


 <small>DEPARTAMENTO DE ENGENHARIA DE COMUNICAÇÃO E SISTEMAS DIGITAIS</small> PCS 2302/2024 Laboratório de Fundamentos da Eng.de Computação	PCS-2302 / PCS-2024 Lab. de Fundamentos de Eng. de Computação
	Aula 02 Montador Absoluto Professores: Marcos A. Simplício Junior Paulo Sergio Muniz Silva



Aula 02:
Montador Absoluto
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 Ago. 2012

1


   <small>DEPARTAMENTO DE ENGENHARIA DE COMUNICAÇÃO E SISTEMAS DIGITAIS</small> PCS 2302/2024 Laboratório de Fundamentos da Eng.de Computação	Roteiro
	<ol style="list-style-type: none">1. Linguagem Simbólica para o simulador MVN2. Montador absoluto para o simulador MVN: esquema geral, estruturas de Dados e algoritmos típicos3. Descrição da implementação do montador absoluto.

Aula 02:
Montador Absoluto
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 Ago. 2012

2

DEPARTAMENTO DE ENGENHARIA DE
COMUNICAÇÃO E SISTEMAS DIGITAIS



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

Aula 02:
Montador Absoluto

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 Ago. 2012


Construção de Programas em Linguagem de Máquina (1)

- Escrever um programa usando diretamente codificação binária não é uma tarefa simples, e tampouco agradável.
- Entretanto, foi este tipo de codificação que permitiu a construção dos primeiros programas no mundo (e também na disciplina).
- Naturalmente, se um programa é muito grande ou se lida com diversas estruturas complexas (listas, etc.), a sua codificação se torna ainda mais difícil e sujeita a erros.

3

DEPARTAMENTO DE ENGENHARIA DE
COMUNICAÇÃO E SISTEMAS DIGITAIS



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

Aula 02:
Montador Absoluto

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 Ago. 2012

Construção de Programas em Linguagem de Máquina (2)

- Por conta disso, torna-se imprescindível construir alguma **abstração** que facilite a programação e a verificação dos programas.
- A primeira ideia, mais natural, é utilizar o modelo de máquina existente e, a partir dele, definir nomes (mnemônicos) para cada instrução da máquina.
- Posteriormente, verifica-se que somente isso não basta, pois é necessário lidar com os endereços dentro de um programa (rótulos, operandos, subrotinas), com a reserva de espaço para tabelas, com valores constantes.
- Enfim, é necessário definir uma **linguagem simbólica**.

4

DEPARTAMENTO DE ENGENHARIA DE
COMUNICAÇÃO E SISTEMAS DIGITAIS
PCS
PCS 2302/2024
Laboratório de
Fundamentos da
Eng.de Computação

Linguagem Simbólica

- Uma instrução de máquina tem usualmente o aspecto seguinte em sua imagem mnemônica:

0012 JZ 042 ; (042) sendo 0012=rótulo, JZ=mnemônico, e 042=operando numérico

- A mesma instrução, em linguagem simbólica, pode ser escrita com ou sem um rótulo simbólico, e pode também referenciar um operando através de um rótulo simbólico ou numérico:

Q JZ R ; Q=rótulo JZ=mnemônico R=operando simbólico

JZ R ; rótulo omitido JZ=mnemônico R=operando simbólico

Q JZ /0042 ; Q=rótulo JZ=mnemônico 0042=operando numérico

JZ /0042 ; rótulo omitido JZ=mnemônico 0042=operando numérico

- Convenciona-se que sempre o primeiro elemento da linha é um rótulo; caso o rótulo seja omitido, deverá haver uma instrução
- Entre os elementos de uma linha deve haver ao menos um espaço
- Cada linha deve conter uma instrução/pseudo-instrução completa
- À direita de um ponto-e-vírgula, todo texto é ignorado (=comentário)
- Mnemônicos e significado das pseudo-instruções:

- @ (Operando numérico: define endereço da instrução seguinte)
- \$ (Reserva de área de dados)
- # (Final físico do texto-fonte. Operando=endereço de execução)
- K (Constante. Operando numérico = valor da constante, em hexadecimal)
- /<valor> (valor numérico em hexadecimal)

