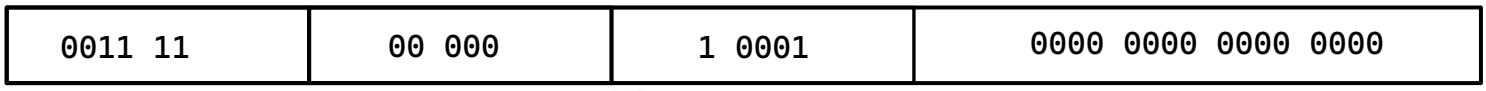
lui $s1, 0x1001 #direccion base del vector

Tipo i



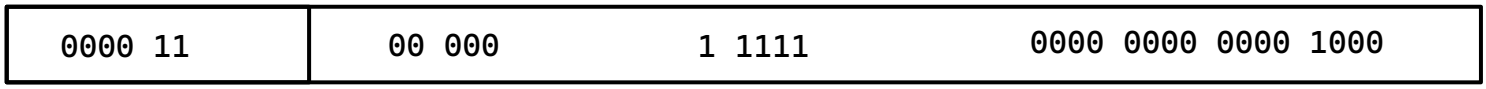
op RS RT

6 bits 5 bits 5 bits 16 bits

0x3C110000 Instruccion hexadecimal

jal longitud_array #CalcularLongitudArray()

