Tabla

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

1. Determine el CPI

Primero hay que determinar la cantidad de instrucciones por tipo

u (aritmética) = 4

v (acceso memoria) = 3

w (direccionamientos) = 2 algun beq y un jump al inicio del for

Notamos que por el ciclo, las instrucciones se realizan 10^6 veces

CPIa = (3\*4 + 7\*3+ 2\*2 ) (10^6)/ (9 \*10^6) = 4,11111

CPIb = (3\*4 + 5\*3+ 1\*2 ) (10^6)/ (9 \*10^6) = 3,22222

1. Determine el tiempo de uso de CPU

TcpuA = (#intrucc \* CPIa) / tasa = (9 \*10^6 \* 4,1111) / (2,8 \*10^9) = 13,21 \*10^-3 [s]

TcpuB = (#intrucc \* CPIa) / tasa = (9 \*10^6 \* 3,2222) / (2,1 \*10^9) = 13,80 \*10^-3 [s]

1. ¿Qué sistema tiene mejor rendimiento? ¿Por cuántas veces más que el otro?

El de mejor rendimiento es el que tiene un menor tiempo. Es decir el compu A

Luego

RendA = 1/(13,70 \*10^-3) = 72,9927

RendB = 1/(13,80 \*10^-3) = 72,4142

Además:

RendA/RendB = 72,9927/72,4142 = 1,008

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamenteEsto igual se podía calcular solo con los tiempos de ejecución.

Suma $rs, $rt se guarda en una dirección de memoria de 32 bits

Nos dice que trabaja con 64 registros, luego el 63 en binario es 111111, por lo que necesita 6 bits para guardar los dos registros fuente que se suman. Además como tenemos 140 instrucciones, considerando un código para cada uno, el 139 en binario es 10001011, por lo que necesita 8 bits.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Opcode | R1 | R2 | X otro | Direccion memoria |
| 8 | 6 | 6 | 12 | 32 |

¿Con qué instrucciones MIPS 32 bits implementaría la función sumar?

Supone $s0 = r1, $s1 = r2, y que en dirección este la dirección de memoria de 32 bits

Add $t0, $s0, $s1

Lui $t1, upper(dirección)

Ori, $t1, $t1, lower(dirección)

Sw $t0, 0($t1)

Tabla

Descripción generada automáticamenteLas pseudo instrucciones son move, li, btg, b

Li $a0, 5: load immediate? Supongo

Addi $a0, $0, 5

#El profe hizo esto, aunque lo mio es mejor porque son menos instrucciones

Lui $a0, 0x0000

Ori $a0, $a0, 0x0005

Move $a0, $v0: es como asignar a a0 el valor de v0 creo

Add $a0, $v0, $0

Btg $a0, $t0, et1: Branch si son si el a0 es menor que t0. NI IDEA, LA LEI

Slt $t1, $t0, $a0

Beq $t1, $0, et1 #si la resta el 0, voy a et1, ergo $t0 > $a0

Indique que se muestra por pantalla (primera llamada al sistema del code):

Muestra el 15

Texto, Carta

Descripción generada automáticamente

#supone $s0=u, $s1=v, $s2=W

add $s0, $0, $0 #inicializo u = 0

addi $s1, $0, 12 #inicializo v = 12

while: slti $t0, $s0, 100 # guardamos signo de u-100

beq $t0, $0, salida #si u-100 es no negativo, salgo del ciclo

sll $t0, $s0, 2 # reuso $t0 y guardo una copia del index u\*4

add $t0, $t0, $s2 # le sumo la dirección base del arreglo

sub $t1, $s1, $s0 # calculo v-u en $t1

sll $t1, $t1, 2 # mult por 4 el valor anterior

sw $t1, 0($t0) # escribimos en la memoria ($t0) el valor calculado # en $t1

addi $s0, $s0, 1 # u++

j while # volvemos a iterar

salida:

Tabla

Descripción generada automáticamente

1. sw $t1, -7($t2) Es tipo I

Antes del desarrollo, convertimos el -7 a binario con C2

7 ----- 0111 C1: 1000 C2: 1001

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 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 | | | |
| 0xAD49FFF9 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

2. bne $t0, $t3, inicio Es tipo I

Notamos que entre el PC+4 y inicio, hay -10 saltos de instrucciones CUENTA BIEN WN NSW

Luego: 10: 01010 C1 10101 C2: 10110

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| opcode | | | | | | rs | | | | | rt | | | | | imm | | | | | | | | | | | | | | | |
| 5 | | | | | | 11 | | | | | 8 | | | | | -9 | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | | | | 5 | | | | 6 | | | | 8 | | | | F | | | | F | | | | F | | | | 5 | | | |
| 0x1568FFF5 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

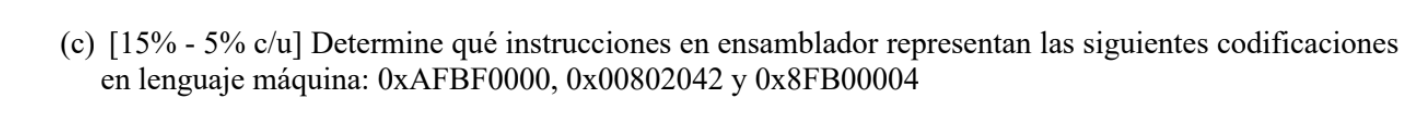
3. j inicio

Notamos que la dirección de memoria de inicio es 0xFC00000C. Ahora cal, el JTA

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| F | | | | C | | | | 0 | | | | 0 | | | | 0 | | | | 0 | | | | 0 | | | | C | | | |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
|  |  |  |  | 3 | | 0 | | | | 0 | | | | 0 | | | | 0 | | | | 0 | | | | 3 | | | |  |  |

Luego, la instrucción

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| opcode | | | | | | adress | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | | | | | |  | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | | | | B | | | | 0 | | | | 0 | | | | 0 | | | | 0 | | | | 0 | | | | 3 | | | |
| 0x0B000003 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |



1. 0xAFBF0000. Notamos que no es tipo R

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| opcode | | | | | | rs | | | | | rt | | | | | imm | | | | | | | | | | | | | | | |
| 43 | | | | | | 29 | | | | | 31 | | | | | 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 |
| A | | | | F | | | | B | | | | F | | | | 0 | | | | 0 | | | | 0 | | | | 0 | | | |
| sw $ra, 0($sp) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

2. 0x00802042. Notamos Tipo R

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| opcode | | | | | | rs | | | | | rt | | | | | rd | | | | | shamp | | | | | funct | | | | | |
| 0 | | | | | | 4 | | | | | 0 | | | | | 4 | | | | | 1 | | | | | 2 | | | | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | | | | 0 | | | | 8 | | | | 0 | | | | 2 | | | | 0 | | | | 4 | | | | 2 | | | |
| srl $a0, $a0, 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Creo que está malo porque los shift de imm no tienen rs, el valor de rs debería ser rt,

Y recuerda que los valores van al revés.

Ej: sll rd, rt, rs para las que usan 3 reg, y sll rd, rt, imm

3. 0x8FB00004 Tiene pinta de j. Es load Word de tipo I jajaja

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| opcode | | | | | | rs | | | | | rt | | | | | imm | | | | | | | | | | | | | | | |
| 35 o 0x23 | | | | | | 29 | | | | | 16 | | | | | 4 | | | | | | | | | | | | | | | |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 8 | | | | F | | | | B | | | | 0 | | | | 0 | | | | 0 | | | | 0 | | | | 4 | | | |
| lw $s0, 4($sp) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |