Computer Structure and Language

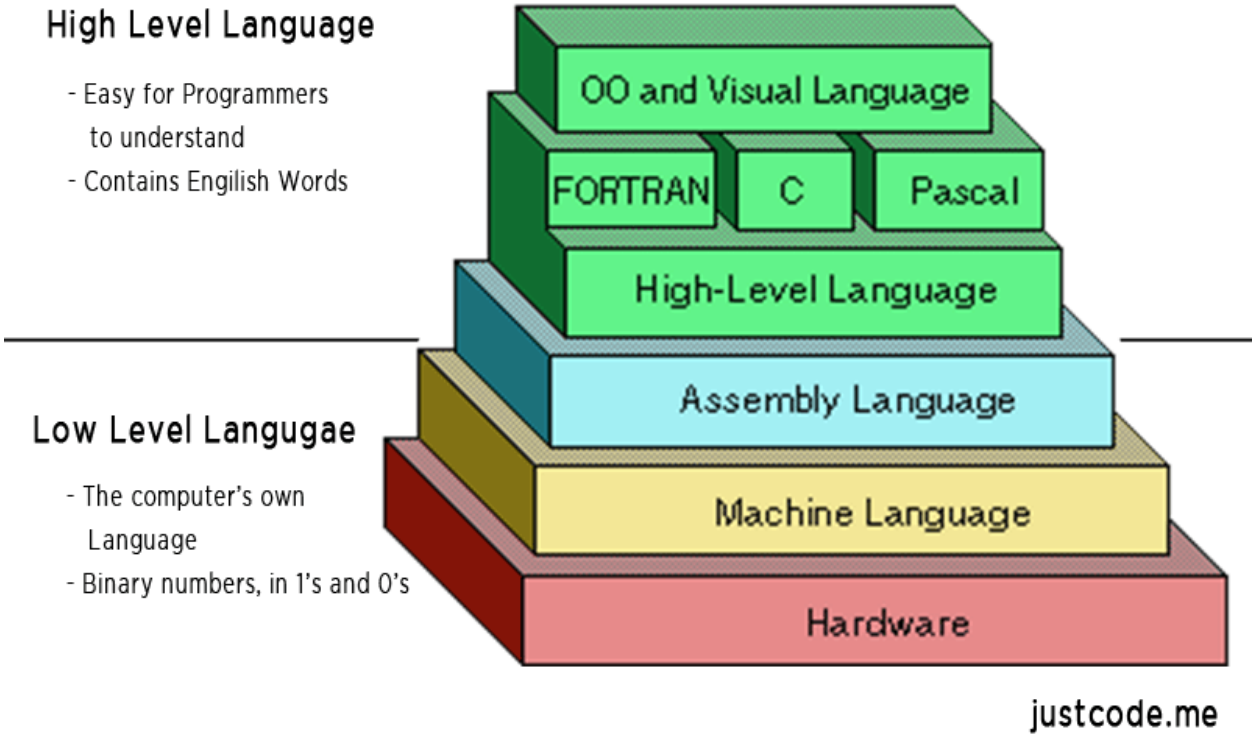
Hamid Sarbazi-Azad
Department of Computer Engineering
Sharif University of Technology (SUT)
Tehran, Iran

Assembly Language

Assembly Language

- A low-level programming language specific to each processor (ISA)
- Human readable names for machine language instructions.

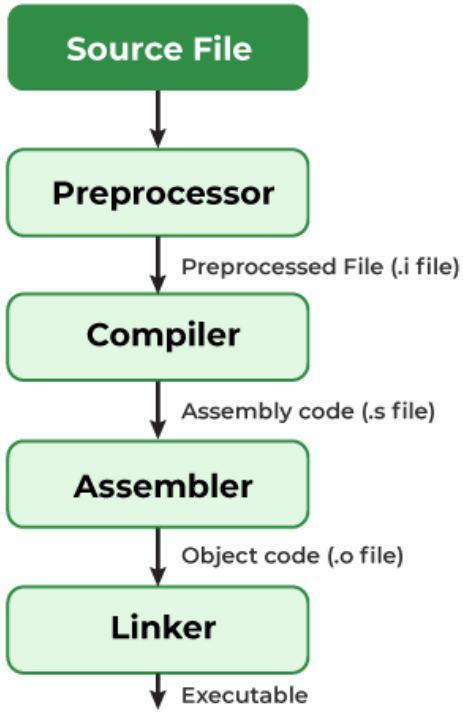
Relation to High-Level Languages



http://justcode.me/assembly/introduction-assembly-language-examples/

Relation to High-Level Languages

- Codes written in high-level languages such as C get “Compiled” to assembly and then “Assembled” to machine Language.



https://www.geeksforgeeks.org/compiling-a-c-program-behind-the-scenes/

Is Assembly still in use?
Why Learn Assembly?

