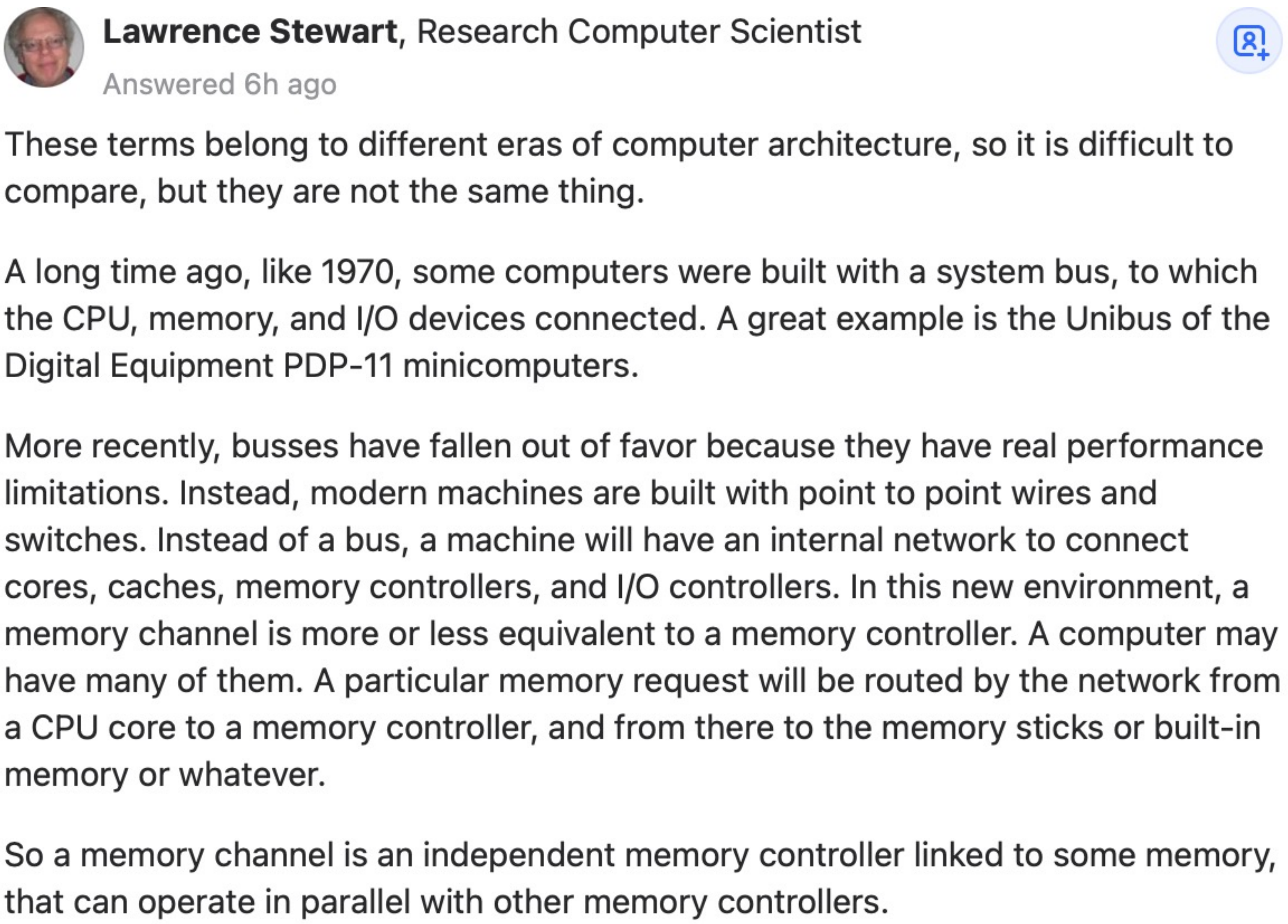
Andrew Silverman, Occupied in design/manufacturing of consumer electronics.

Answered 6h ago

In early computers they were often the same, but today they are generally separate, which allows the memory busses to run at potentially much higher overall data rates, and with different kinds of electrical signaling than peripheral devices on a different bus that don't require the same very high bandwidth. Additionally, having multiple memory channels avoids the bottleneck of all data to/from memory having to flow over a single electrical interface, increasing the amount of data that can be transferred simultaneously even more. In typical PCs today we usually see dual memory channels, and even more in high-end desktop or server architectures.

Memory Buses

Quora



Lawrence Stewart, Research Computer Scientist

Answered 6h ago

These terms belong to different eras of computer architecture, so it is difficult to compare, but they are not the same thing.

A long time ago, like 1970, some computers were built with a system bus, to which the CPU, memory, and I/O devices connected. A great example is the Unibus of the Digital Equipment PDP-11 minicomputers.

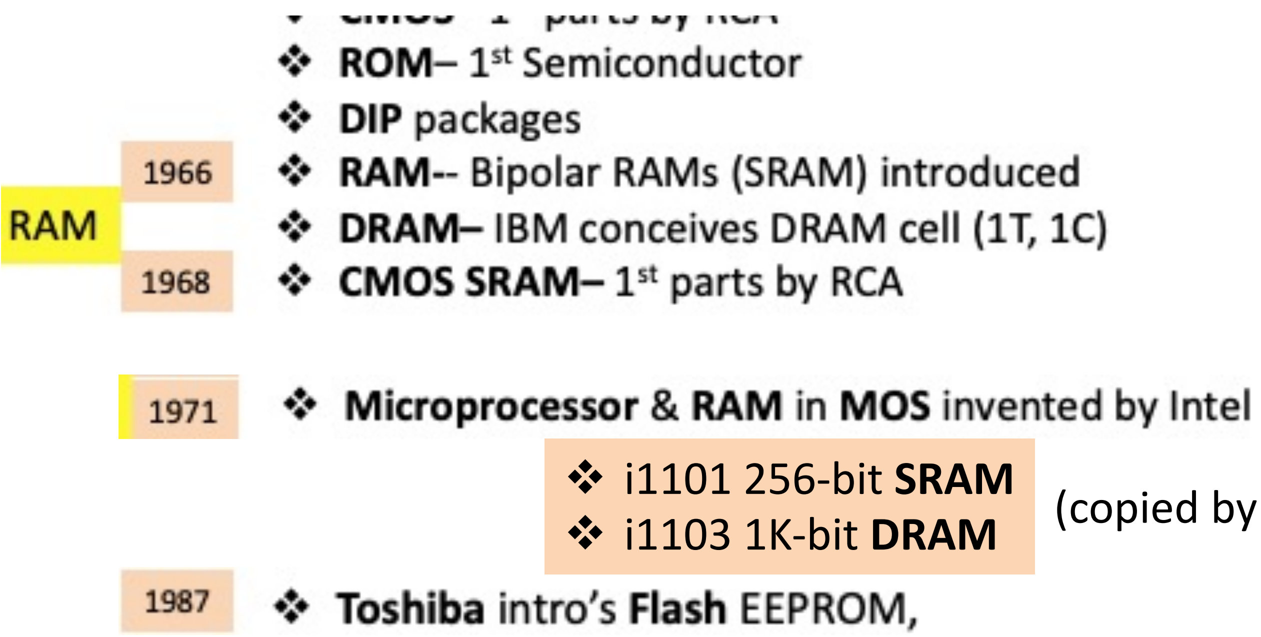
More recently, busses have fallen out of favor because they have real performance limitations. Instead, modern machines are built with point to point wires and switches. Instead of a bus, a machine will have an internal network to connect cores, caches, memory controllers, and I/O controllers. In this new environment, a memory channel is more or less equivalent to a memory controller. A computer may have many of them. A particular memory request will be routed by the network from a CPU core to a memory controller, and from there to the memory sticks or built-in memory or whatever.

So a memory channel is an independent memory controller linked to some memory, that can operate in parallel with other memory controllers.

Computer Architecture

Memory

Memory IC Timeline



- CMOS – 1st parts by RCA
- **ROM**– 1st Semiconductor
- **DIP** packages

RAM

1966

- **RAM**-- Bipolar RAMs (SRAM) introduced
- **DRAM**– IBM conceives DRAM cell (1T, 1C)
- **CMOS SRAM**– 1st parts by RCA

1968

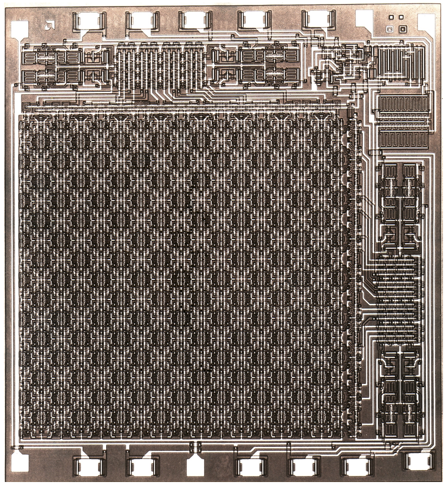
1971

- **Microprocessor** & **RAM** in **MOS** invented by Intel
  - i1101 256-bit **SRAM**
  - i1103 1K-bit **DRAM**

(copied by AMD)

1987

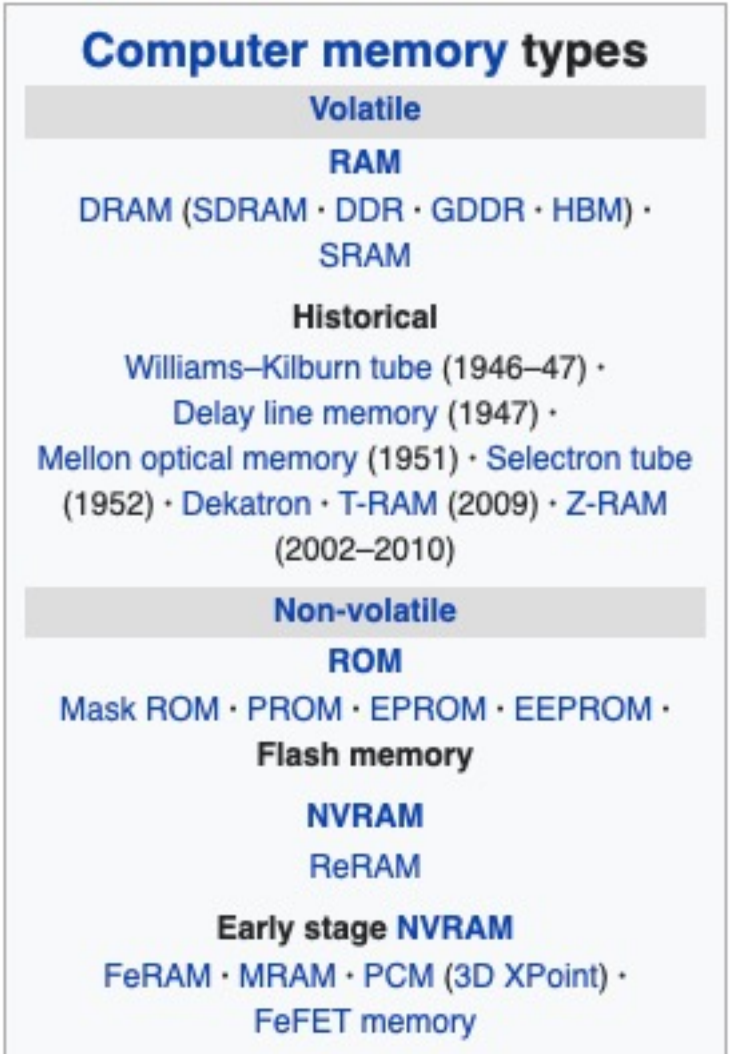
- **Toshiba** intro's **Flash** EEPROM,



Am1101A die

Memory Types

Wiki



Computer memory types

Volatile

RAM

DRAM (SDRAM · DDR · GDDR · HBM) · SRAM

Historical

Williams–Kilburn tube (1946–47) · Delay line memory (1947) · Mellon optical memory (1951) · Selectron tube (1952) · Dekatron · T-RAM (2009) · Z-RAM (2002–2010)

Non-volatile

ROM

Mask ROM · PROM · EPROM · EEPROM ·

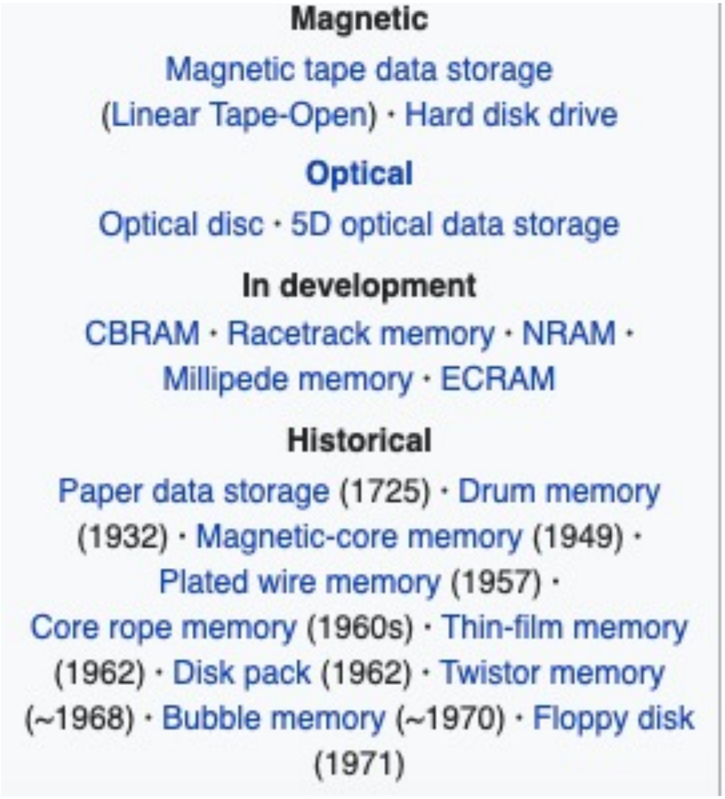
Flash memory

NVRAM

ReRAM

Early stage NVRAM

FeRAM · MRAM · PCM (3D XPoint) · FeFET memory



Magnetic

Magnetic tape data storage (Linear Tape-Open) · Hard disk drive

Optical

Optical disc · 5D optical data storage

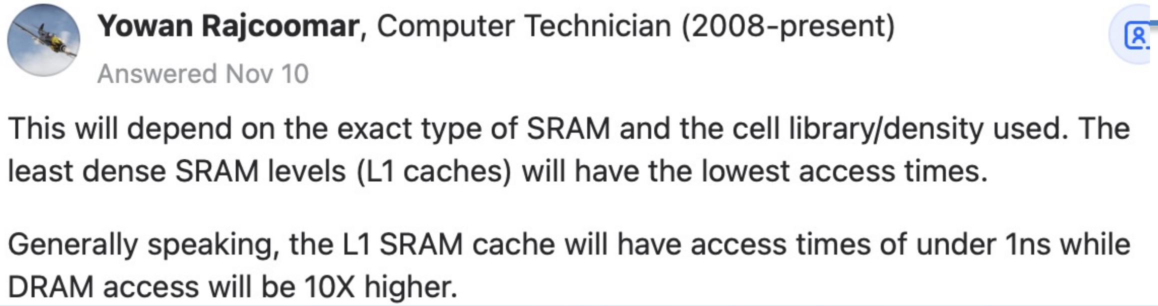
In development

CBRAM · Racetrack memory · NRAM · Millipede memory · ECRAM

Historical

Paper data storage (1725) · Drum memory (1932) · Magnetic-core memory (1949) · Plated wire memory (1957) · Core rope memory (1960s) · Thin-film memory (1962) · Disk pack (1962) · Twistor memory (~1968) · Bubble memory (~1970) · Floppy disk (1971)

Memory Speeds

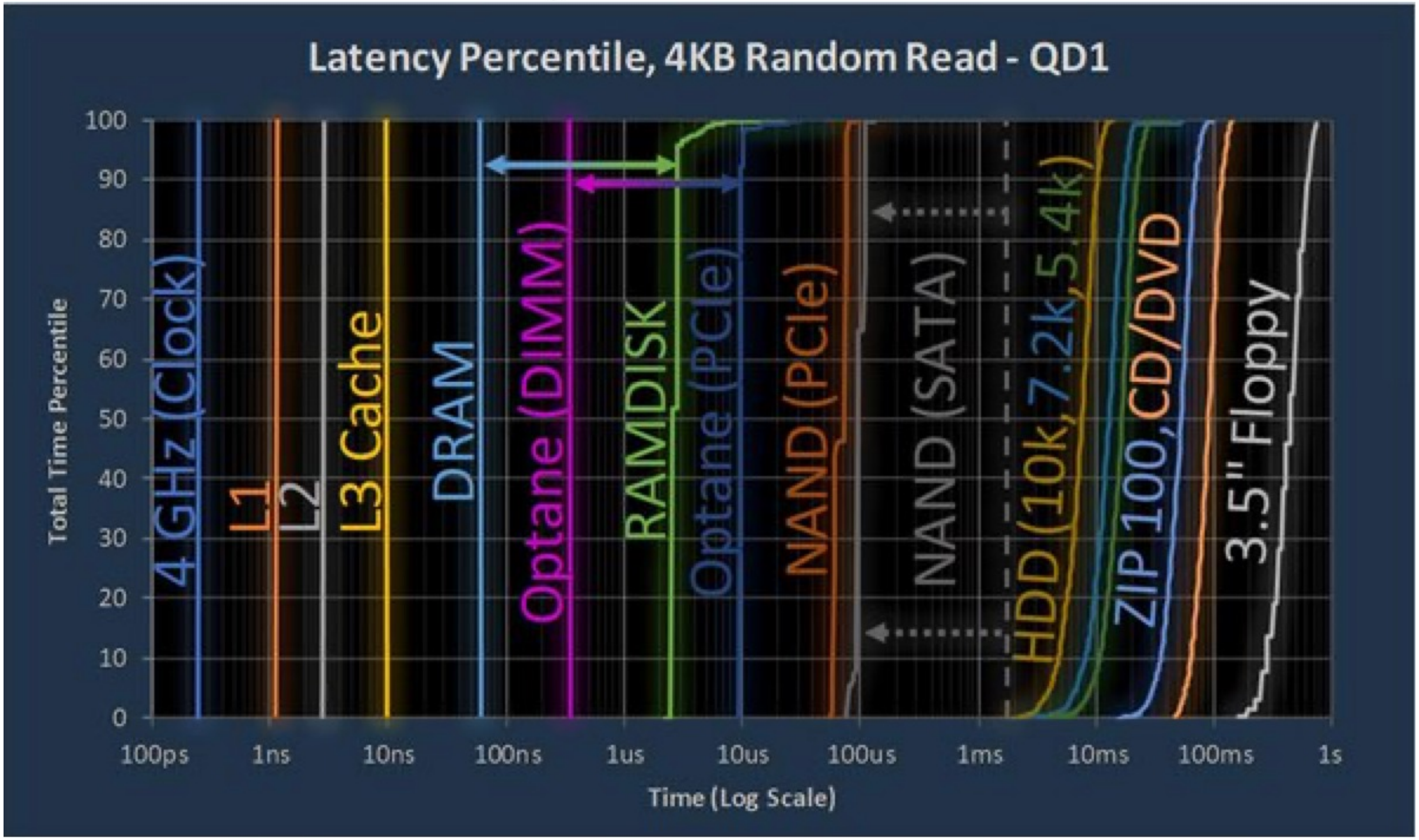


Yowan Rajcoomar, Computer Technician (2008-present)

Answered Nov 10

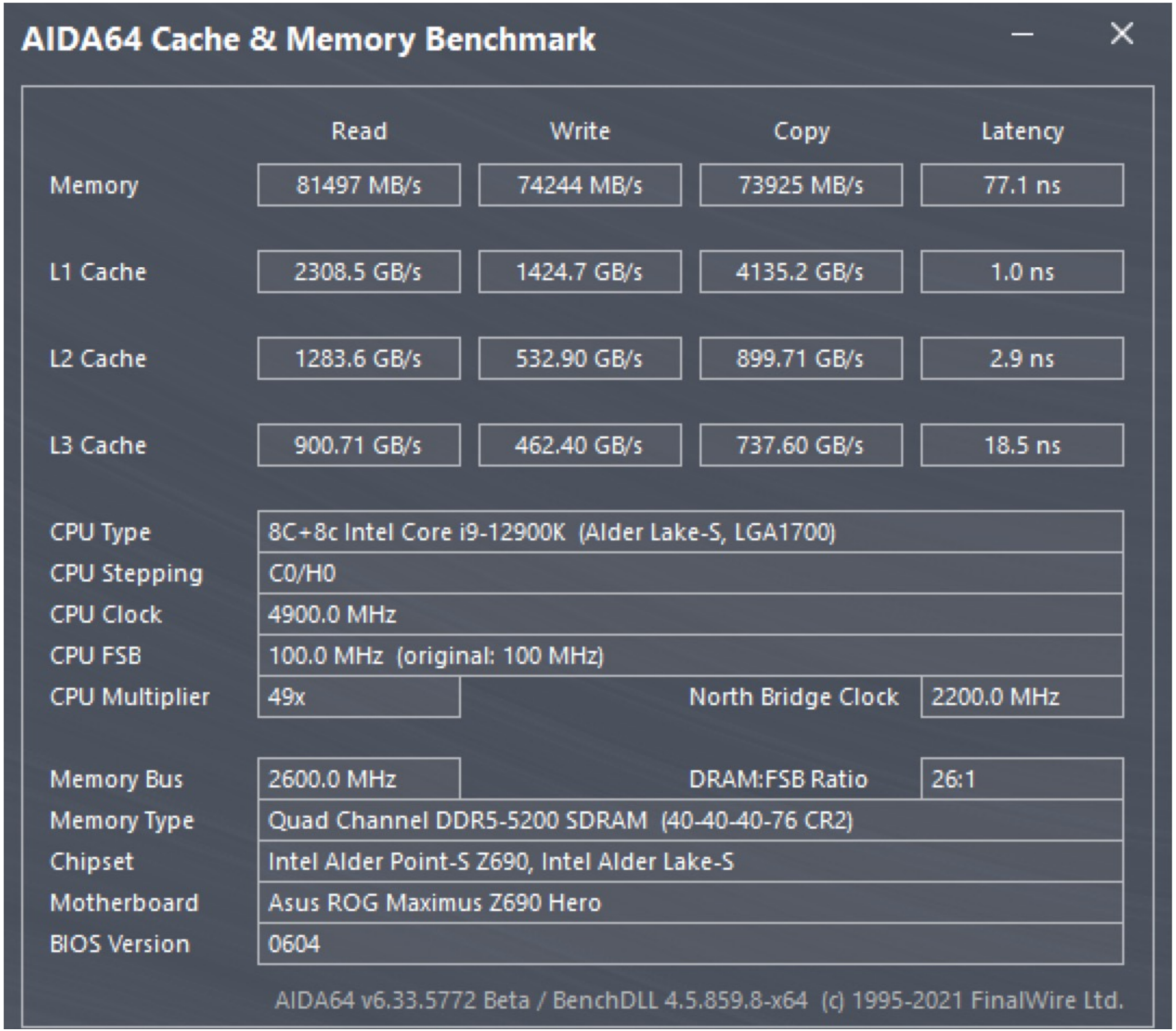
This will depend on the exact type of SRAM and the cell library/density used. The least dense SRAM levels (L1 caches) will have the lowest access times.

Generally speaking, the L1 SRAM cache will have access times of under 1ns while DRAM access will be 10X higher.



Memory Speeds: Core i9-12

Example with a Core i9–12900K showing access times of L1, L2 and L3 caches alongside DDR5 SDRAM:



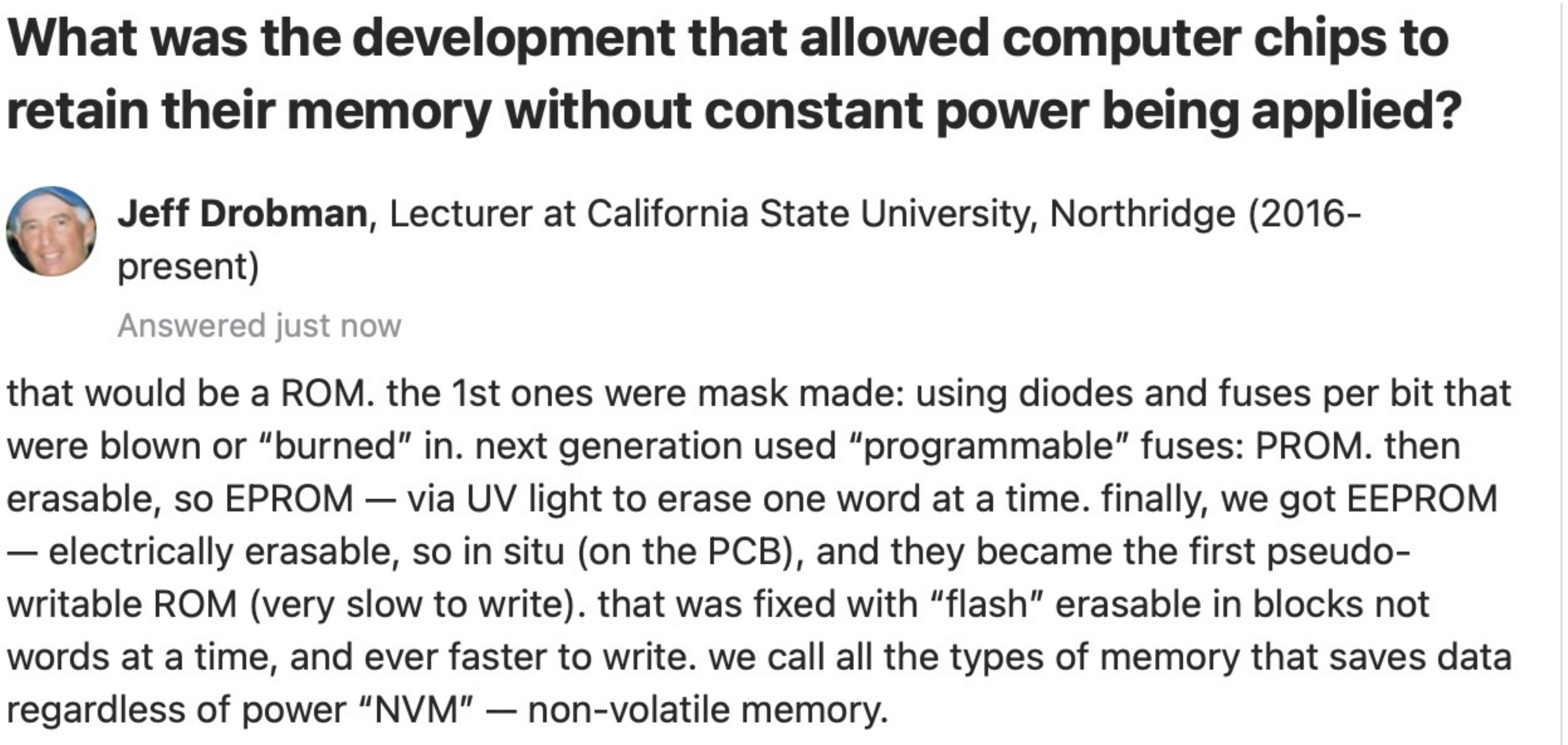
AIDA64 Cache & Memory Benchmark

	Read	Write	Copy	Latency
Memory	81497 MB/s	74244 MB/s	73925 MB/s	77.1 ns
L1 Cache	2308.5 GB/s	1424.7 GB/s	4135.2 GB/s	1.0 ns
L2 Cache	1283.6 GB/s	532.90 GB/s	899.71 GB/s	2.9 ns
L3 Cache	900.71 GB/s	462.40 GB/s	737.60 GB/s	18.5 ns

CPU Type	8C+8c Intel Core i9-12900K (Alder Lake-S, LGA1700)		
CPU Stepping	C0/H0		
CPU Clock	4900.0 MHz		
CPU FSB	100.0 MHz (original: 100 MHz)		
CPU Multiplier	49x	North Bridge Clock	2200.0 MHz
Memory Bus	2600.0 MHz	DRAM:FSB Ratio	26:1
Memory Type	Quad Channel DDR5-5200 SDRAM (40-40-40-76 CR2)		
Chipset	Intel Alder Point-S Z690, Intel Alder Lake-S		
Motherboard	Asus ROG Maximus Z690 Hero		
BIOS Version	0604		

AIDA64 v6.33.5772 Beta / BenchDLL 4.5.859.8-x64 (c) 1995-2021 FinalWire Ltd.

ROM



What was the development that allowed computer chips to retain their memory without constant power being applied?

Jeff Drobman, Lecturer at California State University, Northridge (2016-present)

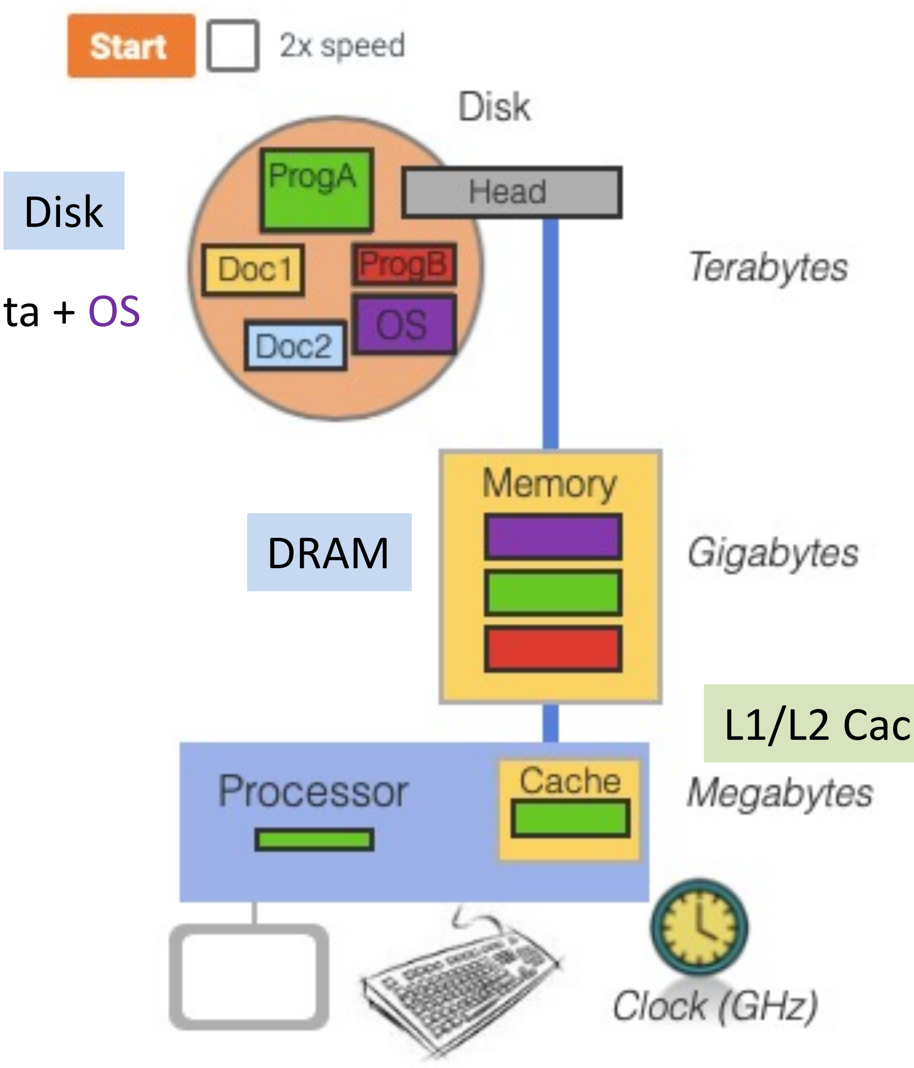
Answered just now

that would be a ROM. the 1st ones were mask made: using diodes and fuses per bit that were blown or "burned" in. next generation used "programmable" fuses: PROM. then erasable, so EPROM — via UV light to erase one word at a time. finally, we got EEPROM — electrically erasable, so in situ (on the PCB), and they became the first pseudo-writable ROM (very slow to write). that was fixed with "flash" erasable in blocks not words at a time, and ever faster to write. we call all the types of memory that saves data regardless of power "NVM" — non-volatile memory.

ROM→ PROM→ EPROM→ EEPROM→ Flash

Computer Memory Org

1.6.1: Some computer components.



Disk

Code + Data + OS

DRAM

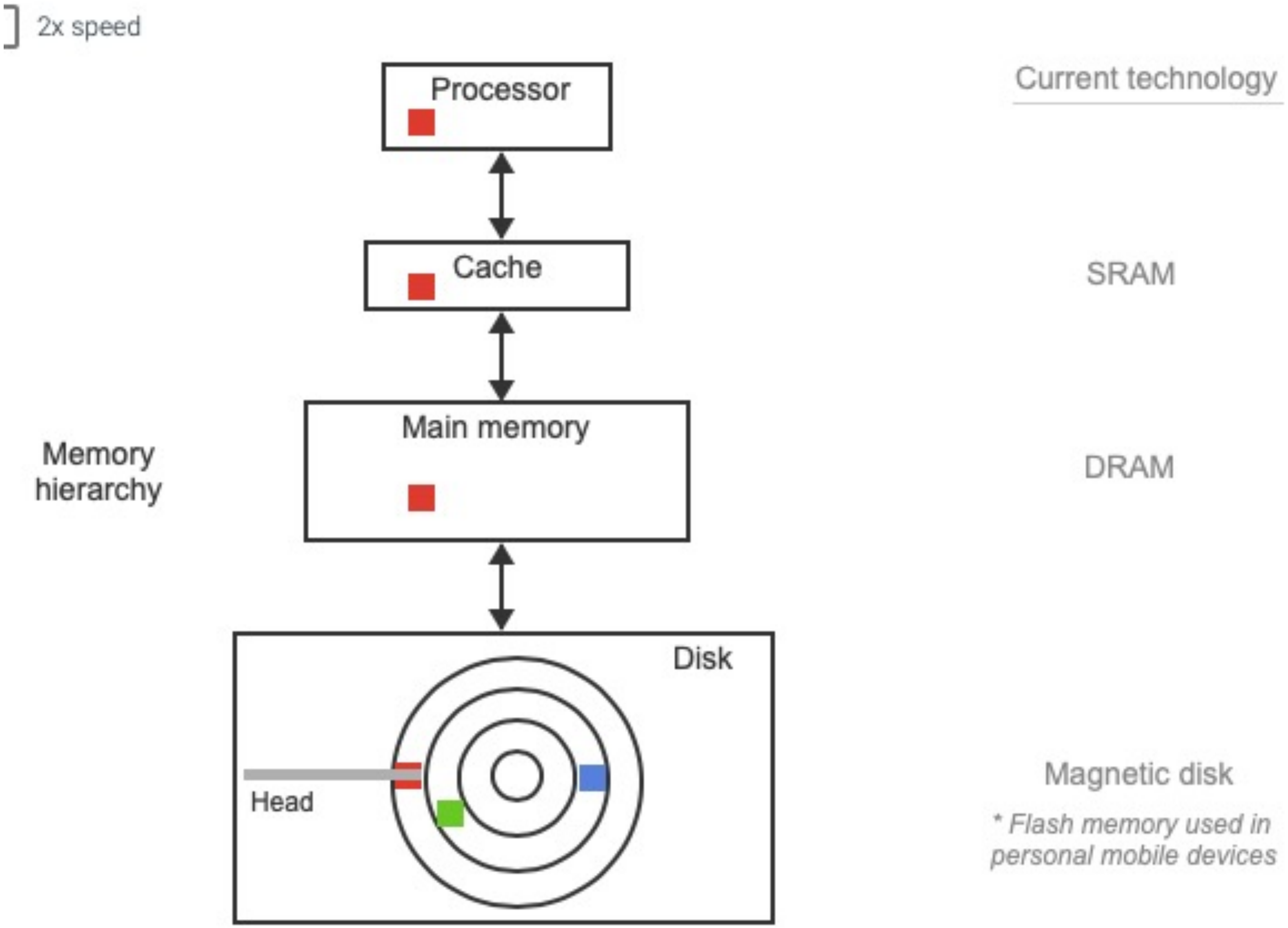
L1/L2 Cache | SRAM

Memory

P&H Ch 5

COMP 122: Computer Architecture and Assembly Language
Spring 2020

5.2.1: Primary technologies used in memories.



* Flash memory used in personal mobile devices

DRAM Timeline

Hennessy & Patterson

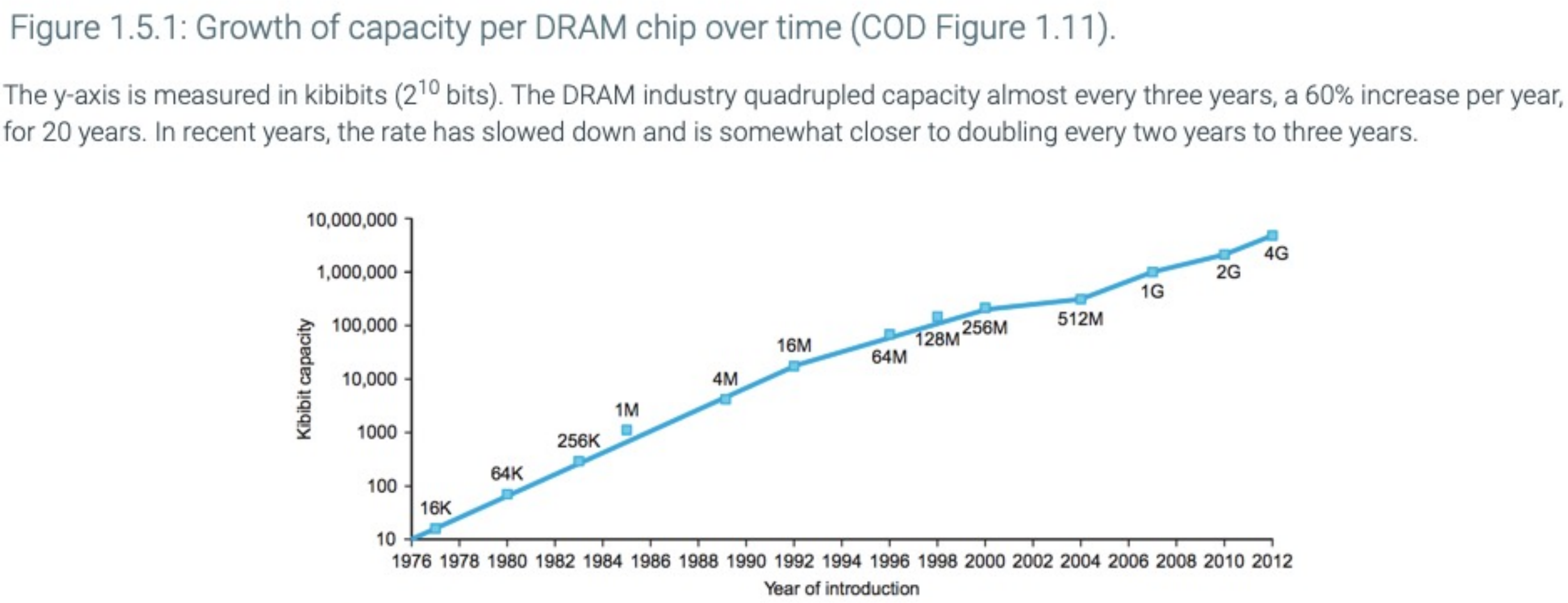
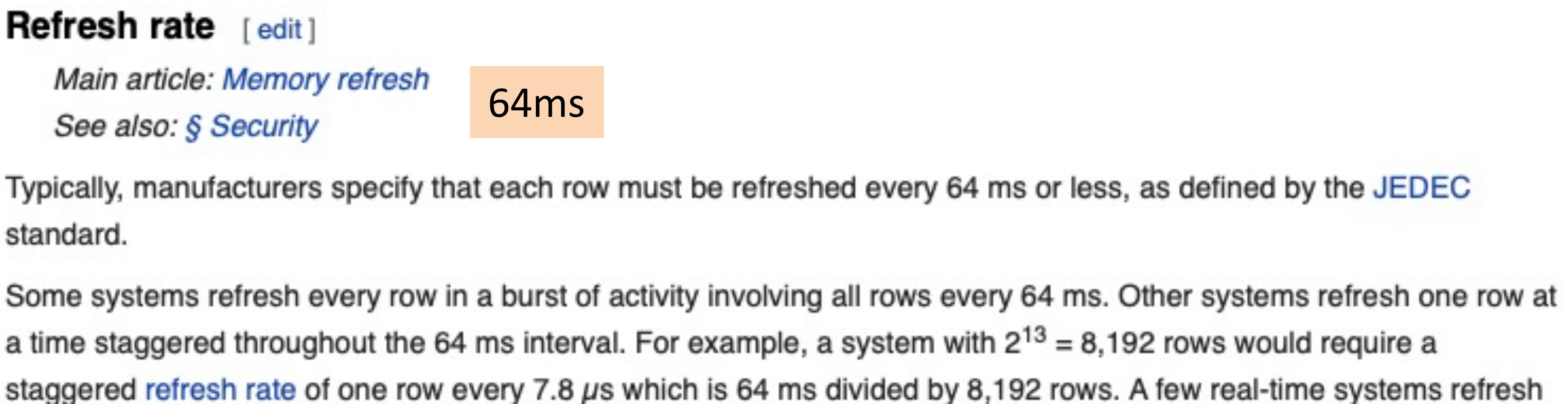


Figure 1.5.1: Growth of capacity per DRAM chip over time (COD Figure 1.11).

The y-axis is measured in kibibits (2^{10} bits). The DRAM industry quadrupled capacity almost every three years, a 60% increase per year, for 20 years. In recent years, the rate has slowed down and is somewhat closer to doubling every two years to three years.



