PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

PCS-2302 / PCS-2024
Lab. de Fundamentos de Eng. de Computação

Aula 01

Introdução
Máquina de von Neumann

Professores:
Anarosa Alves Franco Brandão (PCS 2302)
Jaime Simão Sichman (PCS 2302)
Reginaldo Arakaki (PCS 2024)
Ricardo Luís de Azevedo da Rocha (PCS 2024)

Monitores: Diego Queiroz e Tiago Matos

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

A ideia da Máquina de von Neumann (1)

- O **Modelo de von Neumann** procura oferecer uma alternativa prática, disponibilizando ações mais poderosas e ágeis em seu repertório de operações.

- Isso viabiliza, para os mesmos programas, codificações muito mais expressivas, compactas e eficientes.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

A ideia da Máquina de von Neumann (2)

- Para isso, a Máquina de von Neumann utiliza:
  - **Memória endereçável**, usando acesso aleatório
  - **Programa armazenado** na memória, para definir diretamente a função corrente da máquina (ao invés da MEF)
  - **Dados** representados na memória (ao invés da fita)
  - Codificação numérica **binária** em lugar da unária
  - **Instruções variadas e expressivas** para a realização de operações básicas muito frequentes (ao invés de sub máquinas específicas)
  - **Maior flexibilidade** para o usuário, permitindo operações de entrada e saída, comunicação física com o mundo real e controle dos modos de operação da máquina

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

Elementos da Arquitetura a Simular (1)

- Neste curso pretende-se simular um *processador muito simples*, porém estruturalmente similar aos disponíveis na realidade

- O processador tem um conjunto de elementos físicos de armazenamento de informações
  - **Memória Principal:** para armazenar programas e dados
  - **Acumulador (AC):** funciona como área de trabalho, para a execução de operações aritméticas e lógicas
  - Outros **registradores auxiliares**: empregados em diversas operações intermediárias no processamento dos programas

- O conjunto de dados neles contidos em cada instante constitui o **estado instantâneo** do processamento

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Aula 1:

Introdução
Máquina de Turing
Máq. von Neumann

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

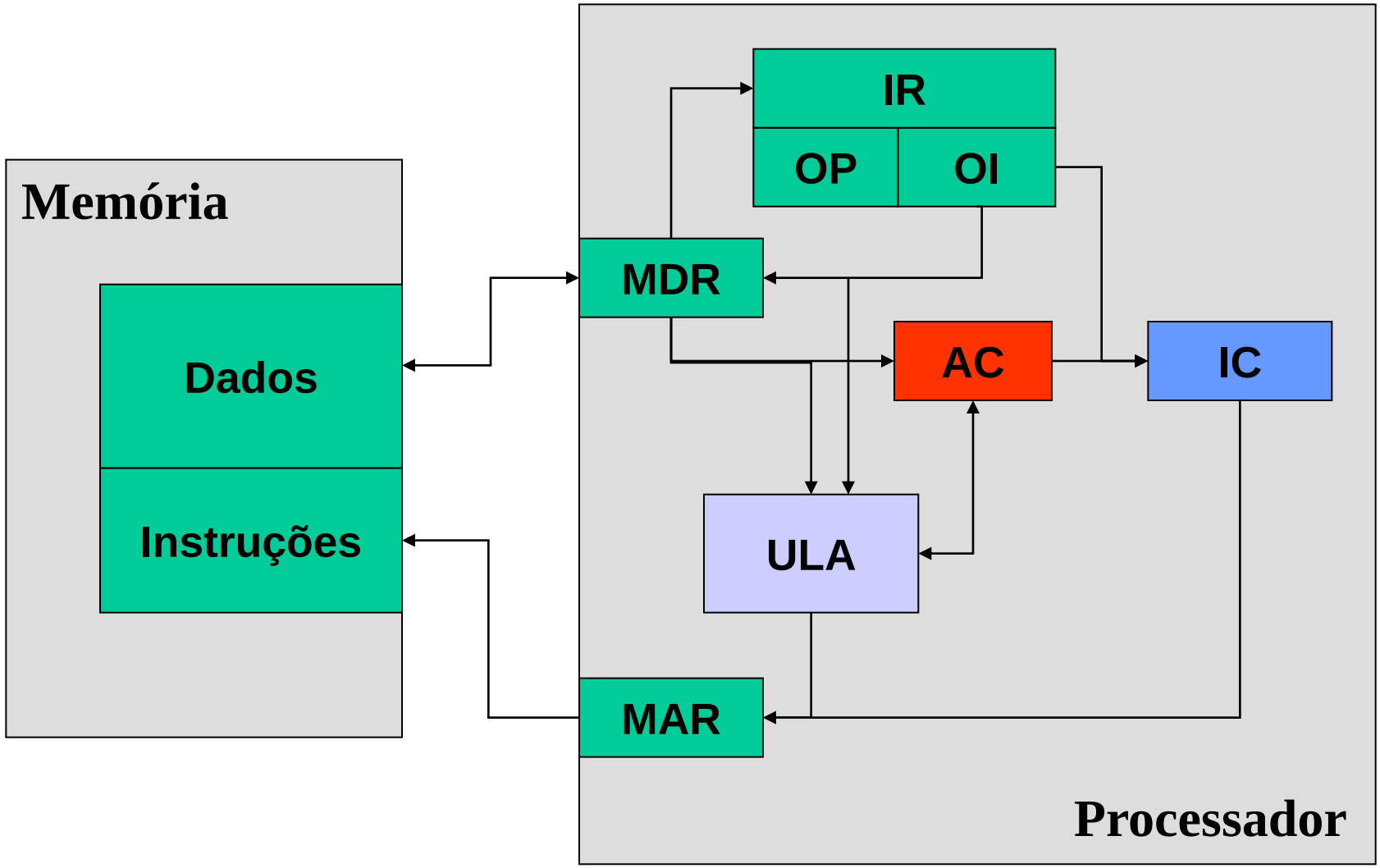
Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

