7.1 Computer Architecture and 7.2 Machine Language & Execution-Reading

Notebook:	How Computers Work [CM1030]		
Created:	2019-10-09 10:09 AM	**Updated:**	2019-11-11 7:22 PM
Author:	SUKHJIT MANN		
Tags:	ALU, Bus, CU, Data Transfer, IR, Jump, Load, Opcode, Operand, PC, Store		

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

Essential Question:

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

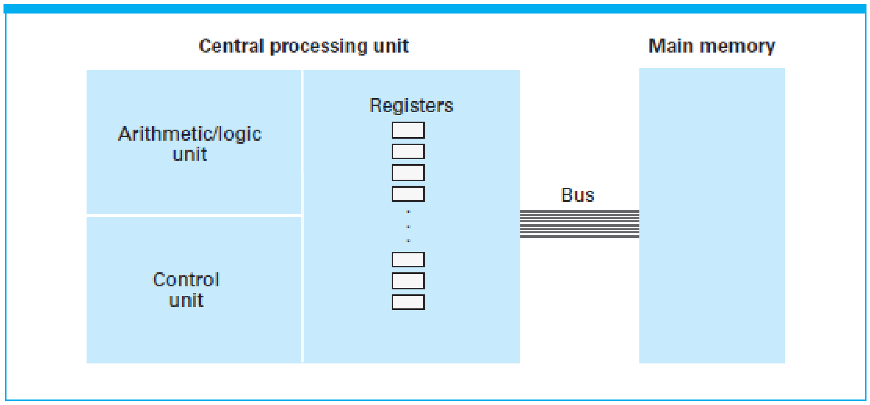
Questions/Cues:

- What controls the manipulation of data?
- What is a Motherboard?
- What is an Arithmetic/Logic Unit?
- What is a Control Unit?
- What is a Register Unit?
- What is a bus?
- What is cache memory?
- What is a stored-program concept?
- What is Machine Language/Machine Instruction?
- What is RISC/CISC?
- How can machine instructions be categorized?
- What is an instruction register?
- What is a program counter?
- What is the machine cycle of a CPU?

Notes

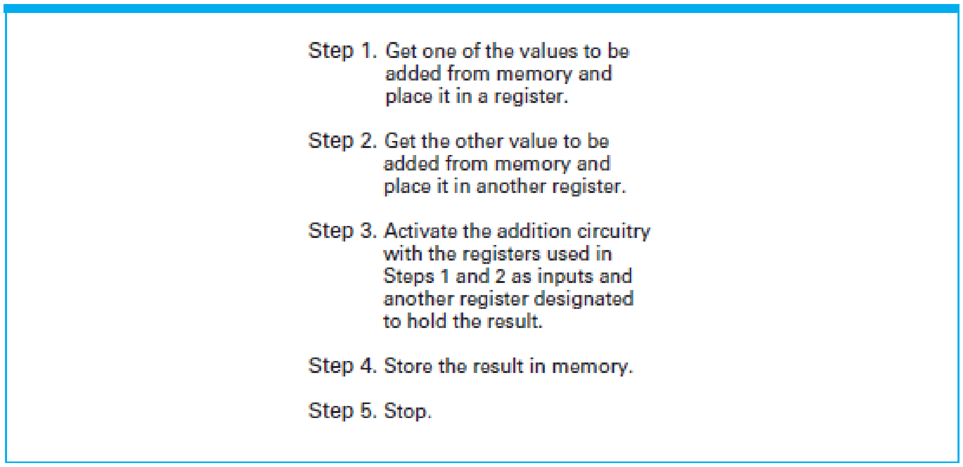
- CPU (Central Processing Unit) = circuitry in a comp that controls the manipulation of data
  - today's desktop comps, CPU packaged as small flat squares (2 x 2 inches)
  - Small size processors in smartphones & mobile internet Devices called microprocessors
- Motherboard = Comp's main circuit board
  - CPU plug into socket on motherboard via connecting pins
- Arithmetic/logic unit = circuitry that performs operations on data (addition & subtraction)
- Control Unit = circuitry for coordinating machine's activities
- Register unit/Registers = contains data storage cells (similar to RAM) called registers; used for temp storage of info within CPU

