# Basic Computer Architecture

# Glitch blamed for Gaston voting woes

**A software problem delays the declaration of a winner in two statewide judicial races and four legislative races.**

By Paul Nowell
The Associated Press

GASTONIA — A "software glitch" created the problems in retrieving vote counts from Gaston County's new, computerized voting machines on Election Day, the county elections board chairman said Thursday.

The glitch has delayed a winner being declared in two statewide judicial races and four legislative races.

Once the problem is fixed, several statewide races that were in jeopardy could be decided by as early as today, said Steve Gheen.

"A glitch in the [software] program stopped the count after 32,000 votes [in each voting machine]," he said after meeting with state elections officials and technicians with the company that makes the voting machines. "It's being corrected now.

# Glitch blamed for Gaston voting woes

**A software problem delays the declaration of a winner in two statewide judicial races and four legislative races.**

BY PAUL NOWELL
THE ASSOCIATED PRESS

GASTONIA — A "software glitch" created the problems in retrieving vote counts from Gaston County's new, computerized voting machines on Election Day, the county elections board chairman said Thursday.

The glitch has delayed a winner being declared in two statewide judicial races and four legislative races.

Once the problem is fixed, several statewide races that were in jeopardy could be decided by as early as today, said Steve Gheen.

"A glitch in the [software] program stopped the count after 32,000 votes [in each voting machine]," he said after meeting with state elections officials and technicians with the company that makes the voting machines. "It's being corrected now.

"A glitch in the software stopped the count after 32000 votes"

# Glitch blamed for Gaston voting woes

**A software problem delays the declaration of a winner in two statewide judicial races and four legislative races.**

By Paul Nowell
The Associated Press

GASTONIA — A "software glitch" created the problems in retrieving vote counts from Gaston County's new, computerized voting machines on Election Day, the county elections board chairman said Thursday.

The glitch has delayed a winner being declared in two statewide judicial races and four legislative races.

Once the problem is fixed, several statewide races that were in jeopardy could be decided by as early as today, said Steve Gheen.

"A glitch in the [software] program stopped the count after 32,000 votes [in each voting machine]," he said after meeting with state elections officials and technicians with the company that makes the voting machines. "It's being corrected now.

## Was the number really 32000?

# Glitch blamed for Gaston voting woes

**A software problem delays the declaration of a winner in two statewide judicial races and four legislative races.**

BY PAUL NOWELL
THE ASSOCIATED PRESS

GASTONIA — A "software glitch" created the problems in retrieving vote counts from Gaston County's new, computerized voting machines on Election Day, the county elections board chairman said Thursday.

The glitch has delayed a winner being declared in two statewide judicial races and four legislative races.

Once the problem is fixed, several statewide races that were in jeopardy could be decided by as early as today, said Steve Gheen.

"A glitch in the [software] program stopped the count after 32,000 votes [in each voting machine]," he said after meeting with state elections officials and technicians with the company that makes the voting machines. "It's being corrected now.

Was the number really 32000?

Or was it 32767?

# Glitch blamed for Gaston voting woes

**A software problem delays the declaration of a winner in two statewide judicial races and four legislative races.**

By Paul Nowell
The Associated Press

GASTONIA — A "software glitch" created the problems in retrieving vote counts from Gaston County's new, computerized voting machines on Election Day, the county elections board chairman said Thursday.

The glitch has delayed a winner being declared in two statewide judicial races and four legislative races.

Once the problem is fixed, several statewide races that were in jeopardy could be decided by as early as today, said Steve Gheen.

"A glitch in the [software] program stopped the count after 32,000 votes [in each voting machine]," he said after meeting with state elections officials and technicians with the company that makes the voting machines. "It's being corrected now.

How was the variable VOTES declared?

# Glitch blamed for Gaston voting woes

**A software problem delays the declaration of a winner in two statewide judicial races and four legislative races.**

By Paul Nowell
The Associated Press

GASTONIA — A "software glitch" created the problems in retrieving vote counts from Gaston County's new, computerized voting machines on Election Day, the county elections board chairman said Thursday.

The glitch has delayed a winner being declared in two statewide judicial races and four legislative races.

Once the problem is fixed, several statewide races that were in jeopardy could be decided by as early as today, said Steve Gheen.

"A glitch in the [software] program stopped the count after 32,000 votes [in each voting machine]," he said after meeting with state elections officials and technicians with the company that makes the voting machines. "It's being corrected now.

Votes declared as short int.

# Glitch blamed for Gaston voting woes

**A software problem delays the declaration of a winner in two statewide judicial races and four legislative races.**

By Paul Nowell
The Associated Press

GASTONIA — A "software glitch" created the problems in retrieving vote counts from Gaston County's new, computerized voting machines on Election Day, the county elections board chairman said Thursday.

The glitch has delayed a winner being declared in two statewide judicial races and four legislative races.

Once the problem is fixed, several statewide races that were in jeopardy could be decided by as early as today, said Steve Gheen.

"A glitch in the [software] program stopped the count after 32,000 votes [in each voting machine]," he said after meeting with state elections officials and technicians with the company that makes the voting machines. "It's being corrected now.

Votes declared as short int.

Up to 32767 votes it worked perfectly.

# Glitch blamed for Gaston voting woes

**A software problem delays the declaration of a winner in two statewide judicial races and four legislative races.**

By Paul Nowell
The Associated Press

GASTONIA — A "software glitch" created the problems in retrieving vote counts from Gaston County's new, computerized voting machines on Election Day, the county elections board chairman said Thursday.

The glitch has delayed a winner being declared in two statewide judicial races and four legislative races.

Once the problem is fixed, several statewide races that were in jeopardy could be decided by as early as today, said Steve Gheen.

"A glitch in the [software] program stopped the count after 32,000 votes [in each voting machine]," he said after meeting with state elections officials and technicians with the company that makes the voting machines. "It's being corrected now.

Votes declared as short int.

At 32768 … the vote count started to appear negative.

# Glitch blamed for Gaston voting woes

**A software problem delays the declaration of a winner in two statewide judicial races and four legislative races.**

By Paul Nowell
The Associated Press

GASTONIA — A "software glitch" created the problems in retrieving vote counts from Gaston County's new, computerized voting machines on Election Day, the county elections board chairman said Thursday.

The glitch has delayed a winner being declared in two statewide judicial races and four legislative races.

Once the problem is fixed, several statewide races that were in jeopardy could be decided by as early as today, said Steve Gheen.

"A glitch in the [software] program stopped the count after 32,000 votes [in each voting machine]," he said after meeting with state elections officials and technicians with the company that makes the voting machines. "It's being corrected now.

## Votes are not a signed quantity

# Glitch blamed for Gaston voting woes

**A software problem delays the declaration of a winner in two statewide judicial races and four legislative races.**

By Paul Nowell
The Associated Press

GASTONIA — A "software glitch" created the problems in retrieving vote counts from Gaston County's new, computerized voting machines on Election Day, the county elections board chairman said Thursday.

The glitch has delayed a winner being declared in two statewide judicial races and four legislative races.

Once the problem is fixed, several statewide races that were in jeopardy could be decided by as early as today, said Steve Gheen.

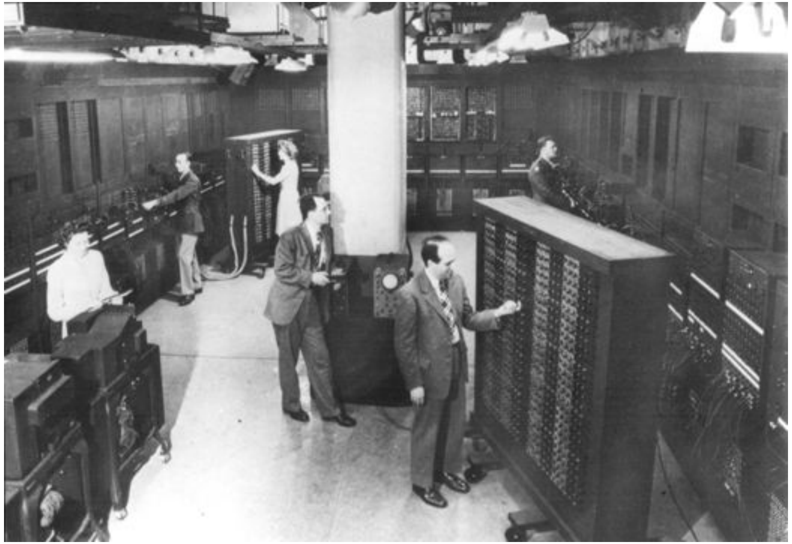
"A glitch in the [software] program stopped the count after 32,000 votes [in each voting machine]," he said after meeting with state elections officials and technicians with the company that makes the voting machines. "It's being corrected now.

C - declare votes as unsigned

Java - declare votes as an int (large enough to hold the anticipated count)

# ENIAC

## Electronic Numeric Integrator And Computer



Circa fall 1945

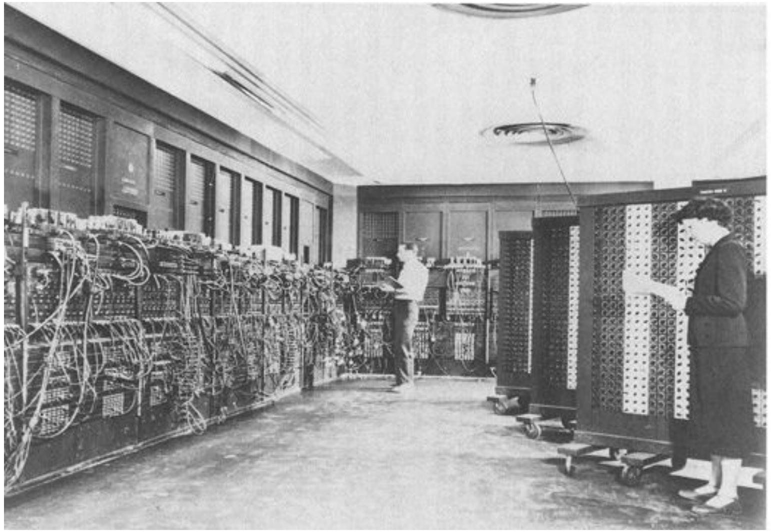
# ENIAC

## Electronic Numeric Integrator And Computer

Function was to calculate artillery shell trajectories

# ENIAC

## Electronic Numeric Integrator And Computer

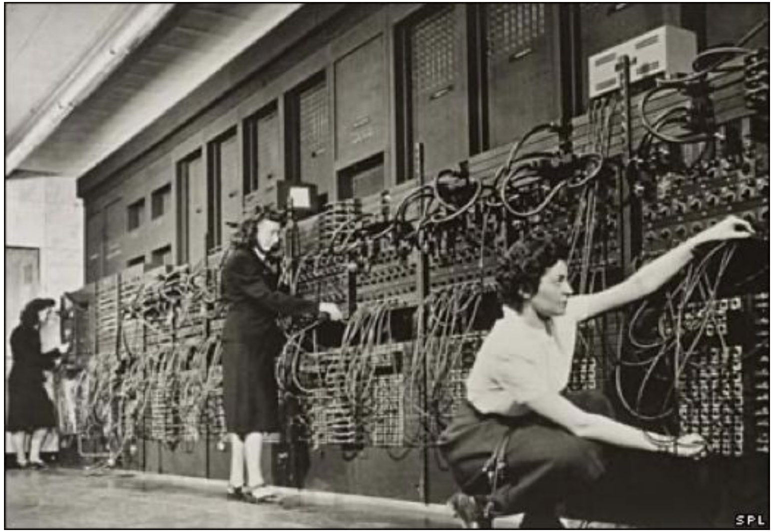


18,000 vacuum tubes

MTF ≈ 2 days

# ENIAC

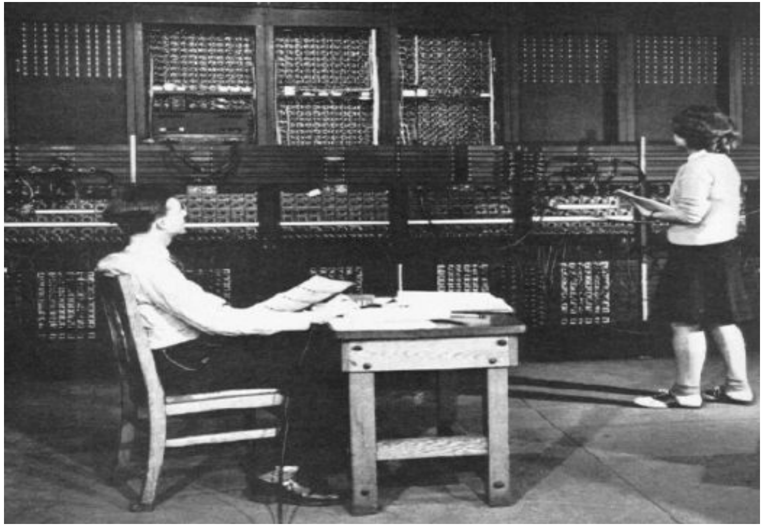
## Electronic Numeric Integrator And Computer




