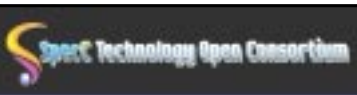
9th IEEE/DATC Electronic Design Processes Workshop, EDP 2002

SpecC Methodology for High-Level Modeling

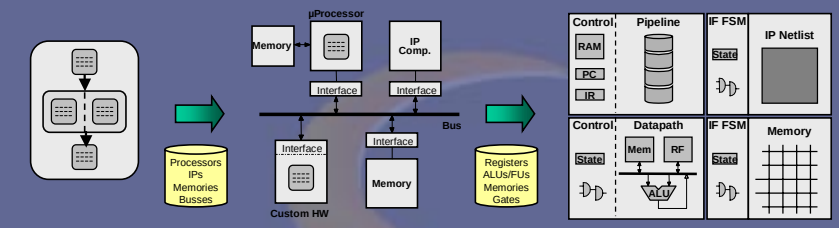
Rainer Dömer
Center for Embedded Computer Systems
University of California, Irvine
http://www.cecs.uci.edu/~specc/

Outline



- System design
- SpecC design methodology
- Specification model
- Architecture model
- Communication model
- Implementation model
- Summary & conclusions

System-On-Chip Design

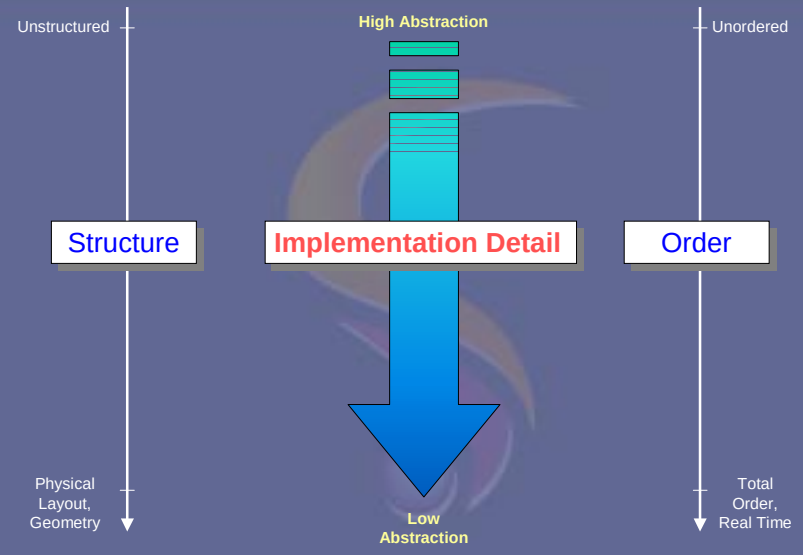


Specification + constraints | **System architecture + estimates** | **RTL/IS Implementation + results**

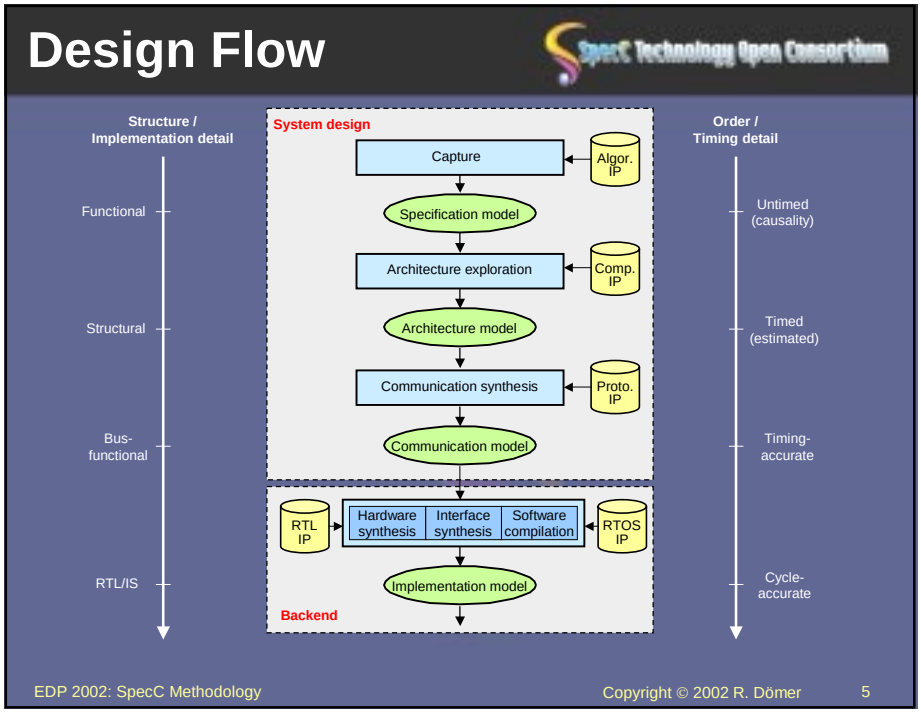
- Specification to architecture to implementation
- Behavior to structure
  1. System level: system specification to system architecture
  2. RT/IS level: component behavior to component microarchitecture