Figure 2.1 CPU and main memory connected via a bus



  - General-purpose registers = serve as temp holding places for data being manipulated by CPU
    - hold inputs to ALU unit & provide storage space for results
- To perform operation on data in main memory: CU transfers data from memory to GPR -> informs ALU which registers hold the data -> activates appropriate circuitry within ALU -> tells ALU which register should receive result
- Bus = purpose of transferring bit patterns between CPU and main memory
  - Through bus, CPU reads data from main memory by supplying address of pertinent memory cell along with electronic signal telling memory circuitry that it's supposed to retrieve data in indicated cell.
  - CPU writes data in memory by providing address of destination cell & data to be stored together with appropriate electronic signal telling main memory that it's supposed to store data being sent to it

Figure 2.2 Adding values stored in memory



Step 1. Get one of the values to be added from memory and place it in a register.

Step 2. Get the other value to be added from memory and place it in another register.

Step 3. Activate the addition circuitry with the registers used in Steps 1 and 2 as inputs and another register designated to hold the result.

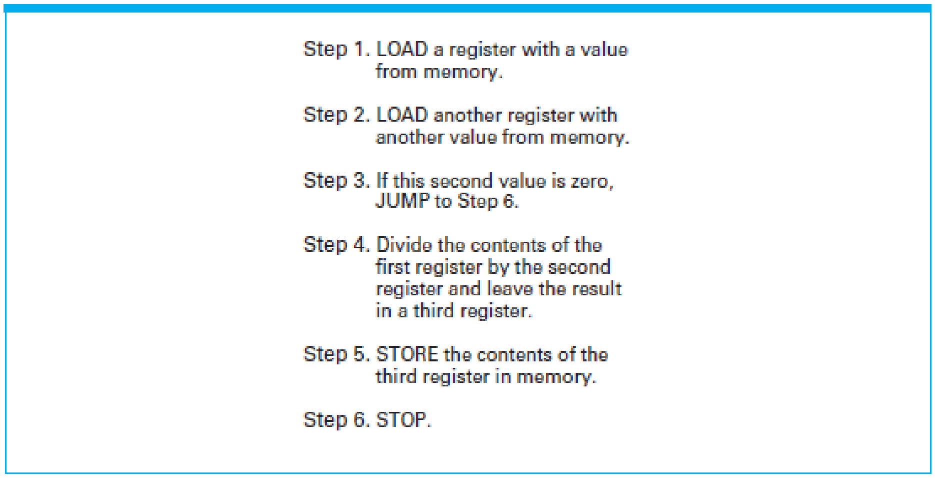
Step 4. Store the result in memory.

Step 5. Stop.

- Cache memory = portion (perhaps several hundred KB) of high-speed memory located within the CPU itself. In this special memory area, comp attempts to keep copy of that portion of main memory that of current interest. Data transfer normally between registers & main memory now reduced to be made between registers & cache memory. Changes made to cache memory then transferred to main memory at more opportune time. Result is CPU that can execute machine cycle more rapidly because not delayed by main memory communication
- Stored-program concept = idea of storing comp's prog in its main memory
- Machine language = collection of instructions along with encoding system encoded as bit patterns
- Machine instruction = instruction expressed in machine lang
- Reduced instruction set computer (RISC) = that a CPU should be designed to execute minimal set of machine instructions
  - machine designed using RISC is efficient, fast & less expensive to manufacture

