

# USP - ICMC - SSC

## SSC 0511 - Sist. Informação - 2o. Semestre 2014

# Disciplina de Organização de Computadores Digitais

**Prof. Fernando Santos Osório**

**Email: fosorio [at] { icmc. usp. br , gmail. com }**

**Página Pessoal: <http://www.icmc.usp.br/~fosorio/>**

**Material on-line: Wiki ICMC:**

**[http://wiki.icmc.usp.br/index.php/SSC-511-2014\(fosorio\)](http://wiki.icmc.usp.br/index.php/SSC-511-2014(fosorio))**



Lab. de Robótica Móvel



Centro de Robótica da USP

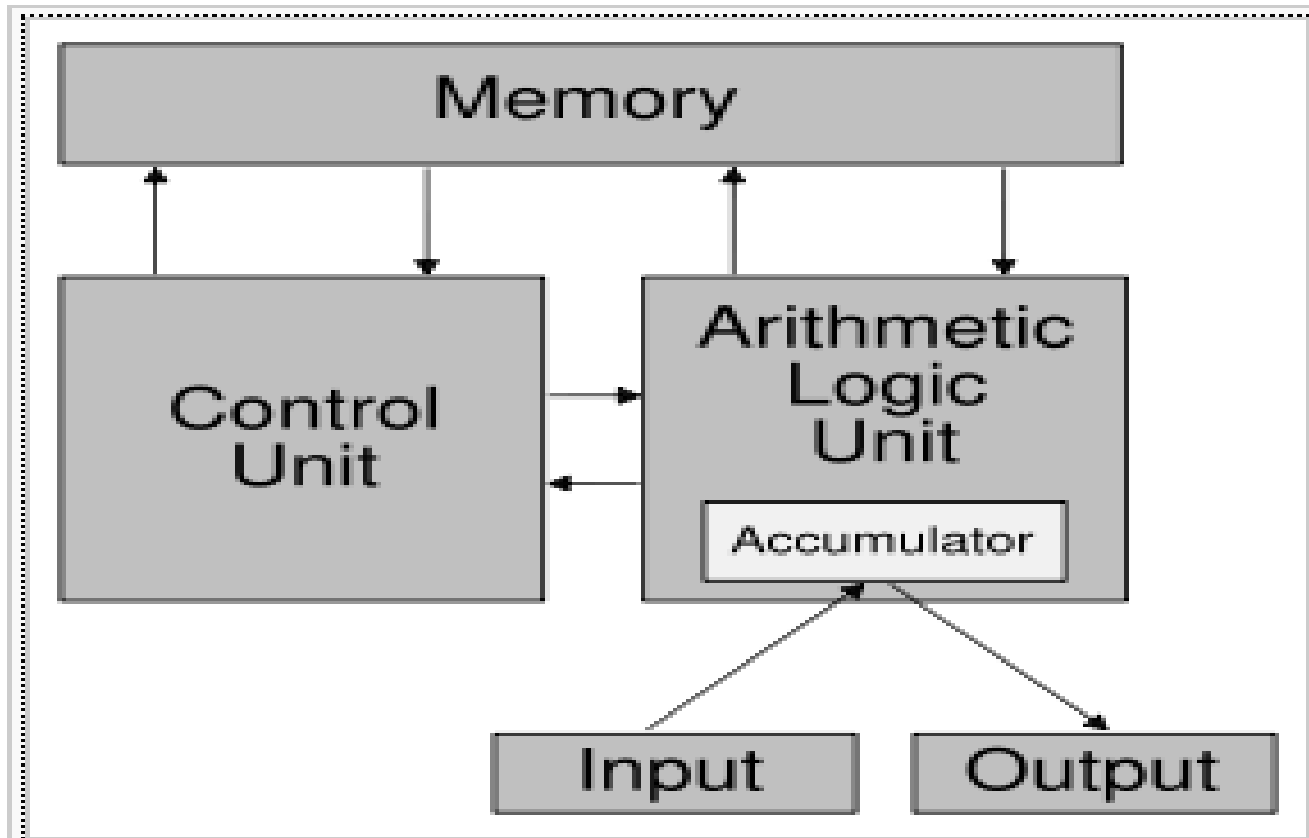
## **Agenda:**

- 1. Arquitetura de Von Neumann:  
ULA, Memória, Registradores, UC e E/S**
- 2. Memória – Conceitos Básicos**
  - Flip-Flops, Registradores e Contadores**
  - Registradores: Acumulador**
  - Registradores: Registro de Instrução**
  - Registradores: Contador de Programa**
  - Registradores: Flags**
  - Memória: Barramento e Acesso a Memória**
- 3. Micro-Processadores...**

# 1. Arquitetura de Von Neumann

## Arquitetura - Modelo Inicial:

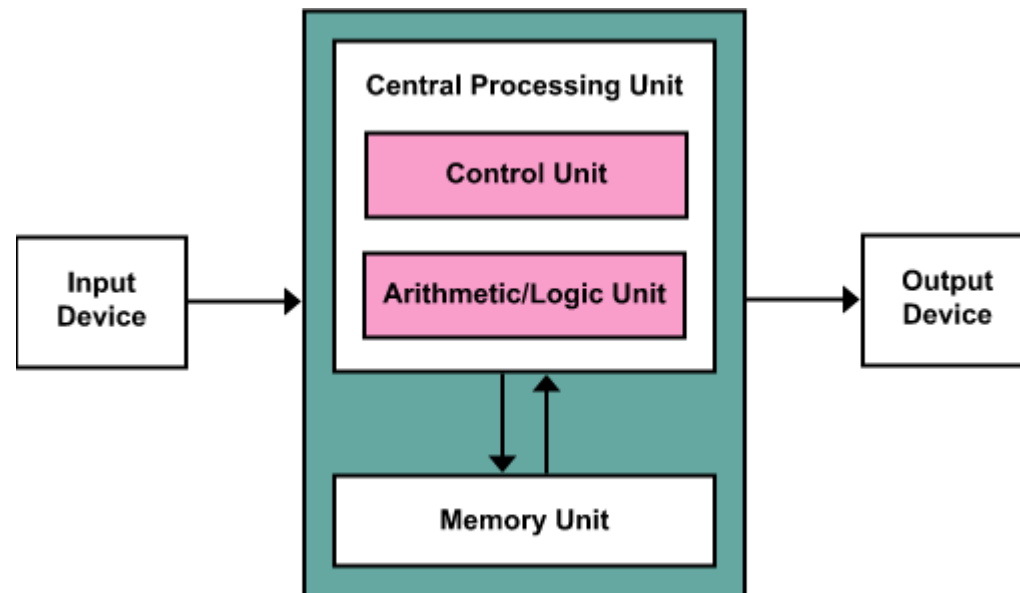
- Máquina de von Neumann



# 1. Arquitetura de Von Neumann

## Arquitetura - Modelo Inicial:

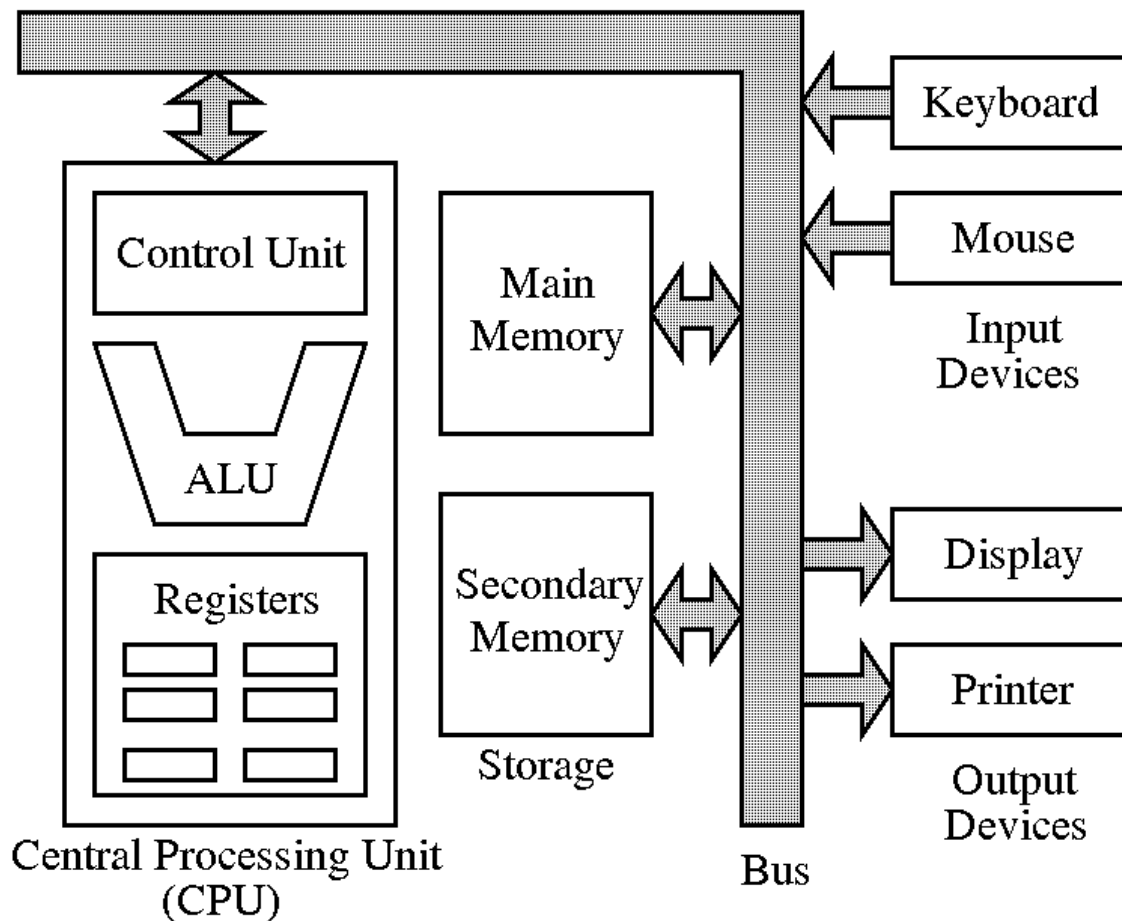
- Máquina de von Neumann



# 1. Arquitetura de Von Neumann

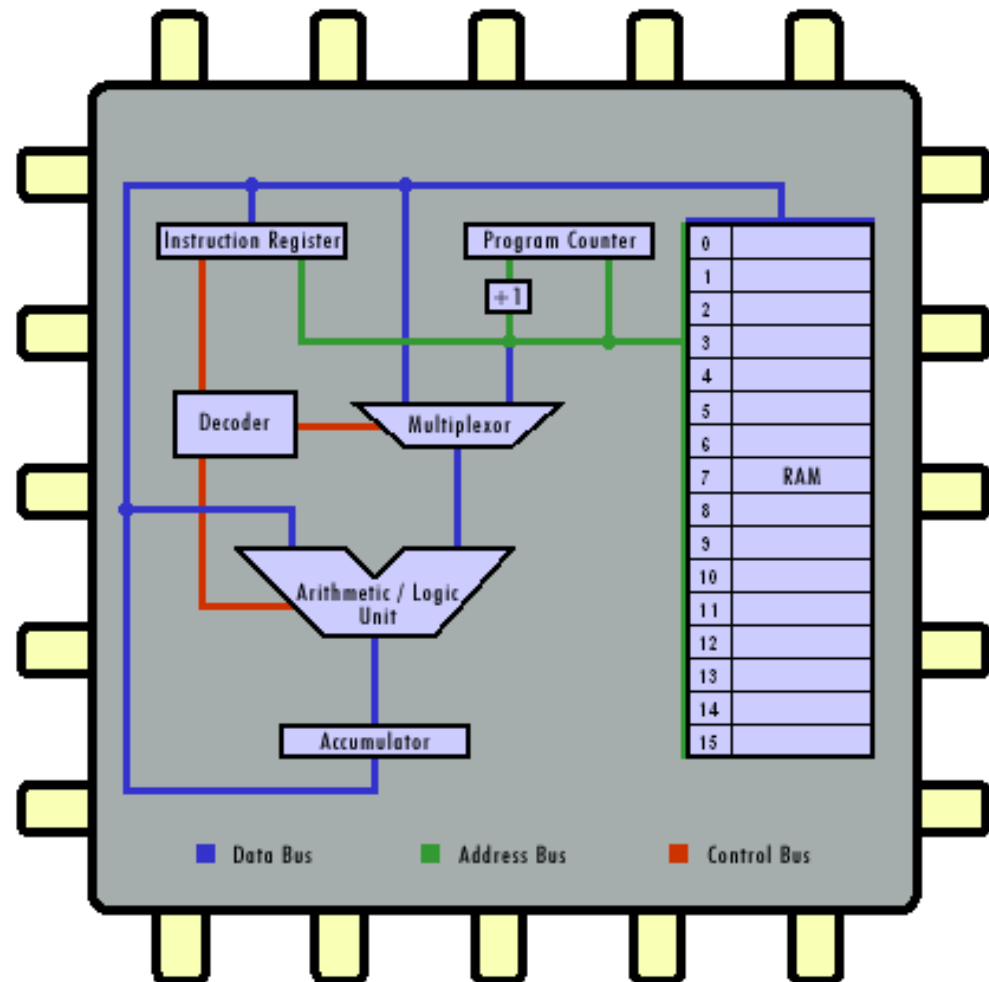
## Arquitetura - Modelo Inicial:

- Máquina de von Neumann



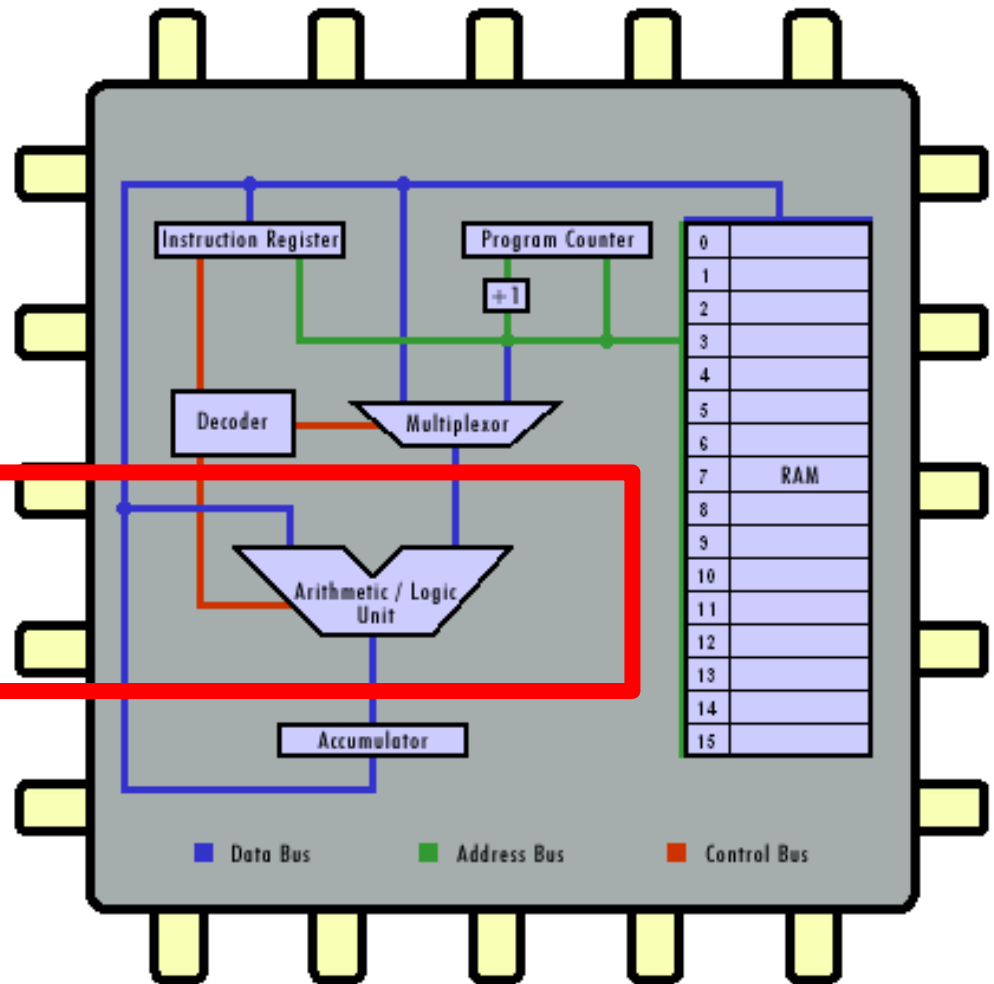
# 1. Arquitetura de Von Neumann

## CPU – Processador



## CPU – Processador

**ULA / ALU**  
Unidade Lógico-Aritmética



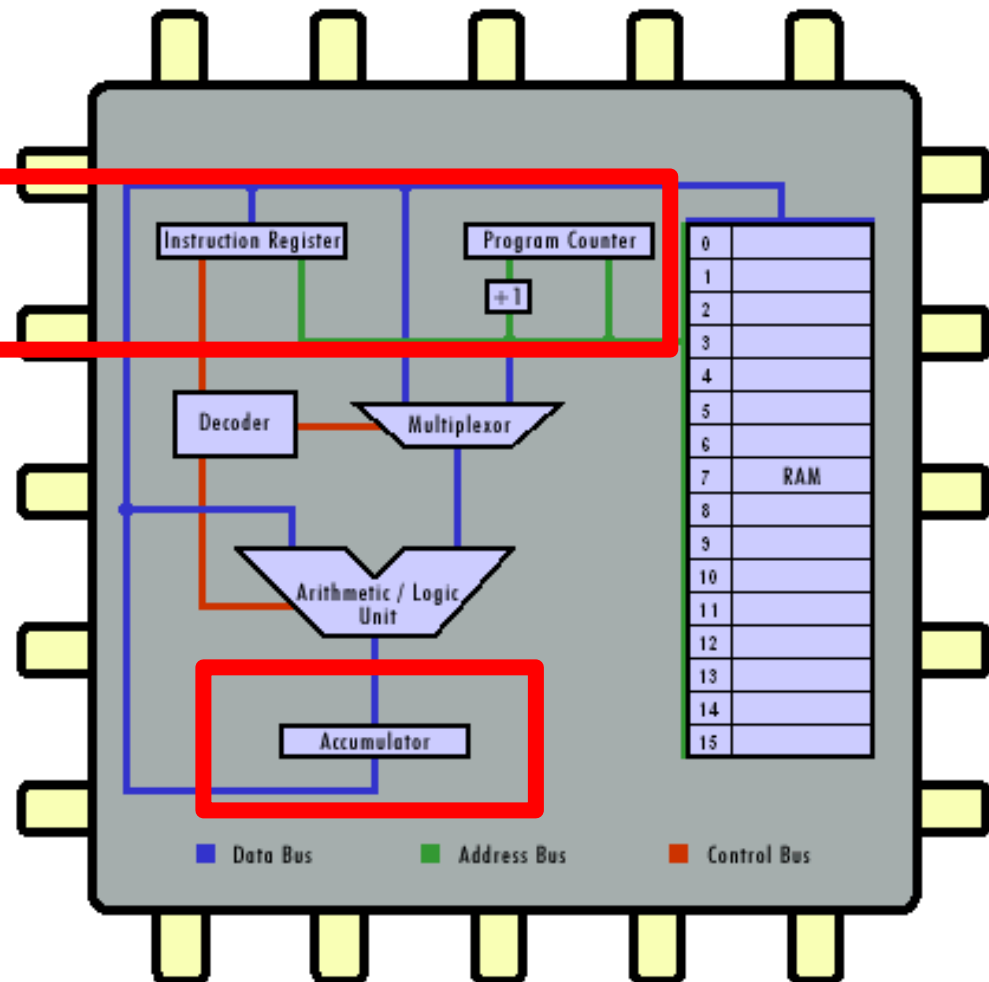
# 1. Arquitetura de Von Neumann

## CPU – Processador

Registradores

### Registradores

Program Counter: PC (CP)  
Instruction Register: IR (RI)  
Accumulator: Acc (AC)  
Status Register - Flags: Flags