Elementos da Arquitetura a Simular (2)

- Os **Registradores Auxiliares** são:
  - **Registrador de Dados da Memória (MDR)** – serve como ponte para os dados que trafegam entre a memória e os outros elementos da máquina
  - **Registrador de Endereço da Memória (MAR)** – indica qual é a origem ou o destino, na memória principal, dos dados contidos no registrador de dados da memória.
  - **Registrador de Endereço de Instrução (IC)** – indica em cada instante qual será a próxima instrução a ser executada pelo processador.
  - **Registrador de Instrução (IR)** – contém a instrução em execução
    - **Código de Operação (OP)** – parte do registrador de instrução que identifica a instrução que está sendo executada
    - **Operando da Instrução (OI)** – complementa a instrução indicando o dado ou o endereço sobre o qual ela deve agir.

Diagrama da Arquitetura a Simular (3)



PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Aula 1:

Introdução
Máquina de Turing
Máq. von Neumann

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Aula 1:

Introdução
Máquina de Turing
Máq. von Neumann

Autores:

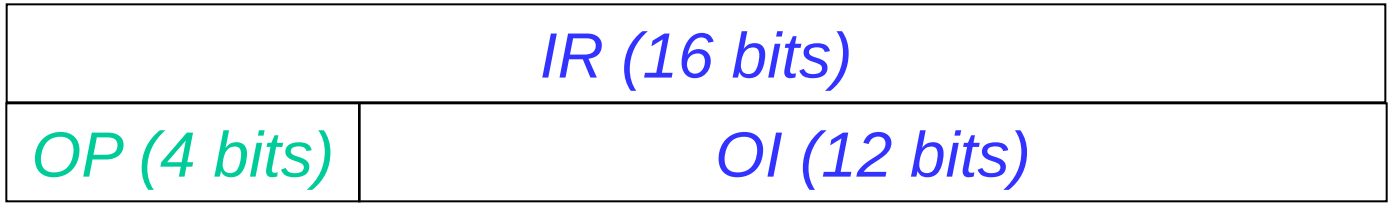
Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

Conjunto de registradores da Máquina de von Neumann (MVN)

MAR	**Registrador de endereço de memória**
MDR	**Registrador de dados da memória**
IC	**Registrador de endereço de instrução**
IR	**Registrador de instrução**
OP	**Registrador de código de operação**
OI	**Registrador de operando de instrução**
AC	**Acumulador**



PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

Funcionamento de um Simulador

Devem-se separar dois conceitos independentes na lógica de um simulador:

- **Comandos de controle do simulador**: esta parte independe da arquitetura do computador que se está simulando, e sua função é de orientar a operação do programa simulador e de permitir ao usuário observar e alterar o conteúdo dos componentes do processador simulado.

