ESD Risk Mitigation Using CUDA-accelerated Algorithm for MOSFET Connectivity Checking

Emanuel Dogaru, PhD[1]; Lenuta Pirghie[1]; Jean-Michel Akre, PhD[2]
[1]**AMSIMCEL**, 480A Vicovu de Jos, Suceava, Romania, www.amsim.ro
[2] **Innovanalyse**, 41 Avenue des Myosotis, 93220 Gagny, France, www.innovanalyse.com
emanuel.dogaru@amsimcel.com; lenuta.pirghie@amsimcel.com; jmakre@innovanalyse.com

ABSTRACT

The design of state of the art Integrated Circuits (ICs) in advanced technologies requires complex verification scenarios, among which **Electrostatic Discharge (ESD)** checking has become an important topic. The increasing complexity of ICs through multiple power domains and the use of IP blocks from various vendors require a comprehensive ESD protection strategy, not only at the IC package boundary but all over the chip. Further, the ESD protection scheme often changes during design phases, to accommodate tight design constraints and meet the required ESD standards. Since it is critical to find all ESD design mistakes as early as possible in the IC design cycle, ESD verification is required to run both in the pre-layout and layout assembly phases. ESD verification is executed by specialized **Electronic Design Automation (EDA)** tools. We propose a novel ESD verification algorithm and the toolset implementation that benefits from the **multi-GPU CUDA** acceleration for very fast time to results. An ESD application for the inter-domain ESD risk mitigation is presented to illustrate the preliminary results of our research.

MOTIVATION

ESD in Advanced Technologies

In the conception of nowadays Integrated Circuits (ICs) in advanced technologies like FinFET or ultra-thin gate oxide (<10nm) [1] ESD failures become more and more the bottleneck, resulting in costly re-spins.

It is commonly accepted that the ESD standards are more difficult to meet as the technology size is shrinking [2] [3].

Nowadays it is not sufficient to verify that on-chip ESD protections are in place, but a very comprehensive design and verification strategy for ESD is a must, since most of core devices in a chip are at risk. This is due to the use of sensitive elements (ultra thin-oxide transistors, ultra-shallow junctions, narrow and thin metal layers), and also due to SoCs complexity that use multiple power domains and IP blocks from various vendors.

The following are the three types of ESD events and corresponding ESD qualifications needed to prevent ESD failures [1]:

ESD	Type	Main target
HBM	Human Body Model	IO/power pads, on-chip protections
CDM	Charge Device Model	Core devices, internal protections
MM	Machine Model	Pads, on-chip protections

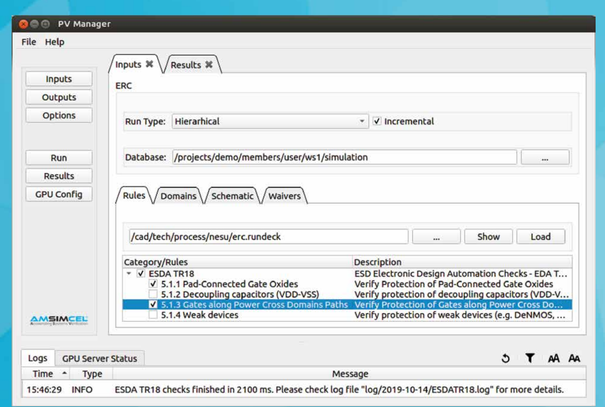
CUDA-accelerated ESD Verification

Despite the rising cost of IC development, EDA tools and mask sets, semiconductor companies continue to use the most advanced CMOS technologies for high performance applications. These incur some unique challenges, like i.e. **CDM ESD** risk rising exponentially [3] for the very large ICs with transistor count in billions. Indeed, since all core devices located at the edge of power domains are potentially at risk, design teams face the challenge to verify all power domains interfaces that can easily outnumber multi-million devices.