Programmed by wiring
a patch panel

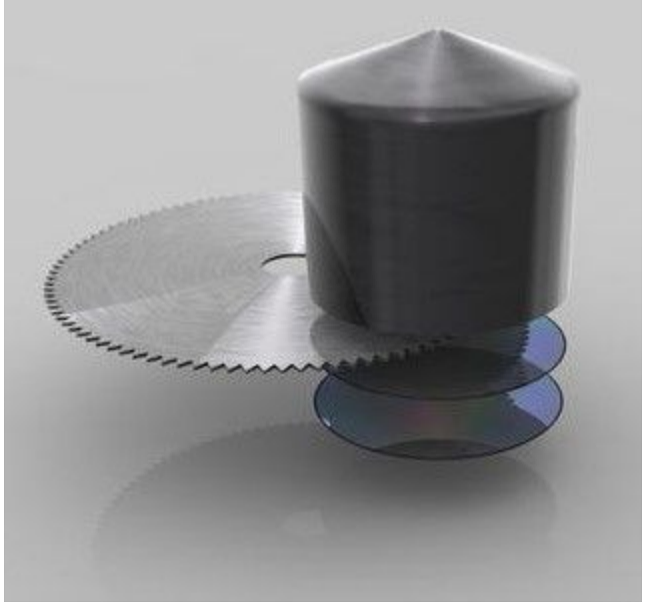
# ENIAC

## Electronic Numeric Integrator And Computer

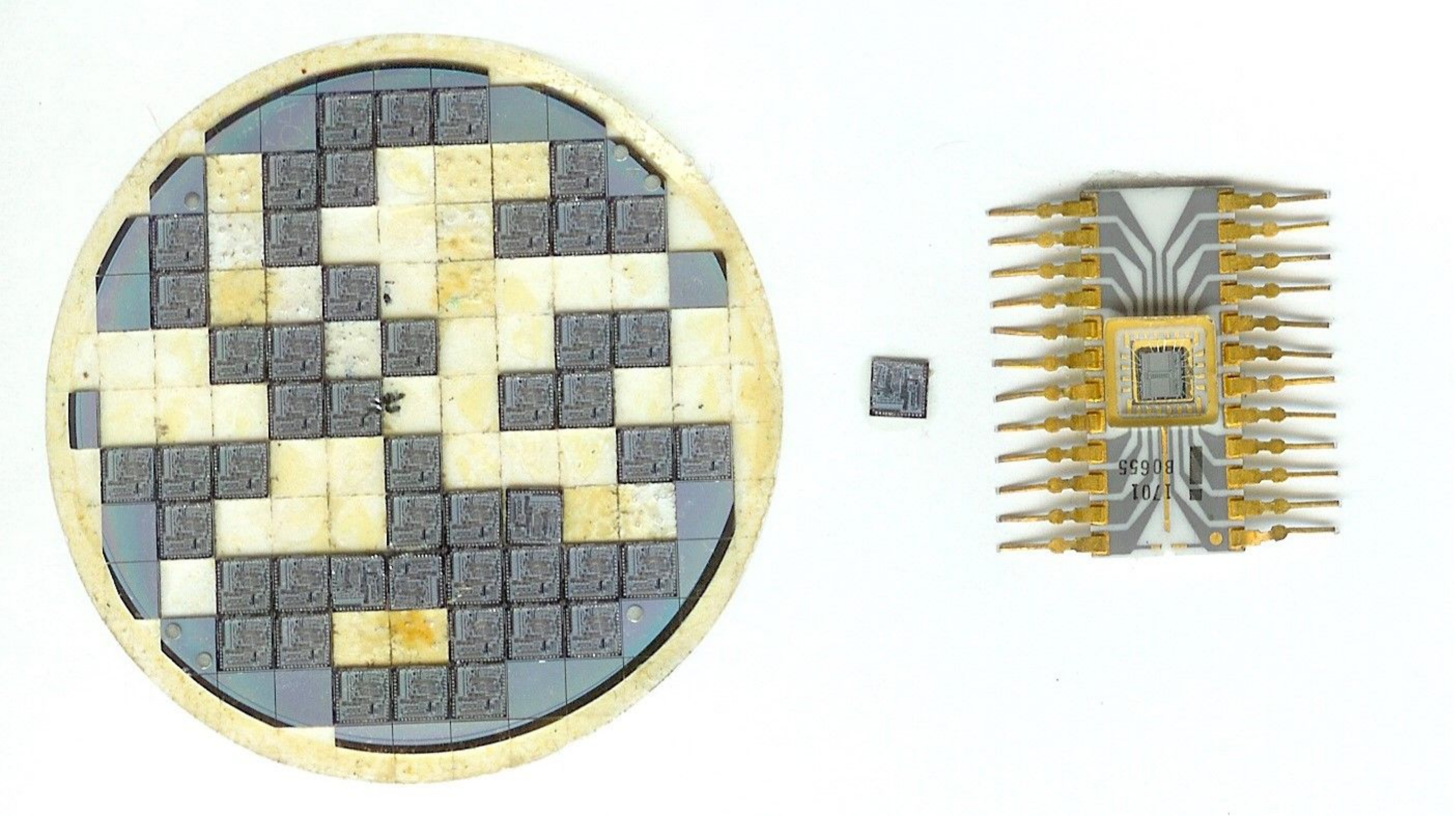


5000 additions per second

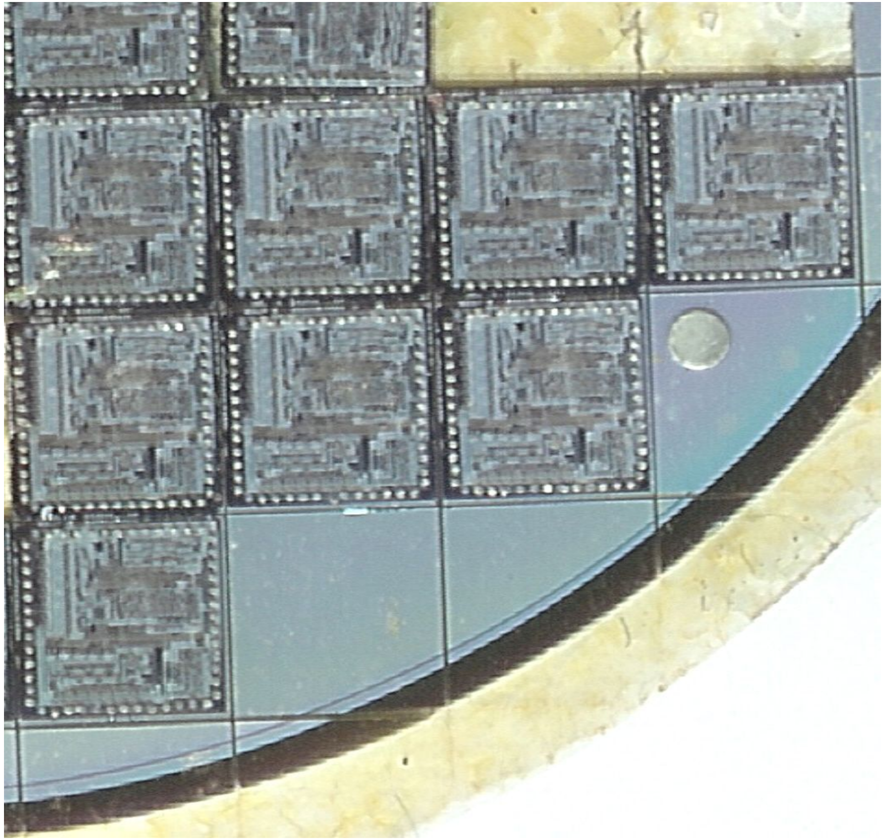
# 21st Century Computers



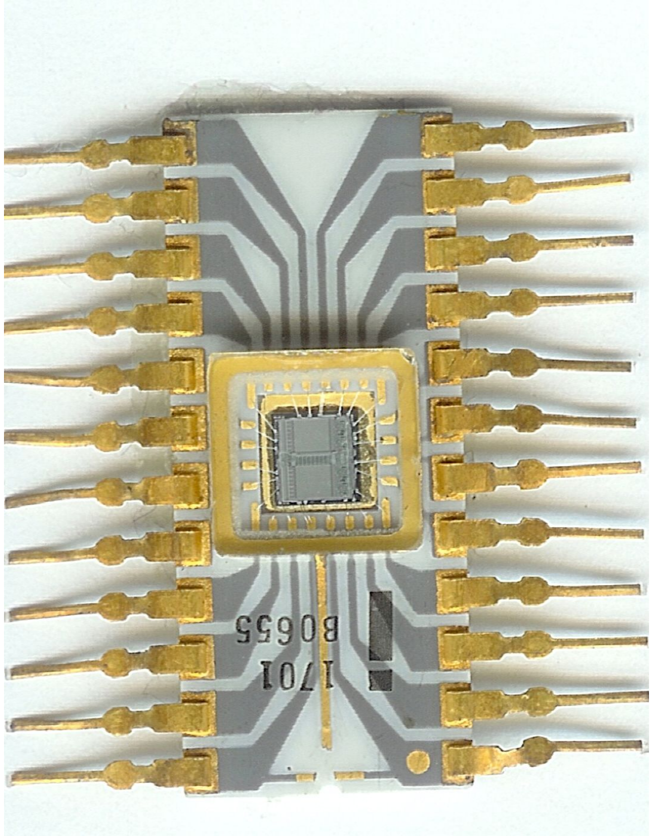
# 21st Century Computers



# 21st Century Computers

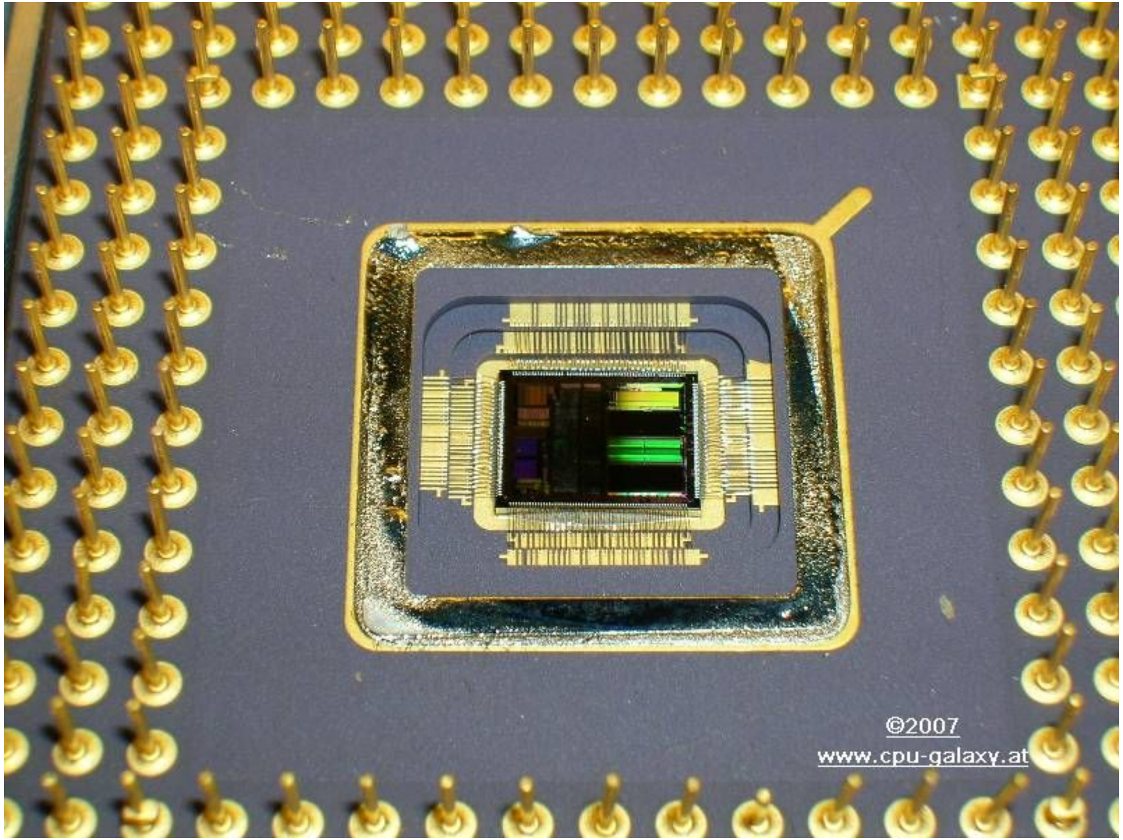


# 21st Century Computers



# 21st Century Computers

# 21st Century Computers



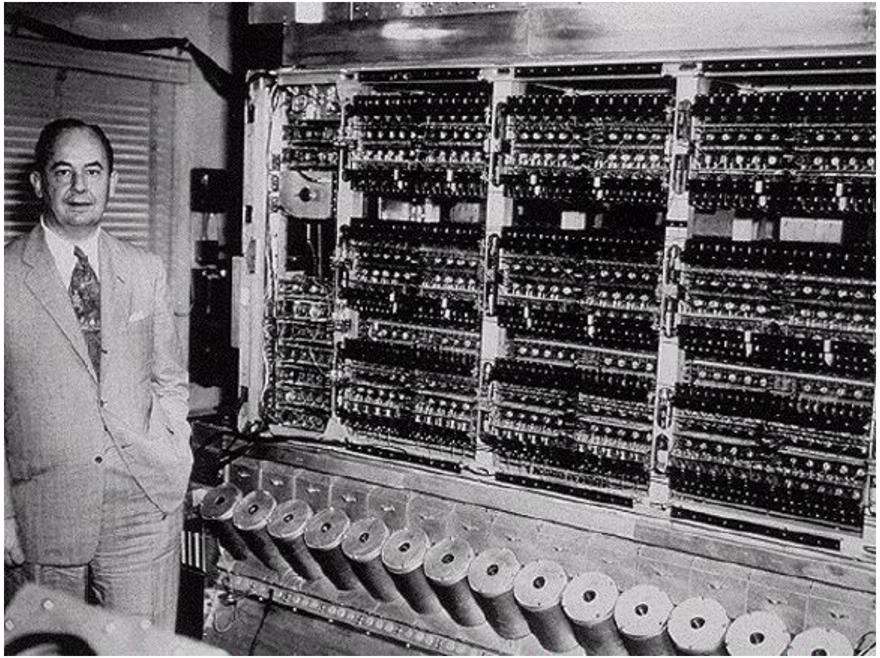
# 21st Century Computers

# 21st Century Computers

Interesting walkthrough of the process:

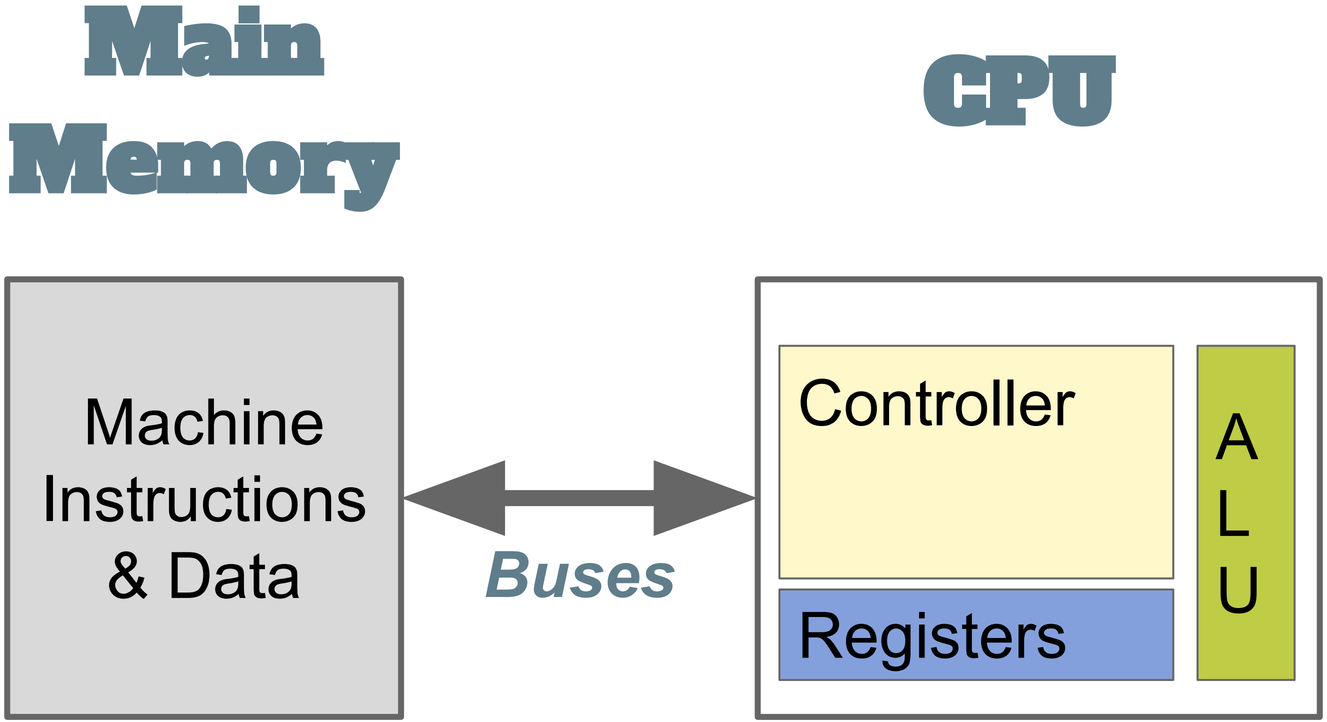
https://www.pcgameshardware.de/CPU-CPU-154106/News/How-an-Intel-CPU-is-created-From-Sand-to-Silicon-Making-of-a-Chip-689436/galerie/1156822/

# John Von Neumann Architecture



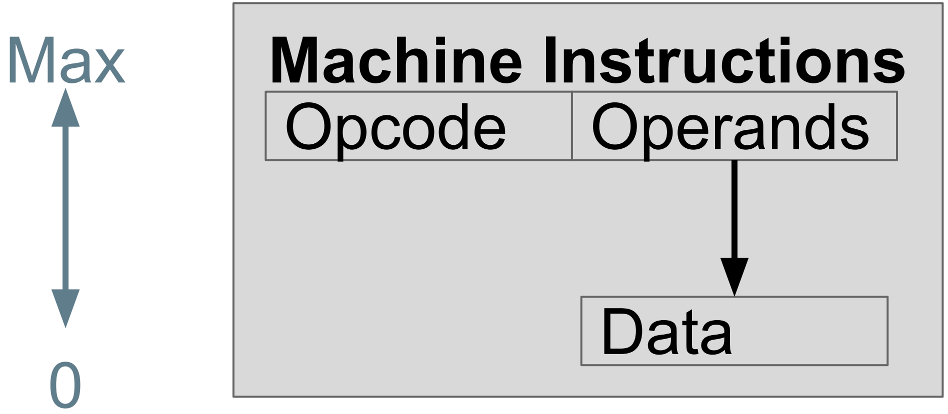
(circa 1944)

**Still the *basic* architecture for modern computers**


Main Memory
CPU
Machine Instructions & Data
Buses
Controller
Registers
A L U

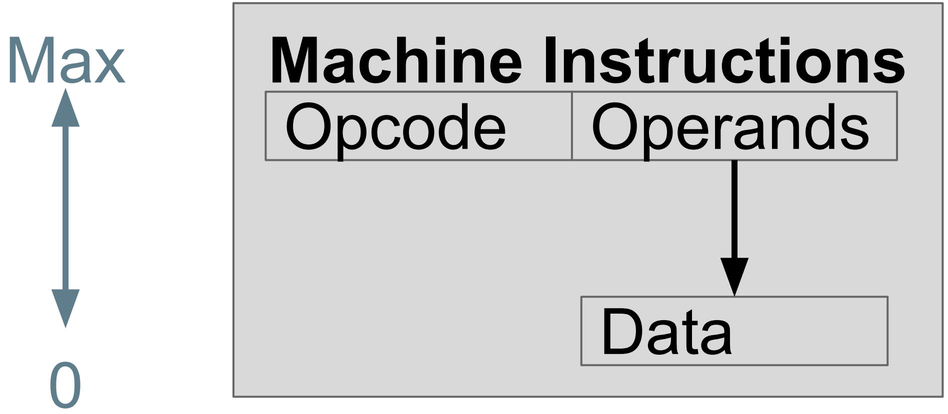
# Memory

Size is measured in bytes



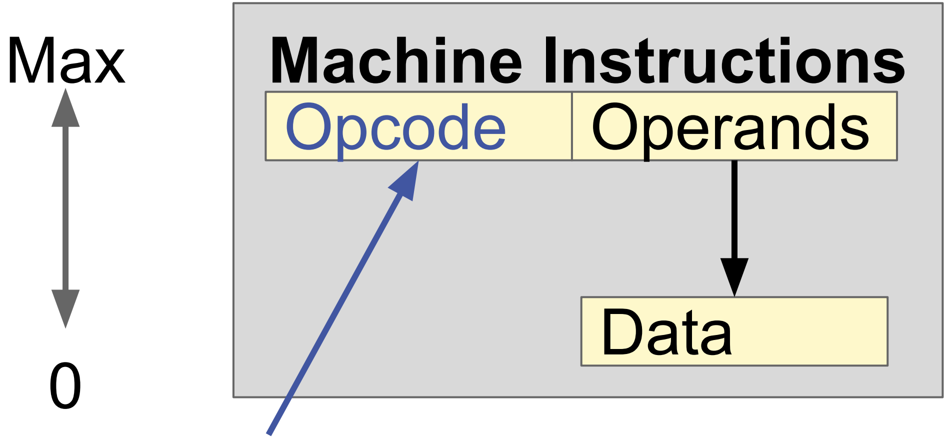

**32-64 GigaBytes typical (today)**

# Memory




**Opcode (Operation Code)**
**Defines the instruction**

# Memory




**Large enough to describe all needed instructions**

**Not so large as to be wasteful**

**Operation code size limits the number of possible different instructions**

# Memory

- Need to consider evolution of the architecture over time
- Intel’s x86 architecture has lasted 50 years
- The original x86 design
    - 6 bits for the opcode
    - allows only 64 unique machine instructions
- Resorted to unusual (creative) techniques to expand set of available opcodes.

# Data Sizes (Intel Hardware Terminology)

| | Byte | Word | Longword |
|---|---|---|---|
| Bits | 8 | 16 | 32 |
| States | 256 | 65536 | ~ 4 billion |

**Can be used for Unsigned or Two's Complement Signed**

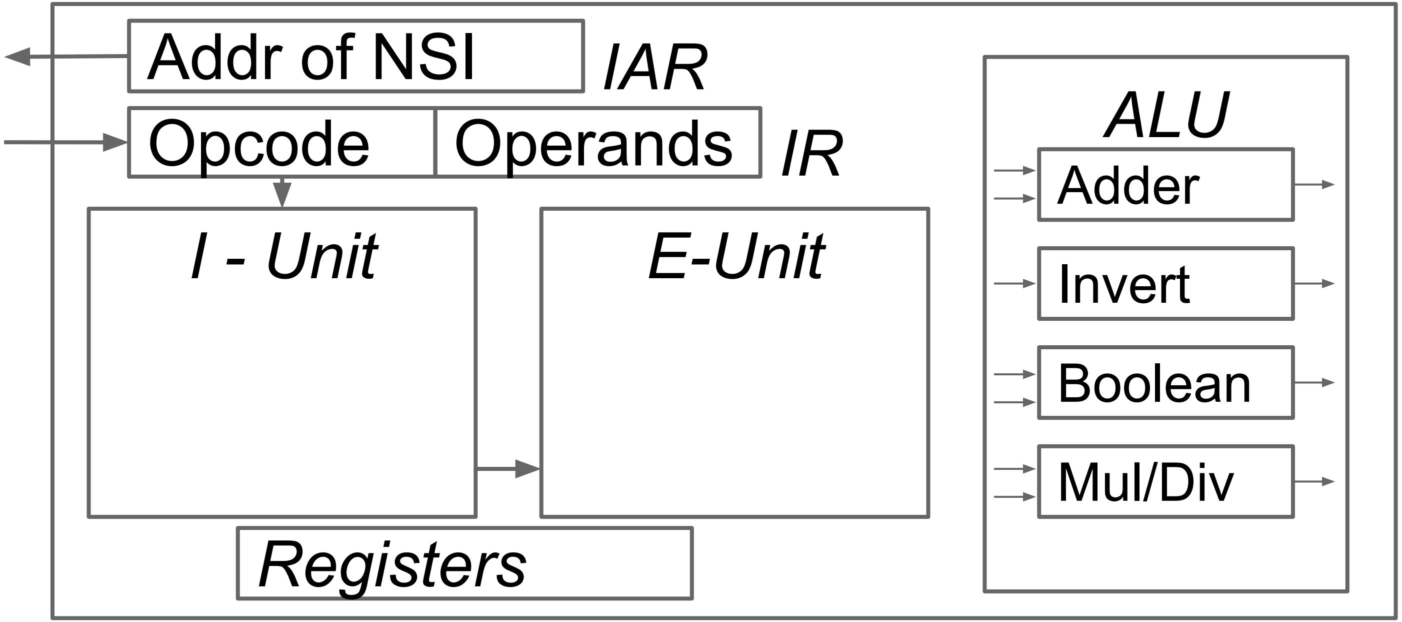
# Data Sizes (Intel Hardware Terminology)

| | Byte | Word | Longword |
|---|---|---|---|
| Bits | 8 | 16 | 32 |
| States | 256 | 65536 | ~ 4 billion |
| UPN | 0 - 255 | 0 - 65535 | 0 - 4 billion |
| Signed | - 128<br>+127 | - 32768 +32767 | - 2 billion<br>+2 billion |

# Memory Technology

- SRAM Static RAM
    - fast / expensive / transistors
    - 6 transistors per bit
    - access time of 5-10 nanosecs
    - too expensive for Main Memory: 20-100x more expensive than DRAM
- DRAM Dynamic RAM
    - slow / less expensive / capacitors
    - 1 transistor and 1 capacitor per bit
    - access time of 50-150 nanosecs
    - used for Main Memory