- **Execução das instruções do processador simulado**: esta parte do simulador depende fortemente da arquitetura da máquina cuja operação se deseja simular, que deve implementar um modelo da máquina simulada, no nível de granularidade mais conveniente em cada caso.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

Comandos de Controle do Simulador

- Conta-se com os seguintes comandos de controle para o programa simulador:
  - **[INITIALIZE]** – atribui valores iniciais padrão a todos os elementos importantes do simulador e da arquitetura.
  - **[LOAD]** – serve para carregar programas e dados para a memória da máquina simulada
  - **[STEP]** – serve para colocar o simulador no modo de operação passo a passo.
  - **[RUN]** – serve colocar o simulador no modo de operação contínuo.
  - **[EXECUTE]** – serve para promover a execução do programa, conforme o modo de operação: execução contínua/uma instrução por vez.
  - **[SHOW]** – serve para mostrar o conteúdo das memórias da máquina simulada, após a execução de um passo (modo STEP) ou após a execução de um programa (modo RUN).

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

[EXECUTE] – Obtenção e Decodificação

EXECUTE - Serve para promover a execução do programa, conforme o modo de operação: contínua/ou uma instrução por vez

1) **Determinação da Instrução a Executar**

2) **Fase de Obtenção da Instrução**

•Obter na memória, no endereço contido no registrador de Endereço de Instrução, o código da instrução desejada

3) **Fase de Decodificação da Instrução**

•Decompor a instrução em duas partes: o código da instrução e o seu operando, depositando essas partes nos registradores de instrução e de operando, respectivamente.

•Selecionar, com base no conteúdo do registrador de instrução, um procedimento de execução dentre os disponíveis no repertório do simulador (passo 4).

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

Conjunto de instruções da Máquina de von Neumann (MVN)

Código (hexa)	Instrução	Operando
0	Desvio incondicional	endereço do desvio
1	Desvio se acumulador é zero	endereço do desvio
2	Desvio se acumulador é negativo	endereço do desvio
3	Deposita uma constante no acumulador	constante relativa de 12 bits
4	Soma	endereço da parcela
5	Subtração	endereço do subtraendo
6	Multiplicação	endereço do multiplicador
7	Divisão	endereço do divisor
8	Memória para acumulador	endereço-origem do dado
9	Acumulador para memória	endereço-destino do dado
A	Desvio para subprograma (função)	endereço do subprograma
B	Retorno de subprograma (função)	endereço do resultado
C	Parada	endereço do desvio
D	Entrada	dispositivo de e/s (*)
E	Saída	dispositivo de e/s (*)
F	Chamada de supervisor	constante (**)

(*) ver slides seguintes
(**) por ora, este operando (tipo da chamada) é irrelevante, e esta instrução nada faz.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

[EXECUTE] – Execução de instrução (1)

4) **Fase de Execução da Instrução**

- Executar o procedimento selecionado em 3, usando como operando o conteúdo do registrador de operando, preenchido anteriormente.

- Caso a instrução executada não seja de desvio, incrementar o registrador de endereço de instrução a executar. Caso contrário, o procedimento de execução já terá atualizado convenientemente tal informação.

4.1) Execução da instrução (decodificada em 3)

– De acordo com o código da instrução a executar (contido no registrador de instrução), executar os procedimentos de simulação correspondentes (detalhados adiante)

4.2) Acerto do registrador de Endereço de Instrução para apontar a próxima instrução a ser simulada:

– Incrementar o registrador de Endereço de Instrução.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

[EXECUTE] – Execução de instrução (2)

- Obs.: Sistema de **numeração e aritmética** adotada: Binário, em complemento de dois
  - representa inteiros e executa operações em 16 bits.
  - o bit mais à esquerda é o bit de sinal (1 = negativo)

