



Estrutura do tema ISC

1. Representação de informação num computador
2. Organização e estrutura interna dum computador
3. Execução de programas num computador
4. O processador e a memória num computador
5. Evolução da tecnologia e da eficiência

Execução de programas num computador (1)



Níveis de abstração:

- nível das linguagens HLL (*High Level Languages*): as linguagens convencionais de programação (puro texto)
 - » imperativas e OO (Basic, Fortran, C/C++, Java, ...)
 - » funcionais (Lisp, Haskell, ...)
 - » lógicas (Prolog, ...)
- nível da linguagem *assembly* (de “montagem”): linguagem intermédia (comandos do CPU em formato texto)
- nível da linguagem máquina: a linguagem de comandos, específica p/ cada PU ou família de PU's (em binário puro)
 - » arquiteturas CISC (*Complex Instruction Set Computers*)
 - » arquiteturas RISC (*Reduced Instruction Set Computers*)



```
int x = x+y;
```

- Código C
 - somar 2 inteiros (c/ sinal)

```
addl 8(%ebp), %eax
```

Idêntico à expressão

```
x = x + y
```

- Assembly (da GNU p/ IA-32)
 - somar 2 inteiros de 4 *bytes*
 - operandos “*long*” em GCC
 - a mesma instrução, c/ ou s/ sinal
 - operandos:
 - x: em registo %eax
 - y: na memória M[%ebp+8]

```
0x401046: 03 45 08
```

- Código *object* em IA-32
 - instrução com 3 *bytes*
 - na memória a partir do endereço 0x401046

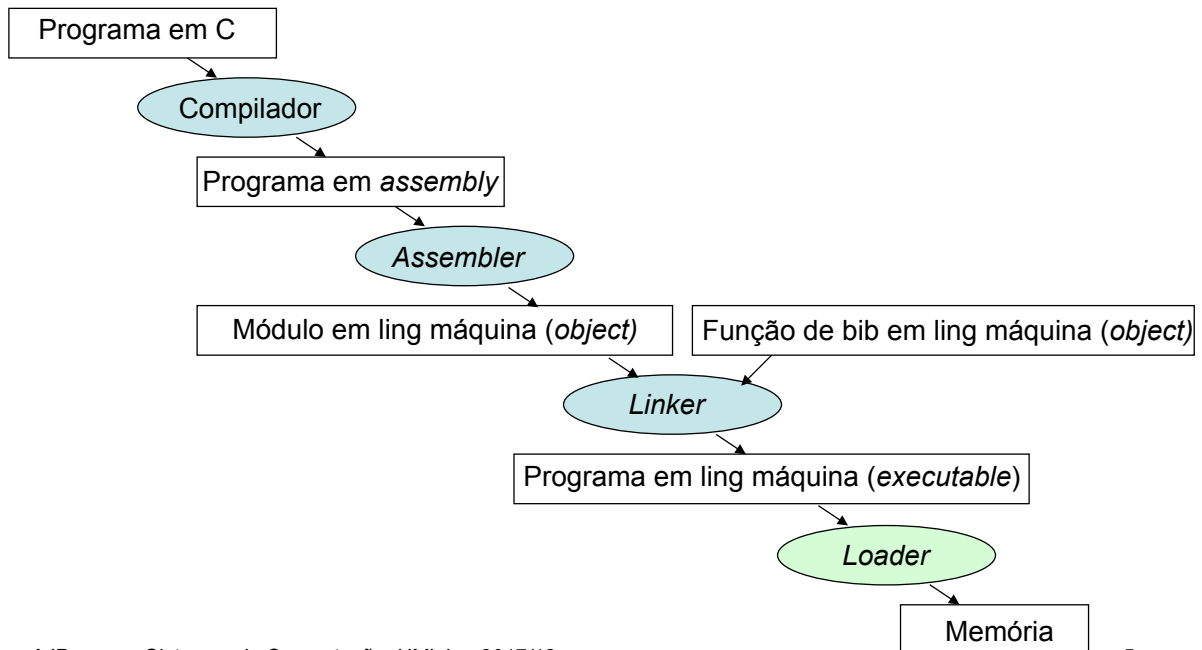


Mecanismos de conversão (para comandos do CPU):

