arm

OpenSHMEM on Arm

Pavel Shamis/Pasha

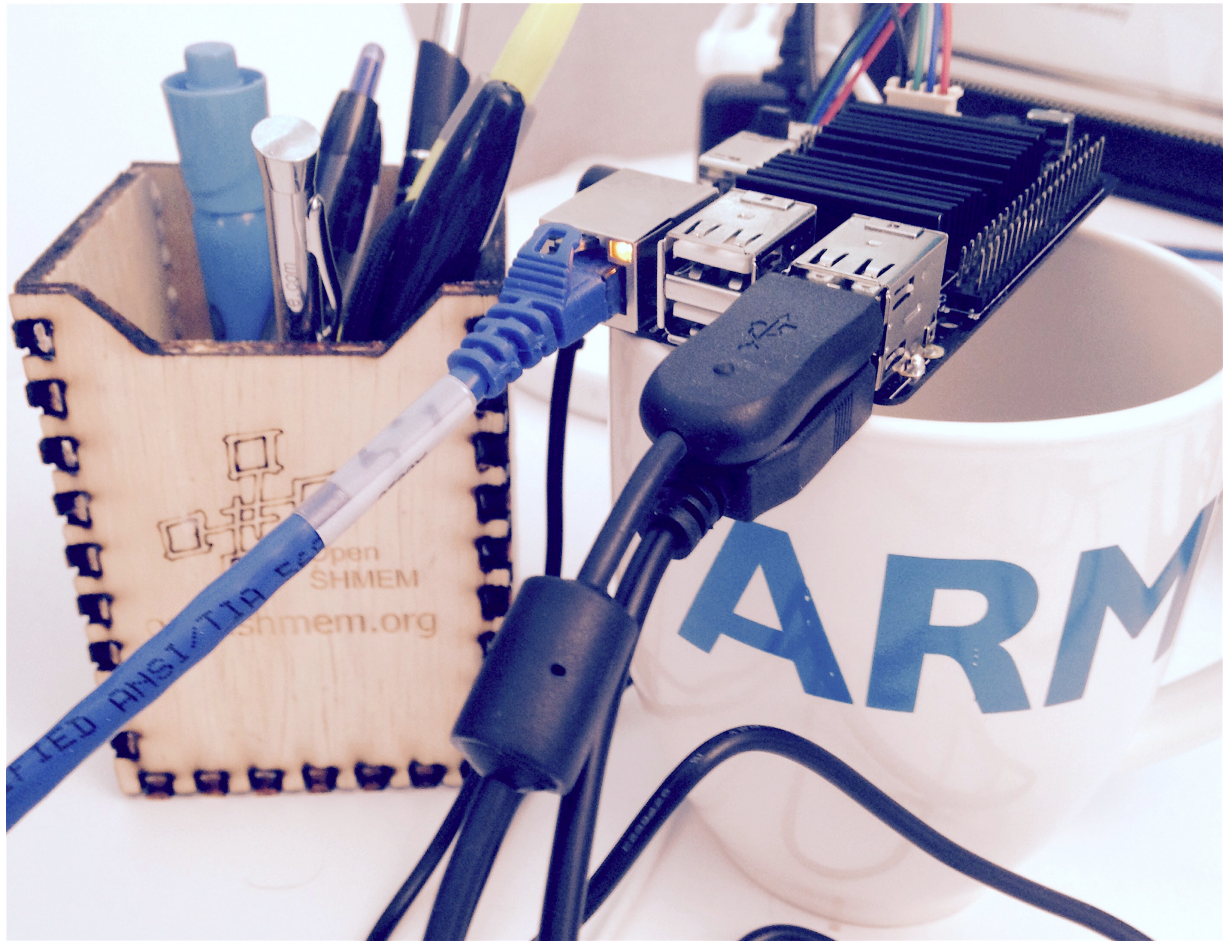
Principal Research Engineer

OpenSHMEM Workshop 2017

Annapolis, MD

© 2017 Arm Limited

OpenSHMEM on Arm ?

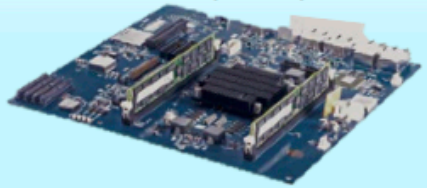


© 2017 Arm Limited

Arm servers from multiple manufacturers

HP ProLiant
(Applied Micro, TI)

Softiron 64-0800
(AMD)



Gigabyte R120-P30
(Applied Micro)

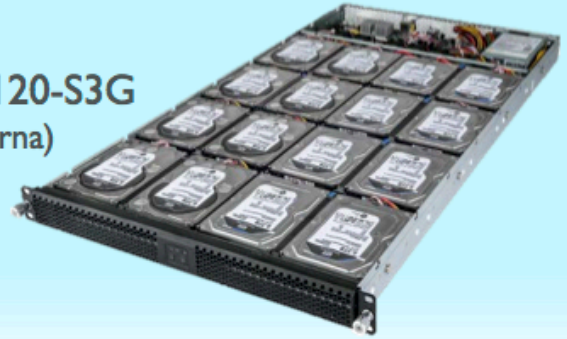
Wiwynn LN1148-10SL
(Marvell)

Gigabyte MT70-HD0
(Cavium)

Cirrascale RM1905D
(Applied Micro)

Mitac Datun
(Applied Micro)

Gigabyte D120-S3G
(Annapurna)



© 2017 Arm Limited

Arm Overview

© 2017 Arm Limited

arm

An introduction to Arm

ARM is the world's leading semiconductor intellectual property supplier.

We license to over 350 partners, are present in 95% of smart phones, 80% of digital cameras, 35% of all electronic devices, and a total of 60 billion ARM cores have been shipped since 1990.

Our CPU business model:

License technology to partners, who use it to create their own system-on-chip (SoC) products.

We may license an instruction set architecture (ISA) such as "ARMv8-A")

or a specific implementation, such as "Cortex-A72".

Partners who license an ISA can create their own implementation, as long as it passes the compliance tests.

...and our IP extends beyond the CPU

© 2017 Arm Limited
arm

A partnership business model

A business model that shares success

- Everyone in the value chain benefits
- Long term sustainability

Design once and reuse is fundamental

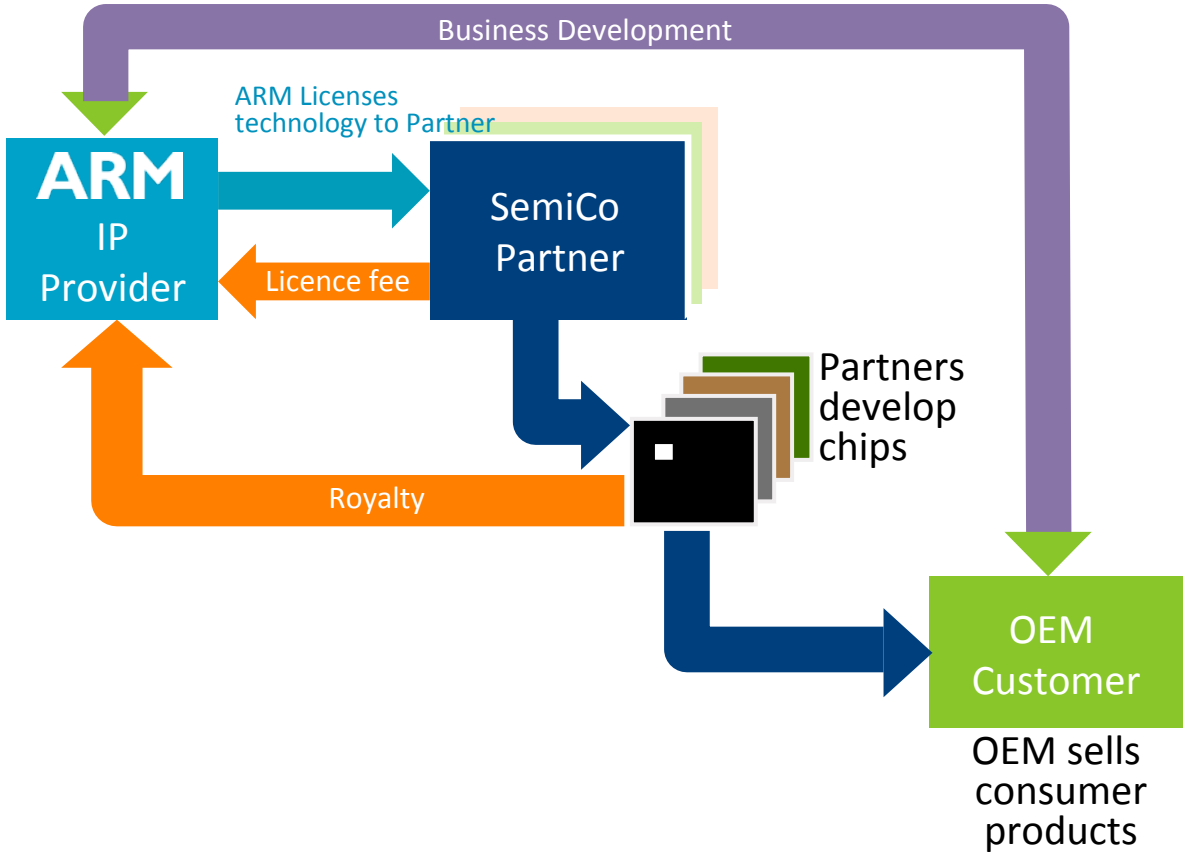
- Spread the cost amongst many partners
- Technology reused across multiple applications
- Creates market for ecosystem to target
  - Re-use is also fundamental to the ecosystem

Upfront license fee

- Covers the development cost

Ongoing royalties

- Typically based on a percentage of chip price
- Vested interest in success of customers



Approximately 1350 licenses
Grows by ~120 every year

More than 440 potential royalty payers

14.8bn+ ARM-powered chips in 2015

© 2017 Arm Limited
arm

Partnership

© 2017 Arm Limited

Range of SoCs addressing infrastructure

XILINX ZYNQ UltraSCALE+

NXP QorIQ® Layerscape 2080A

Mellanox TECHNOLOGIES BlueField

socionext SC2A11

QUALCOMM Centriq 2400

applied micro X-Gene 3™

ALTERA Stratix 10 FPGA ▪ SoC

CAVIUM THUNDERX2

One size does not fit all

© 2017 Arm Limited
arm

Serious Arm HPC deployments starting in 2017

Two big announcements about ARM in HPC in Europe:

insideHPC

Bull Atos to Build HPC Prototype for Mont-Blanc Project using Cavium ThunderX2 Processor

January 16, 2017 by staff

Today the Mont-Blanc European project announced it has selected Cavium's ThunderX2 ARM server processor to power its new HPC prototype.

The new Mont-Blanc prototype will be built by Atos, the coordinator of phase 3 of Mont-Blanc, using its Bull expertise and products. The platform will leverage the infrastructure of the Bull sequana pre-exascale supercomputer range for network, management, cooling, and power. Atos and Cavium signed an agreement to collaborate to develop this new platform, thus making Mont-Blanc an Alpha-site for ThunderX2.

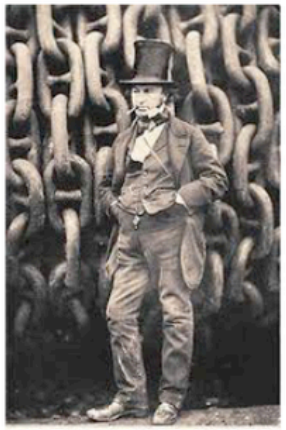
Bull atos technologies

University of BRISTOL GW4 January 17th 2017

Announcing the **GW4 Tier 2 HPC service, 'Isambard'**: named after Isambard Kingdom Brunel

System specs:

- Cray CS-400 system
- **10,000+** ARMv8 cores
- HPC optimised software stack
- Technology comparison:
  - x86, KNL, Pascal
- To be installed March-Dec 2017
- £4.7m total project cost over 3 years



I.K.Brunel 1804-1859

Simon McIntosh Smith, simonm@cs.bris.ac.uk, @simonmcs 5 bristol.ac.uk

© 2017 Arm Limited
arm

Japan

Post-K: Fujitsu HPC CPU to Support ARM v8

ARM FUJITSU

Post-K fully utilizes Fujitsu proven supercomputer microarchitecture

Fujitsu, as a lead partner of ARM HPC extension development, is working to realize ARM Powered® supercomputer w/ high application performance

ARM v8 brings out the real strength of Fujitsu's microarchitecture

HPC apps acceleration feature	Post-K	FX100	FX10	K computer
FMA: Floating Multiply and Add	✔	✔	✔	✔
Math. acceleration primitives*	✔Enhanced	✔	✔	✔
Inter core barrier	✔	✔	✔	✔
Sector cache	✔Enhanced	✔	✔	✔
Hardware prefetch assist	✔Enhanced	✔	✔	✔
Tofu interconnect	✔Integrated	✔Integrated	✔	✔

* Mathematical acceleration primitives include trigonometric functions, sine & cosines, and exponential...

slides from Fujitsu at ISC'16

© 2017 Arm Limited

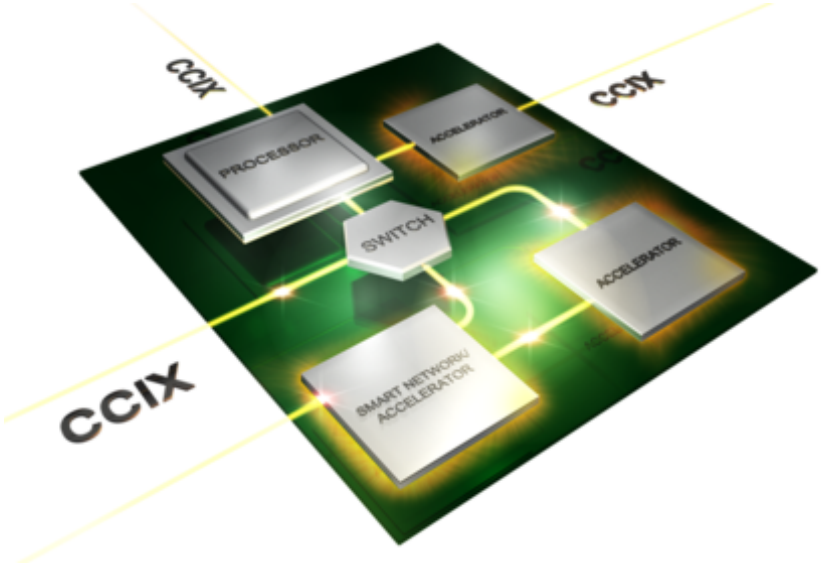
Integration with Network Interconnects

© 2017 Arm Limited

arm

CCIX

CCIX Cache Coherent Interconnect for Accelerators



Accelerators and Network (NIC/HCA/etc.) as a first class "citizen" in the system

Seamless process and accelerator hardware cache coherence support

Low-latency and high-bandwidth

Allow in-line acceleration

- Bump in the wire processing (network packet processing, storage acceleration, etc.)

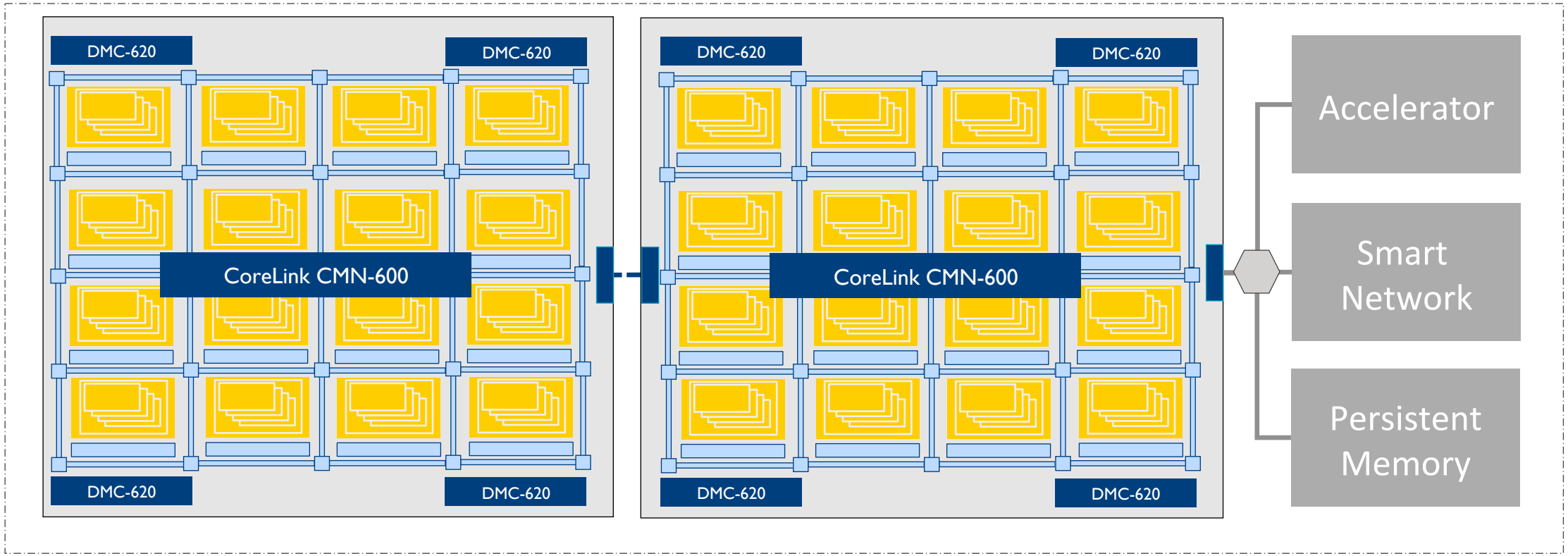
Allows "off-line" acceleration (co-processor model)

Driver-less / interrupt-less usage model

http://www.ccixconsortium.com

© 2017 Arm Limited

Scale-up server node



Shared virtual memory system

© 2017 Arm Limited

CCIX multichip connectivity and topologies

New class of interconnect providing high performance, low latency for new accelerators use cases

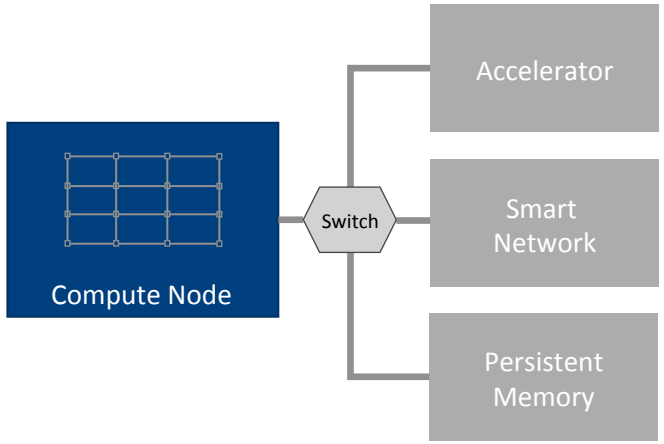
- CCIX defines 25GT/s (3x performance*)
- Examining 56GT/s (7x performance*) and beyond
- Enabling low latency via light transaction layer

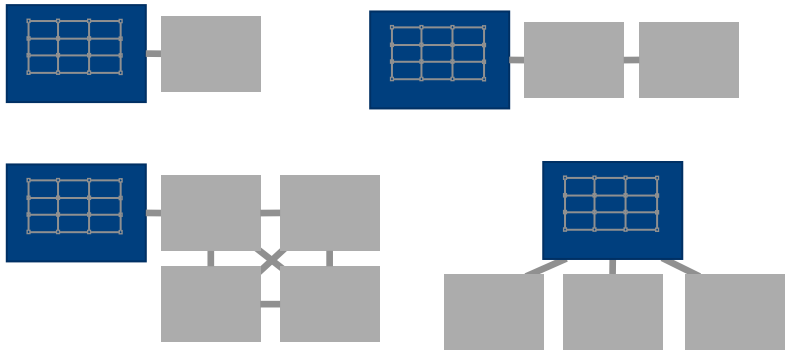
Flexible, scalable interconnect topologies

- Flexible point-to-point, daisy chained and switched topologies

Simplified deployment by leveraging existing PCIe hardware and software infrastructure

- Runs on existing PCIe transport layer and management stack
- Coexist with legacy PCIe designs





* Note: Based on PCIe Gen3 Performance

© 2017 Arm Limited

Building CCIX devices

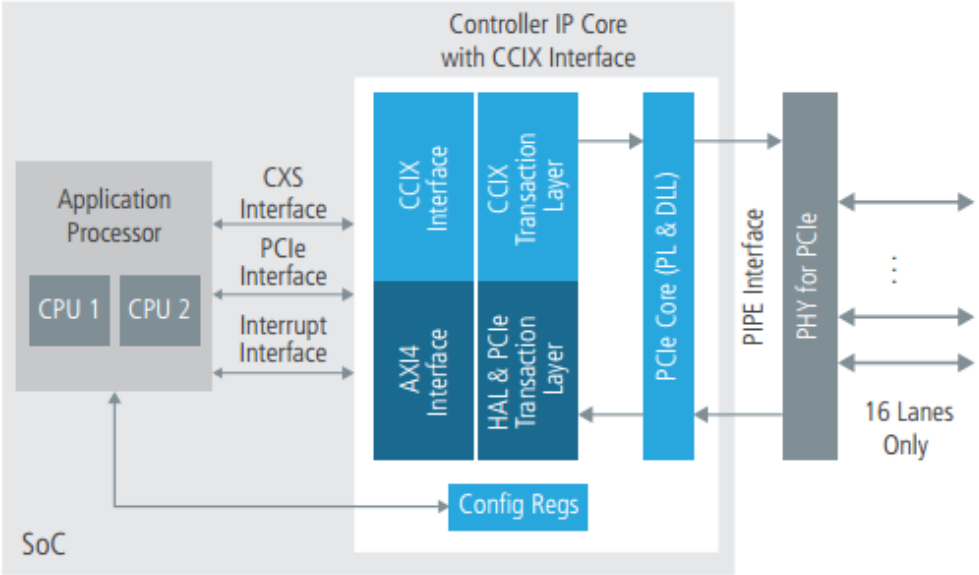
Cadence® IP for CCIX

- Built upon silicon proven PCIe solutions

IP products:

