Organização e Arquitetura de Computadores

Tradução e edição: António Esteves

SC :: LCC :: Universidade do Minho

Fevereiro 2019

Sumário

- Estrutura da máquina de von Neumann (ou IAS):
  - Formatos da memória
  - Estrutura do IAS mais detalhada
  - Registos
  - Fluxograma da execução de algumas operações
  - Conjunto de instruções
- Componentes dum computador
  - Abordagens da programação: em hardware vs em software
  - Principais componentes
  - Principais registos do CPU
  - Ciclo de uma instrução: ciclo de procura e ciclo de execução
  - Principais categorias de ações dum CPU
  - Máquina hipotética similar ao IAS
  - Execução de instruções nesta máquina hipotética

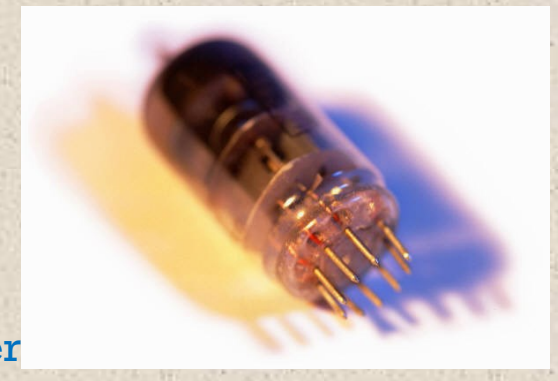
Bibliografia

- COA, 9ª edição, capítulo 2, páginas 16-23
- COA, 9ª edição, capítulo 3, páginas 66-72

História dos Computadores

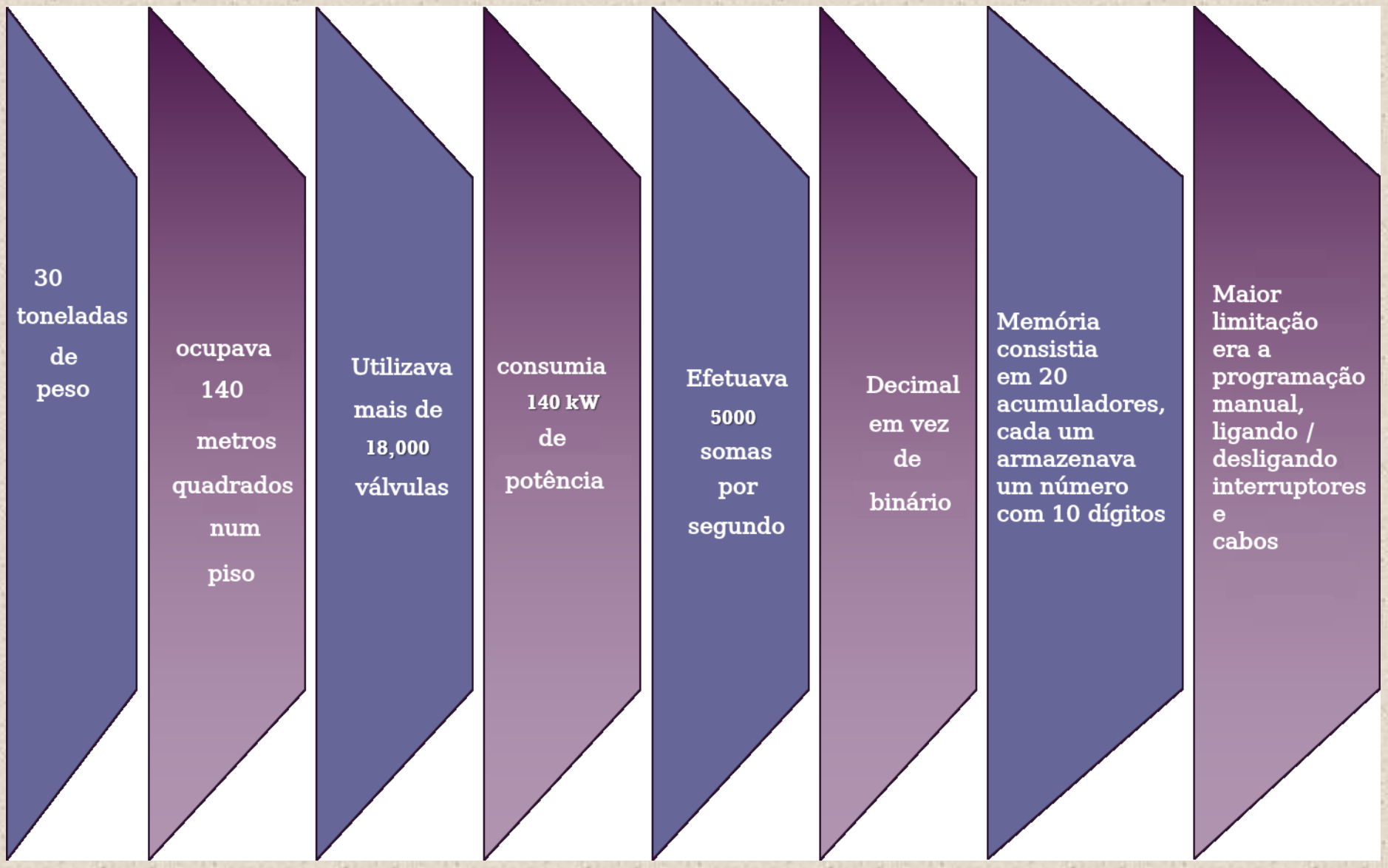
Primeira geração: válvulas

- **ENIAC**
  - **E**lectronic **N**umerical **I**ntegrator **A**nd **C**omputer
- Projetado e construído na Universidade da Pensilvânia
  - Início: 1943
  - Conclusão: 1946
  - Autores: John Mauchly e John Eckert



- Primeiro computador digital, eletrónico, de utilização genér
  - O Laboratório de Pesquisa de Balística do Exército (BRL) precisava de fornecer tabelas com as trajetórias precisas para as novas armas e dentro de um prazo razoável
  - Não foi concluído a tempo de ser utilizado durante a II guerra mundial
- A primeira aplicação foi efetuar os cálculos necessários para determinar a viabilidade da bomba de hidrogénio
- Continuou a operar sob gestão do BRL até 1955, quando foi desmantelado

ENIAC


30 toneladas de peso
ocupava 140 metros quadrados num piso
Utilizava mais de 18,000 válvulas
consumia 140 kW de potência
Efetuava 5000 somas por segundo
Decimal em vez de binário
Memória consistia em 20 acumuladores, cada um armazenava um número com 10 dígitos
Maior limitação era a programação manual, ligando / desligando interruptores e cabos

