2023
DESIGN AND VERIFICATION™
DVCON
CONFERENCE AND EXHIBITION
EUROPE
10 YEAR ANNIVERSARY

Efficient Debugging on Virtual Prototype using Reverse Engineering Method

Sandeep Puttappa, Senior Staff Engineer

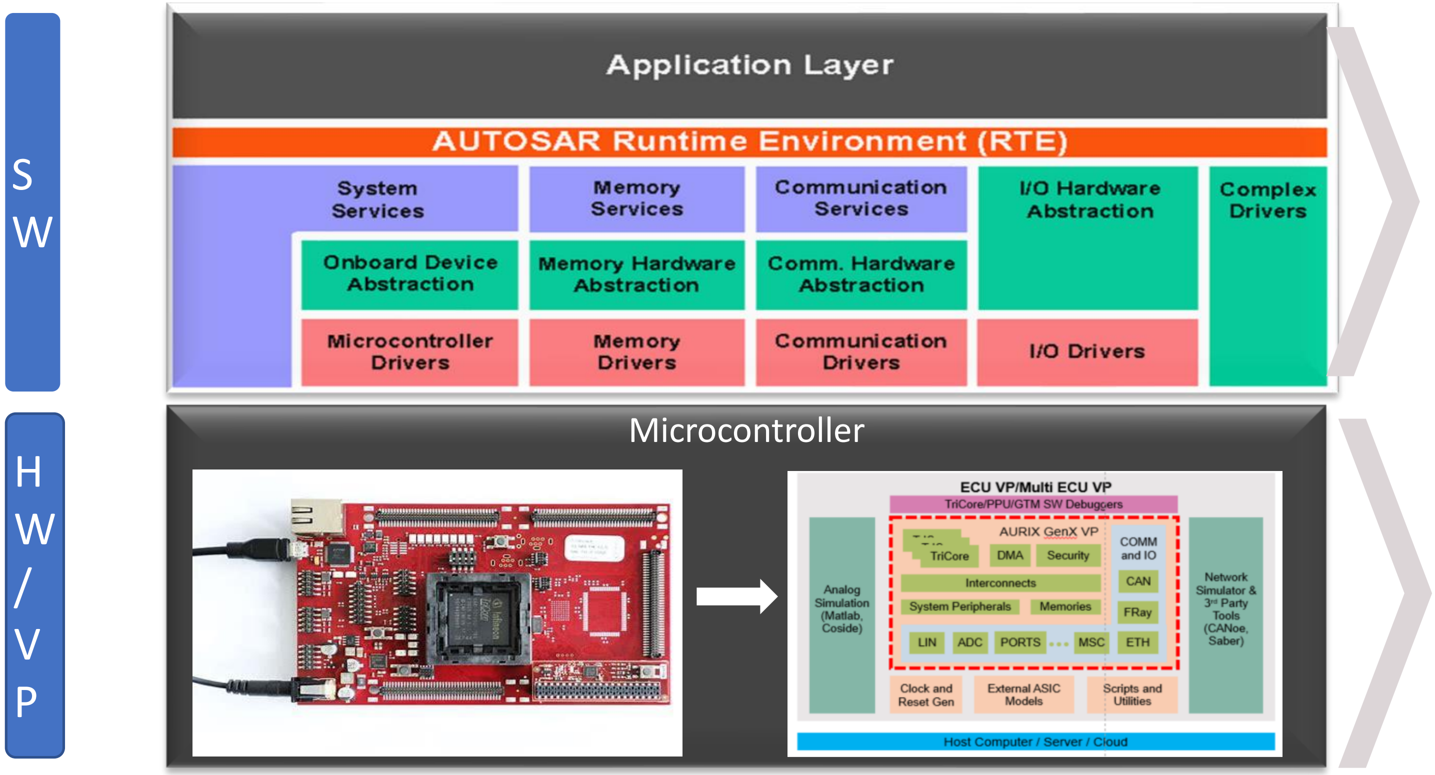
Dineshkumar Selvaraj, Lead Principal Engineer

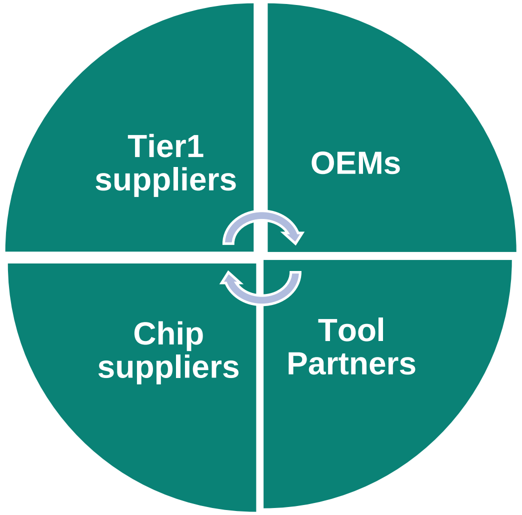
Ankit Kumar, Associate Engineer

Agenda

- Background
- Traditional debugging approaches on VP and its Limitations
- Reverse Engineering Method Implementation
- Proof of Concept
- Next Steps

Background

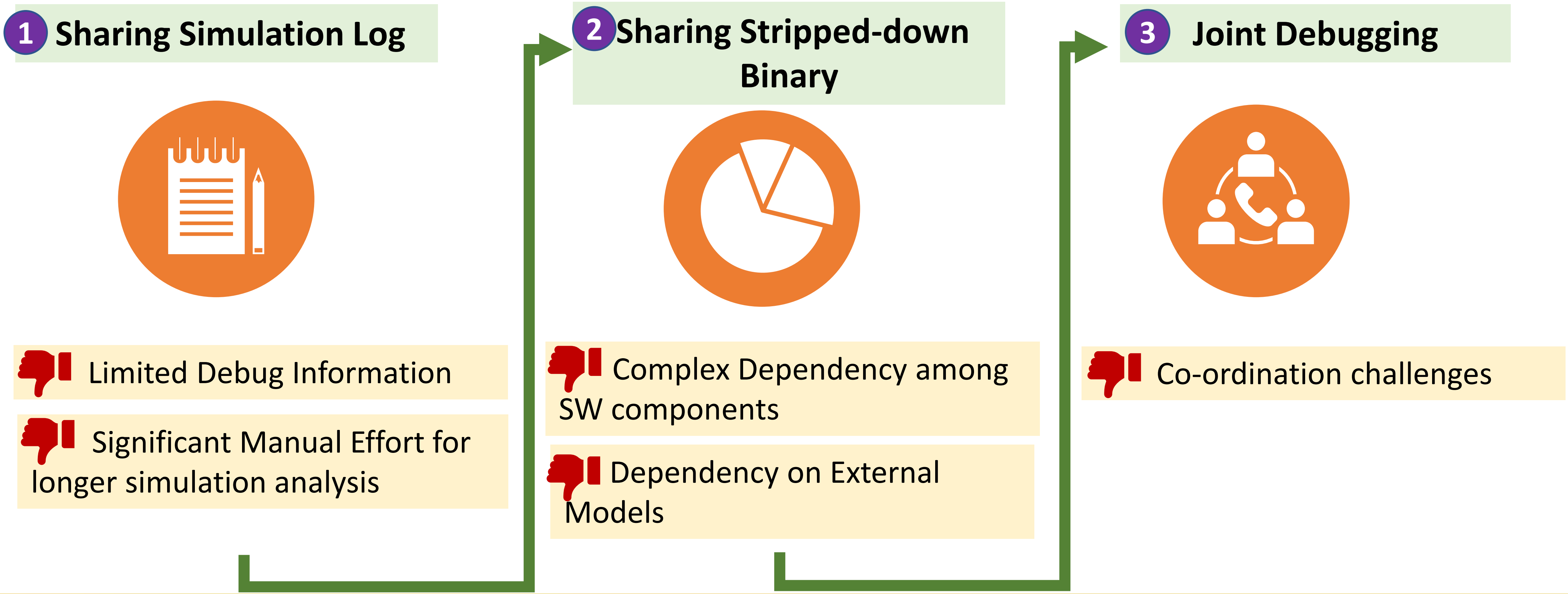




- Early Availability
- Ease of Use
- Longer Maintenance lifespan
- Superior debugging and tracing Capabilities

How to Debug Issues on Virtual Prototype in the absence of SW?

Traditional VP Debugging Approaches



1 **Sharing Simulation Log**

👎 Limited Debug Information

👎 Significant Manual Effort for longer simulation analysis

2 **Sharing Stripped-down Binary**

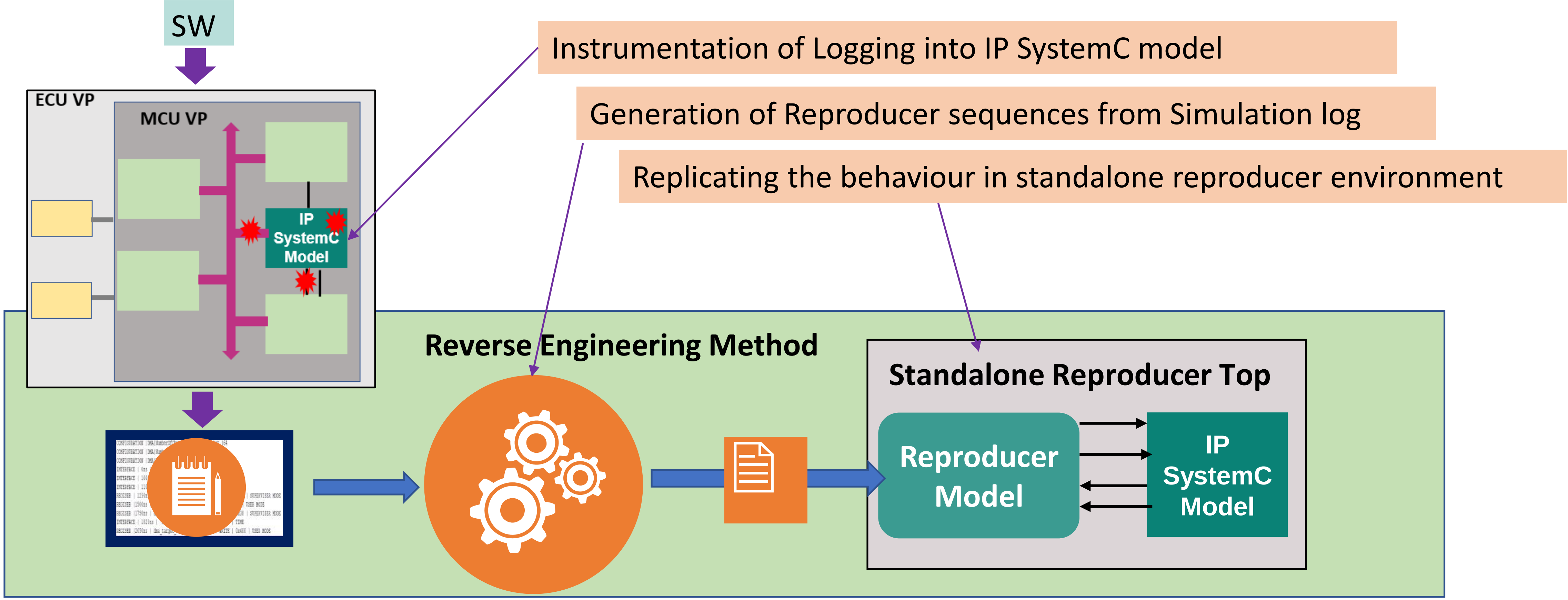
👎 Complex Dependency among SW components

