1. Problem 4.7 from Hwang and Zhiwei, “Scalable Parallel Computing”

2. Problem 4.8 from Hwang and Zhiwei, “Scalable Parallel Computing”

3. Problem 4.9 from Hwang and Zhiwei, “Scalable Parallel Computing”

4. Problem 4.10 from Hwang and Zhiwei, “Scalable Parallel Computing”
1-

\[ CPI = \frac{45 \times 1 + 32 \times 2 + 15 \times 2 + 8 \times 2}{45 + 32 + 15 + 8} = \frac{155}{100} = 1.55 \text{ cycles/instruction} \]

\[ \text{MIPS rate} = \frac{40 \text{ MHz}}{1.55 \text{ CPI}} = 25.8 \text{ MIPS} \]

\[ \text{Execution time} = \frac{(45000 \times 1 + 32000 \times 2 + 15000 \times 2 + 8000 \times 2) \text{cycles}}{(40 \times 10^6) \text{cycles/s}} = 3.875 \text{ ms} \]

The execution time can also be obtained by dividing the total number of instructions by the MIPS rate:

\[ \frac{(45000 + 32000 + 15000 + 8000) \text{instructions}}{(25.8 \times 10^6) \text{instructions/s}} = 3.875 \text{ ms} \]

2-

(a) The speedup factor \( S_k \) of a \( k \)-stage pipeline over an equivalent non-pipelined processor is:

\[ S_k = \frac{T_i}{T_k} = \frac{nk\tau}{k\tau + (n - 1)\tau} = \frac{nk}{k + (n - 1)} \]

where \( \tau \) is the clock period

\( k = \) number of stages = 5

\( n = \) number of instruction = 15000

\( \therefore \) Speedup \( S_k = \frac{15000 \times 5}{5 + (15000 - 1)} \approx 4.9987 \)

(b) The efficiency \( E_k \) of a linear \( k \)-stage pipeline is:

\[ E_k = \frac{S_k}{k} = \frac{n}{k + (n - 1)} \]

\( \therefore \) Efficiency \( E_k = \frac{15000 \times 1.25}{5 + (15000 \times 1.25 - 1)} \approx 0.9984 \)

The throughput \( H_k \) is defined as the number of tasks (operations) performed per unit time.

\[ H_k = \frac{n}{[k + (n - 1)]\tau} = \frac{nf}{[k + (n - 1)]} \]

\[ = \frac{15000 \times 1.25}{5 + (15000 \times 1.25 - 1)} \times 25 \text{MIPS} \approx 24.96 \text{MIPS} \]
(a) Let \( k \) be the number of pipeline stages, \( m \) be the superscalar degree, \( N \) be the number of independent instructions and \( n \) be the superpipeline degree (that is the number of pipeline cycles).

The time required by the scalar base machine is

\[
T(1, 1) = k + N - 1 \quad \text{(base cycles)}
\]

The minimum time needed to execute \( N \) independent instructions on a superpipelined superscalar machine of degree \( (m, n) \) as:

\[
T(m, n) = k + \frac{N - m}{mn} \quad \text{(base cycles)}
\]

Thus the speedup \( S(m, n) \) over the base machine is:

\[
S(m, n) = \frac{T(1, 1)}{T(m, n)} = \frac{k + N - 1}{k + \frac{N - m}{mn}} = \frac{mn(k + N - 1)}{mnk + N - m}
\]

(b) The practical limitations preventing the growth of the \( m \)-issue superscalar processor are:

1. Additional and duplicated hardware is required for the implementation of multiple pipelines. These hardware are limited by the cost, die area, bus cycle speed, internal and external bus bandwidth, etc.
2. Since multiple instructions are issued in each clock cycle, they will compete for different functional units available such as Integer ALU, Floating Point Unit, etc. Therefore, these resource and inherently the complexity of the microprocessor architecture limit the growth of the superscalar degree.
3. Affected by the algorithms and data structures used in the user programs, the average value of Instruction Level Parallelism is measured between 2 to 5 in ordinary program traces. For this reason, the inherent parallelism among instructions existing in the programs may not be sufficient to fill all the available instruction slots.
4. The compiler technology may not be fully capable of optimizing the parallelism in the user program without inducing resource conflicts, data dependency and branches, etc. Therefore, the superscalar architecture unable to be fully utilized even all the above problems are solved.

(c) The practical limitations preventing the growth of the superpipeline with an \( n \)-times faster clock are:

1. A high degree of pipelining requires a high clock rate but since the advance in computer architecture, the increase in processor clock rate far lags behind the increase in the processor speed. Also, because the equipment processing high frequency is much more costly, the high cost further limits the growth of superpipeline.
2. Introducing more pipeline stages will raise the chance of resource conflicts as the same resources may be used in different stages.
3. If the compiler is not smart enough, heavy penalties will arise from the issues of data dependency, control dependency and resource conflicts as a result of increasing the number of pipeline stages.
4. By clock technologies since a high clock rate is needed for a larger degree of pipelining.
(a) X needs 400 (=4×100) cycles to execute the program. It takes 16000 ns (400×40 ns). Y needs 104 (=5+[100-1]) cycles to execute the program. It takes 5200 ns (104×50).

\[
\text{Speedup} = \frac{16000}{5200} = 3.08
\]

(b) MIPS rates are computed as follows:

\[
X : \frac{100}{16\,\mu s} = 6.25\, MIPS
\]

\[
Y : \frac{100}{5.2\,\mu s} = 19.2\, MIPS
\]