

SystemC Verification Working Group

Adam Rose

Networking and Computing Systems Group

Semiconductor Products Sector

Motorola

Why System Level Modeling ?

Market Driven
Architectures

Design Realities

Why System Level Modeling ?

Market Driven
Architectures

optimistic reuse
assumptions

Design Realities

Why System Level Modeling ?

Market Driven
Architectures

optimistic reuse
assumptions

optimistic timescale /
resource issues

Design Realities

Why System Level Modeling ?

Market Driven
Architectures

optimistic reuse
assumptions

optimistic timescale /
resource issues

Inadequate verification
infrastructure

Design Realities

Why System Level Modeling ?

Market Driven
Architectures

Design Realities

Why System Level Modeling ?

Rapid Virtual Prototype

Market Driven
Architectures

Design Realities

Why System Level Modeling ?

Rapid Virtual Prototype

Market Driven
Architectures

Early functional
sanity checks on
existing ip

Design Realities

Why System Level Modeling ?

Rapid Virtual Prototype

Market Driven
Architectures

Early functional
sanity checks on
existing ip

Design Realities

Early and accurate
communication of new
ip requirements

Why System Level Modeling ?

Rapid Virtual Prototype

Market Driven
Architectures

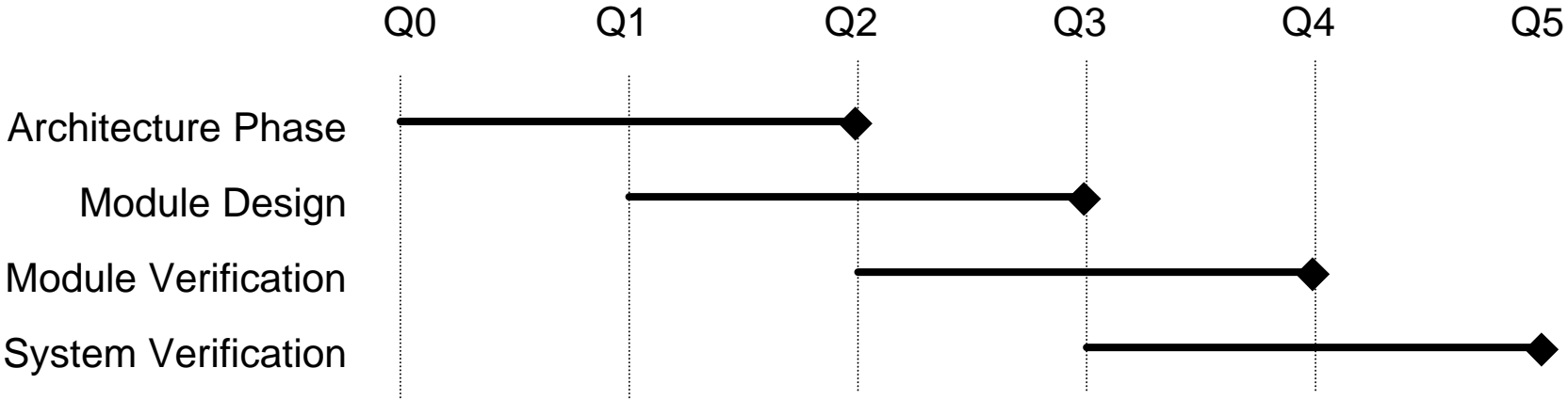
Early functional
sanity checks on
existing ip

Design Realities

Early and accurate
communication of new
ip requirements

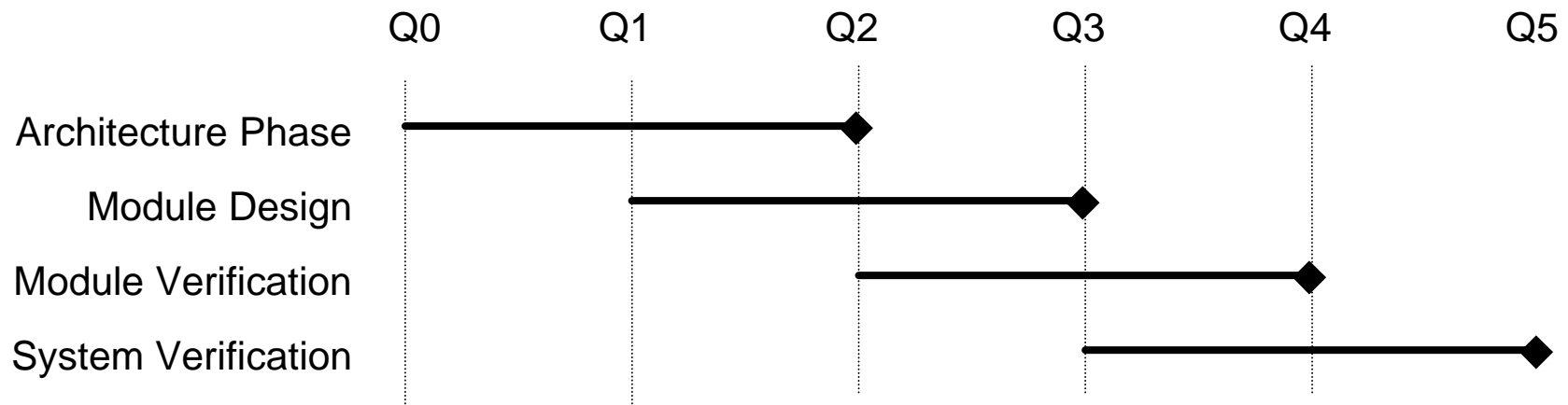
Early development of
verification infrastructure

Why System Level Verification ?

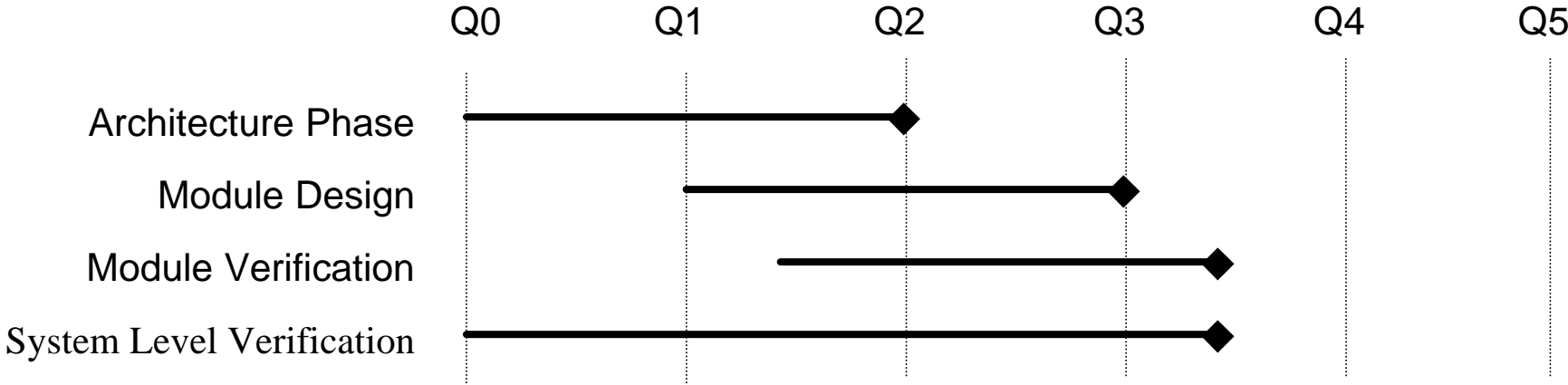


Why System Level Verification ?

