An Insight into Kalray’s OpenCL™ High Performance Implementation

Sébastien Le Duc
Software Engineering Director, Kalray

IWOCL & SYCLcon - April 27, 2021
Agenda

1. Kalray in a Nutshell
2. MPPA® Architecture Overview
3. KAF™: Kalray Acceleration Framework
4. Mapping OpenCL™ on the MPPA® Architecture
5. OpenCL™ on MPPA® Usage
6. Conclusion
Kalray offers a new type of processor targeting the booming market of intelligent systems.

**A Global Presence**
- France (Grenoble, Sophia-Antipolis)
- USA (Los Altos, CA)
- Japan (Yokohama)
- Canada (Partner)
- China (Partner)
- South Korea (Partner)

**Leader in Manycore Technology**
- 3rd generation of MPPA® processor
- ~€100m R&D investment
- 30 Patent families

**Industrial investors**
- NXP
- Renault Nissan Mitsubishi
- Safran
- MBDA

**Financial investors:**
- CEA Investissement, Bpifrance, ACE, INOCAP Gestion, Pengpai

*Public Company (ALKAL)*
*Support from European Govts*
*Working with 500 fortune companies*
INTELLIGENT SYSTEMS / EDGE COMPUTING
At the Heart of Next Decade Industry

Next Gen. Embedded Systems

Next Gen. Data Center

Compute and AI Intensive Critical Systems

MPPA® Processors

PCle Cards & Modules

Acceleration Solutions for Storage, Networking and Compute
Agenda

1. Kalray in a Nutshell
2. MPPA® Architecture Overview
3. KAF™: Kalray Acceleration Framework
4. Mapping OpenCL™ on the MPPA® Architecture
5. OpenCL™ on MPPA® Usage
6. Conclusion
MPPA® COOLIDGE™ SCALABLE APPROACH

MANycORE PROCESSOR
- 80 or 160 CPU cores (SiP)
- Frequency: from 600 to 1200 MHz

System Interconnects
- AXI
- NoC

Interfaces
- 2x 100Gb/s Ethernet
- PCIe GEN4 x16

COMPUTE CLUSTER
- 16 CPU 64-bit cores
- 1x Safety/Security 64-bit core
- 2x Crypto accelerators
- DMA

Memory Hierarchy
- L1 data caches coherent between all cores of a compute cluster
- 4MB local memory that can be configured as either local memory or L2$ (4:0, 3:1, 2:2)

3RD GENERATION VLIW CORE
- 64-bit core
- 6-issue VLIW architecture
- MMU + I&D L1 cache (16KB+16KB)
- 16-bit/32-bit/64-bit IEEE 754-2008 FPU
- Vision/ML Co-processor (TCA)
Agenda

1. Kalray in a Nutshell
2. MPPA® Architecture Overview
3. KAF™: Kalray Acceleration Framework
4. Mapping OpenCL™ on the MPPA® Architecture
5. OpenCL™ on MPPA® Usage
6. Conclusion
KAF™: KALRAY ACCELERATION FRAMEWORK
Easy Integration with Host System

KAF™, for easy integration with host system

- Direct programming on MPPA® supported through OpenCL™ and OpenMP dispatch with OpenCL™
- OpenCL™ used as backend offloading API by KAF™ libraries and KaNN™ ML compiler

Optimized Applications

Optimized Middleware

KAF™

Direct Programming

ML models

API based programming

KaNN™ ML compiler

OpenCV

BLAS
LAPACK
FFTW

OpenCL

OpenMP
dispatch from
OpenCL

MPPA® CLUSTERS
Agenda

1. Kalray in a Nutshell
2. MPPA® Architecture Overview
3. KAF™: Kalray Acceleration Framework
4. Mapping OpenCL™ on the MPPA® Architecture
5. OpenCL™ on MPPA® Usage
6. Conclusion
OpenCL™ PLATFORM MODEL

The OpenCL™ Platform Model

- **Topology**: A Host is connected to one or several OpenCL™ devices
- **Host**: Processor on which the applications / middleware run
- **OpenCL Device**: One or more compute units
- **Compute units**: One or more processing elements
- **Sub-device**: Subset of compute units of the parent device
- **Control flow**: Can be either converged or diverged

Mapping to the MPPA®

- **Topology**: MPPA® PCIe board used as compute accelerator
- **Host**: Intel x86 or ARM
- **OpenCL™ device**: The MPPA®
- **Compute units**: MPPA® clusters
- **Processing elements**: MPPA® cores
- **Sub-device**: Subset of clusters
- **Control flow**: Diverged
OpenCL™ EXECUTION MODEL

Definition of Execution Model

- **Host**: Executes the host program
- **OpenCL device**: Executes the compute kernel
- **Work-item**: One of the parallel executions of the kernel function
- **Work-group**: A group of work-items that execute on a single compute unit
- **NDRange kernel**: The index space for work-item parallel executions

Mapping to the MPPA®

- **Host**: Intel x86 or ARM
- **OpenCL™ device**: MPPA®
- **Work-item**: Executes on one MPPA® core
- **Work-group**: Executes on one MPPA® cluster
- **NDRange**: Device or sub-device
OpenCL™ MEMORY MODEL

Definition of Memory Model

OpenCL™ Device Memory consists of 4 memory regions:

• **Global**: Read/write access to all work-items in all work-groups on any device.
• **Constant**: A region of global memory that remains constant during the execution of a kernel instance.
• **Local**: A memory region local to a work-group.
• **Private**: A memory region private to a work-item.

