Floating point arithmetic operations

Addition or subtraction

For arithmetic floating point addition or subtraction operations, we need to check if both exponents are equal or not. If they are not equal, we need to align the mantissa by shifting it to the right or the lift.

Example:

$.5372400*10^2$
$.1580000*10^{-1}$

In this example, the two number exponents are 2 and -1. Both are not equal, so we need to align either the first number mantissa or the second number mantissa by shifting to the right or left. The second number shifted to the right three positions. The mantissa part is stored in registers. When shifting to the left, the most significant digits are lost. Significant bits are lost when shifting to the right. In this example, the second number shifted to the right three positions.

$.5372400*10^2$
$.0001580*10^2$
____
$.5373980*10^2$

For addition or subtraction operations, two floating point numbers are stored in the AC and BR registers. The result is stored in the AC register after addition or subtraction operations. The addition or subtraction algorithm can be divided into four parts.

1. Check for zeros.
2. Align the mantissas.
3. Add or subtract the mantissas.
4. Normalize the result.

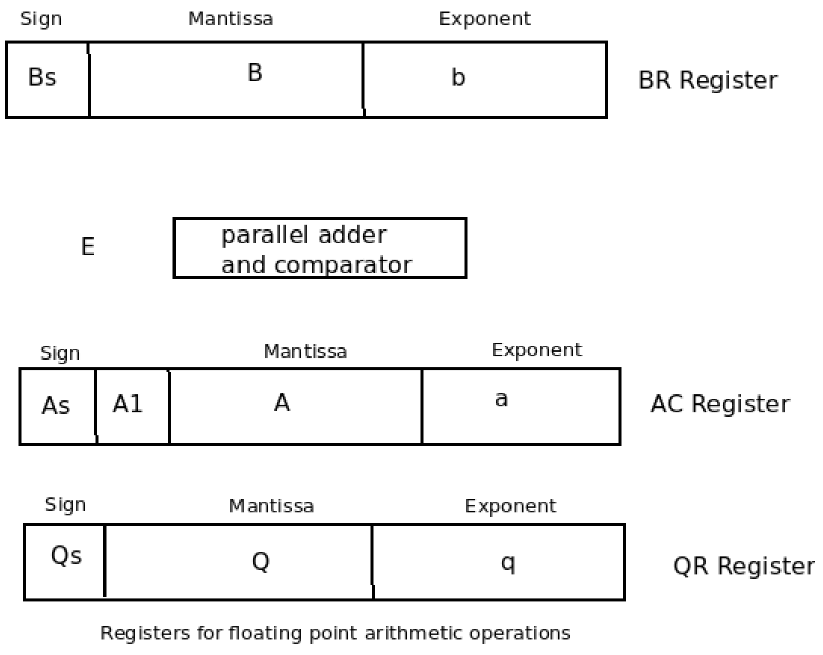

Figure 0 Registers for floating point arithmetic operations

There are three registers: AC, BR, and QR. Each register holds one floating point number, but a floating point number has three parts: sign, mantissa, and exponent. In the AC register, As indicates the floating point number sign, A1 indicates the most significant bit in mantissa, and a indicates the exponents of the floating point number.

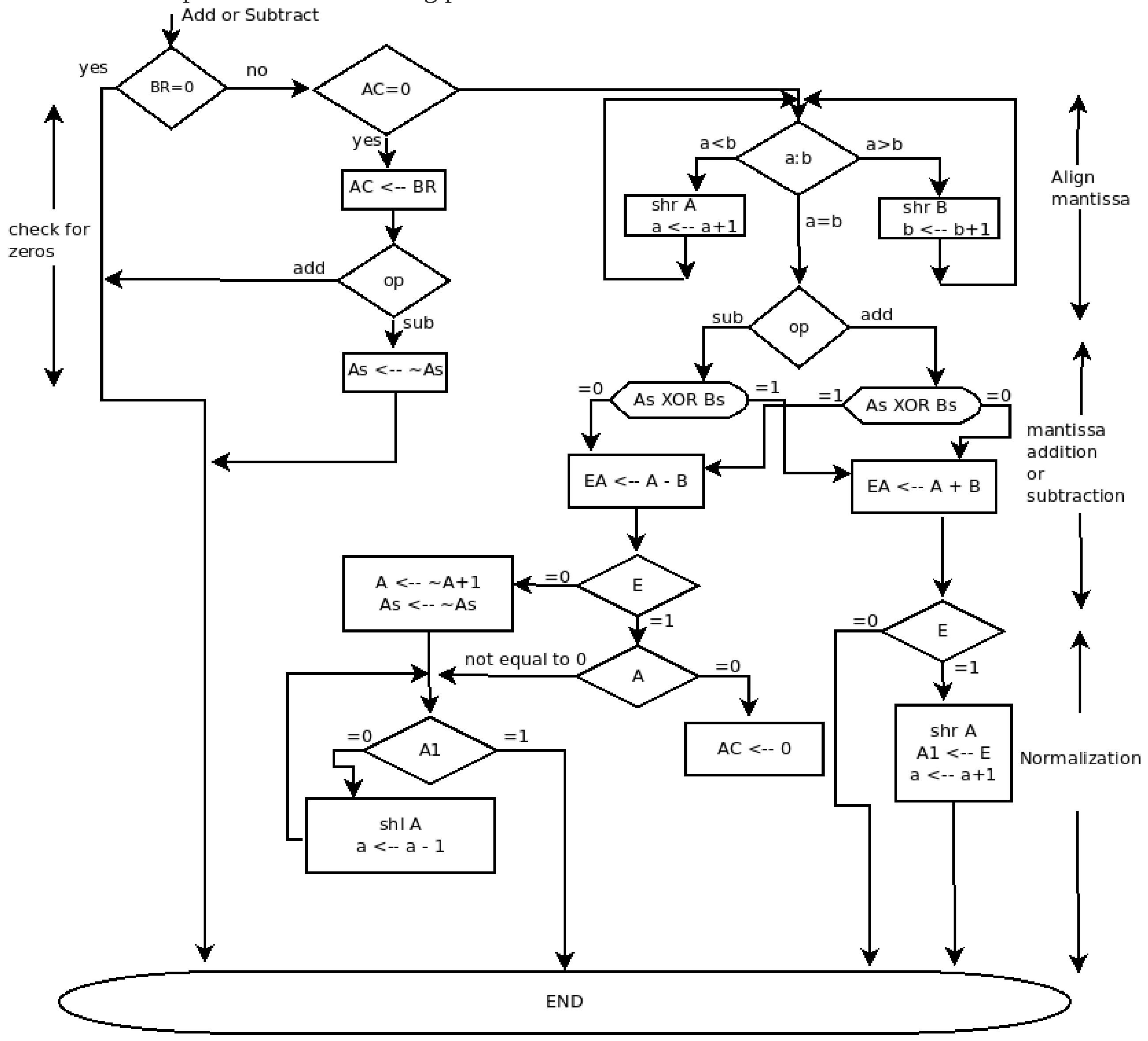

Figure 1 floating point numbers addition or subtraction

In figure 0, the BR register has three parts: Bs indicates sign bit (it is the sign of the second floating point number), B indicates the mantissa part of the second floating point number, and b indicates the exponent part of the second floating point number.

In Figure 1, first we have to check for zeros of two floating point numbers for addition or subtraction operation. If the first floating point number is zero, then the second floating point number, which is stored in BR, is the result of either an addition or subtraction operation. In the subtraction operation, if the BR value is negative, then a positive sign is assigned to the resultant register. If BR has a positive value, then a negative sign is assigned to the resultant register.

If two mantissa exponents are equal, then based on the sign bits of the XOR operation, either an addition or subtraction operation is performed. If not equal, then one of the floating point numbers aligned means shifted to the right or the left. After mantissa alignment based on the XOR gate output, either an addition or subtraction operation is performed.

If the resultant floating point number is not in normalization form and it is identified based on the rightmost bit, If the rightmost bit, that is, A1 is 0 means underflow (meaning not in the normalization form),then the mantissa part is shifted left and the exponent is decremented.

Multiplication

1. Check for zeros
2. Add the exponents
3. Multiply the mantissa
4. Normalize the product

Before multiplication, in floating point multiplication, we need to check if any one of the numbers is zero or not. If any floating point number is zero, then the product is zero. If both floating point numbers are non-zero numbers, then the result is stored in the AC register.

If both numbers are non-zero, then exponents are added and mantissas are multiplied. The product is stored in the AC register. If the final result is not in normalised form, then the result is normalised by shifting to the left or right so that the exponent is either incremented or decremented.

