What is needed on top of TLM-2 for bigger Systems?

Jerome Cornet - ST

Martin Schnieringer - Bosch GmbH

Agenda

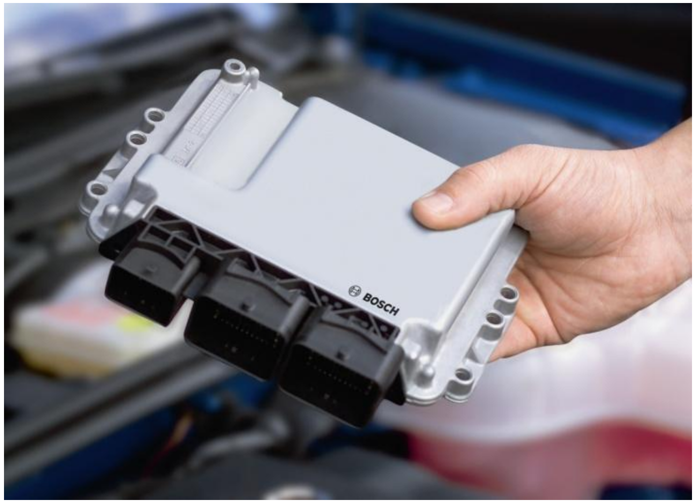
- Introduction
- Example of Serial Protocols
- TLM Standard Design
- Serial Protocol Scenarios
- Relation with Existing Standards
- Outlook

Introduction

Motivation:

- Early & efficient SW development on Virtual Prototypes of Electronic Control Units (ECU)
- ECU (network) is assembled from IP of different vendors communicating via serial protocols e.g. SPI, CAN, I2C, LIN, FlexRay, Ethernet etc.

Avoid effort when connecting simulation models



Goal:

- Establish SystemC TLM modeling standard for serial protocols

Examples of Serial Protocols

- Controller Area Network (CAN)
  - Inter-ECU protocol
  - Standardized frames with identifier/data, asynchronous, collision bus
  - Nodes are both master and slave, multi-master (identifier arbitration)
- I2C
  - Intra-ECU protocol
  - Standardized frames with address, R/W, ACK, Data..., synchronous, collision bus
  - Nodes are both master and slave, multi-master (bit-level arbitration)
- Serial Peripheral Interface (SPI)
  - Intra-ECU protocol
  - No standard frame, synchronous, one line per direction full duplex
  - Different connection schemes e.g. broadcast, daisy chaining, single-master
  - Typically nodes are either master or slave

TLM Standard Design Pitfalls

- Being too generic
  - Pros: Simple to design, everything covered at once
  - Cons
    - Often overlook corner cases of actual protocol being modeled
    - Difficult to tailor a single framework for different modeling needs
    - Complicated rules, limited ease of use as a result
- Being too specific
  - Pros: Easy to ensure interoperability, easy to use
  - Cons:
    - Difficult to design and agree upon
    - Potentially different solutions for common problems

Good Properties of a TLM Standard

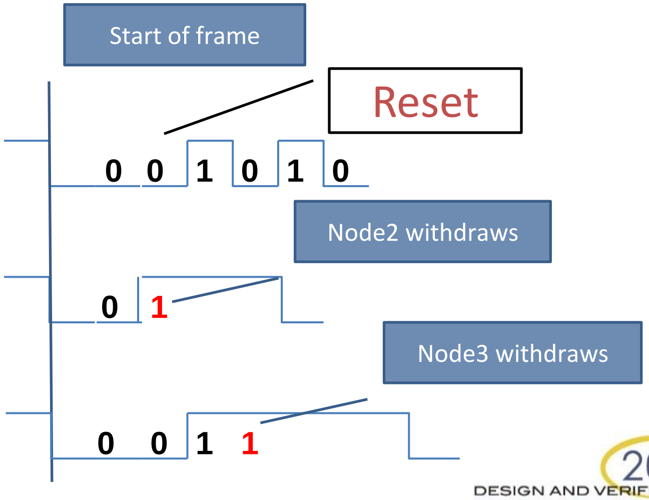
- **Simple**
  - Straightforward protocol rules
  - Easy to understand and to comply with
- **Powerful**
  - Capture relevant protocol features and scenarios
  - Well-defined/identified corner cases
- **Flexible**
  - Does not force modeling objects to be used by the model developer
  - Does not impose artificial limits on modeling context (proper hierarchy support, etc.)

