Introduction on Computer / xPU Architecture

Chixiao Chen

cxchen@fudan.edu.cn

Overview

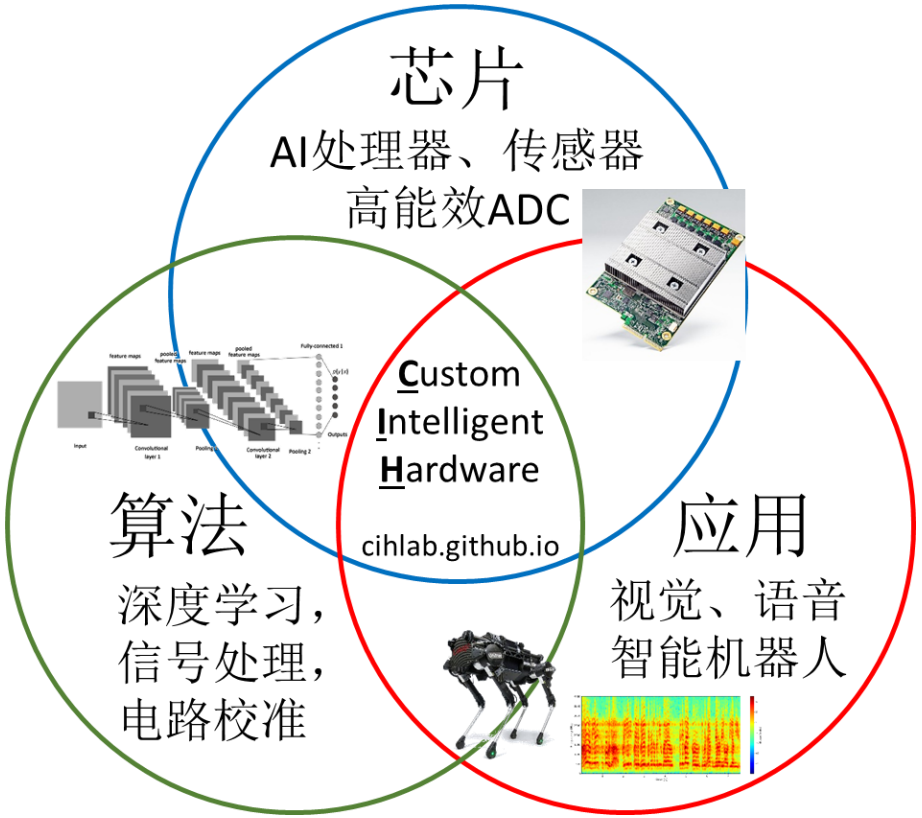
- Course Overview
- A Brief Review of Computer Architecture History
- Why AI need specific architecture?

Course Overview

- Website: https://cihlab.github.io/course/ai19.html

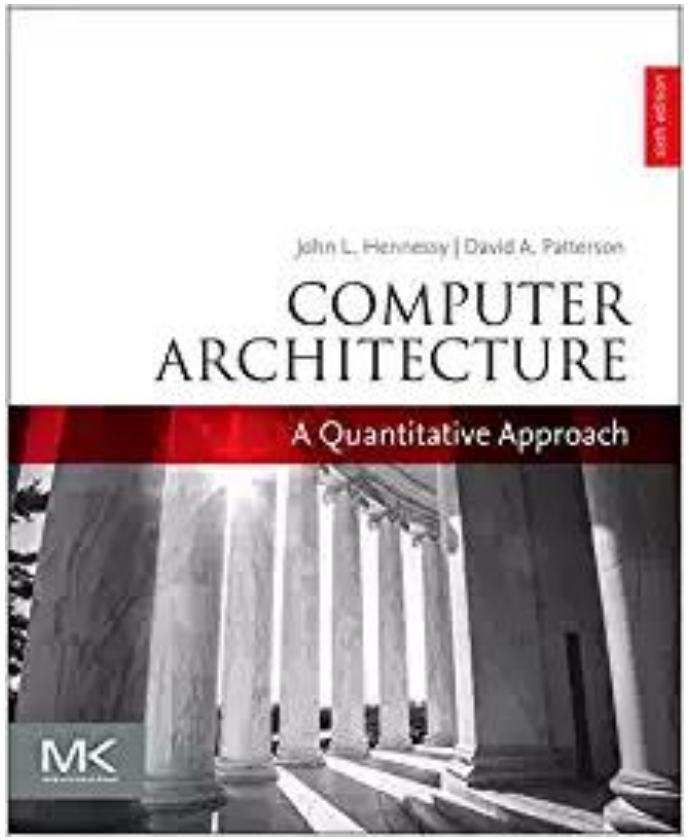
- Instructor:

  Chixiao Chen, Ph.D
  Currently, I am assistant prof. in FAET.

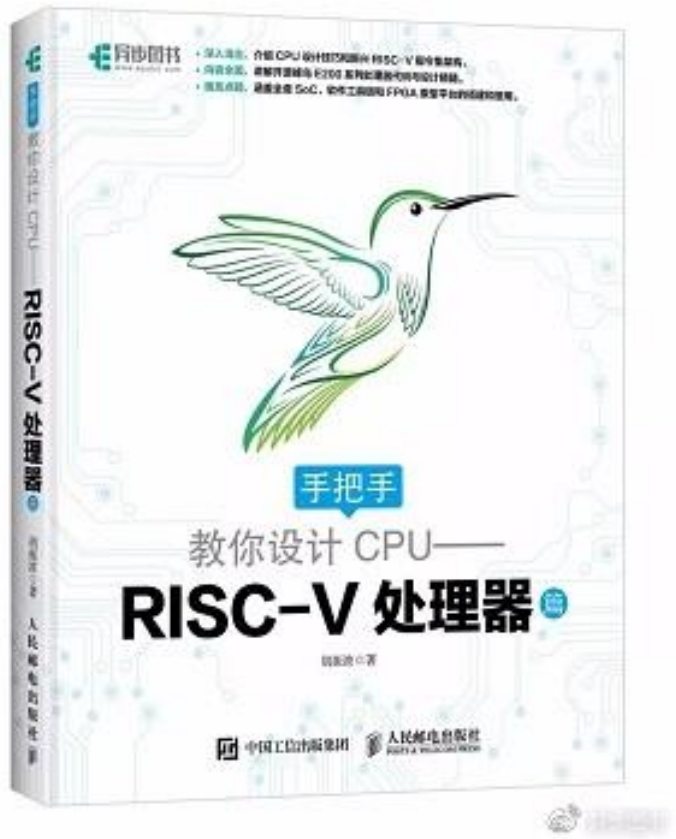


Textbooks

- Computer Architecture Bible



- More “接地气” one



What course about?

- Week 2-6
  Basic computer architecture
  (RISC-V based, RISC-V foundation *)

- Week 7-8
  Introduction on deep learning

- Week 9-14
  AI processor architecture

Course Calendar

Week	Date	Lecture Title	Reading / Slides	Homework
1	Feb 28	Introduction on Computer/AI Architecuture	/	/
2	Mar 7	Instruction Set, CISC vs. RISC		/
3	Mar 14	Pipeline, Branch Prediction		/
4	Mar 21	Von Neumman Arch, Memory Hierarchy		HW1 Due
5	Mar 28	Accelerator and Co-processor		/
7	Apr 4	Introduction on Machine Learning		HW2 Due
8	Apr 11	Convolution Neural Network		/
9	Apr 18	Recurrent Neural Network, LSTM, GRU		/
10	Apr 25	Domain Specific Architecutre, RISC-V (Guest Lecturer)		HW3 Due
11	May 2	Accelerator for AI, Data Flow / TPU		/
12	May 9	ISA for AI, SIMD/VLIW/Superscalar/Cambricon		/
13	May 16	Many-Core for AI, GPU / CGRA		HW4 Due
14	May 23	Algorithm/Software/Hardware Codesign		/
15	Jun 6	No Class (Working on Final Project)		/
16	Jun 13	Presentation Day		Final PJ Due

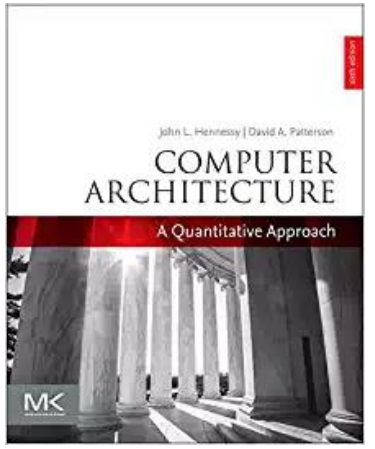
How about grading?

- Homework Assignment (15% x 4)+ Final Project (40%)
- No finals / No mid-terms
- The last day of instruction is the final presentation day.
- Any Questions / Comments?

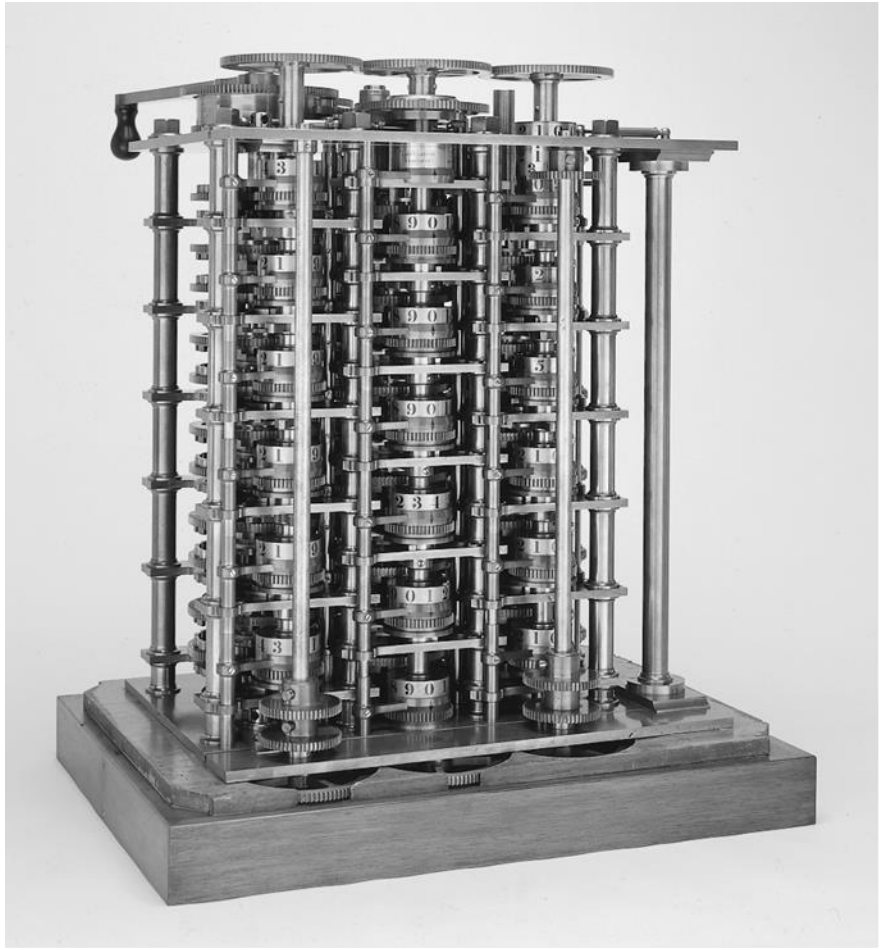
A Brief History Review on Computer Architectures

- Courtesy by "**A New Golden Age for Computer Architecture**" ACM Communications, and Prof. Patterson's ISSCC 2018 Keynote

Turing Award 2017



The First Computer



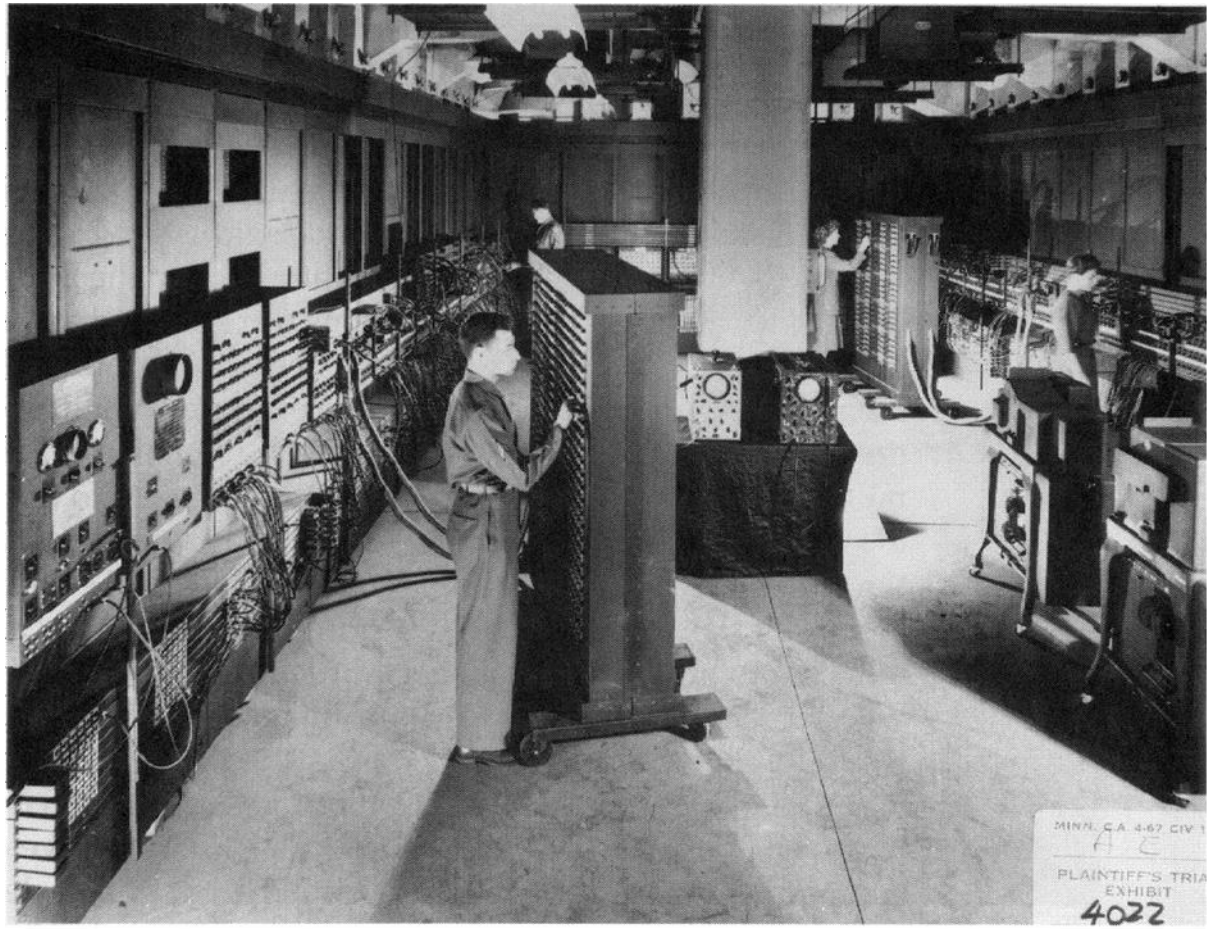
- use the decimal number system rather than the binary representation
-"store" and "mill" (execute), 2 cycle opn.
- used pipelining to speed up the execution of the addition operation
-complexity and the cost problem

The Babbage Difference Engine (1832)