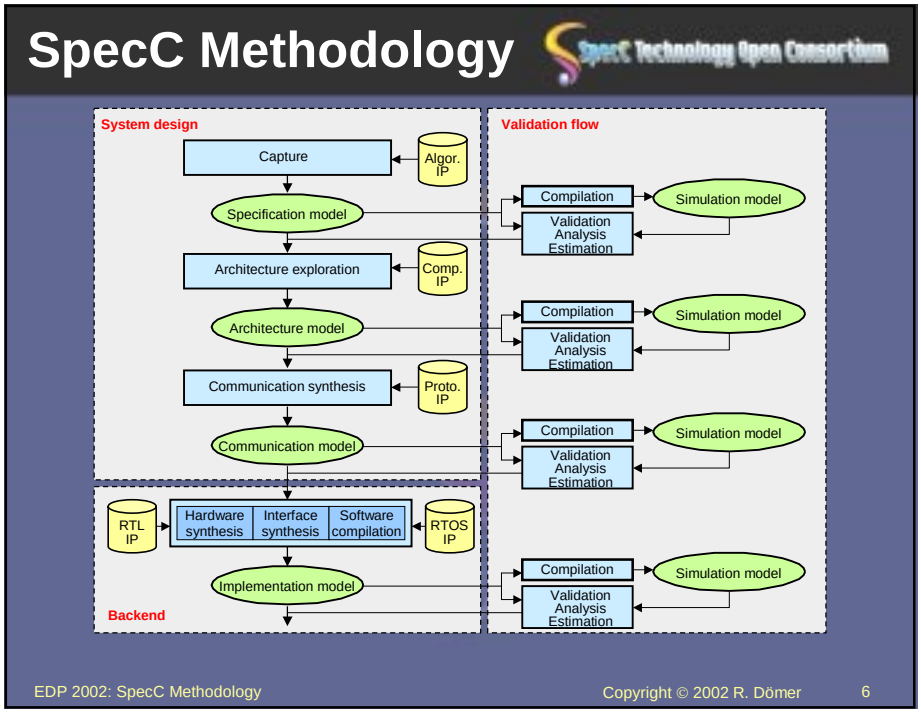
EDP 2002: SpecC Methodology Copyright © 2002 R. Dömer 3

Abstraction Levels



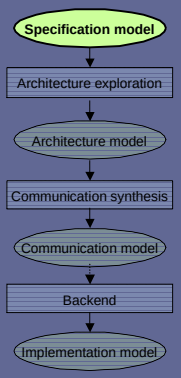
EDP 2002: SpecC Methodology Copyright © 2002 R. Dömer 4


Design Flow
Structure / Implementation detail
Functional
Structural
Bus-functional
RTL/IS
System design
Capture
Algor. IP
Specification model
Architecture exploration
Comp. IP
Architecture model
Communication synthesis
Proto. IP
Communication model
RTL IP
Hardware synthesis
Interface synthesis
Software compilation
RTOS IP
Implementation model
Backend
Order / Timing detail
Untimed (causality)
Timed (estimated)
Timing-accurate
Cycle-accurate
EDP 2002: SpecC Methodology
Copyright © 2002 R. Dömer
5

