Universität Bremen

Towards Checkpointing for SystemC/SystemC AMS -

Requirements & Challenges

07.12.2023 | 10th COSIDE User Group Meeting

Dr. Muhammad Hassan

DFKI GmbH/University of Bremen

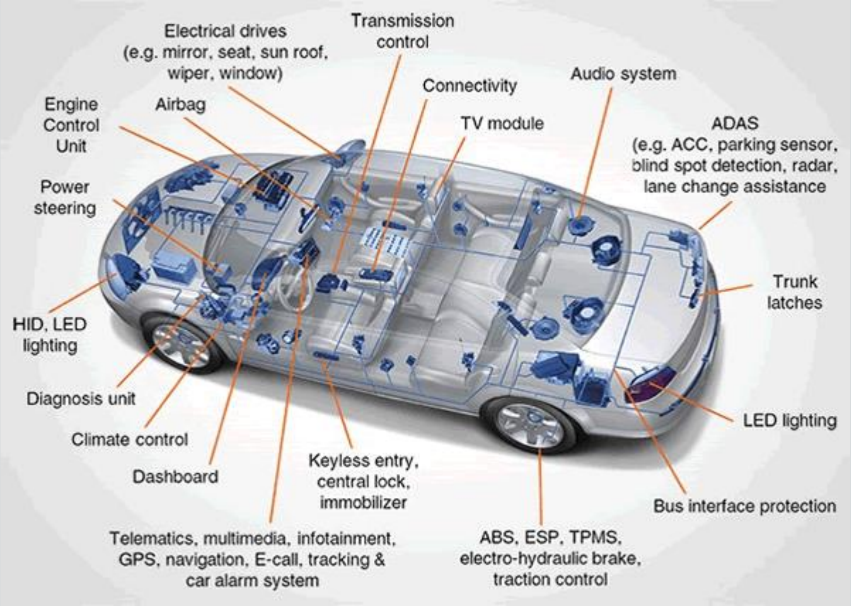
PaSVer

Bundesministerium für Bildung und Forschung

16ME0855

Motivation

- Increasing complex electrical and electronic components in automotives demands high-performance and data processing computing nodes
- Automotive OEMs must consider entire system together for testing & verification instead of testing only individual elements


Electrical drives (e.g. mirror, seat, sun roof, wiper, window)
Transmission control
Connectivity
Audio system
TV module
ADAS (e.g. ACC, parking sensor, blind spot detection, radar, lane change assistance
Engine Control Unit
Airbag
Power steering
Trunk latches
HID, LED lighting
Diagnosis unit
LED lighting
Climate control
Dashboard
Keyless entry, central lock, immobilizer
Bus interface protection
Telematics, multimedia, infotainment, GPS, navigation, E-call, tracking & car alarm system
ABS, ESP, TPMS, electro-hydraulic brake, traction control

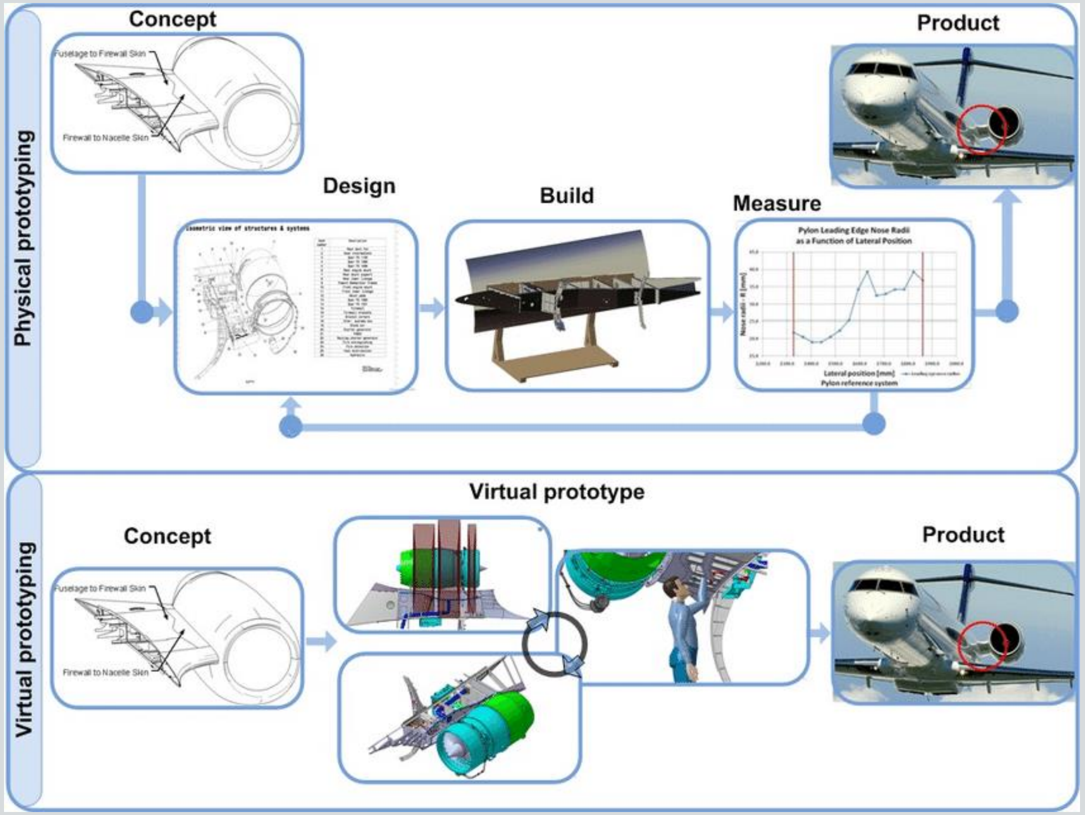
[1] https://connectorsupplier.com/evolution-automotive-connectivity-autonomous-vehicle-technology-drives-need-speed-bandwidth/

Virtual Prototype (VP)

- Virtual Prototype (hardware replica) is used
  - For specification, implementation, verification
  - Reduces costs
  - Helps develop the right system & develop the system correctly (detecting errors early)

- Reduces bottleneck in overall system simulation time

- Create technical base for exchange of scenario along the Electronic design chain (e.g., test and application scenarios)



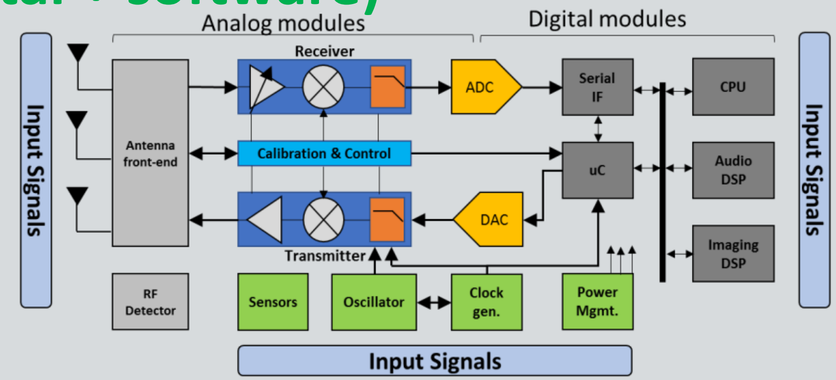
Toche, Boris & Pellerin, Robert & Fortin, Clément & Huet, Greg. (2012). Set-Based Prototyping with Digital Mock-Up Technologies. IFIP Advances in Information and Communication Technology. 388. 299-309. 10.1007/978-3-642-35758-9_26.

