Tabla

Descripción generada automáticamente**Organización de Computadores PEP #1, 01/2023 Diurno**

Recordar:

Tcpu \* tasa = #ciclos

Tcpu = (#instrucciones \* CPI)/tasa

Luego: #ciclos = #instrucciones\*CPI

#ciclosARM = (7\* 10^5) \* 2,8 = 19,6 \* 10^5

#ciclosMIPS = (11\* 10^5) \* 1,5 = 16,5 \* 10^5

(b) Tiempo de uso de CPU

TcpuARM = #ciclos / tasa = (19,6 \* 10^5)/(3\* 10^9) = 6,53333\* 10^-4 [s]

TcpuMIPS= #ciclos / tasa = (16,5 \* 10^5)/(2,1\* 10^9) = 7,857142 \* 10^-4 [s]

(c) ¿Qué sistema tiene mejor rendimiento? ¿Por cuántas veces más que el otro?

Calculamos los tiempos totales

TejecARM = 7 \* 10^-4 + 1+ 1,5 [s] = 2,500653333 [s]

TejecMIPS= 11 \* 10^-4 + 0,03+ 0,01 [s] = 0,0407857 [s]

El que tiene major rendimiento es el MIPS, porque tiene un tiempo mucho menor

Luego:

TejecARM/ TejecMIPS = 2,5006/0,0407 = 61,43

MIPS es 61, 4 veces mejor que ARM, tiene mejor rendimiento

Texto

Descripción generada automáticamente

Por ejemplo, las instrucciones de lectura y escritura de words en la memoria, funcionan de tal forma que ya sea que escriban lo que está en un registro, o lean y guarde lo de la memoria en un registro. Acede a la memoria al realizar la sumar la dirección base con una constante, utilizada para acceder a algún elemento de la memoria al tratarla como un arreglo, considerando esta constante como un multiplo de 4, para acceder a la palabra y no por byte. Tiene la siguiente estructura la lectura

lw $rt, im($rs), donde el registro a guardar la Word es $rt, y el registro con la dirección base en $rs

Texto

Descripción generada automáticamente

Ya que el tamaño de palabra es de 64 bits, cada instrucción es de ese tamaño.

Y las direcciones de memoria son de 48 bits, podría significar que como no puedo utilizar un JTA, puedo guardar la dirección a la instrucción en estos bits.

Además, para los 48 registros, se tiene que 48 es 11 0000 (necesito 6 bits para numerar 48 registros). También, para las 200 instrucciones, notamos que 200 es 1100 1000 (necesito 8 bits)

Op r1 X otro label

8 6 14 36



# lo cargo por partes porque es muy grande

lui $t0, upper(BIG) #guardo la primera parte

ori $t0, $t0, lower(BIG) #guardo la Segunda parte

add $t0, $t0, $t2 # sumo el valor de big mas la direccion

lw $t5, 0($t0) #guardo la palabra

Texto

Descripción generada automáticamente

#supone $s0 = u, $s1 = v, $s2 = i, $s3 = s

add $s2, $0, $0 #i=0

add $s3, $0, $0 # s=0

loop: slti $t0, $s2, 100 # guardamos signo de i-100

beq $t0, $0, salida #caso de salida cuando i-100>=0

sll $t0, $s2, 2 # en t0 guardamos la copia de i mult por 4

add $t1, $t0, $s0 # en t1guardamos dirección de u[i]

add $t2, $t0, $s1 # en t2 guardamos dirección de v[i]

lw $t1, 0($t1) # en t1 guardamos el valor leído de u[i]

lw $t2, 0($t2) # en t2 guardamos el valor leído de v[i]

add $t0, $t1, $t2 # calculamos en t0 = u[i] + v[i]

add $s3, $s3, $t0 # s = s + u[i] + v[i]

addi $s2, $s2, 1 # i++

j loop # volvemos a iterar

salida:

Tabla

Descripción generada automáticamente

* Convierta las instrucciones 2, 5 y 10 a lenguaje máquina.

2. beq $t0, $0, Z

Entre el PC+4 y Z hay 8 instrucciones

Luego:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| opcode | | | | | | rs | | | | | rt | | | | | imm | | | | | | | | | | | | | | | |
|  | | | | | | 8 | | | | | 0 | | | | | 8 | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | | | | 1 | | | | 0 | | | | 0 | | | | 0 | | | | 0 | | | | 0 | | | | 8 | | | |
| 0x11000008 En la pauta el profe lo tiene malo, puso 0x11080008 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

5. lw $ra, 0($sp)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| opcode | | | | | | rs | | | | | rt | | | | | imm | | | | | | | | | | | | | | | |
|  | | | | | | 29 | | | | | 31 | | | | | 0 | | | | | | | | | | | | | | | |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 8 | | | | F | | | | B | | | | F | | | | 0 | | | | 0 | | | | 0 | | | | 0 | | | |
| 0x8FBF0000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

10. j X

Notamos que la direccion de label X es 0xAFBF0000. El JTA

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | | | | F | | | | B | | | | F | | | | 0 | | | | 0 | | | | 0 | | | | 0 | | | |
| 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  |  |  |  | 2 | | E | | | | F | | | | C | | | | 0 | | | | 0 | | | | 0 | | | |  |  |

Luego:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| opcode | | | | | | adress | | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | 0x2EFC000 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | | | | B | | | | E | | | | F | | | | C | | | | 0 | | | | 0 | | | | 0 | | | |
| 0x0BEFC000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

* Convierta las instrucciones 1 y 6 en lenguaje ensamblador

1. 0x03E00008

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| opcode | | | | | | rs | | | | | rt | | | | | rd | | | | | shamp | | | | | funct | | | | | |
| 0 | | | | | | 31 | | | | | 0 | | | | | 0 | | | | | 0 | | | | | 8 | | | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | | | | 3 | | | | E | | | | 0 | | | | 0 | | | | 0 | | | | 0 | | | | 8 | | | |
| jr $ra | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

6. 0xAD49FFF9

Antes, convertamos el numero de C2 a decimal

FFF9

1111 1111 1111 1001 - 1 = 1111 1111 1111 1000 luego: 0000 0000 0000 0111

Es el -7

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| opcode | | | | | | rs | | | | | rt | | | | | imm | | | | | | | | | | | | | | | |
| 43 | | | | | | 10 | | | | | 9 | | | | | -7 | | | | | | | | | | | | | | | |
| 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| A | | | | D | | | | 4 | | | | 9 | | | | F | | | | F | | | | F | | | | 9 | | | |
| sw $t1, -7($t2) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |