The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Comp 411 Computer Organization
Spring 2012

Problem Set #4 Solutions

Problem 1. “May Your Carries Overflowith”

a) The most straightforward way to prove two Boolean equations are identical is to show they have the same truth-tables.

$$V = \overline{A_{n-1}B_{n-1}}S_{n-1} + A_{n-1}B_{n-1}\overline{S_{n-1}}$$

C_{in}	B_{n-1}	A_{n-1}	S_{n-1}	V
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	0

$$V = \text{XOR}(C_{out}, C_{in})$$

C_{in}	B_{n-1}	A_{n-1}	C_{out}	V
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	1	1
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	0

b) Both methods work identically.

Consider the case when a negative number is subtracted from a positive number, yielding a negative result. This is equivalent to adding two positive numbers. Both formulas give identical results:

A[n−1] = 0,B[n−1] = 0 ,S[n−1] = 1, Cout = 0, Cin = 1
V =001+001=XOR(0,1)=1

When a positive number is subtracted from a negative number and a positive number rEsults, it is the same as adding two negative numbers. Again, both methods give the same result.:

A[n−1] = 1, B[n−1] = 1, S[n−1] = 0, Cout = 1, Cin = 0
V =110+110=XOR(1,0)=1

c) Only C is needed. In the case that the MSB adder generates a Cout, the value is clearly too large to be represented in the finite byte stream.

d) The addu instruction can certainly be used for signed 2’s-complement arithmetic. The user must simply be aware, that there will be no indication if an operation’s results are not valid.

Problem 2. "Bits of Floating-Point"

a)

$$2007 = 11111010111_2$$
$$= 1.1111010111_2 \times 2^{10}$$

Sign = 0

Exponent = 10
E = 10 + 127 = 137 = $1000\,1001_2$

Significand = $111\,1010\,1110\,0000\,0000\,0000_2$

S	E	F
1	100 0100 1	111 1010 1110 0000 0000 0000

0x44FAE000

b)

S	E	F
1	100 0101 0	111 1010 1110 0000 0000 0000

0xC57AE000

c)

S	E	F
1	011 1011 1	000 0000 0000 0000 0000 0000

0xBB800000

d)

S	E	F
0	100 1011 0	111 1111 1111 1111 1111 1111

0x4B7FFFFF

e)

S	E	F
0	100 1011 1	000 0000 0000 0000 0000 0000

0x4B800000

f)

S	E	F
0	011 1110 1	000 0000 0000 0000 0000 0000

Sign = 0

E = $0111\,1101_2$ = 125
Exponent = 125 - 127 = -2

Significand = $000\,0000\,0000\,0000\,0000\,0000_2$

$$1.0 \times 2^{-2} = 0.01_2$$
$$= 0.25$$

g)

S	E	F
1	011 1110 1	100 0000 0000 0000 0000 0000

$0.011_2 = 0.375$

h)

S	E	F
0	100 0110 1	000 0000 0000 0000 0000 0000

$1.0_2 \times 2^{14} = 100\,0000\,0000\,0000_2 = 16384$

i)

S	E	F
1	100 0110 0	111 1111 1111 1100 0000 0000

$-1.111\,1111\,1111_2 \times 2^{13} = -11\,1111\,1111\,1100 = -16380$

j)

S	E	F
0	100 0110 1	000 0000 0000 0010 0000 0000

$1.00\,0000\,0000\,0001_2 \times 2^{14} = 100\,0000\,0000\,0001 = 16385$

Problem 3. "Floating-Point Arithmetic"

For this problem, the values of x and y are shown below. Throughout the problem, numbers in red are outside of single-precision accuracy.

x:

S	E	F
0	011 1100 1	000 0000 0000 0000 0000 0000

1mm $1.0_2 \times 2^{-6} = 0.00\,0001_2 = 0.015625$

y:

S	E	F
0	100 1000 1	000 0000 0000 0000 0000 0000

$1.0_2 \times 2^{18} = 100\,0000\,0000\,0000\,0000_2 = 262144$

a) $x + y$

Denormalize and add:

$$\begin{aligned} x &= 0.000\,0000\,0000\,0000\,0000\,0000\,1_2 \times 2^{18} \text{ (the 1 is too small to be represented!)} \\ &= 0.000\,0000\,0000\,0000\,0000\,0000_2 \times 2^{18} \\ y &= 1.000\,0000\,0000\,0000\,0000\,0000_2 \times 2^{18} \\ x+y &= 1.000\,0000\,0000\,0000\,0000\,0000_2 \times 2^{18} \end{aligned}$$

