Desenvolvimento do MIPS em simulador lógico

Apresentação das versões ciclo único e multiciclo

Iago Gouveia Gurgel e Luiz Paulo Assis Barbosa

Universidade Federal do Rio Grande do Norte – UFRN
Centro de Ensino Superior do Seridó – CERES
Departamento de Ciências Exatas e Aplicadas – DCEA
Bacharelado em Sistemas de Informação – BSI

Laboratório de Elementos do Processamento da Informação – LabEPI

20 de dezembro de 2023

Conjunto de instruções

Nome	Formato	Operação	Opcode/Funct
add	R	R[rd] = R[rs] + R[rt]	$0/20_{hex}$
sub	R	R[rd] = R[rs] - R[rt]	$0/22_{hex}$
and	R	R[rd] = R[rs] & R[rt]	$0/24_{hex}$
or	R	R[rd] = R[rs] | R[rt]	$0/25_{hex}$
nand	R	$R[rd] = \sim (R[rs] \& R[rt])$	$0/26_{hex}$
nor	R	$R[rd] = \sim (R[rs] \mid R[rt])$	$0/27_{hex}$
slt	R	R[rd] = (R[rs] < R[rt]) ? 1 : 0	$0/2a_{hex}$
addi	I	R[rt] = R[rs] + SignExtImm	8_{hex}
lw	I	R[rt] = M[R[rs] + SignExtImm]	23_{hex}
sw	I	M[R[rs] + SignExtImm] = R[rt]	$2b_{hex}$
beq	I	if(R[rt] == R[rs]) PC = PC + 4 + BranchAddr	4_{hex}
j	J	PC = JumpAddr	2_{hex}

Formatos de instrução

Formato R					
Opcode	Rs	Rt	Rd	Shamt	Funct
31...26	25...21	20...16	15...11	10...6	5...0

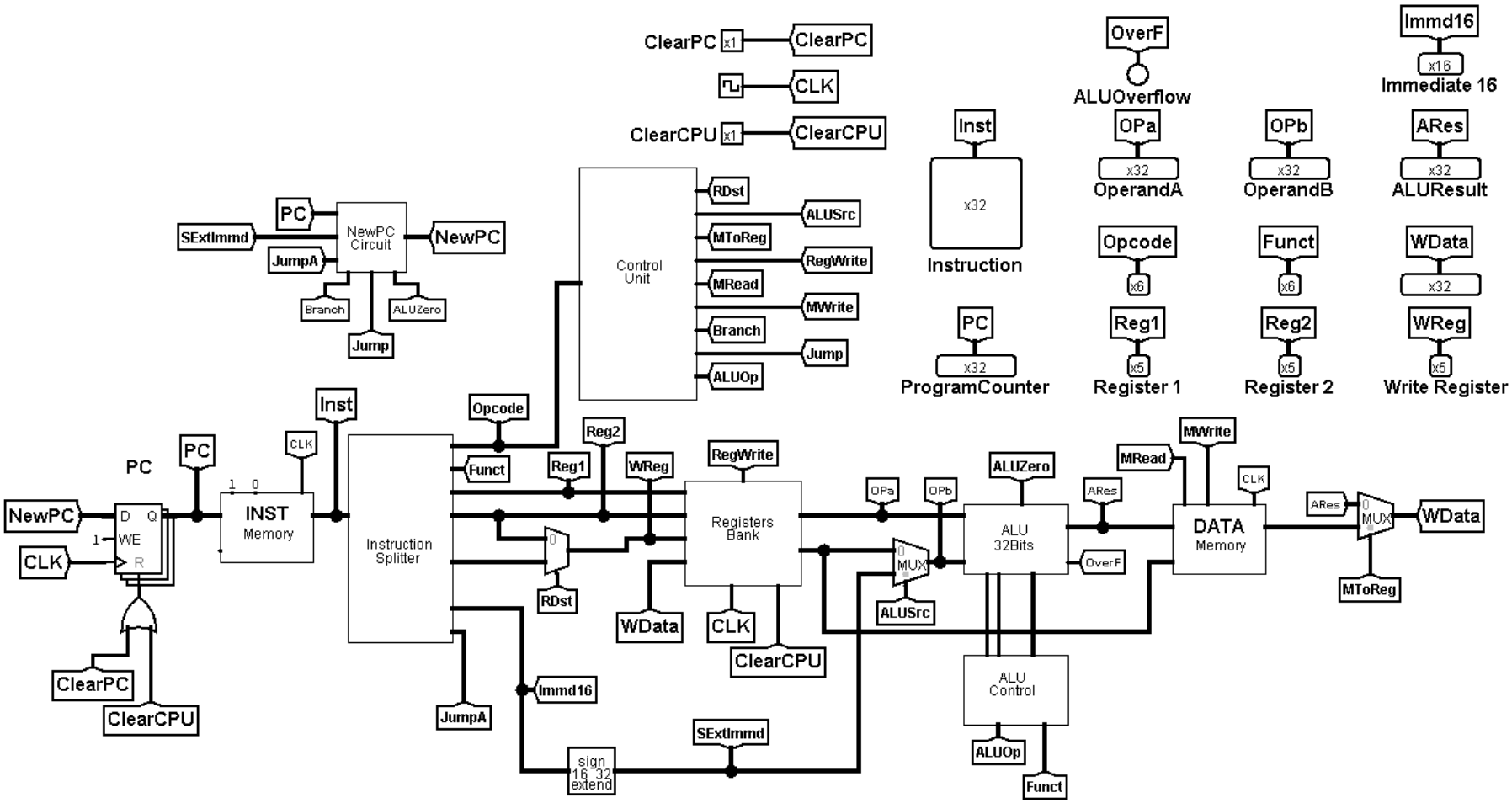
Formato I			
Opcode	Rs	Rt	Imediato
31...26	25...21	20...16	15...0

