

# 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)

Marcos A. Simplício Junior (PCS 2302)

Reginaldo Arakaki (PCS 2024)

Paulo Sergio Muniz Silva (PCS 2024)

**Monitores:** Allan Diego Lima, Luis Gustavo Nardin, Marcelo Amaral

## Roteiro

1. Planejamento da disciplina
2. Da Máquina de Turing à Máquina de von Neumann
  - a. Visão geral da Máquina de Turing
  - b. Problemas práticos da Máquina de Turing
  - c. Exemplo de uma máquina muito simples na arquitetura von Neumann
  - d. Exemplo de um simulador de uma máquina de von Neumann (MVN)
3. Parte Experimental
  - a. Pequenos programas em código da máquina MVN



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

Professores:  
Anarosa A. F. Brandão  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

3

## Planejamento da disciplina (1)

### • Objetivos da disciplina

- Apresentar conceitos fundamentais da engenharia de computação, do ponto de vista do software, tendo os seguintes temas como motivação:
  - Máquina de von Neumann
  - Principais aspectos dos Programas de Sistema
    - Programas de Sistema: programas que dão suporte à operação de um computador (montadores, carregadores, bibliotecas, sistemas operacionais, etc.)



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

Professores:  
Anarosa A. F. Brandão  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

4

## Planejamento da disciplina (1)

### • Objetivos da disciplina (cont.)

- Desenvolver alguns programas de sistema para um simulador da Máquina de von Neumann
- Codificar na linguagem Java partes de programas de sistema existentes, implementados segundo o paradigma da orientação a objetos



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

Professores:  
Anarosa A. F. Brandão  
Marcos A. Símplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

5

## Planejamento da disciplina (2)

### • Método

– Aulas ministradas em laboratório com:

- Exposição conceitual dos problemas a resolver
- Realização experimental dos conceitos apresentados para atender à meta da aula

### • Componentes da Avaliação

- Nota de comprometimento (C) – avaliação individual
- Média das notas dos produtos criados na aula (R)
- Média das notas de duas provas (P)

### • Avaliação final = $(1.C + 4.R + 5.P) / 10$



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

Professores:  
Anarosa A. F. Brandão  
Marcos A. Símplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

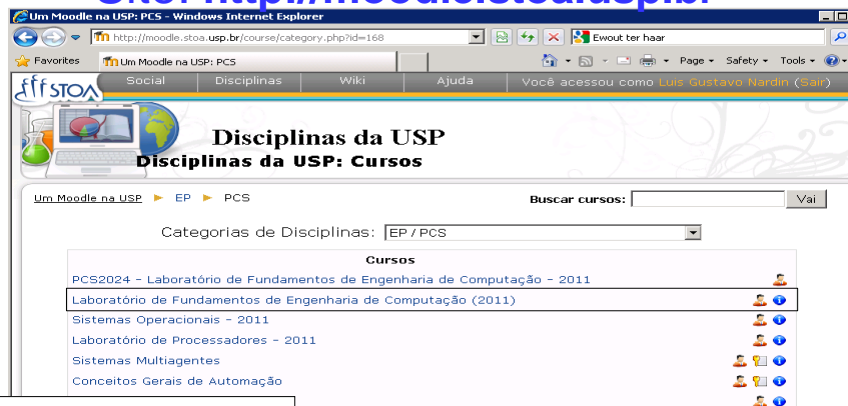
Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

6

## Planejamento da disciplina (3)

Site: <http://moodle.stoa.usp.br>



- Slides das aulas, em PDF
- Material didático de apoio
- Instruções e avisos
- Links úteis

Fazer cadastro em:  
<http://stoa.usp.br/cadastro/>



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

Professores:  
Anarosa A. F. Brandão  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

7

## Planejamento da disciplina (4)

- **Aula 1**
  - Das Máquinas de Turing (MT) à Máquina de von Neumann (MVN).
  - Exemplos de programas em um simulador da Máquina de von Neumann (MVN).
  - Exercícios. Pequenos programas em código de máquina MVN.
- **Aula 2**
  - Exercícios. Rotinas de uma biblioteca elementar de apoio a programas de sistema do simulador da Máquina de von Neumann (MVN) em código de máquina MVN.



