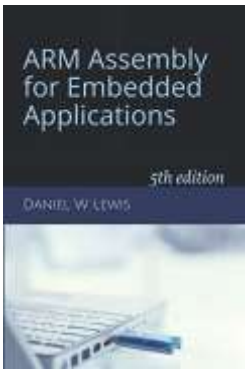


Programming Lab 9G

Floating-Point Reciprocal

Click to download Lab9G-Main.c

Topics: Floating-point arithmetic

Prerequisite Reading: Chapters 1-9
Revised: June 12, 2021

***Background*[1]:** The floating-point instruction set of some processors does not include a divide instruction, or at best it is one of the slowest instructions. For many applications, division can be replaced by multiplying by $1/x$ – provided there is a fast and reasonably accurate way to compute it. A popular approach is to use Newton's root-finding method to improve upon an initial estimate. A good initial estimate may be created by treating the bits representing the floating-point number x as a 32-bit integer and then subtracting it from two times 0x3F800000 (the floating-point representation of +1.0). When interpreted as a float, the resulting bit pattern provides a very good first approximation to the reciprocal (*or inverse*) for values of x in the range:

$$2^{-126} \le |x| \le 2^{126} \quad \approx \quad 1.2 \times 10^{-38} \le |x| \le 8.5 \times 10^{37}$$

The algorithm then runs one iteration of Newton's method, yielding a final approximation that has a worst-case error of less than two percent.

Assignment: The main program will compile and run without writing any assembly. However, your task is to create equivalent replacements in assembly language for the two C functions shown below and found in the main program. The original C versions of the functions have been defined as "weak" so that the linker will automatically replace them in the executable image by those you create in assembly; you do not need to remove the C versions.

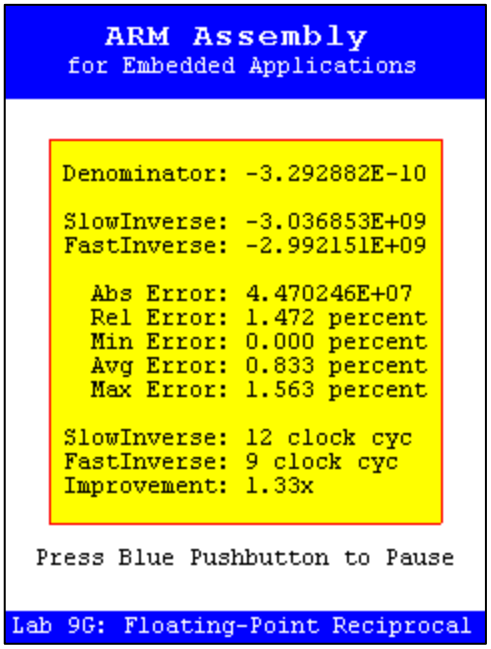
```
float SlowInverse(float denominator) ;
```

Uses the floating-point divide instruction to compute the result.

```
float FastInverse(float denominator) ;
```

Uses Newton's method with first approximation described above.

Test your implementation using the C main program. If your code is correct, the program will call the two functions continuously with random floating-point values, displaying their resulting values, the absolute, relative, minimum, average and maximum error, and compare the execution time performance. Pressing and holding the blue pushbutton will pause the execution; releasing the pushbutton allows execution to resume. If an error occurs, an error message will be displayed as **white text on a red background** and execution halted.



[1] https://www.yumpu.com/en/document/read/6104114/floating-point-tricks-ieee-computer-graphics-and-applications