



INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIAS E TECNOLOGIA DO MARANHÃO

ENGENHARIA DA COMPUTAÇÃO

Francisco Kleyton de Lima Saldanha

RELATÓRIO SIMPLIFICADO: PROCESSADOR (CPU) DE 4 BITS

SANTA INÊS - MA

2025

Francisco Kleyton de Lima Saldanha

RELATÓRIO SIMPLIFICADO: PROCESSADOR (CPU) DE 4 BITS

Relatório simplificado apresentado ao Curso de Engenharia da Computação do IFMA – Campus Santa Inês, como requisito parcial para a obtenção de nota nas disciplinas do semestre.

SANTA INÊS - MA

2025

| | | |
|--------------|--|----------|
| 1 | Sumário | |
| 2 | Estrutura de códigos: | 4 |
| 3 | MOV (11): | 4 |
| 4 | DATA (00): | 4 |
| 5 | STORE e LOAD (10): | 5 |
| | <input type="checkbox"/> STORE NUM (Armazenar valor direto pelo código): | 5 |
| | <input type="checkbox"/> STORE REG (Armazenar valor puxando de um registrador): | 5 |
| | <input type="checkbox"/> LOAD (Carregar de um endereço e salvar em um registrador): | 5 |
| 6 | CIRCUITO: | 6 |
| 6.1 | Circuito Externo: | 6 |
| 6.1.1 | - Nomenclatura de Túneis: | 6 |
| 6.1.2 | - Banco de Registradores (REGFILE) e ULA externamente: | 6 |
| 6.1.3 | - Contador, Registrador de Instrução, ROM e Decodificadores: | 7 |
| 6.1.4 | - Memória RAM: | 7 |
| 6.1.5 | - Circuito Externo Completo: | 8 |
| 6.2 | Circuitos Internos: | 9 |
| 6.2.1 | - REGFILE (Banco de Registradores): | 9 |
| 6.2.2 | - Unidade de Lógica e Aritmética (ULA): | 9 |

2 Estrutura de códigos:

- Códigos de 16 bits: xxxx xxxx xxxx xxxx

- Os dois primeiros bits indicam o tipo de instrução:

- I. 11 – MOV.
- II. 00 – DATA.
- III. 10 – STORE ou LOAD.

- Outros segmentos utilizados nos exemplos e demonstrações:

- I. {xxx} – Três bits de operação (operações aritméticas).
- II. {xx} – Dois bits indicando tipo de operação com RAM (STN, STR ou LOAD).
- III. [xxxx] – Quatro bits indicando valor numérico decimal de 0 a 15.
- IV. (xxx) – Três bits indicando endereço de registrador.
- V. (xxxx xxxx) – Oito bits indicando endereço da RAM.
- VI. #X – Número em sua forma decimal.
- VII. \$XX – Valor hexadecimal de dois dígitos para endereço de RAM.

3 MOV (11):

- 11xx xxxx x[xxx x](xxx) → Alocando memória diretamente no registrador.

- MOV #NUMBER, RD.

- Exemplos testados e funcionais:

- I. 1100 0000 0[001 0](000) → C010 → MOV #2, R0
- II. 1100 0000 0[001 1](001) → C019 → MOV #3, R1
- III. 1100 0000 0[000 1](101) → C00D → MOV #1, R5

4 DATA (00):

- 00{xx x}xx(x xx)(xx x)(xxx) → Manipulando dados e executando operações.

- DATA(Operação) RA, RB, RD.

- Exemplos testados e funcionais:

- I. 00{00 0}00(0 00)(00 1)(010) → 000a → ADD R0, R1, R2
- II. 00{11 1}00(0 01)(00 0)(100) → 3844 → MULT R1, R0, R4
- III. 00{10 1}00(0 00)(10 1)(110) → 282E → SUB R0, R5, R6
- IV. 00{11 0}00(1 10)(00 0)(100) → 3184 → MULT R6, R0, R4

5 STORE e LOAD (10):

- **STORE NUM (Armazenar valor direto pelo código):**

- 10(00) [xxxx] (xxxx xxxx) → armazenar número diretamente.

- STORE NUM #NUM, ENDRESS → (STN).

Exemplos testados e funcionais:

- I. 1000 [1010] (0000 0101) → 8A05 → STORE NUM #10, \$05
- II. 1000 [0100] (0000 0010) → 8402 → STORE NUM #4, \$02

- **STORE REG (Armazenar valor puxando de um registrador):**

- 10(11) x(xxx) (xxxx xxxx) → armazenar de um registrador.

- STORE REG Rr, ENDRESS → (STR).