Registrador de instrução = 0 (desvio incondicional)

- modifica o conteúdo do registrador de Endereço de Instrução (**IC**) armazenando nele o conteúdo do registrador de operando (**OI**)

  IC := OI

Registrador de instrução = 1 (desvio se acumulador é zero)

- se o conteúdo do acumulador for zero, então modifica o conteúdo do registrador de Endereço de Instrução (**IC**), armazenando nele o conteúdo do registrador de operando (**OI**)

  Se AC = 0 então IC := OI

  senão IC := IC + 1

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Aula 1:

Introdução
Máquina de Turing
Máq. von Neumann

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

[EXECUTE] – Execução de instrução (3)

Registrador de instrução = 2 (desvio se negativo)

- se o conteúdo do acumulador (**AC**) for negativo, isto é, se o bit mais significativo for 1, então modifica o conteúdo do registrador de Endereço de Instrução (**IC**) armazenando nele o conteúdo do registrador de operando (**OI**)

  Se AC < 0 então IC := OI

  senão IC := IC + 1

Registrador de instrução = 3 (constante para acumulador)

- Armazena no acumulador (**AC**) o número relativo de 12 bits contido no registrador de operando (**OI**), estendendo seu bit mais significativo (bit de sinal) para completar os 16 bits do acumulador

  AC := OI

  IC := IC +1

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

[EXECUTE] – Execução de instrução (4)

Registrador de instrução = 4 (soma)

- Soma ao conteúdo do acumulador (**AC**) o conteúdo da posição de memória indicada pelo registrador de operando MEM[**OI**]
- Guarda o resultado no acumulador

```
AC := AC + MEM[OI]
IC := IC + 1
```

Registrador de instrução = 5 (subtração)

- Subtrai do conteúdo do acumulador (**AC**) o conteúdo da posição de memória indicada pelo registrador de operando MEM[**OI**]
- Guarda o resultado no acumulador

```
AC := AC - MEM[OI]
IC := IC + 1
```

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

[EXECUTE] – Execução de instrução (5)

Registrador de instrução = 6 (multiplicação)

- Multiplica o conteúdo do acumulador (**AC**) pelo conteúdo da posição de memória indicada pelo registrador de operando MEM[**OI**]
- Guarda o resultado no acumulador

```
AC := AC * MEM[OI]
IC := IC + 1
```

Registrador de instrução = 7 (divisão inteira)

- Dividir o conteúdo do acumulador (**AC**) pelo conteúdo da posição de memória indicada pelo registrador de operando MEM[**OI**]
- Guarda a parte inteira do resultado no acumulador

```
AC := int (AC / MEM[OI])
IC := IC + 1
```

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

[EXECUTE] – Execução de instrução (6)

Registrador de instrução = 8 (memória para acumulador)

- Armazena no acumulador (**AC**) o conteúdo da posição de memória cujo endereço é o conteúdo do registrador de operando MEM[**OI**]

  AC := MEM[OI]

  IC := IC + 1

Registrador de instrução = 9 (acumulador para memória)

- Guarda o conteúdo do acumulador (**AC**) na posição de memória indicada pelo registrador de operando MEM[**OI**]

  MEM[OI] := AC

  IC := IC + 1

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

[EXECUTE] – Execução de instrução (7)

Registrador de instrução = A (desvio para subprograma)

- Armazena o conteúdo do registrador de Endereço de Instrução (**IC**), incrementado de uma unidade, na posição de memória apontada pelo registrador de operando MEM[**OI**]
- Armazena no registrador de Endereço de Instrução (**IC**) o conteúdo do registrador de operando incrementado de uma unidade (**OI**)

```
MEM[OI] := IC + 1
IC := OI + 1
```

Registrador de instrução = B (retorno de subprograma)

- Armazena no registrador de Endereço de Instrução (**IC**) o conteúdo que está na posição de memória apontada pelo registrador de operando MEM[**OI**]

```
IC := MEM[OI]
```

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

[EXECUTE] – Execução de instrução (8)

Registrador de instrução = C (stop)

- Modifica o conteúdo do registrador de Endereço de Instrução (**IC**) armazenando nele o conteúdo do registrador de operando (**OI**)