Refresh rate [edit]

Main article: Memory refresh

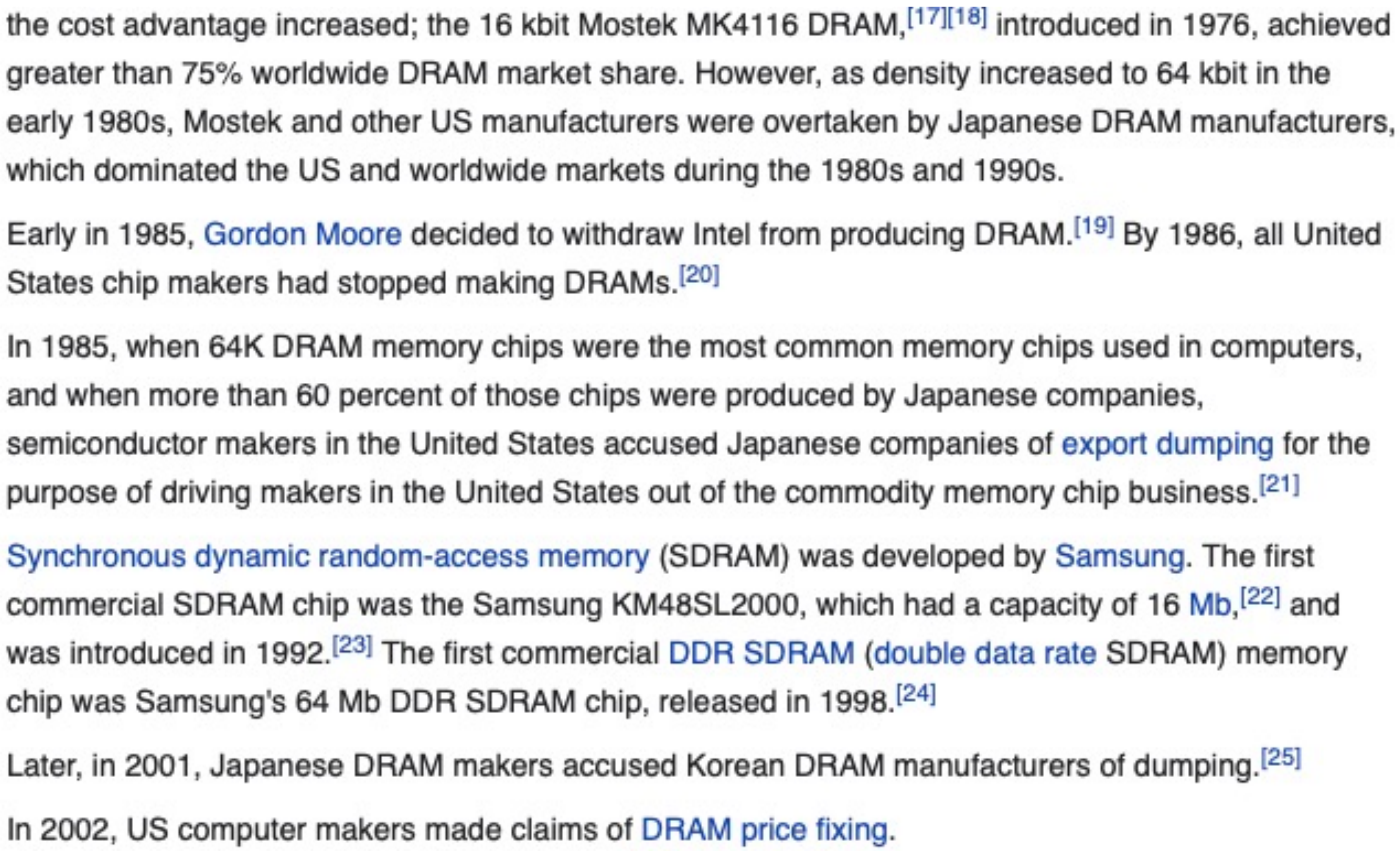
See also: § Security

64ms

Typically, manufacturers specify that each row must be refreshed every 64 ms or less, as defined by the JEDEC standard.

Some systems refresh every row in a burst of activity involving all rows every 64 ms. Other systems refresh one row at a time staggered throughout the 64 ms interval. For example, a system with 2^{13} = 8,192 rows would require a staggered refresh rate of one row every 7.8 μs which is 64 ms divided by 8,192 rows. A few real-time systems refresh

DRAM History



the cost advantage increased; the 16 kbit Mostek MK4116 DRAM,[17][18] introduced in 1976, achieved greater than 75% worldwide DRAM market share. However, as density increased to 64 kbit in the early 1980s, Mostek and other US manufacturers were overtaken by Japanese DRAM manufacturers, which dominated the US and worldwide markets during the 1980s and 1990s.

Early in 1985, Gordon Moore decided to withdraw Intel from producing DRAM.[19] By 1986, all United States chip makers had stopped making DRAMs.[20]

In 1985, when 64K DRAM memory chips were the most common memory chips used in computers, and when more than 60 percent of those chips were produced by Japanese companies, semiconductor makers in the United States accused Japanese companies of export dumping for the purpose of driving makers in the United States out of the commodity memory chip business.[21]

Synchronous dynamic random-access memory (SDRAM) was developed by Samsung. The first commercial SDRAM chip was the Samsung KM48SL2000, which had a capacity of 16 Mb,[22] and was introduced in 1992.[23] The first commercial DDR SDRAM (double data rate SDRAM) memory chip was Samsung's 64 Mb DDR SDRAM chip, released in 1998.[24]

Later, in 2001, Japanese DRAM makers accused Korean DRAM manufacturers of dumping.[25]

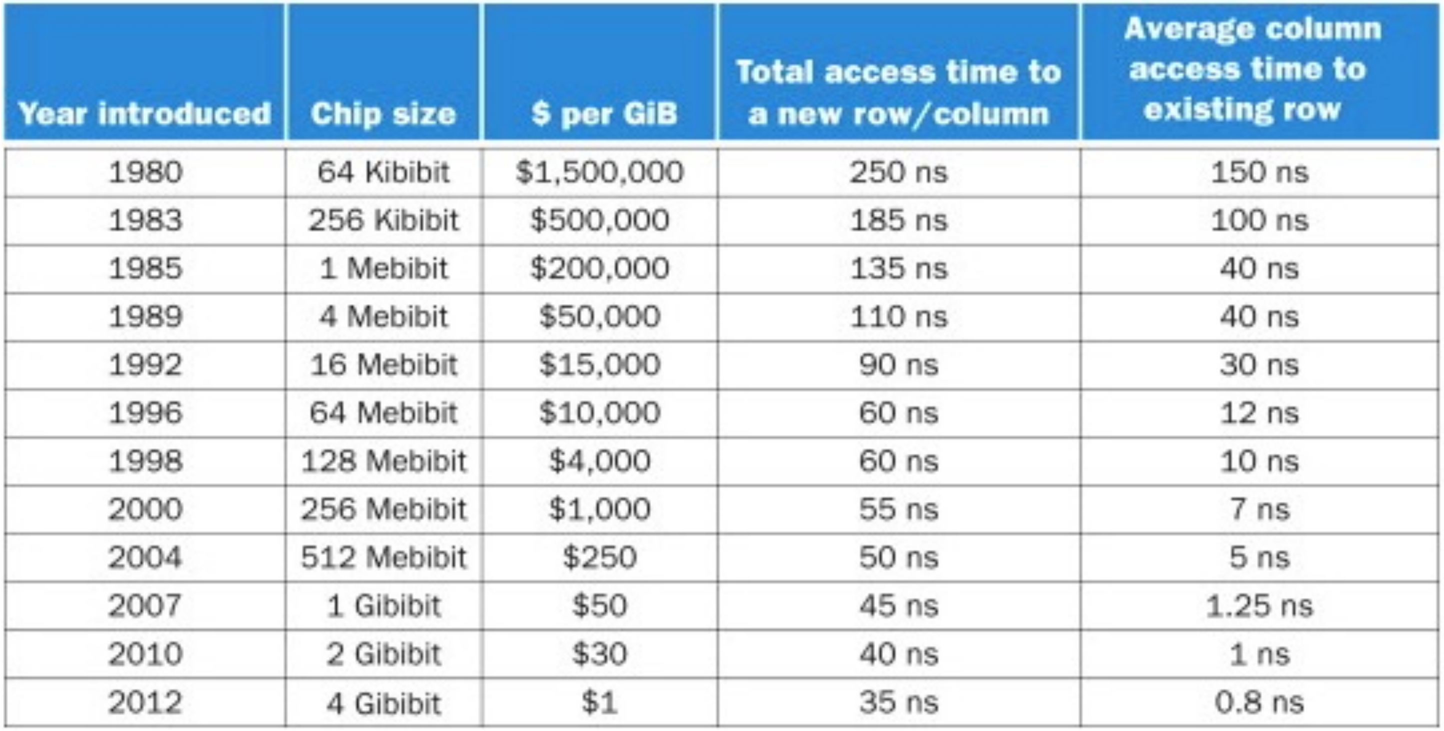
In 2002, US computer makers made claims of DRAM price fixing.

Memory

DRAM

P&H Ch 5

Figure 5.2.1: DRAM size increased by multiples of four approximately once every three years until 1996 and thereafter considerably slower (COD Figure 5.5).



Year introduced	Chip size	$ per GiB	Total access time to a new row/column	Average column access time to existing row
1980	64 Kibibit	$1,500,000	250 ns	150 ns
1983	256 Kibibit	$500,000	185 ns	100 ns
1985	1 Mebibit	$200,000	135 ns	40 ns
1989	4 Mebibit	$50,000	110 ns	40 ns
1992	16 Mebibit	$15,000	90 ns	30 ns
1996	64 Mebibit	$10,000	60 ns	12 ns
1998	128 Mebibit	$4,000	60 ns	10 ns
2000	256 Mebibit	$1,000	55 ns	7 ns
2004	512 Mebibit	$250	50 ns	5 ns
2007	1 Gibibit	$50	45 ns	1.25 ns
2010	2 Gibibit	$30	40 ns	1 ns
2012	4 Gibibit	$1	35 ns	0.8 ns

Micron DRAM

Micron

Our History of Innovation

Micron Technology Is Founded

Micron started as a four-person semiconductor design company in the basement of a Boise, Idaho dental office. Micron's first contract was for the design of a 64K memory chip for Mostek Corporation.

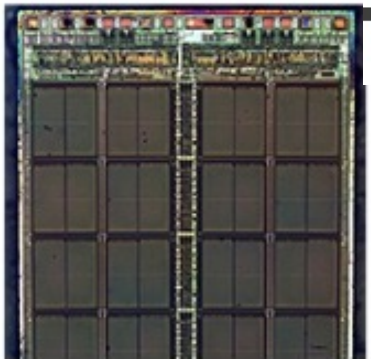
64Kb

1Mb

1987

Micron Brings 1-Megabit DRAM to Market

A milestone in density, the 1Mb DRAM became a staple for main memory in PCs and graphics cards during the late 1980s and 1990s. Micron's 1Mb DRAM enabled high-capacity SIMM modules that supported PCs equipped with Microsoft's new Windows OS.



Micron DDR DRAM

Micron

Double Data Rate

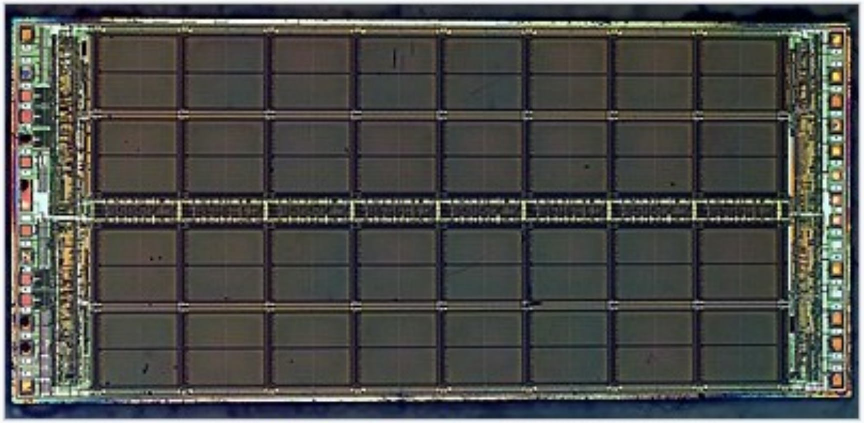
Founded: May 22, 1978

Revenue: $21.44 billion (2020)

Headquarters: Boise, ID

Micron's DDR5 Delivering Next Gen Performance

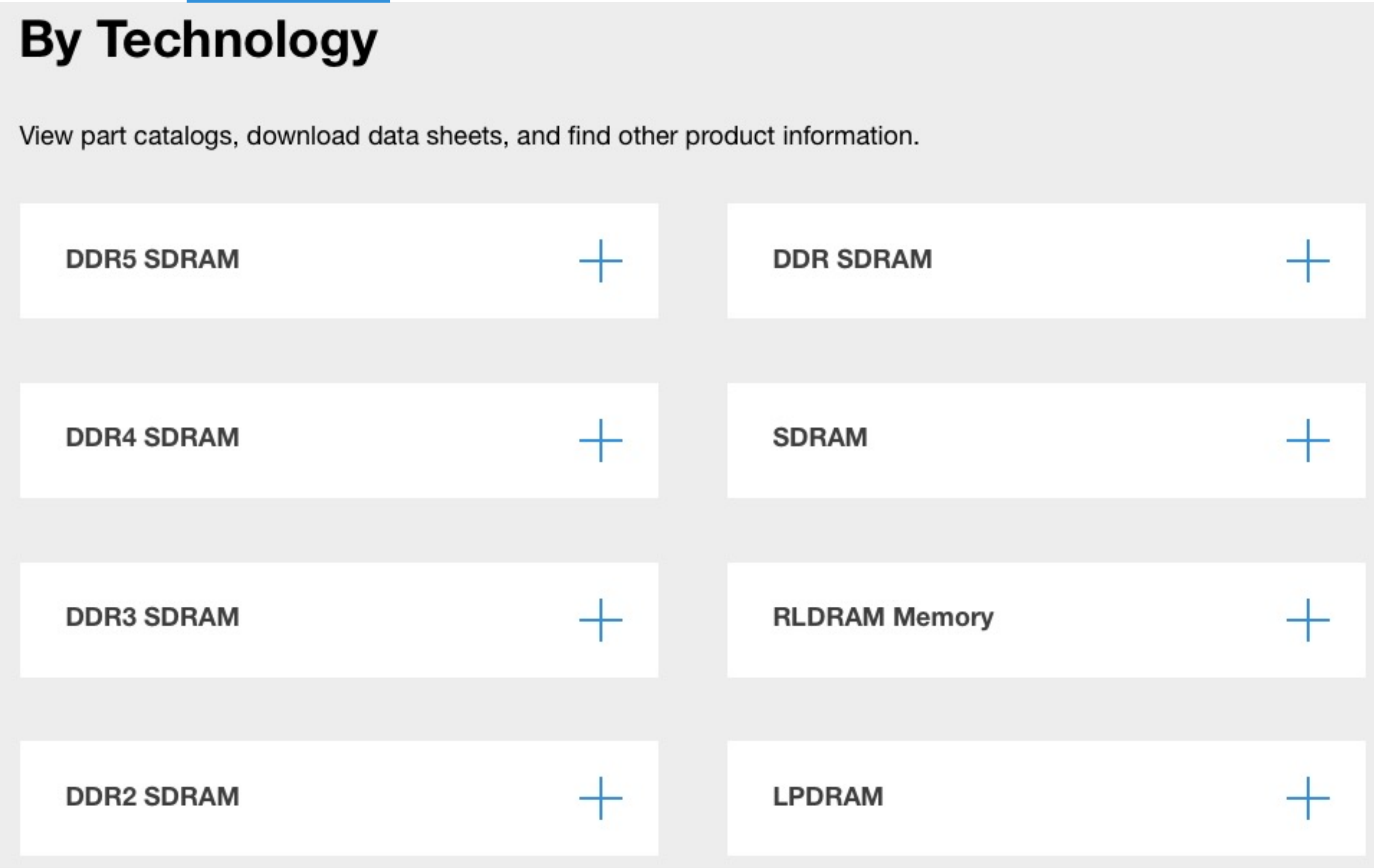
2004
2007
2014
2020
DDR2
DDR3
DDR4
DDR5



A die photograph of the Micron Technology MT4C1024 DRAM integrated circuit. It has a capacity of 1 megabit equivalent of 2^{20} bits or 128 kB.[1]

Micron DRAM

Micron



By Technology

View part catalogs, download data sheets, and find other product information.

DDR5 SDRAM +

DDR SDRAM +

DDR4 SDRAM +

SDRAM +

DDR3 SDRAM +

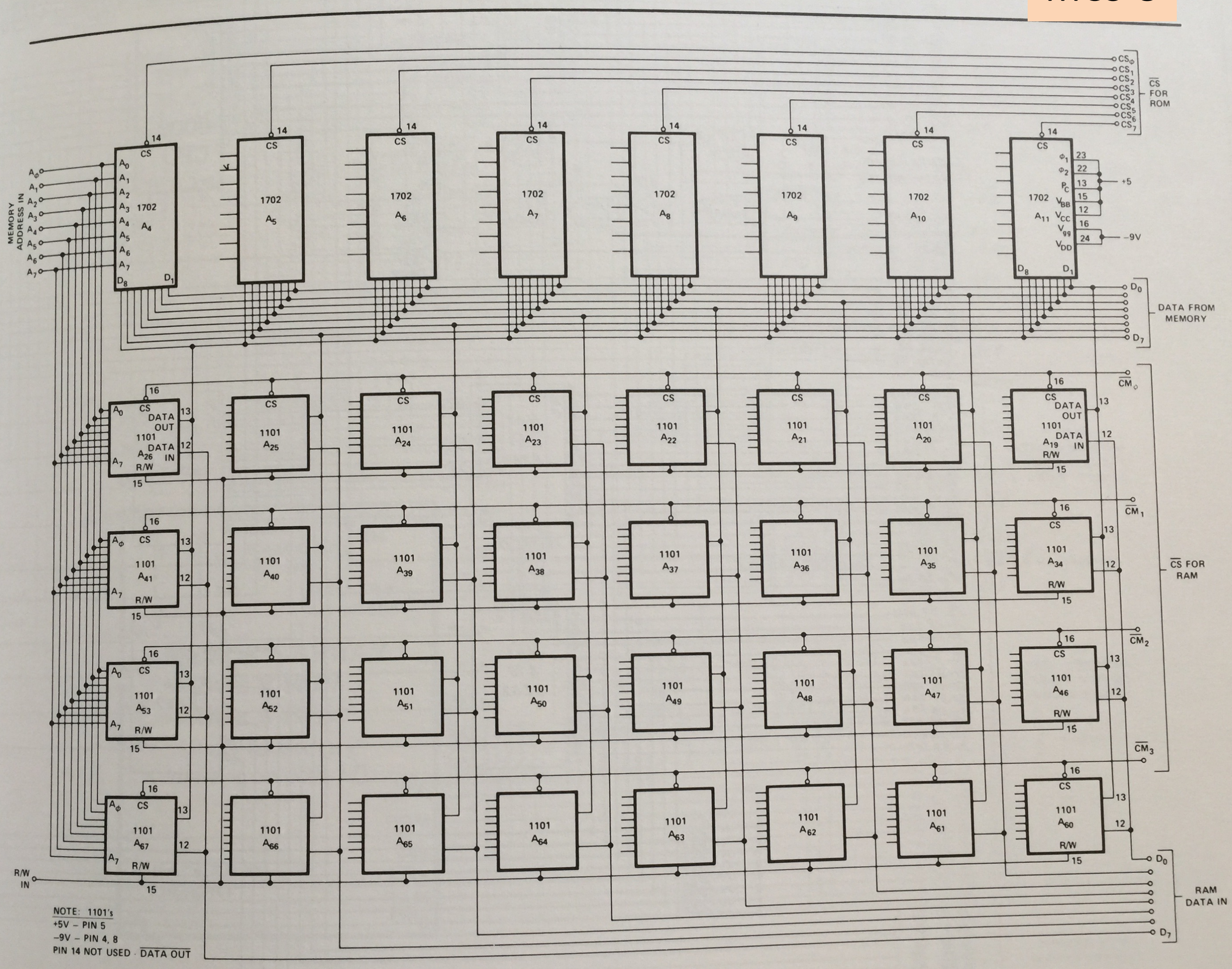
RLDRAM Memory +

DDR2 SDRAM +

LPDRAM +

RAM/ROM (x8) Chips

MCS-8

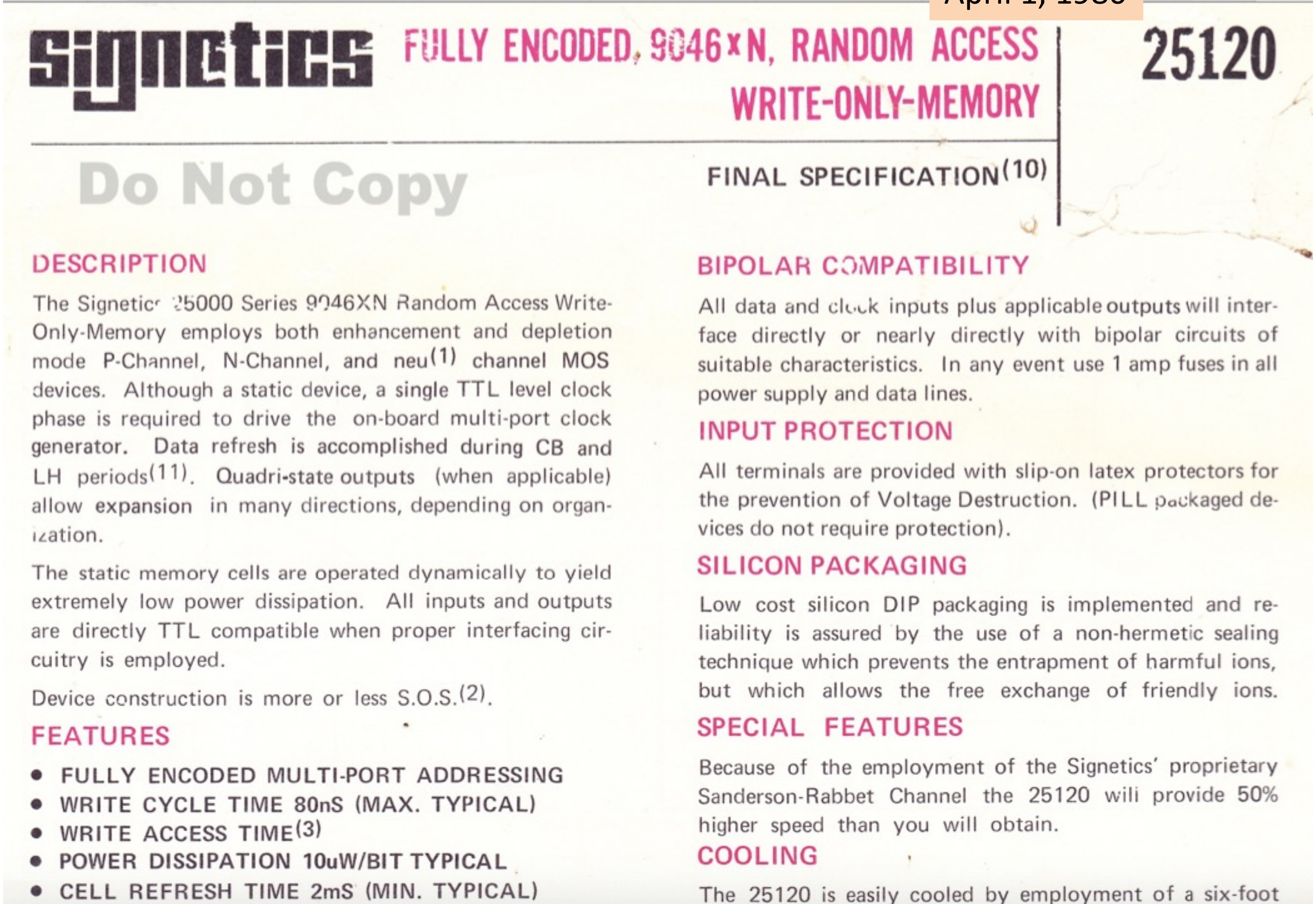


NOTE: 1101's
+5V – PIN 5
−9V – PIN 4, 8
PIN 14 NOT USED · DATA OUT

Figure 9. MCS-8 Memory System

WOM!

April 1, 1980



signetics FULLY ENCODED, 9046×N, RANDOM ACCESS WRITE-ONLY-MEMORY

25120

FINAL SPECIFICATION(10)

DESCRIPTION

The Signetics 25000 Series 9046XN Random Access Write-Only-Memory employs both enhancement and depletion mode P-Channel, N-Channel, and neu(1) channel MOS devices. Although a static device, a single TTL level clock phase is required to drive the on-board multi-port clock generator. Data refresh is accomplished during CB and LH periods(11). Quadri-state outputs (when applicable) allow expansion in many directions, depending on organization.

The static memory cells are operated dynamically to yield extremely low power dissipation. All inputs and outputs are directly TTL compatible when proper interfacing circuitry is employed.

Device construction is more or less S.O.S.(2).

FEATURES

- FULLY ENCODED MULTI-PORT ADDRESSING
- WRITE CYCLE TIME 80nS (MAX. TYPICAL)
- WRITE ACCESS TIME(3)
- POWER DISSIPATION 10uW/BIT TYPICAL
- CELL REFRESH TIME 2mS (MIN. TYPICAL)

BIPOLAR COMPATIBILITY

All data and clock inputs plus applicable outputs will interface directly or nearly directly with bipolar circuits of suitable characteristics. In any event use 1 amp fuses in all power supply and data lines.

INPUT PROTECTION

All terminals are provided with slip-on latex protectors for the prevention of Voltage Destruction. (PILL packaged devices do not require protection).

SILICON PACKAGING

Low cost silicon DIP packaging is implemented and reliability is assured by the use of a non-hermetic sealing technique which prevents the entrapment of harmful ions, but which allows the free exchange of friendly ions.

SPECIAL FEATURES

Because of the employment of the Signetics' proprietary Sanderson-Rabbet Channel the 25120 will provide 50% higher speed than you will obtain.

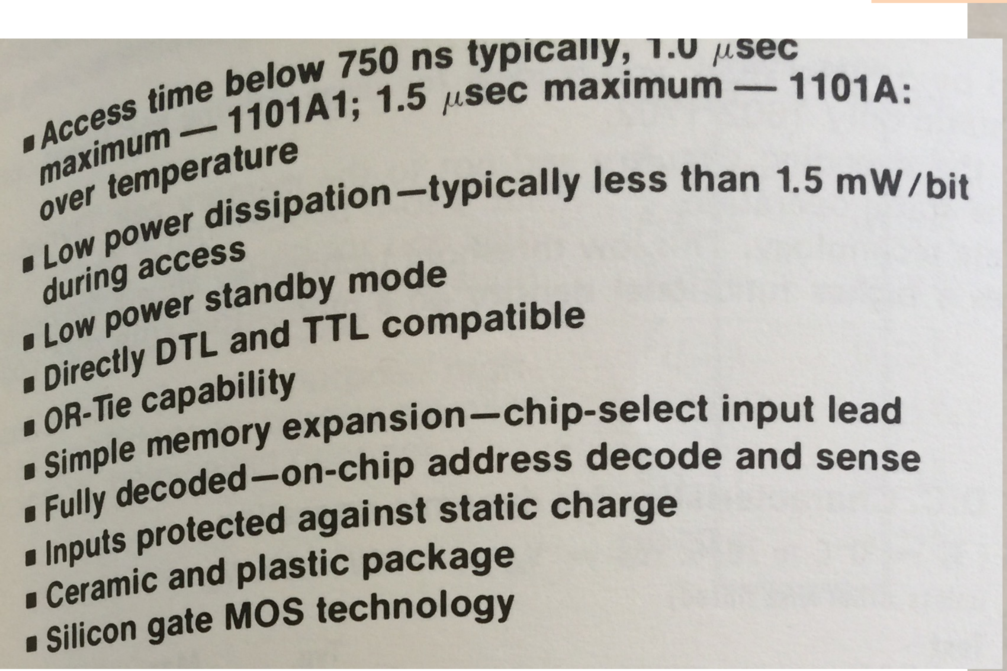
COOLING

The 25120 is easily cooled by employment of a six-foot

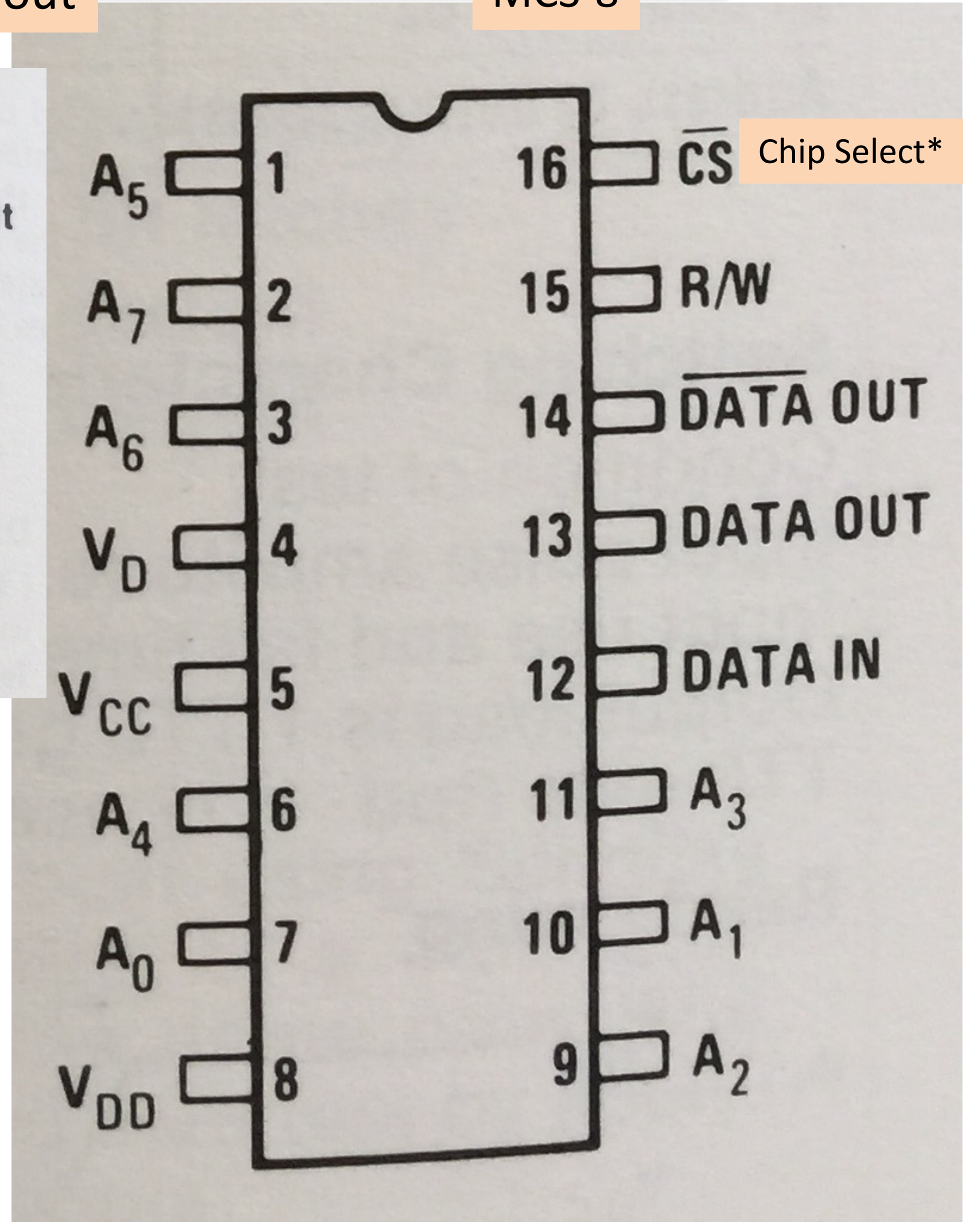
i1101A 256x1 SRAM

Pinout

MCS-8

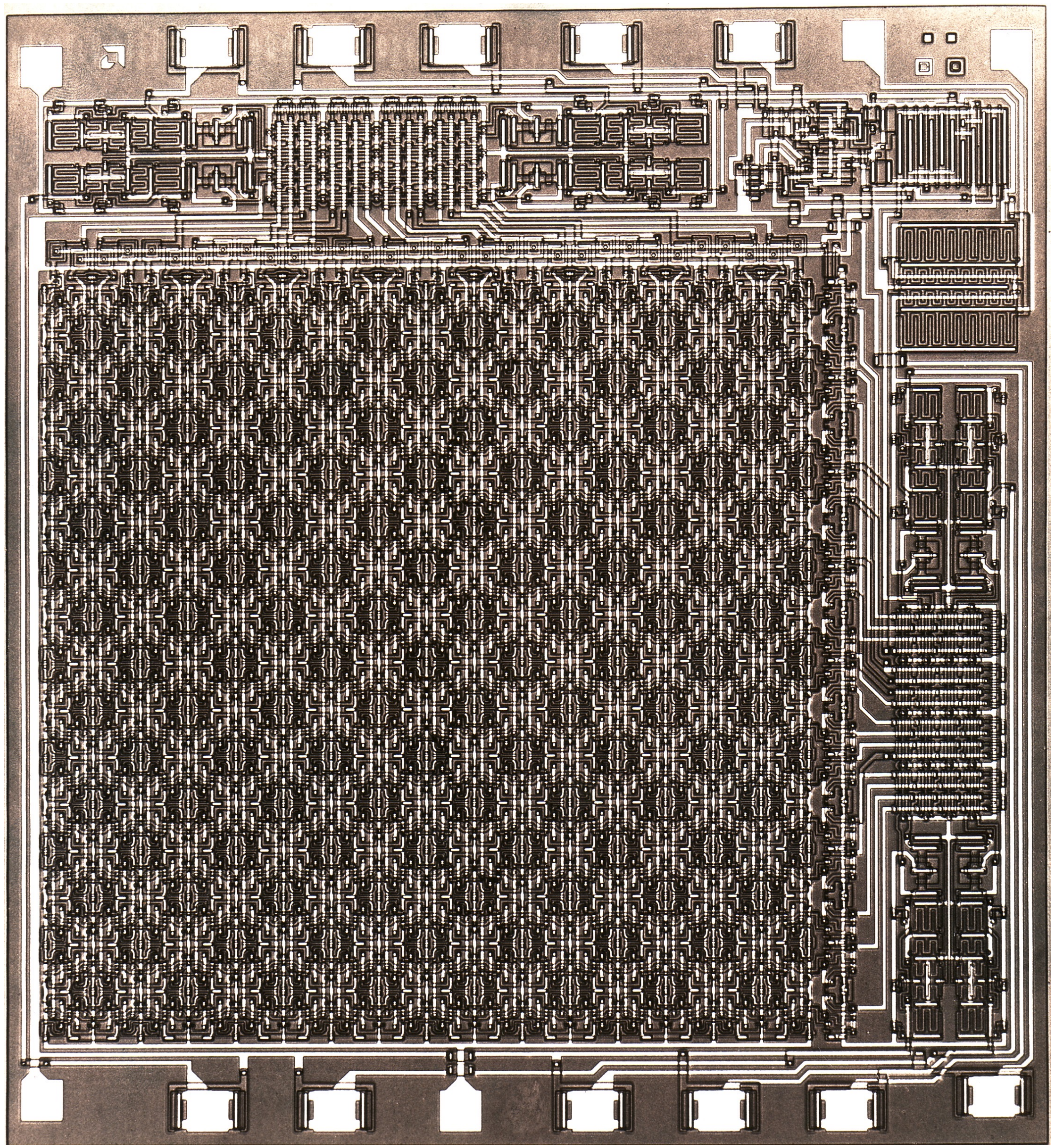


- Access time below 750 ns typically, 1.0 µsec maximum — 1101A1; 1.5 µsec maximum — 1101A: over temperature
- Low power dissipation—typically less than 1.5 mW/bit during access
- Low power standby mode
- Directly DTL and TTL compatible
- OR-Tie capability
- Simple memory expansion—chip-select input lead
- Fully decoded—on-chip address decode and sense
- Inputs protected against static charge
- Ceramic and plastic package
- Silicon gate MOS technology



Chip Select*

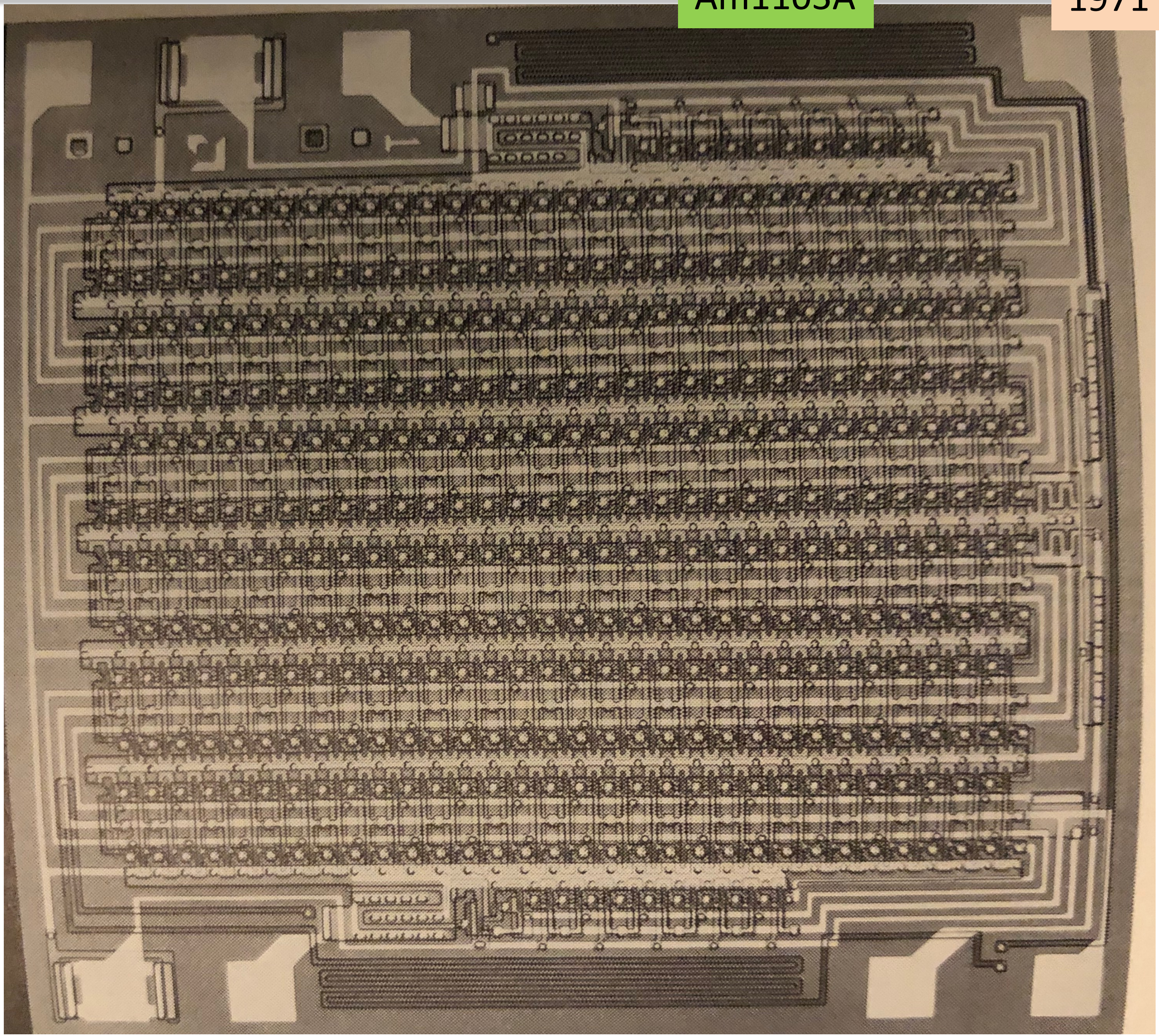
Am1101A 256x1 SRAM



Am1103 1Kx1 DRAM

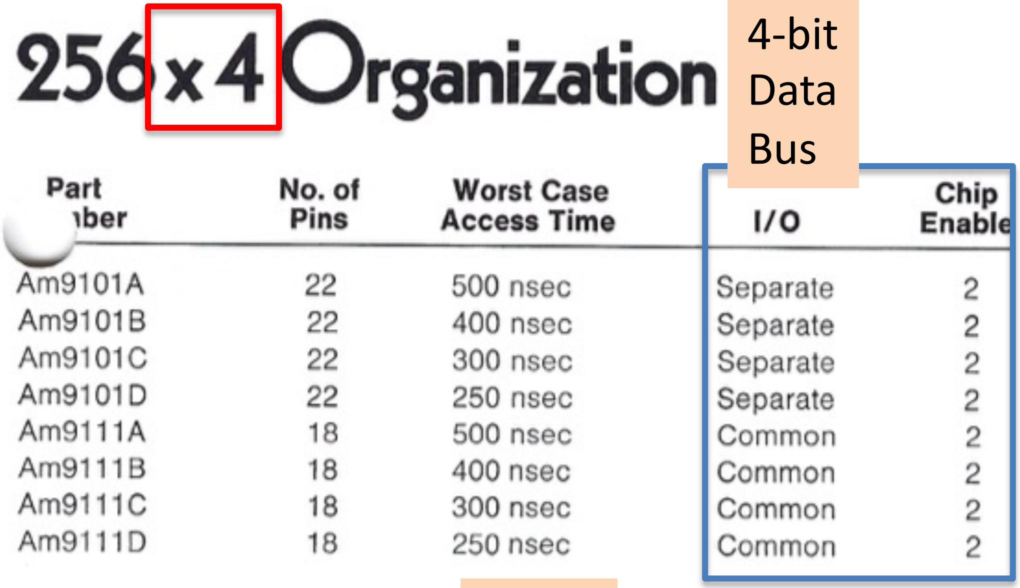
Am1103A

1971



AMD 1K DRAM’s

DRAM



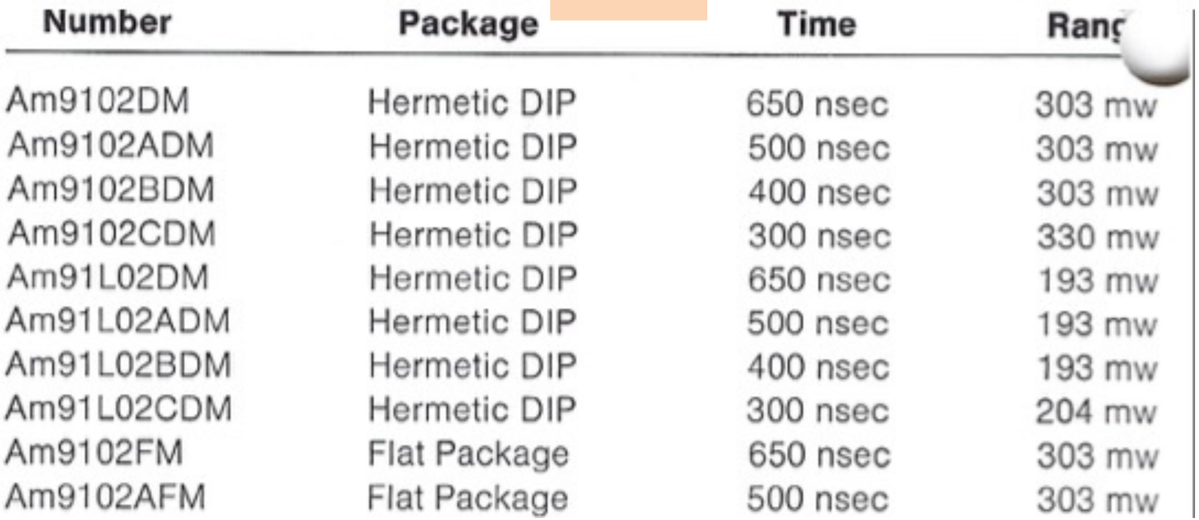
256x4 Organization

4-bit Data Bus

Chip Select*

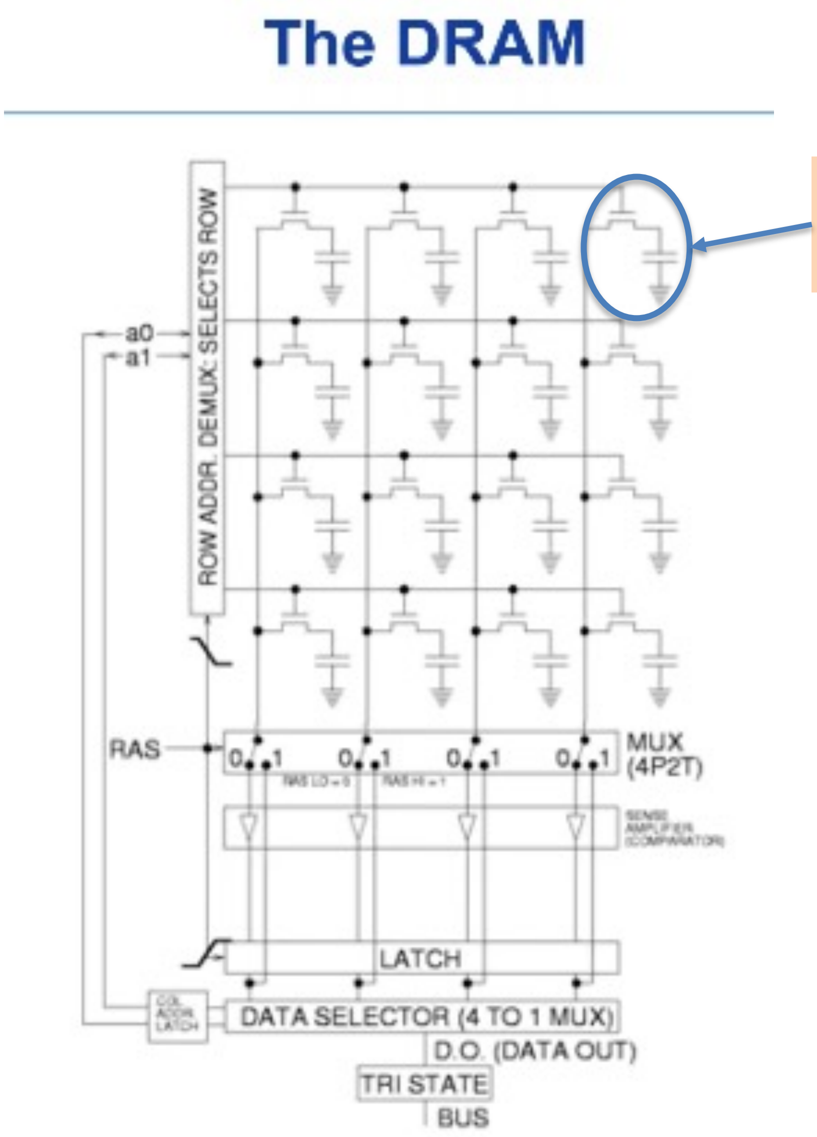
Part ber	No. of Pins	Worst Case Access Time	I/O	Chip Enable
Am9101A	22	500 nsec	Separate	2
Am9101B	22	400 nsec	Separate	2
Am9101C	22	300 nsec	Separate	2
Am9101D	22	250 nsec	Separate	2
Am9111A	18	500 nsec	Common	2
Am9111B	18	400 nsec	Common	2
Am9111C	18	300 nsec	Common	2
Am9111D	18	250 nsec	Common	2

1K x1



Number	Package	Time	Rang
Am9102DM	Hermetic DIP	650 nsec	303 mw
Am9102ADM	Hermetic DIP	500 nsec	303 mw
Am9102BDM	Hermetic DIP	400 nsec	303 mw
Am9102CDM	Hermetic DIP	300 nsec	330 mw
Am91L02DM	Hermetic DIP	650 nsec	193 mw
Am91L02ADM	Hermetic DIP	500 nsec	193 mw
Am91L02BDM	Hermetic DIP	400 nsec	193 mw
Am91L02CDM	Hermetic DIP	300 nsec	204 mw
Am9102FM	Flat Package	650 nsec	303 mw
Am9102AFM	Flat Package	500 nsec	303 mw

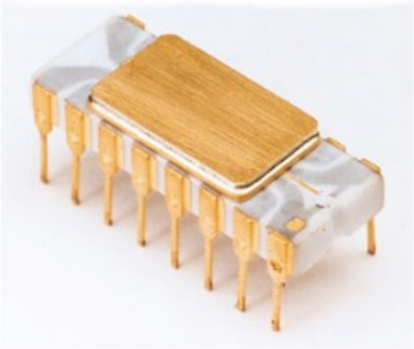
DRAM Schematic



- ❖ 1 transistor
- ❖ 1 cap (parasitic)

MPU/MCU Museum

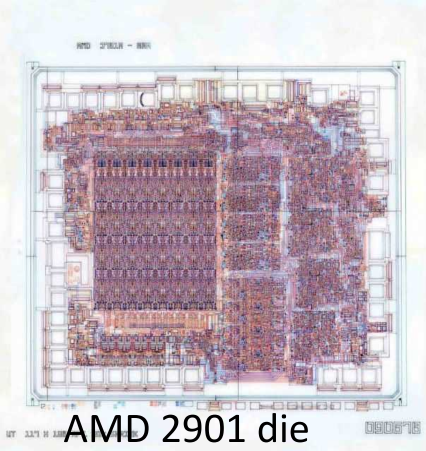
[Nostalgia Dept.]
Intel 4004 is announced, November 15, 1971



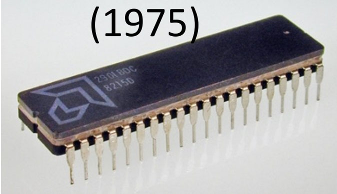
The building-block 4004 CPU held 2300 transistors. The microprocessor, the size of a little fingernail, delivered the same computing power as the first electronic computer built in 1946, which, in contrast, filled a room.

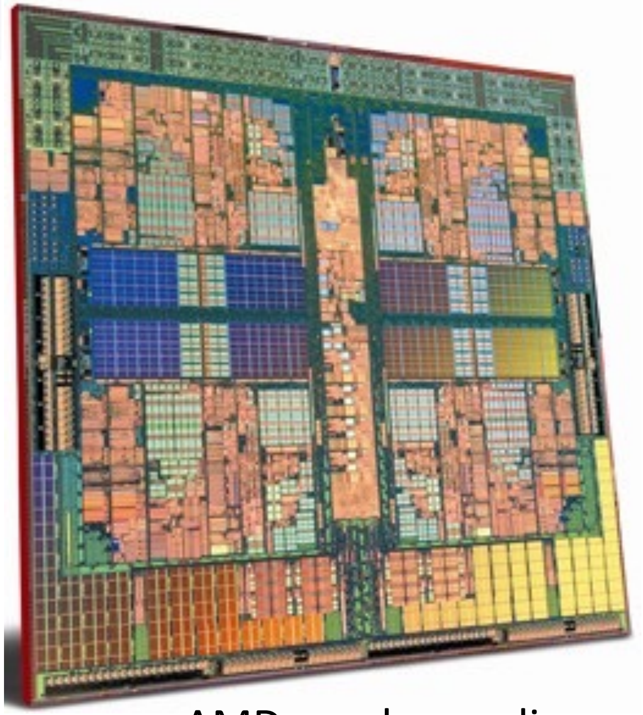
From: EDN Nov 20, 2015
Intel 4004 is announced, November 15, 1971

The venerable 16-pin side-brazed DIP.
(Click image to view full size)

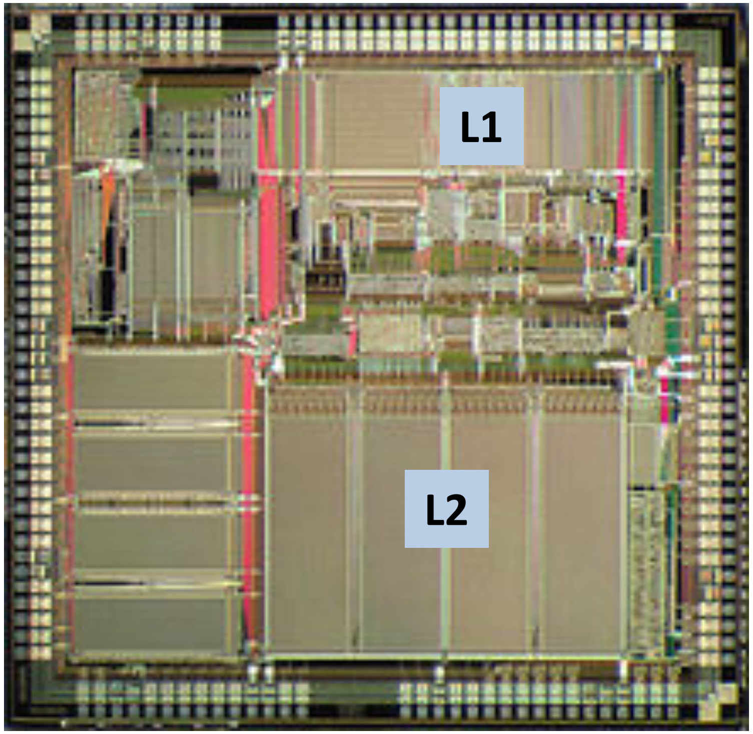


AMD 2901 die
(1975)



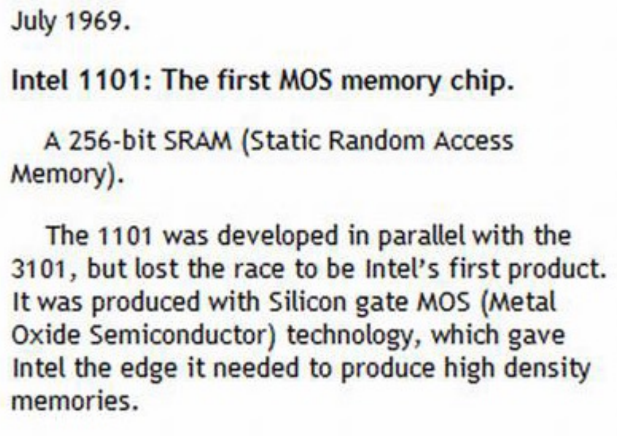


AMD quad-core die



ARM610 sgl-core die

Memory Museum

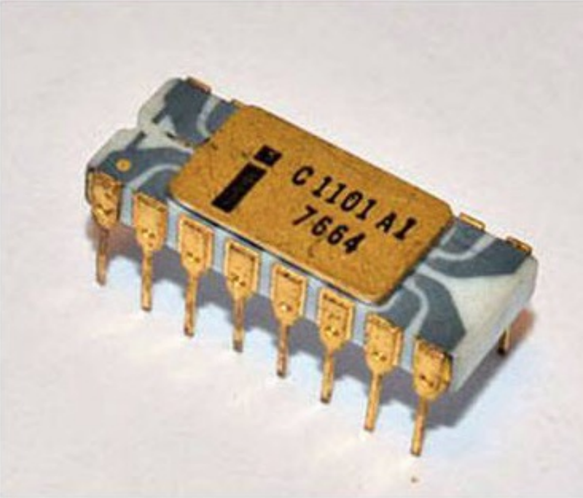


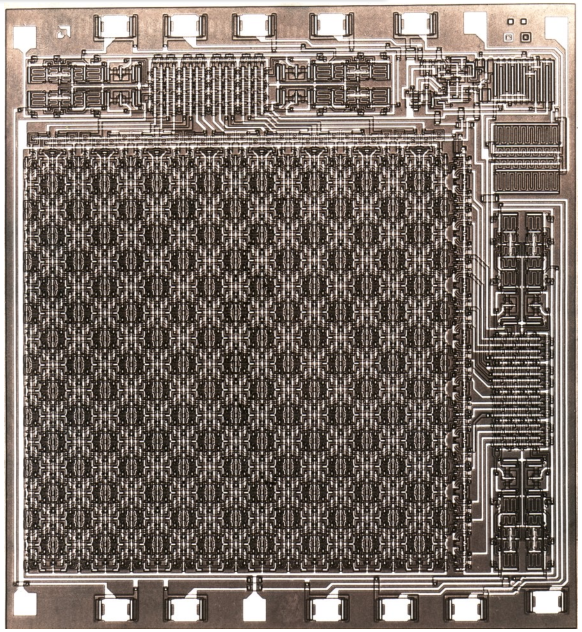
July 1969.

Intel 1101: The first MOS memory chip.

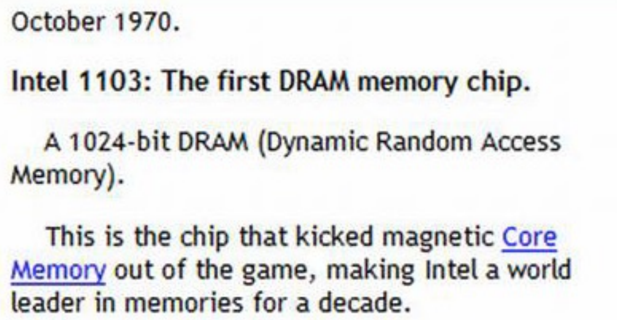
A 256-bit SRAM (Static Random Access Memory).

The 1101 was developed in parallel with the 3101, but lost the race to be Intel's first product. It was produced with Silicon gate MOS (Metal Oxide Semiconductor) technology, which gave Intel the edge it needed to produce high density memories.





Am1101A

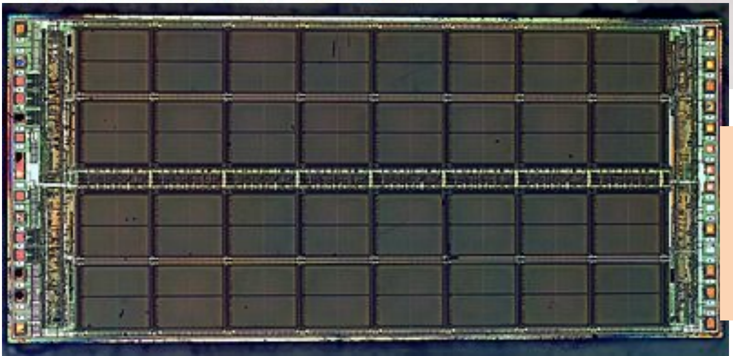


October 1970.

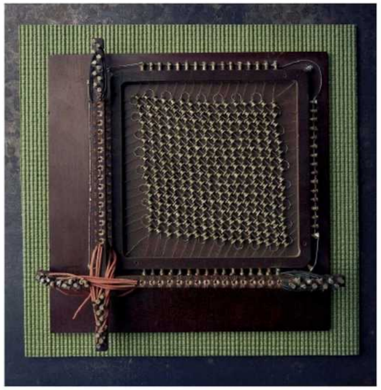
Intel 1103: The first DRAM memory chip.

A 1024-bit DRAM (Dynamic Random Access Memory).

This is the chip that kicked magnetic Core Memory out of the game, making Intel a world leader in memories for a decade.

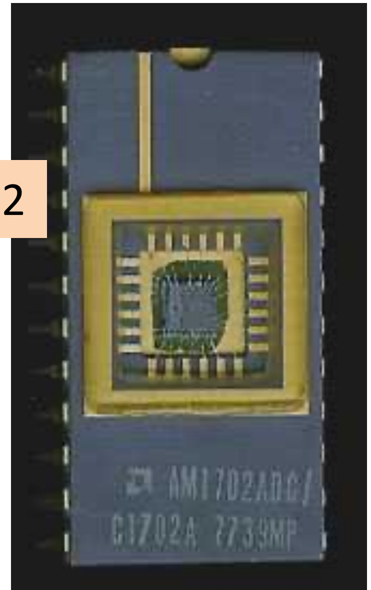


MT4C
1Mb
DRAM



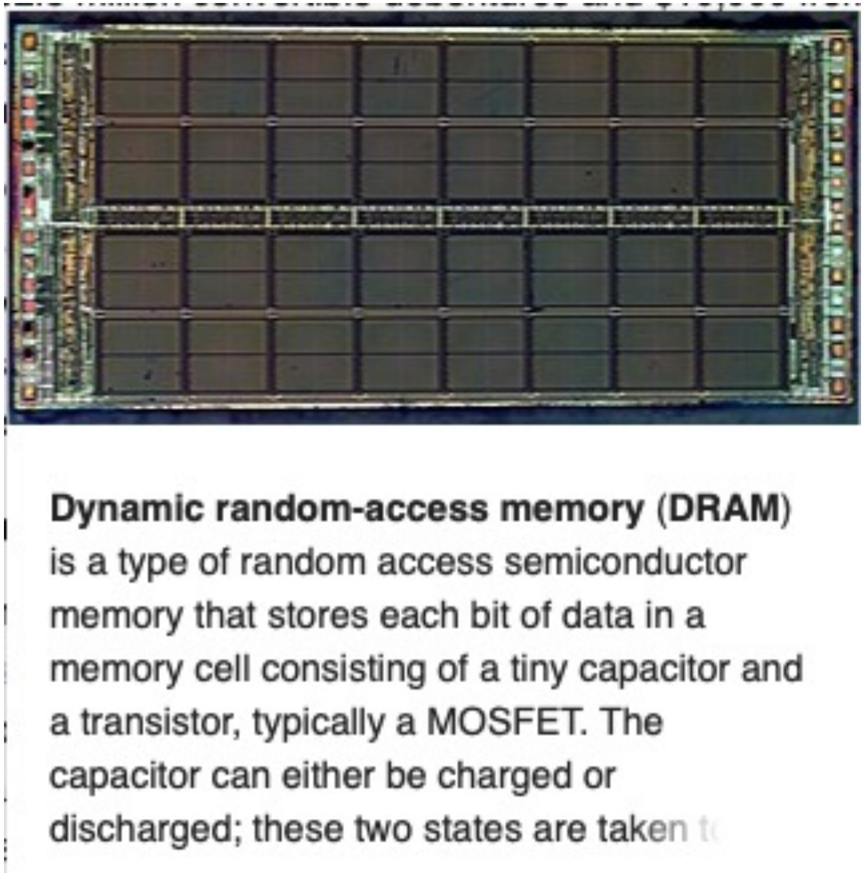
256-bit magnetic core memory (c. 1952) Slow data retrieval and storage speeds limited the utility of early computers. RCA researcher Jan Rajchman's solution was a memory array consisting of a wire matrix with doughnut-shaped magnetic cores at each intersection. By applying a current to a given set of horizontal and vertical wires, you could select a specific core and quickly change the direction of its magnetic field.

Am1702



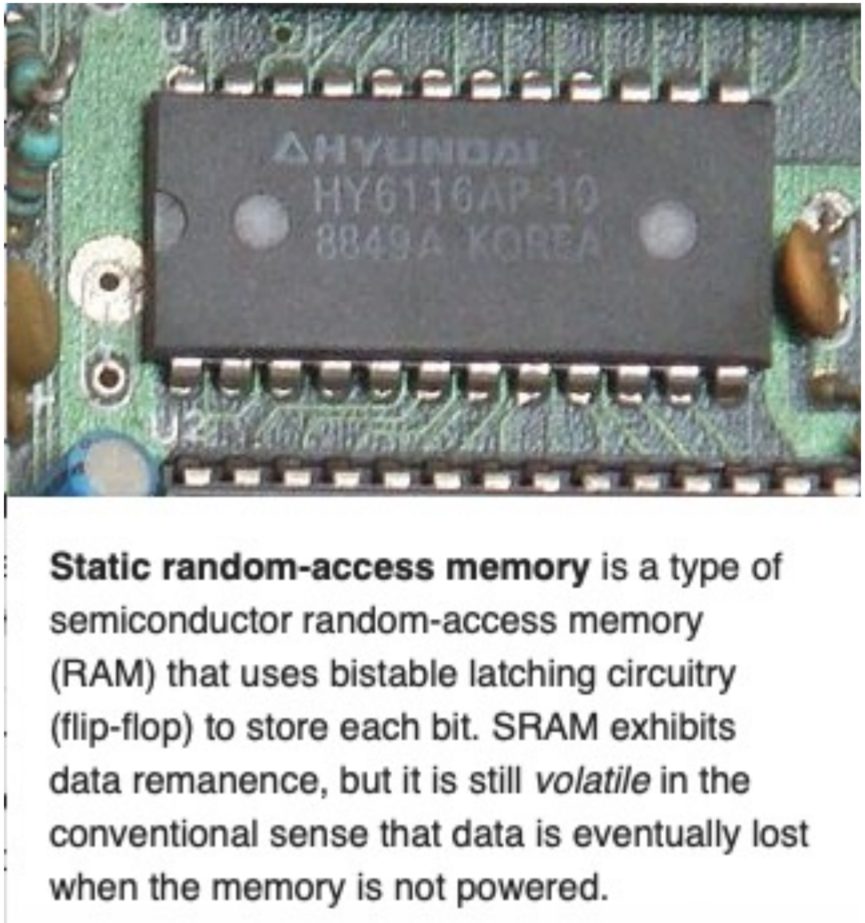
Memory Chips

DRAM	1T



Dynamic random-access memory (DRAM) is a type of random access semiconductor memory that stores each bit of data in a memory cell consisting of a tiny capacitor and a transistor, typically a MOSFET. The capacitor can either be charged or discharged; these two states are taken t

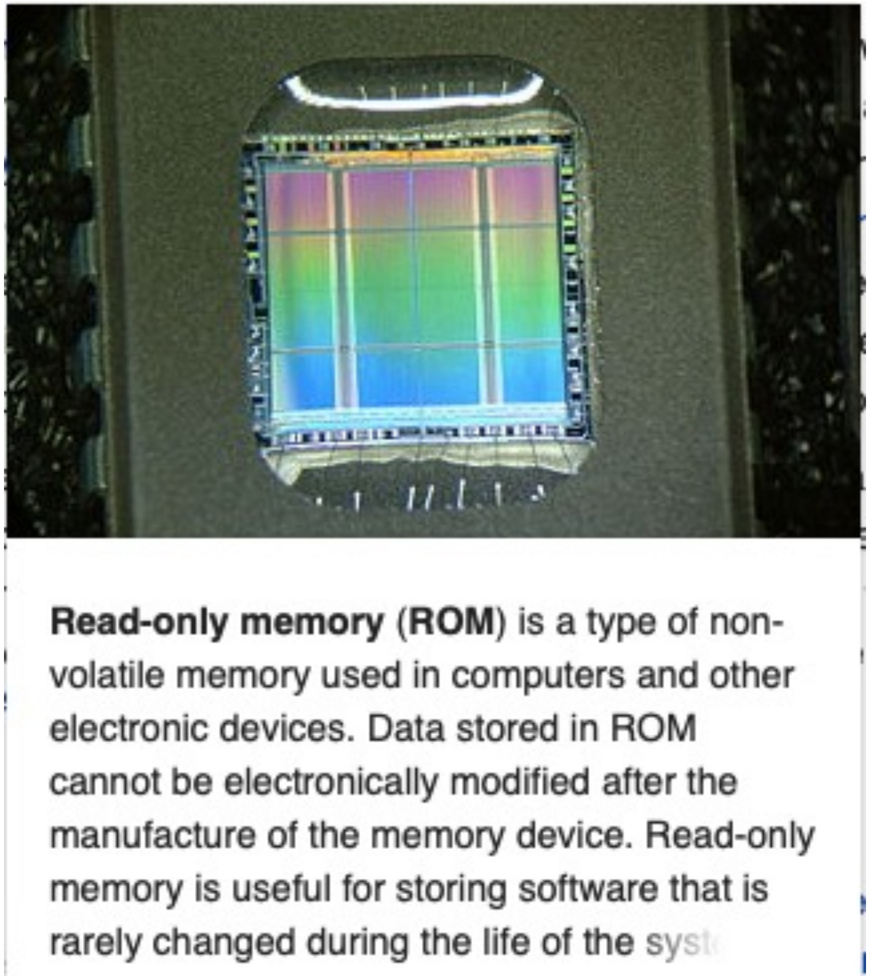
SRAM	4T



Static random-access memory is a type of semiconductor random-access memory (RAM) that uses bistable latching circuitry (flip-flop) to store each bit. SRAM exhibits data remanence, but it is still *volatile* in the conventional sense that data is eventually lost when the memory is not powered.

Memory Chips

ROM

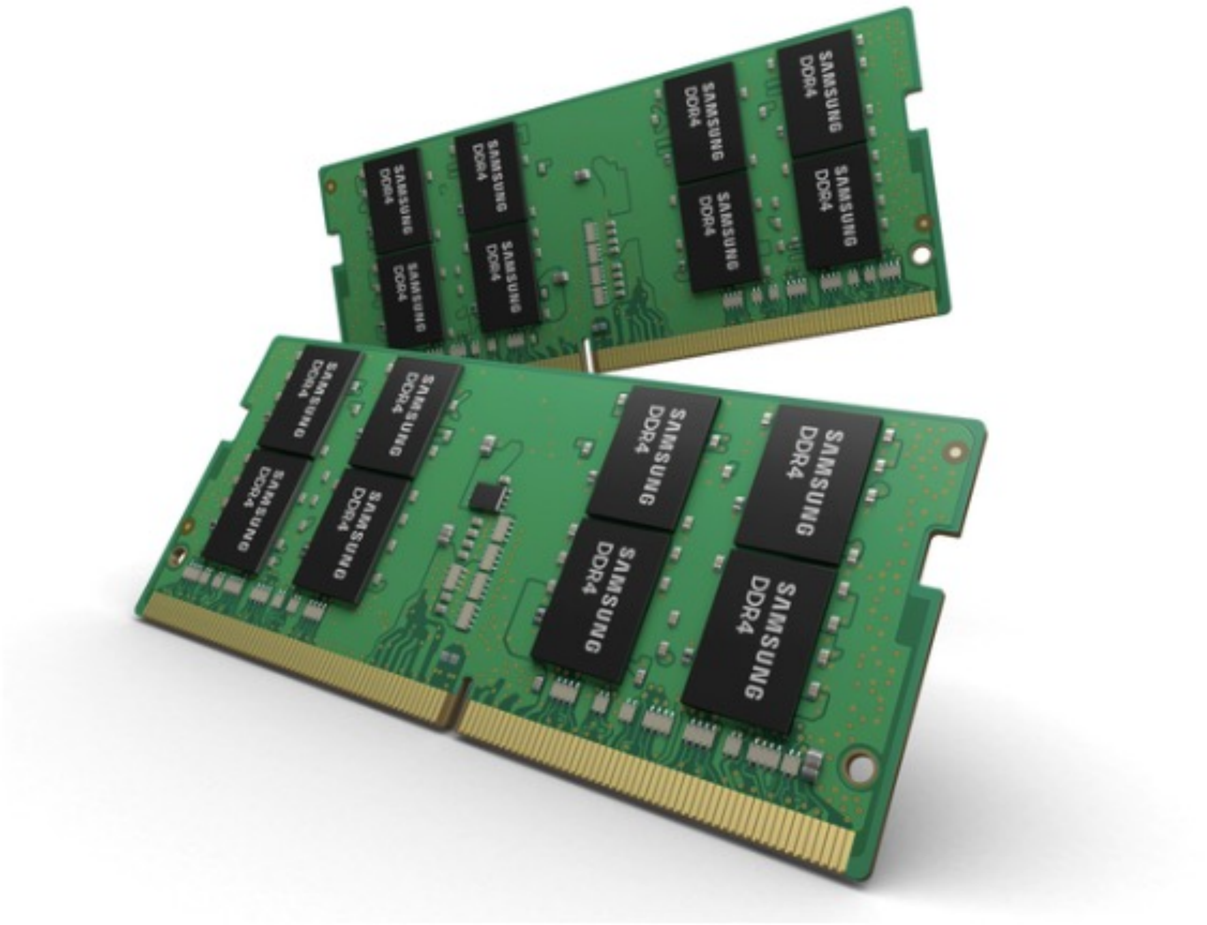


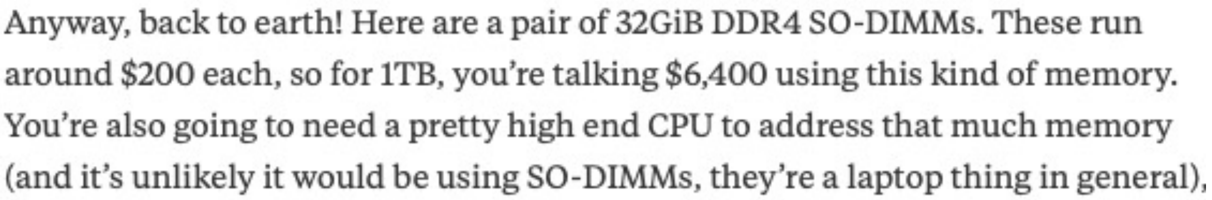
Read-only memory (**ROM**) is a type of non-volatile memory used in computers and other electronic devices. Data stored in ROM cannot be electronically modified after the manufacture of the memory device. Read-only memory is useful for storing software that is rarely changed during the life of the syst

- ❖ROM (masked)
- ❖PROM
- ❖EPROM
- ❖EEPROM
- ❖**Flash E^2**

Modern Memory

32GiB DRAM

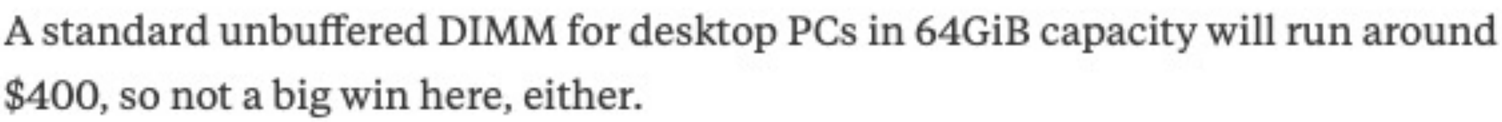




Anyway, back to earth! Here are a pair of 32GiB DDR4 SO-DIMMs. These run around $200 each, so for 1TB, you're talking $6,400 using this kind of memory. You're also going to need a pretty high end CPU to address that much memory (and it's unlikely it would be using SO-DIMMs, they're a laptop thing in general),

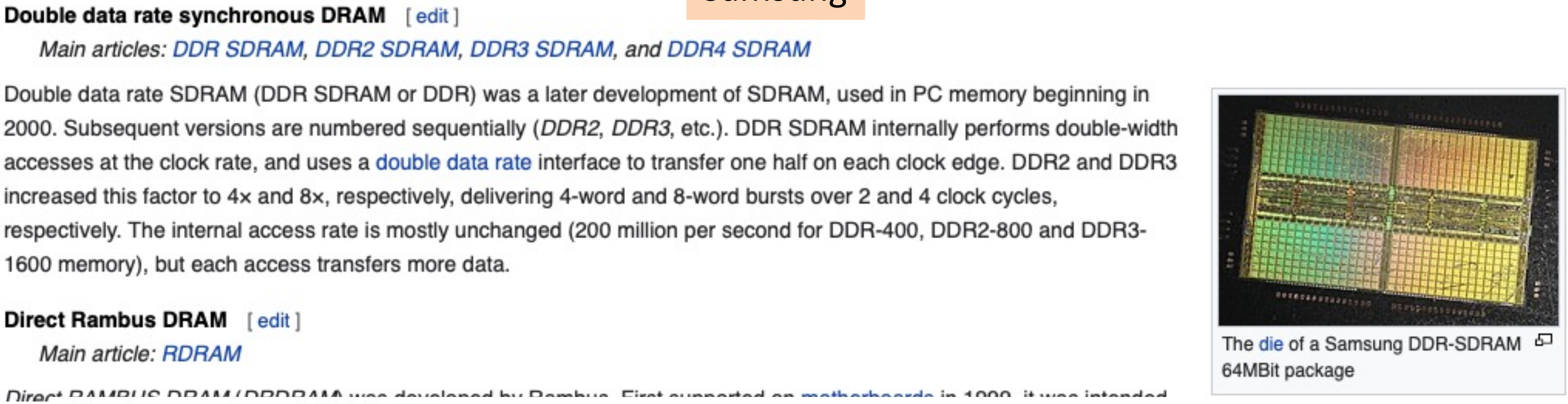
Modern Memory

64GiB DRAM


A standard unbuffered DIMM for desktop PCs in 64GiB capacity will run around $400, so not a big win here, either.

DDR SDRAM

Samsung



Double data rate synchronous DRAM [edit]

Main articles: DDR SDRAM, DDR2 SDRAM, DDR3 SDRAM, and DDR4 SDRAM

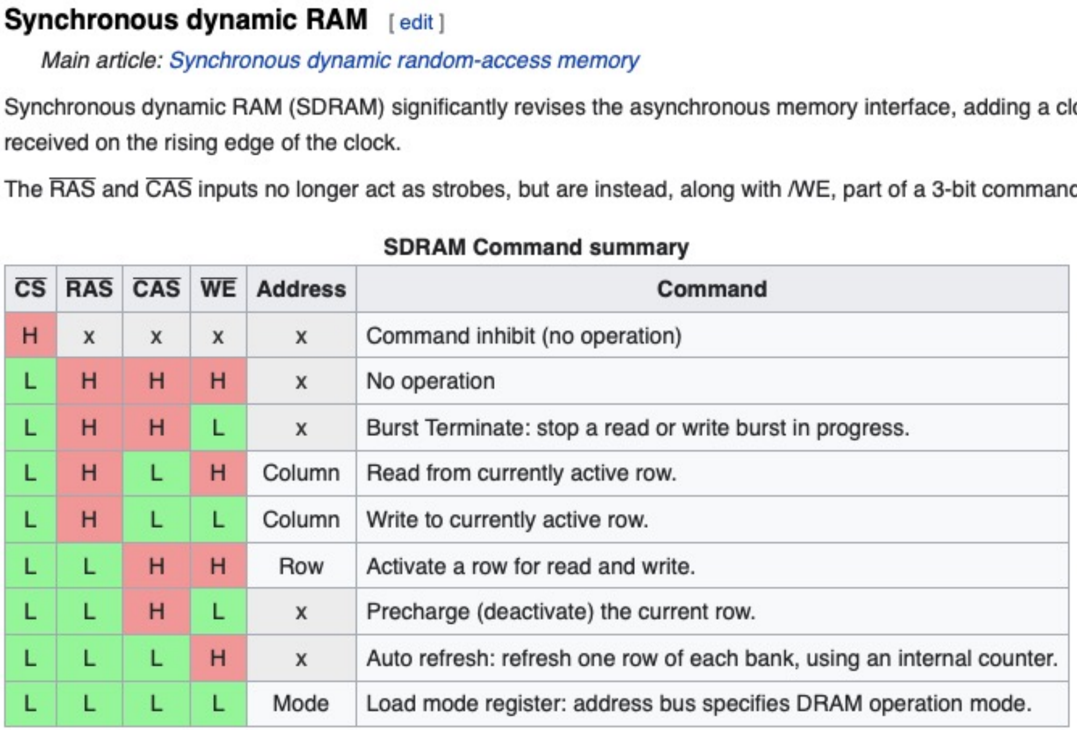
Double data rate SDRAM (DDR SDRAM or DDR) was a later development of SDRAM, used in PC memory beginning in 2000. Subsequent versions are numbered sequentially (*DDR2*, *DDR3*, etc.). DDR SDRAM internally performs double-width accesses at the clock rate, and uses a double data rate interface to transfer one half on each clock edge. DDR2 and DDR3 increased this factor to 4× and 8×, respectively, delivering 4-word and 8-word bursts over 2 and 4 clock cycles, respectively. The internal access rate is mostly unchanged (200 million per second for DDR-400, DDR2-800 and DDR3-1600 memory), but each access transfers more data.

The die of a Samsung DDR-SDRAM 64MBit package

Direct Rambus DRAM [edit]

Main article: RDRAM

Direct RAMBUS DRAM (DRDRAM) was developed by Rambus. First supported on motherboards in 1999, it was intended



Synchronous dynamic RAM [edit]

Main article: Synchronous dynamic random-access memory

Synchronous dynamic RAM (SDRAM) significantly revises the asynchronous memory interface, adding a clock (and a clock enable) line. All other signals are received on the rising edge of the clock.

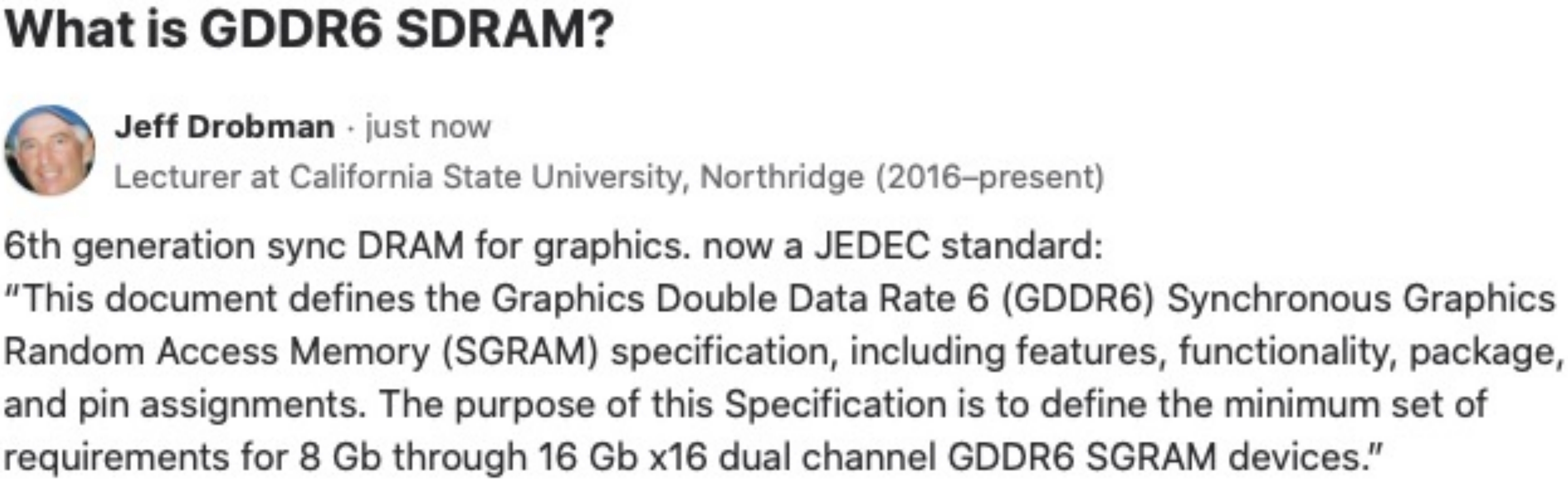
The $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$ inputs no longer act as strobes, but are instead, along with /WE, part of a 3-bit command:

SDRAM Command summary

$\overline{\text{CS}}$	$\overline{\text{RAS}}$	$\overline{\text{CAS}}$	$\overline{\text{WE}}$	Address	Command
H	x	x	x	x	Command inhibit (no operation)
L	H	H	H	x	No operation
L	H	H	L	x	Burst Terminate: stop a read or write burst in progress.
L	H	L	H	Column	Read from currently active row.
L	H	L	L	Column	Write to currently active row.
L	L	H	H	Row	Activate a row for read and write.
L	L	H	L	x	Precharge (deactivate) the current row.
L	L	L	H	x	Auto refresh: refresh one row of each bank, using an internal counter.
L	L	L	L	Mode	Load mode register: address bus specifies DRAM operation mode.

