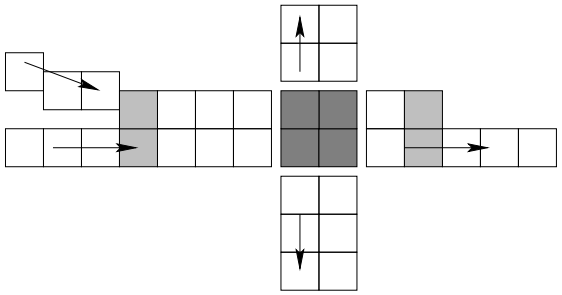
CA-based Traffic Simulation on FPGAs



Matthew S. Nassr, Henning S. Mortveit, CCS-5
Justin L. Tripp, Maya Gokhale, ISR-3
Los Alamos National Labs

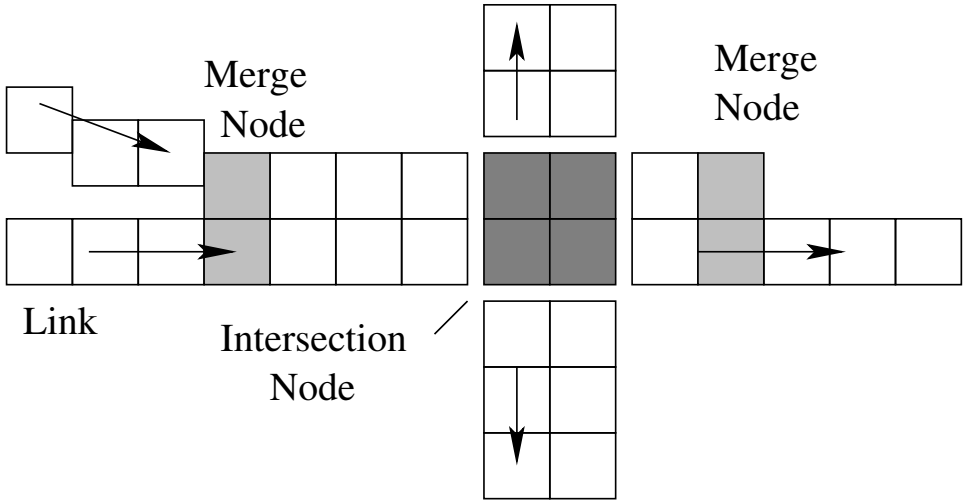
Goals for this Research

- Explore the use of FPGAs in accelerating large simulations
- Increase understanding about trade-offs for very large designs
- Use FPGAs for non-traditional FPGA computation

TRANSIMS: Traffic Simulation

- Micro-simulation using cellular automaton computation on an unstructured grid.
- Road network of nodes and links.
  - ⋆ Nodes - intersections and merge points
  - ⋆ Links - one or more parallel lanes of cells
- Cars
  - ⋆ Four basic rules describe cellular behavior.
  - ⋆ discrete speeds $v \in \{0, 1, 2, 3, 4, 5\}$
  - ⋆ updated once per second.

Example Structure



All clear cell groups are each links. Light gray cells are merge nodes. Dark gray cells belong to an intersection node.

Multi-Lane Traffic Simulation Rules

Rule #1 – *Lane Change Decision:*
When time t is odd and if cell i has a car and a left lane change is:

- *desirable* (car can go faster in target lane),
- *permissible* (there is space for a safe lane change)

flag the car/cell for a left lane change. Similar for even time t.

Rule #2 – *Lane Change:*

When time t is odd and If the car in cell i is flagged for a left lane change clear cell i. Otherwise, if the right neigbor of cell i is flagged for a left lane change then move the car from the neighbor cell to cell i with probability p_α.

Multi-Lane Traffic Simulation Rules

Rule #3 – *Velocity Update:*
Each cell i that has a car updates the velocity using the two-step sequence:

- $v := \min(v + 1, v_{\max}(i), \Delta(i))$ (acceleration)
- If $[\mathrm{UniformRandom}() < p_{\mathrm{break}}]$ and $v > 0]$ then $v := v - 1$ (stochastic deceleration).

Rule #4 – *Position Update:*

If there is a car in cell i with velocity $v = 0$ do nothing. If $v > 0$ then clear cell i.

Otherwise, if there is a car $\delta(i) + 1$ cells behind cell i and the velocity of this car is $\delta(i) + 1$ then move this car to cell i.

Two Approaches

Structural

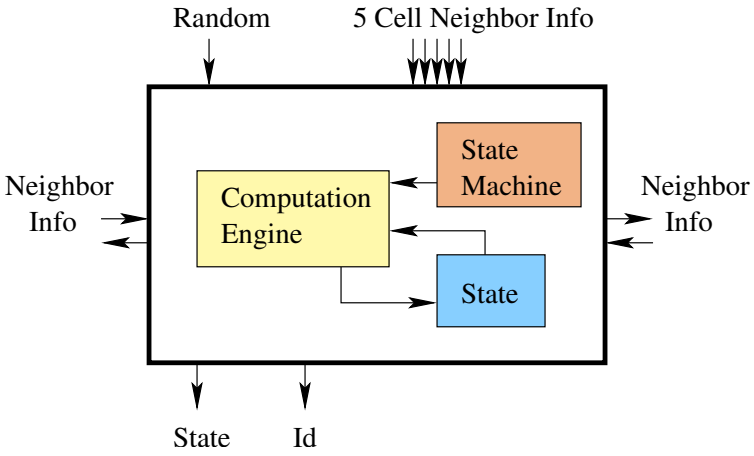
- Entire map is built in hardware
- Updates car next state concurrently
- Cars are assigned next state and moved
- High communication between FPGAs
- Low communication between host and FPGAs

Streaming

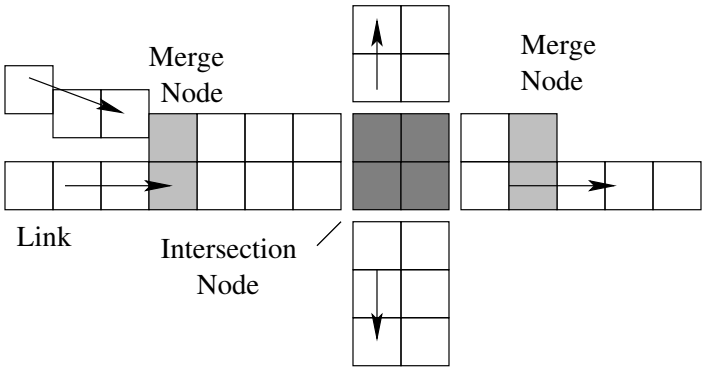
- Map is streamed in and analyzed in pieces
- Updates road cell next state sequentially
- Car is assigned next state, then moved later when destination cell is processed
- High communication between host and FPGAs

Structural Implementation

Individual Cell



Road Network



Structural Design Results for FPGAs

	One-lane		Two-lane	
	XC2V6k	XC2VP100	XC2V6k	XC2VP100
Cells	650	650	400	400
LUTs/Cell	104	97	169	175
Clock(MHz)	48.68	64.17	35.53	54.43
Slices	33790(99%)	31576(71%)	33790(99%)	34999(79%)

The one-lane implementation has no hardware for calculating lane changes, since there are no lanes to change to. The two-lane implementation includes lane changing hardware.

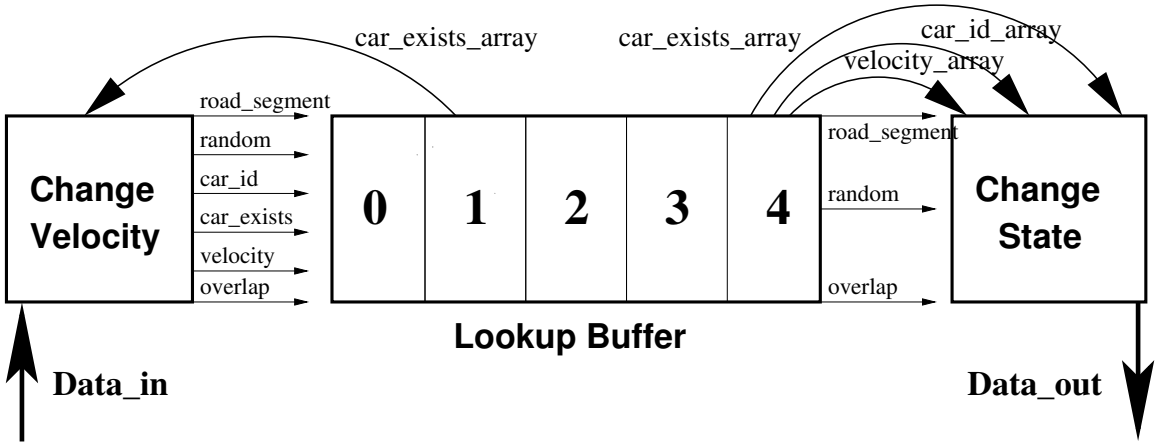
Comparison for Two-lane Implementations

	XC2V6k	XC2VP100	1.7GHz Xeon	3.0GHz Xeon
Cells	400	400	600	600
Cells/sec	2.37×10^9	3.64×10^9	7.82×10^6	1.35×10^7
Speed Up	$303.1\times$	$651.1\times$	$1.0\times$	$1.73\times$

- FPGAs provide more computation power than CPUs.
- Structural approach would require 12,400 FPGAs to simulate the city of Portland (6.2 million road cells).
- Using the streaming approach could reduce the number of FPGAs required.

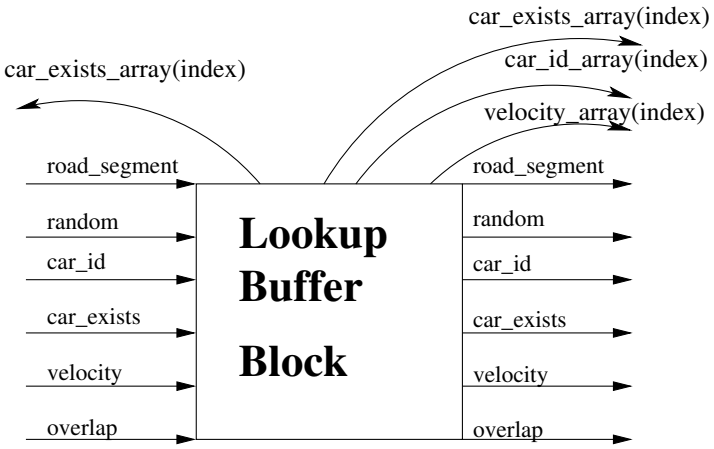
Streaming Approach

Streaming element:



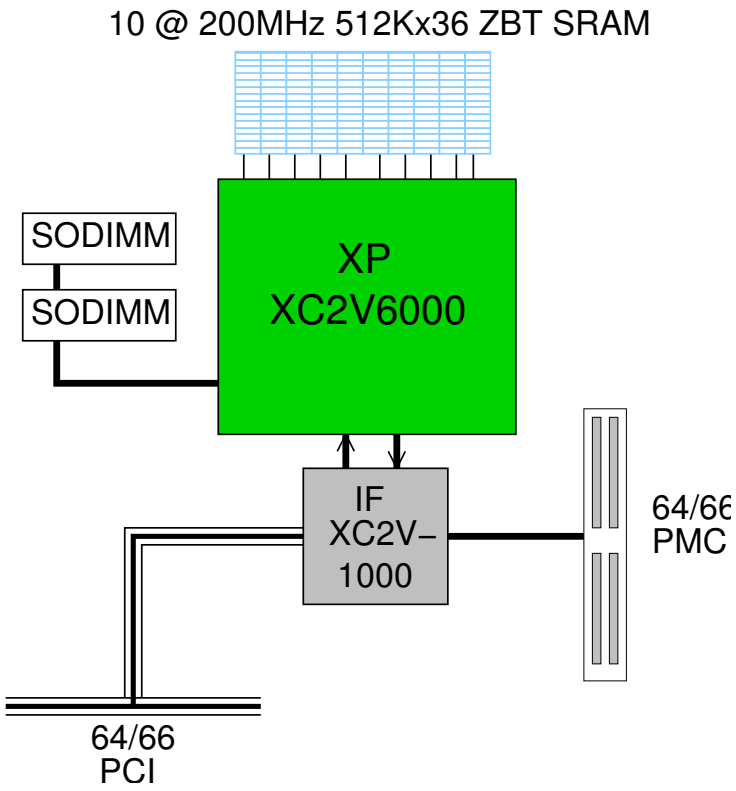
- Streaming allows for an unrestricted map size by processing the road cells in a continuous stream
  - ⋆ Multiple streaming elements are used on a single map
  - ⋆ Streaming elements are sized based on the maximum allowable velocity

Streaming Approach – *continued*



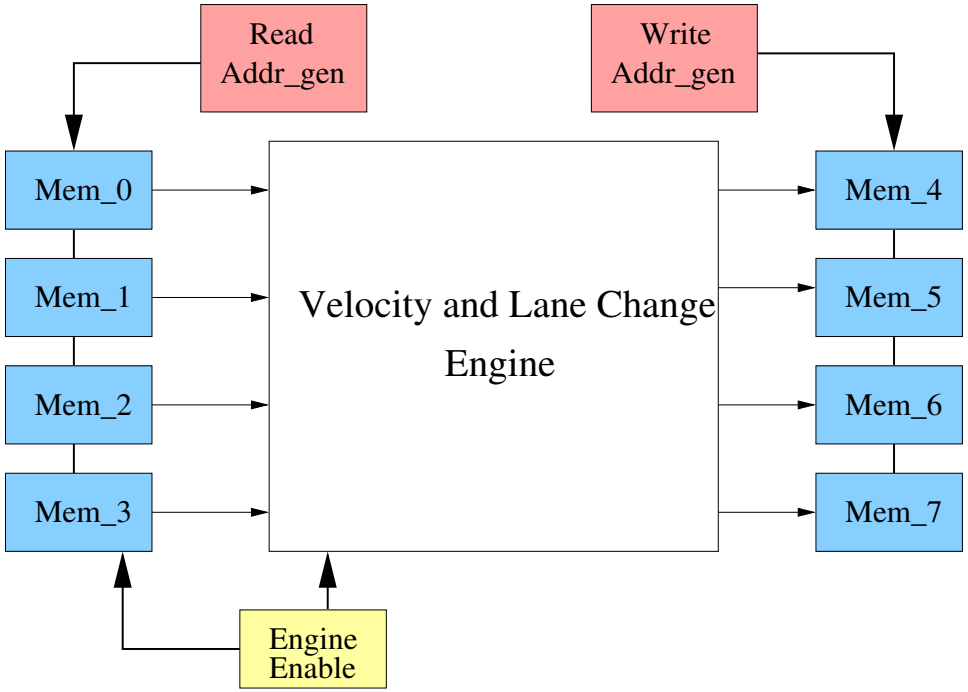
- Single road cell information is passed into the 0th and last element of the lookup buffer
- Multiple road cell information is passed to change velocity and change state in data arrays
- Lookup buffer is made up of (max velocity) lookup blocks, each with 6 inputs and 10 outputs

Mapping to Osiris FPGA Board



The streaming approach was mapped to the Osiris board. Osiris has a Virtex-II 6000, with 10 ZBT SRAMs and a pair of SODIMM sockets. The IF chip provides high speed DMA to and from the PCI bus and the PMC daughter card.

Simple Implementation on Osiris



- Road cell data is read from 4 input SRAMs (4 lanes), updated, and written to 4 output SRAMs
- The update engine is only running when valid data is present.

Data Streaming Processing Rates

	XC2V6k	XC2VP100
Clock(MHz)	61.1	96.6
Slices	2167	2095
Slices/Cell	541	523
Occupied Slices (%)	6%	4%

The first streaming approach uses a small portion of the board, but still manages a significant improvement in processing rate. Four road cells are calculated every clock cycle.

Streaming Comparison With Microprocessors

	XC2V6k	XC2VP100	1.6GHz Xeon	3.0GHz Xeon
Cells/sec	2.44×10^8	3.86×10^8	7.82×10^6	1.35×10^7
Speed Up	$31.2\times$	$49.4\times$	$1.0\times$	$1.73\times$

Although the data-streaming approach is $9.5\times$ slower than the structural, the cost in FPGAs is 12,400 to 1. The streaming approach is viable and is an inherently scalable scheme.

Scaling Issues

Structural vs. Streaming

- Scalability - Structural method cannot scale beyond a small number of road cells due to the high cost in hardware.
- Parallelism - Streaming trades area for sequential execution, but at a low cost in speed.
- Bandwidth - Streaming is bandwidth limited from the host (PCI)
- Cost - Structural has prohibitive cost for FPGAs, but may be possible for nanostructures.

Conclusions

- FPGAs can scale with large simulation requirements (millions of simulation elements).
- Streaming has been integrated into an Osiris board and can accelerated straight-lane traffic simulation by almost $50\times$.
- FPGAs can accelerate large simulations using custom calculations.