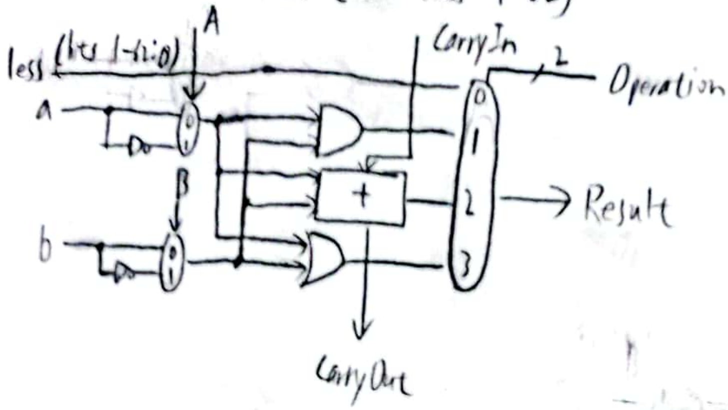
1. A: Ainvert B: Bnegate

| ALU Control | Function |
| --- | --- |
| 0001 | and |
| 0011 | or |
| 0010 | addu |
| 0100 | sltu |
| 1111 | nand |

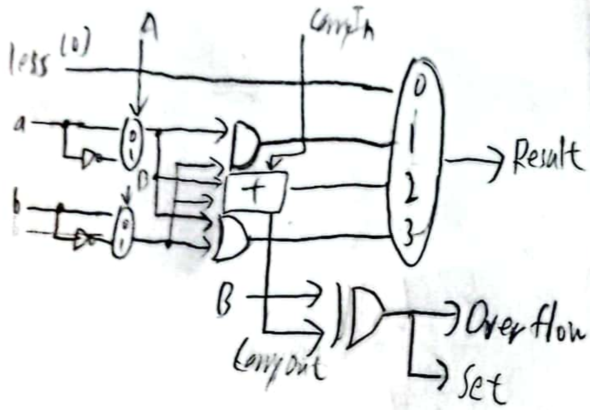
−1

?!

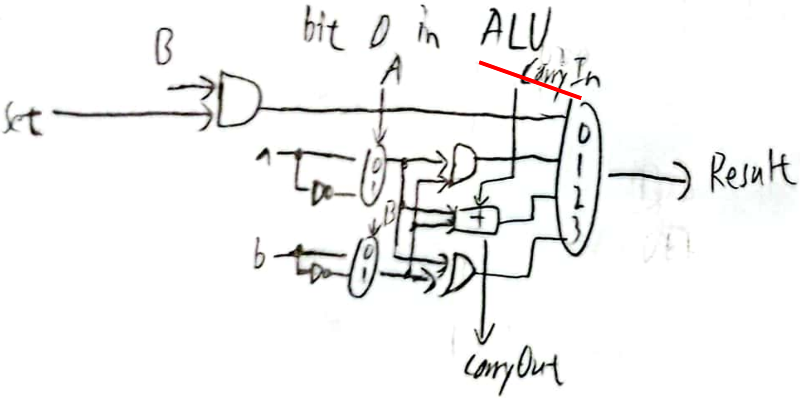
1-bit in ALU (for bits 1-62)

