# **POSH MIXED SIMULATION**

© Copyright 2019 Xilinx Distribution Statement A, Approved for Public Release: Distribution Unlimited

.....

- There are a state of the stat



# EDGAR IGLESIAS

### FRANCISCO IGLESIAS SAKIS PANOU

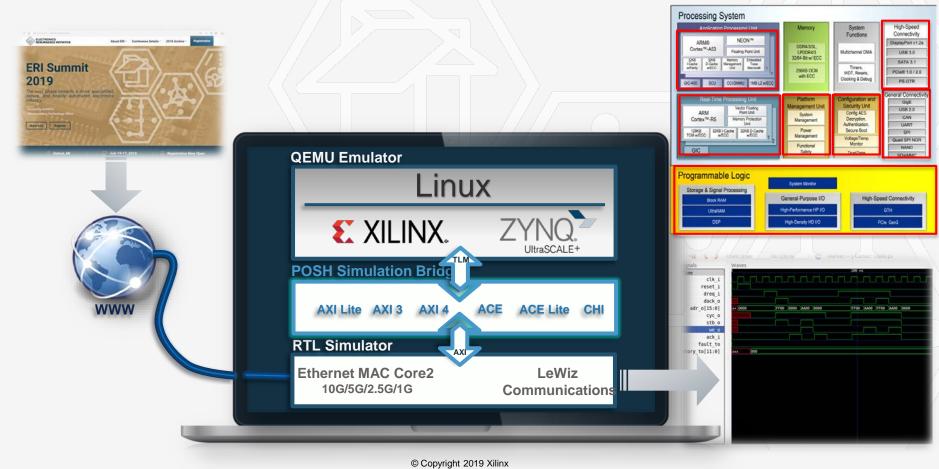
© Copyright 2019 Xilinx Distribution Statement A, Approved for Public Release: Distribution Unlimited



# THE DEMO

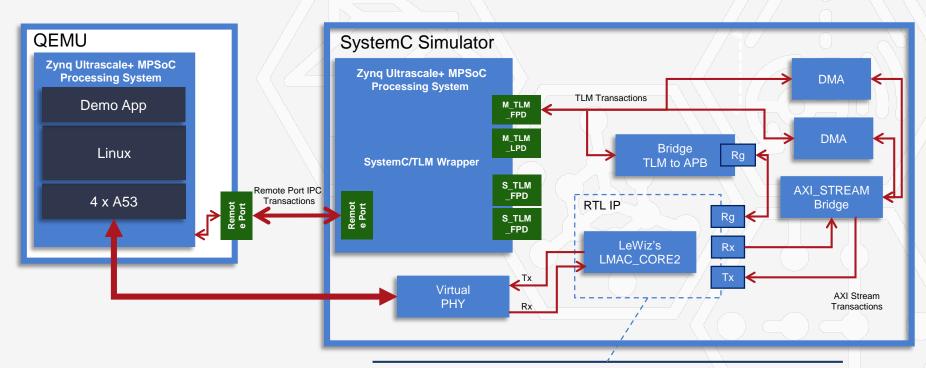
© Copyright 2019 Xilinx Distribution Statement A, Approved for Public Release: Distribution Unlimited

#### **POSH OPEN SOURCE HARDWARE – MIXED SIMULATION LIVE DEMO**



Distribution Statement A, Approved for Public Release: Distribution Unlimited

#### **POSH OPEN SOURCE HARDWARE – MIXED SIMULATION LIVE DEMO**



LeWiz Communications Ethernet MAC Core2 10G/5G/2.5G/1G

https://github.com/lewiz-support/LMAC\_CORE2

### **POSH - THE ALTERNATIVE !**



© Copyright 2019 Xilinx Distribution Statement A, Approved for Public Release: Distribution Unlimited

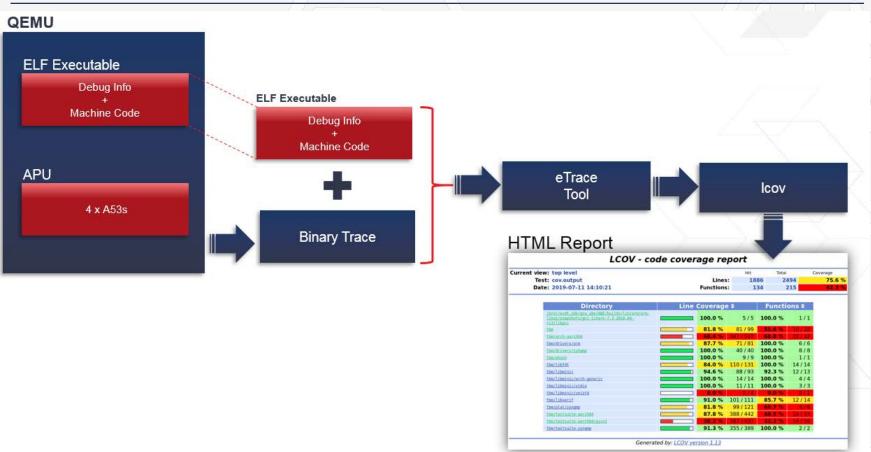
### **GETTING STARTED**

Project Location: <u>https://github.com/Xilinx/libsystemctlm-soc</u>

🖓 Why Git	lub? - Enterprise Explore - Marketplac	e Pricing - Search		I Sign in Sign up
C Xilinx / libsys	temotim-soc wes 2 III fuil requests (0 III Projects (0	E Security in Insights	O Watch 11	★ Star 12 ¥ Fork 0
	GitHub is home to over 36 n	in GitHub today nilion developers working togeth projects, and build software tog Sign up		Darras
No description, w	ebsite, ar topics provided. mits (2 branches	© 6 reterres	11 2 contributors	d Vew lot of
Tearch: master •	Sam put require		1	Find File Clame or clawedged w
	Time put required	el headen into private subfelders 🚟	1	find file Cane or directood +
			1	March Street and Street and Street and Street
[10] Frencisco Igle	sias and Edgar E. Iglesias tim modules. Nove library intern	aders into private aubifolders	8	Latest commit 2047976 7 days ago
())) francisco lgi	aise and Edger E. Iginalias the modules. Nove library inter- tilm bridges: Maye library external the doc: Add documentation for the CPE	aders into private subfoliciers companients		Latest commit 2047976 7 days ago 6 days ago
Frenchers light im checkers im docs	aise and Edger E. Iginalias the modules. Nove library inter- tilm bridges: Maye library external the doc: Add documentation for the CPE	aders into private aubfoliciers components sr posted/inon-posted wire updat		untest convertit zaktryne 7 days ago 6 days ago 26 days ago
Francisco (p) In checkors In docs In literanoto-por	aise and Edger L. Igleise the modules More Remy inter- the bridges: Move Remy internal he doc: Add documentation for the CHE remote-port Add optional exposed for	aders into private subfoldors components or posted/non-posted wire upda aders into private subfoldors		Lafest convert 2047916 7 days ago 6 days ago 25 days ago 5 northe ago
Image: Control of the second secon	ain and Edge E. Igliable the modules Mass Harry intern the bridges: Move Harry external he doc-Add accumentation for the OT remote gott Add optional separat the bridges: Move Harry internal he the bridges: Move Harry internal he	aders into private subfolders components or posted/non-posted wire upda aders into private subfolders aders into private subfolders	tec	Latest convertit auktorne 7 days algo 6 days algo 26 days algo 5 neorethe algo 6 days algo
Freedoors (p) In checkers In doors In Etraenoto-por In Tests In tim-bridget	ain and Edge E. Igliable the modules Mass Harry intern the bridges: Move Harry external he doc-Add accumentation for the OT remote gott Add optional separat the bridges: Move Harry internal he the bridges: Move Harry internal he	aders into private subfolders companents or posted/non-poted vire upda aders into private subfolders aders into private subfolders rtaining CHE transaction attribute	tec	Lafest convirt 2x477x 7 days ago 6 days ago 25 days ago 5 northe ago 6 days ago 6 days ago
Frenches løf in checkers in docs in Sterensko-por in tests in ten-bridger in ten-bridger in ten-bridger	ais of Eiger L (globals the mobiles Mars Herry inter- tion bridges Mars (Brary internal he doc Add documentation for the O I remote york Add optional report the obliges. Nove Brary internal he the obliges. Mars it Brary internal he the international Add an extension oo the international Add an extension oo the international Add an extension oo	aders into private subfolions components er posted/non-posted wire upda aders into private subfolioes aders into private subfolioes rtaining OTE transaction attribute esders into private subfolioes	tec	Lutest sommit bettrin 7 days ega 6 days ega 28 days ega 5 norstik ega 5 days ega 6 days ega 26 days ega
Final Active State	ais of Eiger L (globals the mobiles Mars Herry inter- tion bridges Mars (Brary internal he doc Add documentation for the O I remote york Add optional report the obliges. Nove Brary internal he the obliges. Mars it Brary internal he the international Add an extension oo the international Add an extension oo the international Add an extension oo	aders into private subfoldors components er posted/non-posted vire upda aders into private subfolders aders into private subfolders rtaining OLE transaction attribute eaders into private subfolders CEI transactione	tec	uned commit 247314 21 days age 6 days age 25 days age 5 northe age 6 days age 24 days age 5 days age 5 days age
Provides lyb  declars	the second Edges E Lightline for mobility. More Borry internal the the biologies Mayor Borry external the doc-Add documentation for the OT remarks port Add options' appoint the biologies. More Borry internal the the biologies. More Borry internal the the motion of the add options appoint the motion of the add options appoint the motion of the add options appoint the add options appoint and an extension of the motion. More Borry internal the the motion of the add option of the add option for A	ades into private autofaloes companents er posted, inon-posted virie upda ades into private autofaloes aders into private autofaloes cateri into private autofaloes CE transactions CE transactions	tec	und connit bettyn ty 6 dayn ego 25 dayn ego 5 northe sgo 6 dayn ego 26 dayn ego 8 dayn ego 8 dayn ego 8 dayn ego
Freedoor la ferral de la constante la juit la constante de la	Lightlink the resolution Marks Borry internal the dim bridges: Marks Borry reternal the disc. Add documentation for the CPI ministe gort. Add doptimic account dim bridges: Marks Borry internal the tim-bridges: Marks Borry internal the tim-bridges: Marks Borry internal the tim-modules: Marks Borry internal the manual generators: Add an extension on tim-modules: Marks Borry internal the manual generators: Add an extension on tim-modules: Marks Borry internal the manual generators: Add an extension on methode gort Extended as a sensetables retrole-gort Extended as a sensetables	ades into private autofaloes companents er posted, inon-posted virie upda ades into private autofaloes aders into private autofaloes cateri into private autofaloes CE transactions CE transactions	tec	Lated connet 24470427 days ago 6 days ago 25 days ago 5 nearth ago 5 days ago 5 days ago 3 days ago 3 days ago 8 neartha ago
Provide light  If an electron  If a declars	Lightlast the resultion Mars library inter- the biologics Marce library retensils he doc Add documentation for the OH remeate got 7. Ald optimum separat the biologics Marce library internal he the biologics Marce library internal he the restriction Add an extension co the restriction Add an extension co the restriction Add an extension co the restriction and an extension co restriction and a support for prepage Connect a set of user-defin	ades into private autofaloes companents er posted, inon-posted virie upda ades into private autofaloes aders into private autofaloes cateri into private autofaloes CE transactions CE transactions	tec	utet ommit zentre 2 dags opp 6 dags opp 26 dags opp 6 dags opp 6 dags opp 6 dags opp 26 dags opp

- Getting Started: <u>https://github.com/Xilinx/libsystemctlm-soc/blob/master/README</u>
  - Step 1: Read Dependencies and Configuration Sections in README
  - Step 2: git clone https://github.com/Xilinx/libsystemctlm-soc.git
  - Step 3: cd libsystemctlm-soc/tests/ && make examples-run