- compilador
 - traduz um programa de um nível de abstração para outro inferior (converte um ficheiro de texto noutra de texto);
por ex., de C para *assembly*
 - normalmente inclui mais que um passo de conversão, até chegar à linguagem máquina
- assembler (“montador”)
 - “monta” os comandos / instruções em texto, para binário (*object*), de acordo com as regras do fabricante da PU
- interpretador
 - analisa, uma a uma, as instruções de um programa em HLL, e:
 - » gera código em linguagem máquina para essa instrução, e
 - » executa esse código (nota: não guarda o código gerado)



De um programa em HLL até à sua execução:



AJProença, Sistemas de Computação, UMinho, 2017/18

5

Execução de instruções (em linguagem máquina) numa PU



Ciclo de execução de instruções:

- Leitura de uma instrução da memória
... e incremento do IP
- Descodificação da instrução
- Execução da operação
 - cálculo da localização do(s) operando(s),
e ir buscá-lo(s), se necessário
 - execução da ação especificada
 - guardar resultado, se necessário

Modelo de computação de von Neumann (1945)

Análise de um exemplo: `movl Mem_Loc, %eax`

Modelo de computação de von Neumann, 1945/46 (1)

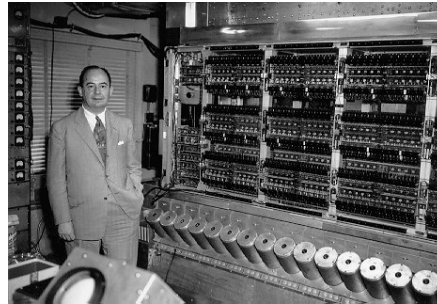
ENIAC (1ª geração, 1945)

- objetivo: cálculo tabelas de artilharia (mas 1º teste foi p/ bomba H)
- máquina **decimal** (base 10)
- 17.468 válvulas, 27 toneladas
- programação: manual, alterando as conexões (cablagem)

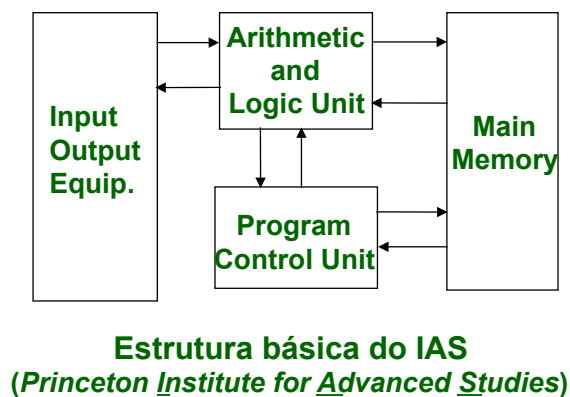


Von Neumann introduz conceito de **stored-program** :

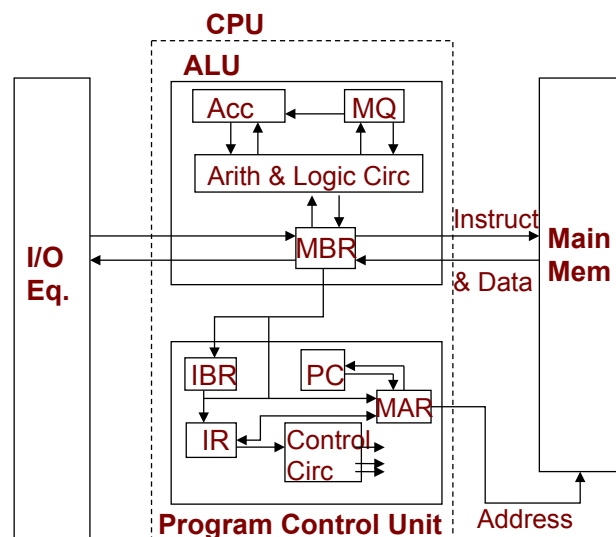
- dados e instruções em **binário**, e armazenados numa memória
- memória acedida pelo endereço da informação
- execução de instruções de modo sequencial (daí o *Program Counter*, PC), interpretadas pela unid. controlo
- constrói novo computador, IAS



Modelo de computação de von Neumann, 1945/46 (2)



Estrutura expandida do IAS



Relato do primeiro bug num computador

IBM, Mark II (09-Set-1947):

