Industry-standard programming models for TI KeyStone II-based ARM + DSP SoCs

Ellen Blinka & Ajay Jayaraj

Texas Instruments

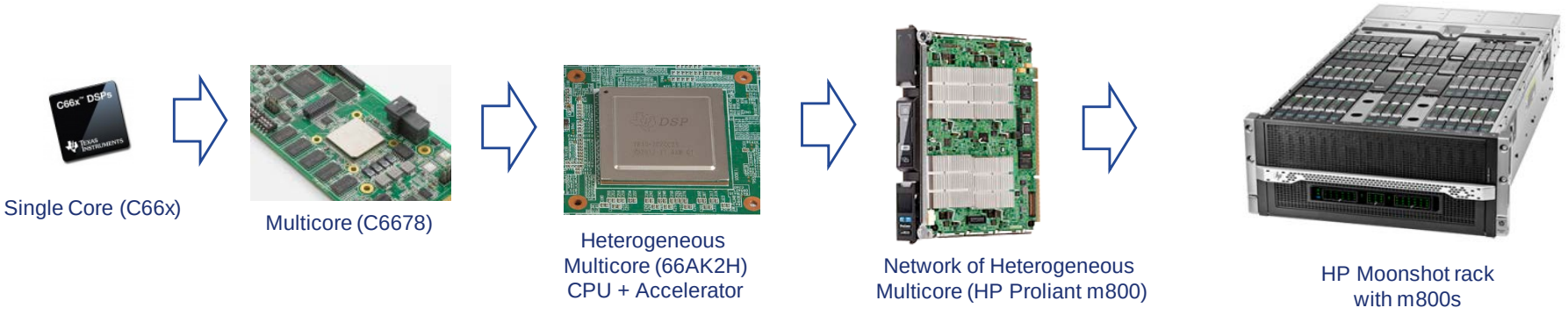
Agenda

- Multicore heterogeneous systems
- Brief overview of KeyStone II SoCs
- Software development philosophy for KeyStone II SoCs
- Tools for parallelization
- Tools for heterogeneous acceleration
- Tools for systems of multiple devices
- Software libraries

Continuous demand for increased processing

The need for multicore heterogeneous SoCs

- Market demand for increased processing performance, reduced power, and efficient use of board area
- Demand satisfied by **adding cores**
  - Mix of general purpose CPUs & accelerators (DSPs, GPUs, hardwired compute engines)
- Challenges:
  - How to efficiently segment tasks between compute engines
  - How to effectively and quickly program multiple cores of different types



Need to scale algorithms to fit available computing power.

The Solution: Programming Models

- Traditional approaches:
  - Manually partition workloads to individual cores
  - Optimize partitioned regions for the core
  - This offers high entitlement

    BUT
  - Partition must be redone for each system configuration
  - Not portable
  - Developer needs detailed knowledge of SoC architecture
    - Increased time to market

- Multicore Software Dev. Tools:
  - Modify code with pragmas and directives
  - Parallelization and load balancing are abstracted from the user
  - This offers high performance

    AND
  - Standard tools are portable to many architectures
  - SoC architecture details are abstracted from the developer
  - Data parallelization, task parallelization, accelerator offload, and more are all possible

TI enables industry-standard multicore software development tools on KeyStone II-based ARM + DSP SoCs

K2H Platform 66AK2H14/12/06 Functional Diagram

C66x Fixed or Floating Point DSP

- 4x/8x 66x DSP cores up to 1.4GHz
- 2x/4x Cotex ARM A15
- 1MB of local L2 cache RAM per C66 DSP core
- 4MB shared across all ARM

