Laboratório de Informática

Leopoldo Teixeira
leo@leopoldomt.com

Até agora vimos…

- A história dos computadores
- Como informação é armazenada…
  - memória, armazenamento em massa
- …e representada
  - sistemas binários, hexadecimais, complemento de dois, notação de excesso, ponto flutuante

O que fazer com a informação armazenada?

CPU

- *Central Processing Unit*
- Primeiros computadores: Unidades enormes…
- Atualmente: pequenos quadrados que encaixam na placa mãe
    - Mobile Internet Devices: microprocessadores

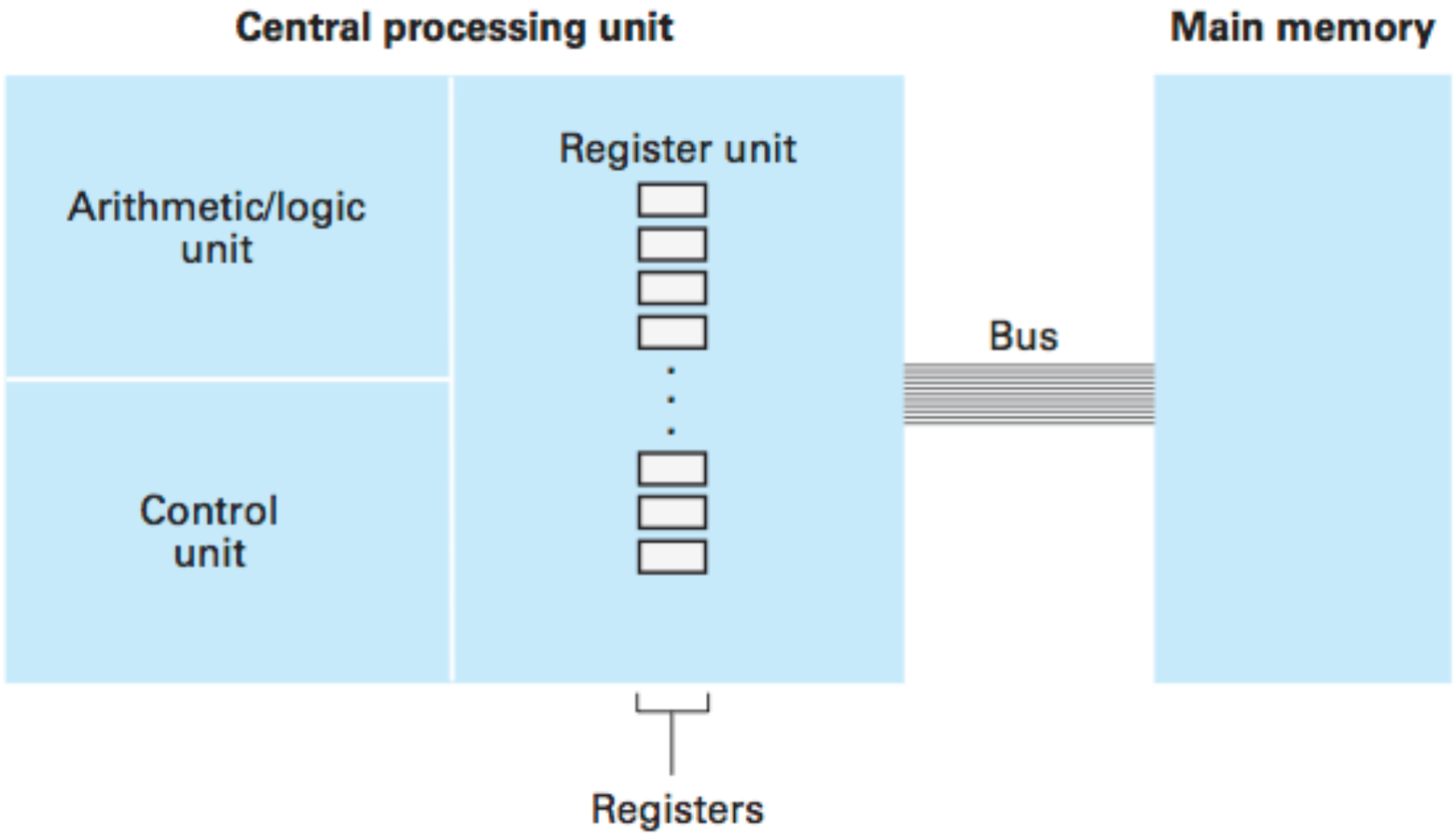
Partes de uma CPU

- Unidade de aritmética e lógica
    - circuitos que executam operações nos dados (adição, subtração)
- Unidade de controle
    - coordenação das atividades da máquina
- Unidade de Registradores
    - células de armazenamento de memória
    - propósito geral vs. propósito específico

Registradores e Via

- Registradores de propósito geral
  - funcionam como posições temporárias de armazenamento para dados que estão sendo manipulados pela CPU
- Via (barramento | *bus*)
  - conjunto de fios que conecta a CPU com a memória principal

Arquitetura CPU



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.

e a memória cache?

Primeiros computadores

- Pouca flexibilidade
  - Passos a ser executados por um dispositivo eram embutidos na unidade de controle, fazendo parte da máquina
  - Para ganhar maior flexibilidade, as máquinas eram projetadas de forma a poder ser reconfiguradas
    - reposicionando cabos
  - Dados e programas vistos como entidades diferentes

Programa Armazenado

- Um programa, da mesma forma que os dados, pode ser codificado e armazenado na memória principal.

- Unidade de controle projetada com a capacidade de extrair o programa da memória, decodificar as instruções, e executá-las.

- Um programa de computador pode ser modificado, simplesmente alterando o conteúdo da memória.

  - (J.P. Eckert, da *Moore School of Electrical Eng.*, *University of Pennsylvania*. John von Neumann foi o primeiro a publicar um trabalho sobre a idéia.)

o que seria necessário para aplicar o conceito de programa armazenado?

Linguagem de Máquina

- Coleção de instruções (padrão de bits), juntamente com o sistema de codificação
- Instrução de máquina
  - uma instrução expressa nessa linguagem
    (instrução em nível de máquina)

RISC

- A lista de instruções de máquina que uma UCP comum deve decodificar e executar é bem pequena.