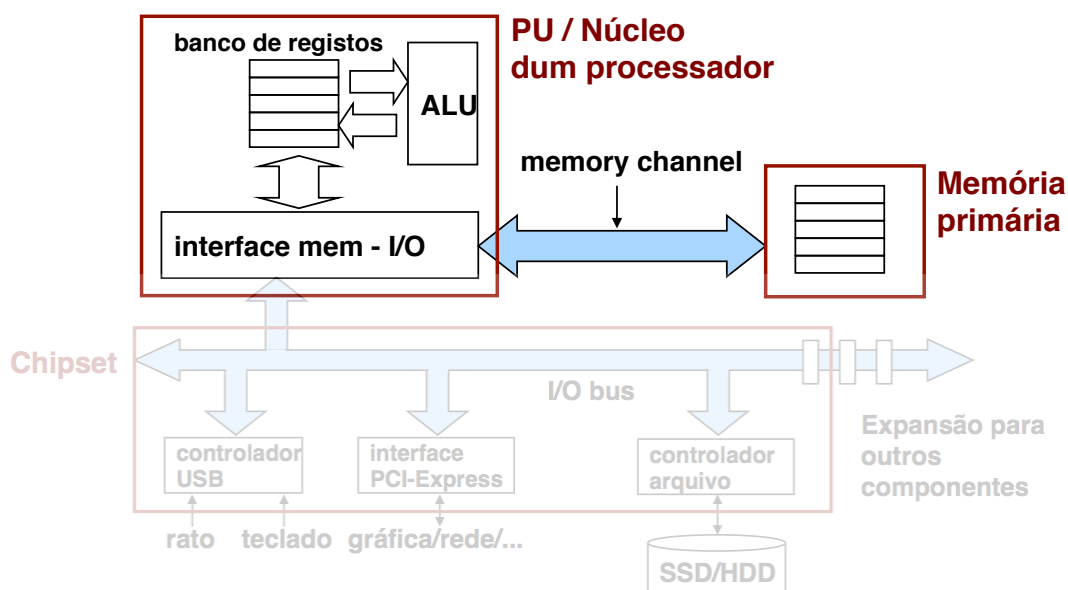
- entrada que os operadores de serviço fizeram no logbook...

9/9

0800 Anttan started
1000 " stopped - anttan ✓ { 1.2700 9.030 547 025
1300 (032) MP-MC 1.582149000 9.037 846 995 connect
(033) PRO 2 2.130476415 4.615925059(-2)
connect 2.130476415
Relays 6-2 in 033 failed special speed test
in relay
Relays changed
1100 Started Cosine Tape (Sine check)
1525 Started Multi Adder Test.
1545 Relay #70 Panel F (moth) in relay.
First actual case of bug being found.
1630 unchanged started.
1700 closed down.

Relay 3145
Relay 3370

Modelo de arquitetura de um computador elementar

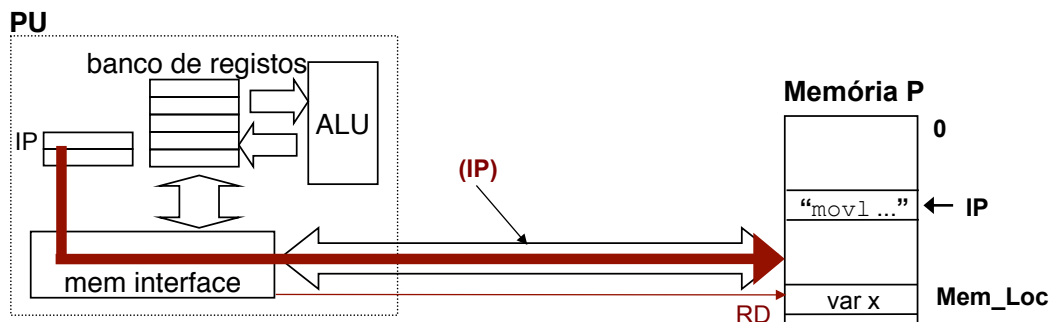


Exemplo de execução de uma instrução em linguagem máquina (1)