DDR SDRAM

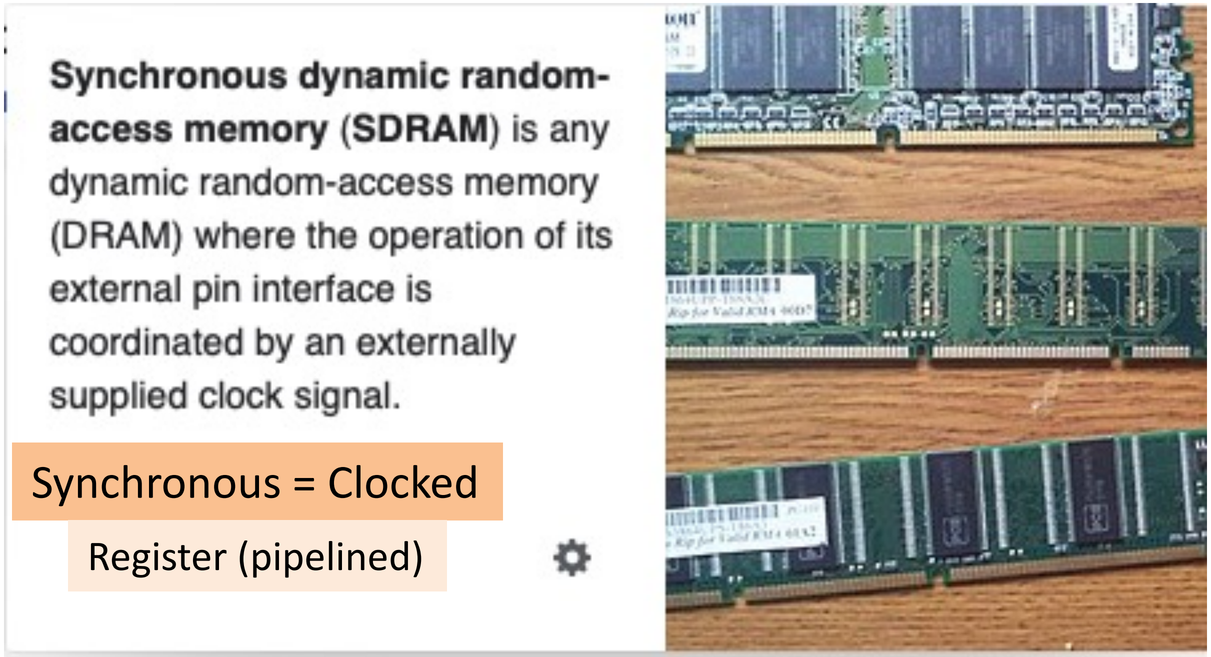
Synchronous **DRAM** | Double Data Rate



What is GDDR6 SDRAM?

Jeff Drobman · just now
Lecturer at California State University, Northridge (2016–present)

6th generation sync DRAM for graphics. now a JEDEC standard:
"This document defines the Graphics Double Data Rate 6 (GDDR6) Synchronous Graphics Random Access Memory (SGRAM) specification, including features, functionality, package, and pin assignments. The purpose of this Specification is to define the minimum set of requirements for 8 Gb through 16 Gb x16 dual channel GDDR6 SGRAM devices."



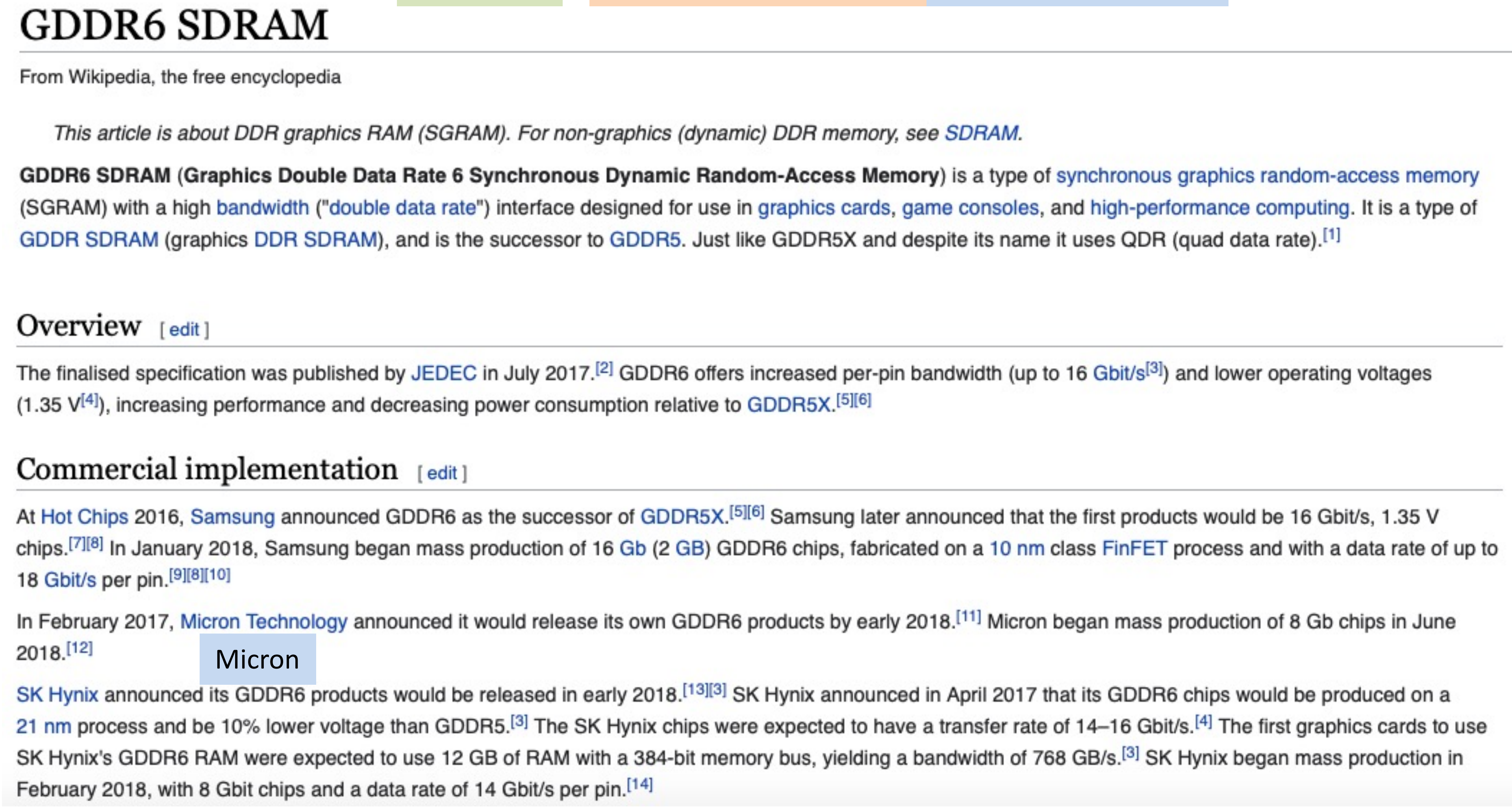
Synchronous dynamic random-access memory (**SDRAM**) is any dynamic random-access memory (DRAM) where the operation of its external pin interface is coordinated by an externally supplied clock signal.

Synchronous = Clocked

Register (pipelined)

DDR SDRAM

Graphics — ***Synchronous*** DRAM | Double Data Rate



GDDR6 SDRAM

From Wikipedia, the free encyclopedia

This article is about DDR graphics RAM (SGRAM). For non-graphics (dynamic) DDR memory, see SDRAM.

GDDR6 SDRAM (**Graphics Double Data Rate 6 Synchronous Dynamic Random-Access Memory**) is a type of synchronous graphics random-access memory (SGRAM) with a high bandwidth ("double data rate") interface designed for use in graphics cards, game consoles, and high-performance computing. It is a type of GDDR SDRAM (graphics DDR SDRAM), and is the successor to GDDR5. Just like GDDR5X and despite its name it uses QDR (quad data rate).[1]

Overview [edit]

The finalised specification was published by JEDEC in July 2017.[2] GDDR6 offers increased per-pin bandwidth (up to 16 Gbit/s[3]) and lower operating voltages (1.35 V[4]), increasing performance and decreasing power consumption relative to GDDR5X.[5][6]

Commercial implementation [edit]

At Hot Chips 2016, Samsung announced GDDR6 as the successor of GDDR5X.[5][6] Samsung later announced that the first products would be 16 Gbit/s, 1.35 V chips.[7][8] In January 2018, Samsung began mass production of 16 Gb (2 GB) GDDR6 chips, fabricated on a 10 nm class FinFET process and with a data rate of up to 18 Gbit/s per pin.[9][8][10]

In February 2017, Micron Technology announced it would release its own GDDR6 products by early 2018.[11] Micron began mass production of 8 Gb chips in June 2018.[12]

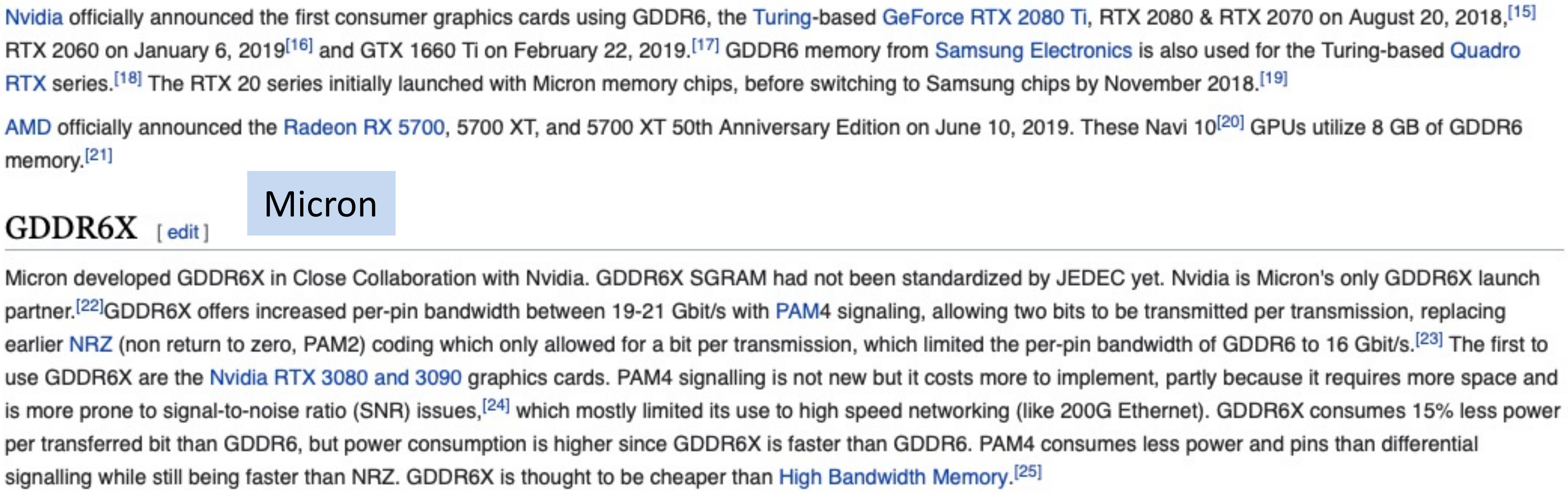
Micron

SK Hynix announced its GDDR6 products would be released in early 2018.[13][3] SK Hynix announced in April 2017 that its GDDR6 chips would be produced on a 21 nm process and be 10% lower voltage than GDDR5.[3] The SK Hynix chips were expected to have a transfer rate of 14–16 Gbit/s.[4] The first graphics cards to use SK Hynix's GDDR6 RAM were expected to use 12 GB of RAM with a 384-bit memory bus, yielding a bandwidth of 768 GB/s.[3] SK Hynix began mass production in February 2018, with 8 Gbit chips and a data rate of 14 Gbit/s per pin.[14]

DDR SDRAM

Graphics | ***Synchronous*** DRAM | Double Data Rate

Nvidia + AMD



Nvidia officially announced the first consumer graphics cards using GDDR6, the Turing-based GeForce RTX 2080 Ti, RTX 2080 & RTX 2070 on August 20, 2018,[15] RTX 2060 on January 6, 2019[16] and GTX 1660 Ti on February 22, 2019.[17] GDDR6 memory from Samsung Electronics is also used for the Turing-based Quadro RTX series.[18] The RTX 20 series initially launched with Micron memory chips, before switching to Samsung chips by November 2018.[19]

AMD officially announced the Radeon RX 5700, 5700 XT, and 5700 XT 50th Anniversary Edition on June 10, 2019. These Navi 10[20] GPUs utilize 8 GB of GDDR6 memory.[21]

Micron

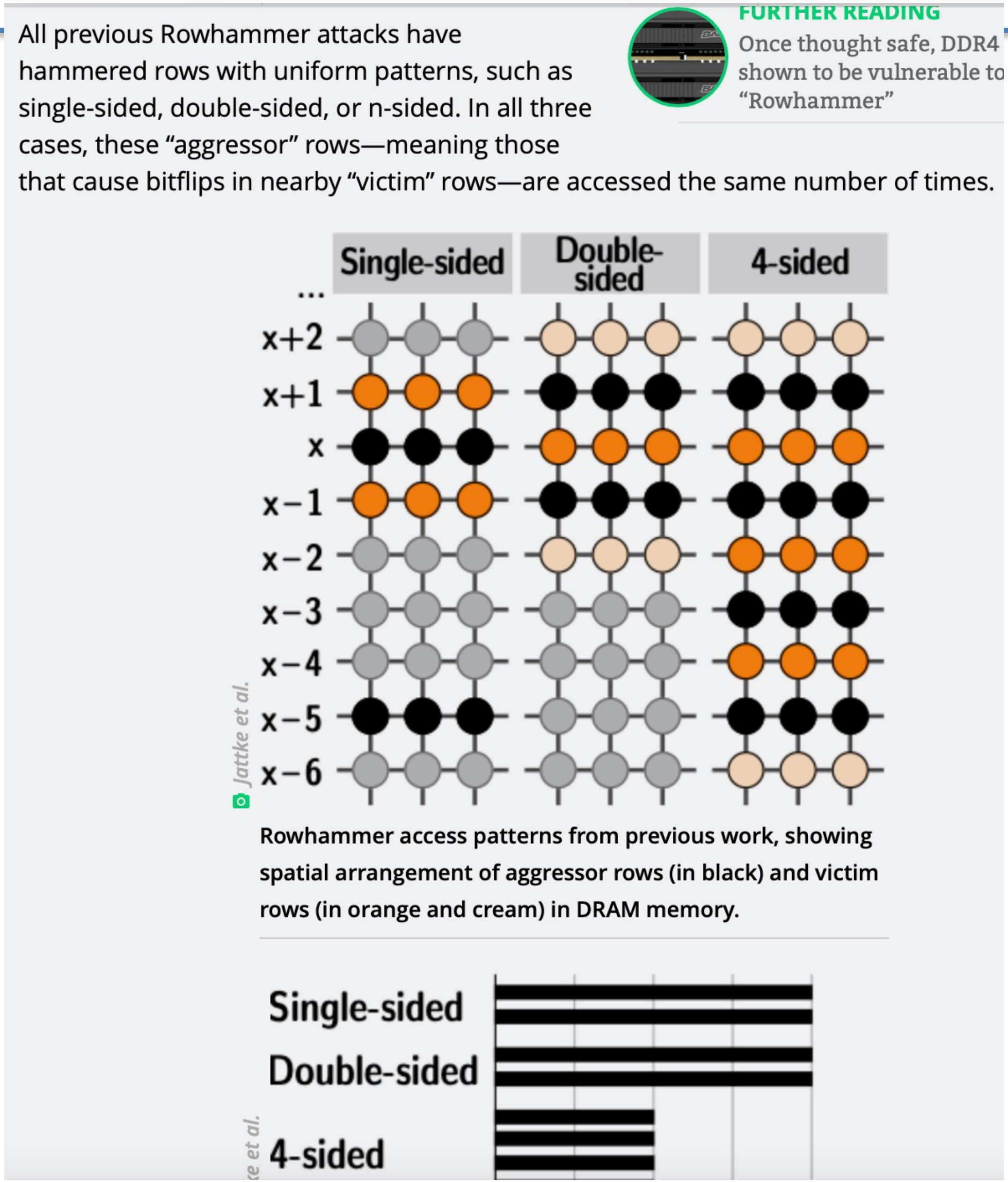
GDDR6X [edit]

Micron developed GDDR6X in Close Collaboration with Nvidia. GDDR6X SGRAM had not been standardized by JEDEC yet. Nvidia is Micron's only GDDR6X launch partner.[22]GDDR6X offers increased per-pin bandwidth between 19-21 Gbit/s with PAM4 signaling, allowing two bits to be transmitted per transmission, replacing earlier NRZ (non return to zero, PAM2) coding which only allowed for a bit per transmission, which limited the per-pin bandwidth of GDDR6 to 16 Gbit/s.[23] The first to use GDDR6X are the Nvidia RTX 3080 and 3090 graphics cards. PAM4 signalling is not new but it costs more to implement, partly because it requires more space and is more prone to signal-to-noise ratio (SNR) issues,[24] which mostly limited its use to high speed networking (like 200G Ethernet). GDDR6X consumes 15% less power per transferred bit than GDDR6, but power consumption is higher since GDDR6X is faster than GDDR6. PAM4 consumes less power and pins than differential signalling while still being faster than NRZ. GDDR6X is thought to be cheaper than High Bandwidth Memory.[25]

DDR4 Vulnerability

- ❖ Attacks
- ❖ Exploits

➢ ***Rowhammer***

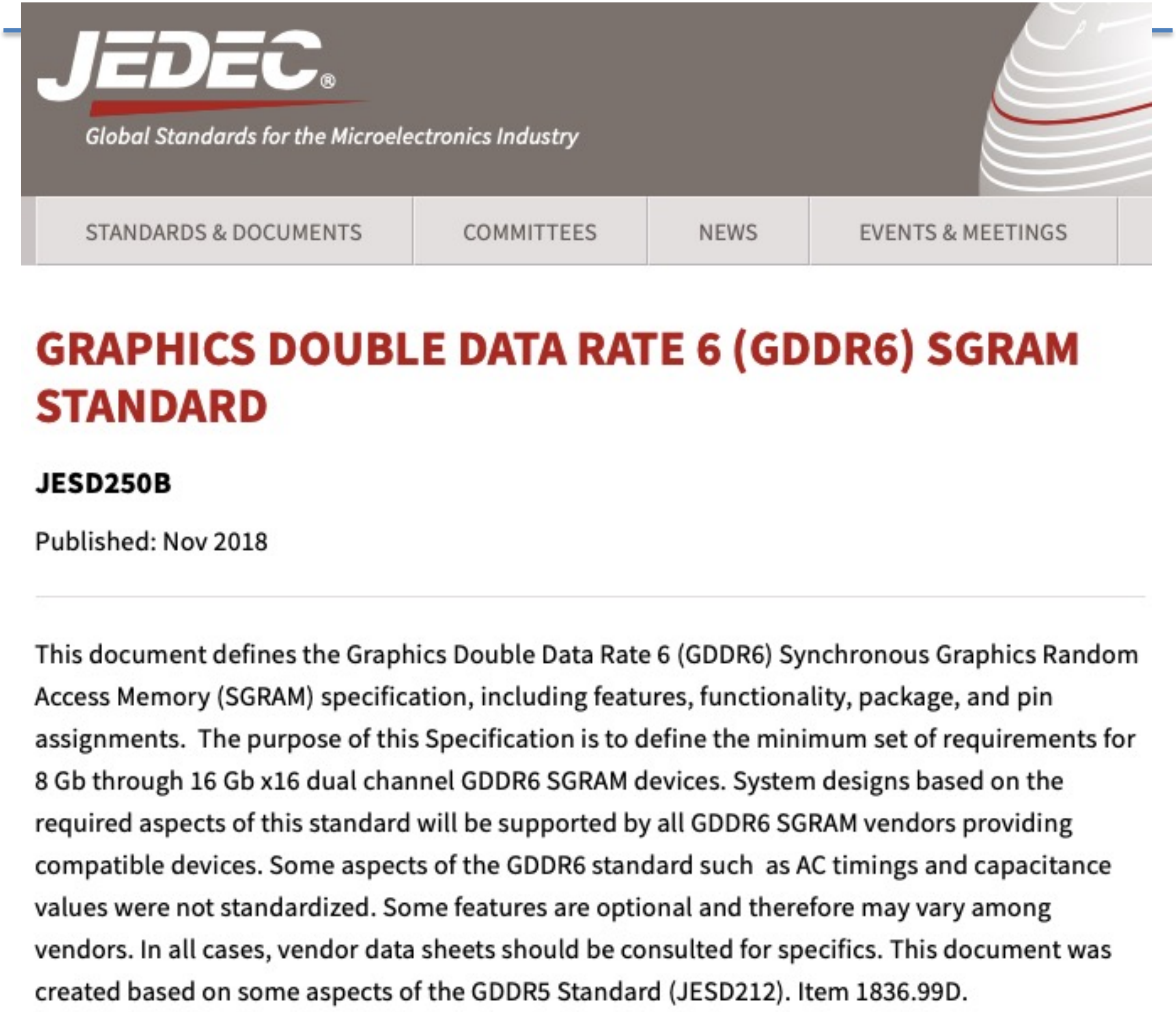


All previous Rowhammer attacks have hammered rows with uniform patterns, such as single-sided, double-sided, or n-sided. In all three cases, these "aggressor" rows—meaning those that cause bitflips in nearby "victim" rows—are accessed the same number of times.

FURTHER READING
Once thought safe, DDR4 shown to be vulnerable to "Rowhammer"

Rowhammer access patterns from previous work, showing spatial arrangement of aggressor rows (in black) and victim rows (in orange and cream) in DRAM memory.

JEDEC (EIA)



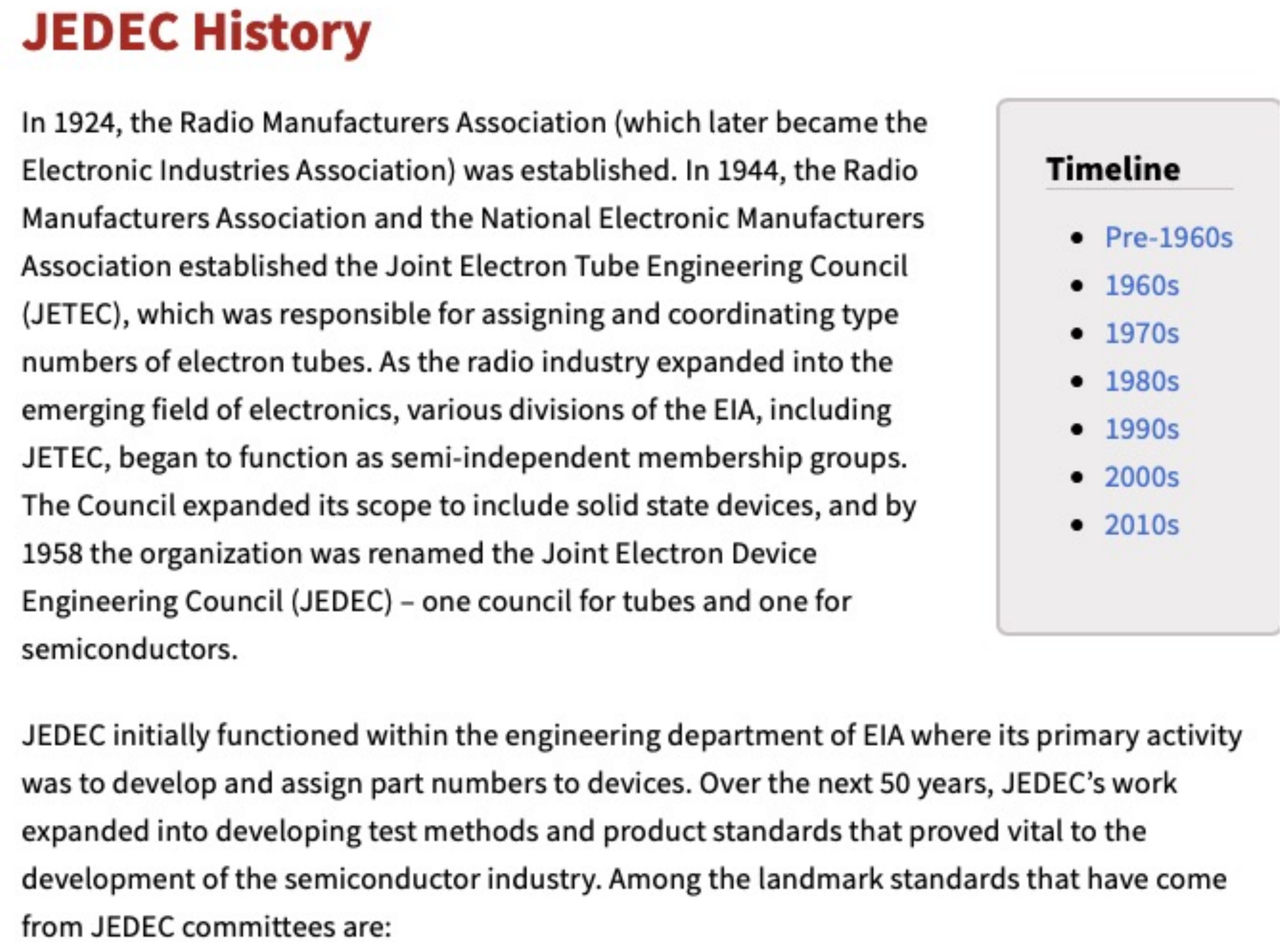
GRAPHICS DOUBLE DATA RATE 6 (GDDR6) SGRAM STANDARD

JESD250B

Published: Nov 2018

This document defines the Graphics Double Data Rate 6 (GDDR6) Synchronous Graphics Random Access Memory (SGRAM) specification, including features, functionality, package, and pin assignments. The purpose of this Specification is to define the minimum set of requirements for 8 Gb through 16 Gb x16 dual channel GDDR6 SGRAM devices. System designs based on the required aspects of this standard will be supported by all GDDR6 SGRAM vendors providing compatible devices. Some aspects of the GDDR6 standard such as AC timings and capacitance values were not standardized. Some features are optional and therefore may vary among vendors. In all cases, vendor data sheets should be consulted for specifics. This document was created based on some aspects of the GDDR5 Standard (JESD212). Item 1836.99D.

JEDEC (EIA)



JEDEC History

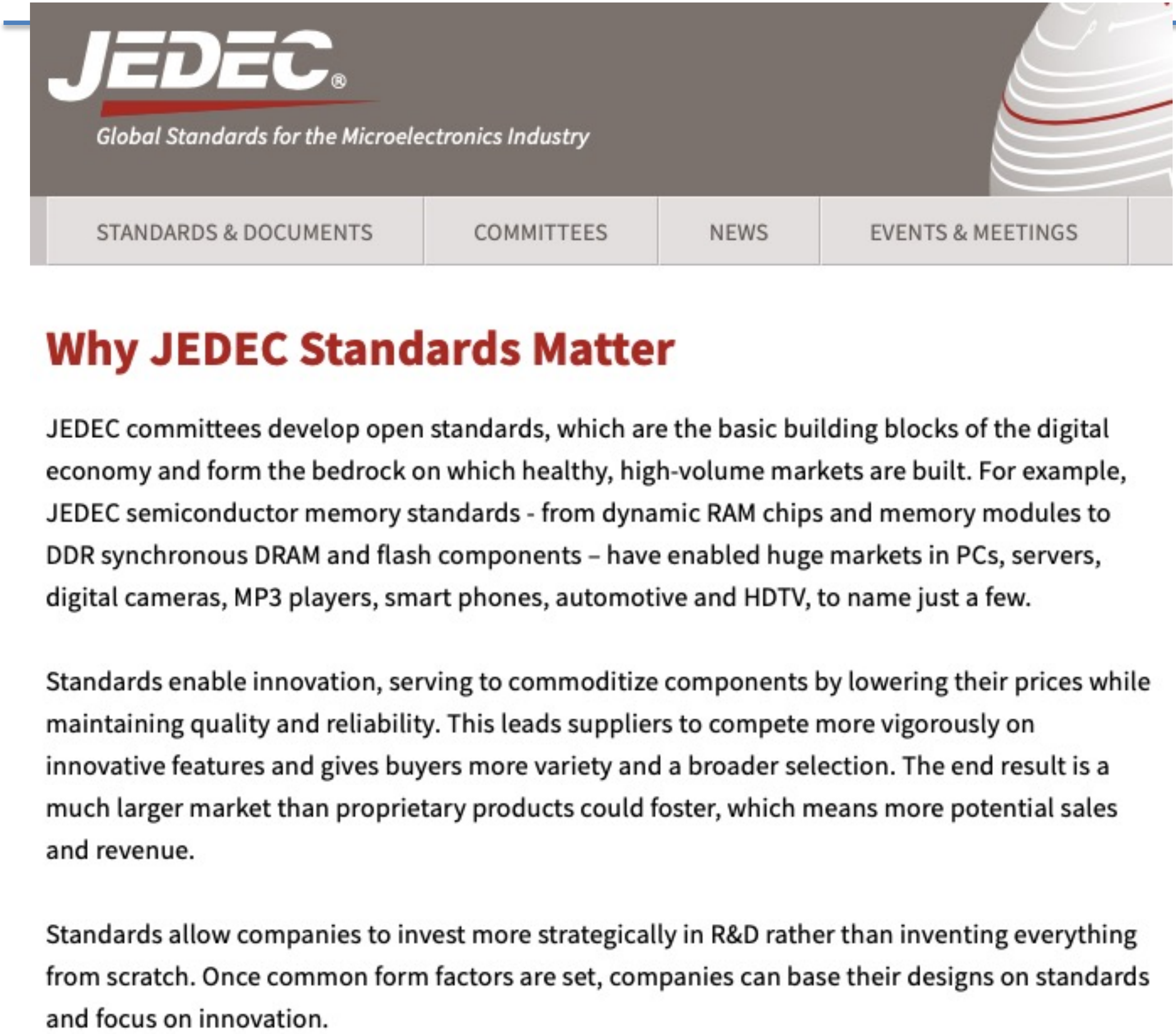
In 1924, the Radio Manufacturers Association (which later became the Electronic Industries Association) was established. In 1944, the Radio Manufacturers Association and the National Electronic Manufacturers Association established the Joint Electron Tube Engineering Council (JETEC), which was responsible for assigning and coordinating type numbers of electron tubes. As the radio industry expanded into the emerging field of electronics, various divisions of the EIA, including JETEC, began to function as semi-independent membership groups. The Council expanded its scope to include solid state devices, and by 1958 the organization was renamed the Joint Electron Device Engineering Council (JEDEC) – one council for tubes and one for semiconductors.

Timeline

- Pre-1960s
- 1960s
- 1970s
- 1980s
- 1990s
- 2000s
- 2010s

JEDEC initially functioned within the engineering department of EIA where its primary activity was to develop and assign part numbers to devices. Over the next 50 years, JEDEC's work expanded into developing test methods and product standards that proved vital to the development of the semiconductor industry. Among the landmark standards that have come from JEDEC committees are:

JEDEC (EIA)



Why JEDEC Standards Matter

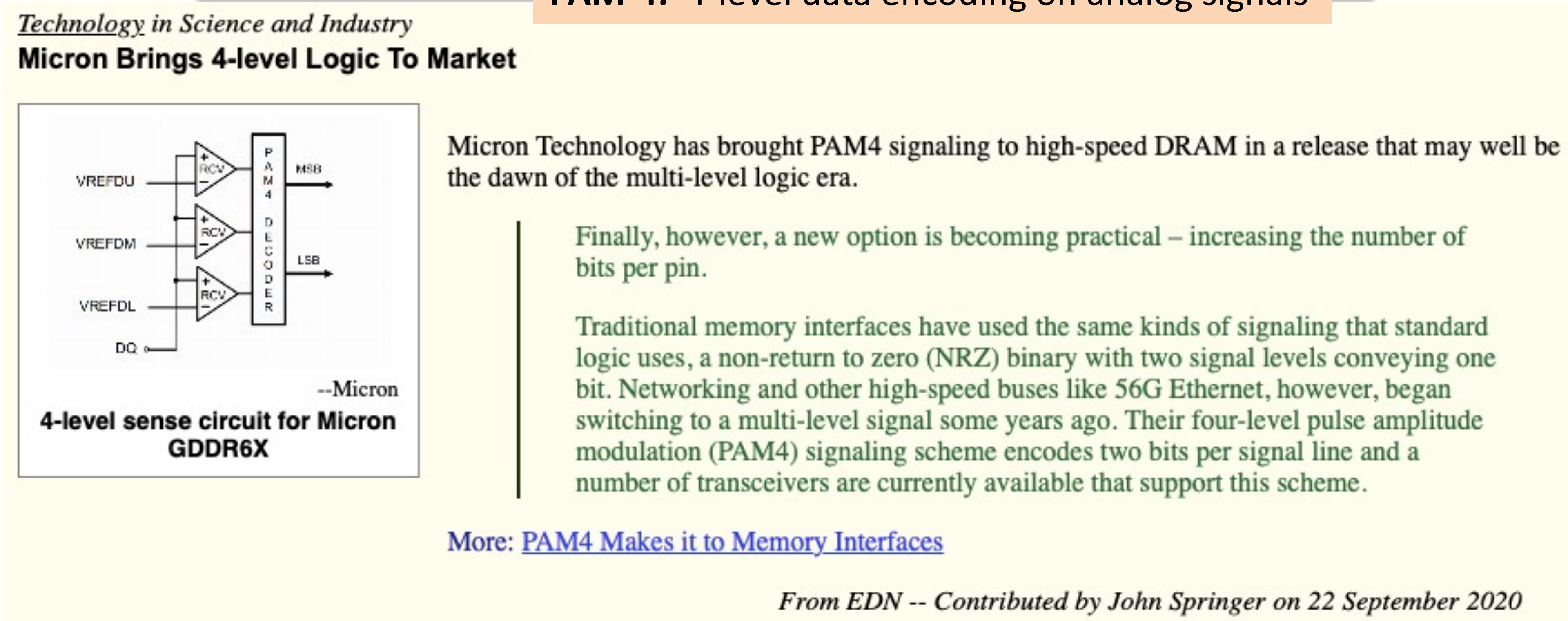
JEDEC committees develop open standards, which are the basic building blocks of the digital economy and form the bedrock on which healthy, high-volume markets are built. For example, JEDEC semiconductor memory standards - from dynamic RAM chips and memory modules to DDR synchronous DRAM and flash components – have enabled huge markets in PCs, servers, digital cameras, MP3 players, smart phones, automotive and HDTV, to name just a few.

Standards enable innovation, serving to commoditize components by lowering their prices while maintaining quality and reliability. This leads suppliers to compete more vigorously on innovative features and gives buyers more variety and a broader selection. The end result is a much larger market than proprietary products could foster, which means more potential sales and revenue.

Standards allow companies to invest more strategically in R&D rather than inventing everything from scratch. Once common form factors are set, companies can base their designs on standards and focus on innovation.

Micron (DRAM) News

PAM-4: 4-level data encoding on analog signals



Technology in Science and Industry

Micron Brings 4-level Logic To Market

--Micron

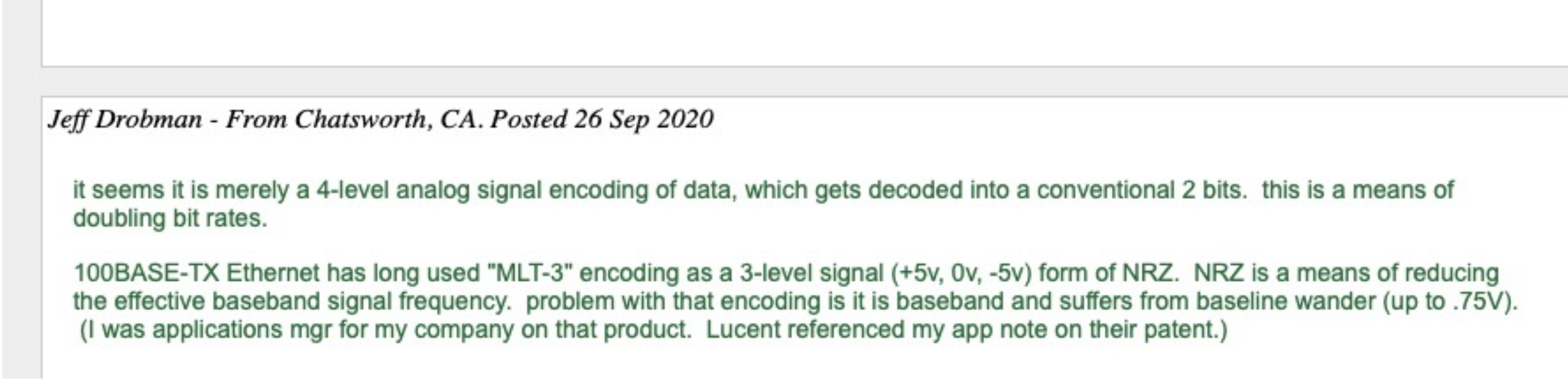
4-level sense circuit for Micron GDDR6X

Micron Technology has brought PAM4 signaling to high-speed DRAM in a release that may well be the dawn of the multi-level logic era.

> Finally, however, a new option is becoming practical – increasing the number of bits per pin.
>
> Traditional memory interfaces have used the same kinds of signaling that standard logic uses, a non-return to zero (NRZ) binary with two signal levels conveying one bit. Networking and other high-speed buses like 56G Ethernet, however, began switching to a multi-level signal some years ago. Their four-level pulse amplitude modulation (PAM4) signaling scheme encodes two bits per signal line and a number of transceivers are currently available that support this scheme.

More: PAM4 Makes it to Memory Interfaces

From EDN -- Contributed by John Springer on 22 September 2020



Jeff Drobman - From Chatsworth, CA. Posted 26 Sep 2020

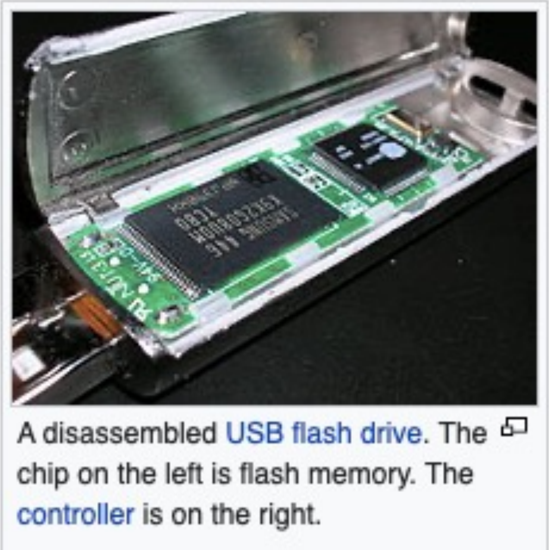
it seems it is merely a 4-level analog signal encoding of data, which gets decoded into a conventional 2 bits. this is a means of doubling bit rates.

100BASE-TX Ethernet has long used "MLT-3" encoding as a 3-level signal (+5v, 0v, -5v) form of NRZ. NRZ is a means of reducing the effective baseband signal frequency. problem with that encoding is it is baseband and suffers from baseline wander (up to .75V). (I was applications mgr for my company on that product. Lucent referenced my app note on their patent.)

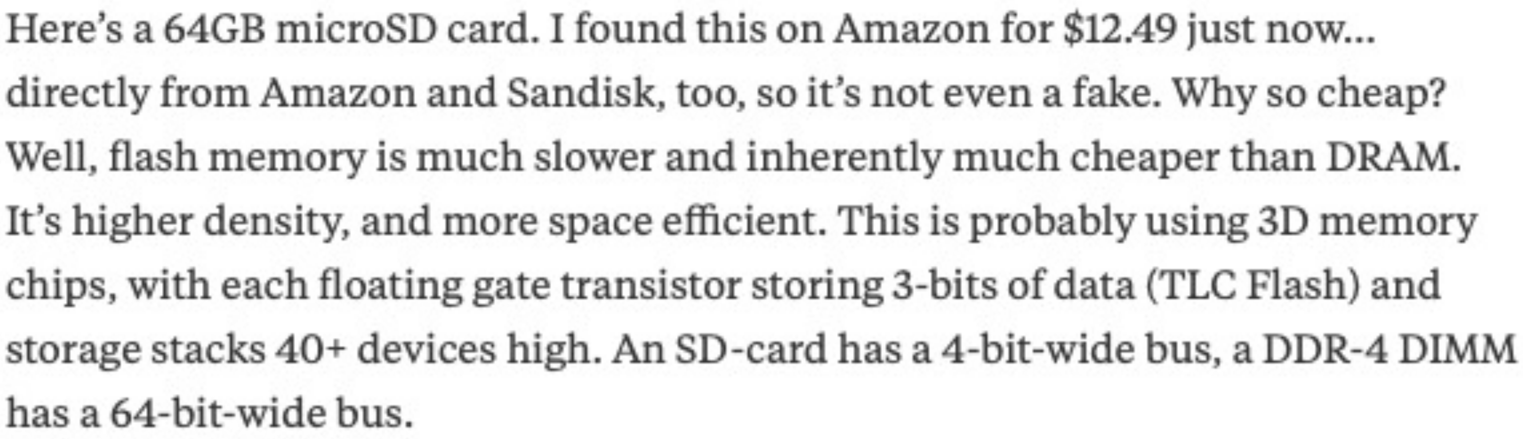
Modern Memory

64GB Flash

- Used on Raspberry Pi 4



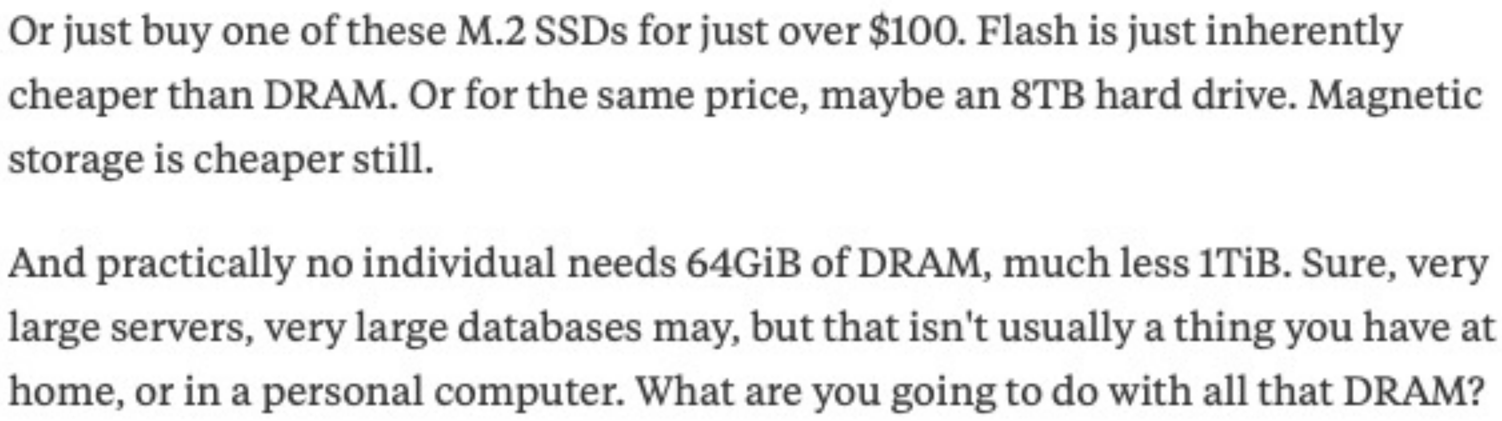
A disassembled USB flash drive. The chip on the left is flash memory. The controller is on the right.