Large on chip and off chip memory

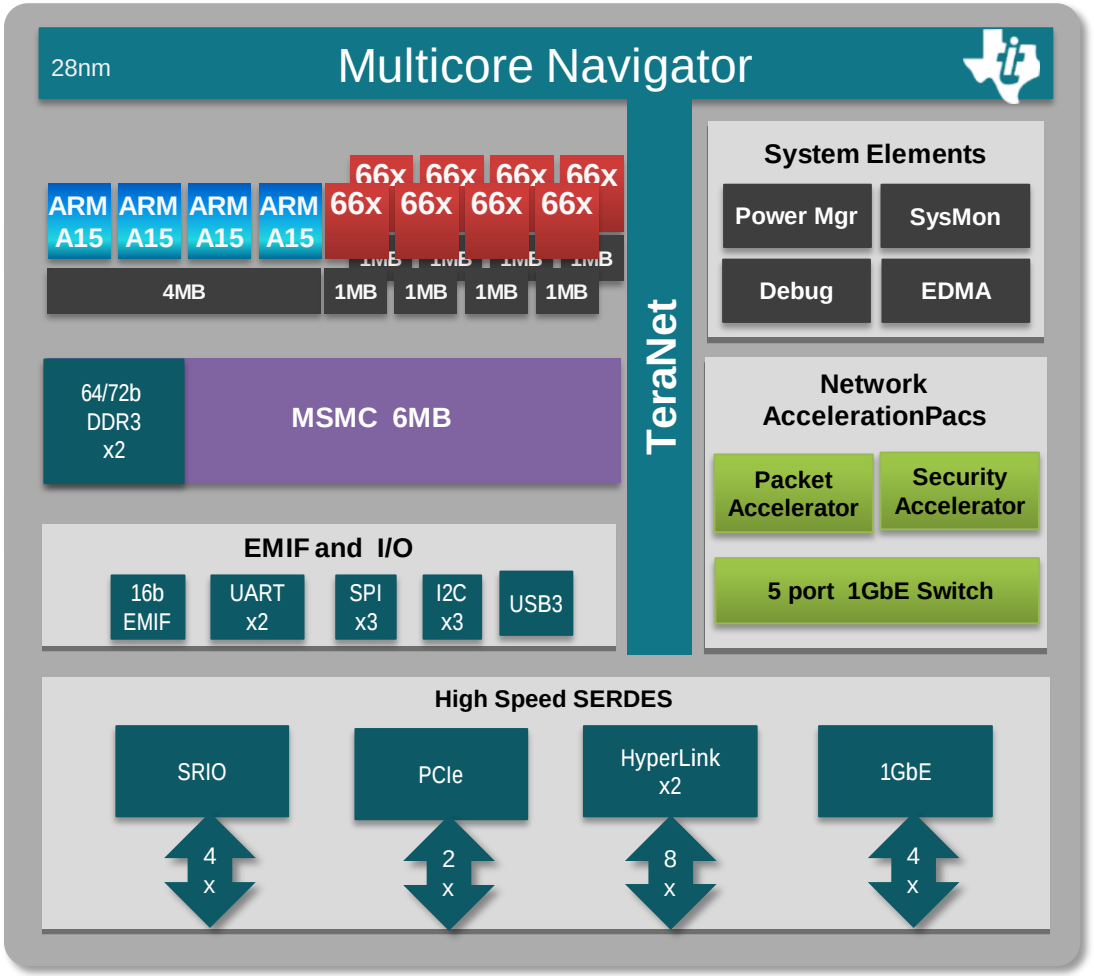
- Multicore Shared Memory Controller provides low latency & high bandwidth memory access
- 18MB (6MB Shared) L2 on-chip with ECC
- 2 x 72 bit DDR3, 72-bit (with ECC), 10GB total addressable, DIMM support (4 ranks total)

KeyStone multicore architecture and acceleration

- Multicore Navigator, TeraNet, HyperLink
- 1GbE Network coprocessor (IPv4/IPv6)
- Crypto Engine (IPSec, SRTP)

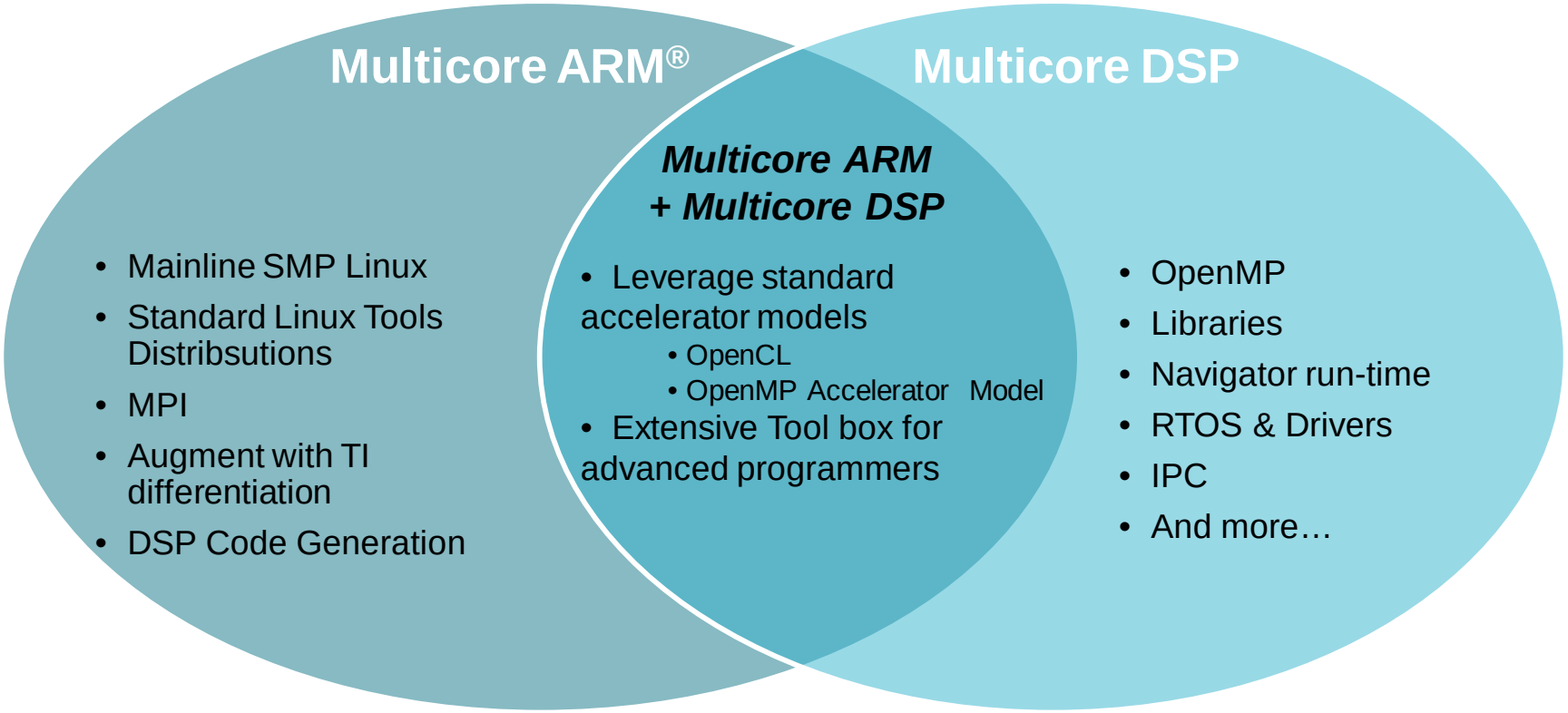
Peripherals

- 4 Port 1G Layer 2 Ethernet Switch
- 2x PCIe, 1x4 SRIO 2.1, EMIF16, USB 3.0 UARTx2, SPI, I²C
- 13-16W depending upon DSP cores, speed, temp & other factors



40mm x 40mm package

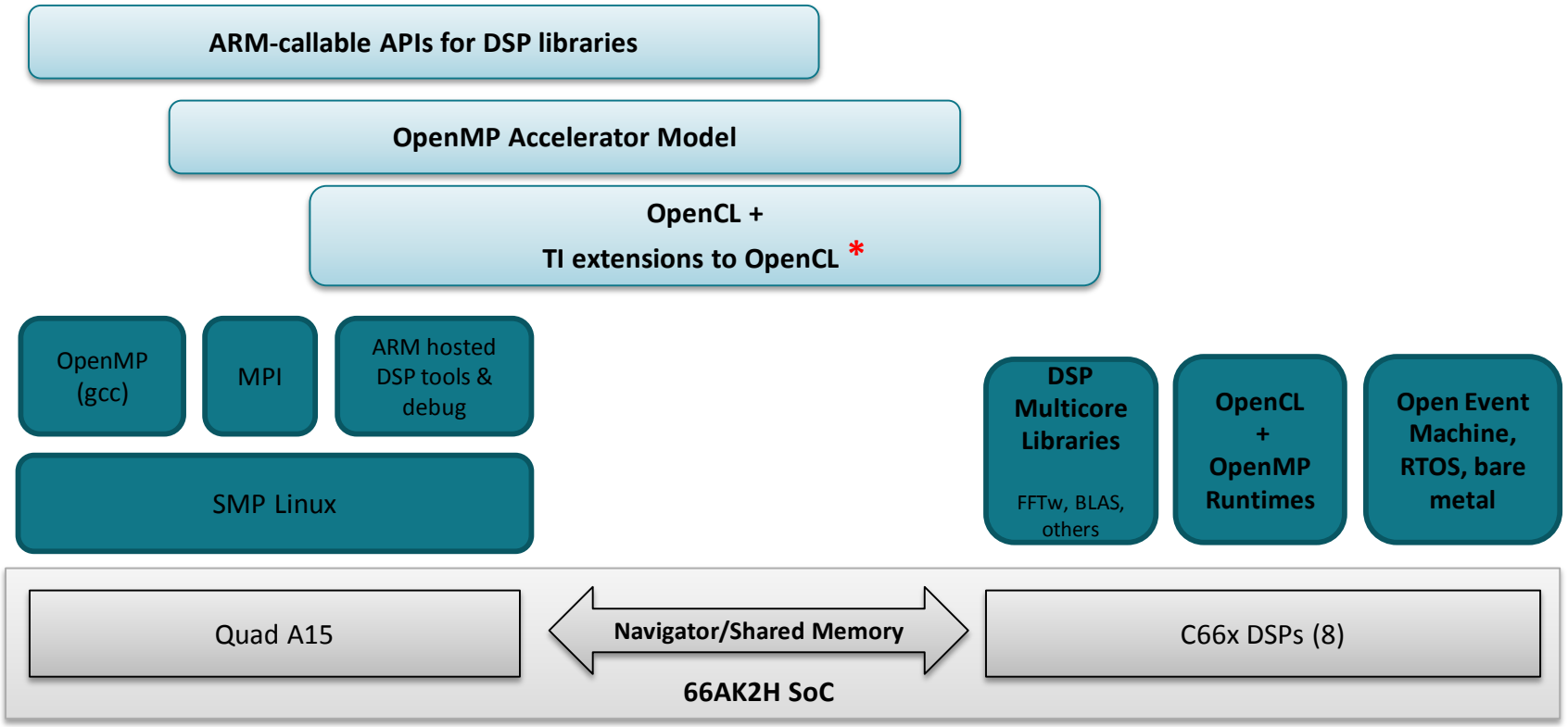
Software Development Philosophy



Development Environment:

- GDB, etc. Apps Debug Environment for ARM & DSP
- Eclipse "Embedded" Development & Debug Environment
- Instrumentation and Trace leveraging embedded hardware capability

Programming Models for KeyStone II



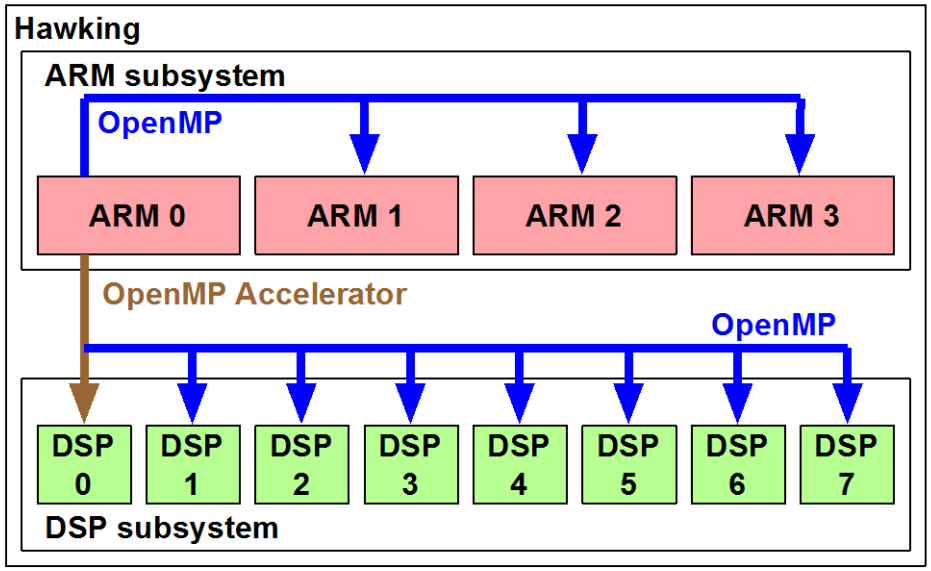
* **TI extensions enable OpenCL kernels to act as wrappers for C code with OpenMP regions**

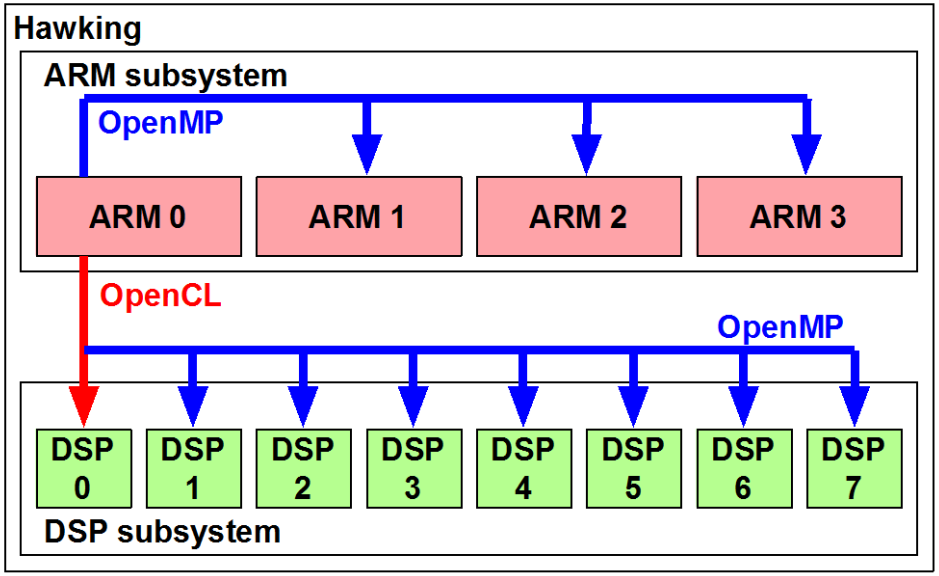
- A node is a 66AK2H SoC
- OpenMP Accelerator model or OpenCL for offloading computation from ARMs to DSPs on a single node
- MPI on ARM to communicate across nodes (multiple transports supported)

OpenMP for Shared Memory Parallel programming

- OpenMP available on ARM and DSP clusters
  - OpenMP supported in TI's C66x compiler and runtime
  - Efficient, low overhead runtime for C66x cores using Multicore Navigator
  - OpenMP supported in standard GCC tooling

- Portable to any shared memory architecture
- Simple to modify existing single-threaded code