  IC := OI

Registrador de instrução = D (input)

- Aciona o dispositivo indicado, fazendo a leitura de dados do mesmo
- Transfere dado para o acumulador

  (solicita dado do dispositivo)

  AC := dado de entrada

  IC := IC + 1

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

[EXECUTE] – Execução de instrução (9)

Registrador de instrução = E (output)

- Aciona o dispositivo indicado
- Transfere o conteúdo do acumulador (**AC**) para o dispositivo, esperando que este termine de executar a operação de gravação

dado de saída := AC

(aciona dispositivo)

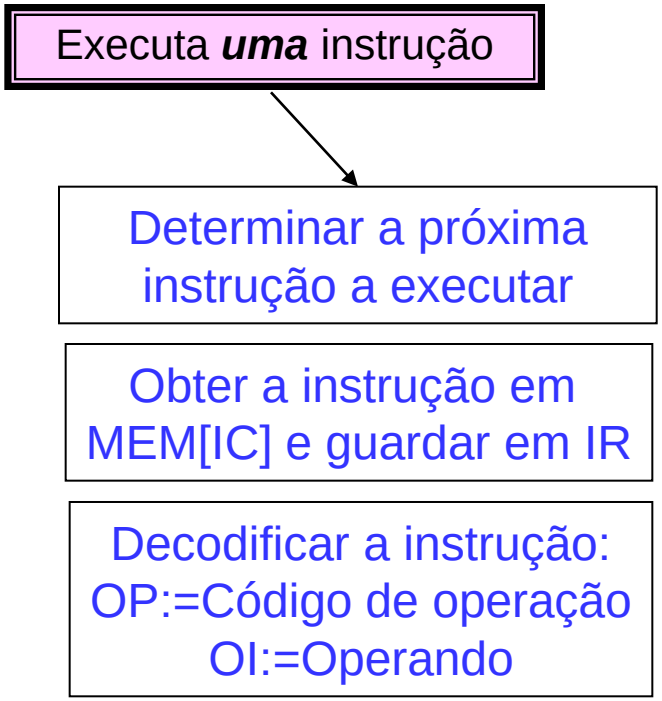
IC := IC + 1

Registrador de instrução = F (supervisor call)

(não implementada: por enquanto esta instrução não faz nada)

IC := IC + 1

Diagrama de fluxo do Interpretador [detalhamento de EXECUTA]



OP (hexa)	Ação a executar
0	IC:=OI
1	Se AC=0 então IC:=OI senão IC:=IC+1
2	Se AC<0 então IC:=OI senão IC:=IC+1
3	AC:=OI ; IC:=IC+1
4	AC:=AC+MEM[OI] ; IC:=IC+1
5	AC:=AC-MEM[OI] ; IC:=IC+1
6	AC:=AC*MEM[OI] ; IC:=IC+1
7	AC:=int(AC/MEM[OI]) ; IC:=IC+1
8	AC:=MEM[OI] ; IC:=IC+1
9	MEM[OI]:=AC ; IC:=IC+1
A	MEM[OI]:=IC+1; IC:=OI+1
B	IC:=MEM[OI]
C	IC:=OI
D	aguarda; AC:= dado de entrada; IC:=IC+1
E	dado de saída := AC ; aguarda ; IC:=IC+1
F	(nada faz por ora) ; IC:=IC+1

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Aula 1:

Introdução
Máquina de Turing
Máq. von Neumann

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

Conjunto de registradores da Máquina de von Neumann (MVN)

Operações de Entrada e Saída

OP	*Tipo*	*Dispositivo*

OP — D (entrada) ou E (saída)

Tipo — Tipos de dispositivo:
- 0 = Teclado
- 1 = Monitor
- 2 = Impressora
- 3 = Disco

Dispositivo — Identificação do dispositivo. Pode-se ter vários tipos de dispositivo, ou *unidades lógicas* (LU). No caso do disco, um arquivo é considerado uma unidade lógica.