Here's a 64GB microSD card. I found this on Amazon for $12.49 just now... directly from Amazon and Sandisk, too, so it's not even a fake. Why so cheap? Well, flash memory is much slower and inherently much cheaper than DRAM. It's higher density, and more space efficient. This is probably using 3D memory chips, with each floating gate transistor storing 3-bits of data (TLC Flash) and storage stacks 40+ devices high. An SD-card has a 4-bit-wide bus, a DDR-4 DIMM has a 64-bit-wide bus.

Modern Memory

1TB Flash



Or just buy one of these M.2 SSDs for just over $100. Flash is just inherently cheaper than DRAM. Or for the same price, maybe an 8TB hard drive. Magnetic storage is cheaper still.

And practically no individual needs 64GiB of DRAM, much less 1TiB. Sure, very large servers, very large databases may, but that isn't usually a thing you have at home, or in a personal computer. What are you going to do with all that DRAM?

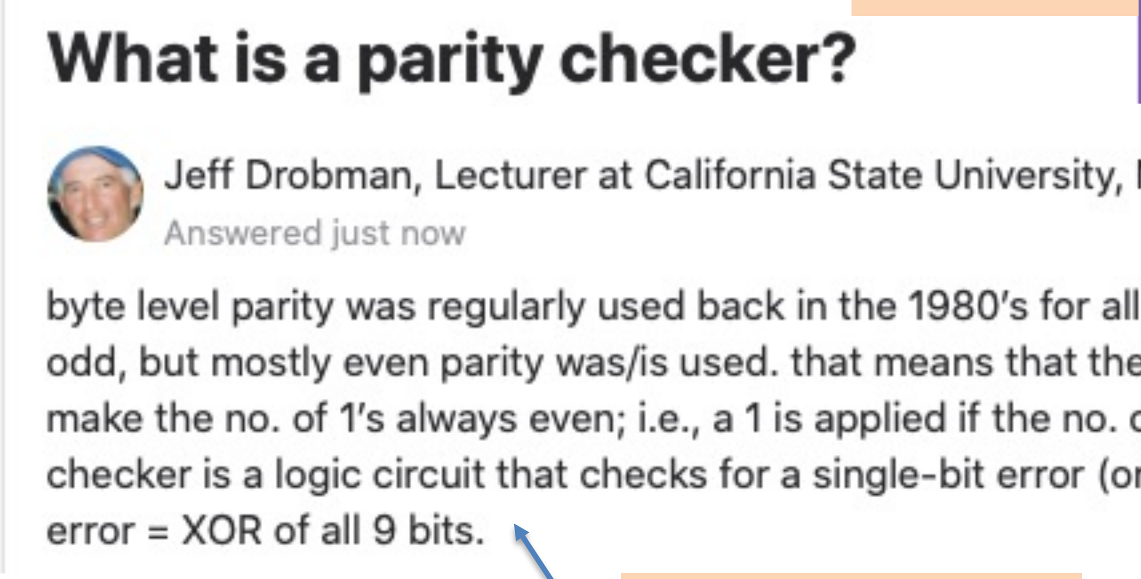
CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE
COMP122

Memory Errors

DR JEFF SOFTWARE
INDIE APP DEVELOPER

P&H Ch 5

COMP 122: Computer Architecture and Assembly Language
Spring 2020



What is a parity checker?

Jeff Drobman, Lecturer at California State University, Northridge (2016-present)
Answered just now

byte level parity was regularly used back in the 1980's for all RAM. the 2 choices are even or odd, but mostly even parity was/is used. that means that the 9th bit, parity, is selected to make the no. of 1's always even; i.e., a 1 is applied if the no. of 1's is odd, else a 0. a "parity" checker is a logic circuit that checks for a single-bit error (or any odd number of errors): error = XOR of all 9 bits.

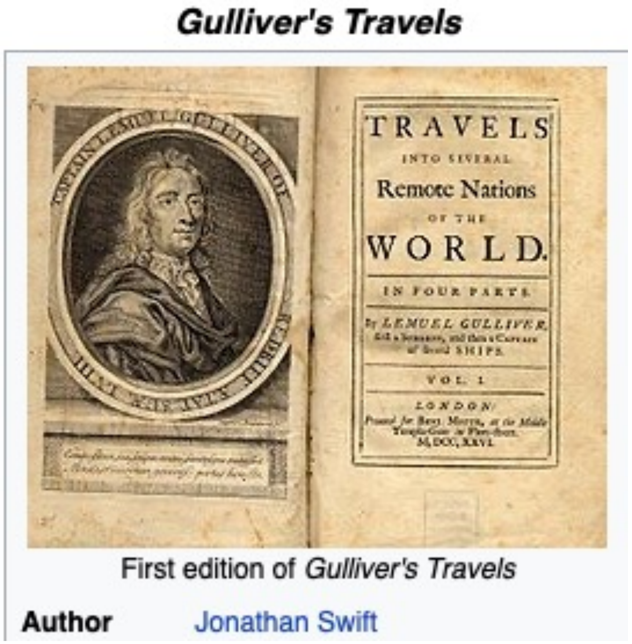
Error Handling (Bit)

- ❖ EDC (detection)
  - ❑ Parity (byte)
  - ❑ EDC
- ❖ ECC (correction)
  - ❑ Hamming Codes

Memory Architecture

Endianness

Endianness

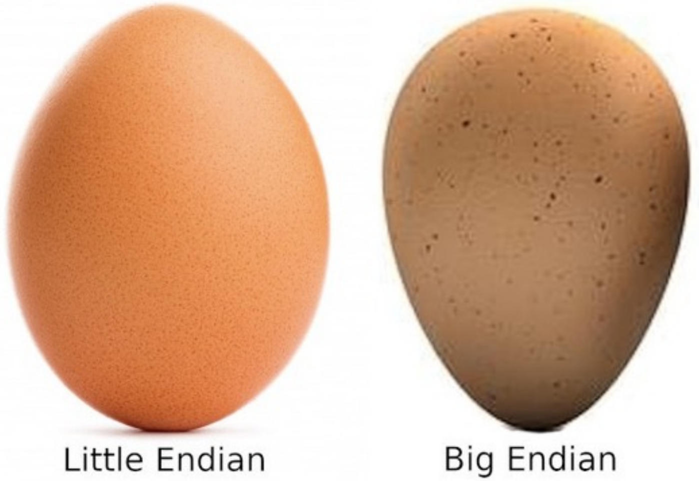


Gulliver's Travels

TRAVELS
INTO SEVERAL
Remote Nations
OF THE
WORLD.
IN FOUR PARTS.
By LEMUEL GULLIVER, First a Surgeon, and then a Captain of several SHIPS.
VOL. I.
LONDON:
Printed for Benj. Motte, at the Middle Temple-Gate in Fleet-street.
M,DCC,XXVI.

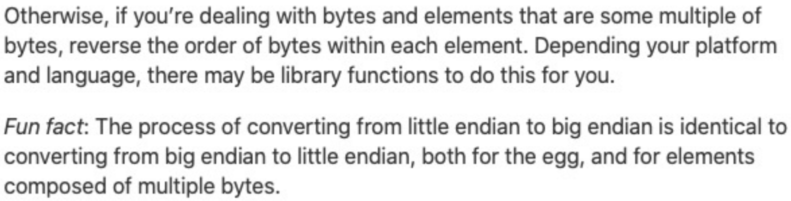
First edition of *Gulliver's Travels*

Author Jonathan Swift



Little Endian

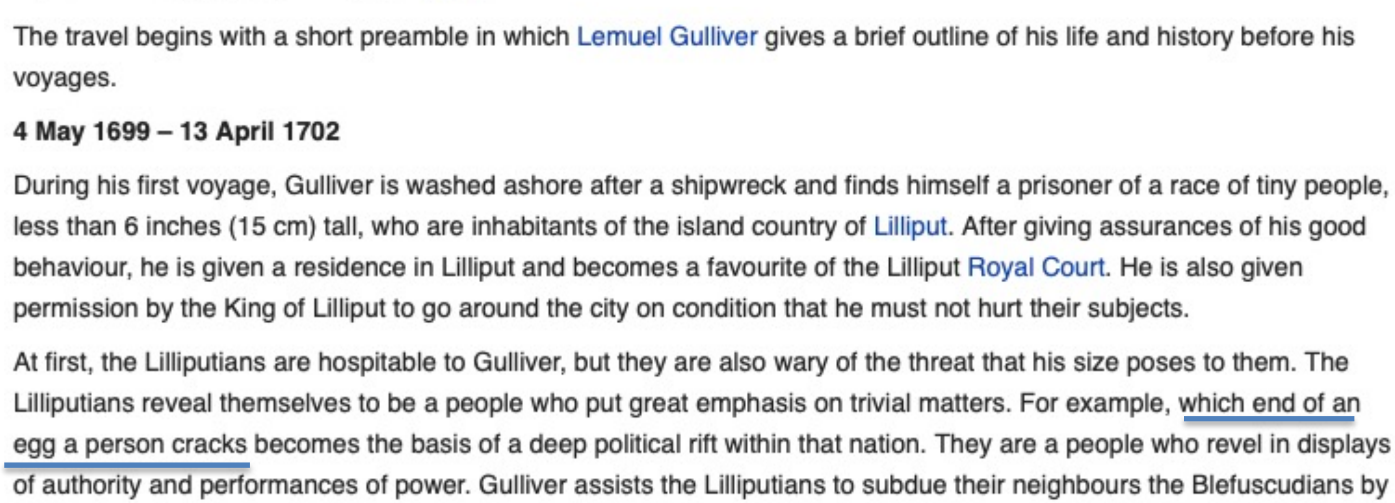
Big Endian



Otherwise, if you're dealing with bytes and elements that are some multiple of bytes, reverse the order of bytes within each element. Depending your platform and language, there may be library functions to do this for you.

Fun fact: The process of converting from little endian to big endian is identical to converting from big endian to little endian, both for the egg, and for elements composed of multiple bytes.

Part I: A Voyage to Lilliput [edit]



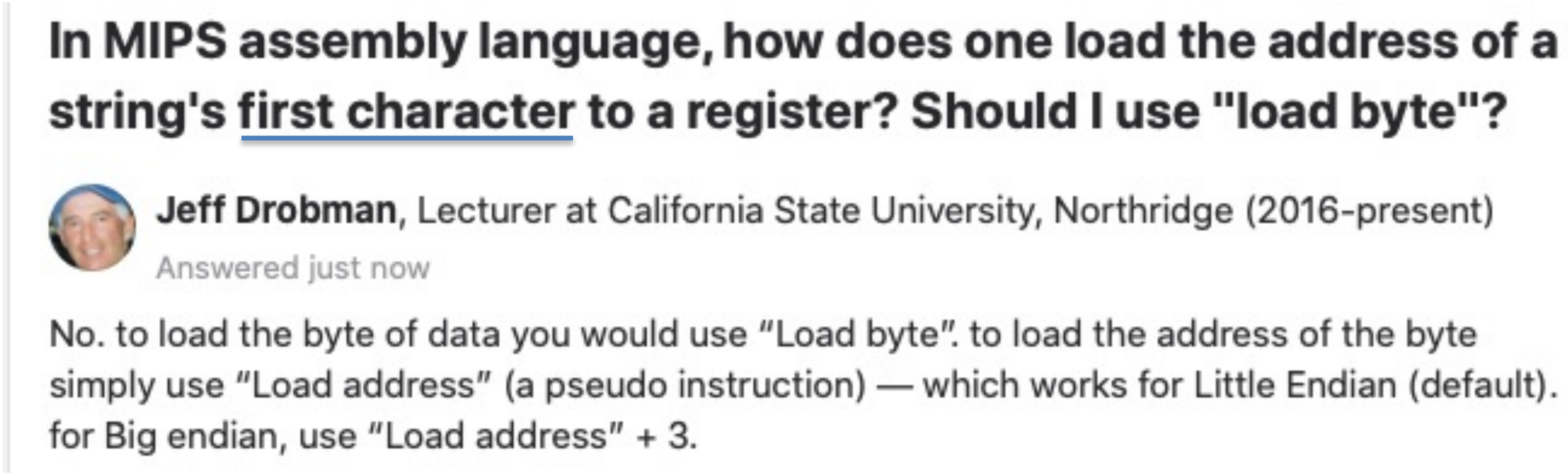
The travel begins with a short preamble in which Lemuel Gulliver gives a brief outline of his life and history before his voyages.

4 May 1699 – 13 April 1702

During his first voyage, Gulliver is washed ashore after a shipwreck and finds himself a prisoner of a race of tiny people, less than 6 inches (15 cm) tall, who are inhabitants of the island country of Lilliput. After giving assurances of his good behaviour, he is given a residence in Lilliput and becomes a favourite of the Lilliput Royal Court. He is also given permission by the King of Lilliput to go around the city on condition that he must not hurt their subjects.

At first, the Lilliputians are hospitable to Gulliver, but they are also wary of the threat that his size poses to them. The Lilliputians reveal themselves to be a people who put great emphasis on trivial matters. For example, which end of an egg a person cracks becomes the basis of a deep political rift within that nation. They are a people who revel in displays of authority and performances of power. Gulliver assists the Lilliputians to subdue their neighbours the Blefuscudians by

MIPS Load Byte

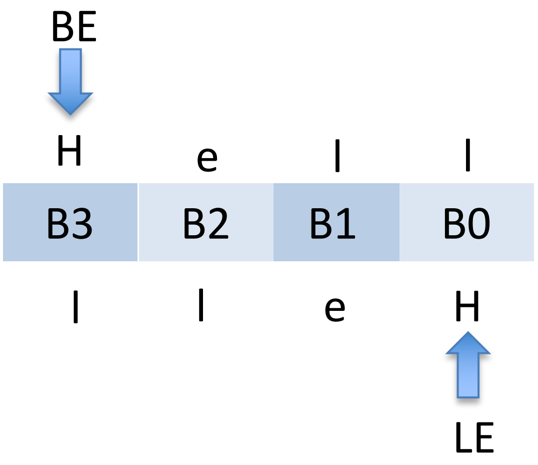


In MIPS assembly language, how does one load the address of a string's first character to a register? Should I use "load byte"?

Jeff Drobman, Lecturer at California State University, Northridge (2016-present)

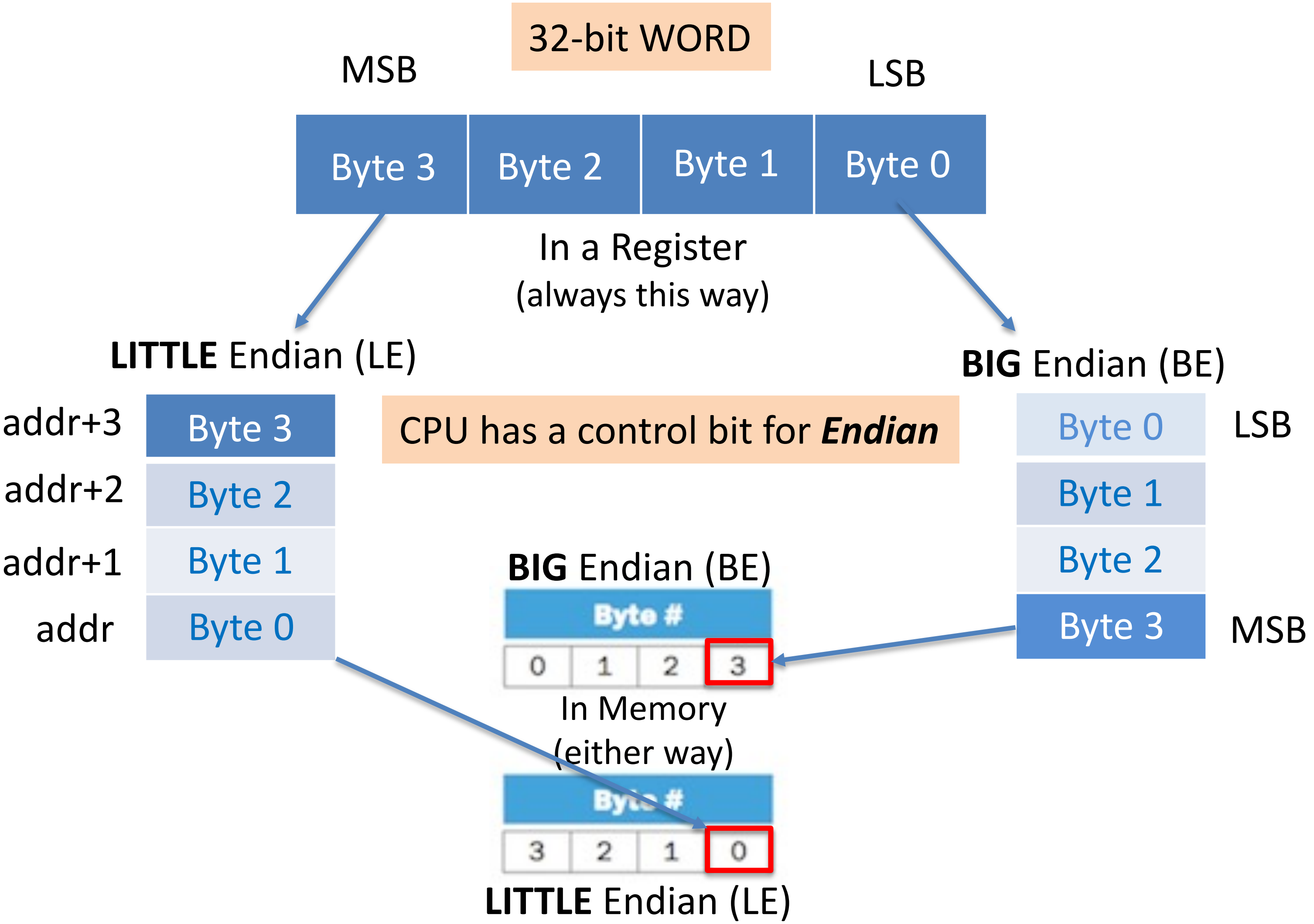
Answered just now

No. to load the byte of data you would use “Load byte”. to load the address of the byte simply use “Load address” (a pseudo instruction) — which works for Little Endian (default). for Big endian, use “Load address” + 3.



Data: Big/Little Endian

Endian



32-bit WORD

MSB

LSB

Byte 3	Byte 2	Byte 1	Byte 0

In a Register
(always this way)

CPU has a control bit for ***Endian***

LITTLE Endian (LE)

addr+3	Byte 3
addr+2	Byte 2
addr+1	Byte 1
addr	Byte 0

BIG Endian (BE)

Byte 0	LSB
Byte 1	
Byte 2	
Byte 3	MSB

BIG Endian (BE)

Byte #			
0	1	2	3

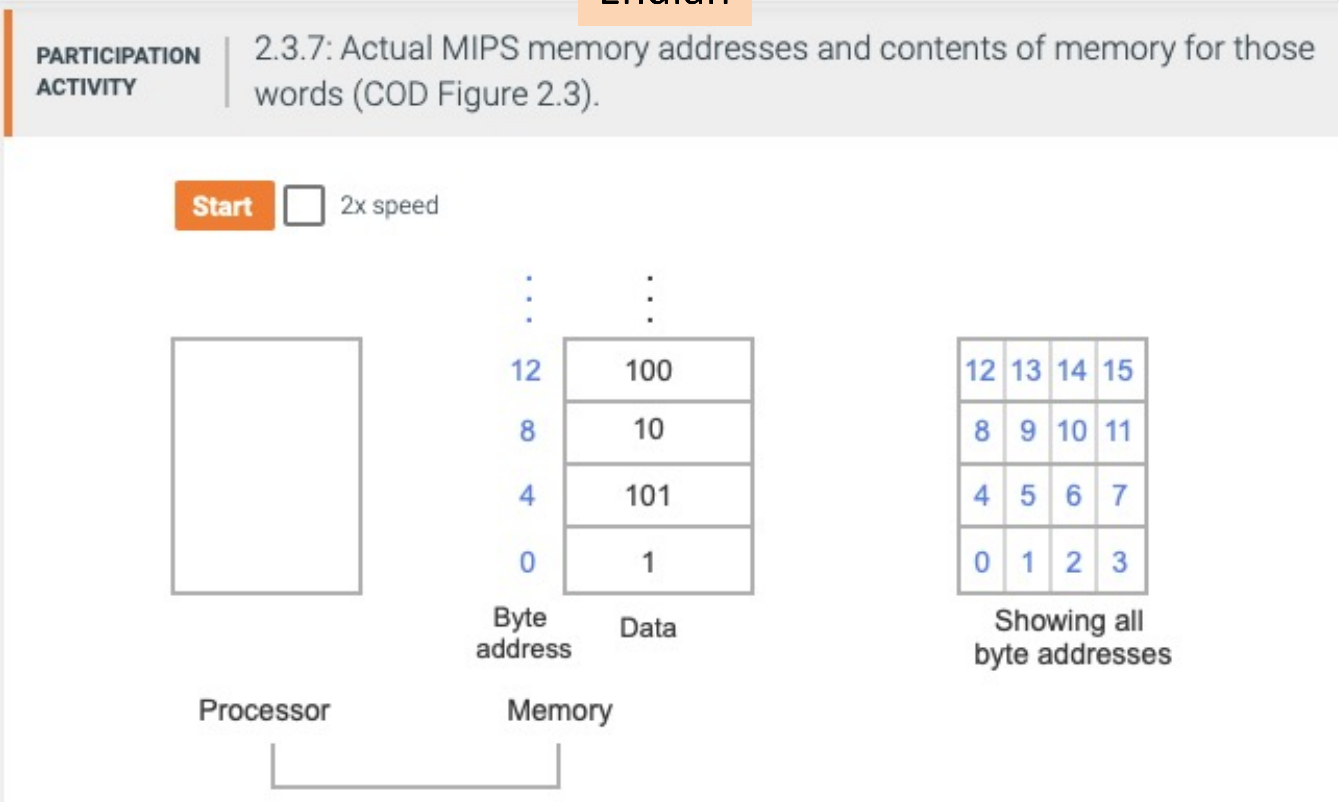
In Memory
(either way)

Byte #			
3	2	1	0

LITTLE Endian (LE)

Endianness

Endian



PARTICIPATION ACTIVITY | 2.3.7: Actual MIPS memory addresses and contents of memory for those words (COD Figure 2.3).

Start □ 2x speed

Byte address	Data
⋮	⋮
12	100
8	10
4	101
0	1

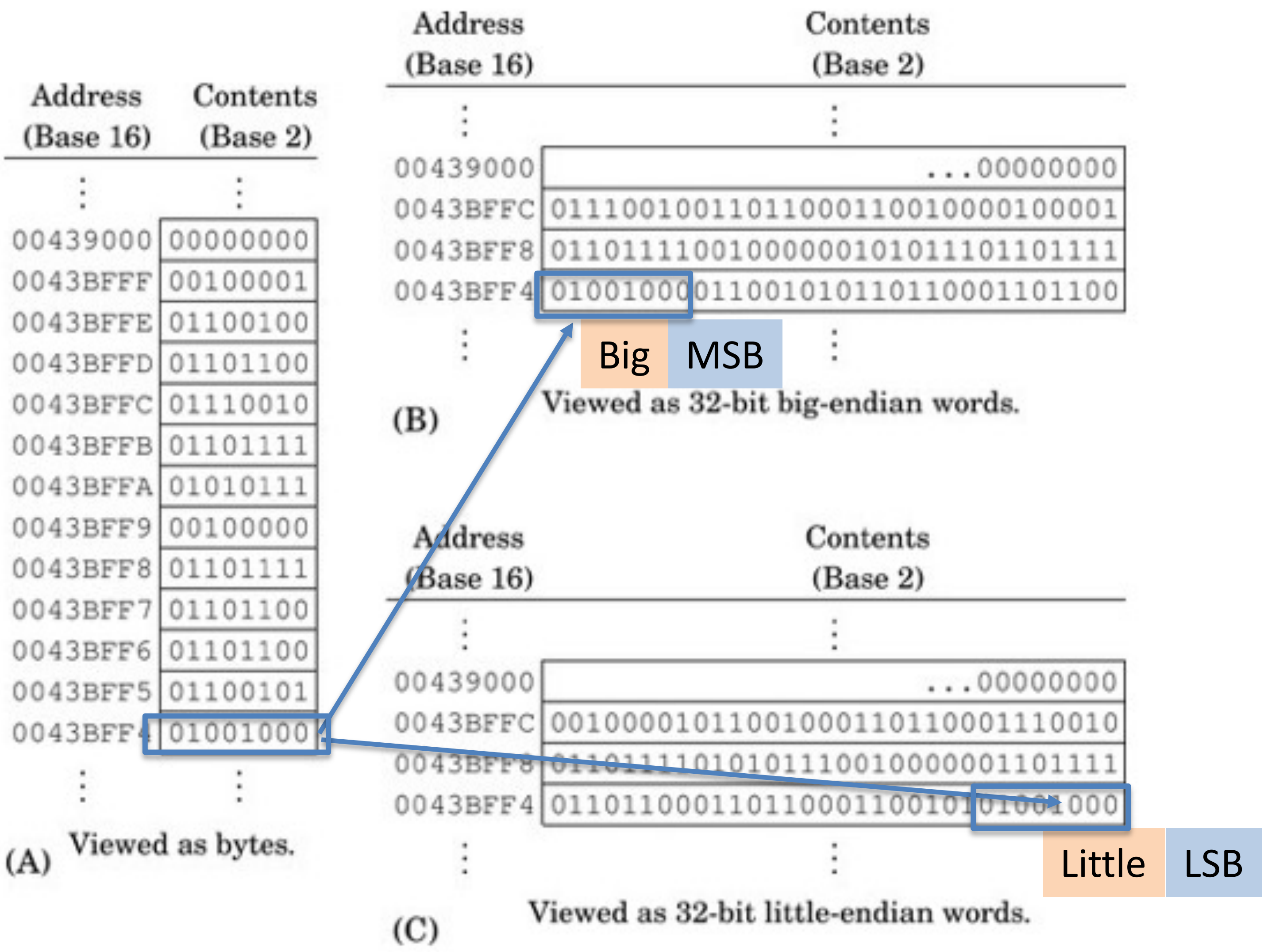
12	13	14	15
8	9	10	11
4	5	6	7
0	1	2	3

Showing all byte addresses

Computers divide into those that use the address of the leftmost or "big end" byte as the word address (_big endian_) versus those that use the rightmost or "little end" byte (_little endian_). MIPS is in the *big-endian camp*. Since the order matters only if you access the identical data both as a word and as four bytes, few need to be aware of the endianess. (COD Appendix A (Assemblers, Linkers, and the SPIM Simulator) shows the two options to number bytes in a word.)

Memory Addresses/Contents

Endian



Address (Base 16)	Contents (Base 2)
⋮	⋮
00439000	00000000
0043BFFF	00100001
0043BFFE	01100100
0043BFFD	01101100
0043BFFC	01110010
0043BFFB	01101111
0043BFFA	01010111
0043BFF9	00100000
0043BFF8	01101111
0043BFF7	01101100
0043BFF6	01101100
0043BFF5	01100101
0043BFF4	01001000
⋮	⋮

(A) Viewed as bytes.

Address (Base 16)	Contents (Base 2)
⋮	⋮
00439000	...00000000
0043BFFC	01110010011011000110010000100001
0043BFF8	01101111001000000101011101101111
0043BFF4	01001000011001010110110001101100
⋮	⋮

Big MSB

(B) Viewed as 32-bit big-endian words.

Address (Base 16)	Contents (Base 2)
⋮	⋮
00439000	...00000000
0043BFFC	00100001011001000110110001110010
0043BFF8	01101111010101110010000001101111
0043BFF4	01101100011011000110010101001000
⋮	⋮

Little LSB

(C) Viewed as 32-bit little-endian words.

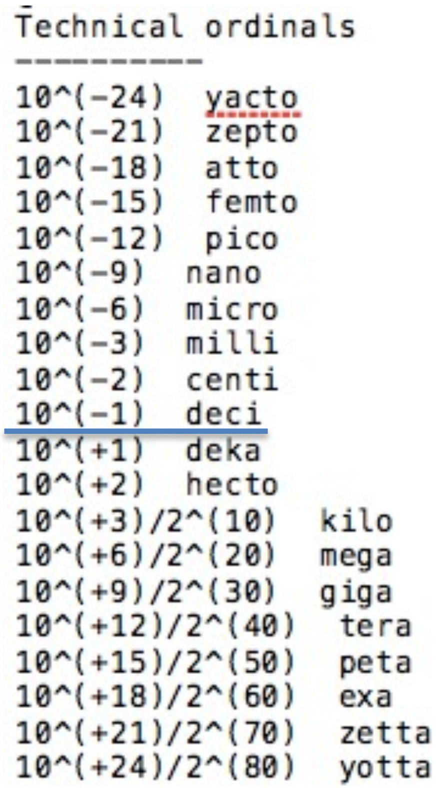
FIGURE 1.6 A section of memory.

Memory Architecture

Address Space

Ordinals

Powers of 2 <> 10: 10:3



```
Technical ordinals
----------
10^(-24)   yacto
10^(-21)   zepto
10^(-18)   atto
10^(-15)   femto
10^(-12)   pico
10^(-9)   nano
10^(-6)   micro
10^(-3)   milli
10^(-2)   centi
10^(-1)   deci
10^(+1)   deka
10^(+2)   hecto
10^(+3)/2^(10)   kilo
10^(+6)/2^(20)   mega
10^(+9)/2^(30)   giga
10^(+12)/2^(40)   tera
10^(+15)/2^(50)   peta
10^(+18)/2^(60)   exa
10^(+21)/2^(70)   zetta
10^(+24)/2^(80)   yotta
```

Gazillions

10^(+6) million
10^(+9) billion
10^(+12) trillion
10^(+15) quadrillion
10^(+18) quintillion
10^(+21) sexillion
10^(+24) septillion
10^(+27) octillion
10^(+30) nonillion
10^(+33) decillion
10^(+36) undecillion
10^(+39) duodecillion
10^(+42) tredecillion
10^(+45) quattuordecillion
10^(+48) quindecillion
10^(+51) sexdecillion
10^(+54) septendecillion
10^(+57) octodecillion
10^(+60) novemdecillion
10^(+63) vigintillion
10^(+100) googol
10^(+303) centillion
10^(10^(+100))
googolplex

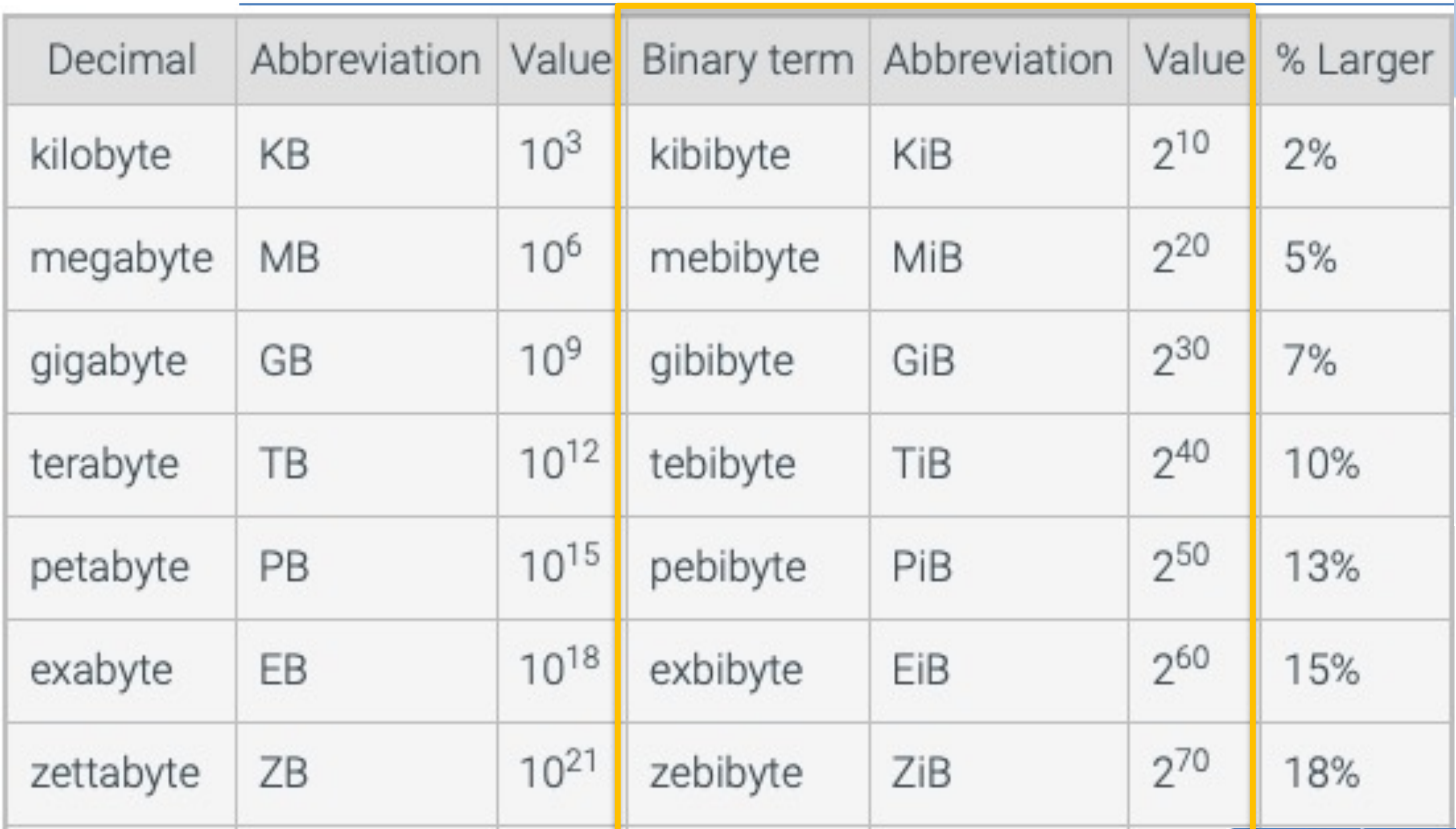
Ordinal	Power of 2	Power of 10	Actual
1K	2^{10}	10^{3}	1024
1M	2^{20}	10^{6}	1,048,576
1G	2^{30}	10^{9}	1.074×10^{9}
1T	2^{40}	10^{12}	1.0995×10^{12}

Name	2^{n}	M/G	Actual
byte	2^{8}	--	256
short	2^{16}	64K	65,536
integer	2^{32}	4B	4.3×10^{9}
long	2^{64}	16 Q	1.84×10^{19}
IPv6	2^{128}	340 uD	3.4×10^{38}

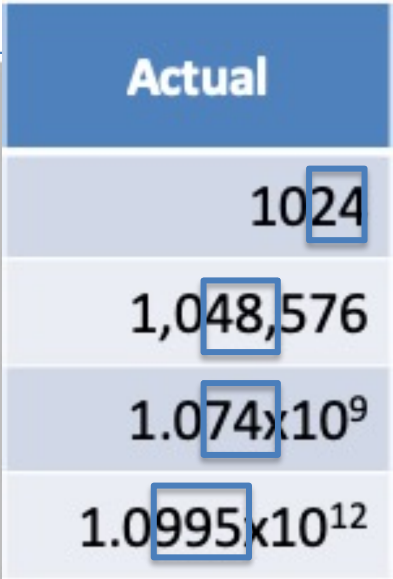
CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE
COMP122

GiB/TiB (2^{30}/2^{40})

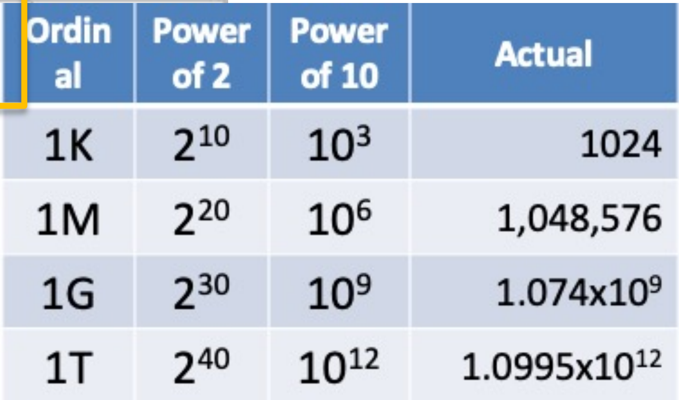
DR JEFF SOFTWARE
INDIE APP DEVELOPER
© Jeff Drobman
2016-2022



Decimal	Abbreviation	Value	Binary term	Abbreviation	Value	% Larger
kilobyte	KB	10^3	kibibyte	KiB	2^{10}	2%
megabyte	MB	10^6	mebibyte	MiB	2^{20}	5%
gigabyte	GB	10^9	gibibyte	GiB	2^{30}	7%
terabyte	TB	10^{12}	tebibyte	TiB	2^{40}	10%
petabyte	PB	10^{15}	pebibyte	PiB	2^{50}	13%
exabyte	EB	10^{18}	exbibyte	EiB	2^{60}	15%
zettabyte	ZB	10^{21}	zebibyte	ZiB	2^{70}	18%
yottabyte	YB	10^{24}	yobibyte	YiB	2^{80}	



Actual
1024
1,048,576
1.074×10^9
1.0995×10^{12}



Ordinal	Power of 2	Power of 10	Actual
1K	2^{10}	10^3	1024
1M	2^{20}	10^6	1,048,576
1G	2^{30}	10^9	1.074×10^9
1T	2^{40}	10^{12}	1.0995×10^{12}

Memory Architecture

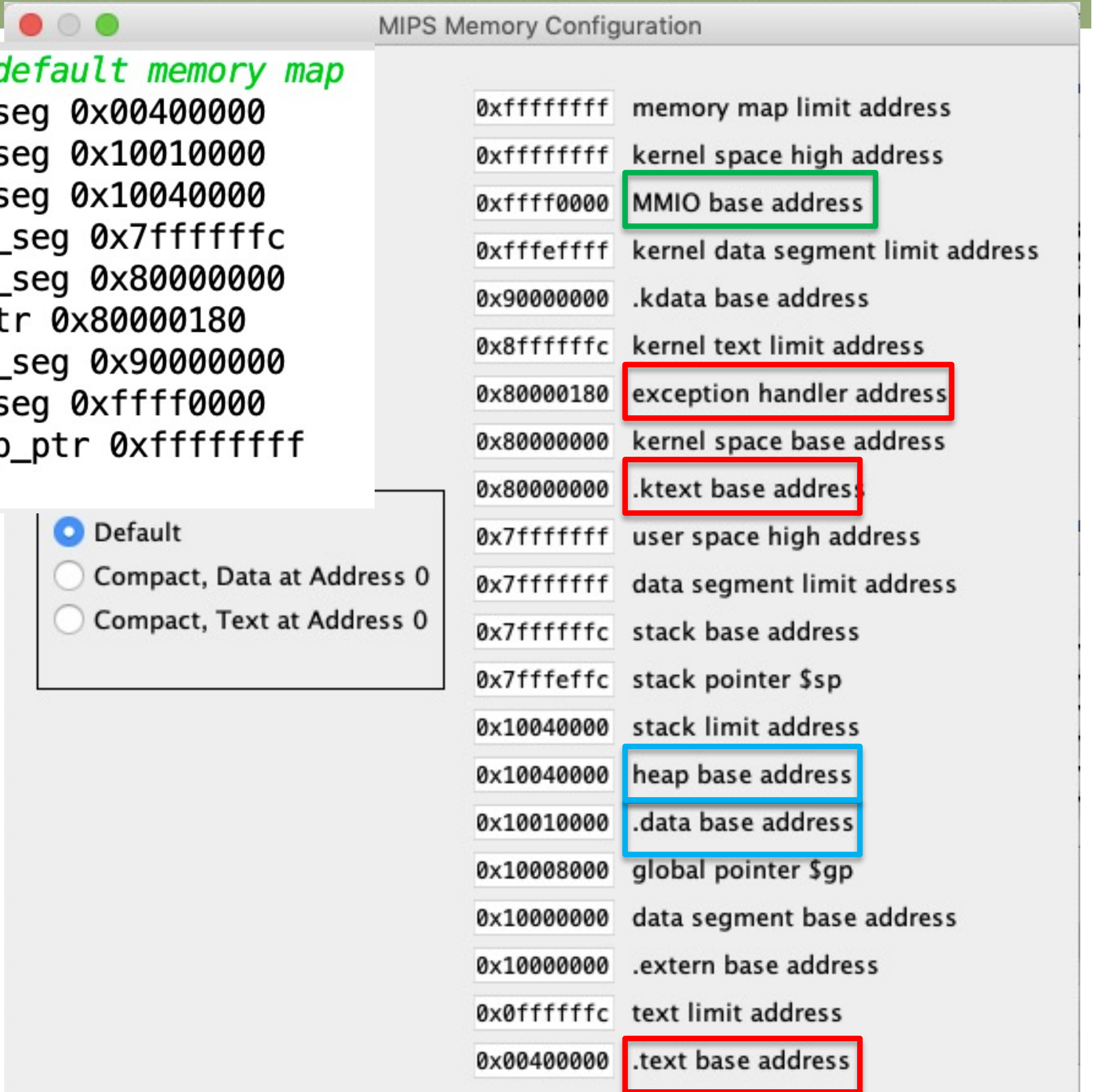
Segments

MARS

MARS (MIPS Assembler and Runtime Simulator)