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

Professores:  
Anarosa A. F. Brandão  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

8

## Planejamento da disciplina (5)

- **Aula 3**
  - Elementos básicos da orientação a objetos (Parte I).
  - Constructos básicos da linguagem Java.
  - Arquitetura de software do simulador MVN.
  - Exercícios. Incremento, em código de máquina MVN, da biblioteca elementar de apoio para o simulador MVN. Extensões de instrução do simulador da MVN em Java.
- **Aula 4**
  - Descrição do montador absoluto para o simulador MVN.
  - Exercício. Pequeno programa de sistema em linguagem simbólica do montador absoluto, utilizando rotinas da biblioteca elementar de apoio.

## Planejamento da disciplina (6)

- **Aula 5**
  - Os programas de sistema *Dumper* e *Loader*.
  - Exercícios. Implementar um *dumper* e um *loader* absoluto para a MVN, em linguagem simbólica do montador absoluto.
- **Aula 6**
  - Exercício. Implementar a primeira parte de um monitor *batch* elementar para a MVN, em linguagem simbólica do montador absoluto, utilizando rotinas da biblioteca elementar de apoio.
- **Aula 7**
  - 1a. Prova

## Planejamento da disciplina (7)

- **Aula 8**
  - Elementos básicos da orientação a objetos (Parte II).
  - Arquitetura de software do montador relocável para o simulador MVN.
  - Exercícios. Implementação de partes do montador relocável em Java. Programas de teste em linguagem simbólica do montador relocável.
- **Aula 9**
  - Arquitetura de software do ligador e relocador para a MVN.
  - Exercícios. Implementação de partes do ligador e relocador em Java. Programas de teste do ligador e do relocador em linguagem simbólica do montador relocável.

## Planejamento da disciplina (7)

- **Aula 10**
  - **Exercício.** Implementar a segunda parte de um monitor *batch* elementar para a MVN, em linguagem simbólica do montador relocável.
- **Aula 11**
  - **Exercício.** Implementar a terceira parte de um monitor *batch* elementar para a MVN, em linguagem simbólica do montador relocável.
- **Aula 12**
  - 2a. Prova
- **Aula 13**
  - Prova de recuperação

## Máquina de Turing (1)

- **Máquina de Turing:** modelo de computação proposto pelo inglês Alan M. Turing em 1936.



Alan M. Turing, disponível em  
[http://en.wikipedia.org/wiki/Alan\\_Turing](http://en.wikipedia.org/wiki/Alan_Turing)



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

Professores:  
Anarosa A. F. Brandão  
Marcos A. S. S. Brandão  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

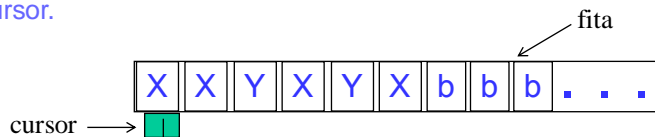
Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

13

## Máquina de Turing (2)

- Uma **Máquina de Turing** compõe-se de:
  - Uma fita infinita, composta de células, cada qual contendo um símbolo de um alfabeto finito disponível (a fita também implementa a memória externa da máquina).
  - Um cursor, que pode efetuar leitura ou escrita em uma célula, ou mover-se para a direita ou para a esquerda.
  - Uma máquina de estados finitos (MEF), que controla o cursor.



**Máquina de Estados Finitos (MEF)**



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

Professores:  
Anarosa A. F. Brandão  
Marcos A. S. S. Brandão  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

14

## Computação em uma MT (3)

- Inicialmente a fita contém somente a cadeia de entrada, com o cursor posicionado (por convenção) no início da cadeia (o restante da fita está em branco “b”).
- Para armazenar algo, a máquina o grava na fita.
- Se a máquina tentar mover o cursor para a esquerda, estando o cursor posicionado na primeira célula da fita, este não se moverá.
- As saídas **aceita** e **rejeita** são obtidas ao entrar a máquina nos estados de aceitação e rejeição, respectivamente.
- Se a máquina não entrar em um estado de aceitação ou de rejeição, continuará sua computação para sempre (loop infinito).



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

Professores:  
Anarosa A. F. Brandão  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

15

## MT como um Conjunto de Ações (4)

- Uma MT pode ser descrita por um conjunto de ações.
- Ações: (s, i, i', s', d) sendo:
  - s: estado corrente da MEF
  - i: símbolo que está sendo lido na fita
  - i': símbolo que é gravado na fita, no lugar de i
  - s': próximo estado da MEF
  - $d \in \{D, E\}$ , indicando que o cursor pode se mover para a Direita ou para a Esquerda.



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

Professores:  
Anarosa A. F. Brandão  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

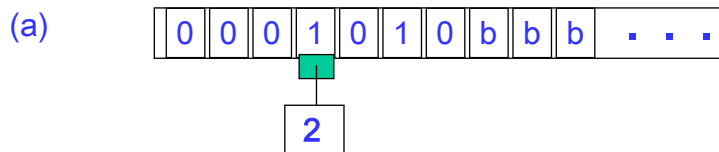
v.1.0 jun. 2011

16

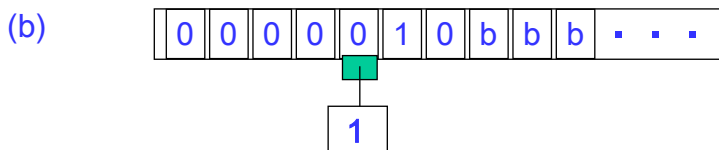
## MT como um Conjunto de Ações (5)

Exemplo:

Estando a máquina na situação (a):



executando a ação (2,1,0,1,D), a nova situação será (b):



(s, i, i', s', d)





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

Professores:  
Anarosa A. F. Brandão  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

17

## Observações sobre a Máquina de Turing

- Uma **Máquina de Turing** deve ser vista como um computador com um único programa fixo. Para alterar o programa, é preciso construir outra máquina.
- Algumas Máquinas de Turing servem como **reconhecedores de linguagens**, outras podem **computar funções**.
- É possível construir uma **Máquina de Turing Universal**, a qual simula a computação de Máquinas de Turing arbitrárias sobre entradas arbitrárias.
- Eliminadas suas limitações de recursos, um **computador moderno** pode ser visto como um dispositivo similar à Máquina de Turing Universal.



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

Professores:  
Anarosa A. F. Brandão  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

18

## Problemas Práticos da Máquina de Turing

- A Máquina de Turing se apresenta, portanto, através de um formalismo poderoso, com fita infinita e apenas quatro operações triviais: ler, gravar, avançar e recuar.
- Isso faz dela um dispositivo detalhista que oferece apenas uma **visão microscópica** da solução do problema que pretende resolver, não permitindo ao usuário usar abstrações mais expressivas.
- Embora a Máquina de Turing Universal permita uma espécie de programação, o seu código é extenso e a sua velocidade final de execução, muito baixa.



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

Professores:  
Anarosa A. F. Brandão  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

19

## 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  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

20

## 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  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

21

## Elementos da Arquitetura a Simular (1)

- Nesta disciplina 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  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

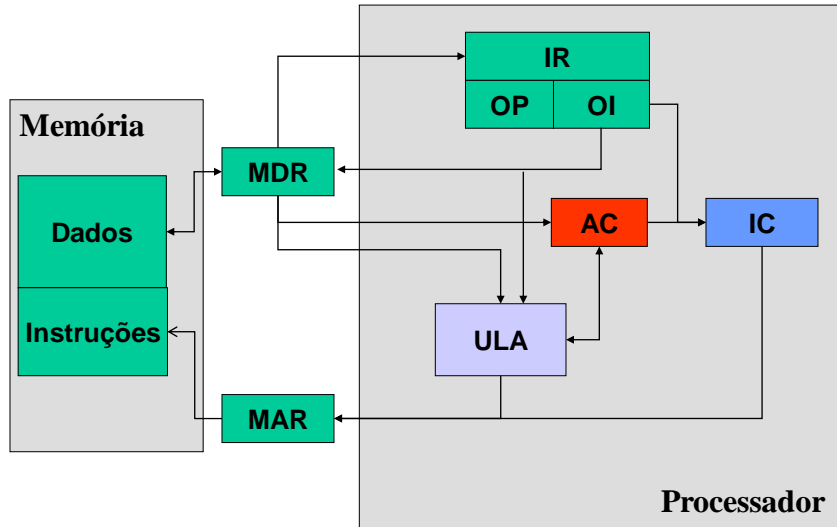
v.1.0 jun. 2011

22

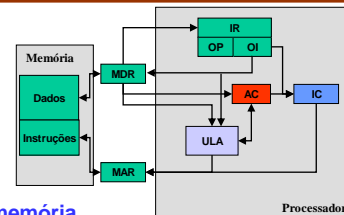
## Elementos da Arquitetura a Simular (2)

- 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.

## Elementos da Arquitetura a Simular (3)



## 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

IR (16 bits)	
OP (4 bits)	OI (12 bits)



## Funcionamento de um Simulador

Deve-se distinguir entre dois conceitos independentes na lógica de um simulador:

- **Comandos de controle do simulador:** esta parte do simulador independe da arquitetura do computador que se está simulando. Sua função é orientar a operação do programa simulador e 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 simulada. É ela que implementa um modelo da máquina simulada no nível de granularidade mais conveniente desejado em cada caso.



## Comandos de Controle do Simulador

- Conta-se com os seguintes comandos básicos de controle para o programa simulador:
  - **[INITIALIZE]** – atribui valores iniciais padrão a todos os elementos importantes do simulador e da arquitetura.
  - **[LOAD]** –carrega programas e dados para a memória da máquina simulada.
  - **[STEP]** – coloca o simulador no modo de operação passo a passo.
  - **[RUN]** – coloca o simulador no modo de operação contínuo.
  - **[EXECUTE]** – promove a execução do programa, conforme o modo de operação: execução contínua/uma instrução por vez.
  - **[SHOW]** – mostra 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  
Marcos A. Símplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

27

## Comandos de Controle do Simulador

```
Z:\>cd mun4
Z:\mun4>java -Dfile.encoding=cp850 -jar mun4.jar
Inicializacao padrao de dispositivos baseada em arquivo: disp.lst
MUN Inicializada

Escola Politécnica da Universidade de São Paulo
PCS2302/PCS2024 - Simulador da Máquina de von Neumann
MUN versão 4.5 (Agosto/2011) - Todos os direitos reservados
```

COMANDO	PARÂMETROS	OPERAÇÃO
i		Re-inicializa MUN
p	[arq]	Carrega programa para a memória
r	[addr] [regs]	Executa programa
b		Ativa/Desativa modo Debug
s		Manipula dispositivos de I/O
g		Lista conteúdo dos registradores
m	[inil] [fim] [arq]	Lista conteúdo da memória
h		Ajuda
x		Finaliza MUN e terminal

```
> -
```



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

Professores:  
Anarosa A. F. Brandão  
Marcos A. Símplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

28

## [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).



## 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.



## [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.

#### 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

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



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

Professores:  
Anarosa A. F. Brandão  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

31

## [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  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

32

## [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  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

33

## [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  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

34

## [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

$$AC := \text{int}(AC / MEM[OI])$$
$$IC := IC + 1$$



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

Professores:  
Anarosa A. F. Brandão  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

35

## [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  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

36

## [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  
Marcos A. S. S. Brandão  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

37

## [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)

$AC := \text{dado de entrada}$

$IC := IC + 1$



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

Professores:  
Anarosa A. F. Brandão  
Marcos A. S. S. Brandão  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

38

## [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

$\text{dado de saída} := AC$

(aciona dispositivo)

$IC := IC + 1$

### Registrador de instrução = F (supervisor call)

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

$IC := IC + 1$



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

Professores:  
Anarosa A. F. Brandão  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

39

## Diagrama de fluxo do Interpretador [detalhamento de EXECUTA]

Executa *uma* instrução

Determinar a próxima  
instrução a executar

Obter a instrução em  
MEM[IC] e guardar em IR

Decodificar a instrução:  
OP:=Código de operação  
OI:=Operando

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  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

40

## Conjunto de registradores da Máquina de von Neumann (MVN)

### Operações de Entrada e Saída

OP	Tipo	Dispositivo
----	------	-------------

OP  
Tipo

D (entrada) ou E (saída)  
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.

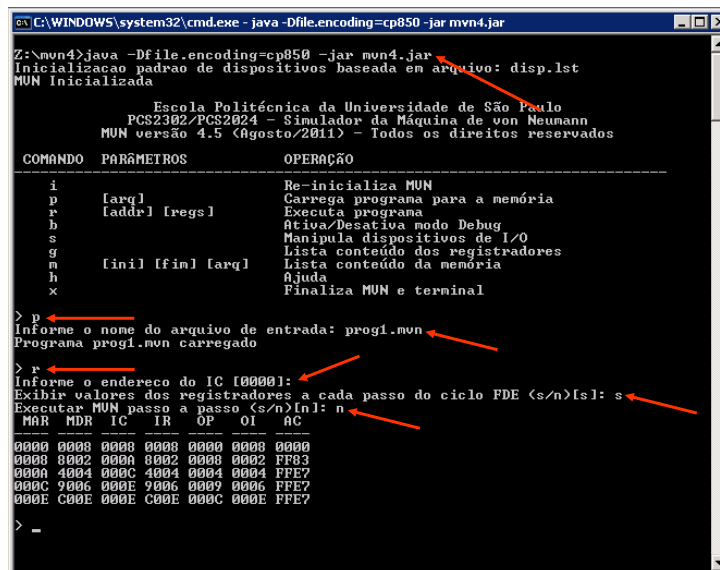
## Exemplo de Programa – Prog1 (1)

- 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.
0000 0008 ; Ponto de entrada: pulo para as instruções
; Constantes do programa
0002 FF83 ; A = 0xFF83 (-125)
0004 0064 ; B = 0x0064 (100)
; Variáveis do programa
0006 0000 ; RESULTADO deverá ser 0xFFE7 (-25)
; Instruções do programa
0008 8002 ; Carrega o valor de A no acumulador
000A 4004 ; Adiciona B ao conteúdo do acumulador
000C 9006 ; Armazena o resultado em RESULTADO
000E C00E ; Para em 0x000E
```

↑ endereços

## Execução de Programa – Prog1 (2)



```
C:\WINDOWS\system32\cmd.exe - java -Dfile.encoding=cp850 -jar mvn4.jar
Z:\mvn4>java -Dfile.encoding=cp850 -jar mvn4.jar
Inicializacao padrao de dispositivos baseada em arquivo: disp.lst
MUN Inicializada

Escola Politécnica da Universidade de São Paulo
PCS2302/PCS2024 - Simulador da Máquina de von Neumann
MUN versão 4.5 (Agosto/2011) - Todos os direitos reservados

COMANDO  PARÂMETROS      OPERAÇÃO
-----
i                Re-inicializa MUN
p [arg]          Carrega programa para a memória
r [addr] [regs]  Executa programa
b                Ativa/Desativa modo Debug
s                Manipula dispositivos de I/O
g                Lista conteúdo dos registradores
n [inil] [fin] [arg]  Lista conteúdo da memória
h                Ajuda
x                Finaliza MUN e terminal

> p
Informe o nome do arquivo de entrada: prog1.mvn
Programa prog1.mvn carregado

> r
Informe o endereço do IC [0000]:
Exibir valores dos registradores a cada passo do ciclo FDE (s/n)!: s
Executar MUN passo a passo (s/n)!: n
MAR  MDR  IC  IR  DP  OI  AC
0000 0008 0008 0008 0008 0008 0000
0008 8002 000A 8002 0008 0002 FF83
000A 4004 000C 4004 0004 0004 FFE7
000C 9006 000E 9006 0009 0006 FFE7
000E C00E 000E C00E 000C 000E FFE7

> -
```



## Exemplo de Programa – Prog2 (1)

- Problema: Desenvolver 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;
; }
;
0000 0010 ; Ponto de entrada: pulo para as instruções
; Constantes do programa
0002 0010 ; A = 0x0010 (16)
0004 0064 ; B = 0x0064 (100)
; Variáveis do programa
0006 0000 ; RESULTADO de subtrair() = 0xFFAC (-84)
```



## Exemplo de Programa – Prog2 (2)

```
; Programa principal
; Chamando SUBTRAIR(A, B)
0010 8002 ; Carrega o conteúdo de A no acumulador
0012 903C ; Armazena no parâmetro X
0014 8004 ; Carrega o conteúdo de B
0016 903E ; Armazena no parâmetro Y
0018 A040 ; Chama a sub-rotina SUBTRAIR
001A 9006 ; Armazena o resultado em RESULTADO
001C C01C ; Para em 0x01C
;
; Sub-rotina SUBTRAIR
; Parâmetros formais
003C 0000 ; X
003E 0000 ; Y
; Corpo da sub-rotina
0040 0000 ; Endereço de retorno
0042 803C ; Carrega o conteúdo de X
0044 503E ; Subtrai Y, resultado no ACUMULADOR
0046 B040 ; Retorna para o endereço contido em 0x040
```



## Execução de Programa – Prog2 (3)

```
> p
Informe o nome do arquivo de entrada: prog2.mvn
Programa prog2.mvn carregado

> r
Informe o endereço do IC [0000]: 0000
Exibir valores dos registradores a cada passo do ciclo FDE (s/n)[s]: s
Executar MUN passo a passo (s/n)[n]: n
```

MAR	MDR	IC	IR	OP	OI	AC
0000	0010	0010	0010	0000	0010	0000
0010	8002	0012	8002	0008	0002	0010
0012	903C	0014	903C	0009	003C	0010
0014	8004	0016	8004	0008	0004	0064
0016	903E	0018	903E	0009	003E	0064
0018	A040	0042	A040	000A	0040	0064
0042	803C	0044	803C	0008	003C	0010
0044	503E	0046	503E	0005	003E	FFAC
0046	B040	001A	B040	000B	0040	FFAC
001A	9006	001C	9006	0009	0006	FFAC
001C	C01C	001C	C01C	000C	001C	FFAC

```
>
```



## 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.



## Algumas práticas de programação (2)

- Suponha que se deseje ler uma sequência de dados armazenados na memória:

034C	0002
034E	0004
0350	0006
0352	0008
end.	dados

- Como fazer isto utilizando as instruções presentes nesta máquina de von Neumann?



## Algumas práticas de programação (3)

- Uma técnica de programação binária, que permite usar uma única instrução para percorrer mais de uma posição de memória, envolve a auto modificação do código. Veja neste exemplo:

End.	Instr.	Comentário
0100	8F00	Obtém o endereço de onde se deseja ler o dado
0102	4F02	Compõe o endereço com o código de operação LOAD
0104	9106	Guarda instrução montada para ser executada
0106	0000	Executa a instrução recém-montada
0108	.....	Usa o valor do acumulador e altera o conteúdo de 0F00 com o valor do próximo endereço da sequência.
.....		
015C	0100	Volta a repetir o procedimento.
.....		
0F00	034C	Endereço (034C) de onde se deseja ler o dado
0F02	8000	Código de operação LOAD, com operando 000

- Notar que o artifício da alteração do código pelo próprio programa, embora condenado pela engenharia de software, é a forma mais prática de percorrer sequências nesta máquina de von Neumann.





## Algumas práticas de programação (3a) Automodificação de código

```
; prog3.mvn
; Programa de ilustração de auto-modificação do código
; Lê uma sequência de dados contidos entre 034C a 0352
0000 0100 ; Ponto de entrada: pulo para as instruções
;
0100 8F00 ; Obtém o endereço de onde se deseja ler o dado
0102 4F02 ; Compõe o endereço com o código de operação LOAD
0104 9106 ; Guarda instrução montada para ser executada
0106 0000 ; Executa a instrução recém-montada
0108 8F00 ; Carrega o endereço da variável na lista
010A 4348 ; Soma com a constante 0002 (desloca uma posição)
010C 9F00 ; Altera o conteúdo de 0F00 com o novo endereço
010E 5F04 ; Subtrai com o endereço de parada
0110 1114 ; Se zero, condição de parada, salta para fora
0112 0100 ; Se não zero, volta para o início
0114 C114 ; Termina o programa
```

Continua no próximo slide...



## Algumas práticas de programação (3b) Automodificação de código

```
;
;
0348 0002 ; Constante 0002 (ADDR+1)
;
;Lista de valores a serem lidos (variáveis)
034C 0002
034E 0004
0350 0006
0352 0008
;
0F00 034C ; Endereço (034C) de onde se deseja ler o dado
0F02 8000 ; Código de operação LOAD, com operando 000
0F04 0354 ; Último endereço a ser lido + 1 (0352 + 0002)
```



