2.2. a. On the IAS, what would the machine code instruction look like to load the contents of memory address 2?
b. How many trips to memory does the CPU need to make to complete this instruction during the instruction cycle?

**Answer:**

<table>
<thead>
<tr>
<th>Opcode</th>
<th>Operand</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000001</td>
<td>000000000010</td>
</tr>
</tbody>
</table>

b. In the beginning, the CPU have to fetch the instruction from the memory. Then, the instruction will include the address of the data which is required to load. Through the execution time, the memory will be accessed in that time to load the data contents which is located at that address for a total of two trips to memory.

-----------------------------------------------------------------------------------------------------------------------------

2.3. On the IAS, describe in English the process that the CPU must undertake to read a value from memory and to write a value to memory in terms of what is put into the MAR, MBR, address bus, data bus, and control bus.

**Answer:**

- For reading a content/value from memory, the CPU puts the address into the MAR. Then, the CPU asserts the Read control line to memory and puts the address on the address bus. After that, the memory will copy the contents of that memory location passed on the data bus. Finally, the data will be transferred to the MBR.
- For writing a content/value to memory, the CPU puts the address into the MAR. Then, the CPU puts the data into the MBR. After that, the CPU will assert the Write control line to memory and puts the address on the address bus and the data on the data bus. Finally, the memory will transfer the data on the data bus into the appropriate memory location.

-----------------------------------------------------------------------------------------------------------------------------

2.4. Given the memory contents of the IAS computer shown below, show the assembly language code for the program, starting at address 08A. Explain what this program does.

**Answer:**

<table>
<thead>
<tr>
<th>Address</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>08A</td>
<td>010FA210FB</td>
</tr>
<tr>
<td>08B</td>
<td>010FA0F08D</td>
</tr>
<tr>
<td>08C</td>
<td>020FA210FB</td>
</tr>
</tbody>
</table>

show the assembly language code for the program, starting at address 08A. Explain what this program does.

**Answer:**

<table>
<thead>
<tr>
<th>Address</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>08A</td>
<td>LOAD M(0FA) STOR M(0FB)</td>
</tr>
<tr>
<td>08B</td>
<td>LOAD M(0FA) JUMP +M(08D)</td>
</tr>
<tr>
<td>08C</td>
<td>LOAD –M(0FA) STOR M(0FB)</td>
</tr>
<tr>
<td>08D</td>
<td></td>
</tr>
</tbody>
</table>
This program is to store the absolute value of content at memory location 0FA into memory location 0FB.

2.5. In Figure 2.3, indicate the width, in bits, of each data path (e.g., between AC and ALU).

Answer:
- Overall data paths to/from MBR is 40 bits.
- Overall data paths to/from MAR is 12 bits.
- All Paths to/from AC is 40 bits.
- All Paths to/from MQ is 40 bits.

2.6. In the IBM 360 Models 65 and 75, addresses are staggered in two separate main memory units (e.g., all even-numbered words in one unit and all odd-numbered words in another). What might be the purpose of this technique?

Answer:
The main purpose for doing that arrangement of the main memory is to increase the performance. Therefore, there is some time delay before the read or write operation could be executed because an address is presented to a memory module. So during this is happening, an address could be presented to the other module. Therefore, the maximum rate will be doubled for a series of requests for successive words.

2.8. While browsing at Billy Bob’s computer store, you overhear a customer asking Billy Bob what is the fastest computer in the store that he can buy. Billy Bob replies, “You’re looking at our Macintoshes. The fastest Mac we have runs at a clock speed of 1.2 gigahertz. If you really want the fastest machine, you should buy our 2.4-gigahertz Intel Pentium IV instead.” Is Billy Bob correct? What would you say to help this customer?

Answer:
Even though the machine have a faster clock speed, it does not ensure that the system will work faster. Different systems are not comparable on clock speed. Therefore, there are other factors such as system components like memory, buses,……etc, and the instruction sets have to be checked also. Both systems have to be run on a benchmark to more accurate measure. The main purpose for producing benchmark programs is to perform particular functions such as performing graphics operations, running office applications, performing floating-point operations, and so on. Then, by comparing both systems to each other, and compute how long they will need to complete these functions. By doing kind of comparing between Apple and Pentium we could say that regarding to Apple computer, the G4 is equivalent or better than a higher-clock speed Pentium on a lot of benchmarks.

