

Modular arithmetic

using OpenCL

Problem description

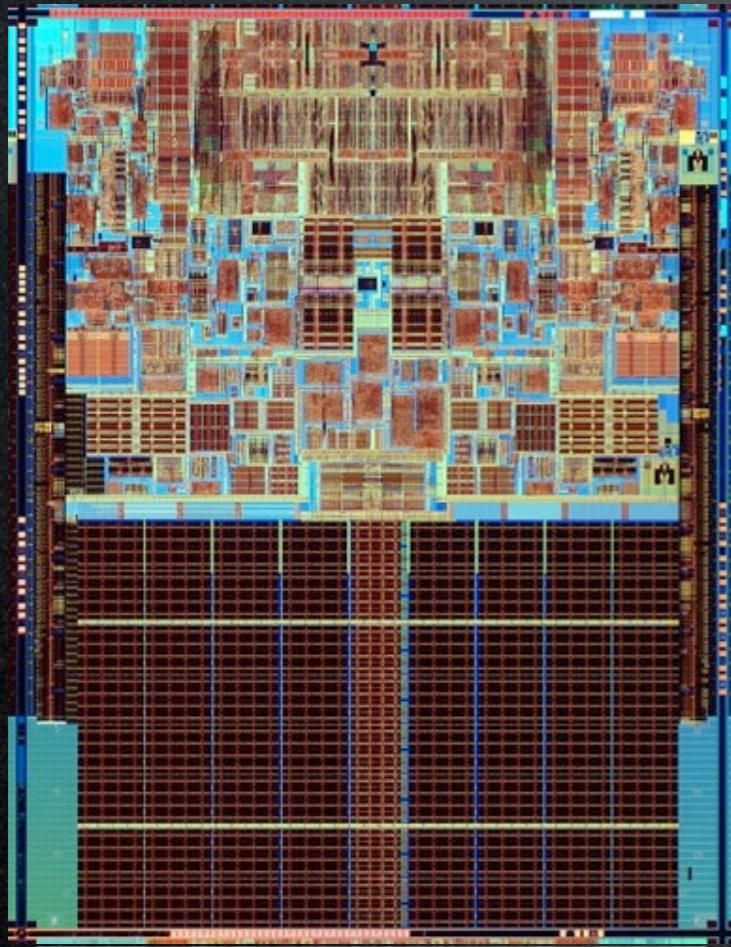
- Most public key algorithms are based on modular arithmetic.
- The simplest, and original, implementation of the protocol uses the multiplicative group of integers modulo p , where p is prime and g is primitive root mod p .
- For this reason public key crypto RSA is much slower than symmetric key algorithms, like DES and AES.
- Recently the field of using Graphics Processing Units (GPUs) for general purpose computing has become more widespread. Many computational problems have gained a significant performance increase by using the highly parallel properties of the GPU.

Modular arithmetic

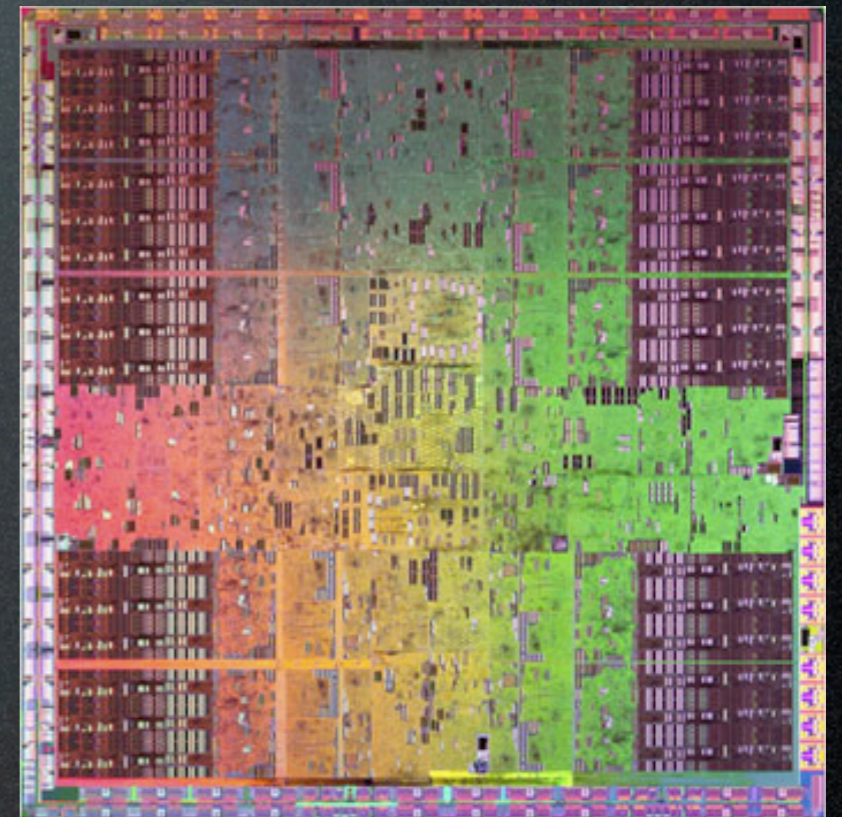
- $c = m^e \bmod n$
- Used in RSA to encrypt/decrypt
- Very expensive processing

CPU vs GPU

Intel Core 2 duo



Nvidia GT200



OpenCL



- OpenCL (Open Computing Language) is an open royalty-free standard for general purpose parallel programming across CPUs, GPUs and other processors, giving software developers portable and efficient access to the power of these heterogeneous processing platforms.
- OpenCL is being created by the Khronos Group with the participation of many industry leading companies and institutions.

