Tamaño y tipo de datos:

* Enteros = 2 bytes
* char = 1 byte

Registros

r10🡪α 🡪 Escalar

r1🡪 Contador de fila 🡪 Escalar

r2🡪 Contador posición de memoria 🡪 Escalar

r3🡪 Espacio vacío🡪 Escalar

r4🡪Rin2🡪Vectorial

r5-r9🡪Vectoriales

r11🡪100 quemado🡪Escalar

Vectoriales:48 bits

Escalares:8 bits

RAM=

OPCODE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | D | Q |
| **0** | **0** | **0** | **0** | Mult |
| **0** | **0** | **0** | **1** | Div |
| **0** | **0** | **1** | **0** | Load |
| **0** | **0** | **1** | **1** | Store |
| **0** | **1** | **0** | **0** | Add |
| **0** | **1** | **0** | **1** | Sub |
| **0** | **1** | **1** | **0** | Addi |
| **0** | **1** | **1** | **1** | Subi |
| **1** | **0** | **0** | **0** | Branch |
| **1** | **0** | **0** | **1** | Branch Cond |
| **1** | **0** | **1** | **0** |  |
| **1** | **0** | **1** | **1** |  |
| **1** | **1** | **0** | **0** |  |
| **1** | **1** | **0** | **1** |  |
| **1** | **1** | **1** | **0** |  |
| **1** | **1** | **1** | **1** |  |

Azul: vector-escalares

Amarillo: vector-vector

Verde: escalar-escalar

Tamaño de instrucciones: 24bits

**Procesamiento de datos**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Opcode | Rd | Rs1 | Rs2 | Imm |
| 23 20 | 19 16 | 15 12 | 11 8 | 7 0 |

**Ex:**

ADD R1, R2, R3

ADD R1, R2, #2

**Para addi y subbi:**

El Rs1 no se puede usar, solo el Rs2.

**Control de flujo**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Opcode | Rb | Rs1 | Rs2 | Imm\*\*\*\* |
| 23 20 | 19 16 | 15 12 | 11 8 | 7 0 |

**Ex:**

**Para B**

R4 sería Rb, tiene que ser un registro escalar.

B R4 \*\* más recomendada

B R4, #5

B #5

**Para BCND**

**R1** debe ser escalar y **R2** vectorial o escalar(solo se toma el primer dato).

BCND R1, R2 #2(Branch on equal)

**Acceso a memoria**

|  |  |  |  |
| --- | --- | --- | --- |
| Opcode | Rd | Rs1 | Immediate\*\*\*\* |
| 23 20 | 19 16 | 15 12 | 11 0 |

LDR-STR R1, R2

LDR-STR R1, R2, #4

**Consideraciones de implementación**

1. Al ejecutarlo el pc va con un offset de 1. (empieza en cero, pero se adelanta a 1 antes de llegar a la segunda instrucción.)
2. Todos los registros son inicializados en cero.
3. Para que la señal que memWrite y regWrite se activen, la ALU debe dar un resultado diferente a cero.