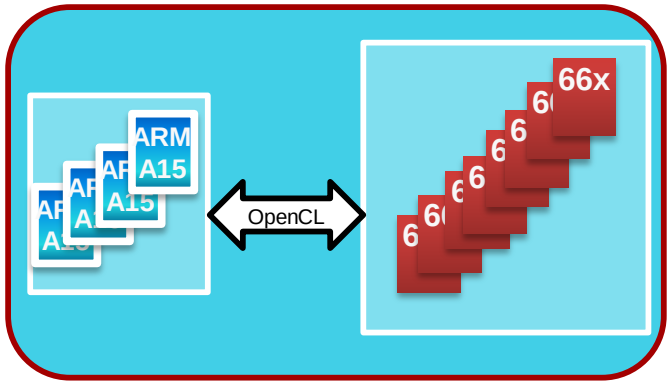


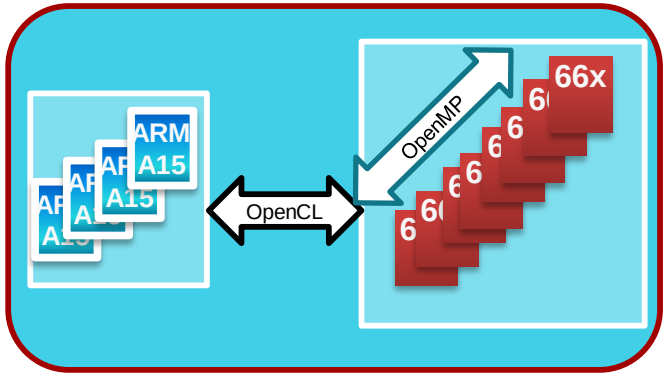


OpenCL for Heterogeneous Acceleration

- OpenCL: used to dispatch tasks from A15s to C66x DSP cores (host to accelerator)
- Multicore Navigator is used to dispatch workgroups and tasks across multiple DSP cores
- TI extension: allows OpenMP dispatch within OpenCL kernel

- Scalable and portable
- Existing OpenMP code integrated into heterogeneous SoC easily with dispatch from OpenCL kernel
- TI extension for OpenMP dispatch also reduces overhead by dispatching only one OpenCL task





OpenCL for Heterogeneous Acceleration

Data Parallel (NDRangeKernel)

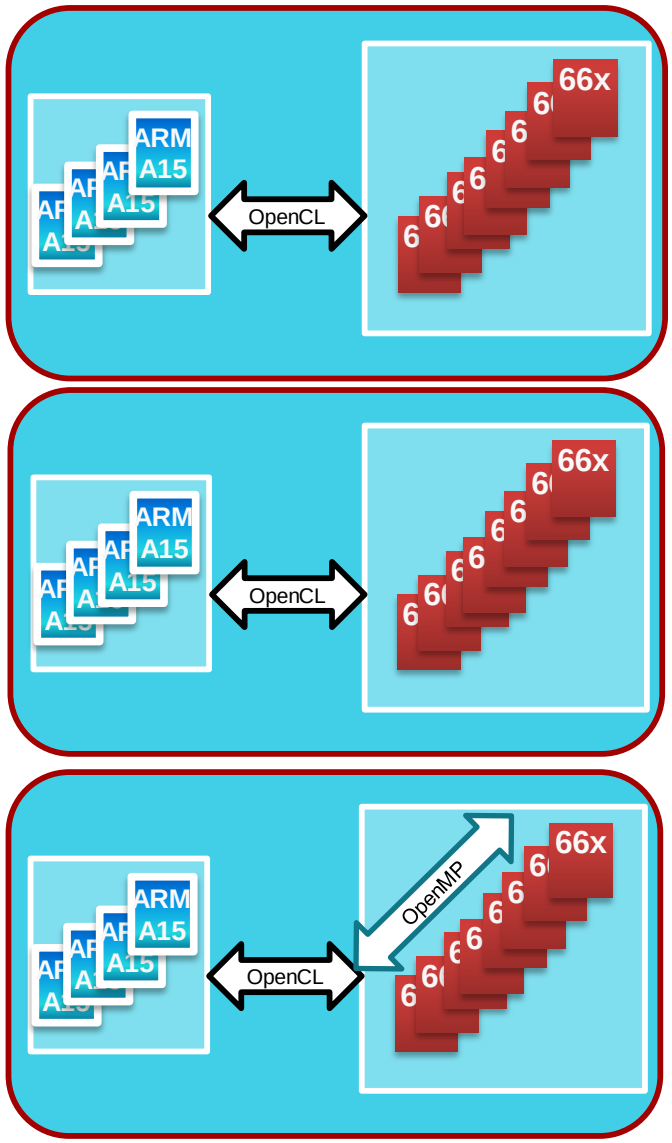
- Kernel is enqueued
- OpenCL subdivides kernel into N workgroups (WG)
- Each WG operates independently on a core, WGs operate concurrently on all cores
- All WGs finish before another Kernel or Task is dispatched
- The host can run asynchronous to the kernel

Task Parallel (Task + Out-Of-Order Queue)