- Complex instruction set computer (CISC) = CPUs with ability to execute large number of complex instructions; even though many are redundant
  - Argument in favour states more complex CPU better cope with ever-increasing complexities of today's software
- Machine's instructions can be categorized into 3 groupings: (1) data transfer, (2) ALU & (3) Control group
- Data Transfer = group consists of instructions that request movement of data from one location to another
  - rare for data being transferred is erased form its original location; process involved more like copying data rather than moving it
  - LOAD instruction = request to fill GPR with contents of memory cell
  - STORE instruction = request to transfer contents of register to memory cell
  - I/O instructions = commands for communicating with devices outside CPU-main memory context
- Arithmetic/Logic = consists of instructions that tell CU to request activity within ALU. ALU capable of performing operations other than basic arithmetic operations. Additional operations are Boolean operations AND, OR, & XOR. Also includes operations that allow contents of registers to be moved right/left within register known as SHIFT or ROTATE
- Control group = instructions that direct execution of program rather than manipulation of data. Contain family of JUMP (or BRANCH) instructions used to direct CPU to execute instruction other than next one in line. JUMP instructions appear in two ways, unconditional jumps "Skip to step 5" or conditional jump "If value obtained is 0, then skip to step 5". Distinction is conditional jump results in "change of venue" only if certain condition is satisfied.

Figure 2.3 Dividing values stored in memory



Step 1. LOAD a register with a value from memory.

Step 2. LOAD another register with another value from memory.

Step 3. If this second value is zero, JUMP to Step 6.

Step 4. Divide the contents of the first register by the second register and leave the result in a third register.

Step 5. STORE the contents of the third register in memory.

Step 6. STOP.

- Encoded version of machine instruction consists of two parts, opcode and operand field