Approach

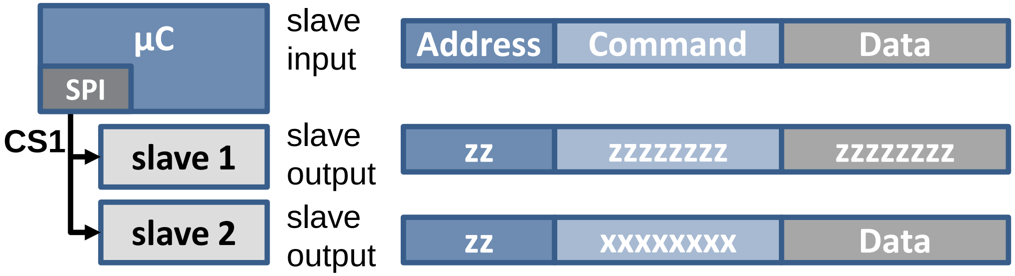
- Initial gathering of small set of participants
  - ASTC, Infineon, ST, Synopsys, Bosch …
- Identify a set of real-world scenarios for each protocol
- Actual code for “testing” modeling approach on scenarios
  - Allows identifying pros/cons of modeling approaches
  - Better exchange and understanding among partners!
- Convergence on single modeling code/approach
- Donation to Accellera

Scenario CAN Reset

- Reset:
  - During arbitration -> winning node changes when node with lowest ID is reset.
  - During frame transmission -> results in error frame sent by detecting node.
- TLM: When to arbitrate? Start/end of arbitration field?
- Implementation:
  - Send arbitration field @ beginning, allow updates for RTL co-sim
  - Determine winning node @ end of arbitration field
    - Interface supports cancellation of transaction that did win



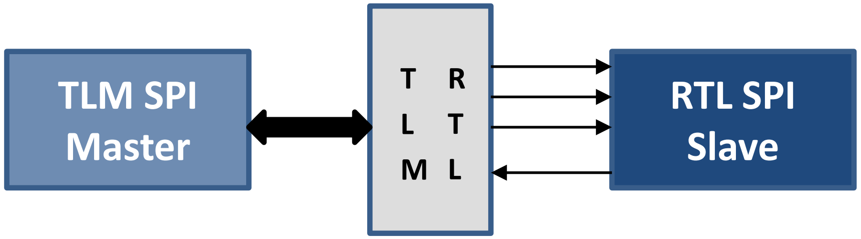
Scenario SPI Broadcast



- µC has less chipselect (CS) ports than slaves to connect
- Multiple slaves are addressed with the same CS line
- Outcome: No slave shall block
  - Blocking does not allow „next“ slave to receive frame with Start Of Frame (SOF) hence can not reply on time
- Implementation: Use of non-blocking calls