- Task is enqueued
- OpenCL dispatches task to one of the cores
- OpenCL can accept additional tasks and dispatch asynchronously
- The host can run asynchronous to the task

OpenMP dispatch through OpenCL (Task + In-Order Queue)

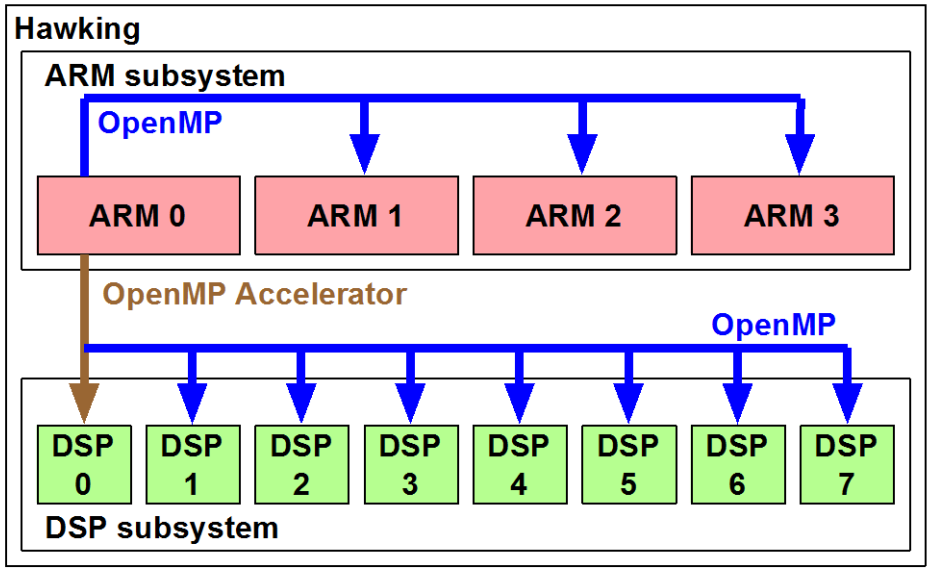
- TI Extension
- Task is enqueued
- OpenCL dispatches task to Core 0
- The task can then leverage other cores by entering OpenMP parallel region
- The host can run asynchronous to the task
- Lowers overhead from dispatching multiple tasks
- Easily integrate existing OpenMP code into heterogeneous environment



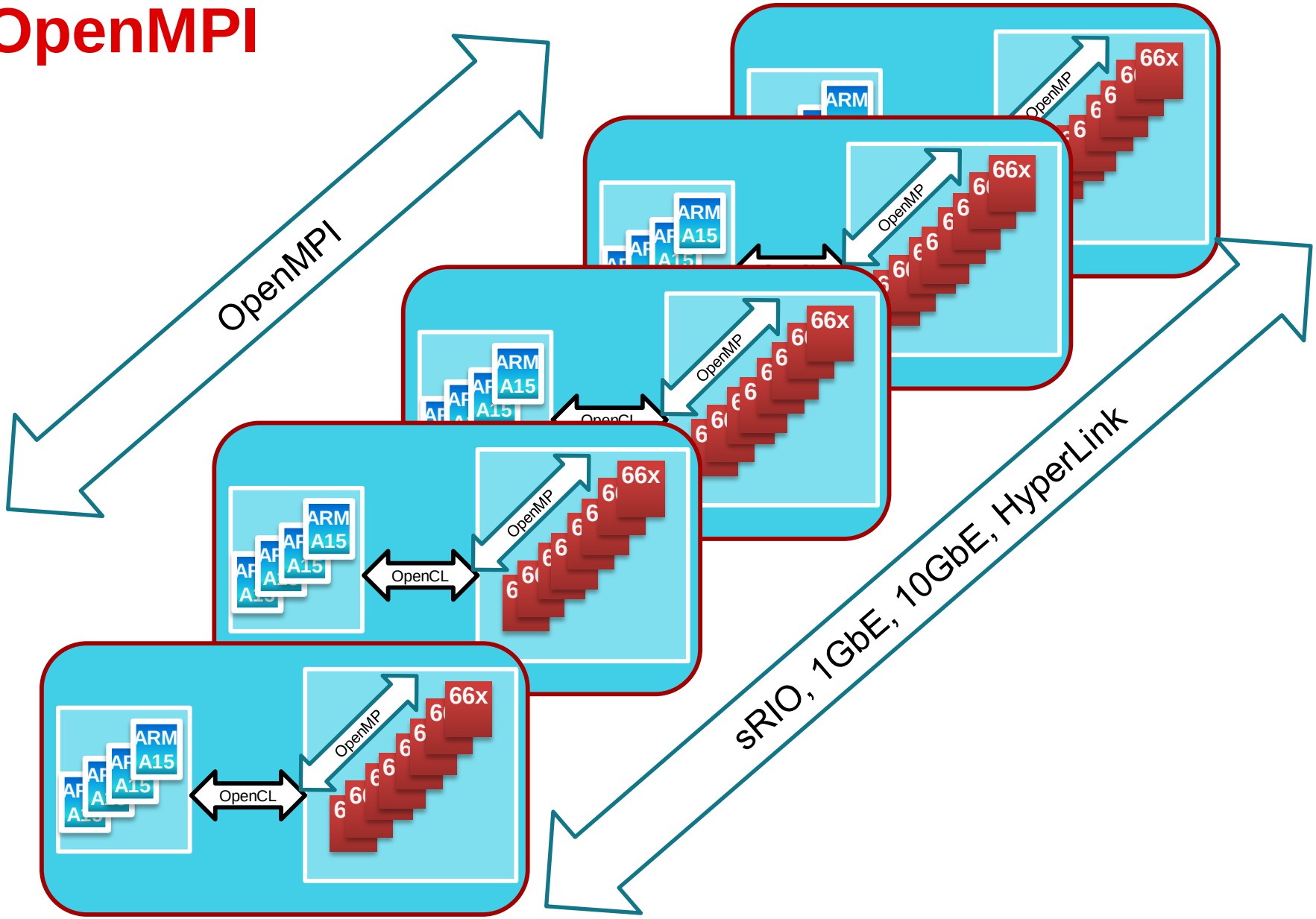
OpenMP Accelerator Model for Heterogeneous Acceleration

