



# Stream Register Files with Indexed Access



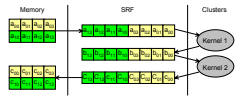
William J. Dally

Nuwan Jayasena  
Jung Ho Ahn

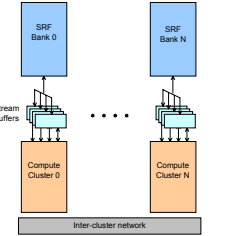
Mattan Erez  
Ujval Kapasi  
Abhishek Das

Francois Labonte  
Timothy Knight

## Stream Register File (SRF)

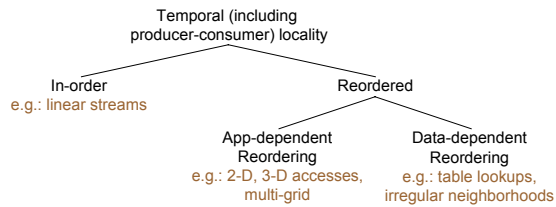


- SRF serves two primary purposes in a stream processor
  - Capture stream temporal locality
  - Staging area for memory transfers



- The SRF is implemented as  $N$  banks, each associated with a single compute cluster
- Unlike a cache, the SRF is software managed, and is a separate address space from the memory system
- Data needed by a particular cluster is placed in the SRF bank associated with it, thereby reducing communication requirements

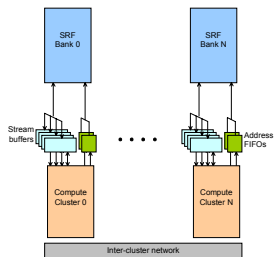
## Locality in Stream Programs



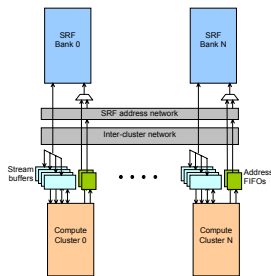
- Conventional SRF implementation only captures in-order stream accesses

## SRF with Indexed Access

- Indexed SRF allows reordered access patterns
- 2 degrees of indexing freedom:
  - In-lane:** a cluster can index in to local SRF bank only
    - Restricted access freedom
    - Lower hardware overhead and complexity
    - Peak bandwidth comparable to sequential streams
  - Cross-lane:** any cluster can index in to any location of SRF
    - Greater flexibility in access patterns
    - Higher hardware overhead and complexity
    - Lower peak bandwidth due to cross-lane communication



In-lane indexed SRF



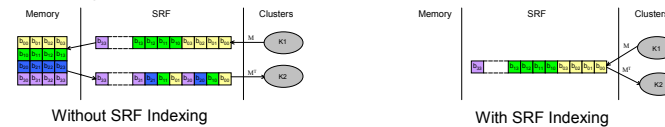
Cross-lane indexed SRF

*The Stream Register File (SRF) is a key element of the storage and bandwidth hierarchy of a stream processor. While conventional SRF designs are restricted to accessing streams in sequential order, more flexible indexed access can be supported with relatively low overhead.*

*Indexed access allows the SRF to capture more of the temporal locality available in applications. For target application classes, this results in a significant reduction in bandwidth demands on the memory system and/or performance improvements.*

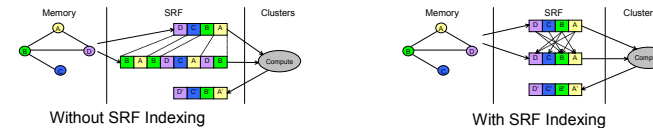
## Performance Impact of Indexed SRF Access

- Eliminates stream reordering memory operations
  - Reduces memory bandwidth demands
  - Shortens lifetime of data set in SRF, reducing the number of live data sets and hence increasing the strip size



- Kernel K1 generates matrix  $M$ , K2 consumes  $M^T$
- Sequential streams (i.e. no SRF indexing) requires transpose to be done through memory
- With indexing,  $M^T$  read directly from  $M$  in SRF

- Eliminates data replication in SRF and redundant memory accesses
  - Reduces space occupied in SRF, allowing for longer strip sizes
  - Reduces memory bandwidth demands by eliminating repeated accesses



- Kernel takes stream of nodes and stream of neighbors
- W/o indexing, repeated elements are replicated in neighbor stream
- SRF indexing allows a single copy to be referenced multiple times

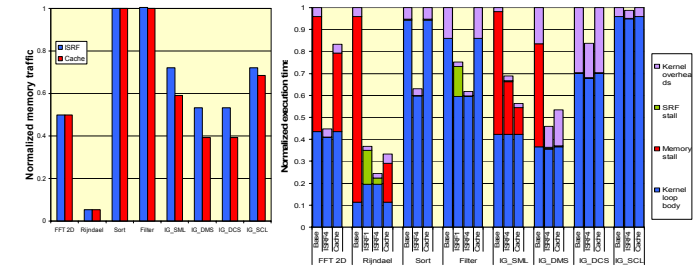
## System Features Enabled by Indexed SRF Access

- Support more concurrent logical streams than available hardware streams
- Local register spilling to SRF
- Unified storage for streams and scratchpad memory
- High bandwidth scratchpad accesses

## Benchmark Results

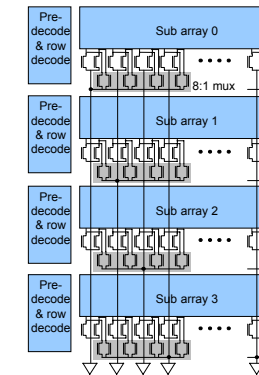
- Benchmarks:
  - FFT\_2D: 64x64 2D FFT models multi-dimensional accesses
  - Rijndael: Encryption algorithm with large numbers of data-dependant table lookups
  - Sort: Merge-sort of 4096 values
  - Filter: 5x5 convolution filter over 256x256 image
  - IG\_\*: Parameterized irregular grid neighborhood access synthetic benchmark. Parameter values for data sets are as follows:

| Data set | Compute density | Avg. graph degree | Avg. strip size |             |
|----------|-----------------|-------------------|-----------------|-------------|
|          |                 |                   | Base SRF        | Indexed SRF |
| IG_SML   | 16              | 4                 | 1812            | 2325        |
| IG_SCL   | 51              | 4                 | 1812            | 2325        |
| IG_DMS   | 16              | 16                | 267             | 527         |
| IG_DCS   | 51              | 16                | 267             | 527         |



- Measurements obtained for machine model with 8 clusters, 4K SRF-words/cluster, and 9.14 GB/s peak DRAM bandwidth
- ISRF1 and ISRF4 correspond to 1 and 4 words/cycle per cluster peak indexed SRF bandwidth (ISRF1 shown only where it differs from ISRF4)

## SRAM Array Implementation



- Each SRF bank implemented as 4 sub-arrays due to VLSI constraints
- Leverage sub-banks to support up to 4 concurrent indexed accesses per bank
  - Low additional overhead:
    - Row decoder per sub array
    - 8:1 muxes at output of sub arrays
- Sequential streams perform wide accesses from a single sub array at a time
  - Energy efficiency of sequential stream accesses comparable to non-indexed SRF

## Status and Future Directions

- Indexed SRF access fully supported in scheduler and cycle-accurate simulator
- Indexed SRF access used in current Merrimac applications
  - StreamFLO:
    - Stencil and group accesses
- Future application targets
  - Sparse matrix operations, irregular meshes etc.
- Future research issues
  - Extracting indexed access patterns from high-level stream code