Universidade Federal de Pelotas
Instituto de Física e Matemática
Departamento de Informática
Bacharelado em Ciência da Computação

Arquitetura e Organização de Computadores I

Aula 4

1. Projeto da Arquitetura e da Organização de um Computador: o Neander

Prof. José Luís Güntzel

guntzel@ufpel.edu.br

www.ufpel.edu.br/~guntzel/AOC1/AOC1.html

1. O Computador Hipotético Neander

A Arquitetura: conjunto de instruções

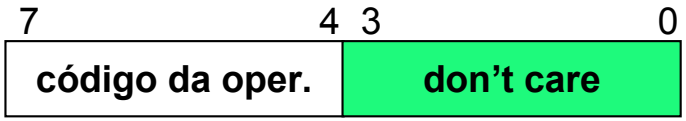
código	instrução	comentário
0000	NOP	Nenhuma operação
0001	STA end	MEM(end) □ AC
0010	LDA end	AC □ MEM(end)
0011	ADD end	AC □ MEM(end) + AC
0100	OR end	AC □ MEM(end) OR AC
0101	AND end	AC □ MEM(end) AND AC
0110	NOT	AC □ NOT AC
1000	JMP end	PC □ end
1001	JN end	IF N=1 THEN PC □ end
1010	JZ end	IF Z=1 THEN PC □ end
1111	HLT	pára processamento

1. O Computador Hipotético Neander

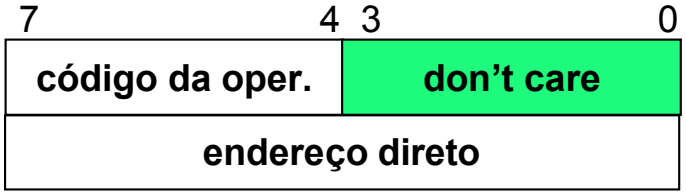
A Arquitetura: formato das instruções

As instruções do Neander possuem **um** ou **dois bytes** (ou seja, ocupam uma ou duas posições de memória)

Instruções com um byte:
NOP, NOT



Instruções com dois bytes:
STA, LDA, ADD, OR, AND, JMP, JN, JZ



1. O Computador Hipotético Neander

A Arquitetura: características gerais

- Largura de dados e endereços de 8 bits
- Dados representados em complemento de 2
- 1 acumulador de 8 bits (AC)
- 1 apontador de programa de 8 bits (PC)
- 1 registrador de estado com 2 códigos de condição: negativo (N) e zero (Z)

1. O Computador Hipotético Neander

A Organização: alguns elementos necessários

- Um registrador de 8 bits para servir de acumulador
- Um registrador de 8 bits para o PC (registrador-contador)
- Dois flip-flops: um para o código de condição N e outro para Z
- Uma memória de 256 posições (endereços) x 8 bits

1. O Computador Hipotético Neander

A Arquitetura: o ciclo de busca (fetch)

1. O Computador Hipotético Neander

Aquitetura/Organização: transferências entre regs.

A fase de busca: é igual para todas as instruções

RI ← MEM(PC)

PC ← PC + 1

- **Novo elemento é necessário: o registrador de instrução (RI)**
- **MEM(PC) corresponde a um acesso à memória, usando o conteúdo do PC como fonte do endereço**

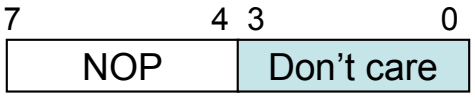
1. O Computador Hipotético Neander

Aquitetura/Organização: transferências entre regs.

Instrução NOP (nenhuma operação)

Simbólico: NOP

RT: -



Passos no nível RT:

Busca:	**RI ← MEM(PC)**
	PC ← PC + 1
Execução:	**nenhuma operação**

As transferências indicam quais caminhos de dados devem existir, mas não indicam os caminhos físicos reais entre os elementos (registradores e ULA)

1. O Computador Hipotético Neander

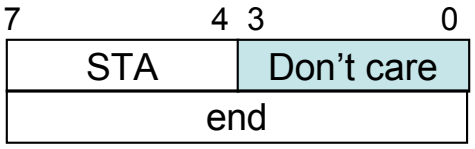
Aquitetura/Organização: transferências entre regs.