Aula 02:



Montador Absoluto

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 Ago. 2012

DEPARTAMENTO DE ENGENHARIA DE
COMUNICAÇÃO E SISTEMAS DIGITAIS
PCS
PCS 2302/2024
Laboratório de
Fundamentos da
Eng.de Computação

Exemplo de programa em linguagem simbólica

O programa abaixo, que foi dado como exemplo na aula 1:

```

0100      8F00  Obtém o endereço para onde se deseja mover o dado
0102      4F02  Compõe o endereço com o código de operação Move
0104      9106  Guarda instrução montada para executar em seguida
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, para outro endereço.
.....
0F00      034C  Endereço (34C) de onde se le o dado
0F02      8000  Código de operação LOAD, com operando 000

```

codificado em linguagem simbólica, fica com o seguinte aspecto:

```

@ /0100 ; @=origem do código 0100=posição de memória (em hexa)
P LD E ; P=rótulo LD=load E=endereço simbólico da constante 034C
+ R ; +=add R=rótulo de onde está uma instrução Load 0000
MM X ; MM=move X=endereço da instrução seguinte
X K /0000 ; reservado para guardar a instrução recém-montada
...
JP P ; JP=jump (desvio) P=rótulo da primeira instrução deste programa
...
E K /034C ; E=rótulo K=constante 034C=operando numérico, em hexadecimal
R LD /0000 ; R=rótulo LD=load 0000=operando zero
# P ; #=final físico P=rótulo da primeira instrução a ser executada

```

Aula 02:

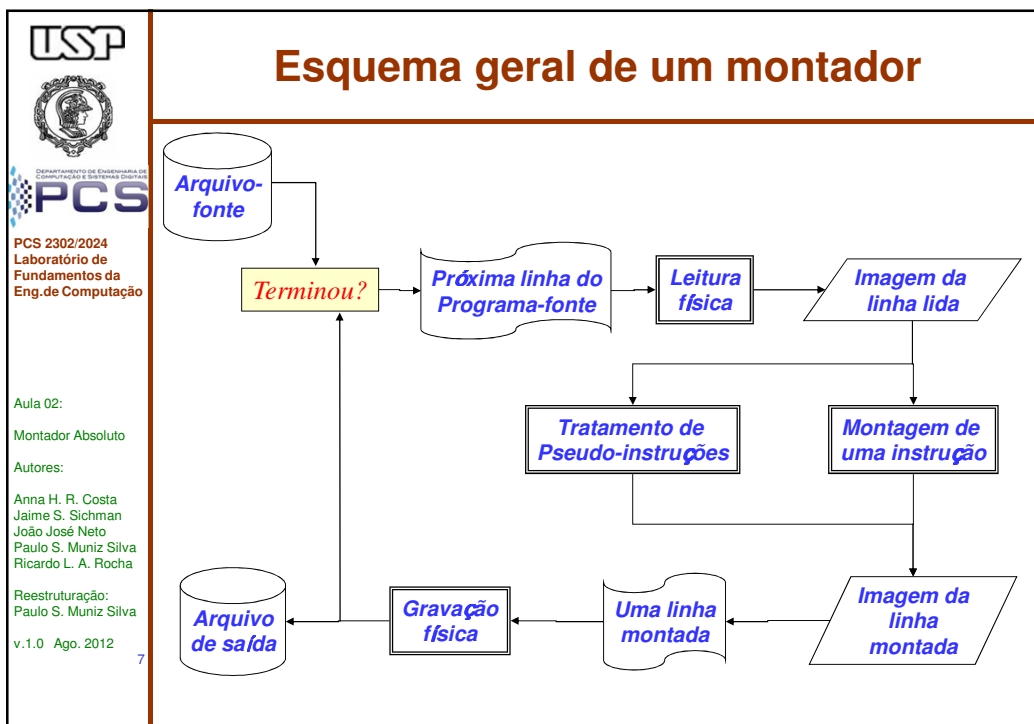
Montador Absoluto

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 Ago. 2012



USP



DEPARTAMENTO DE ENGENHARIA DE
COMUNICAÇÃO E SISTEMAS DIGITAIS

PCS

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

Aula 02:
Montador Absoluto




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 Ago. 2012

Construção de um Montador

- Para a construção de um montador, pressupõe-se que sejam tratadas as seguintes questões:
 - **definição das instruções:** determinar os mnemônicos que as representam simbolicamente;
 - **definição das pseudo-instruções:** determinar os mnemônicos que as representam, bem como sua função para o montador
- Durante a execução de um montador, pressupõe-se que sejam resolvidos os seguintes problemas:
 - **alocação dos rótulos:** determinar qual será o endereço efetivo de um nome encontrado (é necessária uma **tabela de símbolos**);
 - **geração de código:** gerar um arquivo com o código correspondente em linguagem de máquina
- Para cumprir esta tarefa é necessário completar, em primeiro lugar, as definições dos mnemônicos (instruções e pseudo-instruções), para se pensar posteriormente, nos algoritmos.

<div>   </div> <div>  </div> <div> <p>PCS 2302/2024 Laboratório de Fundamentos da Eng.de Computação</p> </div> <div> <p>Aula 02: Montador Absoluto</p> </div> <div> <p>Autores: Anna H. R. Costa Jaime S. Sichman João José Neto Paulo S. Muniz Silva Ricardo L. A. Rocha</p> </div> <div> <p>Reestruturação: Paulo S. Muniz Silva</p> </div> <div> <p>v.1.0 Ago. 2012</p> </div>	<h2>Tabela de mnemônicos para as instruções da MVN (de 2 caracteres)</h2>			
	<p>Operação 0 Jump Mnemônico JP</p>	<p>Operação 1 Jump if Zero Mnemônico JZ</p>	<p>Operação 2 Jump if Negative Mnemônico JN</p>	<p>Operação 3 Load Value Mnemônico LV</p>
	<p>Operação 4 Add Mnemônico +</p>	<p>Operação 5 Subtract Mnemônico –</p>	<p>Operação 6 Multiply Mnemônico *</p>	<p>Operação 7 Divide Mnemônico /</p>
	<p>Operação 8 Load Mnemônico LD</p>	<p>Operação 9 Move to Memory Mnemônico MM</p>	<p>Operação A Subroutine Call Mnemônico SC</p>	<p>Operação B Return from Sub. Mnemônico RS</p>
	<p>Operação C Halt Machine Mnemônico HM</p>	<p>Operação D Get Data Mnemônico GD</p>	<p>Operação E Put Data Mnemônico PD</p>	<p>Operação F Operating System Mnemônico OS</p>

<div>   </div> <div>  </div> <div> <p>PCS 2302/2024 Laboratório de Fundamentos da Eng.de Computação</p> </div> <div> <p>Aula 02: Montador Absoluto</p> </div> <div> <p>Autores: Anna H. R. Costa Jaime S. Sichman João José Neto Paulo S. Muniz Silva Ricardo L. A. Rocha</p> </div> <div> <p>Reestruturação: Paulo S. Muniz Silva</p> </div> <div> <p>v.1.0 Ago. 2012</p> </div>	<h2>Mnemônicos das pseudo-Instruções (1)</h2>	
	<ul style="list-style-type: none"> Mais precisamente, as pseudo-instruções usadas no montador absoluto são as seguintes: <ul style="list-style-type: none"> @ : Recebe um operando numérico, define o endereço da instrução seguinte; K : Constante, o operando numérico tem o valor da constante (em hexadecimal); \$: Reserva de área de dados, o operando numérico define o tamanho da área a ser reservada; # : Final físico do texto fonte. 	

DEPARTAMENTO DE ENGENHARIA DE
COMUNICAÇÃO E SISTEMAS DIGITAIS

PCS

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

Aula 02:
Montador Absoluto

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 Ago. 2012

Mnemônicos das pseudo-Instruções (2)

- Os operandos numéricos podem ter vários formatos.
 - Os exemplos que seguem são traduzidos numa palavra com o valor 03E8 (hexadecimal).
 - /<valor>: Valor em hexadecimal. Ex. /03E8;
 - =<valor>: Valor em decimal. Ex. =1000;
 - @<valor>: Valor em octal. Ex. @1750.
 - #<valor>: Valor em binário. Ex. #1111101000.
 - O exemplo que segue é traduzido numa palavra com o valor 4552 (hexadecimal).
 - ‘<símbolo ASCII><símbolo ASCII>. Ex: ‘ER