Tipo j



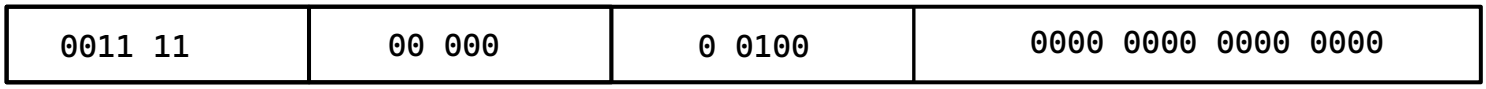
op RS RT Constant

6 bits 26 bits

0x0C1F0008 Instruccion hexadecimal

lui $a0, 0x1001 #direccion base del vector en a0

Tipo i



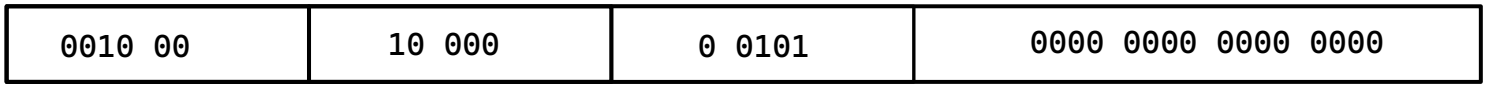
op RS RT Constant

6 bits 5 bits 5 bits 16 bits

0x3C040000 Instruccion hexadecimal

addi $a1, $s0, 0 #a1=longitud

Tipo i



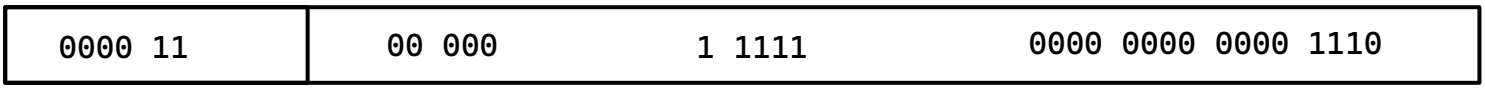
op RS RT Constant

6 bits 5 bits 5 bits 16 bits

0x22050000 Instruccion hexadecimal

jal penultimo #Obtener penultimo elemento

Tipo j



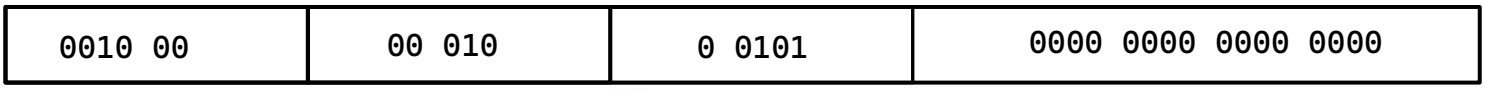
op Constant

6 bits 26 bits

0x0C1F000E Instruccion hexadecimal

addi $a1, $v0, 0 #arg1 = valor penult y=x

Tipo i



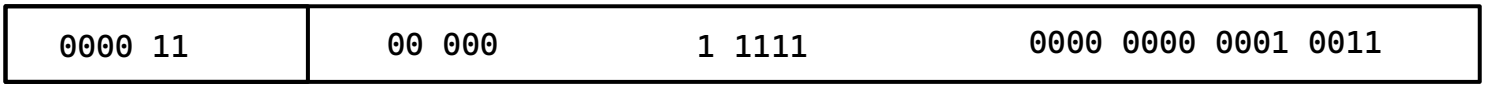
op RS RT Constant

6 bits 5 bits 5 bits 16 bits

0x20450000 Instruccion hexadecimal

jal mult_function # function_multiplicacion()

Tipo j



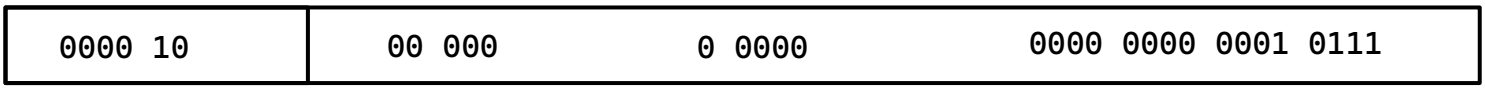
op RS RT Constant

6 bits 26 bits

0x0C1F0013 Instruccion hexadecimal

j exitfinal

Tipo j



op RS RT Constant

6 bits 26 bits

0x08000017 Instruccion hexadecimal

```
lw  $a0, 0($s1) #a0=s1[i]
```

Tipo i

1000 11	10 001	0 0100	0000 0000 0000 0000
op	RS	RT	Constant
6 bits	5 bits	5 bits	16 bits

0x8E240000 Instruccion hexadecimal

```
beq $a0, 0, exit #if(i=0)exit
```

Tipo i

0001 00	00 100	0 0000	0000 0000 0000 0011
op	RS	RT	Constant
6 bits	5 bits	5 bits	16 bits

0x10800003 Instruccion hexadecimal

```
addi $s0, $s0, 1 # longitud=longitud + 1
```

Tipo i

0010 00	10 000	1 0000	0000 0000 0000 0001
op	RS	RT	Constant
6 bits	5 bits	5 bits	16 bits

0x22100001 Instruccion hexadecimal

```
addi $s1, $s1, 1 #direccion = direccion + 1
```

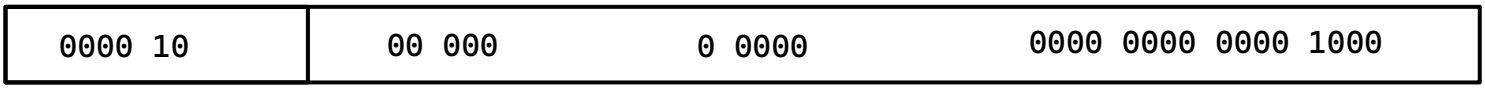
Tipo i

0010 00	10 001	1 0001	0000 0000 0000 0001
op	RS	RT	Constant
6 bits	5 bits	5 bits	16 bits

