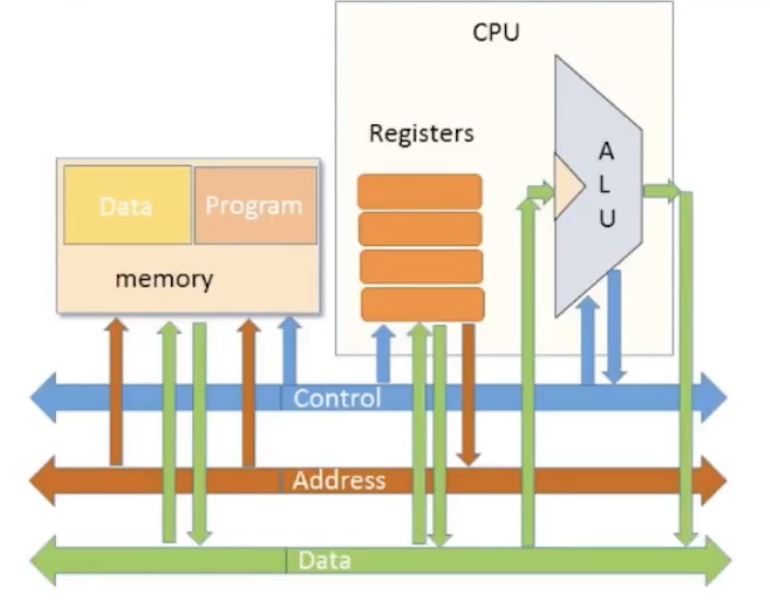
**Module 5: Computer Architecture**

Unit 5.1: Von Neumann Architecture

图示

描述已自动生成



1. Some registers are used to specify addresses.

the way we achieve indirect addressing into a RAM or jump into a ROM address is to **put addresses into a register** and then specifies where we want to access

Unit 5.2: The Fetch-Execute Cycle

1. The basic CPU loop

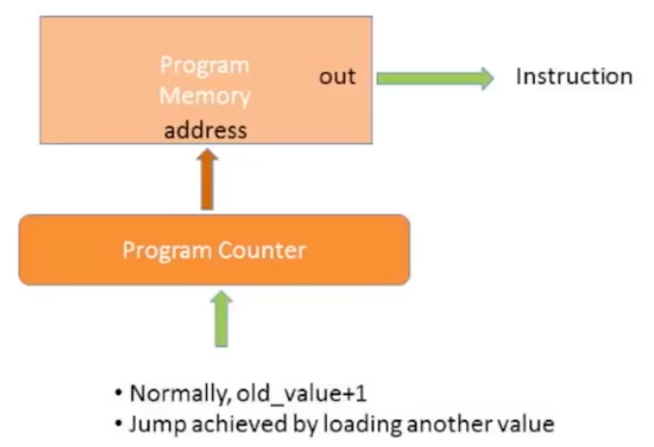
**Fetch** an instruction from the Program memory

**Execute** it

1. **Fetching**

Put the location of the next instruction into the address of the program memory

Get the instruction code itself by reading the memory contents at that location



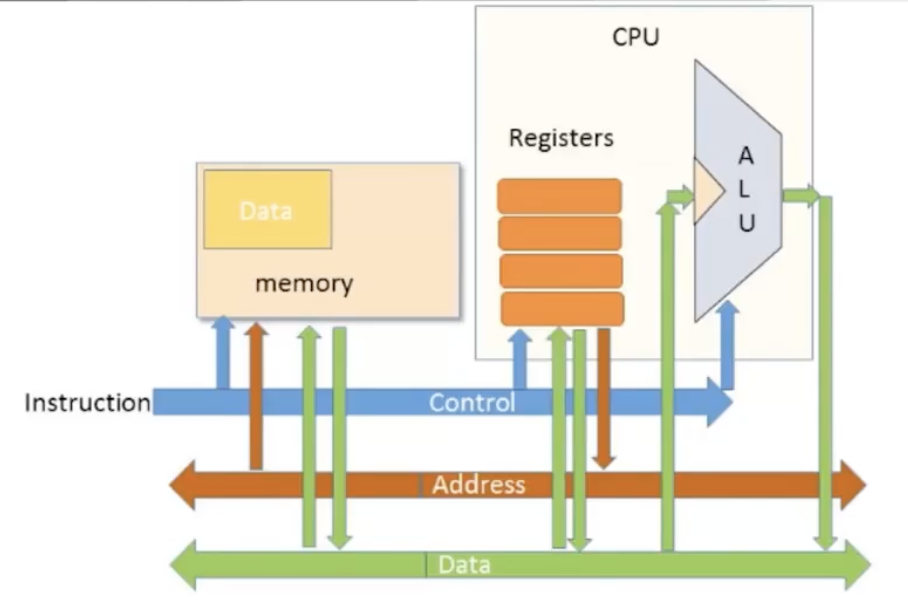
1. **Executing**

The instruction code specifies “what to do”