System Level Verification is always done last, is always on the critical path - and is always done too late for architectural redesign

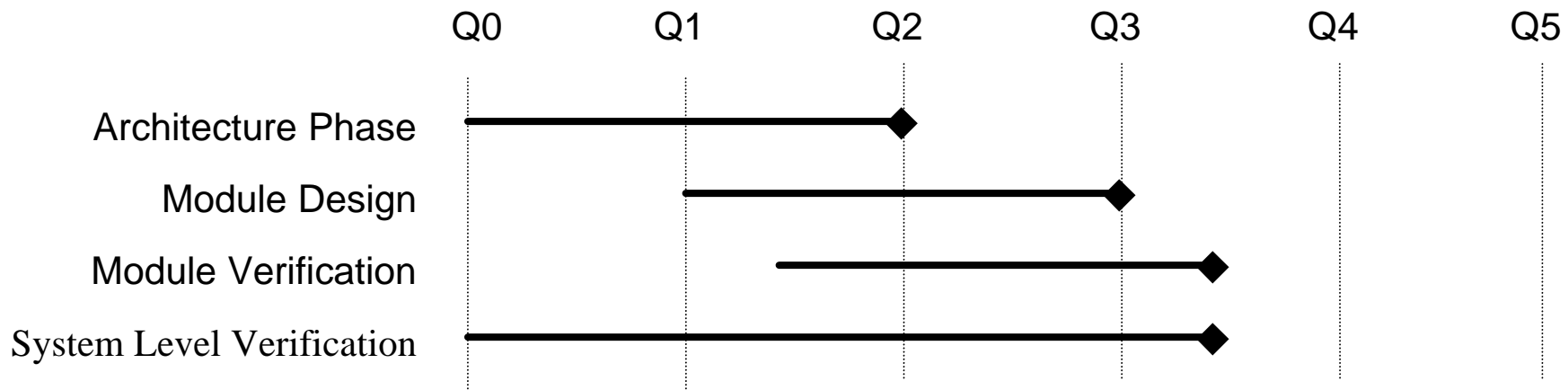


Why System Level Verification ?



Why System Level Verification ?

System Level Verification is done throughout the lifetime of the project



Why C++ ?

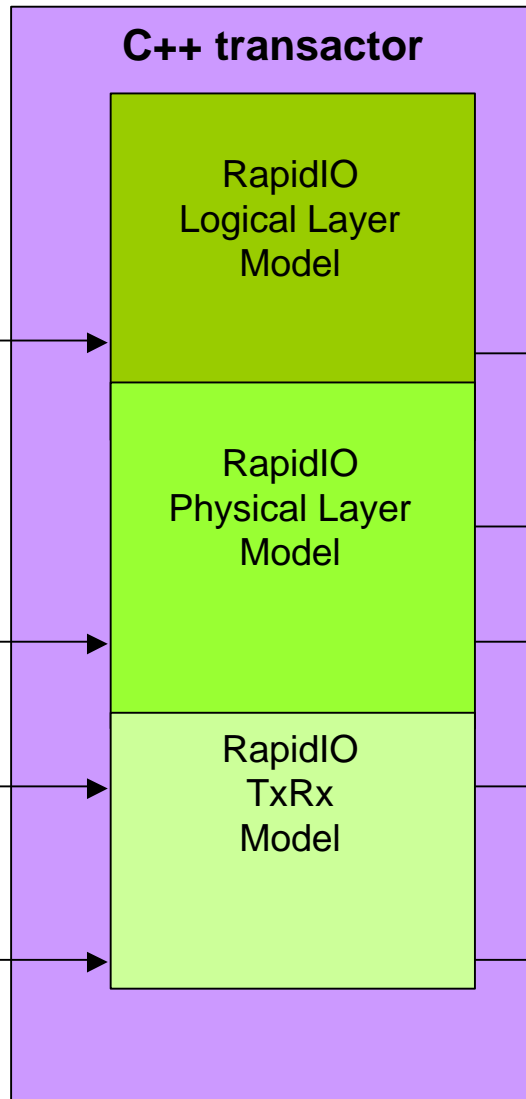
Function interface

- LL model only:*
- Maintenance requests
 - Logical IO requests
 - GSM requests
 - Message requests
 - Doorbell requests

- PL model only:*
- Maintenance requests
 - Logical IO requests
 - GSM requests
 - Message requests
 - Doorbell requests

- Common to all models:*
- Rx link (cycle accurate)

- TxRx model only:*
- Packet requests
 - Symbol request



Callback interface

- LL model only:*
- Maintenance response
 - Local device response
 - Memory response
 - Address translation response
 - GSM Snoop response
 - Messaging response
 - Doorbell response

