Safe and Dynamic Driving
towards Vision Zero.

Chassis & Safety

SensePlanAct

Directed Testing using UVM-SystemC

Agenda

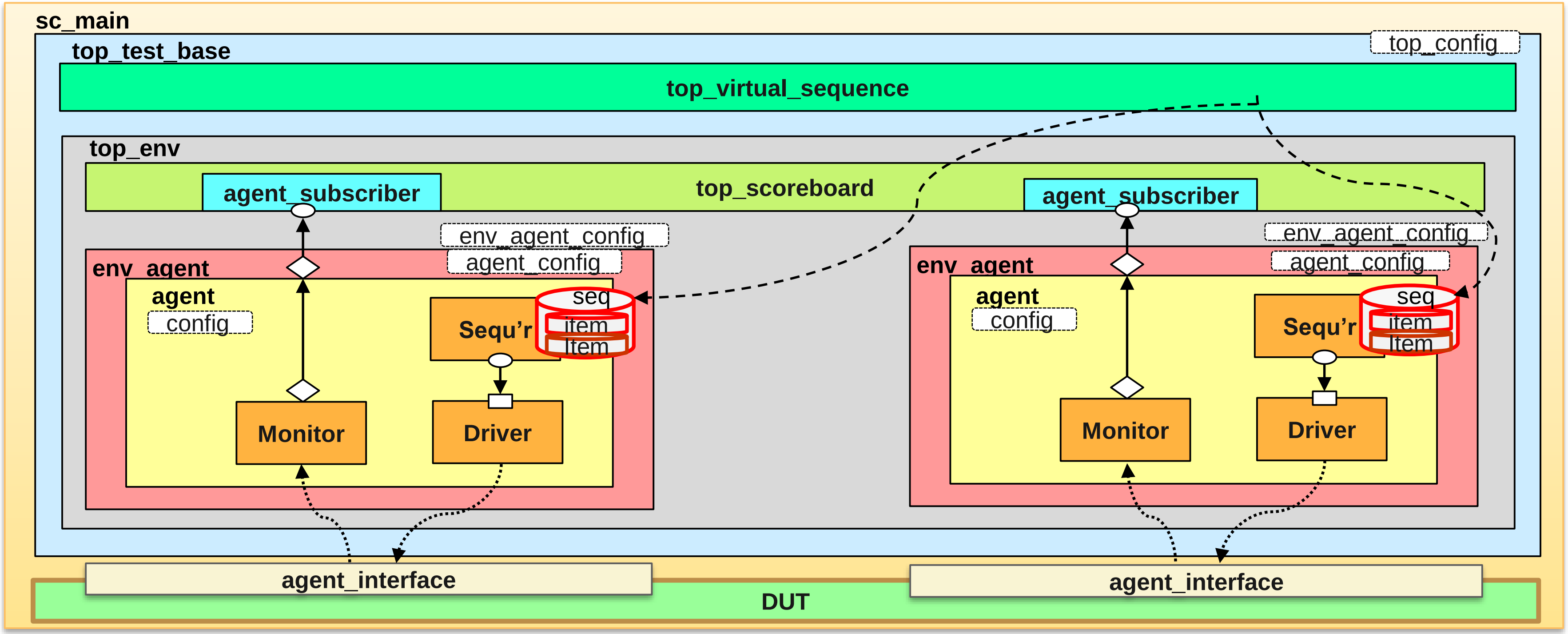
Why abuse UVM for Directed Testing?

- Randomization not yet part of UVM-SystemC
- Would need a reference model
- Definition of how to compare analog values?
- Coverage for analog simulations?