25,000 parts

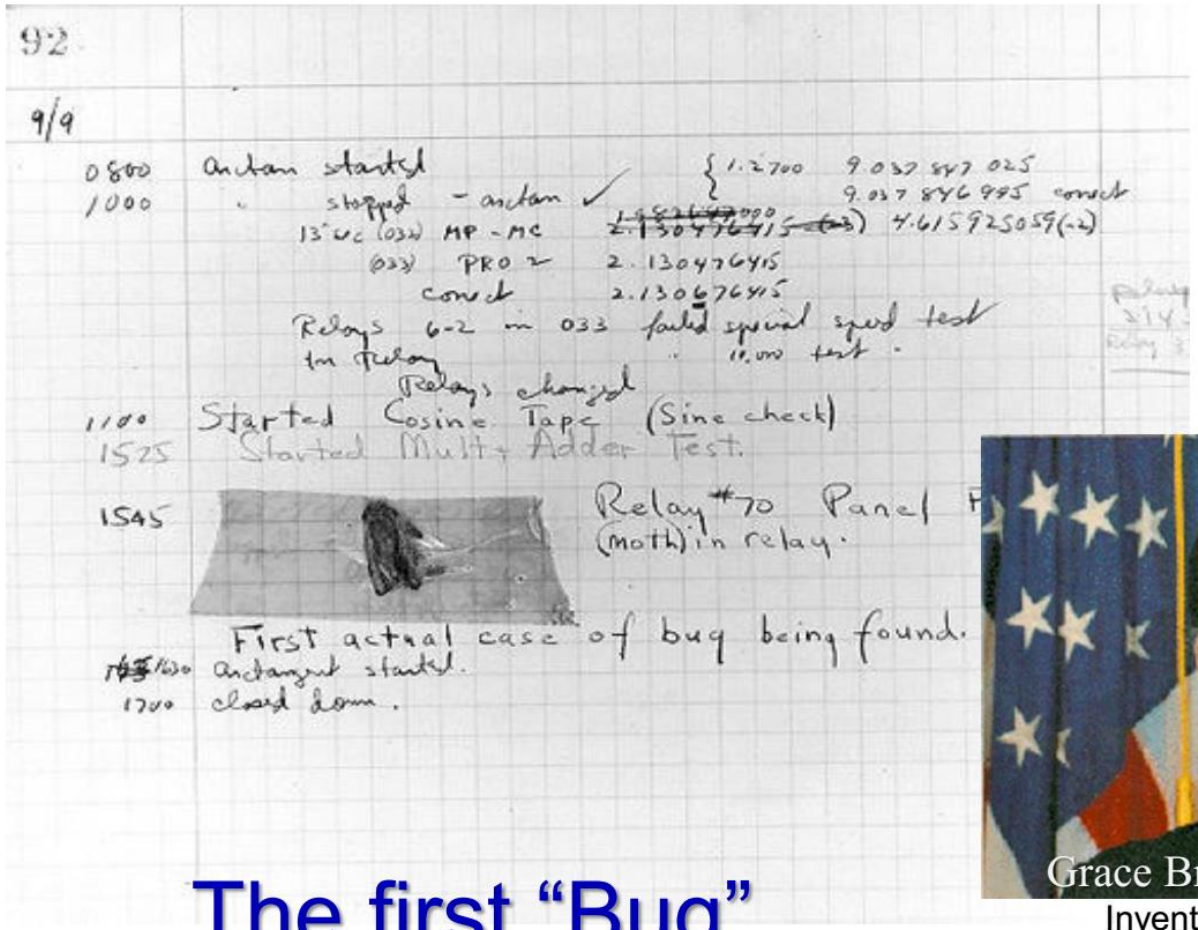
cost: £17,470

The First Electronic Computer

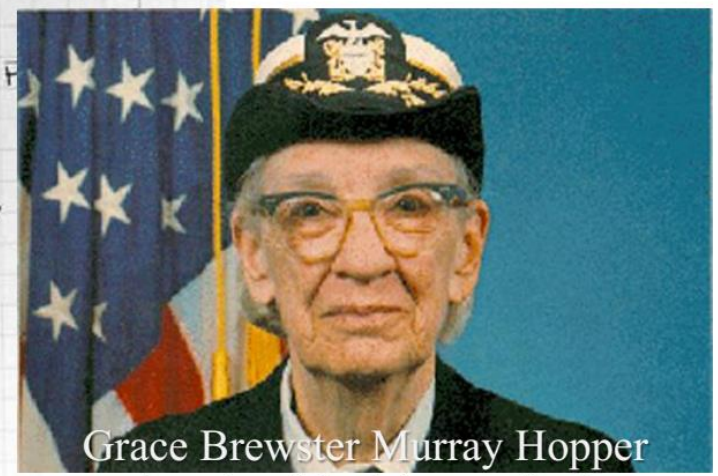


- ***vacuum tube*** *based computer*
- *UNIVAC I (the first successful commercial computer)*
-***Integration Density:*** *80 feet long, 8.5 feet high and several feet wide and incorporated 18,000 vacuum tubes.*
- ***Reliability*** *problems and excessive* ***power consumption*** *made the implementation of larger engines economically and practically infeasible.*

The First Bug

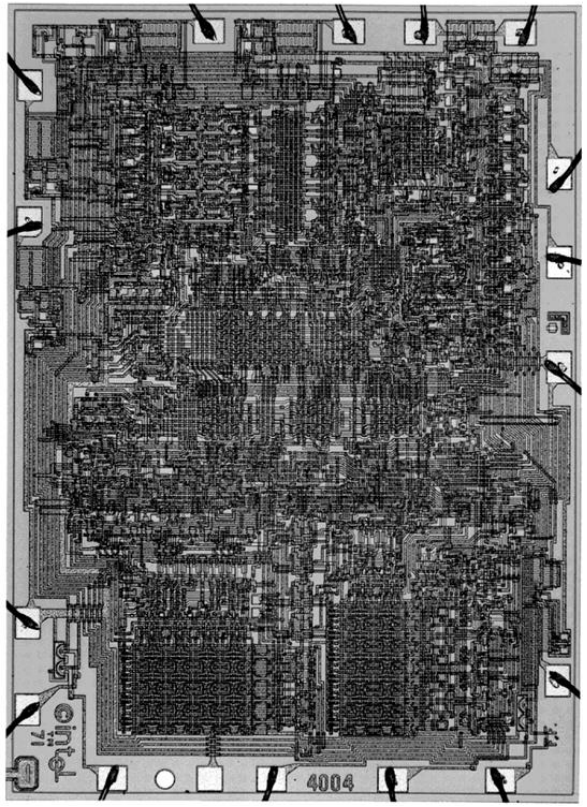


The first "Bug"



Inventor of the infamous Bug!

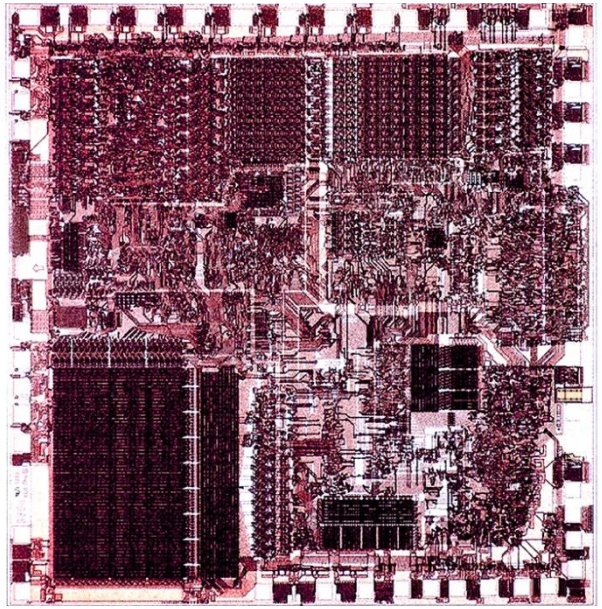
Intel 4004 & 8008 Processor



First 4Kbit MOS memory :1970

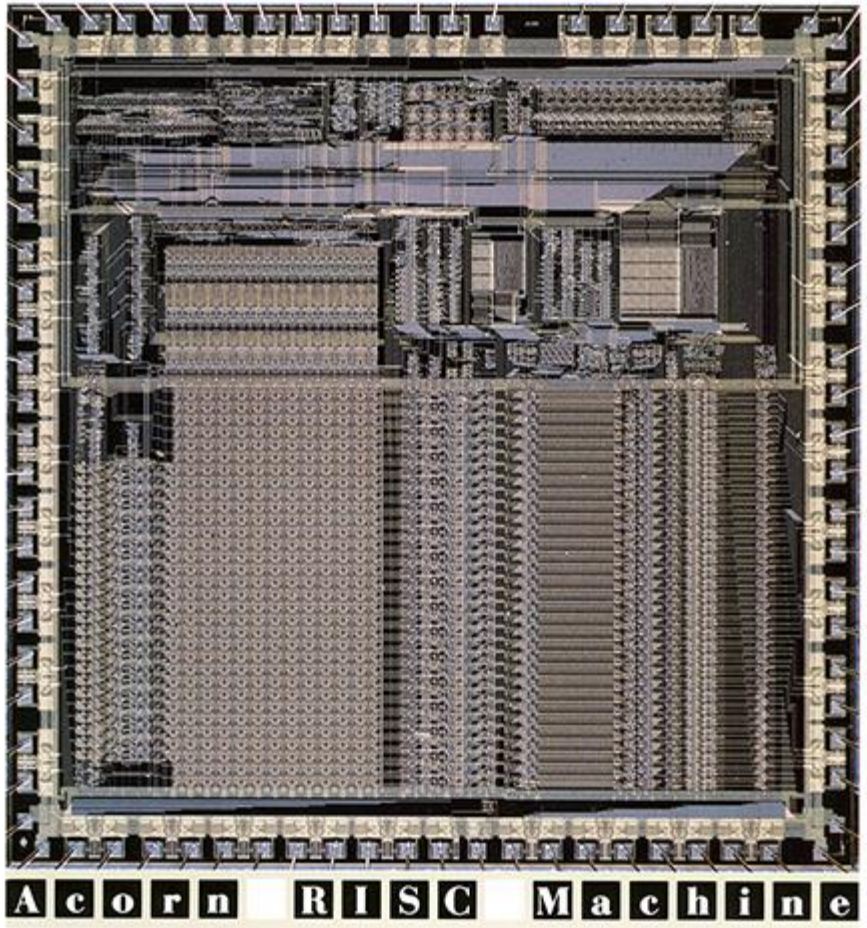
1971
1000 transistors
1 MHz operation
Handcrafted