Ex.: `movl Mem_Loc,%eax`

1. Leitura da instrução na memória (1)



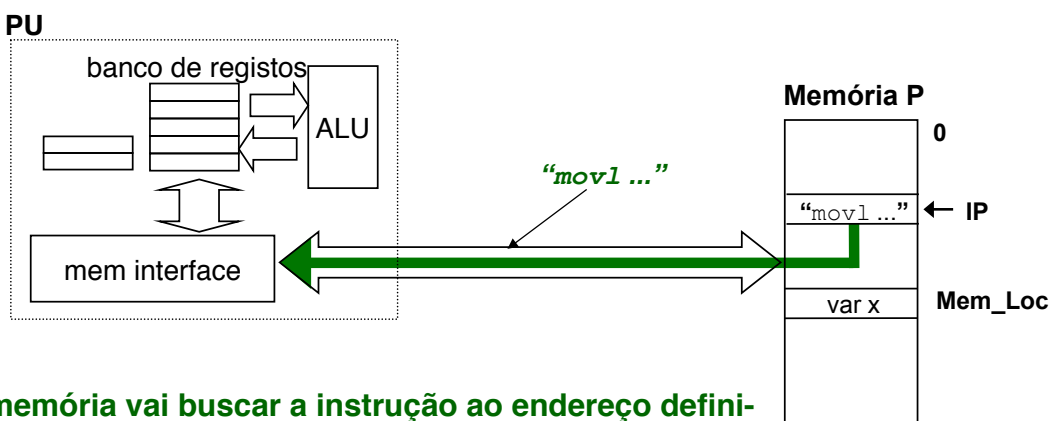
A PU coloca no *address bus* o valor em IP (endereço p/ próxima instrução), e ativa o sinal de controlo RD

Exemplo de execução de uma instrução em linguagem máquina (2)



Ex.: `movl Mem_Loc,%eax`

1. Leitura da instrução na memória (2)



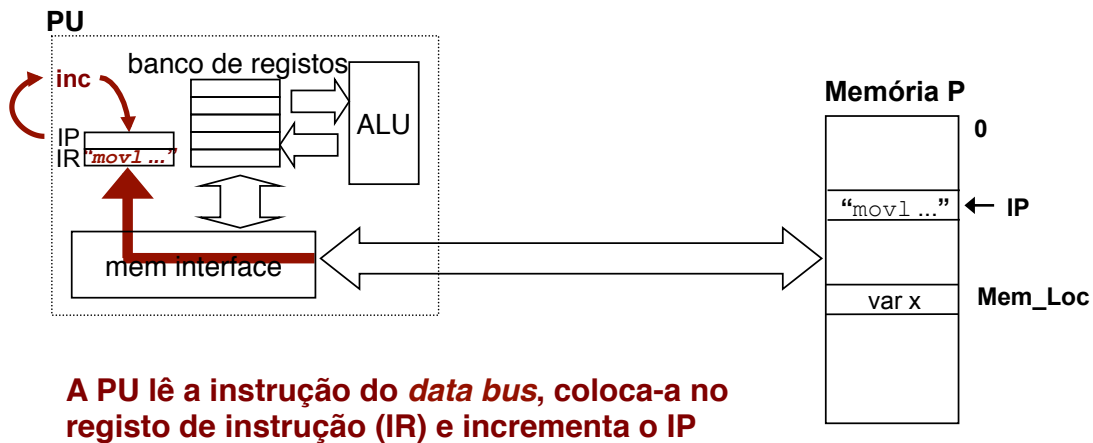
A memória vai buscar a instrução ao endereço definido por IP e coloca-a no *data bus* p/ ser lida pela PU

Exemplo de execução de uma instrução em linguagem máquina (3)

Ex.: `movl Mem_Loc,%eax`

1. Leitura da instrução na memória (3)

... e incremento do IP



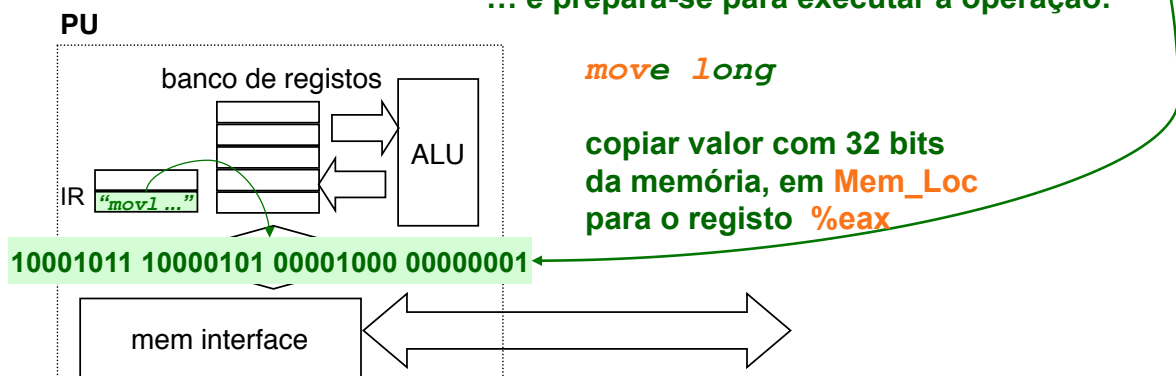
Exemplo de execução de uma instrução em linguagem máquina (4)