- Uma máquina pode executar certas tarefas elementares. Ao adicionar novos recursos a ela, isso não aumenta as suas capacidades teóricas.

- Uma abordagem em arquitetura da CPU é chamada computador com conjunto mínimo de instruções (*reduced instruction set computer* – RISC).

    - Essas máquinas são eficientes e rápidas, ex.: série PowerPC, desenvolvida pela Apple, IBM e Motorola.

CISC

- Outra arquitetura é a de computador com conjunto de instruções complexas (*complex instruction set computer*)

- A CPU mais complexa é mais fácil de programar. Qual a razão?

  - ex: série Pentium da Intel.

Instruções de Máquina

- O repertório de instruções de uma máquina pode ser classificado em três categorias:

Instruções de Máquina

- O repertório de instruções de uma máquina pode ser classificado em três categorias:
  - (1) grupo de transferência de dados

Instruções de Máquina

- O repertório de instruções de uma máquina pode ser classificado em três categorias:
  - (1) grupo de transferência de dados
  - (2) grupo aritmético/lógico

Instruções de Máquina

- O repertório de instruções de uma máquina pode ser classificado em três categorias:
  - (1) grupo de transferência de dados
  - (2) grupo aritmético/lógico
  - (3) grupo de controle

Exemplo de Programa

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

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

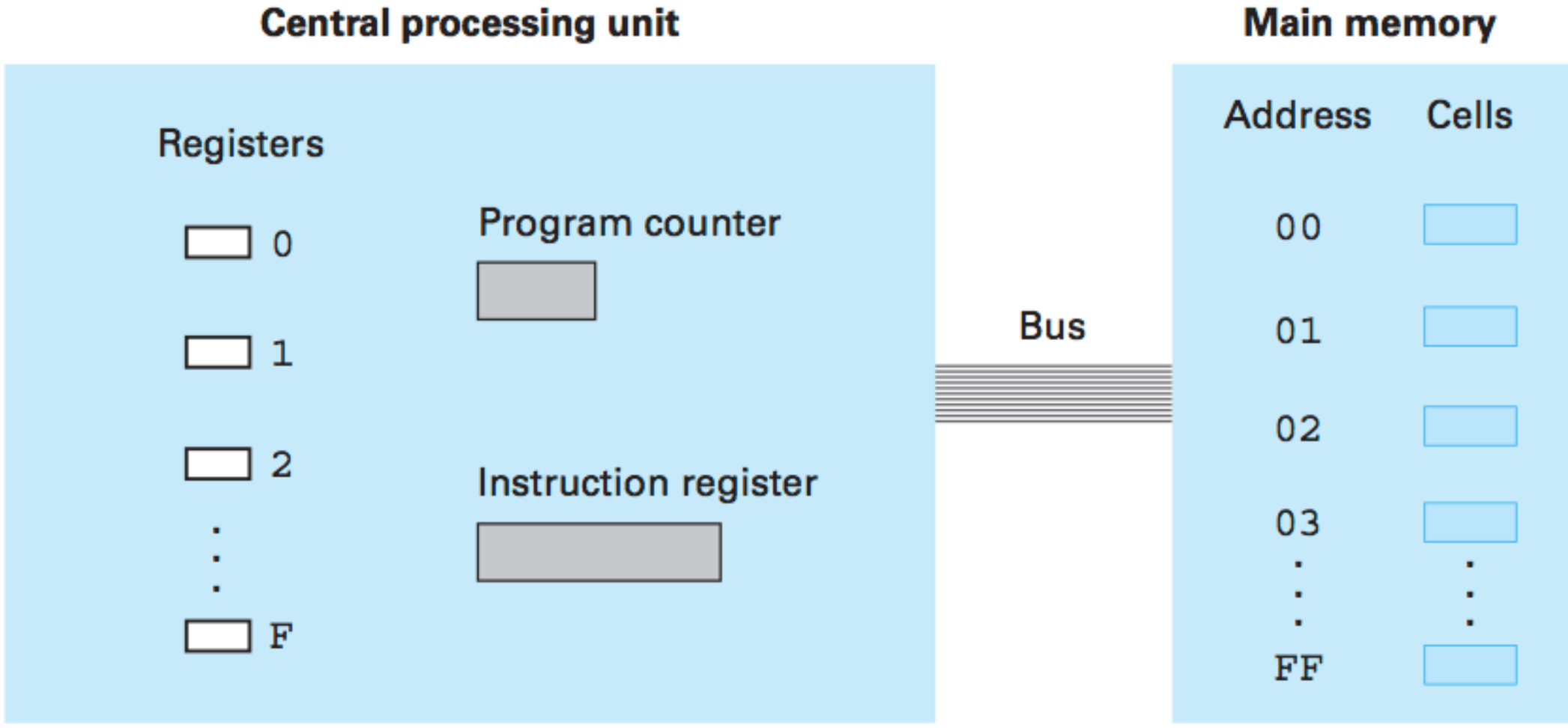
Step 3. If this second value is zero, JUMP to Step 6.