These processors were implemented in **NMOS-only** logic, which has the advantage of **higher speed** over the PMOS logic.



In an odd twist of fate, the chip that established what would become known as the x86 architecture didn't have a name appended with an "86." The 8088 was basically a slightly modified 8086, Intel's first 16-bit CPU.

First ARM Processor



In the early 1980s, Acorn Computers was a small company with a big product. The firm, based in Cambridge, England, had sold over 1.5 million 8-bit BBC Micro desktop computers as part of the BBC's national Computer Literacy Project. It was now time to design a new computer. Unsatisfied with the processors then available on the market, the Acorn engineers decided to make the leap to creating their own 32-bit microprocessor.

"The team was so small that every design decision had to favor simplicity—or we'd never finish it!" says codesigner Steve Furber, now a computer engineering professor at the University of Manchester. In the end, the simplicity made all the difference. The ARM was small, low power, and easy to program.

IBM Compatibility Problem in Early 1960s

By early 1960's, *IBM had 4 incompatible lines of computers!*

701	→	7094
650	→	7074
702	→	7080
1401	→	7010

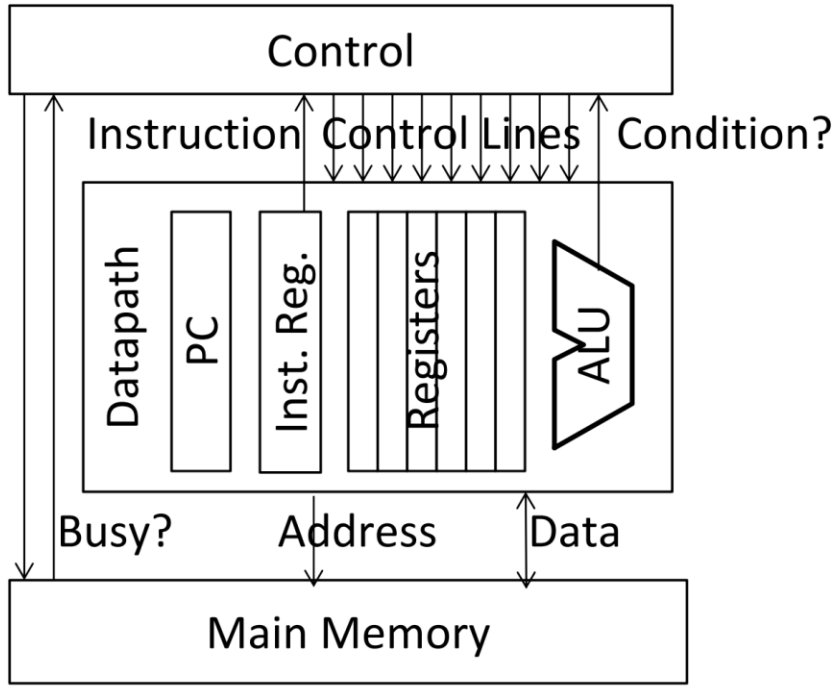
Each system had its own:

- Instruction set architecture (ISA)
- I/O system and Secondary Storage:
- magnetic tapes, drums and disks
- Assemblers, compilers, libraries,...
- Market niche: business, scientific, real time, ...

IBM System/360 – one ISA to rule them all

Control versus Datapath

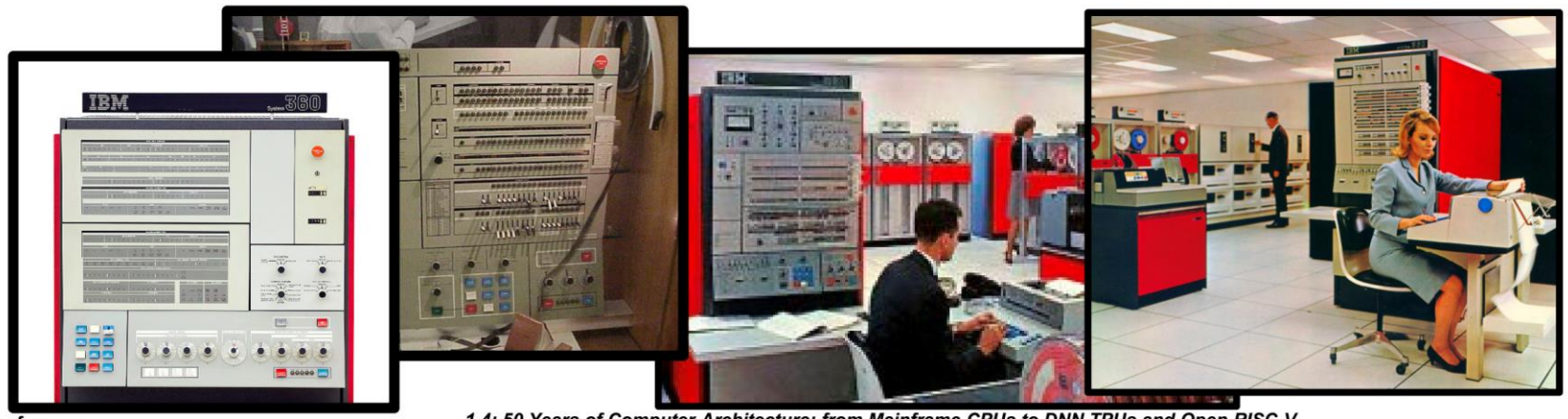
- Processor designs can be split between *datapath*, where numbers are stored and arithmetic operations computed, and *control*, which sequences operations on datapath
- Biggest challenge for early computer designers was getting control circuitry correct



- Maurice Wilkes invented the idea of *microprogramming* to design the control unit of a processor, 1958
  - Logic expensive compared to ROM or RAM
  - ROM cheaper than RAM
  - ROM much faster than RAM

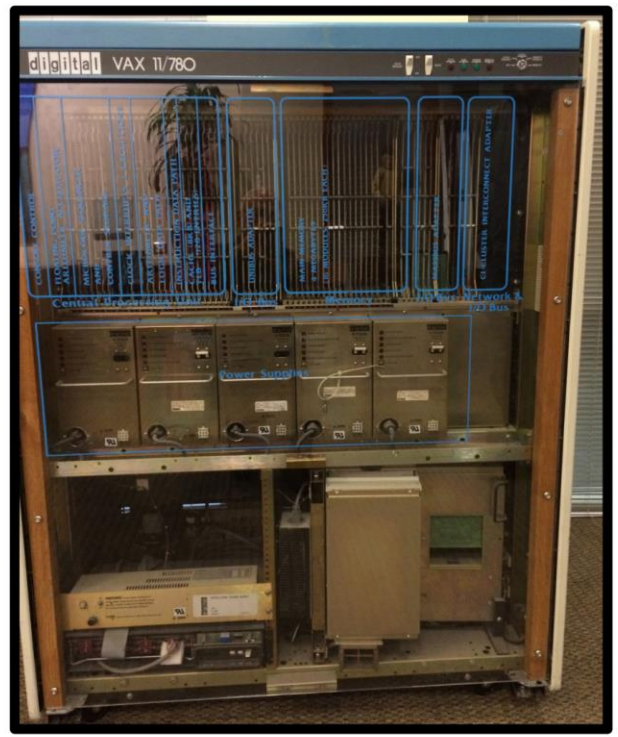
Microprogramming in IBM 360

Model	M30	M40	M50	M65
Datapath width	8 bits	16 bits	32 bits	64 bits
Microcode size	4k x 50	4k x 52	2.75k x 85	2.75k x 87
Clock cycle time (ROM)	750 ns	625 ns	500 ns	200 ns
Main memory cycle time	1500 ns	2500 ns	2000 ns	750 ns
Annual rental fee (1964 $)	$48,000	$54,000	$115,000	$270,000
Annual rental fee (2017 $)	$380,000	$430,000	$920,000	$2,160,000



