Organização e Arquitetura de Computadores

Projeto CPU TOLEDO
Funcionamento da Versão 2

Unidade de Controle Microprogramada e *Hardwired*
Memória ROM
Microprogramação
Microinstruções
Sinais de Controle e Clock

Prof. Raphael Garcia

CPU TOLEDO v.2

- **Características da <u>VERSÃO 1</u> da CPU TOLEDO:**
  - A execução do programa alocado na Memória RAM **não é comandada** pela Unidade de Controle, **mas pelo usuário -** conforme sequência de acionamento dos componentes.
  - Os ***Triggers*** e elementos ***Tri-States*** são acionados no simulador por clique do *mouse*.
  - Exige-se **conhecimento** da rotina de programação de cada instrução.
  - **Poderá haver erro** na execução do programa se acaso algum componente não for acionado no seu **devido instante de tempo**.

CPU TOLEDO v.2

- **Problema**
  - Como a UC fará isso de forma automática, isto é, sem intervenção humana?

- **Solução**
  - Implementar uma **Unidade de Controle** que realiza o sequenciamento das instruções de forma automatizada.
    - O acionamento dos componentes será feito de forma ordenada e sequencial pela UC, conforme sinais específicos de cada instrução.

CPU TOLEDO v.2

- **Há duas possibilidades para isso:** Utilizar uma UC Microprogramada ou uma UC *Hardwired*.

  - **UC Microprogramada**:
    - Utiliza-se uma **Memória ROM**, onde cada sinal de controle é armazenado num endereço desta memória.
    - Os sinais representam o acionamento no tempo ideal dos triggers e elementos tri-states.
    - Cada sinal é uma **microinstrução**.
    - O conjunto de microinstruções é conhecido como **microprograma.**

CPU TOLEDO v.2

- **UC *Hardwired***:
  - Utilizam-se **Portas Lógicas** ao invés de memória.
  - As combinações com as portas resultam em sinais de controle para acionamento dos componentes.

- **Observação**: UC *Hardwired* é mais rápida que UC Microprogramada. No entanto, é mais complexa para se implementar.

