2023 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE

10 YEAR ANNIVERSARY

An Elegant scoreboard eco-system deploying UVM Callbacks, Parameterization for Multimedia designs from Imaging perspective

Chakravarthi Devakinanda, Sahana Ranganathan, Devendra Satish Bilaye, Vivek Kumar, Karthik Majeti

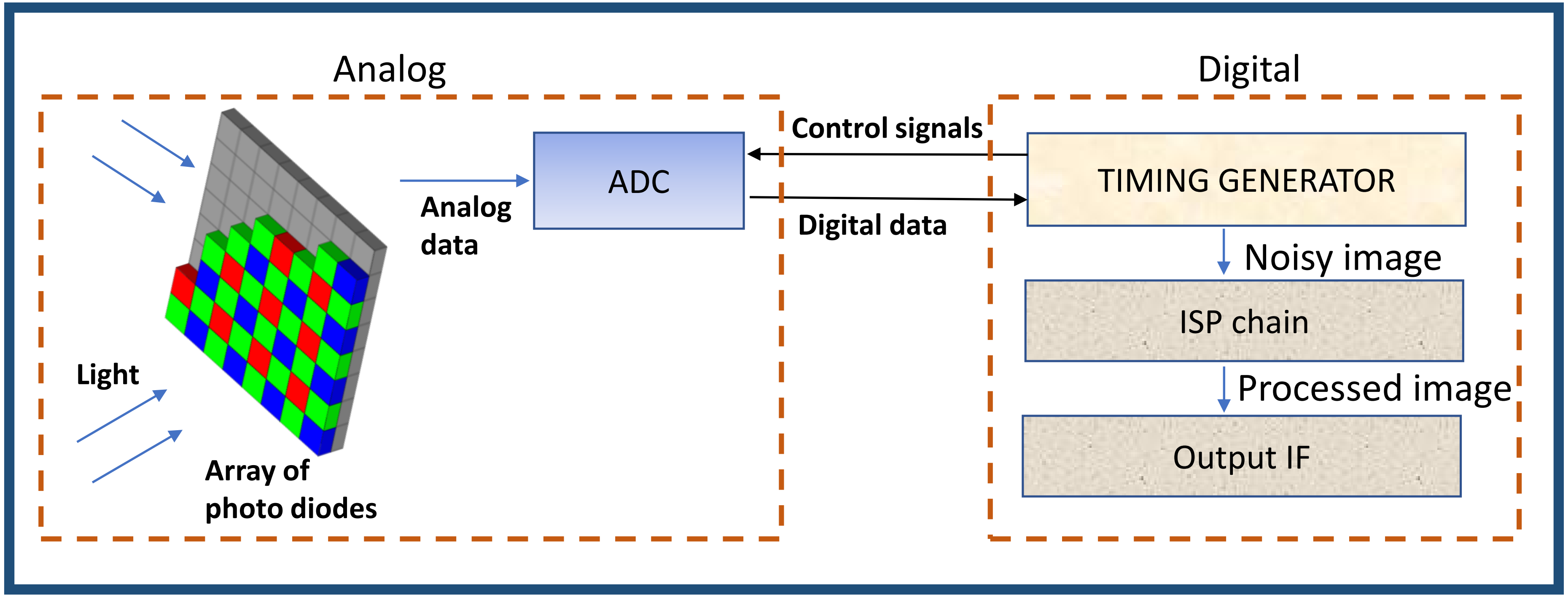
Samsung Semiconductor India Research

accellera SYSTEMS INITIATIVE

Agenda

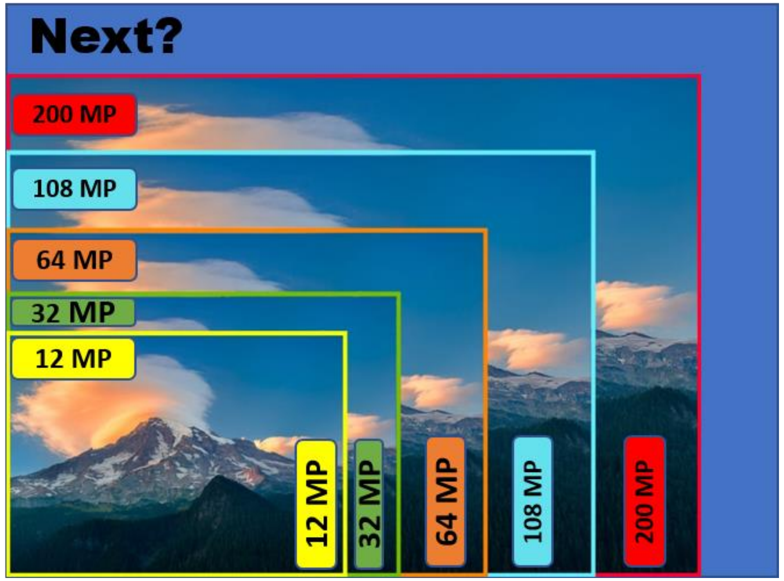
- Overview of CMOS Image Sensors
- Challenges in Multimedia SoC Verification
- Requirements from a Multimedia Testbench
- Addressing the challenges & requirements
- Results and Key Takeaways
- Q&A

