life.augmented

Politecnico di Torino

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

A Hybrid Approach To Interrupt Verification

Giovanni Auditore, Francesco Rua' (STMicroelectronics)

Qibo Peng (Politecnico di Torino)

accellera SYSTEMS INITIATIVE

Introduction

Target: Peripheral interrupts

Purpose: Improve reliability of previous methods

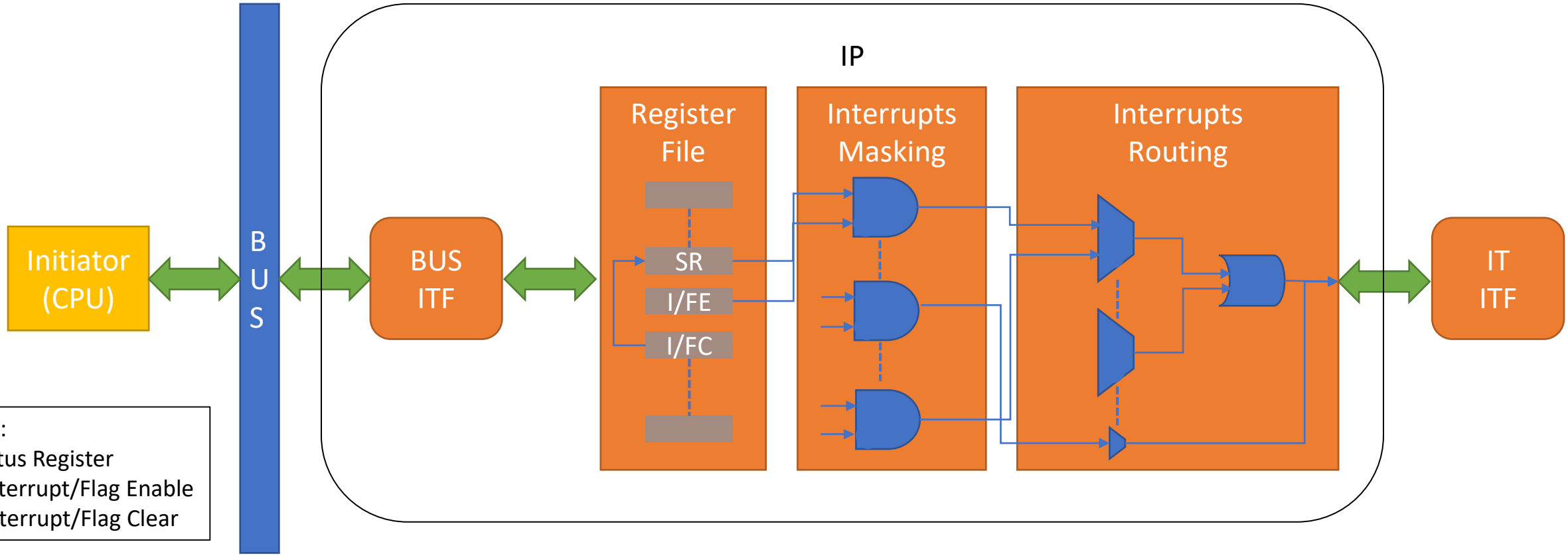
Verification scope: IP level

Applied methodology: Hybrid

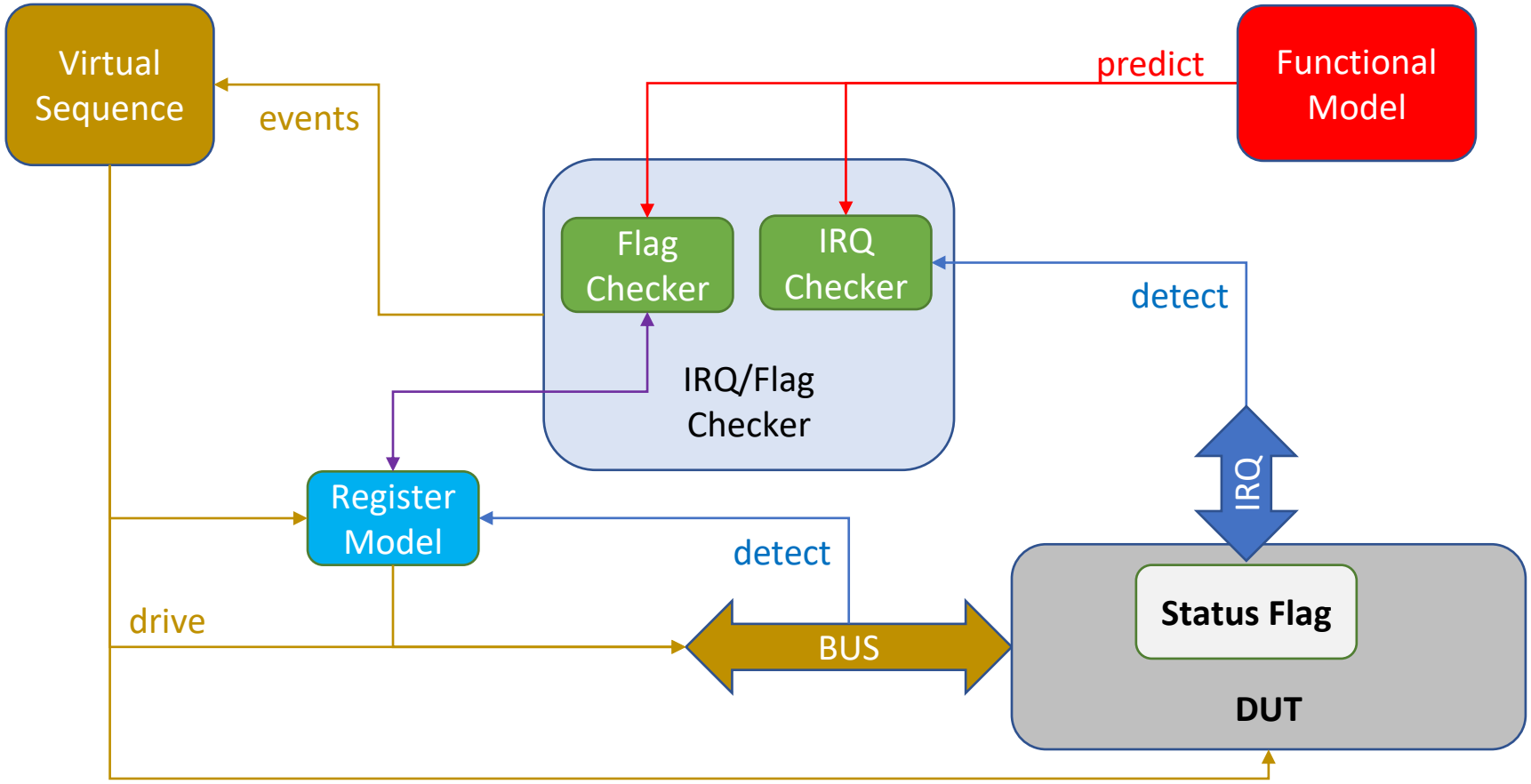
Keywords:

- Quality
- Automation
- Integration

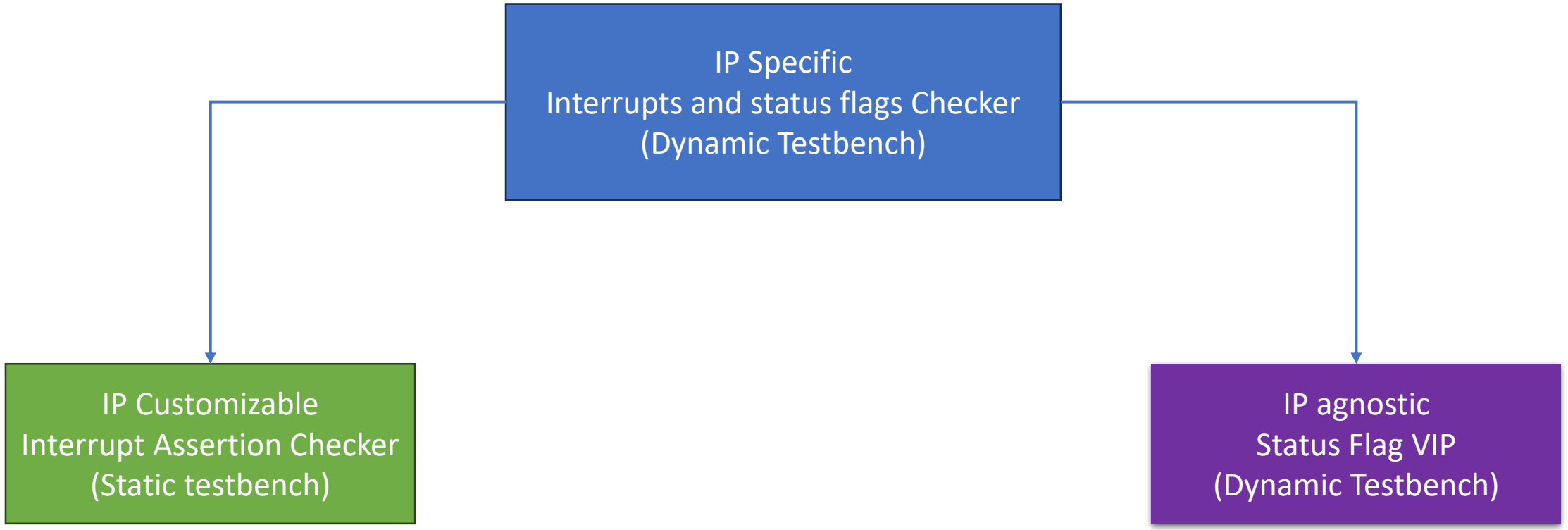
IP Interrupts Architecture



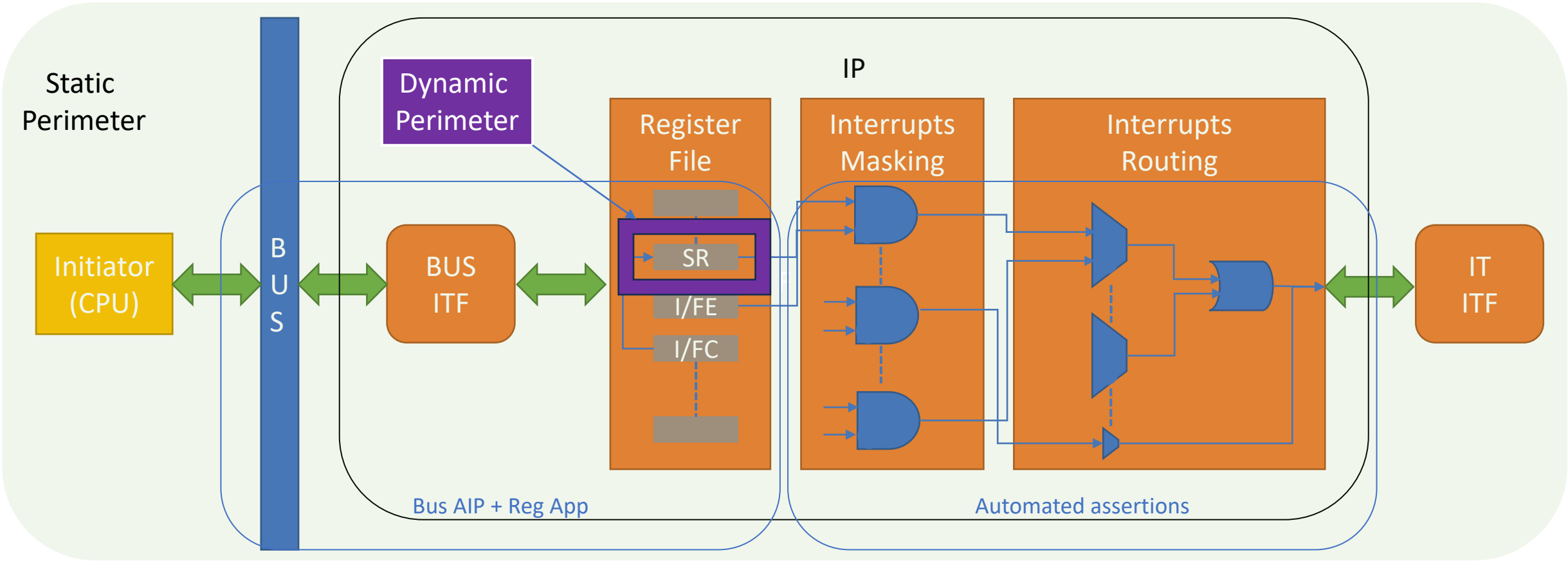
Legacy dynamic verification architecture



Divide and Conquer Strategy



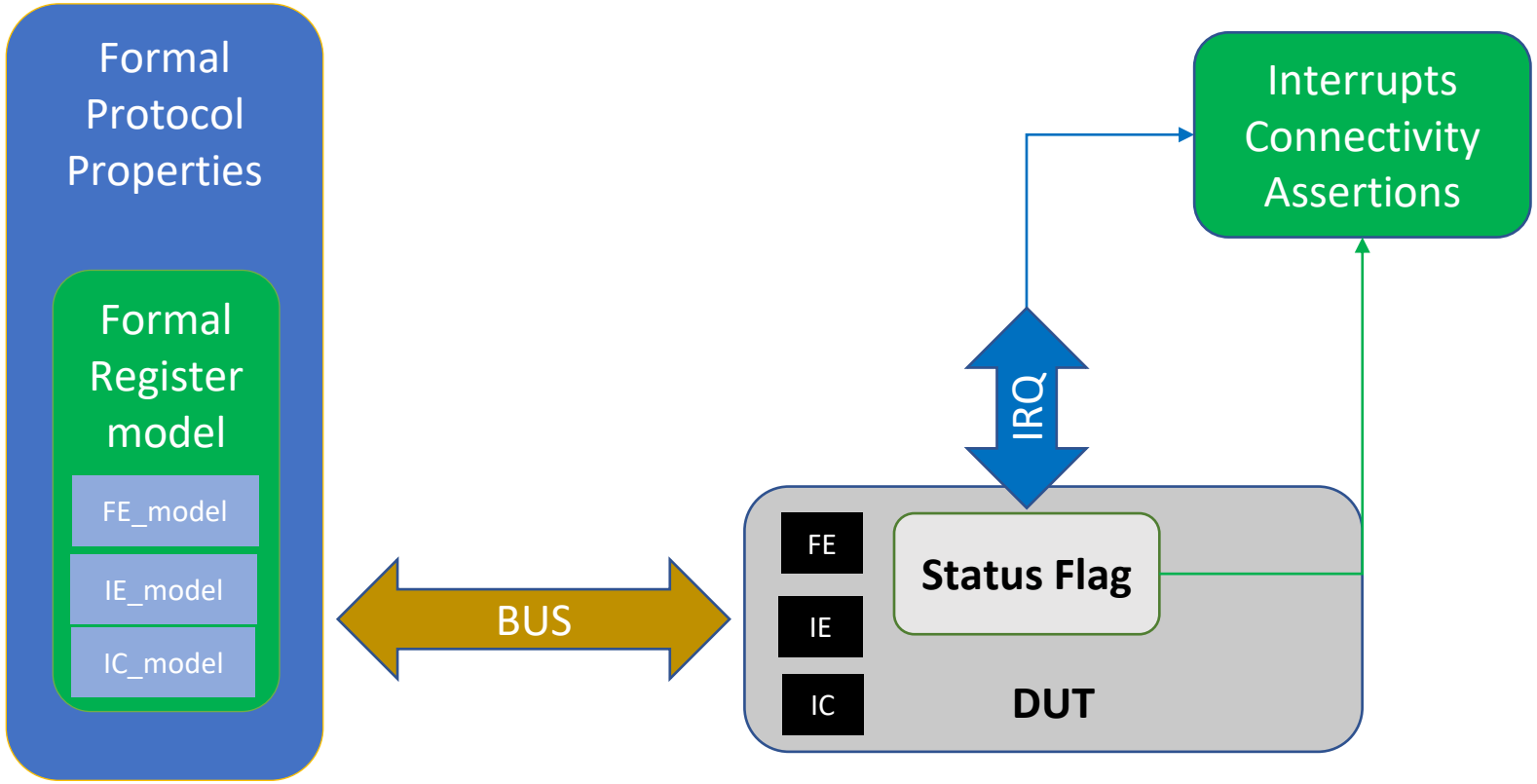
Verification Partitioning



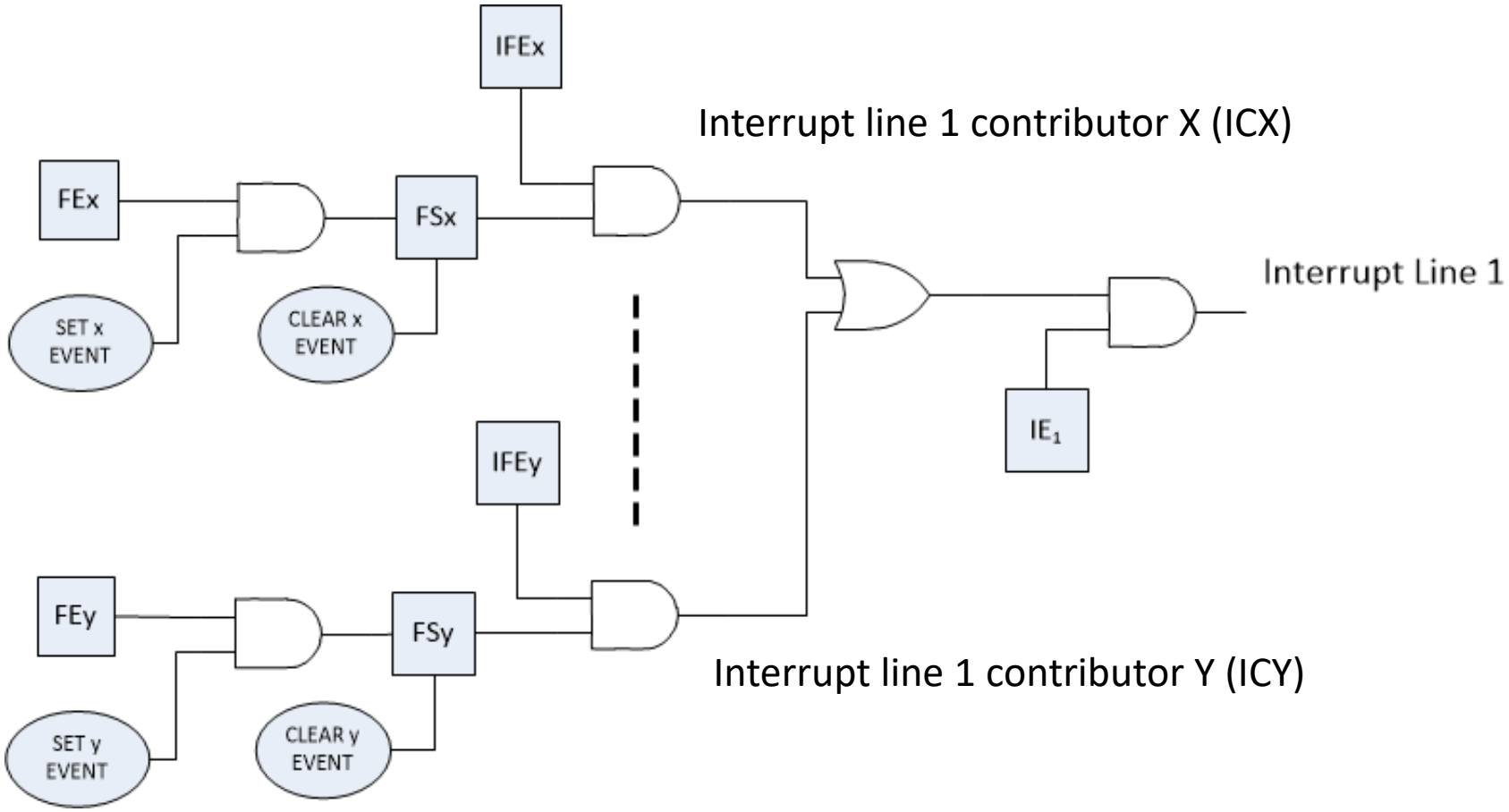
Automation and Reuse Strategy

- Effective
- Reusable
- User friendly
- Standards compliant
- Incremental
- Independent