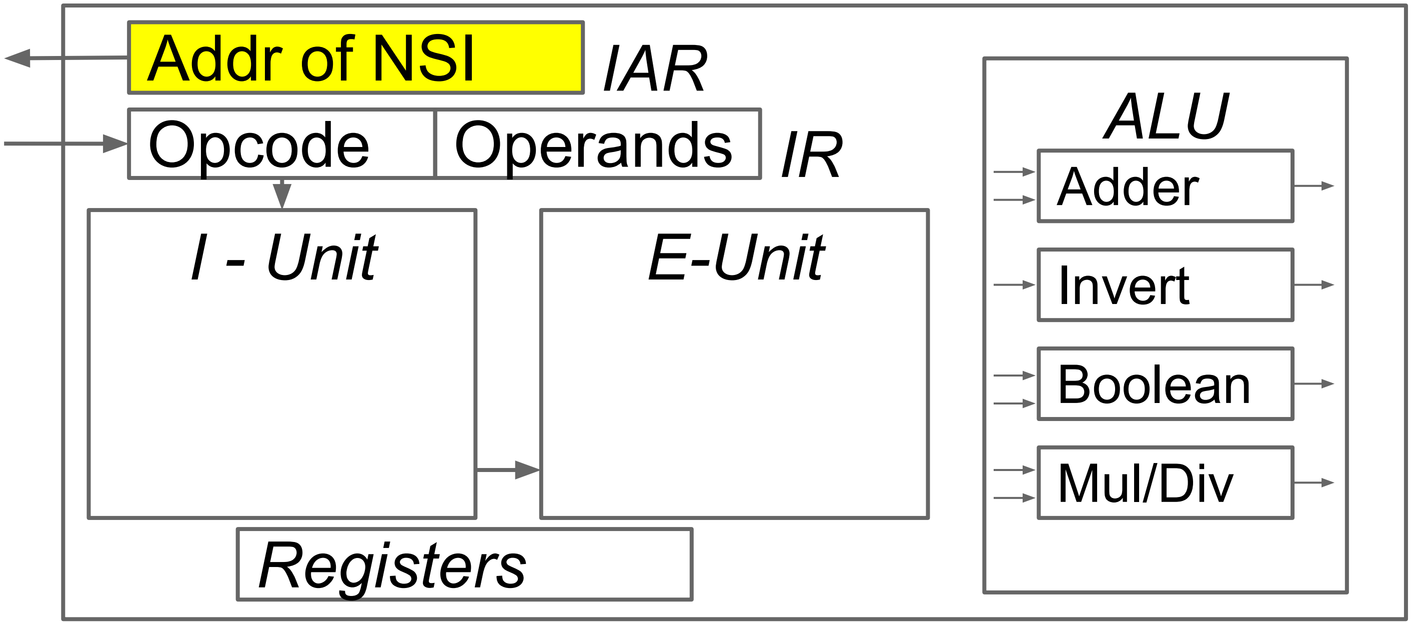
# CPU




**Registers – on-processor temporary storage**

- **Some general purpose**
- **Others special purpose**

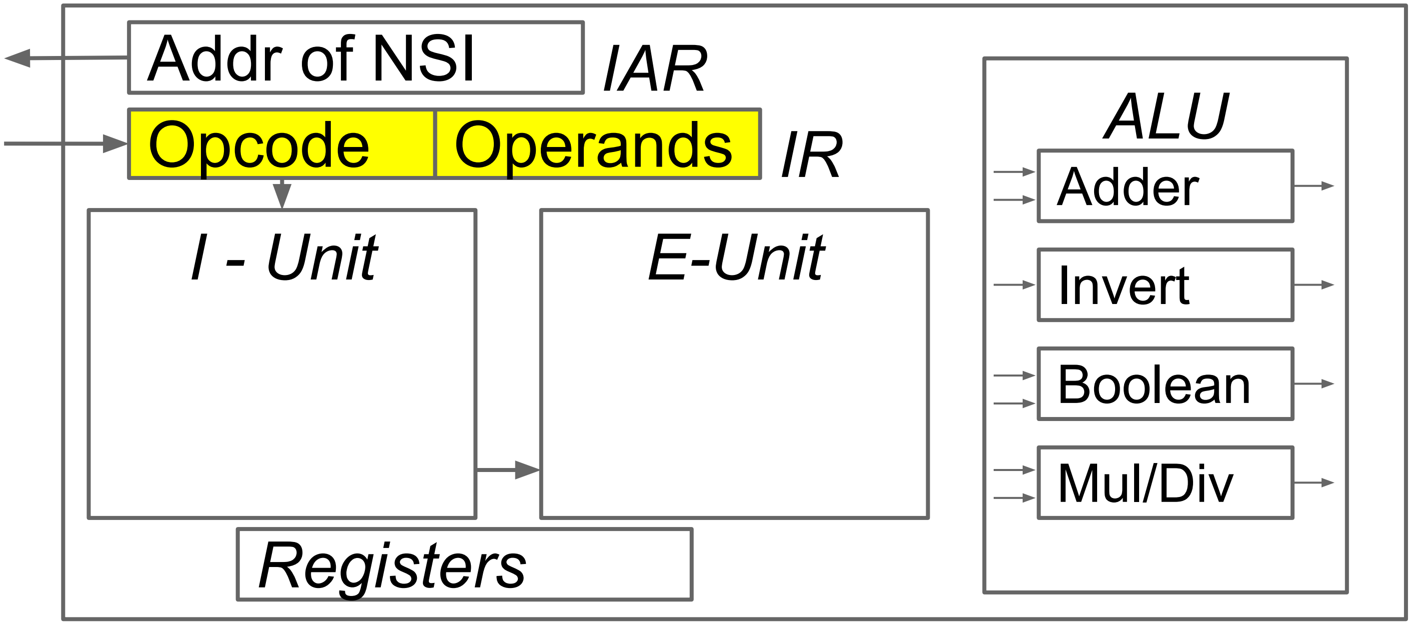
# CPU




**Instruction Address Register (IAR)**

- **Points to the Next Sequential Instruction**
- **AKA: PC (program counter) and IP (inst. pointer)**

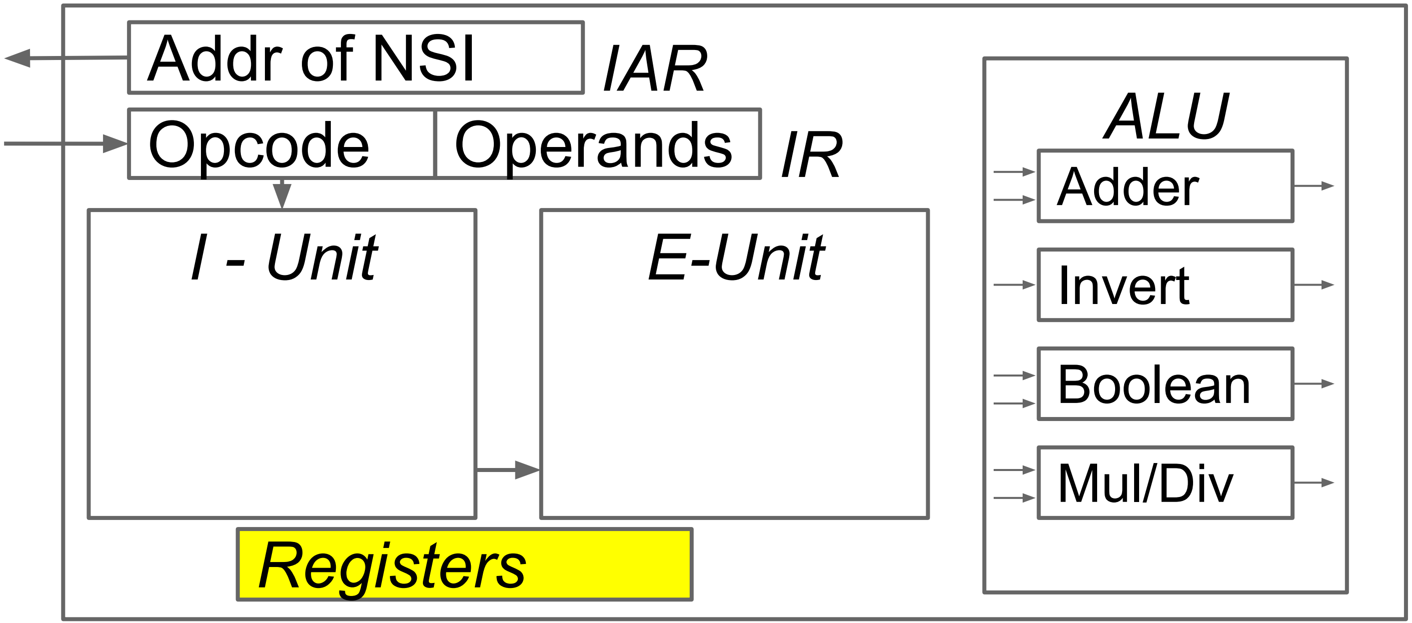
# CPU




**Instruction Register**

- **Holds the instruction being executed**

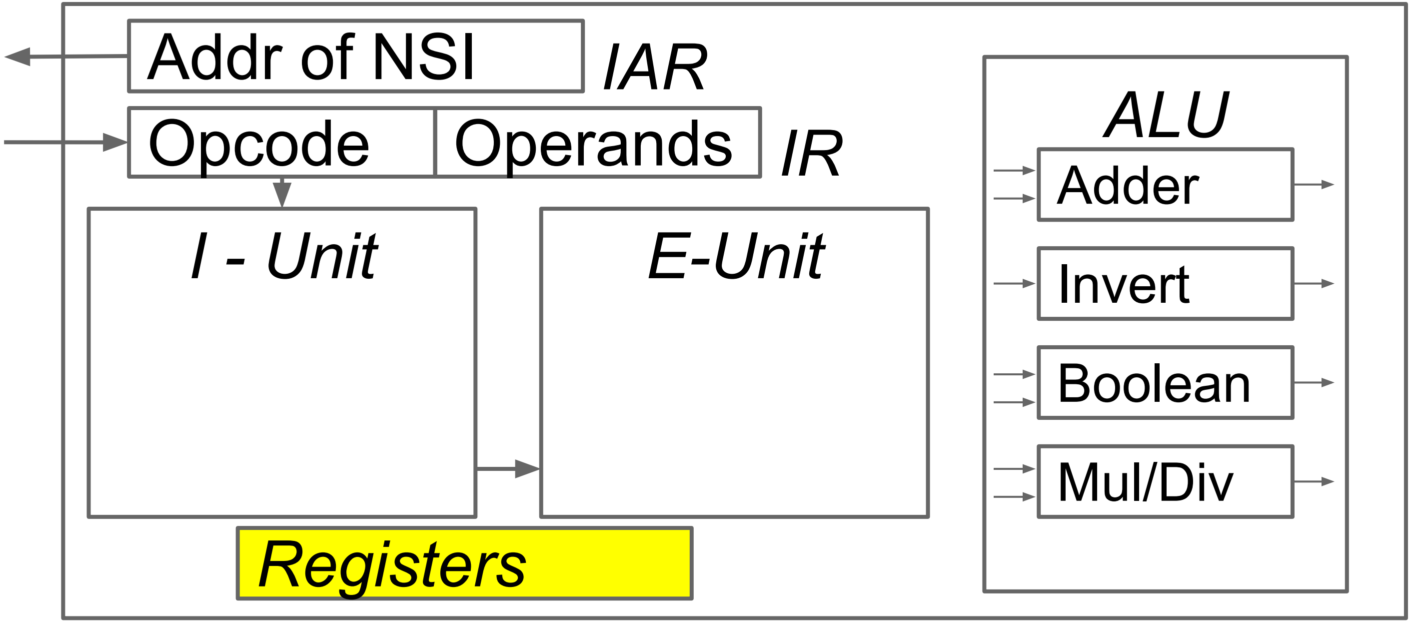
# CPU




**General purpose registers**

- **Scratch space available to the programmer**
  - **compiler writer; asm lang programmer**

# CPU




**General purpose registers**

- **Instructions tend to run faster when data is in registers**
- **Some instructions only work with data in registers**


**Why are registers, which are so important invisible to the HLL programmer?**

# Register architectures vary radically

**8086 (12 Regs)**

| | |
|---|---|
| ax | multiply |
| bx | DS pointer |
| cx | looping |
| dx | multiply |
| si | DS pointer |
| di | DS pointer |
| bp | SS pointer |
| sp | Stack pointer |
| cs | Code Segment |
| ds | Data Segment |
| ss | Stack Segment |
| es | Extra segment |

**M68000 (16 regs)**

| | |
|---|---|
| d0 | (all equivalent) |
| d1 | |
| d2 | |
| d3 | |
| d4 | |
| d5 | |
| d6 | |
| d7 | |
| a0 | (all equivalent) |
| a1 | |
| a2 | |
| a3 | |
| a4 | |
| a5 | |
| a6 | |
| a7 | |

# Register architectures vary radically

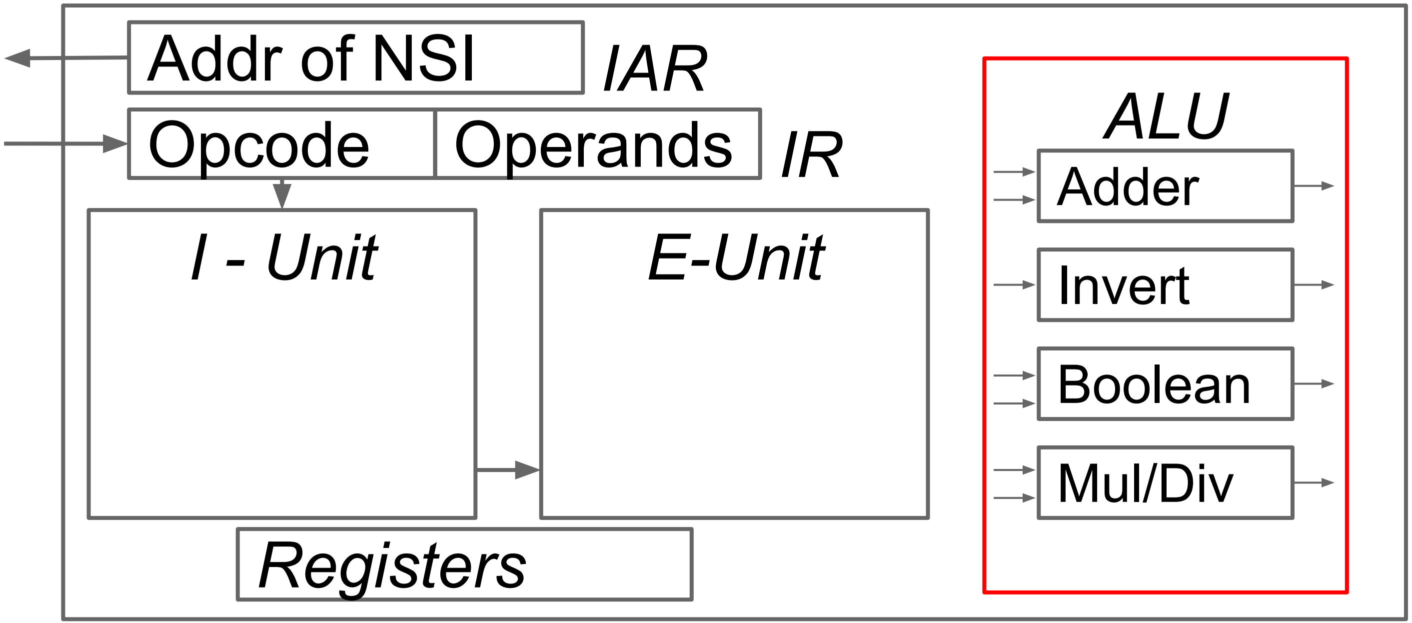
| x86 (12 Regs) | | M68000 (16 regs) | |
|---|---|---|---|
| ax | multiply | d0 | (all equal) |
| bx | DS pointer | d1 | |
| ds | Data Segment | a1 | |
| ss | Stack Segment | a2 | |
| es | Extra segment | a3 | |
| | | a4 | |
| | | a5 | |
| | | a6 | |
| | | a7 | |