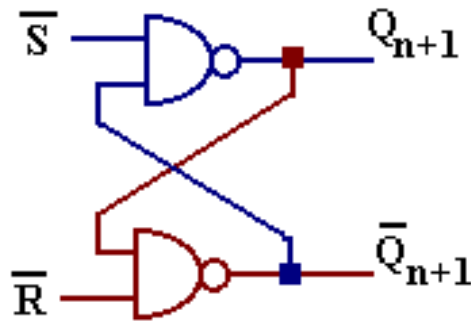


## 2. Memória – Conceitos Básicos

### FLIP-FLOP RS Básico

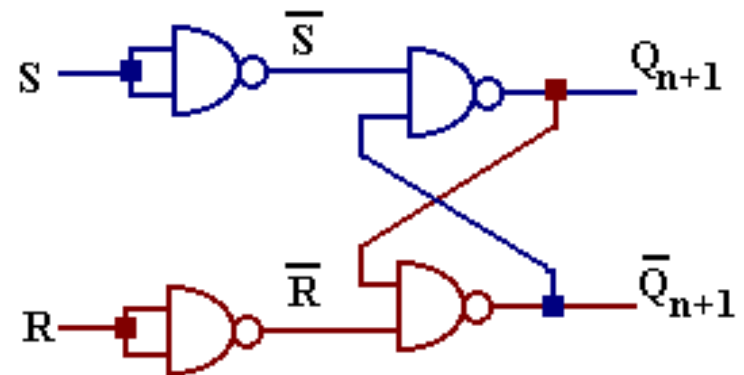
Memória de 1 bit

FLIP-FLOP RS



$\bar{S}$	$\bar{R}$	$Q_{n+1}$	$\bar{Q}_{n+1}$	
0	0	proibida		
0	1	1	0	Estado Set
1	0	0	1	Estado Reset
1	1	$Q_n$	$\bar{Q}_n$	Latch (memória)

FLIP-FLOP RS

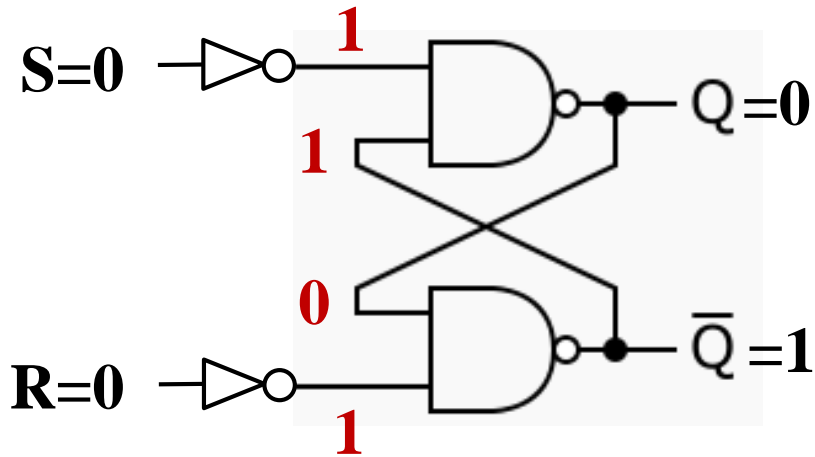


S	R	$Q_{n+1}$	$\bar{Q}_{n+1}$	
0	0	$Q_n$	$\bar{Q}_n$	Latch (memória)
0	1	0	1	Estado Reset
1	0	1	0	Estado Set
1	1	proibida		

## 2. Memória – Conceitos Básicos

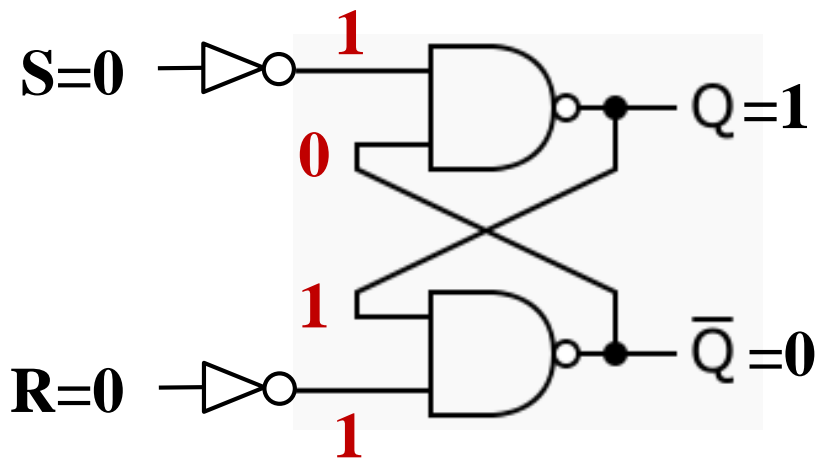
### FLIP-FLOP RS Básico

Memória de 1 bit



$$Q_{(t-1)} = 0$$

$$\bar{Q}_{(t-1)} = 1$$



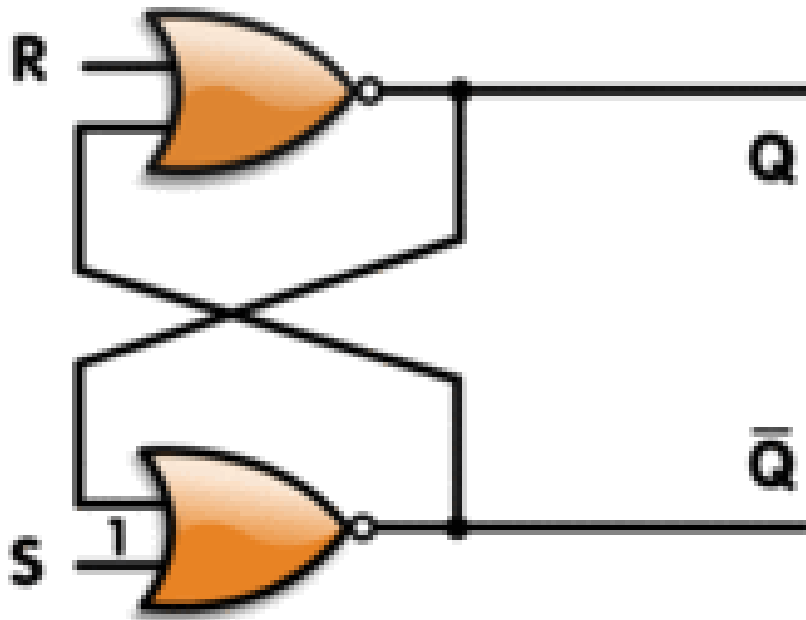
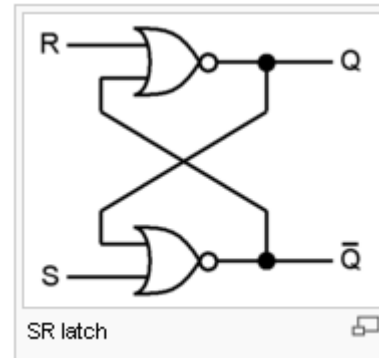
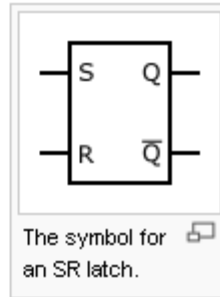
$$Q_{(t-1)} = 1$$

$$\bar{Q}_{(t-1)} = 0$$

## 2. Memória – Conceitos Básicos

### SR LATCH (FLIP-FLOP)

SR latch operation		
S	R	Action
0	0	Keep state
0	1	$Q = 0$
1	0	$Q = 1$
1	1	Restricted combination

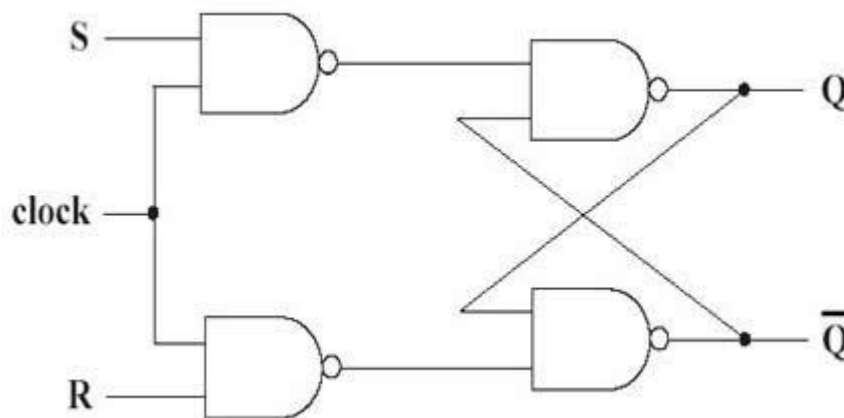
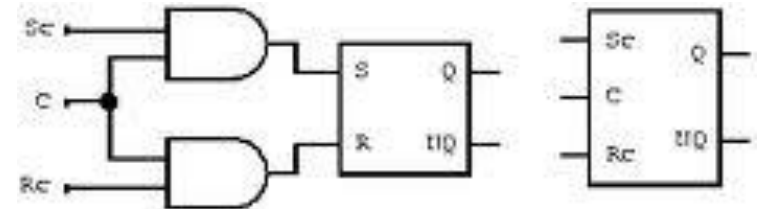
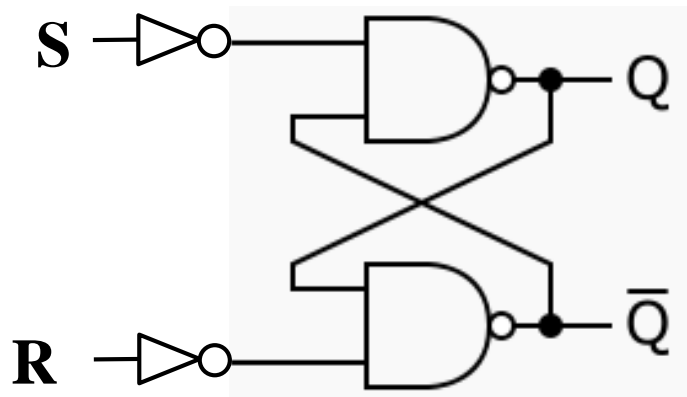


Memória de 1 bit

## 2. Memória – Conceitos Básicos

### FLIP-FLOP RS com Clock

Memória de 1 bit

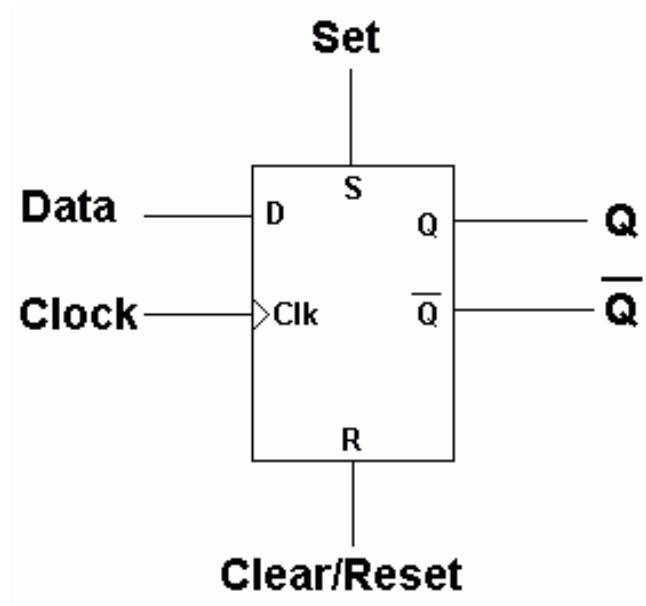


$S_C$	$R_C$	$C$	$Q$	$\bar{Q}$
$x$	$x$	0	no change	
0	0	1	no change	
0	1	1	0	1
1	0	1	1	0
1	1	1	undefined	
0	0	$p$	no change	
0	1	$p$	0	1
1	0	$p$	1	0
1	1	$p$	undefined	

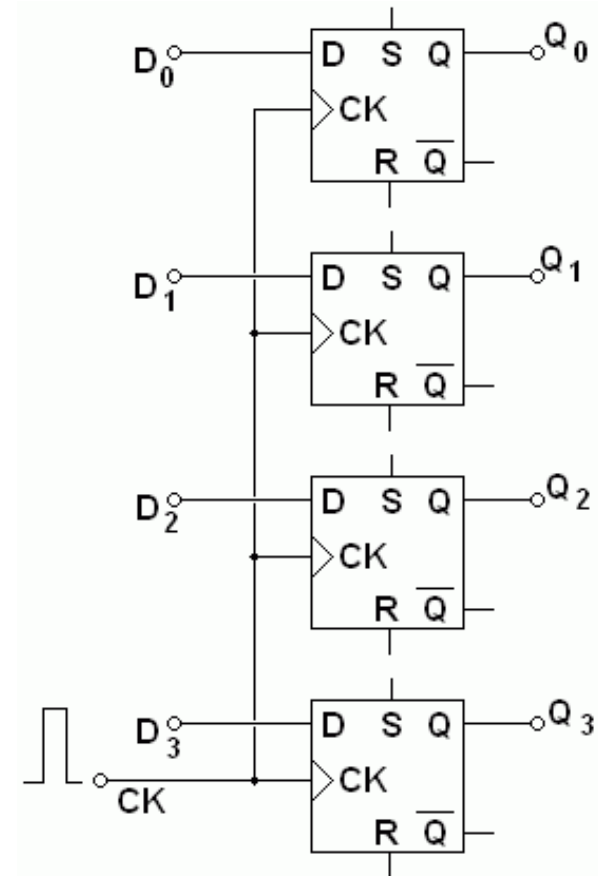
Flip-Flop:  
 RS, RS + Clock,  
 D, JK, Master-Slave

## 2. Memória – Conceitos Básicos

### D LATCH (FLIP-FLOP)



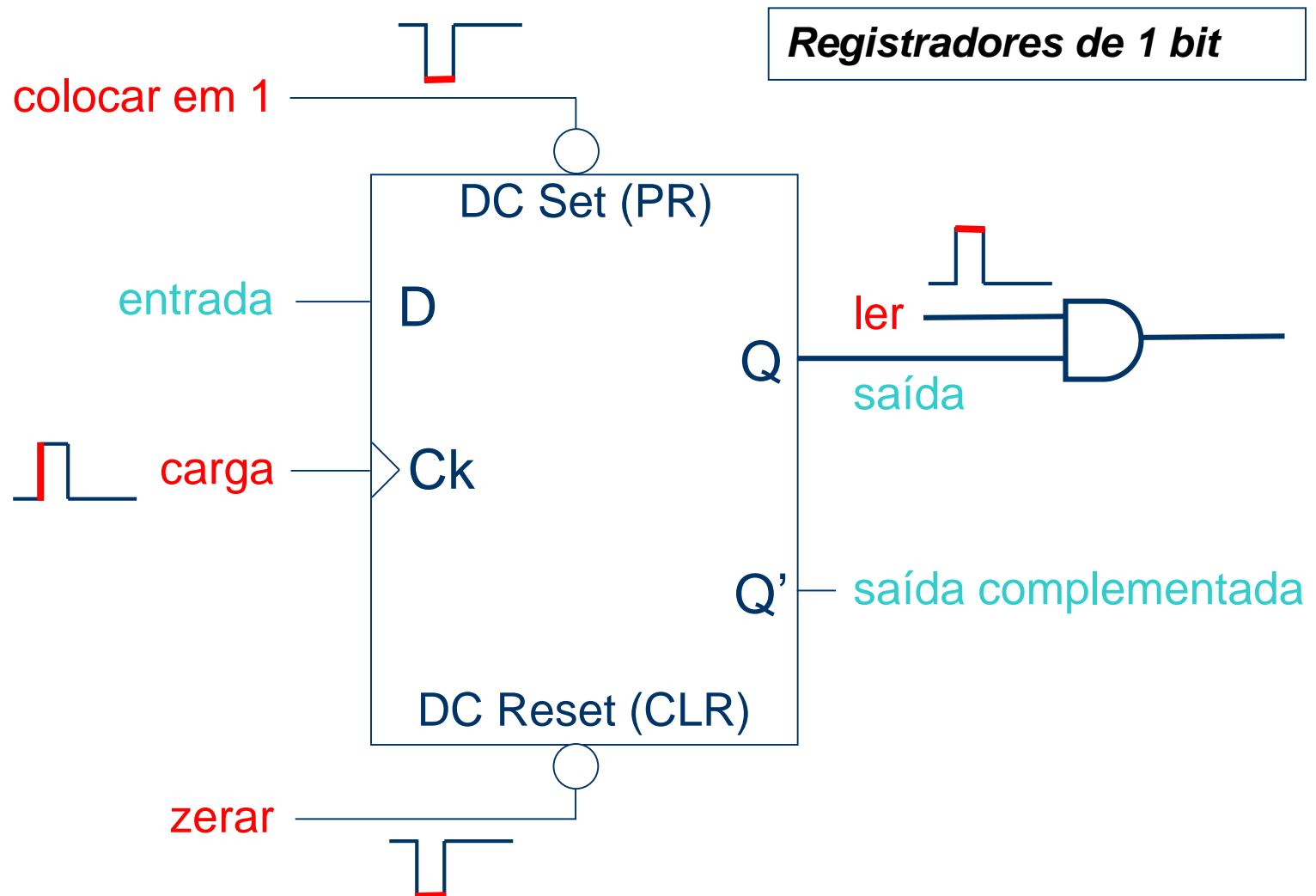
Memória de 1 bits



Memória de 4 bits

## 2. Memória – Conceitos Básicos

### Unidade de Memória

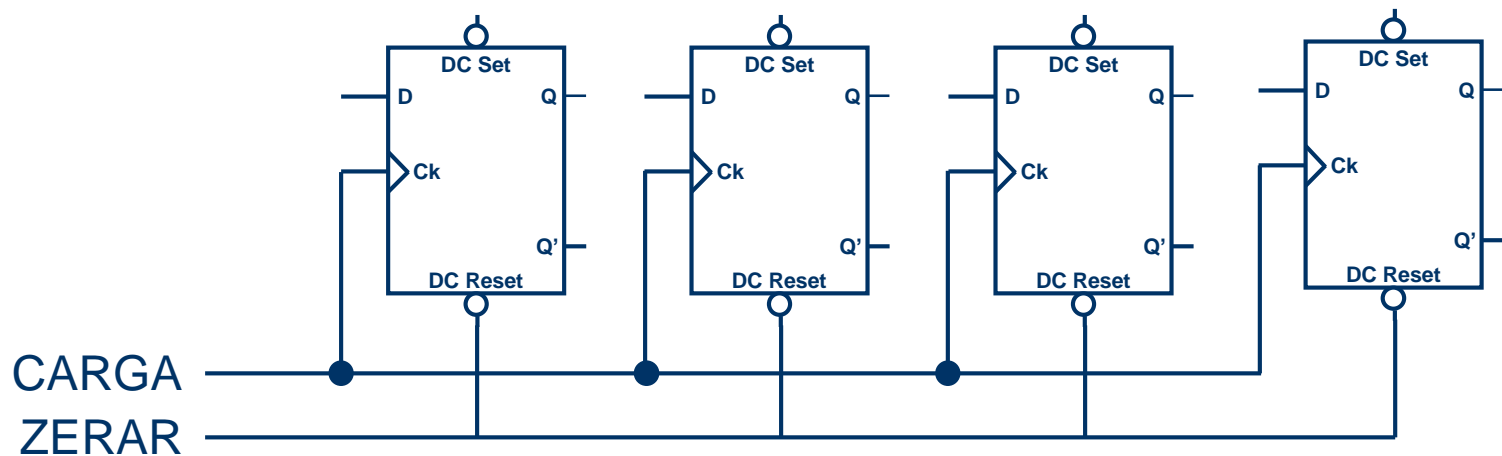


## 2. Memória – Registradores

### Unidade de Memória

- um flip-flop por bit
- sinais de controle comuns a todos os flip-flops

*Registradores de de vários bits*



Quando lê as entradas ?

