Francesco Zhang
301133976

CMPT295
Assignment 7

1. [3 marks] Equal-lengthed Stages Consider the design of a processor, with a max instruction length of 1000 ps. The propagation delay to load a register is 25 ps.

(a) [1 mark] What is the minimum clock cycle time, the instruction latency and CPU throughput using serial execution?
Answer:
Minimum clock cycle time = 1000ps because a non-pipelined execution has to go through all the stages in an instruction.
Instruction latency = 1000ps because the latency is the time to go through 5 stages.
CPU throughput = 1 / 1000ps=1GHz

(b) [1 mark] What is the minimum clock cycle time, the instruction latency and CPU throughput using a pipelined execution with 10 equal stages?
Answer:
Minimum clock cycle time = 1000 / 5 + 25 = 225ps because a pipelined execution has to go through the stage spent most time and the register.
Instruction latency = 225 * 5 = 1125ps because the latency is the time to go through 5 stages.
CPU throughput = 1 / 225ps = 4.444GHz

(c) [1 mark] Consider a design which used n equal stages. What is the minimum clock cycle time, the instruction latency and CPU throughput expressed as a function of n? (You may wish to check that your generalization agrees with your results from parts (a) and (b), i.e., by substituting n = 1, 10.)
Answer:
Serial:
Cycle time = 1000ps, Latency = 1000ps, and Cpu throughput = 1Ghz because serial execution does not concern the number of stages. It is the same result as (a).
Pipeline:
Cycle time= 1000 / n + 25, Latency = (1000 / n + 25) * n, Cpu throughput = 1/(1000/n+25)
When n=1, cycle time=1025ps, Latency=1025ps, Cpu throughput=975.6MHz
When n=10, cycle time=125ps, Latency=1250ps, Cpu throughput=8GHz
They are different from the result of (a) and (b).

2. [2 marks] Serial vs Pipelined Instructions Consider two systems: both have a clock cycle of 450 ps and a register load time of 50 ps. The difference? The first one is serially executed, while the second one uses a pipeline of 5 stages per instruction. What is the instruction throughput of each machine? What is the instruction latency?
Answer:
Serial execution:
Latency = 450ps ,and throughput = 1/450ps = 2.222Ghz because serial execution spend the time of the whole cycle.
Pipeline execution:

Latency = (450 / 5 + 25) * 5 = 575ps, Cpu throughput = 1 / (450 / 5 + 25) = 8.696GHz because the latency is the time to go through 5 stages which all spending the cycle time.

3. [4 marks] Oddly-lengthed Stages Instruction execution design might not always result in stages that are equal in length. As an example, consider a system that can be cleanly divided into 6 stages, in the order (A, B, C, D, E, F), each with a propagation delay (in ps) of (250, 200, 150, 80, 100, 220), for a grand total of 1000 ps. The register loading time is 25 ps.
(a) [1 mark] If you only had one extra set of registers to place between an adjacent pair of stages in order to form a 2-stage pipeline, where would you place them? Compute the minimum clock cycle time and the maximum possible CPU throughput.
Answer:
I will place the set of register between stage F and stage A because they are the two stages spend most time. Making pipeline here can save most time.
Minimum cycle time = 250 + 200 + 150 + 80 + 100 + 25 = 805ps.
Throughput = 1 / 805ps = 1.242GHz.
(b) [1 mark] If you had two extra sets of registers, where would you place them to form a 3-stage pipeline? Again, compute the min clock cycle time and max throughput.
Answer:
The first set of registers will be placed between stage F, stage B and stage A.
Minimum cycle time = 250 + 150 + 80 + 100 + 25 = 605ps.
Throughput=1/805ps = 1.653GHz.
(c) [1 mark] If you had five extra sets of registers, and created a 6-stage pipeline, what would the min clock cycle time and max throughput be?
Answer:
Minimum cycle time = 250 + 25 = 275ps.
Throughput=1/805ps = 3.636GHz.
(d) [1 mark] Suppose you had two extra sets of registers, like in part (b). If you could direct your designers to divide any of A, B, C, D, E, F into two stages, where would you tell them to concentrate their efforts?
Answer:
I will direct them to make the instruction length of two stages to be as same as possible. In this case, I can make A, C, D as stage 1, and B, E, F as stage 2.

4. [3 marks] Persistent Dynamic Memory Build and run the code in the q4 subdirectory from the care-package. When you compare the output with the corresponding C code, it appears that there is a bug in printf(). Perhaps in the other function. There is a bug, however it is not within either function. Explain where the bug really is, and why printf() displays what it does.
Answer:
There is a bug in the functions of main and new_42. The memory allocation of the space for the pointer int_ptr pointing to is done in the function of new_42 that it is stored in the stack. Then, after new_42 is finished, the space allocated is set to be free. The pointer in the main function can still access the data now. However, if the main function continues to allocate memory or call other functions, the data here becomes dangerous. In this situation, calling the function of donotmuchofanything occupy the address used to store 42, then the output become 0.

