HW#5 2017069598 박상지

1)Write a report summarizing the materials discussed in Topic 1-1

Computer Performance

좋은 컴퓨터란 무엇인가? (무엇을 측정해야 하는가?)

what do we measure for Computer Performance?

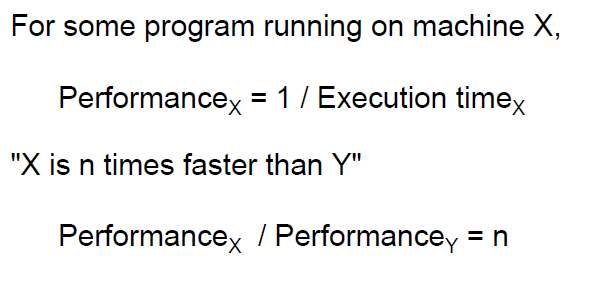
1. Response time: 작업을 끝 마치는데 얼마나 오래 걸리는가?
2. Throughput: 하나의 단위 유닛을 마무리하는데 걸리는 전체 시간

우리는 응답 시간에 보통 초점을 둔다. 왜냐하면 Throughput 자체의 성능을 높이려면 대규모 설계가 필요하다. 하지만 Response time을 줄이면 자동적으로 Throughput도 동시에 성능이 좋아지기 때문에 Throughput보다는 성능 개선이 편한 Response time에 초점을 둔다.

Multiprocessor Systems

1. Parallel Programming: 하나의 큰 문제를 해결하는 것으로 날씨 예보가 대표적인 예이다. 복잡한 공학 계산에서는 공학자가 푸는 문제는 대게 어렵다. 날씨 예보의 프로세서 같은 경우에는 계산 소유 시간이 많아진다. 그러므로 여러 개의 CPU를 사용하여 작업을 나눔으로써 Response time을 줄인다.
2. Natural Parallelism: Parallel 와 대비되는 개념으로 많은 독립적인 사용자가 오기 때문에 서로 다른 작업에 대해 담당 프로세서에 할당을 한다. 그러기 때문에 Response time 보다는 Throughput의 성능을 높이는게 좋다. (전체 소유 시간을 줄이는 것.)

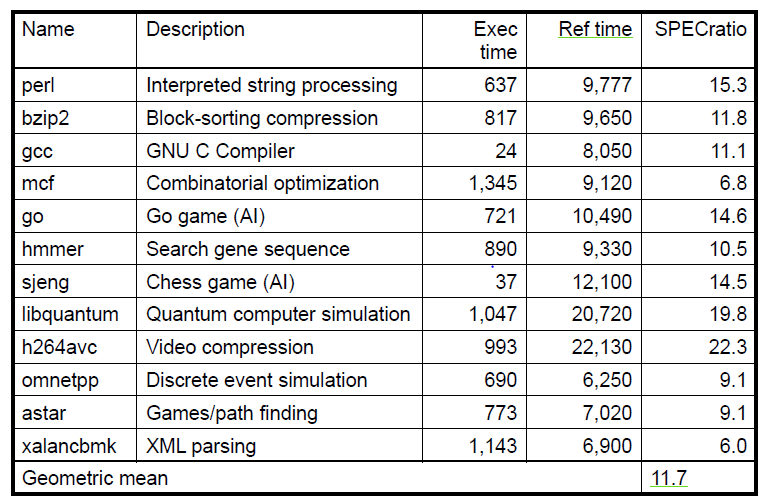
Relative Performance



How do we measure for Computer Performance of Evaluation methods?

SPEC(System Performance Evaluation Corporation)

컴퓨터 수행을 평가하는 방법으로 실제 응용 프로그램을 대표할 수 있는 프로그램이 선정 되어있다. 그것을 Benchmarking이라 불리는데 Benchmarking으로 Response time을 측정한다.