ENIAC

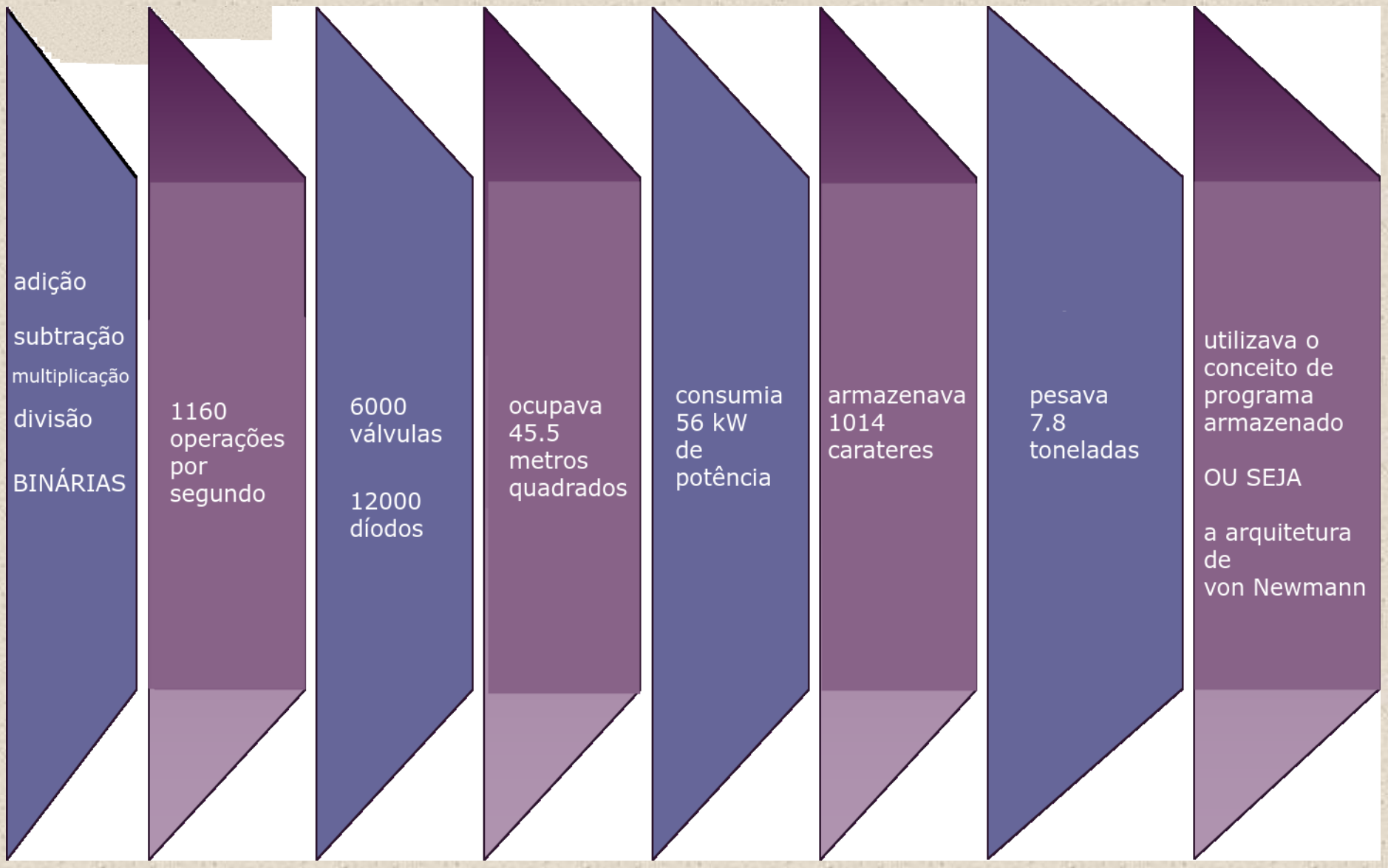
EDVAC (**E**lectronic **D**iscrete **VA**riable **C**omputer)

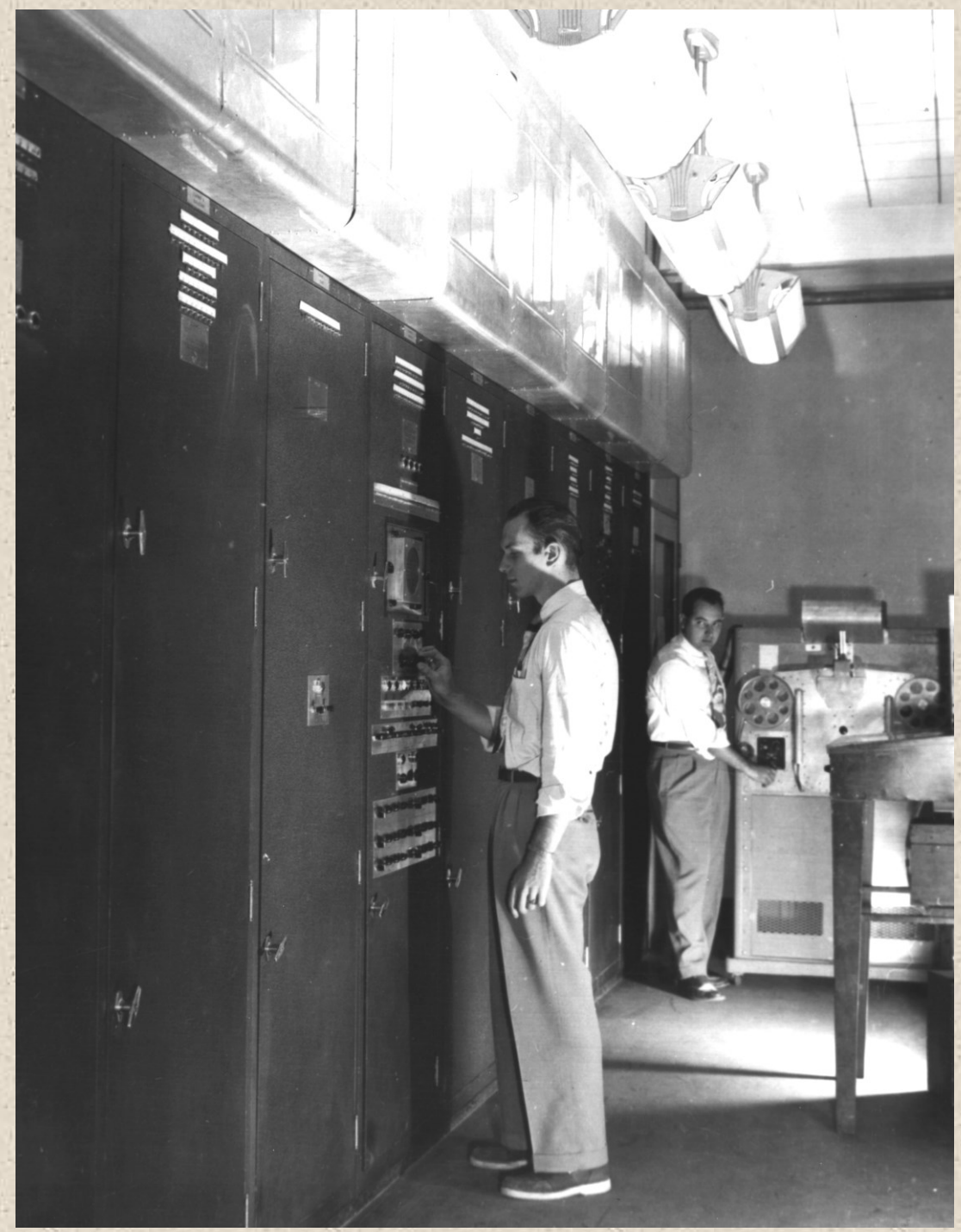
- Computador digital **binário** em vez de decimal
- O primeiro computador nos EUA a utilizar o conceito de **programa armazenado** ou seja a **arquitetura de von Neumann**
- Por isso, já não era preciso refazer as ligações para executar um programa diferente
- Começado em 1945 mas apenas concluído em 1949

Computador IAS

- Princeton **I**nstitute for **A**dvanced **S**tudies
- Construído por John von Newmann com base no trabalho de consultor do projeto EDVAC
- É a base de todos os computadores de uso genérico que se lhe seguiram
- Foi concluído em 1952

