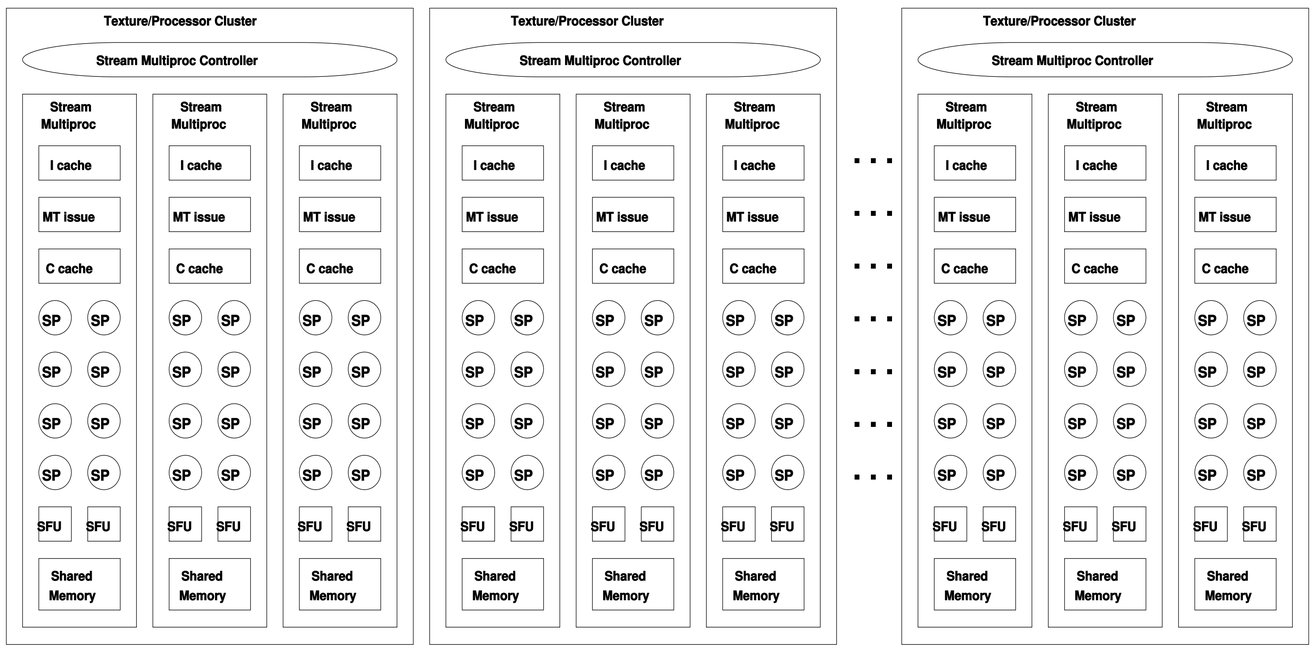
GPGPU Acceleration of Cryptographic Applications

Giovanni Agosta[1], Alessandro Barenghi[1], Fabrizio De Santis[1], Andrea Di Biagio[1], Gerardo Pelosi[2]

[1] *Dipartimento di Elettronica e Informazione*
Politecnico di Milano
Piazza L. da Vinci 32, Milano, ITALY
{agosta,barenghi,dibiagio}@elet.polimi.it
fabrizio.desantis@mail.polimi.it

[2] *Dipartimento di Ingegneria dell'Informazione e Metodi Matematici*
Universita' degli Studi di Bergamo
Viale Marconi 5, Dalmine (BG), ITALY
gerardo.pelosi@unibg.it

CUDA Programming Model



Sketch of the NVIDIA GT200 streaming processors array architecture: each Texture/Processor Cluster contains three stream multiprocessors. In turn, each stream multiprocessor is composed of eight streaming processor cores (SP), plus two special function units (SFU).

Abstracts the actual parallelism implemented by the hardware architecture.
Provides concepts of *block* and *thread* to express concurrency. A block captures the notion of a group of concurrent threads. Blocks are required to execute independently, so that it has to be possible to execute them in any order (in parallel or in sequence).

Fragment of CUDA code: kernel definition

```
__global__ void kernel(
      int *A, int *B, int*C)
{
        /* A B and C are pointers
           to array data structures */
        /* var is local to each thread */
        int var = 5 * ThreadId.idx;
        A[i] = B[i] + C[i] + var;
}
```

Fragment of CUDA code: kernel invocation

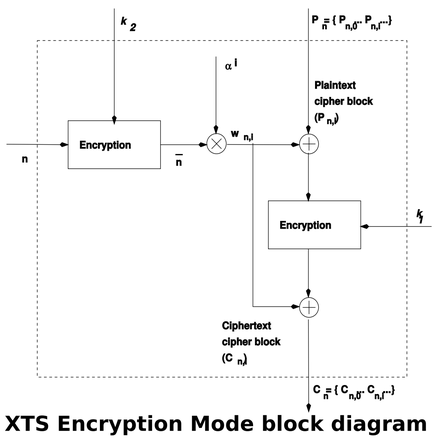
```
int *A, *B, *C;
...
kernel<<<1,N>>>(A, B, C);
```

Primary Design Choice

Packing of *threads* into *blocks*: needed to strike a trade off between synchronization and performance, as well as to distribute execution across stream multiprocessor (a block is assigned to a stream multiprocessor).

Usage of Shared Memory: fast, shared memory is available, but is a critically scarce resource. It can be used for data sharing among threads in the same block, or to supplement registers within a single block.

Case Study: Truecrypt XTS encryption



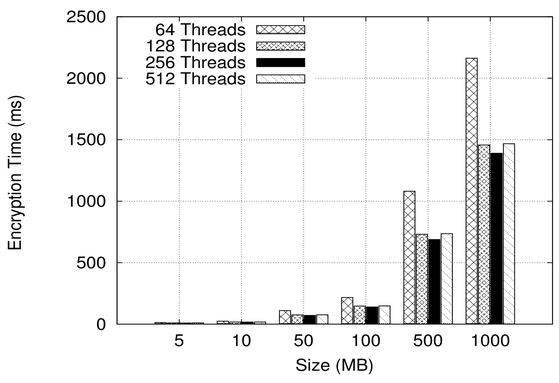
XTS Encryption Mode block diagram

Truecrypt: standard-compliant application for disk volume encryption, used to encrypt large amounts of data.

The XTS mode of operation was designed primarily for the encryption of data on block oriented devices and therefore assumes that the plaintext is naturally split into data units.

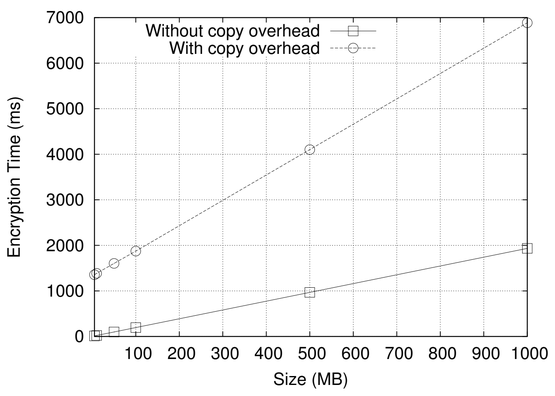
Data units can processed in parallel, providing parallelism to exploit with the GPU.

Design Choices and Experimental Results



Number of threads per block

Exploration of DSE shows that 256 threads per block give best results for all common data sizes.



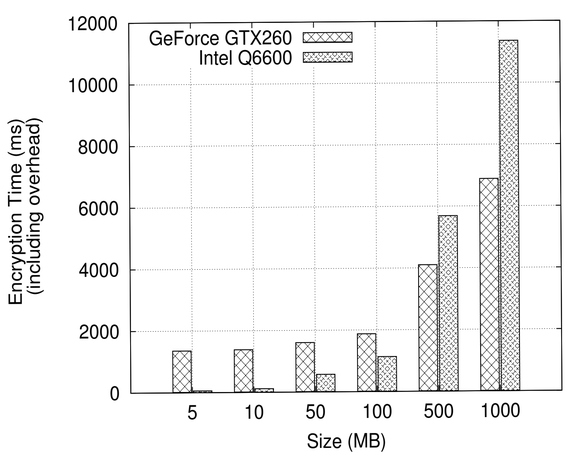
Experimental Setup

Host system: Intel Core 2 Quad Q6600 2.4 GHz, 8 MB L2 cache

Device: NVidia GTX260 with 192 processing cores, 896 Mbytes GDDR3 memory.

CUDA toolkit version 2.1
OS: Gentoo Linux for x86_64

Host to device connection: PCI-Express version 1



Experimental Results

For small sizes of data, CPUs win, due to the GPU wake up overhead.

On heavier workloads, the GPU wins, due to higher computational power.

Tradeoff point: plaintext size of 184 MB against four cores, or 46 MB per core.

Best speedup: one gigabyte write requests, 67%.

Case Study: Fast brute-force DES Breaking

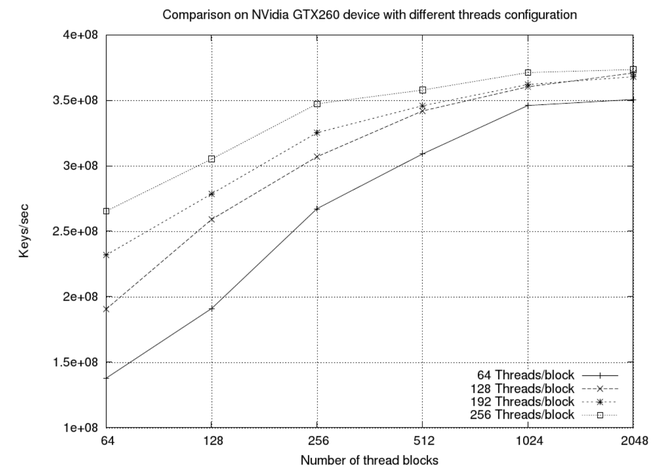
DES has been a NIST and FIPS standard for encryption until 2005.
It is not anymore suitable for high-security applications, but is still used for commercial applications, and is still available in encryption suites such as OpenSSL.
DES was designed specifically for HW implementation, and is known to be slow when implemented in SW. Thus, most brute-force attacks have been conducted using custom hardware (ASIC or FPGA).

Design Choices and Experimental Results

Number of threads per block

Exploration of DSE shows that 128 threads per block give best results in terms of throughput.

Threads per block	Throughput [Mkey/s]
32	61.34
64	75.12
128	75.33
192	72.64
256	75.2



Experimental Results

Throughput: GPU improves by an order of magnitude with respect to CPU, still not on par with FPGA or hardware.

However, we achieve excellent cost/performance metrics, with respect to state of the art FPGA implementations.

In the near future, we expect GPU to become the preferred choice for fast breakers due to the low cost and high availability and programmability of the hardware!

	DES Plain [10^6keys/sec]	DES BS [10^6keys/sec]
CPU	13.32 [11]	36.85 [9]
GPU	75.33 (our GTX260)	373.58 (our GTX260)
FPGA	[2]	–
ASIC	[3]	–

Throughput measures for different DES implementations

nVidia video card model		Graphic Card Cost		Dev cost-equiv to Copacobana	DES Breaking	DES Breaking
Device	**Cores**	**Cost**	**IPS/Euro**	**n*(177$+2*Cost 1 dev)**	**Standard [day]**	**Bitsliced[day]**
GTX295	480	308	897.66	22	100	20
GTX285	240	180	864.00	32	122	24
GTX275	240	136	1117.06	40	100	20
GTX260	192	96	1005.38	44	125	25
GTX260-216	216	110	1462.50	48	90	18
GTX250	128	73	1294.03	56	115	23

Summary of the cost analysis oriented at building at home a DES breaking cluster. (177euro overhead to build the hosts included)

G. Agosta, A. Barenghi, F. De Santis, G. Pelosi: Record-setting software implementation of DES using CUDA, to appear in ITNG 2010
G. Agosta, A. Barenghi, F. De Santis, A. Di Biagio, G. Pelosi: Fast Disk Encryption through GPGPU Acceleration. PDCAT 2009: 102-109
A. Di Biagio, A. Barenghi, G. Agosta, G. Pelosi: Design of a parallel AES for graphics hardware using the CUDA framework. IPDPS 2009: 1-8