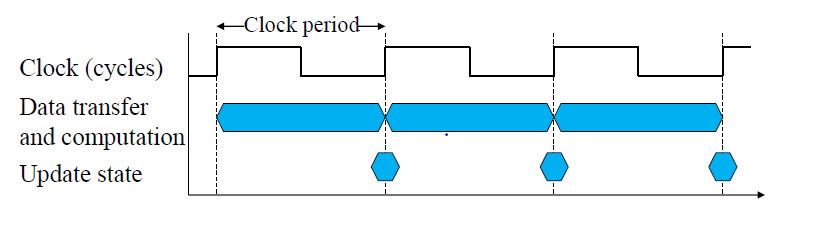


SPEC CPU Benchmark 표를 보자면 이름이 ‘perl’인 첫번째 줄에서 해당 프로그램의 실행 시간은 637이다. 하지면 Reference time을 평균 9,777 이다. ‘perl’ 프로그램에서 다른 평균적인 프로그램보다 15.3배 빠른 것이다.

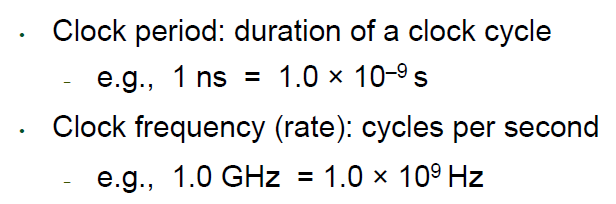
마지막 줄을 보자면 해당 machine은 기존 machine들 보다 Integer performance가 11.7배가 빠르다.

How can we improve for Computer Performance?

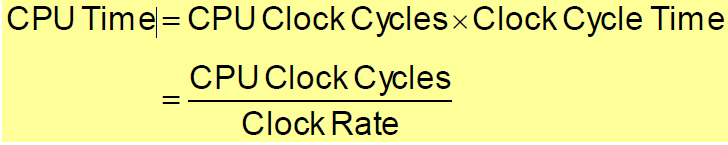
CPU clocking



clock이 올라가게 되면 P-D-E 작업이 실행이 된다. 다음 clock이 올라오게 되면 해당 작업을 저장하고 다음 작업을 실행을 한다.



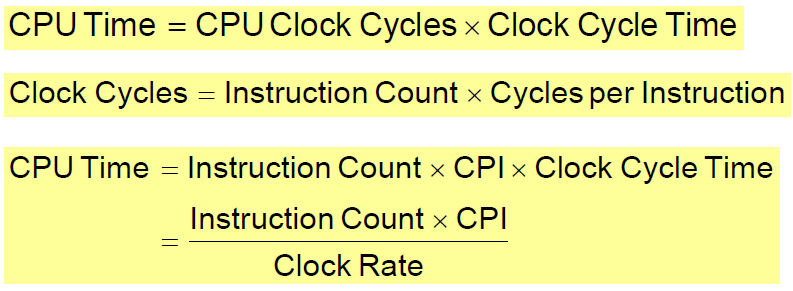
CPU Time



CPU time은 프로그램을 돌리는데 clock cycles이 몇 개가 필요하고 clock 길이, 즉 시간이 얼마나 되는지에 따라 결정된다.

\*clock cycles의 수를 감소시키는 것이 컴퓨터 성능을 높여준다.

\*clock rate을 증가시키는 것이 컴퓨터 성능을 높여준다. (ex. 1GHz -> 2GHz)



Three Factors: IC, CPI, cct

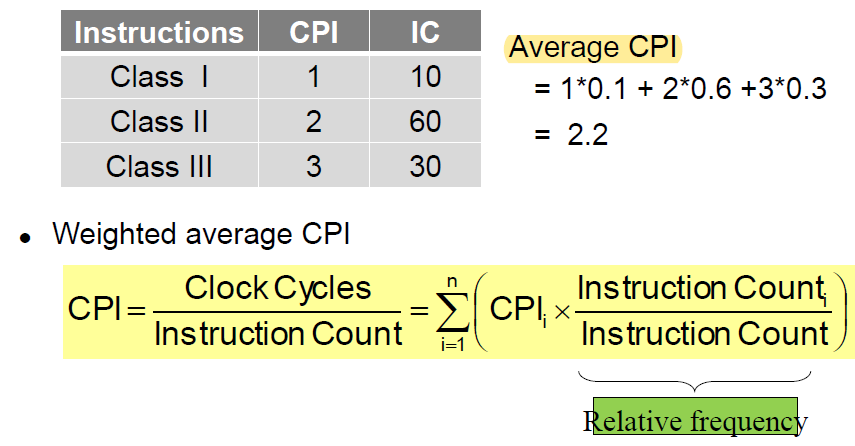
IC(instruction count)