CPU TOLEDO v.2

- **UC implentada na CPU TOLEDO:**

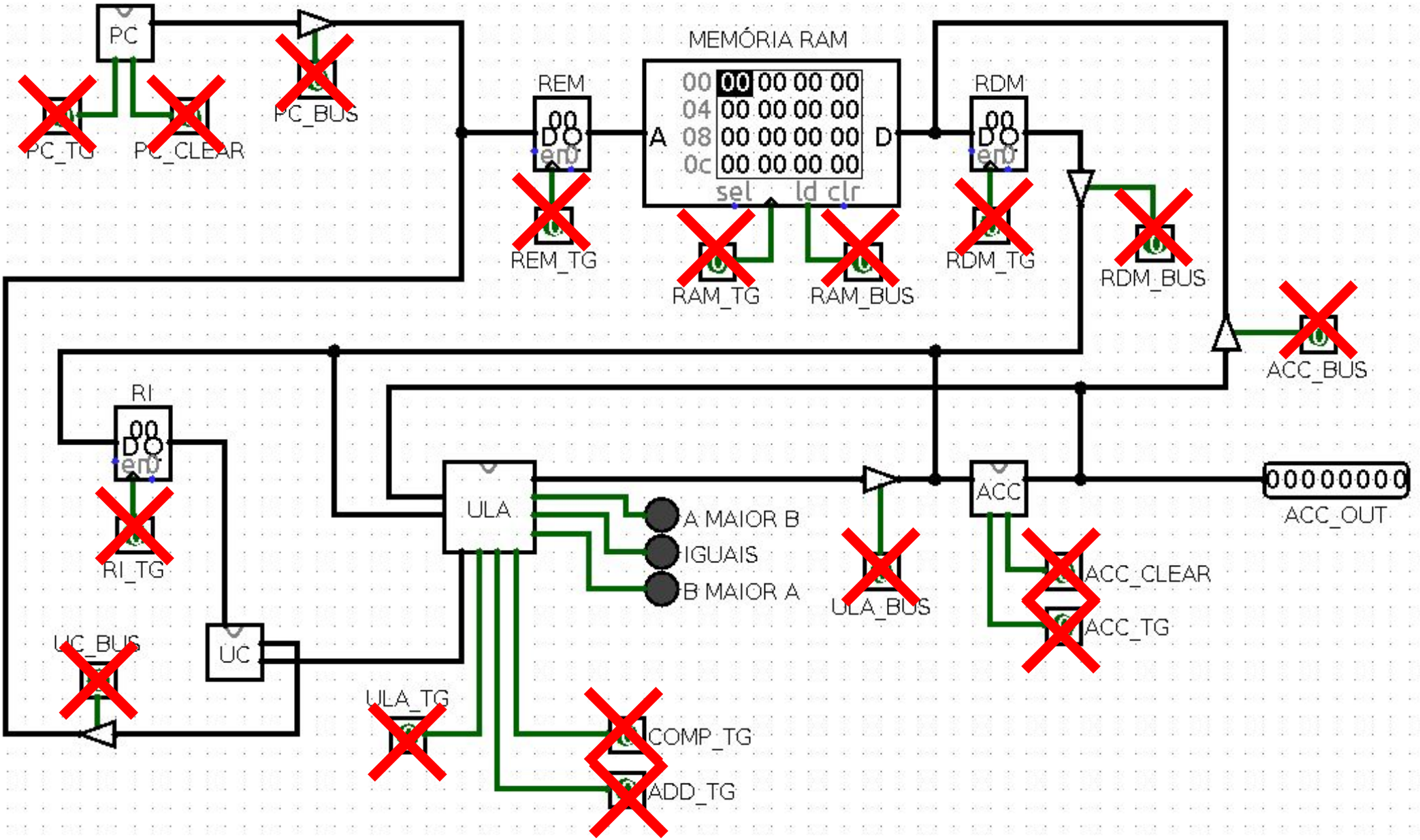
UNIDADE
DE
CONTROLE
MICROPROGRAMADA

→ O processo de **alteração** será realizado em **4 etapas**

CPU TOLEDO v.2

- **Etapas para construção da UC Microprogramada:**
  - **<u>Etapa 1</u>: Eliminação dos controles manuais.**
    - Os ***Triggers*** e elementos ***Tri-States*** presentes na CPU TOLEDO v.1 serão **substituídos** por fios vindos da Unidade de Controle.

