Solutions - Midterm Exam

(February 14th @ 7:30 pm)

Presentation and clarity are very important! Show your procedure!

PROBLEM 1 (20 PTS)

- Compute the result of the following operations. The operands are signed fixed-point numbers. The result must be a signed fixed point number. For the division, use $x = 5$ fractional bits.

1.010001 + 1.011	1001.1101 − 1.011101	0.010101 + 01.11111
10.101 × 0.10011	1.011 × 1.0101	10.10010 ÷ 0.101

```
c8=1 c7=1 c6=0 c5=1 c4=0 c3=0 c2=0 c1=0 c0=0
  1 1.0 1 0 0 0 1 +
  1 1.0 1 1 0 0 0
  ---------------
  1 0.1 0 1 0 0 1
```

```
1 0 0 1.1 1 0 1 0 0 -      c10=0 c9=0 c8=0 c7=1 c6=1 c5=0 c4=0 c3=0 c2=0 c1=0 c0=0
1 1 1 1.0 1 1 1 0 1    ➡   1 0 0 1.1 1 0 1 0 0 +
                           0 0 0 0.1 0 0 0 1 1
                           -------------------
                           1 0 1 0.0 1 0 1 1 1
```

```
c9=0 c8=0 c7=1 c6=1 c5=1 c4=1 c3=1 c2=0 c1=0 c0=0
  0 0 0.0 1 0 1 0 1 +
  0 0 1.1 1 1 1 1 0
  -----------------
  0 1 0.0 1 0 0 1 1
```

```
10.101  x  ➡  01.011  x  ➡     1 0 0 1 1 x
0.10011       0.10011            1 0 1 1
                               ---------
                               1 0 0 1 1
                             1 0 0 1 1
                           0 0 0 0 0
                         1 0 0 1 1
                       -----------------
                       0 1 1 0 1 0 0 0 1
                               ⬇
                       0.1 1 0 1 0 0 0 1
                               ⬇
                       1.0 0 1 0 1 1 1 1
```

```
1.011  x  ➡  0.101  x  ➡    1 0 1 1 x
1.0101       0.1011            1 0 1
                            ---------
                              1 0 1 1
                            0 0 0 0
                          1 0 1 1
                        ---------------
                        0 0 1 1 0 1 1 1
                              ⬇
                        0.0 1 1 0 1 1 1
```

✓ $\frac{10.10010}{0.101}$: To unsigned (numerator) and then alignment, $a = 4$: $\frac{01.0111}{0.101} = \frac{01.0111}{00.1010} = \frac{010111}{001010} \equiv \frac{10111}{1010}$

```
          0001001001
1010 ) 1011100000
       1010
       ----
          1100
          1010
          ----
            10000
             1010
             ----
              110
```

Append $x = 5$ zeros: $\frac{10111\mathbf{00000}}{1010}$

Integer Division:

$Q = 1001001, R = 110$

$\rightarrow Qf = 10.01001 \; (x = 5)$

Final result (2C): $\frac{01.01110}{1.011} = 2C(010.01001) = 101.10111$

PROBLEM 2 (10 PTS)

- Represent these numbers in Fixed Point Arithmetic (signed numbers). Select the minimum number of bits in each case.

✓ -16.375
+16.375 = 010000.011
⇒ -16.375 = 101111.101

✓ 32.3125
+32.3125 = 0100000.0101

- Complete the table for the following fixed point formats (signed numbers): (6 pts.)

Integer bits	Fractional Bits	FX Format	Range	Resolution
6	3	[9 3]	$[-2^5, 2^5 - 2^{-3}]$	2^{-3}
8	5	[13 5]	$[-2^7, 2^7 - 2^{-5}]$	2^{-5}

PROBLEM 3 (40 PTS)

- Calculate the result (provide the 32-bit result) of the following operations with 32-bit floating point numbers. Truncate the results when required. When doing fixed-point division, use 4 fractional bits. Show your procedure.