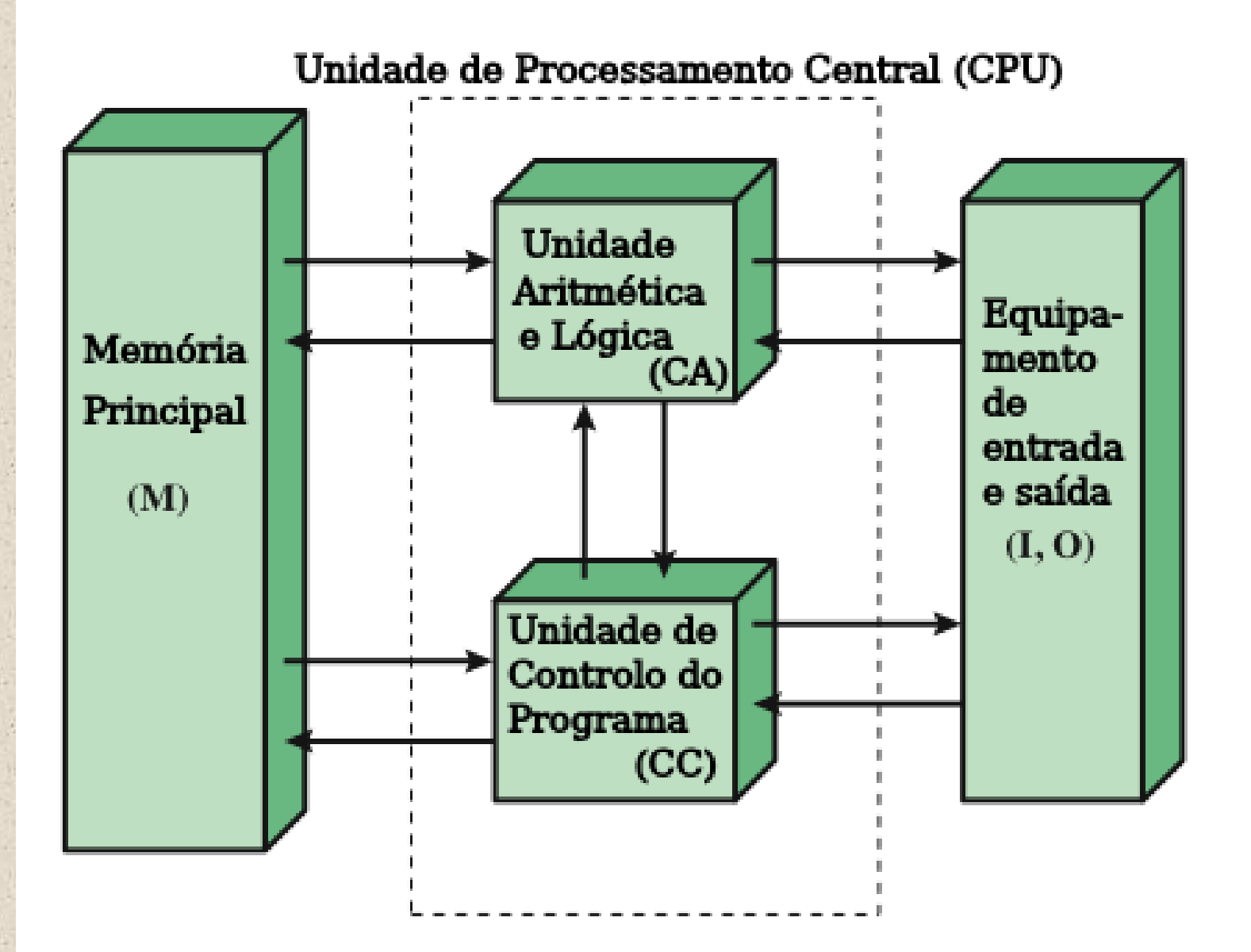
EDVAC


adição
subtração
multiplicação
divisão
BINÁRIAS
1160 operações por segundo
6000 válvulas
12000 díodos
ocupava 45.5 metros quadrados
consumia 56 kW de potência
armazenava 1014 carateres
pesava 7.8 toneladas
utilizava o conceito de programa armazenado
OU SEJA
a arquitetura de von Newmann



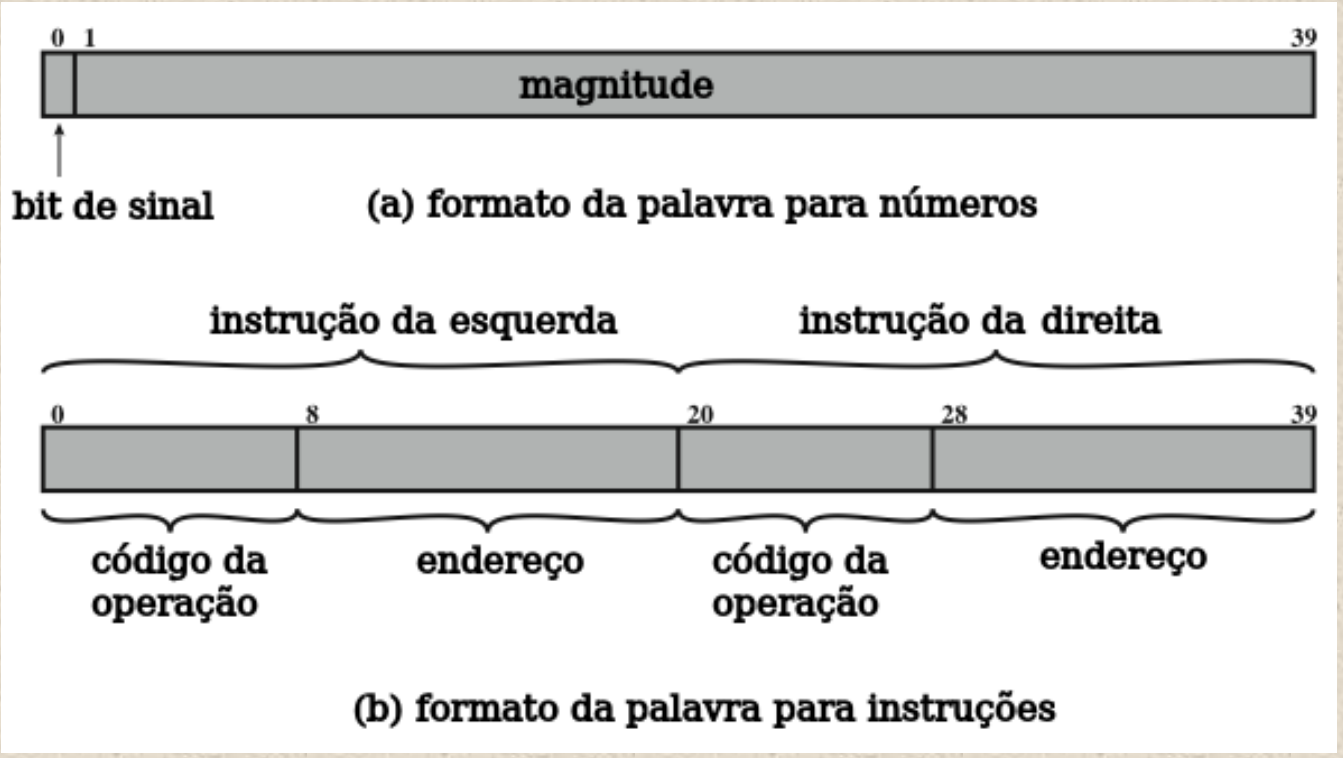
EDVAC

Estrutura da Máquina de von Neumann



Formatos da Memória do IAS

- A memória do IAS consiste em **1000** posições de armazenamento, chamadas **palavras**, com **40 bits** cada
- Tanto os dados como as instruções são armazenados nesta memória
- Os números são representados em binário: 1 bit de sinal + 39 bits para a magnitude
- A uma instrução corresponde um **código** binário de **20 bits**