- nas bordas positivas do sinal **CARGA**

Quando zera todos os bits ?

- quando o sinal **ZERAR** passa de 1 para 0

## 2. Memória – Registradores

### CPU – Processador

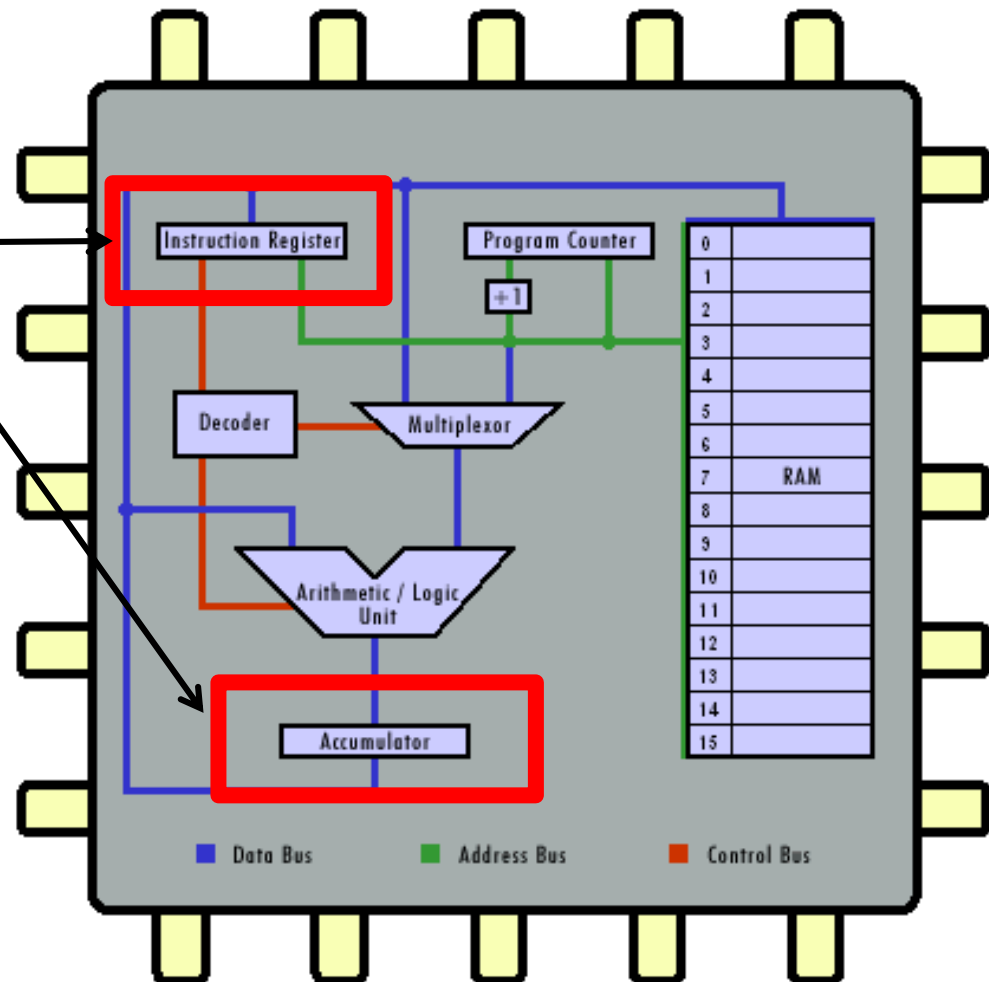
Registradores

#### Registradores

Instruction Register: IR ( RI )

Accumulator: Acc ( AC )