Figure 2.4 The architecture of the machine described in Appendix C

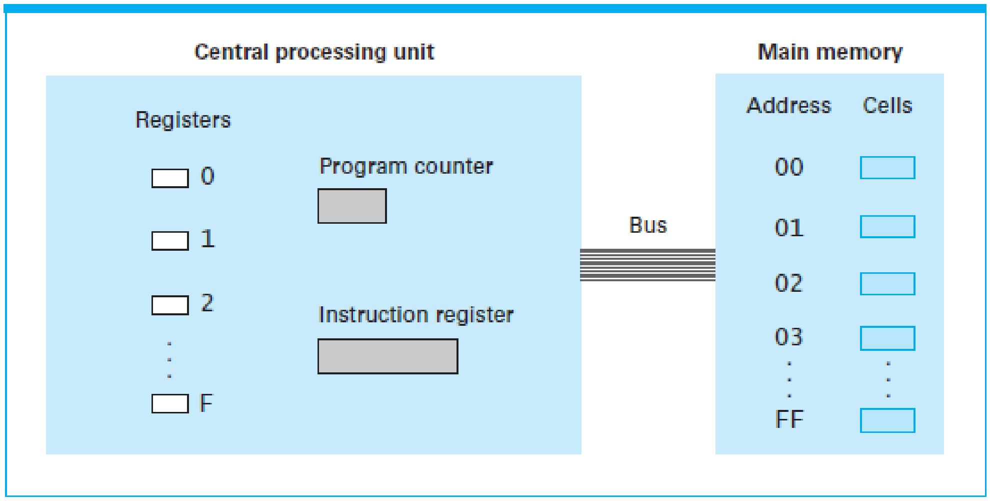


Figure 2.5 The composition of an instruction for the machine in Appendix C

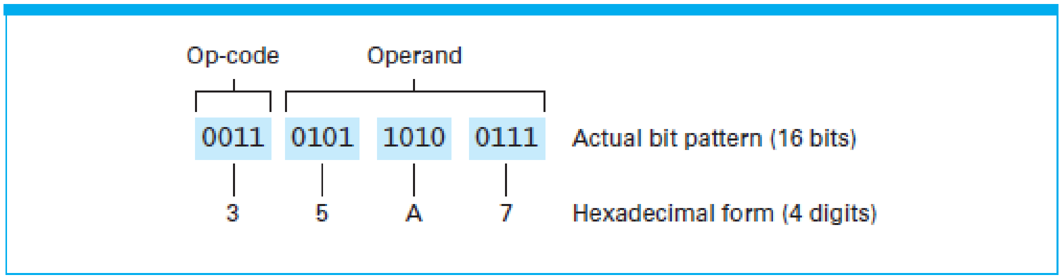


Figure 2.6 Decoding the instruction 35A7

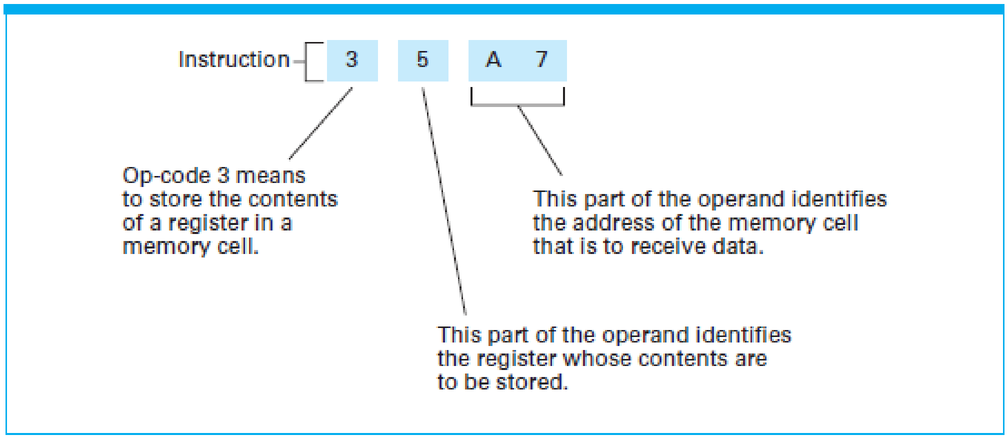
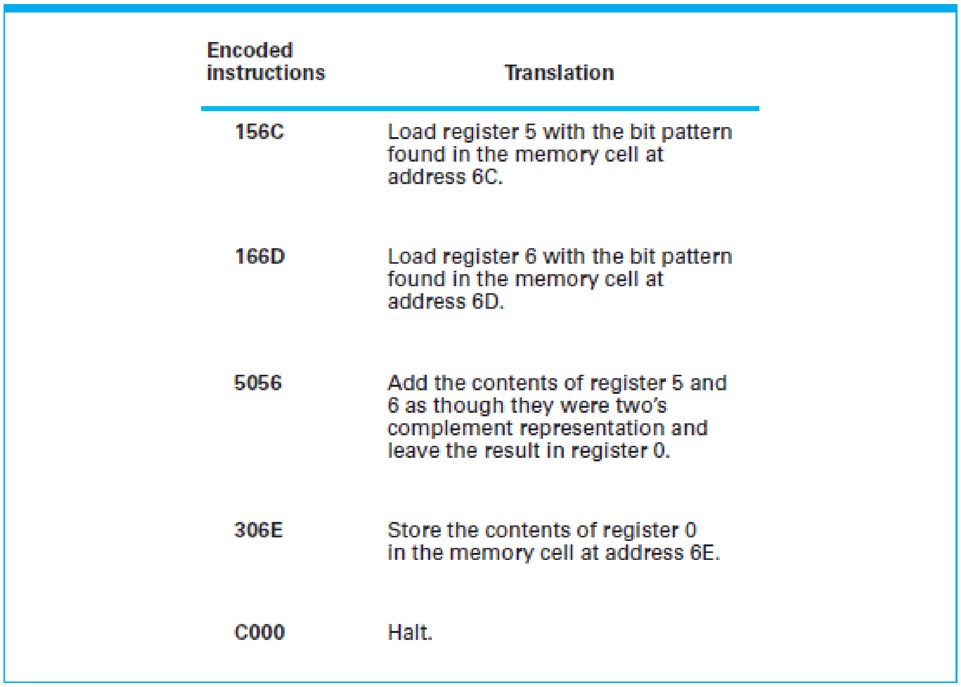

Instruction
3
5
A
7
Op-code 3 means to store the contents of a register in a memory cell.
This part of the operand identifies the address of the memory cell that is to receive data.
This part of the operand identifies the register whose contents are to be stored.