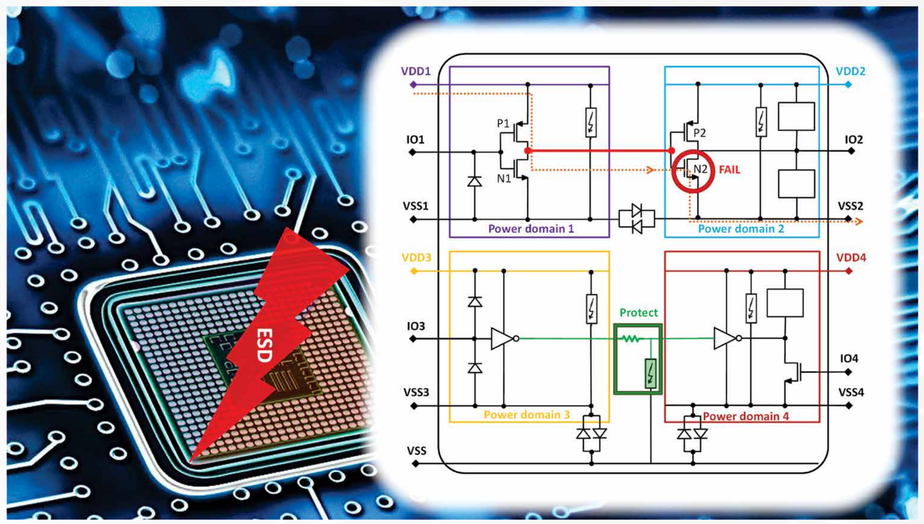
With recurring changes in the schematics and layouts to accommodate design constraints or architecture changes, a comprehensive ESD verification is virtually impossible to carry, due to runtime limitations of the existing tools. Based on the previous research in the field [4] it is known that for these specific applications, the **General Processing(GP) GPU** can yield several orders of magnitude higher performance than a conventional CPU.

We studied the potential of porting an ESD verification algorithm to an NVIDIA-based datacenter and as a result we propose a parallel approach that relies on multi-GPUs CUDA acceleration for very fast time to results.

Our algorithm and toolset **(AMSIMCEL PVManager)** can run at each IC design change and ensure that all circuitry at the domains interfaces is correctly protected.



CUDA ESD ALGORITHM



Experiment Description

In order to test our approach we decided to implement **ESDA TR18, check 5.1.3** [1] (note however that most of the other checks follow the same recipe). Check 5.1.3 is intended to verify the presence of protections on signals that cross a power domain boundary. For example, a violation of this **CDM ESD** rule is shown in the figure above: when the supply pad VDD1 is struck with respect to the ground pad VSS2, a high voltage could be developed across the gate-source oxide of the NMOS N2 in the VDD2 power domain. To implement this rule, we start by identifying the ESD protection mechanisms: to protect a device at the power domain edge we need to ensure that the voltage across it does not exceed the set failure level. This can be achieved by using i.e. an antiparallel diodes protection mechanism or even by increasing the RC constant of the node to be protected.

Implementation Details

Data structures (Spice Netlist representation):

- **List of Nets** - each net having information about the ports it connects to
- **List of Ports** - each port holds information about the device it is attached to and the nets to which is connected
- **List of Devices** - each device has information about the ports it exposes

Algorithm Description

Connectivity algorithm in three steps:

1. Parse the Spice netlist and flatten it in order to build the flat data structures
2. Find all devices at the power domain edge that are at risk of being affected by CDM ESD. This is a more generic step that will provide a power map potentially needed by all ESD checks. The problem can be solved by mapping every net to a power domain. More specifically, the task that needs to be solved is one of distributing power sources information across the circuit.

```
For every power source from devices list
   Distribute power through all devices according to a pre-defined device model (e.g. resistor, MOS transistor, etc.)
   Stop when each net has at least one power domain associated.
End for
```

3. For each such device, check if a protection mechanism is in place. If not, mark the node as violation and indicate the devices at risk. Once we have a power domain map finding all devices that cross a power domain border is a simple matter of filtering. This is a highly parallelizable task: checking for a protection mechanism for every device at risk can be done on a separate thread of a CUDA device.

Memory management

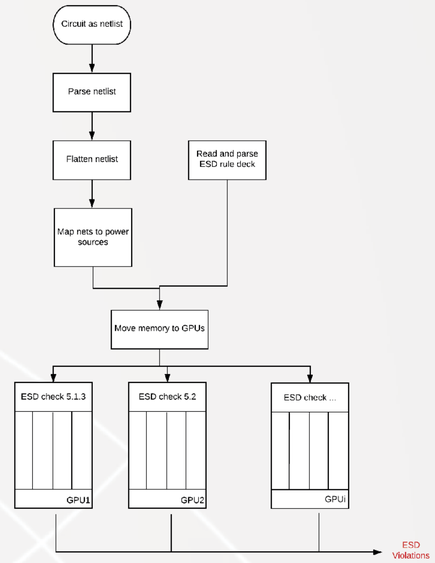
- Memory allocation: expensive -> Allocating memory individually (device by device, or net by net) for a list that can easily grow up to millions of elements, is undoubtedly too expensive => fragmented non-optimized, not localized memory that will limit from using the entire memory storage efficiently.
- Memory copy: memory will be copied between the CPU and GPU element by element, which will further increase execution times, canceling the performances gained by GPU execution.
- Solution: **Custom Memory Allocator** which has the role of creating all lists of elements (nets, ports, devices) in pre-allocated, contiguous memory blocks. This ensures: all elements of the same type will be created in contiguous blocks of memory, free element allocation and fast transfer between CPU and GPU.

