Noah Hanks

ECEN 424 HW 4

1) The time stamp counter (TSC) is a high-resolution timer present in x86 processors that can be used to measure time intervals with very high precision. However, using the TSC for accurate time measurement comes with several challenges. One of the main issues is that the TSC is not always reliable and can produce incorrect results if certain conditions are not met. For example, the TSC is only guaranteed to be monotonic if the processor's frequency is constant, which is not always the case due to power management techniques used by modern processors. Also, if the measurement interval is too short or long, the TSC's results can be affected by factors such as CPU frequency changes, context switches, and other interrupt handling, leading to a loss of repeatability and accuracy. These challenges can make it difficult to use the TSC as a reliable source of timing information in certain situations, and alternative methods such as operating system timers may be needed.
2) To handle the issue of having too small of a measurement interval, the source code loops through and calls the function until the minimum threshold is met. To handle the issue of repeatability, the source code calls the function multiple times and stores the best time in an array.
3) In vector.h there are several parameters that can be changed. "data_t" is the data type that the operations will be on. "OP" is the kind of operation to perform. In the Makefile there are flags that can be set that changes the optimization level when compiling with gcc. In "getcpe.c", "VECVALS" changes the number of unique vector lengths tested. "VECMAX" changes the max vector length. "MEASMAX" changes the number of runs to make each vector length.

   The two parameters I was checking was "VECVALS" and "VECMAX". I found that to match the book's results for integer addition closest, it is best to set them to 1000 and 1024 respetively. So "VECVALS" had to be increased while "VECMAX" stayed the same.
4)

	combine1()	combine2()	combine3()	combine4()
long add	8.9	4.2	4.4	1.1
long multiply	7.5	4.2	4.3	1.1
double add	7.4	7.3	7.1	3.0
double multiply	7.7	7.2	7.1	3.0

5) Comparing the values between what the book had and what I received, I noticed that the trend is pretty much the same. The difference between the values is that mine are usually a little lower. There could be lots of causes that would create this difference. One difference could be this version of gcc is just better at optimizing that the version of gcc that the textbook uses. The processor in my computer is also likely to be more recent than that of the book.