✓ `C1500000 + 436A0000`	✓ `D0A90000 - CF480000`	✓ `80400000 × 7AB80000`	✓ `FBB80000 ÷ 49400000`

✓ X = `C1500000 + 436A0000`:

`C1500000`: `1100 0001 0101 0000 1000 0000 0000 0000`

$e + bias = 10000010 = 130 \rightarrow e = 130 - 127 = 3$ $\qquad Significand = 1.101$

`C1500000` = -1.101×2^3

`436A0000`: `0100 0011 0110 1010 0000 0000 0000 0000`

$e + bias = 10000110 = 134 \rightarrow e = 134 - 127 = 7$ $\qquad Significand = 1.110101$

`436A0000` = 1.110101×2^7

$X = -1.101 \times 2^3 + 1.110101 \times 2^7 = -\dfrac{1.101}{2^4} \times 2^7 + 1.110101 \times 2^7$

$X = (-0.0001101 + 1.110101) \times 2^7$

$$
\begin{array}{cccccccccc}
c_9=1 & c_8=1 & c_7=1 & c_6=1 & c_5=0 & c_4=0 & c_3=0 & c_2=1 & c_1=0 & c_0=0 \\
 & 0 & 1.1 & 1 & 0 & 1 & 0 & 1 & 0 & + \\
 & 1 & 1.1 & 1 & 1 & 0 & 0 & 1 & 1 & \\
\hline
 & 0 & 1.1 & 0 & 1 & 1 & 1 & 0 & 1 &
\end{array}
$$

To subtract these unsigned numbers, we first convert to 2C:

$R = 01.110101 - 0.0001101 = 01.110101 + 1.1110011$

The result in 2C is: $R = 01.1011101$

For floating point, we need to convert to sign-and-magnitude:

$\Rightarrow R(SM) = +1.1011101$

* You can also do unsigned subtraction: $X = (1.110101 - 0.0001101) \times 2^7$

$$
\begin{array}{ccccccccc}
b_8=0 & b_7=0 & b_6=0 & b_5=1 & b_4=1 & b_3=1 & b_2=0 & b_1=1 & b_0=0 \\
 & 1.1 & 1 & 0 & 1 & 0 & 1 & 0 & - \\
 & 0.0 & 0 & 0 & 1 & 1 & 0 & 1 & \\
\hline
 & 1.1 & 0 & 1 & 1 & 1 & 0 & 1 &
\end{array}
$$

$X = 1.1011101 \times 2^7$, $e + bias = 7 + 127 = 134 = 10000110$

X = `0100 0011 0101 1101 0000 0000 0000 0000` = `435D0000`

✓ X = `D0A90000 - CF480000`:

`D0A90000`: `1101 0000 1010 1001 0000 0000 0000 0000`

$e + bias = 10100001 = 161 \rightarrow e = 161 - 127 = 34$ $\qquad Significand = 1.0101001$

`D0A90000` = -1.0101001×2^{34}

`CF480000`: `1100 1111 0100 1000 0000 0000 0000 0000`

$e + bias = 10011110 = 158 \rightarrow e = 158 - 127 = 31$ $\qquad Significand = 1.1001$

`CF480000` = -1.1001×2^{31}

$X = -1.0101001 \times 2^{34} + 1.1001 \times 2^{31} = -1.0101001 \times 2^{34} + \dfrac{1.1001}{2^3} \times 2^{34}$

$X = -(1.0101001 - 0.0011001) \times 2^{34}$ (unsigned subtraction)

$X = -1.001 \times 2^{34}$, $e + bias = 34 + 127 = 161 = 10100001$

X = `1101 0000 1001 0000 0000 0000 0000 0000` = `D0900000`

$$
\begin{array}{ccccccccc}
b_8=0 & b_7=0 & b_6=0 & b_5=1 & b_4=0 & b_3=0 & b_2=0 & b_1=0 & b_0=0 \\
 & 1.0 & 1 & 0 & 1 & 0 & 0 & 1 & - \\
 & 0.0 & 0 & 1 & 1 & 0 & 0 & 1 & \\
\hline
 & 1.0 & 0 & 1 & 0 & 0 & 0 & 0 &
\end{array}
$$

✓ X = `80400000 × 7AB80000`:

`80400000`: `1000 0000 0100 0000 0000 0000 0000 0000`

$e + bias = 00000000 = 0 \rightarrow Denormal\ number \rightarrow e = -126$ $\qquad Significand = 0.1$

`80400000` = -0.1×2^{-126}

`7AB80000`: `0111 1010 1011 1000 0000 0000 0000 0000`

$e + bias = 11110101 = 245 \rightarrow e = 245 - 127 = 118$ $\qquad Significand = 1.0111$

`7AB80000` = 1.0111×2^{118}

$X = (-0.1 \times 2^{-126}) \times (1.0111 \times 2^{118}) = -0.10111 \times 2^{-8} = -1.0111 \times 2^{-9}$

