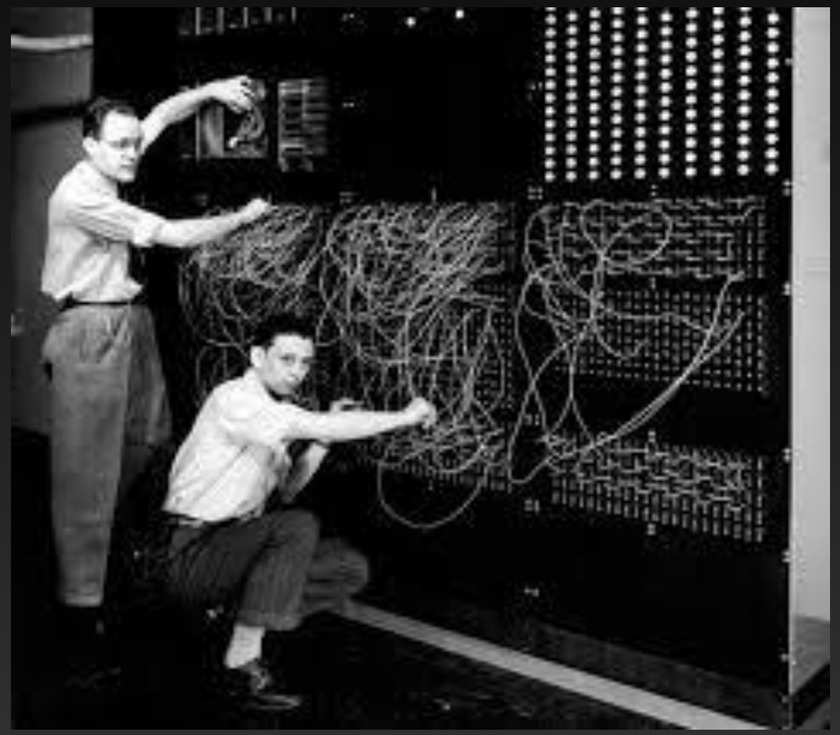
Looking under the hood (or keyboard): Instruction set architec

Eric Liu & Chris O’Brien

March 21, 2022

How do you think a computer works?

Answer waterfall-style in **Slack**!

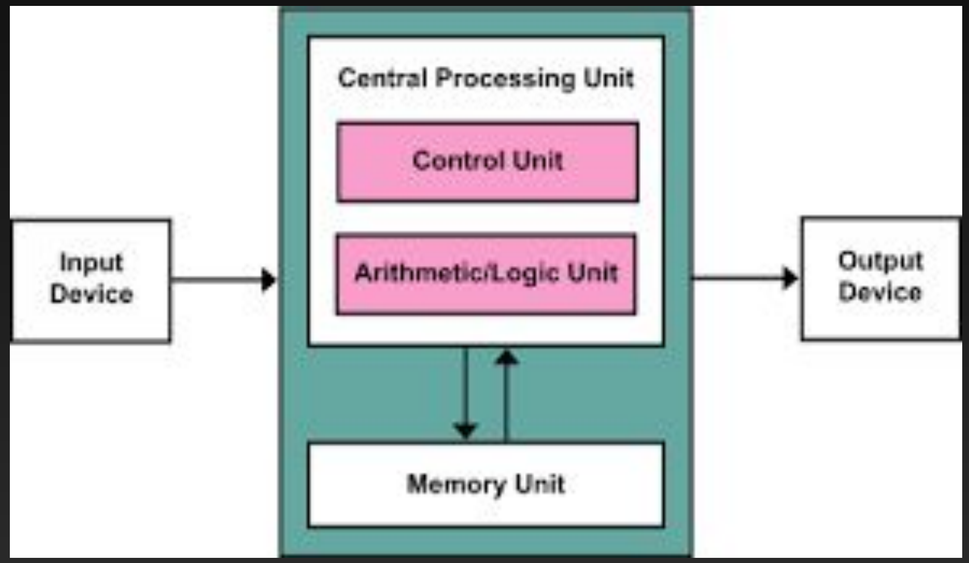


AIM: How does a computer execute instructions?

Computer architecture

The basic set up

- **CPU**: Moves things around, handles arthimetic and logical calculations
- **Registers**: memory with quick access to the CPU
- **Input/Output**: Keyboard, screen. It's what lets humans interact with the computer.



AIM: How does a computer execute instructions?

Activity: The world's slowest computer

- Directly corresponds to real computers
- Runs/Executes at human speed so we can track it
- Everything you do is real computation, not a simulation

AIM: How does a computer execute instructions?