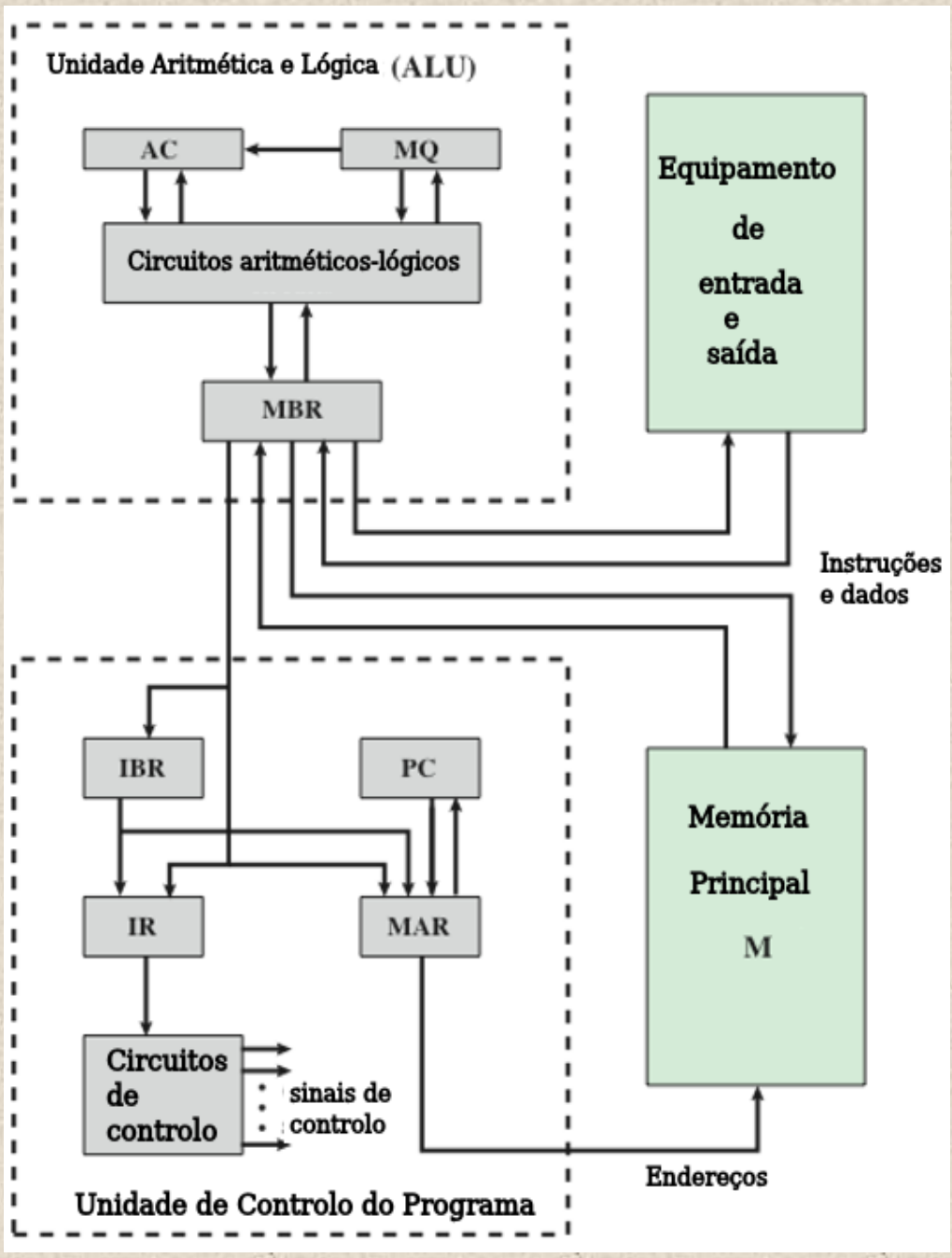


- **8-bits** para o código da operação
- **12-bits** para um endereço duma palavra da memória (0 a 999)

Estrutura do Computador IAS

Registos

IBR ⇔ cache



Registos do IAS

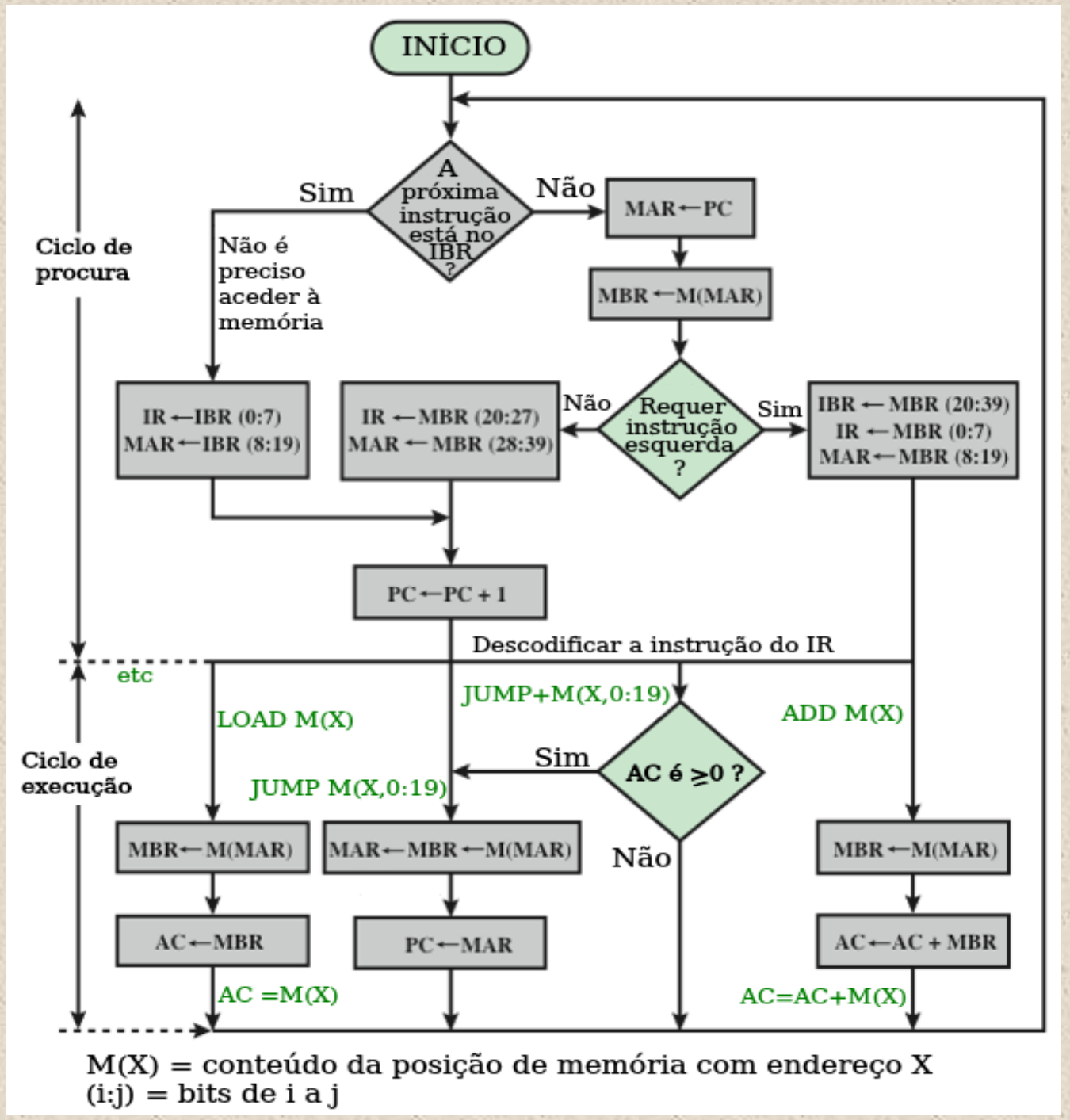
Memory buffer register (MBR)	• Contém uma palavra para ser guardada em memória ou para ser enviada para a unidade de entrada/saída • OU é utilizado para receber uma palavra da memória ou da unidade de entrada/saída
Memory address register (MAR)	• Especifica o endereço de memória onde a palavra será escrita ou o endereço da palavra a ler para o MBR
Instruction register (IR)	• Contém o código de 8 bits da instrução que está a ser executada
Instruction buffer register (IBR)	• Utilizado para guardar temporariamente a instrução da direita de uma palavra da memória
Program counter (PC)	• Contém o endereço do próximo par de instruções a ser procurado na memória
Accumulator (AC) and multiplier quotient (MQ)	• Utilizado para guardar temporariamente os operandos e os resultados das operações da ALU

