Multi-Domain Simulation

Mark Glasser

NVIDIA Corporation

Multiple Domains

- Languages – Verilog/SystgemVerilog, C/C++/SystemC
- Abstractions – RTL, Transaction-level, Mixed Signal
- Media – simulators, emulators, FPGA boards, silicon
- Hierarchy – unit, cluster, system

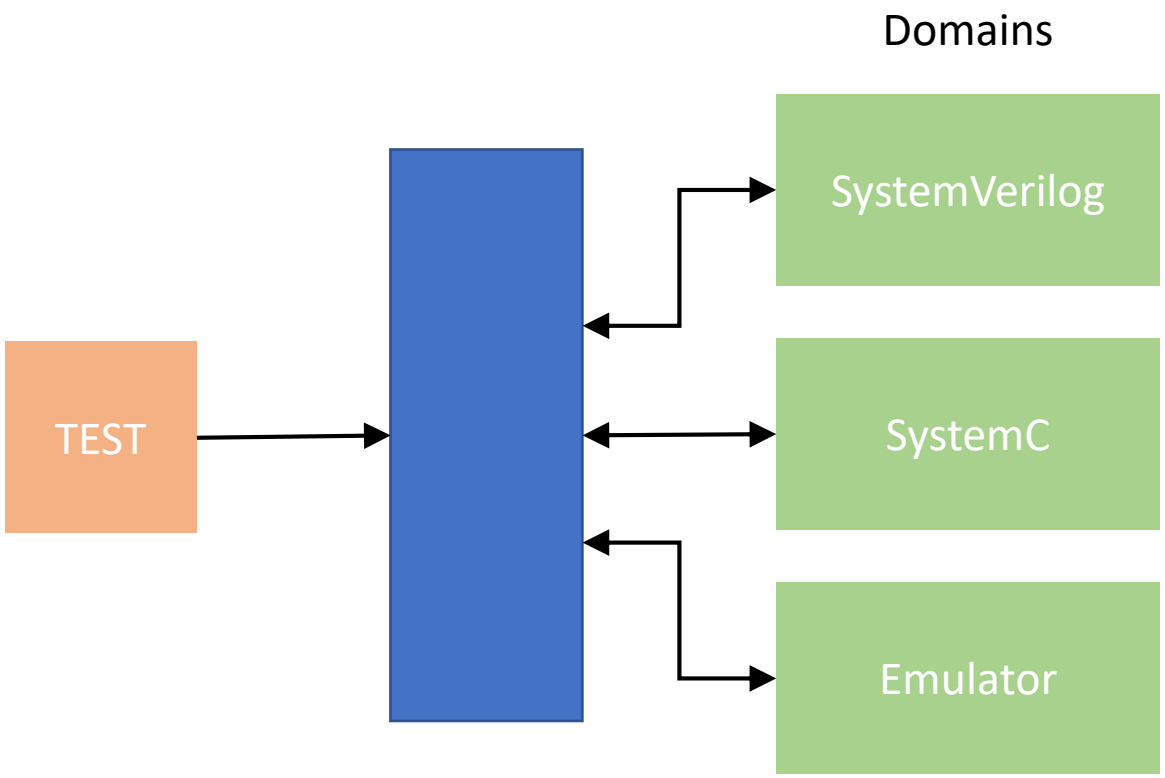
Assembling Large SoC Simulation

- Availability of Models
  - Different abstractions, different languages
  - Available at different times during design cycle
- Test portability across languages, abstractions, and media
- Test portability across hierarchy
- Creating a functioning environment for a block or cluster
- Global address map
- Application stack

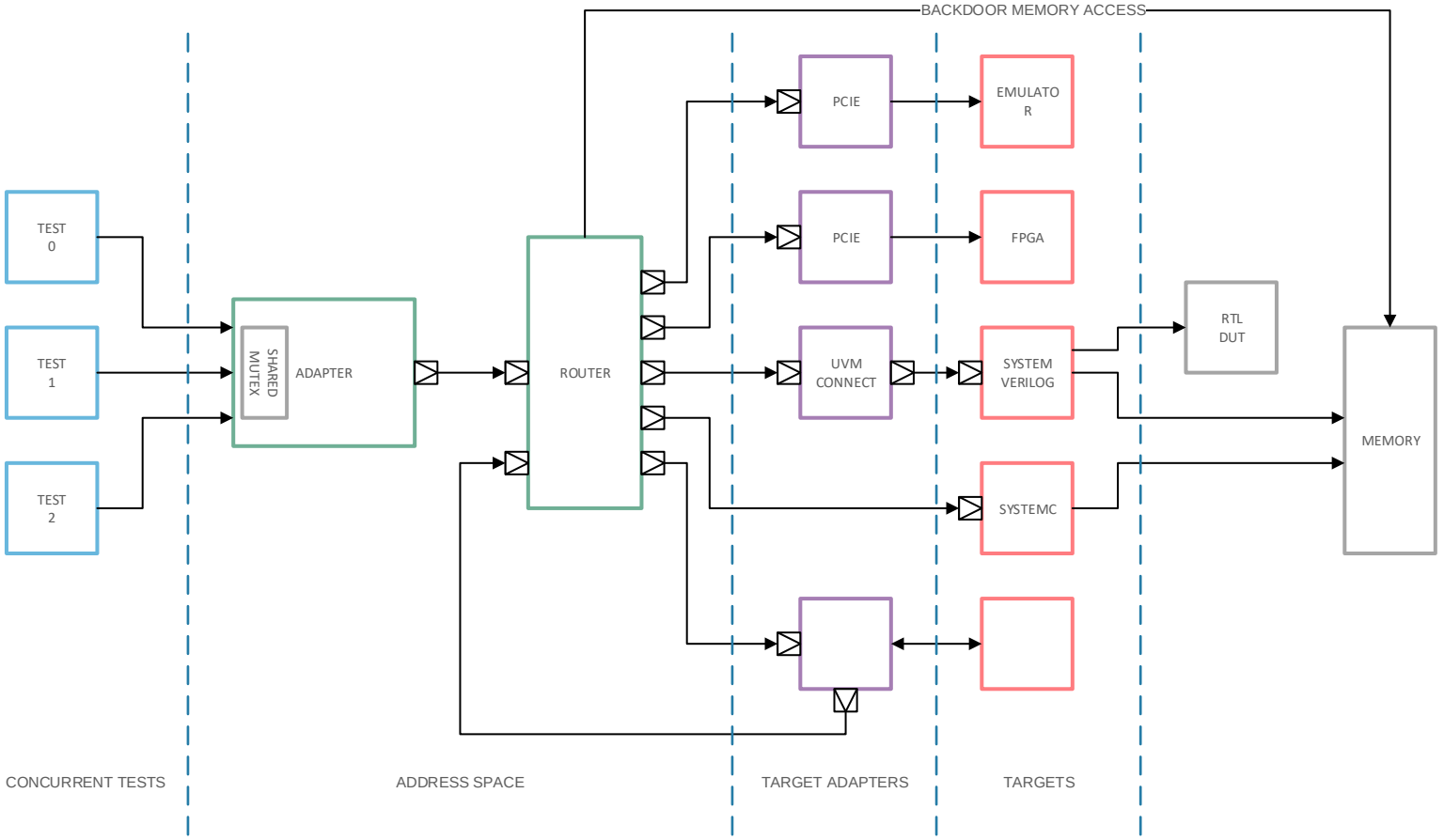
Connect All Those Things Together

- Mix-and-match languages
  - C++, SystemC, SystemVerilog, UVM
- Mix-and-match abstraction levels
  - RTL & TL
- Mix-and-match simulation media
  - Logic simulator, Emulator, FPGA, Silicon
- Etc.

Testing with Multiple Domains



Architecture (canonical topology)



Architectural Foundation

- TLM2 is the medium for constructing topologies
- TLM2 GP is the *lingua franca*
- Share everything between SystemVerilog/UVM and SystemC/C++
- Accommodate existing suites of C/C++-based tests
  - Includes large chunks of application software stack
- UVM Connect to bind SystemC and SytemVerilog/UVM
- Scalable

UVM Connect

- Bind TLM connections between SystemC and SystemVerilog/UVM
- Handles data conversions
- Appears like and ordinary socket on either side
- Open source tool created by Mentor Graphics

Adapter/Test API

FRONT DOOR	BACKDOOR	MISC
Read8	BdRead8	Barrier
Read16	BdRead16	
Read32	BdRead32	
Read64	BdRead64	
Write8	BdWrite8	
Write16	BdWrite16	
Write32	BdWrite32	
Write64	BdWrite64	
BlockRead	BdBlockRead	
BlockWrite	BdBlockWrite	

Adapter + Router

- Owns the address map
- Forms GPs from test API
- Sends transactions to the downstream target

Target Adapters

- Bind targets to fabric
- Bidirectional coversion (as appropriate) between GPs and downstream target
- No limit to number of targets

Features

Shared Memory Model

- Configurable from 8-64 bit address space
- Shared between SystemVerilog and C++
  - Write via SystemC target can be read by SystemVerilog target
  - Vice Versa
- Backdoor and front door paths

Shared Configuration Data

- Var<t> supports any data type
- Centralized database for vars
- Data can be shared between scopes
- Scalar types are automatically shared to UVM resource database

Master/Slave phasing

- UVM is phasing master – if present
- Adapter is phasing master – if UVM is not present – slave otherwise
- Identical phasing semantics – with or without UVM

Interrupts and Resets

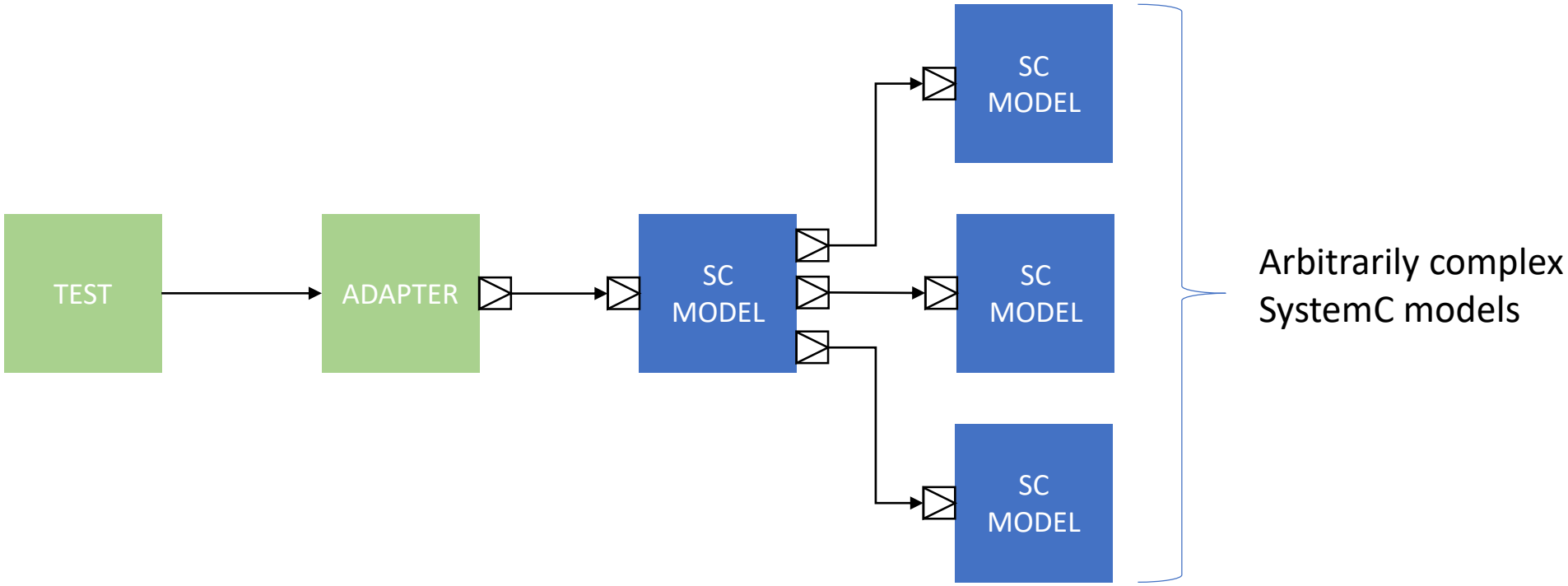
- Interrupt
  - Transfers control to interrupt handler
  - Returns to test upon completion of interrupt handler
- Reset
  - Restarts phasing
  - Both C Test and UVM

