Streams on Wires
A Query Compiler for FPGAs

Rene Mueller  Jens Teubner  Gustavo Alonso
{firstname.lastname}@inf.ethz.ch

Systems Group
Department of Computer Science, ETH Zurich, Switzerland
Technology Trends by Moore’s Law

CPU Transistor Count
- 2x every 2 years

Feature Size
- 0.7x every 2 years

- Multicore CPUs
  - Nehalem: 4 cores, 45 nm

- GPUs
  - nvidia GT200b: 240 cores, 55 nm

- FPGAs
  - Xilinx Virtex-6: 750k logic cells, 40 nm

(adopted from Mark Bohr, ISSCC 2009)
Field-Programmable Gate Arrays (FPGAs)

FPGAs are reconfigurable digital logic devices
Use Case: Automated Financial Trading

▶ Example: Option Price Reporting Authority (OPRA)
  ▶ Aggregated market data streams from NYSE, AMEX, etc.

▶ Latency is critical! \( \mu s \rightarrow \$ \)
▶ High data volume
▶ FAST/FIX messages over UDP
▶ Automatic Trading Systems
  ▶ on small subset of symbols
  ▶ use aggregate statistics
Simplified Trade Stream and Example Queries

<table>
<thead>
<tr>
<th>Seqnr</th>
<th>Symbol</th>
<th>Price</th>
<th>Volume</th>
</tr>
</thead>
<tbody>
<tr>
<td>2246</td>
<td>UBSN</td>
<td>622</td>
<td>47</td>
</tr>
<tr>
<td>2247</td>
<td>NOVN</td>
<td>4637</td>
<td>403</td>
</tr>
<tr>
<td>2248</td>
<td>NESN</td>
<td>2842</td>
<td>166</td>
</tr>
<tr>
<td>2249</td>
<td>UBSN</td>
<td>608</td>
<td>13</td>
</tr>
<tr>
<td>2250</td>
<td>NOVN</td>
<td>4736</td>
<td>118</td>
</tr>
<tr>
<td>2251</td>
<td>ABBN</td>
<td>2505</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Queries:

$Q_1$:
SELECT Price, Volume
FROM Trades
WHERE Symbol = "UBSN"
AND Volume > 100000

$Q_2$:
SELECT Symbol,
    avg(Price) AS AvgPrice
FROM Trades [SIZE 600
    ADVANCE 60 TIME]
GROUP BY Symbol
System Architectures

(a) FPGA in data path
Network Adapter →
Circuit → CPU Adapter

(b) FPGA co-processor
CPU Adapter → Circuit
→ CPU Adapter
Processing Packets at Wire Speed

- Synthetic variable rate stream
  - 16 byte/tuple
  - 64 bytes/Ethernet frame
  - 1,000,000 frames/s → 51% 1 GbE
- Network Adapter (IP/UDP Engine) on FPGA
- PC System drops packets (high interrupt rate)
- No loss in our FPGA solution

Query:

```sql
SELECT count(*)
FROM Trades
[SIZE 1
ADVANCE 1 TIME]
```

- Packets processed at different data input rates:
  - FPGA: 100%
  - software (Linux 2.6): 60% and 36%
Glacier — Component Library and Q2HW Compiler

Glacier

- Component Library in VHDL
  - Streaming Operators: $\sigma$, $\pi$, Aggregation, Grouping, etc.
  - Helper Components: Network and CPU Adapters, Latency Balancing Register, Stream De-multiplexer
- Query-to-Hardware Compiler
  - Translates algebraic query plans
  - Wires up library components
From a Query to a Hardware Circuit

Glacier Compiler

Algebraic Query Plan

VHDL

Digital Circuit

≥ 1

&

VHDL

VHDL Synthesis and Place & Route

bitstream

FPGA
Query Circuits

Example:

```sql
SELECT Price, Volume
FROM Trades
WHERE Symbol = "UBSN"
AND Volume > 100000
```

- **n-bit Tuple → n parallel wires**
- **Additional data valid line**
- **Throughput:** 1 tuple/cycle
  (125 Mtuples/s @125 MHz)
- **Latency:** 5 cycles
  (40 ns @125 MHz)
- **Optimization collapsing stages:** 1 cycle latency
Windowing Operator

```
SELECT sum(Volume) AS svol
FROM Trades[SIZE 60 ADVANCE 15 TUPLES]
```
SELECT Symbol, avg(Price) AS AvgPrice
    FROM Trades [SIZE 600 ADVANCE 60 TIME]
    GROUP BY Symbol
### Operator Summary

<table>
<thead>
<tr>
<th>Operator</th>
<th>Variant</th>
</tr>
</thead>
<tbody>
<tr>
<td>Projection $\pi$</td>
<td>✓</td>
</tr>
<tr>
<td>Selection $\sigma$</td>
<td>✓</td>
</tr>
<tr>
<td>Arithm. &amp; Boolean $\odot$</td>
<td>✓</td>
</tr>
<tr>
<td>Aggregation</td>
<td>algebraic</td>
</tr>
<tr>
<td></td>
<td>holistic</td>
</tr>
<tr>
<td>Window $\ominus$</td>
<td>✓</td>
</tr>
<tr>
<td>Grouping</td>
<td>✓ if space bounded (# groups) and known at design time</td>
</tr>
<tr>
<td>Window-Join</td>
<td>? $\Rightarrow$ future work, seems feasible</td>
</tr>
</tbody>
</table>

External DDR-RAM needed: $\times$

Future work seems feasible: $\Rightarrow$
Current Limitations and Future Work

- Floating point not supported yet
- No VARCHAR support
- Wide tuples (> 32B) are difficult to route to meet timing constraints

- More time in software ⇒ more space on chip
- Tradeoff: serial processing of “wide” data
- Current technological limits
- FPGA can be connected to many memory banks (840 I/O pins)
Conclusions

- FPGAs can do stream processing too
- Efficient generation of query execution circuits
- Performance throughput/latency easily determined from resulting circuits
- Components allow use of FPGA as in data path and CPU co-processor
- FPGA in datapath → processing at wirespeed possible
- 1,000,000 packets/sec (sustained)