**2.10** A benchmark program is run on a 40 MHz processor. The executed program consists  
of 100,000 instruction executions, with the following instruction mix and clock cycle  
count:

|  |  |  |
| --- | --- | --- |
| **Instruction Type** | **Instruction Count** | **Cycles per Instruction** |
| Integer arithmetic | 45,000 | 1 |
| Data transfer | 32,000 | 2 |
| Floating point | 15,000 | 2 |
| Control transfer | 8000 | 2 |

Determine the effective CPI, MIPS rate, and execution time for this program.

- CPI:

CPI = ( 45 000 + ( 32 000 \* 2 ) + ( 15 000 \* 2 ) + ( 15 000 \* 2 ) + ( 8 000 \* 2 ) ) / 100 000 = 1.55

MIPS = [ 100 000 / ( (100 000 \* 1.55) \* 40 \* 10^6 ) ] / 10 ^ 6 = 25.8 MIPS/s

Execution time ( thời gian thực thi chương trình or thời gian xử lý các câu lệnh của chương trình ) =

+ Integer arithimetic: (100 000 \* 1.55) \* 40 \* 10^6 = 0.003875 s

CPI = 1.55; MIPS rate = 25.8; Execution time = 0.003875 s  
**2.11** Consider two different machines, with two different instruction sets, both of which  
have a clock rate of 200 MHz. The following measurements are recorded on the two  
machines running a given set of benchmark programs:

|  |  |  |
| --- | --- | --- |
| **Instruction Type** | **Instruction Count (millions)** | **Cycles Per Instruction** |
| Machine A Arithmetic and logic Load and store Branch Others | 8 4 2 4 | 1  3  4  3 |
| Machine A Arithmetic and logic Load and store Branch Others | 10 8 2 4 | 1  2  4  3 |

1. Determine the effective CPI, MIPS rate, and execution time for each machine.

- CPI A = (( 8 + 4 \* 3 + 2 \* 4 + 4 \* 3 ) \* 106 ) / ( 8 + 4 + 2 + 4 ) \* 106 = 2.22

- MIPS A= (( 8 + 4 + 2 + 4 ) \* 106 ) / [ ( 8 + 4 \* 3 + 2 \* 4 + 4 \* 3 ) \* 106  / ( ( 200 \* 106 ) \* 106 ) ] = 90

- execution time A = ( 8 + 4 \* 3 + 2 \* 4 + 4 \* 3 ) \* 106  / ( 200 \* 106 ) = 0.2 s

-CPI B = (( 10 + 8 \* 2 + 2 \* 4 + 4 \* 3 ) \* 106 ) / ( 10 + 8 + 2 + 4 ) \* 106 = 1.92

- MIPS B = (( 10 + 8 + 2 + 4 ) \* 106 ) / [ ( 10 + 8 \* 2 + 2 \* 4 + 4 \* 3 ) \* 106 / ( ( 200 \* 106 ) \* 106 ) ] = 104

- execution time = ( 10 + 8 \* 2 + 2 \* 4 + 4 \* 3 ) \* 106 / ( 200 \* 106 ) = 0.23 s   
**b.** Comment on the results.

- máy B chạy nhanh hơn máy A bởi vì có tốc độ xử lý nhanh hơn MIPS B cao hơn MIPS A. Số câu lệnh của   
**2.12** Early examples of CISC and RISC design are the VAX 11/780 and the IBM RS/6000,  
respectively. Using a typical benchmark program, the following machine characteristics  
result:

|  |  |  |  |
| --- | --- | --- | --- |
| **Processor** | **Clock Frequency (MHz)** | **Performance (MIPS)** | **CPU Time (seconds)** |
| VAX 11/780 ( A ) | 5 | 1 | 12 *x* |
| IBM RS/6000 ( B ) | 25 | 18 | *x* |

The final column shows that the VAX required 12 times longer than the IBM measured  
in CPU time.  
**a.** What is the relative size of the instruction count of the machine code for this  
benchmark program running on the two machines?