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

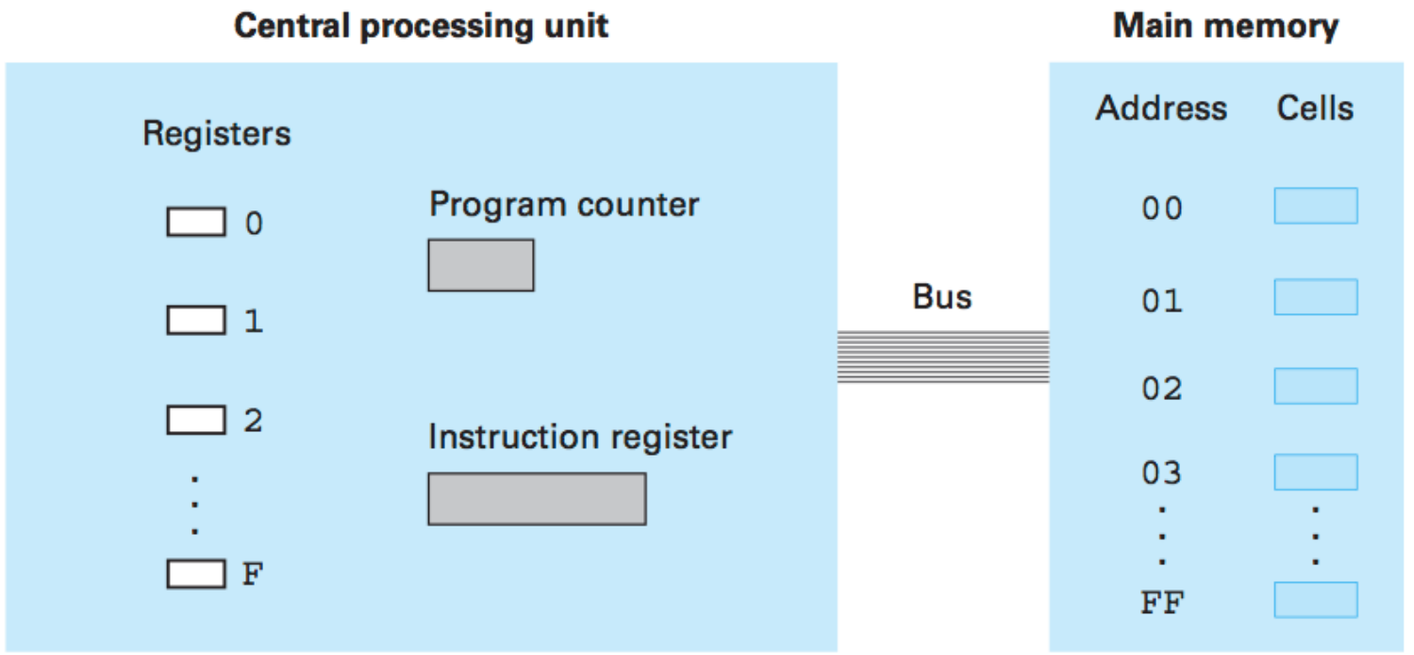
Step 5. STORE the contents of the third register in memory.

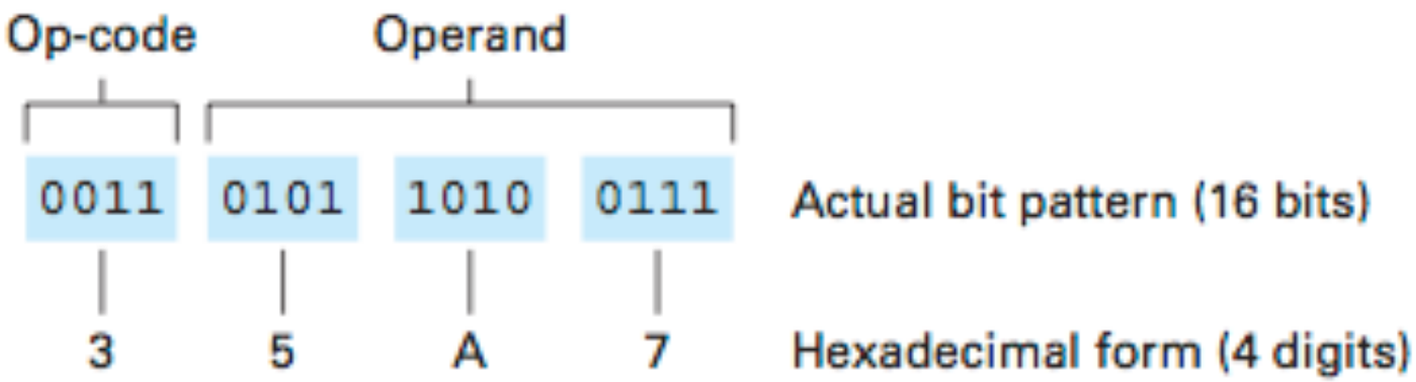
Step 6. STOP.

Arquitetura de CPU



Instruções de Máquina codificadas



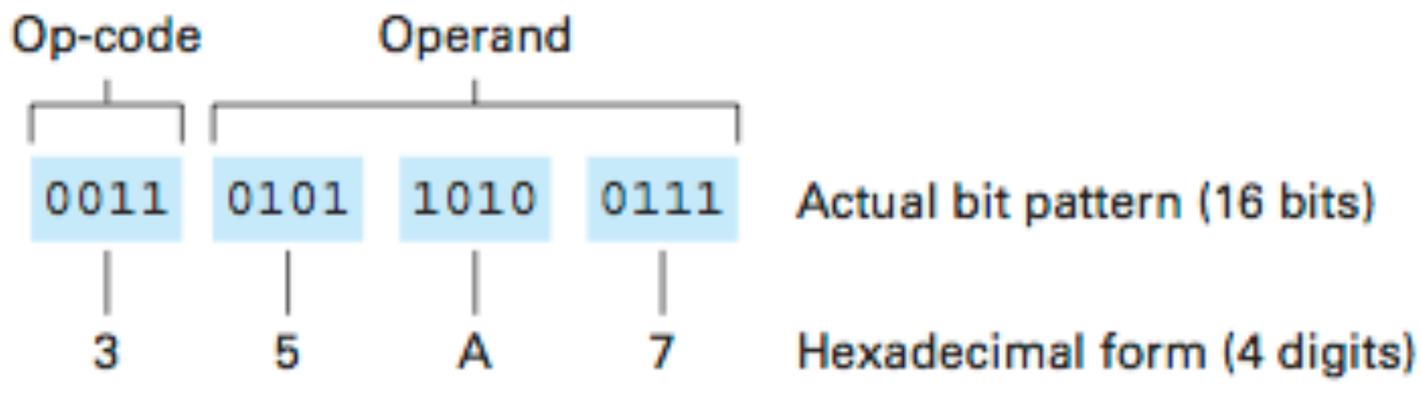


quantas instruções serão necessárias?