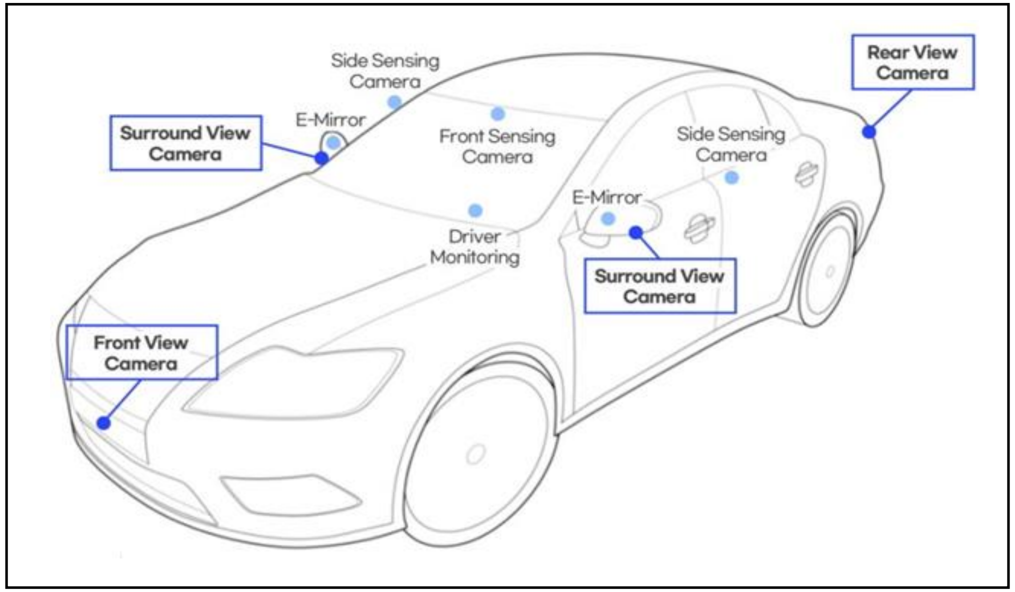
Overview of CMOS Image Sensors



Emerging market trends

- High resolution
- More Sensors
- New features

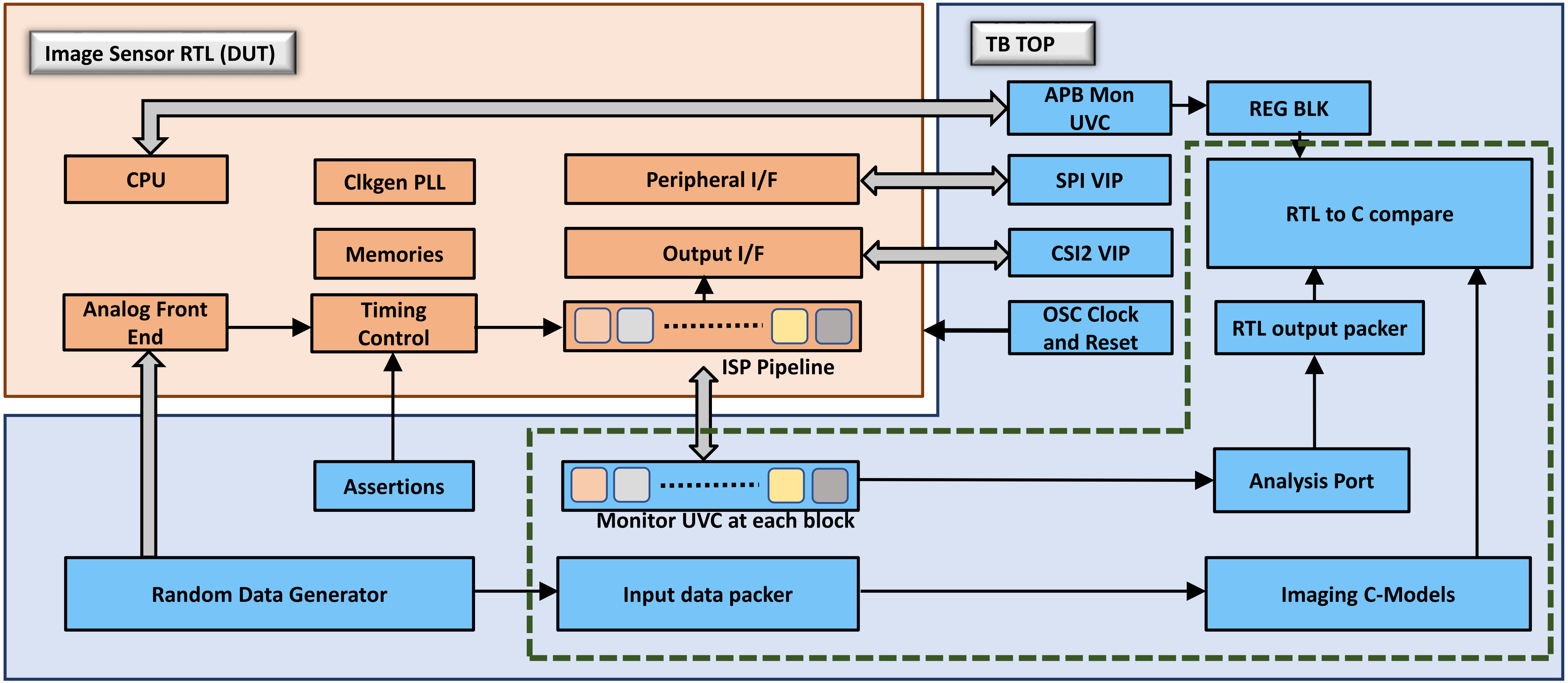




Challenges in Product development

- Shrinking product lifecycles - Spec to silicon < **6 months**
  - < **3 months** to close all verification metrics
- Higher resolutions (**200MP**) -> Simulation time ⇧
- Increasing Design complexity -> Requires exhaustive testing
- Meet stringent security, safety & quality standards

