Question 1

The way we solve this is using Amdahl's Law. The formula for Amdahl's Law is:

$$S = \frac{1}{(1-\alpha) + \frac{\alpha}{k}}$$

Where S is the speedup, α is the fraction of the program that can be parallelized, and k is the perfomance uplift.

In this situation, our α is 0.5, since half of the processing time is spent on floating-point instructions. Since the enhancement has allowed the machine to run floating-point instructions five times faster, our k is 5. Putting all this into our equation, we get

$$\begin{aligned} S &= \frac{1}{(1-0.5) + \frac{0.5}{5}} \\ &= \frac{1}{0.5 + 0.1} \\ &= \frac{1}{0.6} \\ &\approx 1.67 \text{ .} \end{aligned}$$

Thus, our overall speedup is 1.67.

Question 2

a)

Here we need to figure out which one of the optimizations has the greater effect on the performance. If we only make the divide operation 3 times faster, we have an overall speedup of

$$\begin{aligned} S &= \frac{1}{(1-0.2) + \frac{0.2}{3}} \\ &= \frac{1}{0.8 + \frac{1}{15}} \\ &\approx 1.15 \text{ times.} \end{aligned}$$

If we only make the multiple operation 8 times faster, we have an overall speedup of

$$\begin{aligned} S &= \frac{1}{(1-0.6) + \frac{0.6}{8}} \\ &= \frac{1}{0.4 + \frac{3}{40}} \\ &\approx 2.11 \text{ times.} \end{aligned}$$

Therefore, we cannot meet management's goal of achieving a 5 times performance uplift by only making either the divide or multiply operation faster, as the maximum speedup we can achieve is 2.11 times with only the multiply operation being 8 times faster.

b)

Amdahl's law works by diving the old time by the new time to get the speedup, which makes intuitive sense. If the old time is 1, the new time can be found by removing the parts that are sped up and adding the speed adjusted parts back in. Thus we can generalize Amdahl's law to multiple speedups:

$$S = \frac{1}{(1-\alpha-\beta) + \frac{\alpha}{k_1} + \frac{\beta}{k_2}}$$

Where α and β are the fractions of the program that can be parallelized, and k_1 and k_2 are their respective perfomance uplifts. Now we can use this to find the overall speedup if we make both improvements.

$$\begin{aligned} S &= \frac{1}{(1-0.2-0.6) + \frac{0.2}{3} + \frac{0.6}{8}} \\ &= \frac{1}{0.2 + \frac{1}{15} + \frac{3}{40}} \\ &\approx 2.93 \, . \end{aligned}$$

Now, our speedup relative to the old machine is 2.93 times.

Textbook Problems:

Digital Design, 2nd Ed, by Frank Vahid, Wiley publication, 2010

1.8

Binary to decimal:

(a) $100 = 2^2 = 4$

(b) $1011 = 2^3 + 2^1 + 2^0 = 8 + 2 + 1 = 11$

(c) $0000000000001 = 1$

(d) $111111 = 2^0 + 2^1 + 2^2 + 2^3 + 2^4 + 2^5 = 1 + 2 + 4 + 8 + 16 + 32 = 63$

(e) $101010 = 2^1 + 2^3 + 2^5 = 2 + 8 + 32 = 42$

1.15

Decimal to binary using divide by 2:

(a) $19 \div 2 = 9$ remainder 1
$9 \div 2 = 4$ remainder 1
$4 \div 2 = 2$ remainder 0
$2 \div 2 = 1$ remainder 0
$1 \div 2 = 0$ remainder 1
$19 = 10011$

(b) $30 \div 2 = 15$ remainder 0
$15 \div 2 = 7$ remainder 1
$7 \div 2 = 3$ remainder 1
$3 \div 2 = 1$ remainder 1
$1 \div 2 = 0$ remainder 1
$30 = 11110$

(c) $64 \div 2 = 32$ remainder 0
$32 \div 2 = 16$ remainder 0
$16 \div 2 = 8$ remainder 0
$8 \div 2 = 4$ remainder 0
$4 \div 2 = 2$ remainder 0
$2 \div 2 = 1$ remainder 0
$1 \div 2 = 0$ remainder 1
$64 = 1000000$