Memory Map

```
#MIPS std default memory map
.eqv text_seg 0x00400000
.eqv data_seg 0x10010000
.eqv heap_seg 0x10040000
.eqv stack_seg 0x7ffffffc
.eqv ktext_seg 0x80000000
.eqv exc_ptr 0x80000180
.eqv kdata_seg 0x90000000
.eqv MMIO_seg 0xffff0000
.eqv memtop_ptr 0xffffffff
#end map
```

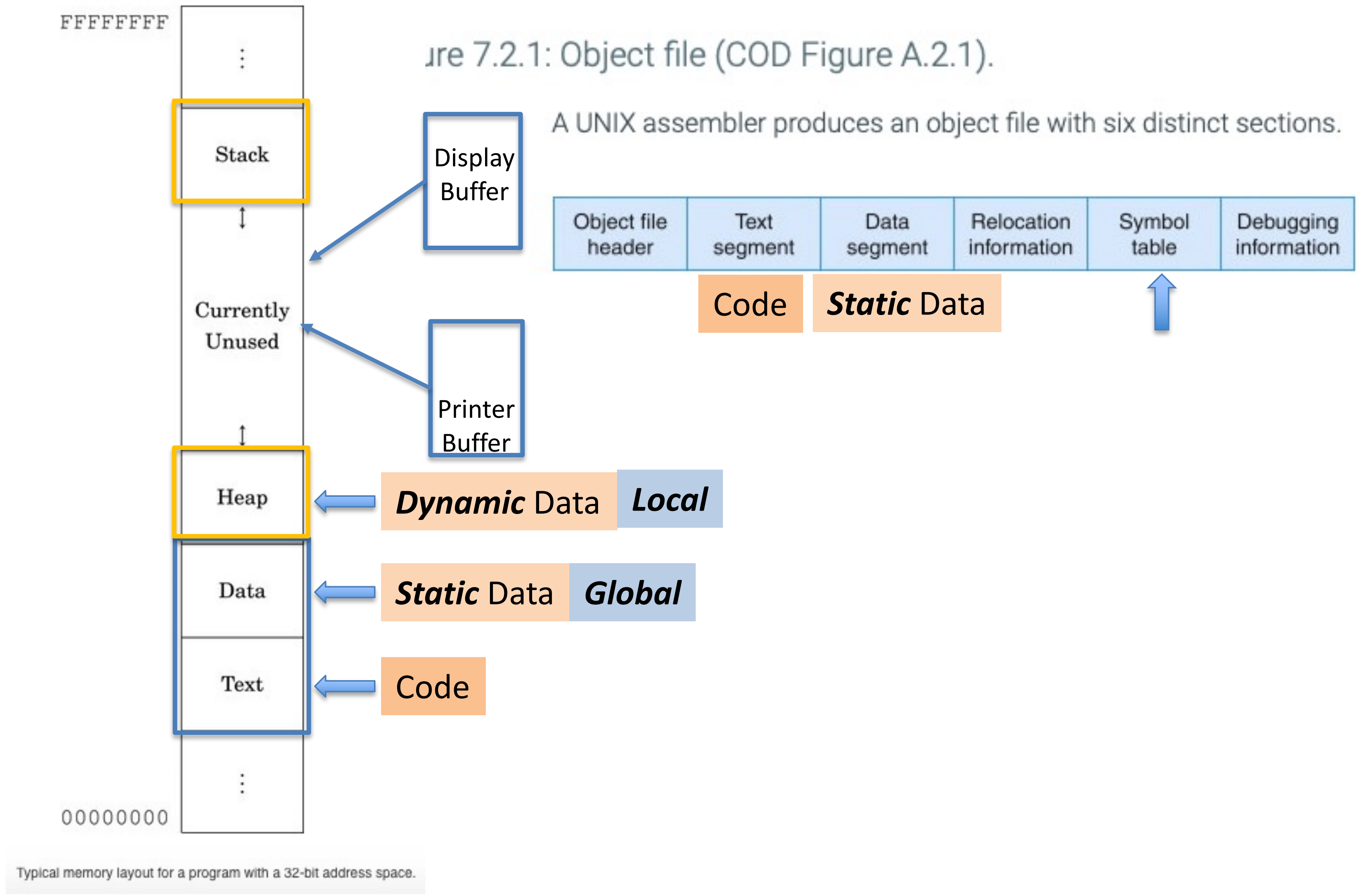


MIPS Memory Configuration

● Default
○ Compact, Data at Address 0
○ Compact, Text at Address 0

0xffffffff	memory map limit address
0xffffffff	kernel space high address
0xffff0000	MMIO base address
0xfffeffff	kernel data segment limit address
0x90000000	.kdata base address
0x8ffffffc	kernel text limit address
0x80000180	exception handler address
0x80000000	kernel space base address
0x80000000	.ktext base address
0x7fffffff	user space high address
0x7fffffff	data segment limit address
0x7ffffffc	stack base address
0x7fffeffc	stack pointer $sp
0x10040000	stack limit address
0x10040000	heap base address
0x10010000	.data base address
0x10008000	global pointer $gp
0x10000000	data segment base address
0x10000000	.extern base address
0x0ffffffc	text limit address
0x00400000	.text base address

Memory Segments



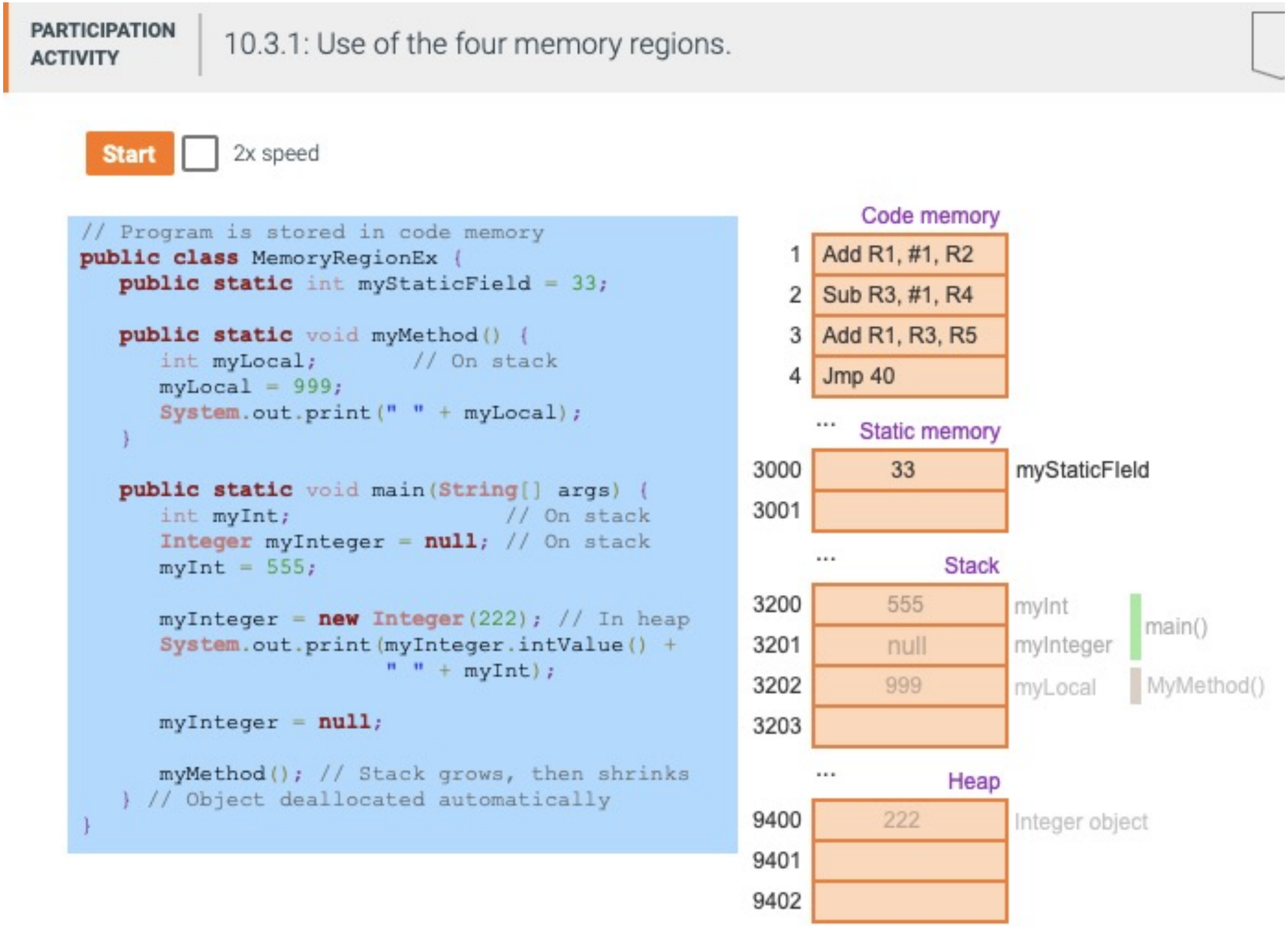
Typical memory layout for a program with a 32-bit address space.

ure 7.2.1: Object file (COD Figure A.2.1).

A UNIX assembler produces an object file with six distinct sections.

Object file header	Text segment	Data segment	Relocation information	Symbol table	Debugging information

Memory Segments



PARTICIPATION ACTIVITY | 10.3.1: Use of the four memory regions.

Start □ 2x speed

```
// Program is stored in code memory
public class MemoryRegionEx {
   public static int myStaticField = 33;

   public static void myMethod() {
      int myLocal;          // On stack
      myLocal = 999;
      System.out.print(" " + myLocal);
   }

   public static void main(String[] args) {
      int myInt;                     // On stack
      Integer myInteger = null;  // On stack
      myInt = 555;

      myInteger = new Integer(222); // In heap
      System.out.print(myInteger.intValue() +
                        " " + myInt);

      myInteger = null;

      myMethod(); // Stack grows, then shrinks
   } // Object deallocated automatically
}
```

Code memory

1	Add R1, #1, R2	
2	Sub R3, #1, R4	
3	Add R1, R3, R5	
4	Jmp 40	

...

Static memory

3000	33	myStaticField
3001		

...

Stack

3200	555	myInt	main()
3201	null	myInteger	
3202	999	myLocal	MyMethod()
3203			

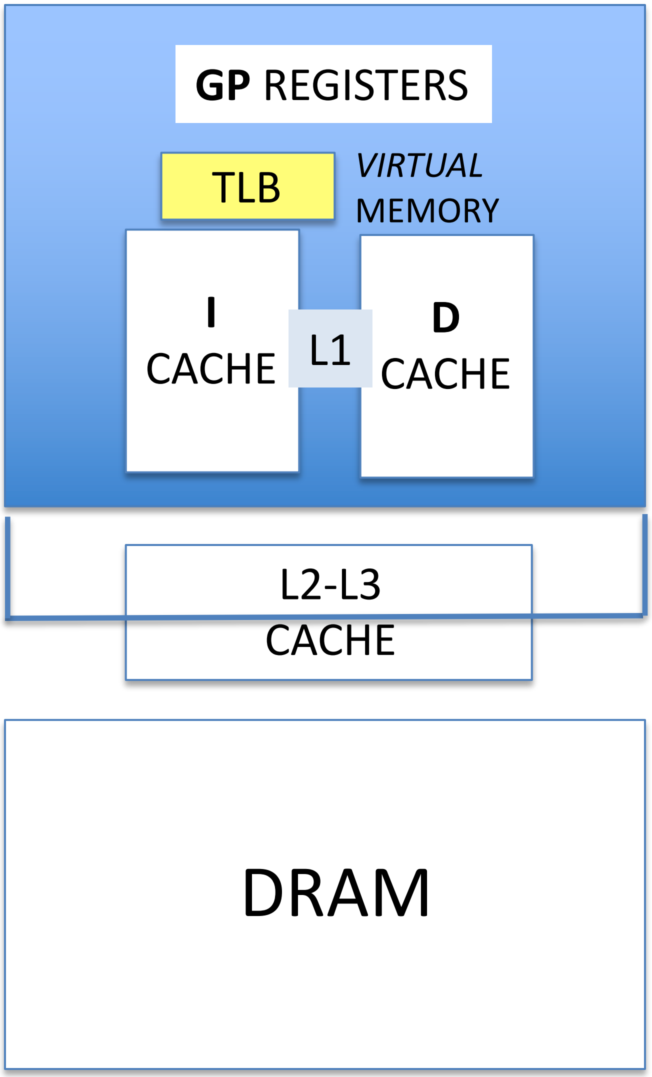
...

Heap

9400	222	Integer object
9401		
9402		

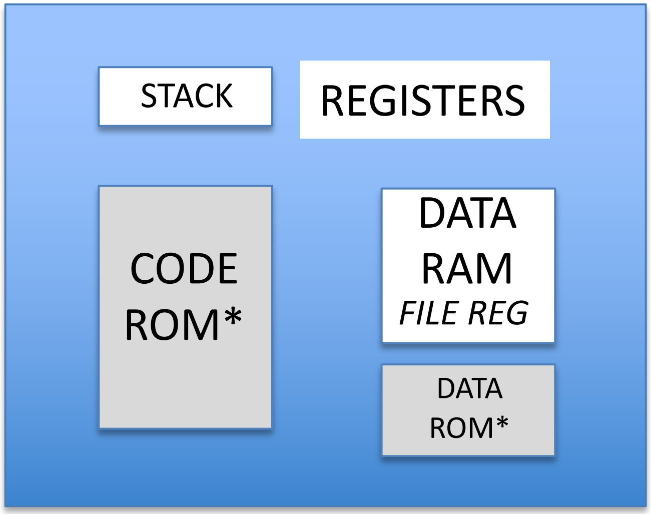
Memory: CPU vs MCU

MICRO**PROCESSOR**



LARGE
EXTERNAL
MEMORY

MICRO**CONTROLLER**

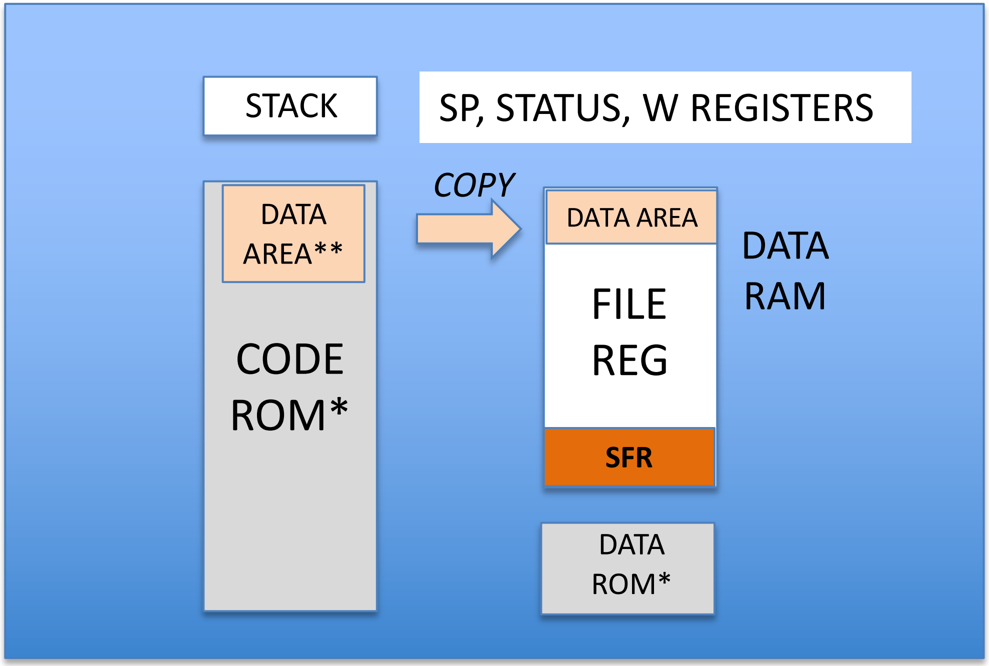


SMALL
INTERNAL
MEMORY

**ROM contents must be "programmed" "burned", or masked*

PIC18F Dual Memory

PIC 18F MICROCONTROLLER



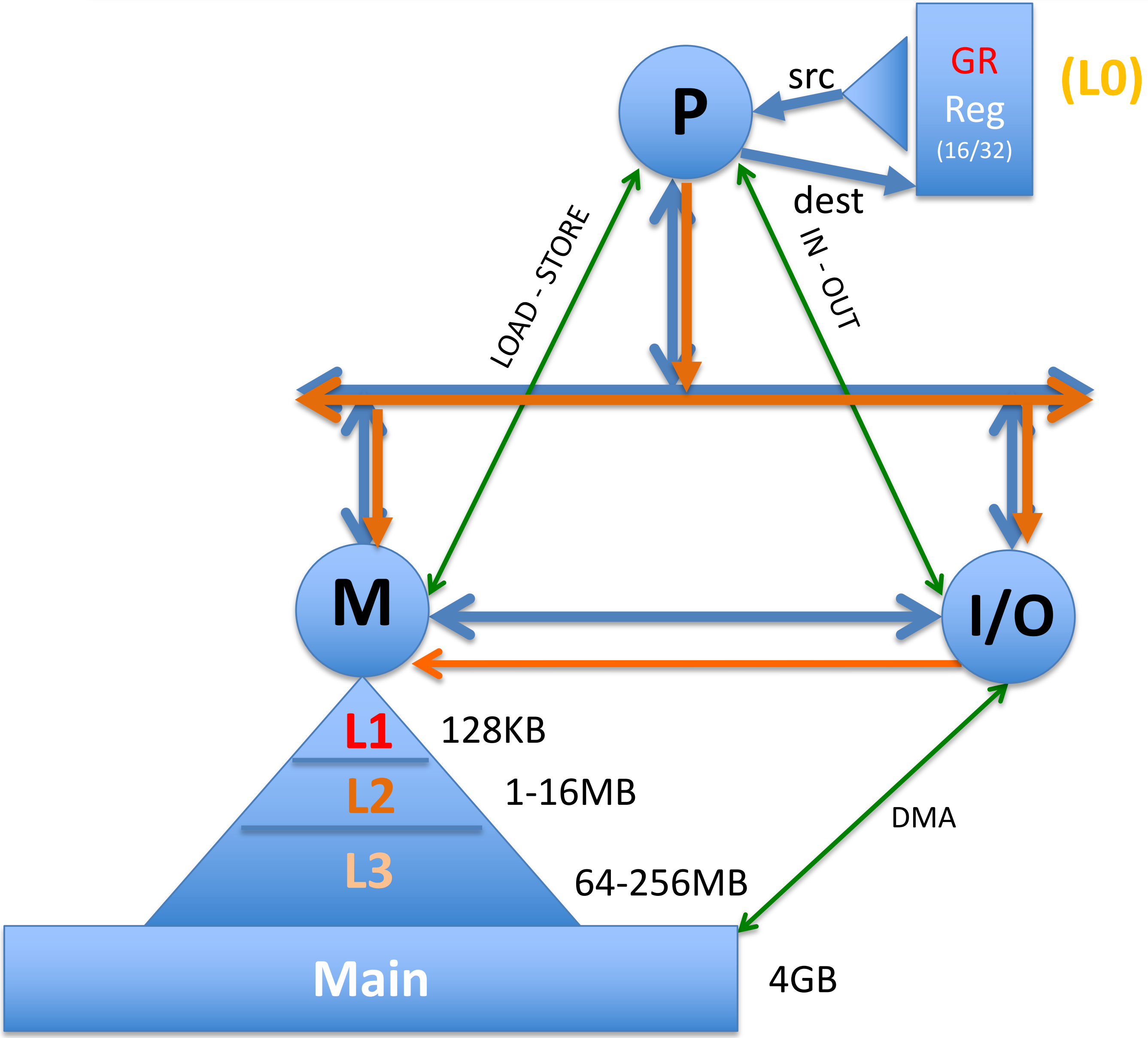
*READ ONLY

use **DB, DW

Memory

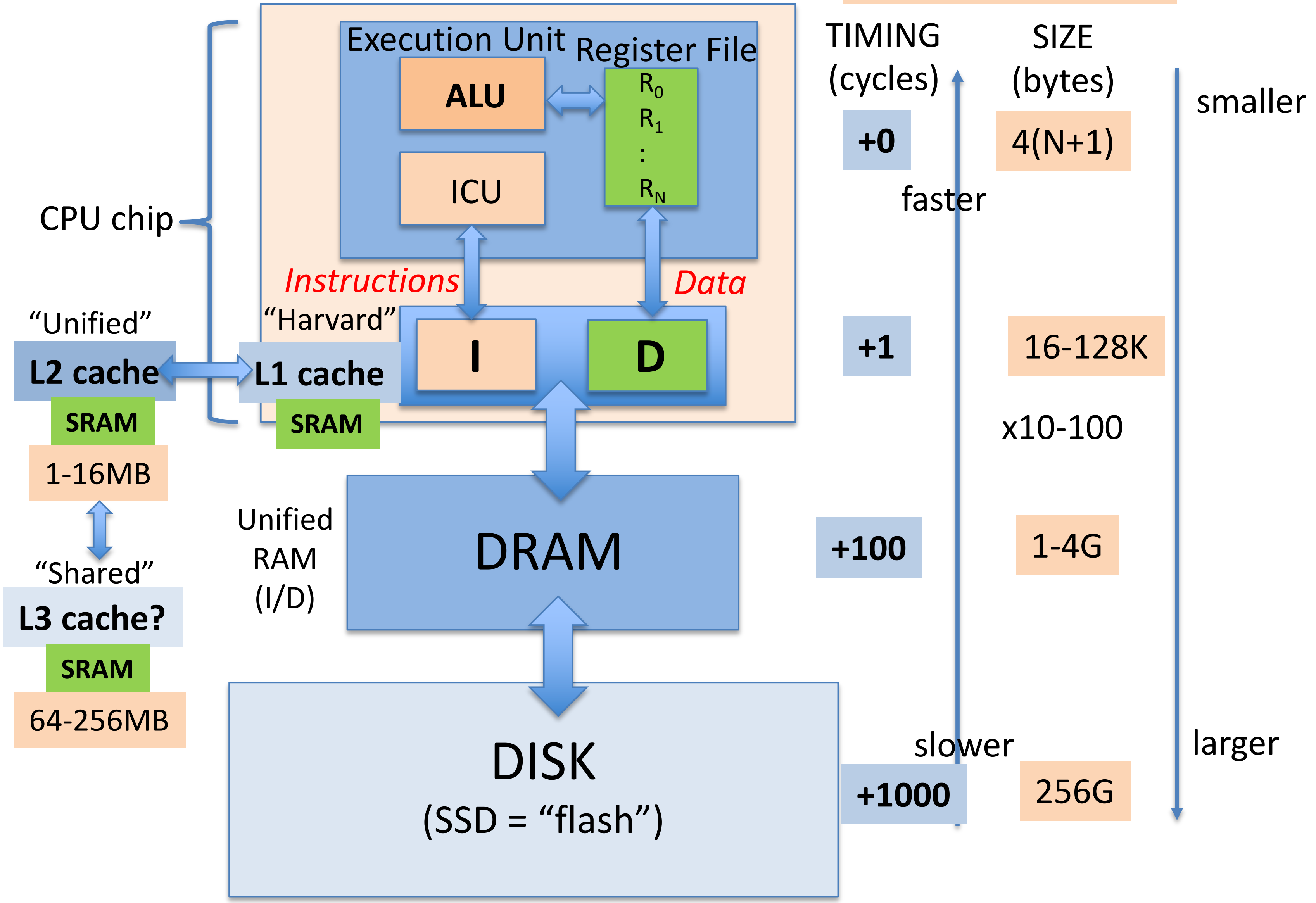
Cache

System Org: Multilevel Memory



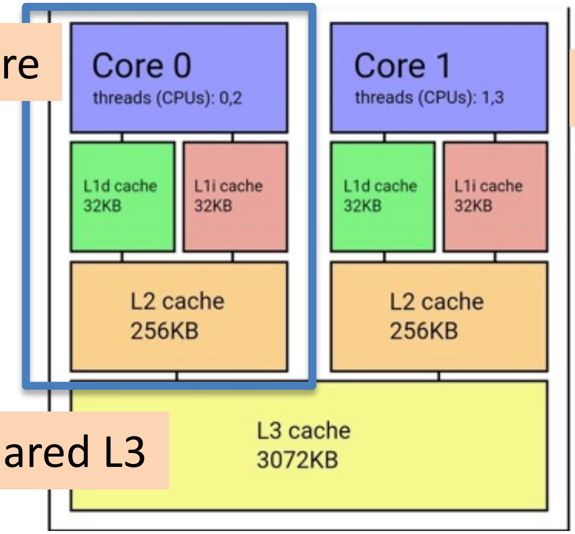
CPU Org + Memory Hierarchy

Multi-level Memory

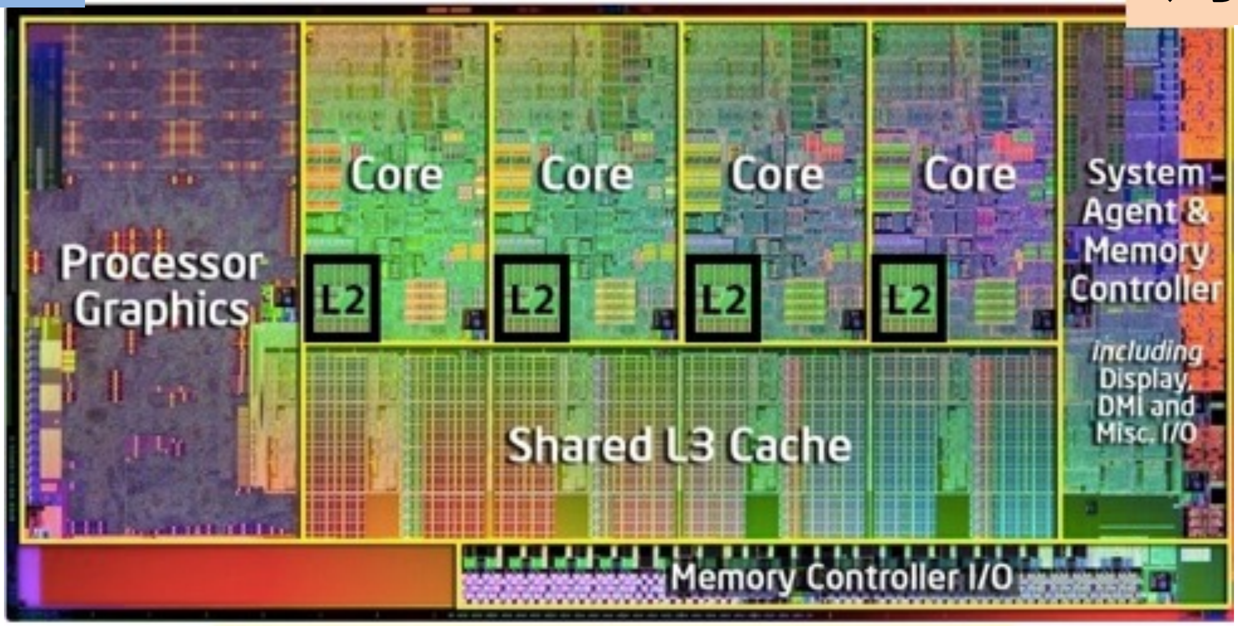


3 Levels of Cache

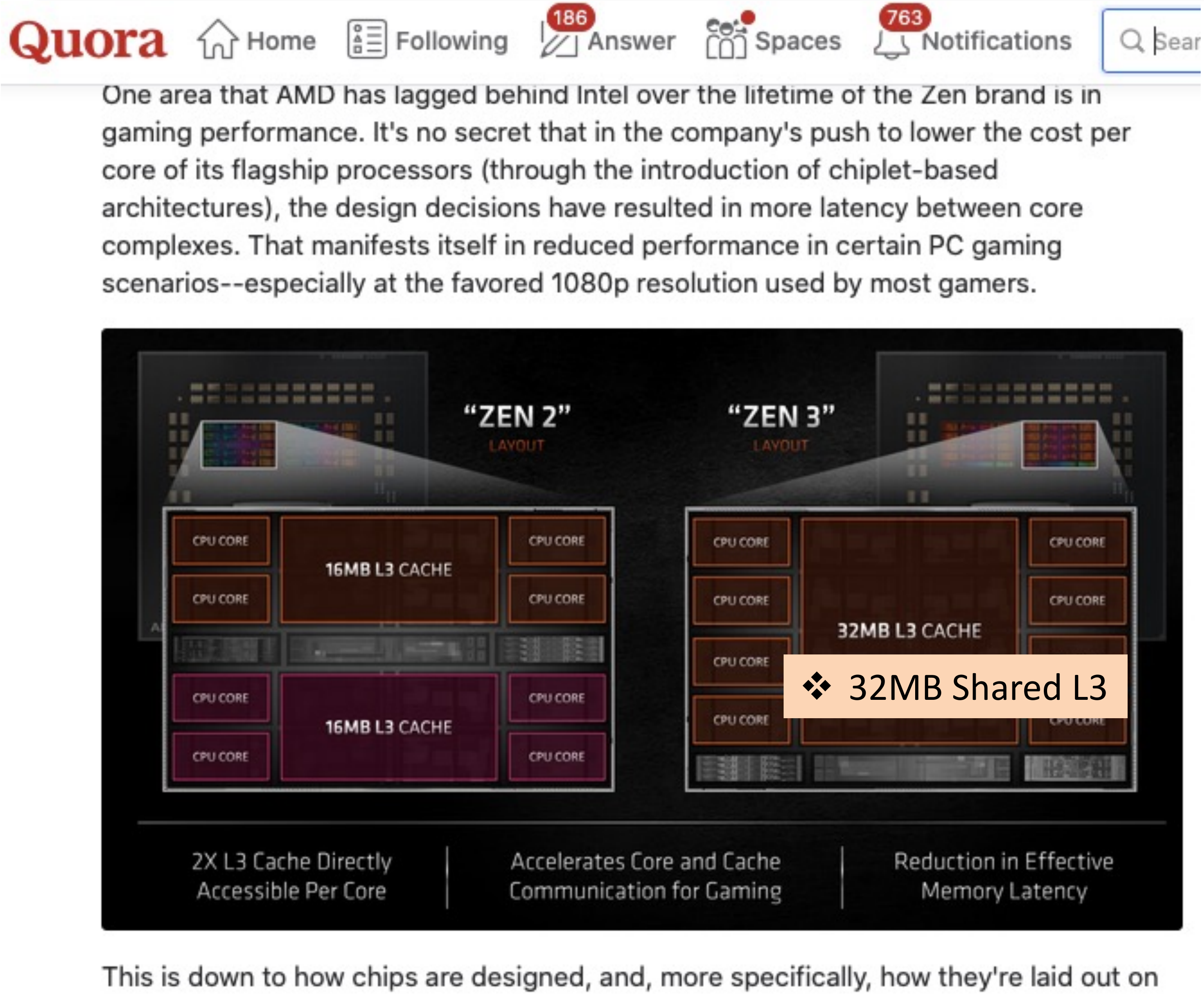
- ❖ Core
- ❖ Dual-Core
- ❖ Floorplan
- ❖ Shared L3



- ❖ Quad-Core
- ❖ Die



AMD Zen L3 Cache



One area that AMD has lagged behind Intel over the lifetime of the Zen brand is in gaming performance. It's no secret that in the company's push to lower the cost per core of its flagship processors (through the introduction of chiplet-based architectures), the design decisions have resulted in more latency between core complexes. That manifests itself in reduced performance in certain PC gaming scenarios--especially at the favored 1080p resolution used by most gamers.

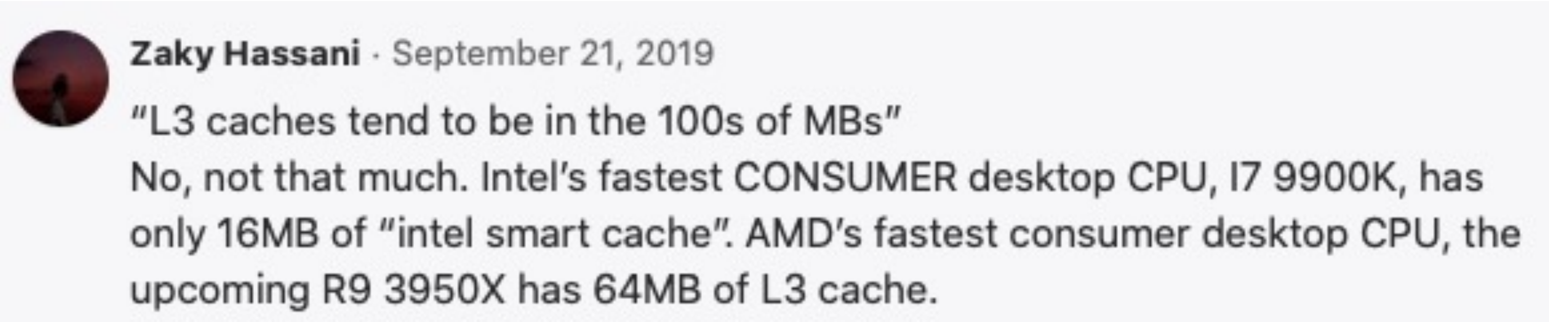
"ZEN 2"
LAYOUT
"ZEN 3"
LAYOUT
CPU CORE
16MB L3 CACHE
CPU CORE
16MB L3 CACHE
CPU CORE
32MB L3 CACHE
2X L3 Cache Directly Accessible Per Core
Accelerates Core and Cache Communication for Gaming
Reduction in Effective Memory Latency

❖ 32MB Shared L3

This is down to how chips are designed, and, more specifically, how they're laid out on

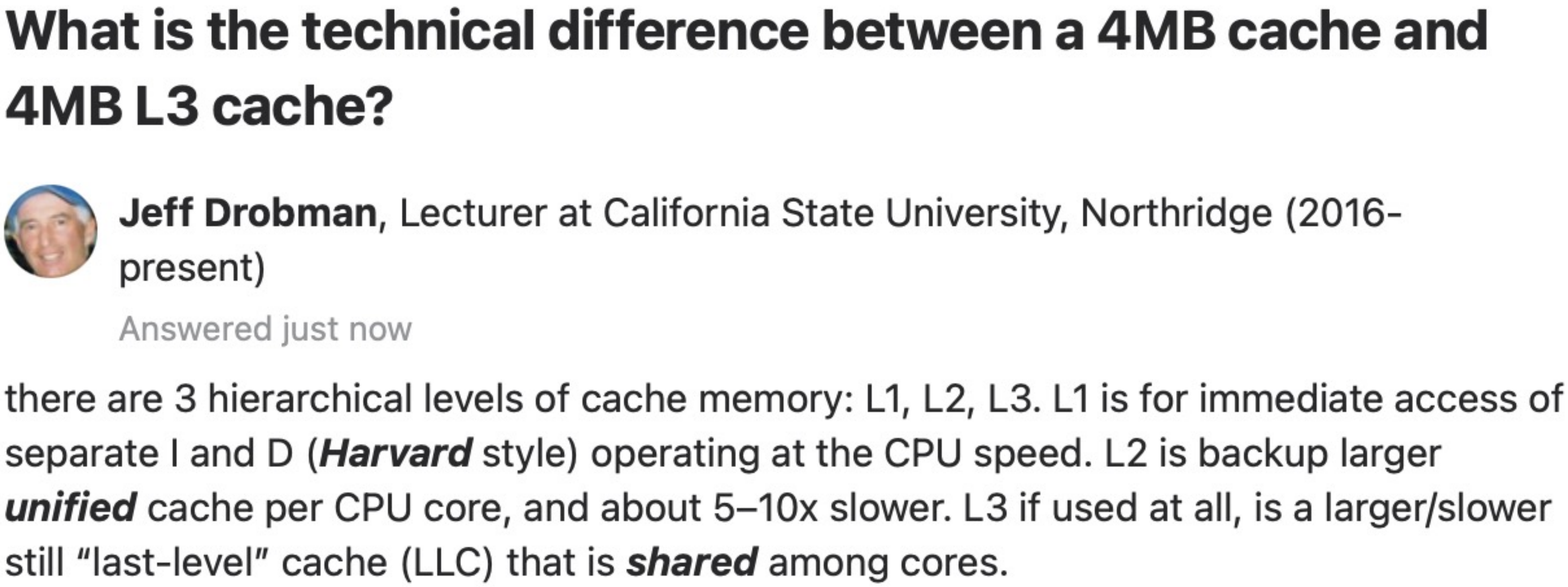
Level 3 (LLC) Cache Size

64-256 MB → 16-64 MB



Zaky Hassani · September 21, 2019

"L3 caches tend to be in the 100s of MBs"
No, not that much. Intel's fastest CONSUMER desktop CPU, I7 9900K, has only 16MB of "intel smart cache". AMD's fastest consumer desktop CPU, the upcoming R9 3950X has 64MB of L3 cache.



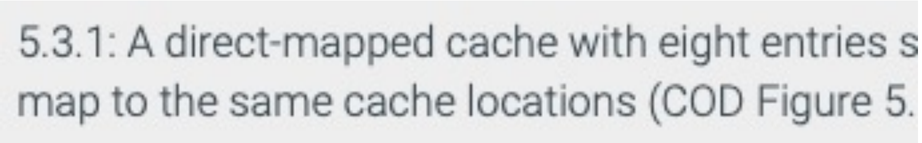
What is the technical difference between a 4MB cache and 4MB L3 cache?

Jeff Drobman, Lecturer at California State University, Northridge (2016-present)

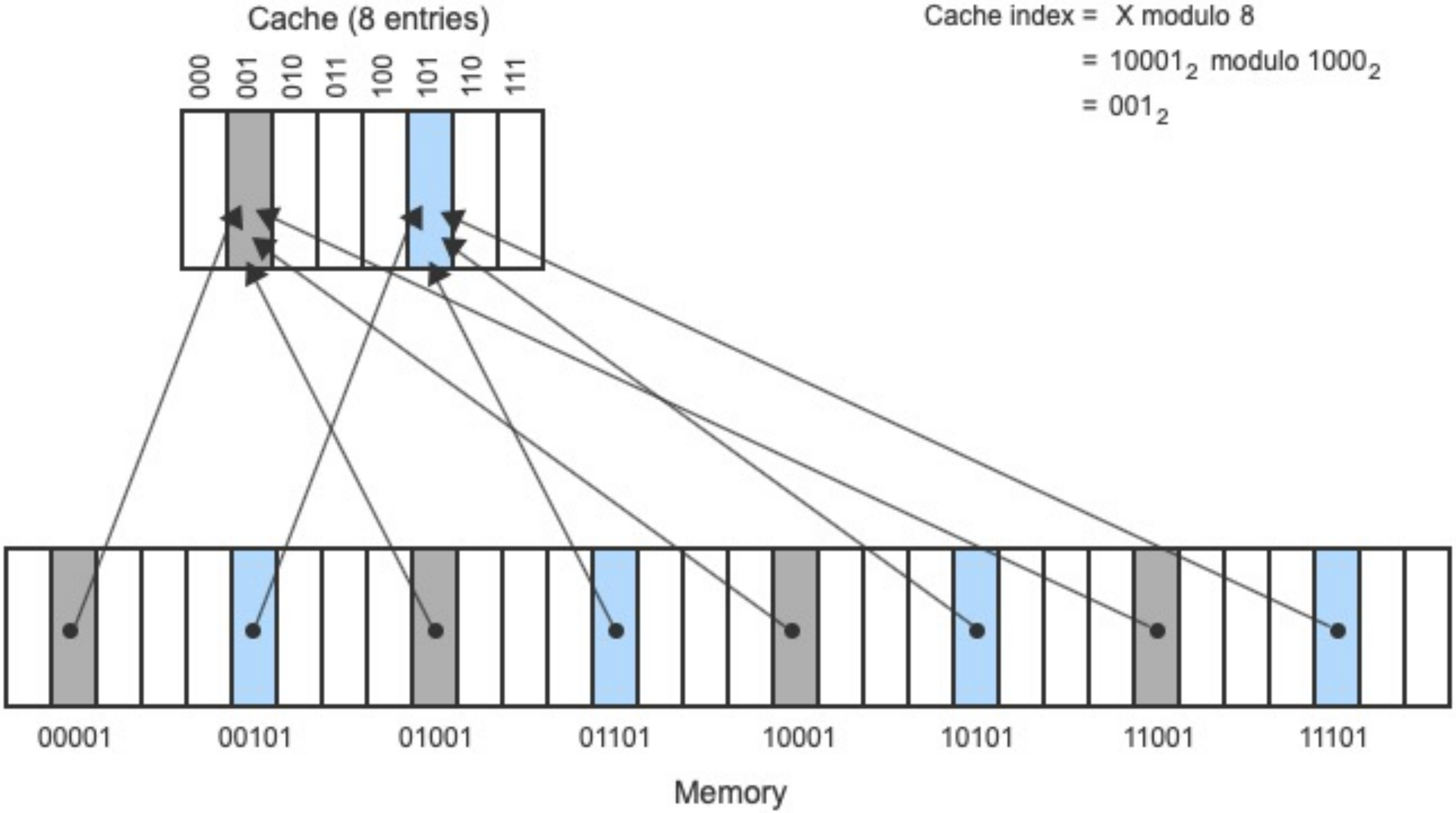
Answered just now

there are 3 hierarchical levels of cache memory: L1, L2, L3. L1 is for immediate access of separate I and D (***Harvard*** style) operating at the CPU speed. L2 is backup larger ***unified*** cache per CPU core, and about 5–10x slower. L3 if used at all, is a larger/slower still "last-level" cache (LLC) that is ***shared*** among cores.