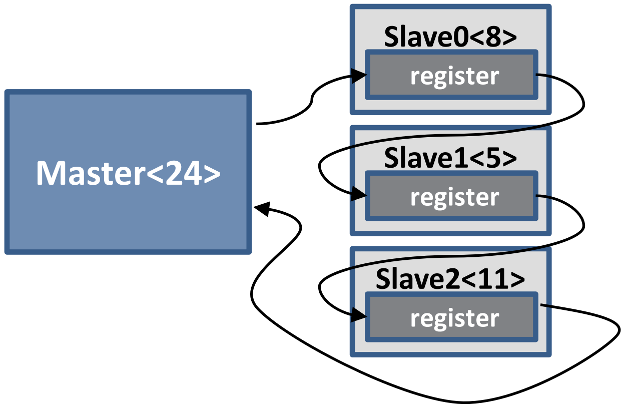
Scenario SPI RTL Co-Simulation

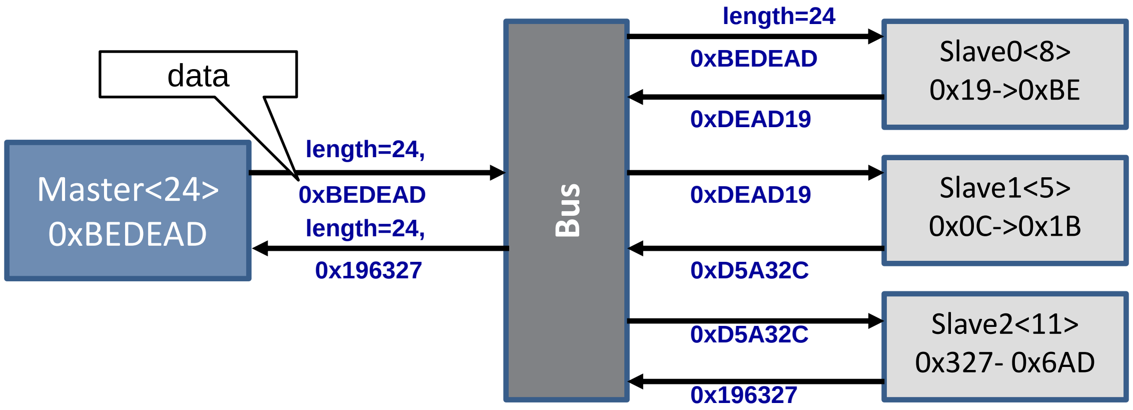
- Reuse HDL code converted to SystemC (legacy IP)
- Outcome:
  - Start of frame (SOF) needed so adapter can start RTL transfer
    - E.g. CAN SOF bit has to be created right away
  - Transaction update mechanism needed as TLM master gets slave data bit by bit and preferably first bit ASAP
- Implementation:
  - SOF and Update „phase“ introduced for TLM implementation



Scenario SPI Daisy-Chain

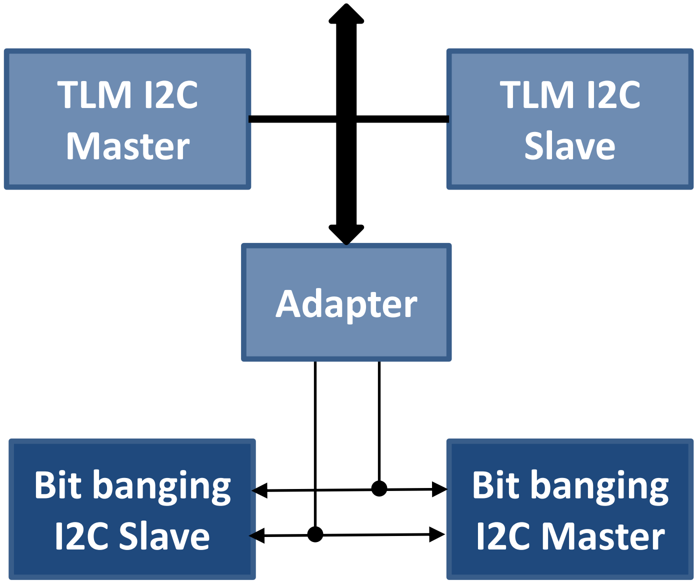
- Master data is shifted (out) e.g. by 24 bits
- Slave IP shall not know master register length upfront
- Outcome: Data container of 8 bit wide Slave0 must store 24 bits to be able to pass on remaining master bits
- Implementation: Transaction container supports dynamic allocation





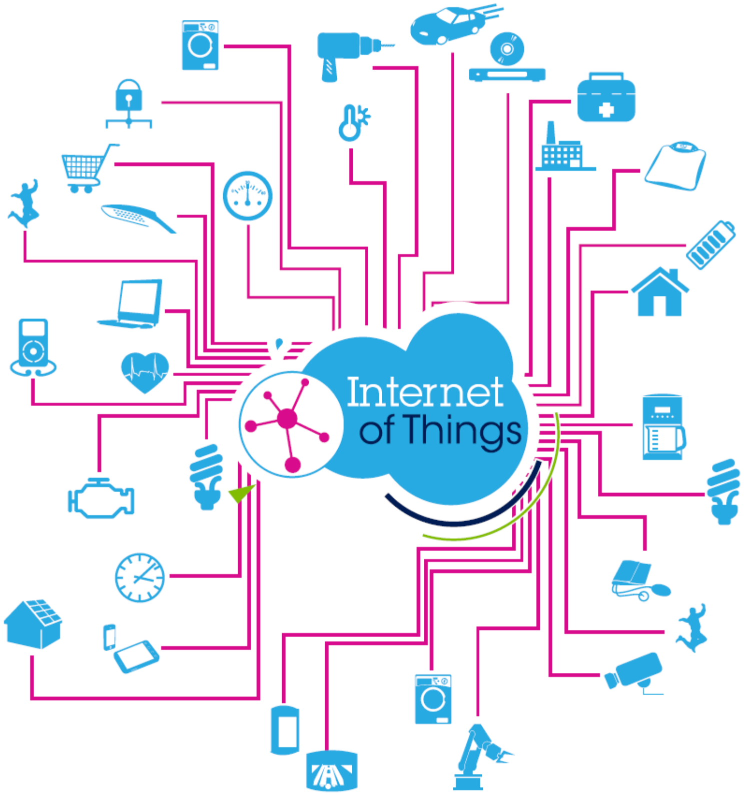
Scenario I2C

- Microcontrollers potentially using software emulation technique
  - GPIO driven directly by embedded software
  - RTL-like level of granularity
  - Should communicate with other modules transparently
- Arbitration done at bit-level
  - How to guarantee proper synchronization of everyone?



Multi-Domain Usage

- SPI and I2C not limited to automotive
- Virtual Prototyping for IoT
  - Connecting very different microcontrollers, hardware, sensors…
  - Validating a global system
  - Interoperability very needed here!
  - Required to take into account all cases
  - One-size-fit-all approach again not appropriate



Relation with Existing Standards

- TLM-1
  - Very generic, sometimes good basis to build upon
  - Many interfaces but still not covering every needs
- TLM-2
  - Designed initially for memory-mapped bus protocols, very flexible
    - Many rules needed to constrain usage for serial protocols
  - sc_export usage complicates hierarchy support
- Guidelines for future standard
  - SystemC based, use sc_port
  - Thin interface layer with non-blocking calls: Interfaces between models
  - Convenience layer: eases modeling and guarantees many rules by construction: Interface with model‘s developer

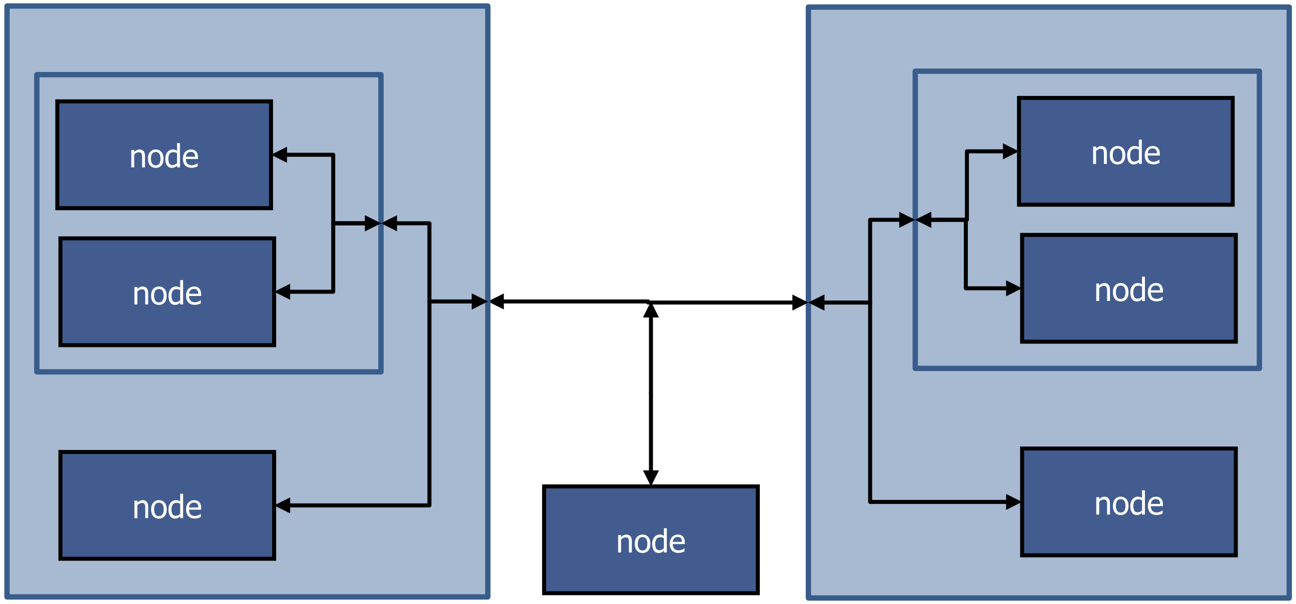
Outlook

- Current status
  - Set of scenarios defined for CAN, SPI and I2C
    - Donated to Accellera TLM Working Group
  - Several prototypes with actual code demonstrated internally
  - Discussions started at Accellera
- Next steps
  - Address more protocols: Ethernet, FlexRay ...
  - Converge on code basis to be donated to Accellera
  - Involvement of more parties: **Please join us!**
  - Standard defined at Accellera, virtual plugfest with multiple vendors

Questions

Backup

Hierarchy example


node
node
node
node
node
node
node