👎 Dependency on External Models

3 **Joint Debugging**

👎 Co-ordination challenges

Proposed Solution – Reverse Engineering Method

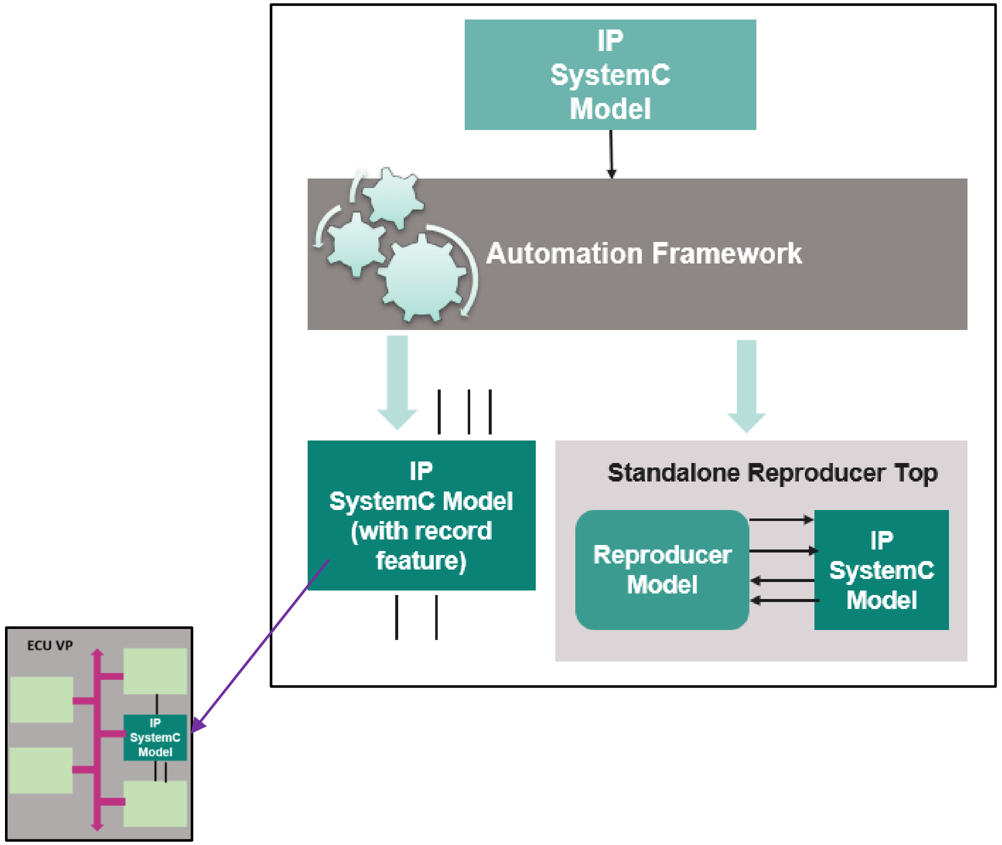


Instrumentation of Logging into IP SystemC model

Generation of Reproducer sequences from Simulation log

Replicating the behaviour in standalone reproducer environment

Implementation



1. Instrumentation Of Logging into IP SystemC model

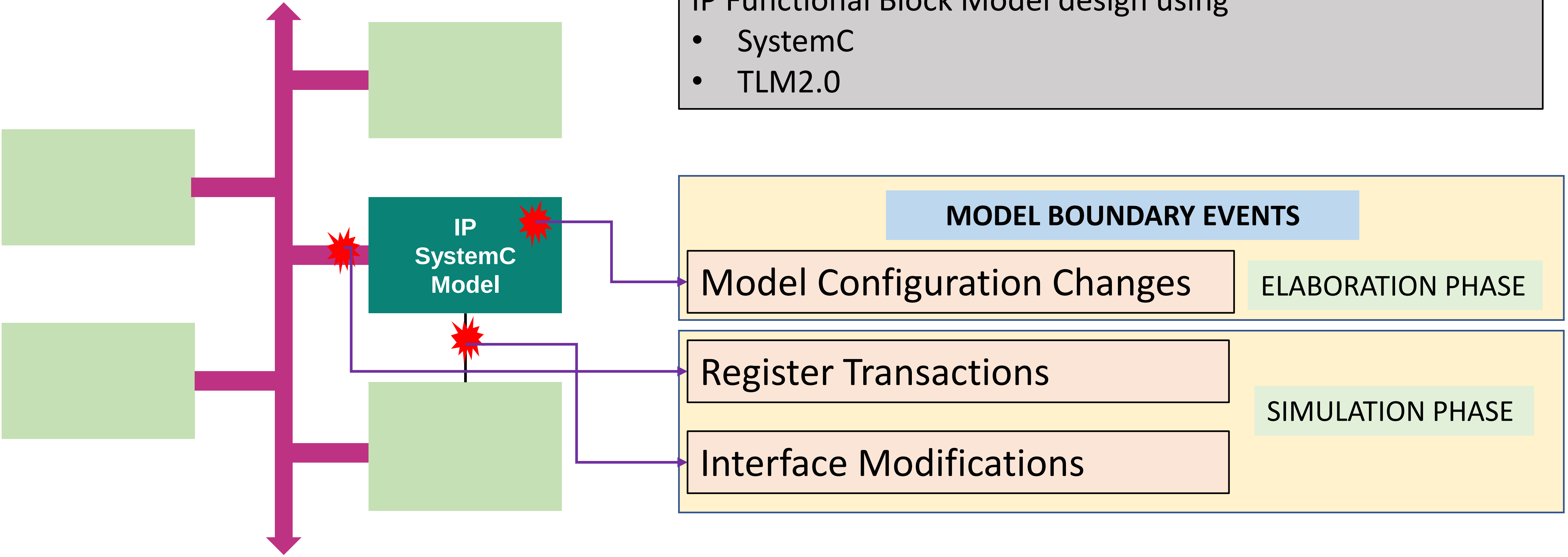
2. Creation of Standalone Reproducer environment