### **CODE COVERAGE FRAMEWORK**



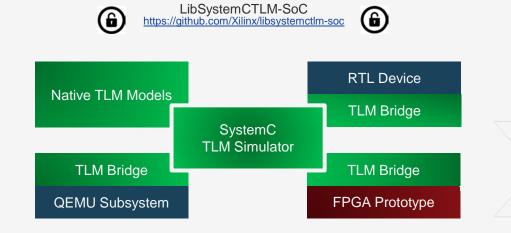


# **TECHNICAL OVERVIEW**

© Copyright 2019 Xilinx Distribution Statement A, Approved for Public Release: Distribution Unlimited

### WHAT ARE WE TRYING TO ACHIEVE

 Goal: Provide a Robust and Scalable Open Source Mixed Simulation Environment For Complex Designs

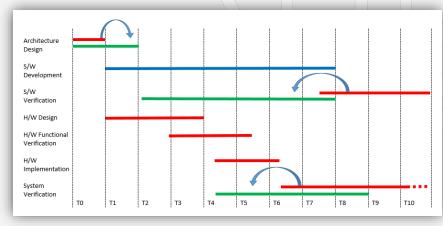




Phase Phase \_\_\_1b

### WHY ARE WE INVESTING

- Model Early, Integrate Continuously, Increase Adoption
  - Behavioral integration bugs are difficult to find and far more costly at later stages
  - Use hardware functional models to drive early and continuous integration
  - Suitable for software development, system architecture, functional hardware IP verification



### Enabling Open Source Hardware & Software Development

- Think System Level Solutions Full System Emulation
- Using Only Open Source Tools & Resources
- Enable the "ARMY" of open source software engineers to cross the hardware threshold

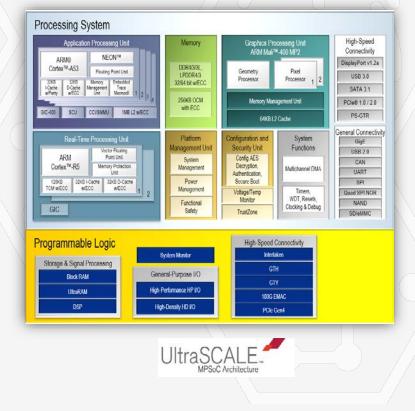
### THE BASICS OF QEMU

- QEMU is a Functionally Accurate Virtual Emulation Platform
  - QEMU Stands For <u>Quick</u> <u>EMU</u>lator
  - Develop, Debug & Run Your Software Exactly As You Would Do On A Board
  - But...Without the Hassle of Cables, Boards. Your Laptop Is All You Need.
- QEMU Platform Puts All the System Pieces Together
  - The Multi Core Heterogeneous SoC + Peripherals
  - The Programmable Logic IP
  - The Various Board Devices
- QEMU is Really Fast !
  - Our Zynq Ultrascale+ MPSoC QEMU can boot Linux Kernel in ~ 20 secs
- QEMU very popular among open source developers
  - KVM
  - GCC
  - Linux
  - Yocto
  - Linux Kernel

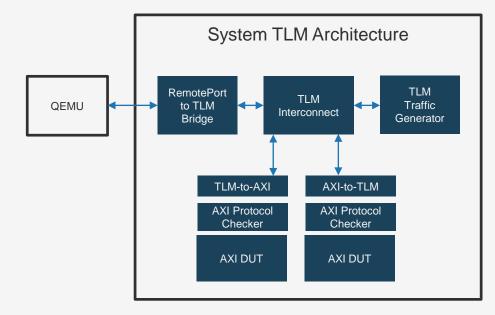


### WE PRACTICE WHAT WE PREACH

- We are just trying to make it better
  - At Xilinx we have successfully delivered 3 generations of products using full system mixed simulation environments
    - Boot ROM
    - Bootloaders
    - OS (Linux, FreeRTOS, Baremetal...)
    - Hypervisors (Xen)
    - Applications
    - Libraries.....
- When talking about complex designs
  - Think Zynq Ultrascale+ MPSoC is a highly powerful, highly flexible platform, featuring 4 x A53s, 2 x R5s and 1 MicroBlaze (PMU) along with rich collection of high speed peripherals and low power IOs



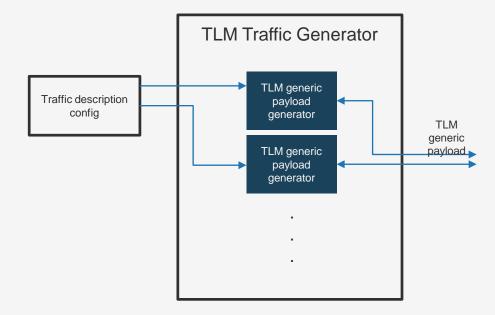
### **POSH QEMU CO-SIM TLM ARCHITECTURE**



#### >> Key Architectural Points

- Remote-port converted to TLM Generic Payloads
  - With Generic Attributes extensions carrying AXI specifics
    - Optional use by intermediate interconnect components
- TLM interconnect allows interconnection of TLM compatible or TLM wrapped devices (such as QEMU)
- TLM/AXI Bridges to direct mapping of TLM to AXI signal wiggling and vice versa but do not add optimizations (such as caching, buffering etc)
- Protocol checker is a passive component monitoring the AXI signals and reporting any protocol errors or inefficiencies that it discovers

### **POSH TECH DETAIL - TRAFFIC GENERATOR**



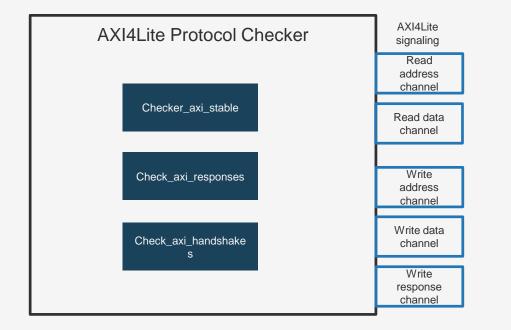
#### >> Key Architectural Points

- Generates TLM generic payloads according to a traffic configuration
- Protocol specifics are carried through TLM generic payload extensions
- Supports generation of multiple simultaneous TLM transactions
- Reusable with different type of TLM bridges

#### >> Next Steps

- Enable parsing of traffic configurations in JSON format

### **POSH TECH DETAIL –** AXI4LITE PROTOCOL CHECKER

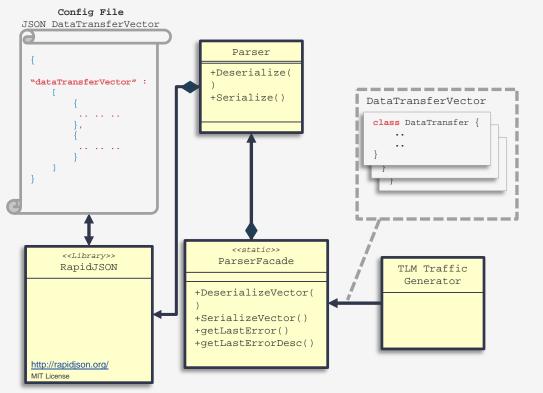


#### Key Architectural Points

>>

- Monitors and verifies AX4Lite signaling
- Consists of internal checkers
  - Each internal checker performs one or several checks on the AXI4Lite signaling
- A configuration enables and disables individual checks to be performed
- Reports errors through systemc's SC\_REPORT\_ERROR
- Examples of checks:
  - Check that signals are stable in any channel while valid is asserted but ready is low
  - Monitors and verifies transaction responses
  - Verifies that expected handshake signals are detected after receiving awvalid arvalid signals

### **POSH TECH DETAIL - CONFIG & PARSER**



- >> Key Architectural Points
  - Optimized human readable/editable config file
  - Atomic, Stateless, Fixed API Interface (Façade)
  - Underlying Lib completely decoupled from client (Parser)
    - No client dependencies on rapidjson
  - Supports description from partial or full object descriptors
  - Supports data randomization through annotations in JSON
  - Fully automated test suite.



# **IN-DEPTH TECHNICAL OVERVIEW**

SUPPORTED PROTOCOLS - AXI 4, AXI 4, AXI LITE, ACE, CHI

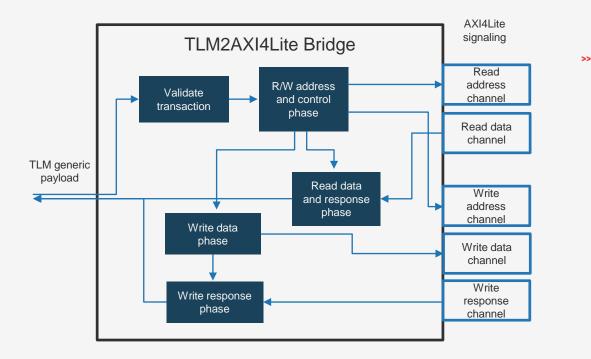
© Copyright 2019 Xilinx Distribution Statement A, Approved for Public Release: Distribution Unlimited



# AXI4, AXI3, AXI LITE

© Copyright 2019 Xilinx Distribution Statement A, Approved for Public Release: Distribution Unlimited

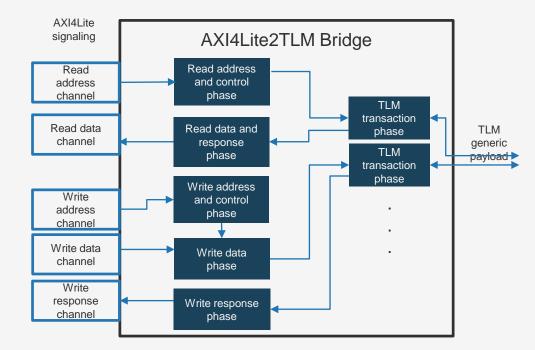
### **POSH TECH DETAIL –** TLM 2.0 TO AXI4LITE BRIDGE



#### Key Architectural Points

- Translates TLM generic payloads into AXI4Lite transactions
  - Large TLM transactions become multiple AXI4Lite transactions
- AXI4Lite Protocol specific details are carried through TLM generic payload extensions (for example AXI access permissions: secure, privileged)
- The phases of the transactions are run in parallel for supporting multiple outstanding AXI4Lite transactions
- An AXI4Lite traffic generator can be created by connecting the bridge to a TLM traffic generator (developed in the first deliverable)
- Examples of generated AXI4Lite transactions:
  - Narrow
  - Unaligned
  - With access permissions

### **POSH TECH DETAIL –** AXI4LITE TO TLM 2.0 BRIDGE



#### >> Key Architectural Points

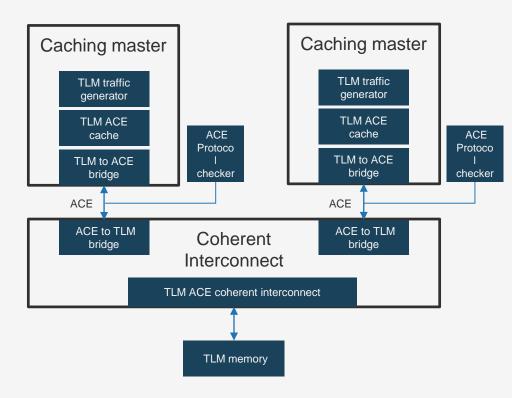
- Translates AXI4Lite transactions into TLM generic payloads
- AXI4Lite Protocol specific details are carried through TLM generic payload extensions (for example AXI access permissions: secure, privileged)
- The phases of the transactions are run in parallel for supporting multiple outstanding AXI transactions
- Issues TLM generic payloads in order (since AXI4Lite does not support AXI IDs)
- Examples of translated AXI4Lite transactions:
  - Narrow
  - Unaligned
  - With access permissions



# ACE

© Copyright 2019 Xilinx Distribution Statement A, Approved for Public Release: Distribution Unlimited

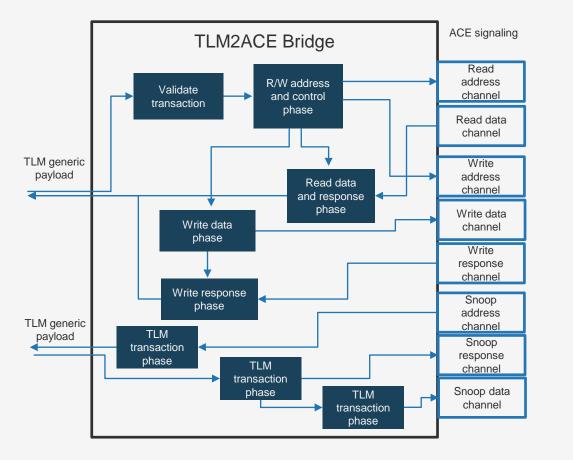
### **POSH TECH DETAIL – ACE TRAFFIC GENERATION**



#### >> Key Architectural Points

- Caching masters generating ACE transactions
- An ACE coherent interconnect generating snoop transactions

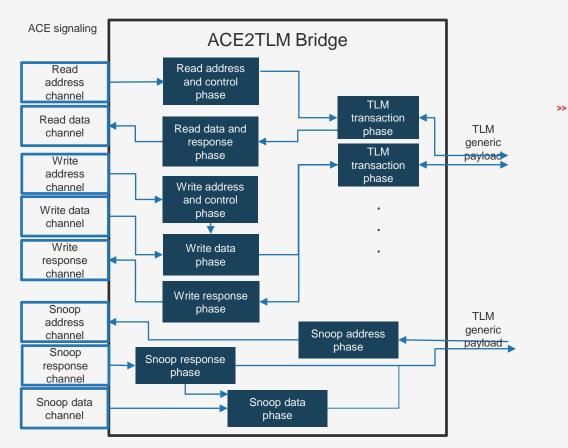
### **POSH TECH DETAIL – TLM 2.0 TO ACE BRIDGE**



#### » Key Architectural Points

- Translates TLM generic payloads into ACE transactions
- Translates ACE snoop transactions to TLM generic payloads
- ACE Protocol specific details are carried through TLM generic payload extensions (for example: snoop type, domain, barrier)
- The phases of the transactions are run in parallel for supporting multiple outstanding transactions
- Examples of translated ACE transactions:
  - Non snooping transactions
  - Coherent transactions
  - Memory update transactions
  - Cache maintenance transactions

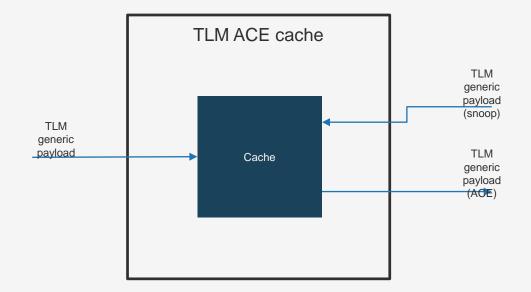
### **POSH TECH DETAIL – ACE TO TLM 2.0 BRIDGE**



#### Key Architectural Points

- Translates ACE transactions into TLM generic payloads
- Translates TLM generic payloads into snoop transactions
- ACE Protocol specific details are carried through TLM generic payload extensions (for example: snoop type, domain, barrier)
- The phases of the transactions are run in parallel for supporting multiple outstanding transactions
- Examples of translated ACE transactions:
  - Non snooping transactions
  - Coherent transactions
  - Memory update transactions
  - Cache maintenance transactions

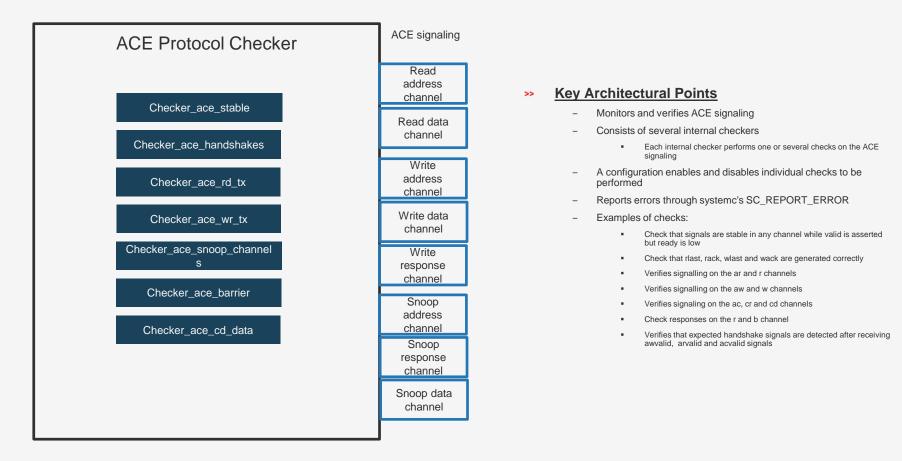
### **POSH TECH DETAIL – TLM ACE CACHE**



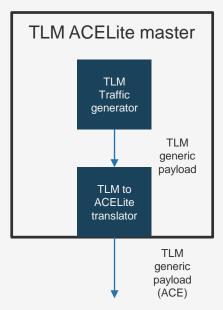
#### » Key Architectural Points

- Contains an internal cache with cache lines in one of the five states defined by the ACE protocol (invalid, unique clean, unique dirty, shared clean and shared dirty)
- Receives TLM generic payload from a master and generates ACE transactions if the required cache lines are not found/ or not in the correct state for serving the transaction
- Receives and acts on snoop transactions from an ACE interconnect