PASVER Project Goals

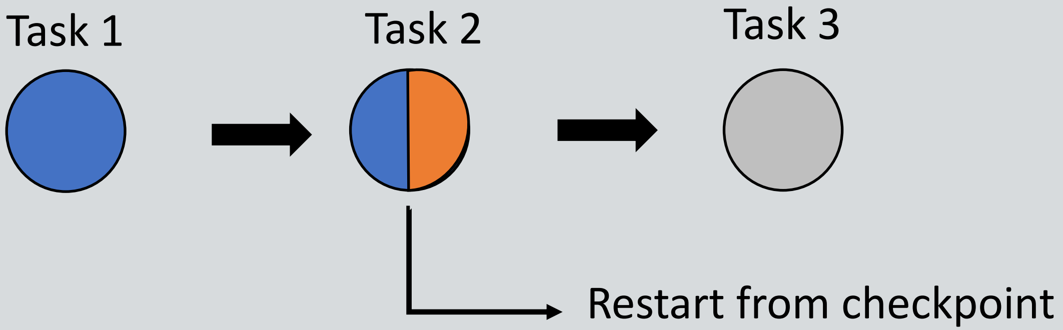
PaSVer

- Accelerate simulations in mixed-signal VPs using checkpointing technique

- **SystemC AMS – development of checkpointing APIS for heterogeneous virtual prototype (analog + digital + software)**



- Checkpointing - technique used to store and restore system state



https://steamcommunity.com/sharedfiles/filedetails/?id=972742773

Checkpoint Applications

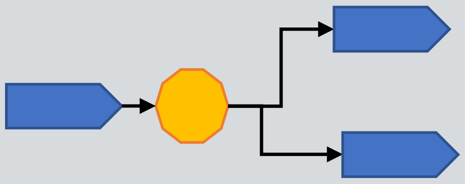
- Checkpoint saves the system config (e.g., saves time in boot sequence, sensor initialization, etc.,)

- Checkpoint can be used as a basis and inputs can be varied for parallel execution of scenarios

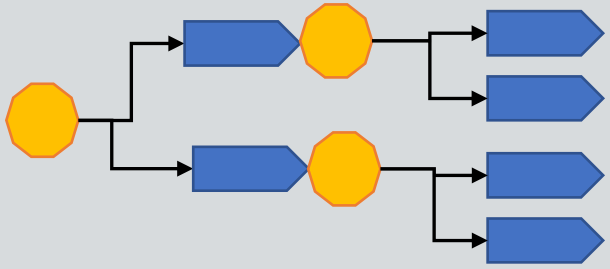
- Checkpoint can be used to analyze the cause of the error

For VP's checkpoints shall include (memory state, processor state, peripheral, active transactions, etc.,)