OpenCL



- Supports both data- and task-based parallel programming models
- Utilizes a subset of ISO C99 with extensions for parallelism
- Defines consistent numerical requirements based on IEEE 754
- Defines a configuration profile for handheld and embedded devices

OpenCL

- OpenCL is made up of tree parts



OpenCL C



- C-based cross-platform programming interface
- Subset of ISO C99 with language extensions - familiar to developers
- Well-defined numerical accuracy - IEEE 754 rounding behavior with defined maximum error
- Online or offline compilation and build of compute kernel executables

OpenCL API



- A hardware abstraction layer over diverse computational resources
- Query, select and initialize compute devices
Create compute contexts and work-queues

OpenCL Runtime

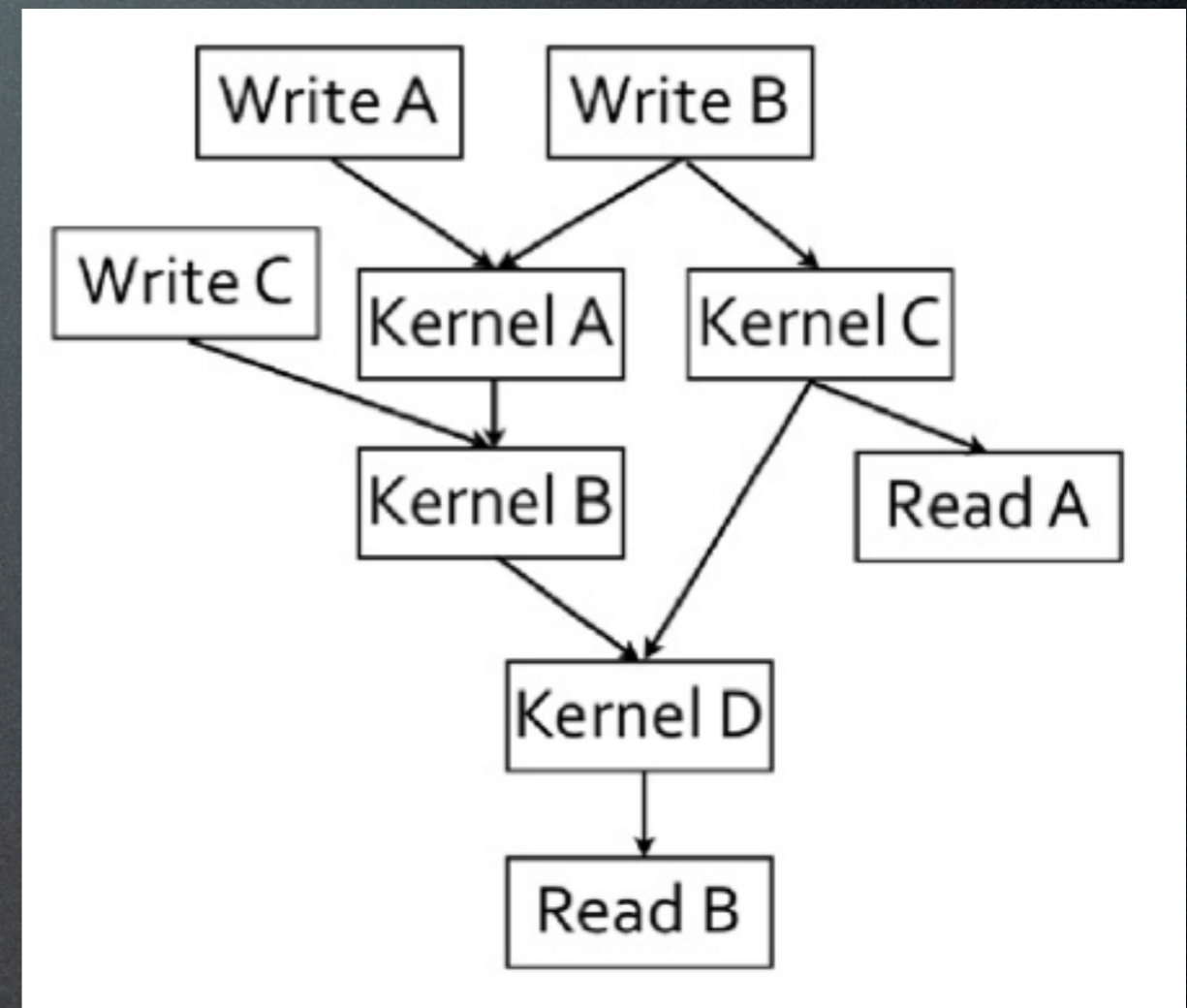


- Execute compute kernels
- Manage scheduling, compute, and memory resources

Command Queues



- To execute a kernel, the kernel is pushed onto a particular command queue.
- Enqueueing a kernel is done asynchronously, so that the host program may enqueue many different kernels without waiting for any of them to complete.
- When enqueueing a kernel, the developer optionally specifies a list of events that must occur before the kernel executes.
- This allows the developer to specify a dependence graph between kernel executions and memory transfers in a particular command queue or between command queues themselves, which the OpenCL runtime will traverse during execution.



OpenCL limitations



- Recursion is not supported.
- The function using the `__kernel` qualifier can only have return type `void` in the source code.
- Variable length arrays and structures with flexible (or unsized) arrays are not supported.
- The C99 standard headers `assert.h`, `ctype.h`, `complex.h`, `errno.h`, `fenv.h`, `float.h`, `inttypes.h`, `limits.h`, `locale.h`, `setjmp.h`, `signal.h`, `stdarg.h`, `stdio.h`, `stdlib.h`, `string.h`, `tgmath.h`, `time.h`, `wchar.h`, and `wctype.h` are not available and cannot be included by a program.

Experiments



Equipment used



- Apple Mac Pro
 - 8 CPU cores - 16 threads
 - 12 GB Ram
 - Nvidia 285 GTX GPU - 240 cores

Algorithm



- Montgomery algorithm used
- Based on the work by Harrisson et al.
- Changed the way the operations mapped to the GPU to further exploit the potential of the GPU at hand.

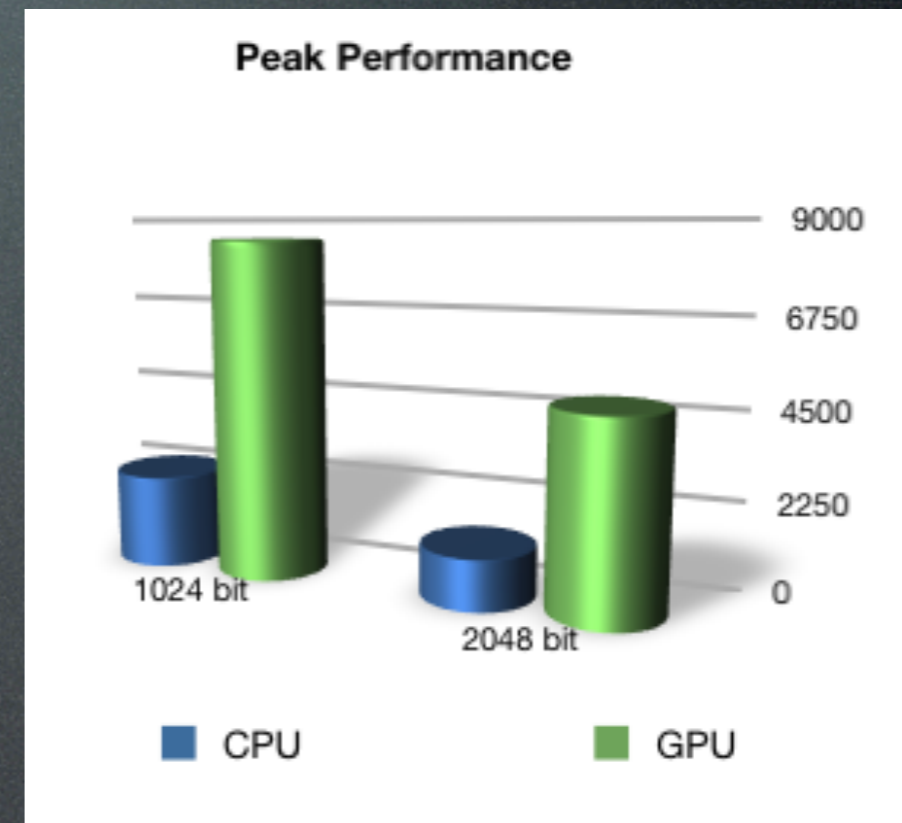
Algorithm



- Harrisson et al. used a highly optimized CUDA kernel mapped to 30 cores to do 1024 bit calculations. Not usable for 2048 bit.
- Needed to change the mapping and how it synchronized it's threads.
- Also optimized the way it used memory, this gave the greatest speed performance gain.

Results

- CPU has good performance
- GPU has even better performance



Results

- Harrison et al.
 - in line with the work Harrison et al. did on 1024 bits.
- Szerwinski et al.
 - Better results than they got.

Results

- Used last generation of the Nvidia GPU series at the time of running the experiments
- Parting the problem into sizes that fit the GPU

Conclusion

- OpenCL is not “there”.
 - Write once and run every where not 100%
 - Still early days missing good development tools
- GPU is faster than CPU for larger operations

Further work

- AMD/ATI GPU
- Extend to larger bit size, 4096 bits and 8192 bits
 - May require new algorithms
- Elliptic curve cryptography (ECC)