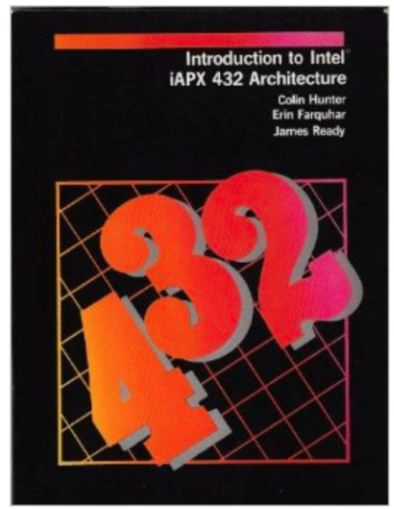
IC Technology, Microcode, and CISC

- Logic, RAM, ROM all implemented using same transistors
- Semiconductor RAM ≈ same speed as ROM
- With Moore's Law, memory for control store could grow
- Allowed more complicated instruction sets (CISC)
- Minicomputer (TTL server) example: Digital Equipment Corp. (DEC) VAX ISA in 1977
- 5K x 96b microcode



Microprocessor Evolution

- Rapid progress in 1970s, fueled by advances in MOS technology, imitated minicomputers and mainframe ISAs
- Intel i432
  - Most ambitious 1970s micro, started in 1975
  - 32-bit capability-based object-oriented architecture
  - Instructions variable number of bits long
  - Heavily microcoded
  - Custom OS written in Ada



"*The vacuum tube, the transistor, the microprocessor—at least once in a generation an electronic device arises to shock and strain designers' understanding. The latest such device is the iAPX 432 micromainframe processor, a processor as different from the current crop of microprocessors (and indeed, mainframes) as those devices are from the early electromechanical analog computers of the 1940's.*"

- Announced 1981
- Severe performance, complexity (multiple chips), and usability problems

From CISC to RISC

- Use fast RAM to build fast instruction *cache* of user-visible instructions, not fixed hardware microroutines
  - Contents of fast instruction memory change to fit what application needs right now
- Use simple ISA to enable hardwired pipelined implementation
  - Simpler encoding allowed pipelined implementations
  - No need for microcoded ISA interpreter; instructions as simple as microinstructions, but not as wide
  - Compiled code only used a few CISC instructions
- Further benefit with integration
  - In early ‘80s, could finally fit 32-bit datapath + small caches on a single chip
  - No chip crossings in common case allows faster operation

"Iron Law" of Processor Performance: How RISC can win

$$\frac{\text{Time}}{\text{Program}} = \frac{\text{Instructions}}{\text{Program}} * \frac{\text{Clock cycles}}{\text{Instruction}} * \frac{\text{Time}}{\text{Clock cycle}}$$

- CISC executes fewer instructions per program
(~1/2X instructions),
but many more clock cycles per instruction
(~ 6X CPI)
⇒ RISC 3X faster than CISC

Time Program Instructions Clock cycles

Dileep Bhandarkar and Douglas Clark, Performance from architecture: comparing a RISC and a CISC with similar hardware organization. *In Proc. Symposium, ASPLOS*, pp. 310-319, 1991.

CISC vs. RISC Today

PC Era

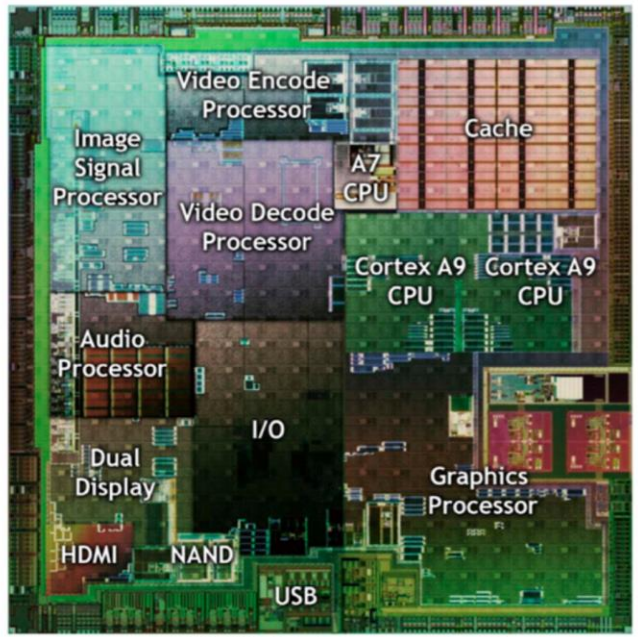
- Hardware translates x86 instructions into internal RISC instructions
- Then use any RISC technique inside MPU
- > 350M / year !
- x86 ISA eventually dominates servers as well as desktops

PostPC Era: Client/Cloud

- IP in SoC vs. MPU
- Value die area, energy as much as performance
- > 20B total / year in 2017
  - x86 in PCs peaks in 2011, declining ~8% per year since (fewer in 2016 than in 2007)
  - x86 servers => Cloud, which have only ~10M servers total (0.05% of 20B)
- 99% Processors today are RISC

Why so many ISAs on an SoC?

- Applications processor (usually ARM)
- Graphics processors
- Image processors
- Radio DSPs
- Audio DSPs
- Security processors
- Power-management processor
- ….
- Apps processor ISA too large for base accelerator ISA
- IP bought from different places, each proprietary ISA
- Home-grown ISA cores
- *Over a dozen ISAs on some SoCs – each with unique software stack*



NVIDIA Tegra SoC

Do we need all these different ISAs?

Must they be proprietary?

What if there was one free and open ISA everyone could use for everything?

RISC-V Origin Story

- UC Berkeley Research using x86 & ARM?
  - impossible – IP issues, too complex
- So we started “3-month project” in summer 2010 to develop our own clean-slate ISA
  - Principal designers: Andrew Waterman, Yunsup Lee, Dave Patterson, Krste Asanovic
- Four years later, we released frozen base user spec

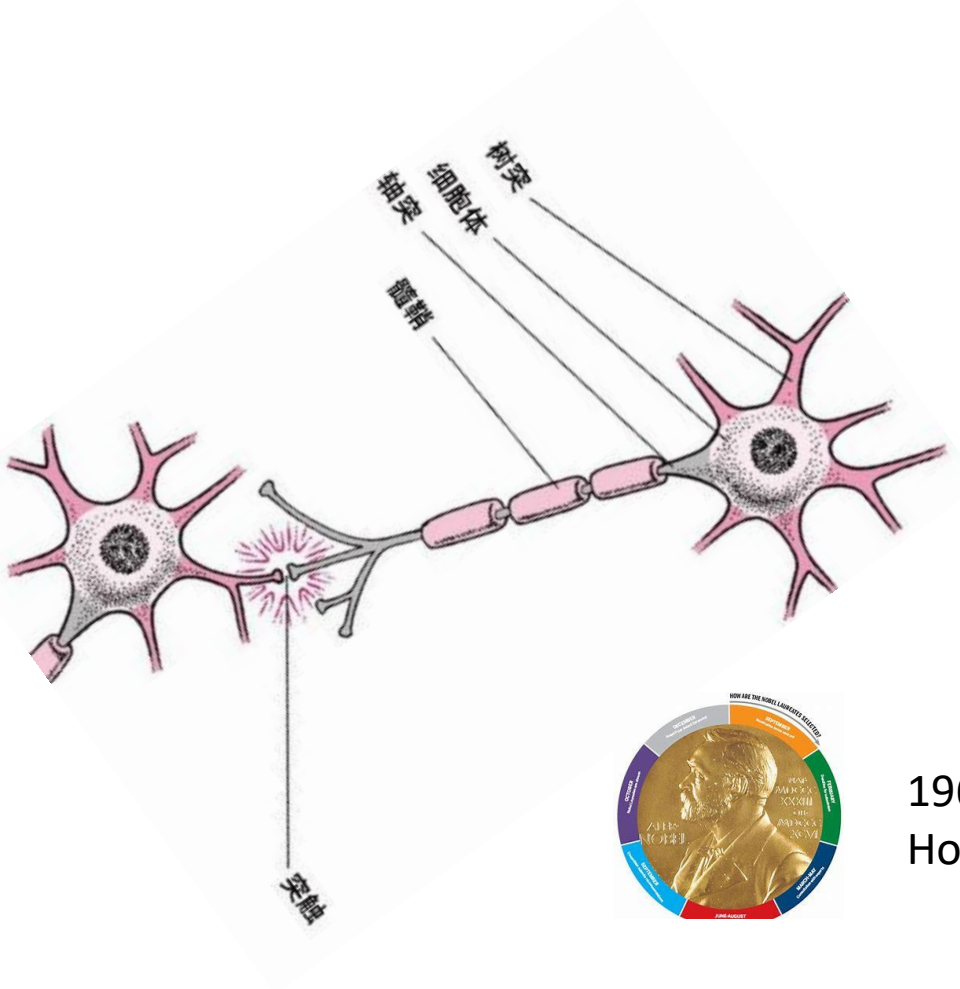
Why are outsiders complaining about changes to RISC-V in Berkeley classes?