11

DEPARTAMENTO DE ENGENHARIA DE
COMUNICAÇÃO E SISTEMAS DIGITAIS

PCS

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

Aula 02:
Montador Absoluto

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 Ago. 2012


Formas de Construção de um Montador

- Há mais de uma forma de construir um Montador. Pelo menos duas são imediatas:
 - Montador de um passo:
 - Lê o código fonte uma única vez;
 - Armazena dinamicamente os rótulos não definidos em uma lista de pendências;
 - Gera o código para cada linha de entrada completamente definida;
 - Resolve uma pendência caso a linha de entrada inicie com um rótulo pendente;
 - Ao final, completa as linhas de código que ainda não haviam sido completamente definidas, resolvendo todos os rótulos pendentes.
 - Montador de dois passos:
 - Lê o código fonte da primeira vez;
 - Num primeiro passo, trata todas as linhas apenas para resolver os endereços dos rótulos e tratar as pseudo-instruções;
 - Lê novamente o código fonte num segundo passo para gerar o código correspondente ao programa

12



DEPARTAMENTO DE ENGENHARIA DE
COMUNICAÇÃO E SISTEMAS DIGITAIS



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

Aula 02:
Montador Absoluto

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 Ago. 2012

Executando o Montador


- Para a execução do montador

```
java -cp MLR.jar montador.MvnAsm [<arquivo asm>]
```
- Exemplo

```
java -cp MLR.jar montador.MvnAsm test.asm
```



DEPARTAMENTO DE ENGENHARIA DE
COMUNICAÇÃO E SISTEMAS DIGITAIS



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

Aula 02:
Montador Absoluto



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 Ago. 2012

Anexo

Estrutura interna do montador

DEPARTAMENTO DE ENGENHARIA DE
COMUNICAÇÃO E SISTEMAS DIGITAIS
PCS
PCS 2302/2024
Laboratório de
Fundamentos da
Eng.de Computação

Aula 02:

Montador Absoluto

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 Ago. 2012

Estruturas de Dados do Montador (1)

- O montador precisará de um conjunto de estruturas de dados que o permitirão conduzir a tarefa. Dentro deste conjunto, há as seguintes estruturas de dados:
 - **locationCounter**: define a localização atual (endereço corrente) de execução.
 - **Tabela de instruções**: define as instruções válidas (símbolo e valor).
 - **Tabela de pseudo-instruções**: define as pseudo-instruções válidas (símbolo e valor).
 - **Tabela de símbolos**: permite armazenar e recuperar os rótulos (símbolo e endereço real).

15

DEPARTAMENTO DE ENGENHARIA DE
COMUNICAÇÃO E SISTEMAS DIGITAIS
PCS
PCS 2302/2024
Laboratório de
Fundamentos da
Eng.de Computação

Aula 02:

Montador Absoluto

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 Ago. 2012

Estruturas de Dados do Montador (2)

- Além destas estruturas, o montador utiliza um conjunto de arquivos (um de entrada e pelo menos dois de saída). Pode ser necessário gerar o texto objeto em algum formato específico, para que um programa *loader* possa carregá-lo na memória.
- Pode-se, ainda, armazenar o conteúdo do texto fonte durante o passo 1 para facilitar a execução do passo 2.

16

DEPARTAMENTO DE ENGENHARIA DE
COMUNICAÇÃO E SISTEMAS DIGITAIS

PCS

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

Aula 02:
Montador Absoluto



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 Ago. 2012

Construção do Montador

- Nesta disciplina foi escolhido realizar um montador de dois passos. As principais ações a serem realizadas em cada um dos dois passos do montador são:
 - **Passo1:** O objetivo é definir os símbolos encontrados, sejam eles rótulos encontrados antes das instruções, ou ainda rótulos de destino de alguma instrução. Para isso deve-se:
 - Manter atualizado o endereço de execução corrente, chamado de **locationCounter**.
 - Armazenar os valores dos símbolos (rótulos) na Tabela de Símbolos (**TS**) para uso posterior no passo 2.
 - Processar as pseudo-instruções.
 - **Passo2:** O objetivo é gerar o código objeto e possivelmente um arquivo de listagem contendo além do código objeto, o texto fonte à direita do código objeto. Para isso, este passo deve:
 - Recuperar os valores dos símbolos (da **TS**).
 - Gerar as instruções.
 - Processar as pseudo-instruções.
- O projeto seguiu o enfoque OO e sua implementação foi realizada na linguagem Java.