(d) $128 \div 2 = 64$ remainder 0
$64 \div 2 = 32$ remainder 0
$32 \div 2 = 16$ remainder 0
$16 \div 2 = 8$ remainder 0

$8 \div 2 = 4$ remainder 0
$4 \div 2 = 2$ remainder 0
$2 \div 2 = 1$ remainder 0
$1 \div 2 = 0$ remainder 1
$128 = 10000000$

1.18

Binary to hexadecimal:

(a) $11110000 = F0$

(b) $11111111 = FF$

(c) $01011010 = 5A$

(d) $1001101101101 = 136D$

1.22

Hexadecimal to binary:

(a) $4F5E = 0100111101011110$

(b) $3FAD = 0011111110101101$

(c) $3E2A = 0011111000101010$

(d) $DEED = 1101111011101101$

1.32

We want to minimize the amount of custom ditital circuitry while maintaining at least 40 transactions per second. With only microprocessors, one transaction consisting of subtasks A, B, and C takes 0.05 s + 0.02 s + 0.02 s = 0.09 s. Thus, the maximum number of transactions per second is $\frac{1}{0.09} \approx 11.11$ transactions per second. Switching out the microprocessor for custom digital circuitry on part A gives us the largest upfront improvement from 50 ms to 1 ms. Now the maximum number of transactions per second is $\frac{1}{0.001+0.02+0.02} \approx 24.39$ transactions per second. Replacing the microprocessor with custom digital circuitry on part C gives us the next largest improvement from 20 ms to 1 ms. Now the maximum number of transactions per second is $\frac{1}{0.001+0.02+0.001} \approx 45.45$ transactions per second. Thus, the best way to achieve the goal of at least 40 transactions per second is to replace the microprocessor with custom digital circuitry on parts A and C.

2.12

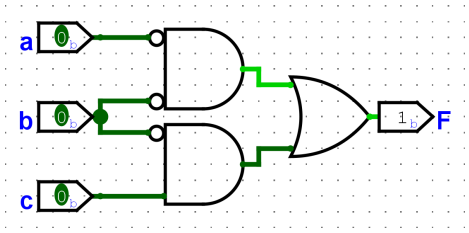
Evaluate $F = (a\ AND\ b)\ OR\ c\ OR\ d$ for the following values of a, b, c, and d:

1. $a = 1$, $b = 1$, $c = 1$, $d = 0$
   $F = 1 * 1 + 1 + 0 = 1$
2. $a = 0$, $b = 1$, $c = 1$, $d = 0$
   $F = 0 * 1 + 1 + 0 = 1$
3. $a = 1$, $b = 1$, $c = 0$, $d = 0$
   $F = 1 * 1 + 0 + 0 = 1$
4. $a = 1$, $b = 0$, $c = 0$, $d = 0$
   $F = 1 * 0 + 0 + 0 = 0$

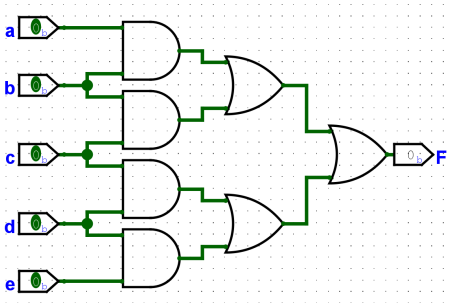
2.18

Convert to gate level circuits:

(a) $F = a'b' + b'c$



(b) $F = ab + bc + cd + de$



(c) $F = ((ab)' + (c)) + (d + ef)'$

2.24

For the function $F = a'd' + a'c + b'cd' + cd$:

(a) List all the variables.

(b) List all the literals.

(c) List all the product terms.

2.28

Convert to sum of products form: $F = a'b(c + d') + a(b' + c) + a(b + d)c$

2.33

$G = ab' + b + a'c$

2.39

$F = a'b'c' + a'bc' + ab'c' + ab'c + abc'$

2.52

Determine if F and G are equivalent using (a) algebraic manipulation and (b) truth tables:

$$F = ab + cd$$
$$G = (1((ab)'(cd)')')'$$

2.58

2.66

2.70

2.71

2.75

3.2

3.4

3.10

3.12

3.21

3.30

3.40

3.43