Figure 2.7 An encoded version of the instructions in Figure 2.2



Encoded instructions	Translation
156C	Load register 5 with the bit pattern found in the memory cell at address 6C.
166D	Load register 6 with the bit pattern found in the memory cell at address 6D.
5056	Add the contents of register 5 and 6 as though they were two's complement representation and leave the result in register 0.
306E	Store the contents of register 0 in the memory cell at address 6E.
C000	Halt.

- Instruction register = used to hold instruction being executed
- Program counter = contains address of next instruction to be executed; serving as machine's way of keep track of where it's in prog
- Machine cycle = three-step process guided by algorithm for CPU; repeated continuously
    - Fetch step = CPU request main memory provide with instruction that is stored at address indicated by PC. Fetch involves retrieving content of 2 memory cells from main memory since instruction is 2 bytes long. CPU places instruction received from memory in instruction register & then increments PC by 2 so counter contains address of next instruction stored in memory
    - Decode step = involves breaking operand field into its proper components based on instruction's opcode.
    - Execute step = CPU executes instruction by activating appropriate circuitry to perform requested task. Once executed finished, CPU begins machine cycle fetch step again.

Figure 2.8 The machine cycle

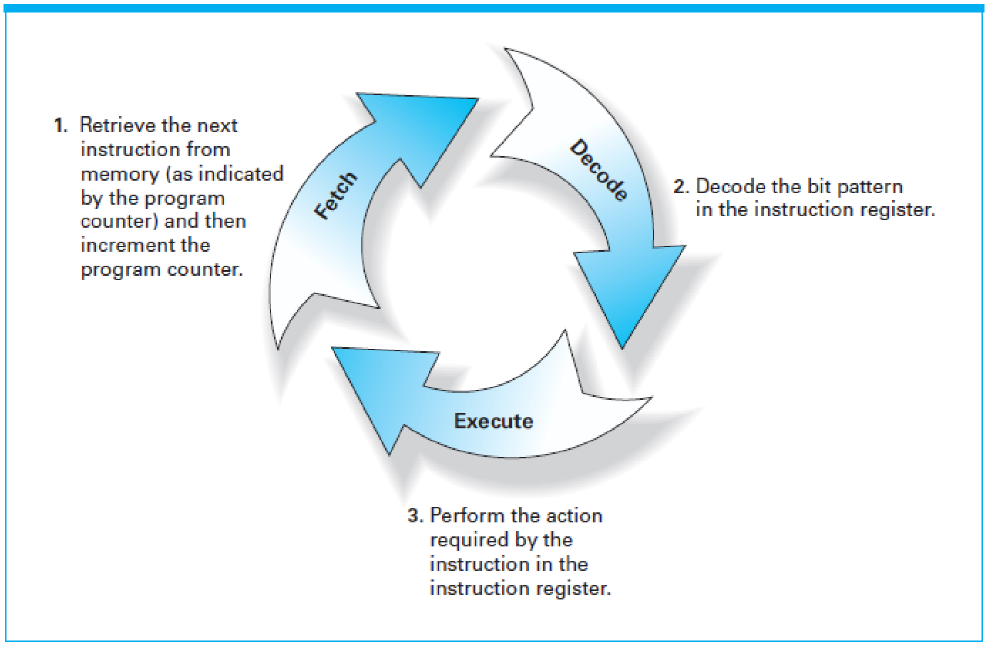

1. Retrieve the next instruction from memory (as indicated by the program counter) and then increment the program counter.
Fetch
Decode
2. Decode the bit pattern in the instruction register.
Execute
3. Perform the action required by the instruction in the instruction register.

