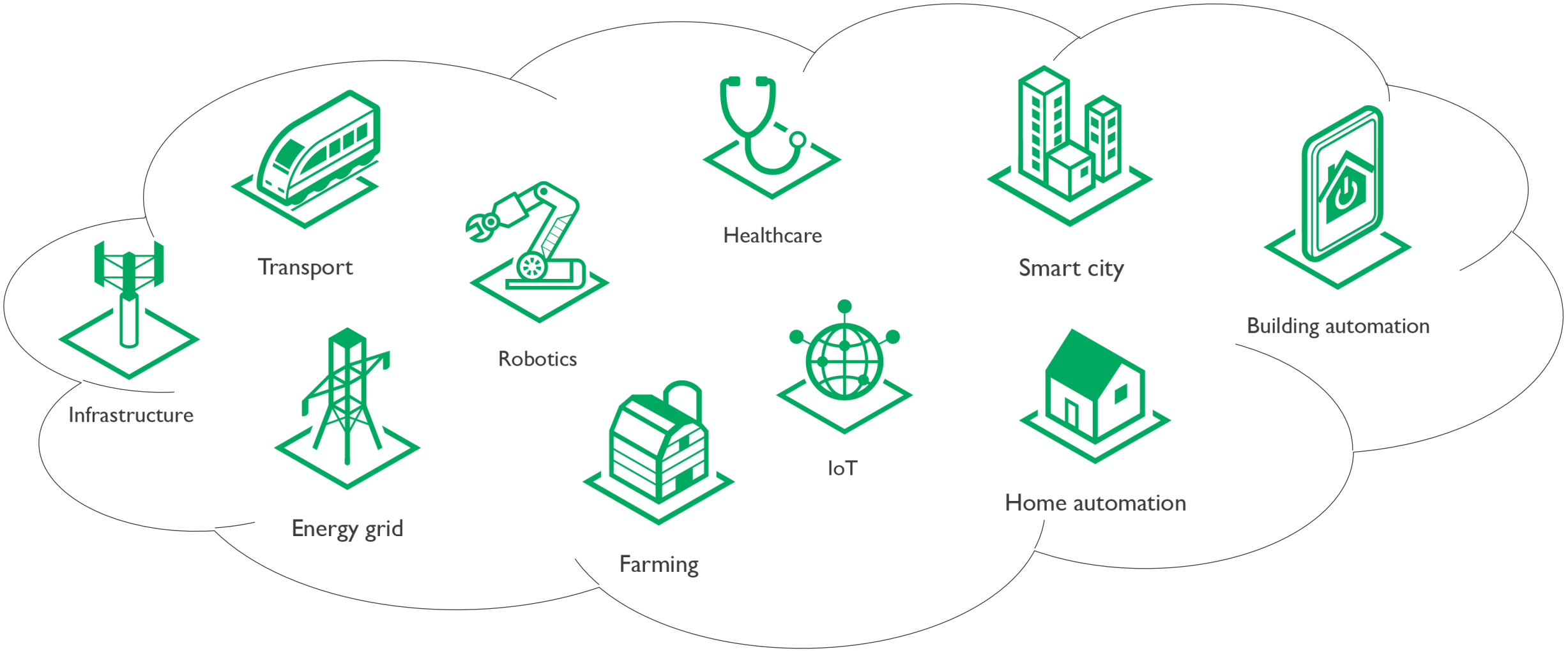
Fast, Scalable and Energy Efficient IO Solutions: Accelerating infrastructure SoC time-to-market

ARM

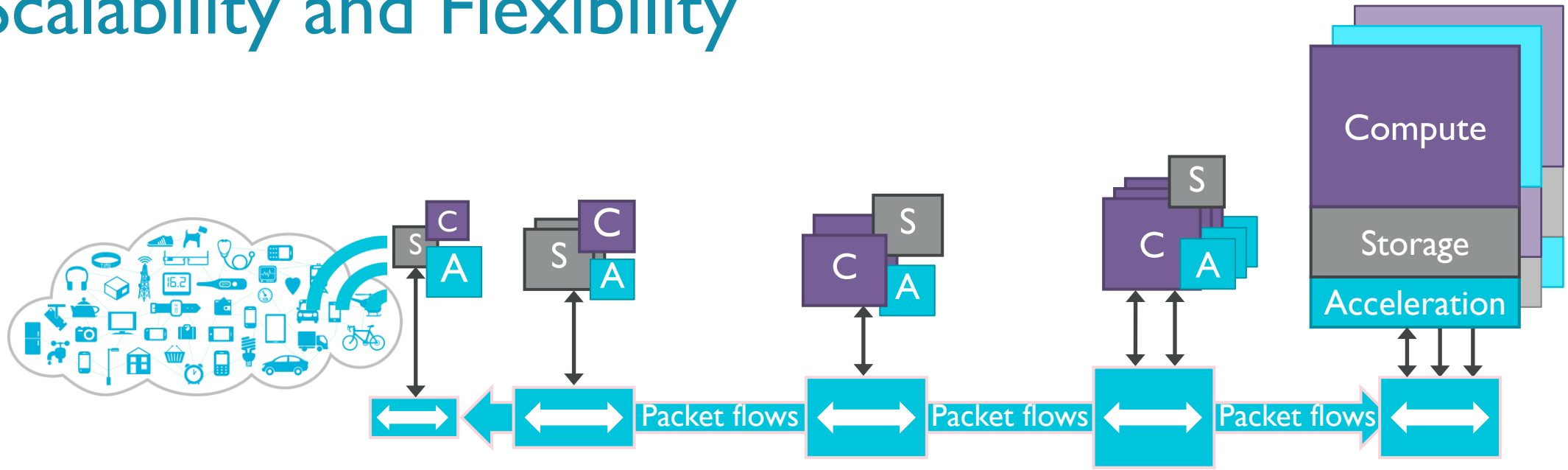
Sridhar Valluru
Product Manager

ARM Tech Symposia 2016

Intelligent Flexible Cloud



Scalability and Flexibility



Access point

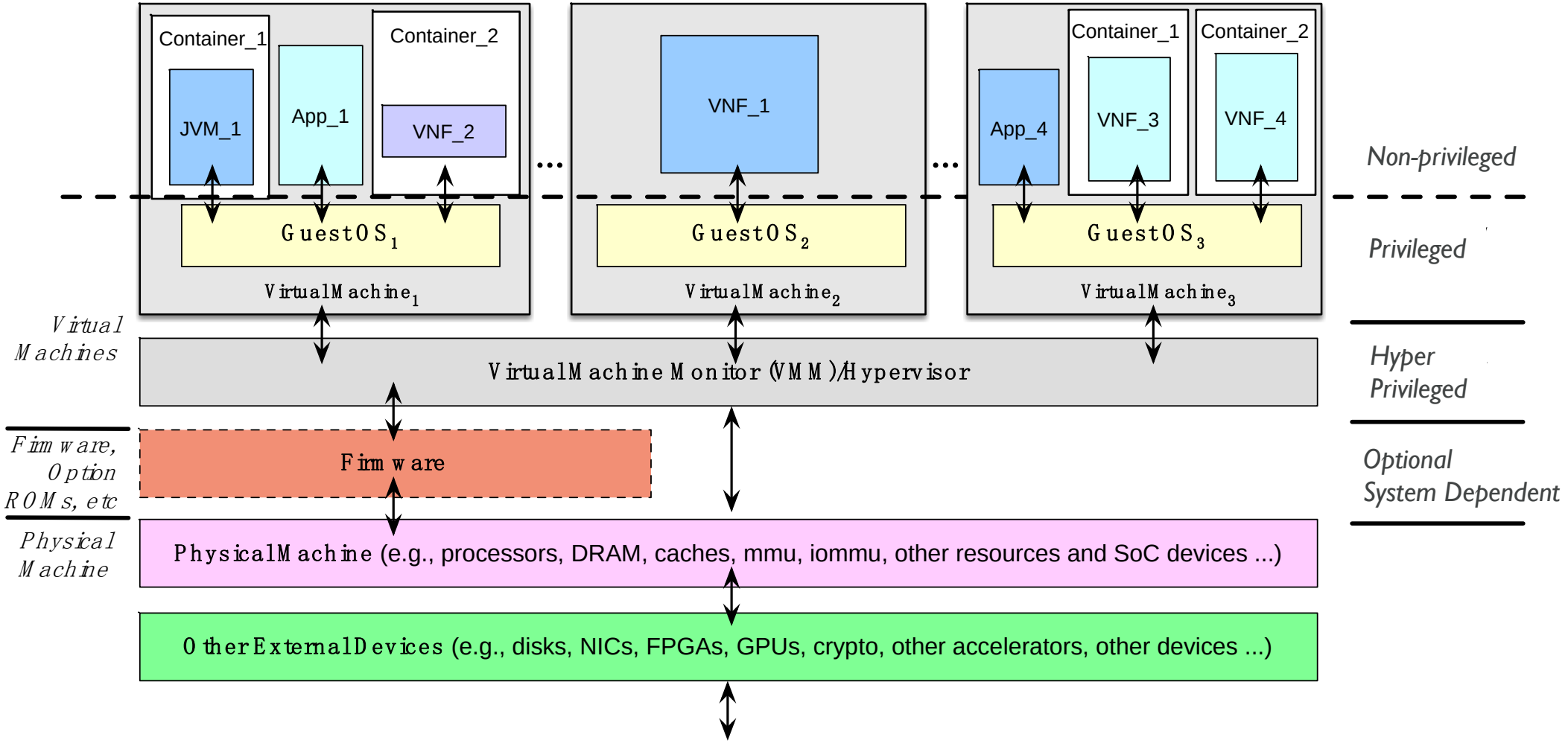
2-5W SoC
4-8 CPUs
~30mm²

Target design space

100W SoC
64-96
CPUs
~300mm²

Data center

Execution environment supporting IFC



IO challenges for next-gen SoC systems

Scalability

- Limited number of hardware IO due to capacity
- Large number of IO stream traffic management

Performance

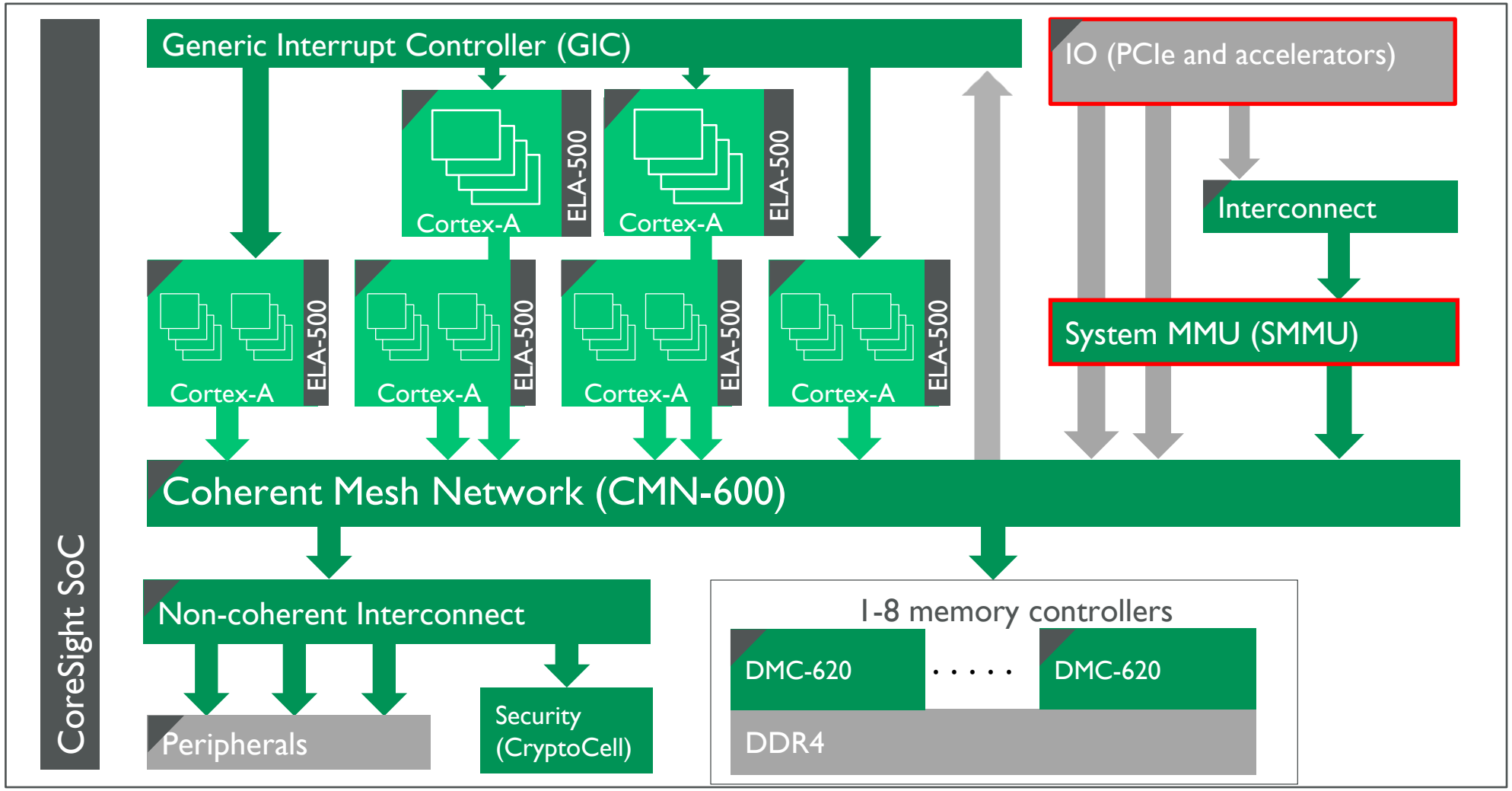
- Large number of translations → Large number of Page Table Walks
- Insufficient TLB in SMMU

Power Efficiency

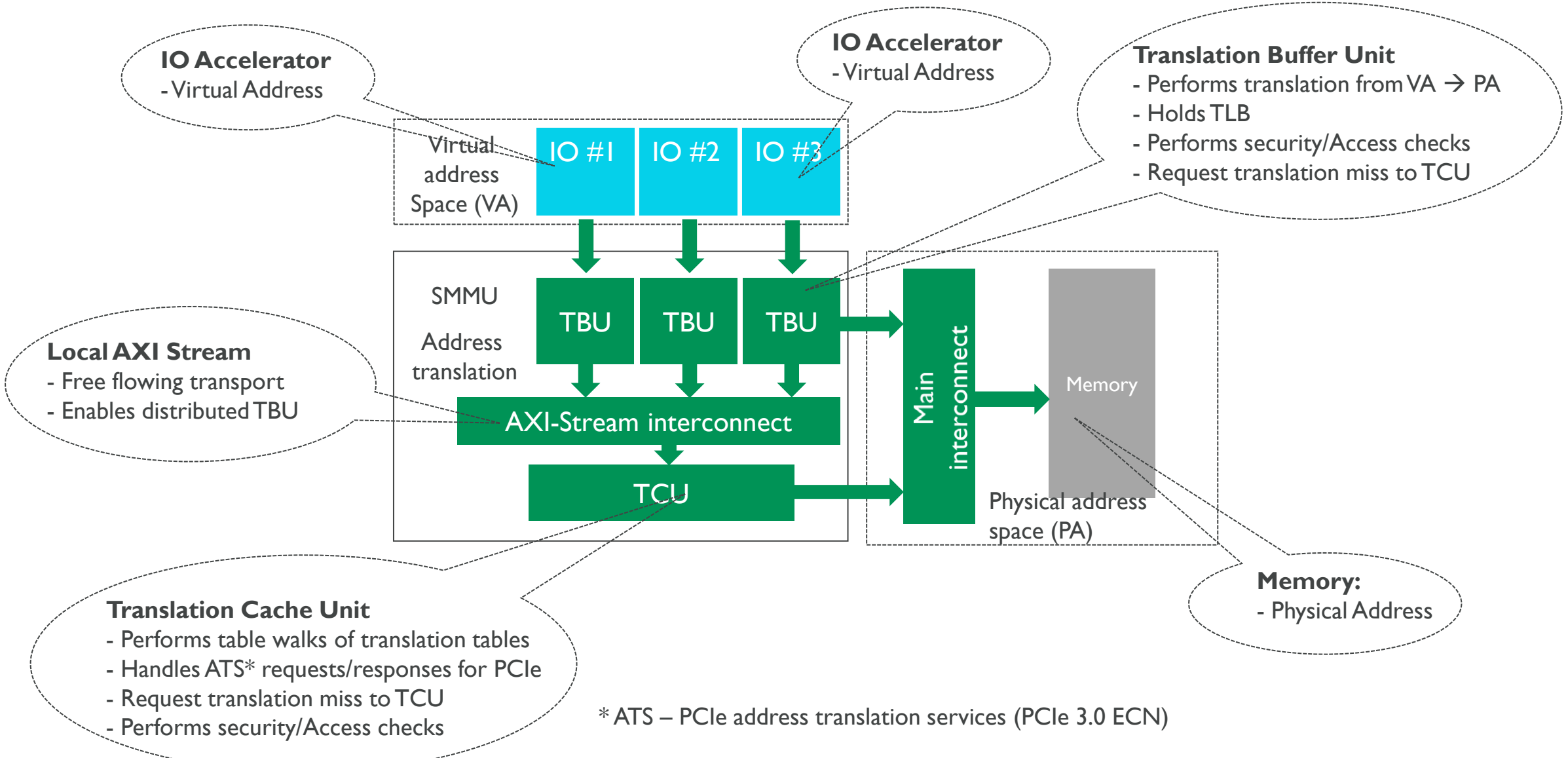
- Large Page Table Walks → Large number memory access
- Enormous TLB → Large dynamic power

System memory management unit (SMMU)

Next-generation example server subsystem



ARM IO MMU or system MMU (SMMU)



* ATS – PCIe address translation services (PCIe 3.0 ECN)

SMMU architecture evolution

SMMUv1

- Features
  - Support for v7 page table for IO virtualization
  - 4k page granule
- Implemented by
  - CoreLink™ MMU-401

SMMUv2

- Adds
  - Up to 128 translation contexts
  - Support for v8 page tables
  - 64k page granule
- Implemented by
  - CoreLink™ MMU-500

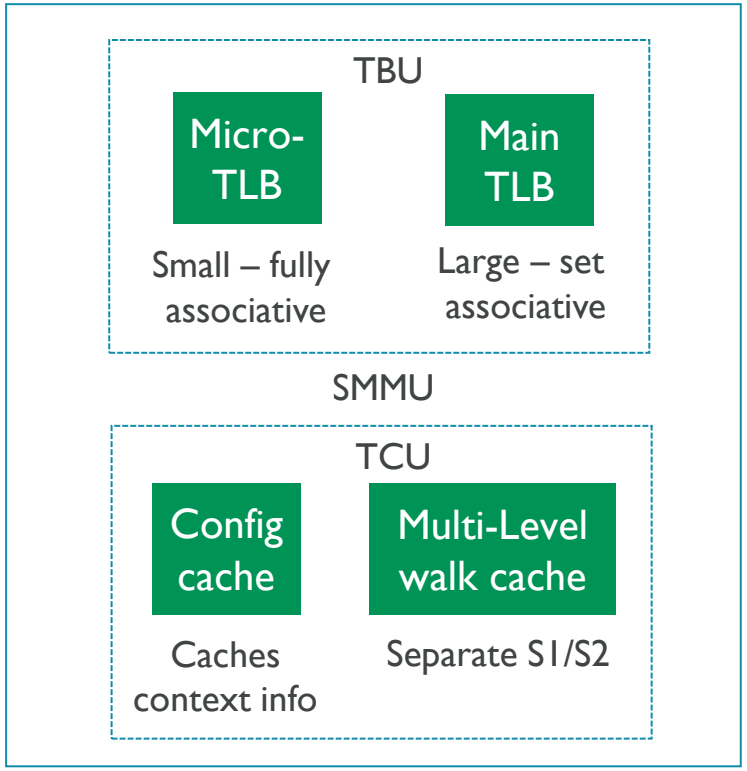
SMMUv3

- Adds
  - Scalability enhancements for millions of translation contexts
  - Context store in memory
  - PCIe address translation services (ATS) for returning translations to end points with address translation caches (ATC)
  - PCIe process address space ID (PASID) for process-specific translations
  - PCIe page requires interface (PRI) support for access to unpinned pages in memory
  - Software communication via memory queues (non-blocking / scalable)
  - Support for message-signalled interrupts

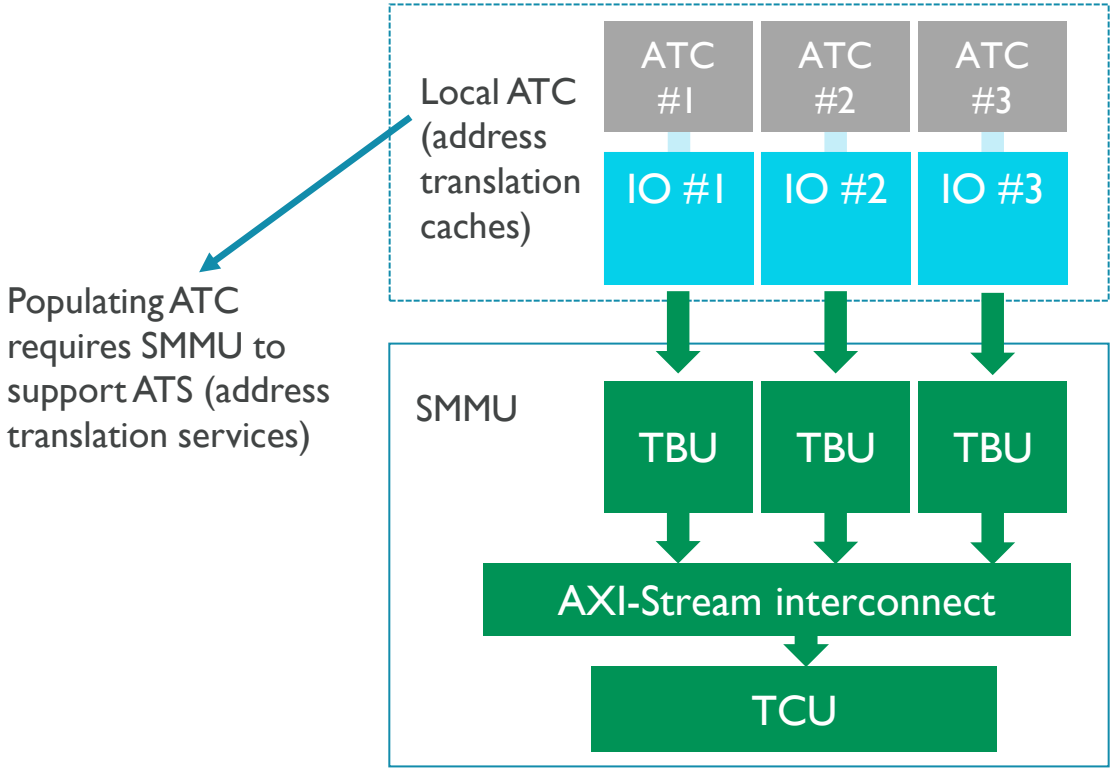
Addressing performance & scalability challenge

SMMU microarchitecture

- VA → PA translation overhead
- Limited TLB scalability with # IO devices



- VA→PA Translation in ATC Cache
- ATC removes dependency on TBU Size



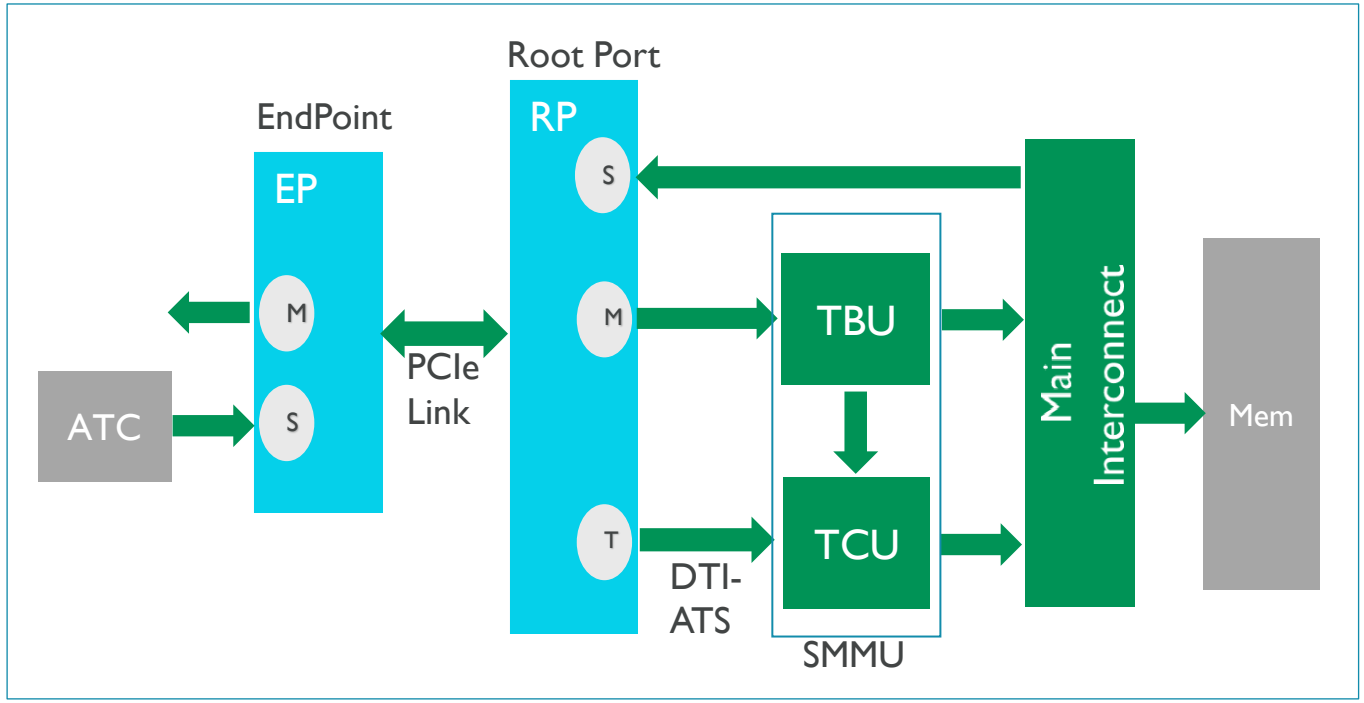
ARM

Advantages of PCIe ATS for IO access performance

- Scalability of ATC
  - Size and number of ATCs grows with number of IO devices, whereas TLB size in SMMU is fixed (however large)
- Independence of ATCs
  - Local ATC accesses are independent of each other and do not result in cache trashing
  - Shared TLB size in a SMMU can suffer from trashing if multiple IO devices access too many scattered locations in memory
- Customizable pre-fetch
  - IO devices can request translations ahead of time according to known access patterns
  - Shared TLB in an SMMU is not aware of IO access patterns and cannot implement a universal pre-fetch policy
- Customizable replacement policies
  - IO devices can prioritize caching of some entries over others based upon known access patterns
    - E.g., an Ethernet NIC might choose to exclusively cache ring descriptor translations and store only data buffer translations temporarily
- Support for unpinned memory without stalling faults with the use of PRI

ARM SMMU and Cadence PCIe RP integration

- M: AXI master interface
  - All normal PCIe packets with or without translated address are seen here
- S: AXI slave interface
- T: DTI-ATS (direct translation interface for PCIe ATS) supports
  - ATS translation requests from EP
  - Invalidation requests from TCU
  - PRI (page request interface) requests from EP



* ATS – PCIe address translation services (PCIe 3.0 ECN)

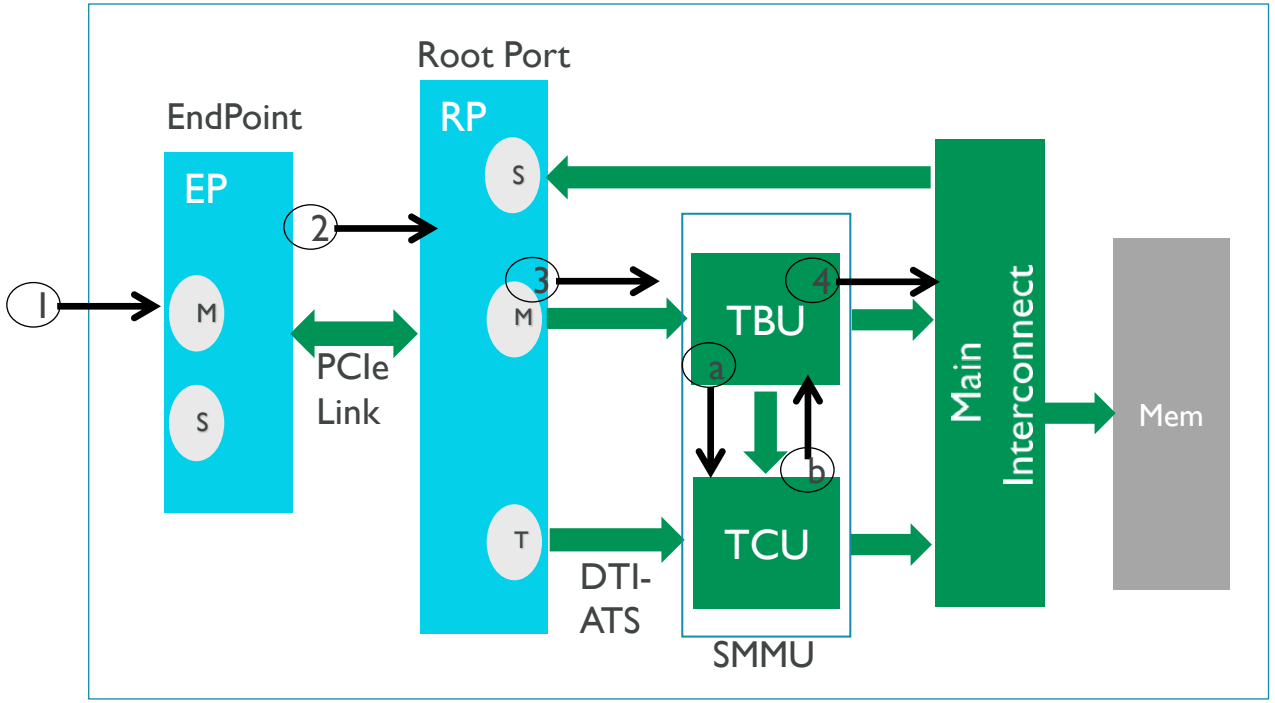
ARM

Cadence PCIe RC's DTI-ATS features

- Separate interface provided with the PCIe RC IP
  - All PCIe ATS related requests, responses, invalidations are routed to this I/F
- DTI-ATS implementation supports additional features
  - PCIe PRI (page request interface)
  - PCIe PASID support (process address space ID)
- DTI-ATS is conveyed using AXI4-Stream
  - Separate master AXI4-Stream and slave AXI4-Stream interfaces
  - Transaction sideband signals to indicate the context information to the TBU
  - DTI-ATS packets can be presented/accepted in one clock cycle
- Debug & status
  - Registers to capture status and error conditions encountered in the DTI-ATS protocol

SMMU with PCIe operation – no ATS

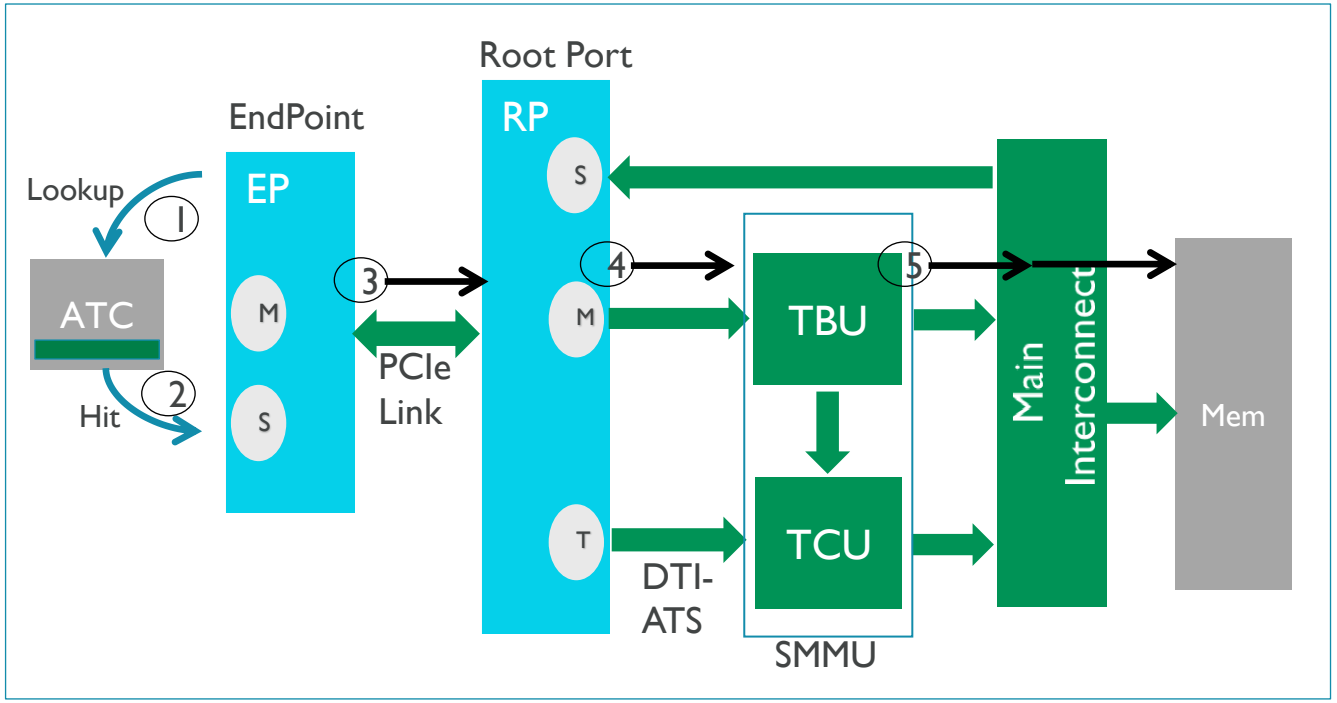
1. Client logic generates a TLP with a un-translated address
2. EP sends this as a PCIe TLP to the RP
3. On receipt by the RP, since the packet is a data flow packet, this is sent on the "M" interface
   a) If the TBU does not have a suitable translation for the address received, it will issue a request to TCU
   b) The TCU will respond with the response for the TBU
4. The TBU then forwards the transaction to the memory