5. [8 marks] Linked Lists and Pipeline Stalls To search a linked list of values costs O(n), i.e., linear time, the same running time for a simple linear search of an array. But not all

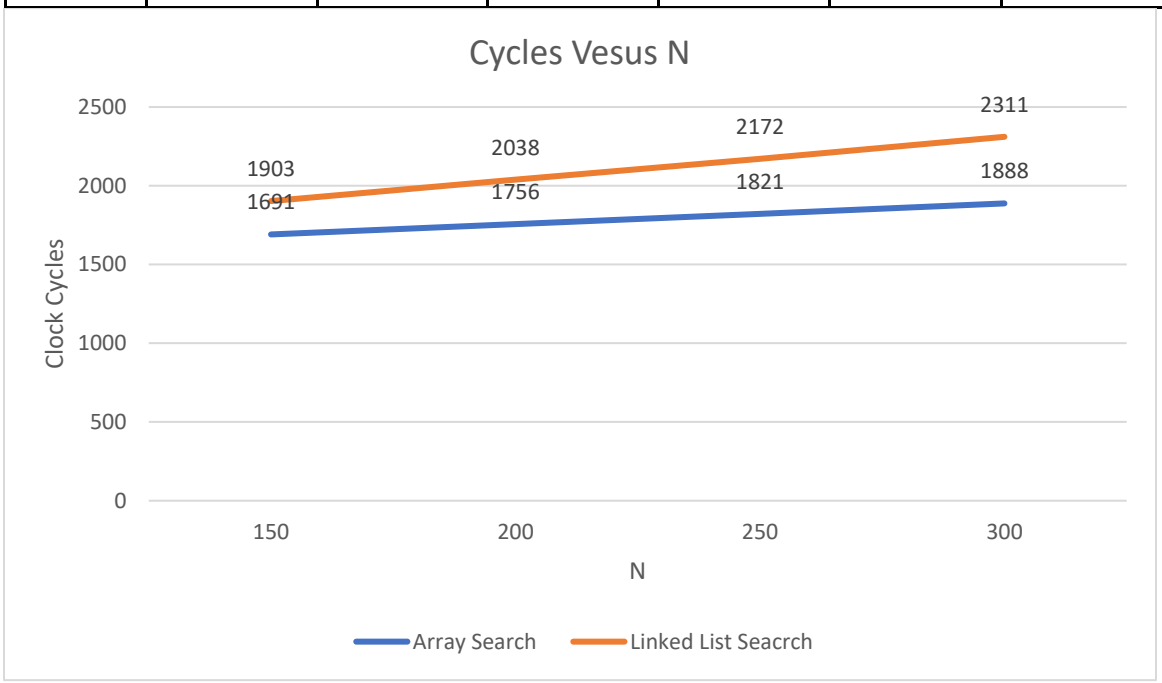
linear time algorithms are created equal: to choose, you can measure the cost per element (CPE) using benchmarking.

(a) [6 marks] Hardcopy: Tabulate your results, one quintet of measurements per row. Compute the average of each set of 5 times. Then, plot both data sets on a graph and draw a straight line through each set. Compute the slope of each line. The slope = the cycles per element (CPE) for each algorithm.
Answer:
Array search

N	t1	t2	t3	t4	t5	Average
150	1693	1691	1690	1691	1691	1691
200	1756	1754	1757	1756	1756	1756
250	1821	1820	1823	1821	1821	1821
300	1888	1887	1889	1888	1890	1888

Linked List search

N	t1	t2	t3	t4	t5	Average
150	1907	1901	1901	1905	1900	1903
200	2036	2041	2036	2038	2038	2038
250	2173	2169	2170	2180	2168	2172
300	2310	2312	2311	2310	2311	2311



The slope of array search=(1888-1691)/(300-150)=1.313
The slope of linked list search=(2311-1903)/(300-150)=2.72

(b) [2 marks] Hardcopy: The main reason for the difference in the CPE is a data dependency in the linked list search that isn't present in the array search. Explain the sequence of instructions that causes the delay.
Answer:
In the linked list search, the following construction cause the delay.

```
.L4:
        movq    8(%rdx), %rdx
        testq   %rdx, %rdx
        jne     .L10
```

The mov instruction read the address of next pointer in the current struct. It needs to first run an addiction to the rdx register, then dereference it. Finally, check whether the pointer is null by using testq. Also, LLsearch use the 64bit register which is spending more time. Therefore, this instruction includes more stages than the one needed in array search. Then the latency for these actions become longer.