Instrução STA (armazena acumulador)

Simbólico: STA end

RT: MEM(end) ← AC

Passos no nível RT:



Busca:	RI ← MEM(PC)
	PC ← PC + 1
Execução:	end ← MEM(PC)
	PC ← PC + 1
	MEM(end) ← AC

1. O Computador Hipotético Neander

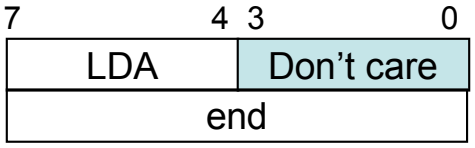
Aquitetura/Organização: transferências entre regs.

Instrução LDA (carrega acumulador)

Simbólico: LDA end

RT: AC ← MEM(end)

Passos no nível RT:



Busca:	RI ← MEM(PC)
	PC ← PC + 1
Execução:	end ← MEM(PC)
	PC ← PC + 1
	AC ← MEM(end); atualiza N e Z

1. O Computador Hipotético Neander

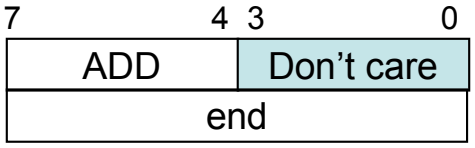
Aquitetura/Organização: transferências entre regs.

Instrução ADD (soma)

Simbólico: ADD end

RT: AC ← MEM(end) + AC

Passos no nível RT:



Busca:	RI ← MEM(PC)
	PC ← PC + 1
Execução:	end ← MEM(PC)
	PC ← PC + 1
	AC ← AC + MEM(end); atualiza N e Z

1. O Computador Hipotético Neander

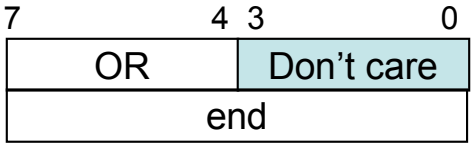
Aquitetura/Organização: transferências entre regs.

Instrução OR ("ou" lógico, bit a bit)

Simbólico: OR end

RT: AC ← MEM(end) OR AC

Passos no nível RT:



Busca:	RI ← MEM(PC)
	PC ← PC + 1
Execução:	end ← MEM(PC)
	PC ← PC + 1
	AC ← AC OR MEM(end); atualiza N e Z

1. O Computador Hipotético Neander

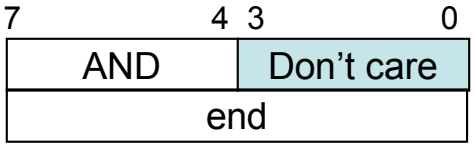
Aquitetura/Organização: transferências entre regs.

Instrução AND ("e" lógico, bit a bit)

Simbólico: AND end

RT: AC ← MEM(end) AND AC

Passos no nível RT:



Busca:	RI ← MEM(PC)
	PC ← PC + 1
Execução:	end ← MEM(PC)
	PC ← PC + 1
	AC ← AC AND MEM(end); atualiza N e Z

1. O Computador Hipotético Neander

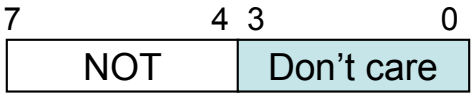
Aquitetura/Organização: transferências entre regs.

Instrução NOT (complementa acumulador)

Simbólico: NOT

RT: AC ← NOT AC

Passos no nível RT:



Busca:	RI ← MEM(PC)
	PC ← PC + 1
Execução:	AC ← NOT(AC); atualiza N e Z

1. O Computador Hipotético Neander

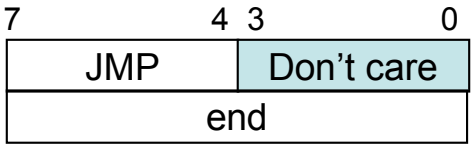
Aquitetura/Organização: transferências entre regs.

Instrução JMP (desvio incondicional - jump)

Simbólico: JMP end

RT: PC ← end

Passos no nível RT:



Busca:	RI ← MEM(PC)
	PC ← PC + 1
Execução:	end ← MEM(PC)
	PC ← end

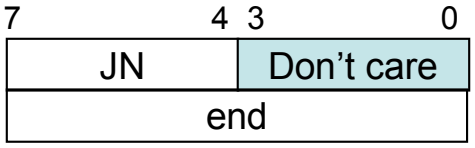
1. O Computador Hipotético Neander

Aquitetura/Organização: transferências entre regs.