Implemented using In-house Automation Framework

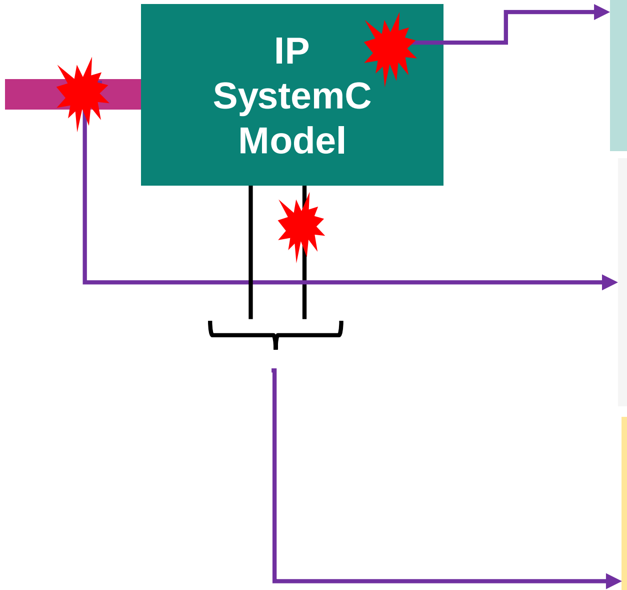
Instrumentation of Logging (1/3)

IP Functional Block Model design using

- SystemC
- TLM2.0



Instrumentation of Logging (2/3)



```
Model Constructor (Model Name, Data Type1 Parameter Name1, Data Type2 Parameter Name2, …)
{
        CONFIGURATION, 1, Data Type1, Parameter Name1, Value1
        CONFIGURATION, 2, Data Type2, Parameter Name2, Value2
        ……
}
```

Configuration Changes

Constructor

```
Register Access Function (Transaction Object)
{
        REGISTER, Simulation Time, Address, Read/Write, Data, Data Length
}
```

Register Transactions

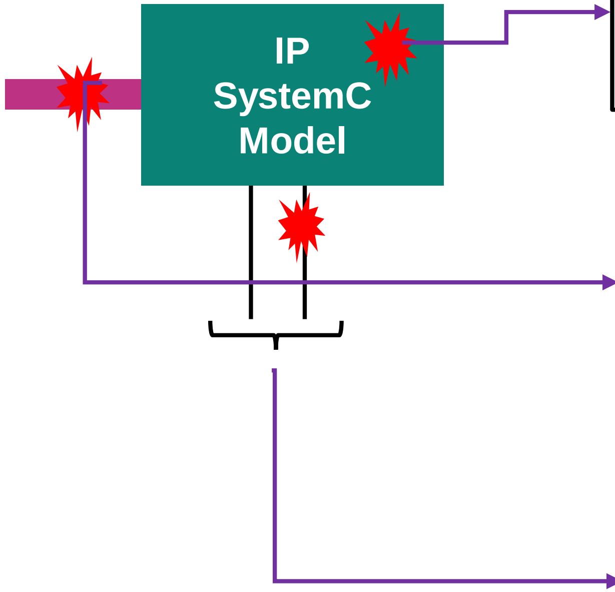
TLM transport function

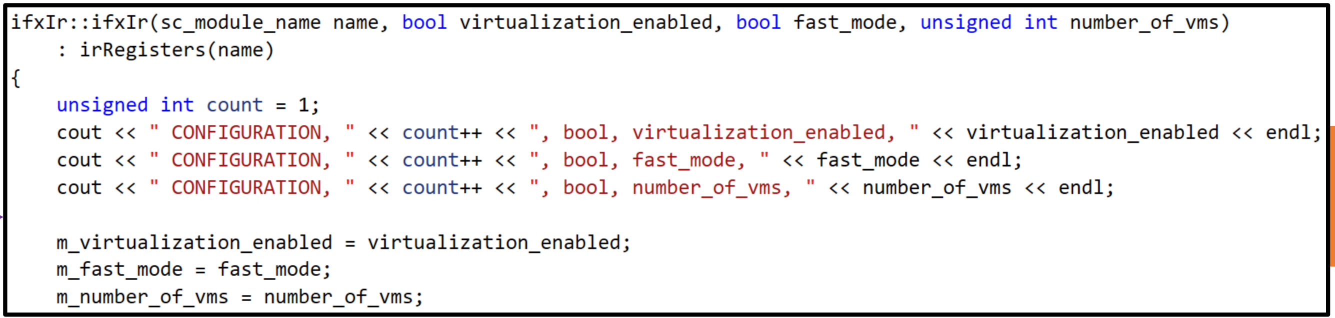
```
SC_METHOD(reset_in_value_changed)
Sensitive << reset_in;
void reset_in _value_changed ()
{
        INTERFACE, Simulation Time, INTERFACE_NAME, TYPE, CURRENT_VALUE
}
```

