Scheduling on a Reconfigurable Processor with Virtual Compute Page

Eylon Caspi, Michael Chu, and Randy Huang
http://www.cs.berkeley.edu/~eylon/cs262/

**Problem:** How to reduce programmer effort in managing large computations on arbitrary reconfigurable resources?

**Solution:** Virtual hardware pages with OS support

---

### Methodology

2. Select and map applications in this programming model.
3. Build an event-driven array simulator.
4. Build a centralized scheduler.
5. Try different scheduling heuristics/algorithms.
6. Understand architectural tradeoffs.

---

### Programming Model

- **Token flow among compute-page operators**
  - Data flow through virtual compute pages is managed by the operating system.

### Scheduling Considerations

- **I/O between resident and non-resident compute pages** is handled efficiently.
- **On array, streaming communication is cheap**.
- **Context switches are expensive**.
- **Combine list-based resource-constrained scheduling and heuristic gang scheduling**.
- **Time-sliced operation is supported**.

### Preliminary Results

- **Random graph generator** is a useful tool for debugging.
- **Sorting the prioritizedReadyQ by resident CMBs** can lead to deadlock.
- **Path-based priority algorithm** seems to work best.
- **Equal number of CMBs and CPUs** might not lead to optimal utilization of area.

### Lessons Learned

- **Random graph generator** is a very useful tool for debugging.
- **Sorting the prioritizedReadyQ** by resident CMBs can lead to deadlock.
- **Path-based priority algorithm** seems to work best.
- **Equal number of CMBs and CPUs** might not lead to optimal utilization of area.

### Status

- **5500 lines of Java code written** (scheduler, simulator)
- **Mapped three benchmark into SCORE programming model**
- **Still collecting data**

### Future Work

- **Improve array cost model**
- **Cross-process Fairness (lottery scheduling)**
- **Memory management for CMBs**
- **Cluster pages (feedback)**

---

### What’s a Reconfigurable Processor?

- **Architecture Features**:
  - Integration of DRAM, microprocessor, and a reconfigurable array.
  - Fast-tree interconnect, characterized by Rent’s parameter.
  - Fast configuration time (µs).
  - Pipelined interconnect to support fast clock period.
  - Highly scalable.

- **Microprocessor**

---

### What’s a virtual compute page?

- **A Compute Page (CP)** is a hierarchically-connected array of programmable computational elements called LookUpTables (LUTs). E.g. A 4-LUT can compute any binary function of 4 inputs.

- **A Virtual Compute Page is a logical entity** which must be scheduled to run in a physical compute page, analogous to virtual memory pages and physical frames.

- **Virtual hardware pages with OS support** automatically manages the virtual mapping and page loading/eviction.

- **Automated virtualization** frees application writers from physical size constraints and enables compatibility across different size arrays.

---

### How to Program Using SCORE

- **Programming Model**
  - **Token flow among compute-page operators**

- **Scheduling Considerations**
  - **I/O between resident and non-resident compute pages** is handled efficiently.
  - **On array, streaming communication is cheap**.
  - **Context switches are expensive**.
  - **Combine list-based resource-constrained scheduling and heuristic gang scheduling**.
  - **Time-sliced operation** is supported.

---

### BRASS Home Page

http://www.cs.berkeley.edu/projects/brass/