**Use a specific set**

⇓

**Lose portability**

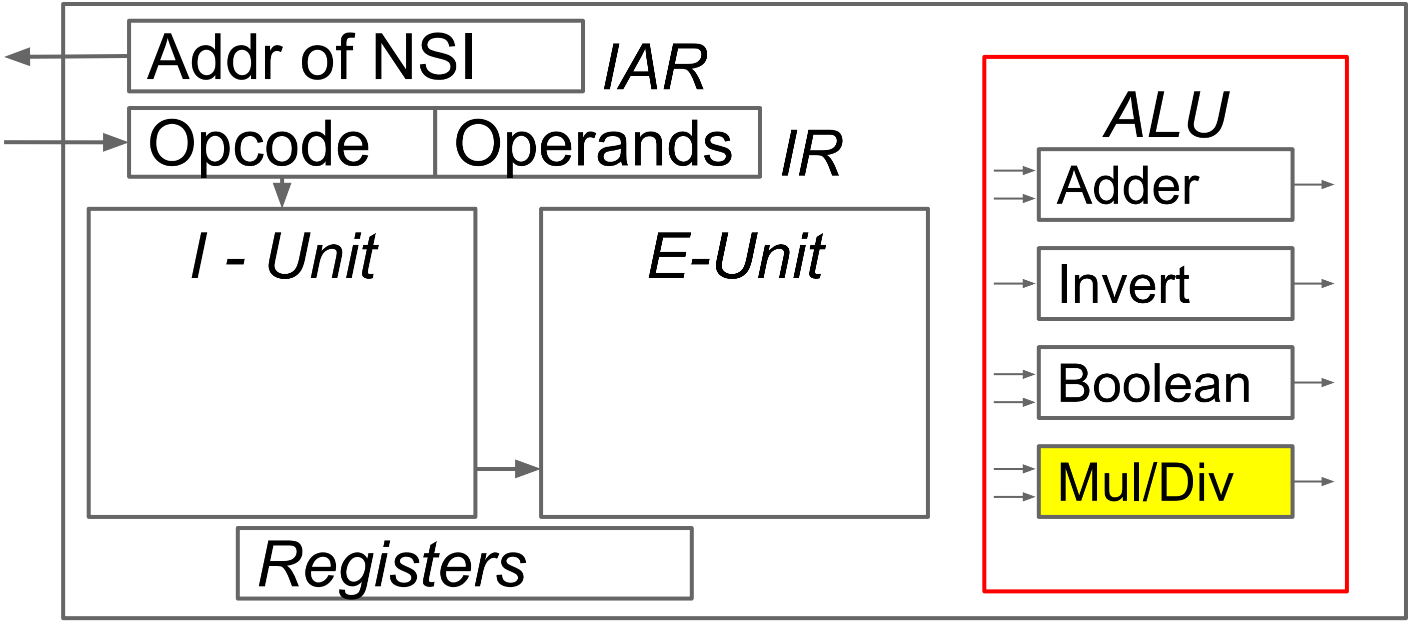
# CPU




**ALU -- Arithmetic Logic Unit**

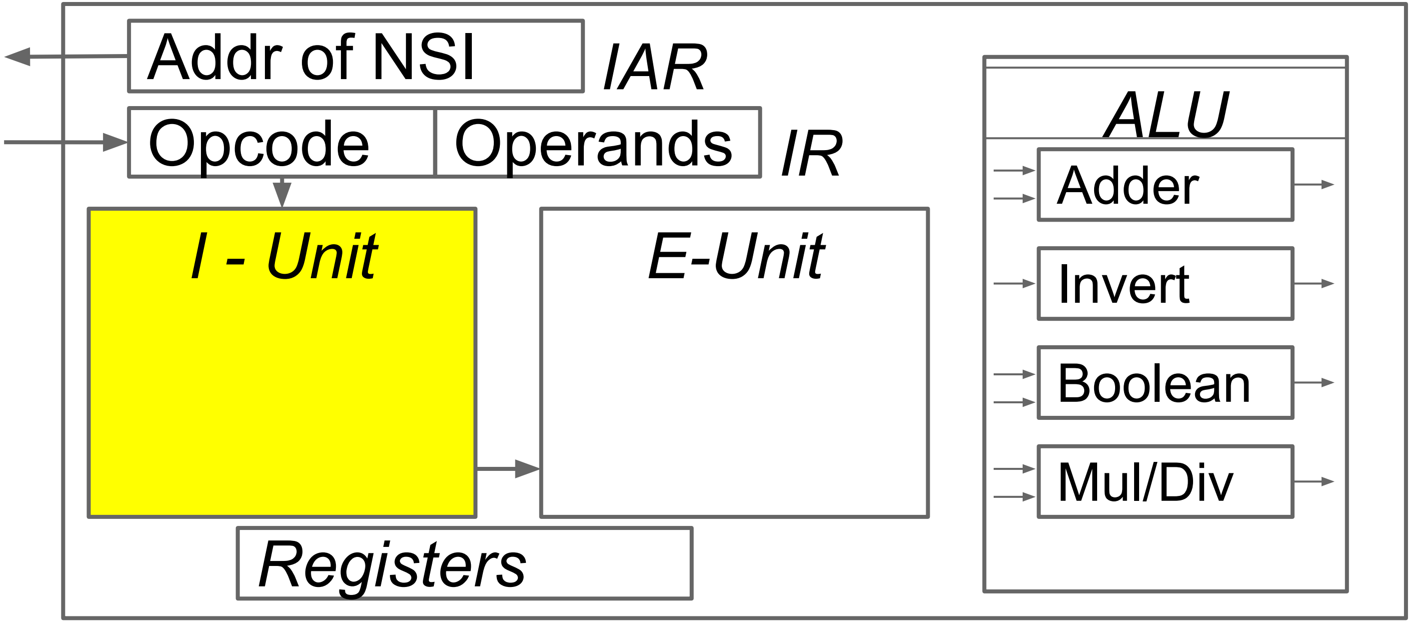
- **Digital circuitry that performs arithmetic and logical operations**

# CPU



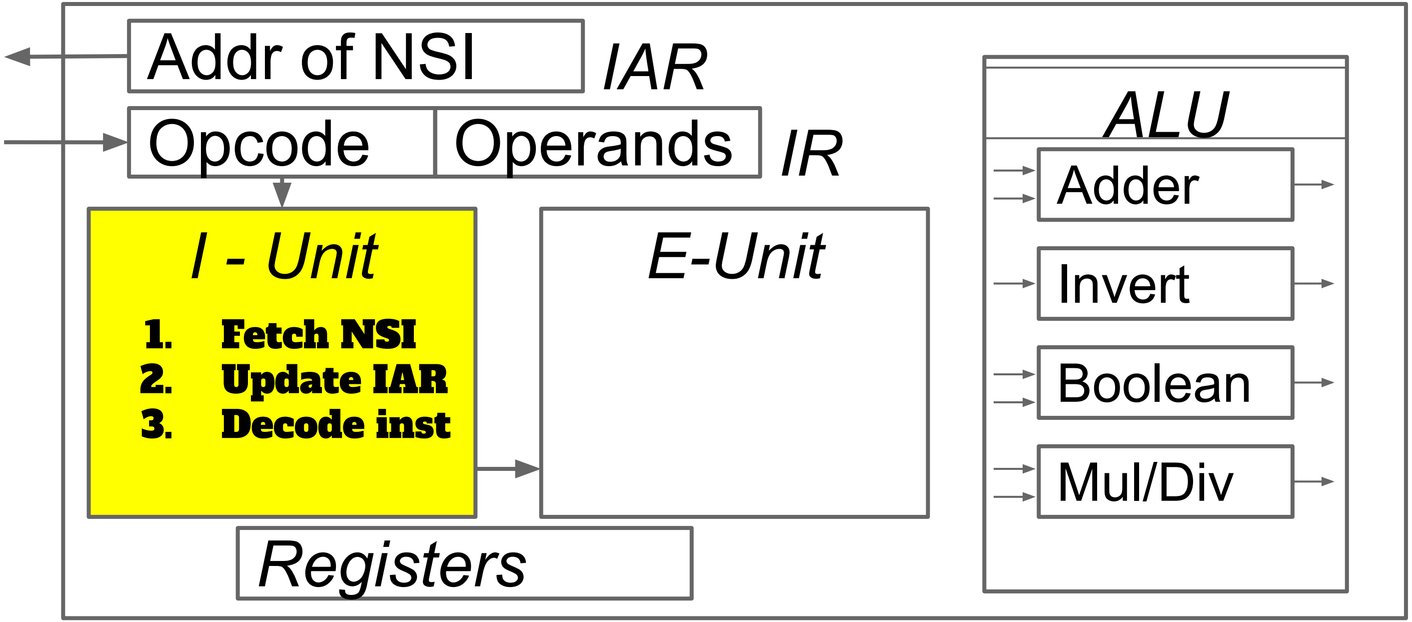

- **It depends on the need**
- **Requires many transistors**

# CPU




**I-Unit controls fetching and decoding instructions**

# CPU




**I-Unit controls fetching and decoding instructions**

# Decode

- What does the instruction do?
- Locate the needed data
    - Register
    - Variable
    - List
    - Structure
    - Immediate constant

# Addressing modes

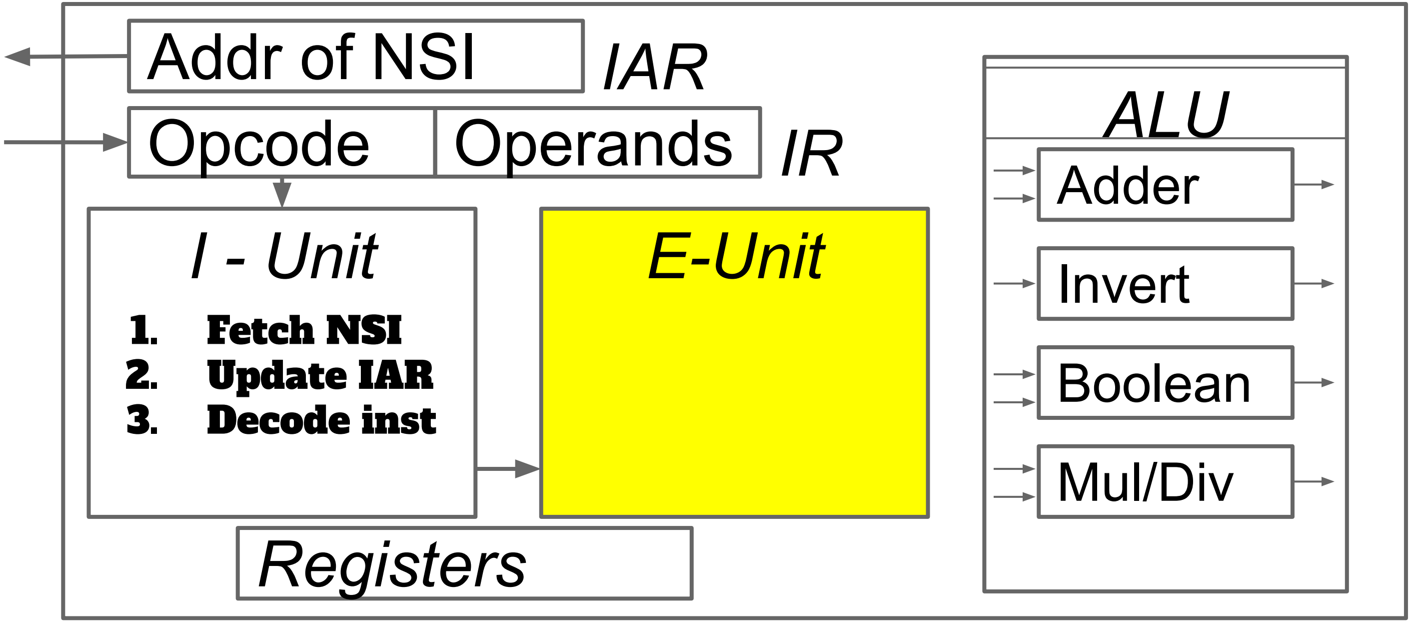
- How does the hardware locate data?
- Address modes
  - Instruction syntax/semantics
  - A handful of different modes for getting operands
  - Determines the *effective address*
- Effective address
  - Actual location of the data
  - Determined by the address mode

# Decode

- What does the instructions do
- Locate the needed data
    - Register
    - Variable
    - List
    - Structure
    - Immediate constant

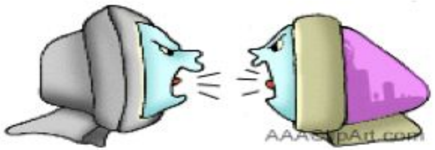
**Effective address**

# CPU




**E-unit executes the instruction**

# CISC vs RISC



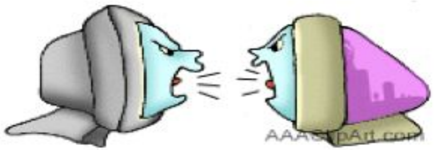
CISC - 1960 ... 1990
Microcoding

RISC - 1990
Hardwired

**Complex Instruction Set Computing**

**Reduced Instruction Set Computing**

# CISC vs RISC



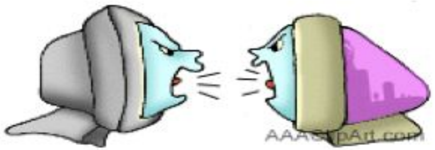
CISC - 1960 ... 1990 Microcoding

RISC - 1990 Hardwired

**CISC - As chip costs rapidly decreased (1960-1990) engineers put more and more function into the hardware.**

# CISC vs RISC

- **Lots of function**
- **Complicated decode**
- **Slow**



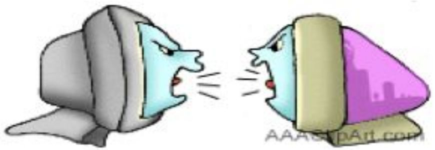
CISC - 1960 ... 1990 Microcoding

RISC - 1990 Hardwired

- **Many ways to access data (address modes)**
- **Machine Instructions**
  - **complicated and fancy**
  - **variable size**
  - **variable execution time**