Tipo de instrução	Código de operação	Representação simbólica	Descrição
Desvio incondicional	00001101	JUMP M(X,0:19)	A próxima instrução a ser executada é buscada na metade esquerda de M(X)
	00001110	JUMP M(X,20:39)	A próxima instrução a ser executada é buscada na metade direita de M(X)
Desvio condicional	00001111	JUMP+M(X,0:19)	Se o número no acumulador é um valor não-negativo, a próxima instrução a ser executada é buscada na metade esquerda de M(X)
	00010000	JUMP+M(X,20:39)	Se o número no acumulador é um valor não-negativo, a próxima instrução a ser executada é buscada na metade direita de M(X)
Aritmética	00000101	ADD M(X)	Soma M(X) a AC; armazena o resultado em AC
	00000111	ADD |M(X)|	Soma |M(X)| a AC; armazena o resultado em AC
	00000110	SUB M(X)	Subtrai M(X) de AC; armazena o resultado em AC
	00001000	SUB |M(X)|	Subtrai |M(X)| de AC; armazena o resto em AC
	00001011	MUL M(X)	Multiplica M(X) por MQ; armazena os bits mais significativos do resultado em AC, armazena os bits menos significativos em MQ.
	00001100	DIV M(X)	Divide AC por M(X); armazena o quociente em MQ e o resto em AC.
	00010100	LSH	Multiplica o acumulador por 2 (isto é, desloca os bits uma posição para a esquerda).
	00010101	RSH	Divide o acumulador por 2 (isto é, desloca os bits uma posição para a direita).

Conjunto de Instruções do IAS (1)

Conjunto de Instruções do IAS (2)

Tipo de Instrução	Código de operação	Representação simbólica	Descrição
Alteração de endereço	00010010	STOR M(X,8:19)	Substitui o campo de endereço à esquerda de M(X) pelos 12 bits mais à direita de AC.
	00010011	STOR M(X,28:39)	Substitui o campo de endereço à direita de M(X) pelos 12 bits mais à direita de AC.
Transferência de dados	00001010	LOAD MQ	Transfere o conteúdo do registo MQ para o acumulador AC
	00001001	LOAD MQ,M(X)	Transfere o conteúdo da posição de memória X para MQ
	00100001	STOR M(X)	Transfere o conteúdo do acumulador para a posição de memória X
	00000001	LOAD M(X)	Transfere M(X) para o acumulador
	00000010	LOAD – M(X)	Transfere – M(X) para o acumulador
	00000011	LOAD |M(X)|	Transfere o valor absoluto de M(X) para o acumulador
	00000100	LOAD – |M(X)|	Transfere – |M(X)| para o acumulador



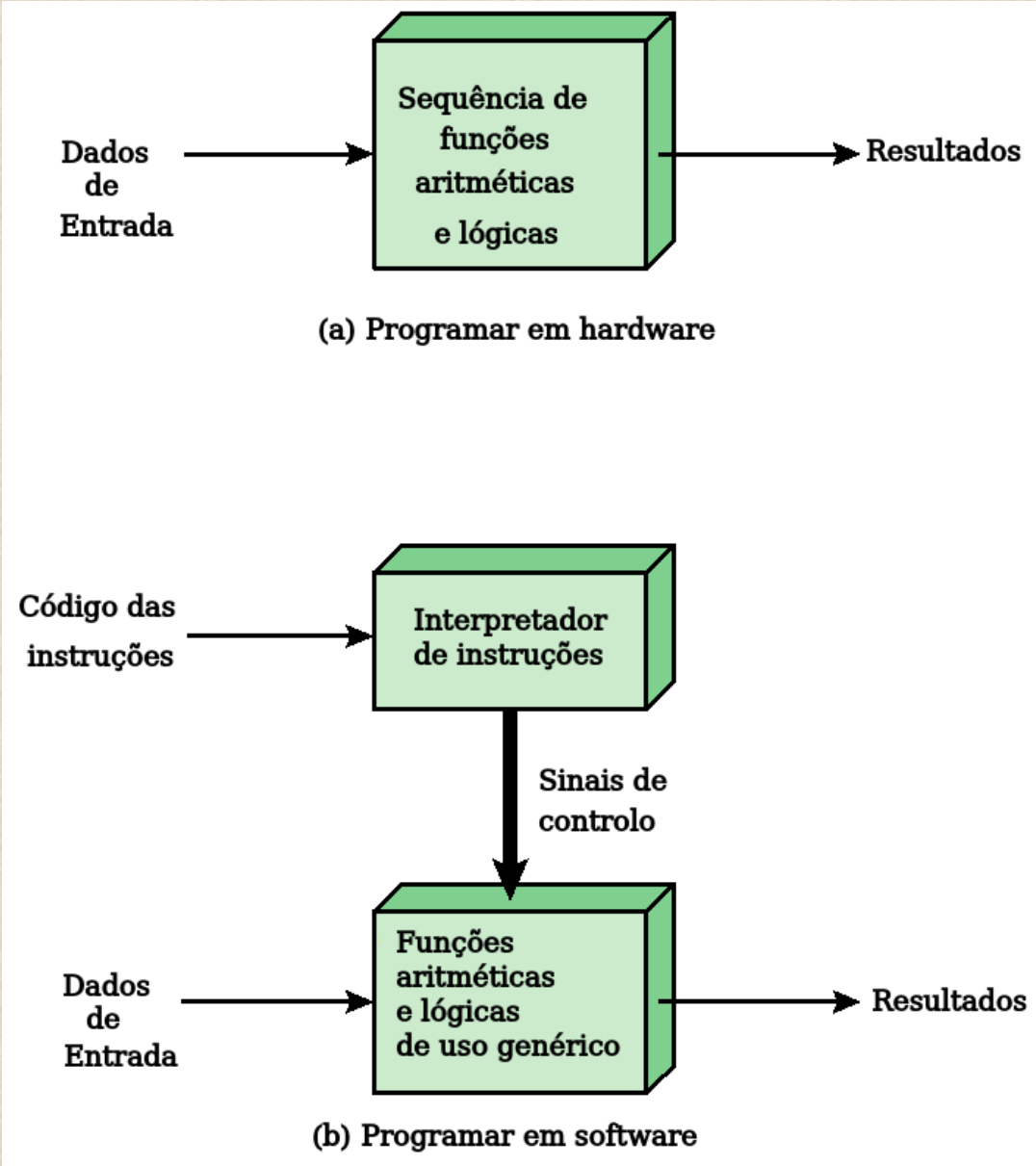
Operações do IAS

Componentes dum Computador

- Originalmente os computadores usavam "**programação" de *hardware***
  - A programação era feita ligando os componentes de acordo com a configuração desejada
  - Para executar um programa diferente, as ligações do computador tinham que ser refeitas
- Os computadores contemporâneos são projetados utilizando princípios desenvolvidos por John von Neumann no Institute for Advanced Studies, Princeton
- A arquitetura era conhecida como **arquitetura de von Neumann**, e utilizava três **conceitos** principais:
  - Os dados e as instruções eram armazenados numa única memória de leitura/escrita
  - O conteúdo desta memória era acedido indicando a posição, sem ter em conta o tipo de dados nela guardados
  - A execução ocorria de forma sequencial, passando de uma instrução para a seguinte – exceto se a ordem fosse explicitamente modificada

Abordagens da programação:

de *hardware* vs de *software*



“programação” de software ⇔ Software

- É uma sequência de **códigos** ou **instruções**
- O *hardware* **interpreta** cada instrução e gera os **sinais de controlo**
- Só é preciso fornecer a sequência de códigos adequada a cada programa, em vez de ter que se religar o *hardware*

Principais components do computador:

- **CPU**
  - **Interpretador** de instruções
  - Módulo de funções aritméticas e lógicas de **uso genérico**
- **Componentes de entrada/saída**
  - **Módulo de entrada**
    - Contém os componentes básicos para aceitar dados e instruções vindos do exterior e convertê-los para um formato utilizável pelo sistema
  - **Módulo de saída**
    - É uma forma de disponibilizar os resultados das operações

Software

CPU

Componentes de entrada/saída

Falta um componente ao computador ...