- Not as much as it used to be. Why?
  - Computers are a lot faster than they used to be.
  - Assembly languages being ISA specific.
  - Compilers are much more advanced.

- But people still write code in assembly. Where?
  - Where performance matters.
  - Power (Resource) constrained environments.
  - Where we need low level access to system.
  - Where we want to use a hardware specific feature.
  - In reverse engineering and security analysis.
  - In porting libraries & systems.

- Is there any other reason to learn assembly?
  - Getting Insight

Getting Insight

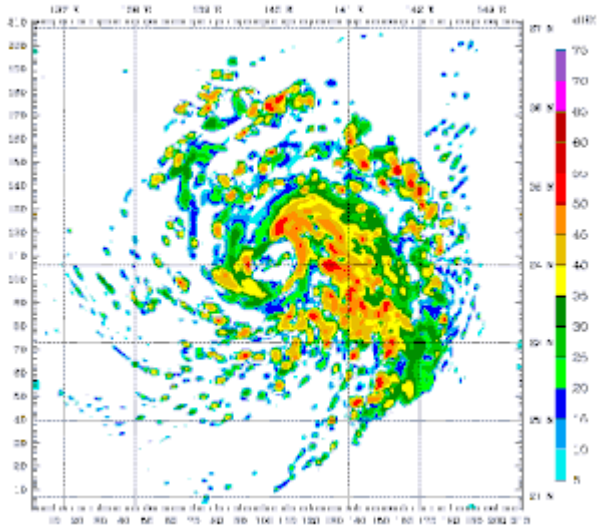
- How programming languages are implemented (code, variables, arrays, functions, etc.)
- How compilers work.
- How operating systems work.
- Computer architecture.
  - Design
  - Implementation
  - …

Performance

- Assembly offers better utilization of resources and less overhead. This leads to better performance.
- Very useful in real-time applications. (robotics, video games, etc.)
- Often the performance critical sections of code or functions are implemented in assembly and then get integrated with rest of the system.
- It is more important in sequential parts of the application.
- This practice used to be more common in the past.

Where does performance matter? (Examples)

- Scientific Computing



https://commons.wikimedia.org/wiki/File:Typhoon_Mawar_2005_computer_simulation.gif

- Video Games

Image credit: Activision

Computers Then vs Now

IBM System 390

- Introduced in 1990s.
- Top model had:
  - x12 637 MHz CPUs
  - 32 GB Memory

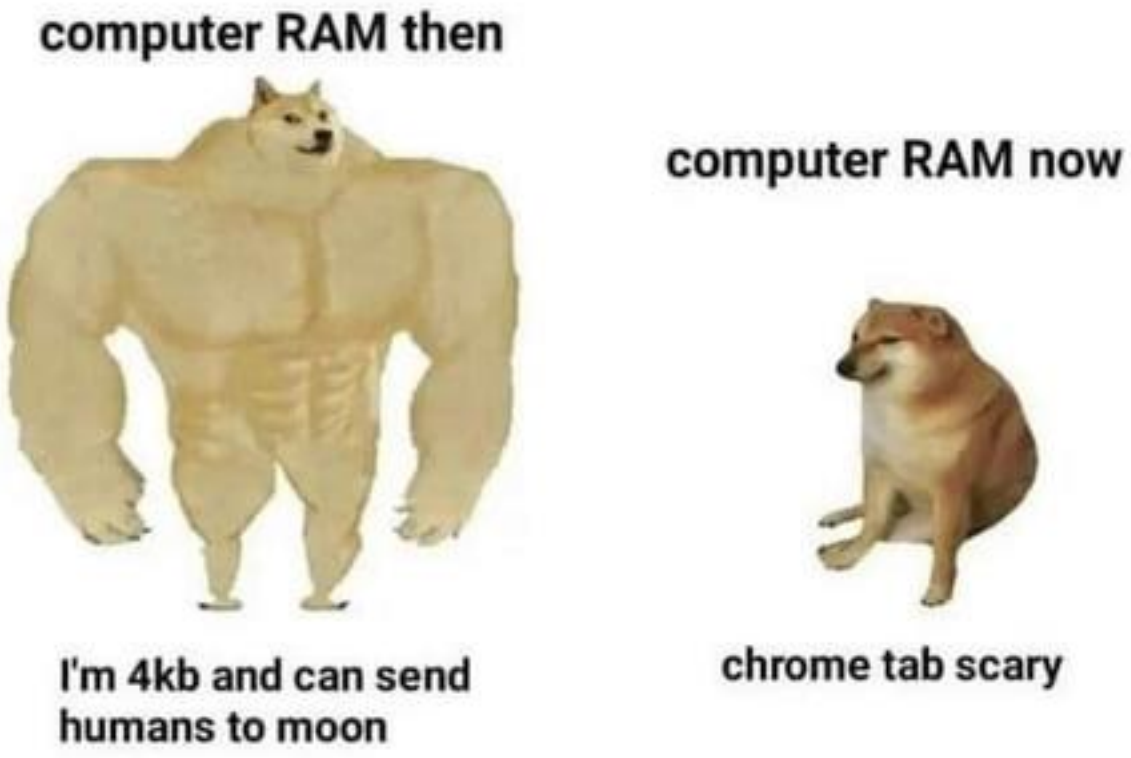
https://en.wikipedia.org/wiki/IBM_System/390

