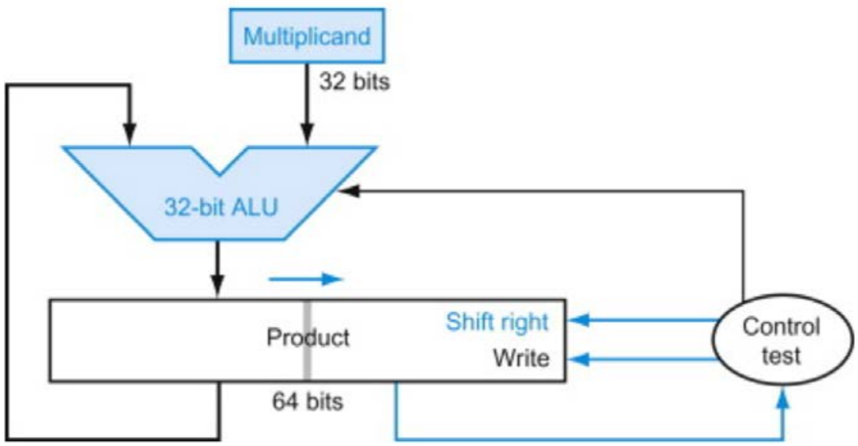
EGC442 Problem Set 9 Dr. Izadi

First Name:__________________________ Last Name: ____________________________

1) Refer to the refined multiplication hardware figure below



a) The refined multiplication hardware halves the width of the Multiplicand register from 64-bits to 32-bits.

- True
- False

b) The Multiplier register is removed and placed inside of the _____ register.

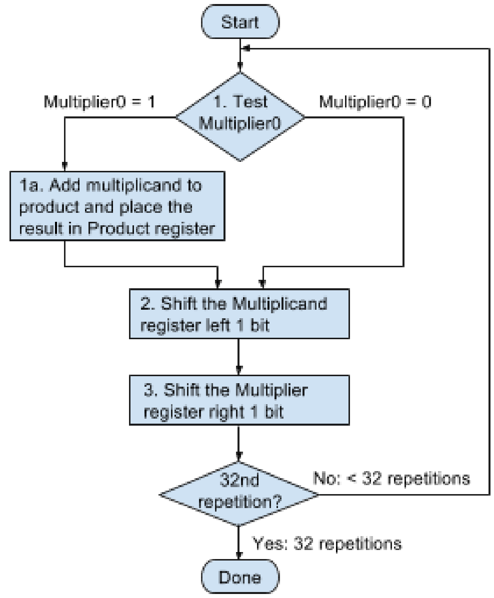
- Product
- Multiplicand

c) The ALU adds the 64-bit Product and 32-bit Multiplicand, and then stores the result into the Product register.

- True
- False

2) Consider the multiplication of $5_{10} \times 12_{10}$, or $0101_2 \times 1100_2$. Fill in the missing values for each of the steps labeled according to COD Figure 3.4 (The first multiplication algorithm …). A copy of the multiplication algorithm figure is shown below to the right.

Iteration	Step	Multiplier	Multiplicand	Product
0	Initial values	1100	0000 0101	0000 0000
1	1: 0 ⇒ No operation	1100	0000 0101	0000 0000
	2: Shift left Multiplicand	1100	0000 1010	0000 0000
	3: Shift right Multiplier	0110	0000 1010	0000 0000
2	(a)	0110	0000 1010	0000 0000
	2: Shift left Multiplicand	0110	(b)	0000 0000
	3: Shift right Multiplier	(c)		0000 0000
3	(d)			0001 0100
	2: Shift left Multiplicand		0010 1000	0001 0100
	3: Shift right Multiplier	0001	0010 1000	0001 0100
4	1a: 1 ⇒ Prod = Prod + Mcand	0001	0010 1000	(e)
	2: Shift left Multiplicand	0001	0101 0000	
	3: Shift right Multiplier	0000	0101 0000	



- **0001 0100**
- **1a: 1 ==> Prod = Prod + Mcand**
- **0011 1100**
- **0011**
- **1: 0 ==> No operation**

(a)

(b)

(c)

(d)

(e)

3)

a. The multiplication hardware supports signed multiplication.

- ○ True
- ○ False

b. The 32-bit registers, called Hi and Lo, combine to form a 64-bit product register.

- True
- False

c. The multiply (mult) instruction ignores overflow, while the multiply unsigned (multu) instruction detects overflow.

- True
- False

4)
a. A calculation that leads to a number being too large to represent is called ______.

- overflow
- underflow
- a fraction

b. Increasing the size of the ______ used to represent a floating-point number impacts the number's precision.

- fraction
- exponent

d. A ______ precision floating-point number is represented with two MIPS words.

- single
- double

5. Show the IEEE 754 binary representation of the number $+0.375_{ten}$ in single precision:

- $11_2 / 2^3$ **or** $\mathbf{0.011_2}$
- **125**
- **.1000 0000 0000 0000 0000 0000**
- $3 / 8$ **or** $3/2^3$
- $\mathbf{1.1_{two} \times 2^{-2}}$
- **0**
- $\mathbf{(-1)^0 \times (1 + .1000\ 0000\ 0000\ 0000\ 0000\ 0000) \times 2^{(125 - 127)}}$

Rewrite as a fraction

Rewrite as a binary number

Rewrite as normalized scientific notation

S = ?

Exponent = ?

Fraction = ?

IEEE 754 binary single precision representation

6. Show the IEEE 754 binary representation of the number -0.9375_{ten} in double precision:

- **1**
- **1022**
- **$(-1)^1 \times (1 + .1110\ 0000 \ldots 0000) \times 2^{(1022 - 1023)}$**
- $1111_{two} / 2^4$ **or 0.1111_{two}**
- **.1110 0000 … 0000**
- 15/16 **or** $15/2^4$
- **$1.111_{two} \times 2^{-1}$**

Rewrite as a fraction

Rewrite as a binary number

Rewrite as normalized scientific notation

S = ?

Exponent = ?

Fraction = ?

IEEE 754 binary double precision representation

7. Convert the single precision binary floating-point representation to decimal.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	0	0	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1 bit	8 bit								23 bit																						

8. Add the following numbers using the floating-point addition algorithm. Assume 4 bits of precision.

a. $1.010 \times 2^{-3} + 0.011 \times 2^{-3} = ?$

- ○ 1.101
- ○ 1.101×2^{-6}
- ○ 1.101×2^{-3}

b. $1.001 \times 2^{-4} + 1.000 \times 2^{-6} = ?$

- ○ 10.001×2^{-4}
- ○ 1.011×2^{-4}

c. $1.000 \times 2^{3} + 0.011 \times 2^{5} = ?$

- ○ 1.010×2^{4}
- ○ 0.101×2^{5}
- ○ 10.001×2^{5}

9. Multiply -14_{ten} and -0.25_{ten}, or $-1.110 \times 2^{3} \times -1.000 \times 2^{-2}$. Assume 4 bits of precision.

- **1.110000**
- **$1.110000_{two} \times 2^{1}$**
- **$1.1100_{two} \times 2^{1}$**
- **+**
- **3 + (-2) = 1**
- **2^{1}**
- **3.5_{ten}**

Adding the non-biased exponents of the operands

Multiply the significands:
$1.110 \times 1.000 = ?$

Product = $1.110000 \times ?$

Normalize the product

Round the product

Set the sign of the product: ? $1.1100_{two} \times 2^{1}$

$-14_{ten} \times -0.25_{ten} = ?$