Basic Multimedia Test Bench



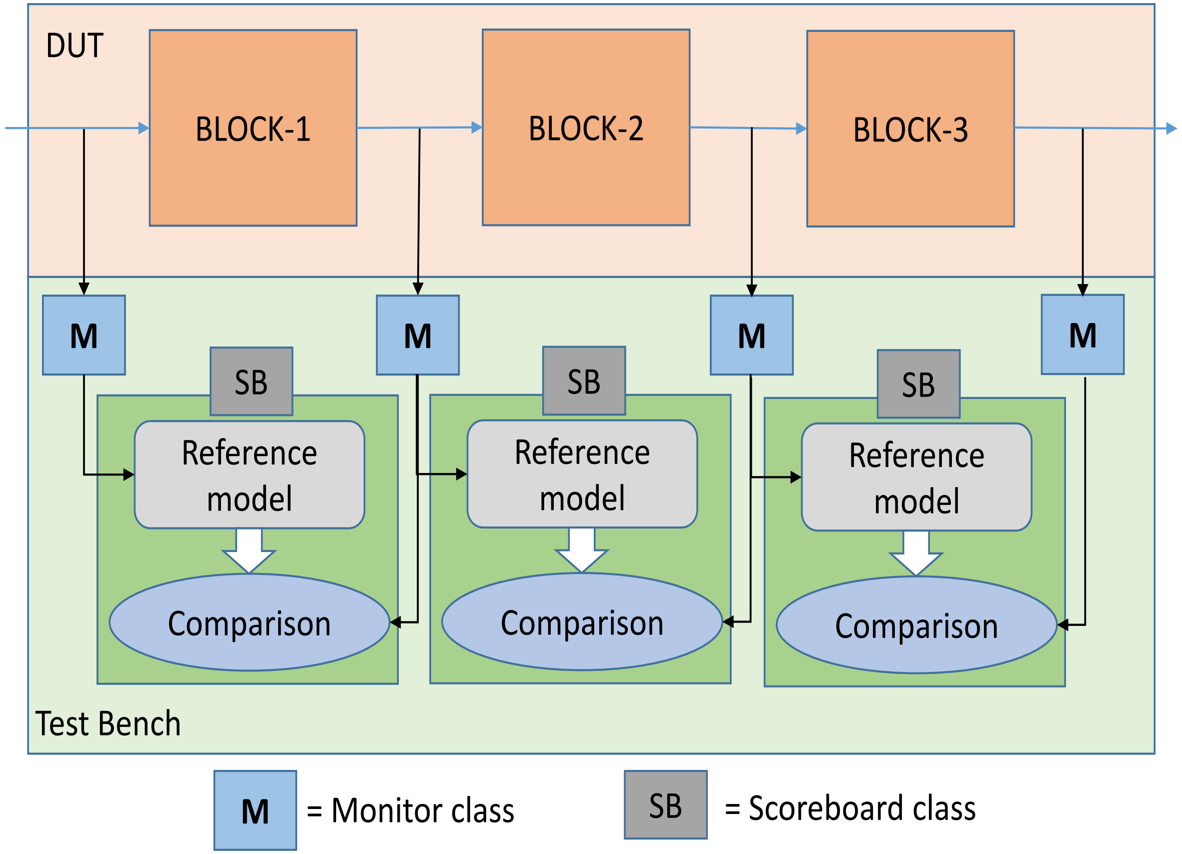
Bottleneck in TB - Scoreboard

- Bigger Designs -> 40+ blocks
- Higher resolutions -> Huge Amount of Data
- More Development time
- Limited Emulator licenses

Basic Scoreboard Structure

- Golden Reference Models :
  - C, C++, python etc
  - end-to-end
  - Unit models

- Unit level score boarding
  - End-end data integrity



Requirements from a Multimedia Scoreboard

Scalable

- New Algorithms -> New IPs
- 40+ blocks
- Higher resolutions

Adaptable

- Mobile Image Sensors
- Automotive and Other Image Sensors

Reusable

- Support for Emulation
- Verification at SOC, Sub-System and IP level
- Support for Gate Level Sims, Power Aware

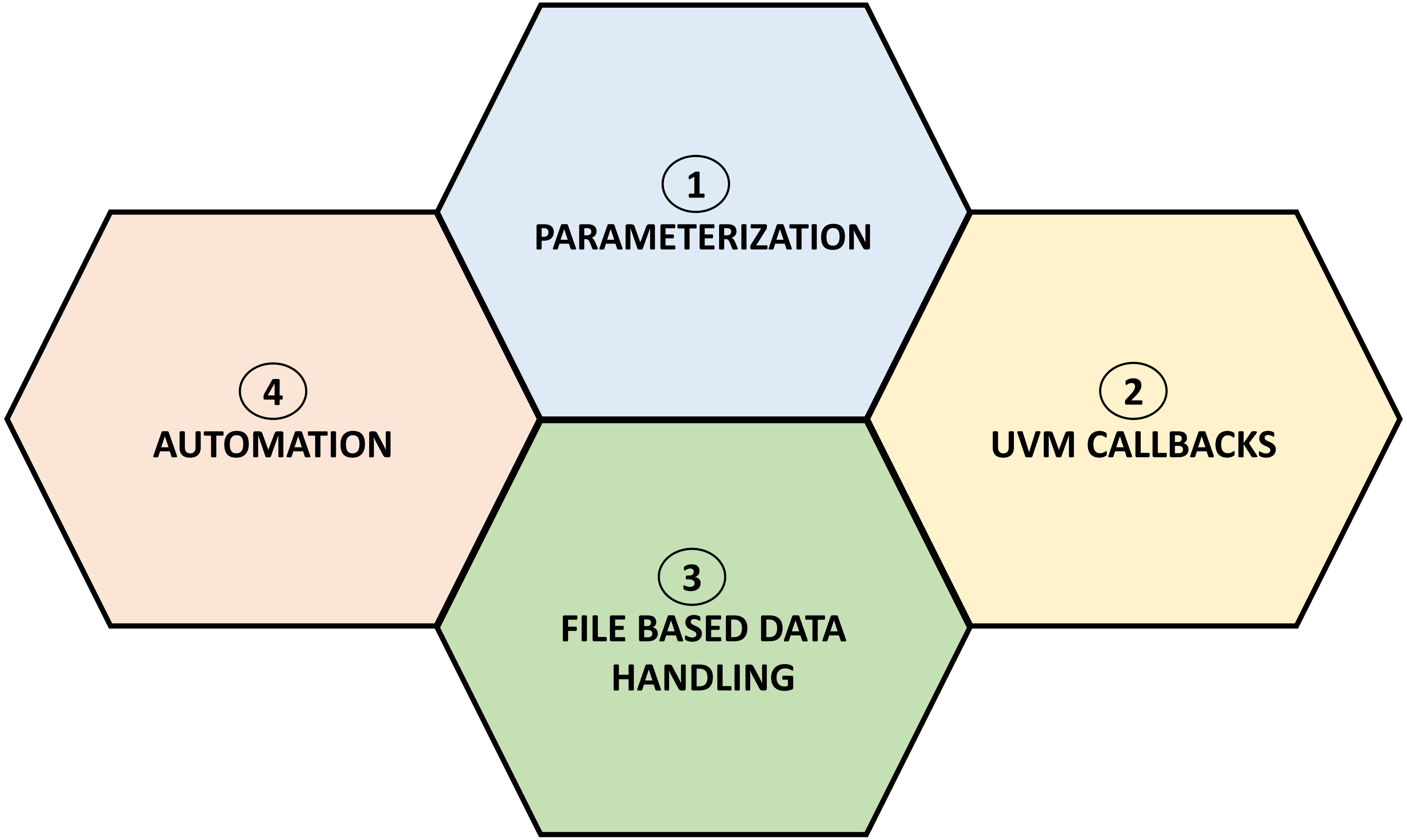
Automation

- Automation friendly at all steps – setup, regression, closure

Resource Management

- Optimal utilization of -> LSF, disk, man-power
- Avoiding Simulation Crashes
- Gain in run-times