# CISC vs RISC

- **Lots of function**
- **Complicated decode**
- **Slow**



CISC - 1960 ... 1990
Microcoding

RISC - 1990
Hardwired

- **Many ways [illegible] access data (add[illegible]odes)**
- **Mach[illegible]tructions**
  - **co[illegible]ed and fancy**
  - **vari[illegible] size**
  - **variable execution time**

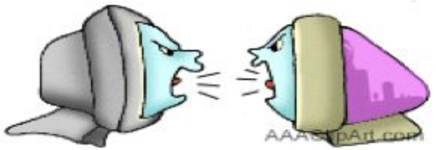
**Too complex**

**Too specific**

**Circa 1990 ... examination of real code showed programmers were not using the complex functions**

# CISC vs RISC

- Lots of function
- Complicated decode
- Slow




CISC - 1960 ... 1990
Microcoding

RISC - 1990
Hardwired

- Lean hardware
- Simple decode
- Fast

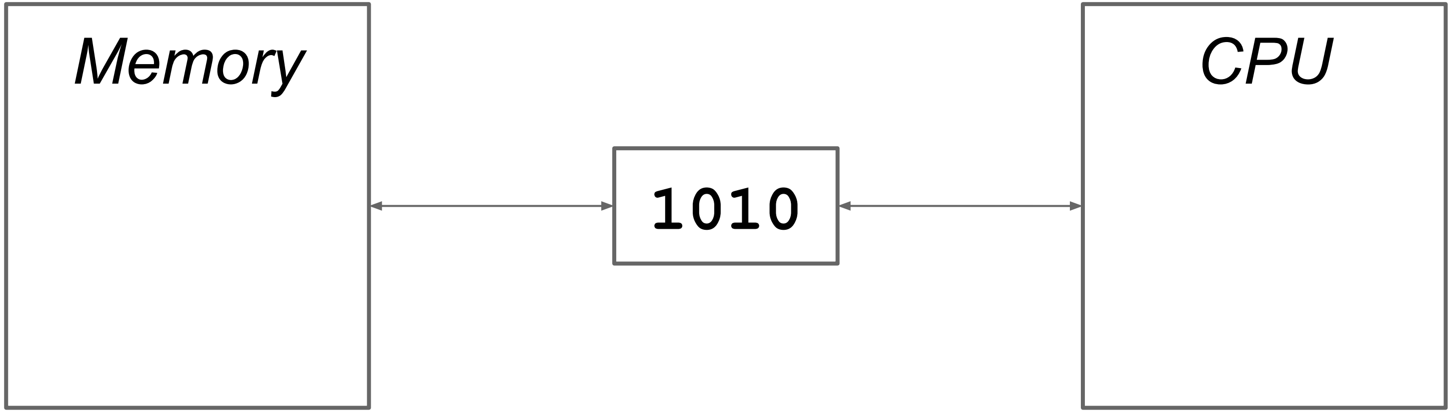
- Many ways to access data (a… …des)
- Machi… …ions
  - complicated and fancy
  - variable size
  - variable execution time

Intel
x86

- Only simple load and stor… …mory
- Mac…
  Inst…
  - only basic operations
  - fixed size
  - fixed execution times

