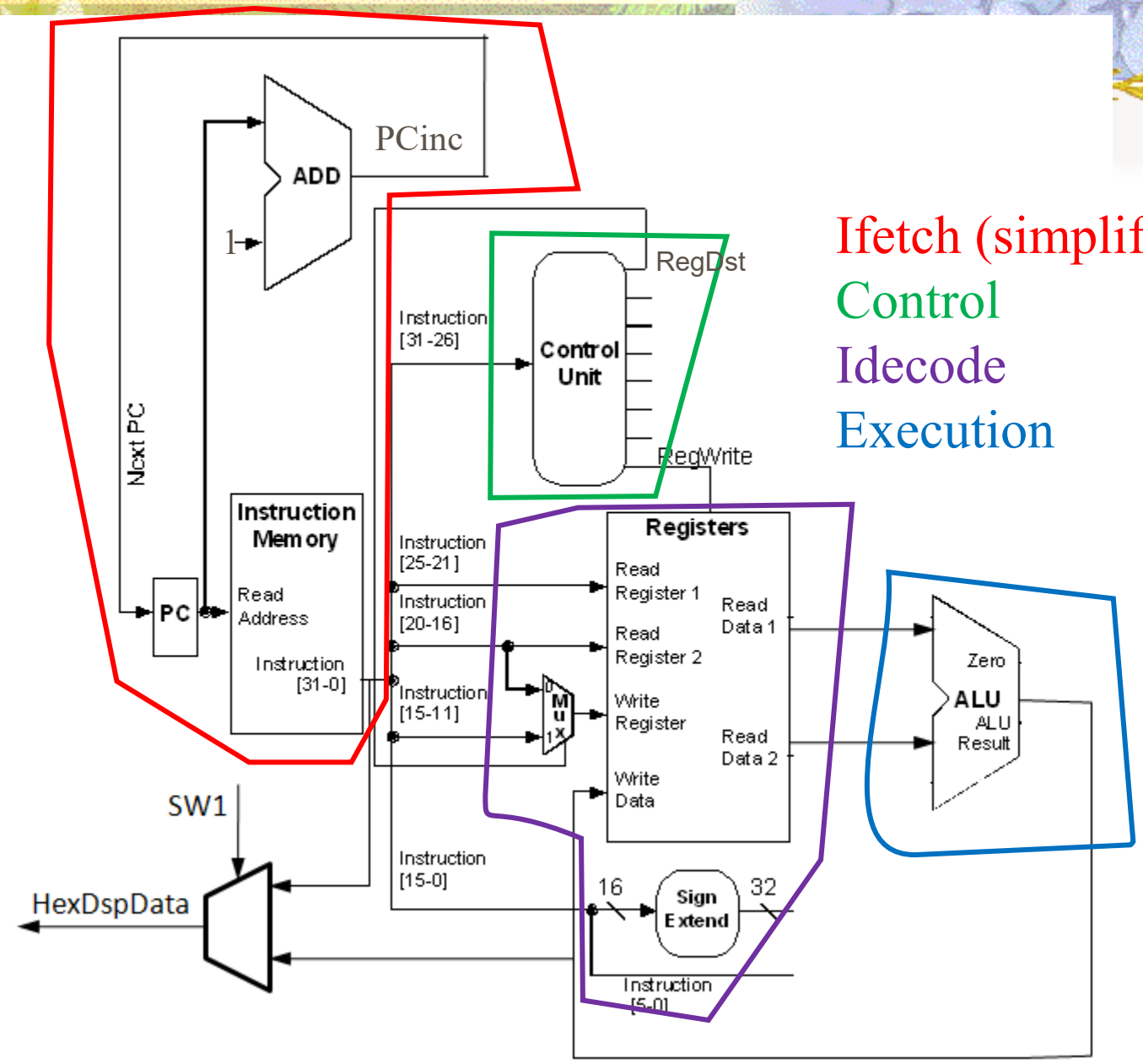
Laboratório de Arquitetura de Computadores

