Power and Area Efficient Sorting Networks using Unary Processing

M. Hassan Najafi, David J. Lilja, Marc Riedel and Kia Bazargan

Najaf011@umn.edu

ICCD 2017, Boston, MA, USA
Overview

• Introduction
  – Sorting, sorting networks: Batcher network
  – Unary processing, unary streams, unary operations

• Complete Sort System
  – Conventional design vs. Unary design
  – Design evaluation

• Highly Efficient Median Filters
  – Circuit design
  – Time-based unary design

• Conclusions
Introduction

• Sorting
  – an important task in different applications
    • Data mining and databases to image and signal processing.

• High performance sorting -> in hardware, but
  • Limited chip area
  • Low-power expectation

• Common approach of hardware sorting
  • Batcher network: a network of compare-and-swap (CAS) units.
    • the lowest latency for hardware-based sorting.

• Hardware and power cost depend on:
  • # of CAS blocks, cost of each CAS block
Introduction

• Sorting

The schematic symbol of a CAS block
  a) ascending b) descending

The CAS network for an 8-input batcher bitonic sorting (24 CAS blocks)

• 16-input, 32-input, and 256-input bitonic sorting networks
  – 80, 240, and 4,608 CAS blocks
Introduction

• **Stochastic computing (SC)**
  – Processing of uniformly distributed random bit-streams
    • **Stochastic bit-stream**: 1011011100 → 0.6
    – **Approximate** computation due to random fluctuation

• **Unary Processing**: An evolution of the idea of SC
  – **Deterministic** computation, completely accurate results
    • **Unary bit-stream**: 1111110000 → 0.6
    • **Unary operations**:
      - Maximum (AND gate)
      - Minimum (OR gate)
      - Absolute value subtraction (XOR gate)
      - Comparison (D-type flip flop)
Complete Sort System

- **Conventional design vs Unary design**
  - Sorting networks are made of CAS blocks

  ![Hardware implementation of a CAS block](image)

  - A much **cheaper** CAS block in **unary** design
  - But additional overhead
    - **Conversion units** to convert data between binary and unary format
    - **Longer processing time** \((2^M \text{ cycles for M-bit resolution})\)
Complete Sort System

- **Design evaluation**
- **Synthesis results for complete sorting networks, data-width of 8-bit**

<table>
<thead>
<tr>
<th># of inputs and outputs</th>
<th># of CAS units</th>
<th>Area ($\mu m^2$)</th>
<th>Critical Path (ns)</th>
<th>Power (@max f) — (@50MHz) (mW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>24</td>
<td>3,086</td>
<td>2,194</td>
<td>1.85</td>
</tr>
<tr>
<td>16</td>
<td>80</td>
<td>10,534</td>
<td>4,511</td>
<td>2.73</td>
</tr>
<tr>
<td>32</td>
<td>240</td>
<td>32,508</td>
<td>9,235</td>
<td>4.06</td>
</tr>
<tr>
<td>64</td>
<td>672</td>
<td>90,691</td>
<td>19,028</td>
<td>5.71</td>
</tr>
<tr>
<td>128</td>
<td>1,792</td>
<td>242,049</td>
<td>33,916</td>
<td>7.49</td>
</tr>
<tr>
<td>256</td>
<td>4,608</td>
<td>586,456</td>
<td>74,719</td>
<td>9.71</td>
</tr>
</tbody>
</table>

- With **unary** design
  - A hardware area saving of up to **87%**.
  - A lower power consumption and critical path latency
- But **total latency = CP latency x number of clock cycles**
  - A much **higher total latency**
  - Leads to a **higher energy consumption** than binary designs
Highly Efficient Median Filters

- **Median filter** a popular non-linear filter widely used in image and signal processing applications

- High computational complexity makes their hardware implementation expensive and inefficient for many applications.

- Sorting network-based architecture, a common approach for hardware implementation of median filters

- The CAS network for a 3x3 Median Filter
Highly Efficient Median Filters

- **Time-based unary design**

  ![Time-encoded unary signal](image)

  - Digital unary stream: 1111111000000000000000

- Time-encoding the input data to address the long latency of processing using unary circuits

  - 1) Convert the sensed data to a time-encoded pulse signal
    - Using an analog-to-time converter (e.g. a pulse width modulator)
  - 2) Process the time-encoded signal using unary circuit
  - 3) Convert back to a desired analog format
    - Using a time-to-analog converter (e.g. a voltage integrator)
Highly Efficient Median Filters

- Synthesis results of the sorting network-based median filter

<table>
<thead>
<tr>
<th>Median Filter</th>
<th>Design Approach</th>
<th>Area ($\mu m^2$)</th>
<th>Latency (ns)</th>
<th>Power (mW) (@max freq)</th>
<th>Energy (pJ)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CAS Logic</td>
<td>Total</td>
<td>Latency</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3x3</td>
<td>Conventional Binary</td>
<td>2,167</td>
<td>2.10</td>
<td>1.03</td>
<td>2.1</td>
</tr>
<tr>
<td></td>
<td>Unary-Bit-Stream-based</td>
<td></td>
<td>2.10</td>
<td>0.95</td>
<td>170.2</td>
</tr>
<tr>
<td></td>
<td>Unary-Time-based</td>
<td></td>
<td></td>
<td>1.78</td>
<td>0.69</td>
</tr>
<tr>
<td></td>
<td>Overheads</td>
<td>917</td>
<td>179.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Total</td>
<td>996</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>776</td>
<td>0.39</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>855</td>
<td>0.39</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **A better area, latency, and energy** consumption with the time-based unary design
- But the down-side is a **slight accuracy loss**.

- MAE (%) of the time-based unary circuit when processing a 128x128 sample input image
Conclusions

- Proposed an area and power efficient implementation of sorting network circuits using unary processing

- **Latency** is the main overhead (e.g. more than 100ns)

- We used a time-based unary design to address the long latency by encoding data in time

- Significant latency and energy saving, at the cost of a slight loss in accuracy
Questions?

M. Hassan Najafi
Najaf011@umn.edu