### **POSH TECH DETAIL – ACE PROTOCOL CHECKER**



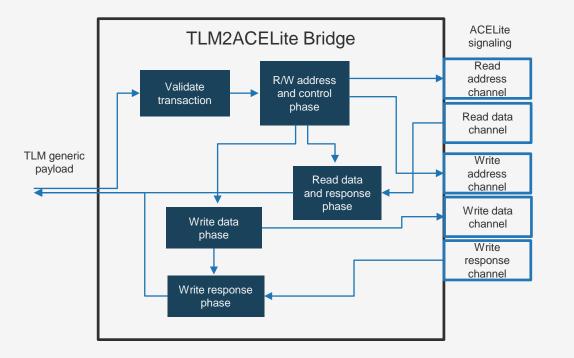
### **POSH TECH DETAIL – TLM ACELITE MASTER**



#### >> Key Architectural Points

- Contains a TLM traffic generator generating TLM generic payloads based on traffic descriptions
- Contains a TLM to ACELite translator translating the TLM generic payloads to ACELite transactions described in TLM (a generic payload with a generic extension containing ACELite attributes)
- Example of ACELite transactions being generated:
  - WriteUnique
  - ReadOnce

### **POSH TECH DETAIL –** TLM 2.0 TO ACELITE BRIDGE

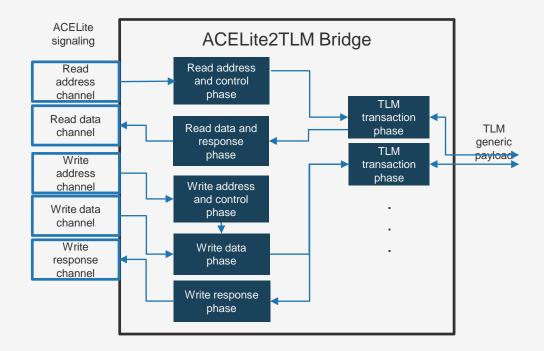


#### Key Architectural Points

>>

- Translates TLM generic payloads into ACELite transactions
- ACE Protocol specific details are carried through TLM generic payload extensions (for example: snoop type, domain, barrier)
- The phases of the transactions are run in parallel for supporting multiple outstanding transactions
- Examples of translated ACELite transactions:
  - Non snooping transactions
  - Coherent transactions
  - Cache maintenance transactions

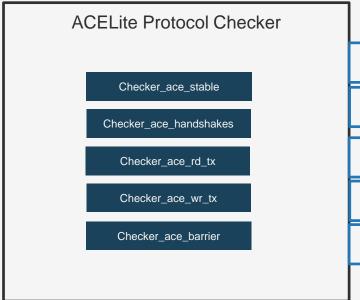
### **POSH TECH DETAIL – ACE TO TLM 2.0 BRIDGE**

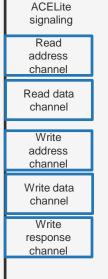


#### >> Key Architectural Points

- Translates ACELite transactions into TLM generic payloads
- ACE Protocol specific details are carried through TLM generic payload extensions (for example: snoop type, domain, barrier)
- The phases of the transactions are run in parallel for supporting multiple outstanding transactions
- Examples of translated ACELite transactions:
  - Non snooping transactions
  - Coherent transactions
  - Cache maintenance transactions

### **POSH TECH DETAIL –** ACELITE PROTOCOL CHECKER





#### Key Architectural Points

>>

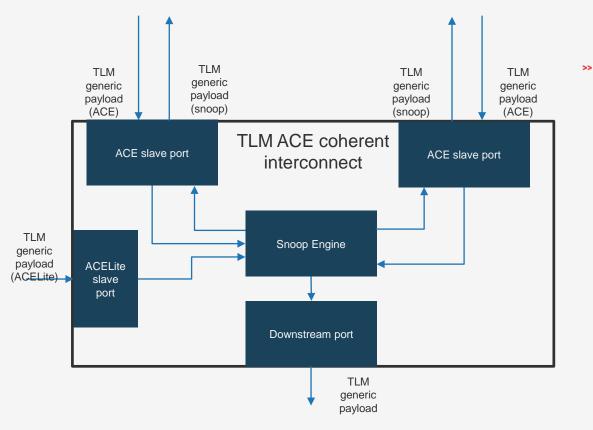
- Monitors and verifies ACELite signaling
- Consists of several internal checkers
  - Each internal checker performs one or several checks on the ACE signaling
- A configuration enables and disables individual checks to be performed
- Reports errors through systemc's SC\_REPORT\_ERROR
- Examples of checks:
  - Check that signals are stable in any channel while valid is asserted but ready is low
  - Verifies signalling on the ar and r channels
  - Verifies signalling on the aw and w channels
  - Checks responses on the r and b channel
  - Verifies that expected handshake signals are detected after receiving awvalid, and arvalid signals



# CHI

© Copyright 2019 Xilinx Distribution Statement A, Approved for Public Release: Distribution Unlimited

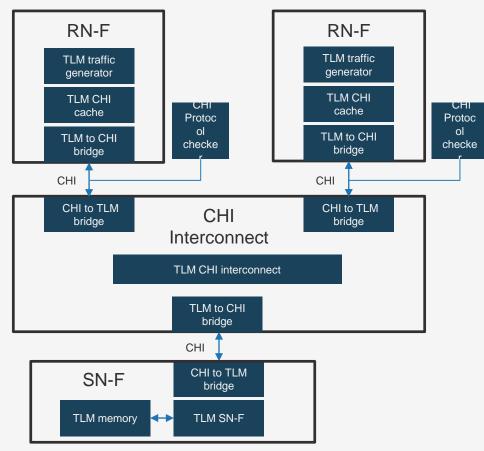
### **POSH TECH DETAIL –** TLM CHI INTERCONNECT



#### Key Architectural Points

- Consists of ACE and ACELite slave ports, a snoop engine and a downstream port
- The ACE slave ports receive TLM generic payloads with ACE transactions and forwards them to the snoop engine
- ACELite slave ports receives TLM generic payloads with ACELite transactions and forwards them to the snoop engine
- The snoop engine handles snooping of masters (through the ACE slave ports). If it is not a snooping transaction it is passed directly to the downstream port