Activity: The world's slowest computers

Instruction set: commands machine responds to.

- **`set a n`**: writes *n* in *cell a*
- **`+ a b c`**: Adds contents of *cell a* to *cell b*. Writes the results in *cell c*.
- **`< a b c`**: Compares *cell a* and *cell b*. If *cell a* is less, write 1 in *cell c*. Otherwise write 0 in *cell c*.
- **`plot a b`**: Fills in the cell at column *cell a*, row *cell b*.
- **`jump a b`**: If *cell a* is 1, go to the instruction numbered *cell b* next. Otherwise ignore.

AIM: How does a computer execute instructions?

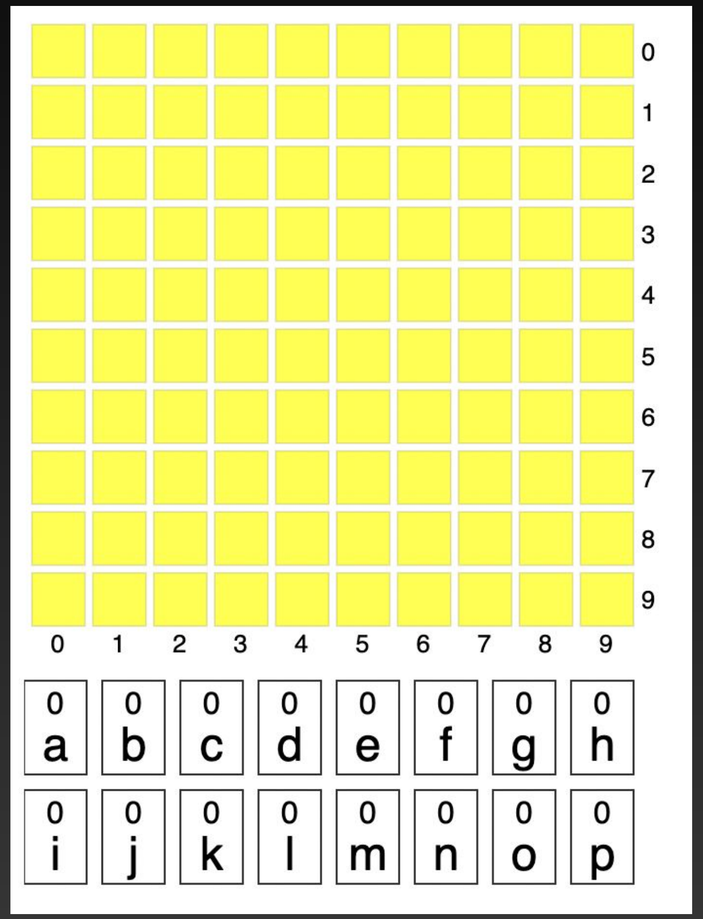
Activity: The world's slowest computers

1. Work with two other partners
   - One is Compiler and *error checker
   - Second partner is CPU/Memory
   - Third partner is the Graphics Card

AIM: How does a computer execute instructions?

Activity: The world's slowest computers

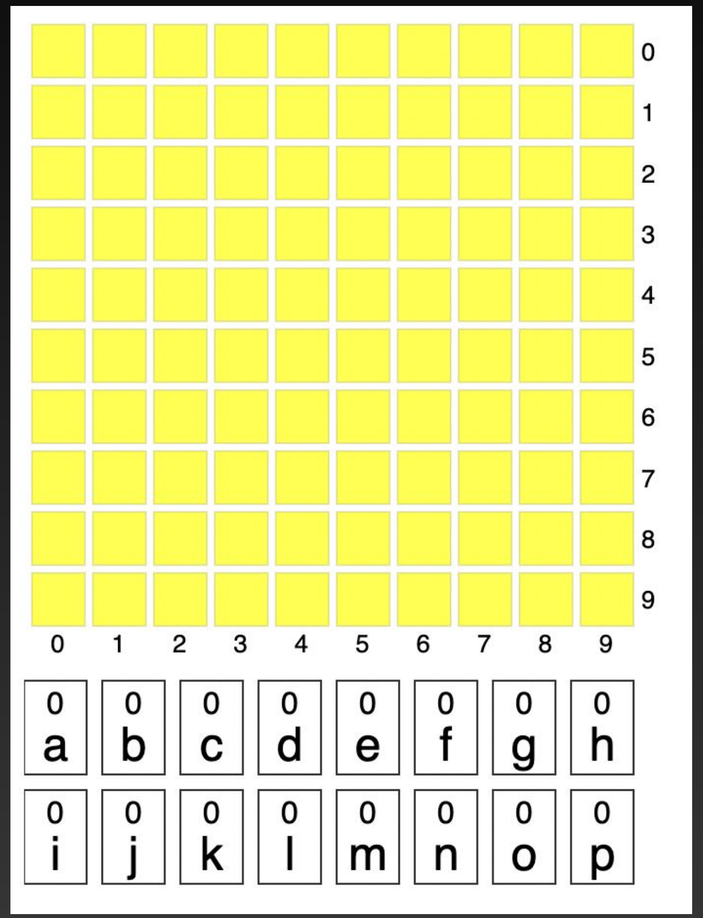
2. Initialize the monitor
   - Draw a 10x10 grid. (Open sheet)
   - Number rows and columns from 0 thru 9. (zero based indexing!)
3. Initialize the computer memory
   - Label your memory registers ***a*** thru ***p***