Ex.: `movl Mem_Loc,%eax`

2. Descodificação da instrução

A unidade de controlo da PU descodifica a instrução...

... e prepara-se para executar a operação:

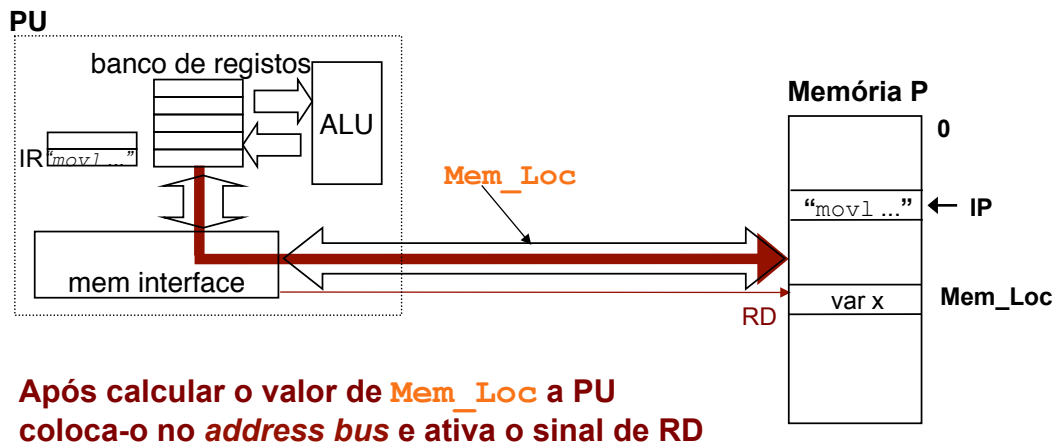


Exemplo de execução de uma instrução em linguagem máquina (5)



Ex.: `movl Mem_Loc,%eax`

3. Execução da operação (1)

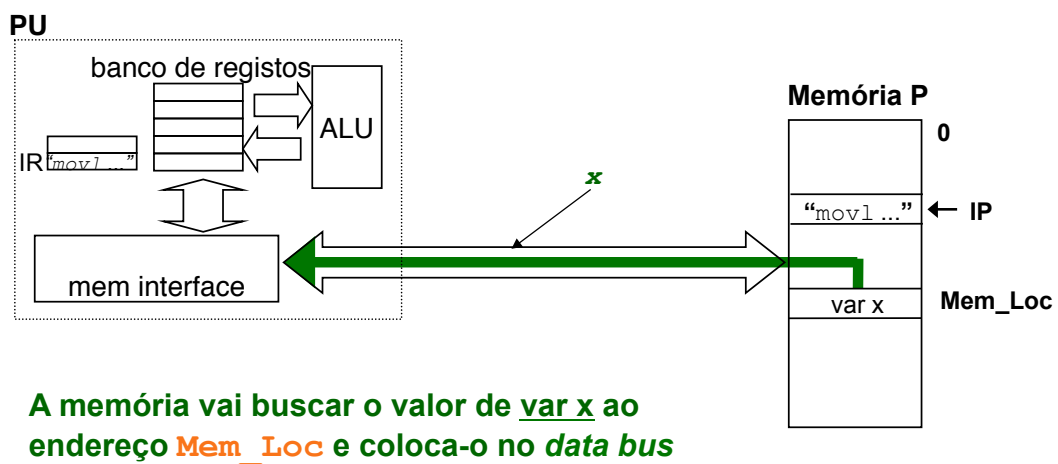


Exemplo de execução de uma instrução em linguagem máquina (6)



Ex.: `movl Mem_Loc,%eax`

3. Execução da operação (2)





Ex.: `movl Mem_Loc,%eax`

3. Execução da operação (3)