Result is still normalized, so the final number is:

$$\begin{aligned} x+y &= 1.000\,0000\,0000\,0000\,0000\,0000_2 \times 2^{18} \\ &= y \end{aligned}$$

Single-precision representation does not have the accuracy to represent the operation $x + y$.

b) $x \times y$

Add exponents:

$$\begin{aligned} E &= 121 + 145 - 12 \text{ (see lecture notes for details)} \\ &= 139 \end{aligned}$$

Multiply significands:

$$\begin{aligned} F &= 1.0 \times 1.0 \\ &= 1.0 \end{aligned}$$

Final number:

S	E	F
0	100 0101 1	000 0000 0000 0000 0000 0000

$1.0_2 \times 2^{12} = 1\,0000\,0000\,0000_2 = 4096$

c) $x - y$

Denormalize and subtract:

$$\begin{aligned} x &= 0.000\,0000\,0000\,0000\,0000\,0000\,1_2 \times 2^{18} \text{ (the 1 is too small to be represented!)} \\ &= 0.000\,0000\,0000\,0000\,0000\,0000_2 \times 2^{18} \\ y &= 1.000\,0000\,0000\,0000\,0000\,0000_2 \times 2^{18} \\ x-y &= -1.000\,0000\,0000\,0000\,0000\,0000_2 \times 2^{18} \end{aligned}$$

Result is still normalized, so the final number is:

$$\begin{aligned} x-y &= -1.000\,0000\,0000\,0000\,0000\,0000_2 \times 2^{18} \\ &= -y \end{aligned}$$

Similar to part a), single-precision cannot accurately represent $x - y$.

Problem 5. The Real "Y2K"

a) The formula for the size of a ROM is 2^n where n is the number of inputs. T DIV3 table has 11 inputs, thus the ROM size is 2^{11} or 2048.

b) Since t_{cd} must be greater than t_h, the value should be > 1ns.

c) The minimum clock rate is specified as $tclk = t_{reg,pd} + t_{rom,pd} + t_{reg,s}$. This the minimum time to compute the circuit is $3 + 11 + 2 = 16$.

d) 11111010011:
S0→ S1→ S0→ S1→ S0→ S1→ S2→ S2→ S1→ S2→ S2→ S2
2003 is not divisible by 3

11111010001:
S0→ S1→ S0→ S1→ S1→ S1→ S2→ S2→ S1→ S2→ S1→ S0
2001 is divisible by 3

10111010100:
S0→ S1→ S2→ S2→ S2→ S2→ S1→ S0→ S0→ S1→ S2→ S1
1492 is not divisible by 3

11011110000:
S0→ S1→ S0→ S0→ S1→ S0→ S1→ S0→ S0→ S0→ S0→ S0
1776 is divisible by 3

11110101010:
S0→ S1→ S0→ S1→ S0→ S0→ S1→ S2→ S2→ S1→ S0→ S0
1962 is divisible by 3

e) The size of the ROM is $2^3 = 8$. The ROM's value table will appear as such:

current MSB	current LSB	input	next MSB	next LSB
0	0	0	0	1
0	0	1	1	0
0	1	0	0	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	x	x
1	1	1	x	x

f) The '3's divisibility rule for binary is this:

For a given binary number n
Create a *list1* consisting of the sequence of digits in the n
Remove every other digit from *list1* and add them to a new *list2*
Let *num1* be the number of '1's in *list1* and *sum2* be the number of '1's in *list2*
Subtract *sum1* from *sum2*. If the result is divisible by three, then n is divisible by three.

Since this rule works based on the relative positions of the '1's in the number, reversing the number does not change its divisibility.

g) Lee's state machine is capable of processing values of any number of digits, while the original design is limited to values of 11 digits. Thus Lee's design is immune to the Y2K problem.

Lee's machine will almost certainly be faster. Since his ROM is smaller, it will likely have a smaller propagation delay. This will allow for a smaller clock delay. His machine will still need eleven clock cycles to compute an 11 digit number, but each cycle will be shorter.

Lee's design is likely to be cheaper, as its ROM is much smaller and fewer registers are required.