Pode-se ter, portanto, até 16 tipos de dispositivos e, cada um, pode ter até 256 unidades lógicas.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

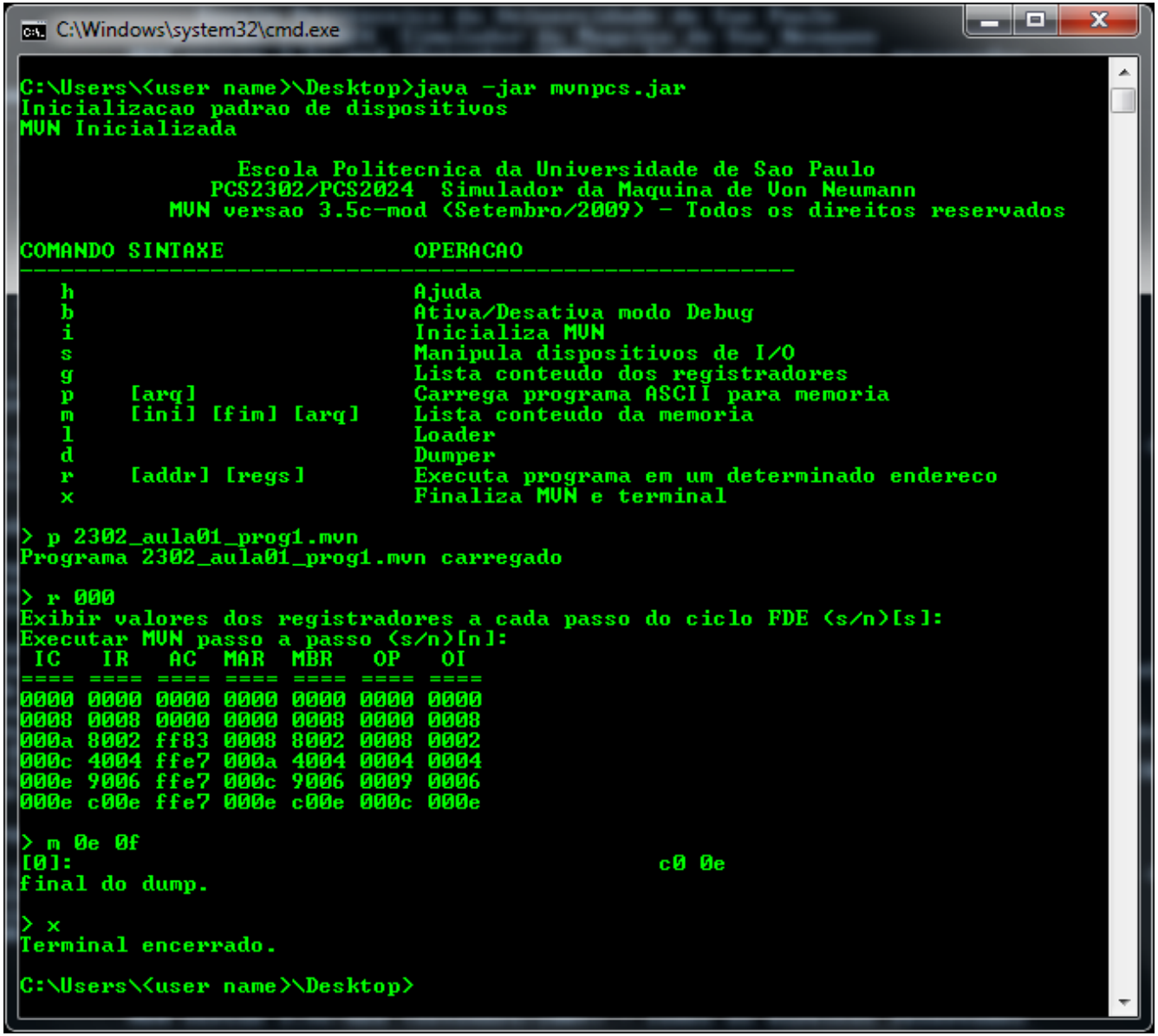
v. 2.2 ago. 2010

Exemplo de Programa – Prog1

- Problema: Somar o valor de duas variáveis iniciadas com os valores -125_{10} e 100_{10}, colocando o resultado em outra variável.

```
; prog1.mvn
; Soma os valores de duas posições de memória e guarda o
; resultado em outra posição de memória, parando na
; instrução final.
000 0008  ; Ponto de entrada: JMP para as instruções
; Constantes
002 FF83  ; A = 0xFF83 (-125)
004 0064  ; B = 0x0064 (100)
; Variáveis
006 0000  ; RESULTADO deverá ser 0xFFE7 (-25)
; Instruções do programa
008 8002  ; Carrega o conteúdo de A no acumulador
00A 4004  ; Adiciona B ao conteúdo do acumulador
00C 9006  ; Armazena o resultado em RESULTADO
00E C00E  ; Para em 0x000E
```

