Graph Streaming Processor

A Next-Generation Computing Architecture

Val G. Cook – Chief Software Architect
Satyaki Koneru – Chief Technology Officer
Ke Yin – Chief Scientist
Dinakar Munagala – Chief Executive Officer
Introduction

• THINCI, Inc. “think-eye” is 5-year-old strategic/venture-backed technology startup

• Develop silicon for machine learning, computer vision and other strategic parallel workloads

• Provide innovative software along with a comprehensive SDK

• 69-person team (95% engineering & operations)

• Key IP (patents, trade secrets)
  – Streaming Graph Processor
  – Graph Computing Compiler

• Product Status
  – Early Access Program started Q1 2017
  – First edition PCIe-based development boards will ship Q4 2017
Architectural Objective

Exceptional efficiency via balanced application of multiple parallel execution mechanisms

Levels of Parallelism
- Task Level Parallelism
- Thread Level Parallelism
- Data Level Parallelism
- Instruction Level Parallelism

Key Architectural Choices
- Direct Graph Processing
- Fine-Grained Thread Scheduling
- 2D Block Processing
- Parallel Reduction Instructions
- Hardware Instruction Scheduling
Task Level Parallelism

Direct Graph Processing
Task Graphs

- **Formalized Task Level Parallelism**
  - Graphs define only computational semantics
  - Nodes reference kernels
  - Kernels are programs
  - Nodes bind to buffers
  - Buffers contain structured data
  - Data dependencies explicit

- **ThinCI Hardware Processes Graphs Natively**
  - A graph is an execution primitive
  - A program is a proper sub-set of graph
Graph Based Frameworks

- **Graph Processing or Data Flow Graphs**
  - They are a very old concept, for example Alan Turing’s “Graph Turing Machine”.
  - Gaining value as a computation model, particularly in the field of machine learning.

- **Graph-based machine learning frameworks have proliferated in recent years.**
Streaming vs. Sequential Processing

• Sequential Node Processing
  – Commonly used by DSPs and GPUs
  – Intermediate buffers are written back and forth to memory
  – Intermediate buffers are generally non-cacheable globally
  – DRAM accesses are costly
    • Excessive power
    • Excessive latency

• Graph Streaming Processor
  – Intermediate buffers are small (~1% of the original size)
  – Data is more easily cached
  – Benefits of significantly reduced memory bandwidth
    • Lower power consumption
    • Higher performance
Thread Level Parallelism
Fine-Grained Thread Scheduling
Fine-Grained Thread Scheduling

- **Thread Scheduler**
  - Aware of data dependencies
  - Dispatches threads when:
    - Resources available
    - Dependencies satisfied
  - Maintains ordered behavior as needed
  - Prevents dead-lock
- **Supports Complex Scenarios**
  - Aggregates Threads
  - Fractures Threads
Graph Execution Trace

- Threads can execute from all nodes of the graph simultaneously
- True hardware managed streaming behavior
Data Level Parallelism

2D Block Processing
Parallel Reduction Instructions
2D Block Processing/Reduction Instructions

- Persistent data structures are accessed in blocks
- Arbitrary alignment support
- Provides for “in-place compute”
- Parallel reduction instructions support efficient processing
  - Reduced power
  - Greater throughput
  - Reduced bandwidth
- Experience better scaling across data types vs. the 2x scaling of traditional vector pipelines
Instruction Level Parallelism

Hardware Instruction Scheduling
Hardware Instruction Scheduling

- Scheduling Groups of Four Processors
  - Hardware Instruction Picker
  - Selects from 100’s of threads
  - Targets 10’s of independent pipelines
Programming Model
Programming Model

• Fully Programmable
  – No a-priori constraints regarding data types, precision or graph topologies
  – Fully pipelined concurrent graph execution
  – Comprehensive SDK with support for all abstraction levels, assembly to frameworks

• Machine Learning Frameworks
  – TensorFlow
  – Caffe
  – Torch

• OpenVX + OpenCL C/C++ Language Kernels (Seeking Khronos conformance post Si)
  – Provides rich graph creation and execution semantics
  – Extended with fully accelerated custom kernel support
Results

• Arithmetic Pipeline Utilization
  – 95% for CNN’s (VGG16, 8-bit)

• Physical Characteristics
  – TSMC 28nm HPC+
  – Standalone SoC Mode
  – PCIe Accelerator Mode
  – SoC Power Estimate: 2.5W