RESULTS

The check 5.1.3. previously described was implemented and executed both on a classic CPU as a single thread application and on an NVIDIA driven GPGPU.

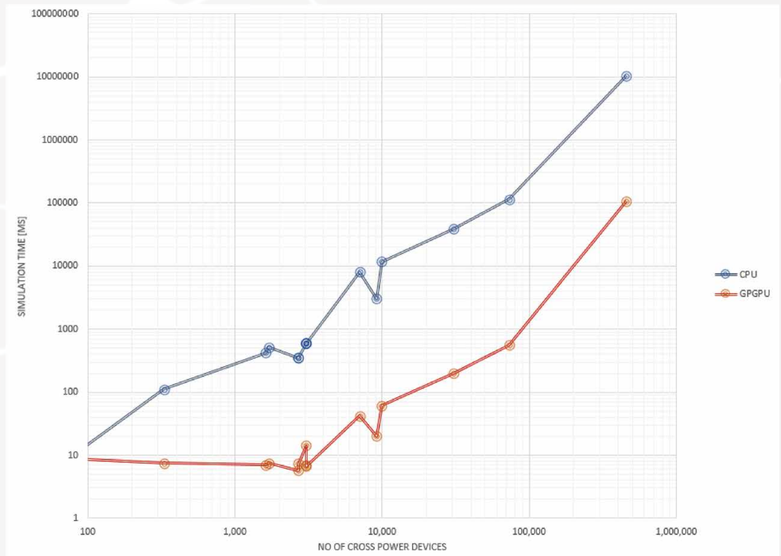
- **System configuration:** Supermicro® 7048GR-TR Workstation with 4 x NVIDIA® GTX 1080Ti, 2 x Intel® Xeon® E5-2603
- **Data:** the performance of the actual ESD check (step 3 of the algorithm) as this is the step ported to GPU. Tested circuits of several sizes, the measurement of interest being the number of devices crossing the power domain border.
- **Comparison:** the algorithm execution times (in ms) for CPU algorithm (in blue) and GPGPU algorithm (in red).
- **Results:** for small circuits the performances are similar, however as the size of the circuits grows, the GPU-based verification can run even 100 X faster. The results are consistent for circuits of different types and sizes.

On top of this performance gain, one could consider a second level of parallelization where other checks can be handled simultaneously by other GPUs from the datacenter, practically dividing the ESD checking time.



CONCLUSIONS

- In this poster we presented an ESD check algorithm designed for GPU that performs up to 100 times faster than a classic CPU based implementation.
- AMSIMCEL approach is scalable to multiple GPU devices, can handle circuits with multi-million to billion transistor counts and can be adapted to most of the ESD checks needed by the industry.



REFERENCES

[1] A. Cester; S. Gerardin; A. Tazzoli; A. Paccagnella; E. Zanoni; G. Ghidini; G. Meneghesso (2005). ESD induced damage on ultra-thin gate oxide MOSFETs and its impact on device reliability. IEEE International Reliability Physics Symposium Proceedings. 84 - 90. 10.1109/RELPHY.2005.1493068.

[2] JEDEC (2014). Human Body Model (HBM) Qualification Issues, JEDEC ESD Technical Tutorial. Online: https://www.jedec.org/sites/default/files/IndustryCouncil_HBM_January2014_JEDECversionMay2014.pdf

[3] JEDEC (2014). Charged Device Model (CDM) Qualification Issues, JEDEC ESD Technical Tutorial. Online: https://www.jedec.org/sites/default/files/IndustryCouncil_CDM_January2014_JEDECversionMay2014.pdf

[4] C. Tugui; G. Donici; E. Dogaru; J-M. Akre (2018). CUDA Accelerated Hybrid Electrical Rule Checking for Floating MOSFET Gates, NVIDIA GPU Technology Conference Europe (GTC 2018), Munich, Germany.

[5] EDA Tool Working Group (2011). ESD Electronic Design Automation Checks (ESD TR18.0-01-11). New York: Electrostatic Discharge Association.