0x22310001 Instruccion hexadecimal

j longitud_array

Tipo j



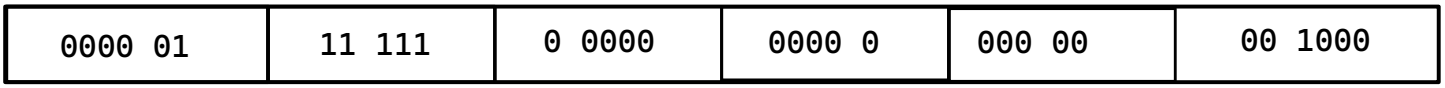
op RS RT Constant

6 bits 26 bits

0x08000008 Instruccion hexadecimal

jr $ra

Tipo r



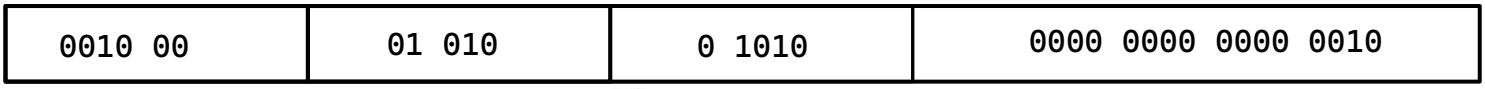
op RS RT RD SHAMT FUNCTION

6 bits 5 bits 5 bits 5 bits 5 bits 6 bits

0x07E00008 Instruccion hexadecimal

addi $t2, $t2, 2

Tipo i



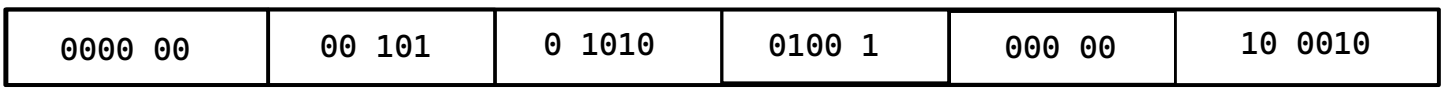
op RS RT Constant

6 bits 5 bits 5 bits 16 bits

0x214A0002 Instruccion hexadecimal

sub $t1, $a1, $t2 #Resta del registro traido

Tipo r



op RS RT RD SHAMT FUNCTION

6 bits 5 bits 5 bits 5 bits 5 bits 6 bits

0x00AA4822 Instruccion hexadecimal

add $s2, $s2, $t1 ## direccion del penultimo

Tipo r

0000 00	10 010	0 1001	1001 0	000 00	10 000
op	RS	RT	RD	SHAMT	FUNCTION
6 bits	5 bits	5 bits	5 bits	5 bits	6 bits

0x02499020 Instruccion hexadecimal

lw $v0,0($s2) #cargamos en v0 el penultimo.value

Tipo i

1000 11	10 010	0 0010	0000 0000 0000 0000
op	RS	RT	Constant
6 bits	5 bits	5 bits	16 bits

0x8E420000 Instruccion hexadecimal

jr $ra

Tipo r

0000 01	11 111	0 0000	0000 0	000 00	00 1000
op	RS	RT	RD	SHAMT	FUNCTION
6 bits	5 bits	5 bits	5 bits	5 bits	6 bits

0x07E00008 Instruccion hexadecimal

sll $t3, $a1, 6 #x * 64

Tipo r

0000 00	00 000	0 0101	0101 1	001 10	00 0000
op	RS	RT	RD	SHAMT	FUNCTION
6 bits	5 bits	5 bits	5 bits	5 bits	6 bits

0x00055980 Instruccion hexadecimal

sll $t4, $a1, 4 #x * 16

Tipo r

0000 00	00 000	0 0101	0110 0	001 00	00 0000
op	RS	RT	RD	SHAMT	FUNCTION
6 bits	5 bits	5 bits	5 bits	5 bits	6 bits

0x00056100 Instruccion hexadecimal

add $v1, $t3, $t4 #(x*64) + (x*16) = x*80

Tipo r

0000 00	01 011	0 1100	0001 1	000 00	10 000
op	RS	RT	RD	SHAMT	FUNCTION
6 bits	5 bits	5 bits	5 bits	5 bits	6 bits

0x016C1820 Instruccion hexadecimal

jr $ra

Tipo r

0000 01	11 111	0 0000	0000 0	000 00	00 1000
op	RS	RT	RD	SHAMT	FUNCTION
6 bits	5 bits	5 bits	5 bits	5 bits	6 bits

0x07E00008 Instruccion hexadecimal

addi $s5, $s5, 20 #Puntero de la dirección base del resultado

Tipo i

0010 00	10 101	1 0101	0000 0000 0001 0100
op	RS	RT	Constant
6 bits	5 bits	5 bits	16 bits

0x22B50014 Instruccion hexadecimal

sw $v1 , 0($s5) #Guardar el valor del resultado

Tipo i

1010 11	10 101	0 0011	0000 0000 0000 0000
op	RS	RT	Constant
6 bits	5 bits	5 bits	16 bits

0xAEA30000 Instruccion hexadecimal