14

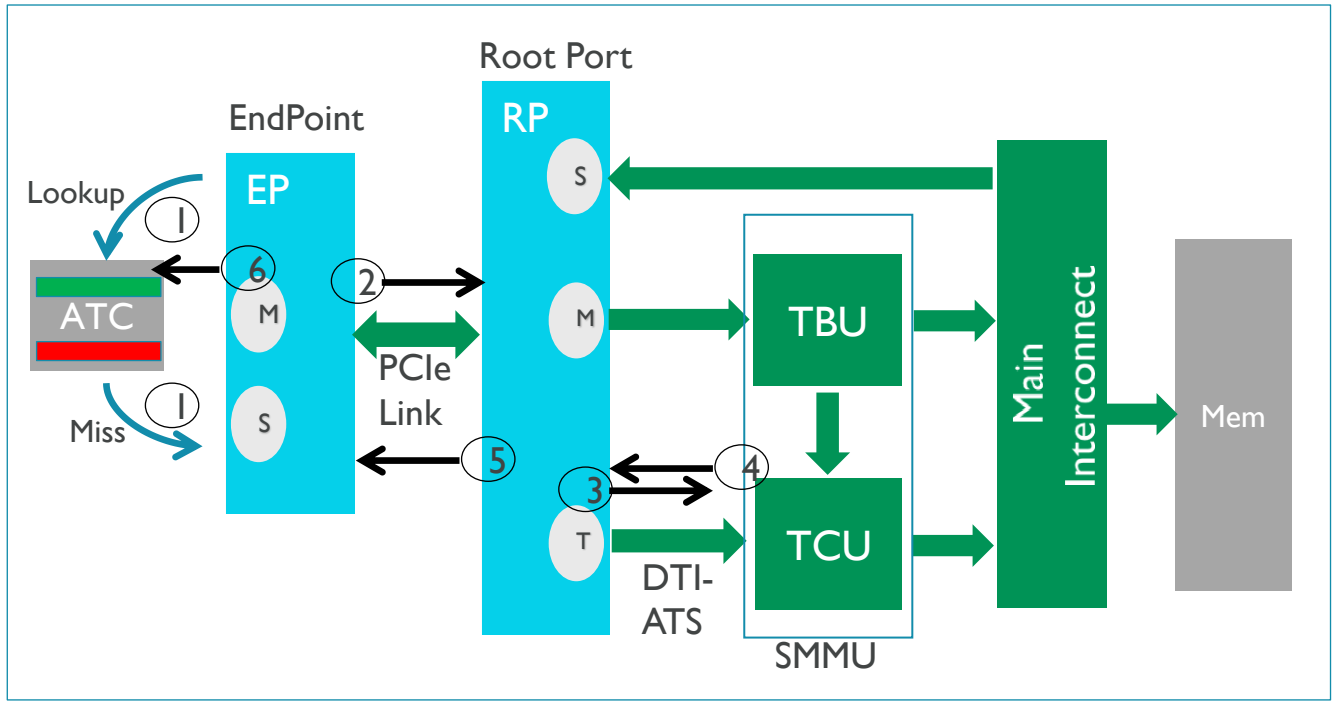
SMMU with PCIe operation – with ATS and ATC hit

1. Client logic generates a TLP with a virtual address
2. The client logic uses the translated Addr if available from the ATC
3. The EP sends this as a PCIe TLP that has translated address
4. On receipt by the RP, since the packet is a data-flow packet, this is sent on the "M" interface
5. The TBU then forwards the transaction to the memory via the main interconnect



SMMU with PCIe operation – with ATS and ATC miss

1. EP client generates a PCIe translation for a particular address that needs translation
2. Translation request goes out on the PCIe link to the RP
3. RP sends the translation request it received on the "T" interface to the TCU
4. The TCU then generates the response completion
5. The RP repacks the translation completion TLP back to the EP
6. Once the EP received this completion for the translation request it generated, it populates the local ATC



IO challenges for next-gen SoC systems

Scalability

- ATC allows PCIe RC to support multiple IO accelerators
- AXI Stream Interface allows distributed TBU's to be connected to a TCU

Performance

- With ATC, no more address translation needed for every transaction
- TCU Cache reduces page table walks

Power Efficiency

- ATC & TCU minimize memory access for page table walks
- Custom ATC in IO accelerator removes the need for very large TLB in SMMU

Summary

- IFC is driving need for scalability, performance and efficiency for IO accesses in infrastructure SoCs
- ARM has been addressing IO virtualization solutions via its SMMU Fast, performant IO such as PCIe Gen4 from Cadence has been efficiently integrated with ARM's SMMU with an architected interface DTI-ATS
- Combined SMMU-PCIe solution delivers high performance access for IO devices with PCIe ATS as well as PRI and PASID support
- SMMU IP from ARM is designed to handle the performance, scalability, and power efficiency demands from SoCs for IFC

Questions?
Want to know more?
Please contact sridhar.valluru@arm.com