ARQUITETURA DE COMPUTADORES - CPU TOLEDO v1.0 - MANUAL



CPU TOLEDO v.2

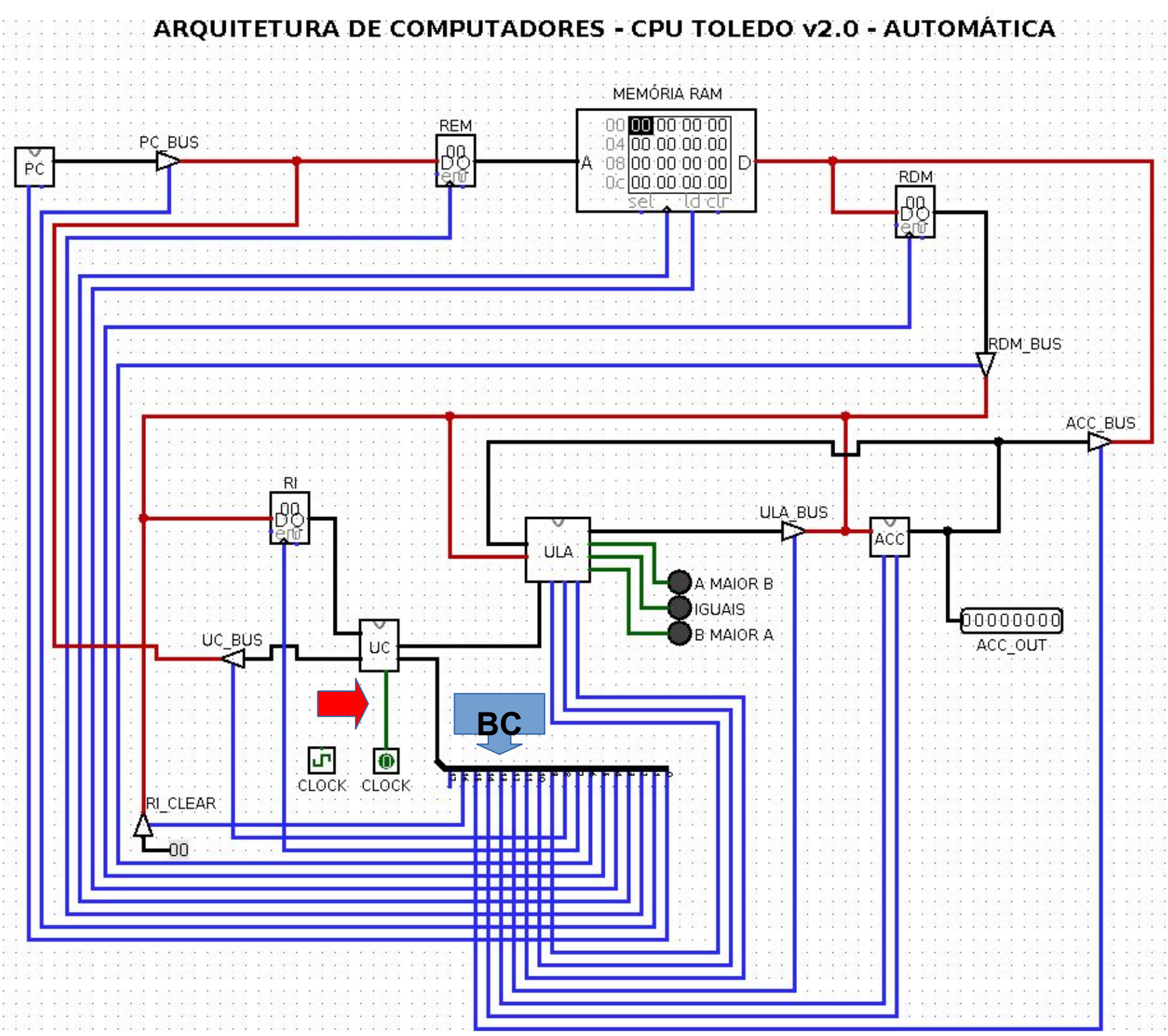
- **<u>Etapa 2</u>: Ligação da UC com todos elementos.**

  Inserção de um Sinal de Clock e Inclusão de um Barramento de Controle (BC).

  - Cada saída do BC será ligado nas entradas dos componentes onde estão as entradas para *Triggers* e *Tri-States*.
  - O **Clock** possibilita **comandar** toda a arquitetura de forma Manual ou Alternada pelo LogiSim, conforme frequência escolhida.