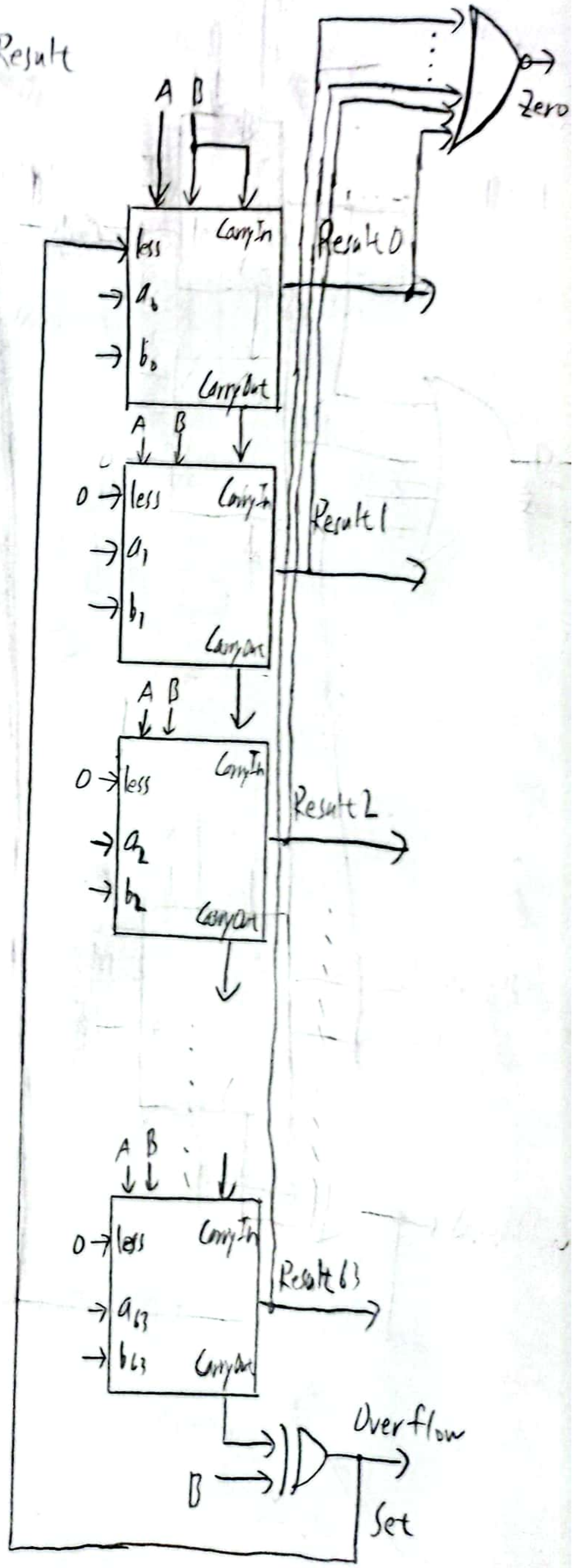



bit 63 in ALU




bit 0 in ALU



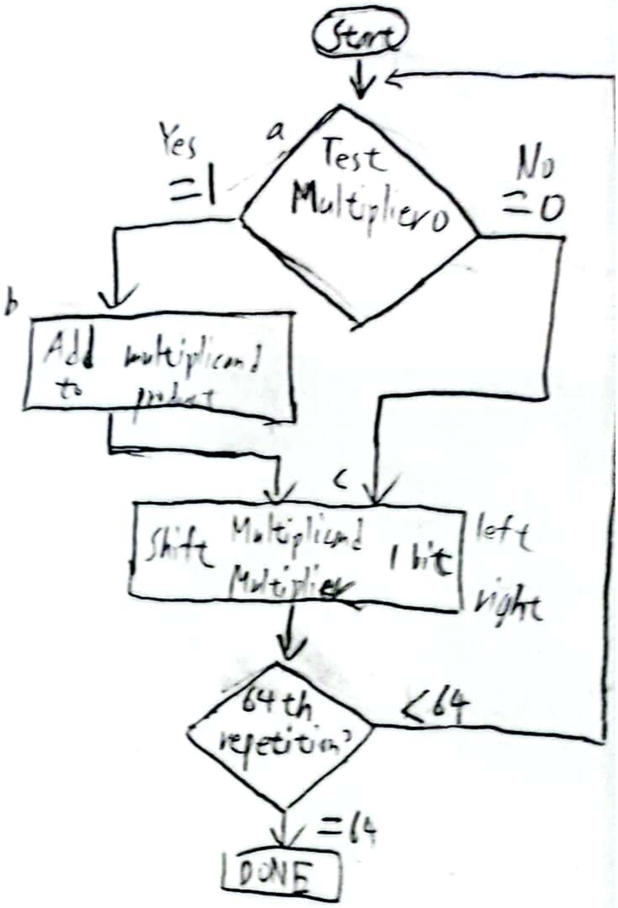



2. M=0111, N=1101

(a) -0

0111 × 1101 = 01011011 = $91_{decimal}$

| Product | Multiplier | Multiplicand | |
|---|---|---|---|
| 00000000 | 1101 | 00000111 | 1.a Yes |
| 00000111 | 0110 | 00001110 | 1.b,c; 2.a No |
| 00000111 | 0011 | 00011100 | 2.c; 3.a Yes |
| 00100011 | 0001 | 00111000 | 3.b,c; 4.a Yes |
| 01011011 | 0000 | 01110000 | 4.b,c; Done |

