Pragmatic Verification Reuse in a Vertical World

Mark Litterick, Verilab, Germany

DVCon 2013 Design & Verification Conference & Exhibition

Sponsored By: accellera SYSTEMS INITIATIVE

Consistent effective verification reuse is not being achieved!

- why?

Top-Level Verification Requirements

- correct overall **application**
- **interaction** of all modules and sub-systems
- access to **shared resources**
- operation with realistic **clock domains**
- **low-power** operation and power domains
- overall **performance** of the system
- **connectivity** of all blocks and sub-systems
- ...

No closure on VR from external interfaces alone

- internal checks & **coverage**
- **block-level reuse** effective

Real-world not conform to bottom-up reuse paradigm

- **different** requirements
- top-level **critical path**
- **ad-hoc** block-levels
- **parallel** development

Validation of perfect chip is not enough

- all chips have **bugs**!
- top **debug time** is huge
- **internal** checks & messages **help debug**

Tape-out top chip + **Vertical reuse** = LESS ⌛ + LESS 🎲 + MORE $

Active/Passive Misuse

- low-level VCs are OK but **environment ignores active/passive**
- drivers post **sequence items** to **scoreboard**
- **drivers** doing functional **timeout checks**
- **coverage** defined in **active** components
- **configuration updates** from sequence or driver
- important **messages** from drivers
- **error injection** traffic reported as a **warning**
- **passive** components controlling **end of test** schedule
- ...

Active/Passive Guidelines

- **complete environment** must consider **active/passive**
- do not connect scoreboards to active components
- perform functional checks in passive components
- collect functional coverage in passive components
- **update configuration** only from **passive** components
- generate **important messages** in **passive** components
- **promote warnings to errors** in passive mode
- do **not** control **end of test** from components in passive mode
- ...

Problems of Scale

- verification environment pulled together only in **base-test**
- expect top-level to **configure** environment again
- provide **multiple interfaces** to top-level
- **macro definitions** having global scope
- **enumeration literals** and types without package prefix
- ...

Scale Guidelines

- **encapsulate** all sub-components in an **environment**
- **encapsulate configuration** objects
- combine multiple interfaces into **hierarchical interface**
- **encapsulate SVA** protocol checks inside interface
- **avoid namespace collisions** by using prefix per scope
- ...