Interface Modifications

sc_method

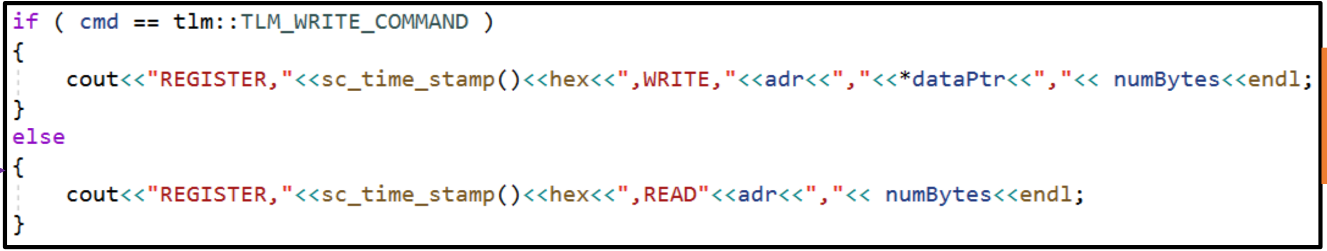
Instrumentation of Logging (3/3)





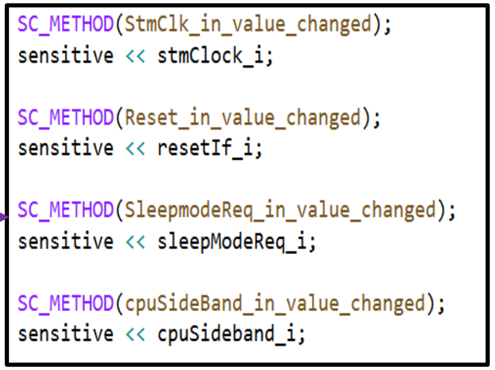
```
ifxIr::ifxIr(sc_module_name name, bool virtualization_enabled, bool fast_mode, unsigned int number_of_vms)
    : irRegisters(name)
{
    unsigned int count = 1;
    cout << " CONFIGURATION, " << count++ << ", bool, virtualization_enabled, " << virtualization_enabled << endl;
    cout << " CONFIGURATION, " << count++ << ", bool, fast_mode, " << fast_mode << endl;
    cout << " CONFIGURATION, " << count++ << ", bool, number_of_vms, " << number_of_vms << endl;

    m_virtualization_enabled = virtualization_enabled;
    m_fast_mode = fast_mode;
    m_number_of_vms = number_of_vms;
```

Configuration Changes

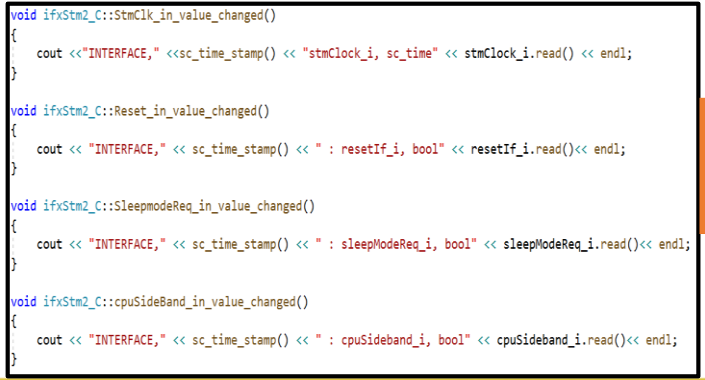


```
if ( cmd == tlm::TLM_WRITE_COMMAND )
{
    cout<<"REGISTER,"<<sc_time_stamp()<<hex<<",WRITE,"<<adr<<","<<*dataPtr<<","<< numBytes<<endl;
}
else
{
    cout<<"REGISTER,"<<sc_time_stamp()<<hex<<",READ"<<adr<<","<< numBytes<<endl;
}
```