What's Different About RISC-V?

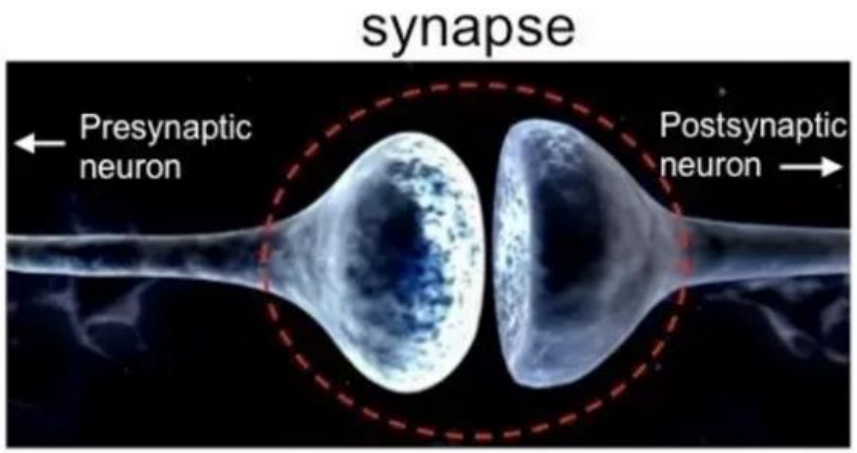
- *Simple*
  - Far smaller than other commercial ISAs
  - 2500 pages/2M words for x86 manual vs 200/75k for RV
- *Clean-slate design (25 years later so learn from mistakes)*
  - Avoids μarchitecture or technology-dependent features
- A *modular* ISA
  - Small standard base ISA
  - Multiple standard extensions
- Designed for *extensibility/specialization*
  - Variable-length instruction encoding for small programs
  - Vast opcode space available for instruction extensions
- *Stable*
  - Base and standard extensions are frozen
  - Additions via optional extensions, not new versions
  - Via open RISC-V Foundation vs internally by corporation

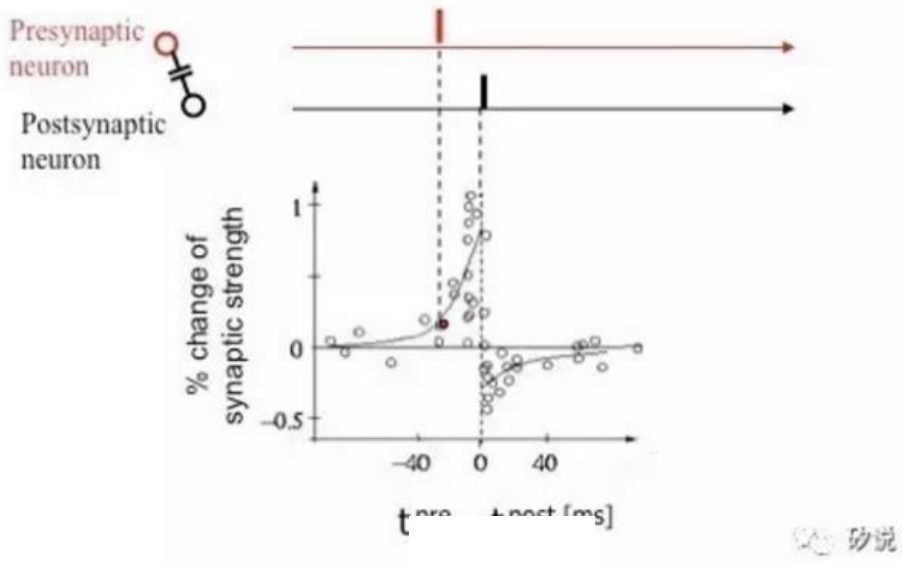
Why AI need Specific Architecture ?

