Subtítulo

**Actividad grupal: Ejercicios sobre Prestaciones**

Autores:

Emiliano González Martín

Darío Vinueza

Pedro Jesús García Ramos

Antonio Luis Ábalos Navarro

Profesora: Paula Lamo Anuarbe

Asignatura: Estructura de Computadores (Abril 2021)



| Histórico de versiones: | | | |
| --- | --- | --- | --- |
| **Versión** | **Fecha** | **Autor** | **Comentarios** |
| 1.0 | 09/04/2021 | Grupo | Apertura documento |
| 1.1 | 12/04/2021 | Grupo | Actualización |
| 1.2 | 13/04/2021 | Grupo | Actualización y cierre de documento |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Índice del Contenido

[1 Introducción y Escenario General 4](#_Toc69237743)

[2 Esquema del Documento 4](#_Toc69237744)

[3 Ejercicio 1 5](#_Toc69237745)

[3.a Obtener los MIPS necesarios para ejecutar ambas opciones en la máquina indicada. 5](#_Toc69237746)

[3.b Teniendo en cuenta el tiempo de CPU, compara y comenta los resultados obtenidos en el apartado anterior 7](#_Toc69237747)

[4 Ejercicio 2 9](#_Toc69237748)

[4.a Aceleracion si sólo se aplica la opción 1 9](#_Toc69237749)

[4.b Aceleracion si sólo se aplica la opción 2 10](#_Toc69237750)

[4.c Evalúa de nuevo la aceleración si se aplican las dos opciones planteadas. 11](#_Toc69237751)

[5 Ejercicio 3 12](#_Toc69237752)

# Introducción y Escenario General

El presente documento contiene la solución dada por los autores en relación a la actividad número uno, **Ejercicios sobre prestaciones**, de la asignatura **Estructura de Computadores** dentro del curso académico **2020/2021**.

Dicha actividad está compuesta por tres ejercicios en los que se solicita la realización de diferentes cálculos, así como la interpretación de los resultados obtenidos en la cada uno de ellos.

# Esquema del Documento

El documento se divide en base a los tres ejercicios que conforman la actividad. En cada uno de ellos se detallan los cálculos realizados, los resultados obtenidos y su interpretación.

# Ejercicio 1

Se pretende ejecutar una aplicación en un computador cuyo procesador emplea únicamente dos tipos de instrucción:

* Instrucción 1 con CPI = 3
* Instrucción 2 con CPI = 2

Se sabe además que el procesador opera a 200 MHz.

Para generar el ejecutable del programa se pueden emplear dos compiladores diferentes, A y B, que ofrecen dos interpretaciones distintas del código, como indica la tabla adjunta.

|  |  |  |
| --- | --- | --- |
| **Compilador** | **Instrucción 1** | **Instrucción 2** |
| A | 6 | 2 |
| B | 4 | 6 |

Se pide:

* 3.a) Obtener los MIPS necesarios para ejecutar ambas opciones en la máquina indicada.
* 3.b) Teniendo en cuenta el tiempo de CPU, compara y comenta los resultados obtenidos en el apartado anterior.

## 3.a Obtener los MIPS necesarios para ejecutar ambas opciones en la máquina indicada.

**Compilador A**

Comenzamos calculando los CPI del compilador A. Para ello, aplicamos la siguiente fórmula:

Donde:

CPI: son el número de ciclos por instrucción que queremos calcular.

NIi: son el número de instrucciones de ese tipo en el programa.

NItotal: son el número de instrucciones total por programa.

Aplicamos fórmula:

Resultado: CPI=2,75 ciclos por instrucción.

Procedemos a calcular las MIPS mediante la siguiente fórmula:

Donde:

MIPS: son los millones de instrucciones por segundo que queremos calcular.

Fcpu: es la frecuencia, en Herzios, a la que trabaja el procesador.

CPI: son el número de ciclos por instrucción que hemos calculado previamente.