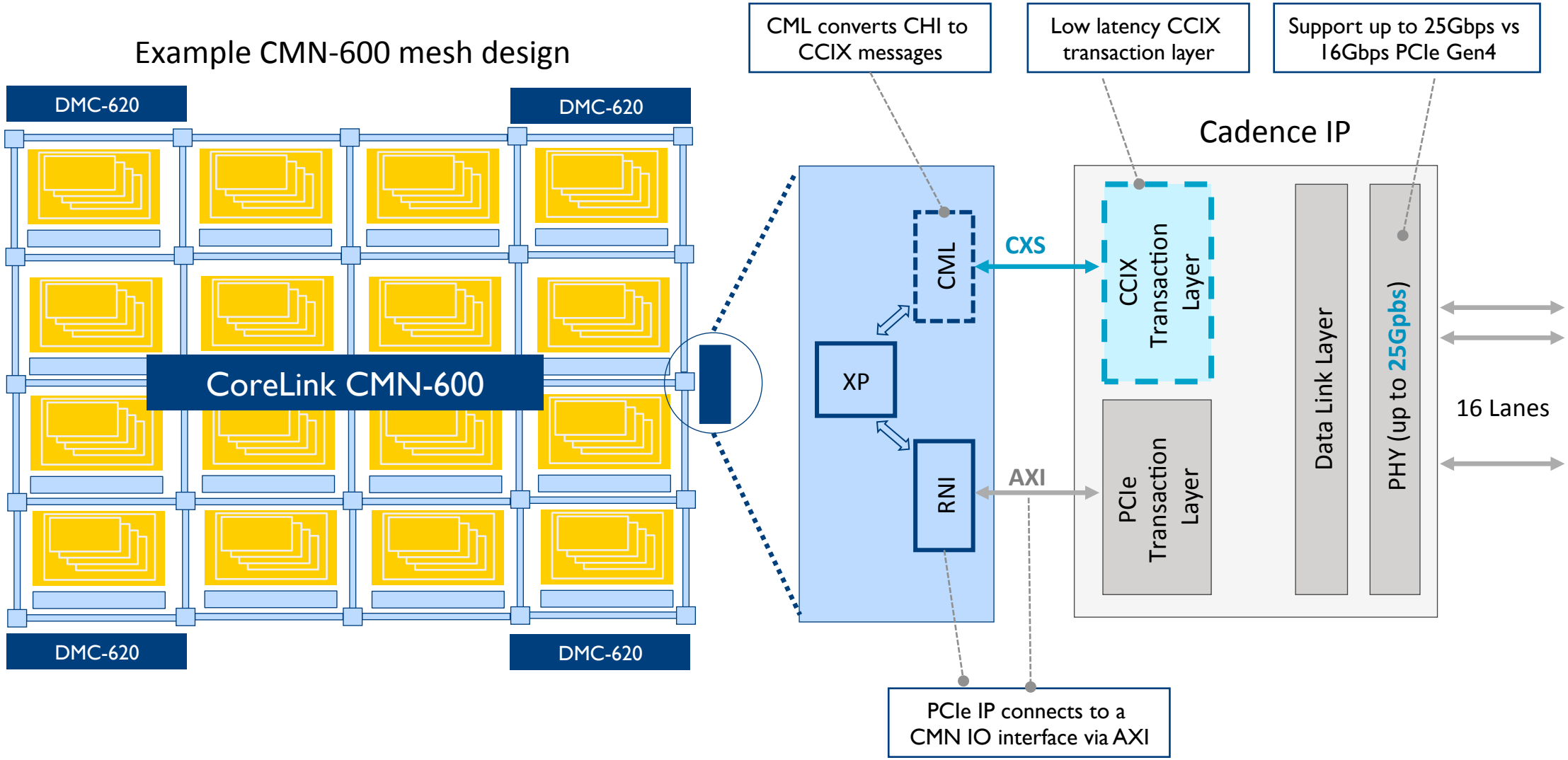
- Controller IP – Provides the CCIX transaction and data link layers.
- PHY IP – Provides the high performance SERDES physical layer supporting speeds up to 25Gpbs.
- Verification IP – Provides the necessary test infrastructure to verify CCIX designs.

Cadence Controller & PHY IP



© 2017 Arm Limited

Cadence CCIX integration



© 2017 Arm Limited

Gen-Z

All data is accessed by some form of a Read or a Write

Example of reads: DDR Row + Column Read, PCI DMA Read, SCSI Write, Socket Read, File Read, RDMA Read

Example of writes: DDR Row + Column Write, PCI DMA Write, SCSI Read, Socket Write, File Write, RDMA Write

The Goal: Simplify world to memory semantic Reads & Writes

 © 2017 Arm Limited

arm

Gen-Z Overview

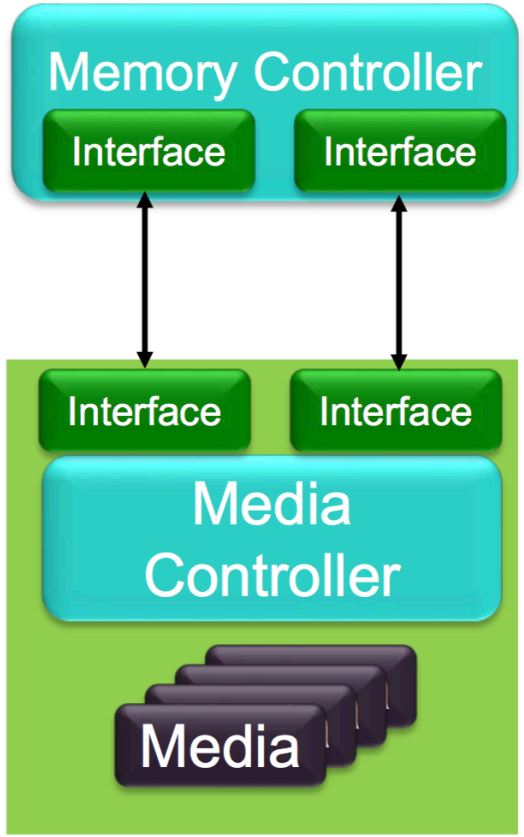
An open, standards-based, scalable, system interconnect and protocol.

Optimized to support memory semantic communications

Breaks Processor-Memory Interlock

Split controller model

- Memory controller
  - Initiates high-level requests—Read, Write, Atomic, Put / Get, etc.
  - Enforces ordering, reliability, path selection, etc.
- Media controller
  - Abstracts memory media
  - Supports volatile / non-volatile / mixed-media
  - Performs media-specific operations
  - Executes requests and returns responses
  - Enables data-centric computing (accelerator, compute, etc.)



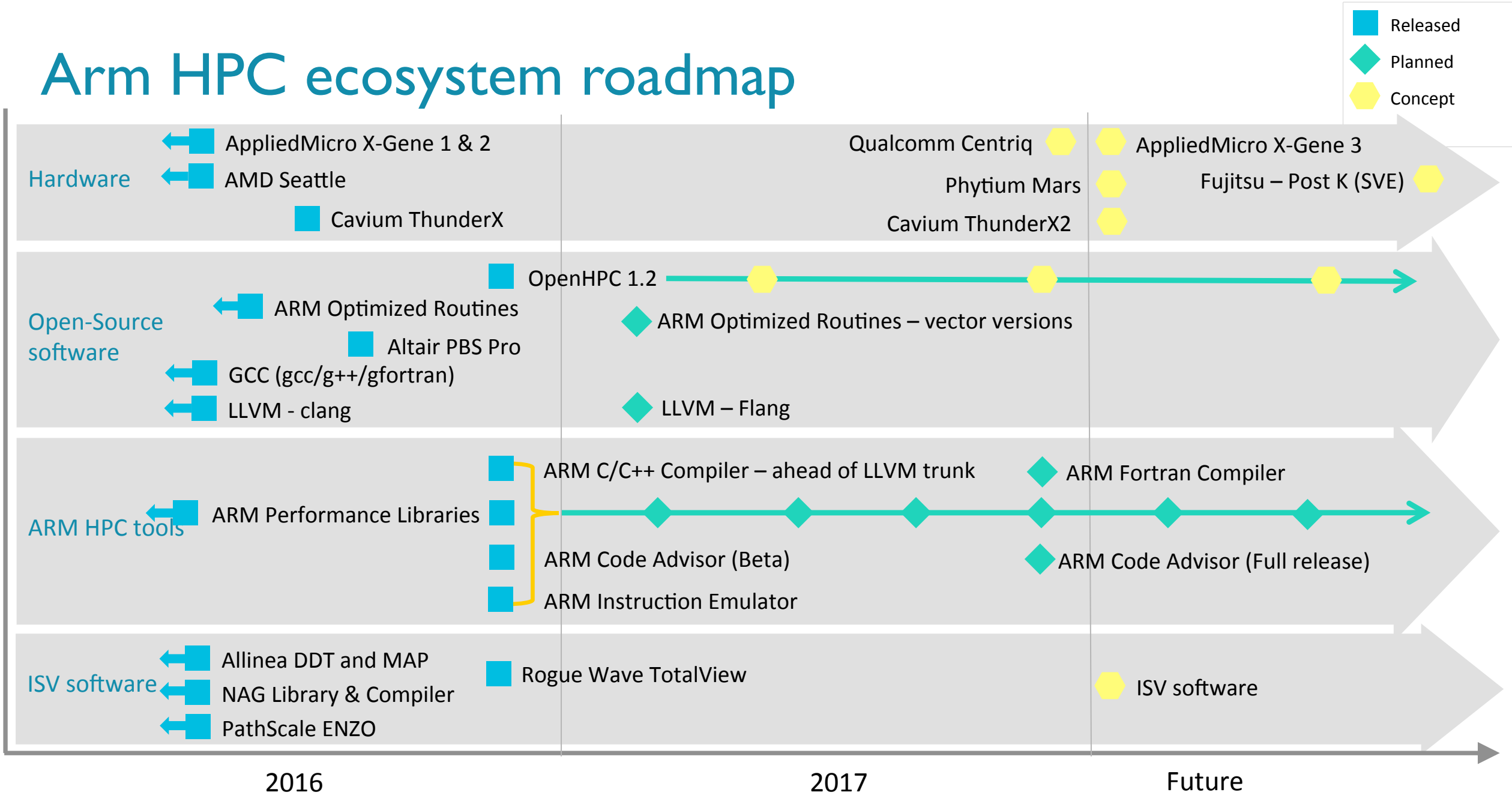
© 2017 Arm Limited

Software Eco-system

© 2017 Arm Limited

arm

Arm HPC ecosystem roadmap



© 2017 Arm Limited

Linux / FreeBSD w/ AARCH64 support

Debian 8 adds AARCH64 – April 2015

12.04LTS & 14.04LTS released ← Also 14.10 & 15.04 releases

Fedora 22 released – May 2015
Fedora 23 released – Nov 2015

Red Hat Enterprise Linux Server for ARM
7.2 BETA – Sept, 2015

CentOS Linux 7 for AArch64
GA – August 2015

OpenSUSE 13.2 – Nov 2014

SUSE Launches Partner Program to Bring
SUSE Linux Enterprise 12 to 64-bit ARM
July 2015 @ ISC

FreeBSD ← Engaged with FreeBSD foundation / Semi-half & Cavium to get FreeBSD on ARMv8
FreeBSD Beta version demo'd by Semihalf – Nov. 2015

© 2017 Arm Limited
arm

Open source and commercial compilers

- GCC
  - C, C++, Fortran
  - OpenMP 4.0

- PathScale
  - C, C++ Fortran
  - OpenACC
  - OpenMP 4.0

- LLVM
  - C, C++
  - OpenMP 3.1, (4.0 coming soon)
  - Fortran coming Q1 2017

- NAG
  - Fortran
  - OpenMP 3.1

- ARM C/C++ Compiler
  - LLVM based
  - Includes SVE

© 2017 Arm Limited

Celebrating 5 years of Open Source Engineering on ARM

"The ARM situation has just improved tremendously over the last several years. It used to be a major pain to me, it has gone to almost being entirely painless."

