**Floating Point Representation**

* What is the value of 14.87510 in binary with binary point?
  + 1410 is **11102**. 0.875 x 2 = **1**.75, 0.75 x 2 = **1**.5, and 0.5 x 2 = **1**.0. Thus 14.87510 is 1110.1112
* What is the value of 1100.10112 in decimal (not in scientific notation)?
  + (1 x 23) + (1 x 22) + (0 x 21) + (0 x 20) + (1 x 2-1) + (0 x 2-2) + (1 x 2-3) + (1 x2-4)

Which is equal to 8 + 4 + 0.5 + 0.125 + 0.0625 = 12.6875

* How to represent binary number with binary point 1100.10110100101102 in IEEE 754 format (32-bit single precision)?
  + 1100.1011010010110 = 1.1001011010010110 x 23. Thus 3 = e - 127 or e = 130 which is 100000102. The above number in IEEE 754 format is:
* MemWrite :
  + 1, if we want to write data to data memory
    - Only sw requires it.

0 10000010 1001011010010110. . . 0

* A 32-bit (single precision) in IEEE 754 format is shown below:

1 10000000 11100000000000000000000

What is above floating-point number in decimal?

* From the above number exponent is 128, thus the number is

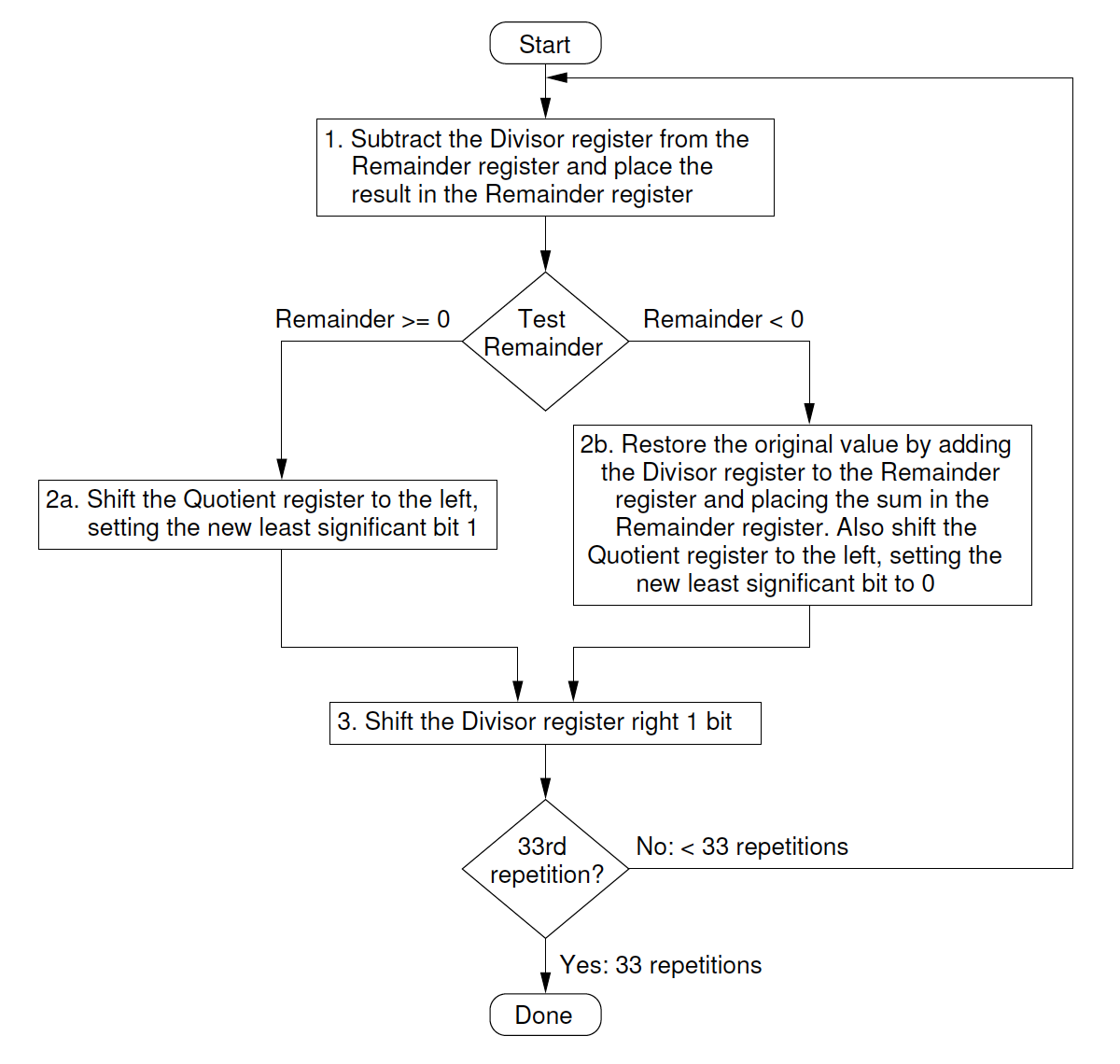
(-1)1 x (1 + 0.111) x 2128-127 = -1.111 x 21 = -11.11

Thus, the number is -3.75

* Consider the following two floating-point number a and b, in IEEE 754 format (single precision)

a = 0 10101010 11001100110011001100110

b = 0 10101011 00110011001100110011001



Which is larger and why?

* b is larger, since the exponent is larger.
* MemWrite:
  + 1, if we want to write data to data memory
  + Only sw requires to write data to data memory
* ALUSrc:
  + 0, if we want second operand of the ALU to come from the read data 2 of the register file
    - Add, sub, or, slt, beq
  + 1, if we want second operand of the ALU to come from sign-extended immediate field of an instruction
    - Lw and sw
* Jump:
  + 1, if the instruction is j
    - Add, sub, and, or, and slt
* Branch:
  + 1, if the instruction is beq
* MemRead:
  + 1, if we want to read data from memory
  + Only lw requires CPU to read data from data memory
* RegDst:
  + 1, if we want rd field to go to Write Register
    - Add, sub, and, or, and slt
  + 0, if we want rt field to go to Write Register
    - lw
  + beq, sw, and j do not write to register file (ignore)
* RegWrite:
  + 1, if we want to write data to register file
    - Add, sub, or, slt, lw
  + 0, if we do not want to write data to register file
    - Beq, sw, j