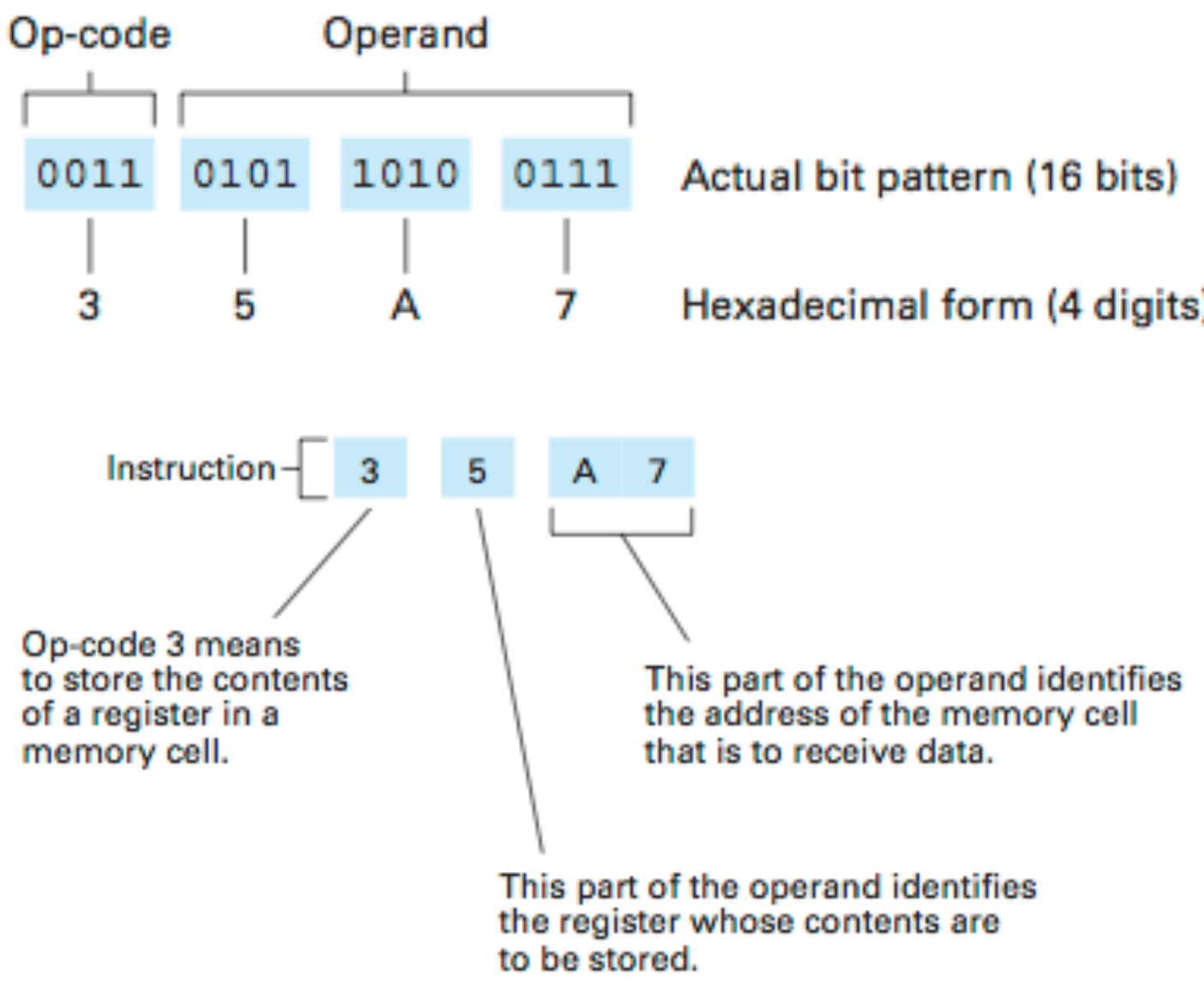
Linguagem de Máquina

Op-code	Operand	Description
1	RXY	LOAD the register R with the bit pattern found in the memory cell whose address is XY. *Example:* 14A3 would cause the contents of the memory cell located at address A3 to be placed in register 4.
2	RXY	LOAD the register R with the bit pattern XY. *Example:* 20A3 would cause the value A3 to be placed in register 0.
3	RXY	STORE the bit pattern found in register R in the memory cell whose address is XY. *Example:* 35B1 would cause the contents of register 5 to be placed in the memory cell whose address is B1.
4	0RS	MOVE the bit pattern found in register R to register S. *Example:* 40A4 would cause the contents of register A to be copied into register 4.
5	RST	ADD the bit patterns in registers S and T as though they were two's complement representations and leave the result in register R. *Example:* 5726 would cause the binary values in registers 2 and 6 to be added and the sum placed in register 7.
6	RST	ADD the bit patterns in registers S and T as though they represented values in floating-point notation and leave the floating-point result in register R. *Example:* 634E would cause the values in registers 4 and E to be added as floating-point values and the result to be placed in register 3.
7	RST	OR the bit patterns in registers S and T and place the result in register R. *Example:* 7CB4 would cause the result of ORing the contents of registers B and 4 to be placed in register C.
8	RST	AND the bit patterns in registers S and T and place the result in register R. *Example:* 8045 would cause the result of ANDing the contents of registers 4 and 5 to be placed in register 0.
9	RST	EXCLUSIVE OR the bit patterns in registers S and T and place the result in register R. *Example:* 95F3 would cause the result of EXCLUSIVE ORing the contents of registers F and 3 to be placed in register 5.
A	R0X	ROTATE the bit pattern in register R one bit to the right X times. Each time place the bit that started at the low-order end at the high-order end. *Example:* A403 would cause the contents of register 4 to be rotated 3 bits to the right in a circular fashion.
B	RXY	JUMP to the instruction located in the memory cell at address XY if the bit pattern in register R is equal to the bit pattern in register number 0. Otherwise, continue with the normal sequence of execution. (The jump is implemented by copying XY into the program counter during the execute phase.) *Example:* B43C would first compare the contents of register 4 with the contents of register 0. If the two were equal, the pattern 3C would be placed in the program counter so that the next instruction executed would be the one located at that memory address. Otherwise, nothing would be done and program execution would continue in its normal sequence.
C	000	HALT execution. *Example:* C000 would cause program execution to stop.

Decodificando Instrução



Decodificando Instrução



Adição em Linguagem de Máquina

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.

Execução de programas

- Um computador efetua a execução de um programa armazenado em sua memória copiando as instruções da memória para a unidade de controle, onde cada instrução é decodificada e executada.

