Cheatsheets / **Computer Architecture: Assembly Language**

Assembly Language

The Compilation Process

The compilation process is the procedure code goes through to go from high-level programming languages into machine code that the hardware understands. Most languages go through some semblance of this four-stage process:

- Preprocessing
- Compiling
- Assembling
- Linking

Stage 1: Preprocessing

Preprocessing is the first step and is used to prepare the user's code for machine code by removing comments, expand included macros, and perform any code maintenance prior to handing the file to the compiler.

Stage 2: Compiling

Compiling is the process of taking the expanded file from the preprocessor and translating the program into the Assembly language that is designated by the ISA. Program optimization is also a key part of this step.

Stage 3: Assembling

Assembling is the process of taking an Assembly language program and using an assembler to generate machine code for use by the computer hardware.

Stage 4: Linking

Linking is the process of filling in function calls, including additional objects, libraries, and source code from other locations into the main binary code so it is ready to be executed by the processor.

Assembly Language

Assembly language is a low-level programming language used to directly correspond with machine code. It begins with an opcode and then references memory locations or data types to operate on.

```
"Hello, World" in x86 Assembly Language:

        .global _start
        .text
_start:
        mov     $1, %rax
        mov     $1, %rdi
        mov     $message, %rsi
        mov     $13, %rdx
        syscall

        mov     $60, %rax
        xor     %rdi, %rdi
        syscall
message:
        .ascii  "Hello, world\n"
```

Print Share