Aplicamos fórmula:

**Resultado: MIPS=72,72 millones de instrucciones por segundo**

**Compilador B**

Comenzamos calculando los CPI del compilador B. Para ello, aplicamos la siguiente fórmula:

Donde:

CPI: son el número de ciclos por instrucción que queremos calcular.

NIi: son el número de instrucciones de ese tipo en el programa.

NItotal: son el número de instrucciones total por programa.

Aplicamos fórmula:

Resultado: CPI=2,4 ciclos por instrucción.

Procedemos a calcular las MIPS mediante la siguiente fórmula:

Donde:

MIPS: son los millones de instrucciones por segundo que queremos calcular

Fcpu: es la frecuencia, en Herzios, a la que trabaja el procesador.

CPI: son el número de ciclos por instrucción que hemos calculado previamente

Aplicamos fórmula:

**Resultado: MIPS=83,33 millones de instrucciones por segundo**

## 3.b Teniendo en cuenta el tiempo de CPU, compara y comenta los resultados obtenidos en el apartado anterior

**Compilador A**

Comenzamos calculando el tiempo de CPU para el compilador A. Para ello, aplicamos la siguiente fórmula:

Donde:

Tcpu: es el tiempo de CPU que queremos calcular.

NI: es el número total de instrucciones de cada programa en este compilador. En este caso NI=8.

CPI: son el número de ciclos por instrucción. En este caso, CPI=2,75.

Fcpu: es la frecuencia, en Herzios, de la CPU. En este caso 200\*10^6 Herzios.

Aplicamos fórmula:

**Resultado: Tcpu=0,00000011 segundos para completar las 8 instrucciones del programa.**

**Compilador B**

Comenzamos calculando el tiempo de CPU para el compilador B. Para ello, aplicamos la siguiente fórmula:

Donde:

Tcpu: es el tiempo de CPU que queremos calcular.

NI: es el número total de instrucciones de cada programa en este compilador. En este caso NI=10.

CPI: son el número de ciclos por instrucción. En este caso, CPI=2,4.

Fcpu: es la frecuencia, en Herzios, de la CPU. En este caso 200\*10^6 Herzios.

Aplicamos fórmula:

**Resultado: Tcpu= 0,00000012 segundos para completar las 10 instrucciones del programa.**

**Comparativa de resultados y conclusiones**

|  |  |  |  |
| --- | --- | --- | --- |
| **Compilador** | **MIPS** | **NI** | **Tcpu** |
| A | 72,72 | 8 | 0,11\*10-6 |
| B | 83,33 | 10 | 0,12\*10-6 |

Comparativamente hablando, el compilador A utiliza un tiempo de CPU menor que el compilador B, así como un menor número de instrucciones. **Es decir, se puede afirmar que, en el compilador A, el programa en cuestión tendrá un mayor rendimiento que en el compilador B.**

# Ejercicio 2

Durante el desarrollo de una CPU se evalúa su rendimiento mediante un *benchmark* y, entre otros resultados, se obtiene la siguiente distribución del tiempo de ejecución:

|  |  |
| --- | --- |
| **Instrucción** | % |
| **Multiplicación** | 20 |
| **Acceso a memoria** | 50 |
| **Resto** | 30 |

En función de estos resultados se estudia introducir dos opciones de mejora en la configuración:

* Opción 1: Acelerar en 4 veces la ejecución de multiplicaciones.
* Opción 2: Mejorar el acceso a operandos en memoria, reduciendo a la mitad el tiempo necesario para ello.

Se pide:

* ¿Cuál es la aceleración si solo se aplica la opción 1?
* ¿Y si solo se aplica la opción 2?

Evalúa de nuevo la aceleración si se aplican las dos opciones planteadas.

## 4.a Aceleracion si sólo se aplica la opción 1

Opción 1: acelerar en 4 veces la ejecución de multiplicaciones.

Utilizamos la siguiente fórmula para calcular el % de mejora.