Addressing the challenges & Requirements



Parameterization

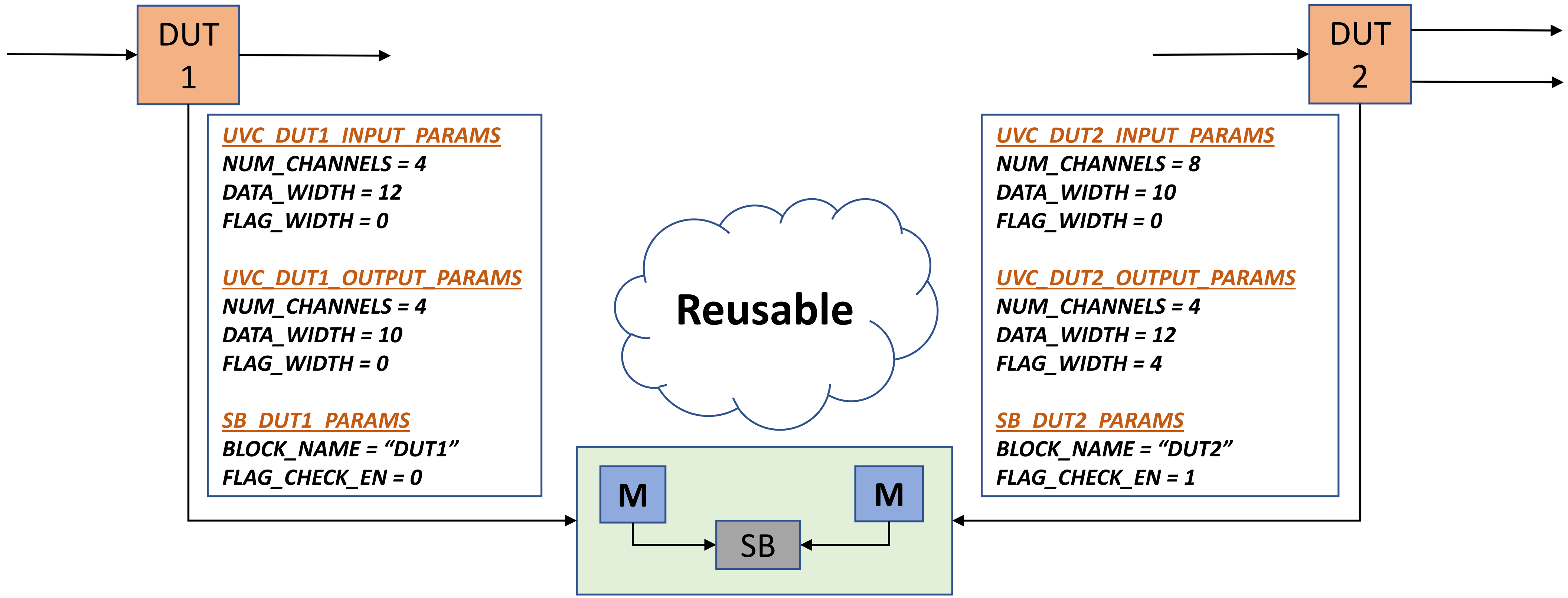
- UVC Parameters –monitor and driver handling
- Scoreboard Parameters –scoreboard handling

```
// UVC PARAMS
parameter uvc_params_t uvc_default_params = '{
  //Default values using defines
  `NUM_CHANNELS,
  `DATA_WIDTH,
  `HADDR_WIDTH,
  `VADDR_WIDTH,
  `FLAG_WIDTH
};
```

```
//SB PARAMS
typedef struct{
  int NUM_OF_IN_CHANNEL;//Reference model data channels
  int NUM_OF_OUT_CHANNEL;//Expected data channels for comparison
  string BLOCK_NAME; // Directory name.
  int ADDR_CHECK_EN; //To enable address checks
  int FLAG_CHECK_EN; //To enable flag checks
} sb_params_t;
```

```
//TRANSACTION PACKET
class uvc_data_packet_c#(uvc_params_t params = uvc_default_params)
extends uvm_sequence_item; //{
  rand bit [(params.DATA_WIDTH-1):0] data[params.NUM_CHANNELS-1:0];
  rand bit [(params.HADDR_WIDTH-1):0] haddr;
  rand bit [(params.VADDR_WIDTH-1):0] vaddr;
  rand bit [(params.FLAG_WIDTH-1):0] flag;
endclass: uvc_data_packet_c //}
```

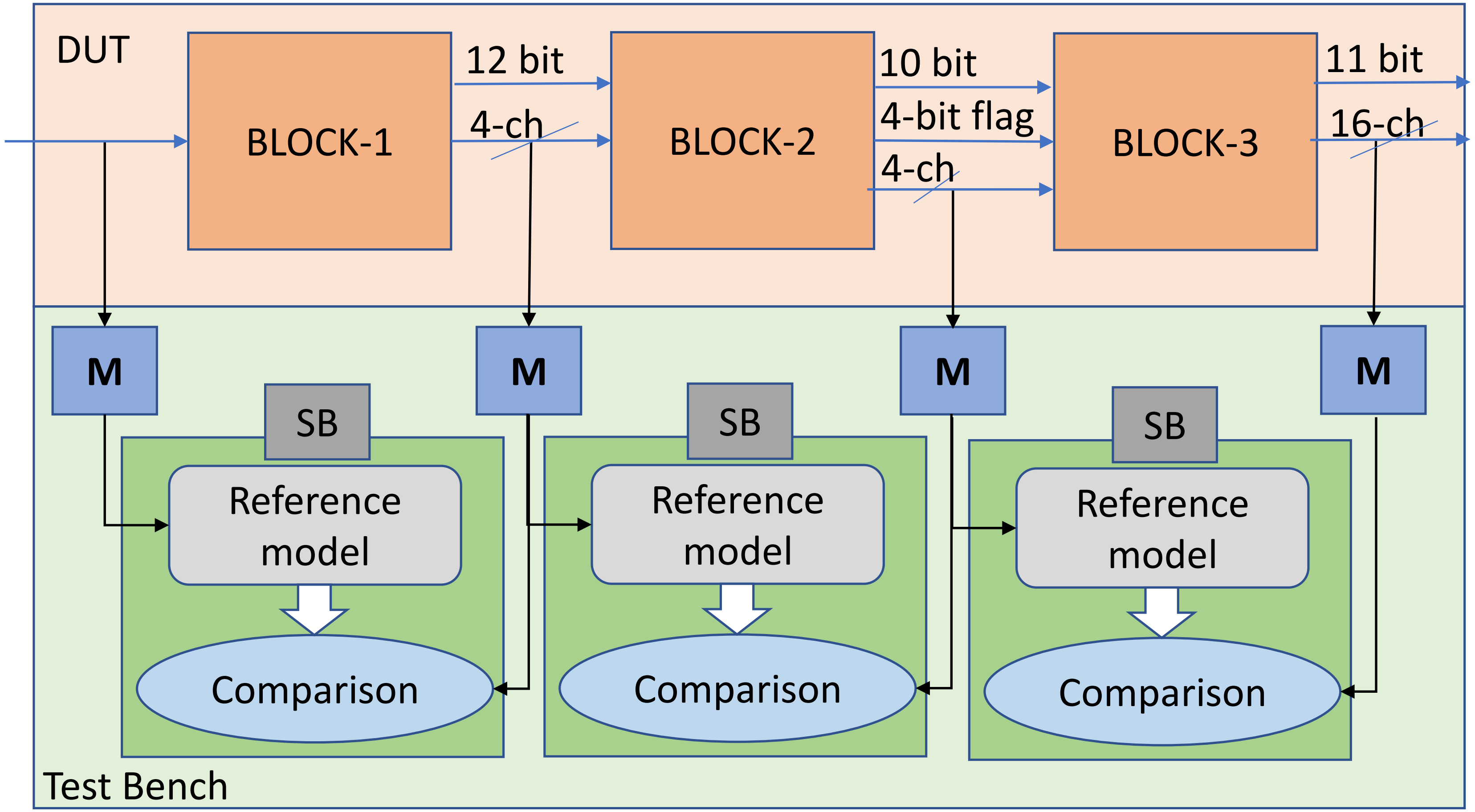
Parameterization – IP



Reusable SB and Monitor

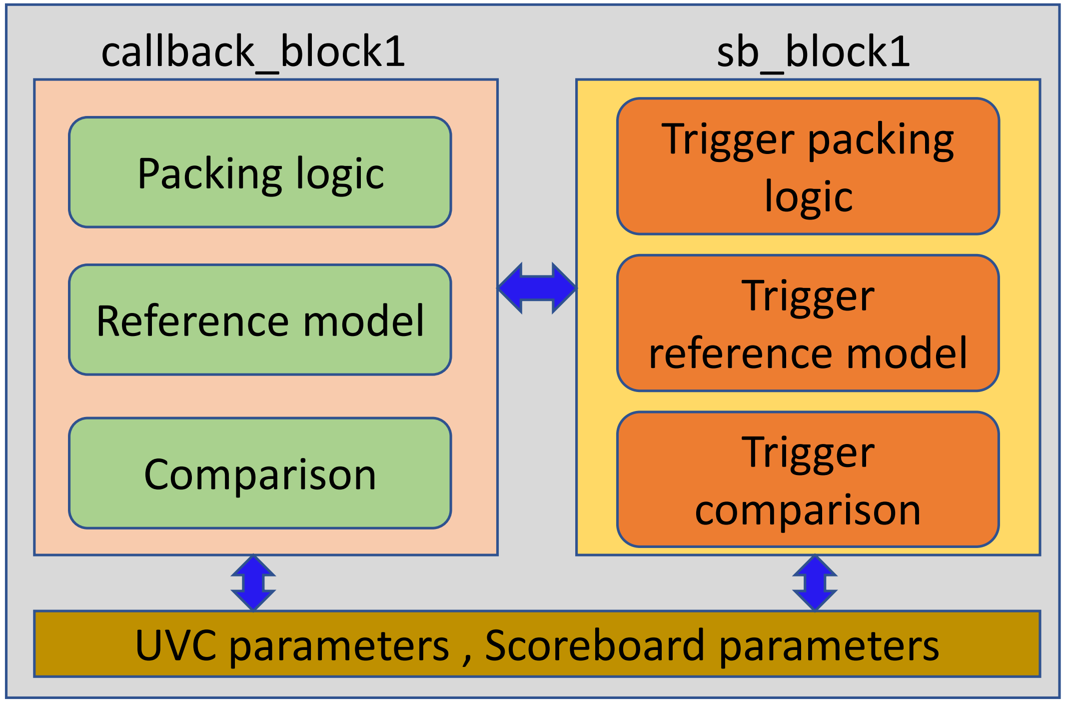
Parameterization - SOC

Scalable



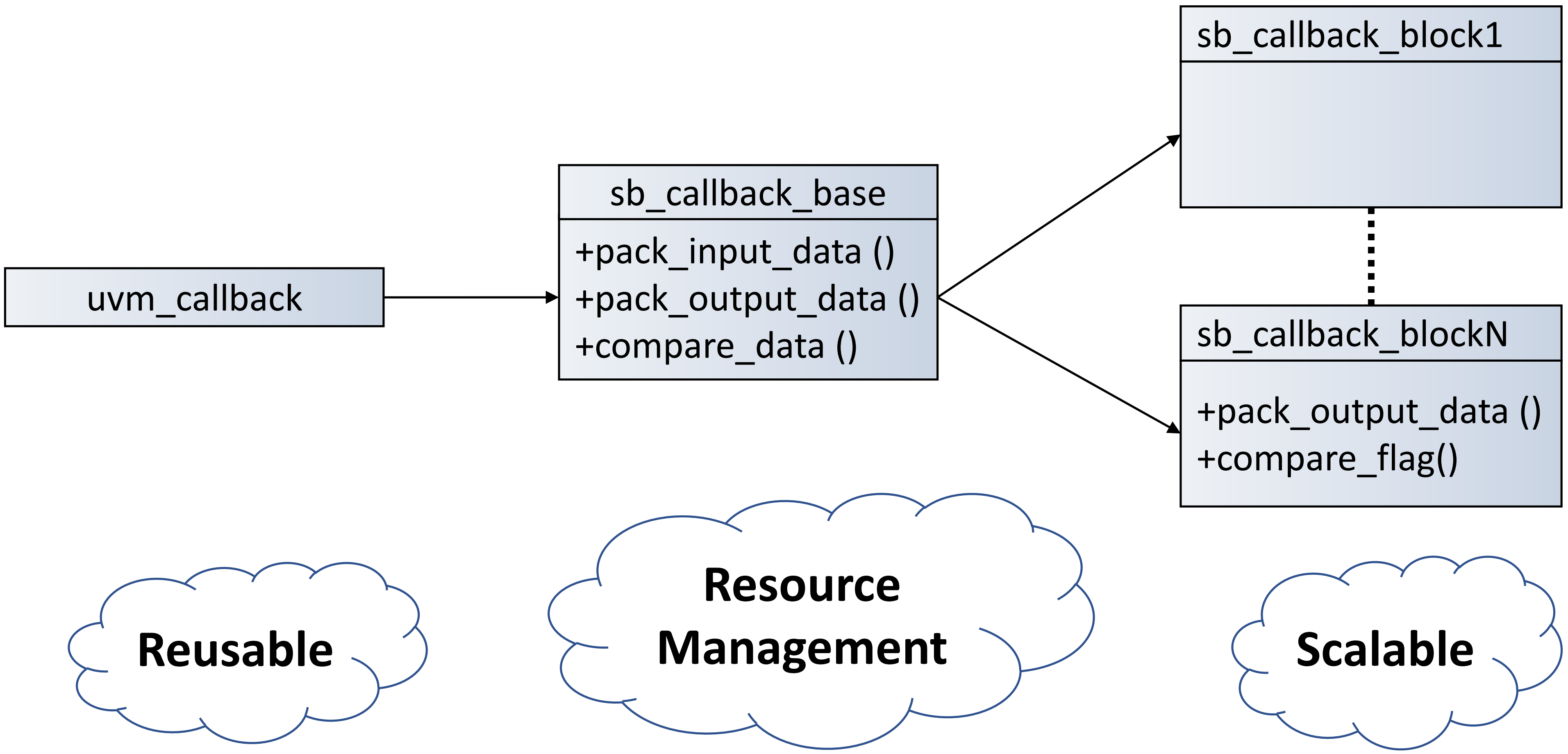
UVM Callbacks

- Scoreboard callback class
  - Implementation logic