Register Transactions



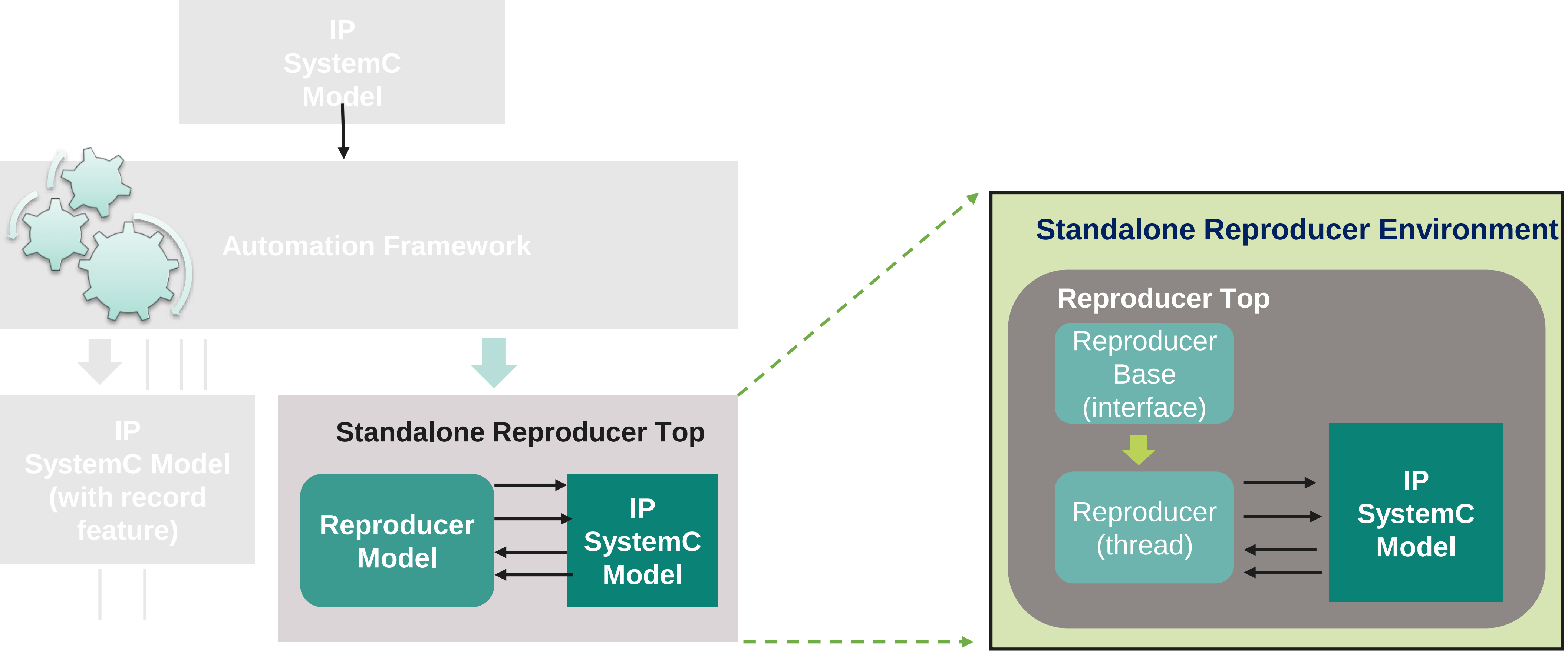
```
SC_METHOD(StmClk_in_value_changed);
sensitive << stmClock_i;

SC_METHOD(Reset_in_value_changed);
sensitive << resetIf_i;

SC_METHOD(SleepmodeReq_in_value_changed);
sensitive << sleepModeReq_i;

SC_METHOD(cpuSideBand_in_value_changed);
sensitive << cpuSideband_i;
```



```
void ifxStm2_C::StmClk_in_value_changed()
{
    cout <<"INTERFACE," <<sc_time_stamp() << "stmClock_i, sc_time" << stmClock_i.read() << endl;
}

void ifxStm2_C::Reset_in_value_changed()
{
    cout << "INTERFACE," << sc_time_stamp() << " : resetIf_i, bool" << resetIf_i.read()<< endl;
}

void ifxStm2_C::SleepmodeReq_in_value_changed()
{
    cout << "INTERFACE," << sc_time_stamp() << " : sleepModeReq_i, bool" << sleepModeReq_i.read()<< endl;
}

void ifxStm2_C::cpuSideBand_in_value_changed()
{
    cout << "INTERFACE," << sc_time_stamp() << " : cpuSideband_i, bool" << cpuSideband_i.read()<< endl;
}
```

Interface Modifications

Creation of Standalone Reproducer environment (1/2)


IP SystemC Model
Automation Framework
IP SystemC Model (with record feature)
Standalone Reproducer Top
Reproducer Model
IP SystemC Model
Standalone Reproducer Environment
Reproducer Top
Reproducer Base (interface)
Reproducer (thread)
IP SystemC Model

Creation of Standalone Reproducer environment(2/2)

Reproducer Base with Complementary Interfaces

```
//IP SystemC Model interfaces
sc_in<sc_time>              Clock_i;
sc_in<bool>                 reset_i;
sc_in<bool>                 sleepMode_i;
sc_in<bool>                 initDone_i;
sc_in<unsigned char>        groupReset_i;
sc_vector<sc_out<bool>> interruptReq1_o;
sc_vector<sc_out<bool>> interruptReq2_o;
sc_out<bool>                alarm_o;
```

```
//Reproducer Model interfaces
sc_out<sc_time>             Clock_o;
sc_out<bool>                reset_o;
sc_out<bool>                sleepMode_o;
sc_out<bool>                initDone_o;
sc_out<unsigned char>       groupReset_o;
sc_vector<sc_in<bool>>      interruptReq1_i;
sc_vector<sc_in<bool>>      interruptReq2_i;
sc_in<bool>                 alarm_i;
```

Reproducer thread to execute Trigger sequences

```
REGISTER, 471170 ns, READ, 0x2460, 0x4
REGISTER, 471220 ns, WRITE, 0x2460, 0x800001, 0x4
REGISTER, 471240 ns, READ, 0x2460, 0x4
REGISTER, 471290 ns, WRITE, 0x2460, 0x4800001, 0x4
```