Instrução JN (desvio condicional - jump on negative)

Simbólico: JN end

RT: IF N = 1 THEN PC ← end



Passos no nível RT:

Se N=1 (desvio ocorre)

Busca:	RI ← MEM(PC)
	PC ← PC + 1
Execução:	end ← MEM(PC)
	PC ← end

Se N=0 (desvio não ocorre)

Busca:	RI ← MEM(PC)
	PC ← PC + 1
Execução:	end ← MEM(PC)
	PC ← PC + 1

a rigor, desnecessário

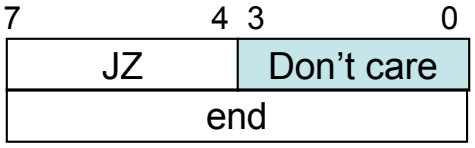
1. O Computador Hipotético Neander

Aquitetura/Organização: transferências entre regs.

Instrução JZ (desvio condicional - jump on zero)

Simbólico: JZ end

RT: IF Z = 1 THEN PC ← end



Passos no nível RT:

Se Z=1 (desvio ocorre)

Busca:	RI ← MEM(PC)
	PC ← PC + 1
Execução:	end ← MEM(PC)
	PC ← end

Se Z=0 (desvio não ocorre)

Busca:	RI ← MEM(PC)
	PC ← PC + 1
Execução:	end ← MEM(PC)
	PC ← PC + 1

a rigor, desnecessário

1. O Computador Hipotético Neander

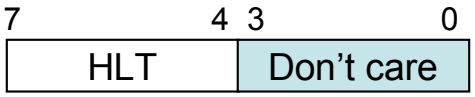
Aquitetura/Organização: transferências entre regs.

Instrução HLT (término de execução - halt)

Simbólico: HLT

RT: --

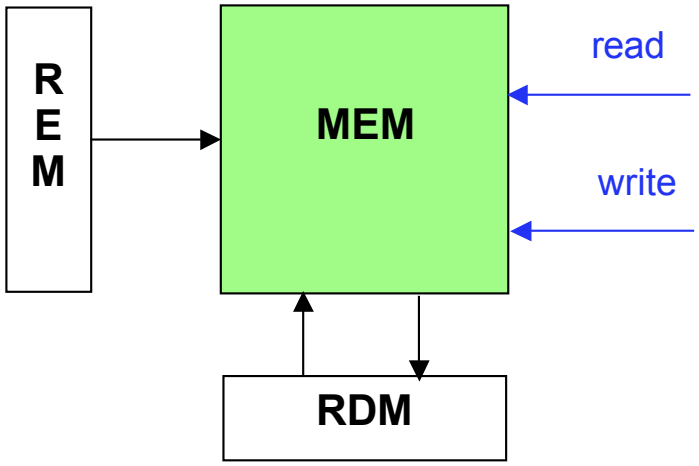
Passos no nível RT:



Busca:	**RI ← MEM(PC)**
	PC ← PC + 1
Execução:	**parar o processamento**

1. O Computador Hipotético Neander

Organização do Sistema de Memória



1. O Computador Hipotético Neander

Aquitetura/Organização

Operações com a memória

x ← MEM(y) descreve uma leitura da memória, que é realizada pelos seguintes passos:

1. **REM ← y**	copia **y** (que é um endereço) para o **REM**
2. **Read**	ativação de uma operação de leitura da memória
3. **x ← RDM**	copia o conteúdo de **RDM** para **x**

- **REM** é o registrador de endereços da memória
- **RDM** é o registrador de dados da memória

1. O Computador Hipotético Neander

Aquitetura/Organização

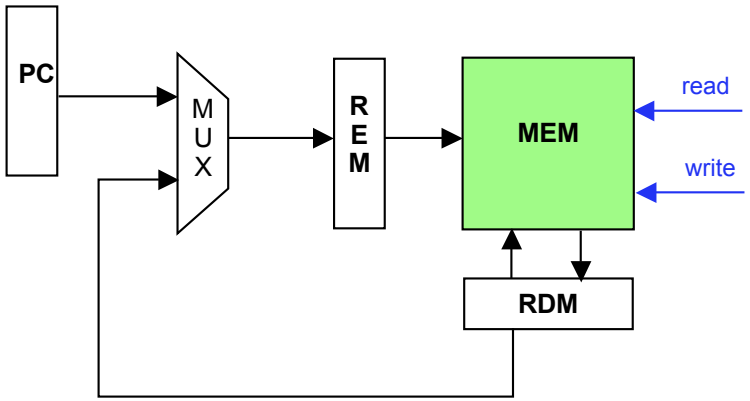
Operações com a memória

MEM(y) ← x descreve uma escrita da memória, que é realizada pelos seguintes passos:

1. REM ← y	copia y (que é um endereço) para o REM
2. RDM ← x	copia x (que é um dado) para o RDM
3. write	ativação de uma operação de escrita na memória

1. O Computador Hipotético Neander

Organização do Sistema de Memória



1. O Computador Hipotético Neander

Aquitetura/Organização

Operações com a memória

Observações - 1

- Após a leitura do PC, o conteúdo deste reg. deve ser incrementado, para apontar para a próxima posição
- O incremento do PC pode ser feito a qualquer instante após a transferência do PC para o REM, podendo ser feito em paralelo com outras operações
- Iremos assumir que o incremento do PC sempre é feito ao mesmo tempo que a operação na memória (read ou write)

1. O Computador Hipotético Neander

Aquitetura/Organização

Operações com a memória

Observações - 2

- **Um desvio condicional que não se realiza não necessita ler o valor do endereço de desvio. Ou seja, basta incrementar o PC**

1. O Computador Hipotético Neander

Aquitetura/Organização

Então, detalhando mais as transferências entre registradores…

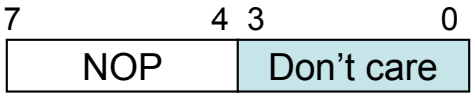
1. O Computador Hipotético Neander

Aquitetura/Organização: transferências entre regs.

Instrução NOP (nenhuma operação)

Simbólico: NOP

RT: -



Passos no nível RT:

Busca:	**REM ← PC**
	Read; PC ← PC + 1
	RI ← RDM
Execução:	**nenhuma operação**

1. O Computador Hipotético Neander

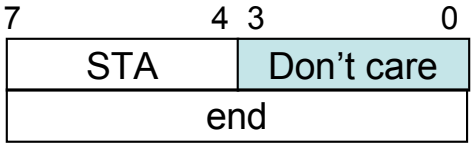
Aquitetura/Organização: transferências entre regs.

Instrução STA (armazena acumulador)

Simbólico: STA end

RT: MEM(end) ← AC

Passos no nível RT:



Busca:	REM ← PC
	Read; PC ← PC + 1
	RI ← RDM
Execução:	REM ← PC
	Read; PC ← PC + 1
	REM ← RDM
	RDM ← AC
	Write

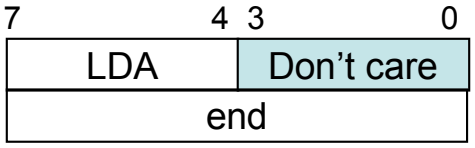
1. O Computador Hipotético Neander

Aquitetura/Organização: transferências entre regs.

Instrução LDA (carrega acumulador)

Simbólico: LDA end

RT: AC ← MEM(end)



Passos no nível RT:

Busca:	REM ← PC
	Read; PC ← PC + 1
	RI ← RDM
Execução:	REM ← PC
	Read; PC ← PC + 1
	REM ← RDM
	Read
	AC ← RDM; atualiza N e Z

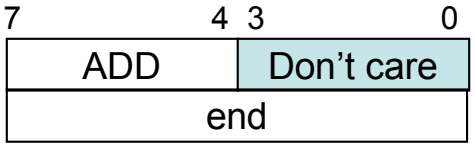
1. O Computador Hipotético Neander

Aquitetura/Organização: transferências entre regs.

Instrução ADD (soma)

Simbólico: ADD end

RT: AC ← MEM(end) + AC



Passos no nível RT:

Busca:	REM ← PC
	Read; PC ← PC + 1
	RI ← RDM
Execução:	REM ← PC
	Read; PC ← PC + 1
	REM ← RDM
	Read
	AC ← AC + RDM; atualiza N e Z

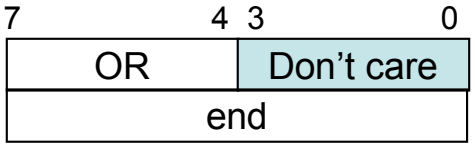
1. O Computador Hipotético Neander

Aquitetura/Organização: transferências entre regs.