- Generic scoreboard class
  - Trigger logic

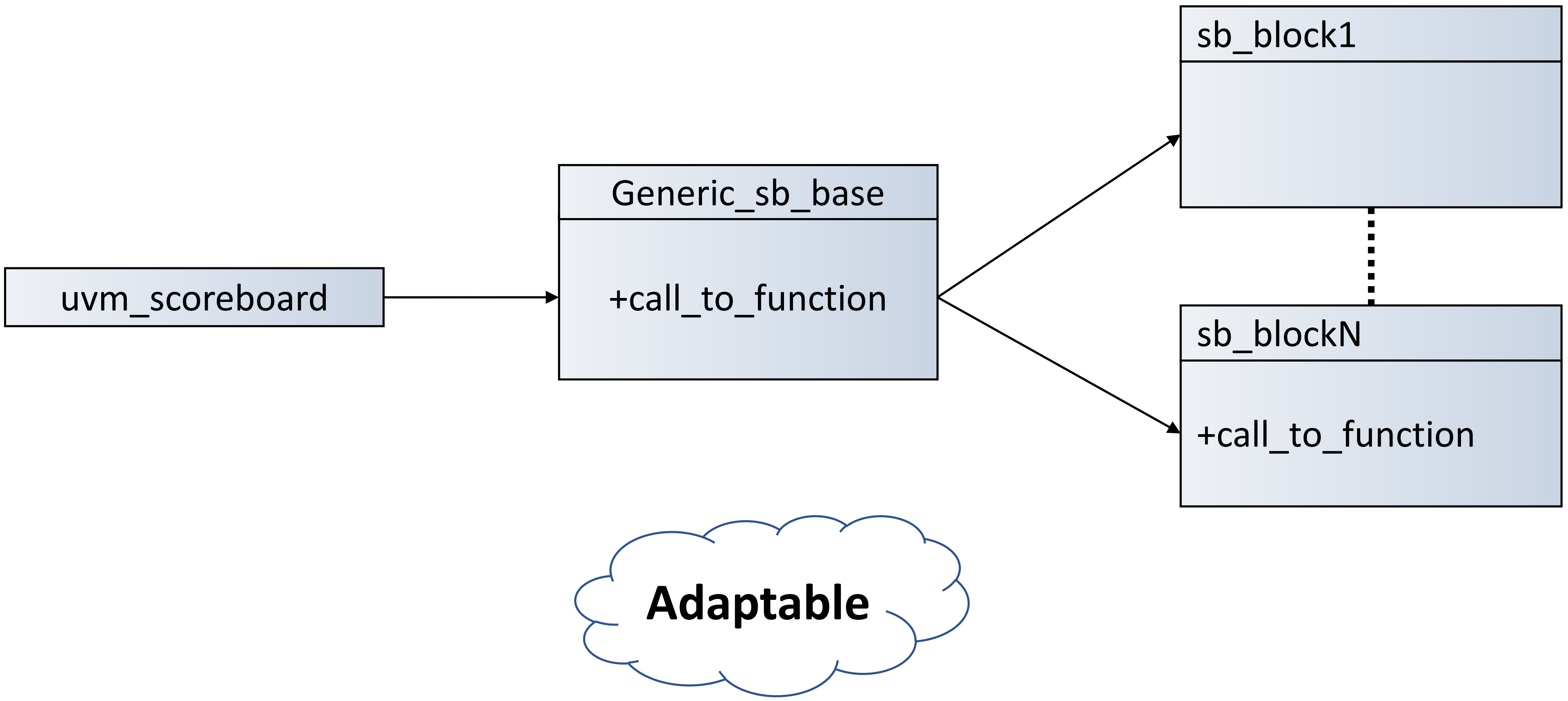



```
//Callback connections
uvm_callbacks#(generic_sb_c#(sb_params_block1,uvc_block1_input_params,uvc_block1_output_params),sb_callback_base)::add(sb_block1,callback_block1);
```

UVM Callbacks – scoreboard callback class



UVM Callbacks – Generic Scoreboard class



File-based monitor

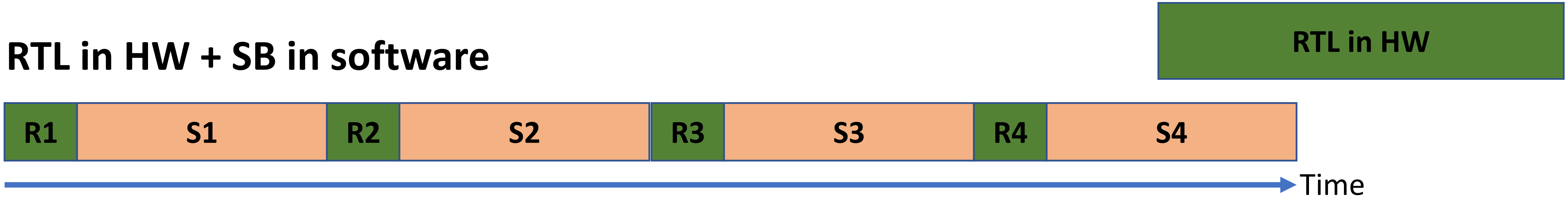
- Queue based monitor – previous method
  - Huge data→ Simulation crash
  - Big_mem LSF
- File based monitor – new method
  - Fewer Big_mem LSF
  - Leverage the benefits of both Emulation and Simulation
- Queue based + file based hybrid approach.

Resource Management

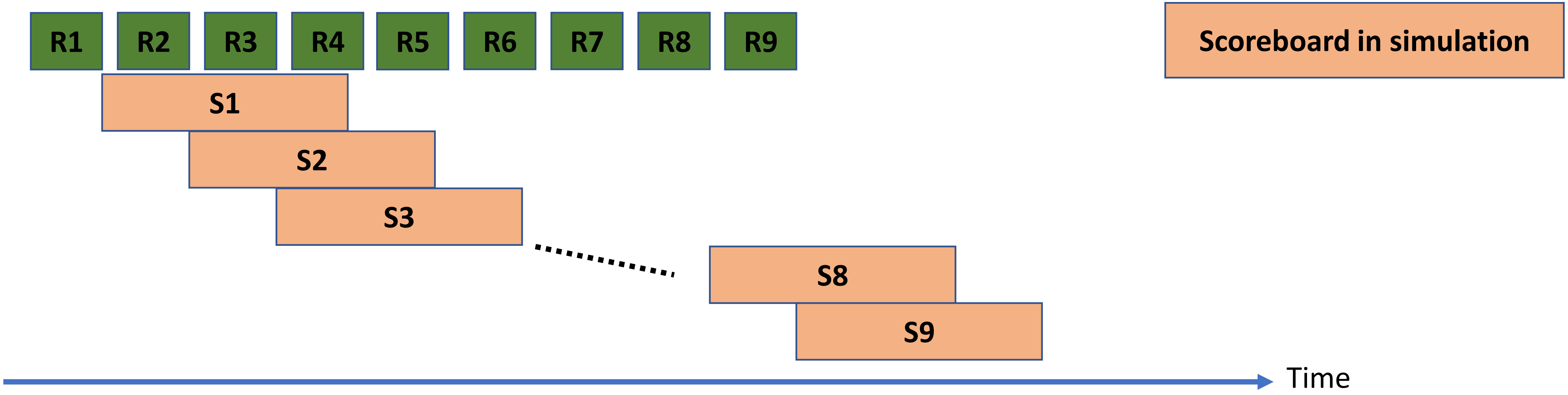
Reusable

File-based approach Advantages

- **RTL in HW + SB in software**

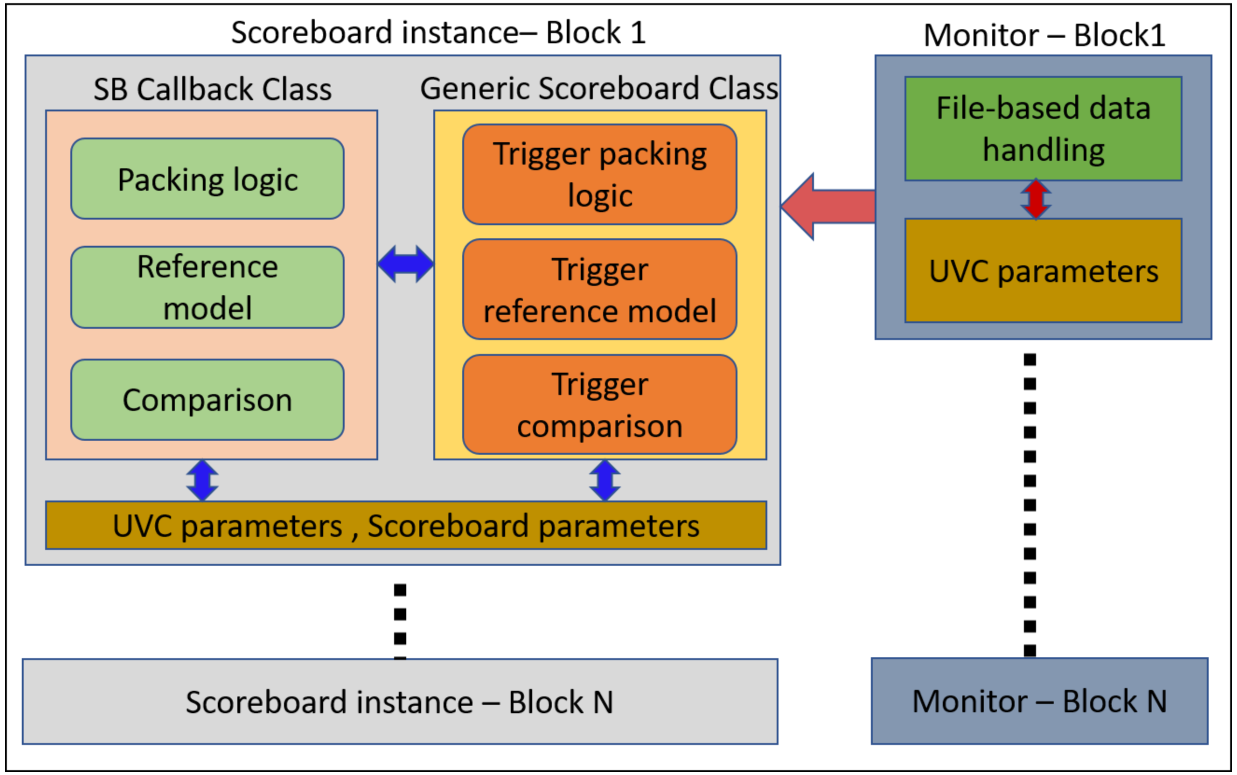


- **Emulation + simulation de-coupled Run with file-based approach.**

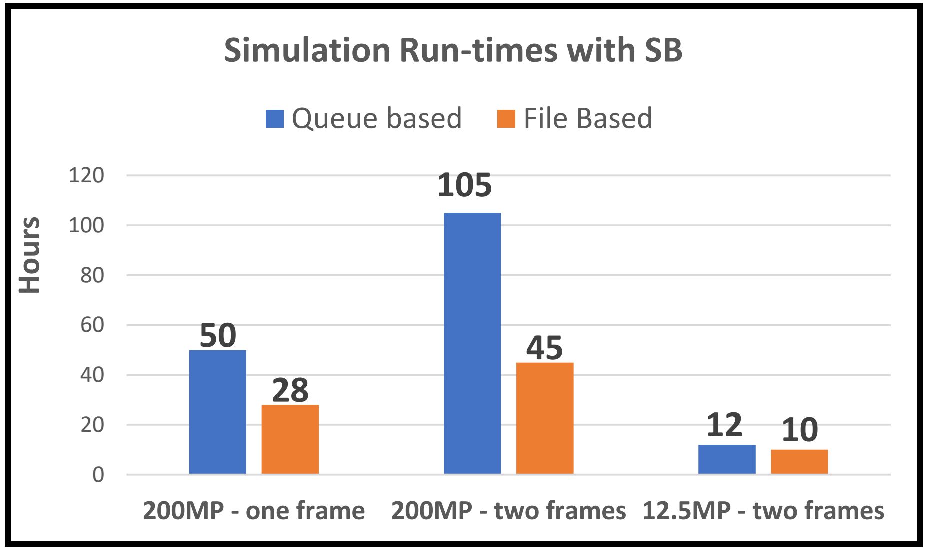


Automation

- XLS
  - Parameters
  - Connections
- XLS -> Script -> DUMP



Results


Simulation Run-times with SB
Queue based
File Based
Hours
120
100
80
60
40
20
0
50
28
105
45
12
10
200MP - one frame
200MP - two frames
12.5MP - two frames

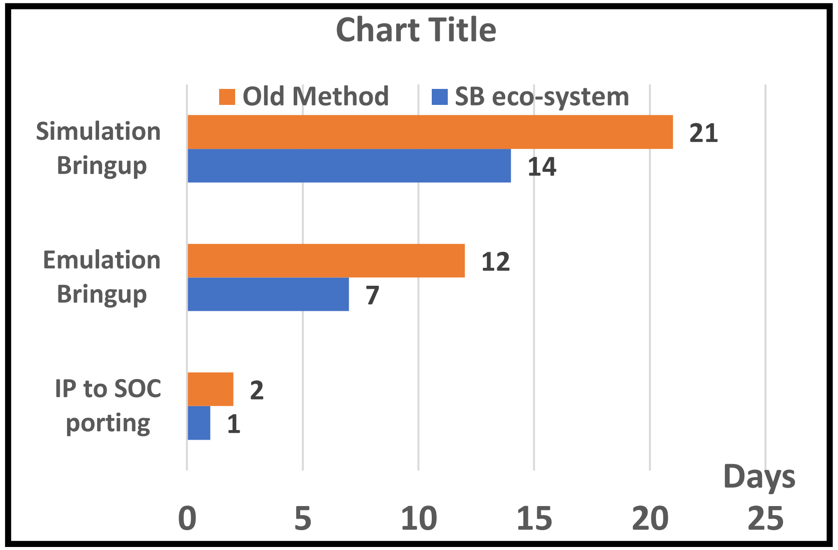


Chart Title
Old Method
SB eco-system
Simulation Bringup
21
14
Emulation Bringup
12
7
IP to SOC porting
2
1
Days
0
5
10
15
20
25


Regression - 1000 tests
Days
4
3
2
1
0
3
1.5
Old Method
SB eco-system

Key takeaways

• Easily scalable to large designs -> members of the team can work on different blocks independently.	**Scalability**
• Allows reuse of scoreboard files in IP, Sub-system & SoC. • Rapid scoreboard deployment - quickly sanitize in acceleration platform and regressions can be run in simulation.	**Reusability**
• Reduced big_mem LSF use, optimize emulator use, gain in run-times, fewer simulator crashes.	**Resource Management**
• Improved code readability – easy rampup. • Easily adaptable for other Multimedia Designs	**Adaptability**

Thank You

Q & A