神经元 及其 可塑性

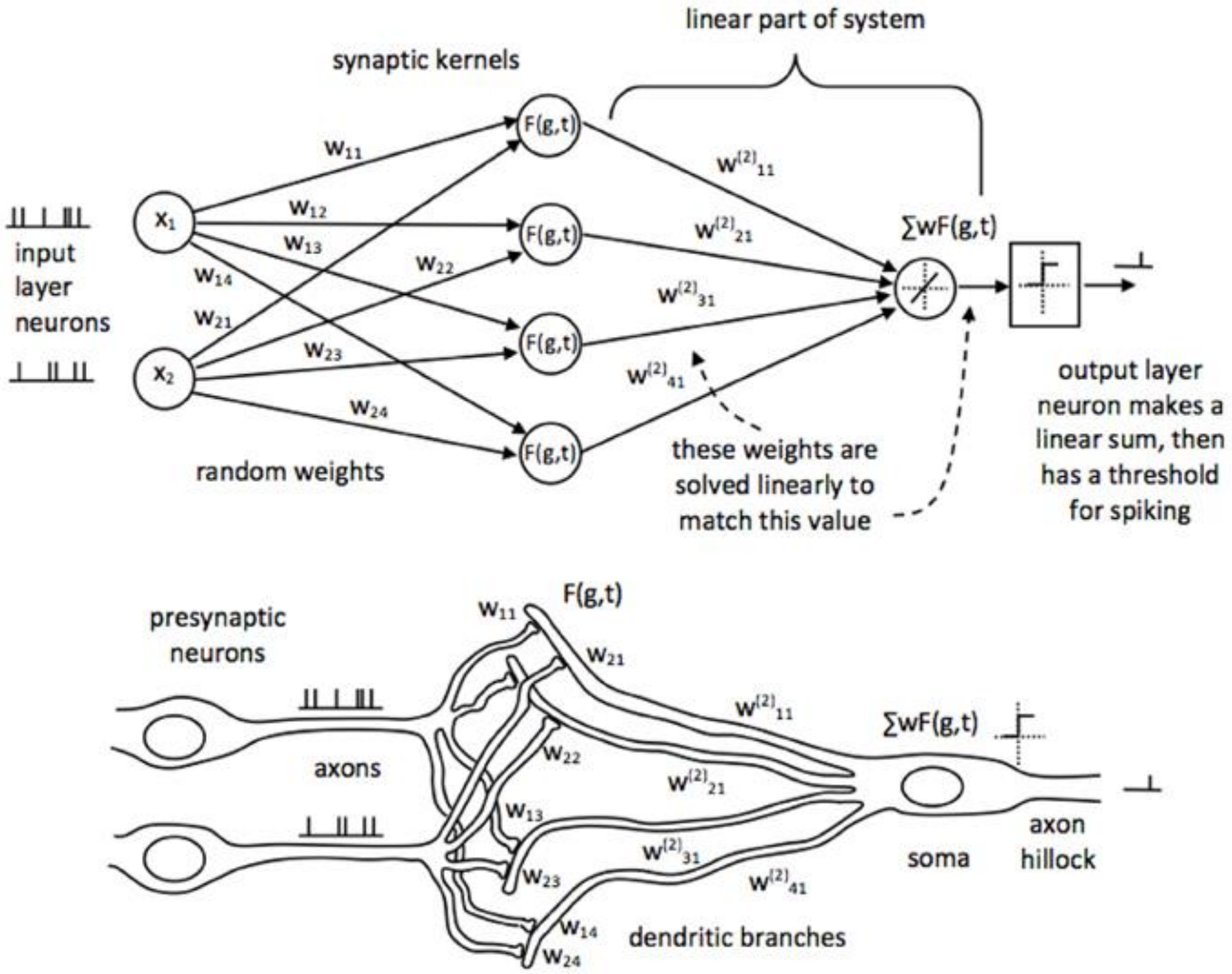


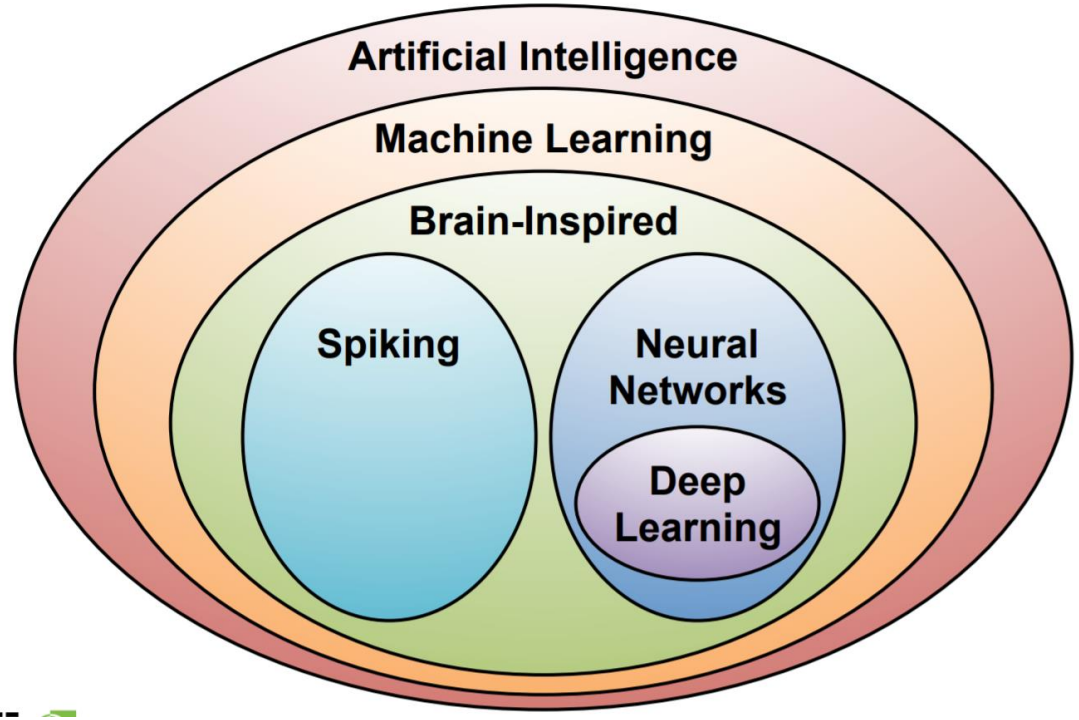
1963 年 诺贝尔奖
Hodgkin-Huxley 模型



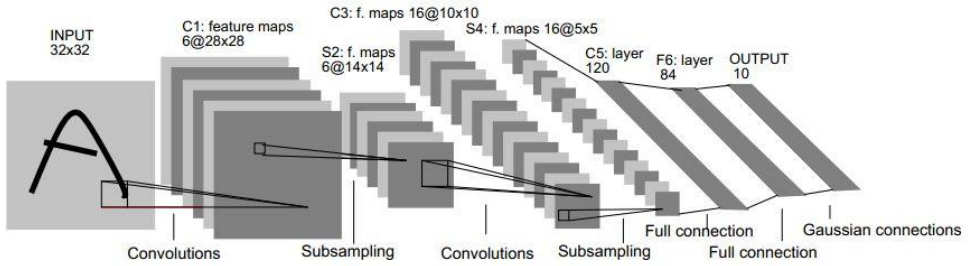


神经元的数学模型 → 神经网络

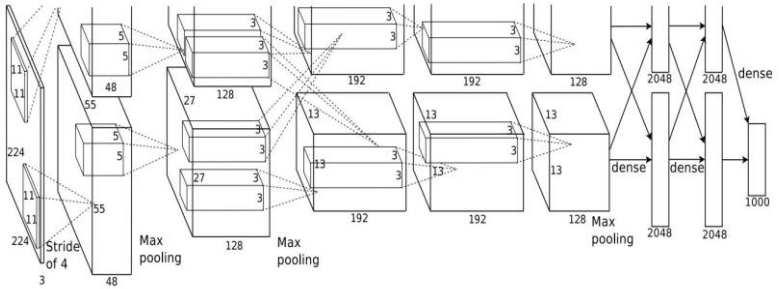




神经网络 在 人工智能 算法 领域的成功



LeNet-5: 手写数字识别



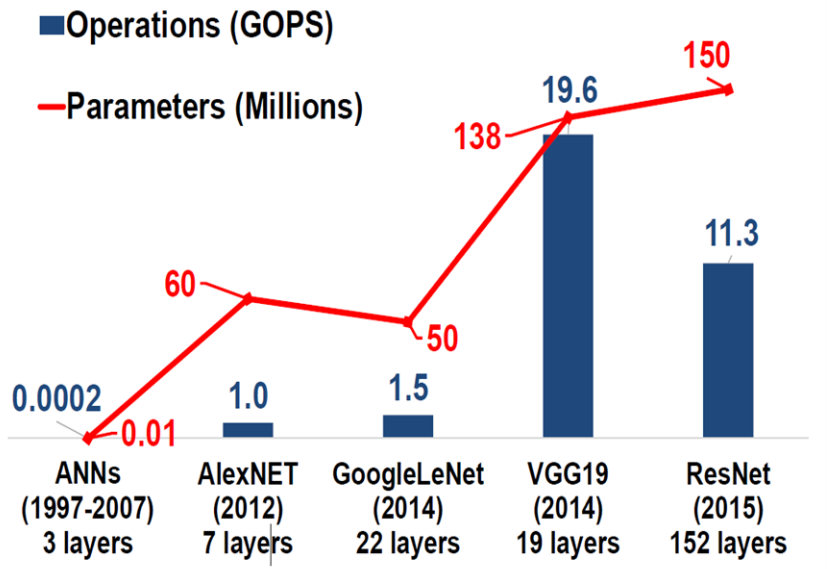
AlexNet: 图像识别首次接近人

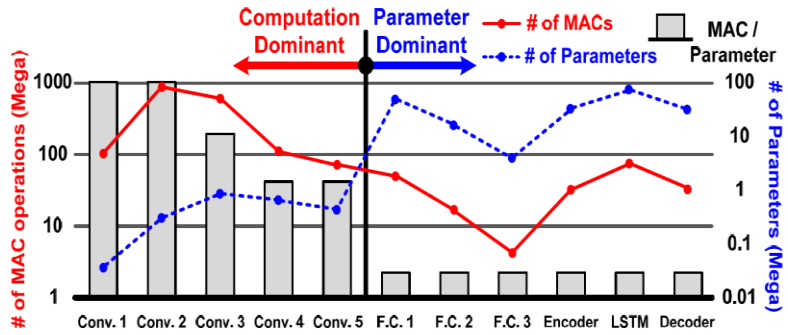
AlphaGo: 围棋上帝

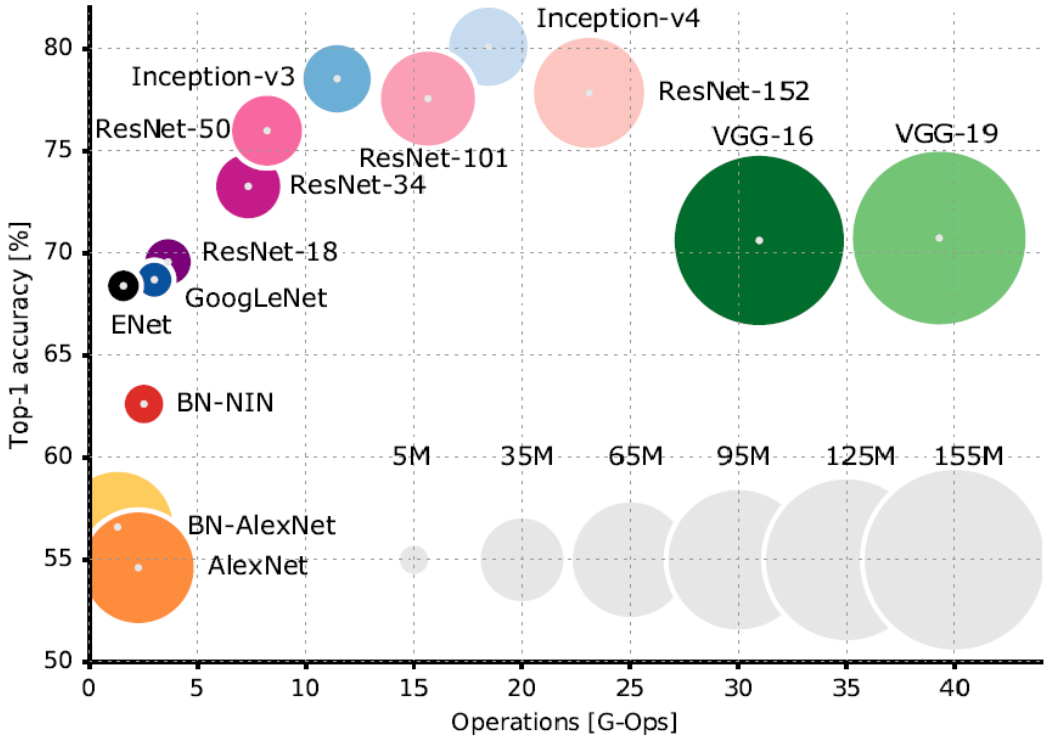
Waymo: Google 车

Echo:
语音助手

深度学习神经网络的演进







算法演进方向由低性能（左下）、多运算（右上）和多参数（大圈）向减小运算和参数同时提高性能（左上、大圈）方向进化，缓和硬件需求。

神经网络在冯诺依曼计算模型

一个神经元操作的计算过程	汇编
累加器清零	MOV
-- 循环开始	BRANCH
从存储器中加载权重	LOAD
从存储器/外设中加载输入	LOAD
权重 乘以 输入	MULT
累加	ADD
-- 判断是否重新循环	GOTO
激活函数	BRANCH
输出 存储	STORE

假设一个神经元有10个突触，每个突触连接需要一个字节（8个比特）存储链接强度，人脑有850亿个神经元，需要多大的存储空间才能放下更个大脑的神经网络？

6.8Tb

冯诺伊曼瓶颈——是访问大规模存储空间占据总线带宽和时间

神经网络在冯诺依曼计算模型

一个神经元操作的计算过程	汇编
累加器清零	MOV
-- 循环开始	BRANCH
从存储器中加载权重	LOAD
从存储器/外设中加载输入	LOAD
权重 乘以 输入	MULT
累加	ADD
-- 判断是否重新循环	GOTO
激活函数	BRANCH
输出 存储	STORE

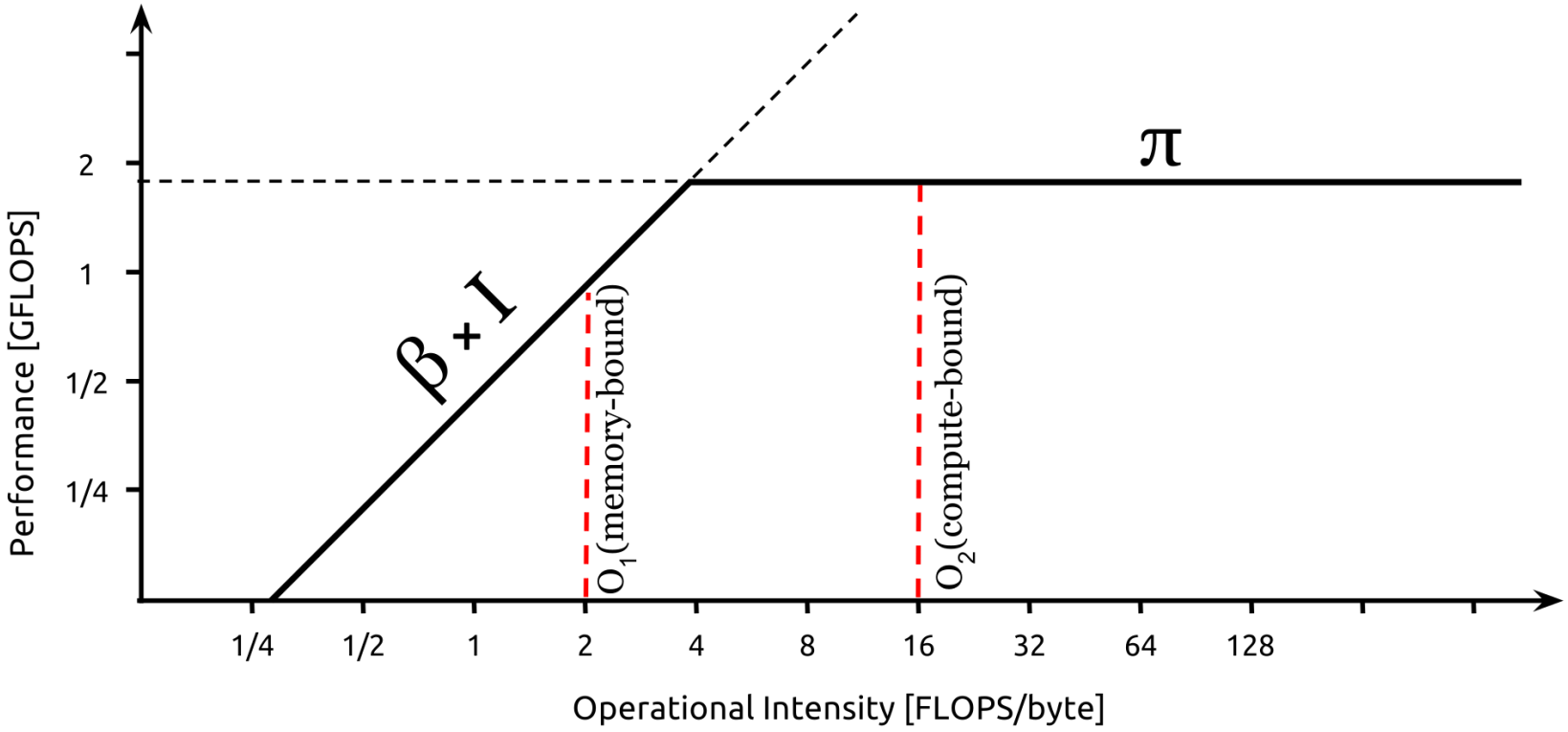
假设一个神经元有10个突触，即64个周期完成一个神经元操作，人脑有850亿个神经元，一个1GHz的冯诺依曼架构处理器需要多少时间来完整模拟一次全人脑？

86分钟

冯诺伊曼是基于逻辑抽象后的计算模型

An Computer Architecture View

The naïve Roofline is obtained by applying simple bound and bottleneck analysis. In this formulation of the Roofline model, there are only two parameters, the peak performance and the peak bandwidth of the specific architecture, and one variable, the arithmetic intensity.