Instrução OR ("ou" lógico, bit a bit)

Simbólico: OR end

RT: AC ← MEM(end) OR AC



Passos no nível RT:

Busca:	REM ← PC
	Read; PC ← PC + 1
	RI ← RDM
Execução:	REM ← PC
	Read; PC ← PC + 1
	REM ← RDM
	Read
	AC ← AC OR RDM; atualiza N e Z

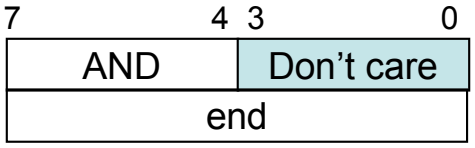
1. O Computador Hipotético Neander

Aquitetura/Organização: transferências entre regs.

Instrução AND ("e" lógico, bit a bit)

Simbólico: AND end

RT: AC ← MEM(end) AND AC



Passos no nível RT:

Busca:	REM ← PC
	Read; PC ← PC + 1
	RI ← RDM
Execução:	REM ← PC
	Read; PC ← PC + 1
	REM ← RDM
	Read
	AC ← AC AND RDM; atualiza N e Z

1. O Computador Hipotético Neander

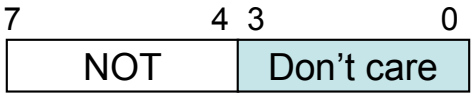
Aquitetura/Organização: transferências entre regs.

Instrução NOT (complementa acumulador)

Simbólico: NOT

RT: AC ← NOT AC

Passos no nível RT:



Busca:	REM ← PC
	Read; PC ← PC + 1
	RI ← RDM
Execução:	AC ← NOT(AC); atualiza N e Z

1. O Computador Hipotético Neander

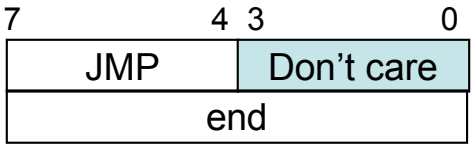
Aquitetura/Organização: transferências entre regs.

Instrução JMP (desvio incondicional - jump)

Simbólico: JMP end

RT: PC ← end

Passos no nível RT:



Busca:	REM ← PC
	Read; PC ← PC + 1
	RI ← RDM
Execução:	REM ← PC
	Read
	PC ← RDM

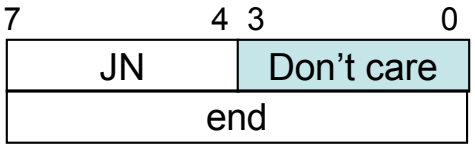
1. O Computador Hipotético Neander

Aquitetura/Organização: transferências entre regs.

Instrução JN (desvio condicional - jump on negative)

Simbólico: JN end

RT: IF N = 1 THEN PC ← end



Passos no nível RT:

Se N=1 (desvio ocorre)

Busca:	REM ← PC
	Read; PC ← PC + 1
	RI ← RDM
Execução:	REM ← PC
	Read
	PC ← RDM

Se N=0 (desvio não ocorre)

Busca:	REM ← PC
	Read; PC ← PC + 1
	RI ← RDM
Execução:	PC ← PC + 1

1. O Computador Hipotético Neander

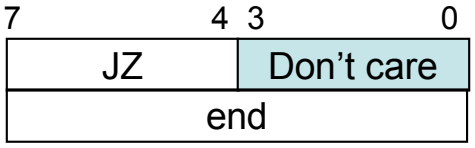
Aquitetura/Organização: transferências entre regs.

Instrução JZ (desvio condicional - jump on zero)

Simbólico: JZ end

RT: IF Z = 1 THEN PC ← end

Passos no nível RT:



Se Z=1 (desvio ocorre)

Busca:	REM ← PC
	Read; PC ← PC + 1
	RI ← RDM
Execução:	REM ← PC
	Read
	PC ← RDM

Se Z=0 (desvio não ocorre)

Busca:	REM ← PC
	Read; PC ← PC + 1
	RI ← RDM
Execução:	PC ← PC + 1

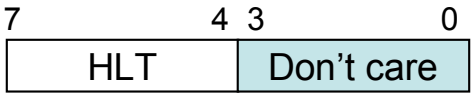
1. O Computador Hipotético Neander

Aquitetura/Organização: transferências entre regs.

Instrução HLT (término de execução - halt)

Simbólico: HLT

RT: --



Passos no nível RT:

Busca:	**REM ← PC**
	Read; PC ← PC + 1
	RI ← RDM
Execução:	**parar o processamento**