System Programming

DHBW Stuttgart, 2019
Prof. Dr.-Ing. Alfred Strey

Project assignment TINF17A

The MIPS floating-point coprocessor only supports all four basic arithmetic operations for both single and double precision IEEE 754 numbers. To achieve a very high performance for many scientific and technical applications, it might be useful to implement also the calculation of some mathematical functions at machine level.

1) The first task consists in writing a function to implement exp(x) in MIPS assembly language. Here a Taylor series can be used for the approximation:

$$\exp(x) = e^x = \sum_{i=0}^{\infty} \frac{x^i}{i!} = \frac{x^0}{0!} + \frac{x^1}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots$$

The following steps have to be performed:

- First implement a high-level program (preferably in C) that approximates exp(x) according to the formula above and compares the result with the corresponding library function. Use the IEEE single precision format here! How many terms of the Taylor series are required for a good approximation?
- Now implement in MIPS assembly language a function `float exp(float x)` that calculates exp(x). Define a suitable limited domain [x_{min}, x_{max}] from which an argument x can be selected with a sufficient low approximation error.
- Test and debug your MIPS function `exp(x)` with SPIM!

2) Also the inverse function ln(x) has to be implemented in MIPS assembly language.

- The natural logarithm of x can be approximated by the following Taylor series:

$$\ln(x) = \sum_{i=1}^{\infty} (-1)^{i+1} \frac{(x-1)^i}{i} = \frac{(x-1)}{1} - \frac{(x-1)^2}{2} + \frac{(x-1)^3}{3} - \frac{(x-1)^4}{4} + \cdots$$

  Please note, that in contrast to exp(x) this Taylor series only converges for $0 < x \leq 2$. Again please write first a high-level program (preferably in C) for the approximation of ln(x). Determine how many terms of the Taylor series are required for a good approximation.
- Then write in MIPS assembly language a function `float ln0(float x)`, that calculates ln(x) for an argument x in the range $0 < x \leq 2$ by the Taylor series above.
- The domain for the argument x can be extended to support also $x > 2$ by applying the equation $x = a \cdot 2^b$ and calculating $\ln(x) = \ln(a) + b \ln(2)$ with $a \leq 2$. Implement a generally applicable function `float ln(float x)` that supports arbitrary arguments x and calls function `ln0` internally.
- Test and debug your implementations of `ln0(x)` und `ln(x)` with SPIM.

3) Finally write in MIPS assembler a test program `main()`, that first asks the user for three values n, x_{min}, x_{max}, then calculates $y_i = \exp(x_i)$ and $z_i = \ln(y_i)$ for n equidistant values x_i in the interval $[x_{min}, x_{max}]$ and finally prints all values x_i, y_i and z_i on the screen.

Test your main program with SPIM!

4) **Optional** improvements:

- Optimize all MIPS functions code for highest efficiency!
- Try to determine an optimal number of terms for the Taylor approximation dependent on the argument x.
- The exponential function for arbitrary (possibly large) arguments x can be calculated by the formula: $e^x = 2^k \cdot e^{x - k \cdot \ln(2)}$. Consider how your above implementation of $\exp(x)$ can be extended to support arbitrary arguments x by applying this formula. Rewrite in MIPS assembly language your function `exp` tos support arbitrary x.
- Give the worst case accuracy of your approximation with respect to the selected domain.
- Check in each function if the argument x *i*s included within the selected domain.

General remarks:

1. Respect the requested calling hierarchy. Use the stack for saving registers, if appropriate. It is not necessary here to use the frame pointer `$fp`!
2. All MIPS assembler implementations should be close to the high-level functions.
3. Stick to all MIPS conventions! In subroutines with floating-point numbers the registers $f0 to $f19 must not be saved on the stack!
4. Comment each line of your assembler source code if useful!
5. The usage of MIPS pseudo instructions is allowed and strongly recommended!
6. Teams of up to two students are admitted.
7. Please submit your project files by email to strey@lehre.dhbw-stuttgart.de before the agreed deadline. The submission (one for each team) should contain the high-level source files, the commented assembler sources (runnable on SPIM simulator) and a short project documentation (max. two A4 pages) as pdf file.