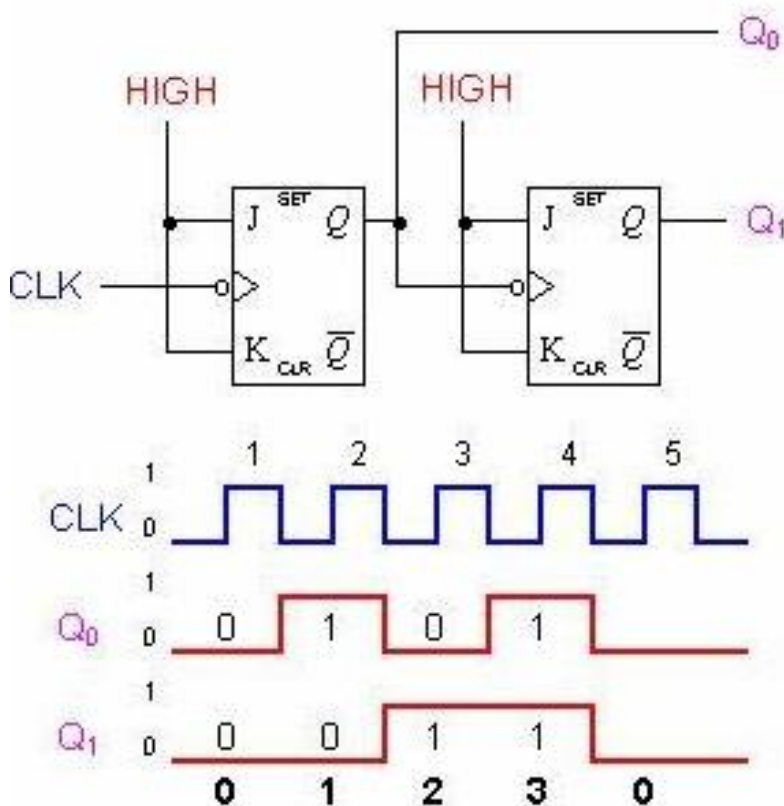
Status Register - Flags: Flags





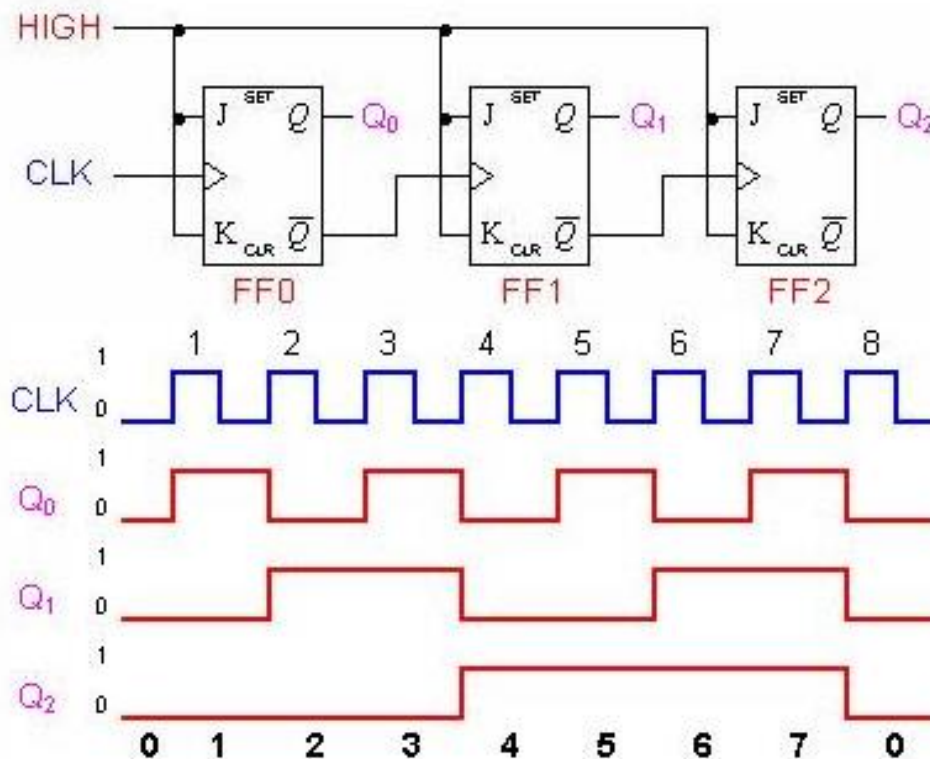
### Registrador - Contadores

JK master-slave sensíveis à borda de **descida**



### Registrador - Contadores

flip-flops JK master-slave sensíveis à borda de **subida** do controle CLOCK

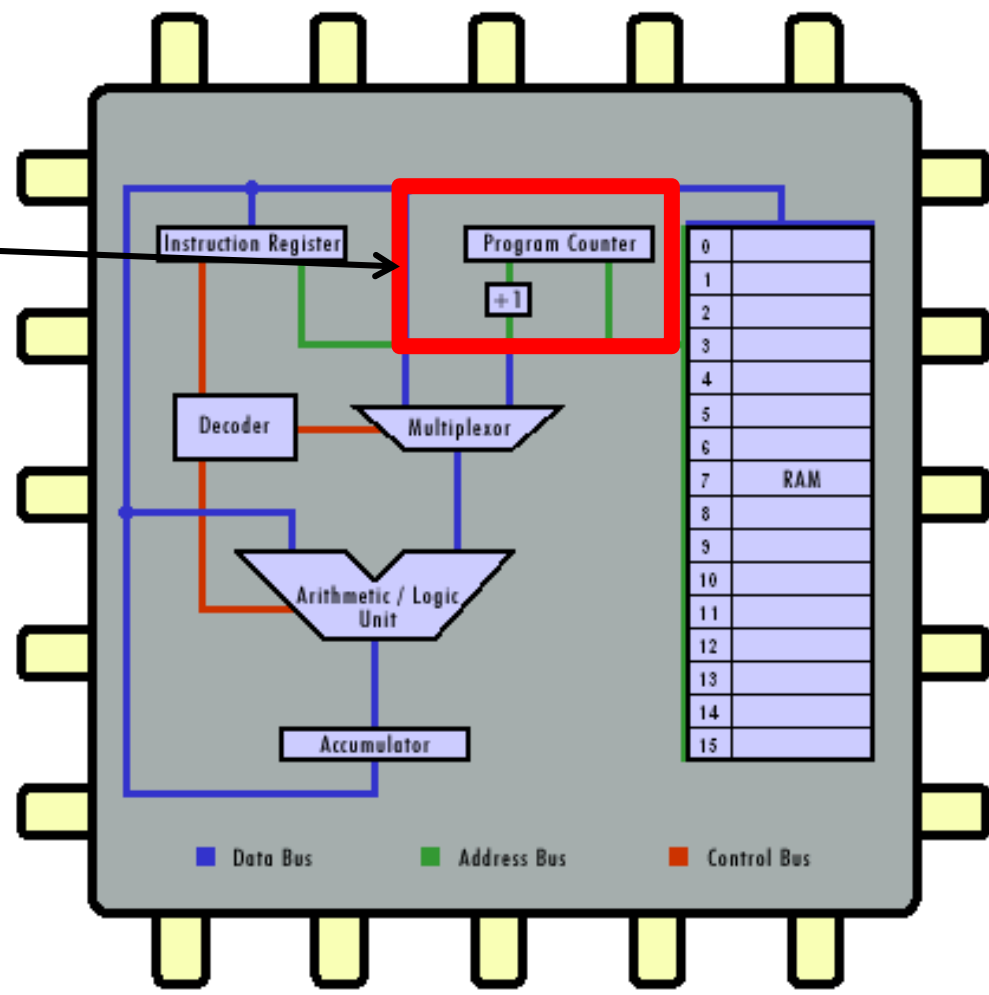


## 2. Memória – Registradores

### CPU – Processador

Registradores

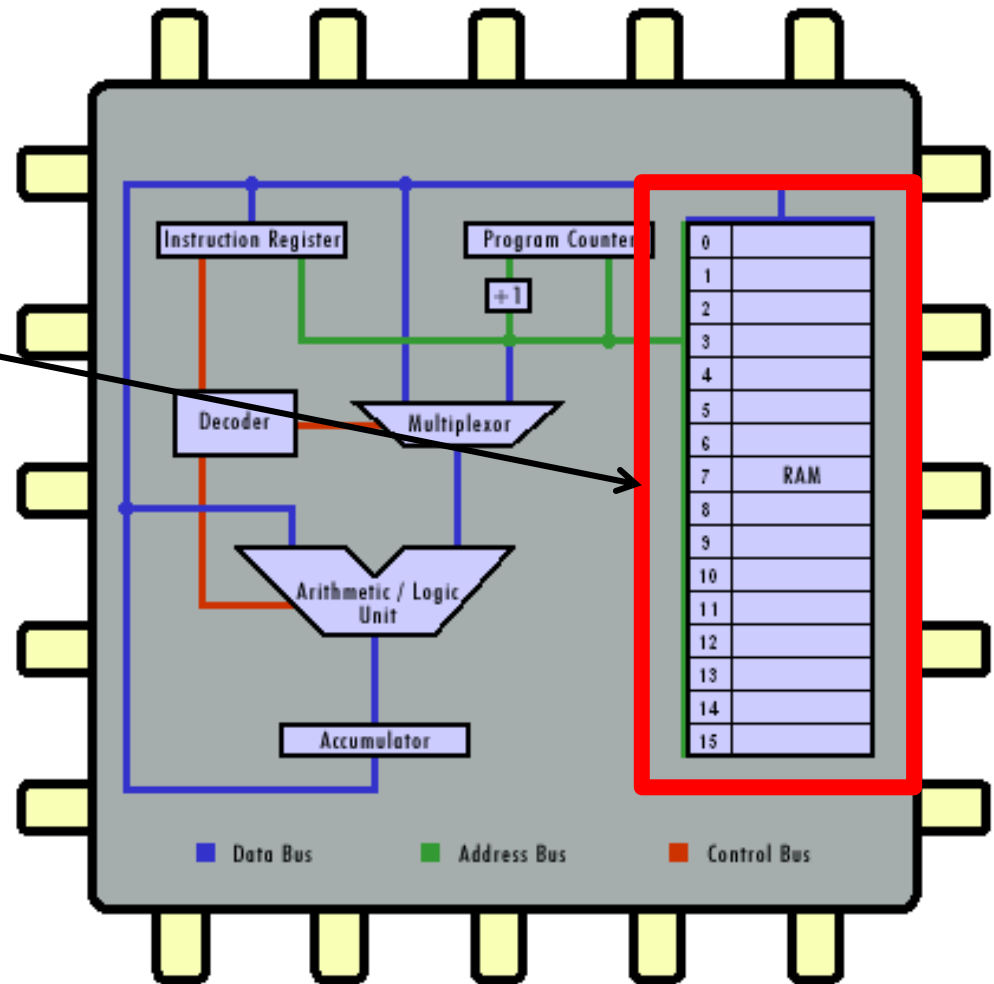
Registradores  
Program Counter: PC



## 2. Memória – Registradores

### CPU – Processador

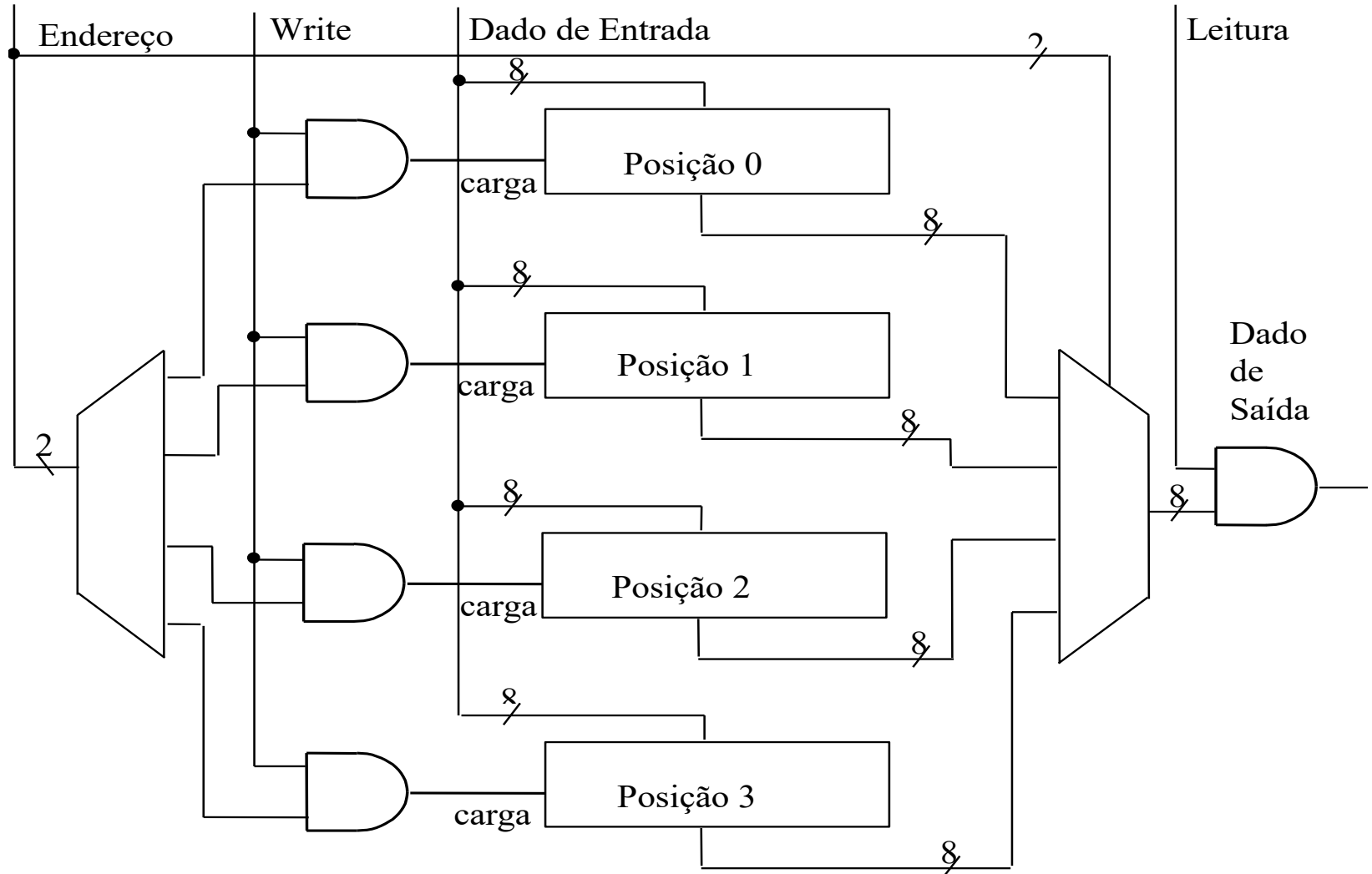
Banco  
de  
Memória



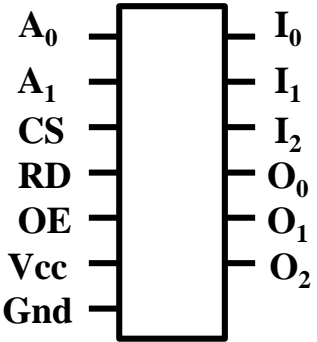
## 2. Memória – Banco de Memória

### Unidade de Memória

#### Memória com seleção linear

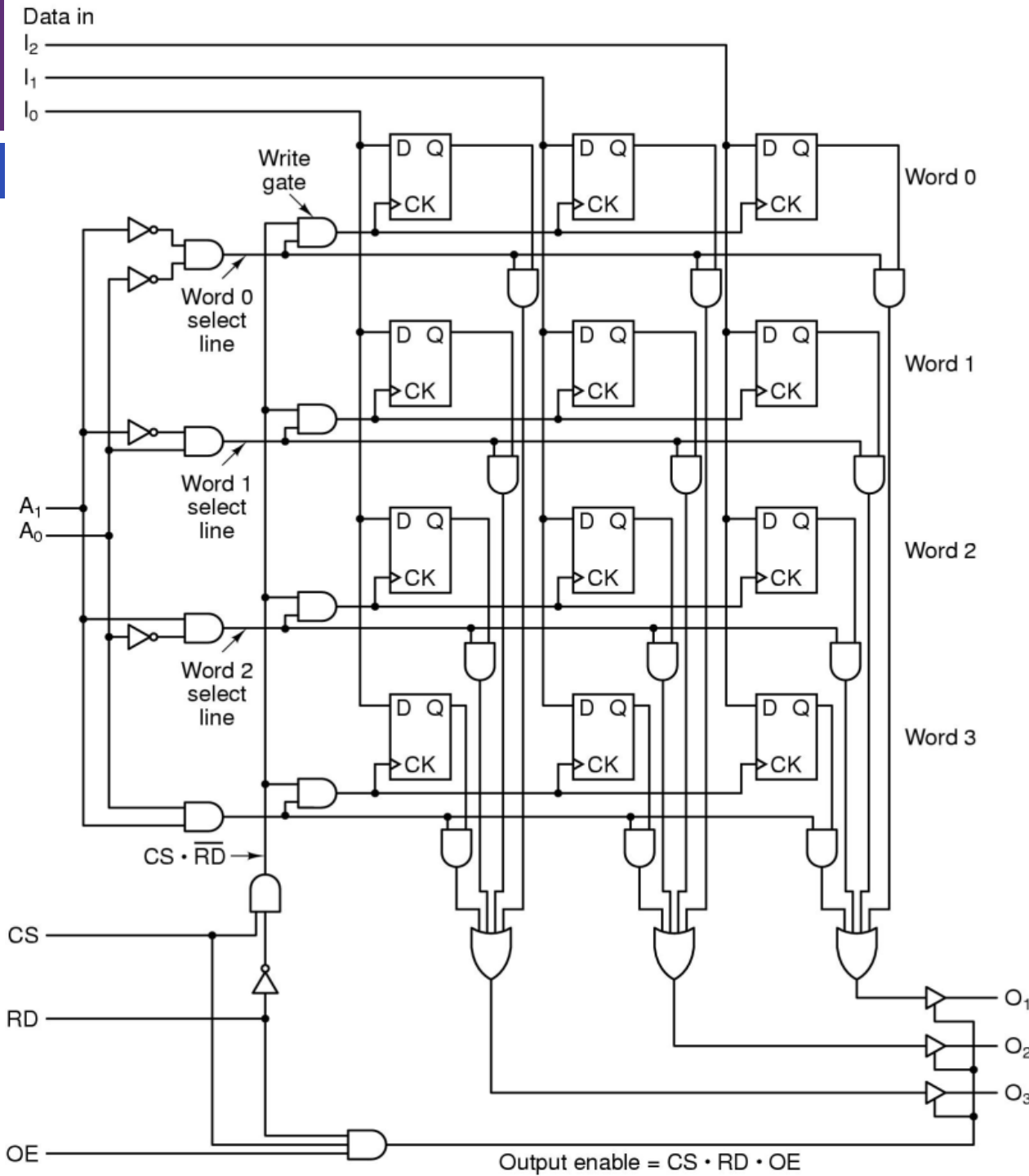


# Memória



**Memória 4 x 3:**  
 4 Endereços de  
 3 bits cada

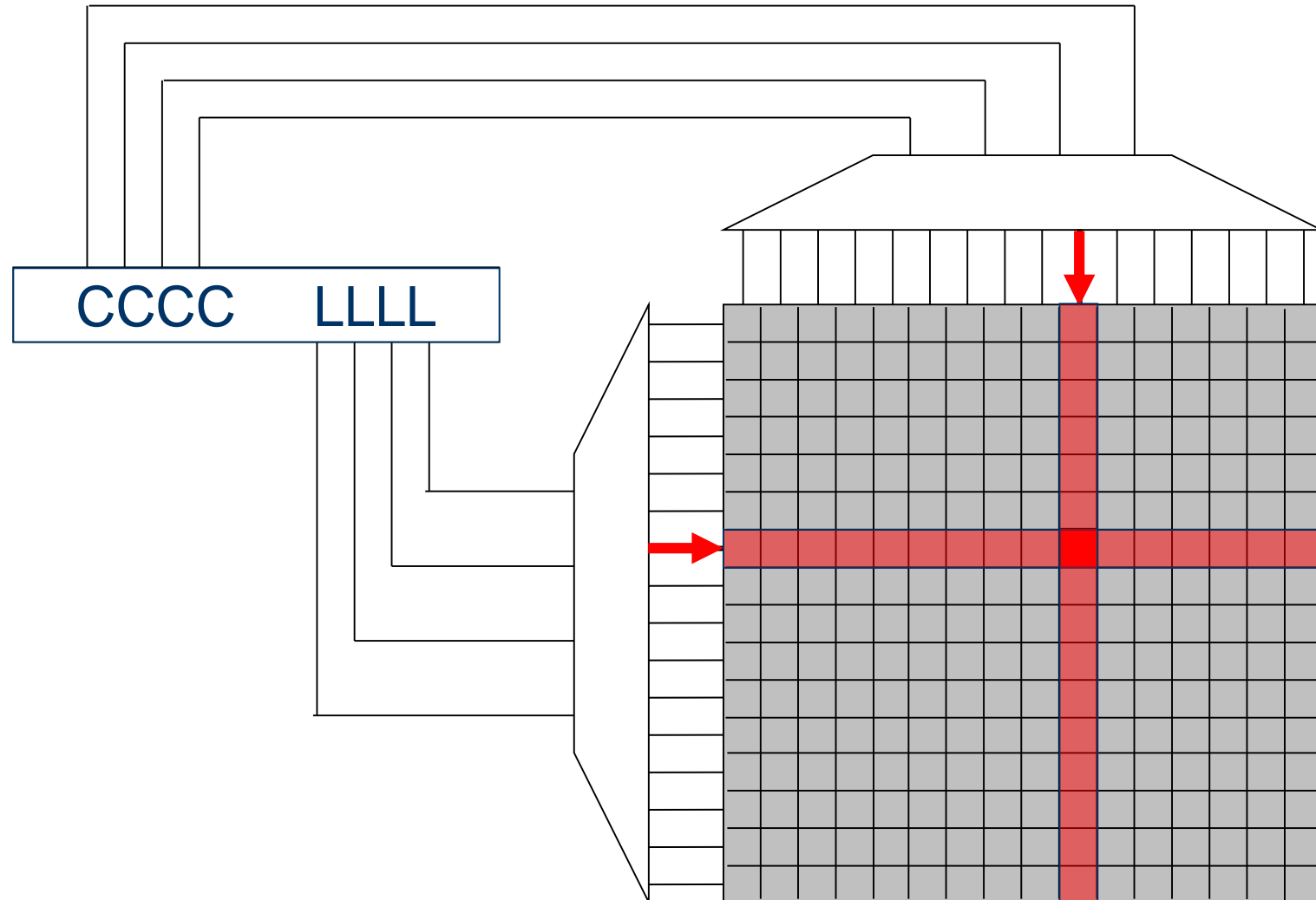
**Endereços:**  $A_0 \dots A_1$   
**Dados (in):**  $I_0 \dots I_2$   
**Dados (out):**  $O_0 \dots O_2$   
**CS = Chip Select**  
**RD = Read**  
**OE = Output Enable**



## 2. Memória – Banco de Memória

### Unidade de Memória

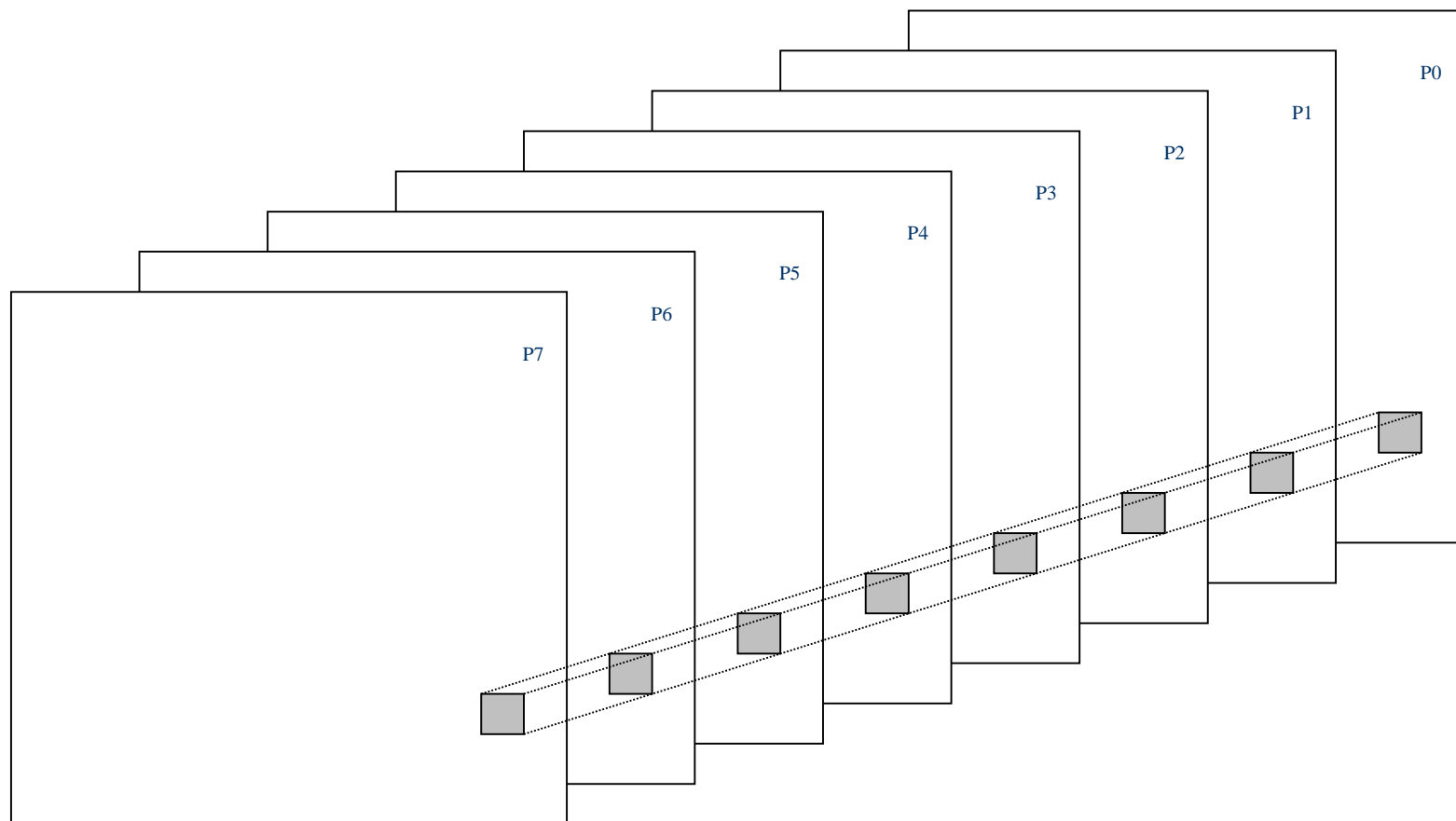
*Memória com seleção matricial  
(um “plano”)*



## 2. Memória – Banco de Memória

### Unidade de Memória

*Memória com seleção de plano  
(1 bit em cada “plano”)*



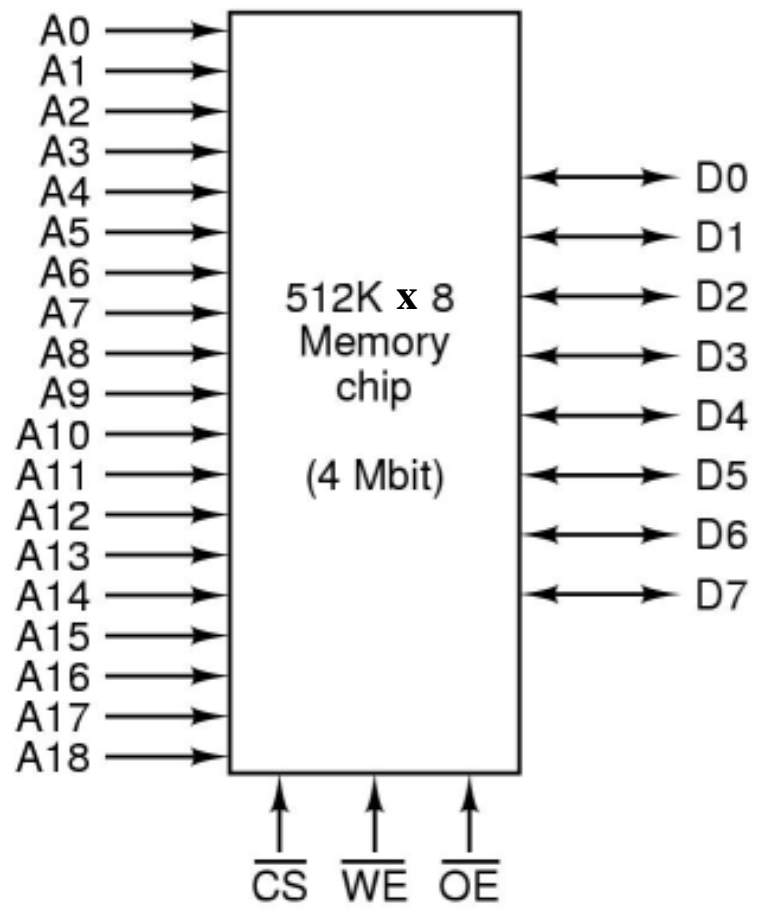
(1 “plano” pode ser 1 circuito integrado de memória “ $nk \times 1$  bit”)



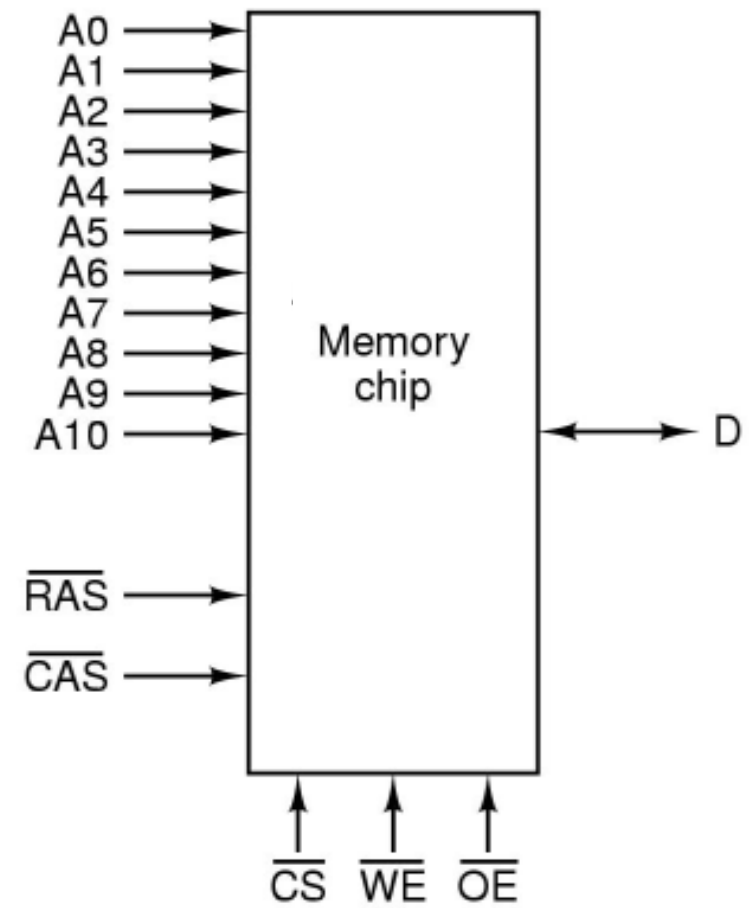
## 2. Memória – Banco de Memória

### Unidade de Memória

### Static and Dynamic RAM



(a)