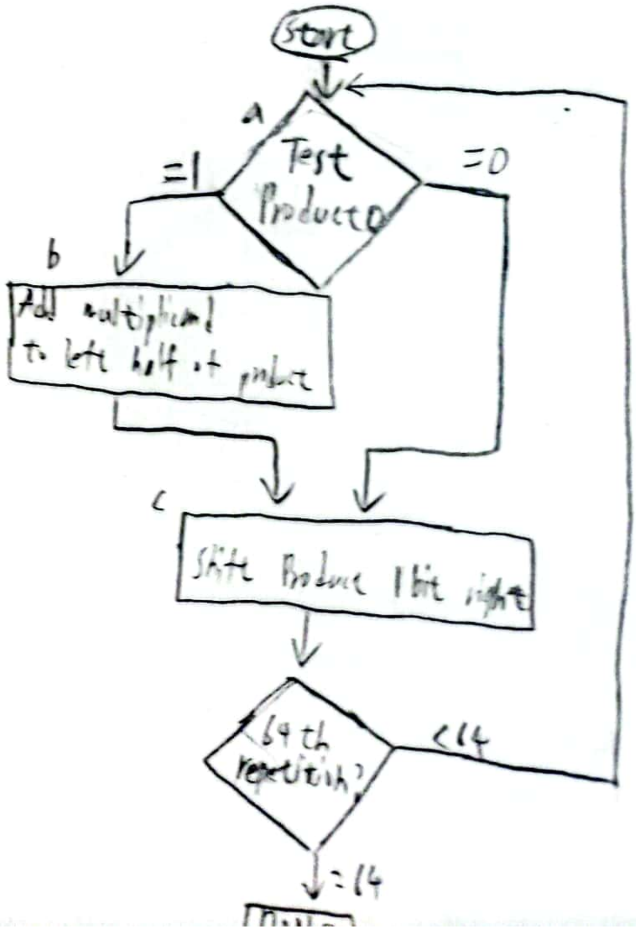



Reference: Lecture slide L3-1

(b)

0111×1101 = 01011011 = $91_{decimal}$

| Multiplicand | Product | |
|---|---|---|
| 0111 | 00001101 | 1.a Yes |
| | 01111101 | 1.b |
| 0111 | 00111110 | 1.c; 2.a No |
| 0111 | 00011111 | 2.c; 3.a Yes |
| | 10001111 | 3.b |
| 0111 | 01000111 | 3.c; 4.a Yes |
| | 10110111 | 4.b |
| 0111 | 01011011 | 4.c; Done |