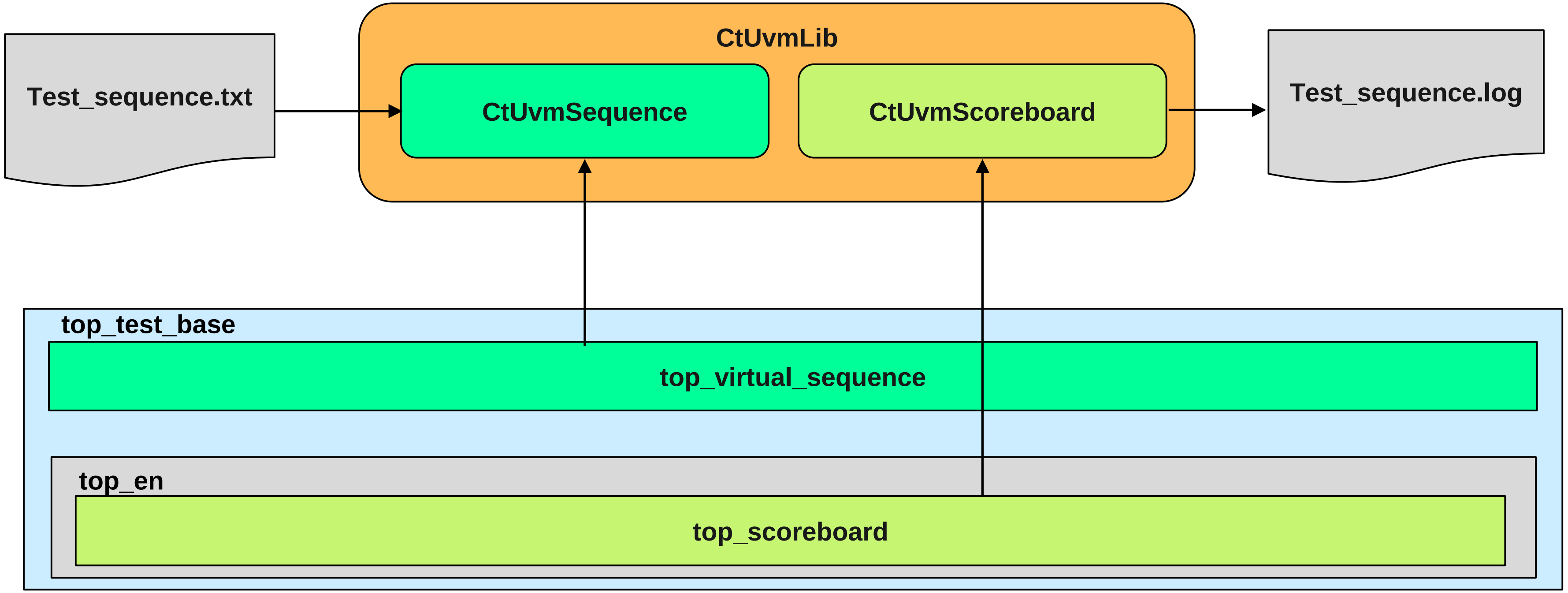
- Standardized Setup
- Coside UVM Generator
- Stimulus separated from test bench
- Reusable

Agenda

Standard UVM Setup



Our Extension to UVM-SystemC



Agenda

CtUvmSequence

- Contains UVM command data base.
- During construction, test sequence file is loaded and it stores the test sequence for further operations.
- Is inherited, constructed and used via top_virtual_sequence
- Has access to the port data base from child class.
- It will read the Commands, Parameters, Variables, Time Parameters, Include Files, Functions and commands from the included files and verify for it's correctness.
- Based on the commands, it will do the required process and execute the same.
- The top_virtual_sequence calls the appropriate sequencer to drive the inputs to DUT.
- After completion of the test sequence, simulation will be stopped.

