Multi-Query Stream Processing on FPGAs

Mohammad Sadoghi
Rija Javed
Naif Tarafdar
Harsh Singh
Rohan Palaniappan
Hans-Arno Jacobsen

University of Toronto

April 2012
Algorithmic Trading

NASDAQ
NYSE
TSX

Market Feeds

Market

News

Broker

Investor

Investor

Buy/Sell

Buy/Sell

Investment Strategies

Strategies (SQL):
SELECT * FROM WHERE GROUP BY

AMGN=58
ORCL=12
HON=24
MSFT=27
IBM=84
INTC=19
JNJ=58

Strategies (BE):
IBM > 145
ORCL < 10
JNJ > 60

fpga-ToPSS
(Complex) Event Processing

Data Streams

Patterns (SQL Continuous Queries)

Real-time Data Analysis

Matched Patterns

Event/Publication

Subscriptions (Boolean Expressions & XPath Expressions)

Content-based Publish/Subscribe

Matched Subscriptions
Why FPGAs

FPGA distinctive features

1. *Hardware reconfigurability*: re-configuring the application on-demand into a highly parallel custom processors

2. *Hardware parallelism*: eliminating inter-processor signalling and message passing overhead associated with the concurrency management at the program and the OS level

3. *Onboard packet processing*: using multiple high bandwidth (giga-bit) I/O pins to eliminate the OS layer latency overhead in moving data between input and output ports

4. *Cost-effective and Energy-efficient*
Overview of parallel join processing - Inter-parallelism

1. \( \text{Event}_{L1} \cdots \text{Event}_{Lk} \)
2. \( \text{Event}_{R1} \cdots \text{Event}_{Rk} \)

Remapping Block

IN Port 1

\( \text{JC}_1 \) Left Window

\( \text{JC}_1 \) Right Window

\( \text{JC}_n \) Left Window

\( \text{JC}_n \) Right Window

Port 1 OUT

1. L. Check-if-Inputs-Present JC_i
2. R. Check-if-Inputs-Present JC_i
3. Port 1 In Mux
4. Port 2 In Mux

Remapping Block

IN Port 2

4. BRAM Block - X

Demux (Dx)

\( \text{JC}_1 \) right window

\( \text{JC}_n \) right window

\( \text{JC}_1 \) left window

\( \text{JC}_n \) left window

BRAM Access Scheduling Block

L/R CIP Flags

BRAM Block-X Port 1 & 2 address vector

Mux / Demux Control

R. Check-if-Inputs-Present
Overview of parallel join processing - Coordination

1. **Existing ‘k’ Events from JC_i Event Window (BRAM)**
   - \(E_n \text{ Age} \quad n \quad \cdots \quad E_1 \text{ Age} \quad 1\)
   - Sort Circuit (Descending order of Age)
   - \(E_{i1} \text{ Age} \quad i_1 \quad \cdots \quad E_{in} \text{ Age} \quad i_n\)
   - Extract Sorted Indices
   - \(i_1 \quad \cdots \quad i_n\)

2. **Check-if-Input-Present**
   - 0/1
   - Compose X-bar Configuration Input

3. **X-bar Driver Look-up table**

4. **k events**

5. **Remapped Events to be written to JC_i Event Window (BRAM)**

---

**Note:**
- fpga-ToPSS
- UNIVERSITY OF TORONTO
- MIDDLEWARE SYSTEMS RESEARCH GROUP
Overview of parallel join processing - Intra-parallelism

Events and Right/Left Window Select Control

New events from JC, Right Event Buffer

New events from JC, Left Event Buffer

k events

[\{E_1, E_2, \ldots, E_k\}]

[\{E_1, E_2, \ldots, E_k\}]

[\{E_1, E_2, \ldots, E_k\}]

k events

[\{E_1, E_2, \ldots, E_k\}]

[\{E_1, E_2, \ldots, E_k\}]

[\{E_1, E_2, \ldots, E_k\}]

Mux

Mux

Mux

JC_i Join Condition

JC_i Join Condition

JC_i Join Condition

... 

... 

... 

University of Toronto

fpga-ToPSS
Complex Event Processing - Multi-query Processing (MQ)

Standard SPJ query plan

Global query plan for Q1 & Q2 via Rete network

\[ Q_1 \]

\[ \sigma_w \quad \sigma_x \quad \sigma_y \quad \sigma_z \]

\[ \sigma_1 \quad \sigma_1 \quad \sigma_3 \quad \sigma_3 \]

\[ \sigma_k \]

Select operator for condition ‘x’

Two-input node with join criteria ‘n’

Memory node

Join operator with condition ‘n’
Select operator for criteria ‘x’

Event data stream

Event data stream

UNIVERSITY OF TORONTO

MIDDLEWARE SYSTEMS RESEARCH GROUP

University of Toronto

fpga-ToPSS
Multi-query Processing Example - Individual Query Plans

\[
\begin{align*}
Q_1 & : \pi_j \sigma_m \pi_i \\
Q_2 & : \pi_i \sigma_k \\
Q_3 & : \sigma_x \pi_k
\end{align*}
\]
Multi-query Processing Example - Global Query Plan
Multi-query Processing Example Running on FPGA

<table>
<thead>
<tr>
<th>Company</th>
<th>Price</th>
<th>Shares</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>AOL</td>
<td>18.97</td>
<td>94.78B</td>
<td>+0.21</td>
</tr>
<tr>
<td>RIM</td>
<td>14.63</td>
<td>524.0M</td>
<td>+0.94</td>
</tr>
<tr>
<td>Yahoo</td>
<td>15.22</td>
<td>1.21B</td>
<td>-0.32</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Company</th>
<th>Volume</th>
<th>P/E Ratio</th>
<th>Mkt Cap</th>
</tr>
</thead>
<tbody>
<tr>
<td>AOL</td>
<td>1.23M</td>
<td>135.4</td>
<td>1.80B</td>
</tr>
<tr>
<td>RIM</td>
<td>6.11M</td>
<td>3.46</td>
<td>7.67B</td>
</tr>
<tr>
<td>Yahoo</td>
<td>15.52M</td>
<td>18.59</td>
<td>18.47B</td>
</tr>
<tr>
<td>SINA</td>
<td>5.39M</td>
<td>0.45</td>
<td>4.28B</td>
</tr>
</tbody>
</table>

Query Count

<table>
<thead>
<tr>
<th>Query</th>
<th>Query Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q₁</td>
<td>1</td>
</tr>
<tr>
<td>Q₂</td>
<td>1</td>
</tr>
<tr>
<td>Q₃</td>
<td>4</td>
</tr>
</tbody>
</table>
Complex Event Processing - MQ Compiler

- Query
  - Single query
  - Multiple queries
  - SPJ query plan
  - SPJ query plan
  - SPJ query plan

Standard Flow:
- Single/Multi-SPJ query optimized Rete-network

Custom Flow:
- Multi-SPJ Query mapped HDL design
- Rete – HDL Compiler
- Custom SPJ Operator HDL lib
Further Information:


**Synthetic/Real Workload Generator (BEGen):**

- [http://msrg.org/datasets/BEGen](http://msrg.org/datasets/BEGen)

**FPGA Project Web Site**

- [http://www.msrg.org/project/fpga-ToPSS](http://www.msrg.org/project/fpga-ToPSS)

**Thank You**