- Part of OpenMP 4.0 Standard, extends OpenMP to heterogeneous SoCs
- Enables dispatch of tasks from A15s to C66x DSPs (host to accelerator)

- Scalable and portable
- Existing code can be adapted quickly with very few additional lines
- OpenCL offers more control over memory management, data movement but OpenMP is a better fit for parallelizing regions

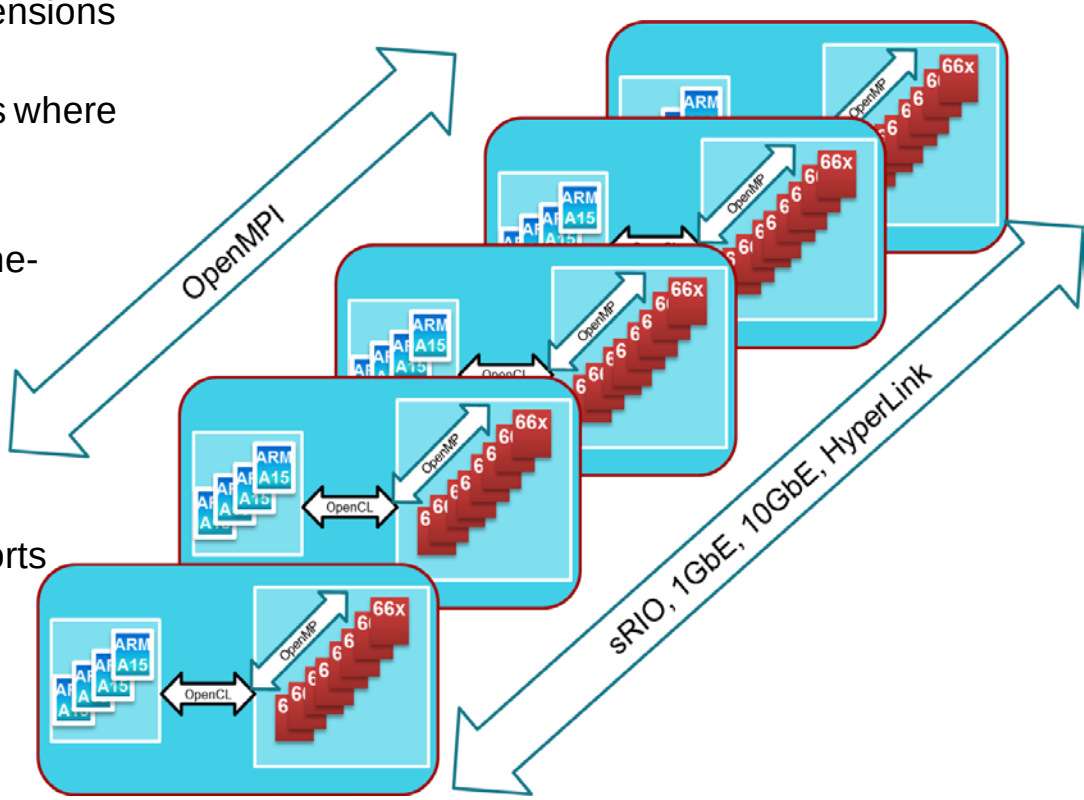


OpenMPI


OpenMPI
ARM
A15
OpenMP
66x
OpenCL
sRIO, 1GbE, 10GbE, HyperLink

OpenMPI for Multiple Devices

- OpenMPI over Ethernet, SRIO or Hyperlink
  - SRIO and Hyperlink are TI-specific extensions to the byte transport layer of OpenMPI
  - Allows use of OpenMPI in architectures where SRIO is the backplane of choice
  - Hyperlink functionality:
    - Message passing between KeyStone-based SoCs
- OpenMPI implementation can pick the "fastest" transport available
  - Uses same API across multiple transports
- Standardized, portable APIs
- Large existing support ecosystem
  - Profilers, debuggers, training, etc.



