2016 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

Analysis of TLM-2.0 and it's Applicability to Non Memory Mapped Interfaces

DELBERGUE Guillaume, BURTON Mark, LE GAL Bertrand and JEGO Christophe

GreenSocs Ltd / Bordeaux INP, CNRS IMS, UMR 5218

université de BORDEAUX

ims

GreenSocs®

2016 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

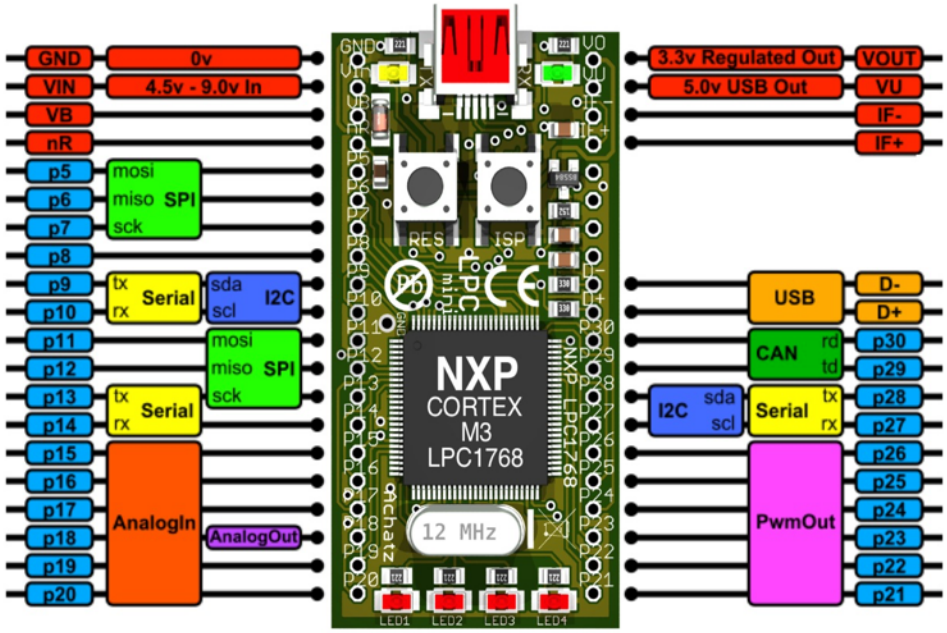
What's the plan ?

- Introduction
- Towards a definition of a transaction
  - OSI Model transaction
- Non Memory Mapped Interfaces:
  - One to One protocols, One to Many protocols, Many to Many protocols
- Ordering and timing of the simulation
- Improved TLM Quantum Keeper
  - Notification system, Experimental results
- Improvements to the existing TLM-2.0 standard
  - Socket and binding, Payload, Phases, Generic Serial Protocol
- TLM Interface Kit Pattern
- Conclusion

2016 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

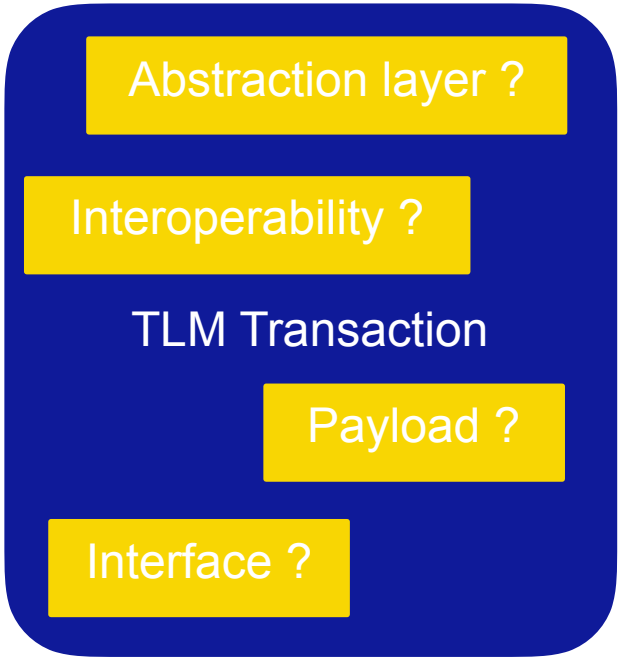
Non memory mapped interfaces

- Modern systems have a lot of interfaces
- One to One serials protocols (GPIO, IRQ, RS232)
- One to Many protocols (RS485, SPI)
- Many to Many protocols (I2C, CAN)