Custom Command Line Parser

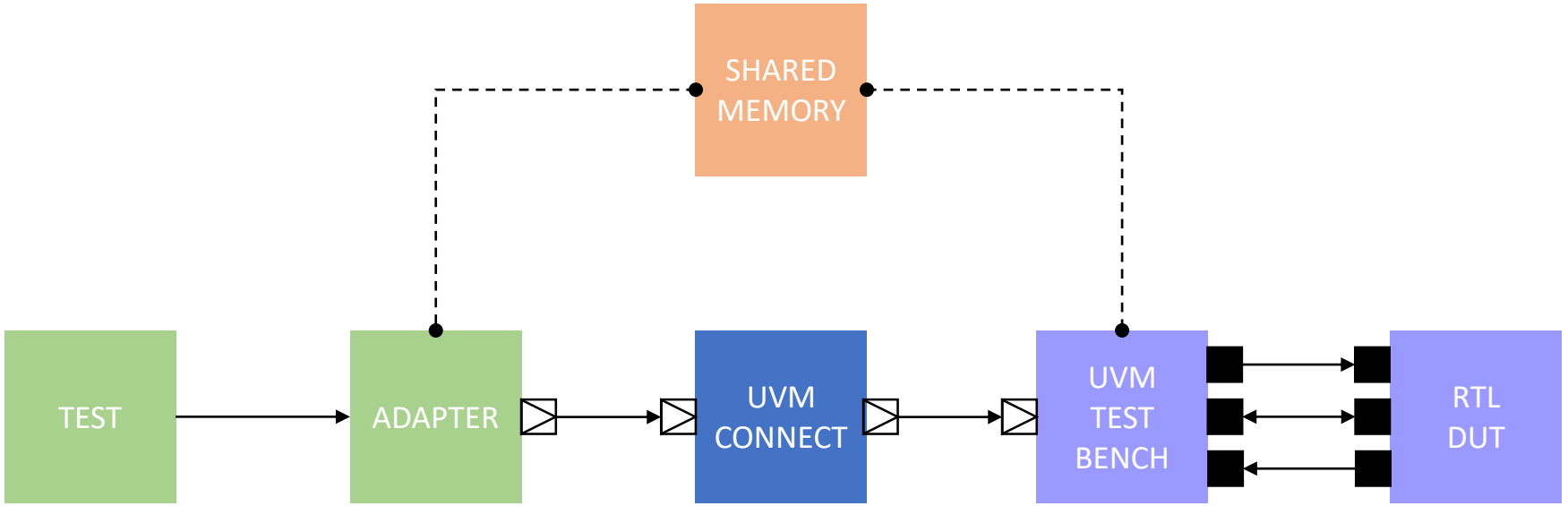
- Command lines are important part of controlling command lines
- Can build custom command line “language”
- Can automatically store options in var<t> database