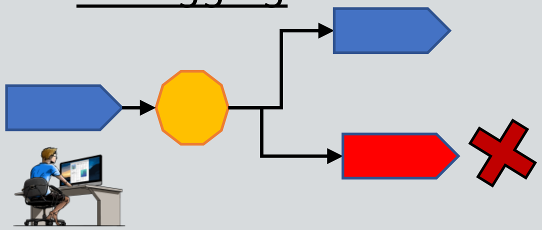
Save system config



Parallel Execution

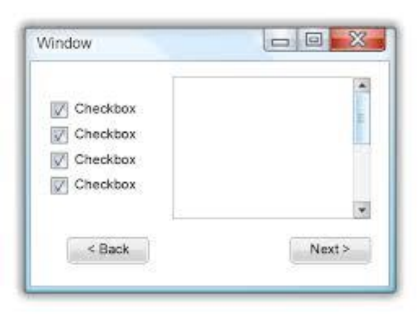


Debugging

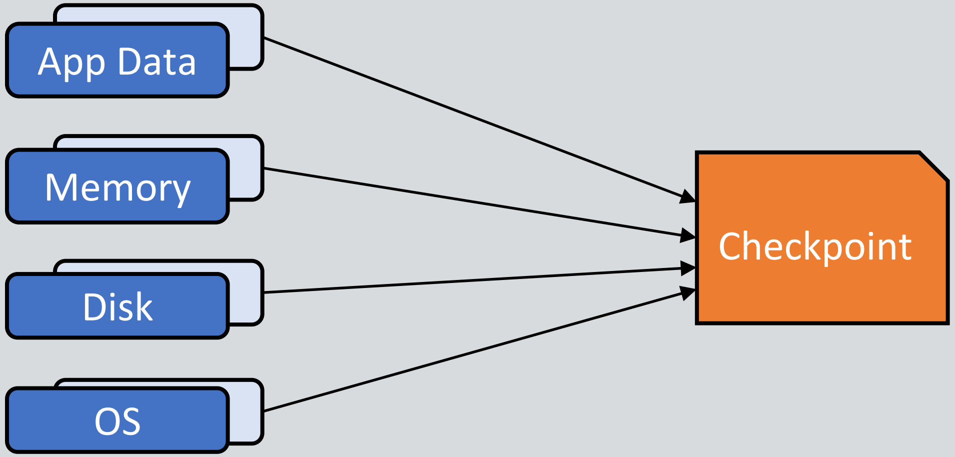


Challenges in Checkpointing

- Manual Checkpointing (variant - 1)
  - requires standardization enabled interfaces (API)



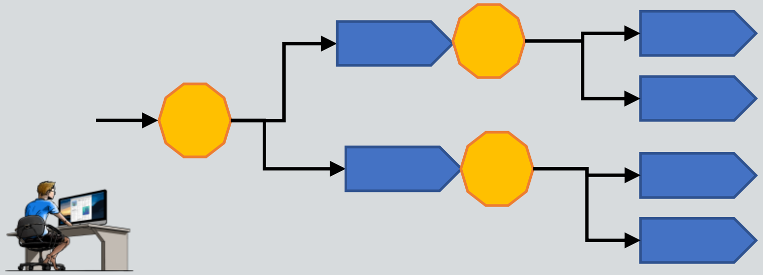
- Checkpointing by storing process context (variant -2)
  - requires efficient storage mechanism
  - Checkpoints cannot be transferred to other System
  - Modification of Checkpoints hardly possible



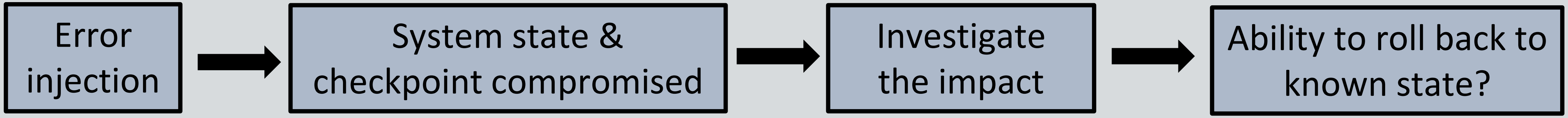
Challenges in CI Integration & Debugging

- System modification in VP for simulating in a Continuous Integration (CI) environment

In a CI flow, a change to input triggers the subsequent simulations runs



- Distribution of checkpoints in the system for debugging failed simulation



Error injection → System state & checkpoint compromised → Investigate the impact → Ability to roll back to known state?

Challenges in Checkpoint metrics

- Metrics to assess checkpoints
- Algorithms to analyze checkpoints and simulation data to evaluate the parallelizable simulation runs
- Remove redundant scenarios to improve simulation time

Roadmap for Checkpointing Implementation

- Development of a methodology to implement checkpointing in mixed-signal systems
- Analyze the performance of checkpointing methods
- Describe necessary changes in VP with checkpointing to implement it in a CI system
- Examine effect of error injection and system’s ability to recover
- Explore metrics to compare checkpoints
- Investigate possible algorithms to eliminate redundant test scenarios and accelerate simulation
- API for Checkpointing
  - Give API to Accellera as long term goal

Food for Thought

- CCI parameters essentially provide the core mechanism needed to represent state outside of a running model
  - Host-independent expression of data values with a name
  - Name-value mappings and a type system for values
- Still need to deal with kernel, structure, and making sure all state gets saved and restored
  - Restoring SystemC threads
- **Store Solver States for SystemC AMS**
- Models might have to be adapted to support checkpointing, in particular:
  - Keep model state explicit and separate from implementation
  - Deal with thread positions
  - Deal with in-place state changes
  - Avoid keeping things on the stack

Towards Checkpointing for SystemC/SystemC AMS -

Requirements & Challenges

07.12.2023 | 10th COSIDE User Group Meeting

Dr. Muhammad Hassan

DFKI GmbH/University of Bremen

Bundesministerium für Bildung und Forschung

16ME0855