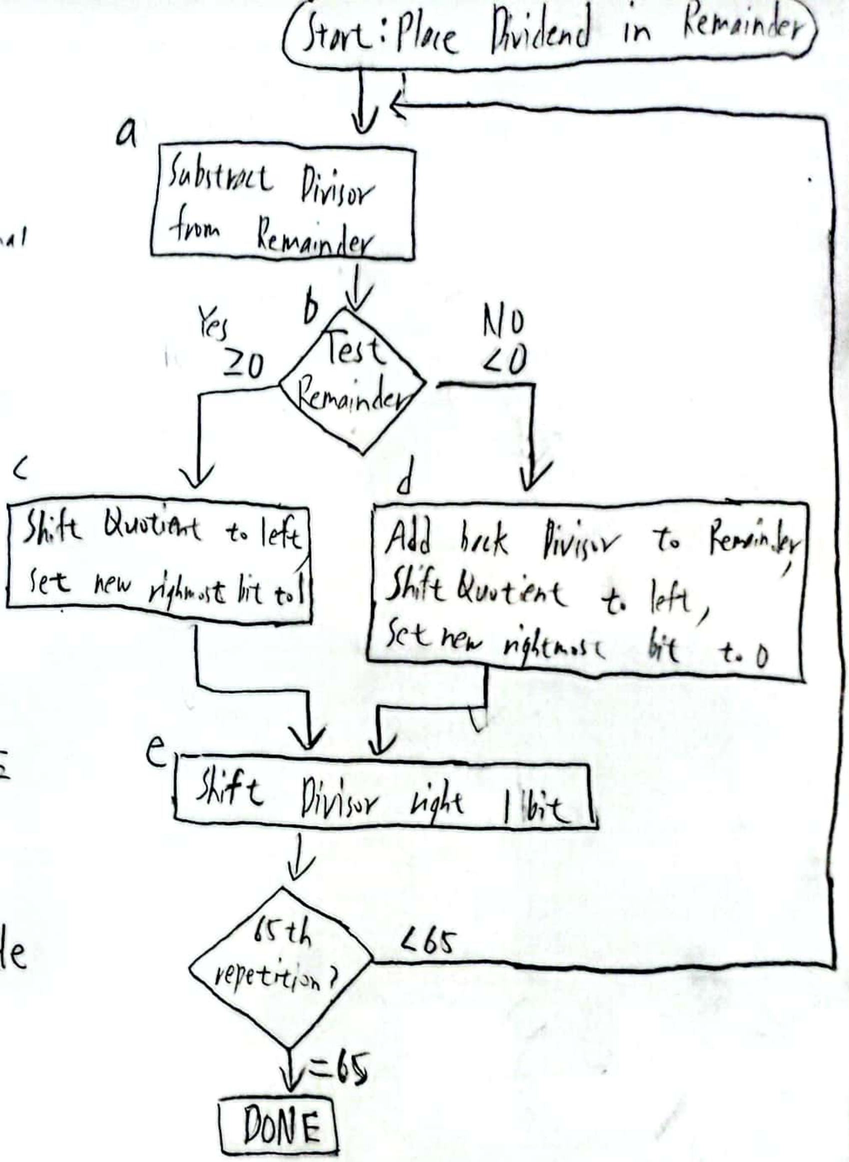


3, M=0111 , N=0110.

(a)

$0111 \div 0110 = 0001 \cdots 0001 = 1 \cdots 1_{decimal}$

| Quot. | Divisor | Remainder | |
|---|---|---|---|
| 0000 | 01100000 | 00000111 | |
| | | 10100111 | 1.a,b No |
| 0000 | 00110000 | 00000111 | 1.d,e |
| | | 11010111 | 2.a,b No |
| 0000 | 00011000 | 00000111 | 2.d,e |
| | | 11101111 | 3.a,b No |
| 0000 | 00001100 | 00000111 | 3.d,e |
| | | 11111011 | 4.a,b No |
| 0000 | 00000110 | 00000111 | 4.d,e |
| | | 00000001 | 5.a,b Yes |
| 0001 | 00000011 | 00000001 | 5.c,e ; DONE |



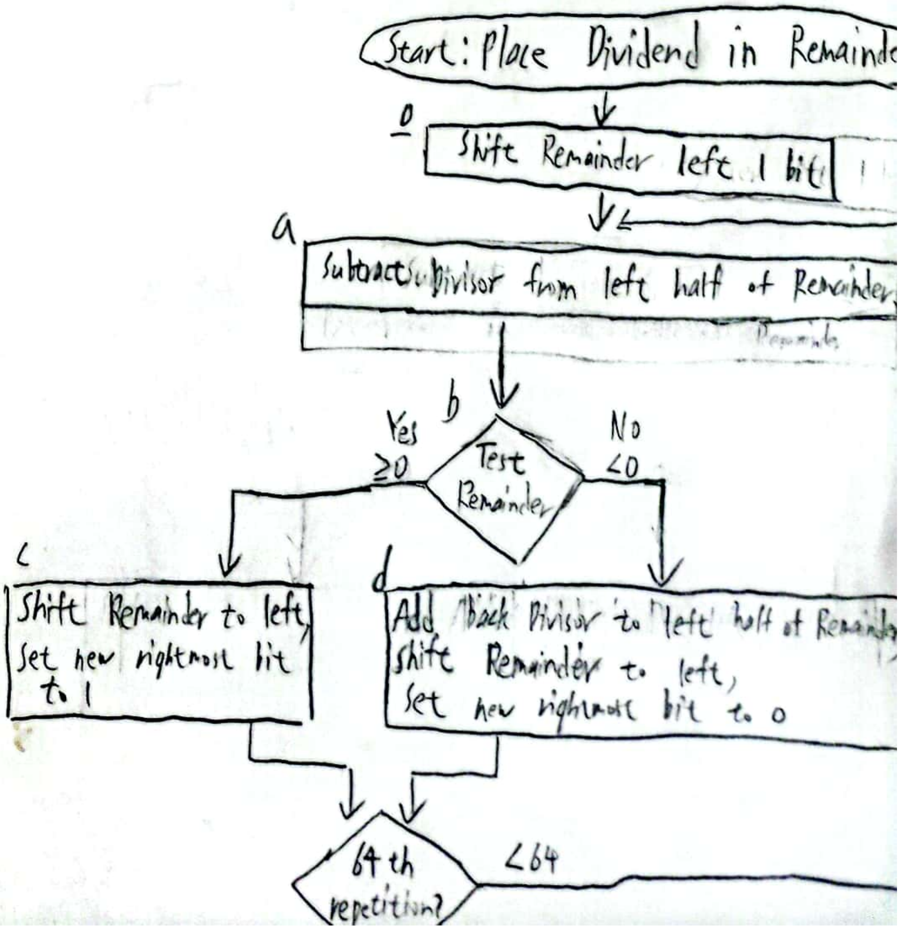

Reference: Lecture slide L3-1

(b)

$0111 \div 0110 = 0001 \cdots 0001 = 1 \cdots 1_{decimal}$

| Remainder | Divisor | |
|---|---|---|
| 00000111 | 0110 | |
| 00001110 | | 0 |
| 10101110 | | 1.a,b No |
| 00011100 | | 1.d |
| 11011100 | | 2.a,b No |
| 00111000 | | 2.d |
| 11011000 | | 3.a,b No |
| 01110000 | | 3.d |
| 00010000 | | 4.a,b Yes |
| 00100001 | | 4.c |
| 0001 0001 | | DONE |

↑ Remainder ↑ Quotient



5.