Formato J	
Opcode	Endereço
31...26	25...0

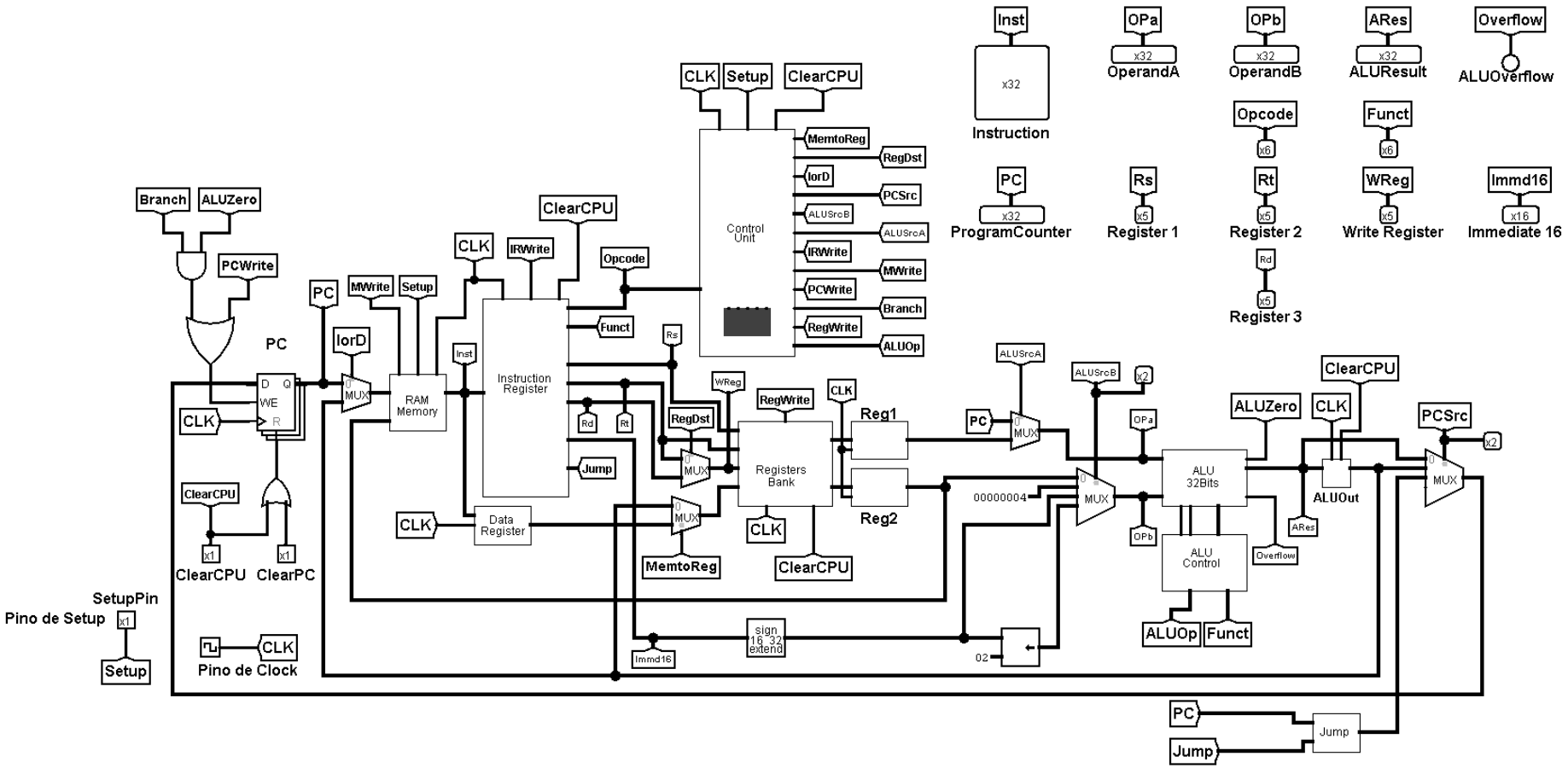
Exemplo de programa

```
1   .data 0x0100
2
3   a: .word 5
4   b: .word 12
5
6   .text 0x0000
7   main:
8           addi $t0, $zero, 0x0100
9           lw $t1, 0($t0)
10          addi $t2, $t1, 2
11          lw $t3, 4($t0)
12          add $t4, $t1, $t2
13          and $t1, $zero, $t1
14          and $t2, $zero, $t2
15          beq $t4, $t3, end
16          sub $t4, $t3, $t2
17  end:
18          j store
19          add $zero, $zero, $zero
20          add $zero, $zero, $zero
21          add $zero, $zero, $zero
22          add $zero, $zero, $zero
23
24  store:
25          sw $t4, 8($t0)
```