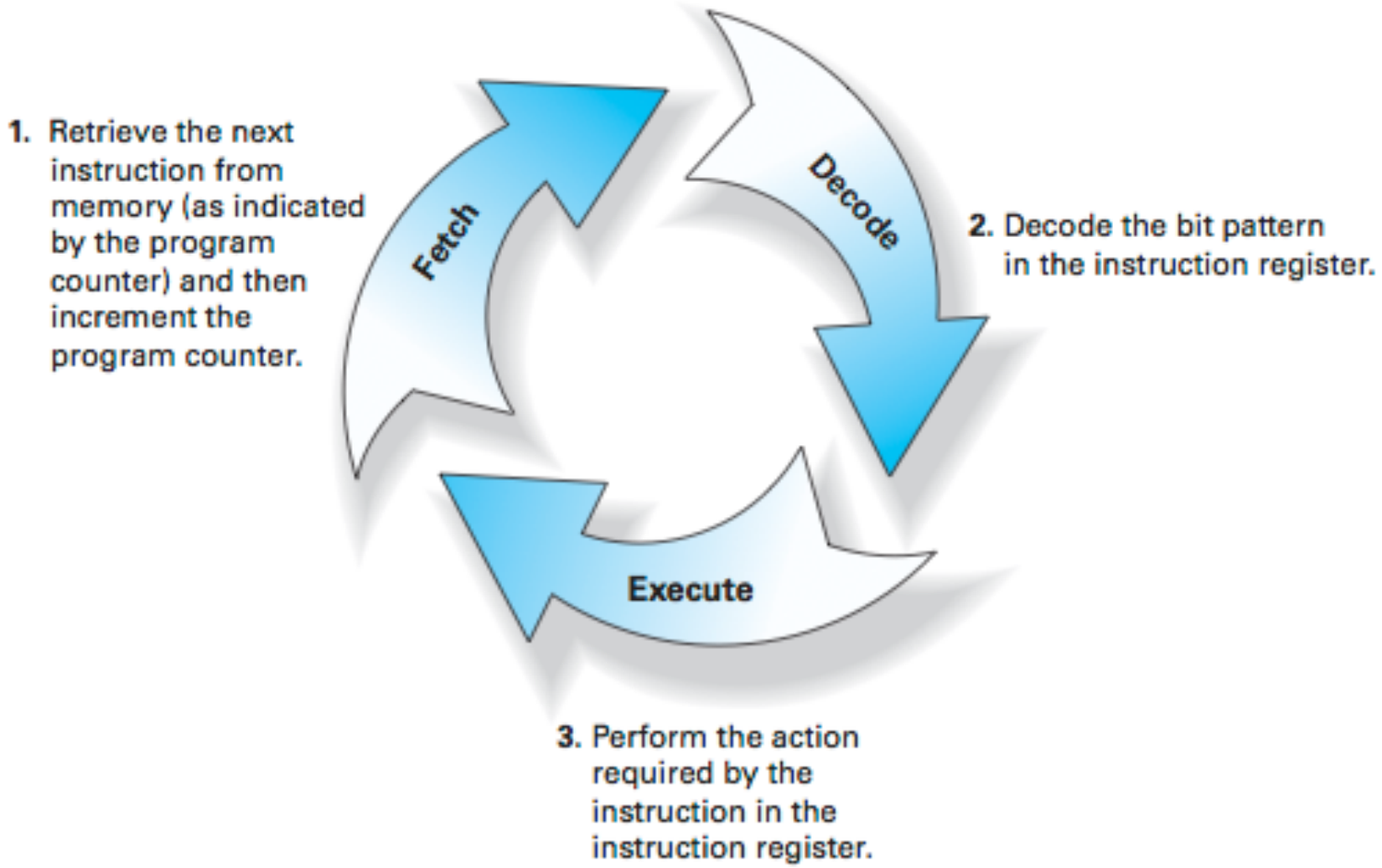
- A ordem que em que as instruções são trazidas da memória corresponde à ordem na qual elas estão armazenadas na memória.

- Registrador de propósito específico:

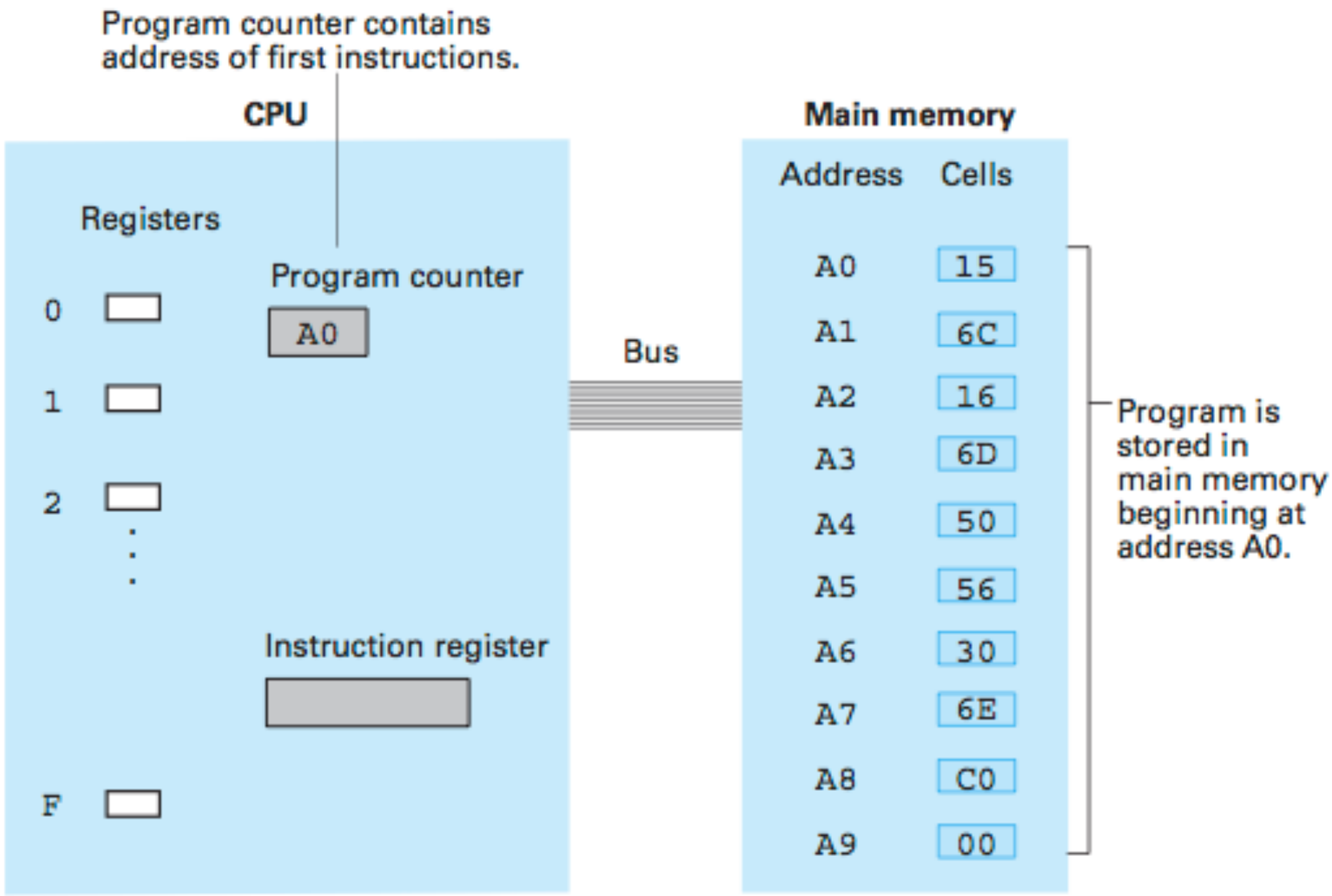
  - Contador de instruções – contém o endereço da próxima instrução a ser executada.

  - Registrador de instruções – usado para manter a instrução que estiver sendo executada.

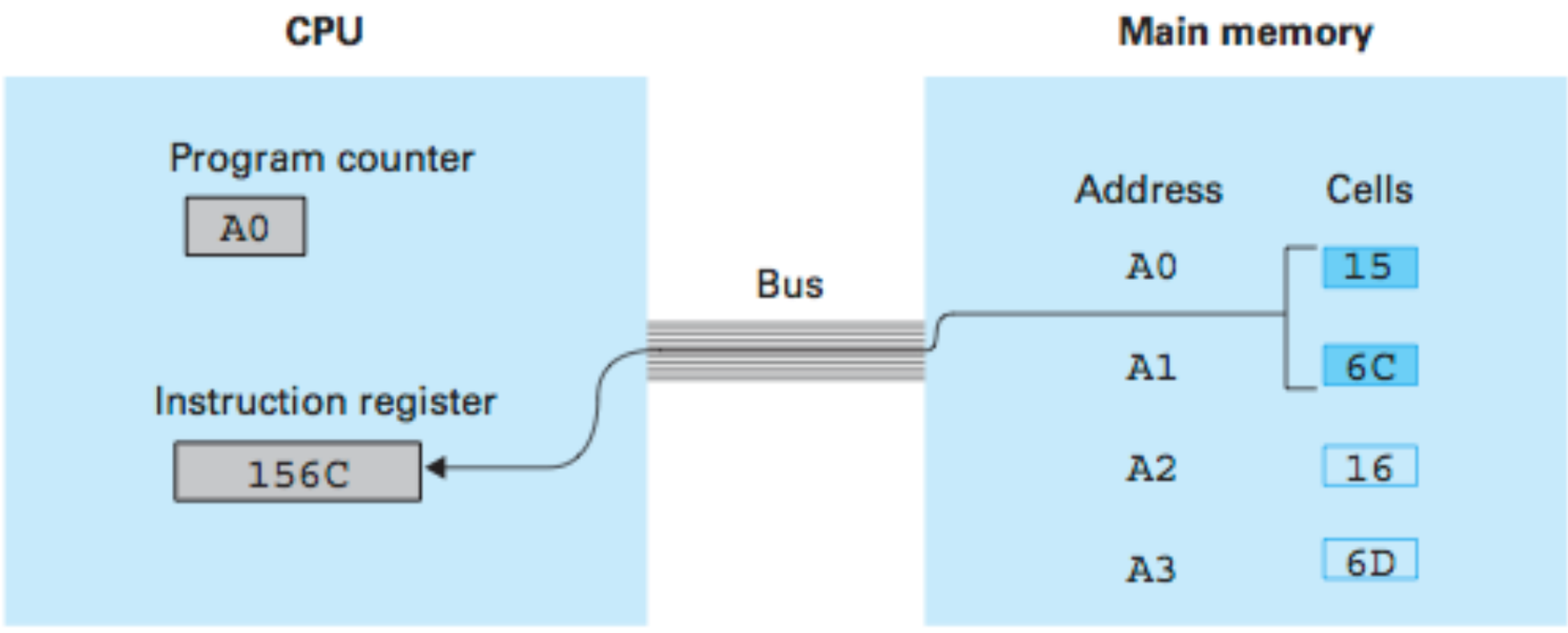
Ciclo da máquina



Programa Armazenado pronto para execução

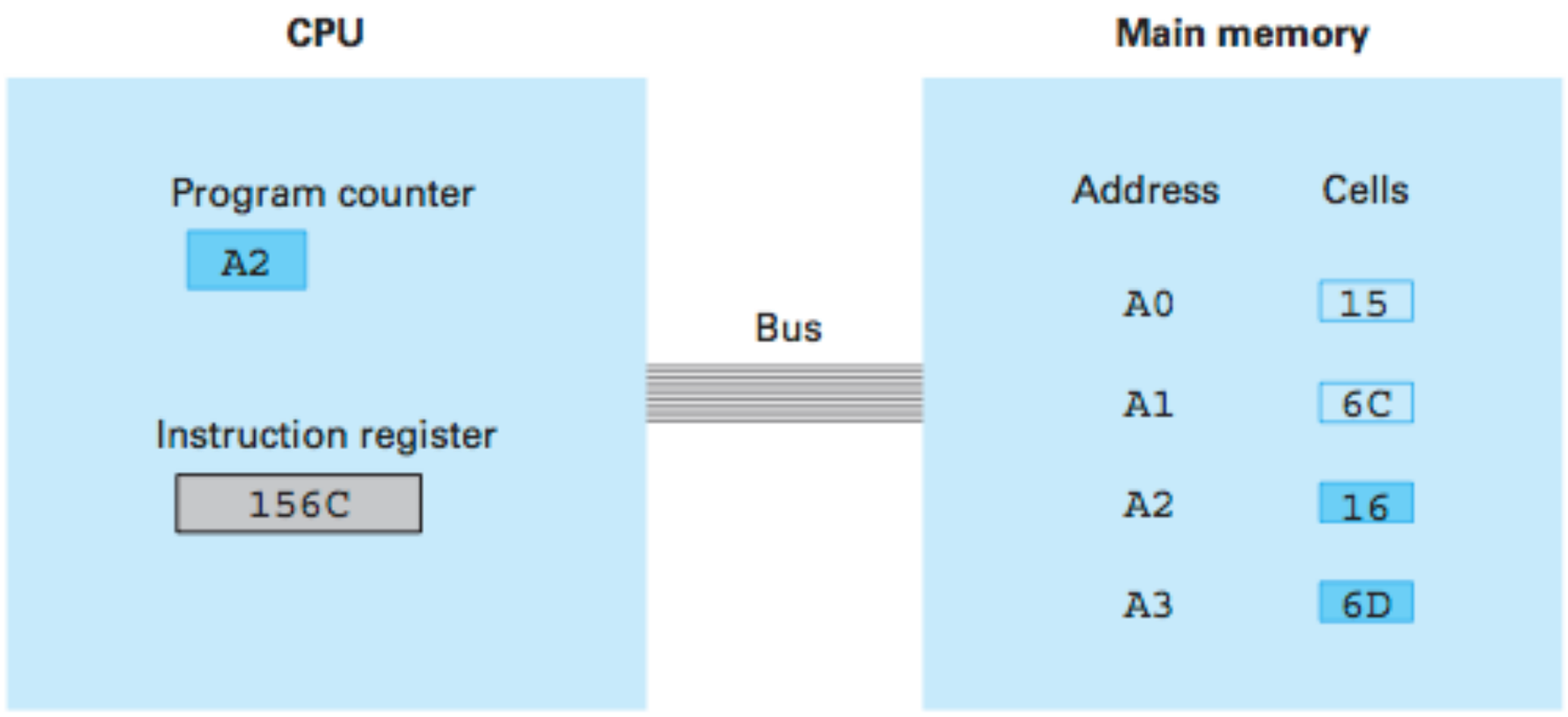


Programa Armazenado pronto para execução



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

Programa Armazenado pronto para execução



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