DEPARTAMENTO DE ENGENHARIA DE
COMUNICAÇÃO E SISTEMAS DIGITAIS

PCS

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

Aula 02:
Montador Absoluto

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 Ago. 2012

Lógica Geral do Montador Absoluto

- O algoritmo geral que serviu de guia para a implementação do montador absoluto foi:


```

begin
  Marque o endereço inicial de geração de código como 0 /* locationCounter */;
  abra o arquivo com o programa;
  while não encontra fim de arquivo do
    | Passo1: leia uma linha preenchendo a Tabela de Símbolos (TS);
  end
  reinicie o arquivo;
  while não encontra fim de arquivo do
    | Passo2: begin
    |   leia uma linha;
    |   monte e gere o código objeto correspondente usando a TS;
    |   escreva o código gerado nos arquivos de saída (em arquivo de saída carregável);
    | end
  end
  fecha os arquivos;
end
            
```

Leitura e Tratamento das Linhas

```

begin
  leia a linha até o final <EOL>;
  separe os tokens da linha (ou seja, palavras);
  while houver tokens do
    | if não é comentário then
      | | armazena temporariamente o token;
    end
  end
  if há token armazenado then
    | analisa a linha;
  end
  incremente o contador de linhas;
  pegue a próxima linha;
end




```

Análise de uma Linha

```

begin
  if há 3 tokens then
    // há um rótulo e deve ser tratado;
    if rótulo não está definido na TS then
      | defina o valor do rótulo na TS como sendo o locationCounter;
      | incremente o locationCounter de 2 (porque a abstração da MVN é Word);
    else
      | erro! o rótulo já estava definido;
    end
  end
  Teste a validade do restante da linha (operador e operando);
end

```

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

Aula 02:
 Montador Absoluto
 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 Ago. 2012




Teste de Operador e de Operando

```

begin
  | verifique se o operador é válido (se é instrução ou pseudo);
  | verifique se o operando é válido (se é número ou rótulo);
end

```

21

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

Aula 02:
 Montador Absoluto
 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 Ago. 2012



Montagem e Geração de Código

```

begin
  | if argumento é instrução then
  |   pegue e monte o código correspondente a partir da Tabela de instruções;
  | else
  |   | if é pseudo then
  |   |   trate a pseudo corretamente;
  |   | end
  | end
end

```

22

Departamento de Engenharia de
 Computação e Sistemas Digitais
PCS
 PCS 2302/2024
 Laboratório de
 Fundamentos da
 Eng.de Computação

Aula 02:

Montador Absoluto

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 Ago. 2012

Montador Absoluto: exmontabs.asm

```

1      @      /0100      ; Endereço da próxima instrução
2  C0      K      /0100
3  C1      K      /0111
4  C2      K      /0122
5  C3      K      /0133
6  C4      K      /0144
7  C5      K      /0155
8  C6      K      /0166
9  C7      K      /0177
10 C8      K      /0188
11 C9      K      /0199
12 CA      K      /01AA
13 CB      K      /01BB
14 CC      K      /01CC
15 CD      K      /01DD
16 CE      K      /01EE
17 CF      K      /01FF
18 CG      K      'ER'      ; ASCII "ER" = 4552
19 CH      K      'R0'      ; ASCII "R0" = 524F
20 CJ      K      =1000      ; 1000 em decimal
21 GK      K      /03E8      ; 3E8 em hexadecimal
22 GL      K      01750      ; 1750 em octal
23 GÇ      K      #1111101000 ; 1111101000 em binário
24
25      @      /0000
26      JP      INICIO
27 DADOS    $      /A      ; Reserva 10 palavras a partir de DADOS
28 INICIO    JP      C0
29          JZ      C1
30          JN      C2
31          LV      C3
32          +      C4
33          -      C5
34          *      C6
35          /      C7
36          LD      C8
37          HM      C9
38          SC      CA
39          RS      CB
40          HM      CC
41          GD      CD
42          PD      CE
43          DS      CF
44          #      INICIO
      
```

