Inadequacy of Processor Load-Store Queues

Overcoming the Limitations of Static Scheduling

- Triggered when a flow decision is made
- Allocate entries for all the memory operations of a group at once
  - The arguments of the memory accesses can arrive out-of-order

Basic idea: allocate groups of memory accesses depending on the dynamic behaviour of the application

Groups
- Sequences of accesses which are statically predefined
- If one access of a group executes, all other accesses belonging to the same group will eventually execute
- The notion of basic blocks already exists in HLS tools

Supplying Sequential Order to the LSQ

The effect of different queue parameters on performance and resource utilization
- The overhead of a single port is minor compared to the overall resources of the LSQ
- The high sensitivity to the number of queue entries is in line with what others achieved in conventional LSQ designs
- Changing the number of groups and the way the ports are organized has minimal influence on cycle time

Use case: Weighted histogram
- The LSQ designs achieve significant speed-ups over statically scheduled designs by dynamically resolving dependencies

Experimental Results

Comparison of the histogram application created by Vivado HLS and a dynamic design using our LSQ in different sizes. We achieve a speed-up of up to 3x over the static design.