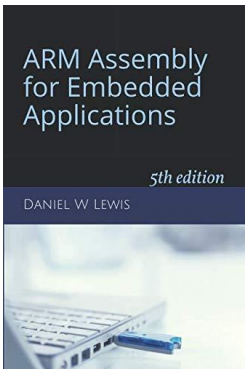


Programming Lab #8f

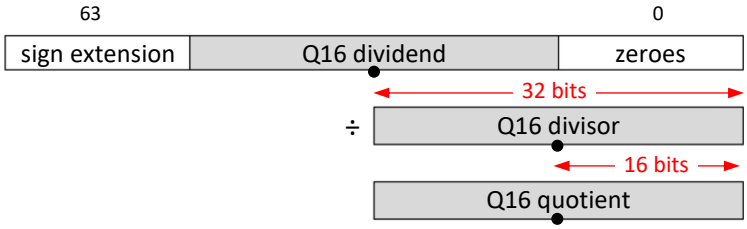
Implementing Division for Q16 Fixed-Point Reals

Topics: Representation of real numbers using Q16 fixed-point.

Prerequisite Reading: Chapters 1-11

Revised: June 3, 2020

Division of one Q16 fixed-point real by another requires that the 32-bit Q16 dividend be positioned in the middle of a 64-bit integer and sign-extended so that the imaginary binary point will be in the middle of the resulting quotient.



Unfortunately, the ARM processor's integer divide instructions only support a 32-bit dividend. Writing a function to do 64÷32 division usually requires a loop that repeats 32 times – once for every bit in the divisor. However, since the Q16 dividend is 32-bits, the code below simply uses the quotient and remainder of a 32÷32 division to extend the result with a loop of only 16 iterations.

To do: Translate the algorithm into an ARM assembly language function. Test your solution with the C main program found here. Since the objective is speed, use the `.rept` directive instead of a loop, avoid conditional branches, and use bitwise operations to change the sign of a value.

```
typedef int32_t Q16 ;

Q16 Q16Divide(Q16 dividend, Q16 divisor)
    {
    uint32_t quotient, remainder ;
    int32_t sign ;
    int k ;

    sign = (int32_t) (dividend ^ divisor) ;
    if (dividend < 0) dividend = -dividend ;
    if (divisor  < 0) divisor  = -divisor ;
    quotient  = dividend / divisor ;
    remainder = dividend % divisor ;
    for (k = 0; k < 16; k++)
        {
        quotient  = quotient  << 1 ;
        remainder = remainder << 1 ;
        if (remainder >= divisor)
            {
            remainder -= divisor ;
            quotient++ ;
            }
        }

    if (sign < 0) quotient = -quotient;
    return quotient ;
    }
```

The main program repeatedly calls your `Q16Divide` function with randomly selected dividends and divisors and compares the quotient and execution time to that of a C reference version based on 64÷32 division. Updates to the display will pause on any error or while the blue push-button is pressed. Errors are displayed as white text on a red background.