2.9. The ENIAC was a decimal machine, where a register was represented by a ring of 10 vacuum tubes. At any time, only one vacuum tube was in the ON state, representing one of the 10 digits. Assuming that ENIAC had the capability to have multiple vacuum tubes in the ON and OFF state simultaneously, why is this representation “wasteful” and what range of integer values could we represent using the 10 vacuum tubes?

Answer:
Regarding to this representation, it is considered a wasteful one because multiple tubes could be on or off simultaneously with each group of 10 vacuum tubes. Therefore, there are two methods to use it as a good one. The first method is to use the 10 tubes as a binary value, giving a range of $2^{10} = 1024$ numbers as signed or unsigned number. The second method is to use only four tubes rather than 10 to
represent the digits from 0 to 9 as binary value. Thus, with ten bits we can represent $2^{10}$ patterns or 1024 patterns. For integer values, these patterns could be used to represent the numbers from 0 through 1023.

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:

<table>
<thead>
<tr>
<th>Instruction Type</th>
<th>Instruction Count (millions)</th>
<th>Cycles per Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Machine A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Arithmetic and logic</td>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>Load and store</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Branch</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>Others</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Machine A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Arithmetic and logic</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>Load and store</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>Branch</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>Others</td>
<td>4</td>
<td>3</td>
</tr>
</tbody>
</table>

a. Determine the effective CPI, MIPS rate, and execution time for each machine.
b. Comment on the results.

**Answer:**

\[
CPI_A = \frac{\sum CPI_i \times I_i}{I_e} = \frac{(8 \times 1 + 4 \times 3 + 2 \times 4 + 4 \times 3) \times 10^6}{(8 + 4 + 2 + 4) \times 10^6} \approx 2.22
\]

\[
MIPS_A = \frac{f}{CPI_A \times 10^6} = \frac{200 \times 10^6}{2.22 \times 10^6} = 90
\]

\[
CPU_A = \frac{I_e \times CPI_A}{f} = \frac{18 \times 10^6 \times 2.2}{200 \times 10^6} = 0.2 \text{ s}
\]

\[
CPI_B = \frac{\sum CPI_i \times I_i}{I_e} = \frac{(10 \times 1 + 8 \times 2 + 2 \times 4 + 4 \times 3) \times 10^6}{(10 + 8 + 2 + 4) \times 10^6} \approx 1.92
\]

\[
MIPS_B = \frac{f}{CPI_B \times 10^6} = \frac{200 \times 10^6}{1.92 \times 10^6} = 104
\]

\[
CPU_B = \frac{I_e \times CPI_B}{f} = \frac{24 \times 10^6 \times 1.92}{200 \times 10^6} = 0.23 \text{ s}
\]

b. Even though, machine B has a higher MIPS than machine A, it needs a longer CPU time to execute the similar set of benchmark programs (instructions).
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:

<table>
<thead>
<tr>
<th>Processor</th>
<th>Clock Frequency</th>
<th>Performance</th>
<th>CPU Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAX 11/780</td>
<td>5 MHz</td>
<td>1 MIPS</td>
<td>12 x seconds</td>
</tr>
<tr>
<td>IBM RS/6000</td>
<td>25 MHz</td>
<td>18 MIPS</td>
<td>x seconds</td>
</tr>
</tbody>
</table>

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. What are the CPI values for the two machines?

Answer:

a. The MIPS rate could be computed as the following:

\[
\left( \frac{\text{MIPS rate}}{10^6} \right) = \frac{I_c}{T}
\]

Thus that:

\[
I_c = T \times \left( \frac{\text{MIPS rate}}{10^6} \right)
\]

Now by computing the ratio of the instruction count of the IBM RS/6000 to the VAX 11/780 which is:

\[
\frac{x \times 18}{12x \times 1} = \frac{18x}{12x} = 1.5
\]

b. Regarding to the VAX 11/780, the CPI = (5 MHz) / (1 MIPS) = 5

Regarding to the IBM RS/6000, the CPI = (25 MHz) / (18 MIPS) = 1.4

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

<table>
<thead>
<tr>
<th>Program</th>
<th>Computer A</th>
<th>Computer B</th>
<th>Computer C</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>10</td>
<td>20</td>
</tr>
<tr>
<td>2</td>
<td>1000</td>
<td>100</td>
<td>20</td>
</tr>
<tr>
<td>3</td>
<td>500</td>
<td>1000</td>
<td>50</td>
</tr>
<tr>
<td>4</td>
<td>100</td>
<td>800</td>
<td>100</td>
</tr>
</tbody>
</table>

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.