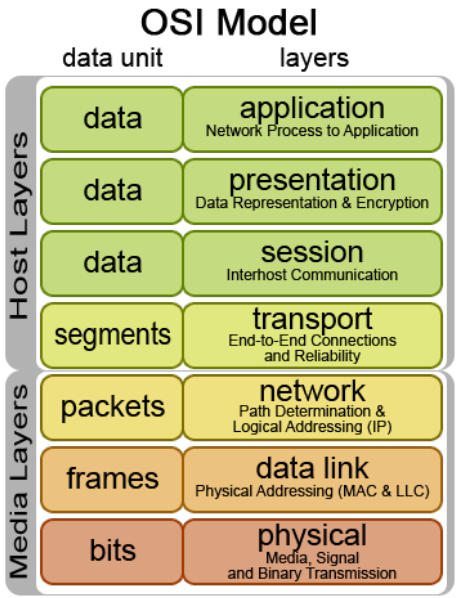
Current definition of a transaction ?

- TLM-2.0 never standardised a TLM transaction
- The TLM-2.0 standard tries to ensure **interoperability** of models written at two abstraction levels
- Transactor can be required to bridge different abstraction levels
- Absence of literature on the subject, no clear agreement
- Confusing for designers
- ! **Critical: Have an agreement of what constitutes a transaction.**



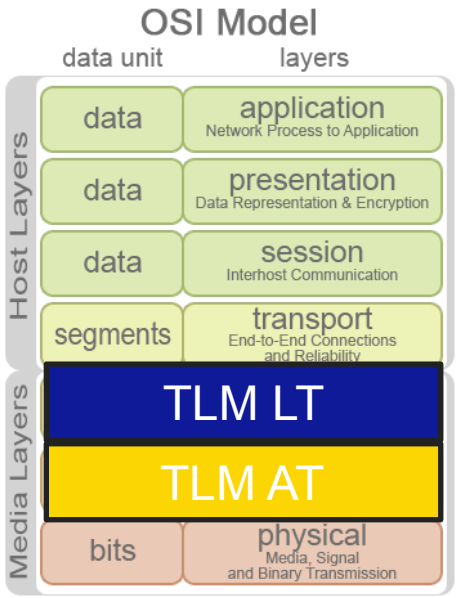
OSI Model

- **Existing standard communication model**
- Divides communication protocols into **multiple layers of abstraction**
- From physical layer (bit level) to raw data on communication channel.
- Layer 1: Bit level, voltage, electrical characteristics
- Layer 2: Data link layer, detect frame borders, manage errors, data congestion, buffering… No data routing
- Layer 3: Routing to support networks and sub networks. Handle “One to Many” and “Many to Many” communications



TLM abstraction layer and OSI Model

- OSI Model : good basis to determine what constitutes a transaction
- ✓ **Proposal:** Second and third layer (data and network) of OSI model map directly onto TLM AT and LT (higher levels are parts of the software domain)
- Layer three provides enough information to route data between nodes (One to Many / Many to Many communications)
- Layer three contains data and address just as a TLM generic protocol transaction (and is route-able)
- ✓ **Proposal: Definition of a transaction which works for both non memory mapped and memory mapped interfaces.**
- We will now go on to analyze real interfaces, and how they can be interpreted using the OSI model



UART example

Bi-directional convenience TLM UART socket

UART IP VENDOR 1

UART IP VENDOR 2

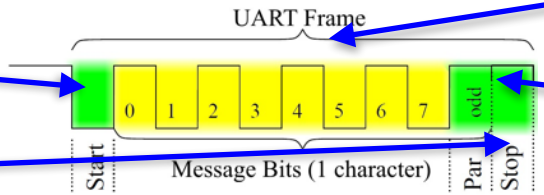
OSI layer 2

- start
- stop

TLM UART PHASES

- start
- stop

OSI / TLM Mapping



TLM UART PAYLOAD

- data_ptr
- data_length
- parity ?

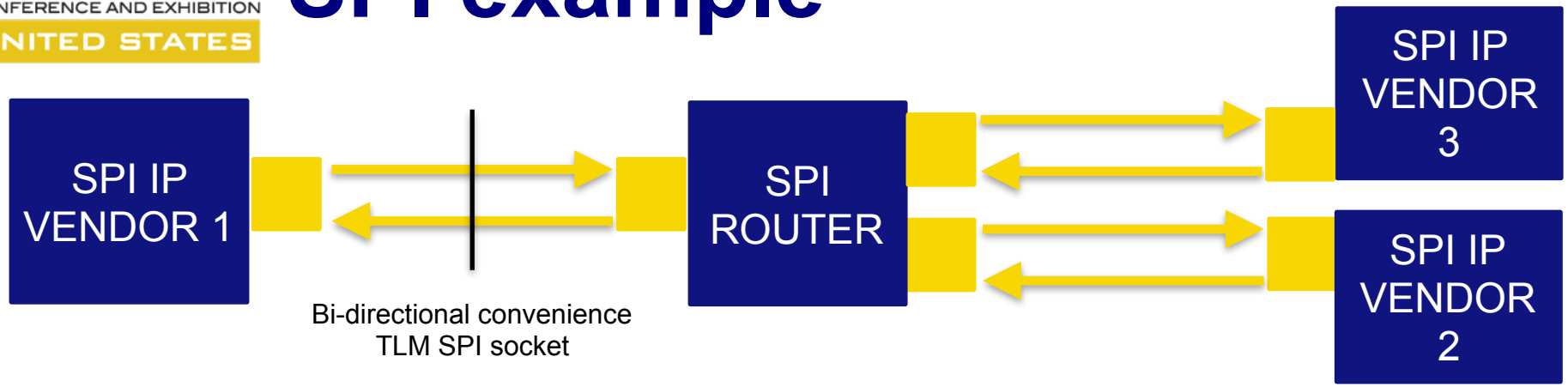
Extension mechanism

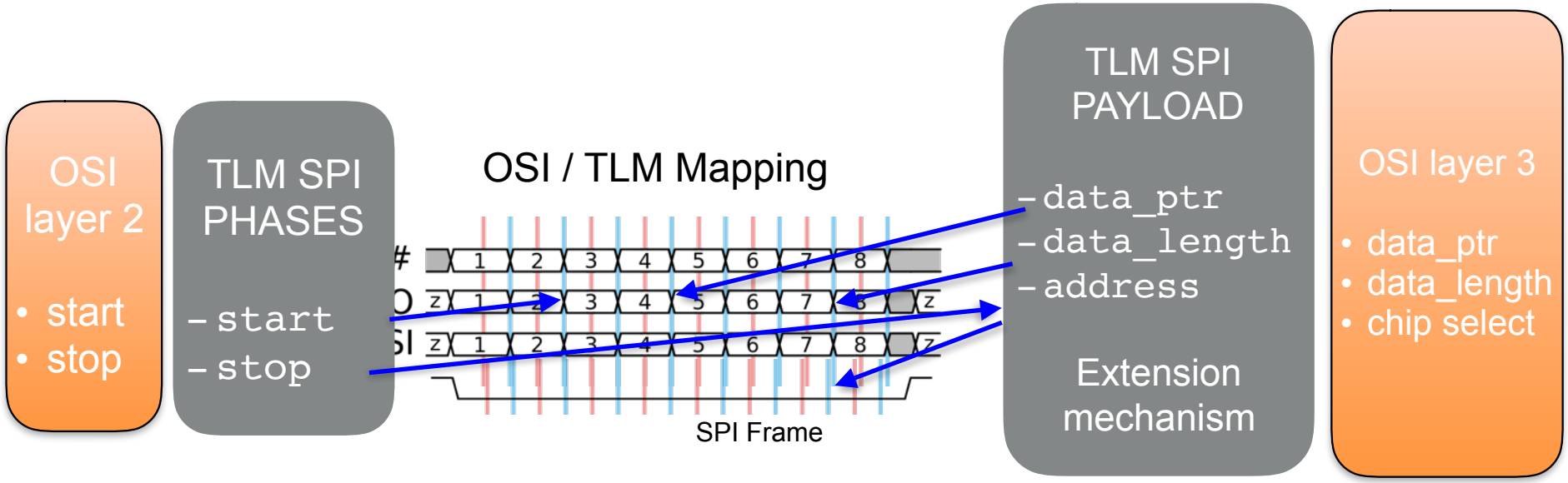
OSI layer 3

- data_ptr
- data_length
- parity ?

- ! Exact content has to be defined by the TLM WG

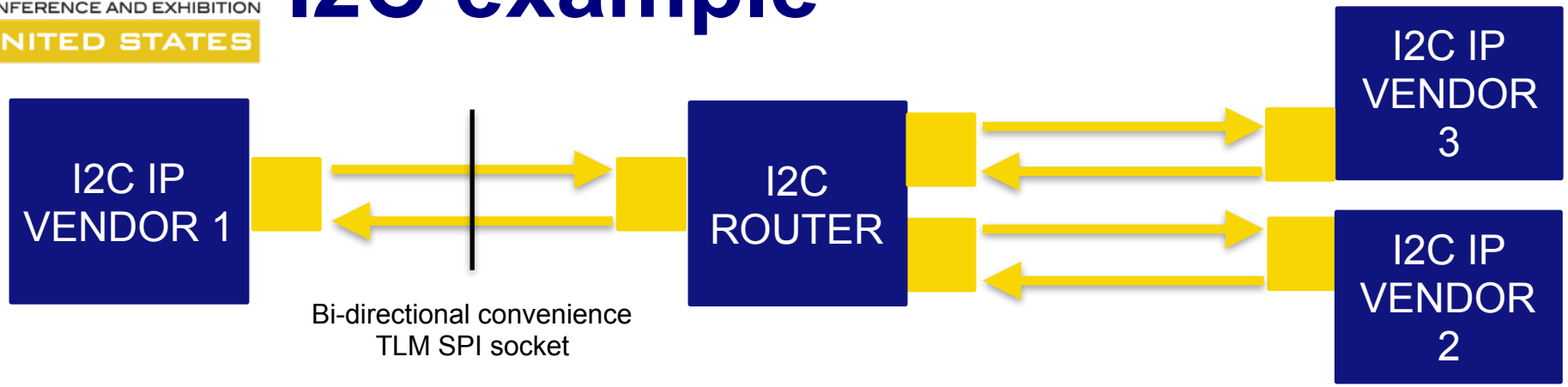
SPI example





- ! Exact content has to be defined by the TLM WG

I2C example



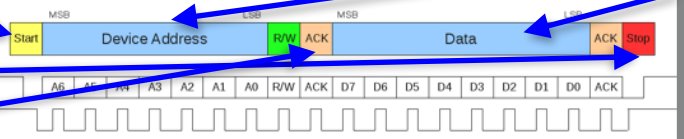
OSI layer 2

- start
- stop
- ack
- pause
- restart

TLM I2C PHASES

- start
- stop
- ack
- pause
- restart

OSI / TLM Mapping



TLM I2C PAYLOAD

- address
- data_ptr
- data_length
- cmd

Extension mechanism

OSI layer 3

- data_ptr
- data_length
- address
- cmd

- ! Exact content has to be defined by the TLM WG

Ordering and timing of the simulation

- SystemC introduced a base of time
  - **SystemC has an event mechanism** to interoperate with
- **TLM-2.0 introduces another base of time : time decoupling** and allows models to run ahead of simulation time which decrease synchronisations and speed up simulation (**quantums**, **notion of annotated time on transactions)**
  - ❗ But **TLM-2.0 didn't implement an event mechanism** to deal with TLM base of time.
- **TLM Quantum Keeper** utils class helps to manage local time
- Hard to find **the best value for the quantum value**
- 🔥 Requirements for a universal timing interface between models

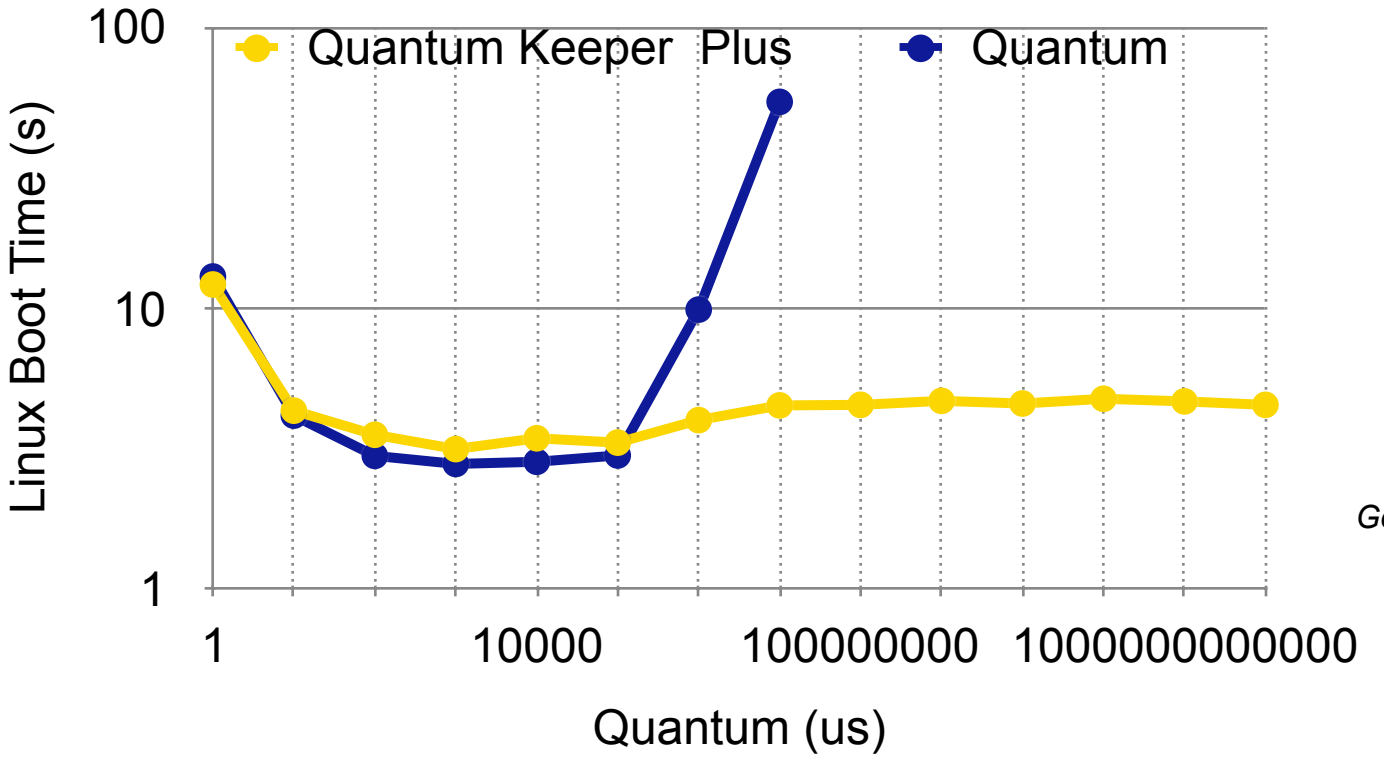
Improved TLM Quantum Keeper : Notification System

👉 : Base time on quantum time instead of simulation time

- ✔ **Proposal:** Improve TLM Quantum Keeper to **provide an event queue based on the local quantum time adding methods to register callback at a certain time**
- A timer can use this mechanism to run a clock without synchronisation with the SystemC kernel
- Example code available in the paper but the key is the API

Improved TLM Quantum Keeper : Experimental results

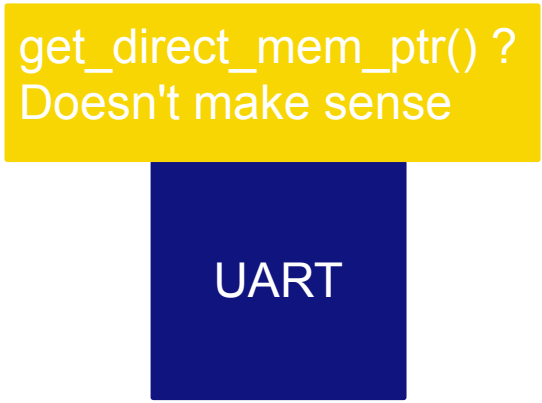
- First issue : TLM Quantum Keeper needs extending (**new API**)
- Second issue : TLM Quantum Keeper needs to be **"findable"** : first class CCI object ?