- B / A = Ic B / Ic A = MIPS B / MIPS A = ( 18 \* CPU time \* 106 ) / ( 1 \* CPU time \* 106 ) = ( 18 \* x ) / 12x = 1.5  
**b.** What are the *CPI* values for the two machines

- MIPS A = Ic A / ( ( ( Ic A \* CPI A ) / ( 5 \* 106 ) \* 106 ) = 1

= Ic A / [ ( Ic A \* CPI A ) / 5 ] = 1

=> CPI A = 5 / 1 = 5

- MIPS B = Ic B / ( ( ( Ic B . CPI B ) / 25 \* 106 ) \* 106 ) = 18

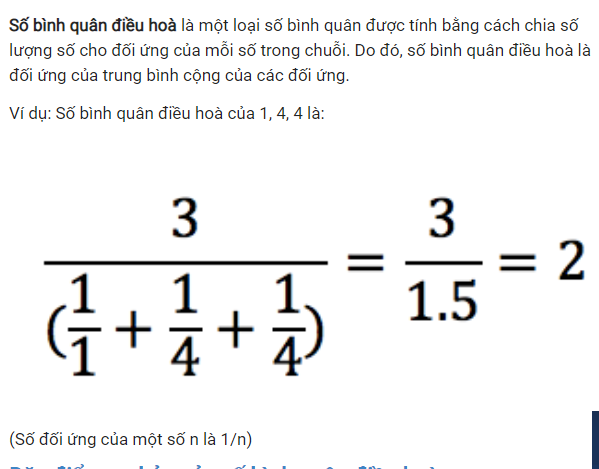
= Ic B / [ ( Ic B \* CPI B ) / 25 ] = 18

=> CPI B = 25 / 18 = 1.4

**2.13** Four benchmark programs are executed on three computers with the following  
results:

|  |  |  |
| --- | --- | --- |
| **Computer A** | **Computer B** | **Computer C** |
| **Program 1** | 1 | 10 | 20 |
| **Program 2** | 1000 | 100 | 20 |
| **Program 3** | 500 | 1000 | 50 |
| **Program 4** | 100 | 800 | 100 |

The table shows the execution time in seconds, with 100,000,000 instructions executed in each of the four programs. Calculate the MIPS values for each computer for  
each program. Then calculate the arithmetic and harmonic means assuming equal  
weights for the four programs, and rank the computers based on arithmetic mean and  
harmonic mean



- A

+ 1: 100

+ 2 : 0.1

+ 3 : 0.2

+ 4 : 2

- B

+ 1: 10

+ 2 : 1

+ 3 : 0.1

+ 4 : 0.125

- C

+ 1: 5

+ 2 : 5

+ 3 : 2

+ 4 : 1

- Arithmetic mean

- A : 25.575 rank 1

- B : 2.8 rank 3

- C : 3.25 rank 2

- Harmonic mean

- A : 0.25 rank 2

- B 0.21 rank 3

- C : 2.1 rank 1

**2.16** Consider the example in Section 2.5 for the calculation of average CPI and MIPS rate,  
which yielded the result of CPI = 2.24 and MIPS rate = 178. Now assume that the  
program can be executed in eight parallel tasks or threads with roughly equal number  
of instructions executed in each task. Execution is on an 8-core system with each  
core (processor) having the same performance as the single processor originally used.  
Coordination and synchronization between the parts adds an extra 25,000 instruction  
executions to each task. Assume the same instruction mix as in the example for each  
task, but increase the CPI for memory reference with cache miss to 12 cycles due to  
contention for memory.  
**a.** Determine the average CPI.  
**b.** Determine the corresponding MIPS rate.  
**c.** Calculate the speedup factor.  
**d.** Compare the actual speedup factor with the theoretical speedup factor determined by Amdhal’s law.

2.17 A processor accesses main memory with an average access time of T2. A smaller cache  
memory is interposed between the processor and main memory. The cache has a  
significantly faster access time of T1 < T2. The cache holds, at any time, copies of some  
main memory words and is designed so that the words more likely to be accessed  
in the near future are in the cache. Assume that the probability that the next word  
accessed by the processor is in the cache is H, known as the hit ratio.  
a. For any single memory access, what is the theoretical speedup of accessing the  
word in the cache rather than in main memory?

- Speedup = T2 /T1  
b. Let T be the average access time. Express T as a function of T1, T2, and H. What is  
the overall speedup as a function of H?

T = H \* T1 + ( 1- H ) \* T2

Speedup = T2 /T = T2 / ( H \* T1 + ( 1 - H ) \* T2 ) = 1 / ( ( 1 - H ) + H \* ( T1 / T2 ) )   
c. In practice, a system may be designed so that the processor must first access the  
cache to determine if the word is in the cache and, if it is not, then access main  
memory, so that on a miss (opposite of a hit), memory access time is T1 + T2.  
Express T as a function of T1, T2, and H. Now calculate the speedup and compare  
to the result produced in part (b).