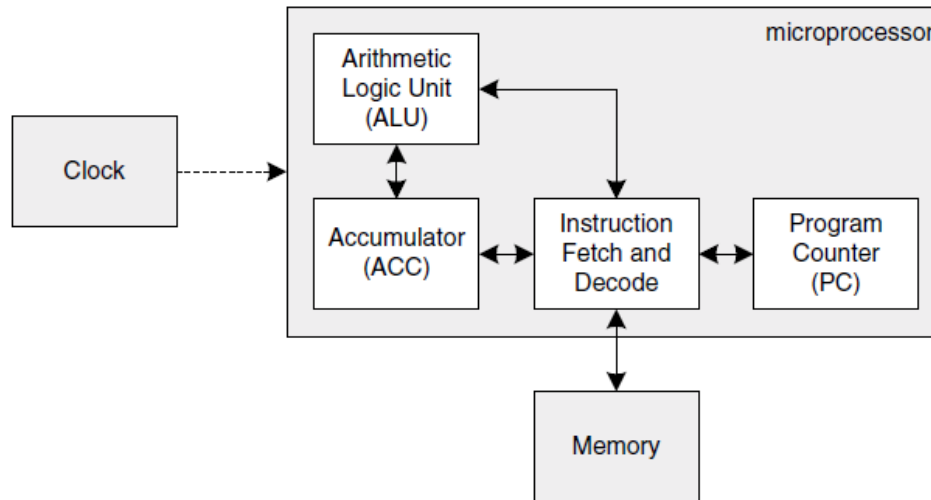


(b)

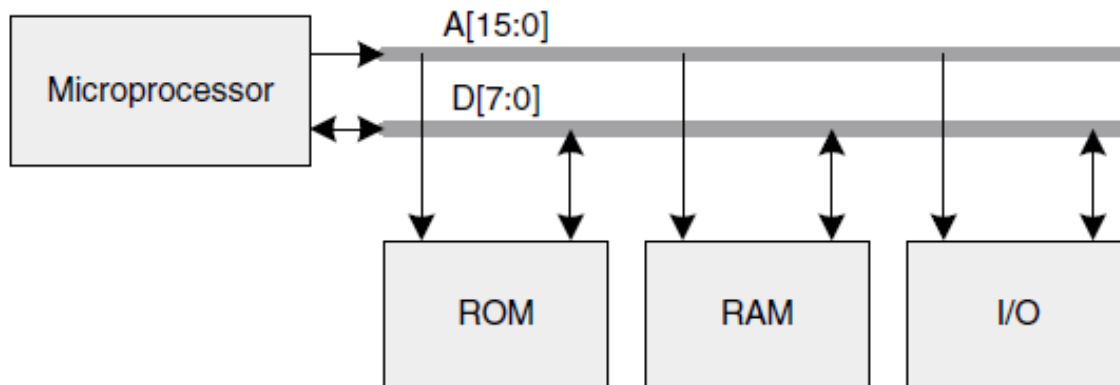
## 2. Memória, Registradores e E/S

### Arquitetura de Computadores

Fonte: Mark Balch - Complete Digital Design



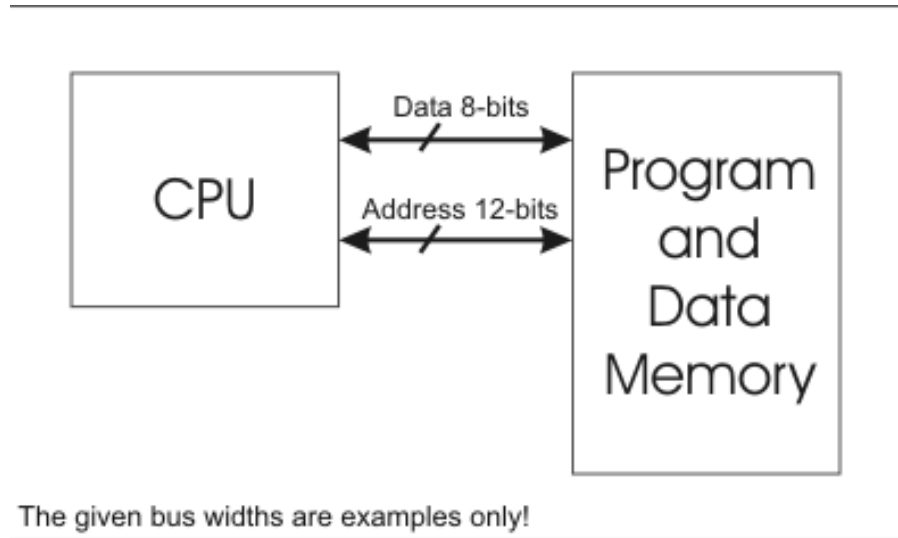
Simple microprocessor



Microprocessor buses

## 2. Arquitetura de Von Neumann

### Arquitetura: *Von Neumann x Harvard*

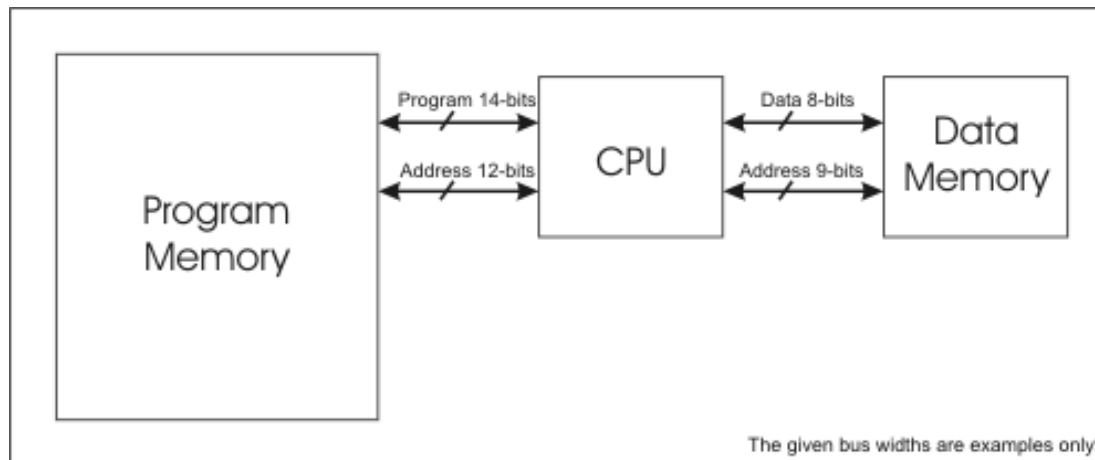


#### Máquina de Von Neumann

Instruções:  
Operador + Operando

IAS

8 bits - Operação (OPcode)  
12 bits - Operando (Endereço)



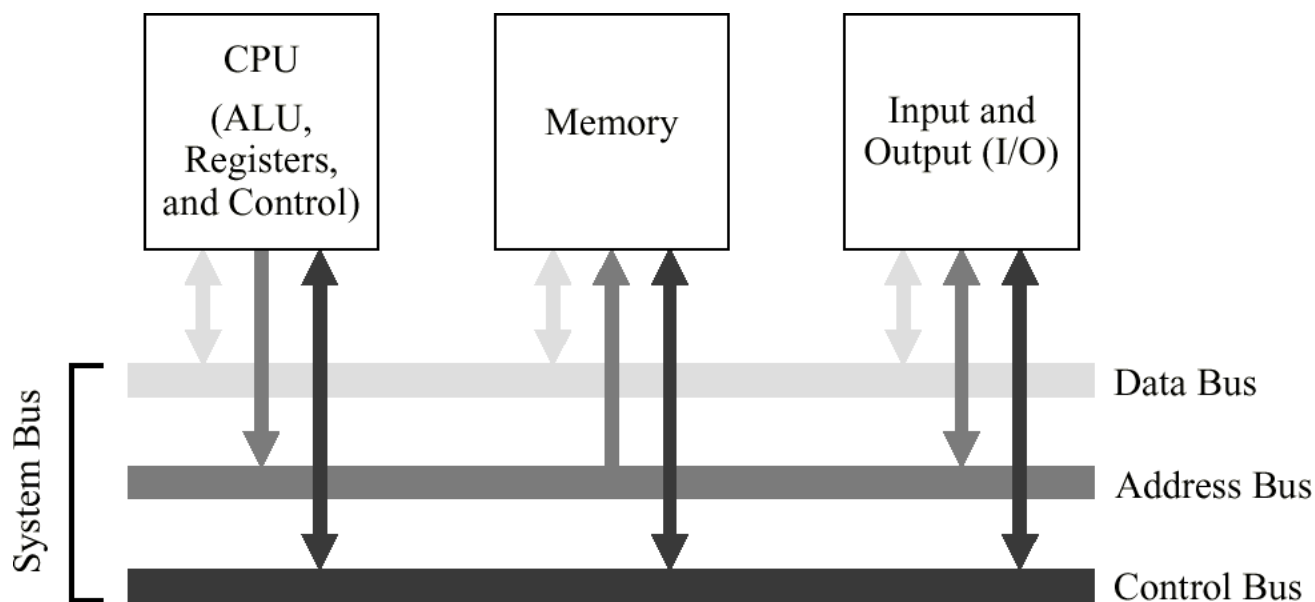
#### Arquitetura de Harvard

- Separação entre memória de programa e memória de dados
- Busca de instruções e operandos pode ser simultânea (barramentos separados)

## 2. Arquitetura de Von Neumann

### Arquitetura de Computadores:

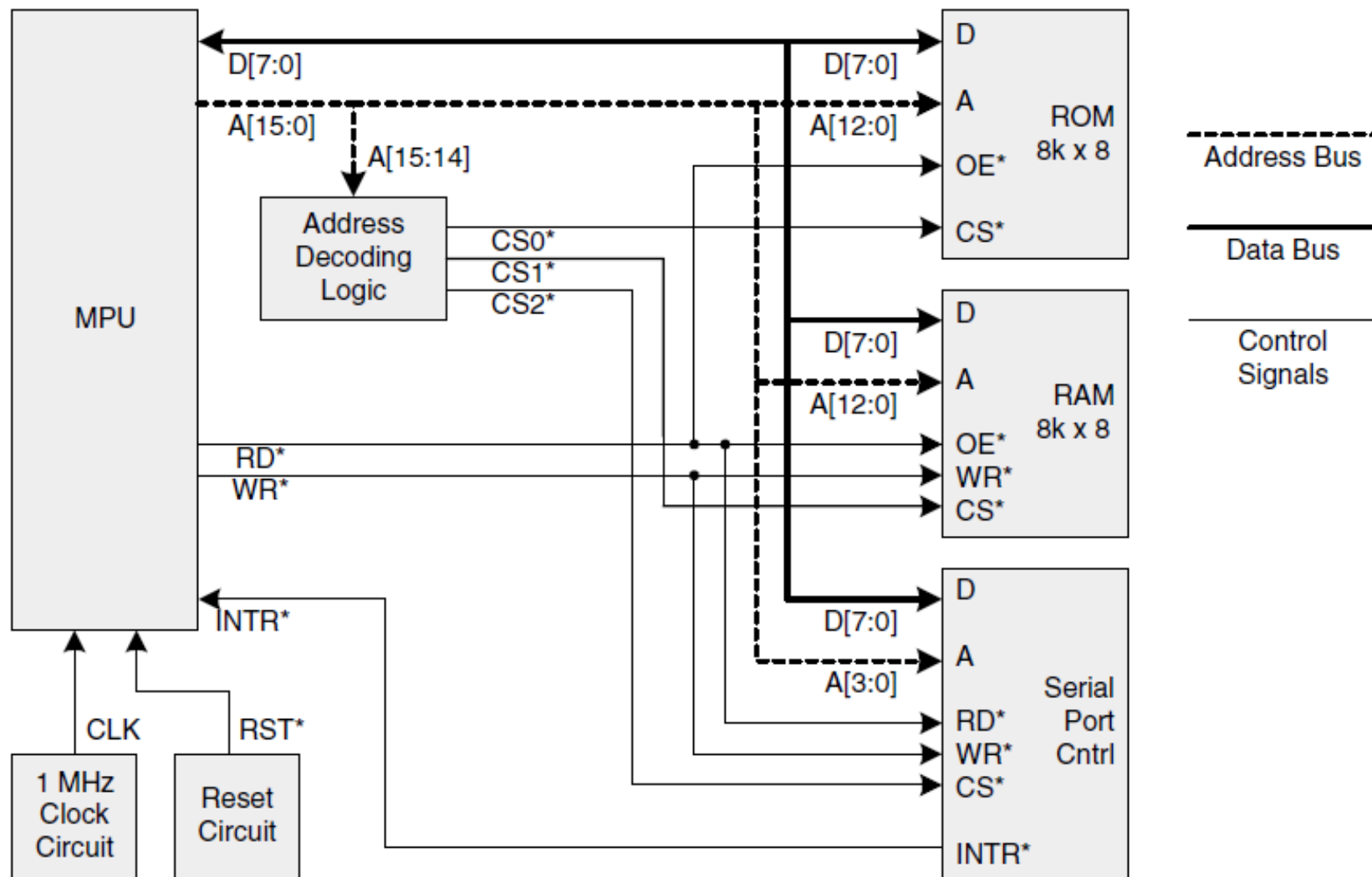
Barramentos: Endereços, Dados e Controle



## 2. Arquitetura de Von Neumann

### Arquitetura de Computadores

Eight-bit computer block diagram



## 2. Arquitetura de Von Neumann

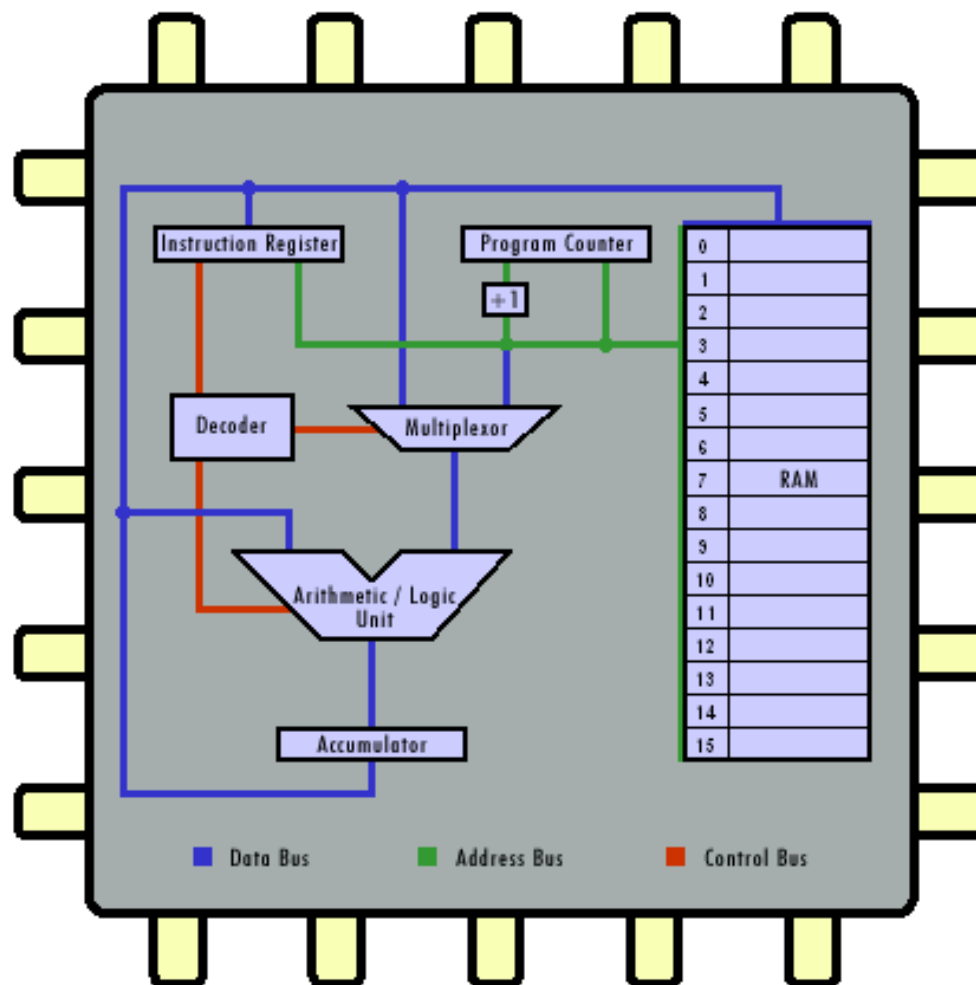
### Unidade de Controle - UC

- Responsável por gerar todos os sinais que controlam as operações no exterior do CPU, e ainda por dar todas as instruções para o correto funcionamento interno do CPU [UC é o "Maestro da Orquestra"];
- Junto a Unidade de Controle temos um decodificador de instruções, que analisa e decodifica o Operador da Instrução (OPcode);
- A unidade de controle executa três ações básicas intrínsecas e pré-programadas pelo próprio fabricante do processador, são elas:
  - (i) busca (fetch)
  - (ii) decodificação
  - (iii) execução
- Assim sendo, todo processador, ao iniciar sua operação, realiza uma operação cíclica, tendo como base essas três ações. A unidade de controle usualmente é fixa, onde esta define o Conjunto de Instruções do Processador:  
***CPU Instruction Set***

## 2. Arquitetura de Von Neumann

### Unidade de Controle - UC

Show time...



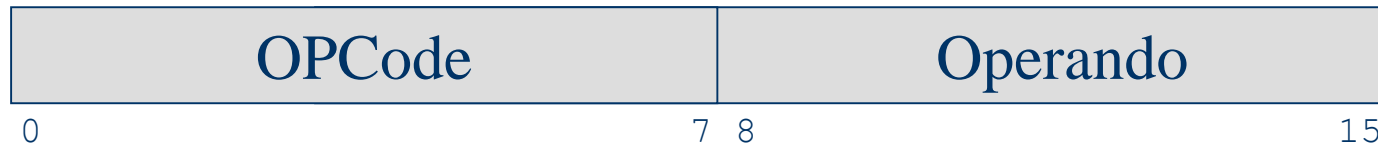
## 2. Arquitetura de Von Neumann

### Unidade de Controle - UC

**CPU Instruction Set**

**Código de Instruções**

**Operador da Instrução + Operando(s)**



**Exemplo: Instrução de 16 Bits**

Tipos de Instruções...

Arquiteturas de 4, 3, 2, 1 e 0 endereços

Tipos de Instruções... Conjunto de Instruções Microprogramado

Tipos de Instruções... RISC, CISC, VLIW

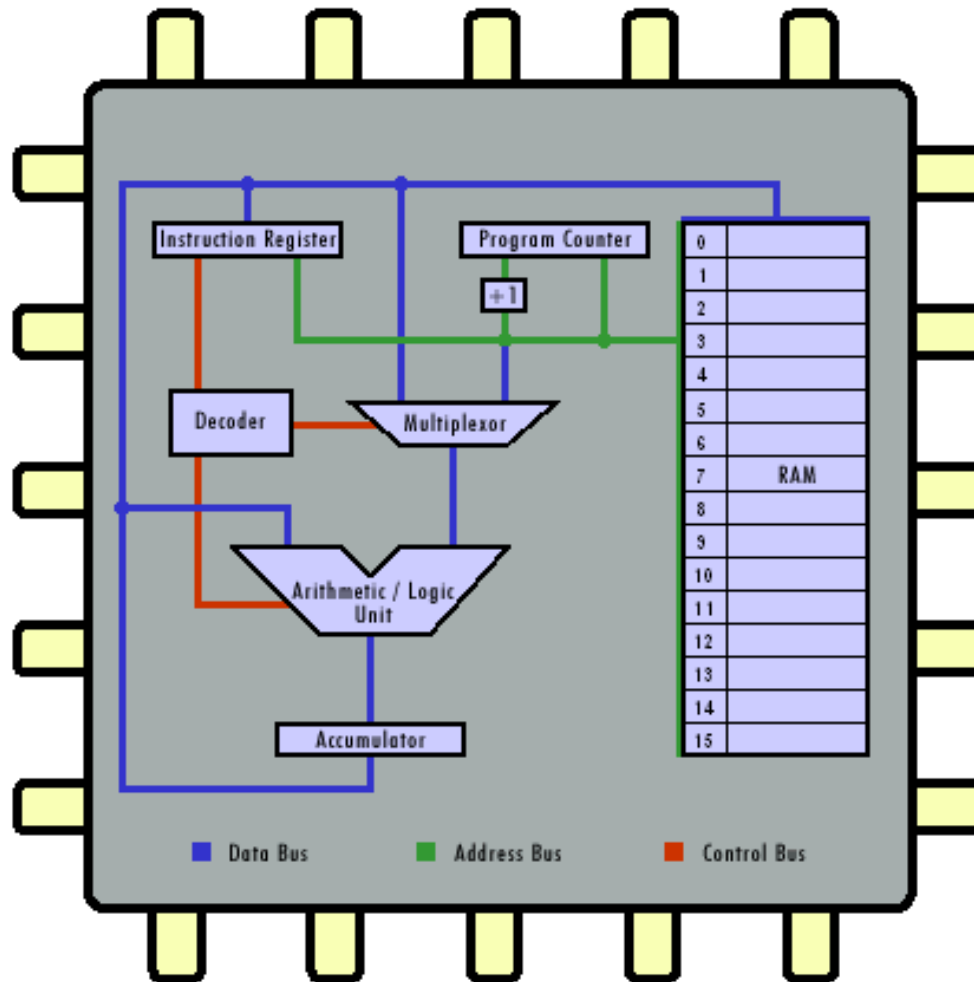
Tipos de Instruções... Modo de Endereçamento

Tipos de Instruções... Instruções de Controle, Privilegiadas, etc.



# 1. Arquitetura de Von Neumann

## Unidade de Controle - UC



### Elementos importantes:

PC - Program Counter

AC - Accumulator

IR - Instruction Register

Address Bus:  $n$  bits

Data Bus :  $m$  bits

Flags da ULA (S - Status)

Z - Zero      C - Carry

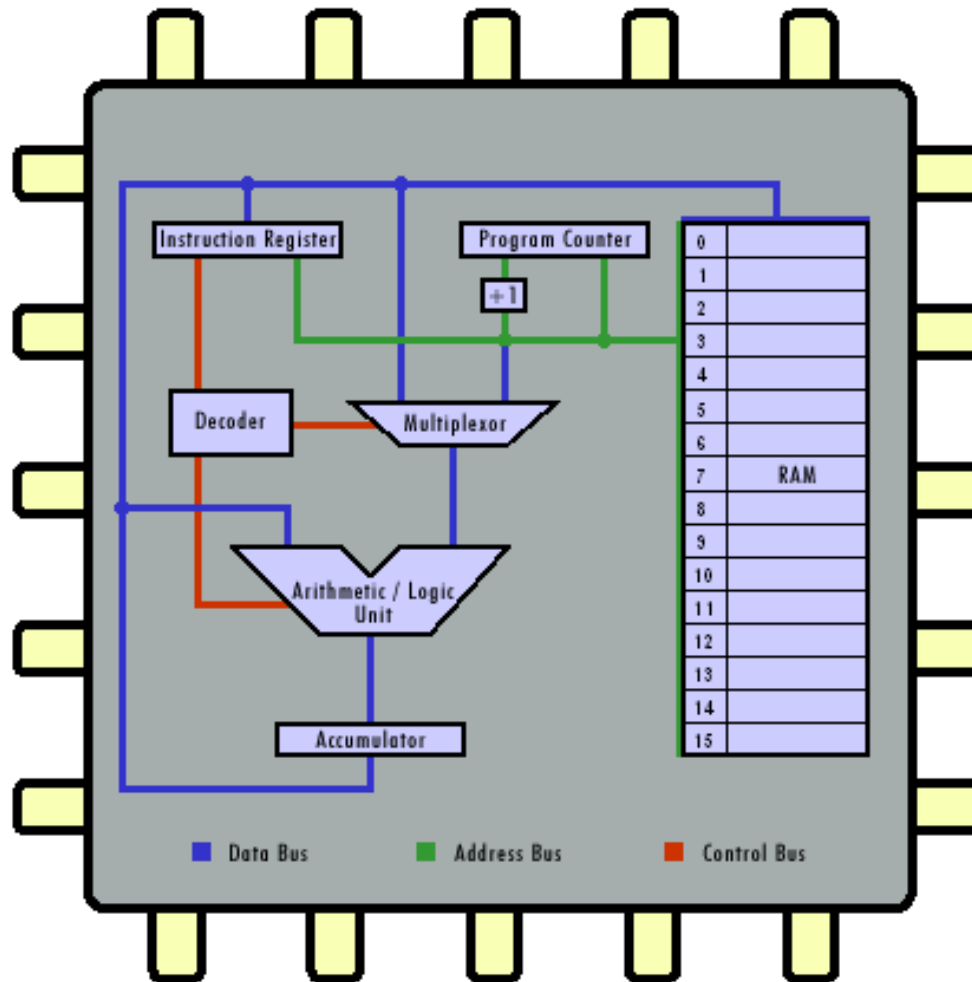
N - Negative    V - Overflow

Memória: RAM / ROM

System/User Program, Data,  
Stack (Pilha), I/O, Interrupts

# 1. Arquitetura de Von Neumann

## Unidade de Controle - UC



### μProc - Elementos

PC - Program Counter  
AC - Accumulator  
IR - Instruction Register  
SP - Stack Pointer  
Ix - Index Register  
BR - Base Registers

Address Bus:  $n$  bits  
Data Bus :  $m$  bits

Flags da ULA (S - Status)  
Z - Zero      C - Carry  
N - Negative    V - Overflow  
P - Parity      I - Interrupt

## 3. Arquitetura de Microprocessadores - Intel 4004

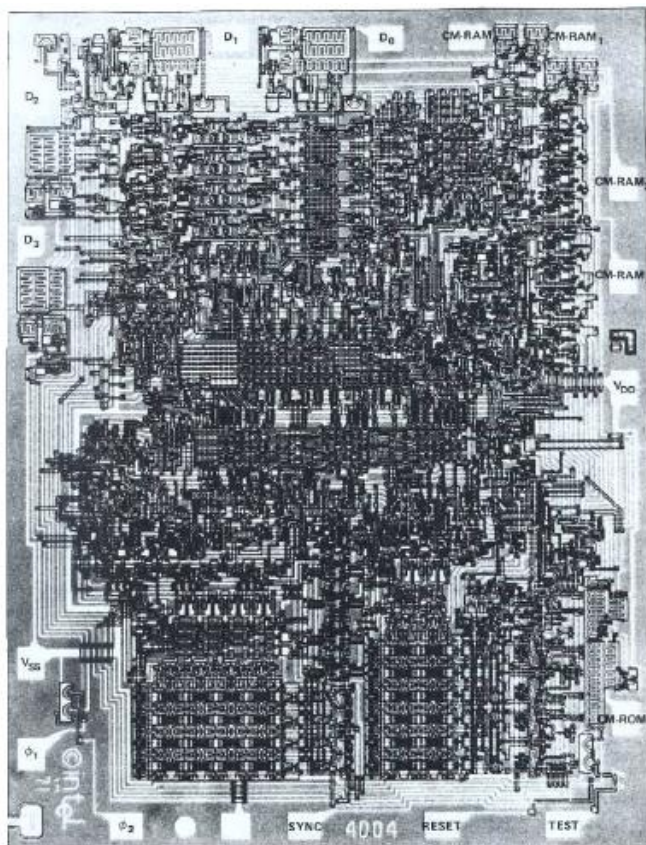
### Intel 4004 (1971)

16-pin DIP package

P-channel Silicon Gate MOS

Minimum system: CPU and one ROM

4-bit parallel CPU with 45 instructions



# MCS-4

## FOUR-BIT PARALLEL MICROCOMPUTER SET

### Features

- Microprogrammable General Purpose Computer Set
- 4-Bit Parallel CPU With 46 Instructions
- Instruction Set Includes Conditional Branching, Jump to Subroutine and Indirect Fetching
- Binary and Decimal Arithmetic Modes
- Addition of Two 8-Digit Numbers in 850 Microseconds
- 2-Phase Dynamic Operation
- 10.8 Microsecond Instruction Cycle
- CPU Directly Compatible With MCS-4 ROMs and RAMs
- Easy Expansion — One CPU can Directly Drive up to 32,768 Bits of ROM and up to 5120 Bits of RAM
- Unlimited Number of Output Lines
- Packaged in 16-Pin Dual In-Line Configuration
- Directly Compatible With 4004 CPU
- Interface 1702A PROMs Directly to 4004 CPU -- Completely Eliminates TTL Interface
- Permits Program Storage in Alterable Memory
- Execute MCS-4 Programs from any Mix of Standard Intel PROMs, ROMs and RAMs
- Expanded I/O Port Capability
- Each Port May be Both Input and Output -- Up to 16 4-bit Input Ports and 16 4-bit Output Ports
- I/O Ports and Control Lines are TTL Compatible
- Number of I/O Ports is Independent of the Size of the Program Memory
- New Instruction WPM (Write Program Memory) is Used for Loading Alterable Program Storage (RAM)

### 3. Arquitetura de Microprocessadores - Intel 4004

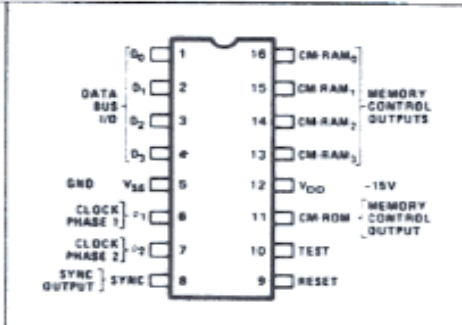
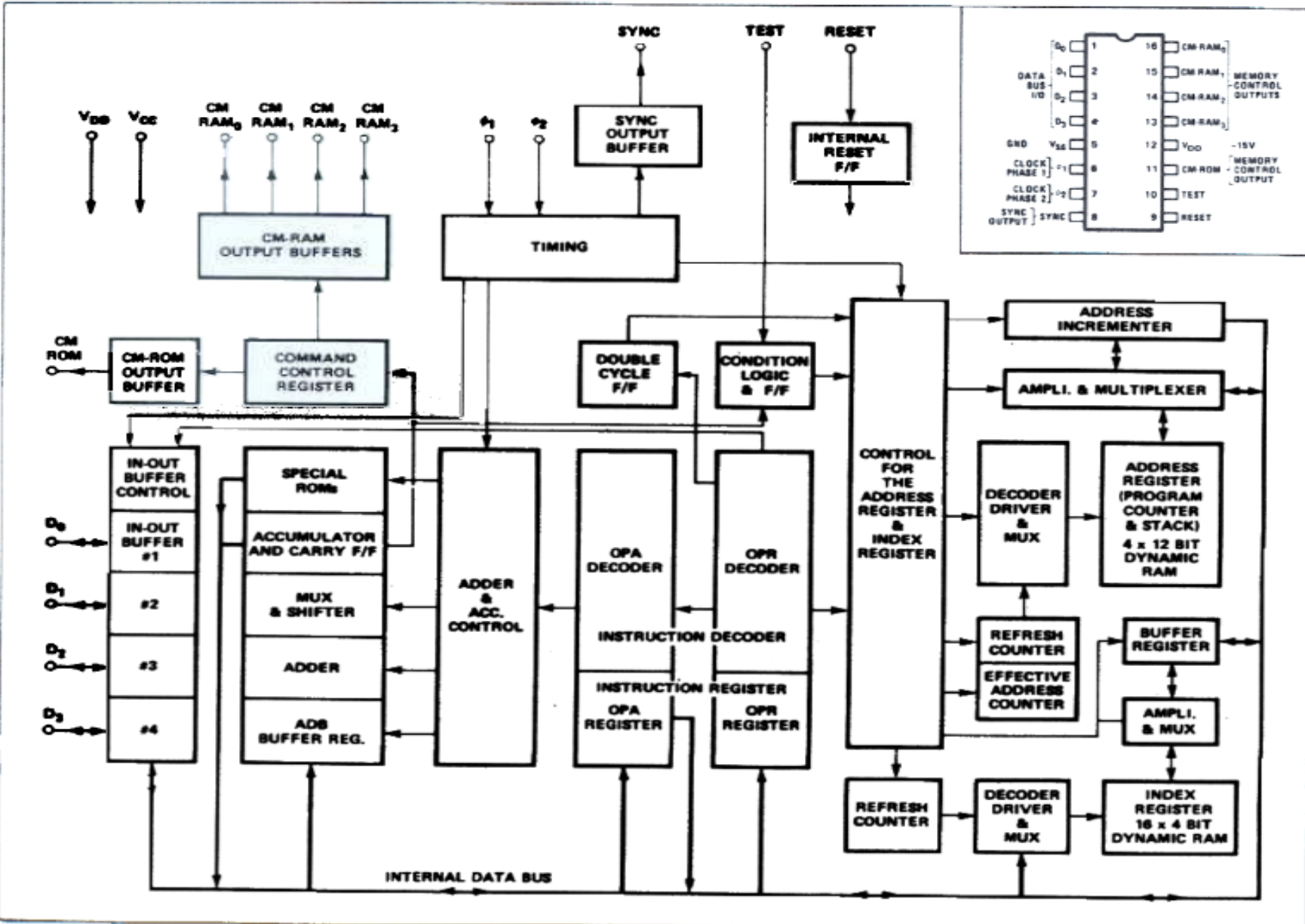
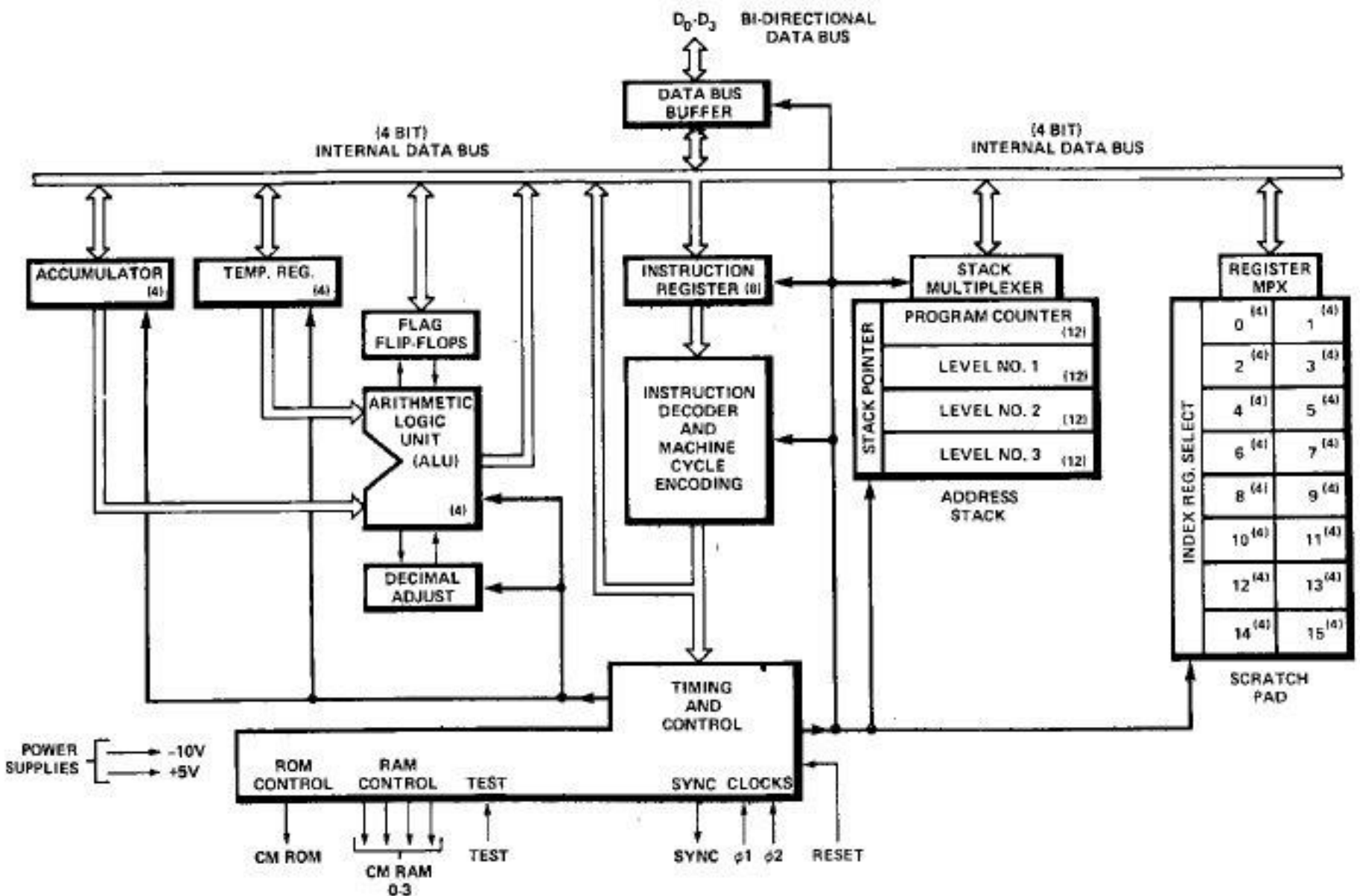


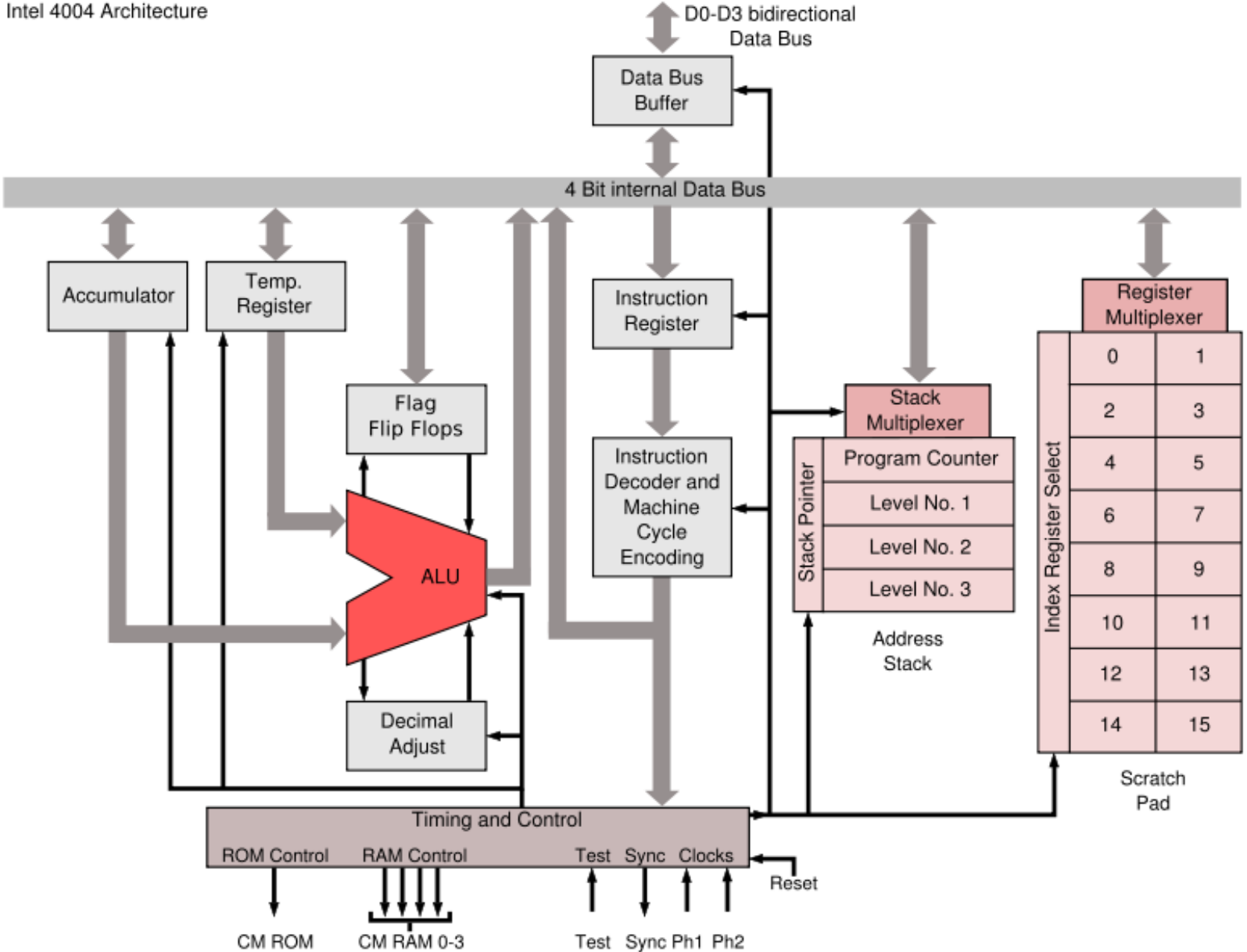
Figure 3. 4004 CPU Block Diagram

### 3. Arquitetura de Microprocessadores - Intel 4004

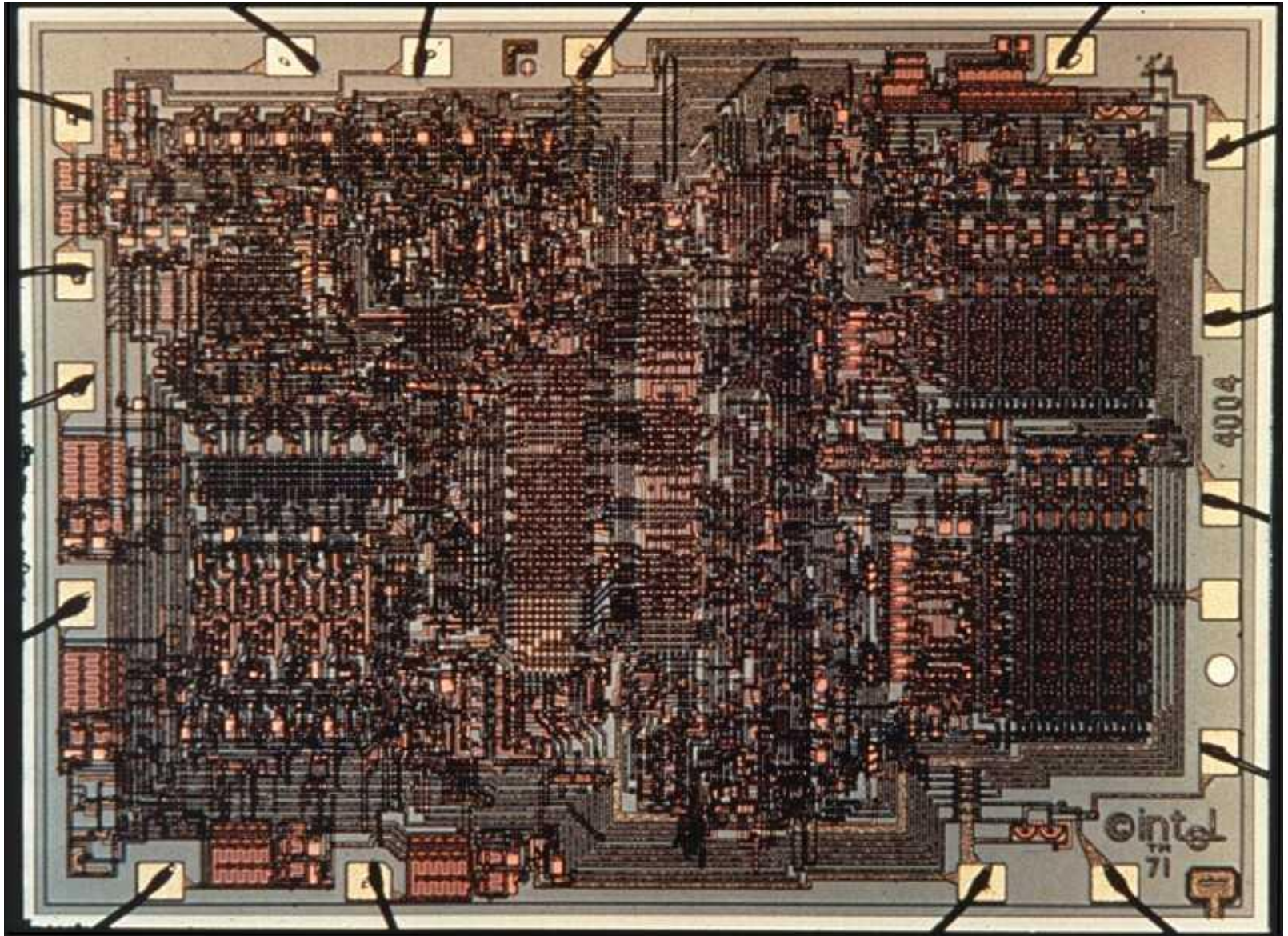


# 3. Arquitetura de Microprocessadores - Intel 4004

Intel 4004 Architecture



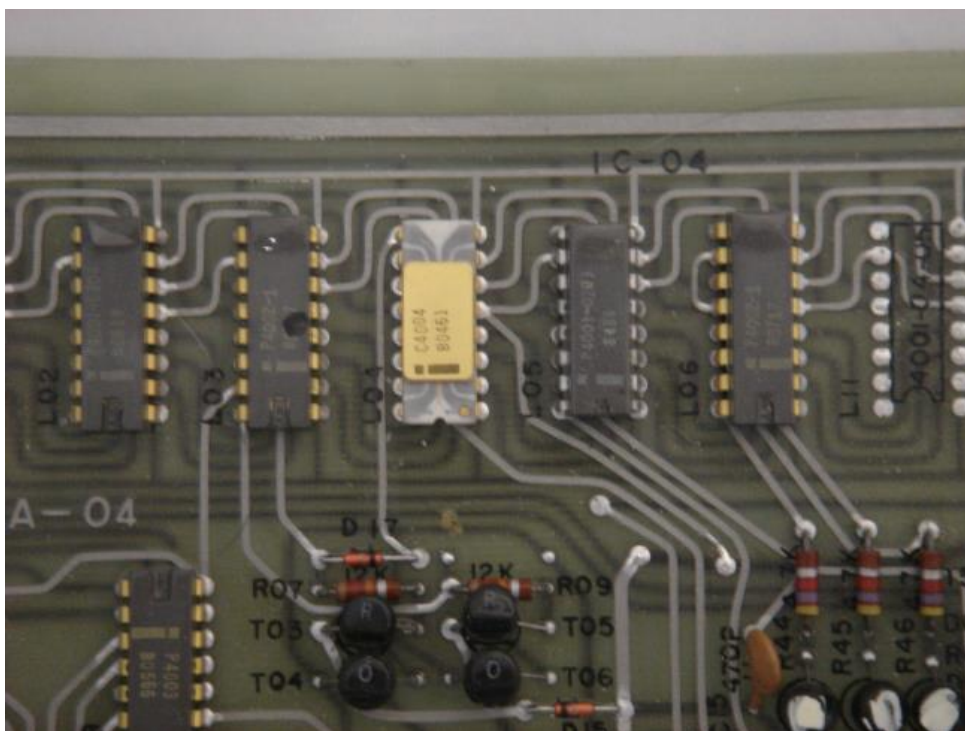
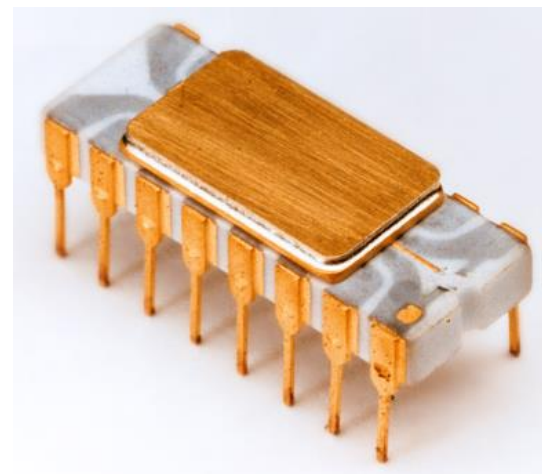
### 3. Arquitetura de Microprocessadores - Intel 4004



### 3. Arquitetura de Microprocessadores - Intel 4004

## Intel 4004

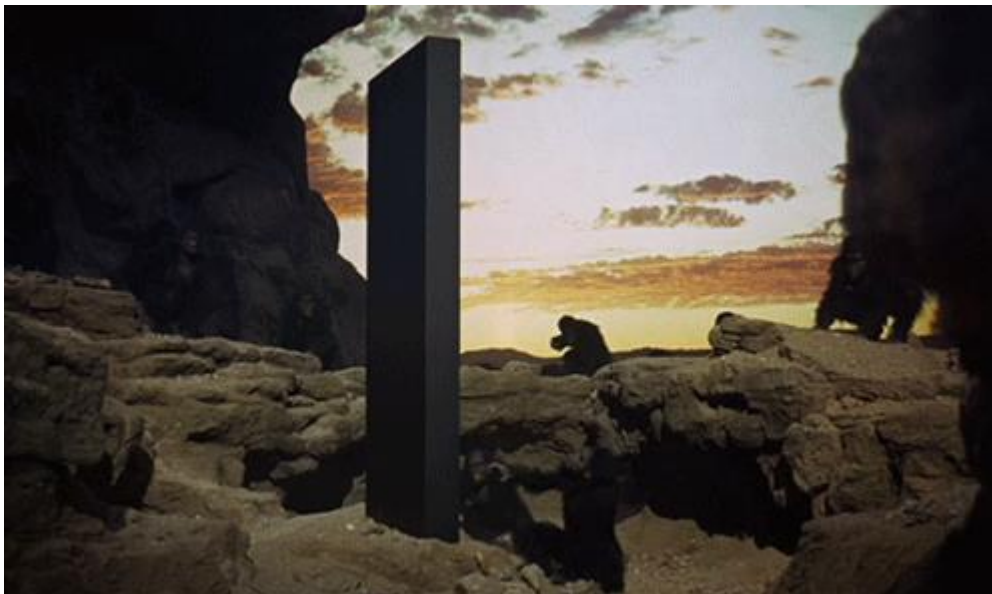
Busicom - Nippon Calculating Machine Corp changed its name to Business Computer Corporation, was a Japanese company that owned the rights to the first microprocessor but sold them back to Intel. They made electronic calculators and the first using the new Intel 4004 processor was the Busicom 141-PF[1]





## 4. Arquiteturas Didáticas

**Começando do Começo...**



## 4. Arquiteturas Didáticas

### **Neander - Computador Hipotético [Weber 2001\*]**

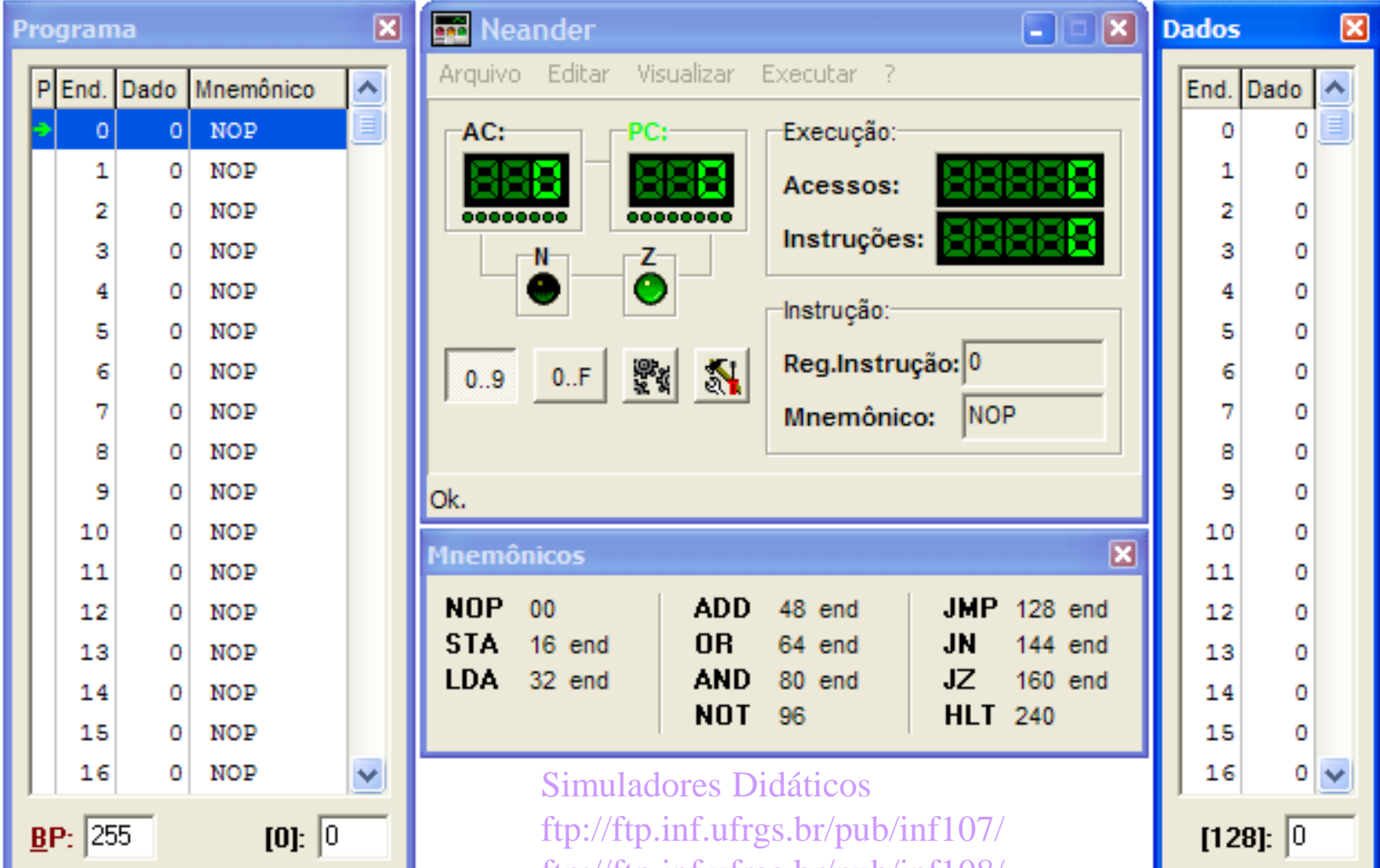
#### **Arquitetura: características gerais**

- **Largura de dados e endereços de 8 bits (bus)**
- **Dados representados em complemento de 2**
- **Acumulador de 8 bits (AC - Accumulator)**
- **Apontador de programa de 8 bits (PC - Program Counter)**
- **Registrador de Instruções de 8 bits (IR - Instruction Reg.)**
- **Registrador de estado (flags) com 2 códigos de condição: Negativo (N) e Zero (Z)**
- **Endereçamento de memória total de 256 bytes**

## 4. Arquiteturas Didáticas

Neander => Simulador WNeander

 **Neander** Versão 2.1  
Julho 2002  
Autores: Raul Fernando Weber  
Taizy Silva Weber  
Versão: Fabio Augusto Dal Castel  
Win32



The screenshot displays the Neander simulator interface with three main windows:

- Programa:** A table showing memory addresses from 0 to 16, all containing the instruction NOP.
- Neander:** The main CPU window showing:
  - AC (Accumulator) and PC (Program Counter) registers, both displaying 000.
  - Execution status: Execução: 00000, Acessos: 00000, Instruções: 00000.
  - Instruction register: Reg.Instrução: 0, Mnemônico: NOP.
  - Control flags: N (Not) and Z (Zero) are shown as indicators.
  - Buttons for 0..9, 0..F, and other functions.
- Dados:** A table showing memory addresses from 0 to 16, all containing the instruction NOP.

At the bottom of the Neander window, there is a table of mnemonics:

Mnemonic	Hex	Hex	Mnemonic	Hex	Hex	Mnemonic	Hex	
NOP	00		ADD	48	end	JMP	128	end
STA	16	end	OR	64	end	JN	144	end
LDA	32	end	AND	80	end	JZ	160	end
			NOT	96		HLT	240	

At the bottom of the Program and Dados windows, there are input fields for BP (255) and [0] (0) on the left, and [128] (0) on the right.

