 PCS 2302/2024 Laboratório de Fundamentos da Eng. de Computação Professores: Anarosa A.F. Brandão Jaime S. Sichman Reginaldo Arakaki Ricardo L.A. Rocha © 2009 Aula 2: Máquina de Von Neumann Exercícios Autores: Jaime S. Sichman João José Neto Paulo S. Muniz Silva Ricardo L. A. Rocha Revisores: Diego Queiroz Tiago Matos v. 1.4 ago 2009	<div>PCS-2302 / PCS-2024 Lab. de Fundamentos de Eng. de Computação</div> <div>Aula 02</div> <div>Máquina de Von Neumann Exercícios</div> <div>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</div>
	1



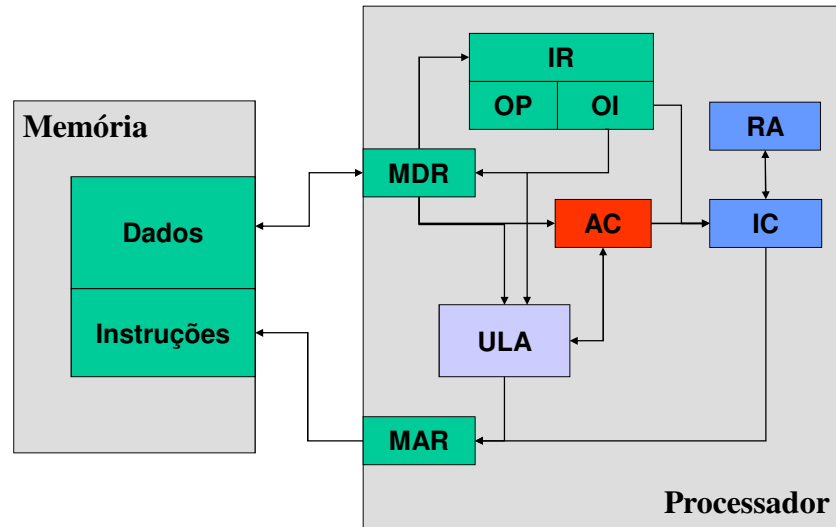
  PCS 2302/2024 Laboratório de Fundamentos da Eng. de Computação Professores: Anarosa A.F. Brandão Jaime S. Sichman Reginaldo Arakaki Ricardo L.A. Rocha © 2009 Aula 2: Máquina de Von Neumann Exercícios Autores: Jaime S. Sichman João José Neto Paulo S. Muniz Silva Ricardo L. A. Rocha Revisores: Diego Queiroz Tiago Matos v. 1.4 ago 2009	<div>Roteiro</div> <div><ol style="list-style-type: none">Máquina de Von Neumann<ol style="list-style-type: none">RecapitulaçãoSequência de DadosParte Experimental<ol style="list-style-type: none">Desenvolvimento de código de máquina MVN</div>
	2

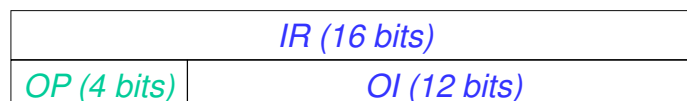



Diagrama da Arquitetura a Simular



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 da próxima instrução
IR	Registrador de instrução
OP	Registrador de código de operação
OI	Registrador de operando de instrução
RA	Registrador de endereço de retorno
AC	Acumulador



<div> <div> <div>USP</div> <div>  </div> </div> <div> <p>PCS 2302/2024 Laboratório de Fundamentos da Eng.de Computação</p> <p>Professores: Anarosa A.F. Brandão Jaime S. Sichman Reginaldo Arakaki Ricardo L.A. Rocha © 2009</p> <p>Aula 2:</p> <p>Máquina de Von Neumann Exercícios</p> <p>Autores:</p> <p>Jaime S. Sichman João José Neto Paulo S. Muniz Silva Ricardo L. A. Rocha</p> <p>Revisores:</p> <p>Diego Queiroz Tiago Matos</p> <p>v. 1.4 ago 2009</p> </div> </div>			<div> <div>Conjunto de instruções da Máquina de Von Neumann (MVN)</div> </div>		
			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 slide seguinte		
			(**) por ora, este operando (tipo da chamada) é irrelevante, e esta instrução nada faz.		

USP




Diagrama de fluxo do Interpretador [detalhamento de EXECUTA]

Executa *uma* 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
© 2009

Aula 2:

Máquina de Von
Neumann
Exercícios

Autores:

Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:

Diego Queiroz
Tiago Matos

v. 1.4 ago 2009

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 se não IC:=IC+1
2	Se AC<0 então IC:=OI se não 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	RA:=IC; IC:=OI
B	AC:=MEM[OI] ; IC:=RA
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



Operações de Entrada e Saída da MVN

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.



