Monociclo Comandos

→ Tabela Instruções:

Name	Fields						Comments
Field size	6 bits	5 bits	5 bits	5 bits	5 bits	6 bits	All MIPS instructions are 32 bits long
R-format	op	rs	rt	rd	shamt	funct	Arithmetic instruction format
I-format	op	rs	rt	address/immediate			Transfer, branch, imm. format
J-format	op	target address					Jump instruction format

→ Tabela Saídas:

Nome	Efeito quando 0	Efeito quando 1
RegDst	Registrador destino = rt	Registrador destino = rd
RegWrite	Nada (faz com que MemToReg e RegDst sejam ignorados)	'Write register' recebe o valor disponível em 'Write data'
ALUSrc	Segundo operando da ALU vem da saída do banco de registradores	Segundo operando da ALU vem do valor estendido da instrução
Branch	PC = PC + 4	Se operandos iguais, PC recebe valor calculado do desvio. Se não, PC = PC + 4
MemRead	Nada	Dado da memória relativo ao endereço especificado é colocado na saída
MemWrite	Nada	Dado da memória relativo ao endereço especificado é substituído pelo que tem no "Write data"
MemToReg	Valor escrito na entrada "Write data" do banco de registradores vem da ALU	Valor escrito na entrada "Write data" do banco de registradores vem da memória

Controle:

→ Tipo R:

Função	RegDst	Jump	Branch	MemRead	MemToReg	ALUOp	MemWrite	ALUSrc	RegWr
add	1	0	0	0	0	1111	0	0	1
sub	1	0	0	0	0	1111	0	0	1
and	1	0	0	0	0	1111	0	0	1
or	1	0	0	0	0	1111	0	0	1
xor	1	0	0	0	0	1111	0	0	1
nor	1	0	0	0	0	1111	0	0	1
slt	1	0	0	0	0	1111	0	0	1
sltu	1	0	0	0	0	1111	0	0	1
sll	1	0	0	0	0	1111	0	0	1
srl	1	0	0	0	0	1111	0	0	1
sra	1	0	0	0	0	1111	0	0	1
sllv	1	0	0	0	0	1111	0	0	1
srlv	1	0	0	0	0	1111	0	0	1
srav	1	0	0	0	0	1111	0	0	1

OBS: Para a instrução do tipo jr deverá ser criado uma nova saída do Control: PCSrc (Personal Count Source) Que irá definir que a próxima instrução virá do reg R[&rs] (Instruction [25-21]).

Descrição:

- **add** - Add (Adicionar)
- **sub** - Subtract (Subtrair)
- **and** - And (E lógico)
- **or** - Or (OU lógico)
- **xor** - Exclusive OR (OU exclusivo)
- **nor** - NOT OR (Não OU)
- **slt** - Set on Less Than (Definir se menor que)
- **sltu** - Set on Less Than Unsigned (Definir se menor que sem sinal)
- **sll** - Shift Left Logical (Deslocamento lógico à esquerda)
- **srl** - Shift Right Logical (Deslocamento lógico à direita) Unsigned

- **sra** - Shift Right Arithmetic (Deslocamento aritmético à direita) Signed
- **sllv** - Shift Left Logical Variable (Deslocamento lógico à esquerda variável)
- **srlv** - Shift Right Logical Variable (Deslocamento lógico à direita variável) Unsigned
- **srav** - Shift Right Arithmetic Variable (Deslocamento aritmético à direita variável) Signed
- **jr** - Jump Register (Pular para o endereço armazenado no registrador)

→ Tipo I:

Função	RegDst	Jump	Branch	MemRead	MemToReg	ALUOp	MemWrite	ALUSrc	RegWr
addi	0	0	0	0	0	0001	0	1	1
andi	0	0	0	0	0	0010	0	1	1
ori	0	0	0	0	0	0011	0	1	1
xori	0	0	0	0	0	0100	0	1	1
beq	0	0	1	0	0	0101	0	0	0
bne	0	0	1	0	0	0110	0	0	0
slti	0	0	0	0	0	0111	0	1	1
sltiu	0	0	0	0	0	1000	0	1	1
lui	0	0	0	0	1	1001	0	1	1
lw	0	0	0	1	1	1010	0	1	1
sw	0	0	0	0	0	1011	1	1	0

- **beq** - Branch if Equal (Desviar se for igual)
- **bne** - Branch if Not Equal (Desviar se não for igual)
- **slti** - Set Less Than Immediate (Definir se menor que imediato)
- **sltiu** - Set Less Than Unsigned Immediate (Definir se menor que imediato sem sinal)
- **lui** - Load Upper Immediate (Carregar imediato superior)
- **lw** - Load Word (Carregar palavra)
- **sw** - Store Word (Armazenar palavra)

→ Tipo J:

Função	RegDst	Jump	Branch	MemRead	MemToReg	ALUOp	MemWrite	ALUSrc	RegWr
j	0	1	0	0	0	1100	0	0	0
jal	0	1	0	0	0	1101	0	0	1

- **j** - Jump (Pular)
- **jal** - Jump and Link (Pular e armazenar o endereço de retorno)