- Linus Torvalds
May 2015

Note:
Linus Torvalds Image from Linux Foundation. Icons made by Freepik. Logos & trademarks remain the property of their respective owners and represent a range of products and services supported by Linaro.

24TB

data from June 2014 - May 2015
615,000 downloads from **>100 countries**

16 Connects
14 Cities on 3 continents

32 member companies
Six members at launch

4,410
gallons consumed at Linaro Connects

1,141,014

minutes of videos showing demos, talks and training sessions watched

More than 220 Engineers
from seed of twenty

#3
company contributor for
Linux Kernels 3.11 - 3.18

11,589
patches upstream since 2011

~20
hardware platforms

www.linaro.org
www.96boards.org

50,217
Wiki pages

>1 Million
website users

>50
Monthly Software Releases

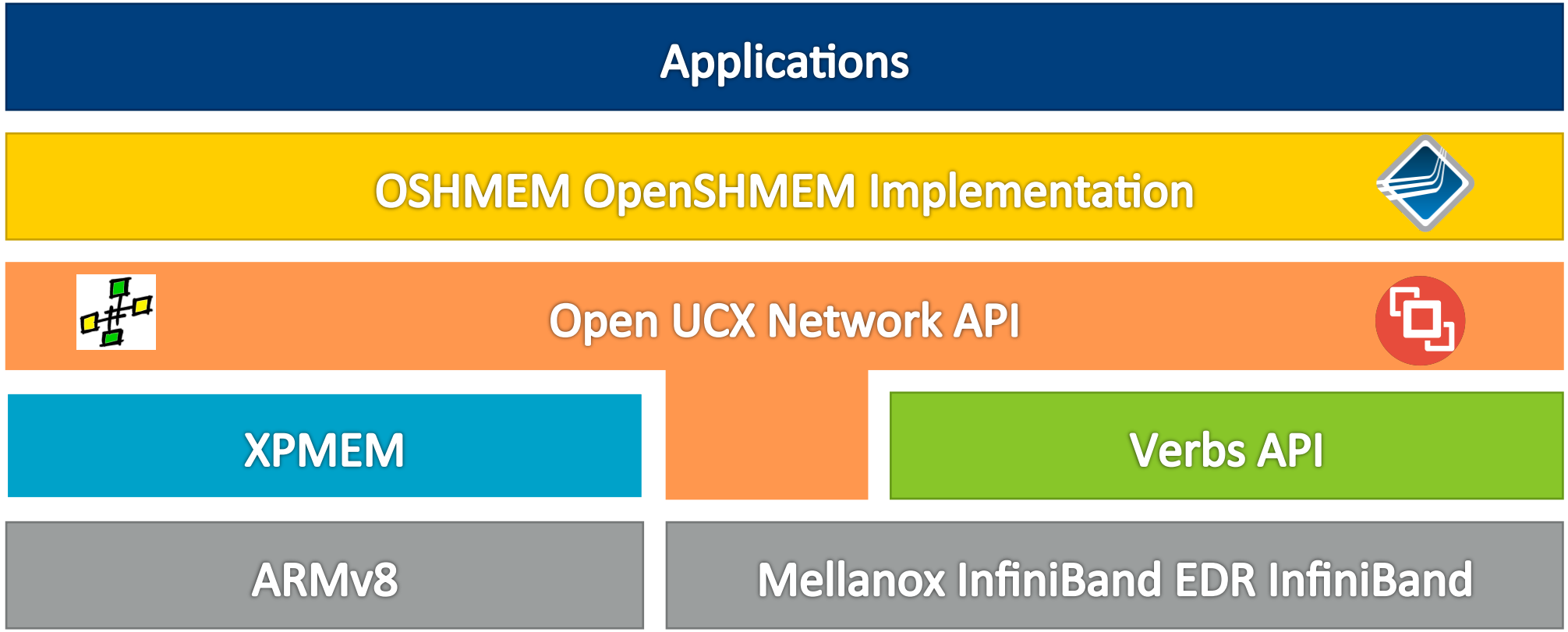
openembedded · yocto PROJECT · redhat

OpenSHMEM

© 2017 Arm Limited

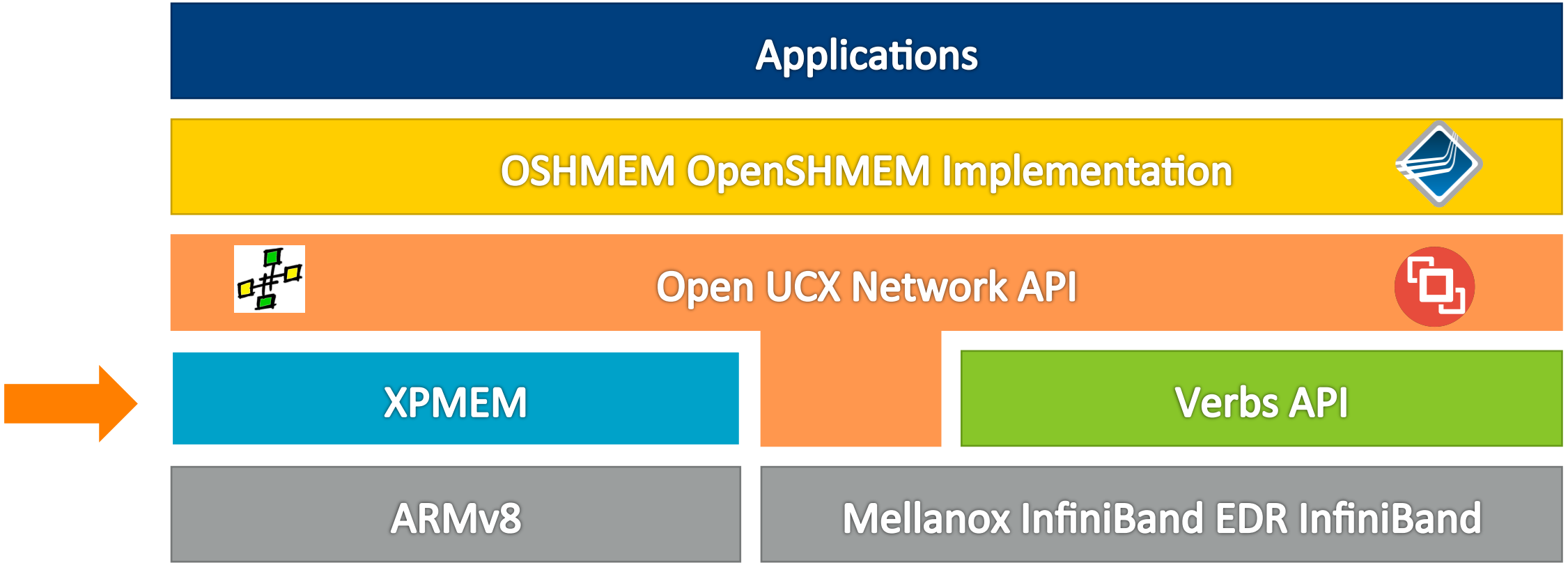
arm

Hardware Software Stack Overview



© 2017 Arm Limited

XPMEM



© 2017 Arm Limited

XPMEM

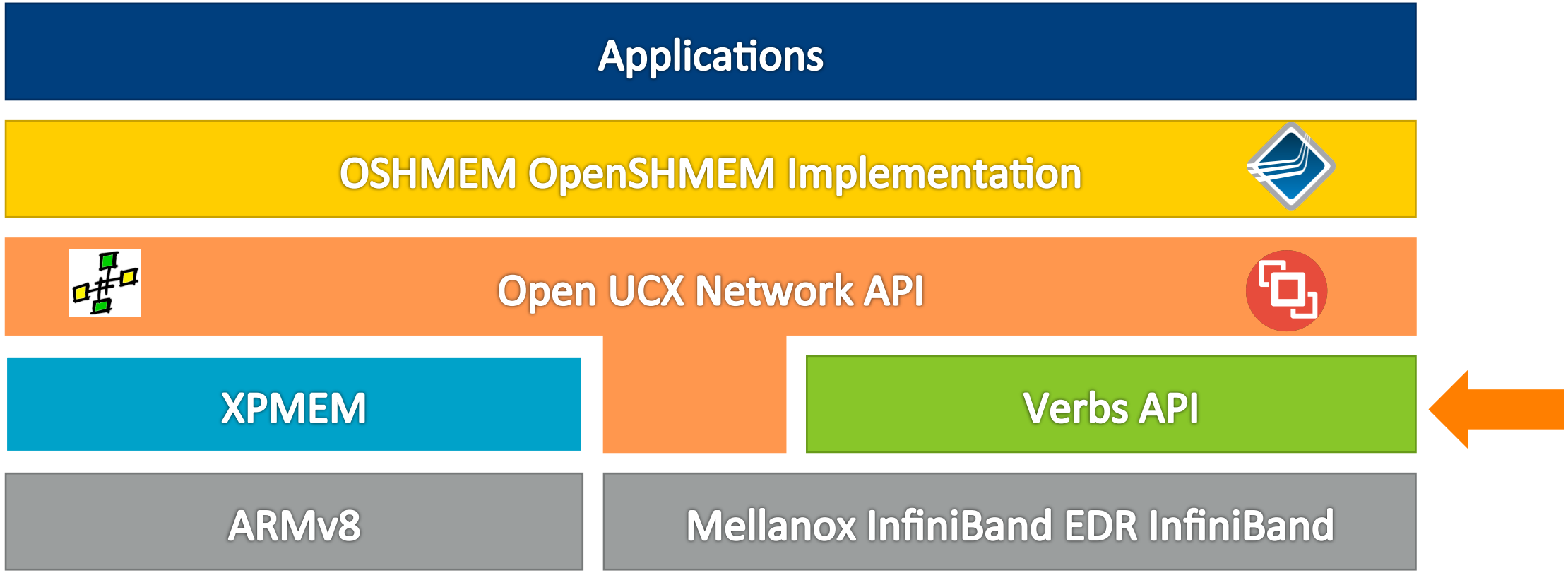
	XPMEM	KNEM	CMA	MMAP
System Call	No	Yes	Yes	Yes
Low Latency	V	X	X	V
High Bandwidth	V	V	V	V
Any Virtual Address	V	V	V	X
Open Source	V	V	V	V
Linux default support	X	X	V	V

XPMEM - Cross Memory Attach

- The code was updated to compile and run on ARMv8 (https://github.com/hjelmn/xpmem)
- 3rd party kernel module based on SGI/Cray XPMEM and maintained by LANL

© 2017 Arm Limited

XPMEM



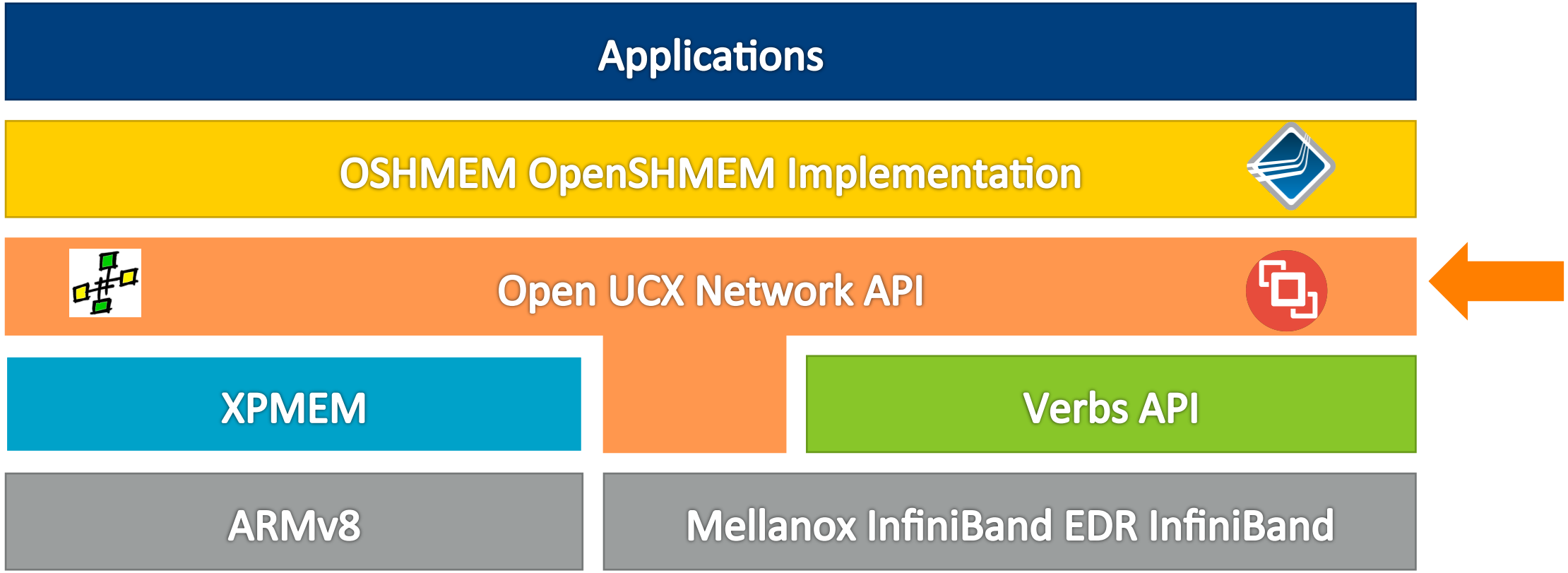
 © 2017 Arm Limited

VERBs API support on Arm

- Besides bug fixes not much work was required
- Mellanox OFED 2.4 and above supports Arm
- Linux Kernel 4.5.0 and above (maybe even earlier)
- Linux Distribution Support – on going process
- OFED – no ARMv8 support

 © 2017 Arm Limited

UCX



© 2017 Arm Limited

OpenUCX v1.2

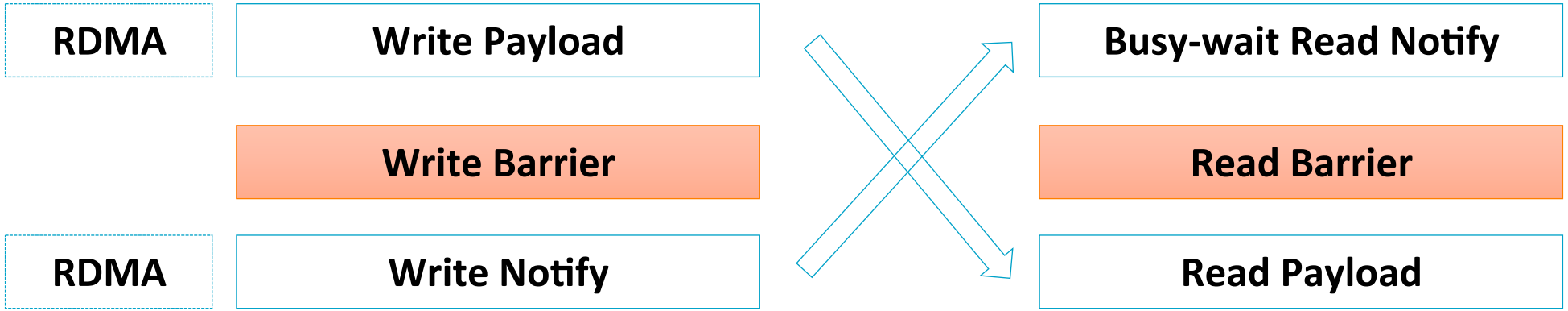
- The first official release from OpenUCX community
  - https://github.com/openucx/ucx/releases/tag/v1.2.0
- Features
  - Support for InfiniBand and RoCE
    - Transports RC, UD, DC
  - Support for Accelerated Verbs – 40% speedup on ARM compared to vanilla Verbs
  - Support for Cray Aries and Gemini
  - Support for Shared Memory: KNEM, CMA, XPMEM, Posix, SySV
  - Support for x86 (with AVX), **ARMv8 (with NEON)**, Power
  - Efficient memory polling – 36% increase in efficiency on ARM
  - UCX interface is integrated with MPICH, OpenMPI, OSHMEM, ORNL-SHMEM, etc.

© 2017 Arm Limited

Relaxed memory ordering

Memory Barriers

- Multithread environment
- Software-hardware interaction
- You can “fish” for these bugs in MPI implementations around Eager-RDMA and shared memory protocols



Maranget, Luc, Susmit Sarkar, and Peter Sewell. "A tutorial introduction to the ARM and POWER relaxed memory models." Draft available from http://www. cl. cam. ac. uk/~ pes20/ppc-supplemental/test7. pdf (2012).

 © 2017 Arm Limited

Memory Barriers

DSB

- Completion semantics
- Data Synchronization Barriers halt execution until:
  - All explicit memory accesses before the instruction complete
  - All cache, branch prediction and TLB operations before the instruction complete
- Interaction with external devices (PCIe doorbells)
  - Device drivers

```
#define ucs_memory_bus_fence()        asm volatile ("dsb sy" ::: "memory");
#define ucs_memory_bus_store_fence()  asm volatile ("dsb st" ::: "memory");
#define ucs_memory_bus_load_fence()   asm volatile ("dsb ld" ::: "memory");
```

Examples https://github.com/openucx/ucx/blob/master/src/ucs/arch/aarch64/cpu.h#L25

© 2017 Arm Limited

Memory Barriers - continued

DMB

- ISH* domain on Linux
- Poll-flag, barrier, data

```
#define ucs_memory_cpu_fence()        asm volatile ("dmb ish" ::: "memory");
#define ucs_memory_cpu_store_fence()  asm volatile ("dmb ishst" ::: "memory");
#define ucs_memory_cpu_load_fence()   asm volatile ("dmb ishld" ::: "memory");
```

Examples https://github.com/openucx/ucx/blob/master/src/ucs/arch/aarch64/cpu.h#L25

© 2017 Arm Limited

Low-level timers

Low-level timers

- Typically found in benchmarks and MPI
- Code examples https://github.com/openucx/ucx/blob/master/src/ucs/arch/aarch64/cpu.h#L35

```
static inline uint64_t ucs_arch_read_hres_clock(void)
{
    uint64_t ticks;
    asm volatile("isb" : : : "memory");
    asm volatile("mrs %0, cntvct_el0" : "=r" (ticks));
    return ticks;
}

static inline double ucs_arch_get_clocks_per_sec()
{
    uint32_t freq;
    asm volatile("mrs %0, cntfrq_el0" : "=r" (freq));
    return (double) freq;
}
```


© 2017 Arm Limited

Cache lines on ARM

Not all cache-lines are 64Byte !

- Implementation dependent
- Don’t make assumptions about 64B cache line size

http://xeroxnostalgia.com/duplicators/xerox-9200/

 © 2017 Arm Limited

MLX5 UCX Transport

MLX5 – specially optimized transport implemented on top of ConnectX Hardware Abstraction Layer

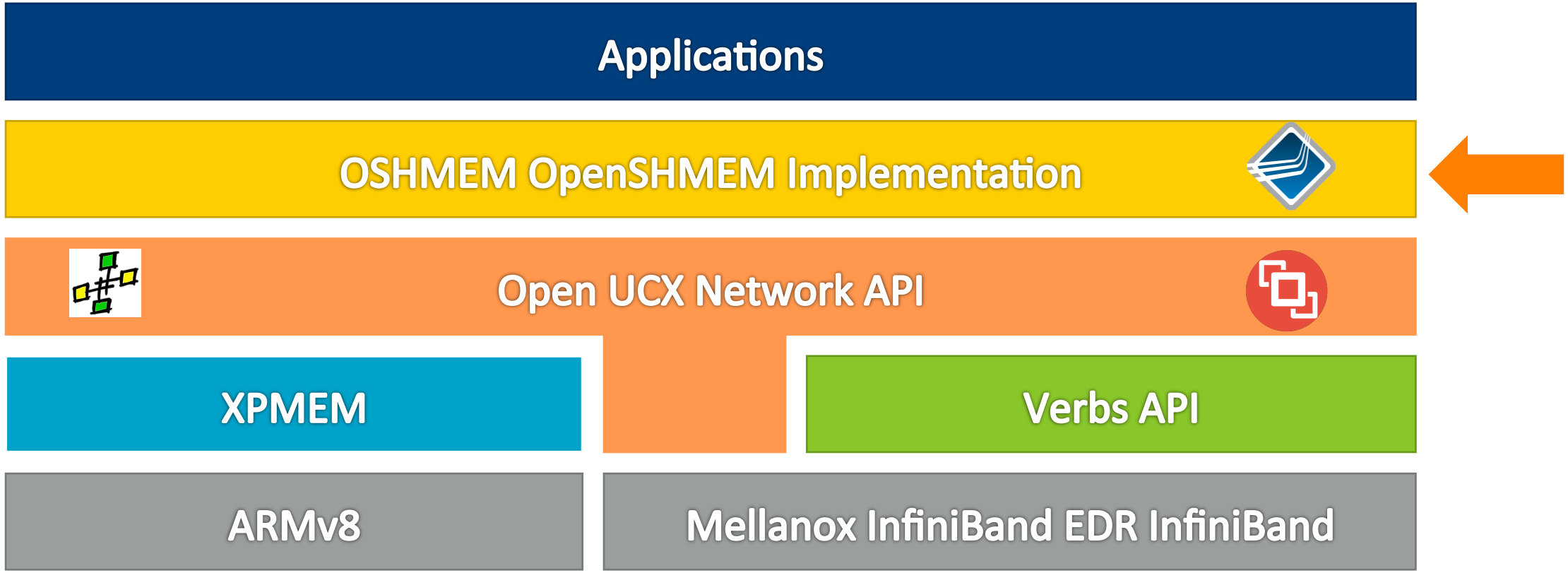
The layer initialize translates UCP request to InifniBand request and rings the doorbell