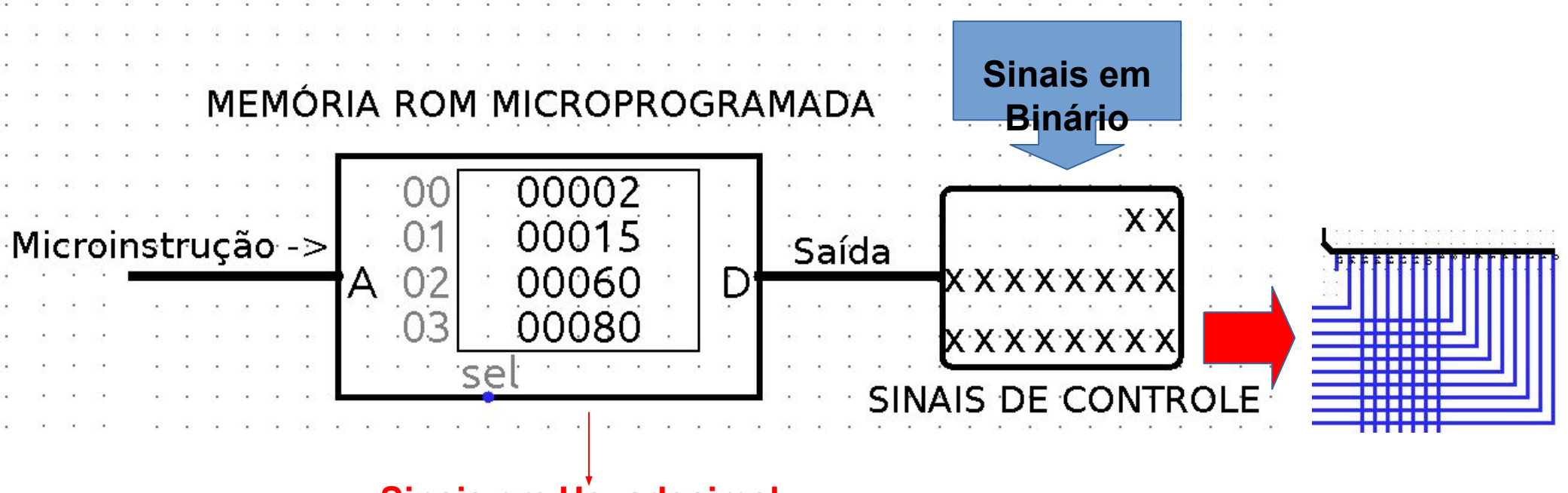
CPU TOLEDO v.2

- A UC enviará os sinais a **todos componentes** por meio do novo barramento, o BC.
- A quantidade de fios/bits deste barramento é, pelo menos, a quantidade total de *Triggers* + *Tri-states*.
- A ordem e ativação destes fios serão representadas pelos conteúdos na Memória ROM em Sistema de Numeração Hexadecimal.


ARQUITETURA DE COMPUTADORES - CPU TOLEDO v2.0 - AUTOMÁTICA
MEMÓRIA RAM
PC
PC_BUS
REM
RDM
RDM_BUS
ACC_BUS
RI
ULA
ULA_BUS
ACC
A MAIOR B
IGUAIS
B MAIOR A
00000000
ACC_OUT
UC_BUS
UC
BC
CLOCK
CLOCK
RI_CLEAR
00

CPU TOLEDO v.2

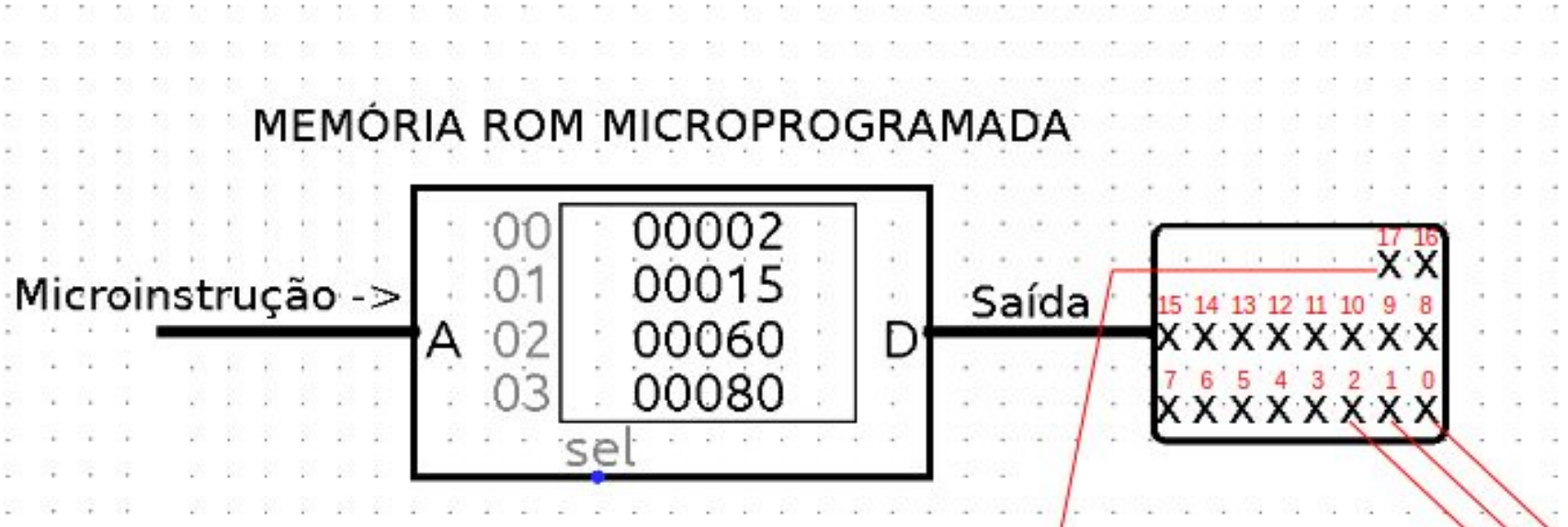
- **<u>Etapa 3</u>: Instalação de uma Memória ROM na UC.**
  - Os sinais de controle ficam armazenados na Memória ROM e representam os **códigos das microinstruções**.
  - Por uma particularidade no LogiSim, os sinais na memória serão representados em valores Hexadecimais.