Answer:

MIPS rate given as

\[
\text{MIPS} = \frac{I_c}{T \times 10^6}.
\]

Arithmetic mean is

\[
R_A = \frac{1}{m} \sum_{i=1}^{m} R_i.
\]

Harmonic mean is

\[
R_H = \frac{m}{\sum_{i=1}^{m} \frac{1}{R_i}}.
\]

By applying \(MIPS = \frac{I_c}{(T \times 10^6)} = 100,000,000/(T \times 10^6) = 100/T\). Therefore, the MIPS values are:
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.

Answer:

a. Since we have the same instruction mix, that means the additional instructions for each task could be allocated appropriately between the instruction types. Therefore, the following table could be gotten:

<table>
<thead>
<tr>
<th>Instruction Type</th>
<th>CPI</th>
<th>Instruction Mix</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arithmetic and logic</td>
<td>1</td>
<td>60%</td>
</tr>
<tr>
<td>Load/store with cache hit</td>
<td>2</td>
<td>18%</td>
</tr>
<tr>
<td>Branch</td>
<td>4</td>
<td>12%</td>
</tr>
<tr>
<td>Memory reference with cache miss</td>
<td>12</td>
<td>10%</td>
</tr>
</tbody>
</table>

The average CPI = (1×0.6) + (2×0.18) + (4×0.12) + (12×0.1) = 2.64. Therefore, the CPI has been increased since the time for memory access is also increased.

b. MIPS = 400/2.64 = 152. There is a corresponding drop in the MIPS rate.
c. The speedup factor equals to the ratio of the execution times. The execution time is calculated as the following:

\[ T = \frac{I_c}{(\text{MIPS} \times 10^6)}. \]

For the one processor, \( T_1 = \frac{(2 \times 10^6)}{(178 \times 10^6)} = 11 \text{ ms.} \)

For the 8 processors, each processor executes \( \frac{1}{8} \) of the 2 million instructions plus the 25,000 overhead instructions. There, the execution time \( T \) for each of the 8 processors is:

\[ \frac{2 \times 10^6}{8} = 0.025 \times 10^6 = 1.8 \text{ ms} \]

Therefore we have

\[ \text{Speedup} = \frac{\text{time to execute program on a single processor}}{\text{time to execute program on N parallel processors}} = \frac{11}{1.8} = 6.11 \]

d. In fact, there are two inefficiencies in the parallel system. The first one is that there are more additional instructions which is added to coordinate between threads. The second one is that there is contention for memory access. Thus, none of the code is inherently serial, and all of it is parallelizable but with scheduling overhead. It could be said that the memory access conflict means some extent memory reference instructions are not parallelizable. By depending on the information given, it is not obvious how to quantify this effect in Amdahl's equation. Therefore, if it is supposed that the fraction of code, which is parallelizable, is \( f = 1 \), then Amdahl's law decreases to Speedup = \( N = 8 \). Therefore, the actual speedup is only about 75% of the theoretical speedup.

2.17. A processor accesses main memory with an average access time of \( T_2 \). A smaller cache memory is interposed between the processor and main memory. The cache has a significantly faster access time of \( T_1 < T_2 \). 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?

b. Let \( T \) be the average access time. Express \( T \) as a function of \( T_1 \), \( T_2 \), and \( H \). What is the overall speedup as a function of \( H \)?

Answer:

a. Speedup = (time to access in main memory) / (time to access in cache)
\[ = \frac{T_2}{T_1} \]

b. The average access time could be calculated as the following:
\[ T = H \times T_1 + (1 - H) \times T_2 \]
By applying the following equation:

\[ \text{Speedup} = \frac{\text{Execution time before enhancement}}{\text{Execution time after enhancement}} = \frac{T_2}{T} = T_2 / (H \times T_1 + (1-H)T_2) \]
\[ = 1 / ((1−H) + H \times (T_1 / T_2)) \]

c. \[ T = H \times T_1 + (1 - H) \times (T_1 + T_2) = T_1 + (1 - H) \times T_2 \]
This is Equation in Chapter 4. Now,
\[ \text{Speedup} = \frac{\text{Execution time before enhancement}}{\text{Execution time after enhancement}} = \frac{T_2}{T} = T_2 / (T_1 + (1-H)T_2) = 1 / ((1−H) + T_1 / T_2) \]
Therefore, since the denominator is larger, the speedup will be less.