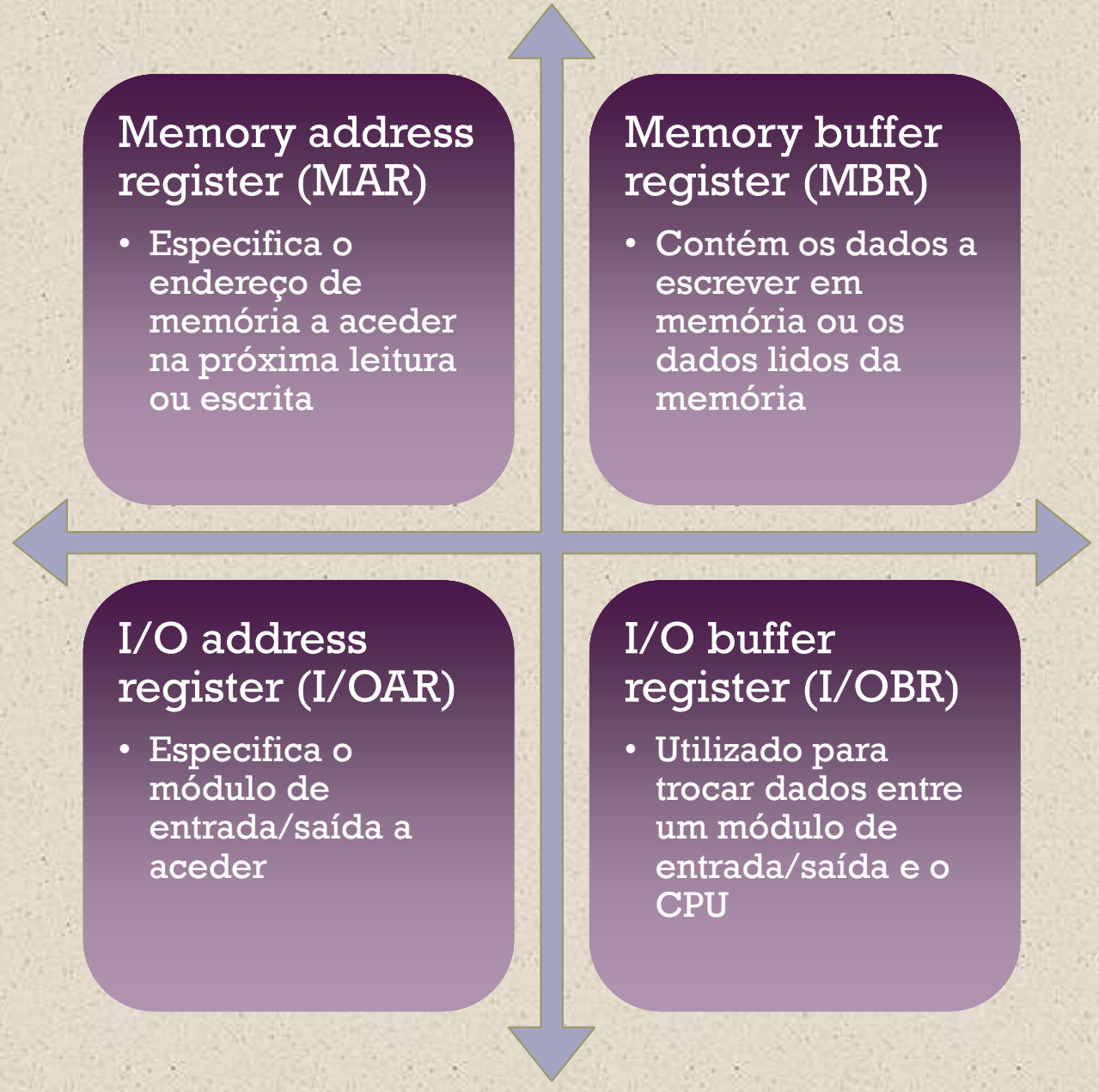
- Um programa não é sempre executado sequencialmente → pode incluir **saltos**
- As operações sobre dados podem precisar de **mais do que um valor** → um de cada vez mas retirado de uma **sequência**
- Deve haver um local para **armazenar temporariamente** as instruções e os dados
- Esse módulo é a **memória** ou memória principal
- Von Neumann sugeriu que a mesma memória podia ser usada para armazenar **instruções e dados**