- The code responsible for initialization of the request was updated to leverage ARM vector instructions (NEON): https://github.com/openucx/ucx/blob/891e20ef90257d1e2721da52461b0261220c82d8/src/uct/ib/mlx5/ib_mlx5.inl#L160

```
#if defined(__SSE4_2__)
    *(__m128i*)raddr = _mm_shuffle_epi8(
                _mm_set_epi64x(rdma_rkey, rdma_raddr),
                _mm_set_epi8(0, 0, 0, 0,            /* reserved */
                             8, 9, 10, 11,          /* rkey */
                             0, 1, 2, 3, 4, 5, 6, 7 /* rdma_raddr */
                             ));
#elif defined(__ARM_NEON)
    uint8x16_t table =  {7,  6,  5, 4, 3, 2, 1, 0, /* rdma_raddr */
                         11, 10, 9, 8,             /* rkey */
                         16,16,16,16};             /* reserved (set 0) */
    uint64x2_t data = {rdma_raddr, rdma_rkey};
    *(uint8x16_t *)raddr = vqtbl1q_u8((uint8x16_t)data, table);
#else
    raddr->raddr = htobe64(rdma_raddr);
    raddr->rkey  = htonl(rdma_rkey);
#endif
```


© 2017 Arm Limited

OSHMEM



 © 2017 Arm Limited

OSHMEM

OSHMEM – a very thin layer on top of UCX UCP API

Worked out of the box, besides minor bugs in the integration layer

 © 2017 Arm Limited

arm

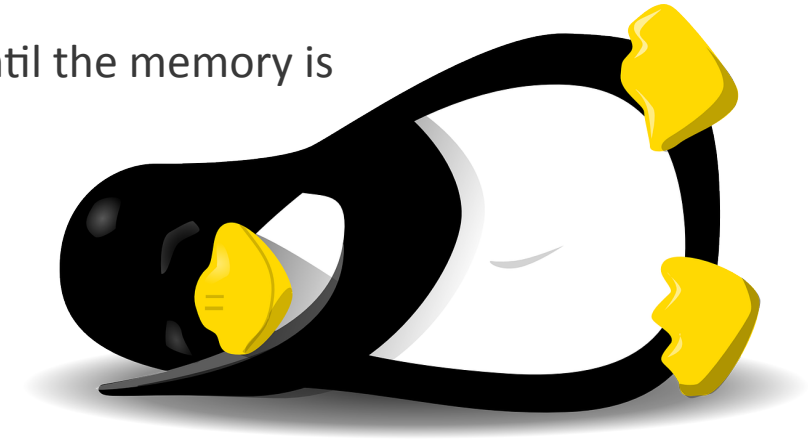
OSHMEM

SHMEM_WAIT/SHMEM_WAIT_UNTIL block until memory is updated by remote process

```
void shmem_int_wait(volatile int *ivar, int cmp_value);
void shmem_int_wait_until(volatile int *ivar, int cmp, int cmp_value);
void shmem_long_wait(volatile long *ivar, long cmp_value);
void shmem_long_wait_until(volatile long *ivar, int cmp, long cmp_value);
void shmem_longlong_wait(volatile long long *ivar, long long cmp_value);
```

Typically implemented as a busy-wait loop

- ARM Wait-For-Event (WFE) – provides an opportunity to pause the core until the memory is updates (or an interrupt occurs)
- It is used in linux spinlock and it is perfect fit for SHMEM



 © 2017 Arm Limited

Preliminary Results

© 2017 Arm Limited

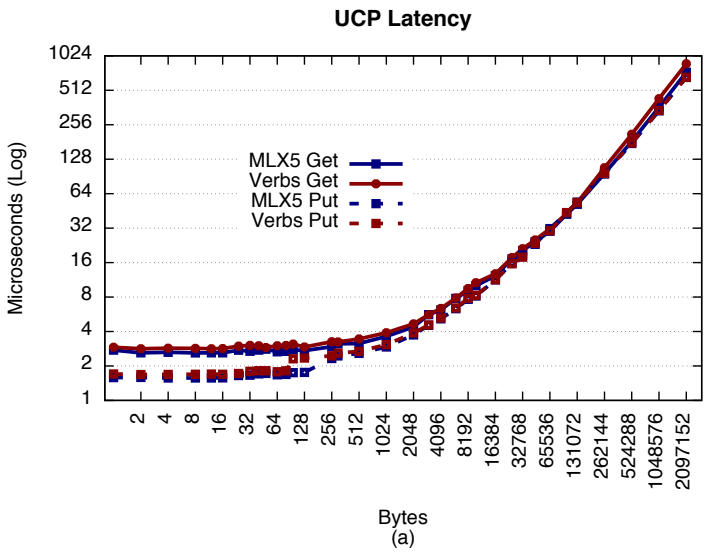
Testbed

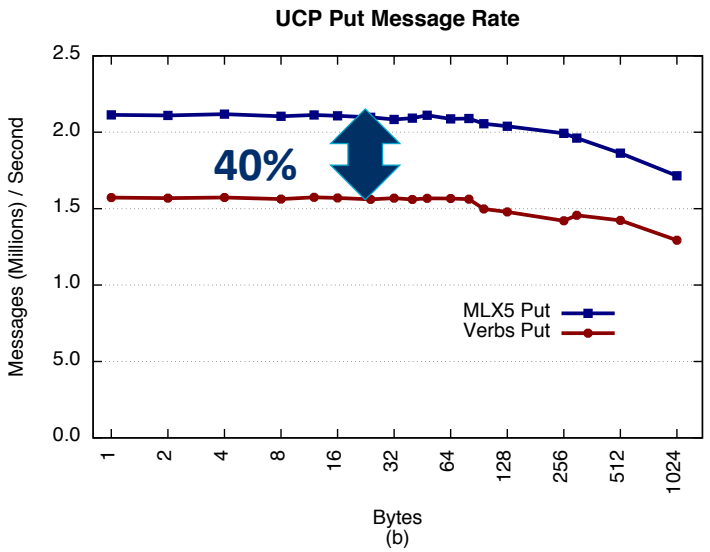
- 2 x Softiron Overdrive 3000 servers with AMD Opteron A1100 / 2GHz
- ConnectX-4 IB/VPI EDR (PCIe gen2 x8)
- Ubuntu 16.04
- MOFED 3.3-1.5.0.0
- UCX [0558b41]
- XPMEM [bdfcc52]
- OSHMEM/OPEN-MPI [fed4849]

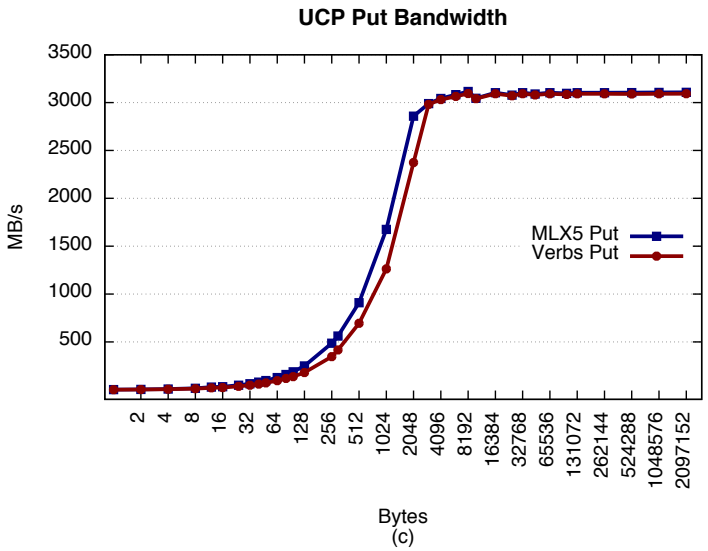
Pavel Shamis, M. Graham Lopez, and Gilad Shainer. "Enabling One-sided Communication Semantics on ARM"