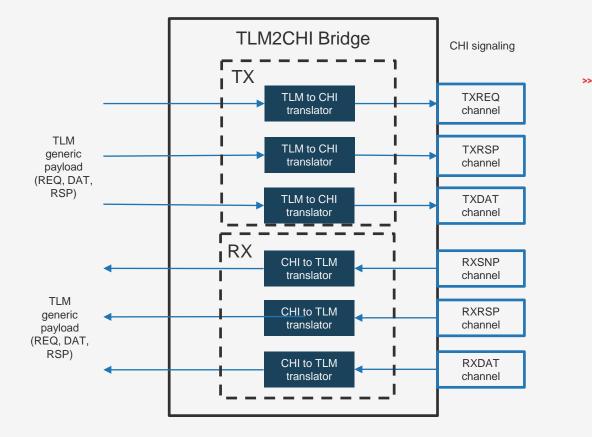
### **POSH TECH DETAIL – CHI TRAFFIC GENERATION**



#### >> Key Architectural Points

- Request nodes (RN-F) generating CHI transactions
- An CHI interconnect (HN-F, MN) generating downstream and snoop transactions

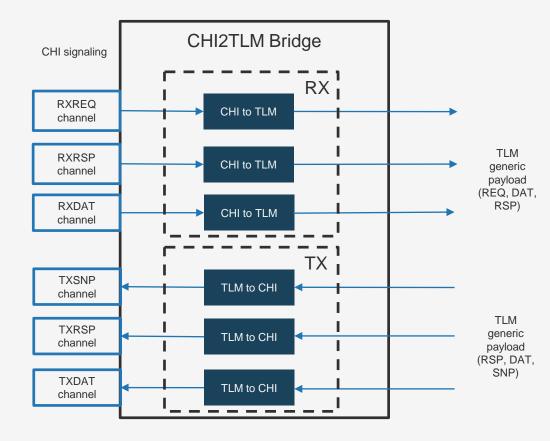
### **POSH TECH DETAIL – TLM 2.0 CHI BRIDGE**



#### Key Architectural Points

- Translates TLM generic payloads into CHI flits on the outbound link
- Translates CHI flits into TLM generic payloads on the inbound link
- CHI Protocol specific details are carried through a TLM generic payload extensions (for example: TgtID, SrcID, Opcode)
- Translators contain queues for supporting multiple outstanding transactions
- Examples of translated CHI flits on the outbound link:
  - Read, dataless, write, atomic requests
  - Data flits containing data for write transactions or snoop requests
  - Requester response flits (for example carrying CompAck)
- Examples of translated CHI flits on the inbound link:
  - Snoop request flits
  - Data flits for read transactions
  - Completer response flits (for example carrying Comp, CompDBIDResp)

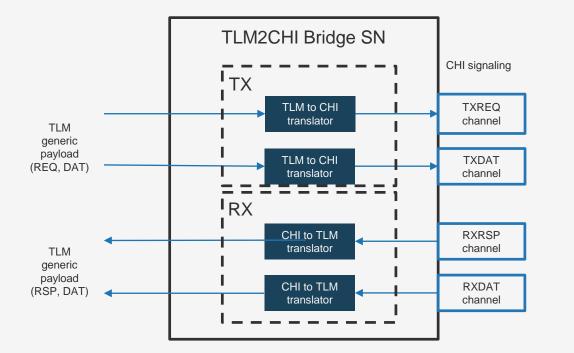
### **POSH TECH DETAIL – CHI TLM 2.0 BRIDGE**



#### » Key Architectural Points

- Translates CHI flits into TLM generic payloads on the inbound link
- Translates TLM generic payloads into CHI flits on the outbound link
- CHI Protocol specific details are carried through a TLM generic payload extensions (for example: TgtID, SrcID, Opcode)
- Translators contain queues for supporting multiple outstanding transactions
- Examples of translated CHI flits on the outbound link:
  - Snoop request flits
  - Data flits for read transactions
  - Completer response flits (for example carrying Comp, CompDBIDResp)
- Examples of translated CHI flits on the inbound link:
  - Read, dataless, write, atomic requests
  - Data flits containing data for write transactions or snoop requests
  - Requester response flits (for example carrying CompAck)

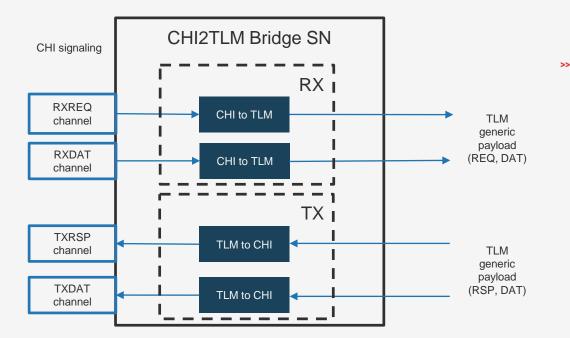
### **POSH TECH DETAIL – TLM 2.0 CHI BRIDGE SN**



#### » Key Architectural Points

- Translates TLM generic payloads into CHI flits on the outbound link
- Translates CHI flits into TLM generic payloads on the inbound link
- CHI Protocol specific details are carried through a TLM generic payload extensions (for example: TgtID, SrcID, Opcode)
- Translators contain queues for supporting multiple outstanding transactions
- Examples of translated CHI flits on the outbound link:
  - Read, dataless, write, atomic requests
  - Data flits containing data for write transactions
- Examples of translated CHI flits on the inbound link:
  - Data flits for read transactions
  - Completer response flits (for example carrying CompDBIDResp)