MEMÓRIA

Saltos

Ciclos

Sequências de valores



MEMÓRIA

Registos

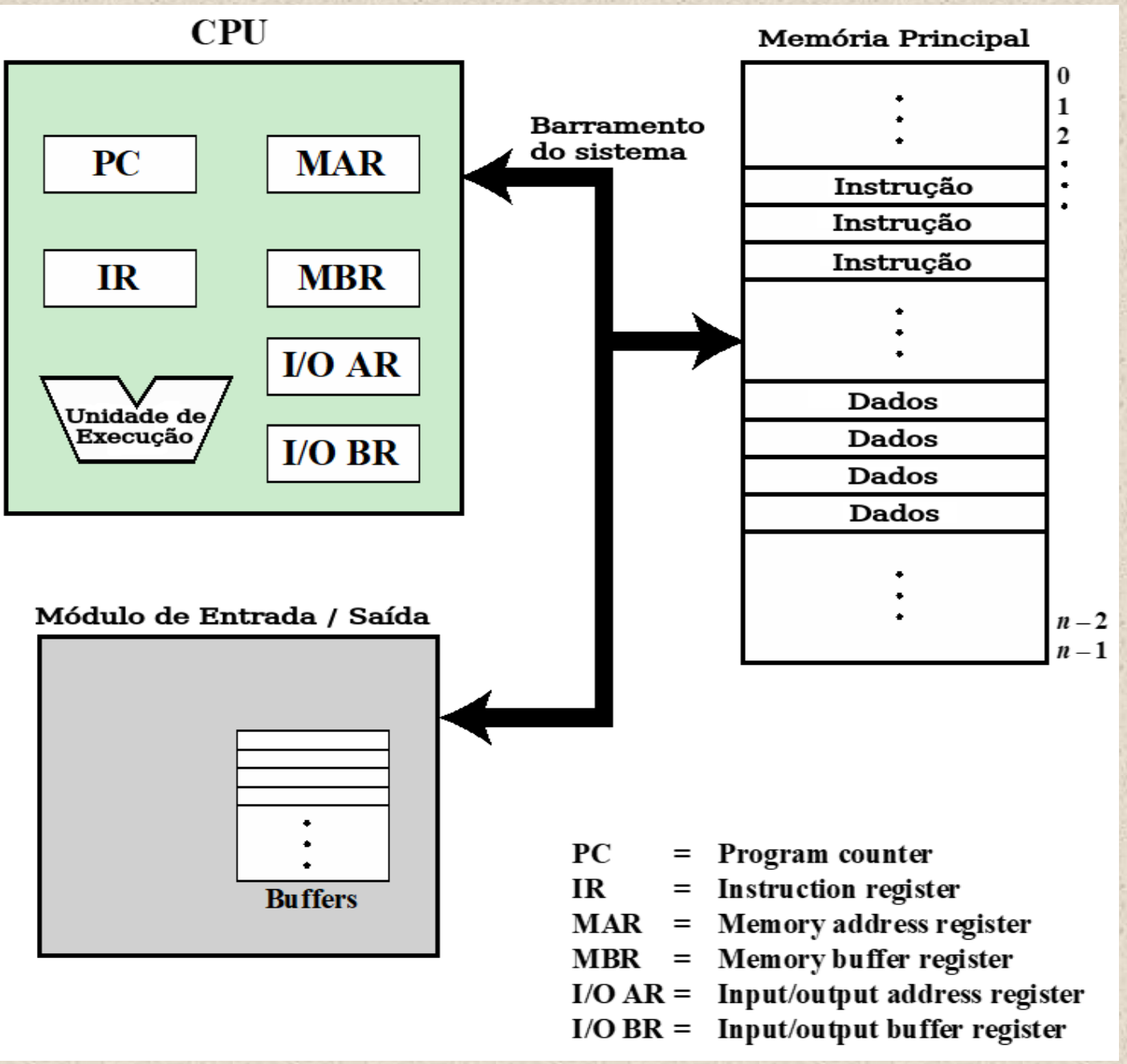
MAR

MBR

Ciclo de procura e ciclo de execução

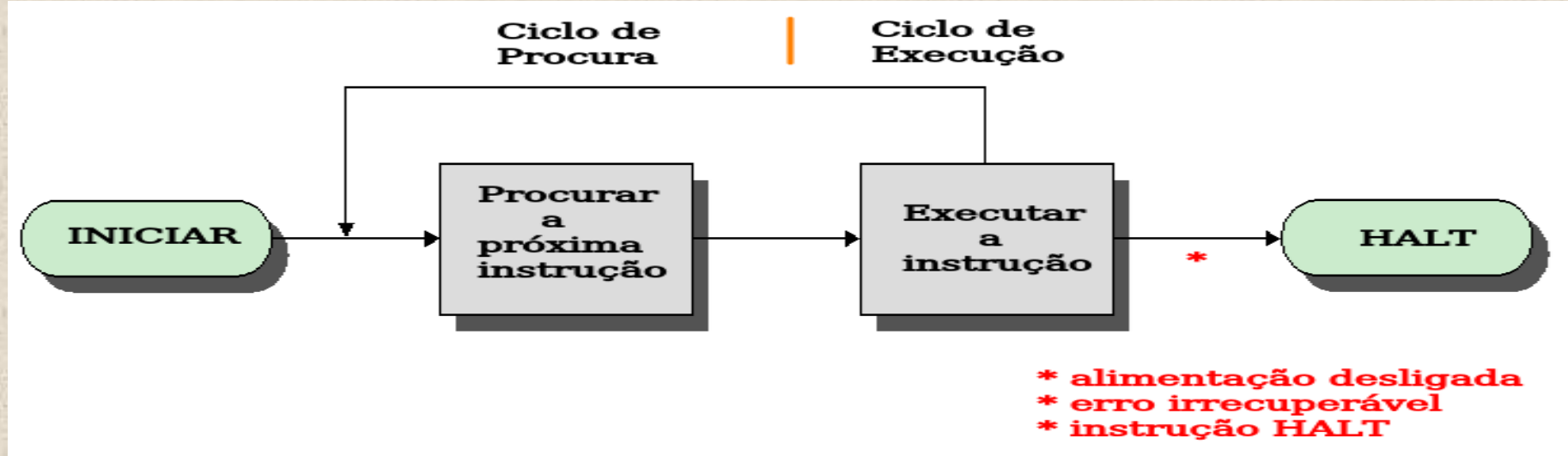
- No início de cada ciclo de instrução, o processador procura uma instrução na memória
- O *program counter* (PC) contém o endereço da próxima instrução a ser procurada
- O processador incrementa sempre o PC depois de ter procurado uma instrução, de modo a obter as instruções em sequência
- A instrução procurada é carregada no registo de instrução (IR)
- O processador interpreta a instrução e executa a ação necessária

Componentes dum computador: visão de alto nível



Ciclo da instrução

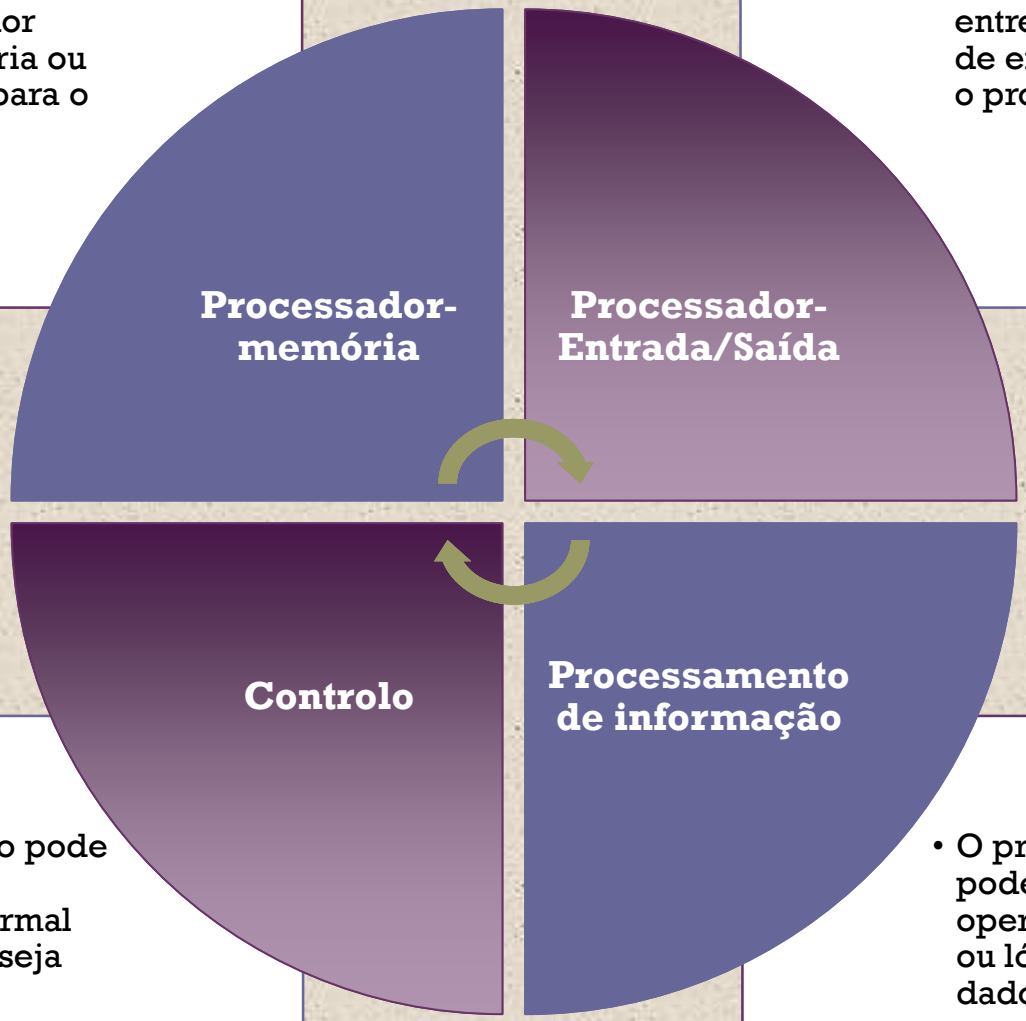
- A função principal dum computador é a execução de um **programa** … que é um **conjunto de instruções armazenadas na memória**
- Na visão mais **simplificada** o ciclo duma instrução inclui duas fases: **procura** e **execução**
- O ciclo de procura-execução será detalhado e clarificado mais à frente