Fundamental Changes in IT Field

- Technology
  - End of Dennard scaling: power becomes the key constraint
  - Ending of Moore's Law: transistor improvement slows
- Architectural
  - Limitation and inefficiencies in exploiting instruction level parallelism end the uniprocessor era in 2004
  - Amdahl's Law and its implications end "easy" multicore era
- Products
  - PC/Server ⇒ Client/Cloud

Source: John Hennessy, "The Future of Microprocessors," Stanford University, March 16, 2017

What's Left?

Since:

- Transistors not getting much better
- Power budget not getting much higher
- Already switched from 1 inefficient processor/chip to N efficient processors/chip

Only path left is *Domain Specific Architectures*

- Just do a few tasks, but extremely well

TPU: High-level Chip Architecture

- The Matrix Unit: 65,536 (256x256) 8-bit multiply-accumulate units
- 700 MHz clock rate
- Peak: 92T operations/second
  - 65,536 * 2 * 700M
- >25X as many MACs vs GPU
- >100X as many MACs vs CPU
- 4 MiB of on-chip Accumulator memory
- 24 MiB of on-chip Unified Buffer (activation memory)
- 3.5X as much on-chip memory vs GPU
- Two 2133MHz DDR3 DRAM channels
- 8 GiB of off-chip weight DRAM memory