Implementation of DES cryptographic algorithm using NVIDIA GPU for a brute-force attack

Miroslav Monok Robert Lorencz

Department of Computer Systems
Faculty of Information technology
Czech Technical University in Prague

June 25, 2013

Schedule

1. Schedule
2. Motivation
3. Data Encryption Standard
4. CUDA GPU
5. Bitslice
6. Reference solutions
7. Implementation
8. Comparison

Motivation

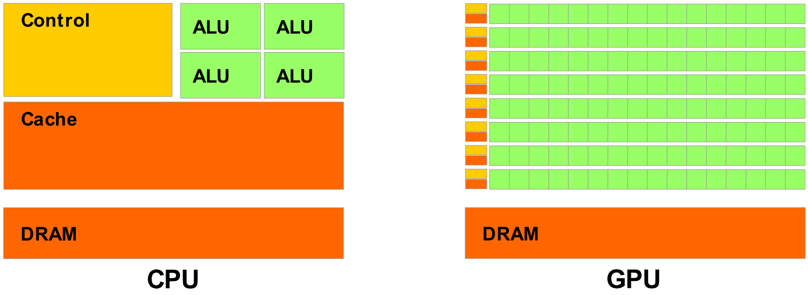
- Create efficient software implementation of DES algorithm using GPU
- Prove that GPU technology can be compared to specialized HW in terms of power/cost ratio

Data Encryption Standard

- Widely used.
- Plenty of different implementations available
- Often used as a "benchmark"
- Till now completely **broken** only by **brutte force attack**

CUDA GPU

GPU as a cheap and powerfull computational platform

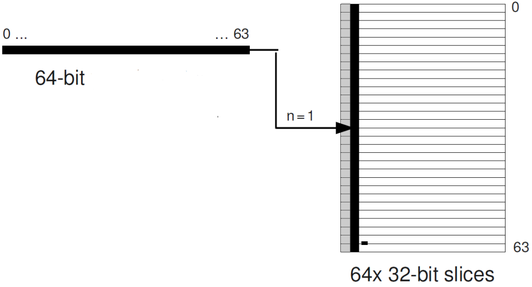


CUDA architecture

- Big amount of simple computational units *Straming Processor*, grouped into *Streaming Multiprocessor*
- CUDA is a complete SW & HW architecture.
- Threads grouped into blocks run in parallel. Parallel part of programme is called *(Kernel)*
- Number of threads and blocks per kernel determines overall power.
- Cuda defines different memory types (registers, shared memory, constant memory (no/cached),main memory,...)
- Memory size vs. memory response time.

Bitslice

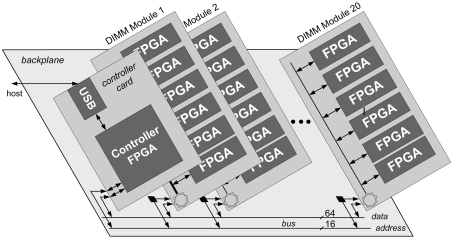
1. Different data representation
2. Bits of word are placed in different variables.
3. Bit-level parallelization
4. SW permutation by addressing → DES is reduced to S-boxes
5. Efective use of memory
6. 1×bitslice DES=32×DES.



Bitslice

- **S-box as a Lookup-table is replaced by logical functions** (using XOR, OR, NOT, AND gates)
- Each of the output bits from S-box is function of all input bits $o_j = f(i_1, i_2, i_3, i_4, i_5, i_6)$
- **DES reduced to S-boxes = one big logical function.**
- Complexity of function determines the speed of algorithm.

COPACOBANA



- Specialized HW based on FPGA technology.
- Up to 120 FPGA cores (Xilinx Spartan-3 XC3S1000)
- **Power:** 65.28×10^9 keys/sec.
- **Price:** 10 000 € (HW), up to 60 000 € as market price.

Record Setting Software Implementation of DES Using CUDA

- Published on (*Seventh International Conference on Information Technology 2010*) by italian group of scientists.
- SW solution running on CUDA GTX 260-216 graphic card.
- using Bitslice data representation.
- **Power:** 373.58 $\times 10^6$ keys/sec.
- configuration: 65535 blocks × 256 threads.
- **Price:** 110 €

HW configuration

Used configuration

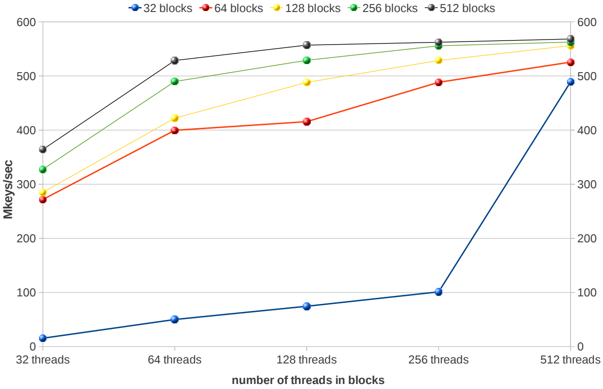
Computational node *Alpha (gpgpu-alpha.fit.cvut.cz)* :

- CPU :Intel i5 760 2.8GHz.
- RAM:8GB.
- **GPU:GeForce GTX 480.**

Basic bitslice DES version

- First version, optimalized by compiler.
- 1 thread=1 bitslice DES= 32 keys.
- Frequent communication with main memory.
- High use of registers = low use of computational units.
- **Power:** 568 330 506.2 keys/sec., version CoreLight 576 792 274.2 keys/sec.

Basic bitslice DES version



Dividing the computation

DES8

- 8 threads, each computing one S-box.
- Variables stored in shared memory.
- Low number of threads per block. Code divergency.
- **speed:** 353 061 687.5 keys/sec (32768 blocks)

DES32

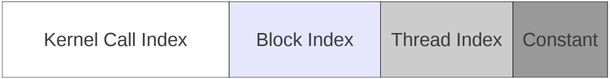
- Each threads computes 1 output bit from S-box.
- Variables stored in shared memory.
- **Power:** 449 628 790.34 keys/sec (32768 blocks)
- Dividing the computation didn't bring desired speedup (code divergency, ...)

DESconst

- Use of constant memory for plaintext and ciphertext and **part of the key**
- **Lower use of registers improved the speed about 20%**
- **Power:** 651 018 433.4 keys/sec.

DESopt

- Key is divided into several parts:



Kernel Call Index

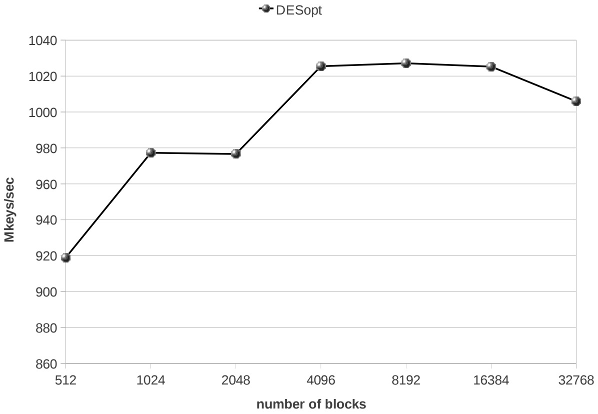
Block Index

Thread Index

Constant

- precomputation of key on CPU = part of the key is constant.
- Does not allow to change number of blocks/threads dynamically
- till now, our most powerfull version
- **Power:** 1.03×10^9 keys/sec. (64 threads / 8192 blocks)

DESopt


DESopt
Mkeys/sec
1040
1020
1000
980
960
940
920
900
880
860
512
1024
2048
4096
8192
16384
32768
number of blocks

Comparison

	Power (Key/sec)	HW Price [€]	Power/price
COPACOBANA	65280×10^6	10 000 (60 000) €	6.528 (1.1)
DES Italy (GTX260)	373.58×10^6	110 €	3.396
DESopt (GTX480)	1030×10^6	210 €	4.904

Implementation - Conclusion

- Speedup **2.76** × comparing with reference SW solution
- Throughput aprox **53.7 Gbit/sec**
- The number of registers available per thread is the limiting factor.
- Speedup was achieved by use of constant memory and by dividing the key.
- Further improvement are possible by optimizing logical functions for S-Boxes

Conclusion

- CUDA is widely supported (forums, big community, SW updates,...)
- No need for special skills and training to develop on GPU
- Different data representation can bring speedup (bitslice)
- FPGA based solutions-more expensive HW and DEVELOPMENT (SW, testing, developers,..)
- GPU technology is significantly cheaper and more flexible, but still slower than dedicated HW

Thank you for attention!

Miroslav Monok

monokmir@fit.cvut.cz