MIPS Ciclo Único



MIPS Multiciclo



Constrastes das implementações

De maneira histórica, os microprocessadores ciclo único, que executam uma instrução por ciclo, foram os primeiros a serem desenvolvidos, mas para essa forma de implementação, como descrito em [2], existem desvantagens como:

- Limitação da frequência do Clock
- Limitação ao uso das unidades funcionais

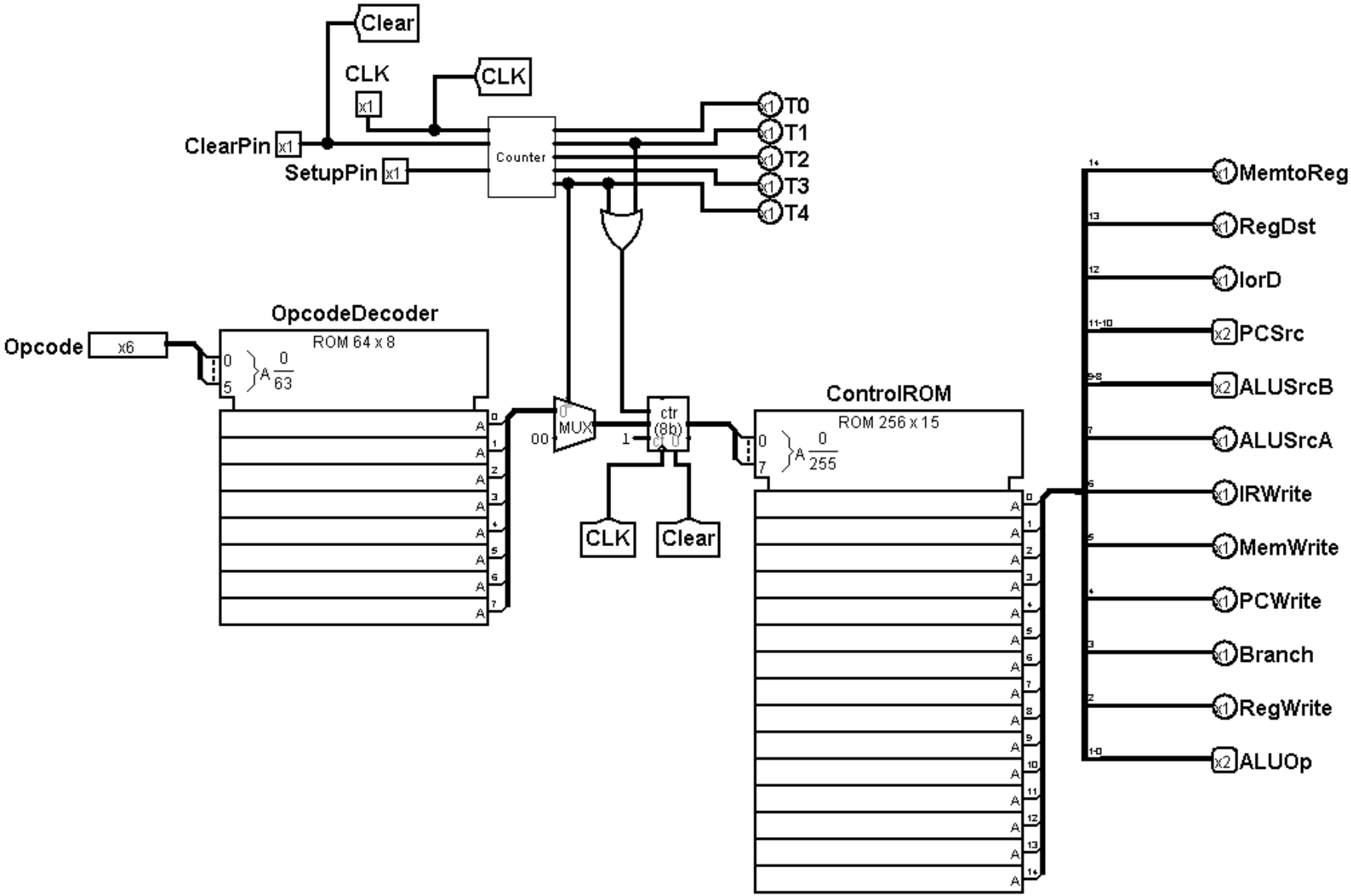
Como principais contrastes entre as versões apresentadas, pode-se ver que:

- Instruções particionadas em estágios
  - Busca da instrução e Incremento do PC
  - Decodificação da instrução e leitura dos registradores
  - Utilização da ULA
  - Lendo ou escrevendo dados na memória
  - Guardando dados no registrador destino
- Melhor utilização da Unidade Lógica Aritmética
- Diferenças em implementação de Unidade de Controle

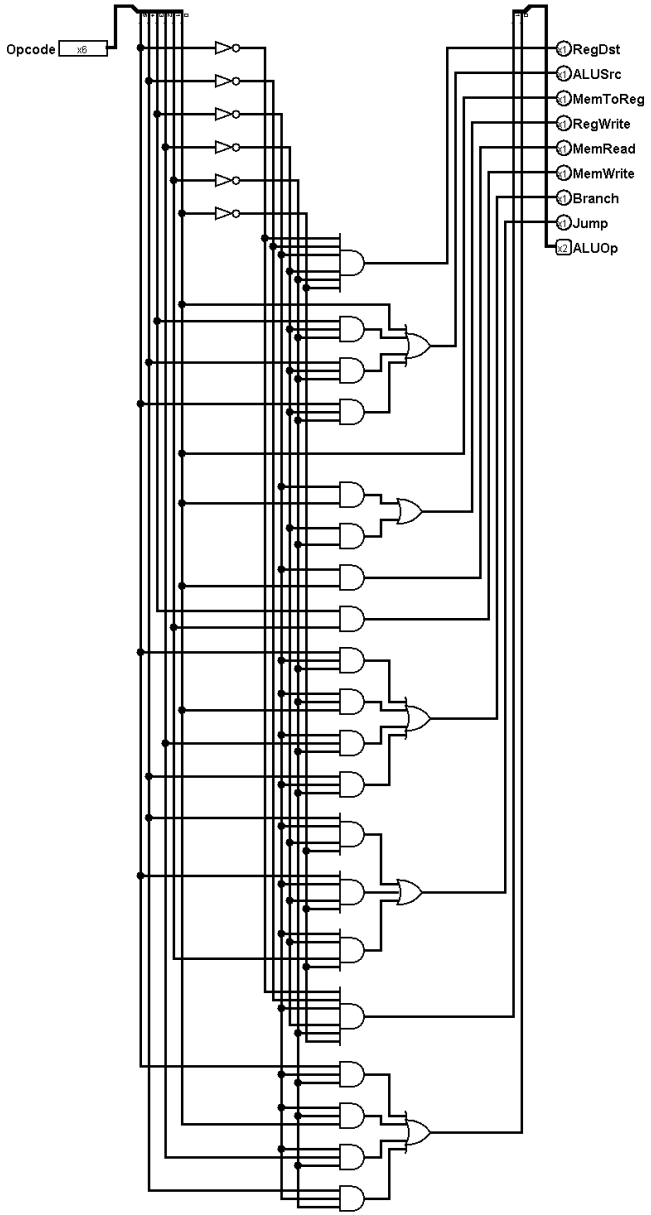
Matriz de controle X Memória de controle

Descrito por Malvino em [1], são apresentadas duas maneiras de implementação de unidades de controle, uma que utiliza uma matriz de controle usando lógica booleana e outra que utiliza uma memória ROM programada com os sinais de controle para cada instrução.
Quando trata-se de conjuntos de instruções maiores, principalmente em processadores multiciclo, é notável a vantagem em nível de complexidade que guardar os sinais de controle em uma memória de controle pode apresentar quando comparada à matriz de controle.

Memória de controle



Matriz de controle



Referências I


Jerald A Brown and Albert Paul Malvino.
Digital Computer Electronics Third Edition.
McGraw Hill, 1992.


Madhav Singh Solanki and Anuska Sharma.
A review paper on the difference between single-cycle and multi cycle processor.
International Journal of Innovative Research in Computer Science & Technology, 9(6):86–90, 2021.