**1.2** [5] <§1.2> The eight great ideas in computer architecture are similar to ideas from other fields. Match the eight ideas from computer architecture, “Design for Moore’s Law”, “Use Abstraction to Simplify Design”, “Make the Common Case Fast”, “Performance via Parallelism”, “Performance via Pipelining”, “Performance via Prediction”, “Hierarchy of Memories”, and “Dependability via Redundancy” to the following ideas from other fields:

**a.** Assembly lines in automobile manufacturing

They use the idea of pipelining, so it matches **Performance via Pipelining**

**b.** Suspension bridge cables

Each cable holds a part of the bridge up so it is similar to multiple processes completing work at the same time therefore it is **Performance via Parallelism**

**c.** Aircraft and marine navigation systems that incorporate wind information

**Performance via Prediction,** it uses wind to predict a better route generation

**d.** Express elevators in buildings

**Make the common case fast**, Express elevators used to go to the most common locations which is traveled to faster

**e.** Library reserve desk

**Hierarchy of Memories**, desk contains things that are most frequently required by students. Similar to Caching

**f.** Increasing the gate area on a CMOS transistor to decrease its switching time

**Dependability via Redundancy**, the increase in gate area is redundant to the increase in switching times.

**g.** Adding electromagnetic aircraft catapults (which are electrically-powered as opposed to current steam powered models), allowed by the increased power generation offered by the new reactor technology

**Design for Moore’s Law**, technological advancement which is Moore’s law

**h.** Building self-driving cars whose control systems partially rely on existing sensor systems already installed into the base vehicle, such as lane departure systems and smart cruise control systems

**Use Abstraction to simplify design**, Using abstraction is the same as self-driving cars that hide all of the complicated sensor systems.

**1.3** [2] <§1.3> Describe the steps that transform a program written in a high-level language such as C into a representation that is directly executed by a computer processor.

Step 1: Translate from C-language program into an assembly language program which is essential from the compiler.

Step 2: Translate from the assembly language into Machine Code. Tool used for this step is the Assembler

**1.4** [2] <§1.4> Assume a color display using 8 bits for each of the primary colors (red, green, blue) per pixel and a frame size of 1280 × 1024.

**a.** What is the minimum size in bytes of the frame buffer to store a frame?

8 bits \* 3 colors / 8 = 3 bytes per pixel

1280 \* 1024 pixels = 1,310,720 pixels = 1,310,720 \* 3 bytes/frame = 3,932,160 bytes/frame

Minimum size is 3,932,160 bytes/frame

**b.** How long would it take, at a minimum, for the frame to be sent over a 100

Mbit/s network?

(3,932,160 bytes \* ( 8 bits/byte))/100E6 bits/second

(3,932,160 \* 8)/100000000 = 0.3145728 seconds

0.3145728 seconds time will take at a minimum for the frame to be sent over a 100 Mbit/s network.

**1.5** [4] <§1.6> Consider three different processors P1, P2, and P3 executing the same instruction set. P1 has a 3 GHz clock rate and a CPI of 1.5. P2 has a 2.5 GHz clock rate and a CPI of 1.0. P3 has a 4.0 GHz clock rate and a CPI of 2.2.

**a.** Which processor has the highest performance expressed in instructions per second?

P1 = 3\*10^9/1.5 = 2\*10^9 instructions per second

P2 = 2.5\*10^9/ 1.0 = 2.5\*10^9 instructions per second

P3 = 4\*10^9/ 2.2 = 1.82\*10^9 instructions per second

P2 has the highest performance expressed in instruction per second

**b.** If the processors each execute a program in 10 seconds, find the number of cycles and the number of instructions.

Clock rate\* time

P1 = 3\*10^9\*10 = 3\*10^10 cycles

P2 = 2.5\*10^10 cycles

P3 = 4 \* 10^10 cycles

Number of instructions:

P1 = clock rate \* time /CPI = 3 \* 10^9 \* 10/ 1.5 = 2\*10^10 instructions

P2 = 2.5 \* 10^9 \* 10 / 1.0 = 2.5 \* 10^10 instructions

P3 = 4\* 10^9 \* 10 / 2.2 = 1.82\* 10^10 instructions

**c.** We are trying to reduce the execution time by 30% but this leads to an increase of 20% in the CPI. What clock rate should we have to get this time reduction?

Execution time = Number of instructions \* CPI / clock rate

Reduce execution time by 30% and cpi increase by 20%

**New clock rate = clock rate \* 1.2 / 0.7 = 1.71 \* clock rate**

**1.8** The Pentium 4 Prescott processor, released in 2004, had a clock rate of 3.6 GHz and voltage of 1.25 V. Assume that, on average, it consumed 10 W of static power and 90 W of dynamic power. The Core i5 Ivy Bridge, released in 2012, had a clock rate of 3.4 GHz and voltage of 0.9 V. Assume that, on average, it consumed 30 W of static power and 40 W of dynamic power.

**1.8.1** [5] <§1.7> For each processor find the average capacitive loads.

For Pentium 4 Prescott, dynamic power

=> 90 = C \* (1.25)^2 \* (3.6 \* 10^9)

=> C = 16 \* 10^-9 Farads

For Ivy Bridge, dynamic power

=> 40 = C \* (0.9)^2 \* (3.4 \* 10^9)

=> C = 14.524 \* 10^-9 Farads

**1.8.2** [5] <§1.7> Find the percentage of the total dissipated power comprised by static power and the ratio of static power to dynamic power for each technology.

Percentage of total dissipated power comprised of static power

= [(Static power) / (Static power + Dynamic power)] \* 100

For Pentium 4 Prescott, percentage

= [(10) / (10 + 90)] \* 100

= 10%

For Ivy Bridge, percentage

= [(30) / (30 + 40)] \* 100

= 42.86%

Ratio is a simple fraction, given by

= (Static power) / (Dynamic power)

For Pentium 4 prescott, ratio

= 10 / 90

= 0.11

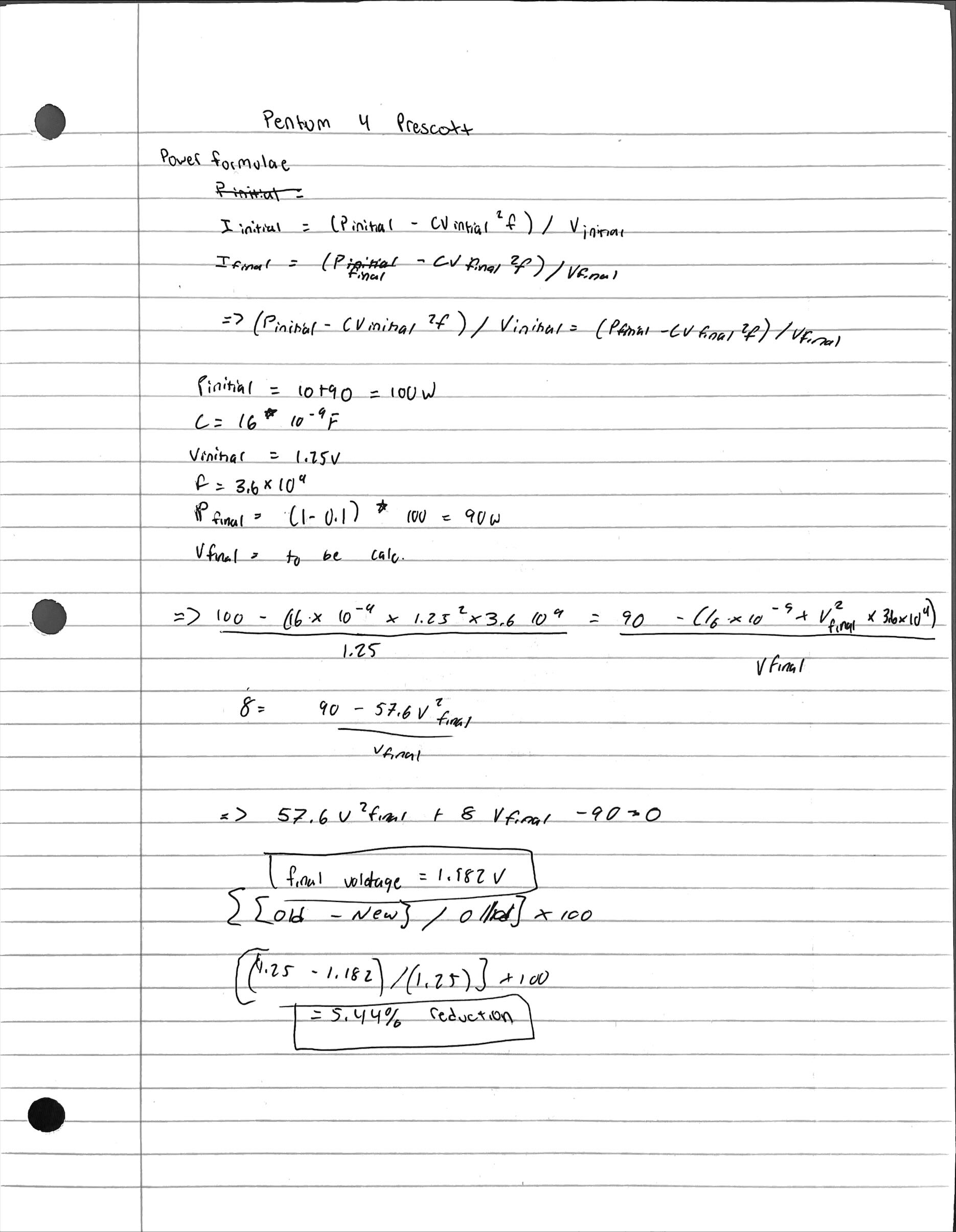
For Ivy Bridge, ratio

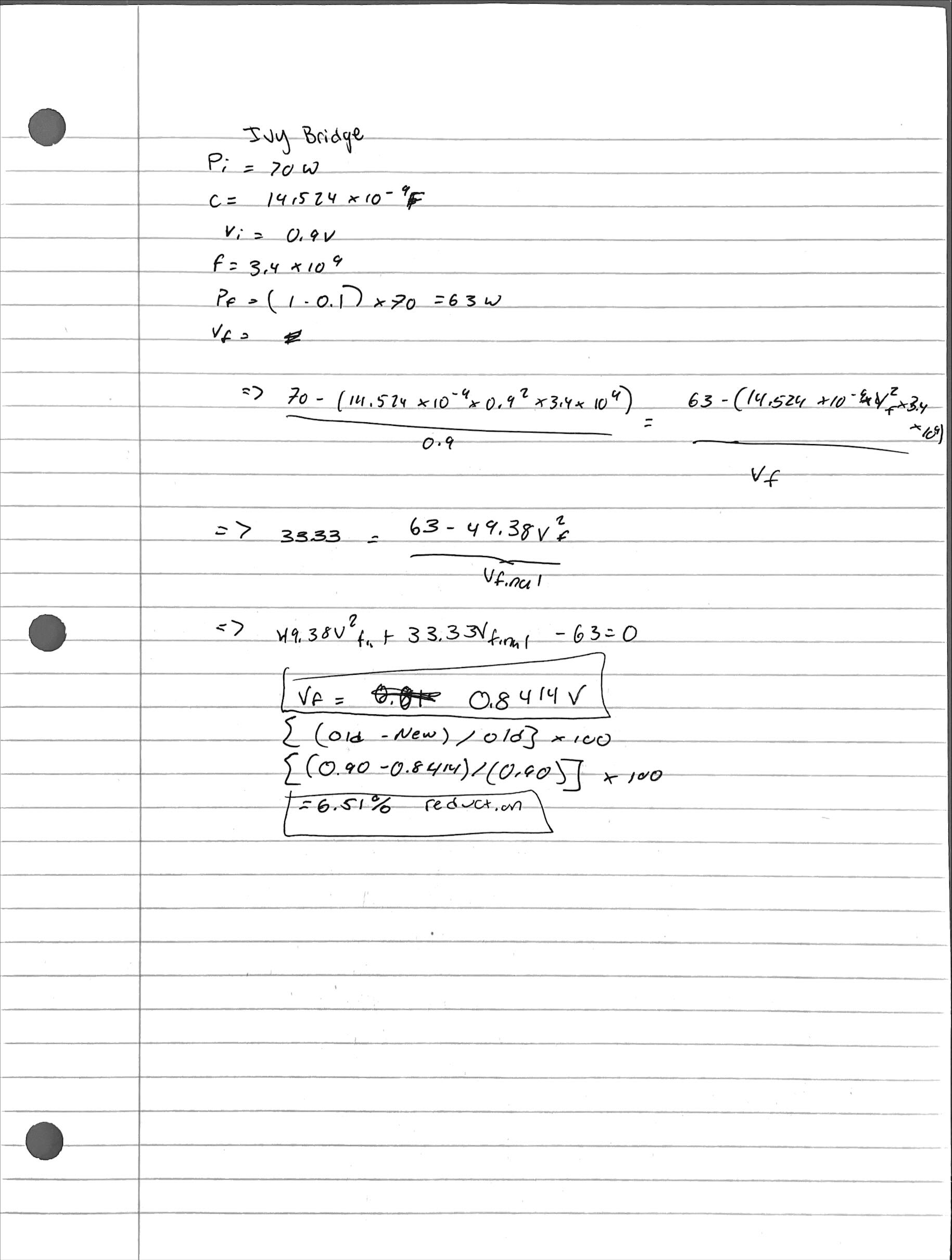
= 30 / 40

= 0.75

**1.8.3** [15] <§1.7> If the total dissipated power is to be reduced by 10%, how much should the voltage be reduced to maintain the same leakage current? Note: power is defined as the product of voltage and current.

Answer on written pictures below.





**1.11** The results of the SPEC CPU2006 bzip2 benchmark running on an AMD Barcelona has an instruction count of 2.389E12, an execution time of 750 s, and a reference time of 9650 s.

**1.11.1** [5] <§§1.6, 1.9> Find the CPI if the clock cycle time is 0.333 ns.

= (Total cycles) / (Total instructions)

= [(Execution time in seconds) / (Cycle time in seconds)] / [Total instructions]

= [(750) / (0.333 \* 10-9)] / [2.389 \* 1012]

= 0.9428

**1.11.2** [5] <§1.9> Find the SPECratio.

= (Reference time) / (execution time)

= 9650/750

= 12.87s

**1.11.3** [5] <§§1.6, 1.9> Find the increase in CPU time if the number of instructions of the benchmark is increased by 10% without affecting the CPI.

CPU time (in seconds)

= (Total number of instructions) \* (CPI)/clock rate

= 825 seconds

Thus, increase in time

= 825 - 750

**= 75 seconds or 10%**

**1.11.4** [5] <§§1.6, 1.9> Find the increase in CPU time if the number of instructions of the benchmark is increased by 10% and the CPI is increased by 5%.

CPU time (in seconds)

= (Total number of instructions) \* (CPI)/clock rate

Total number of instructions = 110/100 \* number of instructions = 1.1 \* number of instructions

CPI = 105/100 \* CPI = 1.05 \* CPI

New execution time = ( 1.05 \* CPI \* Instruction Count) / Clock Cycle time

= 866.25s

Thus, increase in time

= 866.25 - 750

**= 116.25 seconds**

**1.11.5** [5] <§§1.6, 1.9> Find the change in the SPECratio for this change.

= 9650s / 866.25s

= 11.14s

Change in spec ratio Is 12.87 - 11.14 = 1.73 or 13.44%

**1.11.6** [10] <§1.6> Suppose that we are developing a new version of the AMD Barcelona processor with a 4 GHz clock rate. We have added some additional instructions to the instruction set in such a way that the number of instructions has been reduced by 15%. The execution time is reduced to 700 s and the new

SPECratio is 13.7. Find the new CPI.

2.389 x 10^12 - 2.389 x 10^12 \* (15/100) = 2.03\*10^12

CPU execution time = CPI \* Number of instructions / Clock Rate

700 = CPI \* 2.03 \* 10^12 / 4 \* 10^9

**CPI = 1.37**

**1.11.7** [10] <§1.6> Th is CPI value is larger than obtained in 1.11.1 as the clock rate was increased from 3 GHz to 4 GHz. Determine whether the increase in the CPI is similar to that of the clock rate. If they are dissimilar, why?

Change in CPI = (1.37 – 0.94) / 0.94 = 0.43

Clock rate change = (4.0 \* 10 ^9 – 3\* 10^9) / 3\* 10^9 = 0.3333

They are similar since instruction count reduced

**1.11.8** [5] <§1.6> By how much has the CPU time been reduced?

(750 – 700 / 750) \* 100 = 6.66%

**1.11.9** [10] <§1.6> For a second benchmark, libquantum, assume an execution time of 960 ns, CPI of 1.61, and clock rate of 3 GHz. If the execution time is reduced by an additional 10% without affecting to the CPI and with a clock rate of 4 GHz, determine the number of instructions.

New execution time = actual execution time - 10% of initial execution time

= 960 – 96 = 864

Number of instructions 864 \* 4 \* 10 ^9 / 1.61

= 3456 \* 10^9/1.61

=2146.58\*10^9 = 2.15 \* 10 ^ 11

The number of instructions needed is 2.15 \* 10 ^ 11

**1.11.10** [10] <§1.6> Determine the clock rate required to give a further 10% reduction in CPU time while maintaining the number of instructions and with the CPI unchanged.

864 – 864 \* 10/100 = 777.6

New Rate of Clock = 2146 \* 10 ^9 \* 1.61 / 864 = 3.33GHz

Required clock rate is 3.33 GHz for 10 percent reduction

**1.11.1**1 [10] <§1.6> Determine the clock rate if the CPI is reduced by 15% and the CPU time by 20% while the number of instructions is unchanged.

Reduced execution time 960 -0.2 \* 960 = 768 seconds

Reduced CPI = 1.61-0.15 \* 1.62 = 1.37

Clock Rate = 2146 \* 10^9 \* 1.37/768

= 3.28 \* 10^9

= 3.82 GHz

Clock rate is 3.82 GHz