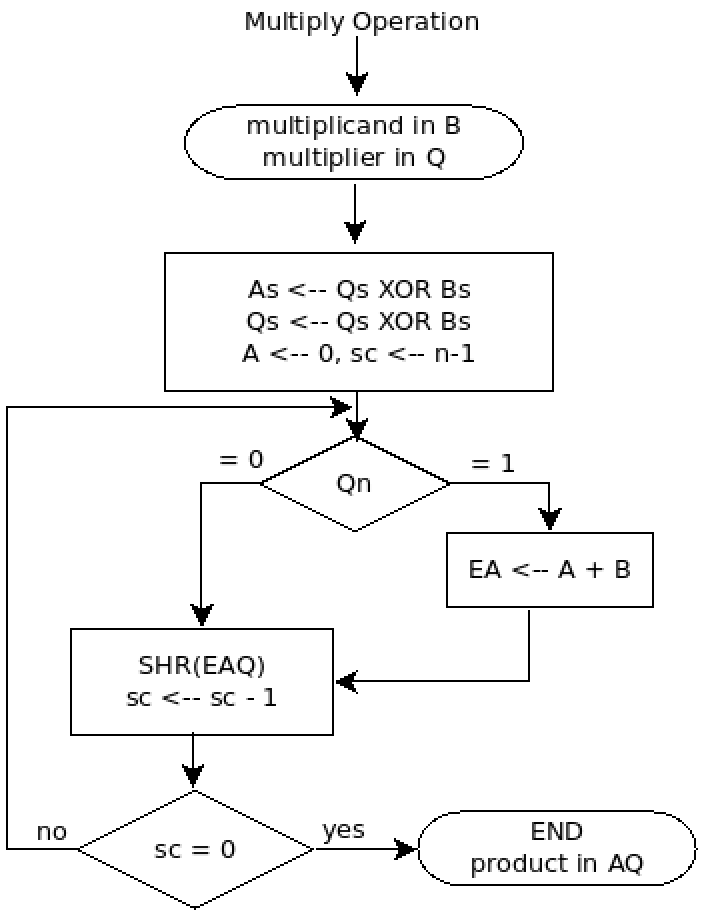

Figure 2 magnitude multiplications

Example:

$.5*10^2$

$.3*10^4$

$.15*10^6$

In this example, two floating point numbers' exponents are different. So exponents are added and the resultant exponent is stored in "a", which is part of the AC register. After that, mantissas are multiplied and the resultant mantissa is stored in A. Based on the rightmost bit of the mantissa, the result is normalized. If A1 is 0, indicating underflow, the mantissa is shifted to the left and the exponent is decremented by one. In floating multiplication from resultant exponent biased value is subtracted.