Ações necessárias à execução de instruções

- Dados transferidos do processador para a memória ou da memória para o processador

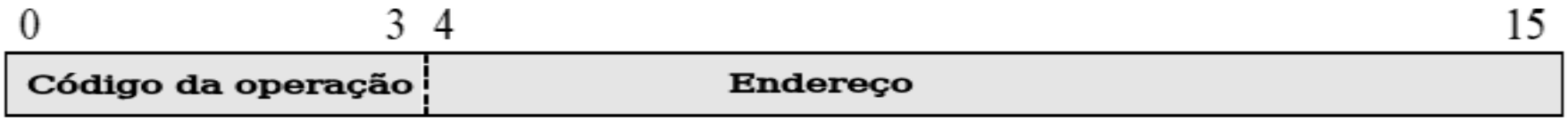
- Dados transferidos entre um dispositivo de entrada/saída e o processador



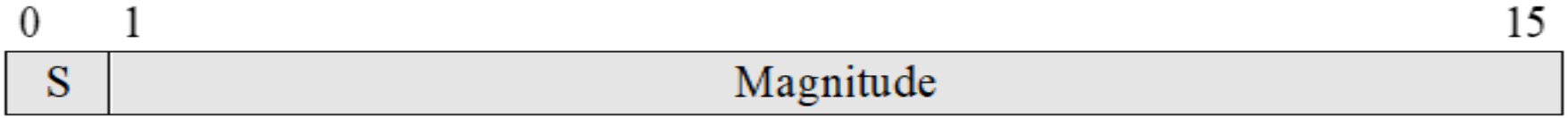
- Uma instrução pode indicar que a sequência normal de execução seja alterada

- O processador pode executar uma operação aritmética ou lógica com os dados

Máquina hipotética similar ao IAS



(a) Formato das instruções



(b) Formato dos inteiros

Program Counter (PC) = Endereço da instrução a executar
Instruction Register (IR) = Instrução em execução
Acumulador (AC) = Armazenamento temporário

(c) Registos internos ao CPU

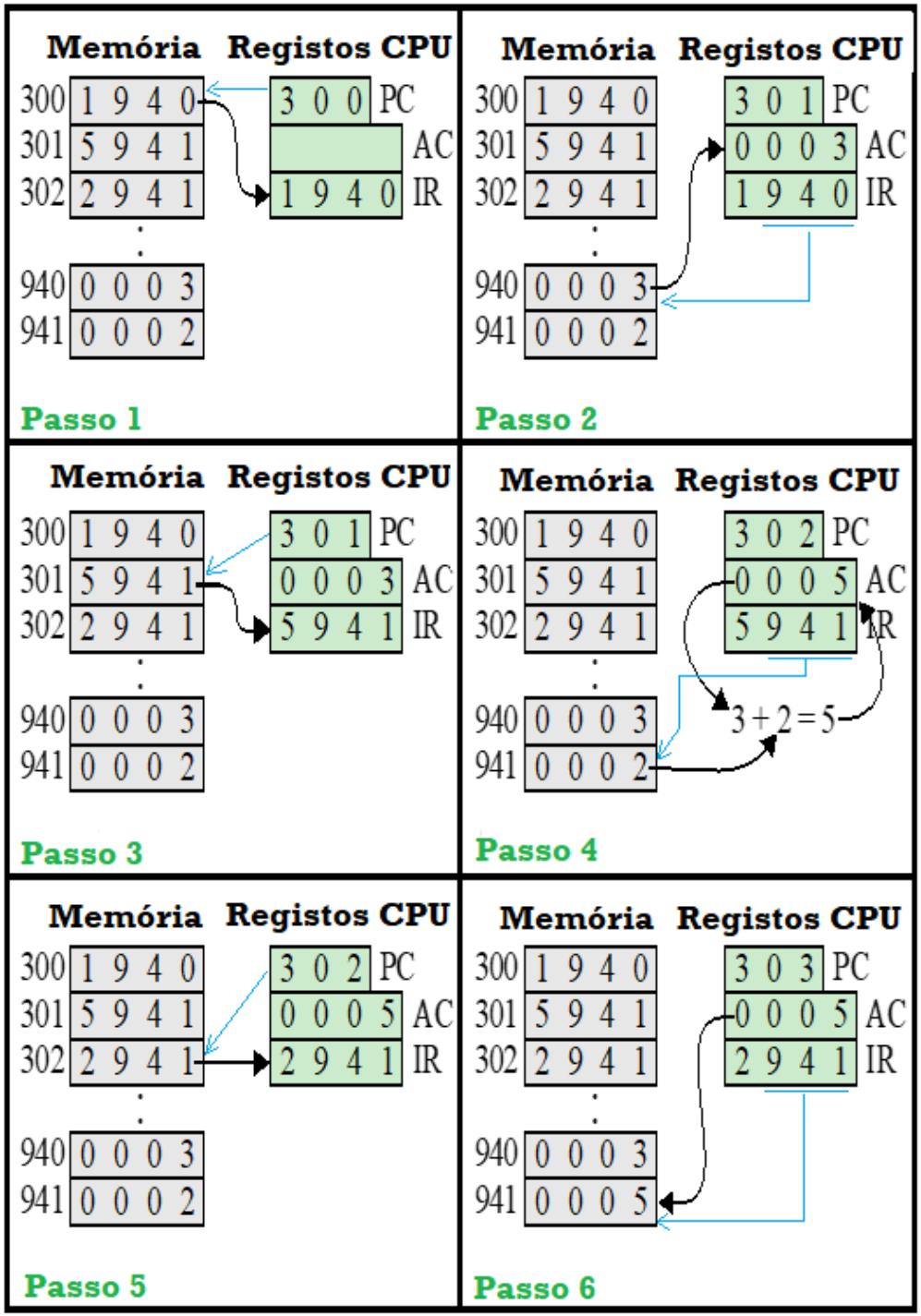
0001 = Ler da Memória para o Acumulador AC
0010 = Guardar o Acumulador AC em Memória
0101 = Adicionar ao Acumulador AC um valor da Memória
...

(d) Lista incompleta de códigos da operação

Máquina hipotética similar ao IAS

- Baseado no conjunto de instruções do IAS, sugira um código de instrução para cada uma das seguintes instruções:

```
SUB    M(X)
MUL    M(X)
DIV    M(X)
JUMP+  M(X)
JUMP   M(X)
```



Exemplo de execução dum programa

Exemplo de execução dum programa

1. O registo **PC** contém o valor **300**, o endereço da primeira instrução.
   - O conteúdo guardado no endereço de memória 300 (**1940**) é carregado para **IR**
   - O primeiro dígito, igual a **1**, é o código da operação → **LOAD AC**
   - Os restantes três dígitos são o endereço → **940**
2. A primeira instrução é executada (**0003** é carregado para o registo **AC**) e o **PC** é incrementado
3. A próxima instrução, que está guardada no endereço **301,** é carregada para o **IR** → **IR = 5941**
   - O primeiro dígito, igual a 5, é o código da operação → **ADD com AC**
   - Os restantes três dígitos são o endereço → **941**
4. Os dados guardados no endereço de memória **941** são somados ao registo **AC** e o conteúdo do **PC** é incrementado
5. A próxima instrução, que está guardada no endereço **302,** é carregada para o **IR** → **IR =2941**
   - O primeiro dígito, igual a **2**, é o código da operação → **STORE AC**
   - Os restantes três dígitos são o endereço → **941**
6. O conteúdo do **AC** é escrito na posição de memória **941** e o PC é incrementado.