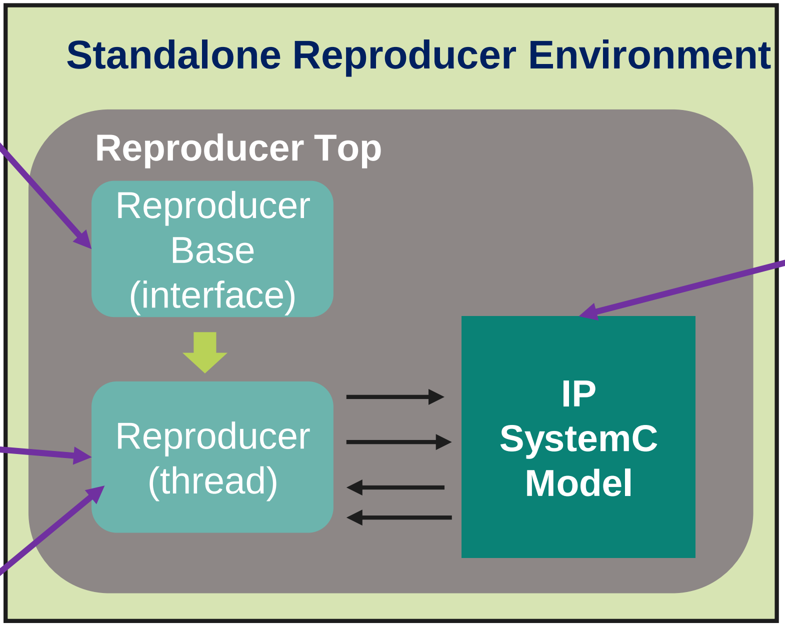
```
wait(sc_time(20, SC_NS));
read(0x2460, read_data, 0x4);
wait(sc_time(50, SC_NS));
write(0x2460, 0x800001, 0x4);
wait(sc_time(20, SC_NS));
read(0x2460, read_data, 0x4);
wait(sc_time(50, SC_NS));
write(0x2460, 0x4800001, 0x4);
```

```
INTERFACE, 471322 ns, ifxIrIspCtrl_i_o[0], unsigned int, 0x1182001
INTERFACE, 471330 ns, ifxIrIspCtrl_i_o[0], unsigned int, 0x1186001
INTERFACE, 471332 ns, ifxIrIspCtrl_i_o[0], unsigned int, 0x1180001
```

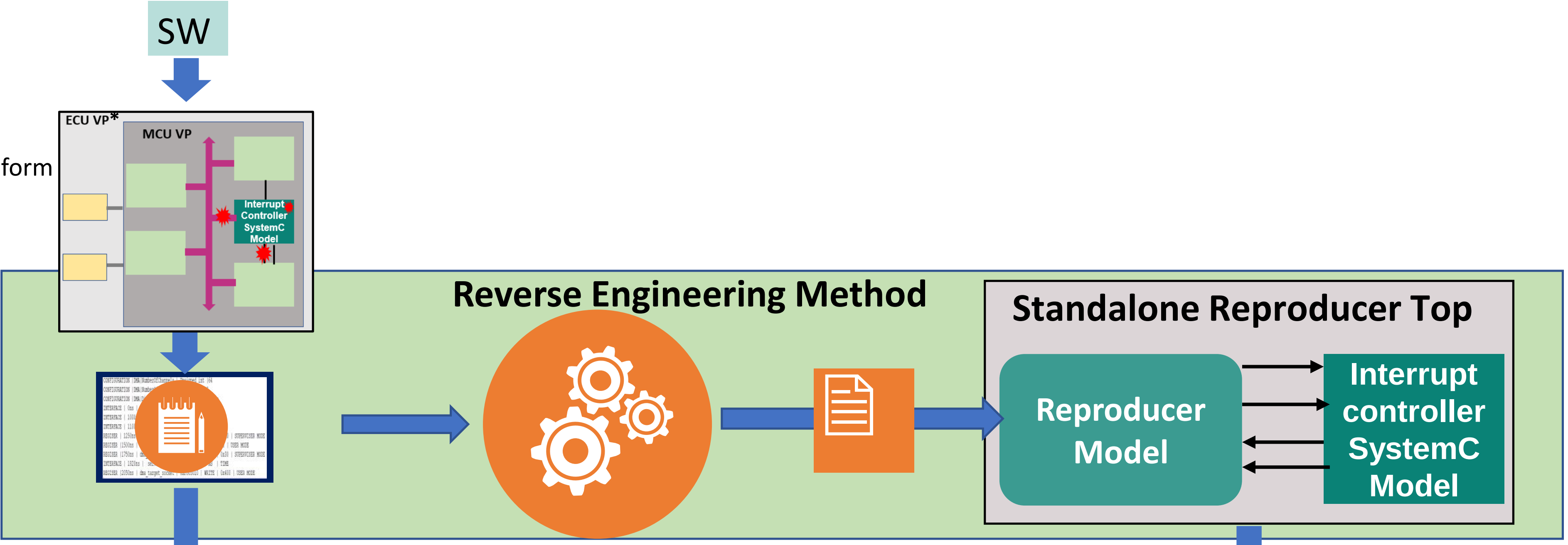
```
ifxIrIspCtrl_i_o[0].write(0x1182001);
wait(sc_time(8, SC_NS));
ifxIrIspCtrl_i_o[0].write(0x1186001);
wait(sc_time(2, SC_NS));
ifxIrIspCtrl_i_o[0].write(0x1180001);
```

```
CONFIGURATION, 1, bool, virtualization_enabled, true
CONFIGURATION, 2, bool, fast_mode, false
CONFIGURATION, 3, unsigned int, number_of_vms, 8
```