23

Departamento de Engenharia de
 Computação e Sistemas Digitais
PCS
 PCS 2302/2024
 Laboratório de
 Fundamentos da
 Eng.de Computação

Aula 02:

Montador Absoluto

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 Ago. 2012

Montador Absoluto: exmontabs.asm

- Arquivo de saída exmontabs.lst (1)

```

1      ;      @      /0100      ; Endereço da próxima instrução
2  0100 0100 ; C0      K      /0100
3  0102 0111 ; C1      K      /0111
4  0104 0122 ; C2      K      /0122
5  0106 0133 ; C3      K      /0133
6  0108 0144 ; C4      K      /0144
7  010a 0155 ; C5      K      /0155
8  010c 0166 ; C6      K      /0166
9  010e 0177 ; C7      K      /0177
10 0110 0188 ; C8      K      /0188
11 0112 0199 ; C9      K      /0199
12 0114 01aa ; CA      K      /01AA
13 0116 01bb ; CB      K      /01BB
14 0118 01cc ; CC      K      /01CC
15 011a 01dd ; CD      K      /01DD
16 011c 01ee ; CE      K      /01EE
17 011e 01ff ; CF      K      /01FF
18 0120 4552 ; GG      K      'ER'      ; ASCII "ER" = 4552
19 0122 524F ; GH      K      'R0'      ; ASCII "R0" = 524F
20 0124 03e8 ; GJ      K      =1000      ; 1000 em decimal
21 0126 03e8 ; GK      K      /03E8      ; 3E8 em hexadecimal
22 0128 03e8 ; GL      K      01750      ; 1750 em octal
23 012a 03e8 ; GÇ      K      #1111101000 ; 1111101000 em binário
24
      
```

24

Montador Absoluto: exmontabs.asm

• Arquivo de saída exmontabs.lst (2)

```

25      ;                                @      /0000
26      0000 0016 ;                      JP      INICIO
27      0002 0000 ; DADOS                $      /A      ; Reserva 10 palavras a partir de DADOS 0
28      0004 0000 ; DADOS                $      /A      ; Reserva 10 palavras a partir de DADOS 2
29      0006 0000 ; DADOS                $      /A      ; Reserva 10 palavras a partir de DADOS 4
30      0008 0000 ; DADOS                $      /A      ; Reserva 10 palavras a partir de DADOS 6
31      000a 0000 ; DADOS                $      /A      ; Reserva 10 palavras a partir de DADOS 8
32      000c 0000 ; DADOS                $      /A      ; Reserva 10 palavras a partir de DADOS a
33      000e 0000 ; DADOS                $      /A      ; Reserva 10 palavras a partir de DADOS c
34      0010 0000 ; DADOS                $      /A      ; Reserva 10 palavras a partir de DADOS e
35      0012 0000 ; DADOS                $      /A      ; Reserva 10 palavras a partir de DADOS 10
36      0014 0000 ; DADOS                $      /A      ; Reserva 10 palavras a partir de DADOS 12
37      ; DADOS                $      /A      ; Reserva 10 palavras a partir de DADOS
38      0016 0100 ; INICIO                JP      C0
39      0018 1102 ;                      JZ      C1
40      001a 2104 ;                      JN      C2
41      001c 3106 ;                      LU      C3
42      001e 4108 ;                      +      C4
43      0020 510a ;                      -      C5
44      0022 610c ;                      *      C6
45      0024 710e ;                      /      C7
46      0026 8110 ;                      LD      C8
47      0028 9112 ;                      HM      C9
48      002a a114 ;                      SC      CA
49      002c b116 ;                      RS      CB
50      002e c118 ;                      HM      CC
51      0030 d11a ;                      GD      CD
52      0032 e11c ;                      PD      CE
53      0034 f11e ;                      DS      CF
    
```