* machine instruction이 실행되는 수로 동적인 수라고 말한다. 동적인 수(Dynamic(run time) count)란? 실제 실행되는 instruction의 수로 개발자가 작성한 코드에는 많은 machine instruction이 있다. instruction 마다 실행이 되는 것과 실행이 안되는 것이 존재한다. 또는 Loop 라는 instruction이 존재하는데 만약 n이 10000이라면 이 Loop라는 IC는 10000이 된다. 만약 n이 input data 라면 해당 프로그램은 input data마다 IC가 변한다. 여기에서 Input을 고정하거나 여러 input에 대한 평균을 취하여 IC를 check을 한다.
* 최상위 개념 IC는 ISA(instruction set architecture)(I/F) 설계에서 결정되며, ISA가 확정이 된다면 컴파일(Compiler) 설계가 가능하다.

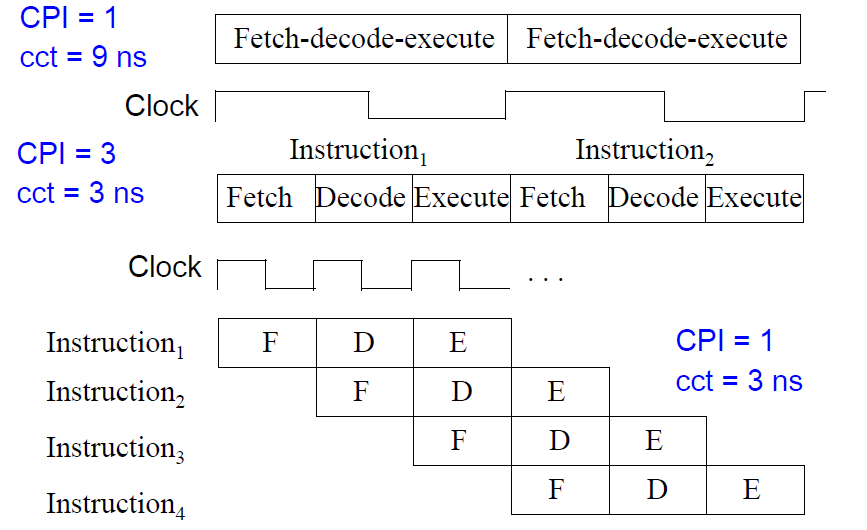
CPU 시뮬레이터로 Benchmark을 돌리면 IC를 얻고, 이에 따라 implementation(CPI & cct)가 영향을 받는다.

CPI(cycle per instruction)

* 차상위 개념, high-level implementation – 한 instruction이 실행되는데 필요한 clock cycles의 수로 instruction 마다 다르다.
* 최종적인 CPI는 빈도에 따른 weighted average로 계산한다. – 실제 실행되는 Dynamic count를 고려한다.
* 설계된 ISA 구현시 CPI가 결정된다.



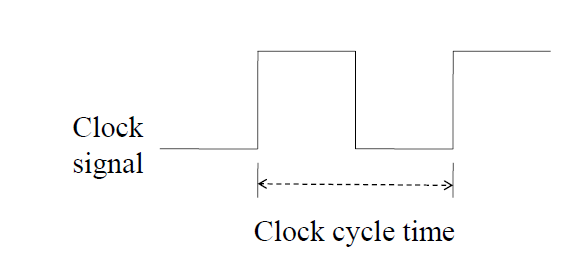
\*Pipelining: 3-Stage Pipeline



첫 번째 그림에서 CPI는 1이고, cct는 9ns 로 가정하자. 해당 CPI를 3등분을 하면 자동적으로 cct도 3등분이 되어 진다. 그렇다면 해당 CPI와 cct를 가지고 3-stage로 Pipeline을 설계를 한다면 한 cycle에 instruction의 수는 1이 되고 cct는 3ns 이기 때문에 IC는 3 ns 가 된다.

cct(clock cycle time)

하위 개념으로 clock 한 주기의 길이를 의미한다. ISA 및 high level implementation이 결정되면, 한 클럭 내에서 할 일 들일 결정되는데 cct를 줄일수록 성능이 좋아진다.