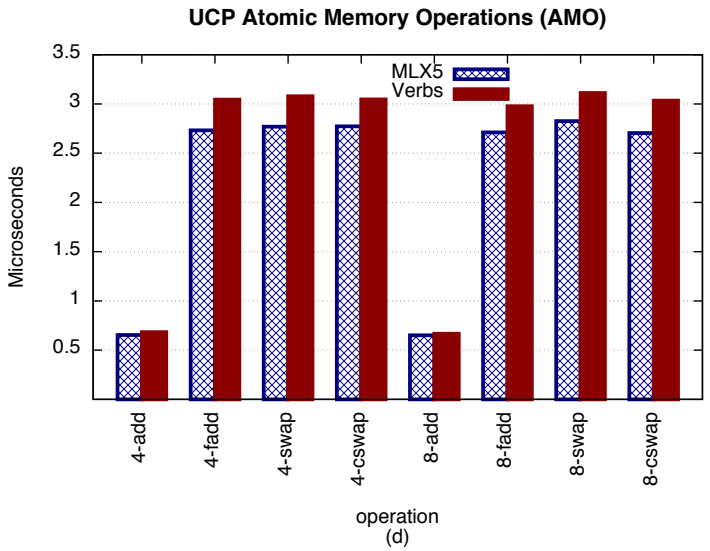
 © 2017 Arm Limited

OpenUCX IB: MLX5 vs Verbs



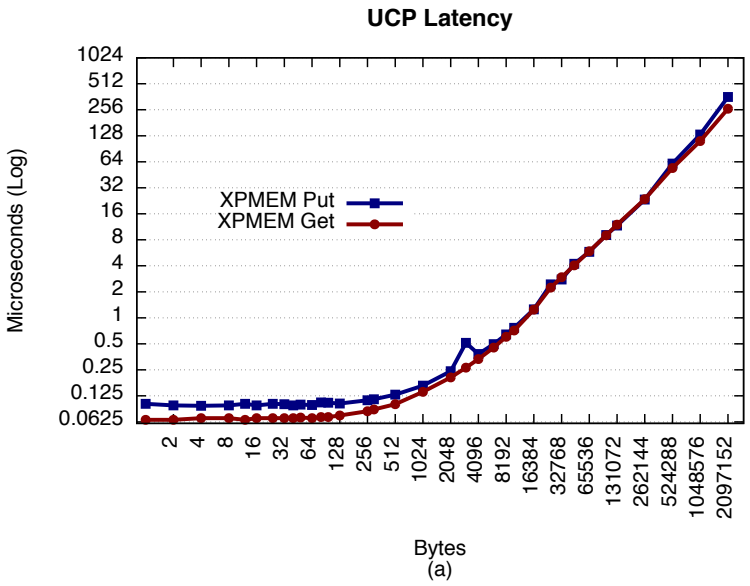


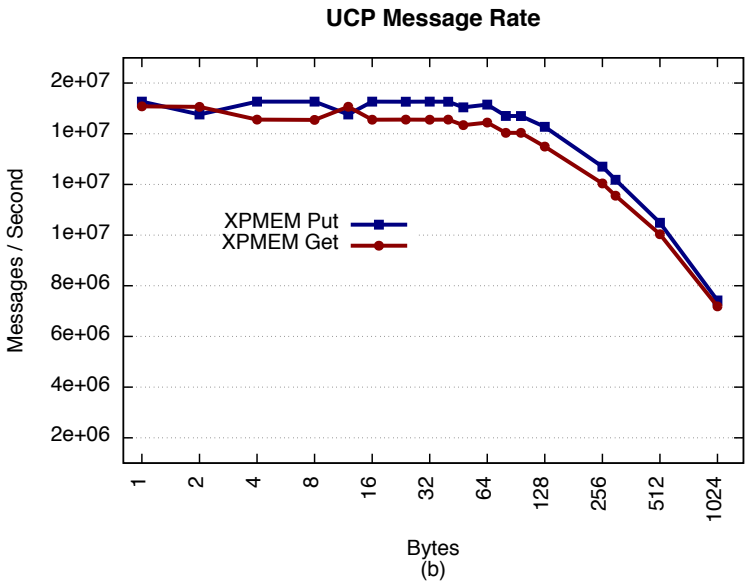


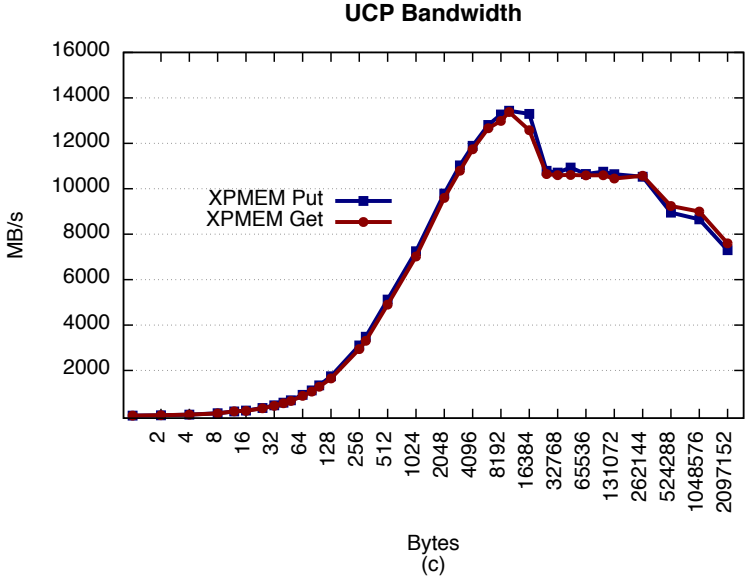


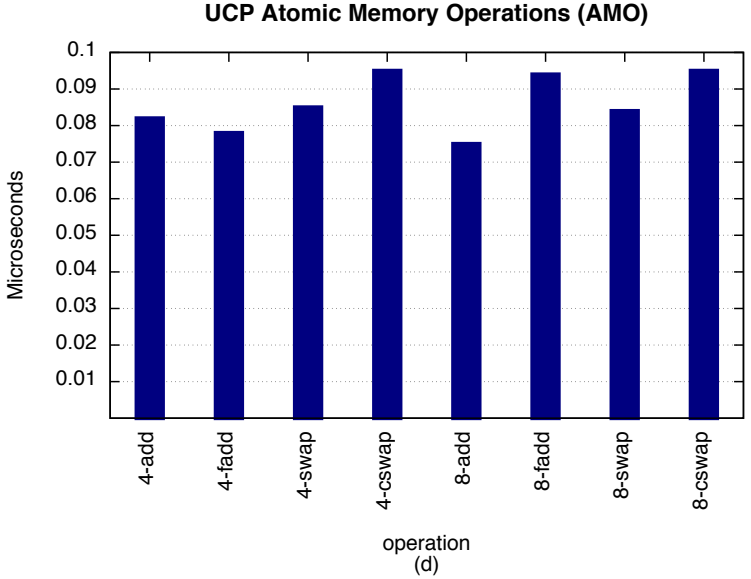
© 2017 Arm Limited

OpenUCX: XPMEM



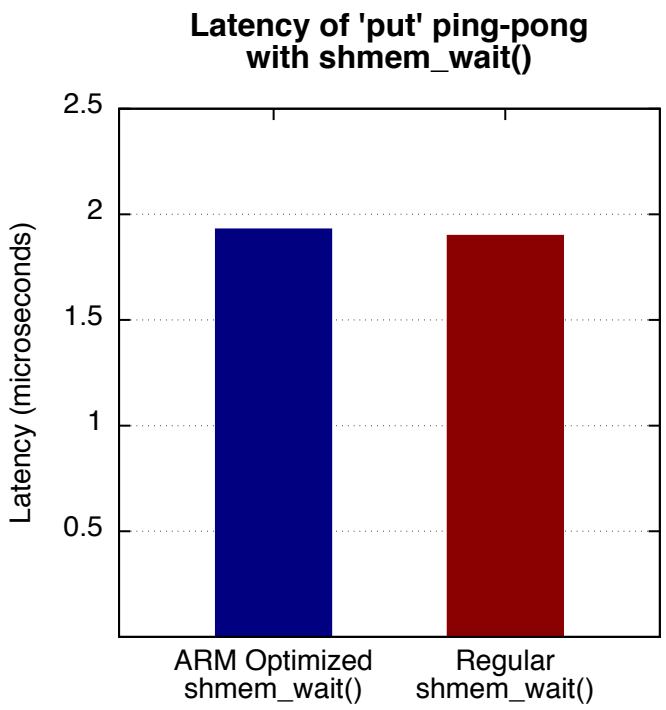




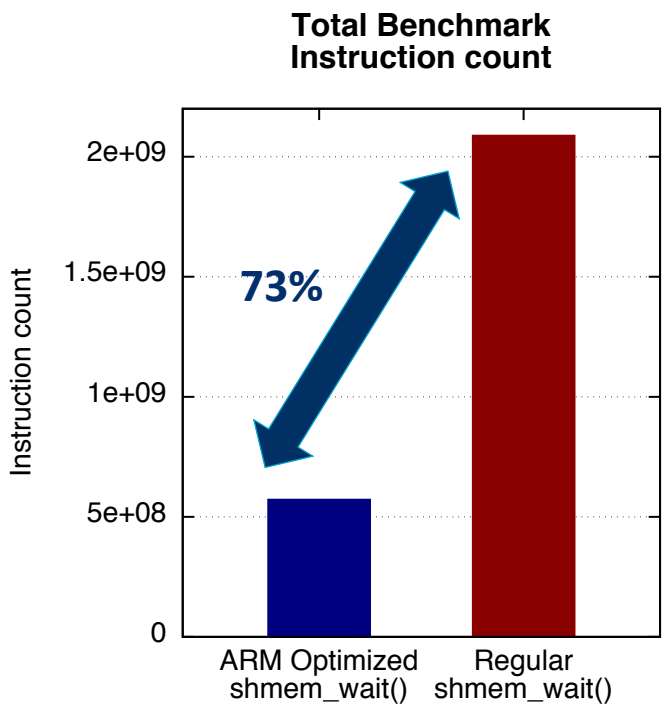


© 2017 Arm Limited

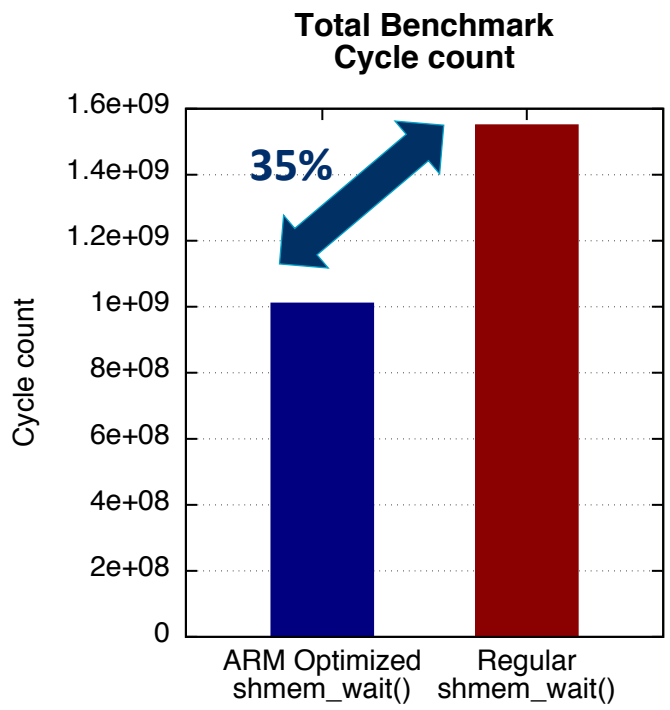
SHMEM_WAIT()



(a)



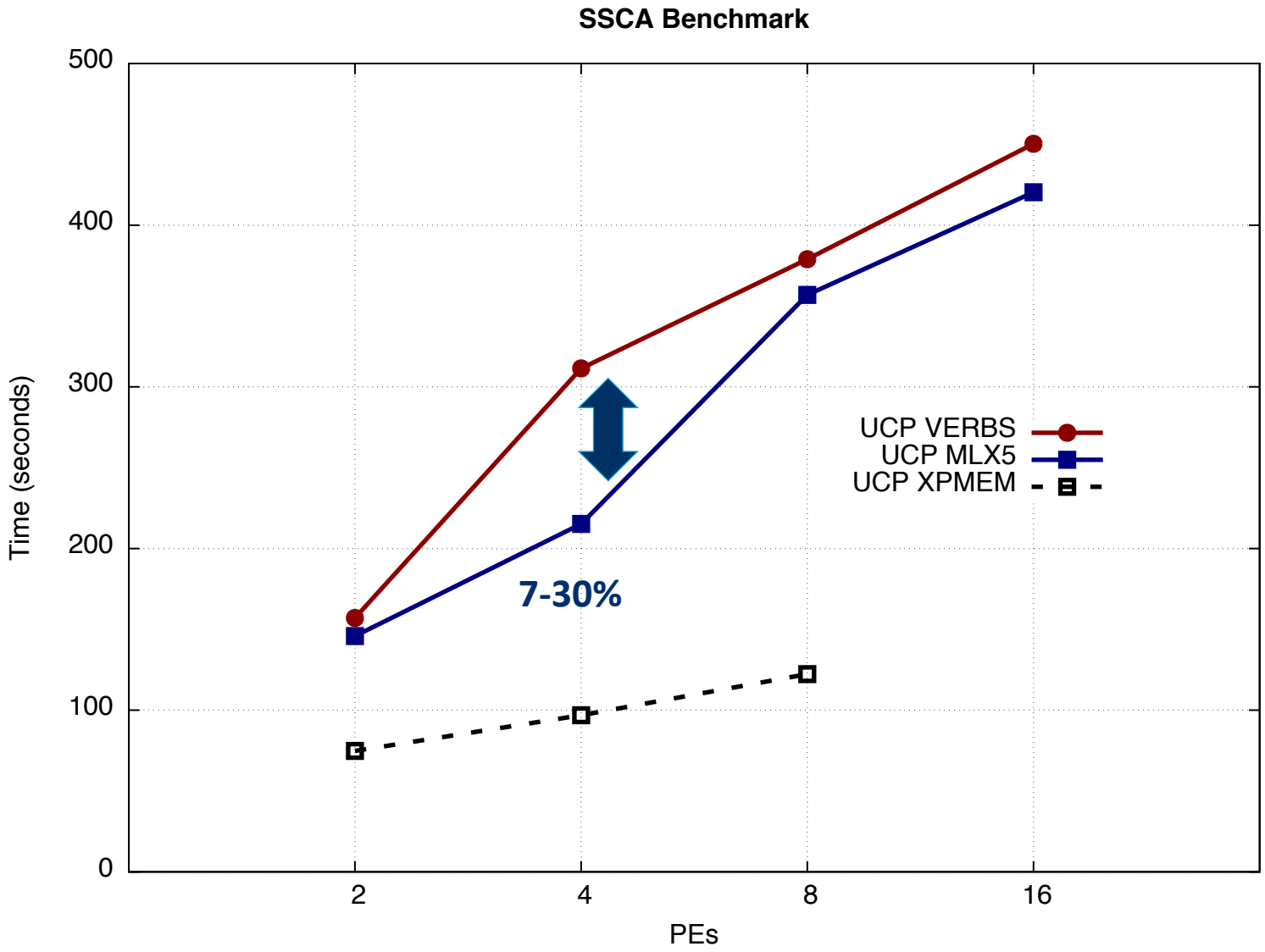
(b)