Figure 2.9 Decoding the instruction B258

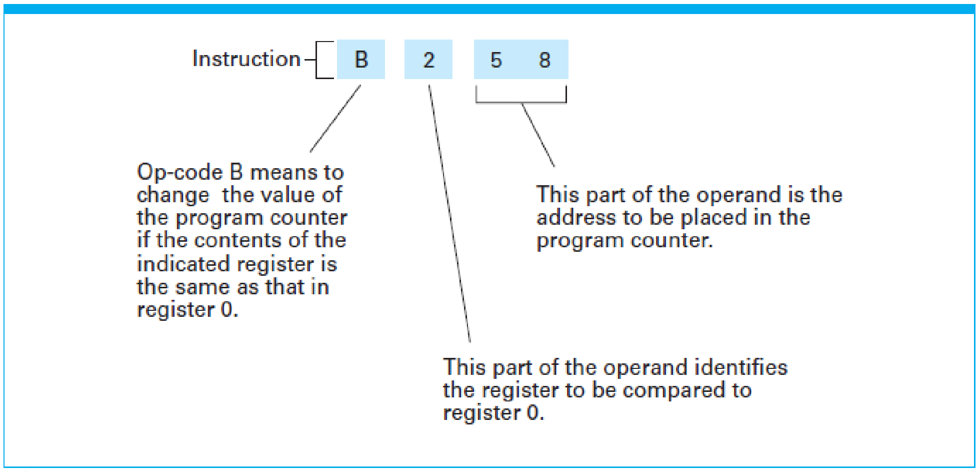

Instruction
B
2
5 8
Op-code B means to change the value of the program counter if the contents of the indicated register is the same as that in register 0.
This part of the operand is the address to be placed in the program counter.
This part of the operand identifies the register to be compared to register 0.

Figure 2.10 The program from Figure 2.7 stored in main memory ready for execution

