The CPU

Class 28

John Von Neumann

John von Neumann

- one of the greatest mathematicians of all time
- born 1903 Budapest, Hungary
- died 1957 Washington, DC

A Few Accomplishments

- at 6 spoke classical Greek
- at 7 could divide two 8-digit numbers in his head
- at 8 had mastered basic calculus
- from age 19 to age 26 published 32 papers in mathematics — more than 4 per year
- at 22 earned his bachelor's in chemical engineering
- and simultaneously earned a PhD mathematics with minors in physics and chemistry

New Jersey

In 1930, at age 27, was appointed as one of the three founding faculty of the Institute for Advanced Study, Princeton, along with Albert Einstein and Kurt Gödel

He worked at IAS for the rest of his life

A Polymath

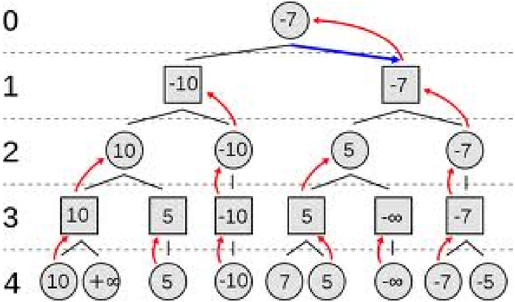
He made major contributions to the fields of

- set theory
- ergodic theory
- operator theory
- lattice theory
- quantum mechanics
- geometry
- numerical analysis
- hydrodynamics
- economics
- statistics

Game Theory

He is considered the founder of game theory

He stated and proved the minimax theorem



Nuclear Physics

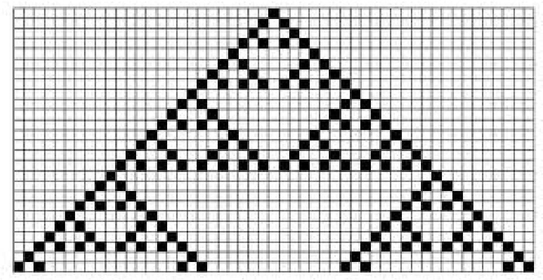
A member of the Manhattan Project which built the first atomic (fission) bombs

Central in the development of the hydrogen (fusion) bomb

Computer Scientist

Before there were working computers

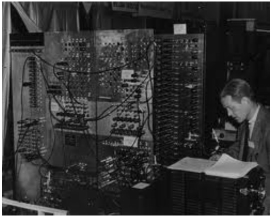
- created the field of cellular automata
- invented the mergesort algorithm



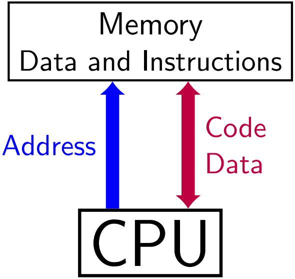
The von Neumann Architecture

Worked at U Penn with Mauchley and Eckert on the EDVAC
Wrote a paper in 1945 describing a computer architecture in which both data and program are stored together in memory in the same address space

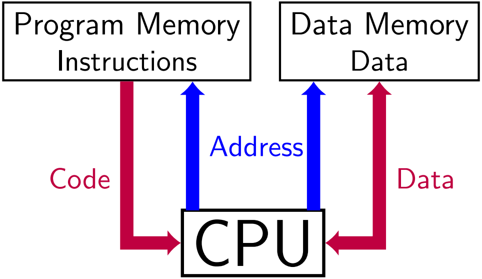
This became known as the von Neumann architecture and every computer you'll ever use is designed partially or completely on this architecture