Cache Memory

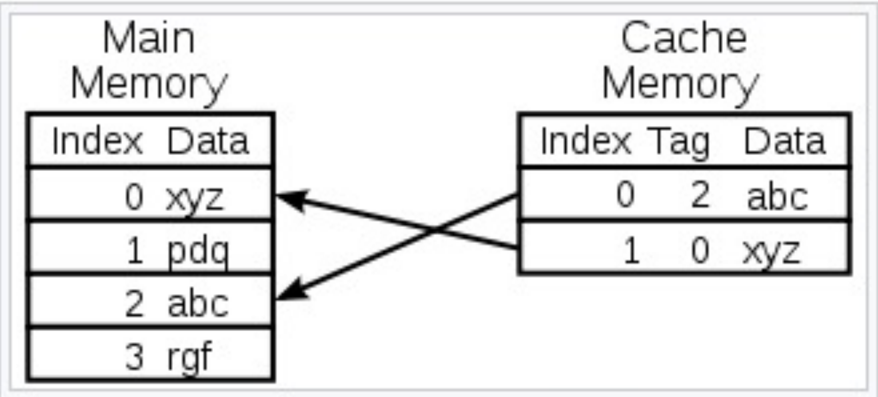


5.3.1: A direct-mapped cache with eight entries s
map to the same cache locations (COD Figure 5.



$$\text{Cache index} = X \text{ modulo } 8$$
$$= 10001_2 \text{ modulo } 1000_2$$
$$= 001_2$$

MLM – Cache



Main Memory

Index	Data
0	xyz
1	pdq
2	abc
3	rgf

Cache Memory

Index	Tag	Data
0	2	abc
1	0	xyz

Diagram of a CPU memory cache operation

Direct-mapped

16	16
Tag	address

SRAM

Associative

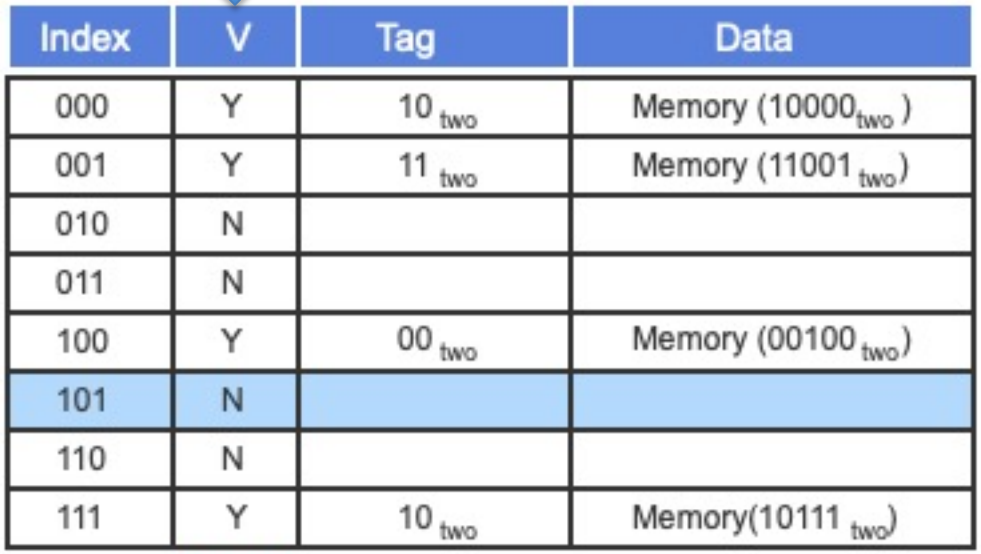
Tag=Full 32-bit address

CAM

Cache Memory

P&H Ch 5

Valid Bit



Index	V	Tag	Data
000	Y	10_{two}	Memory (10000_{two})
001	Y	11_{two}	Memory (11001_{two})
010	N		
011	N		
100	Y	00_{two}	Memory (00100_{two})
101	N		
110	N		
111	Y	10_{two}	Memory(10111_{two})

Invalid data

Cache *flush* V=0

CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE

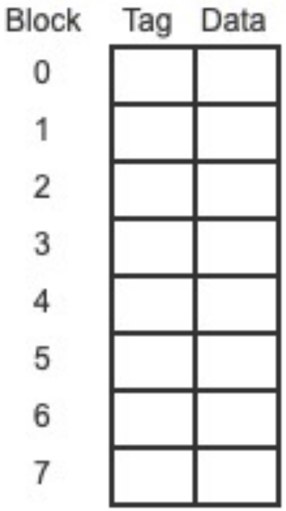
COMP122

Cache Memory

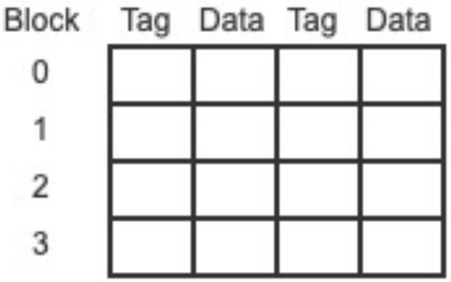
P&H Ch 5

COMP 122: Computer Architecture and Assembly Language
Spring 2020

One-way set associative
(direct mapped)

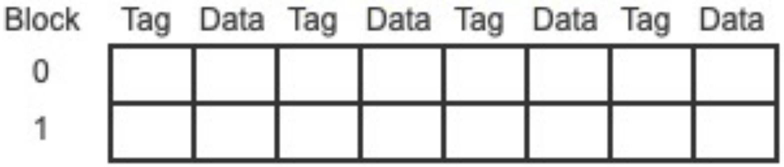


Two-way set associative

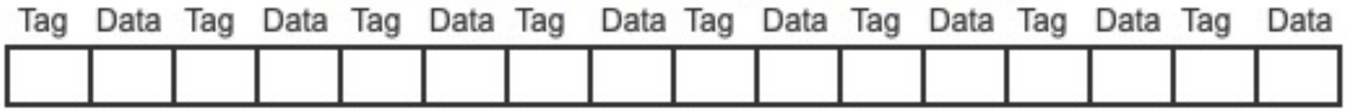


“Ways”

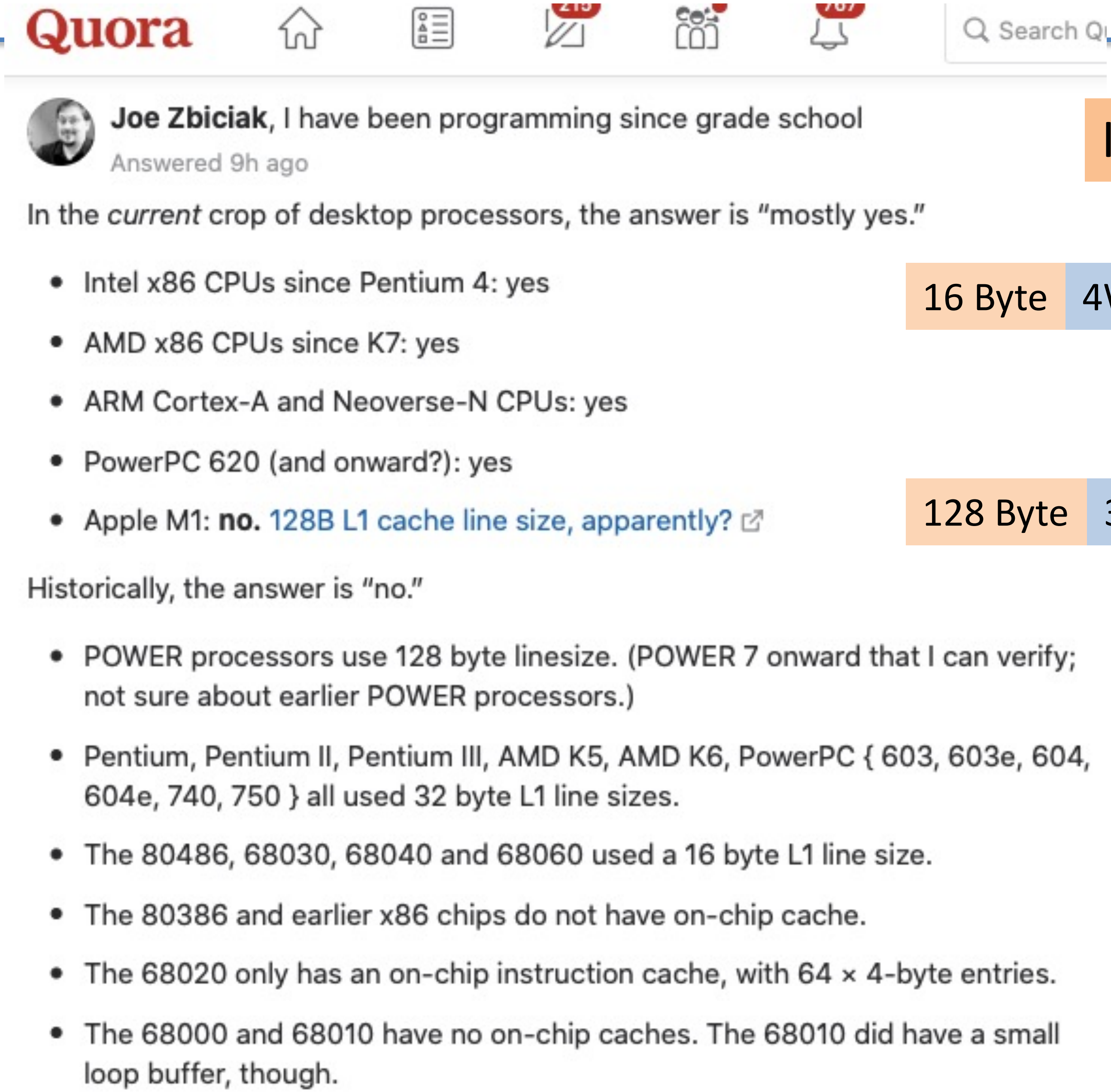
Four-way set associative



Eight way set associative (fully associative)



L1 Cache Line Sizes



Joe Zbiciak, I have been programming since grade school
Answered 9h ago

In the *current* crop of desktop processors, the answer is "mostly yes."

- Intel x86 CPUs since Pentium 4: yes
- AMD x86 CPUs since K7: yes
- ARM Cortex-A and Neoverse-N CPUs: yes
- PowerPC 620 (and onward?): yes
- Apple M1: **no.** 128B L1 cache line size, apparently?

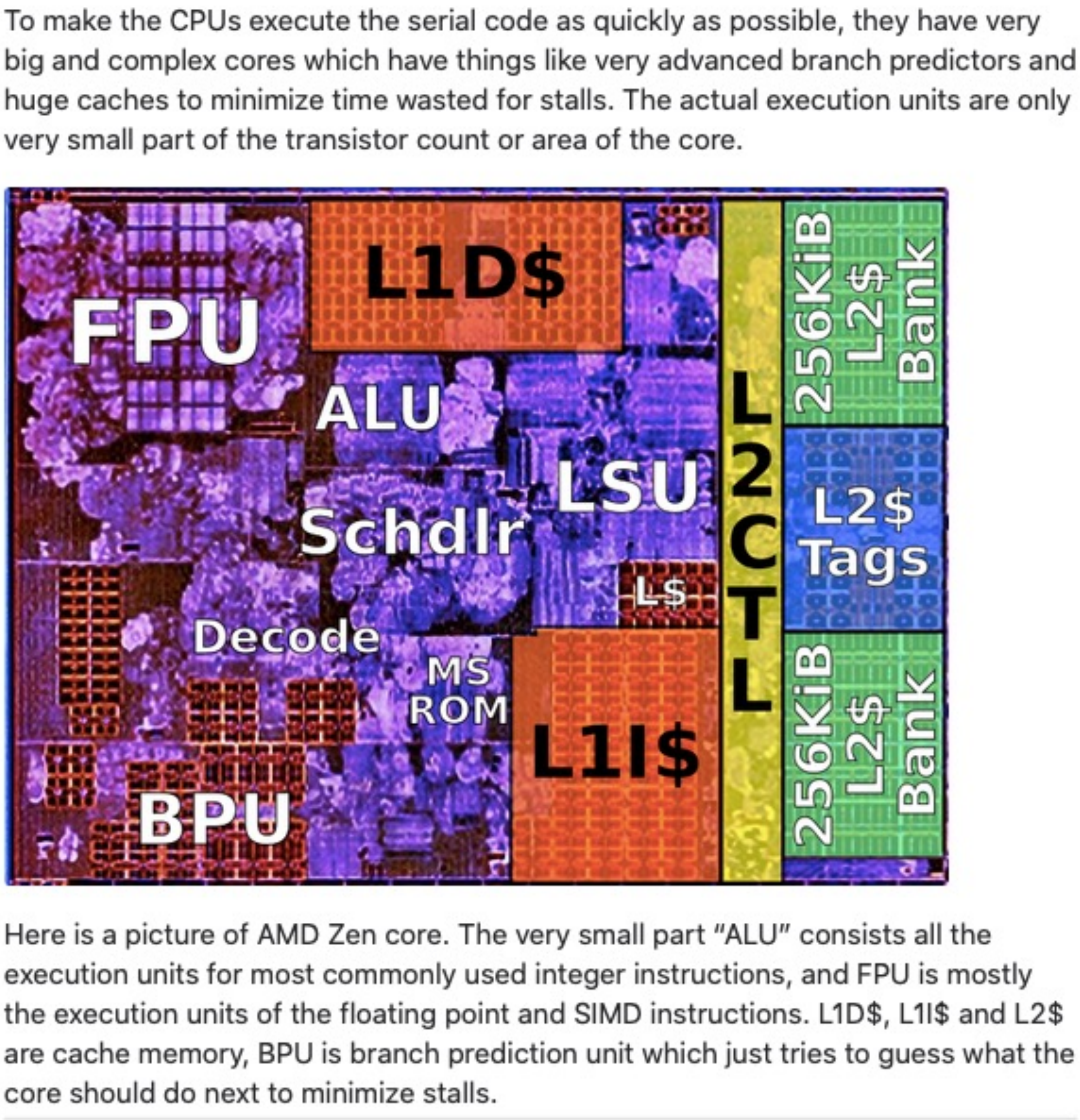
Historically, the answer is "no."

- POWER processors use 128 byte linesize. (POWER 7 onward that I can verify; not sure about earlier POWER processors.)
- Pentium, Pentium II, Pentium III, AMD K5, AMD K6, PowerPC { 603, 603e, 604, 604e, 740, 750 } all used 32 byte L1 line sizes.
- The 80486, 68030, 68040 and 68060 used a 16 byte L1 line size.
- The 80386 and earlier x86 chips do not have on-chip cache.
- The 68020 only has an on-chip instruction cache, with 64 × 4-byte entries.
- The 68000 and 68010 have no on-chip caches. The 68010 did have a small loop buffer, though.

I & D

16 Byte	4W
128 Byte	32W

AMD Zen Cores



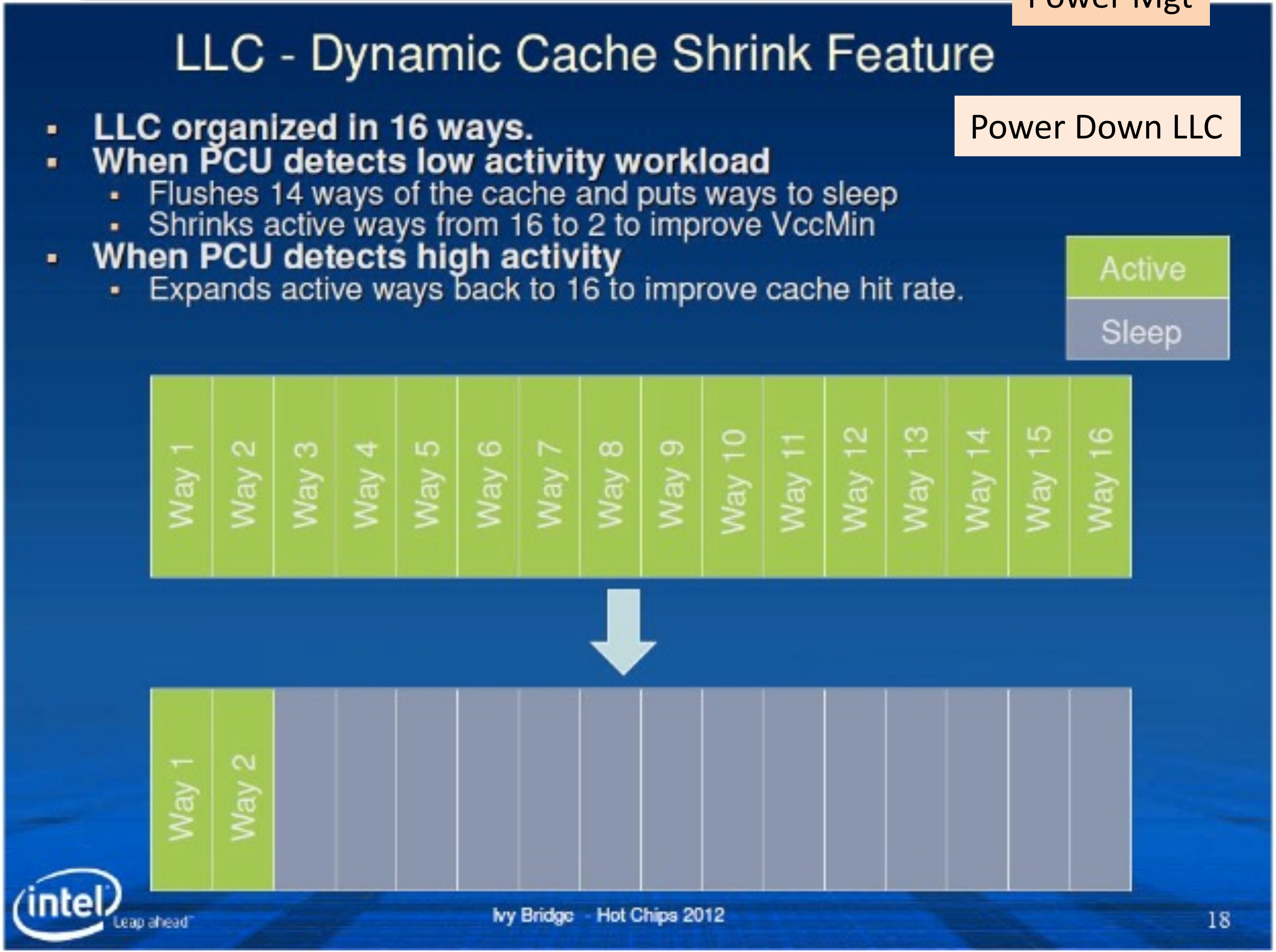
To make the CPUs execute the serial code as quickly as possible, they have very big and complex cores which have things like very advanced branch predictors and huge caches to minimize time wasted for stalls. The actual execution units are only very small part of the transistor count or area of the core.

Here is a picture of AMD Zen core. The very small part "ALU" consists all the execution units for most commonly used integer instructions, and FPU is mostly the execution units of the floating point and SIMD instructions. L1D$, L1I$ and L2$ are cache memory, BPU is branch prediction unit which just tries to guess what the core should do next to minimize stalls.

Intel CPU’s

Power Mgt

Power Down LLC



LLC - Dynamic Cache Shrink Feature

- **LLC organized in 16 ways.**
- **When PCU detects low activity workload**
  - Flushes 14 ways of the cache and puts ways to sleep
  - Shrinks active ways from 16 to 2 to improve VccMin
- **When PCU detects high activity**
  - Expands active ways back to 16 to improve cache hit rate.

🟩	Active
▨	Sleep

Way 1	Way 2	Way 3	Way 4	Way 5	Way 6	Way 7	Way 8	Way 9	Way 10	Way 11	Way 12	Way 13	Way 14	Way 15	Way 16

↓

Way 1	Way 2														

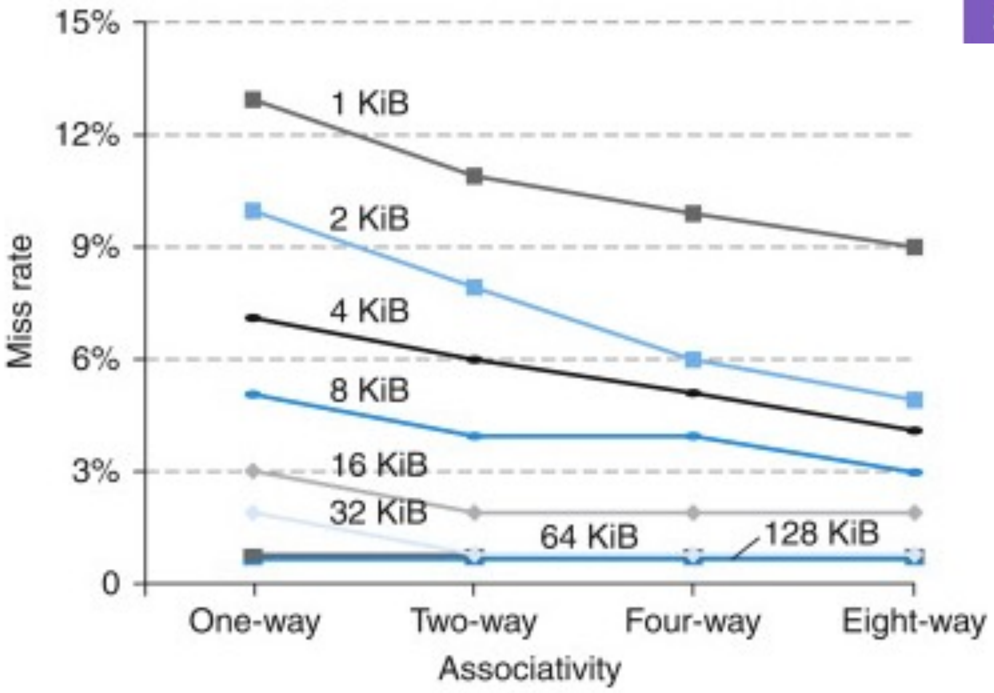
intel Leap ahead

Ivy Bridge - Hot Chips 2012

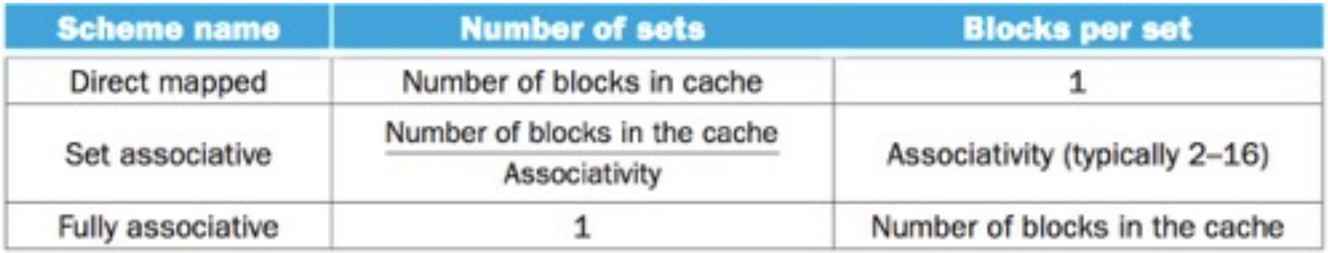
18

Cache Memory

P&H Ch 5



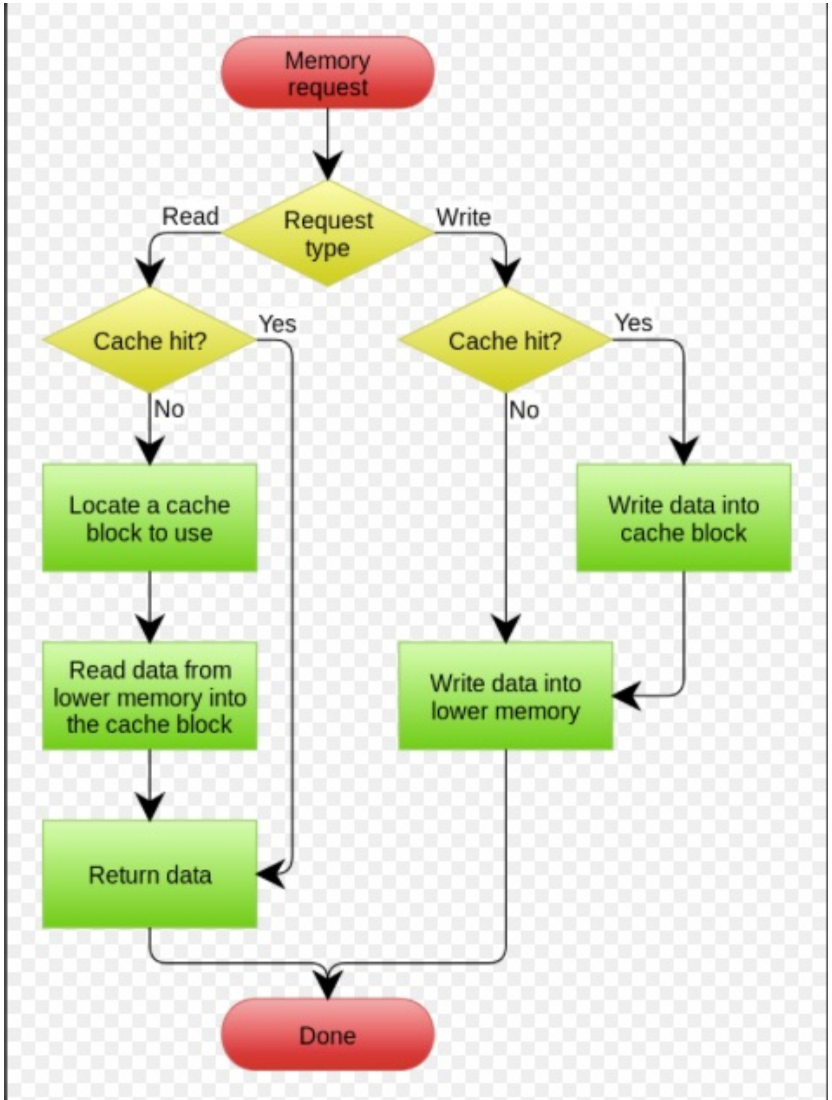
“Ways”



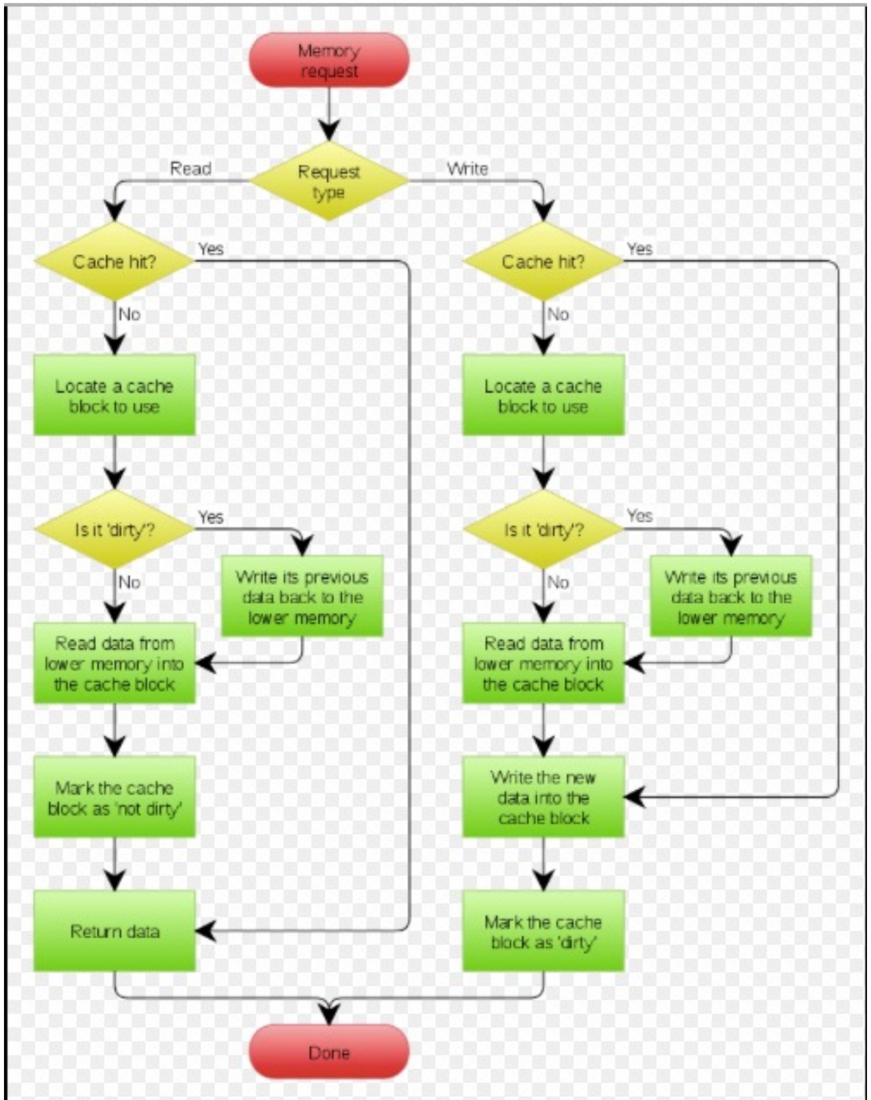
Scheme name	Number of sets	Blocks per set
Direct mapped	Number of blocks in cache	1
Set associative	$\frac{\text{Number of blocks in the cache}}{\text{Associativity}}$	Associativity (typically 2–16)
Fully associative	1	Number of blocks in the cache

Cache Write Policies

A write-through cache with no-write allocation

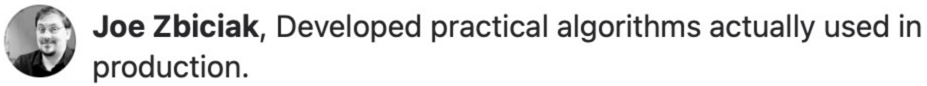

Memory request
Request type
Read
Write
Cache hit?
Yes
No
Cache hit?
Yes
No
Locate a cache block to use
Write data into cache block
Read data from lower memory into the cache block
Write data into lower memory
Return data
Done

A write-back cache with write allocation


Memory request
Request type
Read
Write
Cache hit?
Yes
No
Cache hit?
Yes
No
Locate a cache block to use
Locate a cache block to use
Is it 'dirty'?
Yes
No
Is it 'dirty'?
Yes
No
Write its previous data back to the lower memory
Write its previous data back to the lower memory
Read data from lower memory into the cache block
Read data from lower memory into the cache block
Mark the cache block as 'not dirty'
Write the new data into the cache block
Return data
Mark the cache block as 'dirty'
Done

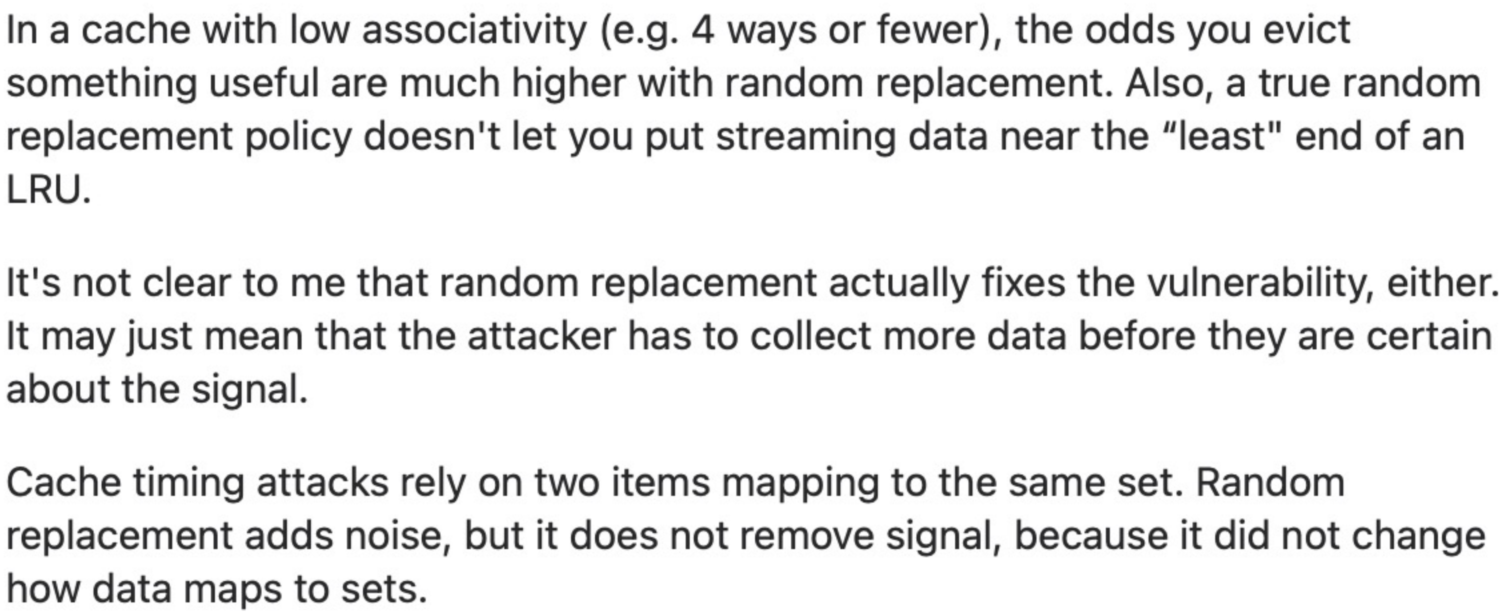
Cache Replacement

Quora



Joe Zbiciak, Developed practical algorithms actually used in production.

Hacking vulnerability	LRU vs Random replacement



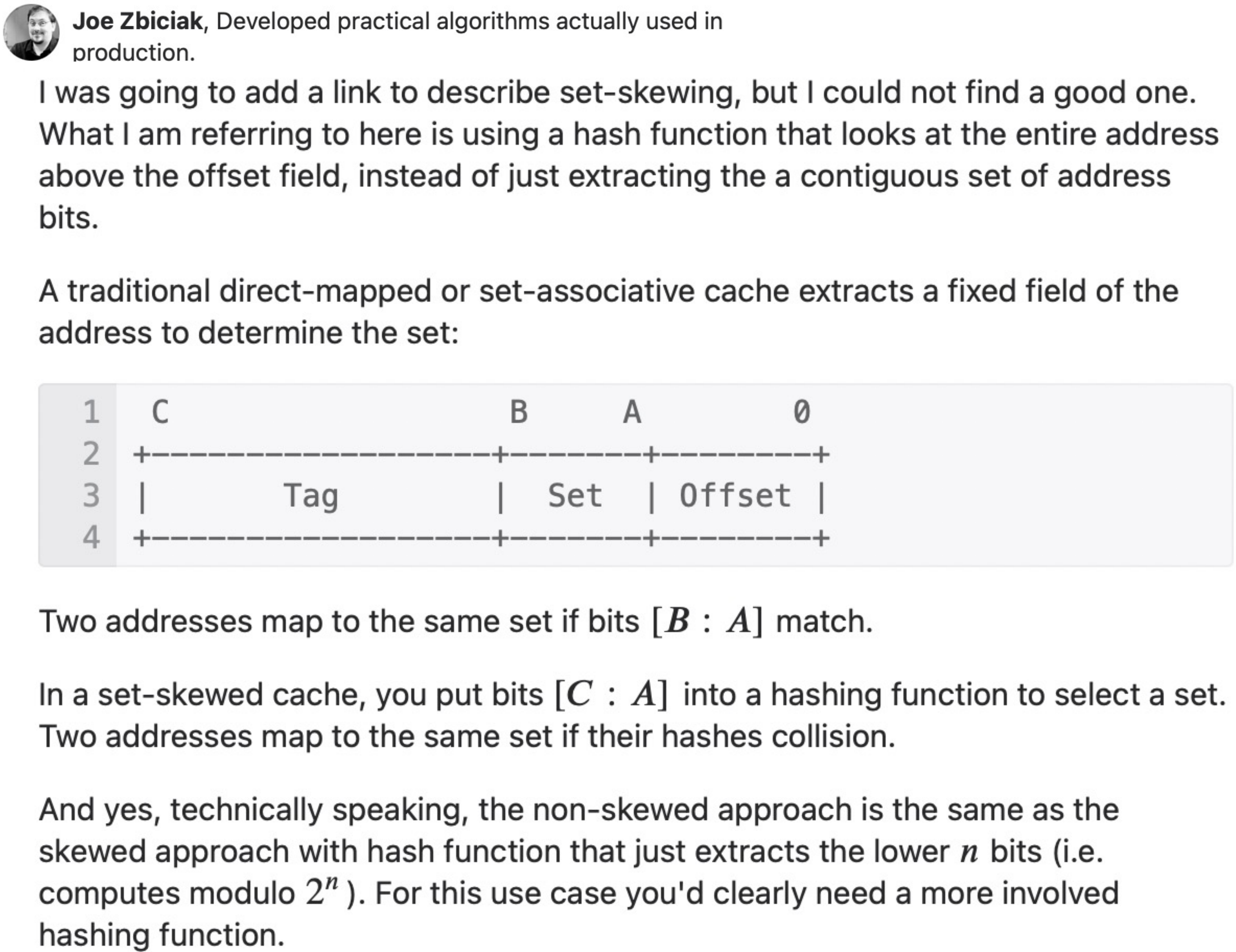
In a cache with low associativity (e.g. 4 ways or fewer), the odds you evict something useful are much higher with random replacement. Also, a true random replacement policy doesn't let you put streaming data near the "least" end of an LRU.

It's not clear to me that random replacement actually fixes the vulnerability, either. It may just mean that the attacker has to collect more data before they are certain about the signal.

Cache timing attacks rely on two items mapping to the same set. Random replacement adds noise, but it does not remove signal, because it did not change how data maps to sets.

Cache Replacement

Quora



Joe Zbiciak, Developed practical algorithms actually used in production.

I was going to add a link to describe set-skewing, but I could not find a good one. What I am referring to here is using a hash function that looks at the entire address above the offset field, instead of just extracting the a contiguous set of address bits.

A traditional direct-mapped or set-associative cache extracts a fixed field of the address to determine the set:

```
 C                   B      A         0
+-------------------+-------+---------+
|        Tag        |  Set  | Offset |
+-------------------+-------+---------+
```

Two addresses map to the same set if bits $[B : A]$ match.

In a set-skewed cache, you put bits $[C : A]$ into a hashing function to select a set. Two addresses map to the same set if their hashes collision.

And yes, technically speaking, the non-skewed approach is the same as the skewed approach with hash function that just extracts the lower n bits (i.e. computes modulo 2^n). For this use case you'd clearly need a more involved hashing function.