Donde:

S: es el % de mejora que queremos calcular.

f: es el % (en tanto \* 1) que NO aprovecha la mejora. En este caso, las partes que no aprovechan la mejora son el ‘Acceso a memoria’ (50%) y el ‘Resto’ (30%). Es decir, 50%+30%=80%, **0,8** (en tanto \* 1).

1-f: es el % (en tanto \* 1) que SÍ aprovecha la mejora. En este caso, la parte que sí aprovecha la mejora es la ‘Multiplicación’, que corresponde a un 20%. Es decir, en tanto \* 1, **0,2**.

m:la mejora que queremos introducir. En este caso, queremos acelerar en 4 veces la multiplicación. Por tanto, m=**4**.

Aplicamos fórmula:

**Resultado: en %, la mejora sería de un 18% sobre la instrucción de la multiplicación.**

## 4.b Aceleracion si sólo se aplica la opción 2

Opción 2: Mejorar el acceso a operandos en memoria, reduciendo a la mitad el tiempo necesario para ello.

Utilizamos la siguiente fórmula para calcular el % de mejora.

Donde:

S: es el % de mejora que queremos calcular.

f: es el % (en tanto \* 1) que NO aprovecha la mejora. En este caso, las partes que no aprovechan la mejora son la ‘Multiplicación’ (20%) y el ‘Resto’ (30%). Es decir, 20%+30%=50%, **0,5** (en tanto \* 1).

1-f: es el % (en tanto \* 1) que SÍ aprovecha la mejora. En este caso, la parte que sí aprovecha la mejora es el ‘Acceso a memoria’, que corresponde a un 50%. Es decir, en tanto \* 1, **0,5**.

m:la mejora que queremos introducir. En este caso, queremos acelerar en 2 veces el acceso a memoria. Por tanto, m=**2**.

Aplicamos fórmula:

**Resultado: en %, la mejora sería de un 33% sobre la instrucción de acceso a memoria.**

## 4.c Evalúa de nuevo la aceleración si se aplican las dos opciones planteadas.

Utilizamos la siguiente fórmula:

Aplicamos los valores que obtuvimos en los apartados 4.a y 4.b a la fórmula:

**Resultado: si se aplican las dos mejoras, se obtiene una aceleración del 67% para el cómputo global.**

# Ejercicio 3

Suponga que en un programa la frecuencia de fallos de la cache de instrucciones es el 2%, y la frecuencia de fallos de la cache de datos es el 4%. Si un procesador tiene un CPI igual a 2 sin incluir paradas debidas a la memoria, y la penalización por fallo es de 100 ciclos para todo tipo de fallo, determinar cuánto más rápido es un procesador cuya cache es perfecta, es decir, que nunca falla. Suponga que la frecuencia de las instrucciones de carga y almacenamiento es el 36%

Datos

* Frecuencia de fallos de la caché de instrucciones: 2% 🡪 en tanto \*1 es **0,02**.
* Frecuencia de fallos de la caché de datos: 4% 🡪 en tanto\*1 es **0,04**.
* CPI (Ciclos por instrucción) del procesador: **2 ciclos por instrucción** (si paradas por acceso a memoria).
* Penalización por fallo: **100 ciclos.**
* Frecuencia de instrucciones de carga y almacenamiento: 36% 🡪 en tanto\*1 es **0,36**.

Fórmula de los ciclos de reloj de parada debido a las instrucciones.

Fórmula de los ciclos de reloj de parada debido a la memoria.

Sumamos Crpi + Crpm para obtener la parada total por ciclo: en este caso 3,44 paradas por ciclo.

El procesador tiene un CPI=2 (sin incluir paradas debidas a la memoria). A esto hay que sumarle el valor calculado previamente (que incluye las paradas por fallo de instrucción y memoria).

La relación de CPI con parada y CPI perfecta sería así. En este caso, **un procesador con caché perfecta sería un 2,72 veces más rápido.**