Software Libraries

User view

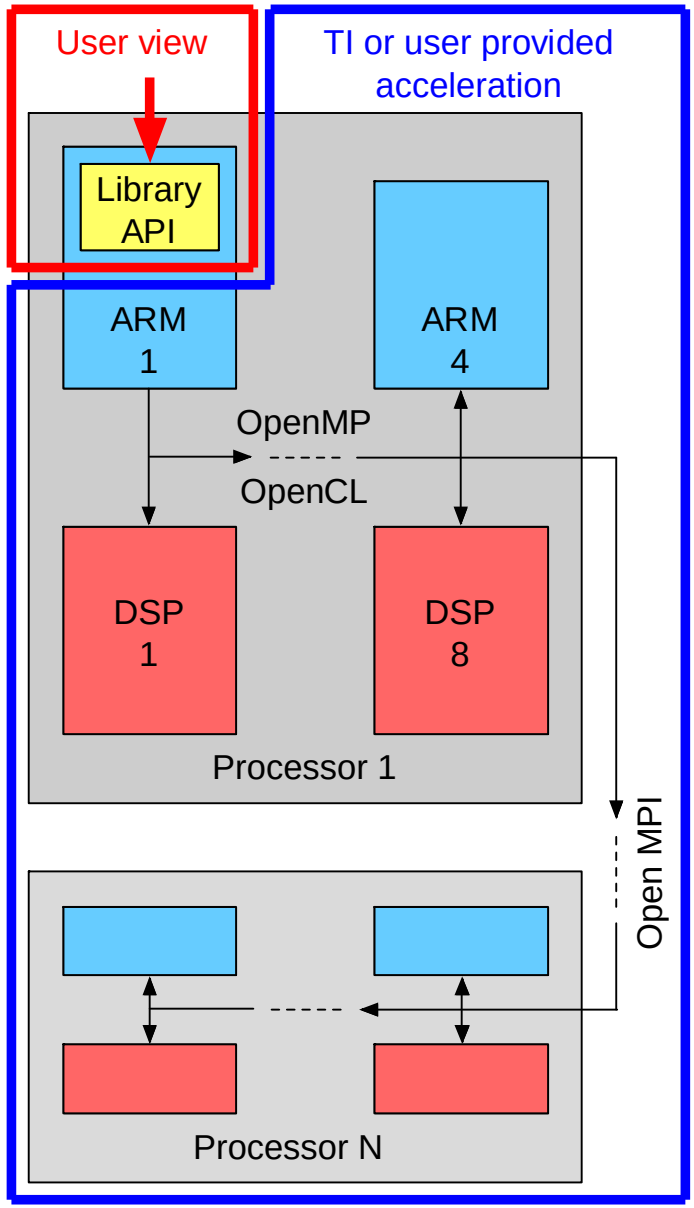
- Embedded Linux running on the ARM
- Standard gcc tool chain
- Simply link to a TI provided library with an ARM callable API to accelerate applications using multiple ARM cores, DSP cores and processors as appropriate
- Use TI provided tools and examples to write new applications and libraries which use multiple ARM cores, DSP cores and processors to accelerate performance
- **Includes BLAS and FFTW libraries**

Using multiple cores on a single processor

- OpenMP for shared memory parallelization across ARM cores
- OpenCL or OpenMP Accelerator for heterogeneous acceleration with multiple DSP cores

Using multiple processors

- Open MPI over Ethernet, SRIO or Hyperlink



K2H Performance with OpenCL/OpenMP-DSP

Benchmark	ARM MPCore (4 Cortex-A15) Gflops	DSP (8 C66x) Gflops	Speedup	Notes
sgemm (single precision general matrix multilply)	7.08	104.79	**14.80**	• 4K x 4K matrix • OpenCL used to dispatch to all DSP cores
dgemm (double precision general matrix multiply)	5.97	26.99	**4.52**	• 4K x 4K matrix • OpenCL used to dispatch to one DSP core, OpenMP across DSP cores
FFTW 3.3.4 (www.fftw.org)	1.21 (single core performance * 4)	6.07	**5.02**	• 1 D, double precision, complex to complex • 1M points • OpenCL used to dispatch to one DSP core, OpenMP across DSP cores

K2H Board Configuration

- XTCIEVMK2X Rev 4.0
- MCSDK HPC 3.0.2
- C66x tools 8.0.0B2
- ARM gcc 4.7.2
- ARM cores clocked at 1.2 GHz
- DSP cores clocked at 1.228 GHz
- 8GB Memory, 6.5 for ARM/Linux, 1.5 for OpenCL

Benchmark Information

sgemm/dgemm

- ARM version tuned with ATLAS (Automatically Tuned Linear Algebra Software)
- Scalar-matrix-matrix product and add the result to a scalar matrix product
  - C = alpha*A*b + beta*C

Thank You

Questions?