7.1 Computer Architecture and 7.2 Machine Language & Execution

Notebook:	How Computers Work [CM1030]		
Created:	2019-10-09 10:09 AM	**Updated:**	2019-11-11 7:23 PM
Author:	SUKHJIT MANN		
Tags:	ALU, Bus, Compiler, CU, Data Transfer, Interpreter, IR, Jump, Load, Opcode, Operand, PC...		

Cornell Notes	**Topic:** 7.1 Computer Architecture 7.2 Machine language & Execution	Course: BSc Computer Science Class: How Computer Work [CM1030]-Lecture Date: November 11, 2019

Essential Question:

What are the inner workings of a CPU and how it processes/stores machine readable code?

Questions/Cues:

- How is data moved between Memory and the CPU
- What is a bus?
- What is ALU?
- What is CU?
- What are Registers?
- What is a Cache?
- What is an instruction?
- What is PC?
- What is IR?
- What is Opcode and Operands?
- What are various instructions performable on data?
- What is Conditional Jump?
- What is a compiler versus an interpreter?

Notes

- Data moved between Memory & CPU via bus
- Bus = Set of wiring connect CPU & Memory and allows fast interchange between two of them
- Arithmetic logic unit (ALU) = performs numerical mathematical calculations inside CPU
- Control Unit (CU) = control running of prog
- Registers = Another bit of memory that store data, much tinier than Main Memory, but actually on CPU; accessed very quickly compared to main memory
    - CPU does calculations with data stored on registers
- Load from Memory to Register -> From register, manipulate data using ALU -> ALU writes back to Register --> data written back to memory
- Cache = smaller, but faster set of memory you can access more quickly than main memory, slower than registers
    - located either on CPU chip or very close to it

    - used to store values CPU likely to use
    - Lot more data than registers, act similar to RAM but faster
    - Performance of comp depends on access to memory; how big & fast cache is
- Instruction = for now, code that acts on data and registers
- Code is stored in memory like data, called stored program concept
- Program Counter (PC) = tells CPU where the current instruction is in memory
- Instructions Register (IR) = which actually current instruction is stored
    - Whenever CPU finish current instruction, looks at PC and gets data from that memory location, loads into instruction register, whatever binary pattern is in IR it will intrepreted as instruction and executed. Sometimes called **fetch-execute cycle**.
- Opcode (Operation Code) = start of instruction
    - Opcode like function name in prog lang
- Operands = rest of the instruction, various other data ie. like numbers to add, those numbers are operands
    - Operands lot like arguments of that function, data passing into function
    - Operands not actual values like 312 but number of a register on CPU. So ADD 312 means Add value in register 1 and add value in register 2 and store new value in register 3
- Load instruction = load data from memory into registers
- Data Transfer Instruction = moves data from 1 part of a comp to another
- Store instruction = Store is exactly the same in reverse, grabs something out of register and puts in memory
- Jump instruction = example of a control statement, instead of moving on to next instruction, jump to different instruction somewhere else in memory
    - puts different number in PC
    - takes last 2 parts of operand & copies them into PC
- Conditional Jump = takes value in register given by 1st operand, and compares it to value of register zero, if values same; jump is done, otherwise CPU carries on & moves to next instruction in sequence
- Compiler = takes high-level prog lang & convert directly machine instructions
- Interpreter = instead of converting code all at once, it will convert prog lang like JS into machine instruction as it running or line-by-line.

In this week, we learned about computer architecture, the different units inside of a CPU about the basics of machine readable code and simple instructions that are given to the CPU and how they are performed.