Execução de Programa – Prog1



PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Aula 1:

Introdução
Máquina de Turing
Máq. von Neumann

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

Exemplo de Programa – Prog2 (1)

- Problema: Implementar uma sub-rotina que subtrai dois inteiros. Os valores dos argumentos estão armazenados em duas variáveis do programa principal. O resultado é armazenado em uma variável do programa principal.

```
; prog2.mvn
; Programa de ilustração para chamada de sub-rotina
;   int subtrair(int x, int y) {
;     return x - y;
;   }
;
000 0010  ; Pula para o início das instruções
; Constantes
002 0010  ; A = 0x0010 (16)
004 0064  ; B = 0x0064 (100)
; Variáveis
006 0000  ; RESULTADO de subtrair() = 0xFFAC (-84)
```

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Aula 1:

Introdução
Máquina de Turing
Máq. von Neumann

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

Exemplo de Programa – Prog2 (2)

```
; Programa principal
; Chamando SUBTRAIR(A, B)
010 8002  ; Carrega o conteúdo de A no acumulador
012 903C  ; Armazena no parâmetro X
014 8004  ; Carrega o conteúdo de B
016 903E  ; Armazena no parâmetro Y
018 A040  ; Chama a sub-rotina SUBTRAIR
01A 9006  ; Armazena o resultado em RESULTADO
01C C01C  ; Para em 0x01C
;
; Sub-rotina SUBTRAIR
; Parâmetros formais
03C 0000  ; X
03E 0000  ; Y
; Corpo da sub-rotina
040 0000  ; Endereço de retorno
042 803C  ; Carrega o conteúdo de X
044 503E  ; Subtrai Y, resultado no ACUMULADOR
046 B040  ; Retorna para o endereço contido em 0x040
```

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Aula 1:

Introdução
Máquina de Turing
Máq. von Neumann

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

Execução de Programa – Prog2

```
> p 2302_aula01_prog2.mvn
Programa 2302_aula01_prog2.mvn carregado

> r 000
Exibir valores dos registradores a cada passo do ciclo FDE (s/n)[s]:
Executar MVN passo a passo (s/n)[n]:
 IC   IR   AC  MAR  MBR   OP   OI
==== ==== ==== ==== ==== ==== ====
0000 0000 0000 0000 0000 0000 0000
0010 0010 0000 0000 0010 0000 0010
0012 8002 0010 0010 8002 0008 0002
0014 903c 0010 0012 903c 0009 003c
0016 8004 0064 0014 8004 0008 0004
0018 903e 0064 0016 903e 0009 003e
0042 a040 0064 0018 a040 000a 0040
0044 803c 0010 0042 803c 0008 003c
0046 503e ffac 0044 503e 0005 003e
001a b040 ffac 0046 b040 000b 0040
001c 9006 ffac 001a 9006 0009 0006
001c c01c ffac 001c c01c 000c 001c

>
```

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

Algumas práticas de programação (1)

- O conjunto de instruções desta máquina de von Neumann é extremamente limitado, exigindo alguns artifícios para a obtenção dos efeitos necessários:
  - Não há operações lógicas. Tudo deve ser feito com operações aritméticas.
  - Não há endereçamento indireto nem indexado. Tudo deve ser feito alterando-se convenientemente as instruções disponíveis, no próprio programa, antes de executá-las.
  - Incrementos e decrementos de variáveis devem ser feitos somando-se ou subtraindo-se as constantes desejadas (tipicamente 1 ou 2) às variáveis alvo.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

Algumas práticas de programação (2)

- Não há instruções específicas para todos os testes. Tudo deve ser feito combinando-se as instruções de desvios condicionais e usando-se lógica invertida quando necessário.