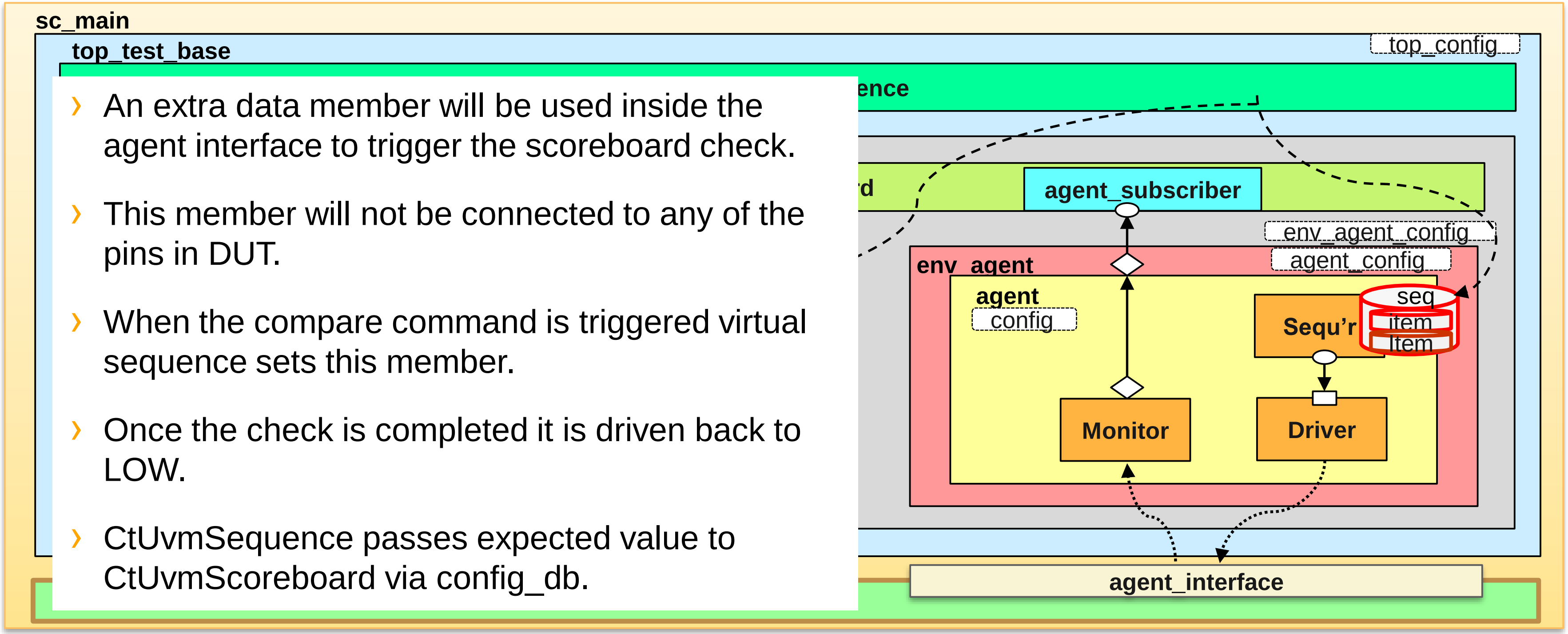
Available Commands

- Communication Commands
- Variable manipulation
- Time measurement
- Loop and Conditional Commands
- Parameter handling
- Function calls

CtUvmScoreboard

- During construction, it reads the test sequence name from uvm_config_db
- When Comparison command is driven from top_virtual_sequence, agent_subscriber will recognize the same and trigger the comparison of result.
- Depending of the type of comparison command, processing will be done and result is validated.
- Test results will be captured in log file after the result check.

Standard UVM Setup



- An extra data member will be used inside the agent interface to trigger the scoreboard check.
- This member will not be connected to any of the pins in DUT.
- When the compare command is triggered virtual sequence sets this member.
- Once the check is completed it is driven back to LOW.
- CtUvmSequence passes expected value to CtUvmScoreboard via config_db.