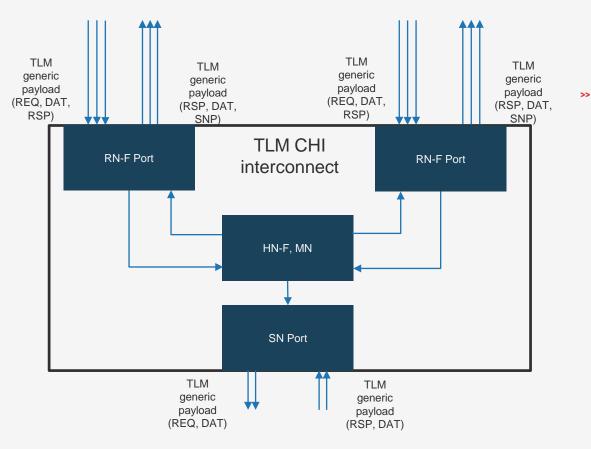
### **POSH TECH DETAIL – CHI TLM 2.0 BRIDGE SN**



#### Key Architectural Points

- Translates CHI flits into TLM generic payloads on the inbound link
- Translates TLM generic payloads into CHI flits on the outbound link
- CHI Protocol specific details are carried through a TLM generic payload extensions (for example: TgtID, SrcID, Opcode)
- Translators contain queues for supporting multiple outstanding transactions
- Examples of translated CHI flits on the inbound link:
  - Read, dataless, write, atomic requests
  - Data flits containing data for write transactions
- Examples of translated CHI flits on the outbound link:
  - Data flits containing data for read transactions
  - Completer response flits (for example carrying CompDBIDResp)

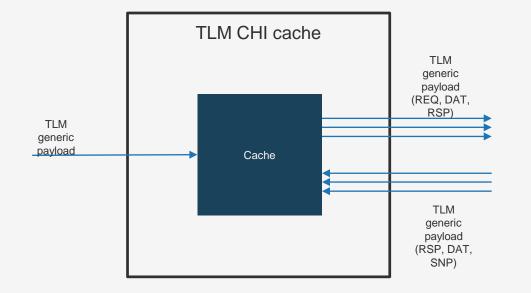
### **POSH TECH DETAIL –** TLM CHI INTERCONNECT



#### Key Architectural Points

- Consists of RN-F ports, an SN port, a HN-F and an MN
- The RN-F ports receive TLM generic payloads describing CHI request, dat and response flits and forwards them to the HN-F (DVM transactions are forwarded to the MN)
- The HN-F snoops the required RN-Fs (through the RN-F ports).
  When required the HN-F issues a transaction towards the Slave Node (through the SN port)
- DVM transactions received on the RN-F ports are forwarded to and processed by the MN

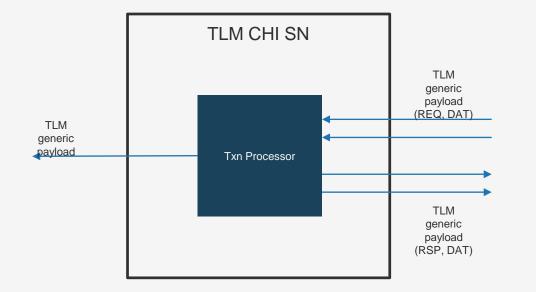
### **POSH TECH DETAIL – TLM CHI CACHE**



#### >> Key Architectural Points

- Contains an internal cache with cache lines in one of the states defined by the CHI protocol (invalid, unique clean, unique clean empty, unique dirty, unique dirty partial, shared clean and shared dirty)
- Receives TLM generic payloads and generates CHI transactions (TLM generic payloads containing an CHI attributes extension) if the required cache lines are not found / or not in the correct state for serving the transaction
- Receives and acts on snoop transactions from an CHI interconnect

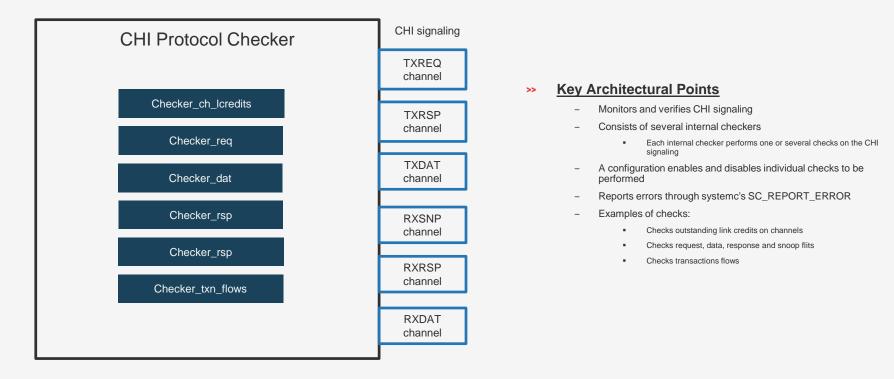
### **POSH TECH DETAIL – TLM CHI SN**



#### » Key Architectural Points

- Receives TLM generic payloads describing CHI requests from an ICN
- Operates as an SN-F on CHI request and generates TLM generic payloads towards a connected TLM module (memory) and or generic payloads towards the ICN describing CHI dat / responses as required by the request
- Example of processed CHI requests:
  - Non-snoopable read, write and atomic requests

### **POSH TECH DETAIL – CHI PROTOCOL CHECKER**





## ERI ELECTRONICS RESURGENCE INITIATIVE S U M M I T

### 2019 | DETROIT, MI | JULY 15-17

This research was developed with funding from the Defense Advanced Research Projects Agency (DARPA). The views, opinions and/or findings expressed are those of the author and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

© Copyright 2019 Xilinx, Inc. Xilinx, the Xilinx logo, Artix, ISE, Kintex, Spartan, Virtex, Zynq, and other designated brands included herein are trademarks of Xilinx in the United States and other countries. All other trademarks are the property of their respective owners. DISCLAIMER To the maximum extent permitted by applicable law: (1) the information contained herein (the "Materials") are made available "AS IS" and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not modify, the Materials without prior written consent. If You reproduce, distribute, or publicly display the Materials, you must retain the foregoing copyright notice and this disclaimer

© Copyright 2019 Xilinx Distribution Statement A, Approved for Public Release: Distribution Unlimited