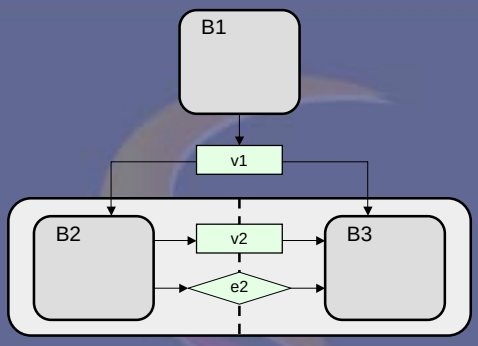

SpecC Methodology
System design
Capture
Algor. IP
Specification model
Architecture exploration
Comp. IP
Architecture model
Communication synthesis
Proto. IP
Communication model
RTL IP
Hardware synthesis
Interface synthesis
Software compilation
RTOS IP
Implementation model
Backend
Validation flow
Compilation
Validation Analysis Estimation
Simulation model
EDP 2002: SpecC Methodology
Copyright © 2002 R. Dömer
6

Specification Model

- **High-level, abstract model**
  - Pure system functionality
  - Algorithmic behavior
  - No implementation details
- **No implicit structure / architecture**
  - Behavioral hierarchy
- **Untimed**
  - Executes in zero (logical) time
  - Causal ordering
  - Events only for synchronization



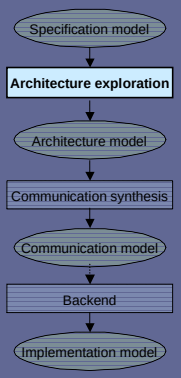
Specification Model Example



- **Simple, typical specification model**
  - Hierarchical parallel-serial composition
  - Communication through ports and variables, events

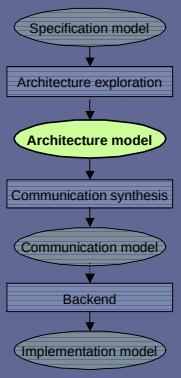
Architecture Exploration

- **Component allocation / selection**
- **Behavior partitioning**
- **Variable partitioning**
- **Scheduling**

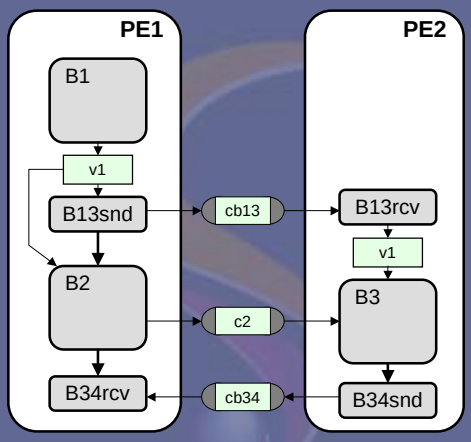


Architecture Model

- **Component structure/architecture**
  - Top level of behavior hierarchy
- **Behavioral/functional component view**
  - Behaviors grouped under top-level component behaviors
  - Sequential behavior execution
- **Timed**
  - Estimated execution delays



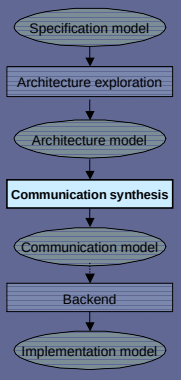
Architecture Model Example



EDP 2002: SpecC Methodology Copyright © 2002 R. Dömer 11

Communication Synthesis

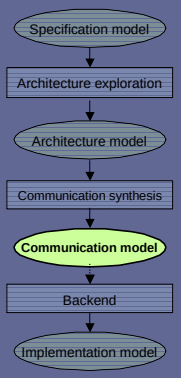
- **Bus allocation / protocol selection**
- **Channel partitioning**
- **Protocol, transducer insertion**
- **Inlining**



EDP 2002: SpecC Methodology Copyright © 2002 R. Dömer 12

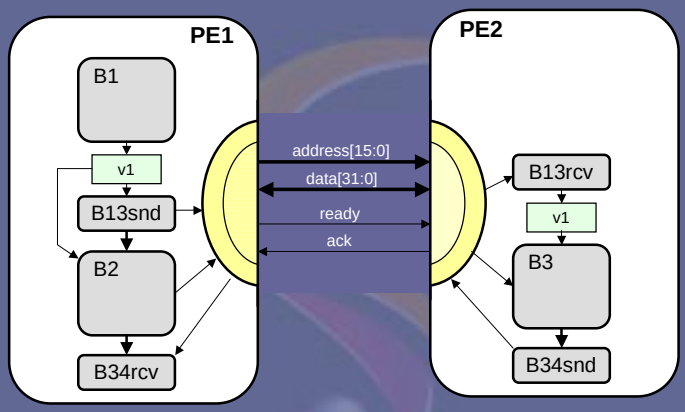
Communication Model

- **Component & bus structure/architecture**
  - Top level of hierarchy
- **Bus-functional component models**
  - Timing-accurate bus protocols
  - Behavioral component description
- **Timed**
  - Estimated component delays