IBM Z16

- Introduced in 2023.
- Fully fitted model has:
  - x200 5.2 GHz CPUs
  - 40 TB Memory

https://en.wikipedia.org/wiki/IBM_Z

Code Optimization Then vs Now



Power (Resource) Constrained Environments.

- Micro-Controllers
- IOT devices
- Mobile Computing

Low Level Access to System

- Operating Systems
  - Specially in case of atomic operations, context switching, protection, etc.
- Drivers
- Compilers

Using Hardware Specific Features

- Most modern CPUs offer hardware accelerators for some common tasks
  - Such as SIMD capabilities for vector operations in x86 or AI accelerators in IBM/Z.
- These features are often absent from high level languages
  - Many of these features require low level programming.
  - They are only present in some architectures.
  - Compilers take time to catch up with latest technologies.

Reverse Engineering

- Process in which a software is deconstructed to extract design information from them.
- Used to understand how viruses works.
- Common Practice in security analysis.
- Machine code could be easily disassembled to assembly language but it’s often very hard to decompile into a high level programming language.

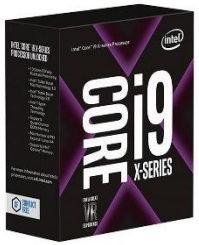
Porting Libraries

- Each programming language has a calling convention for functions.
- By implementing an interface in assembly these calling conventions can be converted and function can be called from other languages.

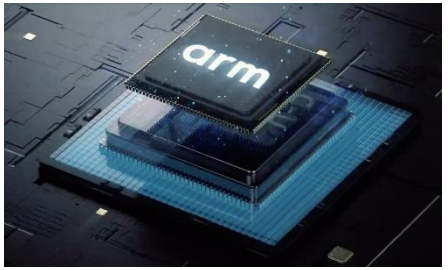
So, how many assemblies are there?

- Each ISA has it's associated assembly language(s).
- This means each processor architecture has an assembly language.
- Virtual Machines like JVM have an assembly language too!
  - One exciting language in this category is the WebAssembly!
- There are also some intermediatory representations that are called assembly for some reason.
  - Like PTX (used for CUDA capable GPUs)

ISA Examples

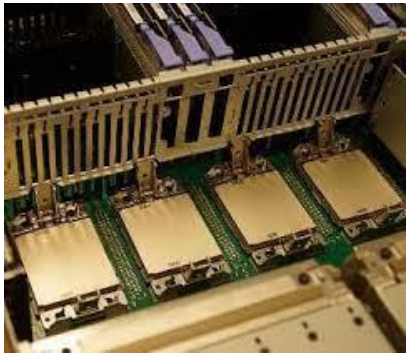


x86



ARM

RISC-V



IBM Z

IBM Power

VM Examples

WEBASSEMBLY

Sample Codes (x86, Linux)

Hello World!

```
.asm
global start

section .data
msg:    db      "Hello, World!",
10
len:    dq      14

section .text
start:
    mov     rax, 0x2000004 ; write
    mov     rdi, 1 ; stdout
    mov     rsi, msg
    mov     rdx, len
    syscall

    mov     rax, 0x2000001 ; exit
    mov     rdi, 0
    syscall
```

Check Prime (C Function) – P.1

```
segment .text
    global is_prime

is_prime:
    mov rcx, 2
    mov rax, rdi
    mov rdx, 0

    cmp rcx, rax
    ja return_false ; if(input < 2)
-> return false
    je return_true ; if (input ==
2) -> return true
    div rcx
    cmp rdx, 0
    je return_false ; if (input % 2
== 0) -> return false
```

Check Prime (C Function) – P.2

```
    mov rax, rdi
    mov rcx, 3
    check_prime_loop:
        cmp rcx, rax
        jae check_prime_loop_end ;
if (rcx(counter) >= rax (rdi(input)
/ c)) -> end loop

        mov rax, rdi
        mov rdx, 0
        div rcx

        cmp rdx, 0
        je return_false ; if
(rdi(input) % c == 0)

        add rcx, 2
        jmp check_prime_loop
    check_prime_loop_end:
```

Check Prime (C Function) – P.3

```
    return_true:
        mov rax, 1
        jmp return
    return_false:
        mov rax, 0
    return:

    ret
```

END OF SLIDES