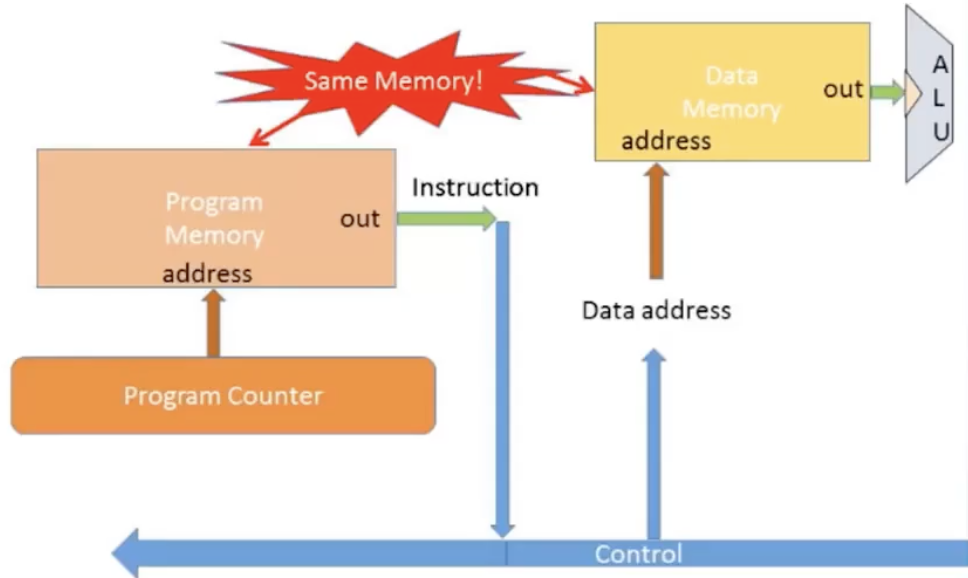
* which arithmetic or logical instruction
* what memory to access
* if/where to jump

Often, different subsets of the bits control different aspects of the operation

Executing the operation involves also accessing registers and/or data memory



1. Fetch-Execute Clash



Solution: Multiplex

图示

描述已自动生成

**W5: Memory**

1. Program Counter
2. Bus structure: used to connect CPU and memory

图示, 示意图

描述已自动生成

1. Fetch-Decode-Execute Cycle

Fetch: The opcode for the instruction is fetched from memory

Decode: Opcode decoded to work what parts of the CPU are needed

Execute: CPU processes the instruction

1. **Fetch-Execute Algorithm**

**文本

描述已自动生成**

1. Fetch

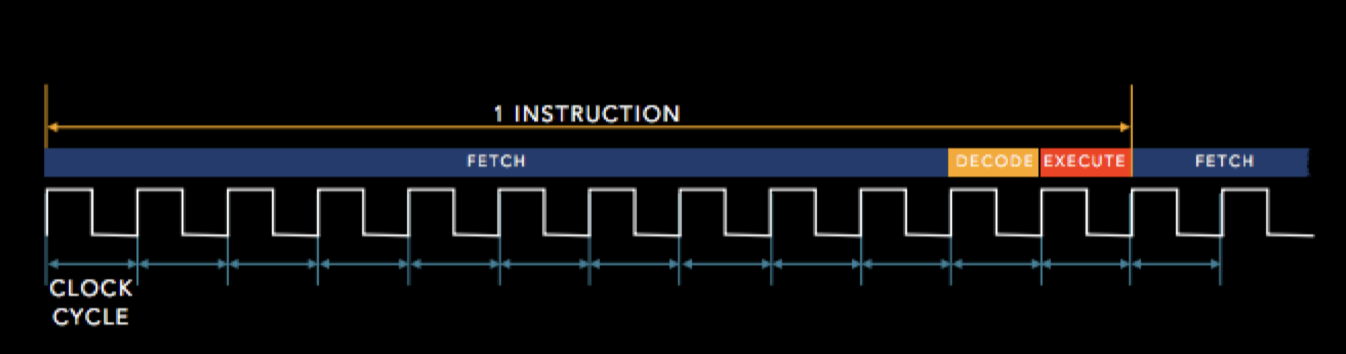
* The address of the instruction to be fetched is defined by the PC
* Memory system places the required value onto the data bus
* The instruction is copied from memory to IR (instruction register)
* Load into instruction cache (a very fast memory)