Architecture



Von Neumann



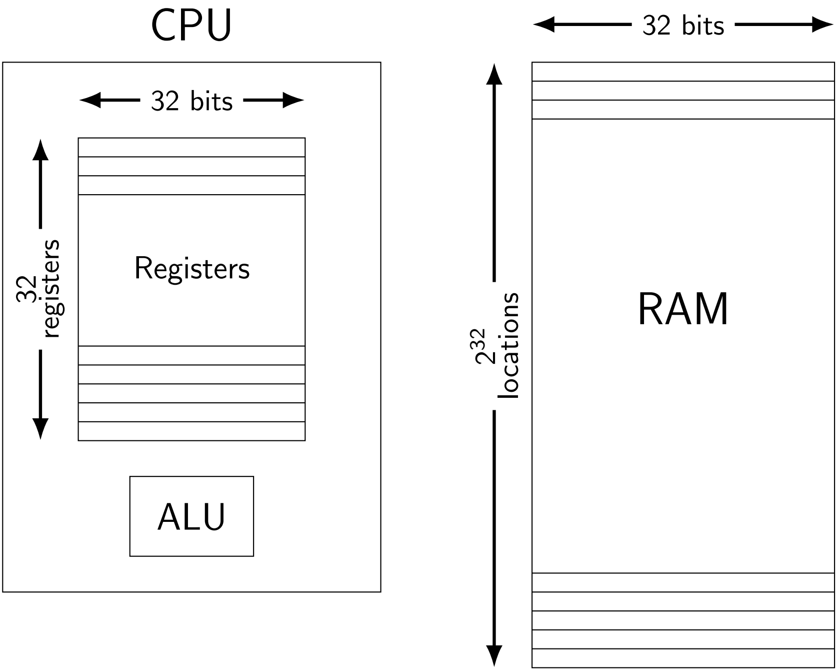
Harvard

MIPS

- MIPS is a CPU architecture
- originally created in 1985 for Unix workstations, has gone through much evolution
- biggest heyday was in the late 1990s
- Nintendo 64, several PlayStation models
- used today in embedded systems in cars, routers, cable modems, robotics

- extremely influential in the development of RISC CPU concepts

- big enough to be a real system, simple enough for undergrads to understand
- used for the rest of this semester and all of CS330



CPU

32 bits

32 registers

Registers

ALU

32 bits

2^{32} locations

RAM

Registers

- every CPU has similar types of things
- they differ in how many, how they are referred to, etc
- we will learn enough about MIPS for this class
- much more in CS330

- a fundamental characteristic of a CPU is its set of registers
- MIPS has 32 registers
- registers get used in almost every instruction, so we must know about them

Register

- a register is the most fundamental data storage structure in a processor
- a register is a place in a CPU where a value is stored
- in MIPS, all registers are exactly 32 bits in size
- registers have symbolic names
- in MIPS, all register names start with a $

MIPS Registers

Number	Name	Purpose
0	$zero	hard-wired to all zeroes
1	$at	used for pseudo-instructions
2 – 3	$v0 – $v1	return values from functions
4 – 7	$a0 – $a3	function arguments, not preserved
8 – 15	$t0 – $t7	temporary (scratch) data, not preserved
16 – 23	$s0 – $s7	saved data, preserved
24 – 25	$t8 – $t9	more temp data, not preserved
26 – 27	$k0 – $k1	reserved for kernel; we don't use these
28	$gp	global pointer (start of global data)
29	$sp	stack pointer
30	$fp	frame pointer
31	$ra	return address

Instruction Set Architecture

- ISA
- an abstract model of a computer
- the central topic of CS330 — we'll just touch on a few parts
- a physical CPU implements an ISA

- an ISA defines
  - data types
  - registers
  - addressing modes
  - the input-output model
  - the actual instructions
- examples
  - x86 Intel processors
  - Arduino microcontrollers
  - MIPS

ISA Types

- based on instruction access to memory
- there are three main types of ISA
  - memory-memory architecture: an instruction can perform an operation with both operands in memory and put the result back into memory
  - register-memory architecture: an instruction can perform an operation with one operand in memory and the other in a register, and put the result in a register
  - load-store architecture: an instruction can perform an operation with both operands in registers, and put the result in a register

Load-Store

- MIPS is a load-store architecture
- there are three categories of instruction
    - memory access: load or store; load a value from memory into a register, or store a value from a register to memory
    - computations: operate on value(s) in registers
    - miscellaneous