AN FPGA-BASED ARCHITECTURE TO SIMULATE CELLULAR AUTOMATA WITH LARGE NEIGHBORHOODS IN REAL TIME

NIKOLAOS KYPARISSAS, APOSTOLOS DOLLAS

School of Electrical and Computer Engineering
Technical University of Crete, Chania, Greece
nkyparissas@isc.tuc.gr, dollas@ece.tuc.gr

FPL 2019 – Sept 9 – Barcelona, Spain
The Hodgepodge Machine with a 29X29 neighborhood

...but, the Cellular Automaton which is commonly known as the Hodgepodge Machine is really the Belousov-Zhabotinsky Reaction “a classical example of non-equilibrium thermodynamics, resulting in the establishment of a nonlinear chemical oscillator”
Example: The Hodgepodge Machine

- Normally a $q$-state CA with a $3 \times 3$ Moore neighborhood
- Extended to a CA with a $29 \times 29$ Moore neighborhood
- A cell can be “healthy” (state 0), “infected” (states 1 to $q-1$) or “ill” (state $q$). In our example: $q = 255$.

The cell’s transition function is defined as:

$$c_{t+1}(i, j) = \begin{cases} 
\frac{\text{number of infected and ill neighbors}}{k} & \text{if } c_t(i, j) = 0 \\
0 & \text{if } c_t(i, j) = q \\
\frac{\text{sum of all neighbors}}{\text{sum of infected neighbors}} + g & \text{otherwise}
\end{cases}$$
SIMULATION EXAMPLES

Example:
The Greenberg-Hastings Model
with 16 states per cell.

1. $r = 1$ Von Neumann,
2. $r = 14$ von Neumann,
3. $r = 14$ Circular

Qualitative differences:
- vortices become curved and wider.
Example:
Anisotropic Rule
with 256 states per cell,
r = 14 Moore
1. 1 generation
2. 120 generations
3. 500 generations
4. 10000 generations

- Self-organization properties
- Not possible with small, r = 1 neighborhoods
NEW CAPABILITIES

Example:
The Hodgepodge Machine
with 256 states per cell.

1. \( r = 1 \) Moore,
2. \( r = 9 \) Moore,
3. \( r = 14 \) Moore

Qualitative differences:
- Vortices become wider
- Small, stable, vortex-like patterns located in the center of the larger vortices
FPGAS AND CELLULAR AUTOMATA: A VERY OLD (BUT CHANGING) STORY

1. Toffoli and Margolus’s Cellular Automata Machines (CAM): 1980s and 1990s
   - Streaming architecture using LUTs to calculate the transition function

2. Cellular Processing Architecture (CEPRA): 1990s
   - Streaming architecture using arithmetic logic to calculate the transition function

   - Implementing the CA as an array of Processing Elements (PE) within the FPGA

   - A streaming architecture using an array of PEs to calculate the CA

5. Many other significant projects since then, most of which have been custom to a specific CA rule without the use of large neighborhoods
FPGAS AND GPU’S – CROSSED AT 11 X 11

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Neighborhood Size</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Margolus, 1993-2001, CAMs</td>
<td>experimented with up to 11x11</td>
<td>10 gen./sec for a 512x512 grid with 3-bit cells</td>
</tr>
<tr>
<td>Gibson et al., 2015, Workstation with Nvidia GTX 560 Ti</td>
<td>experimented with up to 11x11</td>
<td>≈ 65x over serial for Game of Life on a 2048x2048 grid</td>
</tr>
<tr>
<td>Millan et al., 2017, Nvidia TitanX GPU</td>
<td>experimented with up to 11x11</td>
<td>21.1x over serial for Game of Life on a 4096x4096 grid</td>
</tr>
<tr>
<td>Kyparissas &amp; Dollas, 2019, Artix-7 FPGA</td>
<td>experimented with up to 29x29</td>
<td>51x over serial for the Hodgepodge Machine on a 1920x1080 grid</td>
</tr>
</tbody>
</table>

- FPGAs: “game changer” as far as large-neighborhood CA are concerned
- Today’s FPGAs can simulate complex rules with very large neighborhoods on very large grids
# PERFORMANCE RESULTS (WITH A MODEST FPGA)

<table>
<thead>
<tr>
<th>Cellular Automaton</th>
<th>i7 – 7700 HQ, 1000 generations</th>
<th>Our Design, 1000 generations</th>
<th>Speedup of Our Design</th>
</tr>
</thead>
<tbody>
<tr>
<td>Artificial Physics, 21 x 21</td>
<td>538.77 sec</td>
<td>16.67 sec</td>
<td>32x</td>
</tr>
<tr>
<td>Greenberg-Hastings Model, 29 x 29</td>
<td>469.58 sec</td>
<td>16.67 sec</td>
<td>28x</td>
</tr>
<tr>
<td>The Hodgepodge Machine, 29 x 29</td>
<td>851.29 sec</td>
<td>16.67 sec</td>
<td>51x</td>
</tr>
</tbody>
</table>
DESIGN AND ARCHITECTURE

For a kXk neighborhood applied to a nXn data grid:

- \((k-1)Xn + k\) input data points on-FPGA
- kXk weights on-FPGA
- Rules compiled in w/ a tool
- Each piece of data enters FPGA once
- kXk parallelism

System specifications:

- Initialization via UART / USB
- 1080p Full-HD Graphical Display
- Datapath running at 200 MHz
DESIGN AND ARCHITECTURE

The CA Engine’s Buffer:
- Receives memory bursts at 81.25 MHz
- Sends cells at 200 MHz
- Each cell needs to enter the FPGA only once per CA generation
# Resource Utilization

<table>
<thead>
<tr>
<th>Resource</th>
<th>Utilization</th>
<th>Utilization %</th>
</tr>
</thead>
<tbody>
<tr>
<td>LUT</td>
<td>20375</td>
<td>32.14</td>
</tr>
<tr>
<td>LUTRAM</td>
<td>1555</td>
<td>8.18</td>
</tr>
<tr>
<td>FF</td>
<td>27224</td>
<td>21.47</td>
</tr>
<tr>
<td>BRAM</td>
<td>65</td>
<td>48.15</td>
</tr>
<tr>
<td>DSP</td>
<td>1</td>
<td>0.42</td>
</tr>
<tr>
<td>IO</td>
<td>73</td>
<td>34.76</td>
</tr>
<tr>
<td>BUFG</td>
<td>7</td>
<td>21.88</td>
</tr>
<tr>
<td>MMCM</td>
<td>3</td>
<td>50</td>
</tr>
<tr>
<td>PLL</td>
<td>1</td>
<td>16.67</td>
</tr>
</tbody>
</table>
This video is from the 2018 Xilinx Hardware Design Competition.

- The neighborhood is not yet 29X29 but the design process remains the same.
- This design placed in the top-12 among more than 100 entries, however it has not been published to date.
- The example is from Artificial Physics.

Xilinx Open Hardware 2018 Design Contest

A Parallel Framework for Simulating Cellular Automata on FPGA Logic

Participant: Nikolaos Kyparissas
Supervisor: Prof. Apostolos Dollas

Team number: XOHW18-220

https://github.com/nkyparissas/XOHW18