Lab# 7 INTEGER MULTIPLICATION AND DIVISION

Instructor: I Putu Danu Raharja.

Objectives:
Learn how to perform integer multiplication and division operations in MIPS assembly language programs.

Method:

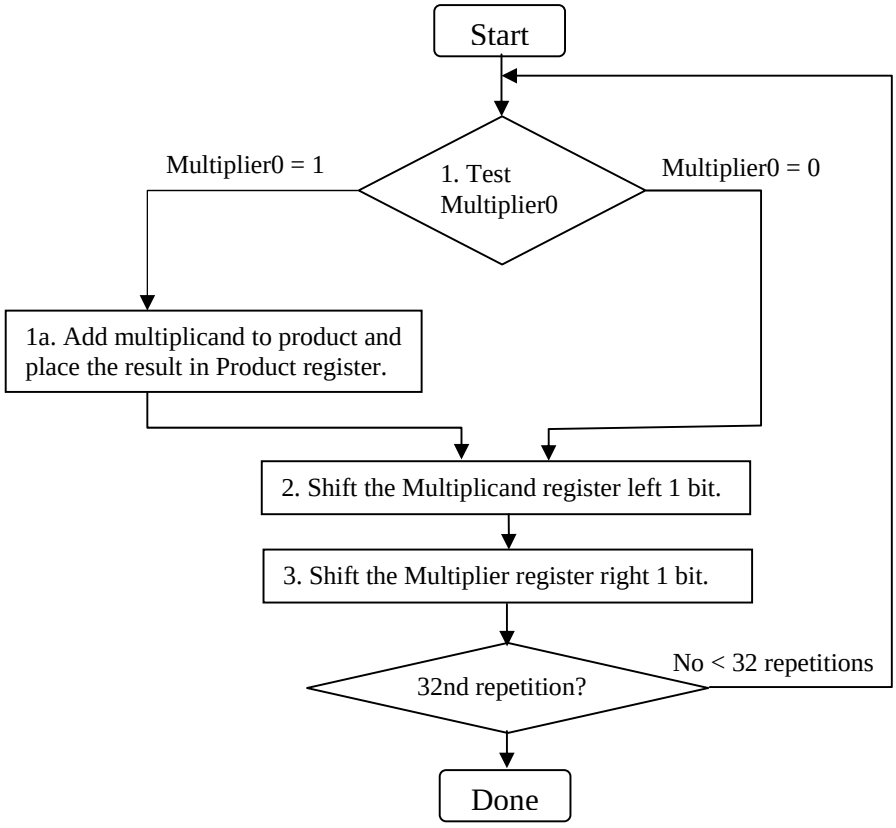
Translate an algorithm from pseudo-code into assembly language.

Preparation:

Read the chapter 3 of lecture textbook.

7.1 MULTIPLICATION

The following figure illustrates the process of an integer multiplication of two 32-bit registers to produce a value of 64-bit.



MIPS provides a separate pair of 32-bit registers to contain the 64-bit product, called Hi and Lo. To produce a properly signed or unsigned product, MIPS has two instructions:

a. multiply (**`mult`**)

b. multiply unsigned (**`multu`**)

To fetch the integer 32-bit products, the programmer uses the following instructions:

a. move from Lo (**`mflo`**)

b. move from Hi (**`mfhi`**)

Both MIPS multiply instructions ignore overflow, so it is up to the software to check to see if the product is too big to fit in 32 bits. There is no overflow if Hi is 0 for **`multu`** or the replicated sign of Lo for **`mult`**. The instruction move from Hi (**`mfhi`**) can be used to transfer Hi to a general-purpose register to test for overflow.

7.2 DIVISION

MIPS uses the 32-bit Hi and 32-bit Lo registers for divide. And after the divide instruction completes, the Hi register contains the remainder, and the Lo register contains the quotient.

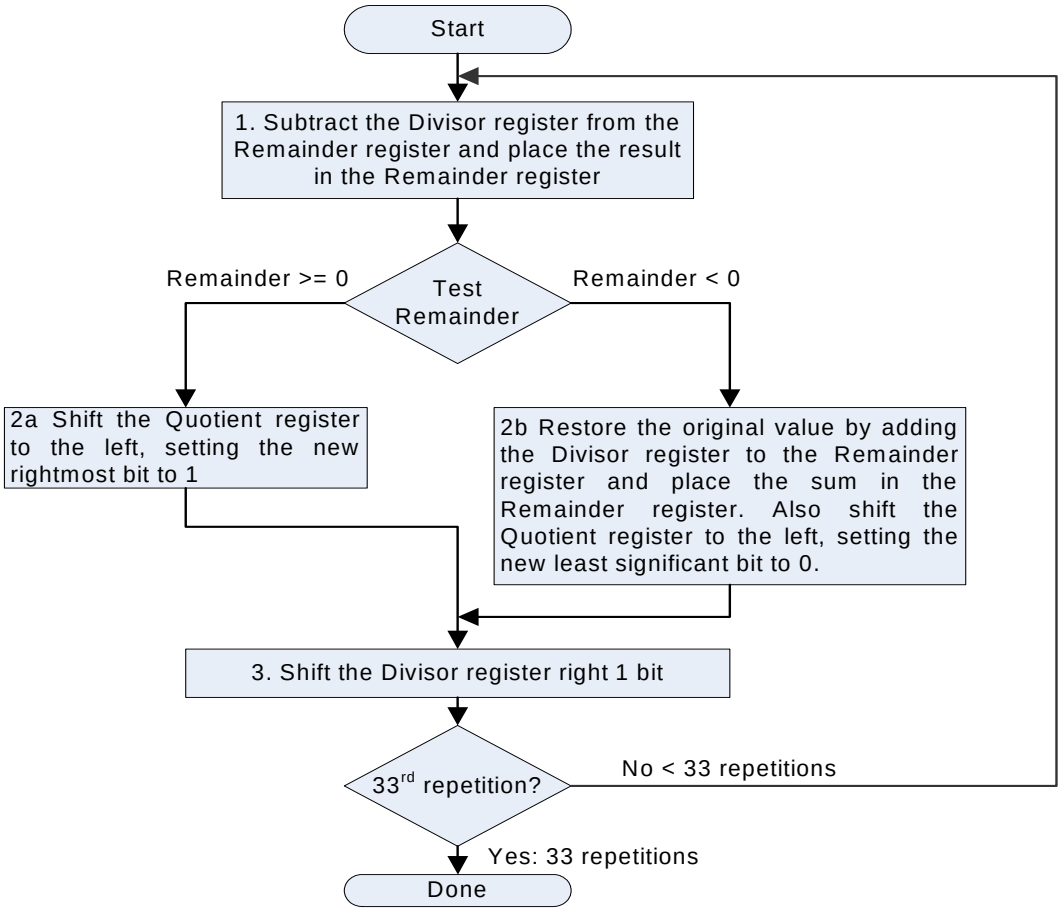
To handle both signed integers and unsigned integers, MIPS has two instructions:

a. divide (**`div`**),

b. divide unsigned (**`divu`**).

MIPS divide instructions ignore overflow, so software must determine if the quotient is too large. In addition to overflow, division can also result in an improper calculation: division by 0. MIPS software must check the divisor to discover division by 0 as well as overflow.

The following figure shows the process of an integer division.



7.3 EXERCISE:

1. Write a function to find the determinant of a two-by-two matrix. The address of the array is passed to the function in register **`$a0`** and the result is returned in **`$v0`**.
2. Implement the algorithm of multiplication mentioned in 7.1 in MIPS assembly language program.
3. Implement the algorithm of division mentioned in 7.2 in MIPS assembly language program.