Mapping to the MPPA®

• **Global / Constant**: MPPA® DDR
• **Local**: Per cluster shared segment in cluster local memory
• **Private**: Per core private segment in cluster local memory

![Diagram of MPPA® architecture and memory mapping](image-url)
**OpenCL™ C: MAPPING TO THE MPPA®**

MPPA® is a DMA-based architecture

- DMA transfers exposed thru OpenCL™ C async copy built-in functions
  - `cl_khr_extended_async_copies`: 2D/3D async copies
  - `cl_khr_async_workgroup_copy_fence`: ordering of async copy operations
  - Kalray vendor extensions: `async_work_item_copy` functions, more 2D/3D extensions

MPPA® supported data types

- `cles_khr_int64`: 64-bit integers
- `cl_khr_fp16`: 16-bit floats
- `cl_khr_fp64`: 64-bit floats
- OpenCL C vector types and vector math built-in functions: MPPA extensive SIMD support
- Kalray vendor extension: tensor data type and associated built-in functions

MPPA® atomic operations

- OpenCL C 2.0 scoped atomic functions
## OpenCL™ Programming: MPPA® vs GPU

<table>
<thead>
<tr>
<th>GPU</th>
<th>MPPA®</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GPUs have a SIMT architecture</strong></td>
<td><strong>MPPA® has a clustered Many-core architecture with SIMD cores</strong></td>
</tr>
<tr>
<td>• Execution of kernels based on warps: group of threads that all execute the same instruction in lockstep manner</td>
<td>• No divergence cost</td>
</tr>
<tr>
<td>• Warp divergence cost is usually high</td>
<td>• DMA based architecture: to achieve high performance, data in DDR must be copied to local cluster memory before processing it</td>
</tr>
<tr>
<td>• DDR latency hidden by hardware hyper-threading and hardware scheduling of warps</td>
<td>• Amount of local cluster memory much higher than on a GPU</td>
</tr>
</tbody>
</table>

**Consequence of OpenCL™ kernels written with GPU in mind:**

- Small work-item functions that process one point of the index space
- Direct access to global (DDR) memory

**Consequence of OpenCL™ kernels**

- OpenCL™ code written with GPU in mind will be functional on the MPPA®
- Better to have bigger work-item functions that process several data points
- Better to use OpenCL™ async workgroup copy built-in functions to copy data to local cluster memory
Agenda

1. Kalray in a Nutshell
2. MPPA® Architecture Overview
3. KAF™: Kalray Acceleration Framework
4. Mapping OpenCL™ on the MPPA® Architecture
5. OpenCL™ on MPPA® Usage
6. Conclusion
**OpenCL™ ON MPPA®: SUPPORTED FEATURES**

### HOST API: Based on OpenCL™ 1.2 Embedded Profile + Extensions

- Conformance achieved for OpenCL™ 1.2 Embedded Profile
- No Image support (optional in OpenCL)

### Standard Extensions supported

- cles_khr_int64: 64-bit integers
- cl_khr_fp16: 16-bit floats
- cl_khr_fp64: 64-bit floats
- cl_khr_extended_async_copies: 2D/3D async copies
- cl_khr_async_workgroup_copy_fence: Ordering of async copy operations

### Kalray Extensions

- **Buffer flags**
  - Allocation of OpenCL™ buffer in any of the clusters’ local memory
  - Allocation of OpenCL™ Buffers in memory region suitable for PCIe peer-to-peer
- **OpenMP dispatch from OpenCL** (see next slide)
- **Async copy extensions**
  - Async_work_item_copy functions: similar to async_workgroup_copy but not synchronizing
  - More 2D/3D extensions
OpenMP DISPATCH FROM OpenCL™

- OpenCL™ used to execute one WI on Core0 of each cluster
- OpenMP used to parallelize within the cluster
The MPPA® can be partitionned:
- at boot time
- or at runtime into several isolated accelerators
Agenda

1. Kalray in a Nutshell
2. MPPA® Architecture Overview
3. KAF™: Kalray Acceleration Framework
4. Mapping OpenCL™ on the MPPA® Architecture
5. OpenCL™ on MPPA® Usage
6. Conclusion
CONCLUSION

Kalray reached OpenCL™ conformance end of 2020 on Coolidge™ MPPA® processor, 3rd generation of MPPA® intelligent processor

What it brings to Kalray?

• Leverage existing OpenCL™ code base
• Leverage tools or libraries that support offloading on an accelerator with OpenCL™ (SyCL, TVM, OpenCV, ...)
• Leverage existing OpenCL development tools (CodeXL, oclgrind, ...)
• Less effort on documentation: focus only on MPPA® specific information
• MPPA® adoption: standard programming model

What it brings to our customers?

• OpenCL™ available resources to accelerate learning curve
• Easier to find skilled OpenCL™ engineers
• Flexibility for porting from one hardware architecture to another
• Long term maintenance
• Much easier porting methodology