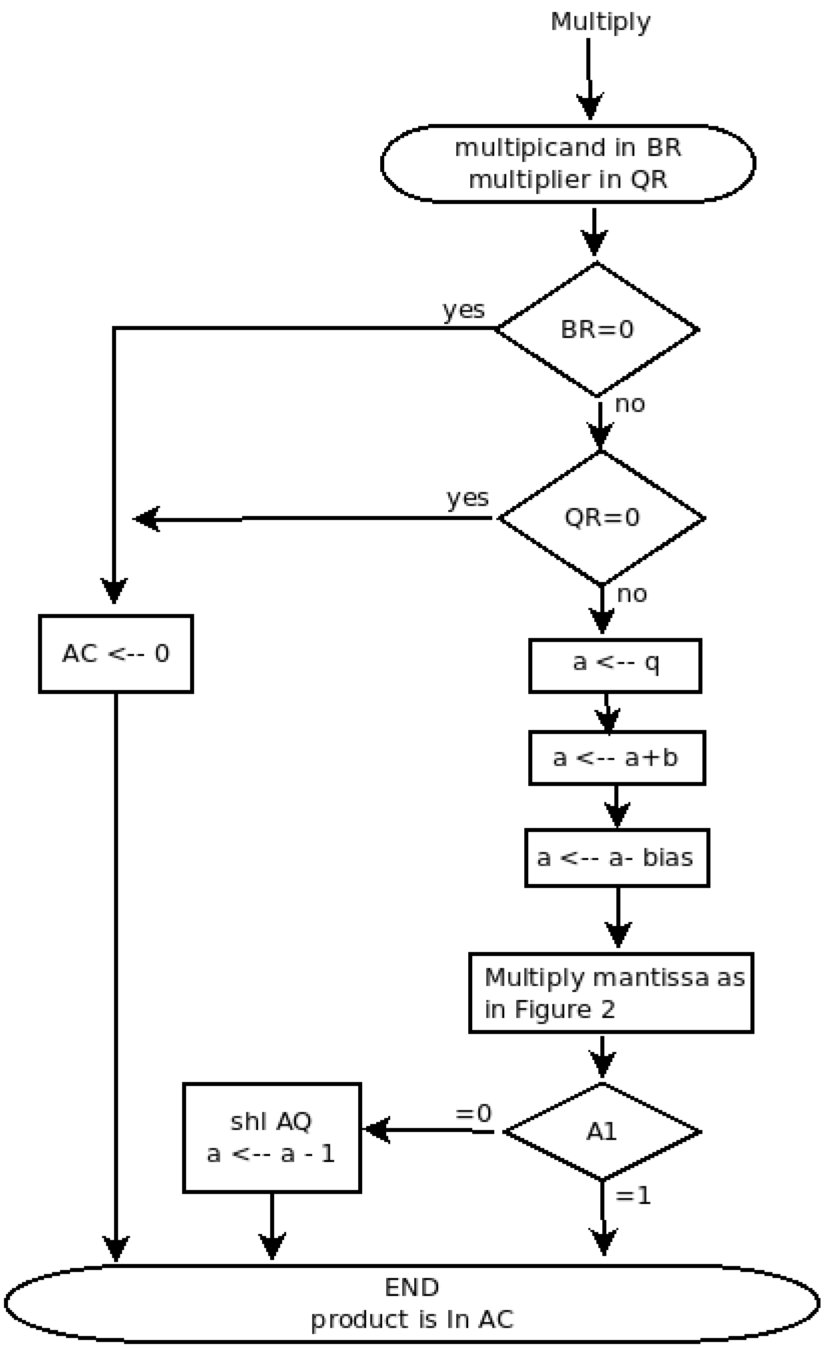

Figure 3 floating point numbers multiplication

Division

1. check for zero’s
2. Initialize registers and evaluate the sign
3. Align the dividend
4. Subtract the exponents
5. Divide the mantissa.

In division operations, first we need to check if the divisor is zero or not. If the divisor is zero, then divide overflow will occur. If the divisor is not zero, then exponents are subtracted. After that, mantissas are divided.

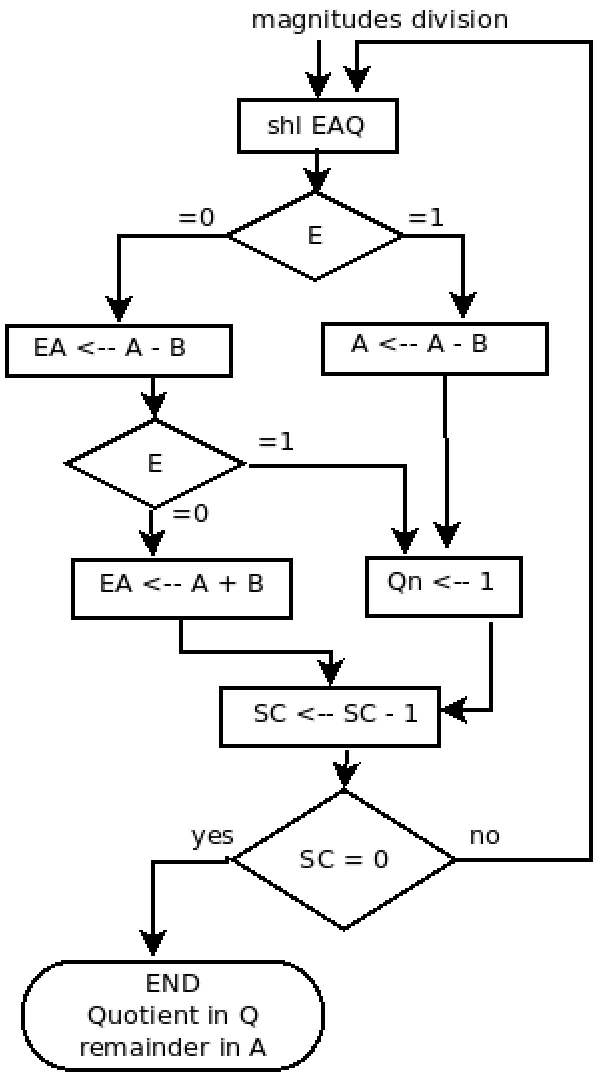

Fig 4 magnitude division

If the divisor is a non-zero floating point number but the dividend is zero, then the result is also zero after the division operation. If both are non-zero floating point numbers, then exponents are subtracted and the resultant exponent is stored in **"q"** because after division operation, the resultant floating point number is stored in the quotient register. In the QR register, there are three parts for floating point numbers. In the first part, **Qs** indicates the sign of the result, the second part **Q** indicates mantissa, and the third part **q** indicates the exponent of the result. After exponents subtraction biased value is added to the exponent which is stored in “a”.