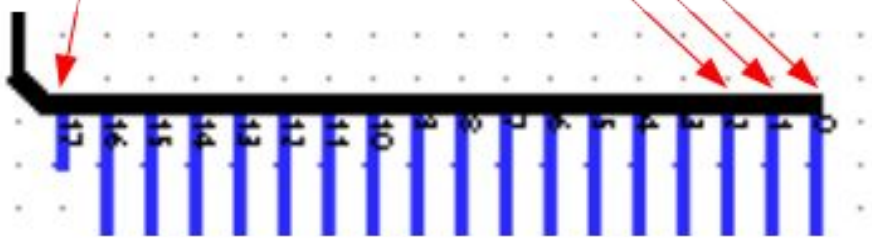


CPU TOLEDO v.2

- Etapas para construção da UC Microprogramada:

  Etapa 3 – Instalação de uma Memória ROM na UC.





Ativa/Desativa componentes

Cada bit/fio da Saída da Memória ROM está associado a um fio do barramento que, por fim, liga-se a um componente

Exemplo:

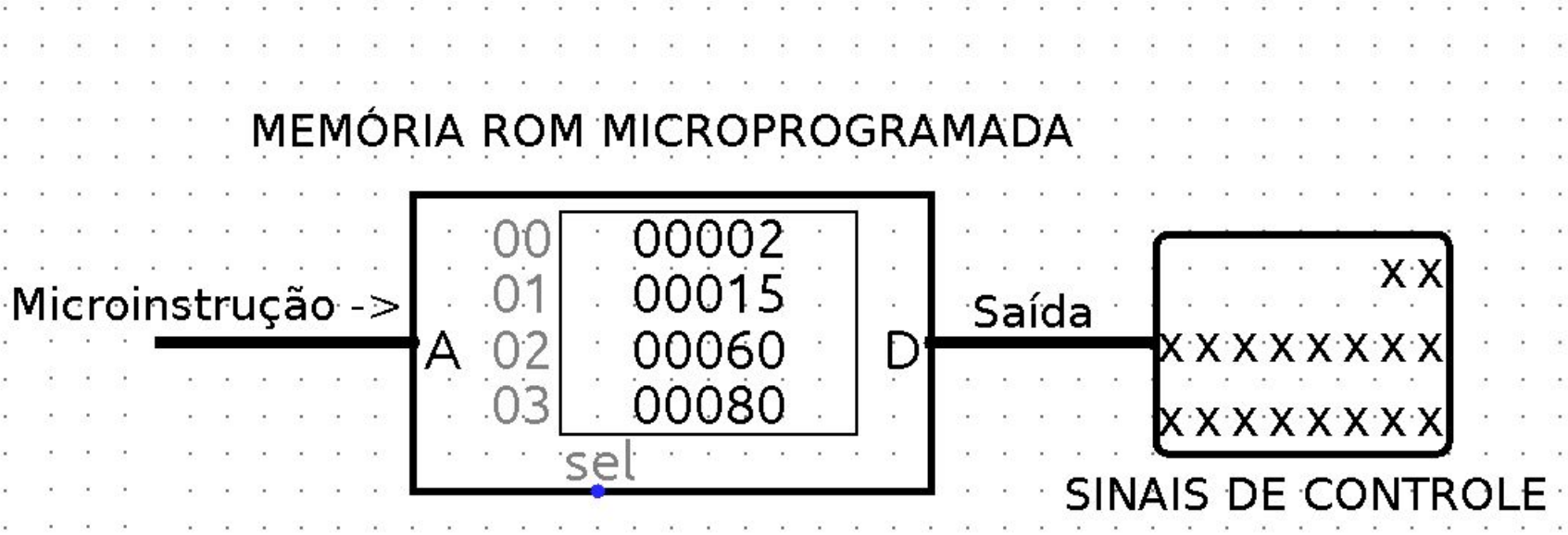
Saída 0 → Fio 0 → PC_TG
Saída 1 → Fio 1 → PC_BUS
Saída 2 → Fio 2 → REM_TG
Saída 3 → Fio 3 → RAM_TG

CPU TOLEDO v.2

- **Observação:**
  - O Uso do **bifurcador** serve para **desmembrar** os sinais, onde para cada sinal há um fio associado.
  - Os sinais saem da Memória ROM em Sistema de Numeração Hexadecimal. No entanto, o bifurcador faz a **conversão em Sistema Binário,** permitindo ativar cada componente de forma individual.

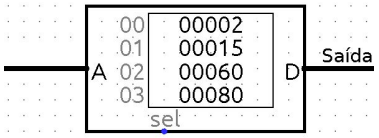
CPU TOLEDO v.2

- **Etapas para construção da UC Microprogramada:**
  - **<u>Etapa 4</u> – Programação dos sinais de controle.**
    - Em cada endereço da Memória ROM, há códigos em Hexadecimal.
    - Os códigos representam valores da posição de saída de cada componente da arquitetura



ENDEREÇO		MICRO-INSTRUÇÕES																				EXECUÇÃO DA INSTRUÇÃO			
		20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
		SINAIS DE SAÍDA DA UC																								
Hexa	Binário					RI_CLEAR	ACC_BUS	ACC_CLEAR	ACC_TG	ULA_BUS	COMP_TG	ADD_TG	ULA_TG	UC_BUS	RI_TG	RDM_BUS	RDM_TG	RAM_BUS	RAM_TG	REM_TG	PC_BUS	PC_TG	Instrução na Memória RAM	Ação após término	Label	Microinstruções
00	00000000																				1					000002
01	00000001																	1		1		1				000015
02	00000010															1	1						00		FETCH (busca)	000060
03	00000011														1											000080
04	00000100	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	#VALOR!
05	00000101	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	#VALOR!
06	00000110													1												000100
07	00000111																	1		1						000014
08	00001000															1	1						1A		LOAD (leitura)	000060
09	00001001								1																	002000
0A	00001010					1																				010000
0B	00001011														1									FETCH		000080
0C	00001100	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	#VALOR!
0D	00001101													1												000100
0E	00001110																	1		1						000014
0F	00001111												1			1	1									000260
10	00010000									1		1											2B		ADD (soma)	001400
11	00010001								1																	002000
12	00010010					1																				010000
13	00010011														1									FETCH		000080
14	00010100	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	#VALOR!
15	00010101													1												000100
16	00010110																			1						000004
17	00010111						1																			008000
18	00011000																1						3C		STORE (escrita)	000020
19	00011001						1																			008000
1A	00011010																		1							000008
1B	00011011					1																				010000
1C	00011100														1									FETCH		000080
1D	00011101	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	#VALOR!
1E	00011110													1												000100
1F	00011111																	1		1						000014
20	00100000												1			1	1						4B		COMP (comparação)	000260
21	00100001										1															000800
22	00100010					1																				010000
23	00100011														1									FETCH		000080
24	00100100	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	#VALOR!
25	00100101					1																				010000

MEMÓRIA ROM MICROPROGRAMADA



CPU TOLEDO v.2

- **Algumas ações dos microprogramas na Memória ROM**
  - Instrução **FETCH** → código: **00**
    - **Microinstruções correspondentes:**
      - **000002**
      - **000015**
      - **000060**
      - **000080**

Hexa	Binário					RI_CLEAR	ACC_BUS	ACC_CLEAR	ACC_IG	ULA_BUS	COMP_IG	ADD_IG	ULA_IG	UC_BUS	RI_IG	RDM_BUS	RDM_IG	RAM_BUS	RAM_IG	REM_IG	PC_BUS	PC_IG	Instrução na Memória RAM	Ação após término	Label	Microinstruções
00	00000000																				1					000002
01	00000001																	1		1		1				000015
02	00000010															1	1						00		FETCH (busca)	000060
03	00000011														1											000080

CPU TOLEDO v.2

- **Algumas ações dos microprogramas na Memória ROM**
  - Instrução **FETCH** → código: **00**
    - **Microinstrução: 000002**
    - → **PC_BUS = 1** e todos os demais com valor 0
      - Ação: Libera o valor que está no PC para o barramento, atingindo o REM.