Profa. Yeda

Aula 2 – Introduções ao Processador MIPS



Ifetch (simplificado)

Control

Idecode

Execution

Mnemonic	Format	Opcode Field	Function Field	Instruction
Add	R	0	32	Add
Addi	I	8	-	Add Immediate
Addu	R	0	33	Add Unsigned
Sub	R	0	34	Subtract
Subu	R	0	35	Subtract Unsigned
And	R	0	36	Bitwise And
Or	R	0	37	Bitwise OR
Sll	R	0	0	Shift Left Logical
Srl	R	0	2	Shift Right Logical
Slt	R	0	42	Set if Less Than
Lui	I	15	-	Load Upper Immediate
Lw	I	35	-	Load Word
Sw	I	43	-	Store Word
Beq	I	4	-	Branch on Equal
Bne	I	5	-	Branch on Not Equal
J	J	2	-	Jump
Jal	J	3	-	Jump and Link (used for Call)
Jr	R	0	8	Jump Register (used for Return)

TIPOS DE INSTRUÇÕES

Table 14.1 MIPS 32-bit Instruction Formats.

Field Size	6-bits	5-bits	5-bits	5-bits	5-bits	6-bits
R- Format	Opcode	Rs	Rt	Rd	Shift	Function
I - Format	Opcode	Rs	Rt	Address/immediate value		
J - Format	Opcode	Branch target address				

MIPS R-format

- Instruction fields
  - op: operation code (opcode)
  - rs: first source register number
  - rt: second source register number
  - rd: destination register number
  - shamt: shift amount (00000 for now)
  - funct: function code (extends opcode)

Mnemonic	Format	Opcode Field	Function Field	Instruction
Add	R	0	32	Add
Addi	I	8	-	Add Immediate
Addu	R	0	33	Add Unsigned
Sub	R	0	34	Subtract
Subu	R	0	35	Subtract Unsigned
And	R	0	36	Bitwise And
Or	R	0	37	Bitwise OR
Sll	R	0	0	Shift Left Logical
Srl	R	0	2	Shift Right Logical
Slt	R	0	42	Set if Less Than
Lui	I	15	-	Load Upper Immediate
Lw	I	35	-	Load Word
Sw	I	43	-	Store Word
Beq	I	4	-	Branch on Equal
Bne	I	5	-	Branch on Not Equal
J	J	2	-	Jump
Jal	J	3	-	Jump and Link (used for Call)
Jr	R	0	8	Jump Register (used for Return)

Introdução

- Linguagem de alto nível x linguagem assembly x linguagem de máquina: (P1.2*)
  - C = A + B linguagem C
  - add C, A, B linguagem assembly
  - 00000000011000010001000000100000 ling.Máquina

00611020 Hexa

- Princípio da simplicidade e regularidade: operações simples, básicas e padronizadas (P2.2*)
  - Ex.: soma e subtração sempre possuem 3 operandos
  - Então como faz D = A + B + C ?
    - add D, A, B
    - add D, D, C

* Seção do livro do Patterson

Represetação das Instruções

- Instruções são codificadas em binário (Código de máquina)
- Instruções MIPS: Codificação em palavras de 32-bits
- Número dos Registradores
  - $zero é reg 0: sempre com valor zero
  - $at é reg 1: temporário para constantes grandes
  - $v0 - $v1 são reg's 2 – 3: resultado funções e avaliação expressões
  - $a0 - $a3 são reg's 4 – 7: argumentos de funções
  - $t0 – $t7 são reg's 8 – 15: temporários
  - $s0 - $s7 são reg's 16 – 23: temporários salvos
  - $t8 – $t9 são reg's 24 – 25: temporários
  - $k0 – $k1 são reg's 26 – 27: reservado para o kernel
  - $gp, $sp, $fp são reg's 28 – 30: ponteiros global, pilha e quadro
  - $ra é reg 31: reservado para endereço de retorno de função