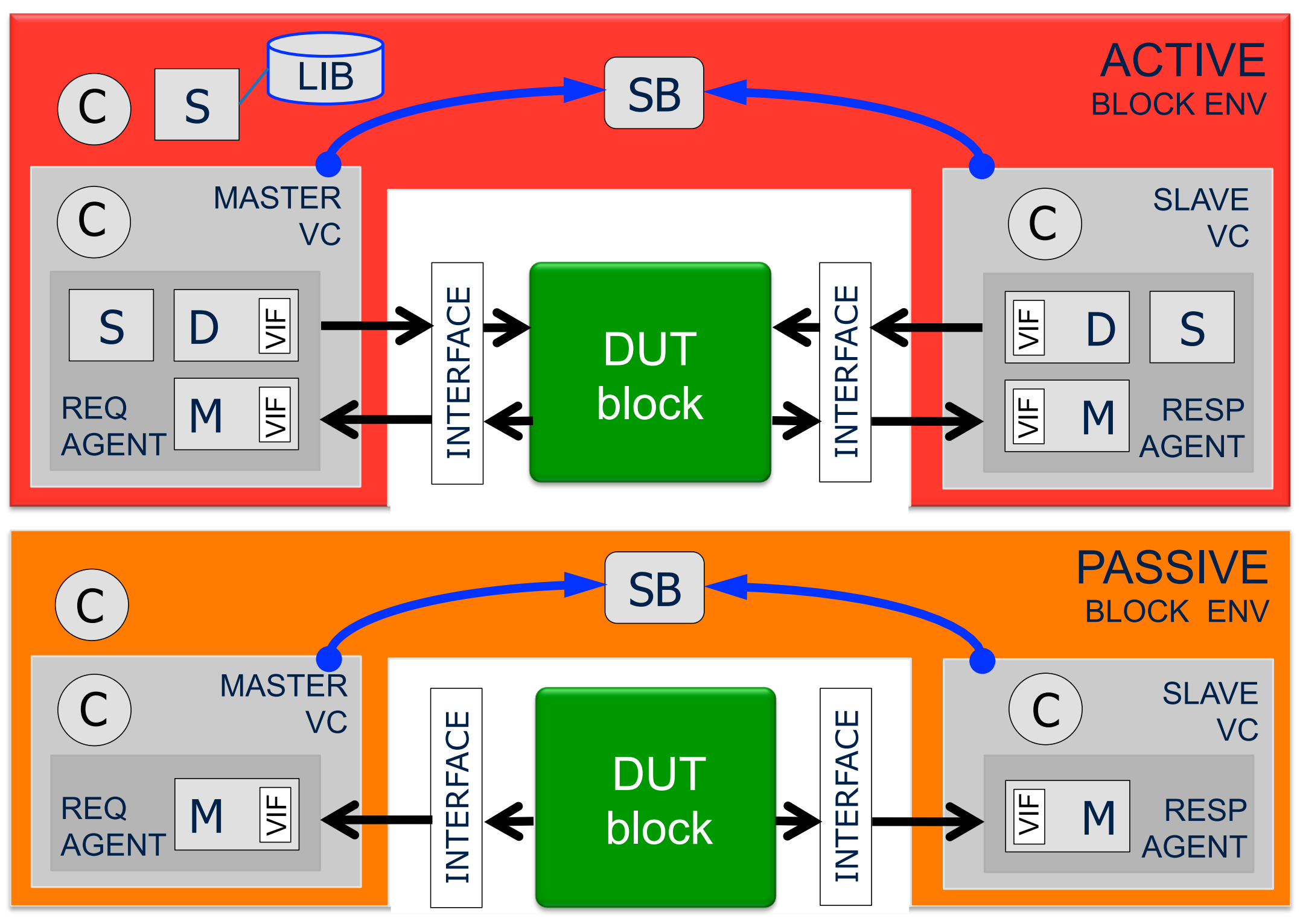
Additional Concerns

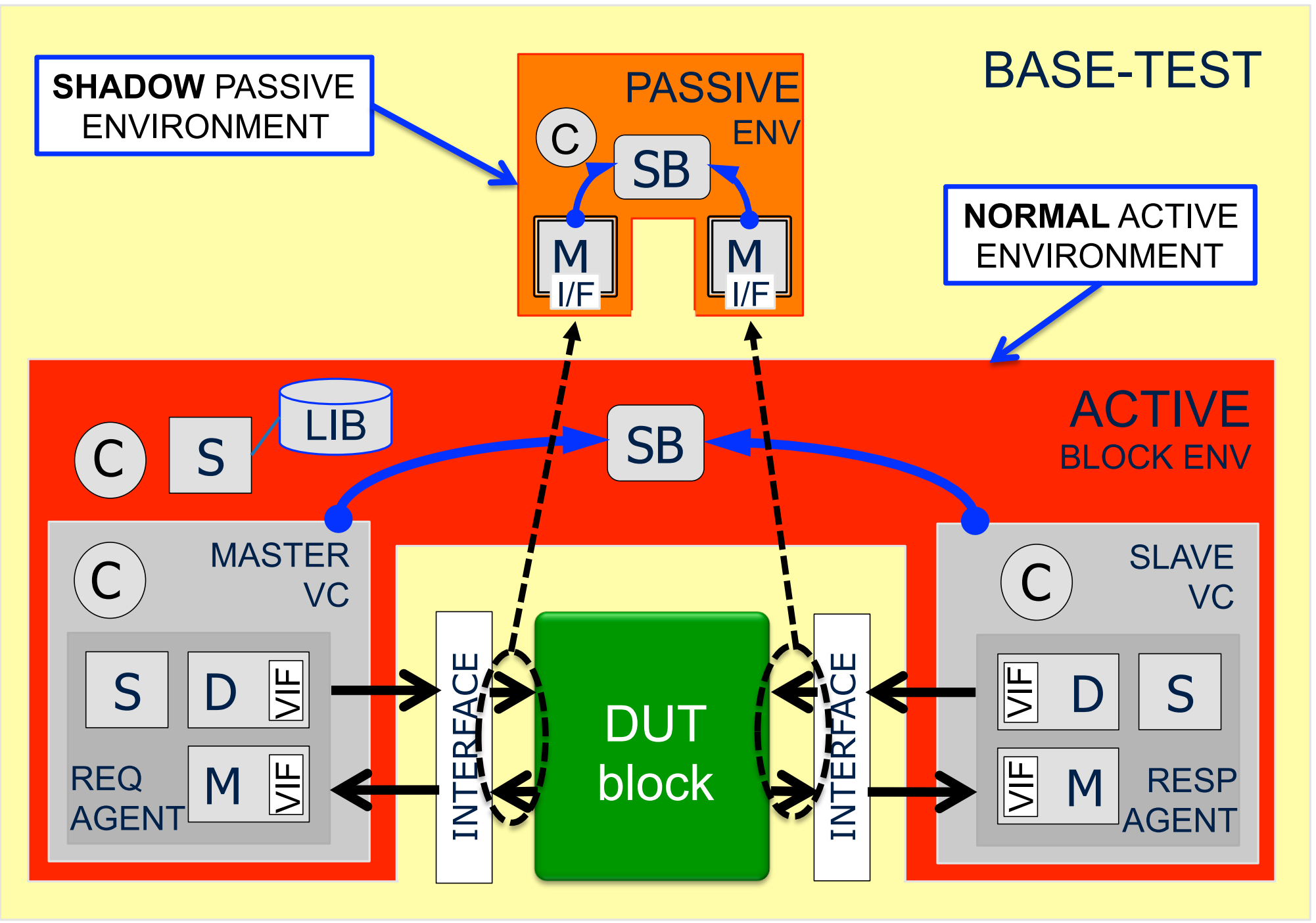
- **formal** verification carried out at block-level
- **power-aware** simulations introduced at top-level
- inefficient **design assertions** generate performance bottleneck
- **CDC operation** not fully explored at block-level
- **CDC waivers** that prove to be invalid at top-level
- uncontrollable or **inappropriate AMS assertion** performance
- ...

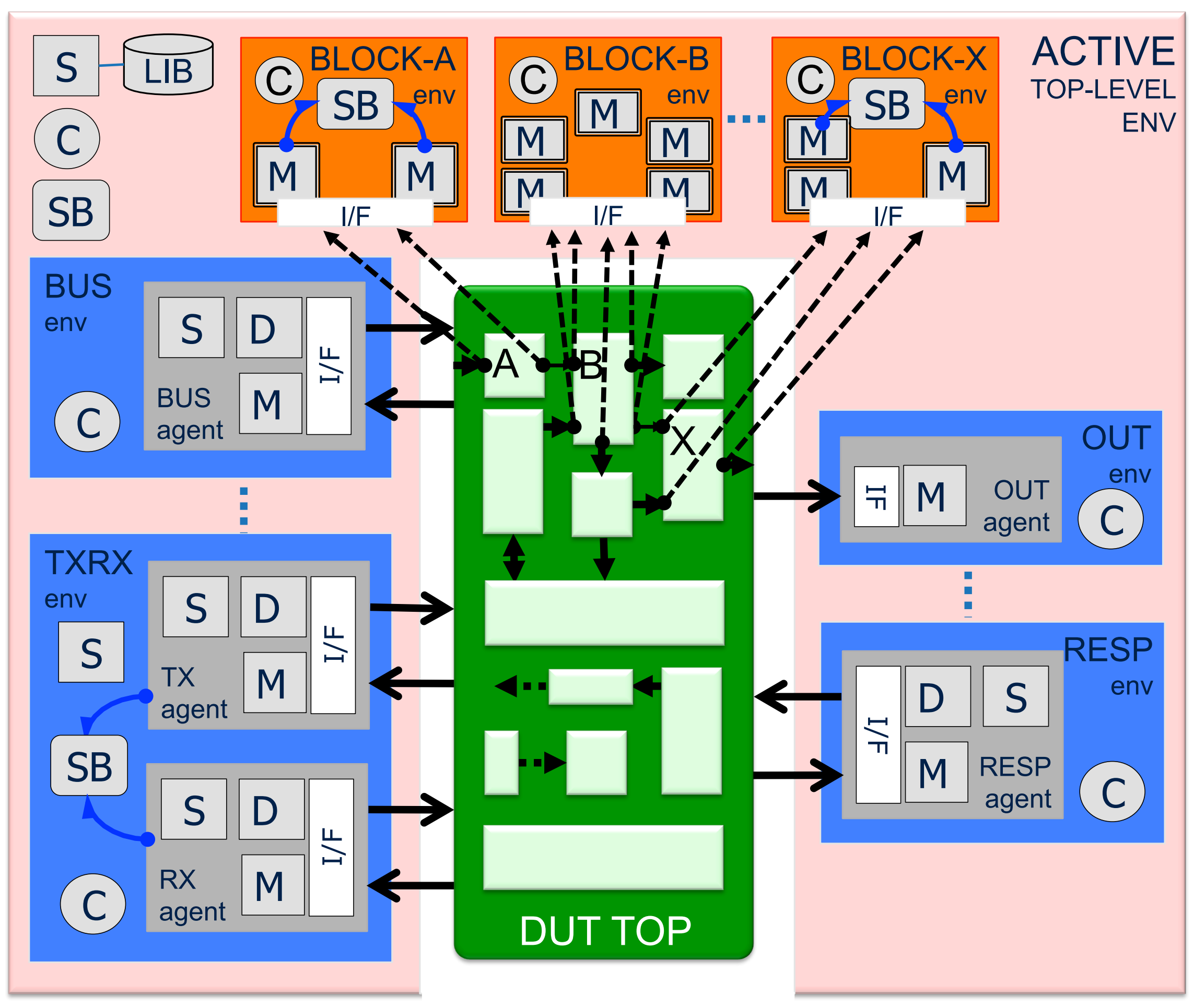
Additional Guidelines

- **validate assumptions** from formal in top-level
- **verification components** need to be **power-aware**
- enable **only appropriate RTL assertions** for top-level
- reuse **CDC assertions** at top-level
- exercise **caution** with bottom-up **CDC waiver** reuse
- enable **only appropriate AMS assertions** for top-level
- ...

Retrofit Reuse → **Test @ block-level** → **Integrate @ top-level**







verilab

mark.litterick@verilab.com