- Convém separar sub-rotinas já testadas e muito usadas, bem como variáveis e constantes, dos programas em desenvolvimento.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

Algumas práticas de programação (3)

- Esta MVN suporta endereçamento de 12bits

- À medida que os programas ficam maiores e/ou tem-se mais de um programa na memória, é importante planejar um **mapa de memória**. Uma sugestão para um mapa bem simples é reservar os endereços 0x0000 – 0x01FF para área de dados, constantes, tabelas, etc., e os endereços a partir de 0x0200 para programas principais e sub-rotinas.

- Projete sempre no papel seus programas em linguagem de máquina, e simule seu funcionamento no papel antes de utilizar o computador. Economiza-se muito tempo e esforço evitando-se a depuração de erros na base da tentativa e de testes.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

Algumas práticas de programação (4)

- Documente todos os programas desenvolvidos com comentários informativos no código, e no papel, com diagramas de fluxo e com desenhos ilustrativos das estruturas de dados utilizadas e das operações efetuadas. Em programação binária, é muito raro que, passados alguns dias, mesmo o autor consiga lembrar-se exatamente de como funciona o programa que ele próprio criou.

- Projete bem e anote os testes realizados e os resultados esperados. É frequente ter de repeti-los para as novas versões de um programa em desenvolvimento.

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Aula 1:

Introdução
Máquina de Turing
Máq. von Neumann

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

Bibliografia (Programação de Sistemas)

Relíquias Preciosas

Barron, D. W. ***Assemblers and Loaders*** (3rd. ed.) MacDonald/Elsevier, 1978

Beck, L. L. ***System Software - An Introduction to Systems Programming*** Addison-Wesley, 1985

Calingaert, P. ***Assemblers, Compilers and Program Translation*** Computer Science Press, 1979

Donovan, J. J. ***Systems Programming*** McGraw-Hill, 1972

Duncan, F.G. ***Microprocessor Programming and Software Development*** Prentice Hall, 1979.

Freeman, P. ***Software System Principles*** SRA, 1975

Gear, C. W. ***Computer Organization and Programming (3rd. ed.)*** McGraw-Hill, 1981

Graham, R. M. ***Principles of Systems Programming*** John Wiley & Sons, 1975

Gust, P. ***Introduction to Machine and Assembly Language Programming*** Prentice Hall, 1986

Maginnis, J. B. ***Elements of Compiler Construction*** Appleton-Century-Crofts, Meredith Co., 1972

Presser, L. and White, J. R. ***Linkers and Loaders*** ACM Comp. Surveys, vol. 4, n. 3, pp. 149-168

Rosen, S. (ed.) ***Programming Systems and Languages*** McGraw-Hill, 1967

Tseng, V. (ed.) ***Microprocessor Development and Development Systems*** McGraw-Hill, 1982

Ullman, J. D. *Fundamental Concepts of Programming Systems* Addison-Wesley, 1976

Wegner, P. ***Progr. Languages, Inf. Structures and Machine Organization*** McGraw-Hill, 1968.

Welsh, J. and McKeag, M. ***Structured System Programming*** Prentice-Hall,1980

PCS 2302/2024
Laboratório de Fundamentos da Eng.de Computação

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha

Aula 1:

Introdução
Máquina de Turing
Máq. von Neumann

Autores:

Anna H. R. Costa
Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:
Diego Queiroz
Tiago Matos

v. 2.2 ago. 2010

Referências Bibliográficas

Costa, A.H.R., Sato, L.M., Sichman, J.S., Tori, R. ***Material didático da disciplina PCS 2214 – Fundamentos da Engenharia de Computação I***, PCS/EPUSP, São Paulo, SP. 2004-2005.

Sipser, M. ***Introduction to the Theory of Computation***. PWS Publishing Company, Boston, MA. 1997.

Leitura complementar:

UM SIMULADOR-INTERPRETADOR PARA A LINGUAGEM DE MÁQUINA DO PATINHO FEIO.

(João José Neto, Aspectos do Projeto de Software de um Minicomputador, Dissertação de Mestrado, EPUSP, S. Paulo, 1975, cap.3)