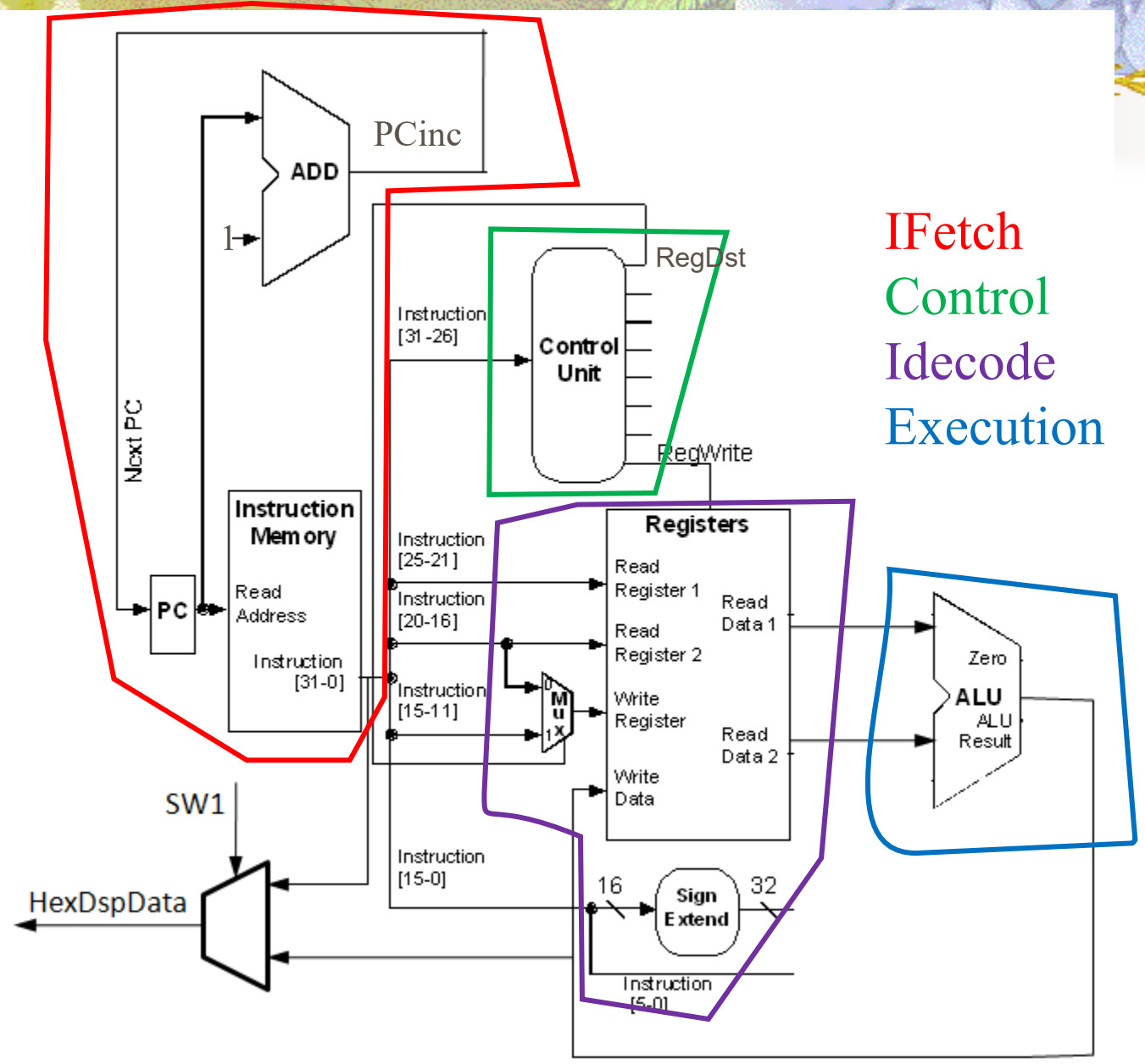
Exemplo R-format

op	rs	rt	rd	shamt	funct
6 bits	5 bits	5 bits	5 bits	5 bits	6 bits

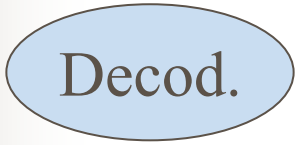
```
add $t0, $s1, $s2
```

special	$s1	$s2	$t0	0	add
0	17	18	8	0	32
000000	10001	10010	01000	00000	100000

$0000.0010.0011.0010.0100.0000.0010.0000_2 = 02324020_{16}$


IFetch
Control
Idecode
Execution
PCinc
ADD
1
Next PC
RegDst
Instruction [31-26]
Control Unit
RegWrite
Instruction Memory
Read Address
Instruction [31-0]
PC
Registers
Instruction [25-21]
Instruction [20-16]
Instruction [15-11]