## Algumas práticas de programação (3c) execução Prog3.mvn

```
Programa prog3.mvn carregado
> r
Informe o endereço do IC [0000]: 0000
Exibir valores dos registradores a cada passo do ciclo FDE (s/n)[s]: s
Executar MUN passo a passo (s/n)[n]: n
MAR  MDR  IC   IR   OP   OI   AC
0000 0100 0100 0100 0000 0100 0000
0100 8F00 0102 8F00 0008 0F00 034C
0102 4F02 0104 4F02 0004 0F02 034C
0104 9106 0106 9106 0009 0106 034C
0106 034C 0108 034C 0008 034C 0002
0108 8F00 010A 8F00 0008 0F00 034C
010A 4348 010C 4348 0004 0348 034E
010C 2F00 010E 2F00 0009 0F00 034E
010E 5F04 0110 5F04 0005 0F04 FFFA
0110 1114 0112 1114 0001 0114 FFFA
0112 0100 0100 0100 0000 0100 FFFA
0100 8F00 0102 8F00 0008 0F00 034E
0102 4F02 0104 4F02 0004 0F02 034E
0104 9106 0106 9106 0009 0106 034E
0106 034E 0108 034E 0008 034E 0004
0108 8F00 010A 8F00 0008 0F00 034E
010A 4348 010C 4348 0004 0348 0350
010C 2F00 010E 2F00 0009 0F00 0350
010E 5F04 0110 5F04 0005 0F04 FFFC
0110 1114 0112 1114 0001 0114 FFFC
0112 0100 0100 0100 0000 0100 FFFC
0100 8F00 0102 8F00 0008 0F00 0350
0102 4F02 0104 4F02 0004 0F02 0350
0104 9106 0106 9106 0009 0106 0350
0106 0350 0108 0350 0008 0350 0006
0108 8F00 010A 8F00 0008 0F00 0350
010A 4348 010C 4348 0004 0348 0352
010C 2F00 010E 2F00 0009 0F00 0352
010E 5F04 0110 5F04 0005 0F04 FFFE
0110 1114 0112 1114 0001 0114 FFFE
0112 0100 0100 0100 0000 0100 FFFE
0100 8F00 0102 8F00 0008 0F00 0352
0102 4F02 0104 4F02 0004 0F02 0352
0104 9106 0106 9106 0009 0106 0352
0106 0352 0108 0352 0008 0352 0008
0108 8F00 010A 8F00 0008 0F00 0352
010A 4348 010C 4348 0004 0348 0354
010C 2F00 010E 2F00 0009 0F00 0354
010E 5F04 0110 5F04 0005 0F04 0000
0110 1114 0114 1114 0001 0114 0000
0114 C114 0114 C114 000C 0114 0000
```

- Monta instrução
- Dado no AC
- Atualiza endereço
- Cond. de parada



## Algumas práticas de programação (4)

- Incrementos e decrementos de variáveis devem ser feitos somando-se ou subtraindo-se as constantes desejadas (tipicamente 1 ou 2) às variáveis alvo.
- 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  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

53

## Algumas práticas de programação (5)

- O simulador tem suporte para 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**.
  - A estratégia típica é reservar os endereços mais baixos para programas e os endereços mais altos para a área comum de dados, constantes, tabelas, etc. Por exemplo, no simulador, um mapa simples pode reservar os endereços 0x0000 – 0x0DFF (3584 bytes) para programas principais e sub-rotinas e os endereços a partir de 0x0E00 (512 bytes) para a área comum.
  - Na primeira parte da disciplina, em que os programas são carregados em endereços absolutos (fixos) da memória, pode-se, para simplificar, não dividir a memória. No entanto, os **programas deverão ser carregados nos endereços mais baixos**. Os exemplos da primeira parte da disciplina adotam um dos estilos clássicos de programação em “código de máquina” quando não há divisão da memória.



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

Professores:  
Anarosa A. F. Brandão  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

54

## Algumas práticas de programação (6)

- Projete sempre no papel seus programas 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.
- 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  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

## 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, 1996
- 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, 1980
- Graham, R. M. *Principles of Systems Programming* John Wiley & Sons, 1975
- Gust, P. *Introduction to Machine and Assembly Language Programming* Prentice Hall, 1985
- 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, 1972
- 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

55



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

Professores:  
Anarosa A. F. Brandão  
Marcos A. Simplicio Jr.  
Reginaldo Arakaki  
Paulo S. Muniz Silva  
© 2011

Aula 1:

Introdução  
Máq. von Neumann

Autores:

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

Reestruturação:  
Paulo S. Muniz Silva

v.1.0 jun. 2011

## Referências Bibliográficas

Costa, A. H. R., Sichman, J. S., Tori, R., Brandão, A. A. F..  
*Material didático da disciplina PCS2214 –  
Fundamentos da Engenharia de Computação I*,  
PCS/EPUSP, São Paulo. 2010-2011.

Sipser, M. *Introduction to the Theory of Computation*.  
(2o. Edition) Course Technology, Boston, MA. 2005.

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)**

56