회로 설계자는 cct를 줄이려 노력을 하는데, clock 내에 할 일들 중에서 가장 오래 걸리는 작업을 더욱 빠르게 만들려고 노력을 한다. 1clock cycle은 이미 결정되었지만 그 안에 signal을 줄이려고 노력을 한다.

2) Solve chapter 1 exercises: 2, 5, 7, 9, 11, 14

1.2

a) Performance via Pipelining

b) Dependability via Redundancy

c) Performance via Prediction

d) Performance via Parallelism

e) Make the Common Case Fast

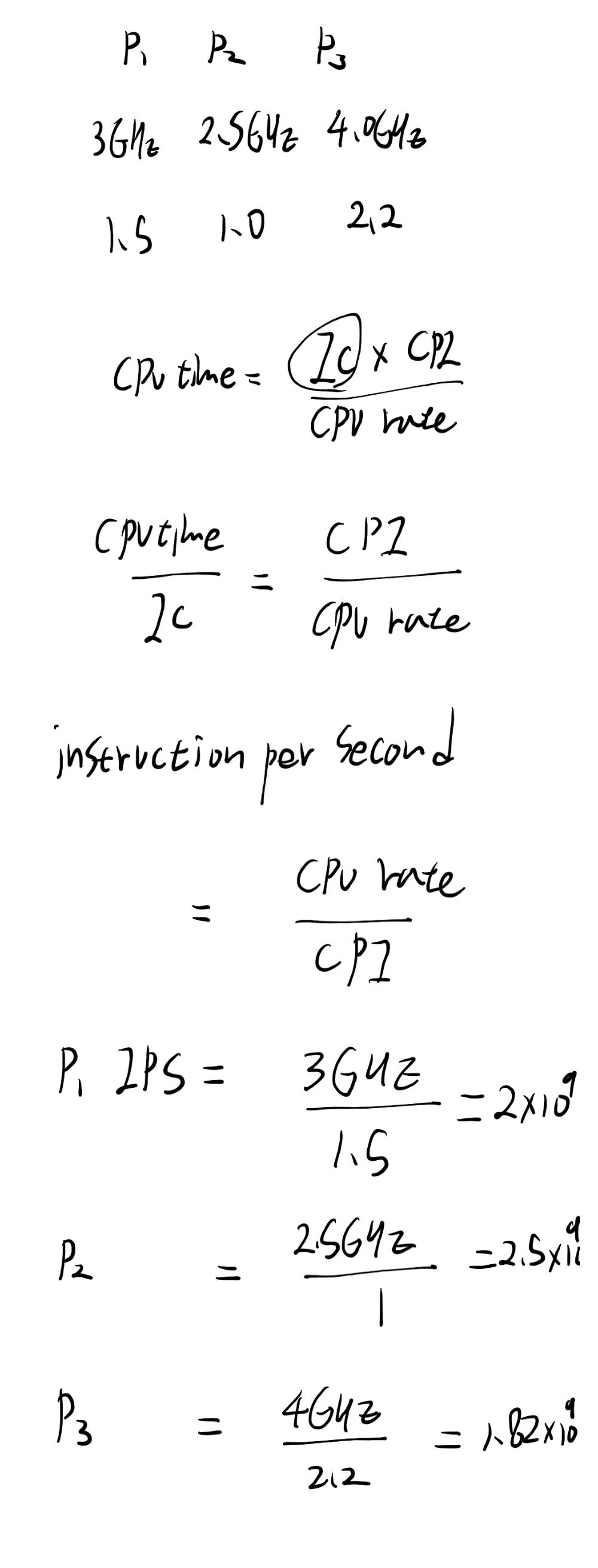
f) Hierarchy of Memories

g) Design for Moores Law

h) use Abstraction to Simplify Design

1.5

a) processor2



b)

