TIZIANO DE MATTEIS, JOHANNES DE FINE LICHT AND TORSTEN HOEFLER

FBLAS: Streaming Linear Algebra Kernels on FPGA

5TH International Workshop on Heterogeneous High-performance Reconfigurable Computing
FPGA for HPC

Modern high-performance FPGAs are attractive for HPC workloads:
- they are offered with native floating points units (DSPs), HBM, Network interfaces ...

However, they are rarely considered in HPC
- Productivity: HLS and OpenCL ease programmers life
- Tools and libraries: lack of maintained, publicly available and re-usable components;

We contribute with fBLAS, an open-source projects:
- First open source (HLS) and complete BLAS available for FPGA;
- Numerical module interfaces are designed to natively support streaming communication across on-chip connections

github.com/spcl/FBLAS
FBLAS: library design

**HLS Modules**: implement numerical routines (e.g. DOT, GEMV, ...) :
- exploit spatial parallelism and fast on-chip memory
- have a *streaming* interface to enable communications through on-chip FIFO buffers: **data arrives/is produced using input/output channels**

**Host Layer**: allows the user to invoke numerical routines from the host
- the API is written in C++, and provides a set of library calls matching BLAS API
- can be used to offload single routine to FPGA

**FBLAS** currently targets the Intel ecosystem (e.g. Stratix 10)
- Eventually both SDx and Intel OpenCL support with the *same interface*
Modules implementation

FBLAS modules are pre-optimized with key HLS transformations, such as *pipelined loops, replication, and tiling*

**Tiling** has implications for how data is streamed to/from modules

For **GEMM**, computation is organized in a 2D Systolic array

---

Optimizations are configurable by the user according to desired performance or utilization requirements
Module composition

Streaming interface enables **communication through on-chip memory rather than through off-chip DRAM**

**Example:** consider the following computation

\[ y = (A + uv^T)x + y \]

**I/O:** 3\(N^2\) + 5\(N\)

Reduces costly off-chip memory accesses and allows pipelined parallel modules execution
Streaming Composition

A computation is expressed by a *Module Directed Acyclic Graph* (MDAG)

An MDAG is **valid** if:

- it expresses a composition that will terminate
- all the edges are valid. An edge is valid if:
  - # of elements produced = # of elements consumed
  - order in which elements are consumed = order in which they are produced

Composition of multi-trees

A multi-tree module composition, with valid edges, is always valid. E.g. axpydot:

\[
\begin{align*}
  z & \leftarrow w - \alpha v \\
  \beta & \leftarrow z^T u
\end{align*}
\]

Requires 3 BLAS calls. **I/O = 7N**

I/O = 3N + 1

(and modules run in parallel)
Streaming Composition

A computation is expressed by a **Module Directed Acyclic Graph** (MDAG)

An MDAG is **valid** if:

- it expresses a composition that will terminate
- all the edges are valid. An edge is valid if:
  - # of elements produced = # of elements consumed
  - order in which elements are consumed = order in which they are produced

**Composition of non multi-trees**

Invalid graphs could occur in generic compositions

Solved by:

- setting the channel size appropriately (according to the size of input data)
- breaking the MDAG into multiple valid components
Results

Target architecture: FPGA: Stratix 10, 5.7K DSPs, 29 MB BRAM, 32 GB DRAM. Host: 10 cores Intel Xeon, 64 GB DRAM.

Module evaluation: scaling with different vectorization width/tiling. Input data generated on chip

Streaming composition: speedup wrt. DRAM implementation, evaluated over various meaningful compositions.
CONCLUSIONS

**fBLAS**, is the first HLS-based BLAS implementation available for FPGA

User can offload routines from an host program or integrate them into HLS codes

HLS modules have a streaming interface to enable communications through on-chip FIFO buffers rather than DRAM

---

**fBLAS**

[GitHub Repository](https://github.com/spcl/FBLAS)
Thanks!

Any Questions?