Example:

$.15*10^{12}$

$.5*10^{7}$

$.3*10^{5}$

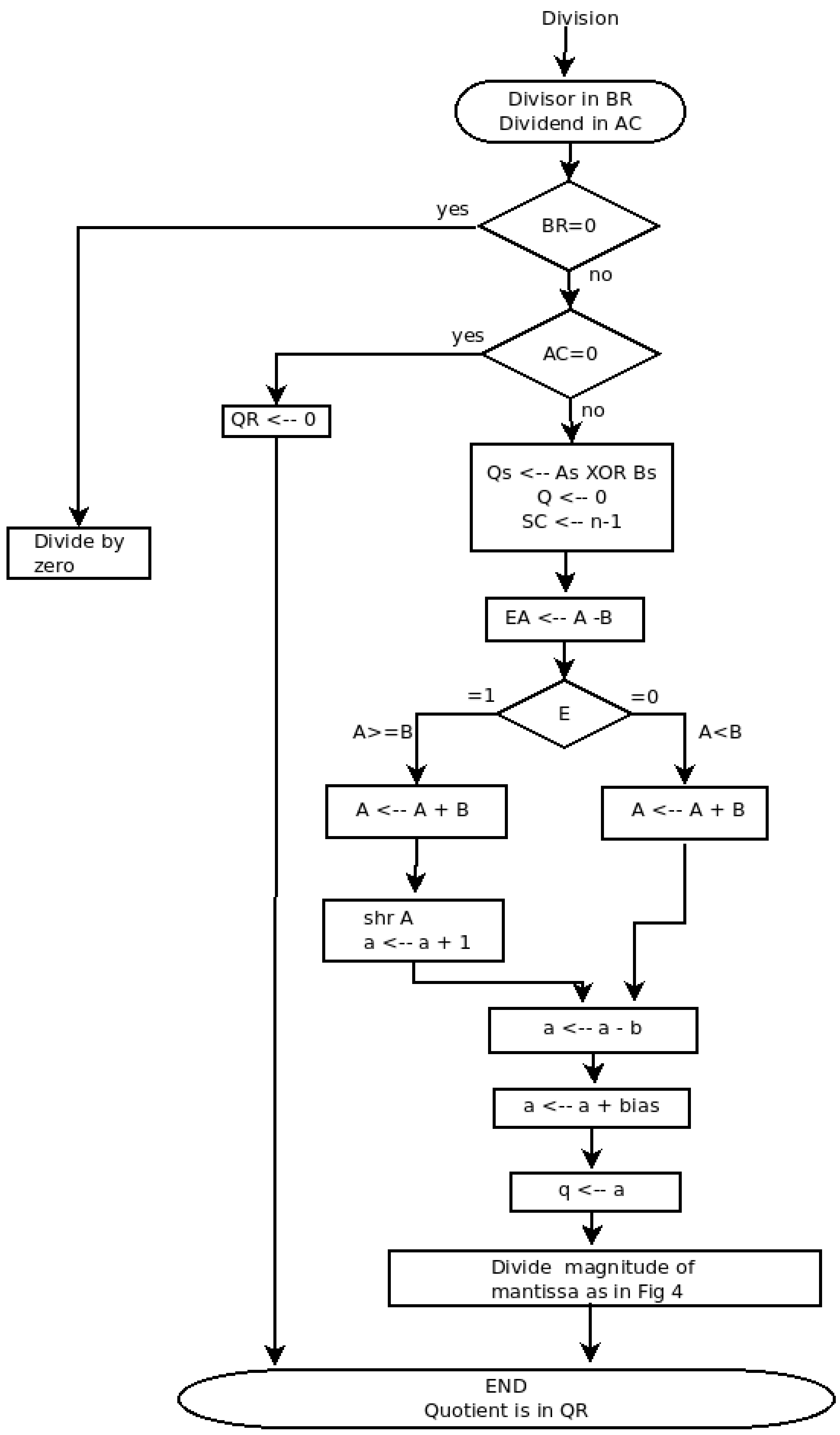

Figure 5 floating point number division