Use Models

TL Model Validation

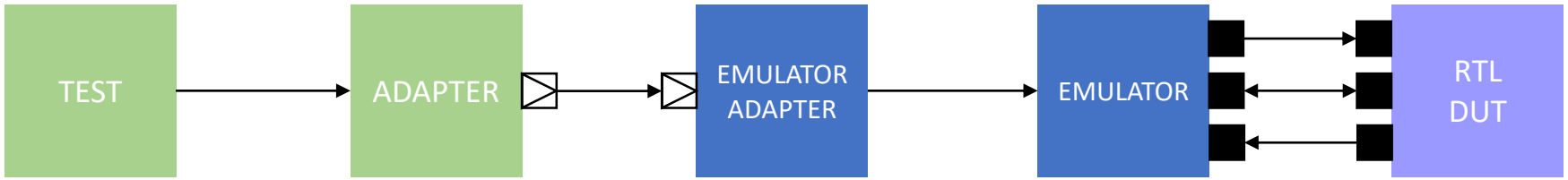


RTL Testbench

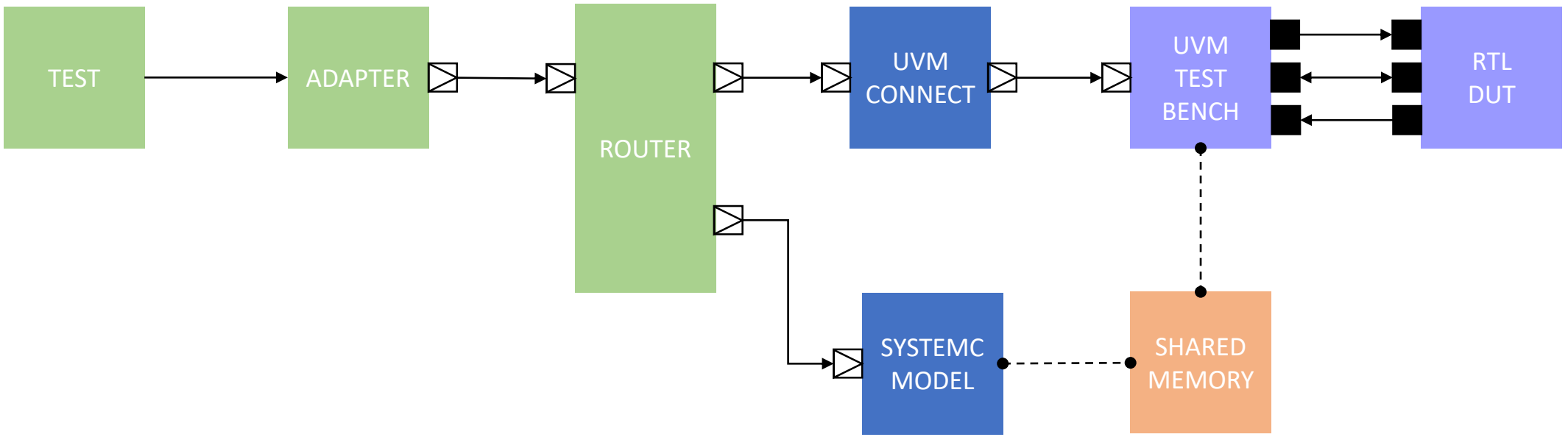


Backdoor and front door access to memory

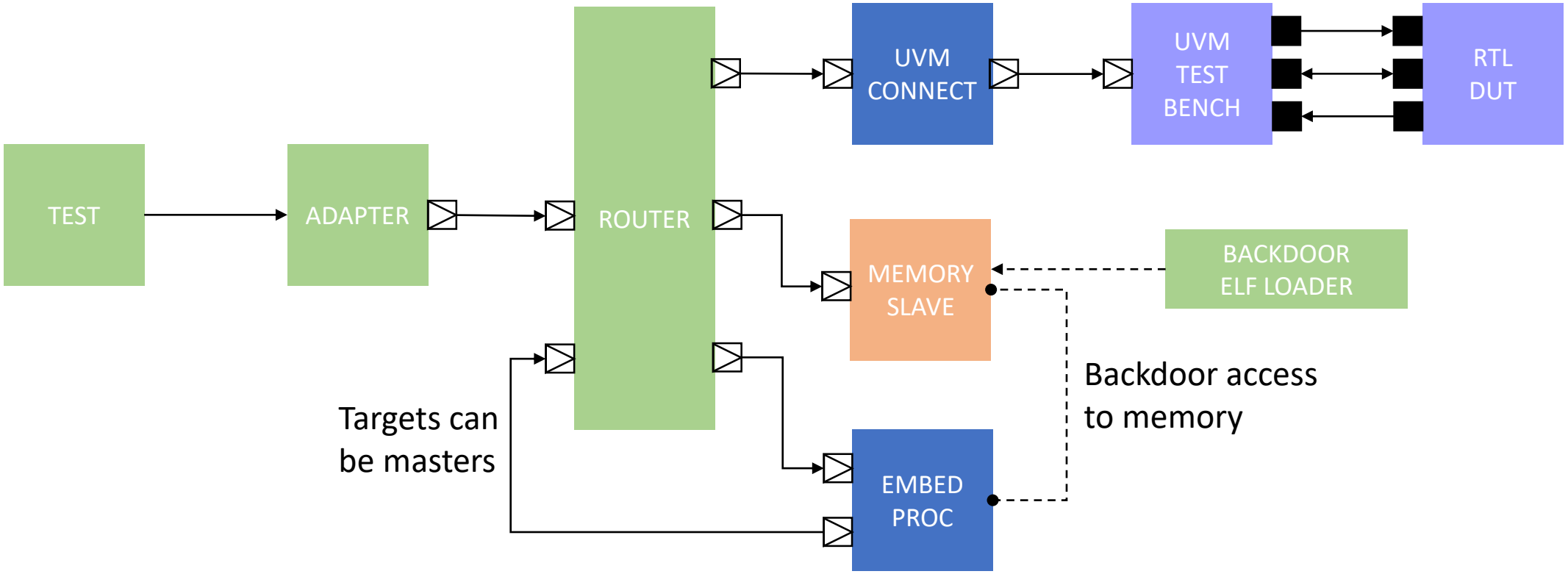
Emulator



Combine SystemC & SystemVerilog/UVM



Embedded Processor & Shared Memory



More Use Models…

- Many topologies are possible
- SystemC + Emulator and/or FPGA
- Unit-level FPGA + RTL + embedded processor

- Endless combinations:
  - SystemC/TLM
  - RTL
  - Emulator
  - FPGA
  - Embedded Processor

Test Portability

- Test harness is identical across abstractions and language
- Test semantics are identical across abstractions and language

Other Possible Solutions

- Portable Stimulus
  - PSS is yet another language
  - Does not fit in with existing suite of C-based tests and software stack
  - Does not support multiple simultaneous domains
- Multi-Language Working Group
  - Backplane architecture
  - Too generic – attempts to accommodate every kind of logic simulation engine
  - No longer active(?)

Pragmatic Approach

- Only need to support SystemC and SystemVerilog engines/languages
- No expectations for commercially available tools that do not already exist
- Must fit in with C/C++-based software stack and existing test suites

Conclusion

- Higly flexible, scalable system for mutli-domain simulation
- Provides test portability across abstractions and across languages
- Provides a foundation for driving emulators and FPGA boards
- By sharing as much as possible between SystemC and SystemVerilog we blur the lines between languages and abstractions
- Enable complex simulations in an SoC design environment

THANK YOU!