Read Register 1
Read Register 2
Write Register
Write Data
Read Data 1
Read Data 2
M u x
0
1
Zero
ALU
ALU Result
SW1
HexDspData
Instruction [15-0]
16
Sign Extend
32
Instruction [5-0]

Processador MIPS Simplificado



Decodificação

- Blocos Operacionais
  - Reconhece os tipos de instruções,
  - Define endereço (índice) dos registradores de origem (entradas) e destino (saída) ,
  - “Conecta” registradores de entrada ao barramento de dados,
  - “Conecta” registrador de saída ao barramento de dados.

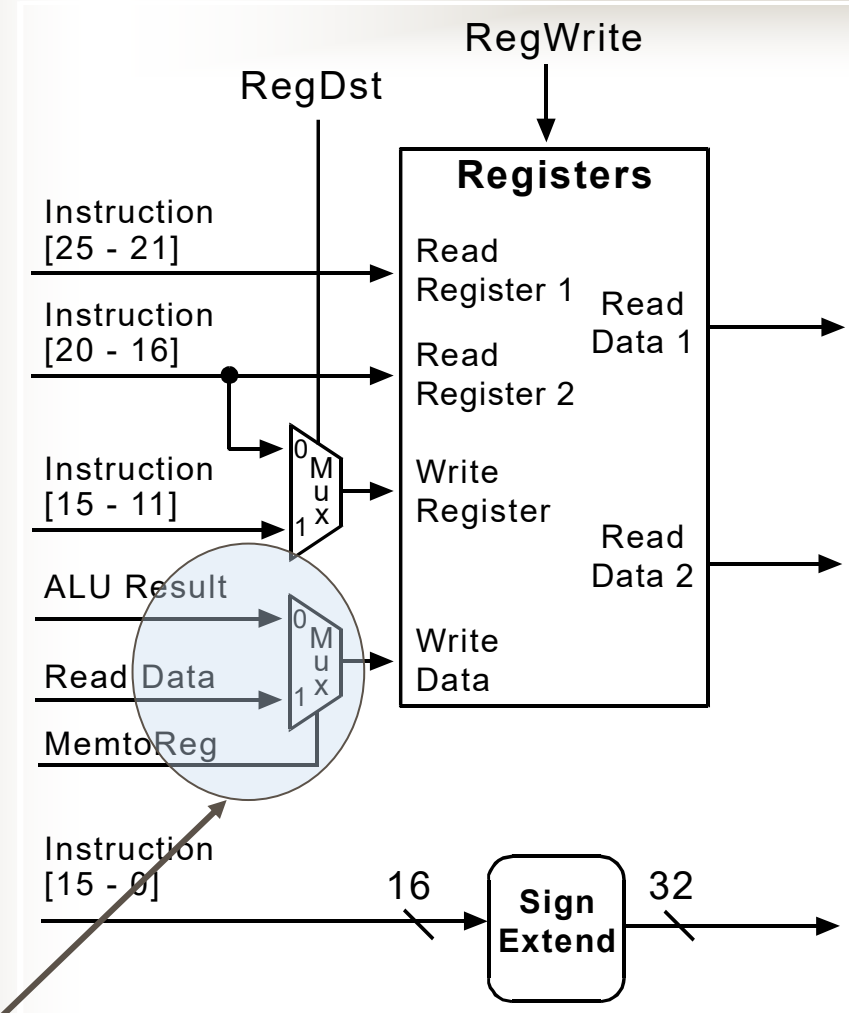
Registrador Origem e Destino?