$e + bias = -9 + 127 = 118 = 01110110$

X = `1011 1011 0011 1000 0000 0000 0000 0000` = `BB380000`

✓ X = `FBB80000 ÷ 49400000`:

`FBB80000`: `1111 1011 1011 1000 0000 0000 0000 0000`

$e + bias = 11110111 = 247 \rightarrow e = 247 - 127 = 120$ $\qquad Significand = 1.0111$

`FBB80000` = -1.0111×2^{120}

`49400000`: `0100 1001 0100 0000 0000 0000 0000 0000`

$e + bias = 10010010 = 146 \rightarrow e = 146 - 127 = 19$ $\qquad Significand = 1.1$

`49400000` = 1.1×2^{19}

$$X = -\frac{1.0111 \times 2^{120}}{1.1 \times 2^{19}} = -\frac{1.0111}{1.1} \times 2^{101}$$

```
          000001111
11000  ) 101110000
          11000
           101100
            11000
            101000
             11000
             100000
              11000
               1000
```

Alignment:

$$\frac{1.0111}{1.1} = \frac{1.0111}{1.1000} = \frac{10111}{11000}$$

Append $x = 4$ zeros: $\frac{101110000}{11000}$

Integer division

$$Q = 1111 \rightarrow Qf = 0.1111$$

Thus: $X = -0.1111 \times 2^{101} = -1.111 \times 2^{100}$

$e + bias = 100 + 127 = 227 = 11100011$

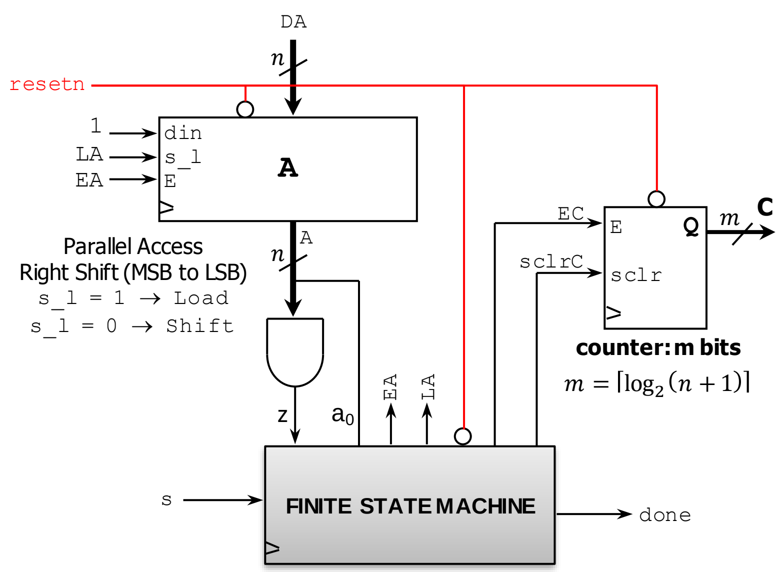
X = 1111 0001 1111 0000 0000 0000 0000 0000 = F1F00000

PROBLEM 4 (30 PTS)

- "Counting 0's" Circuit: It counts the number of bits in register A that has the value of '0'.
  The digital system is depicted below: FSM + Datapath. Example: For $n = 8$: if $A = 00110110$, then $C = 0100$.
  - ✓ m-bit counter: $sclr$. If $E = sclr = 1$, the count is initialized to zero. If $E = 1, sclr = 0$, the count is increased by 1.
  - ✓ Parallel access shift register: If $E = 1$: $s\_l = 1 \rightarrow$ Load, $s\_l = 0 \rightarrow$ Shift.

- Sketch the Finite State Machine diagram (in ASM form) given the algorithm (for $n = 8$, $m = 4$). (18 pts.)
  - ✓ The process begins when s is asserted, at this moment we capture DA on register A. Then the process starts by shifting A one bit at a time. The process is concluded when $A = 2^n - 1$. The signal $done$ is asserted when we finish counting.
  - ✓ Note: If $A = 2^n - 1 \rightarrow z = 1$, else $z = 0$. As A is being shifted, each time $a_0 = 0$, we need to increase the count C.

- Complete the timing diagram (next page) where $n = 8, m = 4$. (12 pts.)



ALGORITHM

```
C ← 0
while A ≠ 11...1 (2ⁿ-1)
    if a₀ = 0 then
       C ← C + 1
    end if
    right shift A
end while
```

Finite State Machine: