A Parallel Version of the MiNa's Quantum Dot Cellular Automata IDE - QCADesigner

Riccardo Cattaneo, Giuseppe Chindemi

Politecnico di Milano
HPPS

June 22 2010

Why?

- Si based technology is reaching its physical limits due to aggressive miniaturization, progressively increasing packaging densities, clock distribution and dissipation issues
- Alternatives are being studied right now in order to overcome these limits (different technologies)
- QCA cells are one of these
- QCADesigner is the most mature tool for layouting and simulating QCA cells based circuits
- QCADesigner is unreasonably slow to simulate more bigger-than-toy circuits

QCA cells? What?

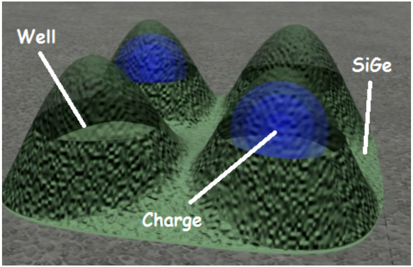


Figure: QCA cells: physical view.

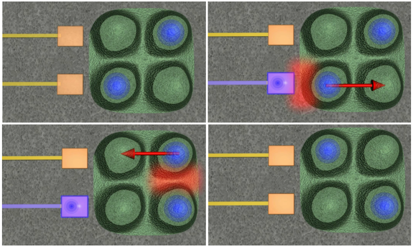


Figure: QCA cells: evolution of local state.

QCA are CA: evolution depends on previous status of cell itself and neighborhood

The Problem

QCADesigner is slow... too slow

- A common MUX can take more than 4 hours to be simulated!

How to improve the performance of QCADesigner?

- Parallel programming on GPUs with Nvidia Cuda

Cuda Overview

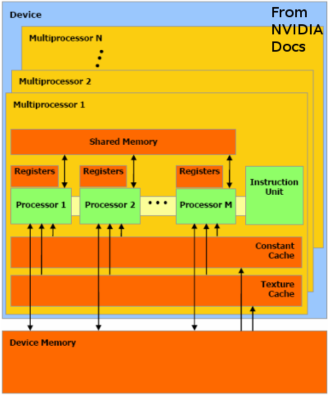
What is Cuda?

- It is a software layer that allow programmers to exploit the capability of Nvidia GPUs as general purpose processors

Why Cuda for QCAD?

- Because GPUs offer parallelism and QCAs are parallel by nature
- Because GPUs are specialized in FP operations
- Because GPUs offer the lowest price per core

GPU Logical Organization and Programming Model



From NVIDIA Docs

Figure: Cuda GPUs: A MIMD Array of SIMD processors

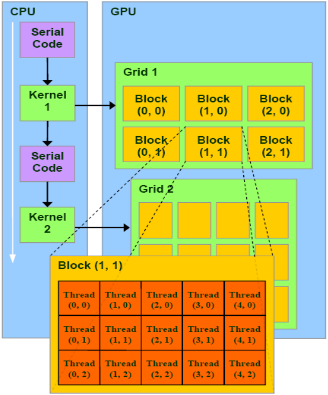


Figure: Cuda GPUs: Heterogeneous Programming

Implementation Overview

QCADesigner

- Every cell - one thread approach
- Additive error when evolving system
- Time complexity: $O(2^i * n * b)$

CudaQCADesigner

- One cell - one thread approach
- No additive error when evolving system
- Time complexity: $(\frac{2^i * n * b}{T})$ where T is the number of running threads

Implementation

QCADesigner Every cell is simulated one after the other even though they could be evolved in parallel

CudaQCADesigner Every thread is responsible for the evolution of its cell. The larger the number of running threads, the better the performances (upper bound: T=number of cells in the layout)

Implementation

choiches

Tests Description

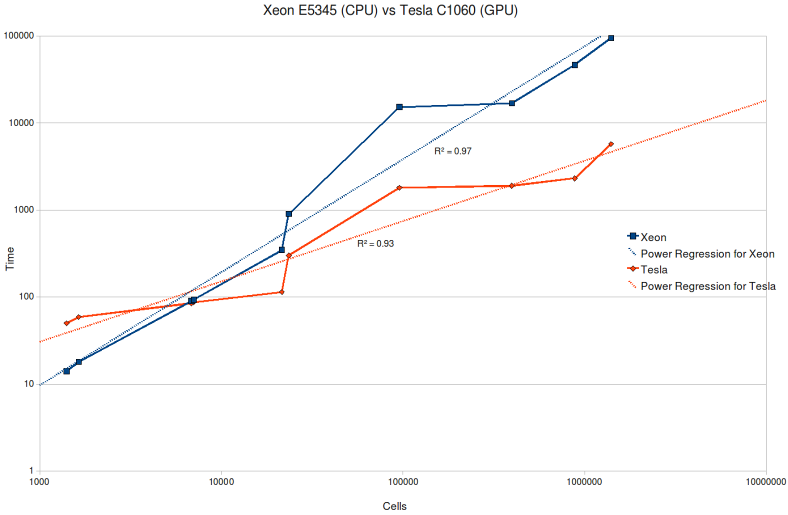
The "Lucifero" Workstation

CPU	Intel Xeon E5345
GPU	Nvidia Testa C1060

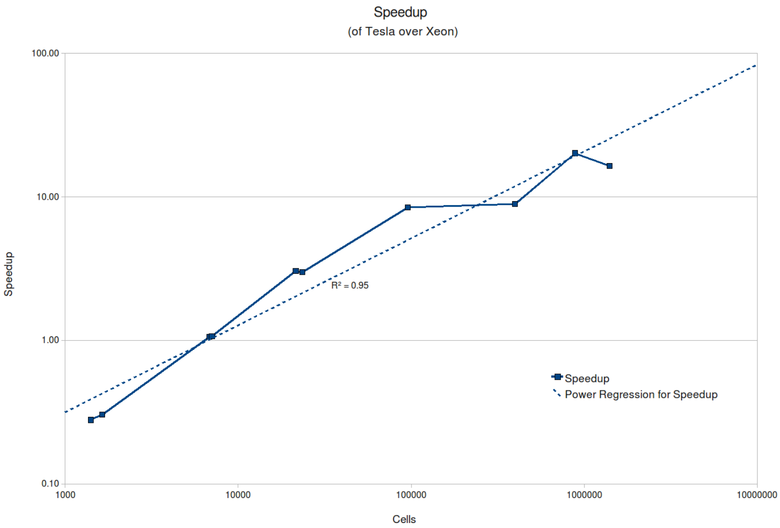
Which Tests?

Test 1	QCAD vs CudaQCAD
Test 2	CudaQCAD Profiling

Test 1: QCAD vs CudaQCAD



Test 1: QCAD vs CudaQCAD



Test 2: CudaQCAD Profiling - Memory Transfert Rate

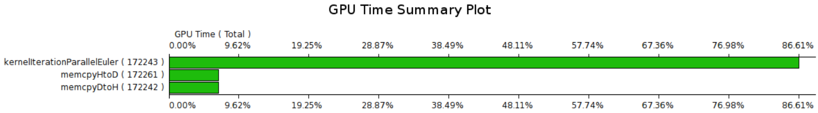


Figure: Memory Tranfer for NAND circuit (1642 cells)

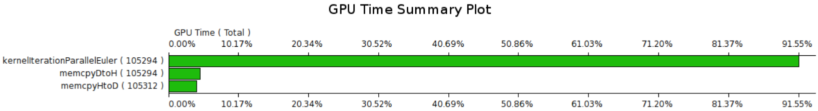
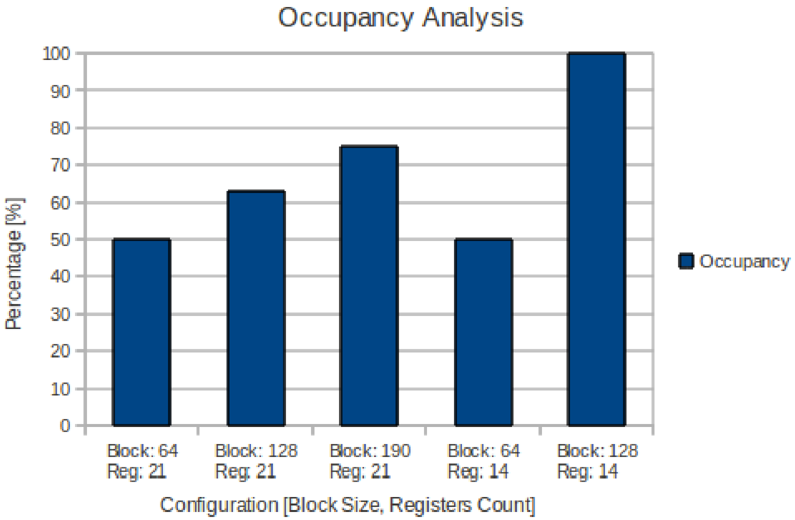


Figure: Memory transfers for MUX42 circuit (21551 cells)

Test 2: CudaQCAD Profiling - GPU Occupancy



Conclusions

OBJ 1: Design a QCA simulator faster than QCADesigner

- CudaQCADesigner can significantly outperform QCADesigner for big circuits

OBJ 2: Produce a good Software

- CudaQCADesigner well exploits GPU resources

Questions?

Questions?