EE 445 COMPUTER ARCHITECTURE I
HOMEWORK 3

DUE DATE
12.01.2020
23.59

In this homework, you are going to implement a simple calculator to calculate factorial, exponent, sine and cosine functions based on Booth's multiplication algorithm, fixed point arithmetic and Taylor series expansion.

a. Implement Booth's multiplication algorithm as a module on Verilog. Your implementation should take two 16 bit integers as input and return the result as a 32 bit integer.

b. Implement a module to calculate the factorial of a small integer N by multiplying the integers $1 \leq k \leq N$. In this step, you might assume that the end result fits in 32 bits, and all intermediate calculations fit in 16 bits.
(Hint: In the intermediate steps, your multiplication module returns a 32 bit result. However, the multiplication fits in 16 bits. Hence, you need to use the least significant 16 bits for the next step.)

c. In this step, you are going to use fixed point arithmetic. In this homework, we use unsigned numbers and 16 bits fractional part. In this representation, a 16 bit number represents the fractional number which is equal to its integer equivalent divided by 2^{16}.
Ex: 0010 0100 1001 0011 = 9363 (decimal) = 9363/65536 = 0.1429 (fixed point)
Implement three different modules to calculate exp(x), sin(x) and cos(x) using your multiplication module and truncated Taylor series expansion. Stop the Taylor series at x^{10} term. That is, use 10 terms for exp(x), and 5 terms for sin(x) and cos(x).
(Hint: In this representation, multiplication of two numbers is equal to $N_1/2^{16} * N_2/2^{16} = N_1N_2/2^{32}$. To convert this to the closest 16 bit fixed point representation, we rewrite it as $(N_1N_2/2^{16})/2^{16}$. Hence, you need to use the most significant 16 bits of the multiplication for the next step.)
(Hint: As you do not implement a division module, you might consider precalculating fixed point representations of 1/n! for n=1,2,…,9 by hand and generating a lookup table for them.)

Test your codes for a small number (e.g. 0.1429). Submit your verilog codes and simulation results.