AIM: How does a computer execute instructions?

Activity: The world's slowest computers

- **Partner 1 - Compiler:**
  - Read each instruction carefully
- **Partner 2:**
  - executes instruction
  - Reads/Writes in memory as needed
- **Partner 3:**
  - Share your screen and Plot graphics when directed

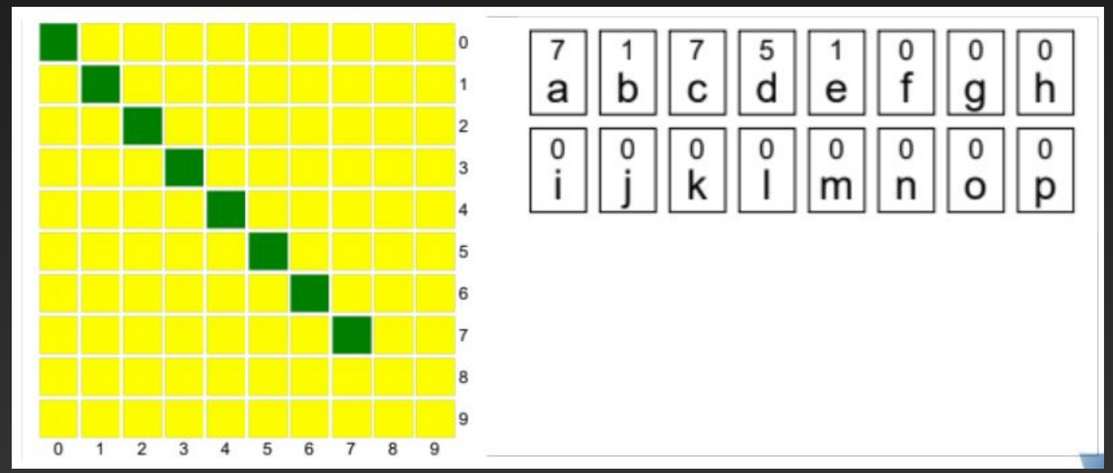


AIM: How does a computer execute instructions?

Activity: The world's slowest computers

- Get started! What's the output

Answer:

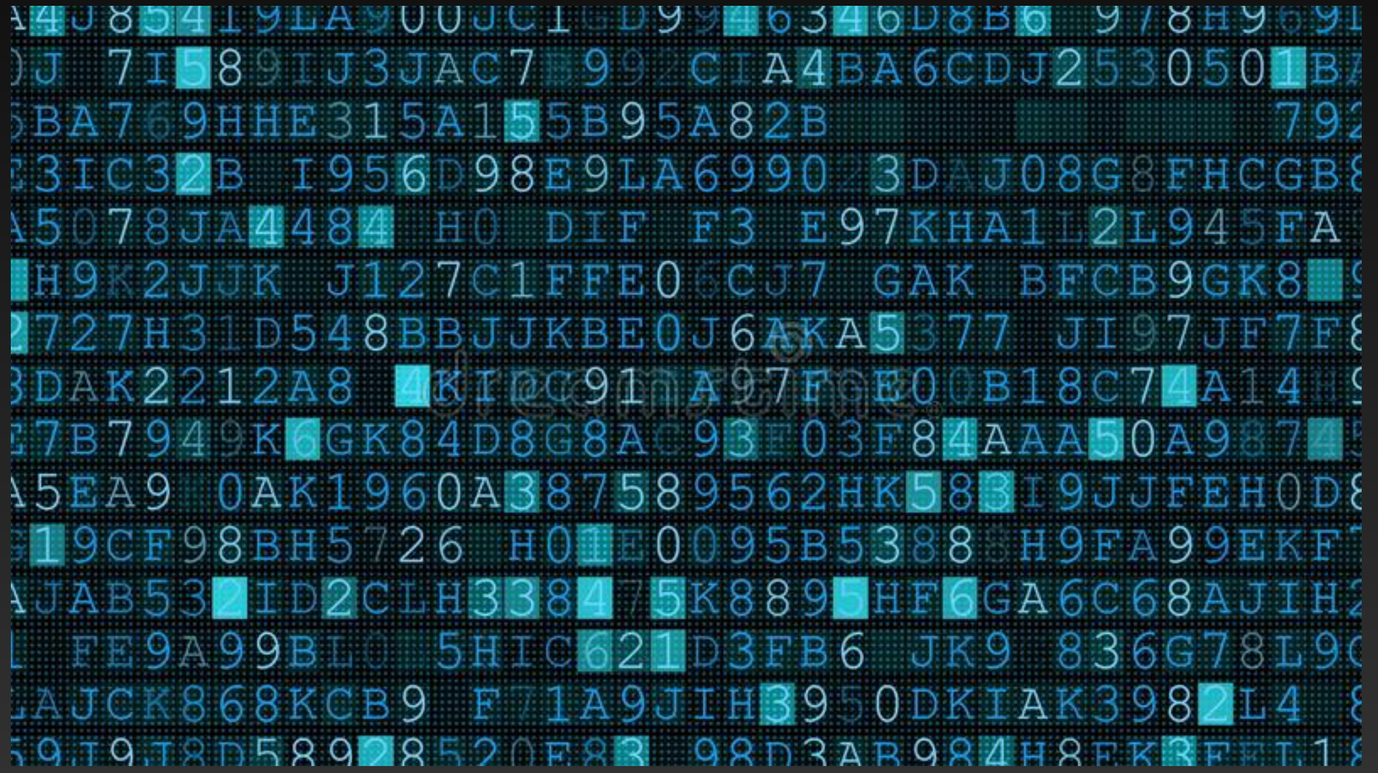


AIM: How does a computer execute instructions?

Instruction set architecture

Machine and assembly language

- **Machine code** on real computers is typically written in binary numbers.
- **Assembly language** is a human-readable mnemonic for machine code.
- An **instruction set architecture** defines what commands control the CPU.



AIM: How does a computer execute instructions?

The “Little Man Computer” (LMC)

A simple(-ish) ISA:

- **One clerk (the “little man”)**
  - fetches, decodes, and executes instructions
- **100 mailbox registers, numbered 0-99**
  - Each mailbox holds one instruction
- **Two trays:** input **and** output
- **Two special registers:**
  - A program counter
  - An accumulator

Central Pension Office, Prague, 1937.

AIM: How does a computer execute instructions?

THE LMC Instruction set

Mnemonic code (Numeric code) : Definition

- `INP (901)` : **take a value from the** input **tray, place in the** accumulator**.**
- `STA N (3xx)` : **Store value in** accumulator **in mailbox** #N
- `LDA N: (5xx)` **: takes value from mailbox #N, place in** accumulator
- `ADD N (1xx)` : **Add value from mailbox #N to the** accumulator**.**
- `SUB N (2xx)` : **Subtracts value in mailbox #N from value in the** accumulator**.**
- `OUT (902)` : **Places value from** accumulator **into the output**
- `HLT (000)` **: halt**
- `DAT ()` **: Reserve spot as** data **the memory address**
- `BRA (6xx)` **: Branch always. Sets the program counter to address xx**
- `BRZ (7xx)` **: Branch if zero. If accumulatr == 0, set program counter to addres xx.**
- `BRP (8xx)` **: Branch if zero or positive. If accumulator >= 0, set program counter to address xx**

AIM: How does a computer execute instructions?

Activity
LMC programming

1. Write a program that input 3 values and returns the sum.
2. **More challenging:**
   Write a program that outputs values 10 through 1 (hint: branching instructions will be useful

- `INP (901)` : **take a value from the** input **tray, place in the** accumulator**.**
- `STA N (3xx)` : **Store value in** accumulator **in mailbox** #N
- `LDA N: (5xx)` **: takes value from mailbox #N, place in** accumulator
- `ADD N (1xx)` : **Add value from mailbox #N to the** accumulator**.**
- `SUB N (2xx)` : **Subtracts value in mailbox #N from value in the** accumulator**.**

AIM: How does a computer execute instructions?

Discussion
LMC vs. other ISAs

- How are the models we explored today different from ‘real’ computers?
- How could you use these models in your classroom?
- What other models (if any have you heard about)?

AIM: How does a computer execute instructions?