Example test program

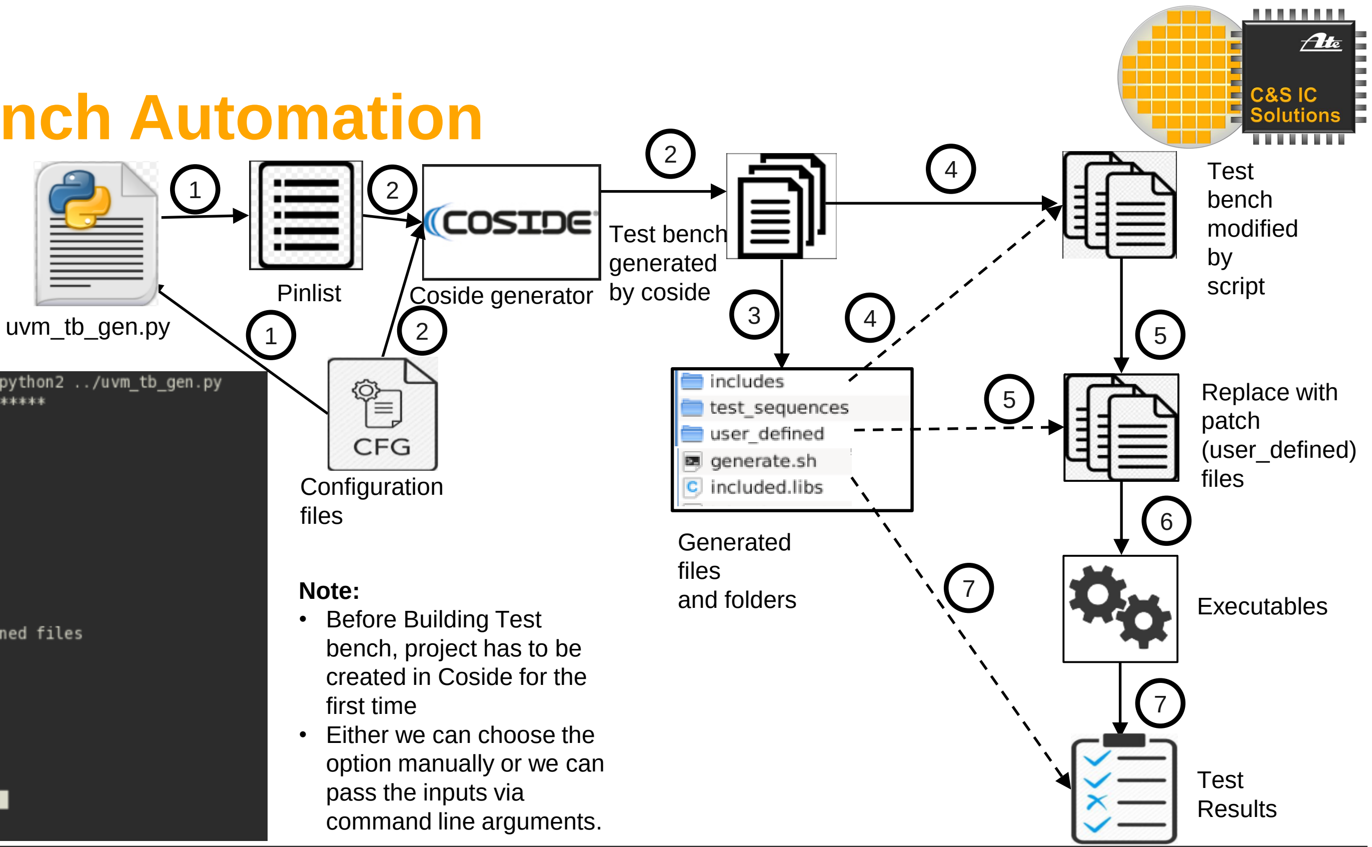
```
port_write(POR_N,3,3);
#wait for tp
wait(50e-6);
read12(DATAREG);
cmp_rx_data(EXPDATA,“Data register");
```