Table 14.1 MIPS 32-bit Instruction Formats.

Field Size	6-bits	5-bits	5-bits	5-bits	5-bits	6-bits
R- Format	Opcode	Rs	Rt	Rd	Shift	Function
I - Format	Opcode	Rs	Rt	Address/immediate value		
J - Format	Opcode	Branch target address				

- Rs: 1º registrador fonte
- Rt: 2º registrador fonte ou registrador destino
- Rd: registrador destino

- Um multiplexador deve escolher



Registrador pode armazenar resultados de:
ULA
Memória

Registrador destino pode ser definido por:
rd para R-format
rt para I-format

I-format: endereço de offset precisa ser extendido.

Implementar no Exp3

Figure 14.6 Block Diagram of MIPS Decode Unit.

Processador MIPS Simplificado

Controle

- Blocos Operacionais
  - Gera sinais para as demais unidades de acordo com a instrução.

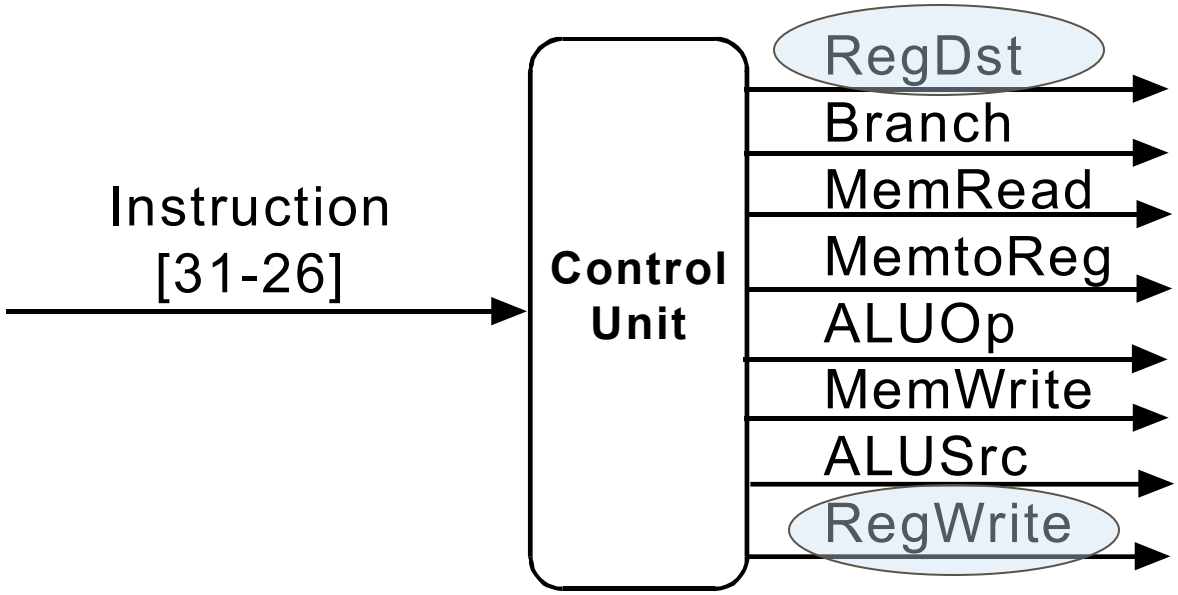
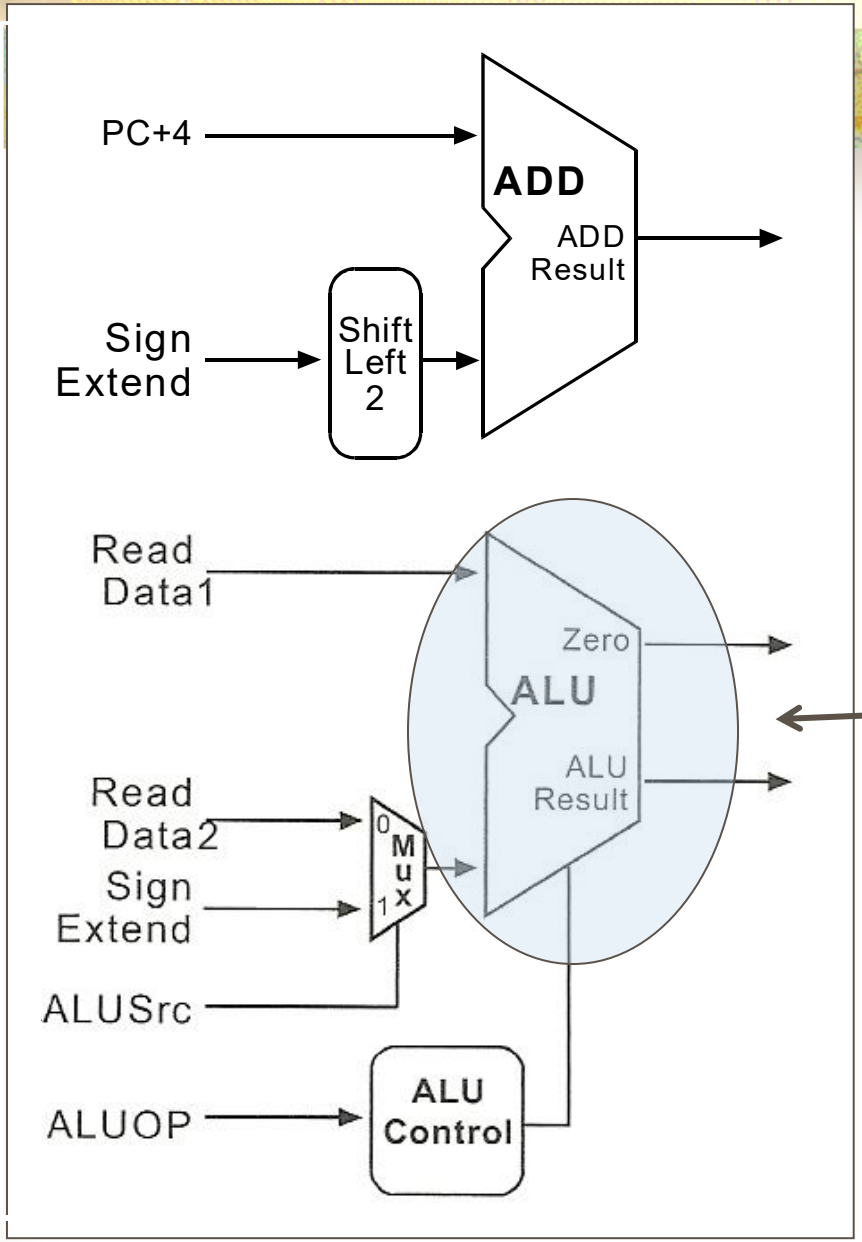