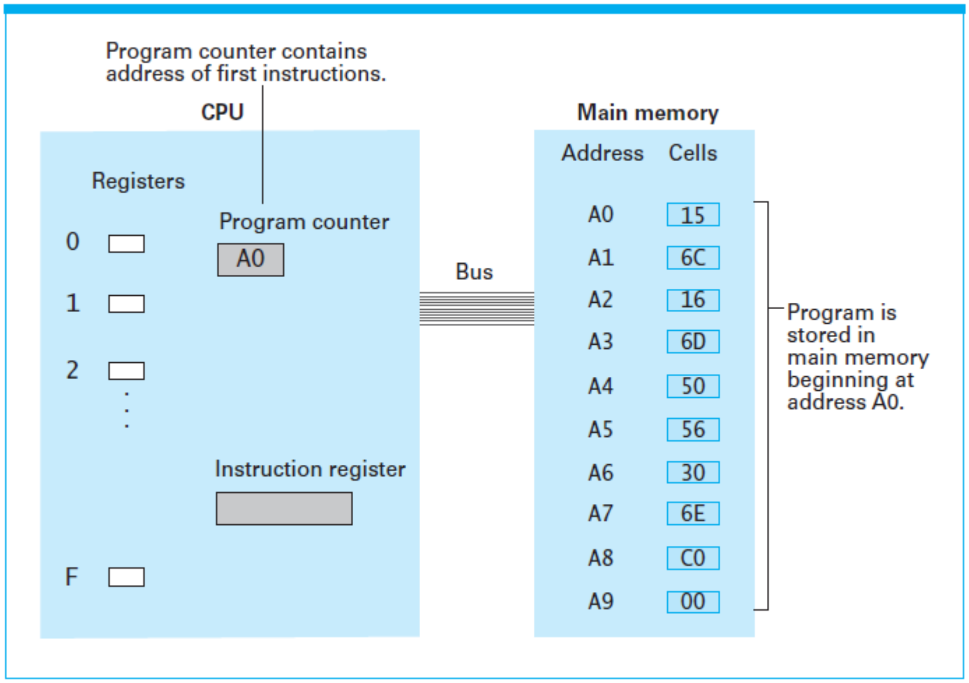
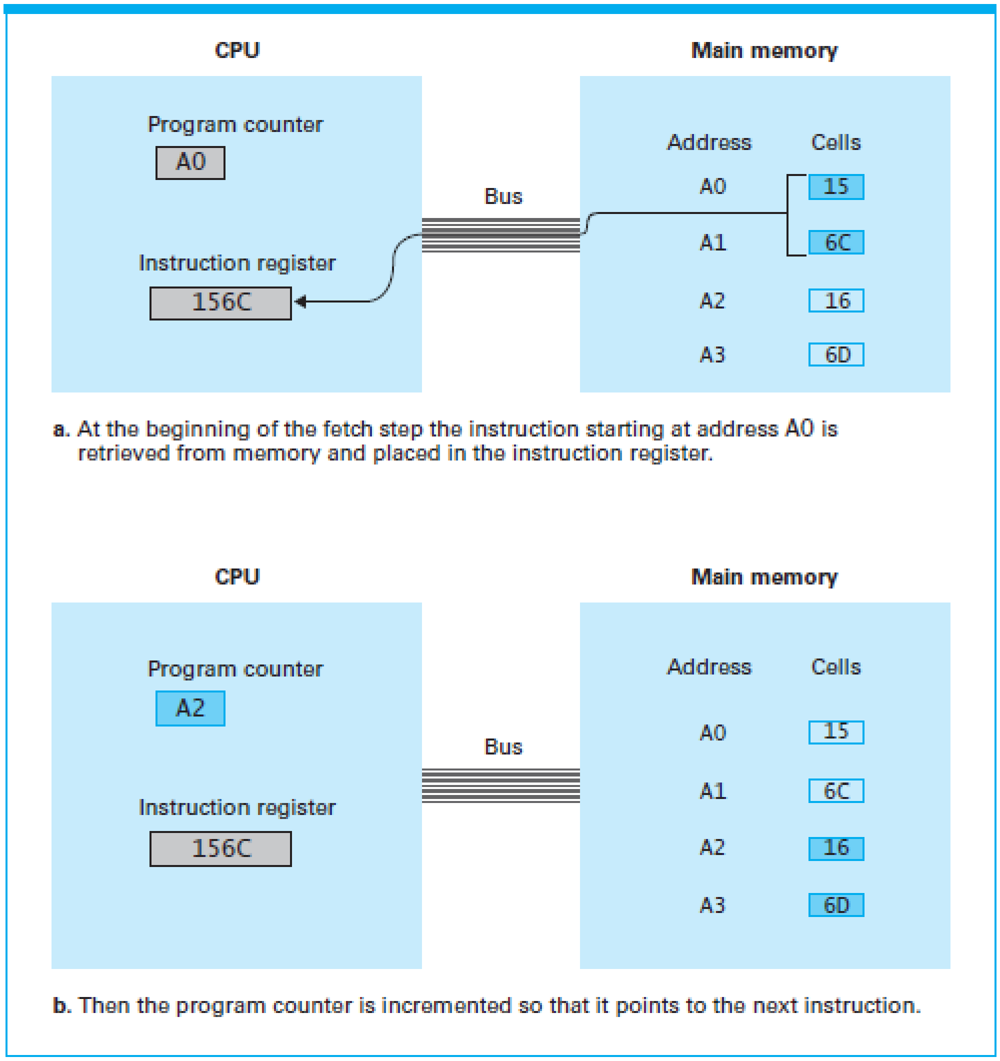


Figure 2.11 Performing the fetch step of the machine cycle



a. At the beginning of the fetch step the instruction starting at address A0 is retrieved from memory and placed in the instruction register.

b. Then the program counter is incremented so that it points to the next instruction.

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.