Hexa	Binário					RI_CLEAR	ACC_BUS	ACC_CLEAR	ACC_IG	ULA_BUS	COMP_IG	ADD_IG	ULA_IG	UC_BUS	RI_IG	RDM_BUS	RDM_IG	RAM_BUS	RAM_IG	REM_IG	PC_BUS	PC_IG	Instrução na Memória RAM	Ação após término	Label	Microinstruções
00	00000000																				1					000002

CPU TOLEDO v.2

- **Algumas ações dos microprogramas na Memória ROM**
  - Instrução **FETCH** → código: **00**
    - **Microinstrução: 000015**
    - → **PC_TG = 1**, **REM_TG = 1** e **RAM_BUS = 1** e todos os demais com valor 0
      - Ação: Deixa o PC pronto para o próximo endereço, busca o conteúdo posicionado na Memória RAM e libera o valor no Barramento de Dados – saída D.

Hexa	Binário					RI_CLEAR	ACC_BUS	ACC_CLEAR	ACC_TG	ULA_BUS	COMP_TG	ADD_TG	ULA_TG	UC_BUS	RI_TG	RDM_BUS	RDM_TG	RAM_BUS	RAM_TG	REM_TG	PC_BUS	PC_TG	Instrução na Memória RAM	Ação após término	Label	Microinstruções
01	00000001																	1		1		1				000015

CPU TOLEDO v.2

- **Algumas ações dos microprogramas na Memória ROM**
  - Instrução **FETCH** → código: **00**
    - **Microinstrução: 000060**
    - → **RDM_TG = 1** e **RDM_BUS = 1** e todos os demais com valor 0
      - Ação: O conteúdo lido na Memória RAM é registrado no RDM e já o libera no Barramento de Dados para ir até o registrador RI.

Hexa	Binário					RI_CLEAR	ACC_BUS	ACC_CLEAR	ACC_TG	ULA_BUS	COMP_TG	ADD_TG	ULA_TG	UC_BUS	RI_TG	RDM_BUS	RDM_TG	RAM_BUS	RAM_TG	REM_TG	PC_BUS	PC_TG	Instrução na Memória RAM	Ação após término	Label	Microinstruções
02	00000010															1	1									000060

CPU TOLEDO v.2

- **Algumas ações dos microprogramas na Memória ROM**
  - Instrução **FETCH** → código: **00**
    - **Microinstrução: 000060**
    - → **RI_TG = 1** e todos os demais com valor 0
      - Ação: Armazena o conteúdo no RI, permitindo entrar na UC para Decodificação.

Hexa	Binário					RI_CLEAR	ACC_BUS	ACC_CLEAR	ACC_TG	ULA_BUS	COMP_TG	ADD_TG	ULA_TG	UC_BUS	RI_TG	RDM_BUS	RDM_TG	RAM_BUS	RAM_TG	REM_TG	PC_BUS	PC_TG	Instrução na Memória RAM	Ação após término	Label	Microinstruções
03	00000011														1											000080

CPU TOLEDO v.2

- **Como as microinstruções são executadas?**
  - A Instrução **FETCH** → código: **00**, sempre executará a MESMA sequência de microinstruções de forma ordenada:
    - **1º: 000002**
    - **2º: 000015**
    - **3º: 000060**
    - **4º: 000080**