Figure 14.3 Block Diagram of MIPS Control Unit

Processador MIPS Simplificado

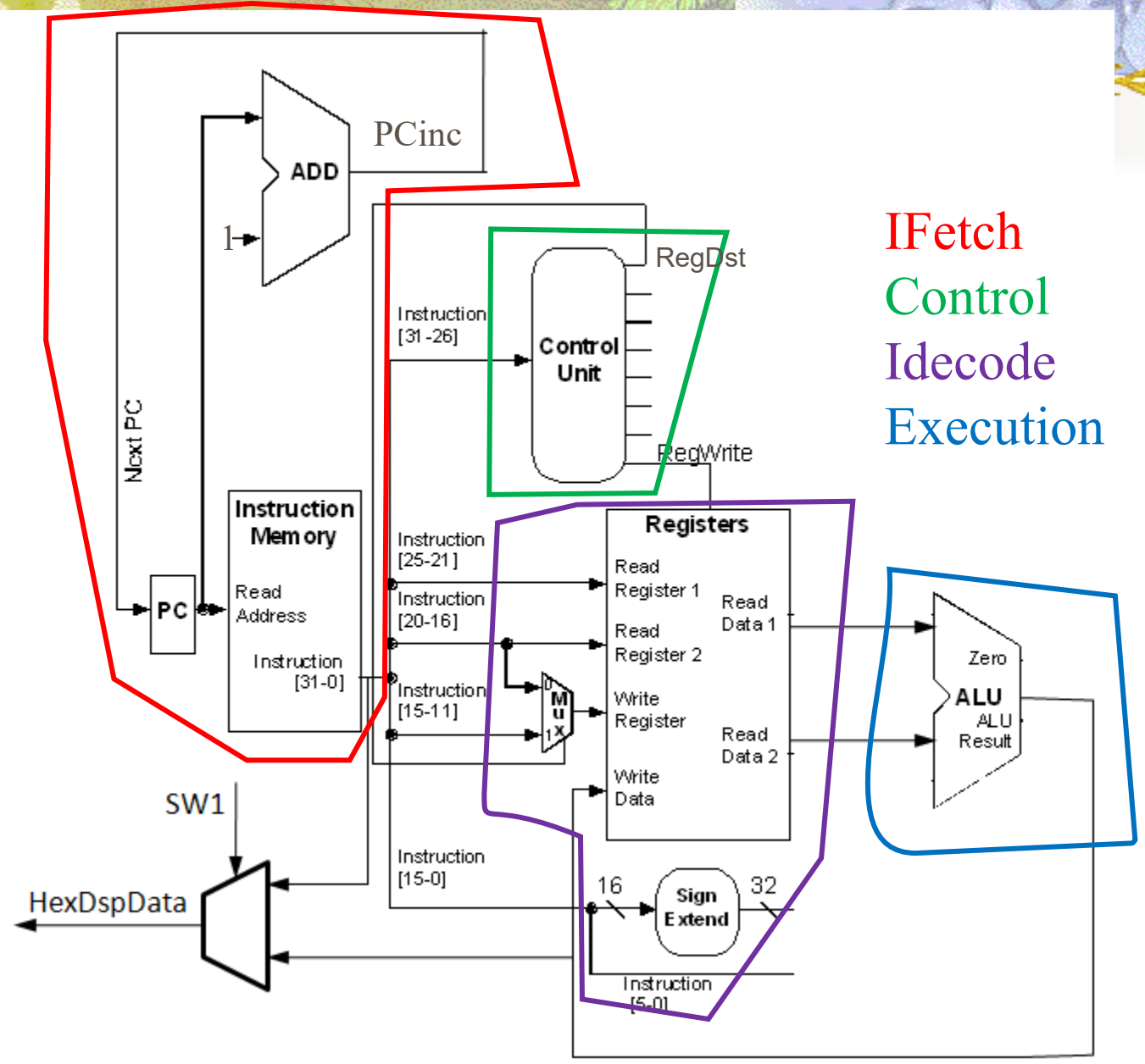
Executa

- Blocos Operacionais
  - Realiza as operações aritméticas e lógicas,
    - Com Rs e Rt
    - Com Rs e constante/endereço
  - Calcula endereço da próxima instrução para:
    - Salto relativo condicional (branch).
      - Com PC + 4 + endereço



Implementar somente a ULA

Figure 13.7 Block Diagram of MIPS Execute Unit.


IFetch
Control
Idecode
Execution
PCinc
ADD
1
Next PC
RegDst
Instruction [31-26]
Control Unit
RegWrite
Instruction Memory
Read Address
Instruction [31-0]
PC
Registers
Instruction [25-21]
Instruction [20-16]
Instruction [15-11]
Read Register 1
Read Register 2
Write Register
Write Data
Read Data 1
Read Data 2
M u x
0
1
Zero
ALU
ALU Result
SW1
HexDspData
Instruction [15-0]
16
Sign Extend
32
Instruction [5-0]