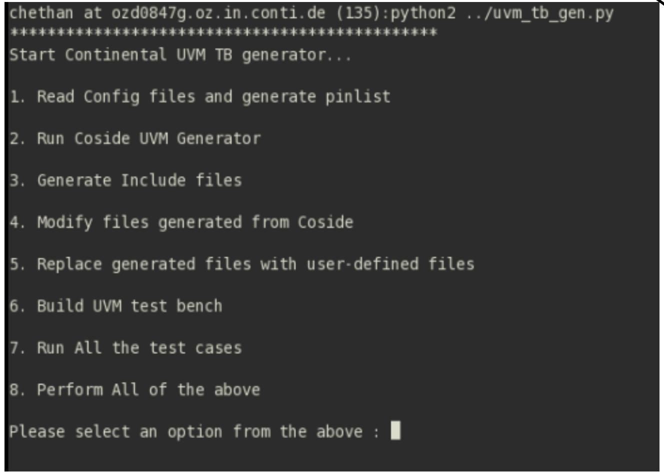
Agenda

Changes to Generated UVM files

- Each test bench generated with Coside UVM generator needs to be adapted
- ~ 10 files are affected for each test bench (sc_main, agents, subscribers, top_scoreboard and top_virtual_sequence
- Changes would need to be redone with each regeneration (e.g. interface update)

UVM Test Bench Automation




```
chethan at ozd0847g.oz.in.conti.de (135):python2 ../uvm_tb_gen.py
**********************************************
Start Continental UVM TB generator...

1. Read Config files and generate pinlist

2. Run Coside UVM Generator

3. Generate Include files

4. Modify files generated from Coside

5. Replace generated files with user-defined files

6. Build UVM test bench

7. Run All the test cases

8. Perform All of the above

Please select an option from the above : █
```

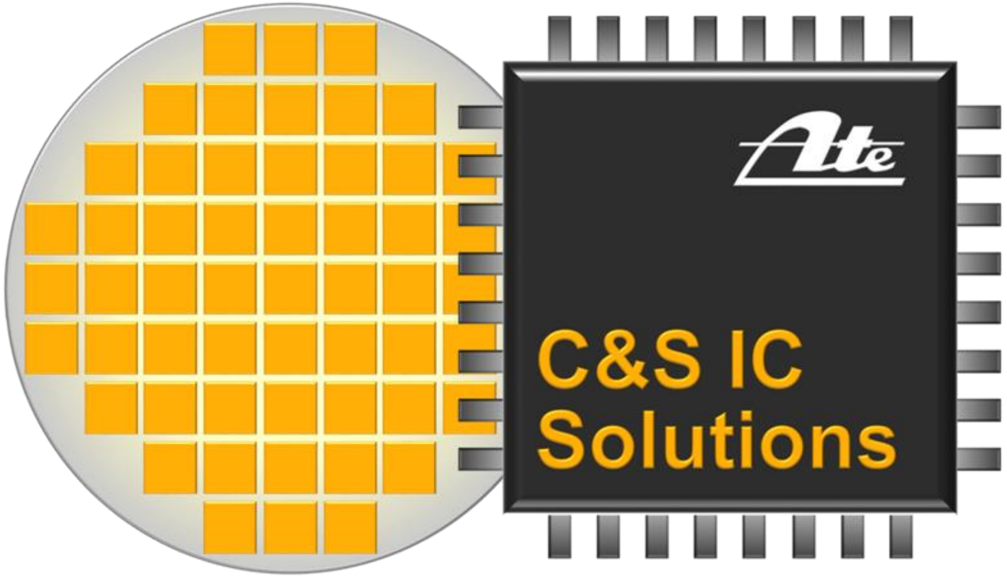
Note:
- Before Building Test bench, project has to be created in Coside for the first time
- Either we can choose the option manually or we can pass the inputs via command line arguments.

Agenda

Conclusion

- The structured approach of UVM is not only paying of in randomized scenarios but also in directed testing.
- By attaching a parser to the top_virtual_sequence many different test cases can be run without recompilation.
- The required adjustments to the UVM library were completely automated.

Thank you
for your attention!



ASIC solutions for ADAS

Safe and Dynamic Driving towards Vision Zero

SensePlanAct