(c)

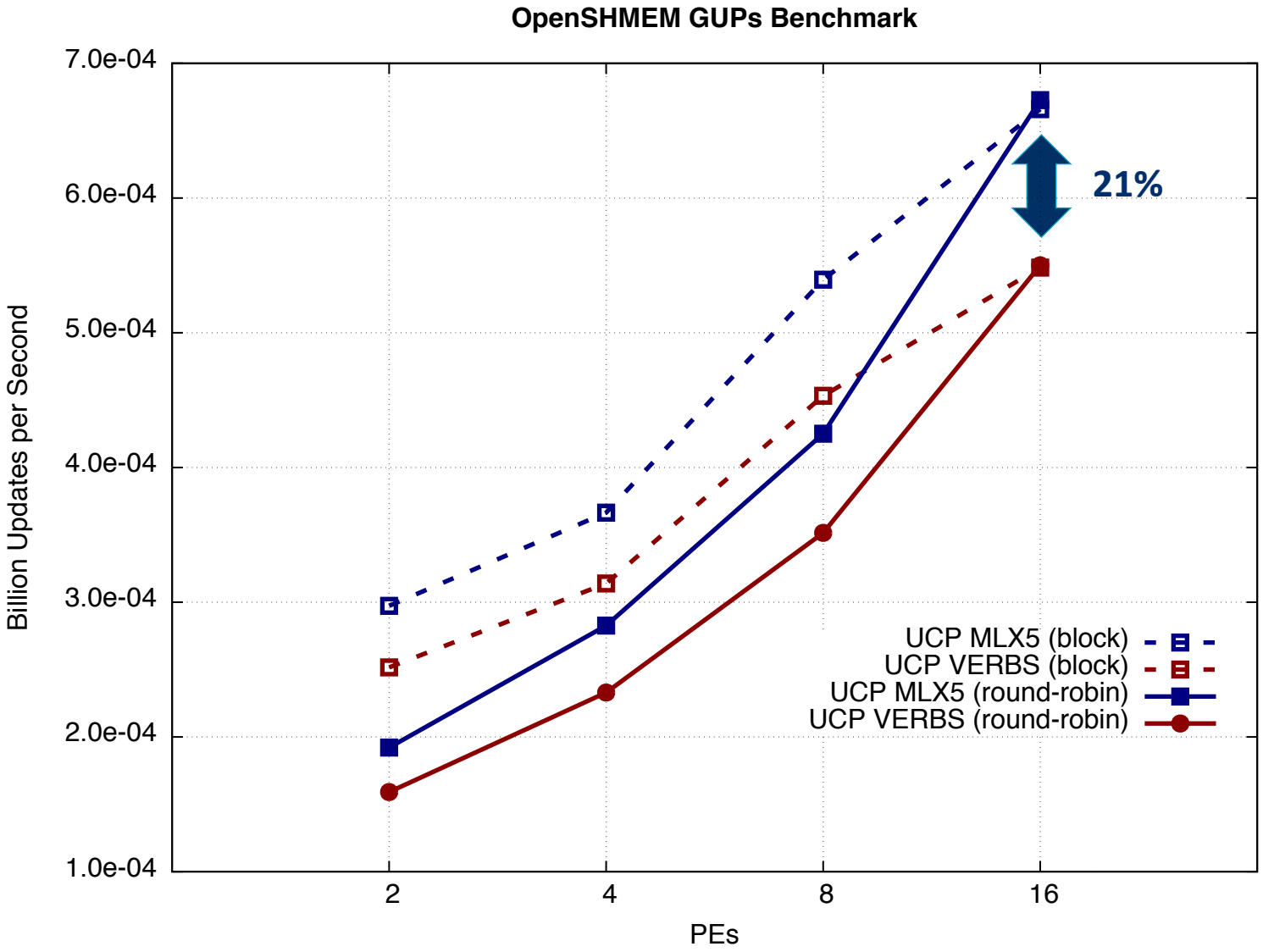
© 2017 Arm Limited

OpenSHMEM SSCA



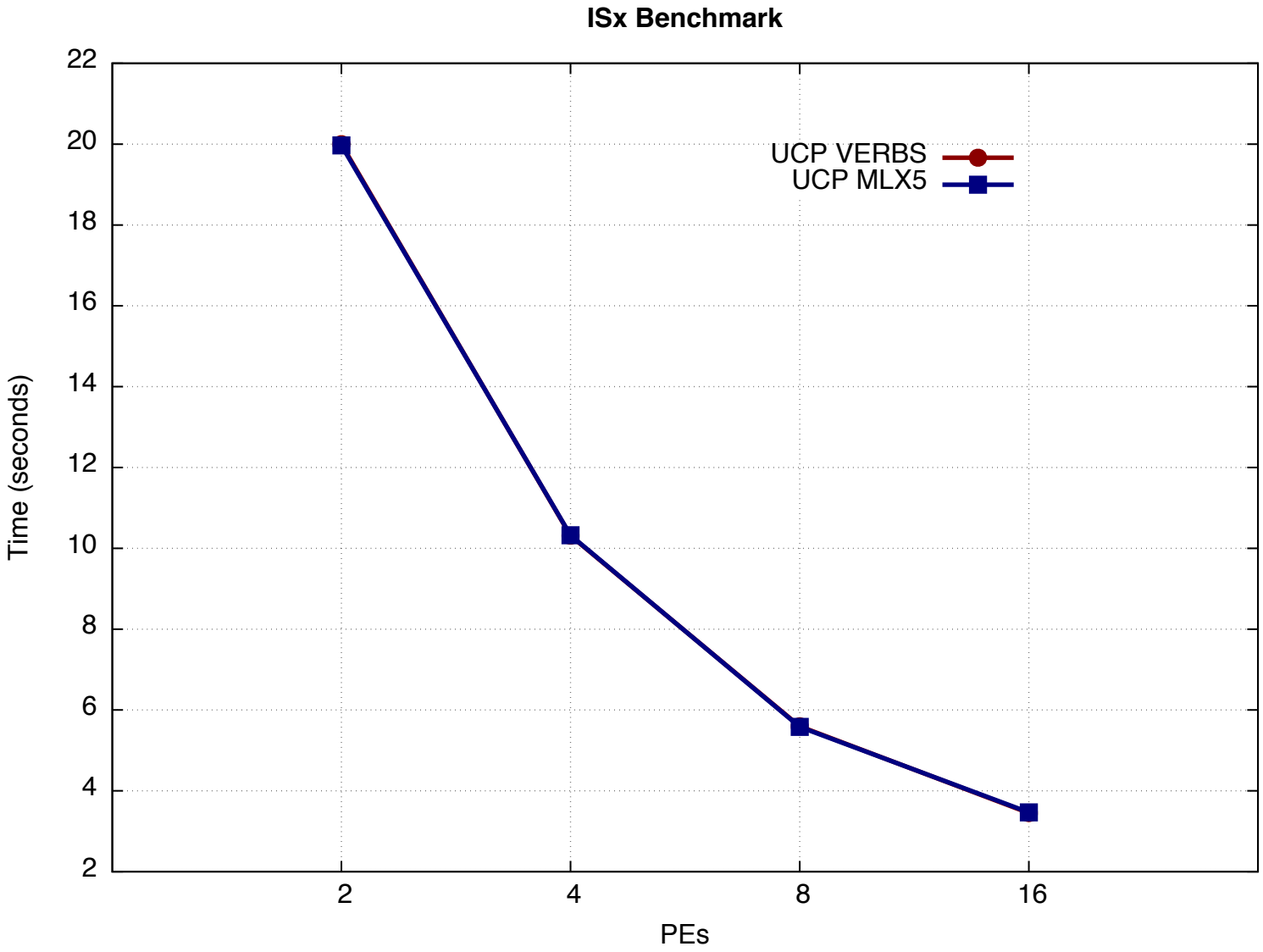
© 2017 Arm Limited

OpenSHMEM GUPs



© 2017 Arm Limited

OpenSHMEM ISx



© 2017 Arm Limited

Summary

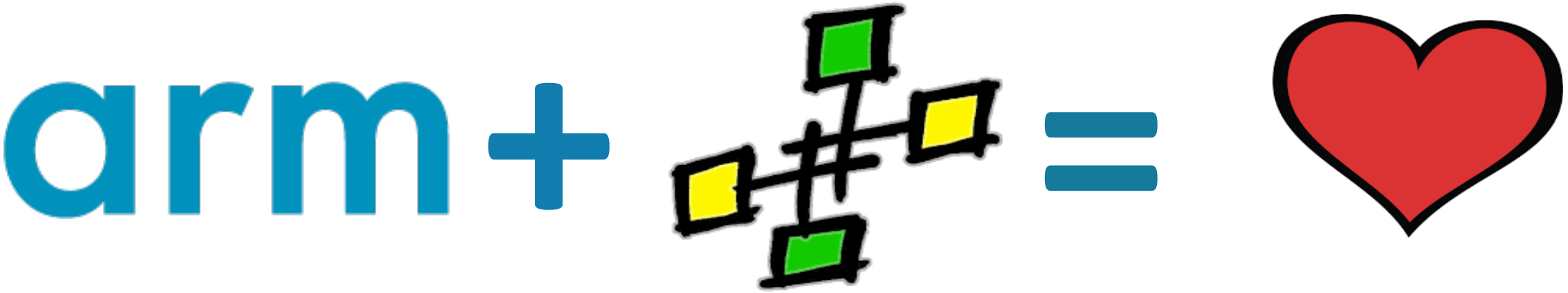
Open UCX over Mellanox InfiniBand reaches 90% of practical bandwidth with a single 2Ghz ARM core

Open UCX with MLX5 transport on Arm demonstrate 40% increase in message rate when compared to Verbs transport

OpenSHMEM shmem_wait demonstrate 35% decrease in number of cycles using ARM WFE instruction

GUPs and SSCA OpenSHMEM codes demonstrate between 7-35% acceleration in execution time using Open UCX MLX5 transport

© 2017 Arm Limited



 © 2017 Arm Limited

The Arm trademarks featured in this presentation are registered trademarks or trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. All rights reserved. All other marks featured may be trademarks of their respective owners.

www.arm.com/company/policies/trademarks

arm

 © 2017 Arm Limited