Memory

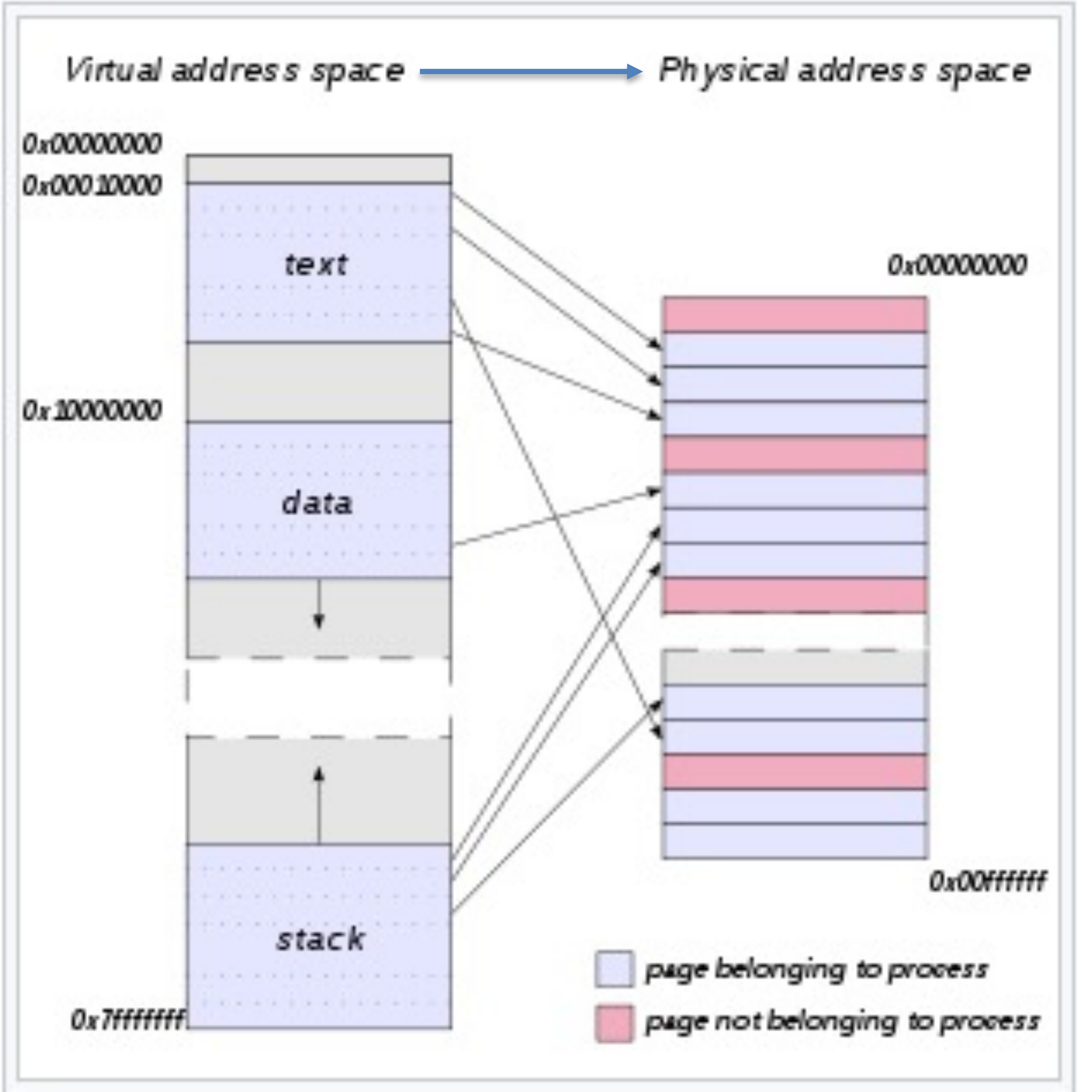
Virtual Memory

MMU/TLB

CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE
COMP122

Virtual Memory

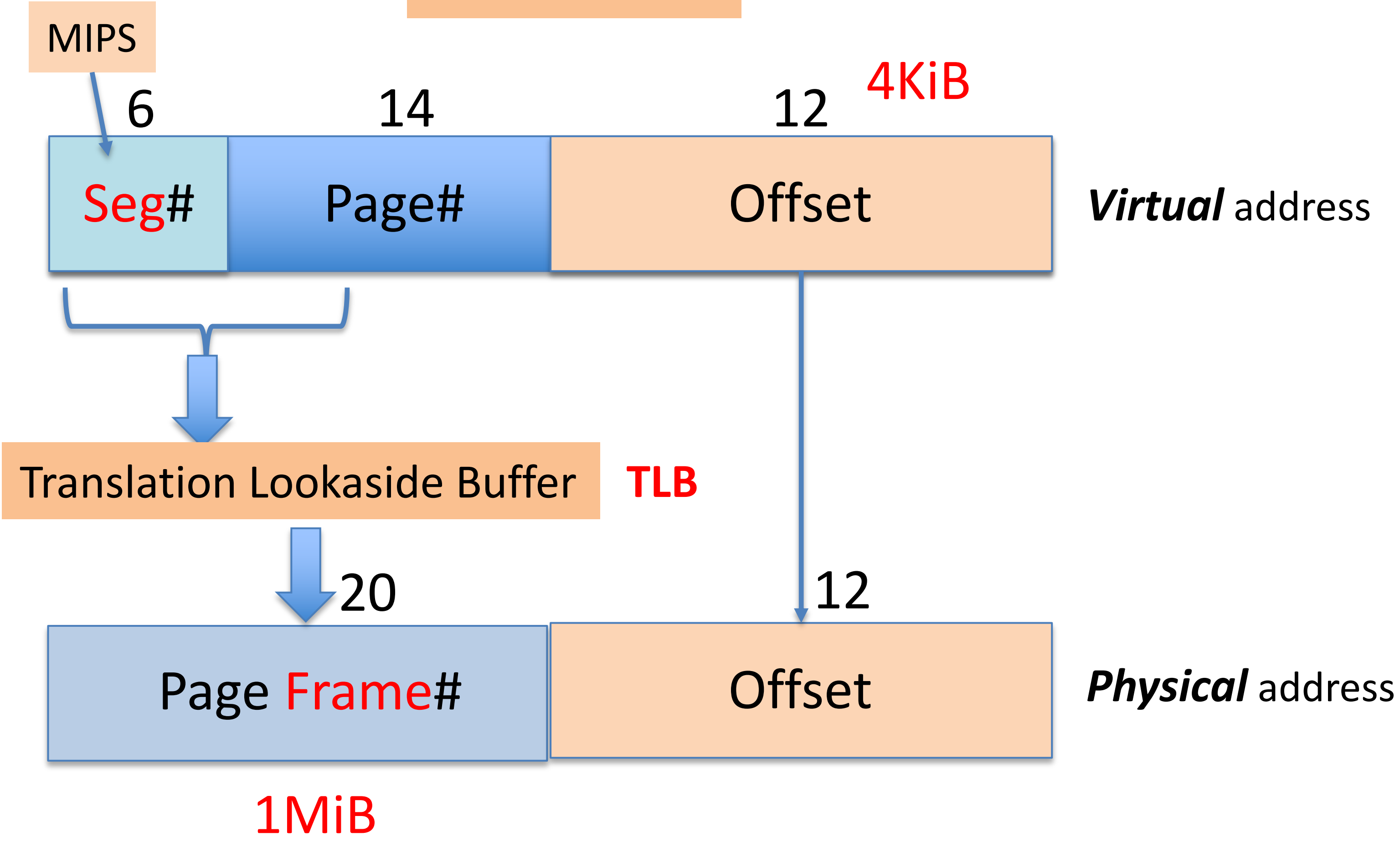
DR JEFF SOFTWARE
INDIE APP DEVELOPER
© Jeff Drobman
2016-2022



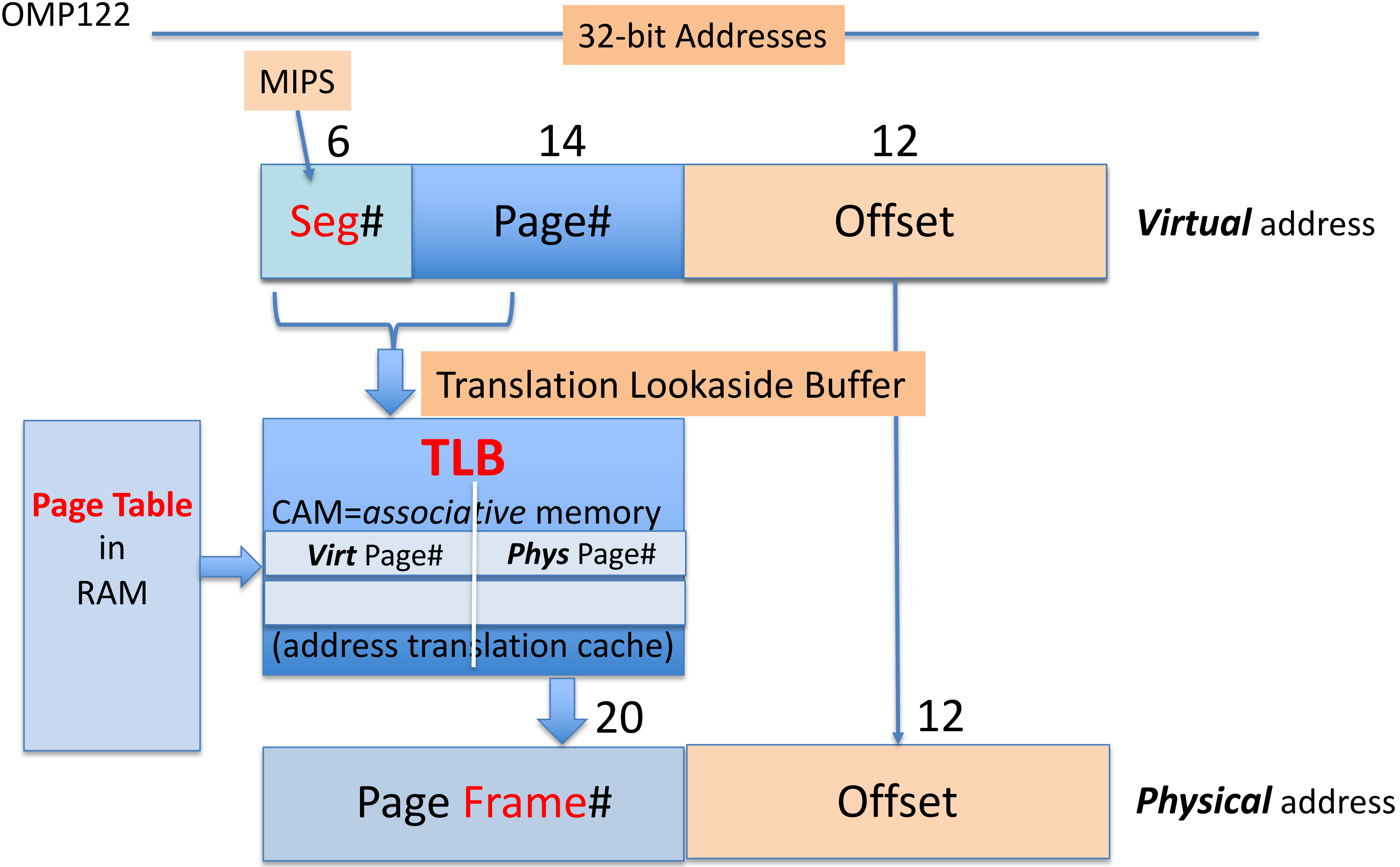
CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE
COMP122

Virtual Memory

32-bit Addresses



Virtual Memory



32-bit Addresses

MIPS

6 | 14 | 12

Seg# | Page# | Offset

Virtual address

Translation Lookaside Buffer

Page Table
in
RAM

TLB

CAM=*associative* memory

Virt Page# | ***Phys*** Page#

(address translation cache)

20 | 12

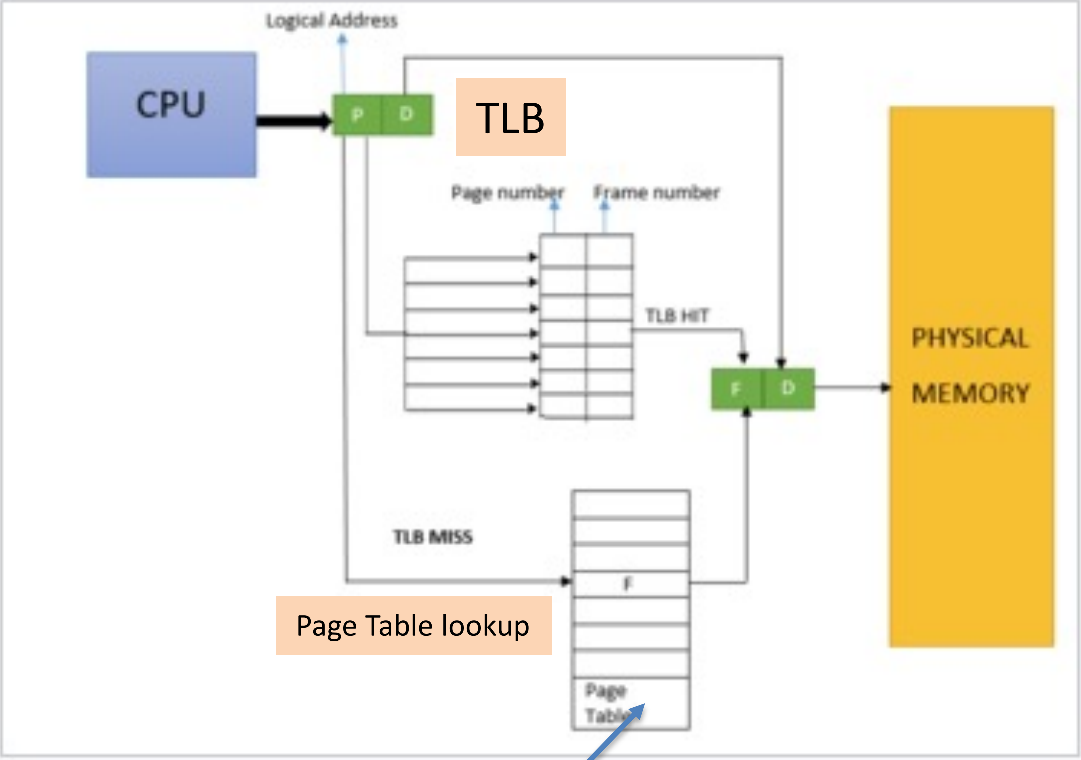
Page **Frame**# | Offset

Physical address

CSUN
CALIFORNIA STATE UNIVERSITY NORTHRIDGE
COMP122

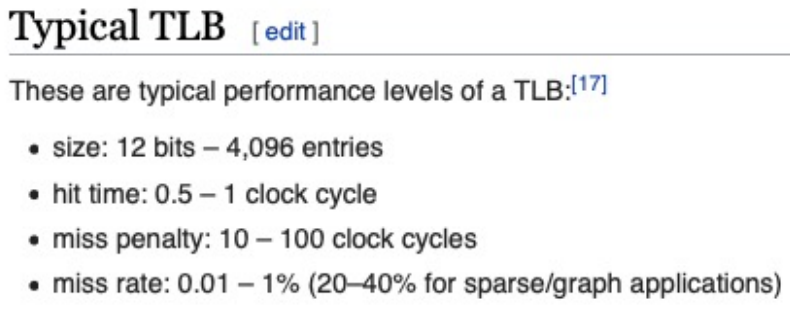
Virtual Memory

DR JEFF SOFTWARE
INDIE APP DEVELOPER
© Jeff Drobman
2016-2022



General working of TLB[4]

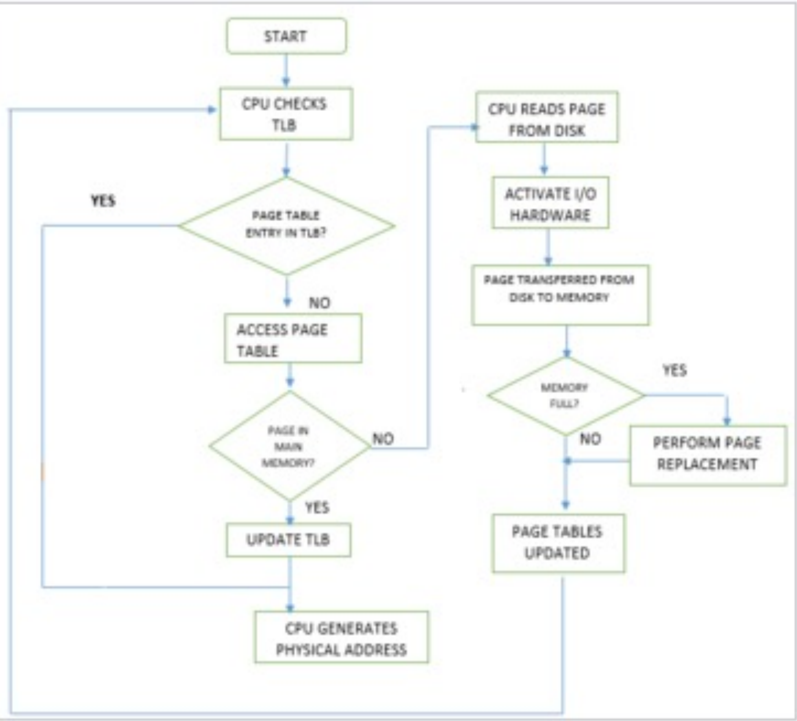
Page Table



Typical TLB [edit]

These are typical performance levels of a TLB:[17]

- size: 12 bits – 4,096 entries
- hit time: 0.5 – 1 clock cycle
- miss penalty: 10 – 100 clock cycles
- miss rate: 0.01 – 1% (20–40% for sparse/graph applications)



Flowchart[6] shows the working of a translation lookaside buffer. For simplicity, the page-fault routine is not mentioned.

CSUN
CALIFORNIA
STATE UNIVERSITY
NORTHRIDGE

COMP122

Virtual Memory

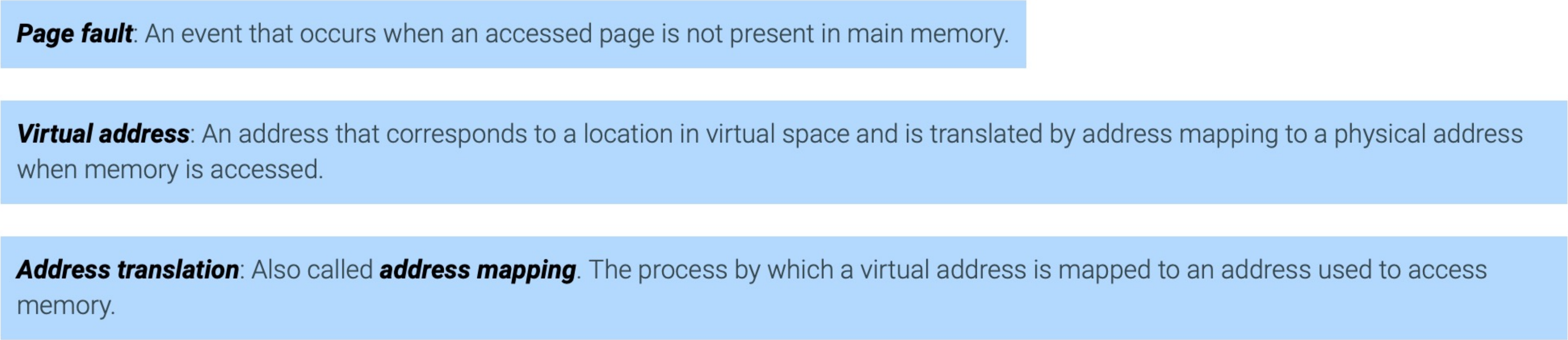
DR JEFF
SOFTWARE
INDIE APP DEVELOPER

P&H Ch 5

COMP 122: Computer Architecture and Assembly Language
Spring 2020

5.7 Virtual memory

Figure 5.7.2: Mapping from a virtual to a physical address


Page fault: An event that occurs when an accessed page is not present in main memory.

Virtual address: An address that corresponds to a location in virtual space and is translated by address mapping to a physical address when memory is accessed.

Address translation: Also called ***address mapping***. The process by which a virtual address is mapped to an address used to access memory.

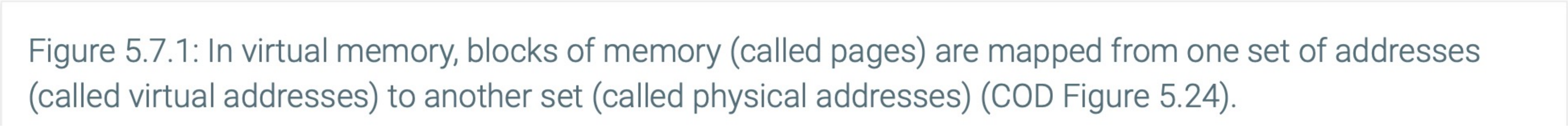


Figure 5.7.1: In virtual memory, blocks of memory (called pages) are mapped from one set of addresses (called virtual addresses) to another set (called physical addresses) (COD Figure 5.24).

Virtual Memory

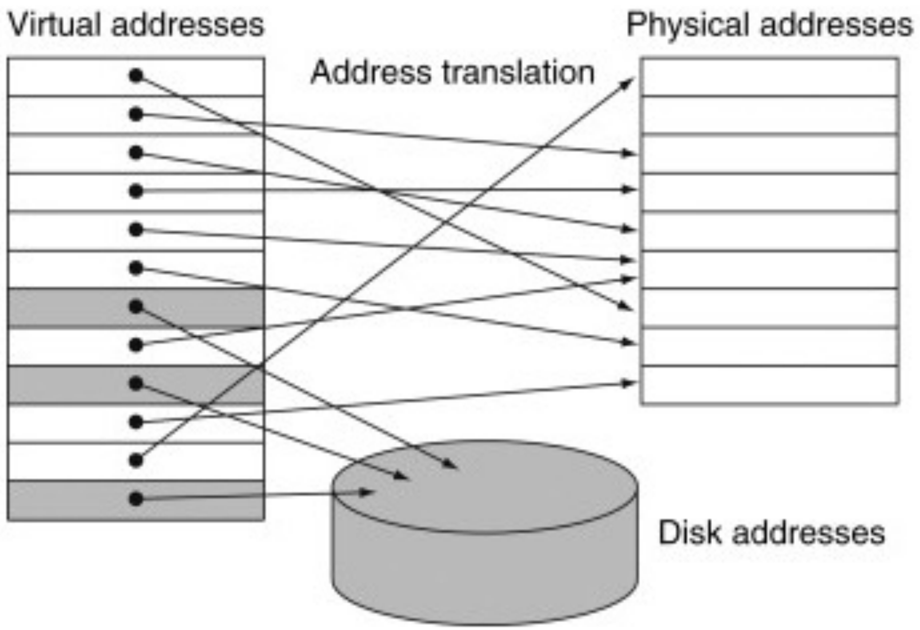
P&H Ch 5

COMP 122: Computer Architecture and Assembly Language
Spring 2020

5.7 Virtual memory

Figure 5.7.2: Mapping from a virtual to a physical address

Figure 5.7.1: In virtual memory, blocks of memory (called pages) are mapped from one set of addresses (called virtual addresses) to another set (called physical addresses) (COD Figure 5.24).

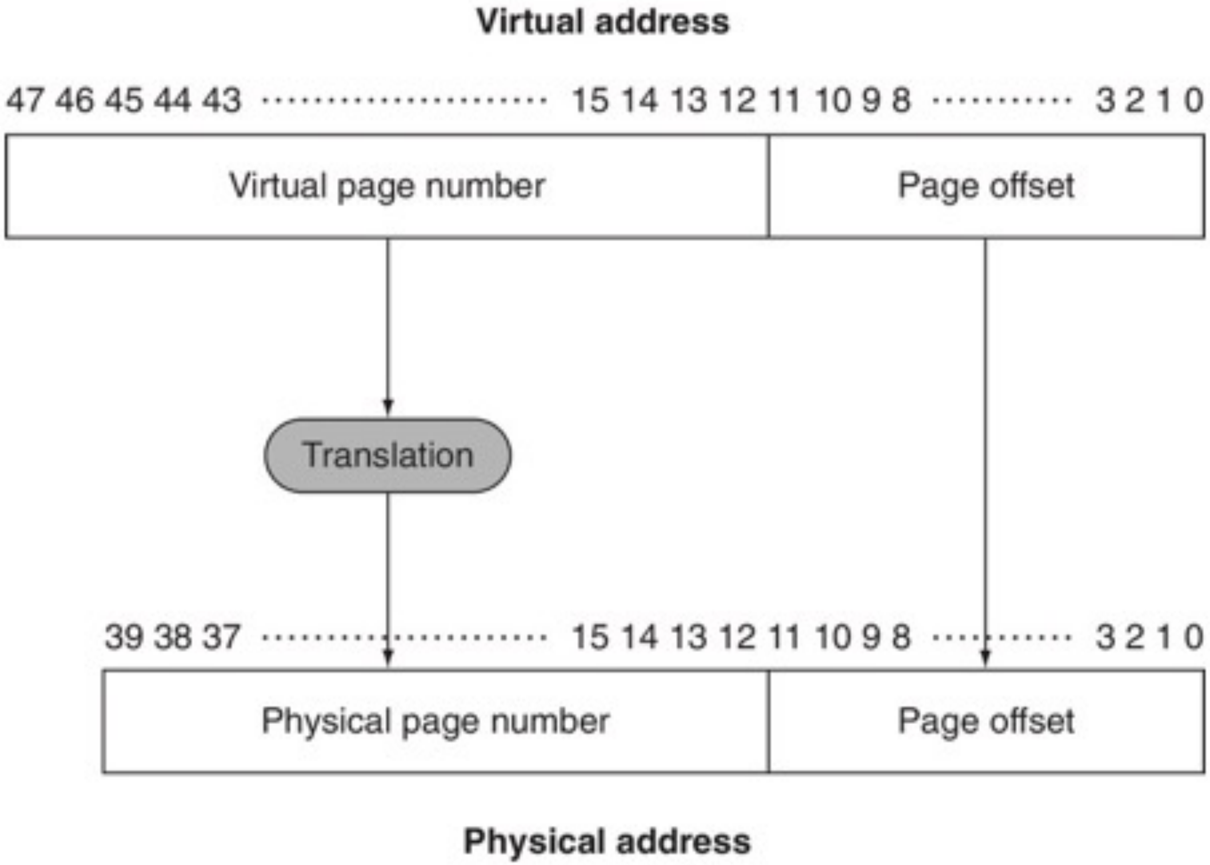


Virtual Memory

TLB

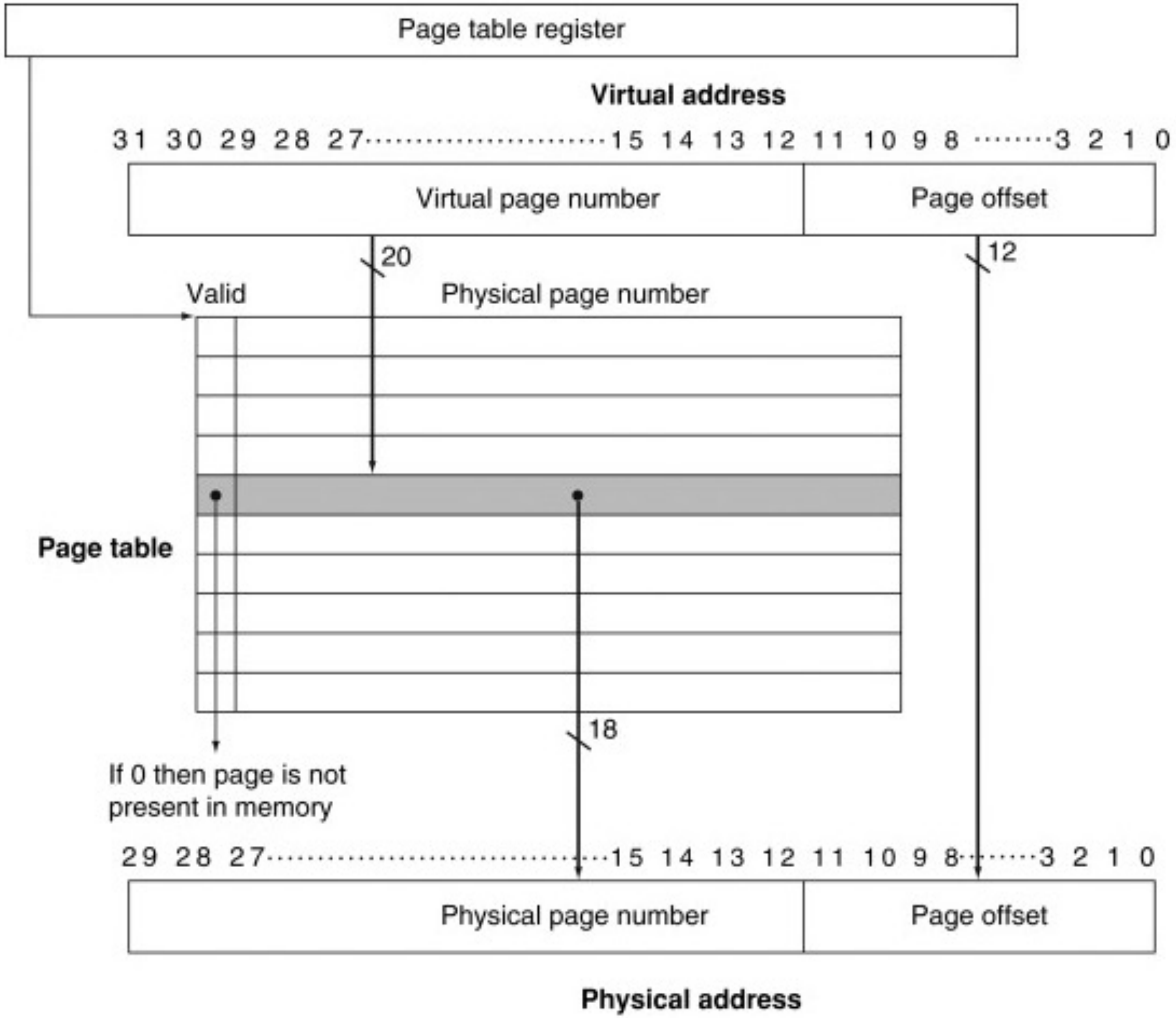
P&H Ch 5

Figure 5.7.2: Mapping from a virtual to a physical address



Virtual Memory

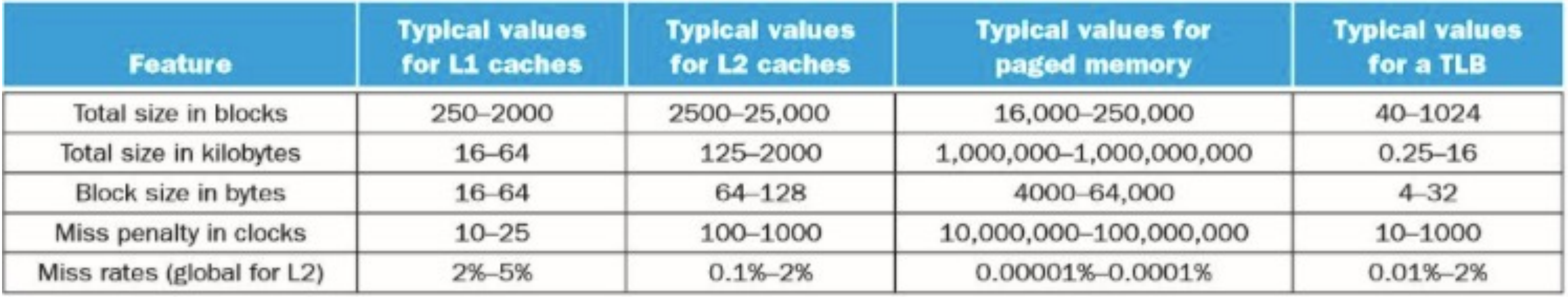
P&H Ch 5



MLM Sizes/Hits

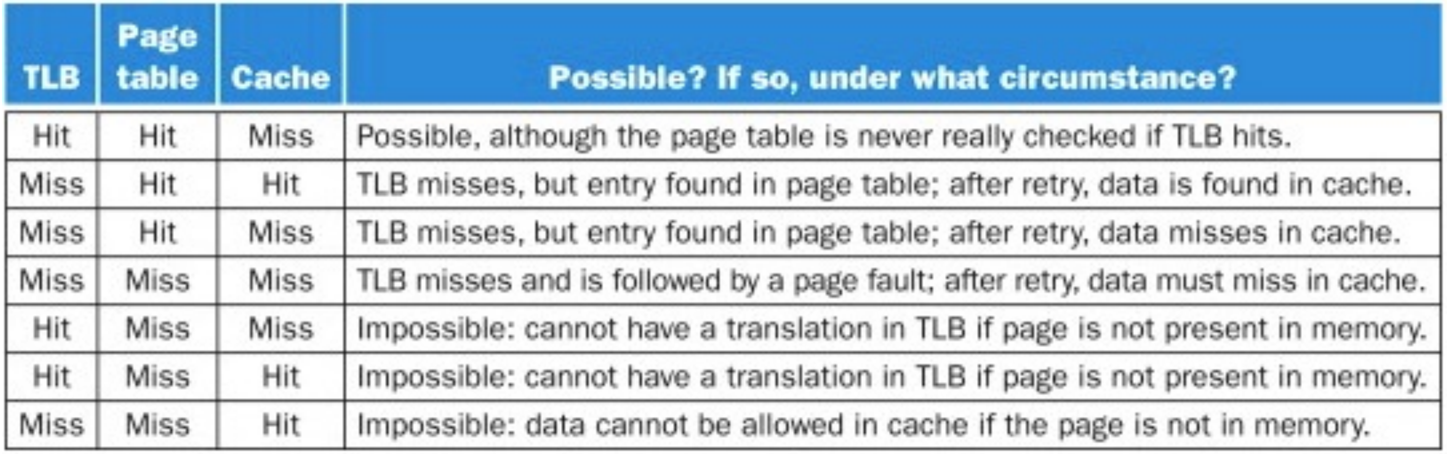
P&H Ch 5

COMP 122: Computer Architecture and Assembly Language
Spring 2020



Feature	Typical values for L1 caches	Typical values for L2 caches	Typical values for paged memory	Typical values for a TLB
Total size in blocks	250–2000	2500–25,000	16,000–250,000	40–1024
Total size in kilobytes	16–64	125–2000	1,000,000–1,000,000,000	0.25–16
Block size in bytes	16–64	64–128	4000–64,000	4–32
Miss penalty in clocks	10–25	100–1000	10,000,000–100,000,000	10–1000
Miss rates (global for L2)	2%–5%	0.1%–2%	0.00001%–0.0001%	0.01%–2%

TLB

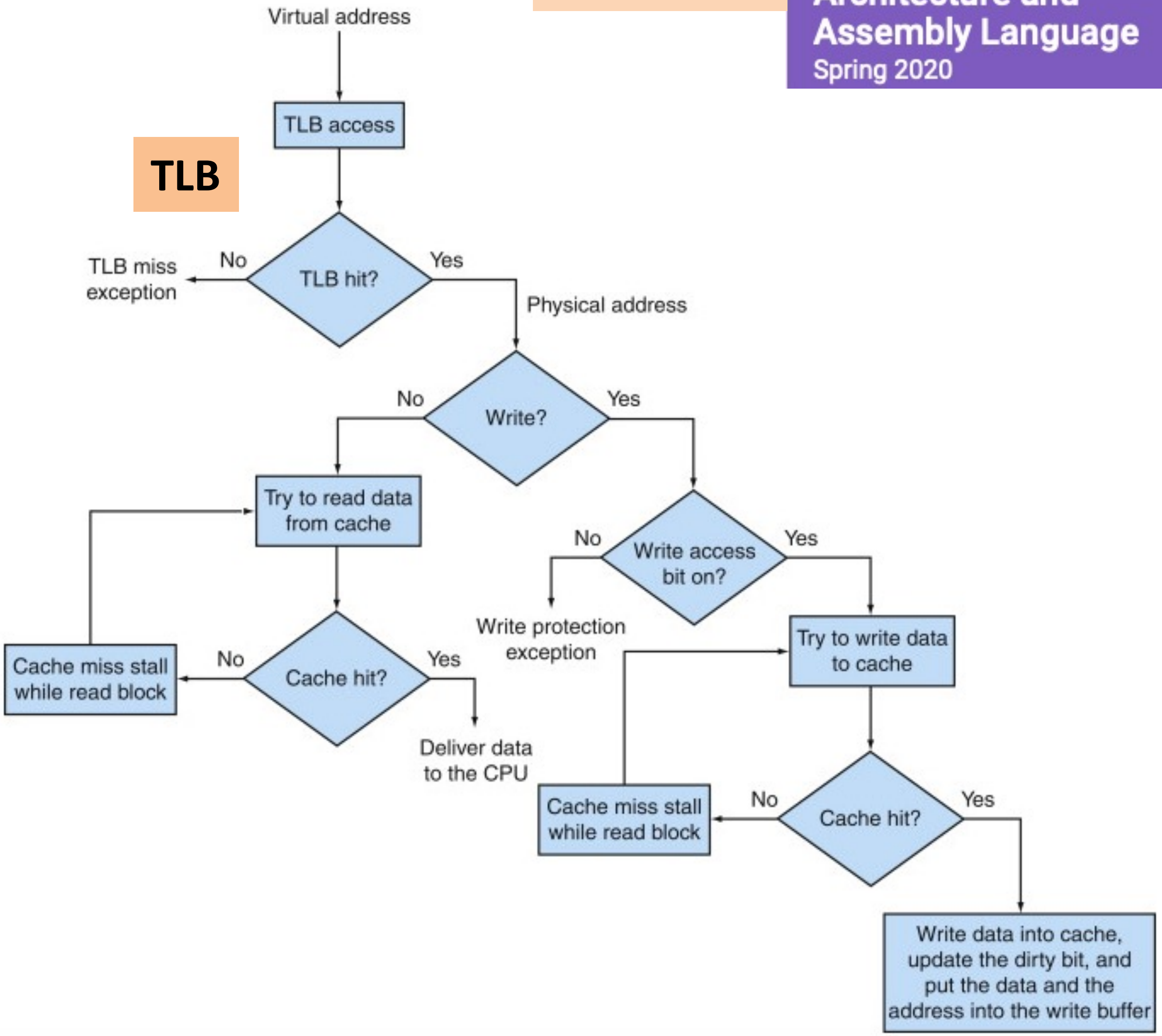


TLB	Page table	Cache	Possible? If so, under what circumstance?
Hit	Hit	Miss	Possible, although the page table is never really checked if TLB hits.
Miss	Hit	Hit	TLB misses, but entry found in page table; after retry, data is found in cache.
Miss	Hit	Miss	TLB misses, but entry found in page table; after retry, data misses in cache.
Miss	Miss	Miss	TLB misses and is followed by a page fault; after retry, data must miss in cache.
Hit	Miss	Miss	Impossible: cannot have a translation in TLB if page is not present in memory.
Hit	Miss	Hit	Impossible: cannot have a translation in TLB if page is not present in memory.
Miss	Miss	Hit	Impossible: data cannot be allowed in cache if the page is not in memory.

Virtual Memory

P&H Ch 5

TLB

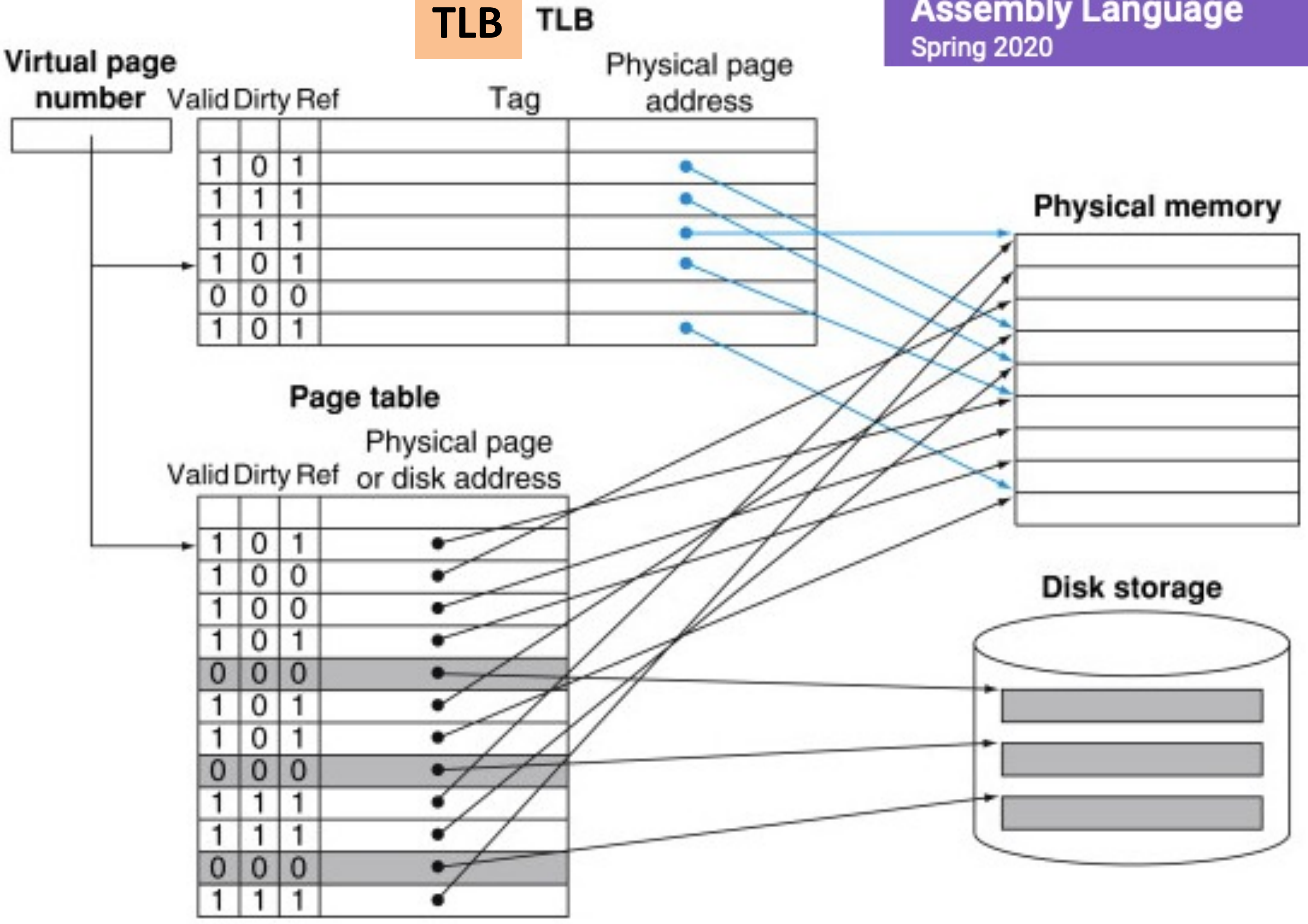

Virtual address
TLB access
TLB hit?
No
TLB miss exception
Yes
Physical address
Write?
No
Yes
Try to read data from cache
Write access bit on?
No
Yes
Write protection exception
Try to write data to cache
Cache miss stall while read block
Cache hit?
No
Yes
Deliver data to the CPU
Cache miss stall while read block
Cache hit?
No
Yes
Write data into cache, update the dirty bit, and put the data and the address into the write buffer

Virtual Memory

P&H Ch 5

COMP 122: Computer Architecture and Assembly Language
Spring 2020

TLB



TLB

Valid	Dirty	Ref	Tag	Physical page address
1	0	1		
1	1	1		
1	1	1		
1	0	1		
0	0	0		
1	0	1		

Page table

Valid	Dirty	Ref	Physical page or disk address
1	0	1	
1	0	0	
1	0	0	
1	0	1	
0	0	0	
1	0	1	
1	0	1	
0	0	0	
1	1	1	
1	1	1	
0	0	0	
1	1	1	