Executar programas

- A execução de um programa consiste em seguir uma lista detalhada de instruções

- A CPU marca “onde está” na execução com o contador de programas

Programas vs. Dados

- Muitos programas podem ser armazenados na memória principal de um computador

- Muitos dados podem ser armazenados na memória principal de um computador

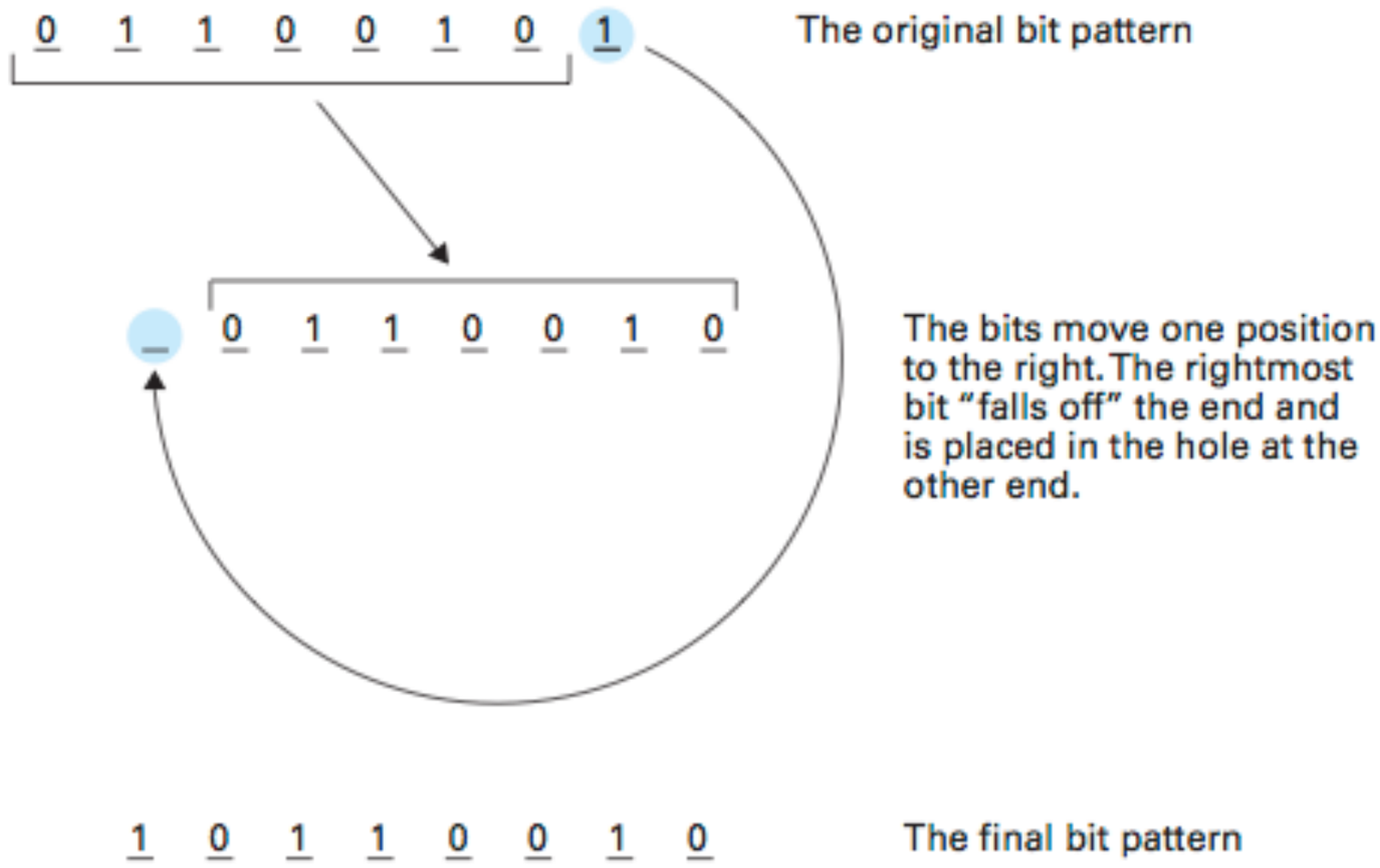
- Como a máquina diferencia programas de dados?

