Trabalho de Projetos

ISA

Green Card

OPCODE	TIPO	INSTRUÇÃO	NOME	OPERAÇÃO
0000	R	BRZR	Branch On Zero Register	if (R[ra] == 0) PC = R[rb]
0001	I	JAL	Jump And Link	rd = PC + 4 PC += IMM
0010	R	JALR	Jump And Link Register	rd = PC + 4 PC = R[rb]
0011	R	LW	Load Word	R[ra] = M[R[rb]]
0100	R	SW	Store Word	M[R[rb]] = R[ra]
0101	I	ADDI	Add Imediato	R[ra] = R[ra] + IMM
0110	R	MUL	Multiplicação	R[ra] = R[ra] * R[rb]
0111	R	DIV	Divisão	R[ra] = R[ra] / R[rb]
1000	R	SLR	Shift Left Register	R[ra] = R[ra] << R[rb]
1001	R	SRR	Shift Right Register	R[ra] = R[ra] >> R[rb]
1010	R	NOT	Not	R[ra] = not R[rb]
1011	R	AND	And	R[ra] = R[ra] and R[rb]
1100	R	OR	Or	R[ra] = R[ra] or R[rb]
1101	R	XOR	Xor	R[ra] = R[ra] xor R[rb]
1110	R	ADD	Add	R[ra] = R[ra] + R[rb]
1111	R	SUB	Sub	R[ra] = R[ra] - R[rb]

Formatos de Instrução

TIPO R												
Bits	11	10	9	8	7	6	5	4	3	2	1	0
	OPCODE				Ra				Rb			

TIPO I												
Bits	11	10	9	8	7	6	5	4	3	2	1	0
	OPCODE				Ra				Imm			

Implementação

- Componentes utilizados:
    - PC: 1 registrador de 12 bits, 2 somadores de 12 bits, 2 MUXes
    - Banco de registradores: 15 registradores de 12 bits e 1 constante de 0.
    - Memória ROM: Uma memória ROM é utilizada para armazenar o programa, e outra para armazenar a codificação da Unidade de Controle
    - Memória RAM: Utilizada para guardar ou carregar valores passados pelas instruções SW e LW.
    - Multiplexadores: 1 MUX na ULA, 1 DEMUX e 2 MUXes no banco de registradores, 15 MUXes para cada registrador
    - Operadores Lógico Aritméticos: 1 MUX e 1 operador de 12 bits de cada uma das seguintes operações: AND, OR, XOR, NOT, somador, subtrator, shift left, shift right, multiplicador, divisor, comparador.
    - Componentes utilizados nas conexões do circuito main: 2 MUXes, 1 extensor de sinal para o imm

- Funcionamento do imediato: Número em complemento de dois, deve ser estendido respeitando que o bit mais significativo representa o sinal. Portanto o imediato é capaz de armazenar valores de 7 a -8.

- Informações gerais da implementação da ISA:
    - mux_pc: MUX que controla o valor que será guardado no registrador PC, utilizado pelas instruções de saltos e desvios. A entrada 00 é a padrão, 01 é utilizada pela instrução JAL, 10 é utilizada pela instrução BRZR e 11 é utilizada pela instrução JALR.

Entradas	Dado
00	PC + 1
01	PC + Imm
10	R[rb] ou PC + 1
11	R[rb]

    - mux_mem: Define qual informação será gravada no registrador destino

Entradas	Dado
00	Resultado ULA
01	M[rb]
10	PC + 1

    - mux_ula: Define se a ula receberá um valor imediato ou um valor armazenado em um registrador

Entradas	Dado
0	R[rb]
1	Imm

    - opula: Utiliza um MUX dentro da ula para devolver o resultado da operação correspondente

- Unidade de controle: Foi codificada de acordo com as devidas necessidades, ela controla se é para escrever na memória ou registrador, qual caminho que o sinal deve seguir por cada MUX e operação da ULA.
  Tabela utilizada:

Opcode	mux_pc	mux_ula	opula	wm	wreg	mux_mem
0000	10	0	0000	0	0	00
0001	01	1	0000	0	1	10
0010	11	0	0000	0	1	10
0011	00	0	0000	0	1	01
0100	00	0	0000	1	0	00
0101	00	1	0100	0	1	00
0110	00	0	1000	0	1	00
0111	00	0	1001	0	1	00
1000	00	0	0110	0	1	00
1001	00	0	0111	0	1	00
1010	00	0	0011	0	1	00
1011	00	0	0000	0	1	00
1100	00	0	0001	0	1	00
1101	00	0	0010	0	1	00
1110	00	0	0100	0	1	00
1111	00	0	0101	0	1	00

Convenção

Registrador	Nome	Descrição
x0	zero	Constante zero
x1	ra	Endereço de retorno
x2	sp	Ponteiro da stack
x5 - x10	t0 - t4	Registradores temporários
x11 - x13	a0 - a1	Argumentos de função
x14 - x15	s0 - s1	Registradores salvos

Decisões da ISA

Formatos de instruções: Os formatos foram escolhidos para maximizar a quantidade de registradores, 4 bits possibilita guardar endereços para 16 registradores diferentes.
O tipo I foi definido com o objetivo de ser flexível para ser usado em qualquer registrador com uma pequena desvantagem no fato dos imediatos serem um pouco pequenos. Além disso, foi escolhido o formato de 2 operandos para maximizar a quantidade de registradores como já mencionado e para que os bits sejam distribuídos igualmente entre os dois operandos.

Memória: A Memória ROM foi utilizada para escrever o programa pois ele não será alterado no decorrer da execução, também foi utilizada na unidade de controle pois também não é alterada, além de tornar mais flexível a modificação de seu comportamento. Memória RAM foi utilizada pois seus valores podem ser modificados e lidos a qualquer momento.

ULA: A unidade lógico aritmética foi feita com suporte para 10 operações, para facilitar operações de soma, subtração, multiplicação, divisão, shift right e shift left. Além de 1 operação, zero, com uma saída exclusiva, que serve para dizer se o valor de R[ra] é zero, ela possui uma saída exclusiva pois ela tem um função bem diferente das outras operações, porque ela é utilizada para escolher o caminho do PC.

Tradução assembly para binário

Para traduzir basta substituir o nome da instrução com o seu opcode correspondente, o nome dos registradores (x0, x1, … , x15) por números em binário, portanto, x0 é equivalente ao endereço 0000, x1 ao 0001 e assim por diante. Caso tenha um imediato basta traduzir o número de decimal para binário em complemento de 2.