- Exemplos testados e funcionais:

- I. 1011 0(000) (0000 0000) → B000 → STORE REG R0, \$00
- II. 1011 0(001) (0000 0001) → B101 → STORE REG R1, \$01

- **LOAD (Carregar de um endereço e salvar em um registrador):**

- 10(10 ou 01) x(xxx) (xxxx xxxx) → carregar de um endereço e armazenar.

- LOAD ENDRESS, Rd → LOAD.

- Exemplos testados e funcionais:

- I. 1010 0(000) (0000 0101) → A005 → LOAD \$05, R0
- II. 1010 0(001) (0000 0010) → A102 → LOAD \$02, R1

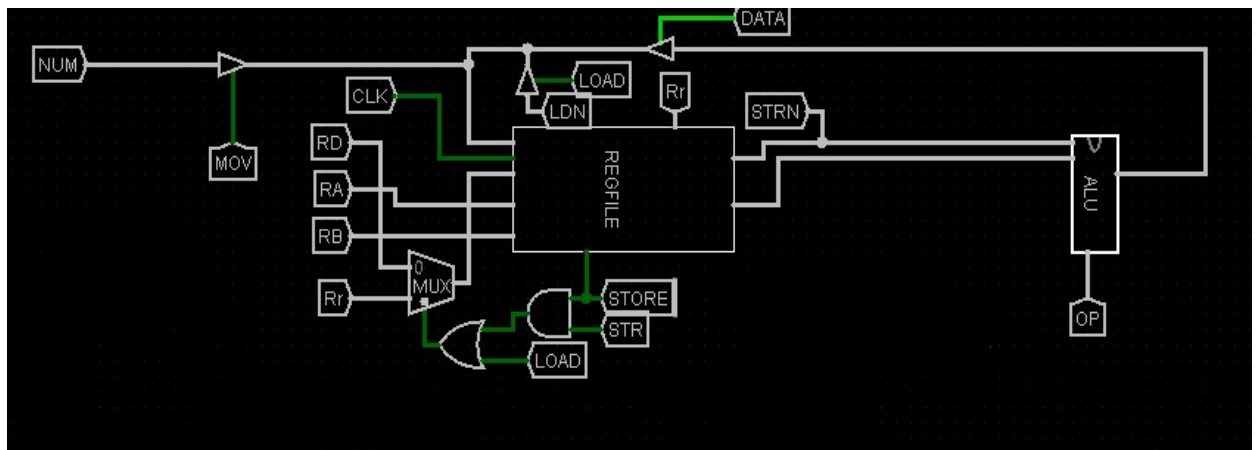
6 CIRCUITO:

6.1 Circuito Externo:

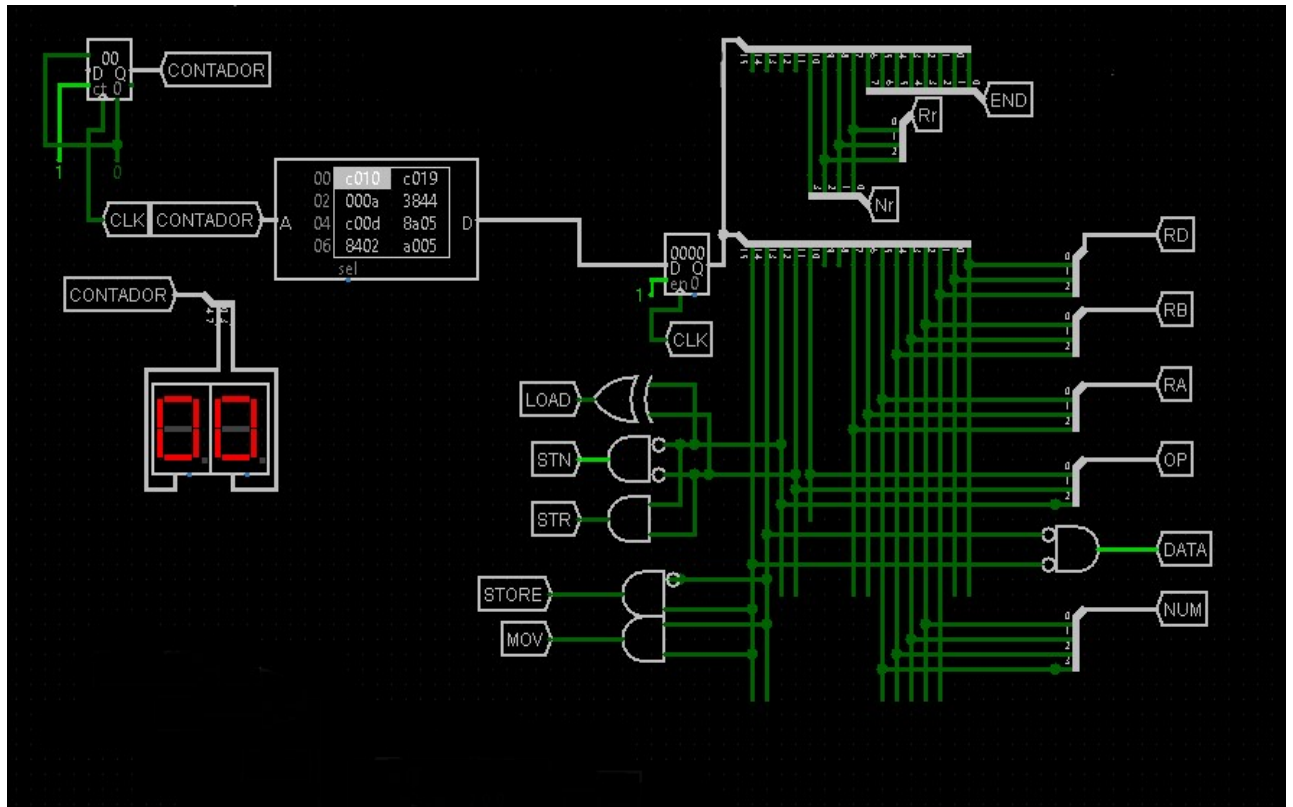
6.1.1 - Nomenclatura de Túneis:

- CLK: Clock universal.
- MOV: 2 Bits de indicação de instrução MOV.
- DATA: 2 Bits de indicação de instrução DATA.
- OP: 3 Bits de indicação de tipo de operação aritmética na ULA.
- NUM: 4 Bits do valor numérico que será registrado ou operado.
- STORE: 2 Bits de indicação de instrução STORE e LOAD.
- STR: 2 Bits de Indicação de STORE REG.
- STN: 2 Bits de indicação de STORE NUM.
- Nr: 4 Bits de valor numérico a ser salvo na RAM via STN.
- Rr: 3 Bits de endereço de registrador em operações de STORE e LOAD.
- LOAD: 2 Bits de indicação de LOAD.
- STRN: 4 Bits de valor numérico a ser salvo na RAM via STR.
- LDN: 4 Bits de valor numérico lido na RAM.
- END: 8 Bits de endereço utilizado para a RAM.
- RA: 3 Bits de Endereço para registrador primário.
- RB: 3 Bits de Endereço para registrador secundário.
- RD: 3 Bits de Endereço para registrador destino.
- CONTADOR: 8 Bits de endereçamento para RAM e ROM.

6.1.2 - Banco de Registradores (REGFILE) e ULA externamente:

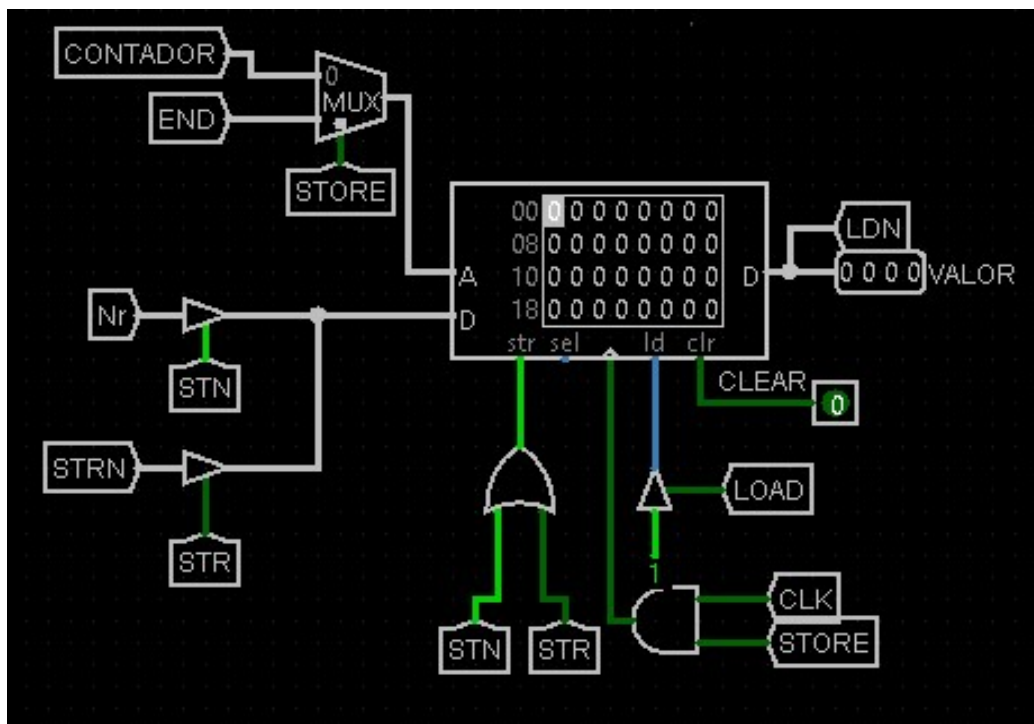


6.1.3 - Contador, Registrador de Instrução, ROM e Decodificadores:



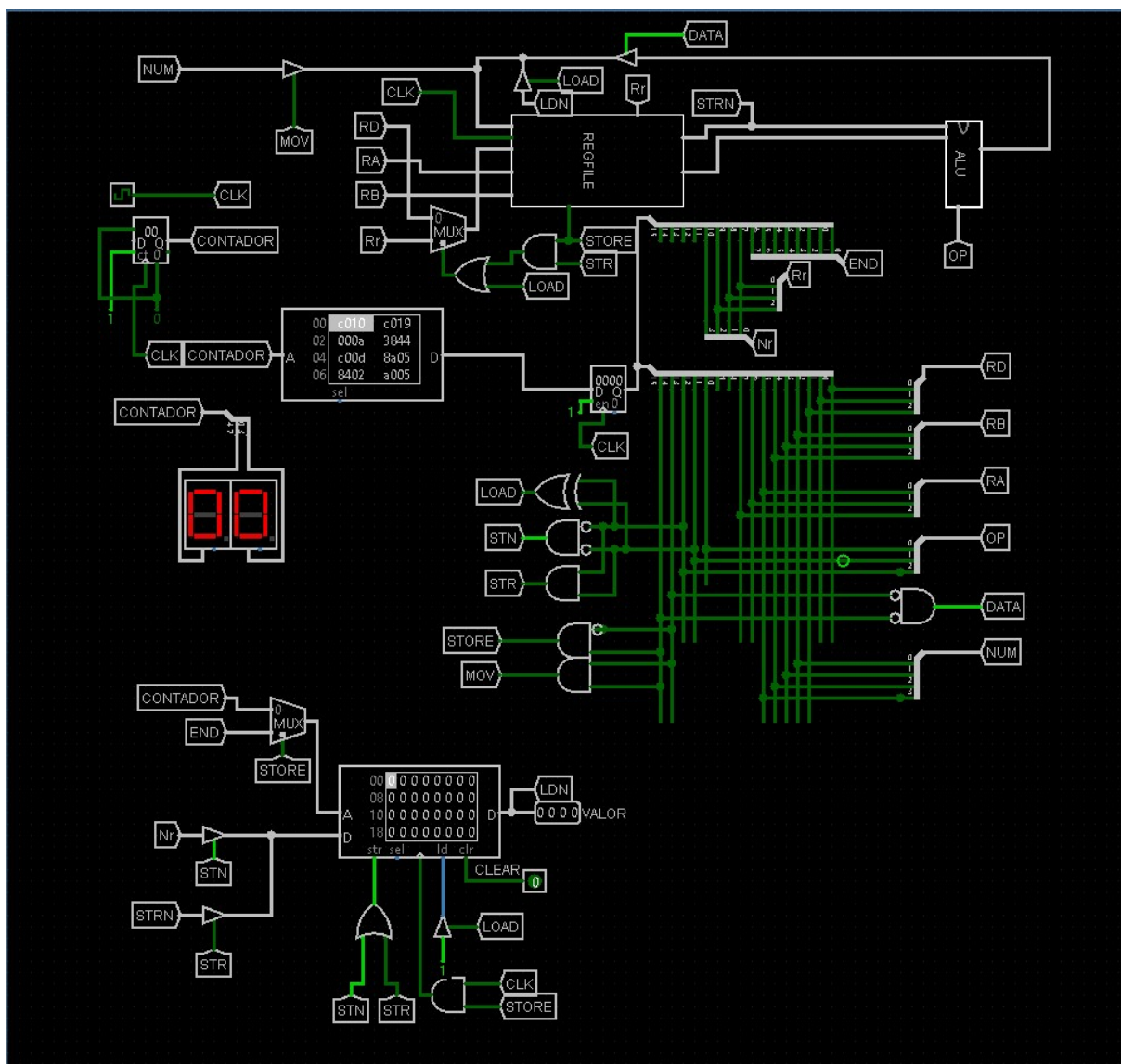
Barramentos e Contador – 2025

6.1.4 - Memória RAM:



RAM - 2025

6.1.5 - Circuito Externo Completo:



The screenshot displays a digital logic circuit simulation of a 74181 ALU. The circuit features an 8-bit register (R0-R7) and a 4-bit data multiplexer (DMUX). The inputs are INP (0000), Clock (0), and RD (0000). The outputs are A (0000) and B (0000). The circuit is configured to perform a specific operation, likely addition or subtraction, as indicated by the 'STORE' input being 0.