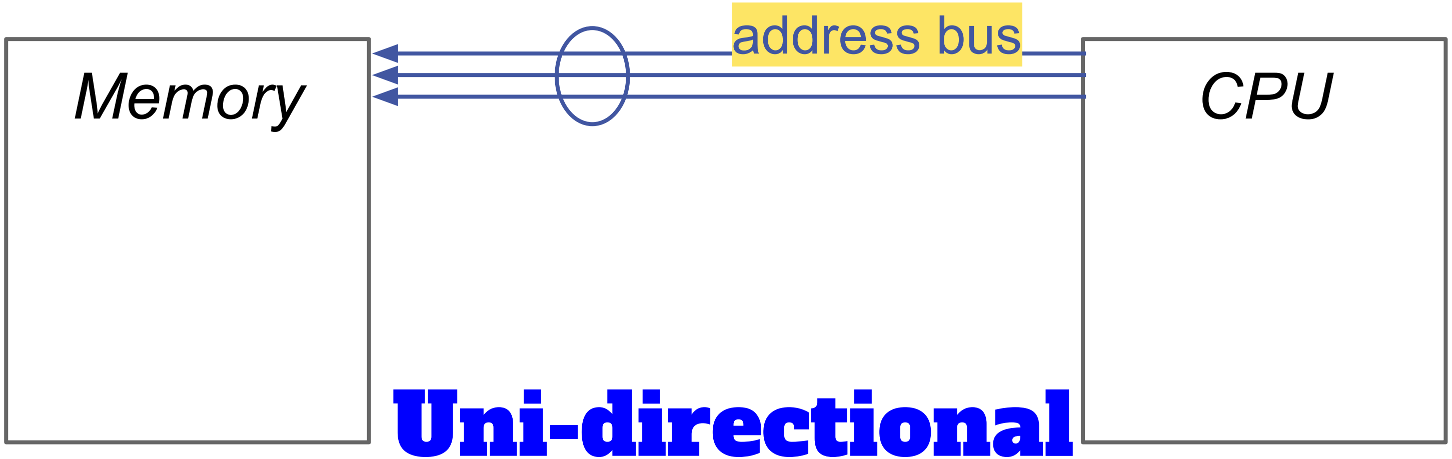
ARM
Advanced
RISC Machine

# Buses & Protocols




Rules for moving data between the CPU and memory

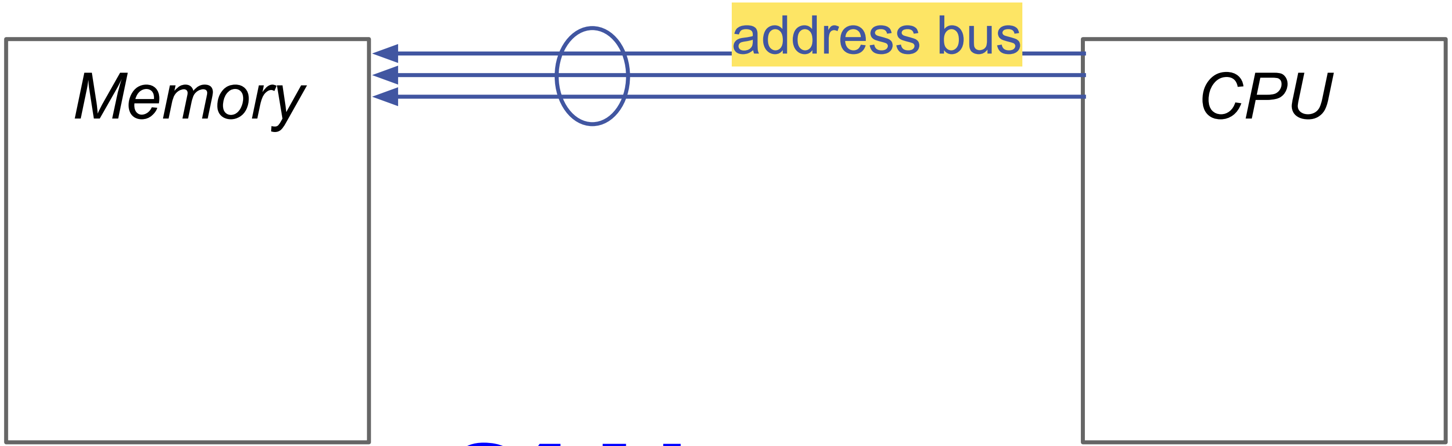
# Buses & Protocols




**Uni-directional**

**Carries the desired address from the CPU to memory**

# Buses & Protocols




**32-bit system**

**Can address**

**$2^{32}$ bytes = 4 GB**

**64-bit system**

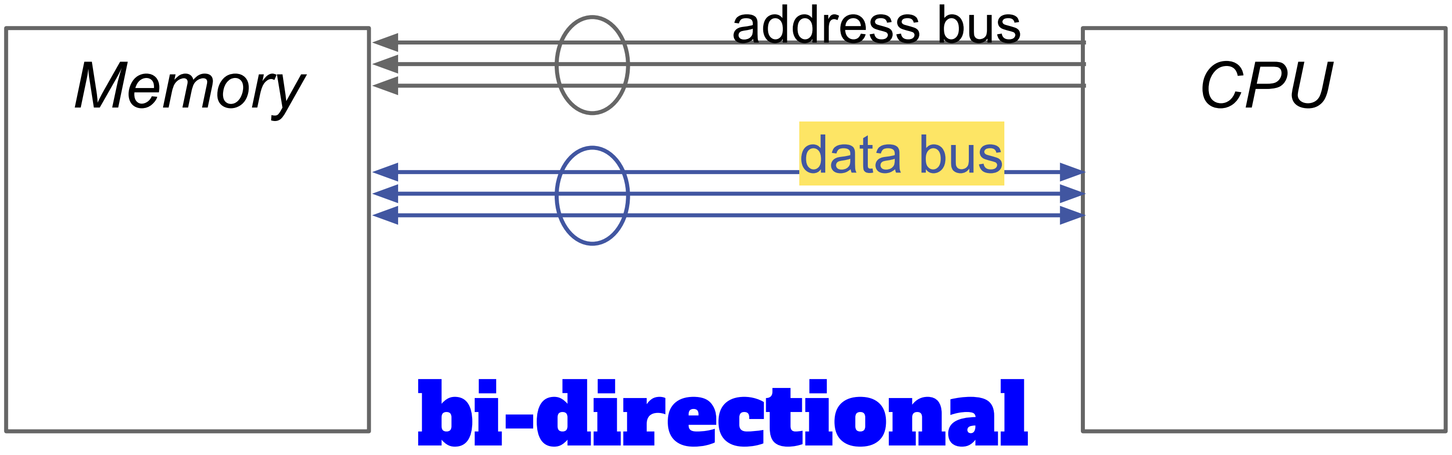
**Can address**

**$2^{64}$ bytes (about 16 million terabytes)**

**Most only implement 36 bits**

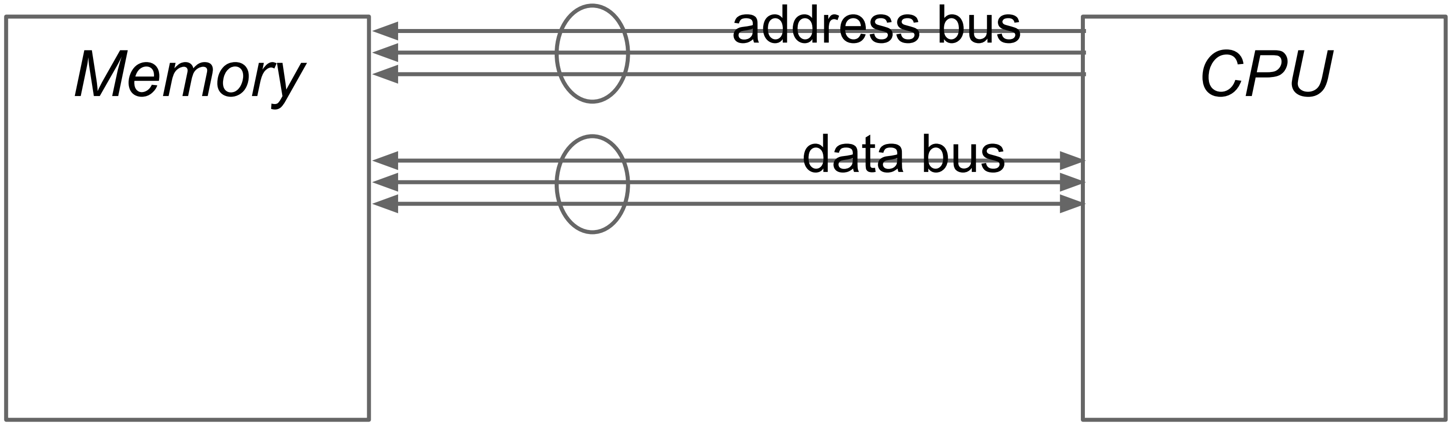
**That's $2^{36}$ bytes = 64 GB**

# Buses & Protocols



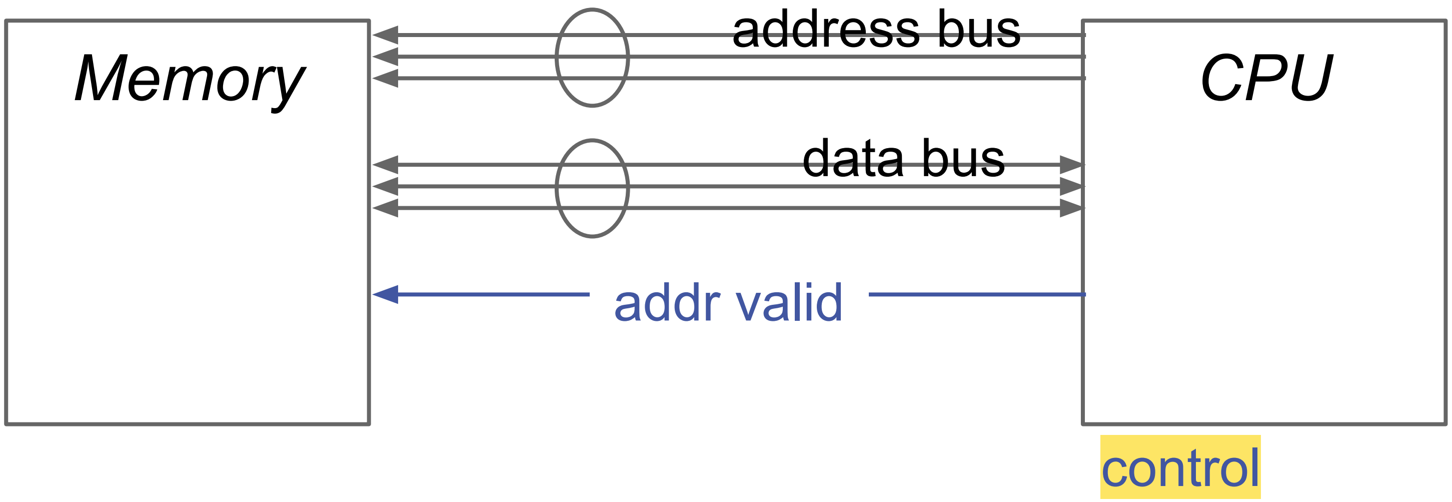

**Carries data between the CPU and memory**

# Buses & Protocols




control

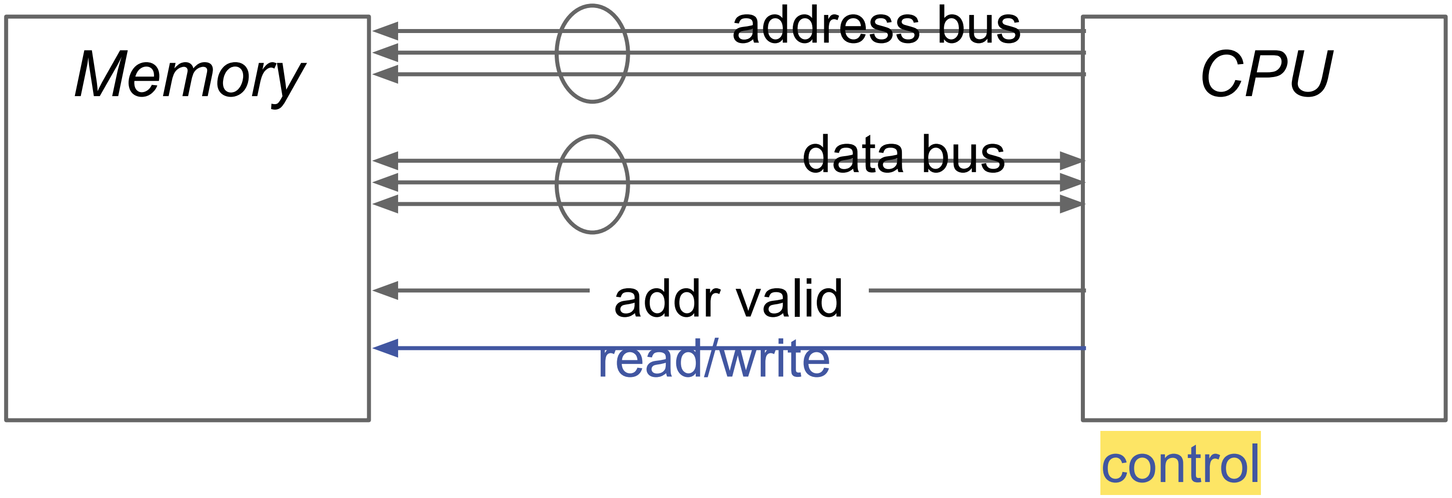
# Buses & Protocols




**0 = no work for memory**

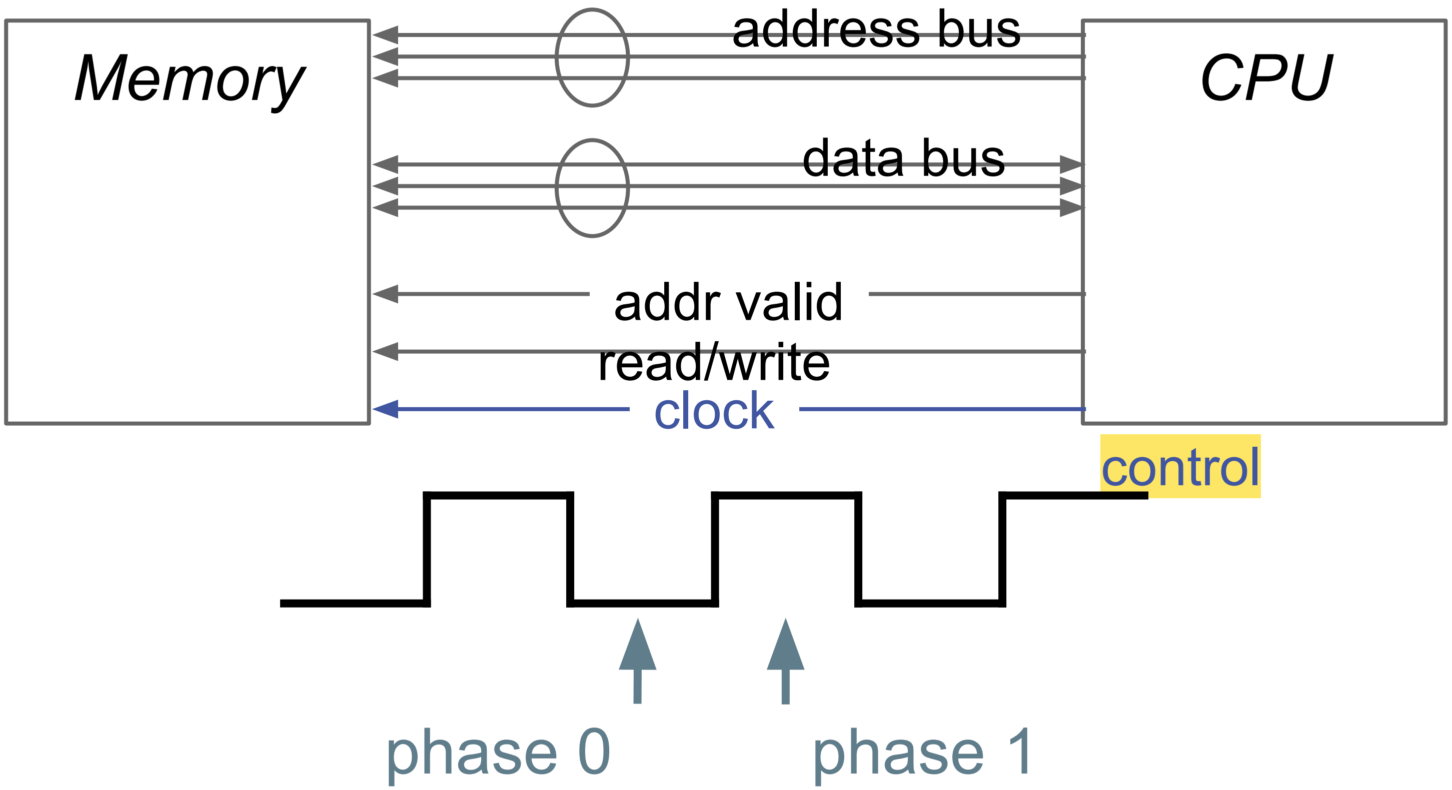
**1 = work for memory**

# Buses & Protocols




**0 = read operation**
**1 = write operation**

# Buses & Protocols

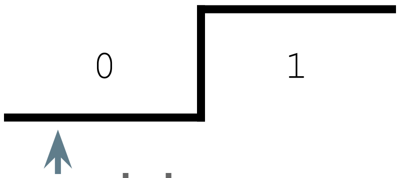


# Buses & Protocols

Example - Move 1 byte from memory to CPU

(Our own bus protocol - rules for communicating)

**Phase 0**




**Phase 1**

Initial condition: address valid line set to false

# Buses & Protocols

Example - Move 1 byte from memory to CPU

(Our own bus protocol - rules for communicating)

**Phase 0**


**Phase 1**

Initial condition: address valid line set to false

**CPU**

- puts address on address bus

# Buses & Protocols

Example - Move 1 byte from memory to CPU

(Our own bus protocol - rules for communicating)

**Phase 0**


**Phase 1**

Initial condition: address valid line set to false

**CPU**

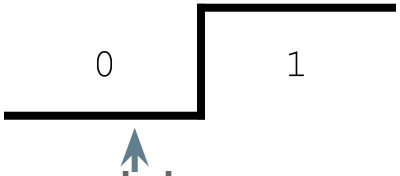
- puts address on address bus
- sets R/W to read

# Buses & Protocols

Example - Move 1 byte from memory to CPU

(Our own bus protocol - rules for communicating)

**Phase 0**




**Phase 1**

Initial condition: address valid line set to false

**CPU**

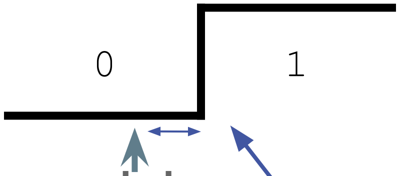
- puts address on address bus
- sets R/W to read
- sets Address Valid to true

# Buses & Protocols

Example - Move 1 byte from memory to CPU

(Our own bus protocol - rules for communicating)

Phase 0




Phase 1

Initial condition: address valid line set to false

**CPU**

- puts address on address bus
- sets R/W to read
- sets Address Valid to true

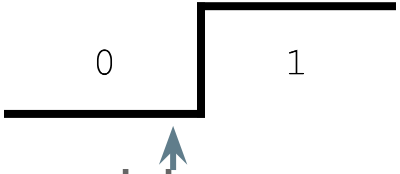
**Design must allow memory to complete its work before phase 0 ends**

# Buses & Protocols

Example - Move 1 byte from memory to CPU

(Our own bus protocol - rules for communicating)

**Phase 0**




**Phase 1**

Initial condition: address valid line set to false

**CPU**

- puts address on address bus
- sets R/W to read
- sets Address Valid to true

**Memory**

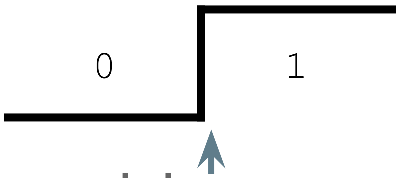
- puts data on Data Bus

# Buses & Protocols

Example - Move 1 byte from memory to CPU

(Our own bus protocol - rules for communicating)

**Phase 0**




**Phase 1**

Initial condition: address valid line set to false

**CPU**
- puts address on address bus
- sets R/W to read
- sets Address Valid to true

**Memory**
- puts data on Data Bus

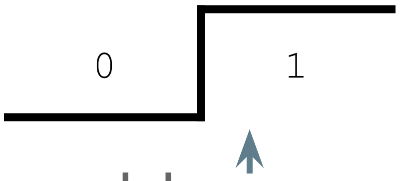
**CPU**
- removes data

# Buses & Protocols

Example - Move 1 byte from memory to CPU

(Our own bus protocol - rules for communicating)

**Phase 0**




**Phase 1**

Initial condition: address valid line set to false

**CPU**
- puts address on address bus
- sets R/W to read
- sets Address Valid to true

**Memory**
- puts data on Data Bus

**CPU**
- removes data
- sets Address Valid to false

# What errors can occur?

## as instructions are executed

# What errors can occur?

- Reference non-existent memory
- Fetch an instruction from a data field
- Store data on top of an instruction
- Bits are corrupted as they move over buses

# What errors can occur?

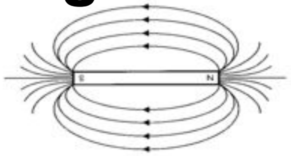
- Reference non-existent memory
- Fetch an instruction from a data field
- Store data on top of an instruction
- Bits are corrupted as they move over buses

A bus is just a wire

# What errors can occur?

- Reference non-existent memory
- Fetch an instruction from a data field
- Store data on top of an instruction
- Bits are corrupted as they move over buses

Introduce a
**Magnetic Field**



Bit

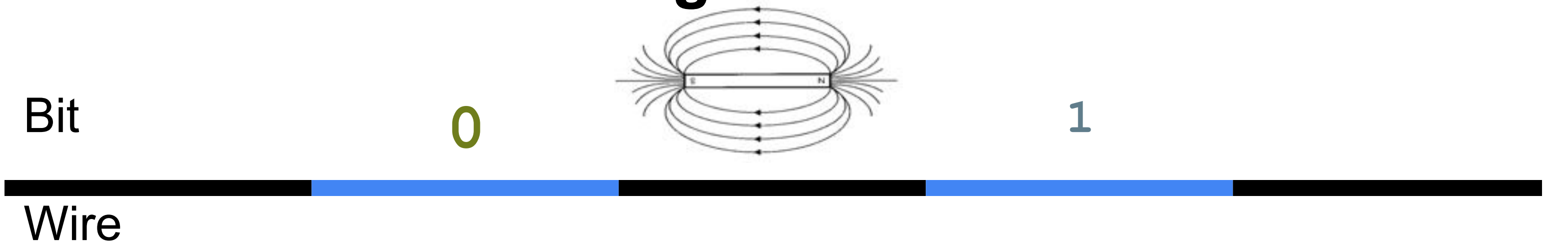
0

Wire

# What errors can occur?

- Reference non-existent memory
- Fetch an instruction from a data field
- Store data on top of an instruction
- Bits are corrupted as they move over buses

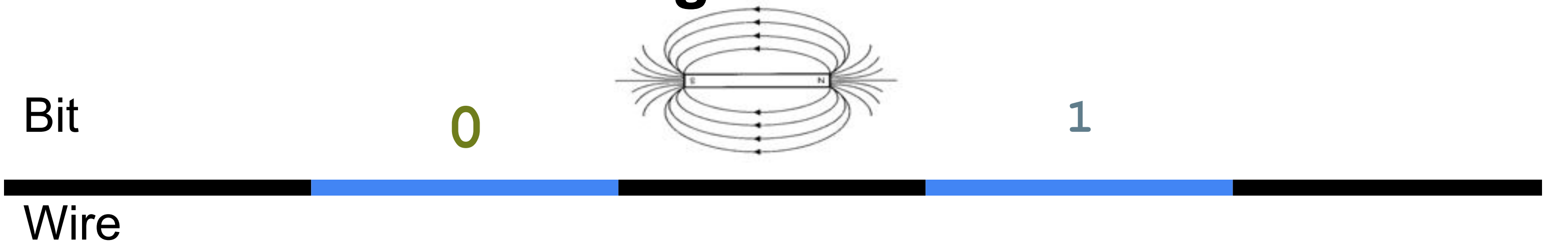
Introduce a **Magnetic Field**



# What errors can occur?

- Reference non-existent memory
- Fetch an instruction from a data field
- Store data on top of an instruction
- Bits are corrupted as they move over buses

What causes a
**Magnetic Field**



# What errors can occur?

- Reference non-existent memory
- Fetch an instruction from a data field
- Store data on top of an instruction
- Bits are corrupted as they move over buses

# What errors can occur?

- Reference non-existent memory
- Fetch an instruction from a data field
- Store data on top of an instruction
- Bits are corrupted as they move over buses

# What errors can occur?

- Reference non-existent memory
- Fetch an instruction from a data field
- Store data on top of an instruction
- Bits are corrupted as they move over buses


# What errors can occur?

- Reference non-existent memory
- Fetch an instruction from a data field
- Store data on top of an instruction
- Bits are corrupted as they move over buses
- Overflow or divide by zero
- Bug in the microcode for an instruction

**What factors affect the system's ability to detect and handle these errors?**

# Factor 1 … Cost

While mainframes cost millions, the driving force for the PC is low cost

A way to save money is leave out error detection and correction hardware

# Factor 2 … Complexity

| CPU | Circuits | Text Pages |
|---|---:|---:|
| 8086 | 29,000 | 1 |
| Pentium 4 | 42,000,000 | 1,448 |
| Core i7 Quad | 731,000,000 | 25,207 |

Could you proofread and guarantee no errors in 25,207 pages of text?