Instruções Aritméticas e Lógicas

- AND, OR, XOR, Rotation e Shift, Add, Subtract…

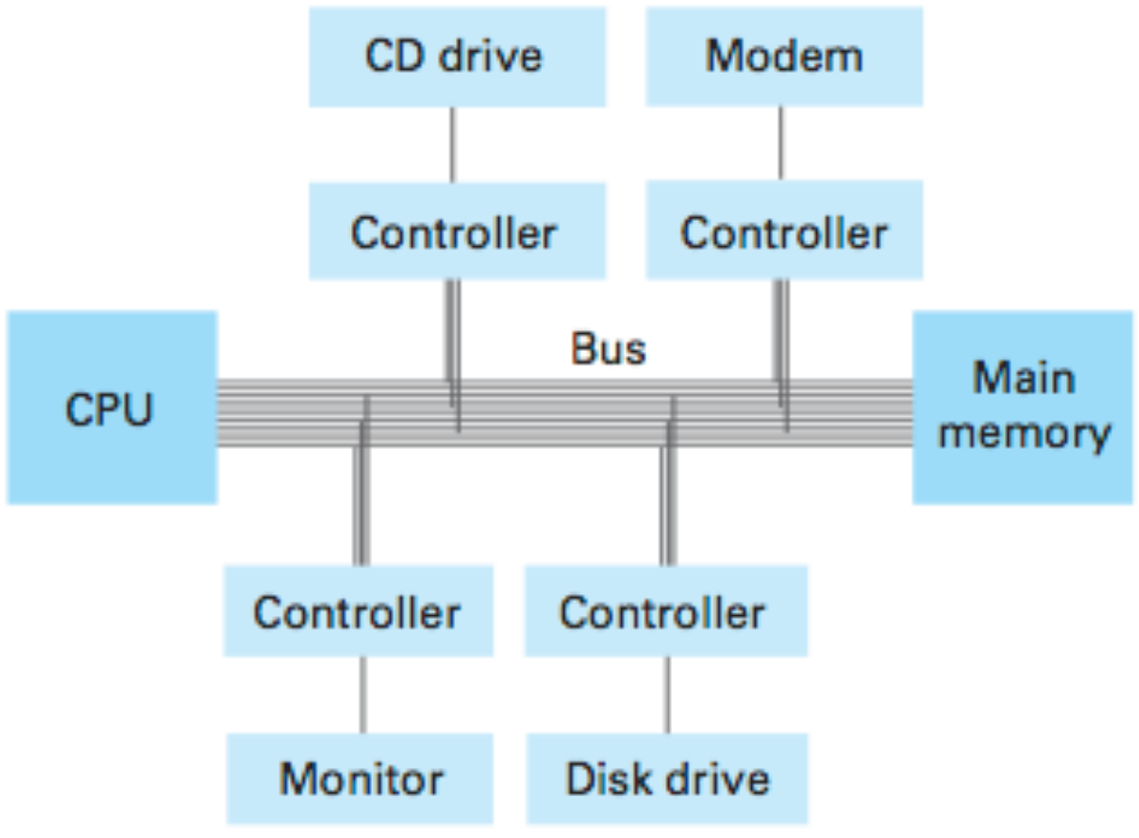
Instruções Aritméticas e Lógicas

- AND, OR, XOR, Rotation e Shift, Add, Subtract…

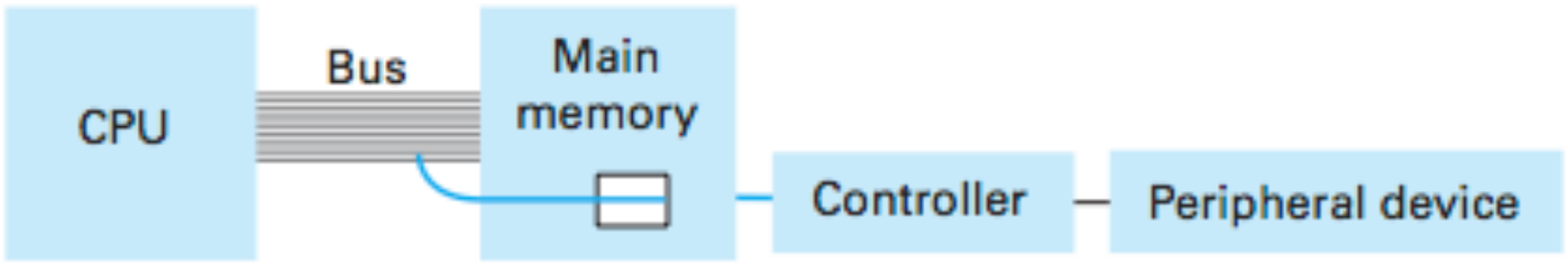


Como a CPU se comunica com os demais dispositivos de um computador?

Comunicação com outros dispositivos



Direct Memory Access (DMA)



Handshaking

- Transferências de dados quase sempre são bilaterais
- O diálogo entre dispositivos é mediado por protocolos
  - trocar informações sobre o status
  - coordenar atividades
- Geralmente envolvem uma palavra de status
  - Impressora, quais são possíveis status?

Meios de Comunicação

- Comunicação paralela

Meios de Comunicação

- Comunicação paralela
- Comunicação serial

Meios de Comunicação

- Comunicação paralela
- Comunicação serial
- Comunicação entre dispositivos distantes
  - modem

Meios de Comunicação

- Comunicação paralela
- Comunicação serial
- Comunicação entre dispositivos distantes
  - modem
  - DSL

Arquiteturas

- Pipelining
- Processamento paralelo
  - MIMD (*multiple-instruction stream, multiple-data stream*)
  - SISD (*single-instruction stream, single-data stream*)
  - SIMD (*single-instruction stream, multiple-data stream*)