EDP 2002: SpecC Methodology Copyright © 2002 R. Dömer 13

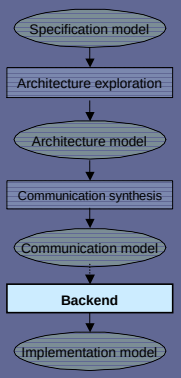
Communication Model Example



EDP 2002: SpecC Methodology Copyright © 2002 R. Dömer 14

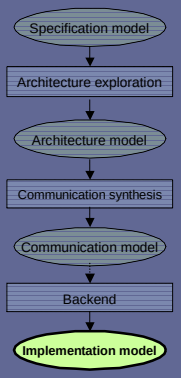
Backend

- **Clock-accurate implementation of PEs**
  - Hardware synthesis
  - Software development
  - Interface synthesis

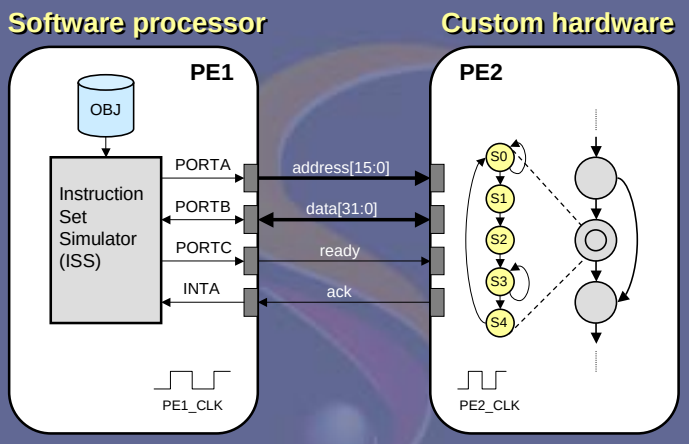


Implementation Model

- **Cycle-accurate system description**
  - RTL description of hardware
    - Behavioral/structural FSMD view
  - Object code for processors
    - Instruction-set co-simulation
  - Clocked bus communication
    - Bus interface timing based on PE clock



Implementation Model Example



EDP 2002: SpecC Methodology Copyright © 2002 R. Dömer 17

Summary and Conclusions

- **SpecC system-level design methodology**
  - Specification to RTL
    - System synthesis
    - Backend
  - Four levels of abstraction
    - Specification model: untimed, functional
    - Architecture model: estimated, structural
    - Communication model: timed, bus-functional
    - Implementation model: cycle-accurate, RTL/IS
  - Well-defined, formal models & transformations
    - Automatic, gradual refinement
    - Executable models, testbench re-use
    - Simple verification
  - Customizable
    - Easy integration with existing design flow

EDP 2002: SpecC Methodology Copyright © 2002 R. Dömer 18

Further Information

- **Literature**
  - "*SpecC: Specification Language and Methodology*" by Gajski, Zhu, Dömer, Gerstlauer, Zhao, Kluwer Academic Publishers, 2000.
  - "*System Design: A Practical Guide with SpecC*" by Gerstlauer, Dömer, Peng, Gajski, Kluwer Academic Publishers, 2001.
  - "*System-level Modeling and Design with the SpecC Language*", Ph.D. Thesis R. Dömer, University of Dortmund, 2000.
- **Online**
  - SpecC web pages at UCI **`http://www.cecs.uci.edu/~specc/`**
  - SpecC Open Technology Consortium (STOC) **`http://www.specc.org/`**

EDP 2002: SpecC Methodology 19

Backup Slides

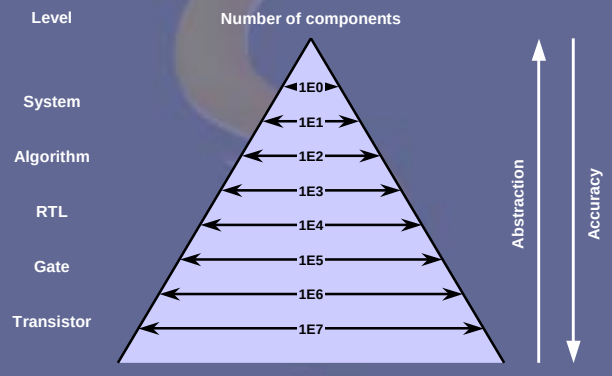
SpecC Technology Open Consortium

- Abstraction Levels
- SpecC Model
- Plug-and-Play

Introduction

SpecC Technology Open Consortium

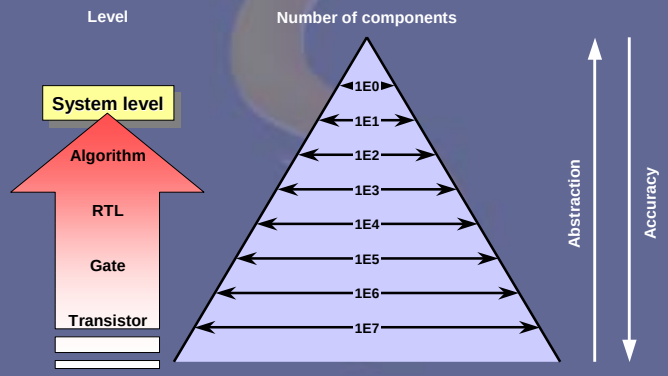
- System-on-Chip (SOC) design
- Increase of design complexity



EDP 2002: SpecC Methodology Copyright © 2002 R. Dömer 22

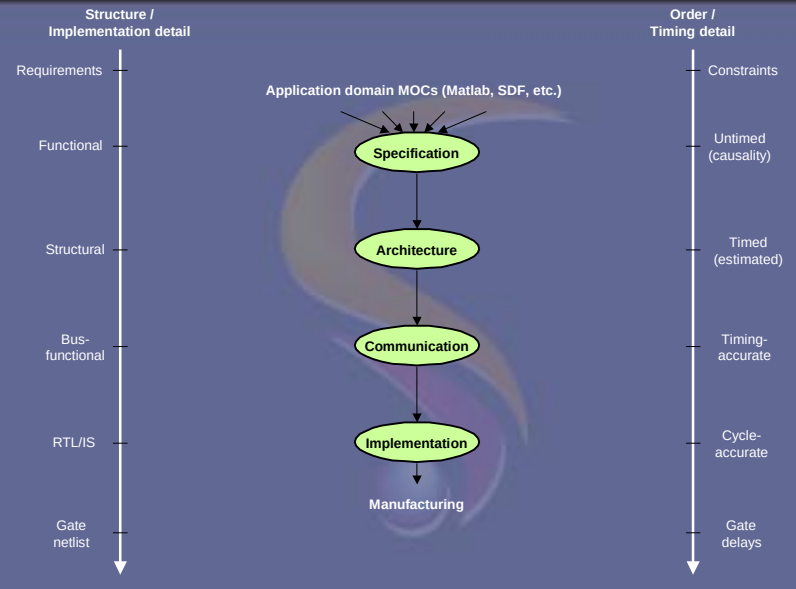
Introduction

- System-on-Chip (SOC) design
- Increase of design complexity
- Move to higher levels of abstraction



EDP 2002: SpecC Methodology Copyright © 2002 R. Dömer 23

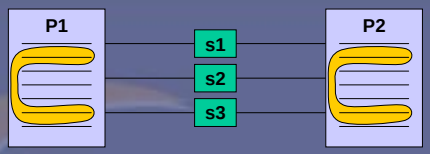
Abstraction Levels



EDP 2002: SpecC Methodology Copyright © 2002 R. Dömer 24

The SpecC Model

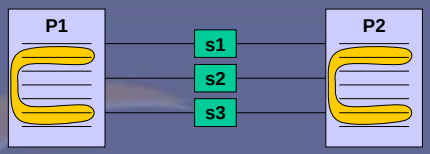
- **Traditional model**



  - Processes and signals
  - Mixture of computation and communication
  - Automatic replacement impossible