Get the best of your quantum 😉

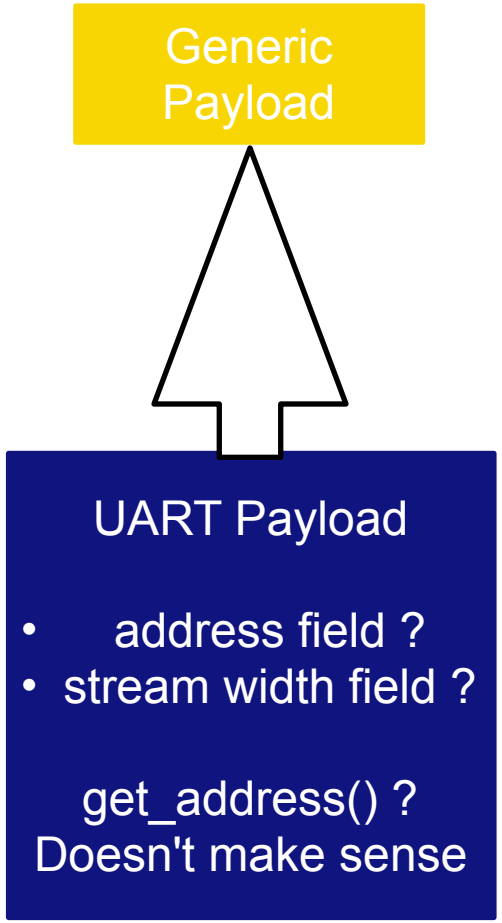
Improvements to the existing TLM-2.0 standard : Transports and sockets

- TLM sockets class are based on forward and backward transport interfaces containing blocking/non blocking function, DMI, debug…
- DMI is not suitable for all interfaces (Signal, UART…)
- Need for bidirectional sockets. GreenSocket instances a pair of initiator and target socket : good candidate
- If no router used, need support for bidirectional multi sockets



Improvements to the existing TLM-2.0 standard : Payload and phases

- Generic payload : the name is somewhat misleading. The “generic” is specific to memory mapped interfaces. Unused fields for some protocols.
- TLM default enumeration phases doesn’t cover the needs of each protocol
- Phases: Can easily add new phases using a macro but default one are also too specific for some protocols



2016 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

TLM-2.0 proposals

- ✓**Proposal:** Cleaned version of the generic payload containing only the extension mechanism to easily inherit and add specific protocol fields (see paper for details)
- ✓**Proposal:** Cleaned version adding a template parameter to specify phases
- ✓**Proposal:** Cleaned version of transport classes to be more generic and less specific to memory mapped cases (See paper for details)
- We believe this can be done while maintaining backward compatibility

TLM Interface Kit Pattern

- OCP SLD kit : Open full TLM kit supporting many protocols, based on TLM-2.0 generic protocol, provides transactors. Based on that kit, we suggest the following:
  - OSI analysis : Protocol analysis to guide choice of payload, phases and documentation
  - Protocol : TLM implementation of payload, phases…
  - Convenience sockets: for common use cases
  - Loggers and checkers
  - Transactors: supports real hardware / RTL level model
  - Host interfaces: “backends” for interfaces
  - UVM and routers
  - Documentation: mostly inherit from current TLM
  - Legal: need a clear license, IP using this interface will become a derivative work

2016 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

Conclusion

- Formal approach to **define a transaction in TLM**
- Introduced the use of OSI layered communication model to define transaction
- Review of common protocols
- Analysis of time and models
- Proposed an **improved version of TLM Quantum Keeper** in order to solve models interoperability
- Examination of TLM-2.0 and proposals for an improved version of TLM-2.0
- Blue prints of a full TLM Interface Kit
- Work will be upstreamed to TLM-WG and will be open source