FLOATING-POINT UNIT GETS MASK UPDATE

# Intel fixes Pentium glitch

*Santa Clara, Calif.* — To correct an anomaly that caused inaccurate results on some high-precision calculations, Intel Corp. last week confirmed that it had updated the floating-point unit (FPU) in the Pentium microprocessor.

The company said that the glitch was discovered midyear and was fixed with a mask change in recent silicon. "This was a very rare con-

posting of a private e-mail communication from Lynchburg College (Lynchburg, Va.) mathematics professor Thomas Nicely. "The Pentium floating-point unit is returning erroneous values for certain division operations," he wrote. "For example, 1/824633702441 is calculated incorrectly (all digits beyond the eighth significant digit are in error). This can be verified . . . by computing (824633702441.0) × (1/824633702441.0), which should equal 1 exactly (within some extremely small rounding error; in general, coprocessor results should contain 19 significant decimal digits). However, the Pentiums tested return 0.999999996274709702 for this calculation."

"The bug has been observed on all Pentiums I have tested or had tested to date, including a Dell P90, a Gateway P90, a Micron P60, an Insight P60 and a Packard-Bell P60.

It has not been observed on any 486 or earlier system, even those with a PCI bus. If the floating-point unit is locked out (not always possible), the error disappears."

majority of people, it will be irrelevant. But engineers may have a different outlook."

A spot check conducted at *EE Times* last week tested out Nicely's expression on an AcerPower Minitower Pentium/60 machine, which was just received from Acer America. The result was 0.999999996247.

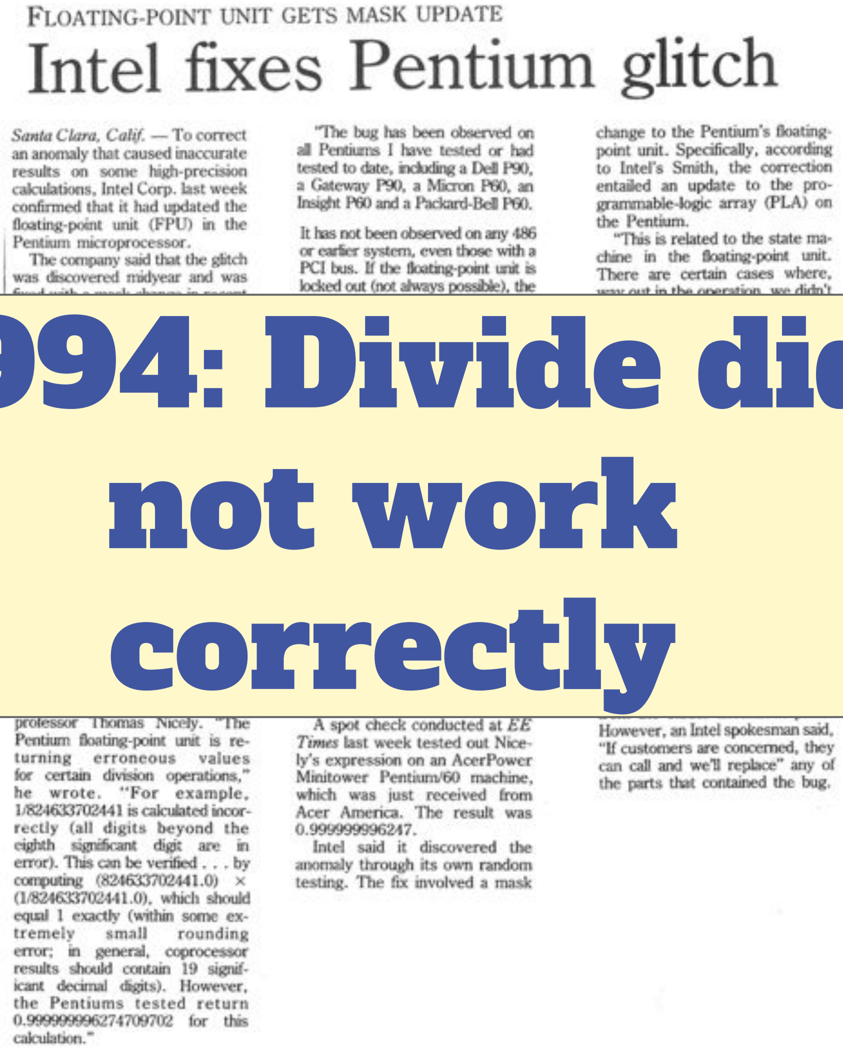
Intel said it discovered the anomaly through its own random testing. The fix involved a mask change to the Pentium's floating-point unit. Specifically, according to Intel's Smith, the correction entailed an update to the programmable-logic array (PLA) on the Pentium.

"This is related to the state machine in the floating-point unit. There are certain cases where, way out in the operation, we didn't handle the precision correctly," he

which became available in the last few months—to differentiate them from the earlier anomalous parts. However, an Intel spokesman said, "If customers are concerned, they can call and we'll replace" any of the parts that contained the bug.

# Chips *will* ship with defects

# 1994: Divide did not work correctly



FLOATING-POINT UNIT GETS MASK UPDATE

## Intel fixes Pentium glitch

*Santa Clara, Calif.* — To correct an anomaly that caused inaccurate results on some high-precision calculations, Intel Corp. last week confirmed that it had updated the floating-point unit (FPU) in the Pentium microprocessor.

The company said that the glitch was discovered midyear and was

professor Thomas Nicely. "The Pentium floating-point unit is returning erroneous values for certain division operations," he wrote. "For example, 1/824633702441 is calculated incorrectly (all digits beyond the eighth significant digit are in error). This can be verified . . . by computing (824633702441.0) × (1/824633702441.0), which should equal 1 exactly (within some extremely small rounding error; in general, coprocessor results should contain 19 significant decimal digits). However, the Pentiums tested return 0.999999996274709702 for this calculation."

"The bug has been observed on all Pentiums I have tested or had tested to date, including a Dell P90, a Gateway P90, a Micron P60, an Insight P60 and a Packard-Bell P60.

It has not been observed on any 486 or earlier system, even those with a PCI bus. If the floating-point unit is locked out (not always possible), the

A spot check conducted at *EE Times* last week tested out Nicely's expression on an AcerPower Minitower Pentium/60 machine, which was just received from Acer America. The result was 0.999999996247.

Intel said it discovered the anomaly through its own random testing. The fix involved a mask change to the Pentium's floating-point unit. Specifically, according to Intel's Smith, the correction entailed an update to the programmable-logic array (PLA) on the Pentium.

"This is related to the state machine in the floating-point unit. There are certain cases where,

However, an Intel spokesman said, "If customers are concerned, they can call and we'll replace" any of the parts that contained the bug.

# How to *not* handle a problem

FLOATING-POINT UNIT GETS MASK UPDATE

## Intel fixes Pentium glitch

*Santa Clara, Calif.* — To correct an anomaly that caused inaccurate results on some high-precision calculations, Intel Corp. last week confirmed that it had updated the floating-point unit (FPU) in the Pentium microprocessor.

The company said that the glitch was discovered midyear and was fixed with a mask change in recent silicon. "This was a very rare con-

Pentium floating-point unit is returning erroneous values for certain division operations," he wrote. "For example, 1/824633702441 is calculated incorrectly (all digits beyond the eighth significant digit are in error). This can be verified . . . by computing (824633702441.0) × (1/824633702441.0), which should equal 1 exactly (within some extremely small rounding error; in general, coprocessor results should contain 19 significant decimal digits). However, the Pentiums tested return 0.999999996274709702 for this calculation."

"The bug has been observed on all Pentiums I have tested or had tested to date, including a Dell P90, a Gateway P90, a Micron P60, an Insight P60 and a Packard-Bell P60.

It has not been observed on any 486 or earlier system, even those with a PCI bus. If the floating-point unit is locked out (not always possible), the error disappears."

*Times* last week tested out Nicely's expression on an AcerPower Minitower Pentium/60 machine, which was just received from Acer America. The result was 0.999999996247.

Intel said it discovered the anomaly through its own random testing. The fix involved a mask change to the Pentium's floating-point unit. Specifically, according to Intel's Smith, the correction entailed an update to the programmable-logic array (PLA) on the Pentium.

"This is related to the state machine in the floating-point unit. There are certain cases where, way out in the operation, we didn't handle the precision correctly," he

"If customers are concerned, they can call and we'll replace" any of the parts that contained the bug.

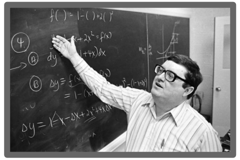
July 1994 - Intel discovers a bug but decides to follow normal procedures meaning 3-5 million bad chips will ship.

Sept 1994 - A math professor discovers the bug. After calling Intel technical support and getting no reaction, he posts his discovery.

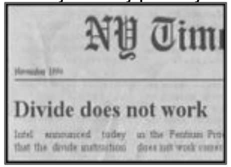
Nov 1994 - Story hits newspapers. Intel calls it a "glitch" will not replace faulty processors.

Dec 1994 - IBM disputes Intel's claim and halts shipment of PCs with the bad CPU.

Dec 1994 - Intel apologizes and agrees to replace bad CPUs at a cost of  $300 million.



https://images.app.goo.gl/L7jDXhrKjqS5B2jn6



Divide does not work

Intel replaces chips

# *Factor 3 … consumer demand*

Do consumers request error handling?

What prompts a manufacturer *to put safety & error handling* features in a product?

What pr… facturer

*to put s… andling*

Ethics

features in a product?

What pr facturer

*to put s andling*

~~Sued~~

features in a product?

What prompts a manufacturer *<u>not</u> to put safety & error handling* features in a product?

Wha turer

*not to* ndling

f ?

# Boeing 737 MAX (2018-2019)

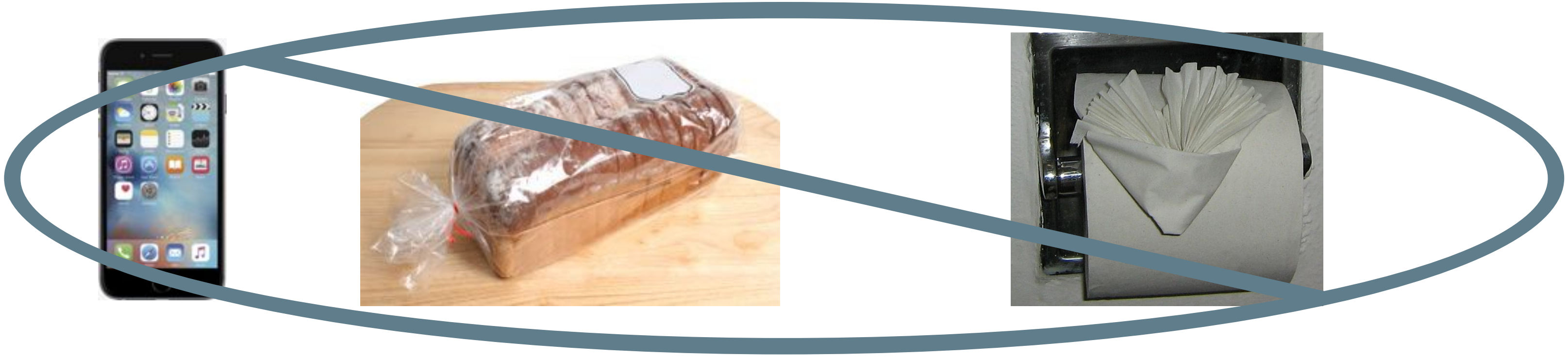
Two plane crashes kill 346 people.

Multiple human and system errors

One of the flaws revealed by the investigation: a system that influenced the direction of the plane was triggered by a single sensor. Any malfunction of the sensor and/or associated software would have had to be dealt with by manually deactivating the software.

https://commons.wikimedia.org/wiki/File:Boeing_737_S7_VQ-BKW_cockpit_May_2011.jpg

What is *the best* product ever created?




Kellogg's
pop-tarts
FROSTED
Brown Sugar Cinnamon

# New Jersey couple sues Kellogg over Pop-Tart fire

July 28, 2001

WOODBURY, N.J. (AP)--A Gloucester County couple has filed a lawsuit against Kellogg Co., claiming a flaming Pop-Tart sparked a fire that caused $100,000 in damage to their home.

Brenda J. Hurff of Washington Township put a cherry Pop-Tart in a toaster before taking her children to preschool. When she returned about 10 to 20 minutes later, smoke was coming from the Gloucester County home and firefighters already had arrived, said Mauro C. Casci, the Hurffs' attorney.

# Question ...

**If the engineers at Kelloggs cannot make Pop-Tarts foolproof ...**


**what hope is there for a *defect free* 731 million circuit computer chip?**