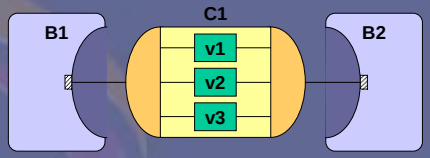
The SpecC Model

- **Traditional model**



  - Processes and signals
  - Mixture of computation and communication
  - Automatic replacement impossible

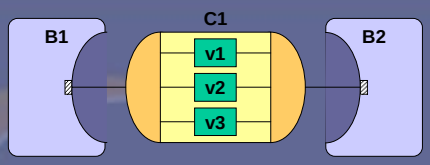
- **SpecC model**



  - Behaviors and channels
  - Separation of computation and communication
  - Plug-and-play

The SpecC Model: Protocol Inlining

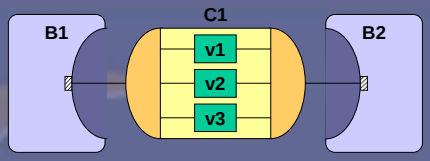
- **Specification model**
- **Exploration model**



  - Computation in behaviors
  - Communication in channels

The SpecC Model: Protocol Inlining

- **Specification model**
- **Exploration model**



  - Computation in behaviors
  - Communication in channels

- **Implementation model**

  - Channel disappears
  - Communication inlined into behaviors
  - Wires exposed

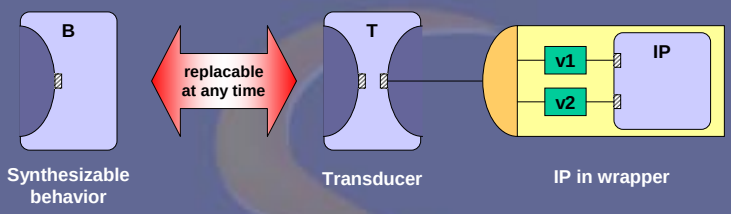
The SpecC Model: Plug-and-Play

- **Computation IP: Wrapper model**

EDP 2002: SpecC Methodology Copyright © 2002 R. Dömer 29

The SpecC Model: Plug-and-Play

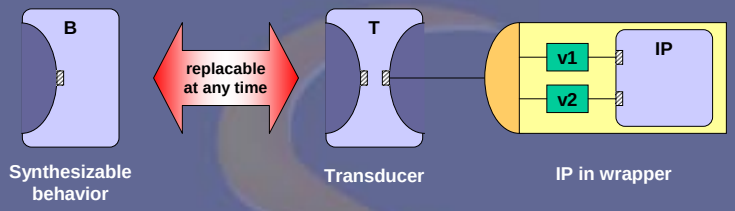
- **Computation IP: Wrapper model**



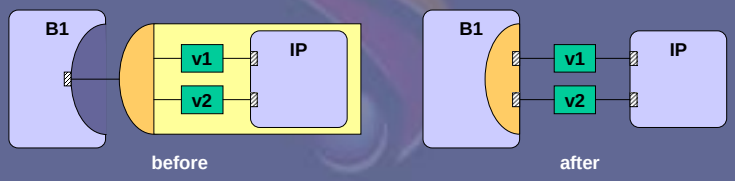
EDP 2002: SpecC Methodology Copyright © 2002 R. Dömer 30