表格

描述已自动生成

1. Memory latency

Affect both instruction access and data access, thus affects fetch



1. Differences

ROM：断电之后存储的数据不会消失，例如：U盘和固态硬盘，存储空间

RAM：断电之后数据会消失，例如：CPU缓存，电脑的内存

* Dynamic RAM: 例如内存，需要定期充电，速度慢但便宜
* Static RAM：例如CPU缓存，速度更快但更贵

1. We have a small piece of static RAM in between

called a **cache** and it stores a **temporary copy** of the instructions

1. Cache(高速缓存)是Register和Memory之间的桥梁

容量和速度介于Register和Memory之间

Cache is divided into indivisible units called **cache lines**

**Cache hit**: if a piece of data is found in the cache

**Cache miss**: if a piece of data is not found in the cache

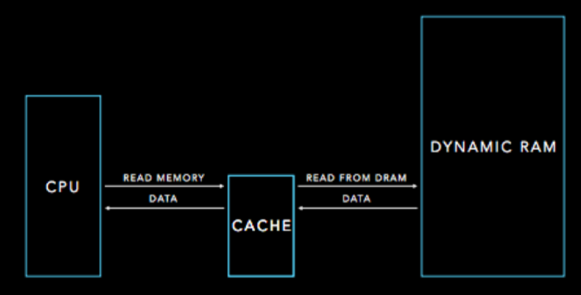
图示

描述已自动生成

1. Cache Basic Principle

cache hit - fast: cache miss - slow:

手机屏幕的截图

描述已自动生成 

如果在cache中没有找到那么需要到内存里去找数据，所以速度会变慢

1. Hit or Miss Measures
2. Can measure the performance of cache

Hit rate: fraction of memory accesses that are cache hits

Miss rate: fraction of memory accesses that are not found in the cache

1. Hit Time: time required to access the cache
2. Miss penalty: time required to fetch a cache line into the cache from memory
3. The Hack Computer

* A 16-bit Von Neumann platform
* The instruction memory and the data memory are physically separated
* Screen:256 rows by 512 columns (row[0] – row[255])
* Keyboard: standard

1. The Hack Computer: Main Parts:

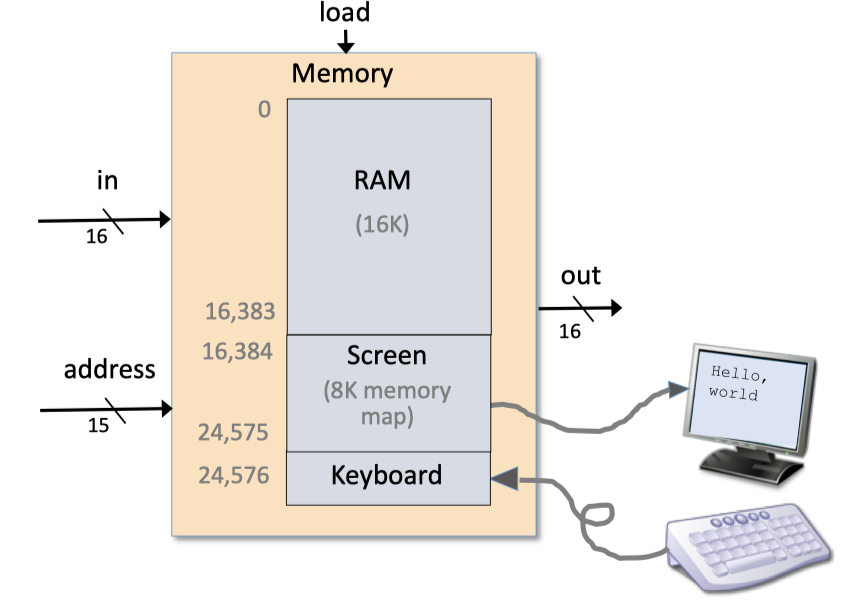
* Instruction memory (ROM)
* Memory: (RAM)

data memory, screen and keyboard

* CPU
* Computer (the logic that holds everything together)

1. Memory implementation

图片包含 图示

描述已自动生成

1. Screen memory map
2. word = screen[32\*row + col/16]

(RAM[16382 + 32\*row + col/16])

1. Set the (col%16)-th bit of word to 1
2. Commit word to the RAM
3. Keyboard

a 16-bit register is used to keep the key stroke

1. The Hack Computer

图示

描述已自动生成

1. Turing Machine

A theoretical machine that has infinite tape separated into squares

Each square contains a symbol or be blank.

The tape may serve as:

1. the input device
2. the memory available for use during the computation
3. the output device

A Turing Machine will have two halt states, one denoting acceptance and the other denoting rejection.

The Halting Problem:

It is impossible to write a program that determine whether another program halts

Turing machines are theoretical concepts of describing computations.

The Von-Neumann architecture is an architecture for constructing actual computers