1. p1 instructions = instruction per second(1) \* CPU time1 = 2 \* 10 ^ 10

p1 clock cycles = CPU time1 \* clock rate1 = 10 \* 3 GHz = 3 \* 10 ^10

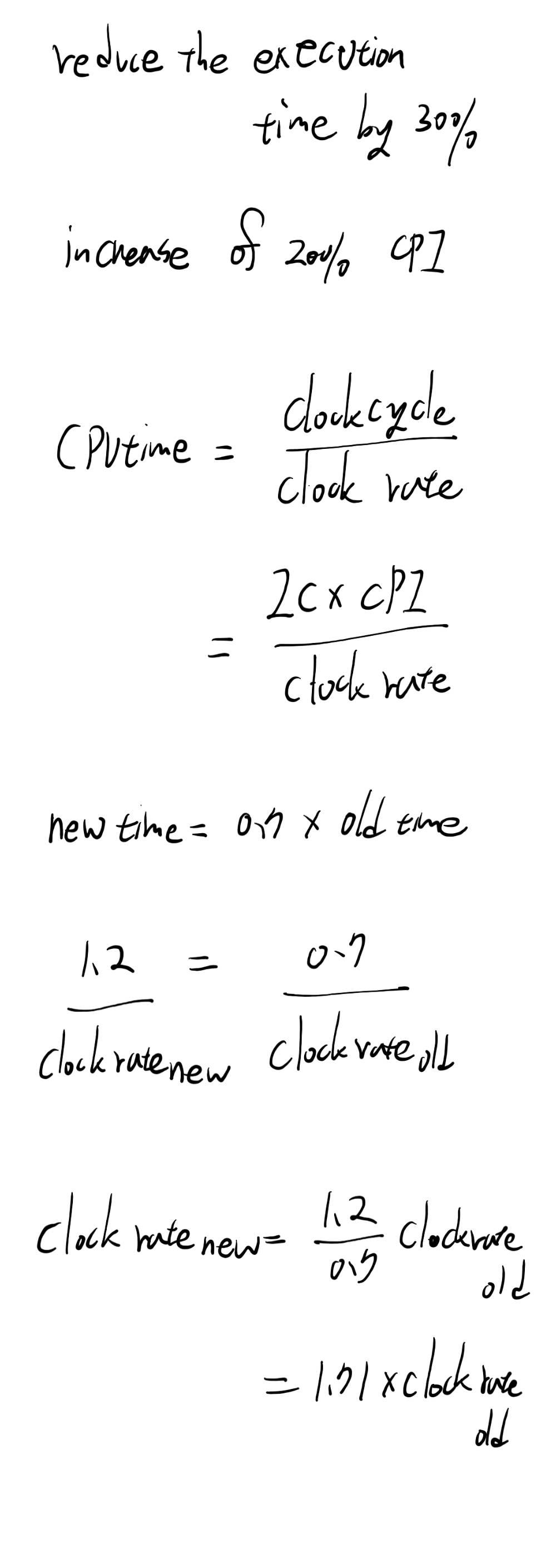
2. p2 instructions = instruction per second(2) \* CPU time2 = 2.5 \*10 ^ 10

p2 clock cycles = CPU time2 \* clock rate2 = 10 \* 2.5GHz = 2.5 \* 10 ^ 10

3. p3 instructions = instruction per second(3)\* CPU time3= 1.82 \* 10 ^ 10

p3 clock cycles = CPU time3\* clock rate3= 10 \* 4 GHz = 4 \* 10 ^ 10

c) 대략 71 %



1.7

a) A : 1, B : 1,25

b) 프로세서 1이 대략 33% 느리다

c) A보다 대략 1.67 빠르고, B보다 대략 2.27 빠르다.

1.9

1)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 1 | 2 | 4 | 8 |
| Execution time in seconds | 9.6 | 7.04 | 3.84 | 2.24 |
| Speed- up | 1 | 1.36 | 2.5 | 4.29 |

2)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 1 | 2 | 4 | 8 |
| Execution time in seconds | 10.88 | 7.965 | 4.3025 | 2.47 |
| Speed- up | 1.13 | 1.13 | 1.12 | 1.1 |

3) 3

1.11

1) 0.942759

2) 12.866667

3) 1.1

4) 1.155

5) 11.19

6) 1.378868

7) CPI는 clock rate보다 더 증가된다. 왜냐하면 instruction 수가 줄어들었기 때문이다.

8) 6.67%

9) 2146.583851

10) 4.445 GHz

11) 4.72 GHz

1.14

1) 계산을 하면 CPI 값이 -4.13이 나온다. CPI는 음수 값을 가질 수 없기 때문에 불가능하다.

2) CPI는 0.79 값을 가지고 80.25% 가 감소율을 보여준다.

3) 33.33%