- Common to LL and PL model:*
- Packet/Symbol send
 - Packet/Symbol received

- PL model only:*
- Request response

- Common to all models:*
- Tx link (cycle accurate)

- TxRx model only:*
- Packet responses
 - Symbol responses

- Common to all models:*
- Model information

OSCI Verification WG History

OSCI Verification WG History

- System Level Design and Verification
 - SystemC 2.0 enables genuine top down system level design
 - *events, separation between behaviour and communication, abstract interfaces*
 - Refinement is the key concept in any top down system design methodology
 - *we gradually add more detail to our model until we arrive at something we can consider to be the final implementation*

OSCI Verification WG History

- System Level Design and Verification
 - SystemC 2.0 enables genuine top down system level design
 - *events, separation between behaviour and communication, abstract interfaces*
 - Refinement is the key concept in any top down system design methodology
 - *we gradually add more detail to our model until we arrive at something we can consider to be the final implementation*
 - We must verify and re-verify as we refine our models

OSCI Verification WG History

- System Level Design and Verification
 - SystemC 2.0 enables genuine top down system level design
 - *events, separation between behaviour and communication, abstract interfaces*
 - Refinement is the key concept in any top down system design methodology
 - *we gradually add more detail to our model until we arrive at something we can consider to be the final implementation*
 - We must verify and re-verify as we refine our models
- Joint Proposal to OSCI by Fujitsu, STM, and Motorola
 - SystemC 2.0 provides a platform upon which various **design** methodologies and tools can be built
 - SystemC must also provide a platform which forms the basis for various **verification** methodologies and tools

OSCI Verification WG History

- System Level Design and Verification
 - SystemC 2.0 enables genuine top down system level design
 - *events, separation between behaviour and communication, abstract interfaces*
 - Refinement is the key concept in any top down system design methodology
 - *we gradually add more detail to our model until we arrive at something we can consider to be the final implementation*
 - We must verify and re-verify as we refine our models
- Joint Proposal to OSCI by Fujitsu, STM, and Motorola
 - SystemC 2.0 provides a platform upon which various **design** methodologies and tools can be built
 - SystemC must also provide a platform which forms the basis for various **verification** methodologies and tools
 - *Same idea arose at same time in different companies - “Great minds think alike” ☺*

Initial Investigations

Initial Investigations

- Working Group set up in Q3 2001
- What has SystemC 2.0 already got ?
 - Object Orientation
 - *required for complex protocol stacks*
 - Separation of Communication and Behaviour
 - *required for (complex hierarchies of) transactions, clean split between test, transactor and DUT*
 - Synchronisation
 - *sc_event, wait, notify, sc_mutex, sc_semaphore, sc_fifo, etc*

Initial Investigations

- Working Group set up in Q3 2001
- What has SystemC 2.0 already got ?
 - Object Orientation
 - *required for complex protocol stacks*
 - Separation of Communication and Behaviour
 - *required for (complex hierarchies of) transactions, clean split between test, transactor and DUT*
 - Synchronisation
 - *sc_event, wait, notify, sc_mutex, sc_semaphore, sc_fifo, etc*
- What will other OSCI working groups give us ?
 - Software modeling, dynamic threads
 - API's to non systemC verification environments

Initial Investigations

- Working Group set up in Q3 2001
- What has SystemC 2.0 already got ?
 - Object Orientation
 - *required for complex protocol stacks*
 - Separation of Communication and Behaviour
 - *required for (complex hierarchies of) transactions, clean split between test, transactor and DUT*
 - Synchronisation
 - *sc_event, wait, notify, sc_mutex, sc_semaphore, sc_fifo, etc*
- What will other OSCI working groups give us ?
 - Software modeling, dynamic threads
 - API's to non systemC verification environments
- What do we need to develop ourselves ?
 - Randomisation, Constraints, Weighting
 - Coverage and Assertions
 - Transactors and Transaction recording
 - Automated translation between abstraction levels

Status

Status

- Members

- EDA Companies : Cadence, Forte, Synopsys
- Systems / IP Companies : ARM, Elixent, Fujitsu, Infineon, Motorola, Philips
- Universities : Chemnitz, Tuebingen