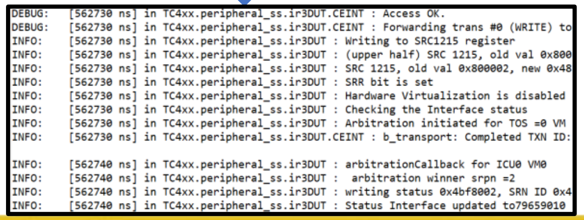
```
bool virtualization_enabled = true
bool fast_mode = false
unsigned int number_of_vms = 8
ifxIr * irPtr = new ifxIr("IR", virtualization_enabled, fast_mode, number_of_vms);
```


Standalone Reproducer Environment
Reproducer Top
Reproducer Base (interface)
Reproducer (thread)
IP SystemC Model

Proof of Concept using AURIX^{TM} Interrupt Controller

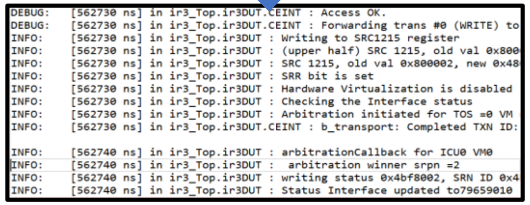

SW
ECU VP*
MCU VP
Interrupt Controller SystemC Model
Reverse Engineering Method
Standalone Reproducer Top
Reproducer Model
Interrupt controller SystemC Model

* ECU like Virtual Platform



```
DEBUG:   [562730 ns] in TC4xx.peripheral_ss.ir3DUT.CEINT : Access OK.
DEBUG:   [562730 ns] in TC4xx.peripheral_ss.ir3DUT.CEINT : Forwarding trans #0 (WRITE) to
INFO:    [562730 ns] in TC4xx.peripheral_ss.ir3DUT : Writing to SRC1215 register
INFO:    [562730 ns] in TC4xx.peripheral_ss.ir3DUT : (upper half) SRC 1215, old val 0x800
INFO:    [562730 ns] in TC4xx.peripheral_ss.ir3DUT : SRC 1215, old val 0x800002, new 0x48
INFO:    [562730 ns] in TC4xx.peripheral_ss.ir3DUT : SRR bit is set
INFO:    [562730 ns] in TC4xx.peripheral_ss.ir3DUT : Hardware Virtualization is disabled
INFO:    [562730 ns] in TC4xx.peripheral_ss.ir3DUT : Checking the Interface status
INFO:    [562730 ns] in TC4xx.peripheral_ss.ir3DUT : Arbitration initiated for TOS =0 VM
INFO:    [562730 ns] in TC4xx.peripheral_ss.ir3DUT.CEINT : b_transport: Completed TXN ID:

INFO:    [562740 ns] in TC4xx.peripheral_ss.ir3DUT : arbitrationCallback for ICU0 VM0
INFO:    [562740 ns] in TC4xx.peripheral_ss.ir3DUT :  arbitration winner srpn =2
INFO:    [562740 ns] in TC4xx.peripheral_ss.ir3DUT : writing status 0x4bf8002, SRN ID 0x4
INFO:    [562740 ns] in TC4xx.peripheral_ss.ir3DUT : Status Interface updated to79659010
```

=



```
DEBUG:   [562730 ns] in ir3_Top.ir3DUT.CEINT : Access OK.
DEBUG:   [562730 ns] in ir3_Top.ir3DUT.CEINT : Forwarding trans #0 (WRITE) to
INFO:    [562730 ns] in ir3_Top.ir3DUT : Writing to SRC1215 register
INFO:    [562730 ns] in ir3_Top.ir3DUT : (upper half) SRC 1215, old val 0x800
INFO:    [562730 ns] in ir3_Top.ir3DUT : SRC 1215, old val 0x800002, new 0x48
INFO:    [562730 ns] in ir3_Top.ir3DUT : SRR bit is set
INFO:    [562730 ns] in ir3_Top.ir3DUT : Hardware Virtualization is disabled
INFO:    [562730 ns] in ir3_Top.ir3DUT : Checking the Interface status
INFO:    [562730 ns] in ir3_Top.ir3DUT : Arbitration initiated for TOS =0 VM
INFO:    [562730 ns] in ir3_Top.ir3DUT.CEINT : b_transport: Completed TXN ID:

INFO:    [562740 ns] in ir3_Top.ir3DUT : arbitrationCallback for ICU0 VM0
INFO:    [562740 ns] in ir3_Top.ir3DUT :  arbitration winner srpn =2
INFO:    [562740 ns] in ir3_Top.ir3DUT : writing status 0x4bf8002, SRN ID 0x4
INFO:    [562740 ns] in ir3_Top.ir3DUT : Status Interface updated to79659010
```

Next Steps

- Implementation for all models of Infineon next generation automotive microcontrollers
- Adaptation of the methodology for
  - serial communication interfaces
  - bus master interfaces
- Configuration to enable/disable the instrumentation of logging
  - Instrumentation should be enabled only for debugging purpose
  - For a normal simulation, Instrumentation should be disabled, as it degrades the performance

Questions