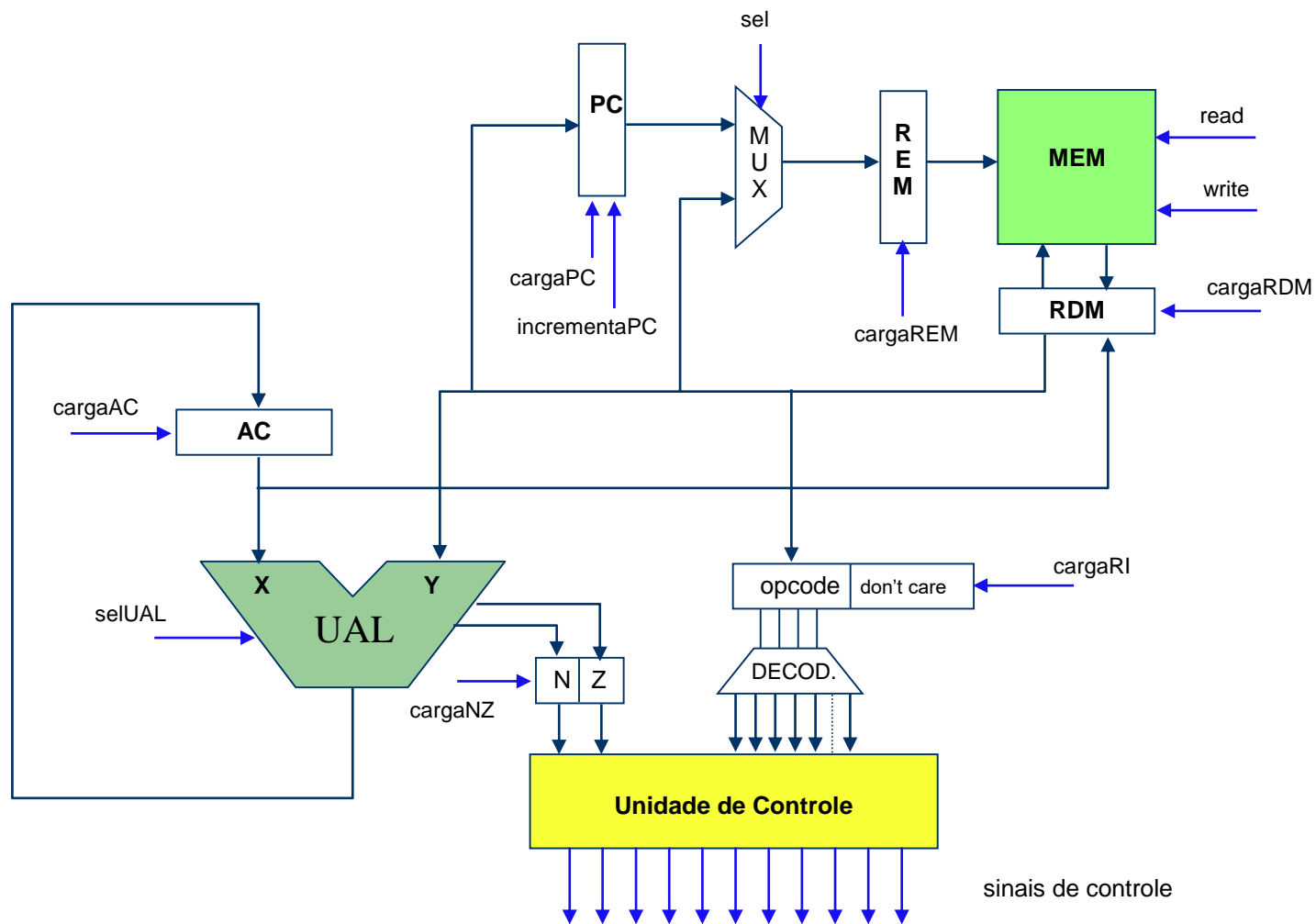
Simuladores Didáticos

<ftp://ftp.inf.ufrgs.br/pub/inf107/>

<ftp://ftp.inf.ufrgs.br/pub/inf108/>

## 4. Arquiteturas Didáticas

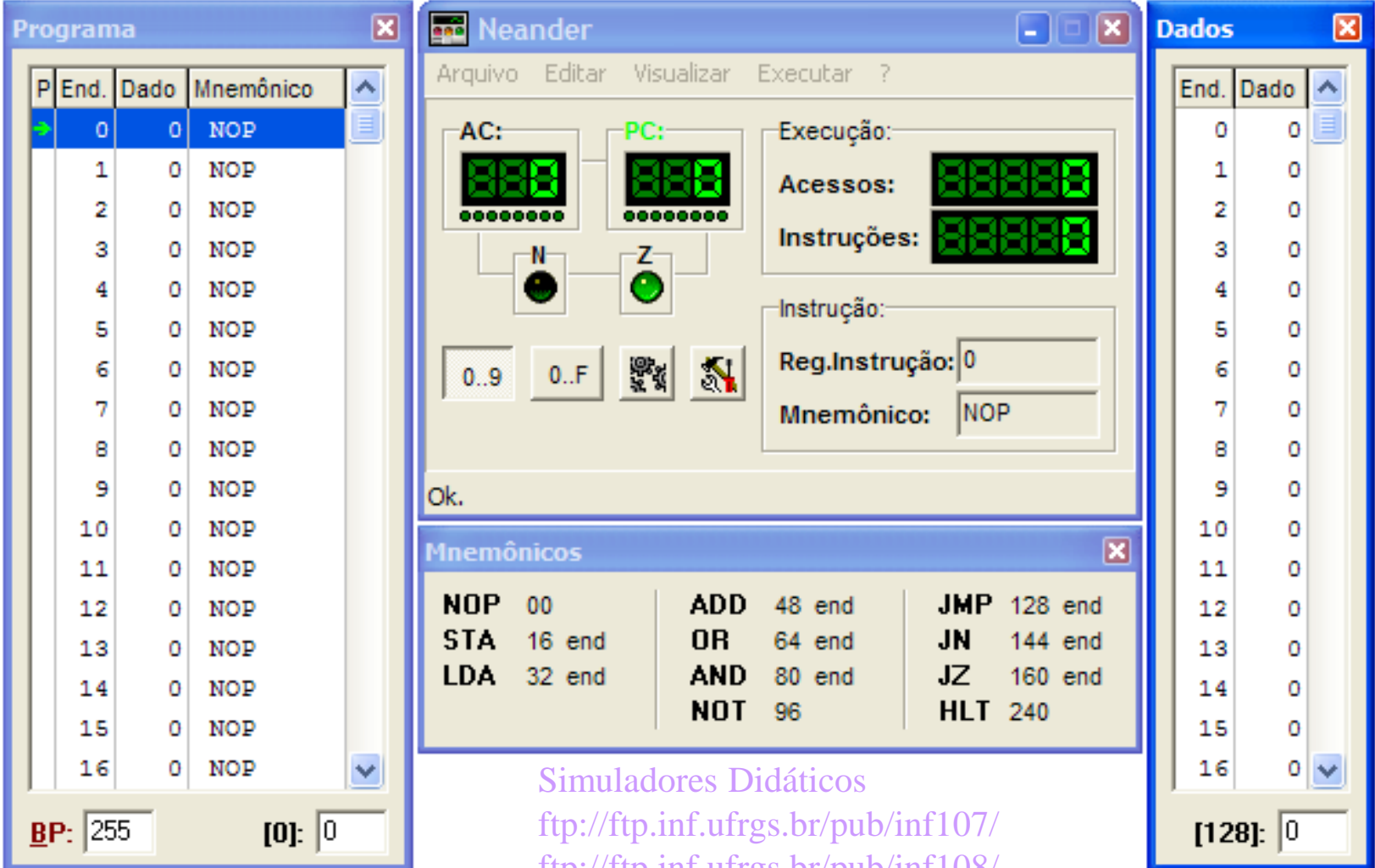
### Neander - Computador Hipotético [Weber 2001\*]



## 4. Arquiteturas Didáticas

Neander => Simulador WNeander

 **Neander** Versão 2.1  
Julho 2002  
Autores: Raul Fernando Weber  
Taizy Silva Weber  
Versão: Fabio Augusto Dal Castel Win32



The screenshot displays the Neander simulator interface with three main windows:

- Programa:** A table showing memory addresses from 0 to 16, all containing the instruction NOP.
- Neander:** The main CPU window showing:
  - AC (Accumulator) and PC (Program Counter) registers, both displaying 000.
  - Execution status: Acessos (Accesses) and Instruções (Instructions) both displaying 00000.
  - Instruction register: Reg.Instrução: 0, Mnemônico: NOP.
  - Control flags: N (Not) and Z (Zero) indicators.
  - Buttons for 0..9, 0..F, and other functions.
- Dados:** A table showing memory addresses from 0 to 16, all containing the value 0.

At the bottom of the Neander window, there is a Mnemônicos table:

NOP	00	ADD	48 end	JMP	128 end
STA	16 end	OR	64 end	JN	144 end
LDA	32 end	AND	80 end	JZ	160 end
		NOT	96	HLT	240

At the bottom of the Programa window, there are fields for BP: 255 and [0]: 0.

At the bottom of the Dados window, there is a field for [128]: 0.

Simuladores Didáticos

<ftp://ftp.inf.ufrgs.br/pub/inf107/>

<ftp://ftp.inf.ufrgs.br/pub/inf108/>

## 4. Arquiteturas Didáticas

### Evolução do Neander... Ahmes, Ramses, Cesar

#### Quadro comparativo

Arquitetura	Endereços	Dados	Nro. Instruções	Registradores
<b>NEANDER</b>	8 bits 256 bytes	8 bits Compl.2	11 instruções (OpCode: 4bits)	AC, PC, IR, Flags (N,Z) REM, RDM
<b>AHMES</b>	8 bits	8 bits	24 instruções (Neander ext.)	PC, IR, REM, RDM Flags (N, Z, C, B, V)
<b>RAMSES</b>	8 bits	8 bits	Modos de End. 4 modos x 16 instr.	PC, IR, RA, RB, RX Flags (N, Z, V, C)
<b>CESAR</b>	16 bits 64 Kbytes	16 bits	Inúmeras	R0 a R6 (uso geral) R7 (PC)

#### Simuladores Didáticos

<ftp://ftp.inf.ufrgs.br/pub/inf107/>

<ftp://ftp.inf.ufrgs.br/pub/inf108/>

[http://pt.wikipedia.org/wiki/Máquinas\\_hipotéticas\\_da\\_Universidade\\_Federal\\_do\\_Rio\\_Grande\\_do\\_Sul](http://pt.wikipedia.org/wiki/Máquinas_hipotéticas_da_Universidade_Federal_do_Rio_Grande_do_Sul)

## 4. Arquiteturas Didáticas

### Evolução do Neander... Ahmes, Ramses, Cesar

The image displays the Ahmes educational simulator interface, which is divided into several windows:

- Programa:** A table showing the program memory. The first row is highlighted, indicating the current instruction.
- Ahmes:** The main CPU window showing the current state of the processor. It includes:
  - AC (Accumulator):** A 7-segment display showing 000.
  - PC (Program Counter):** A 7-segment display showing 000.
  - Flags:** Five indicator lights for N (Negative), Z (Zero), V (Overflow), C (Carry), and B (Break).
  - Execução:** A section with two 7-segment displays for 'Acessos' and 'Instr.', both showing 00000.
  - Instrução:** Fields for 'R I:' (0) and 'Mnem:' (NOP).
  - Buttons:** '0..9' and '0..F' for entering values, and 'Ok.' for execution.
- Dados:** A table showing the data memory. The first row is highlighted, indicating the current data location.
- Mnemônicos:** A table listing the instruction set with their addresses and lengths.

**Programa**

P	End.	Dado	Mnemônico
0	0	0	NOP
1	0	0	NOP
2	0	0	NOP
3	0	0	NOP
4	0	0	NOP
5	0	0	NOP
6	0	0	NOP
7	0	0	NOP
8	0	0	NOP
9	0	0	NOP
10	0	0	NOP
11	0	0	NOP
12	0	0	NOP
13	0	0	NOP
14	0	0	NOP
15	0	0	NOP
16	0	0	NOP

**Ahmes**

Arquivo Editar Visualizar Executar ?

AC: 000 PC: 000

N Z V C B

Execução:

Acessos: 00000 Instr.: 00000

Instrução:

R I: 0 Mnem: NOP

0..9 0..F

Ok.

**Mnemônicos**

NOP	00	JMP	128 end	SHR	224
STA	16 end	JN	144 end	SHL	225
LDA	32 end	JP	148 end	ROR	226
ADD	48 end	JV	152 end	ROL	227
OR	64 end	JNV	156 end	HLT	240
AND	80 end	JZ	160 end		
NOT	96 end	JNZ	164 end		
SUB	112 end	JC	176 end		
		JNC	180 end		
		JB	184 end		
		JNB	188 end		

**Dados**

End.	Dado
128	0
129	0
130	0
131	0
132	0
133	0
134	0
135	0
136	0
137	0
138	0
139	0
140	0
141	0
142	0
143	0
144	0

[128]: 0

**Ahmes** Versão 2.1 Julho 2002

Autores: Flávil Fernando Weber Taicy Silva Weber

Versão: Fabio Augusto Dal Castel Win32

OK

Simuladores Didáticos

<ftp://ftp.inf.ufrgs.br/pub/inf107/>

<ftp://ftp.inf.ufrgs.br/pub/inf108/>

## 4. Arquiteturas Didáticas

### Evolução do Neander... Ahmes, Ramses, Cesar

**Programa**

P	End.	Dado	Mnemônico
0	0	0	NOP
1	0	0	NOP
2	0	0	NOP
3	0	0	NOP
4	0	0	NOP
5	0	0	NOP
6	0	0	NOP
7	0	0	NOP
8	0	0	NOP
9	0	0	NOP
10	0	0	NOP
11	0	0	NOP
12	0	0	NOP
13	0	0	NOP
14	0	0	NOP
15	0	0	NOP

BP: 255 [0]: 0

**Ramses v1.2**

Arquivo Editar Visualizar Executar ?

RA: [000] RB: [000] RX: [000] PC: [000]

N [ ] Z [ ] C [ ]

Execução:

Acessos: [0000] Instr.: [0000]

Instrução:

RI: 0 Mnem: NOP

0..9 0..F [ ] [ ]

Ok.

**Códigos das instruções**

NOP	0	JMP	128	end	Modo: 0: Dir: n 1: Ind: n,l 2: lmd: #n 3: ldx: nX
STR	16	JN	144	end	
LDR	32	JZ	160	end	
ADD	48	JC	176	end	
OR	64	JSR	192	end	Registrador: 0: A 2: X 1: B 3: ?
AND	80	NEG	208	r	
NOT	96	SHR	224	r	
SUB	112	HLT	240		

**Dados**

End.	Dado
128	0
129	0
130	0
131	0
132	0
133	0
134	0
135	0
136	0
137	0
138	0
139	0
140	0
141	0
142	0
143	0

[128]: 0

**Ramses** Versão 1.2 Outubro 2003

Autores: Raul Fernando Weber Taisy Siva Weber

Versão: Win32 Fábio Augusto Dal Castel

Ok

Simuladores Didáticos  
<ftp://ftp.inf.ufrgs.br/pub/inf107/>  
<ftp://ftp.inf.ufrgs.br/pub/inf108/>



## 4. Arquiteturas Didáticas

### Evolução do Neander... Ahmes, Ramses, Cesar

The image shows a screenshot of the Cesar 16 simulator interface, which is used for teaching computer architecture. It consists of three main windows: 'Programa', 'Cesar 16', and 'Dados'.

**Programa Window:** Displays a table of memory addresses and instructions.

P	Ender.	Dado	Mnemônico
	0	0	NOP
	1	0	NOP
	2	0	NOP
	3	0	NOP
	4	0	NOP
	5	0	NOP
	6	0	NOP
	7	0	NOP
	8	0	NOP
	9	0	NOP
	10	0	NOP
	11	0	NOP
	12	0	NOP
	13	0	NOP
	14	0	NOP
	15	0	NOP

**Cesar 16 Window:** Shows the internal state of the processor. It includes seven registers (R0-R7) with green LED displays. R6 is labeled as SP (Stack Pointer) and R7 as PC (Program Counter). Below the registers are status indicators for N (Negative), Z (Zero), V (Overflow), and C (Carry). There are also fields for 'Acessos' (Accesses) and 'Instr.' (Instruction). At the bottom, there are input fields for 'RI' (0) and 'Mnem:' (NOP). A 'BP:' field shows 65535 and an '[0]:' field shows 0.

**Dados Window:** Shows a memory dump with addresses and data values.

Ender.	Dado
1024	0
1025	0
1026	0
1027	0
1028	0
1029	0
1030	0
1031	0
1032	0
1033	0
1034	0
1035	0
1036	0
1037	0
1038	0
1039	0

The bottom of the simulator features a blue keyboard graphic.

Simuladores Didáticos

<ftp://ftp.inf.ufrgs.br/pub/inf107/>

<ftp://ftp.inf.ufrgs.br/pub/inf108/>

## 4. Arquiteturas Didáticas

### Evolução do Neander... Ahmes, Ramses, Cesar

#### Quadro comparativo

Arquitetura	Endereços	Dados	Nro. Instruções	Registradores
<b>NEANDER</b>	8 bits 256 bytes	8 bits Compl.2	11 instruções (OpCode: 4bits)	AC, PC, IR, Flags (N,Z) REM, RDM
<b>AHMES</b>	8 bits	8 bits	24 instruções (Neander ext.)	PC, IR, REM, RDM Flags (N, Z, C, B, V)
<b>RAMSES</b>	8 bits	8 bits	Modos de End. 4 modos x 16 instr.	PC, IR, RA, RB, RX Flags (N, Z, V, C)
<b>CESAR</b>	16 bits 64 Kbytes	16 bits	Inúmeras	R0 a R6 (uso geral) R7 (PC)

#### Simuladores Didáticos

<ftp://ftp.inf.ufrgs.br/pub/inf107/>

<ftp://ftp.inf.ufrgs.br/pub/inf108/>

[http://pt.wikipedia.org/wiki/Máquinas\\_hipotéticas\\_da\\_Universidade\\_Federal\\_do\\_Rio\\_Grande\\_do\\_Sul](http://pt.wikipedia.org/wiki/Máquinas_hipotéticas_da_Universidade_Federal_do_Rio_Grande_do_Sul)

## INFORMAÇÕES SOBRE A DISCIPLINA

**USP - Universidade de São Paulo - São Carlos, SP**

**ICMC - Instituto de Ciências Matemáticas e de Computação**

**SSC - Departamento de Sistemas de Computação**

**Prof. Fernando Santos OSÓRIO**

**Web institucional: <http://www.icmc.usp.br/ssc/>**

**Página pessoal: <http://www.icmc.usp.br/~fosorio/>**

**E-mail: [fosorio \[at\] icmc. usp. br](mailto:fosorio@icmc.usp.br) ou [fosorio \[at\] gmail. com](mailto:fosorio@gmail.com)**

**Disciplina de Organização de Computadores Digitais / BSI**

**Web disciplina: Wiki ICMC - [Http://wiki.icmc.usp.br](http://wiki.icmc.usp.br)**

**> Programa, Material de Aulas, Critérios de Avaliação,**

**> Lista de Exercícios, Trabalhos Práticos, Datas das Provas**