Hexa	Binário					RI_CLEAR	ACC_BUS	ACC_CLEAR	ACC_IG	ULA_BUS	COMP_IG	ADD_IG	ULA_IG	UC_BUS	RI_IG	RDM_BUS	RDM_IG	RAM_BUS	RAM_IG	REM_IG	PC_BUS	PC_IG	Instrução na Memória RAM	Ação após término	Label	Microinstruções
00	00000000																				1		00		FETCH (busca)	000002
01	00000001																	1		1		1				000015
02	00000010															1	1									000060
03	00000011														1											000080

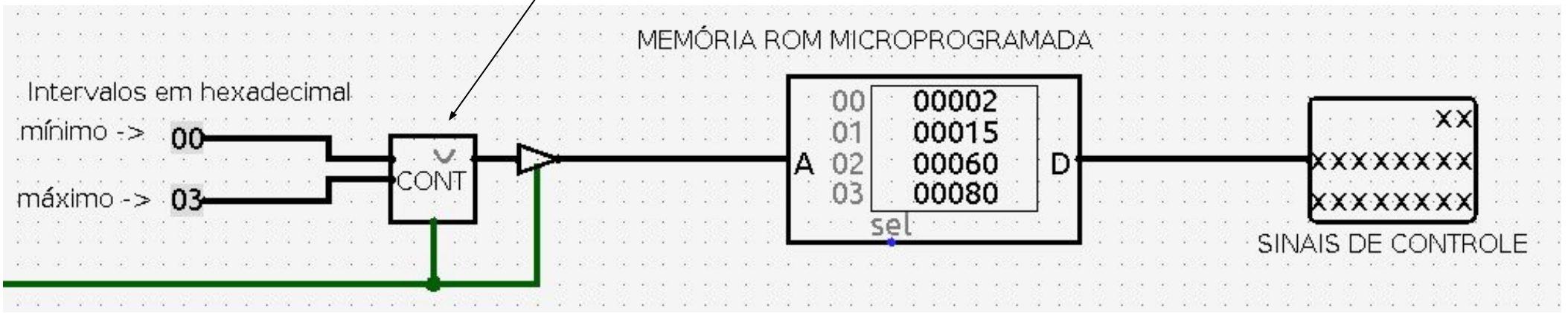
CPU TOLEDO v.2

- **Como as microinstruções são executadas?**
  - Para que a Instrução **FETCH** → código: **00**, **SEJA SEMPRE** executada de forma sequencial e ordenada, é utilizado um **CONTADOR** que vai do endereço 00 até 03
    - **00: 000002**
    - **01: 000015**
    - **02: 000060**
    - **03: 000080**

    - **Os valores 00 a 03 são correspondentes aos endereços da Memória ROM que contém cada microinstrução.**

CPU TOLEDO v.2

- **Como as microinstruções são executadas?**
  - Instrução **FETCH** - CONTADOR de **00** até **03**
    - **00**: 000002
    - **01**: 000015
    - **02**: 000060
    - **03**: 000080



CPU TOLEDO v.2

- **Algumas ações dos microprogramas na Memória ROM**
  - Instrução **LOAD X** → código: **1A**
    - **Microinstruções correspondentes:**
      - **000100**→**UC_BUS = 1** – Acessa o Barramento de Endereços e posiciona o endereço no REM
      - **000014**→**REM_TG = 1** e **RAM_BUS = 1** - Busca o valor que está no endereço Ah que corresponde à variável X (valor 6) e o transfere ao RDM
      - **000060**→**RDM_TG = 1** e **RDM_BUS = 1** - O conteúdo lido na Memória RAM é registrado no RDM e já o libera no Barramento de Dados, mas desta vez, para ir até o registrador ACC
      - **002000**→**ACC_TG = 1** – O valor é armazenado no registrador ACC
      - **010000** →**RI_CLEAR = 1** - Limpa o RI injetando 00 na UC para forcar a arquitetura a buscar uma nova instrução
      - **000080** →Libera o valor 00 na UC que “entende” que deve buscar uma nova instrução em outro endereço da Memória RAM.

CPU TOLEDO v.2

- **Algumas ações dos microprogramas na Memória ROM**
  - Instrução **ADD Y** → código: **2B**
    - **Microinstruções correspondentes:**
      - **000100**
      - **000014**
      - **000260**
      - **001400**
      - **002000**
      - **010000**
      - **000080**

      - **E assim por diante nas demais instruções...**