(a) $a0 = -1.0000_2 \times 2^{-2}$

(b) $a1 = -0.1111_2 \times 2^{-2}$

$a2 = -0.1110_2 \times 2^{-2}$

(c) $a0 - a1 = -0.0001_2 \times 2^{-2}$

$a1 - a2 = -0.0001_2 \times 2^{-2}$

The difference between a0 and a1 are the same as difference between a1 and a2 (and difference between any two consecutive "denormalized" numbers.

(d)

$0x5C = 0\,101\,1100 = 1.11_2 \times 2^2 = 4+2+1 = 7_{decimal}$

(e)

$U = 1\,101\,0111 = -1.0111_2 \times 2^2 = -5.75_{decimal}$

approximation error $= |(-5.75) - (-5.7)|$

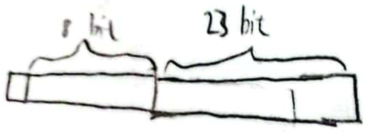
$= 0.05$ #

4.

$X = 0.3125$, $Y = -15.96875$

a)

Single-precision:




$X = \frac{5}{16} = \frac{5}{4} \times \frac{1}{4} = 1.01_2 \times 2^{-2} \Rightarrow 0\,\underbrace{01111101}_{125_{decimal}}\,\underbrace{0100\ldots}_{23\text{ bit}} = 0x3EA00000$ #

$Y = -(8+4+2+1+0.5+0.25+0.125+0.0625+0.03125)$

$= -1.11111111_2 \times 2^3 \Rightarrow 1\,\underbrace{10000010}_{130_{decimal}}\,\underbrace{11111111\,0000\ldots}_{23\text{ bit}} = 0xC17F8000$ #

b)

$1.01_2 \times 2^{-2} + -1.11111111_2 \times 2^3$ $(0.3125 + -15.96875)$

1) Align binary points

$= 0.0000101_2 \times 2^3 + -1.11111111_2 \times 2^3$

2) Add significands

$= -1.1111010 \times 2^3$

3) Normalize result and check for over/underflow (no change) (no over/underflow)

4) Round and renormalize if necessary

$= -1.11110101 \times 2^3 \equiv 1\,\underbrace{10000010}_{130_{decimal}}\,\underbrace{11110101000\ldots}_{23\text{ bit}} = 0xC17A8000$ #

$3+127=130$

4 (c) $X = 0.3125$, $Y = -15.96875$

$$1.01_2 \times 2^{-2} \times -1.11111111_2 \times 2^3$$

$$
\begin{array}{r}
1.11111111 \\
\times \quad 1.01 \\
\hline
111111111 \\
000000000 \\
+\,111111111 \quad \\
\hline
10.01111111011
\end{array}
$$

1) Add exponents

— $-2+3=1$

2) Multiply significands

— $1.01_2 \times 1.11111111_2 \Rightarrow 10.0111111011 \times 2^1$

3) Normalize result and check for over/underflow

$= 1.00111111011 \times 2^2$ with no over/underflow

4) Round and renormalize if necessary (no change)

5) Determine sign: +ve × -ve ⇒ -ve

$= -1.00111111011 \times 2^2 = |1|\underbrace{10000001}_{129\ \text{decimal}}|\overbrace{00111111011\,0000\ldots}^{23\ \text{bit}}| = 0x\,C09FB000$ #

$2+127=129$

6.

X -8

List out the order of types of $\boxed{\text{float numbers}}$:

unsigned comparison

-INF < -Normalized numbers < -Denormalized numbers < ±0 < +Denormalized numbers < +Normalized numbers < +INF.

unsigned comparison

If we interpret as unsigned, the order will become like this

|-INF > -Normalized numbers > -Denormalized numbers > -0| > +|INF > +Normalized numbers > +Denormalized numbers > +0|

(Based on the rules of unsigned comparison, we can actually see that the negative part reversed and switch to the bigger side.)

So, for X and Y having same signed bits, expression 3, 4 handle correctly.

And the expression 2 handle the case that X and Y are different signed. (+ must > -)

Last, leaving a special case that X, Y are both ±0, they are considered the same, so return 1. Yes, So expression 1, 2, 3, 4 handle all of the cases correctly.