The SpecC Model: Plug-and-Play

- **Computation IP: Wrapper model**



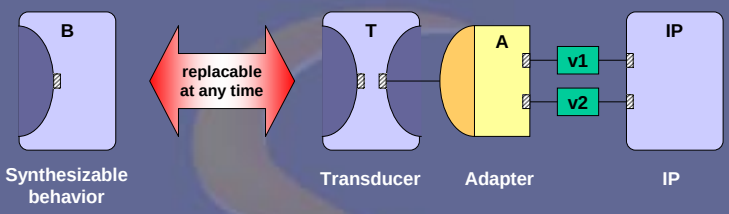
- **Protocol inlining with wrapper**



EDP 2002: SpecC Methodology Copyright © 2002 R. Dömer 31

The SpecC Model: Plug-and-Play

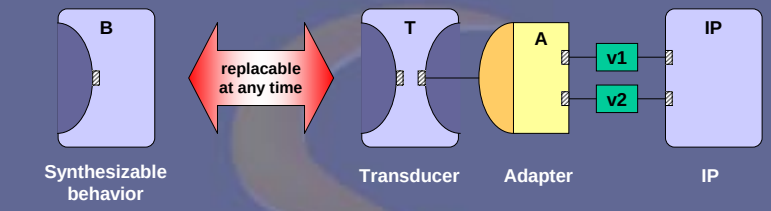
- **Computation IP: Adapter model**



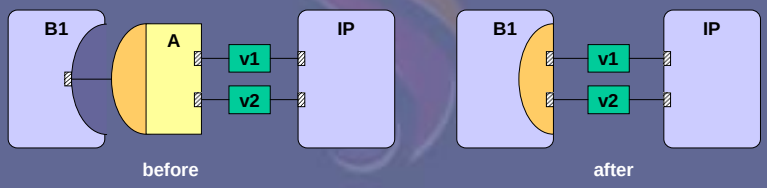
EDP 2002: SpecC Methodology Copyright © 2002 R. Dömer 32

The SpecC Model: Plug-and-Play

- **Computation IP: Adapter model**



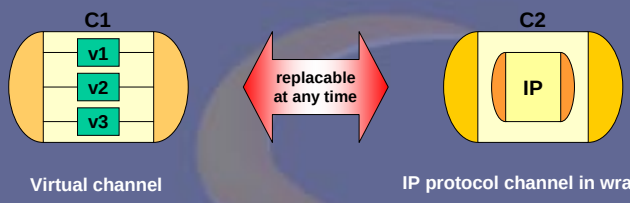
- **Protocol inlining with adapter**



EDP 2002: SpecC Methodology Copyright © 2002 R. Dömer 33

The SpecC Model: Plug-and-Play

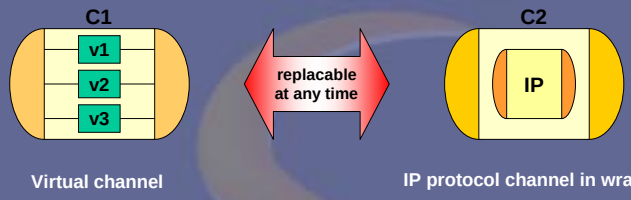
- **Communication IP: Channel with wrapper**



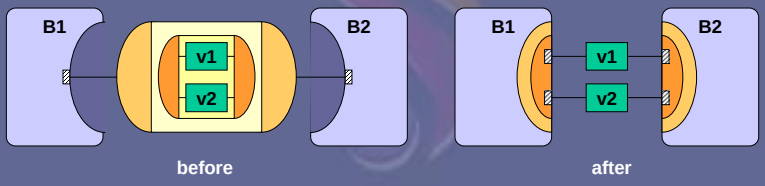
EDP 2002: SpecC Methodology Copyright © 2002 R. Dömer 34

The SpecC Model: Plug-and-Play

- **Communication IP: Channel with wrapper**



- **Protocol inlining with hierarchical channel**



EDP 2002: SpecC Methodology Copyright © 2002 R. Dömer 35

Summary

- **SpecC model**
  - PSM model of computation
  - Separation of communication and computation
  - Hierarchical network of behaviors and channels
  - Plug-and-play
- **SpecC language**
  - True superset of ANSI-C
    - ANSI-C plus extensions for HW-design
  - Support of all concepts needed in system design
    - Structural and behavioral hierarchy
    - Concurrency
    - State transitions
    - Communication
    - Synchronization
    - Exception handling
    - Timing

EDP 2002: SpecC Methodology Copyright © 2002 R. Dömer 36

Conclusion

- **SpecC language**
  - Executable and synthesizable
  - Precise coverage of system language requirements
  - Orthogonal constructs for orthogonal concepts
- **Impact**
  - Adoption of SpecC in industry and academia
  - SpecC Open Technology Consortium (STOC)
- **Future**
  - Standardization effort in progress by STOC
  - Improvement with your participation

Conclusion

- **Using the SpecC language and the advanced architecture exploration capabilities of the SpecC methodology, architecture exploration is quick and easy to implement in the abstract level.**
- **Reuse can be done from the component level to the system level.**
- **Productivity gain from working at higher levels of abstraction.**