Status

- Members

- EDA Companies : Cadence, Forte, Synopsys
- Systems / IP Companies : ARM, Elixent, Fujitsu, Infineon, Motorola, Philips
- Universities : Chemnitz, Tuebingen

- Approximate Timelines

- Phase 1 : Randomisation, Constraints, Weighting, Transactors, etc
 - *Requirements capture complete*
 - *Spec released for DAC 2002*
 - *Production Code end of 2002*

Status

- Members

- EDA Companies : Cadence, Forte, Synopsys
- Systems / IP Companies : ARM, Elixent, Fujitsu, Infineon, Motorola, Philips
- Universities : Chemnitz, Tuebingen

- Approximate Timelines

- Phase 1 : Randomisation, Constraints, Weighting, Transactors, etc
 - *Requirements capture complete*
 - *Spec released for DAC 2002*
 - *Production Code end of 2002*

Status

- Members

- EDA Companies : Cadence, Forte, Synopsys
- Systems / IP Companies : ARM, Elixent, Fujitsu, Infineon, Motorola, Philips
- Universities : Chemnitz, Tuebingen

- Approximate Timelines

- Phase 1 : Randomisation, Constraints, Weighting, Transactors, etc
 - *Requirements capture complete*
 - *Spec released for DAC 2002*
 - *Production Code end of 2002*
- Phase 2 : Assertions and Coverage
 - *requirements capture complete H2 2002*

Status

- Members

- EDA Companies : Cadence, Forte, Synopsys
- Systems / IP Companies : ARM, Elixent, Fujitsu, Infineon, Motorola, Philips
- Universities : Chemnitz, Tuebingen

- Approximate Timelines

- Phase 1 : Randomisation, Constraints, Weighting, Transactors, etc
 - *Requirements capture complete*
 - *Spec released for DAC 2002*
 - *Production Code end of 2002*
- Phase 2 : Assertions and Coverage
 - *requirements capture complete H2 2002*
- *All dates are for guidance only !*

Status

- Members

- EDA Companies : Cadence, Forte, Synopsys
- Systems / IP Companies : ARM, Elixent, Fujitsu, Infineon, Motorola, Philips
- Universities : Chemnitz, Tuebingen

- Approximate Timelines

- Phase 1 : Randomisation, Constraints, Weighting, Transactors, etc
 - *Requirements capture complete*
 - *Spec released for DAC 2002*
 - *Production Code end of 2002*

- Phase 2 : Assertions and Coverage
 - *requirements capture complete H2 2002*

- *All dates are for guidance only !*

- Verification WG will deliver high quality code

- Reuse successful systemC 2.0 process
- Significant experience of C++ verification tools in Working Group
 - *testbuilder (Cadence) , QuickBench (Forte) , SystemC-SV Library (Chemnitz)*

Why SystemC ?

Why SystemC ?

- Is SystemC optimised for every task ?

Why SystemC ?

- Is SystemC optimised for every task ?
 - SystemC may not be the best language for hardware implementation

Why SystemC ?

- Is SystemC optimised for every task ?
 - SystemC may not be the best language for hardware implementation
 - SystemC may not be the best language for verification

Why SystemC ?

- Is SystemC optimised for every task ?
 - SystemC may not be the best language for hardware implementation
 - SystemC may not be the best language for verification
 - SystemC may not even be the best language for some specialized aspects of system level design

Why SystemC ?

- Is SystemC optimised for every task ?
 - SystemC may not be the best language for hardware implementation
 - SystemC may not be the best language for verification
 - SystemC may not even be the best language for some specialized aspects of system level design
- SystemC is the **only** language that can do **all** of these tasks **well**

Why SystemC ?

- Is SystemC optimised for every task ?
 - SystemC may not be the best language for hardware implementation
 - SystemC may not be the best language for verification
 - SystemC may not even be the best language for some specialized aspects of system level design
- SystemC is the **only** language that can do **all** of these tasks **well**
 - SystemC is the **best** language for step by step **refinement** of models and testbenches