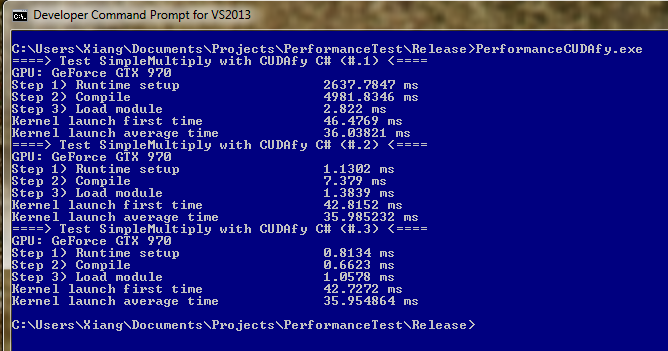
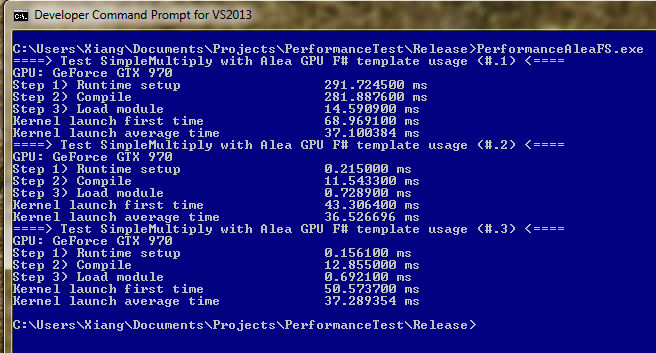
Performance Test

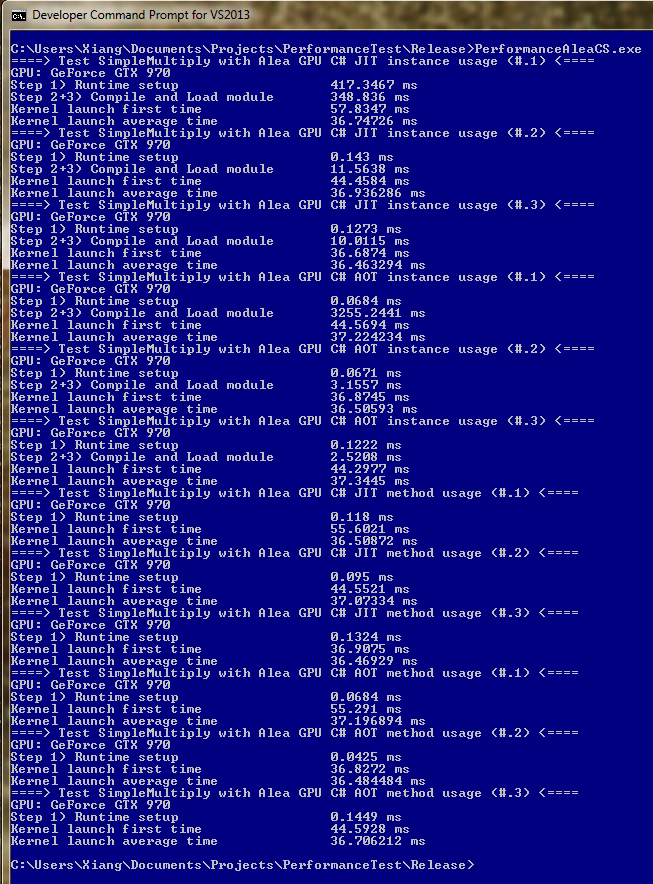
The performance of CUDAfy. Startup code has 3 steps, setup runtime; compile; and load module. Since .NET has JIT compile, so I run it three times:



Now, we use Alea GPU F# template technique, it also has 3 steps. But in Alea GPU, in order to locate natives such as LLVM and NVVM, we use a dynamic P/Invoke tech, so each P/Invoke function is a lazy value, and when they are first used, it is created by reflection to generate P/Invoke IL code. This results that the first time when it compiles or calling CUDA Driver API, it is slow, but the second time, it will be fast. But the kernel launching time is a little slow compare to CUDAfy, should be checked further, especially check CUDAfy’s code.



Now, we test Alea GPU with C#, I coded 4 tests, for instance/method usage + AOT/JIT combination.



So I wonder why Alea launching is a little slow, I guess it is mostly because of some runtime stuff, cause we have an extendable mashaller registery. I use Nsight to profile the kernel time. Left side is CUDAfy, right side is Alea. So looks like CUDAfy generated kernel (through nvcc) is a little faster. Especially, they use 22 registers, alea uses 24 registers.