Como visitar uma seqüência de dados

- Suponha que se deseje visitar (alterar) uma seqüência de dados na memória:

0F00

0F02

0F04

0F06

0002
0004
0006
0008

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



Como visitar uma seqüência de dados

- 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 para onde se deseja armazenar o dado
0102	4F02	Compõe o endereço com o código de operação STORE
0104	9106	Guarda instrução montada para executar em seguida
0106	9000	Executa a instrução recém-montada
0108	Provavelmente, o código seguinte altera o conteúdo de 0F00
.....		
015C	0100	Volta a repetir o procedimento, para outro endereço.
.....		
0F00	034C	Endereço (34C) para onde se deseja armazenar o dado
0F02	9000	Código de operação STORE, 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 seqüências nesta máquina de Von Neumann.



Exercícios

TYGXXA02E01_09

Desenvolva duas sub-rotinas cujas finalidades são:

- **OP2MNEM**: converte um número dado $0 \leq \text{OPCODE} \leq 15$, em MNEM, mnemônico formado de dois caracteres ASCII, conforme a tabela de mnemônicos fornecida adiante;
- **MNEM2OP**: faz a conversão oposta, transformando um mnemônico MNEM válido, dado como dois caracteres ASCII de 7 bits, no número $0 \leq \text{OPCODE} \leq 15$ correspondente, conforme a tabela de mnemônicos acima mencionada.

- Um pequeno programa principal deve ilustrar o uso das duas sub-rotinas.

- Atenção: ambos os parâmetros MNEM e OPCODE são representados como inteiros, ocupando cada qual dois bytes de memória

Comente devidamente os programas desenvolvidos.



Exercícios

Exemplo:

- Dado o código de operação 1, nas posições OPCODE e OPCODE+1 de memória, contendo respectivamente os bytes hexadecimais 00 e 01, respectivamente, **OP2MNEM** retorna como resultado o par de letras JZ (mnemônico de JUMP if Zero), cujos códigos ASCII são 4A e 5A, respectivamente. Em outras palavras, na posição MNEM retornará o byte hexadecimal 4A e na posição MNEM+1, o byte hexadecimal 4E.
- De forma inversa, dado o mnemônico JZ em (MNEM, MNEM+1), ou seja, o par de bytes (4A, 4E), a sub-rotina **MNEM2OP** retornará em (OPCODE, OPCODE+1) o par de bytes (00, 01)



Tabela de mnemônicos para a MVN (de 2 caracteres)

Operação 0 Jump Mnemônico JP	Operação 1 Jump if Zero Mnemônico JZ	Operação 2 Jump if Negative Mnemônico JN	Operação 3 Load Value Mnemônico LV
Operação 4 Add Mnemônico +	Operação 5 Subtract Mnemônico –	Operação 6 Multiply Mnemônico *	Operação 7 Divide Mnemônico /
Operação 8 Load Mnemônico LD	Operação 9 Move to Memory Mnemônico MM	Operação A Subroutine Call Mnemônico SC	Operação B Return from Sub. Mnemônico RS
Operação C Halt Machine Mnemônico HM	Operação D Get Data Mnemônico GD	Operação E Put Data Mnemônico PD	Operação F Operating System Mnemônico OS



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

Professores:
Anarosa A.F. Brandão
Jaime S. Sichman
Reginaldo Arakaki
Ricardo L.A. Rocha
© 2009

Aula 2:

Máquina de Von
Neumann
Exercícios

Autores:

Jaime S. Sichman
João José Neto
Paulo S. Muniz Silva
Ricardo L. A. Rocha

Revisores:

Diego Queiroz
Tiago Matos

v. 1.4 ago 2009

Tabela de caracteres ASCII (7 bits. Ex: "K" = 4b)

	0	1	2	3	4	5	6	7
0	NUL		SP	0	@	P	`	p
1			!	1	A	Q	a	q
2			"	2	B	R	b	r
3			#	3	C	S	c	s
4			\$	4	D	T	d	t
5			%	5	E	U	e	u
6			&	6	F	V	f	v
7	BEL		'	7	G	W	g	w
8			(8	H	X	h	x
9)	9	I	Y	i	y
a	LF		*	:	J	Z	j	z
b		ESC	+	;	K	[k	{
c			,	<	L	\	l	
d	CR		-	=	M]	m	}
e			.	>	N	^	n	~
f			/	?	O	_	o	DEL

13