Interrupts Static Checker



Generic Interrupt Line Architecture

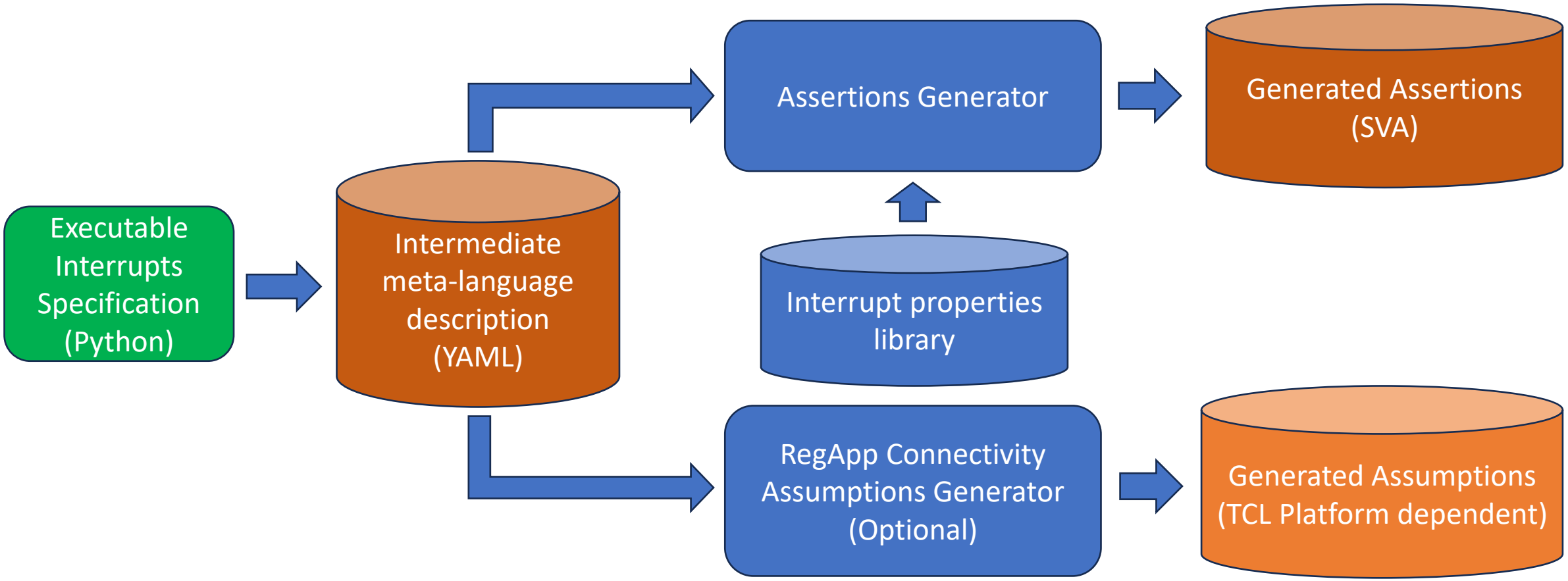


Executable Specification

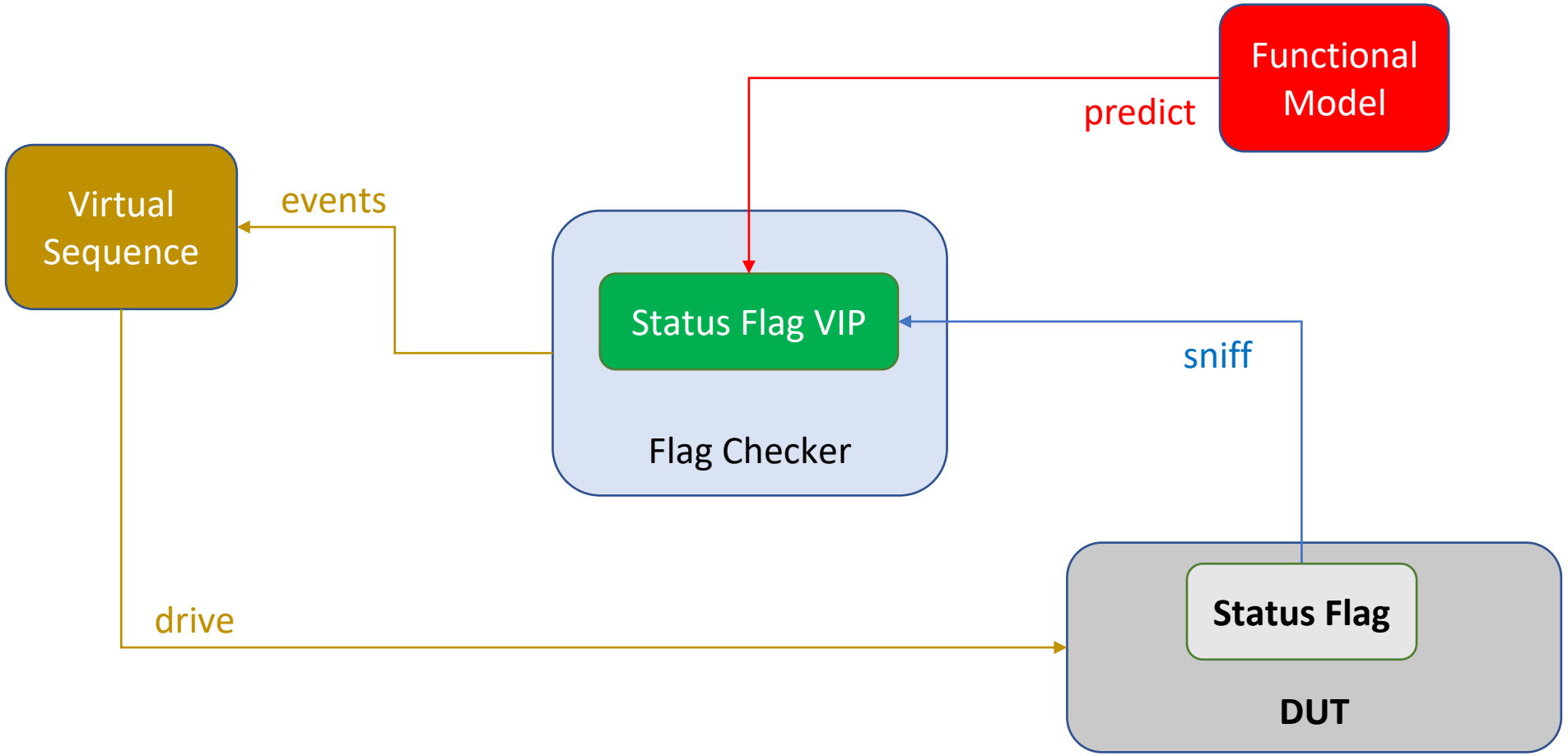
Register fields modelled in regApp Syntax: $(REG.FIELD)

```
interrupts['it_line_1']  = interruptLine(hdl_path = 'dut.it_1', it_enable = '$(GIER.IE1)')

interrupts['it_line_1']['contributors']['ICx'] = interruptContributor(      hdl_path = 'dut.i_reg.FSx',
                                                                         it_flag_enable = '$(IER.IFEx)',
                                                                            flag_enable = '$(FER.FEx)',
                                                                              set_event = 'dut.i_kernel.setX_evt',
                                                                            clear_event = 'dut.i_kernel.clrX_evt',
                                                                   simultaneous_set_clear = 'clear' )

interrupts['it_line_1']['contributors']['ICy'] = interruptContributor(      hdl_path = 'dut.i_reg.FSy',
                                                                         it_flag_enable = '$(IER.IFEy)',
                                                                            flag_enable = '$(FER.FEy)',
                                                                              set_event = 'dut.i_kernel.setY_evt',
                                                                            clear_event = 'dut.i_kernel.clrY_evt',
                                                                   simultaneous_set_clear = 'set' )
```

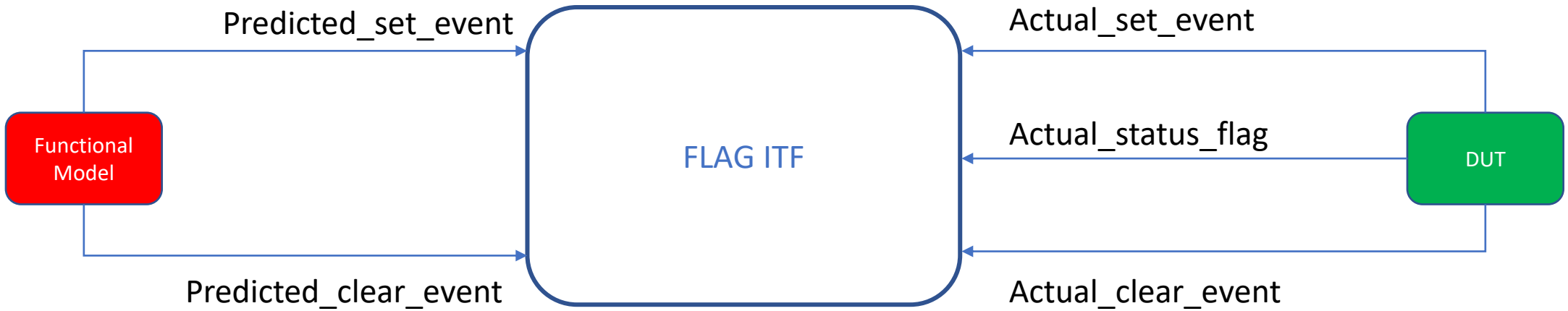
Static Flow Automation



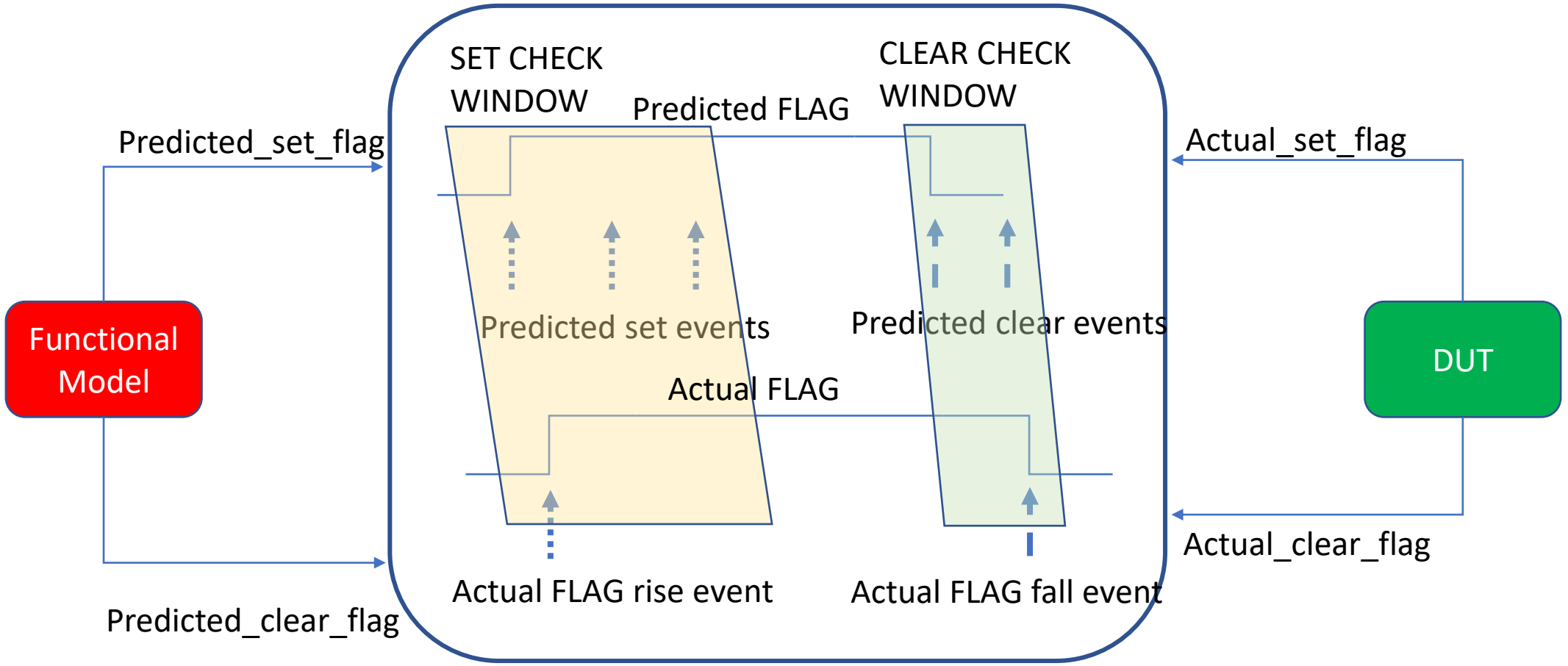
Dynamic Status Flag VIP



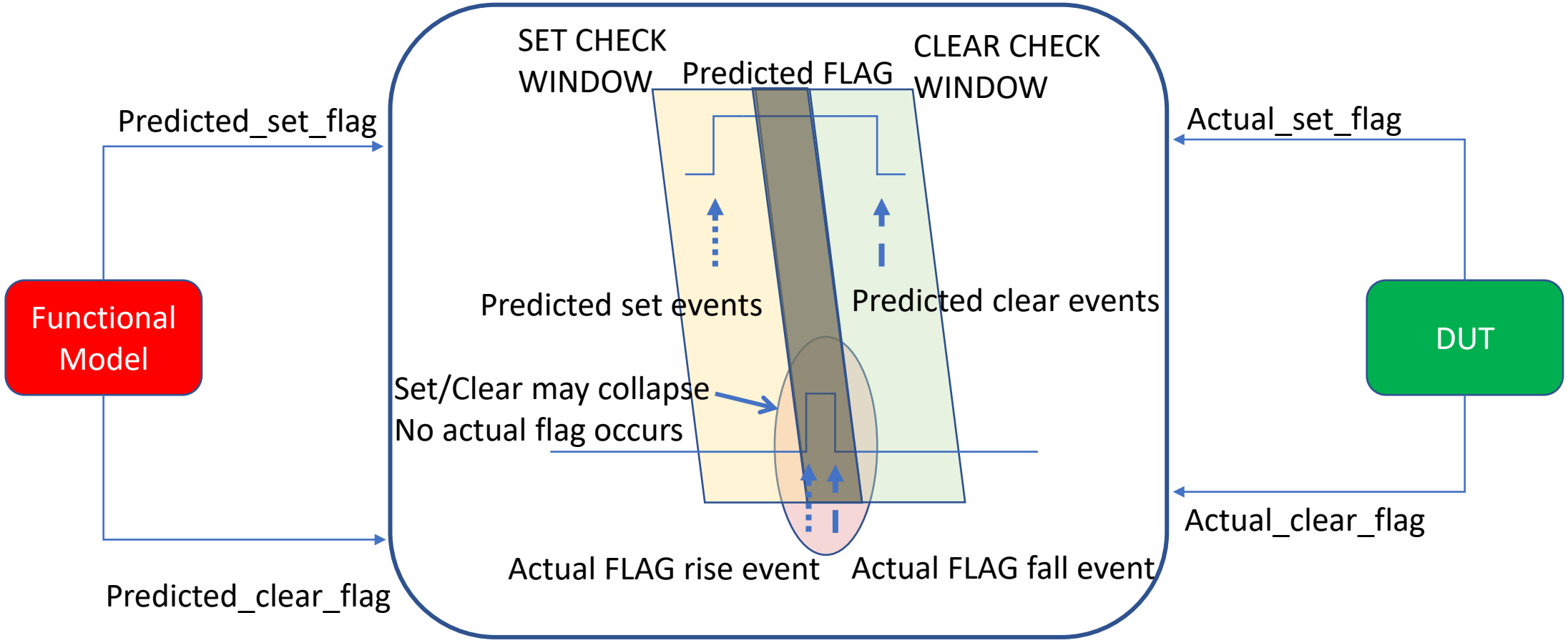
Flag Interface



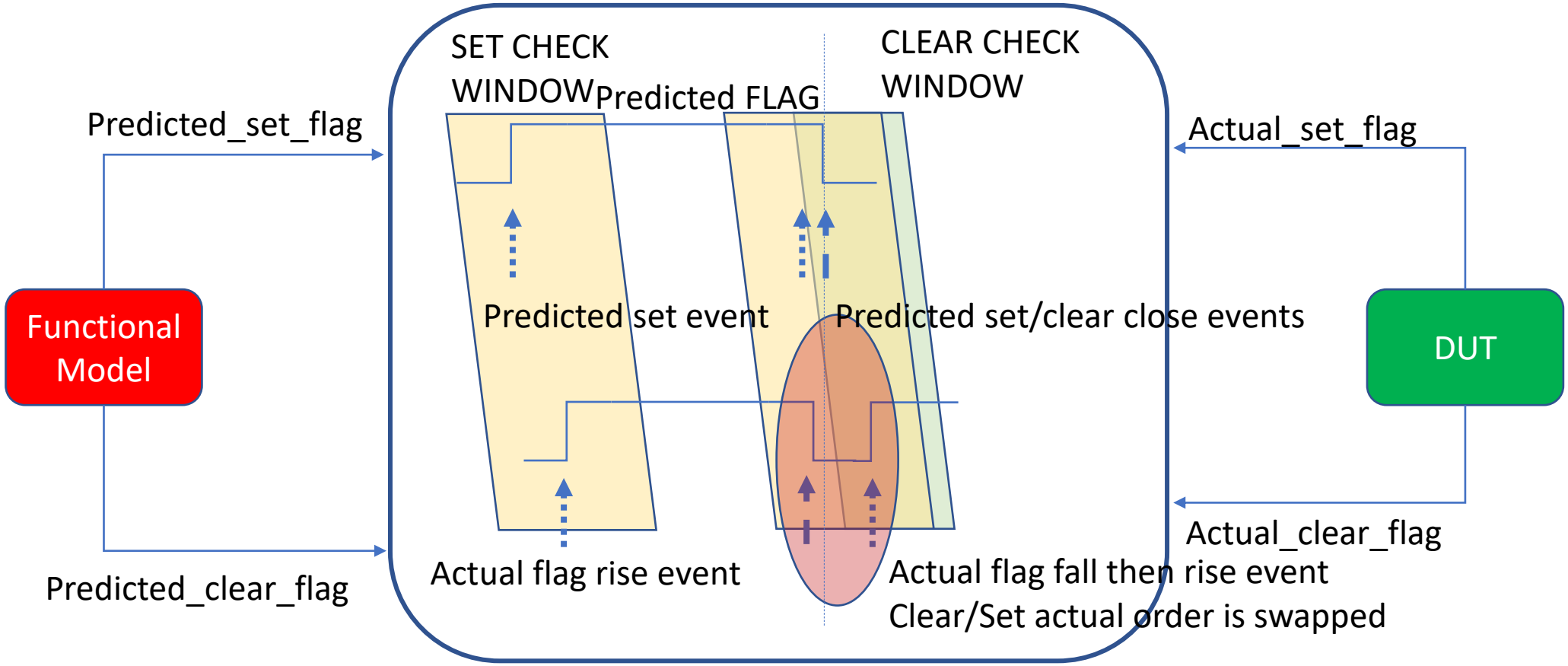
Typical Scenario



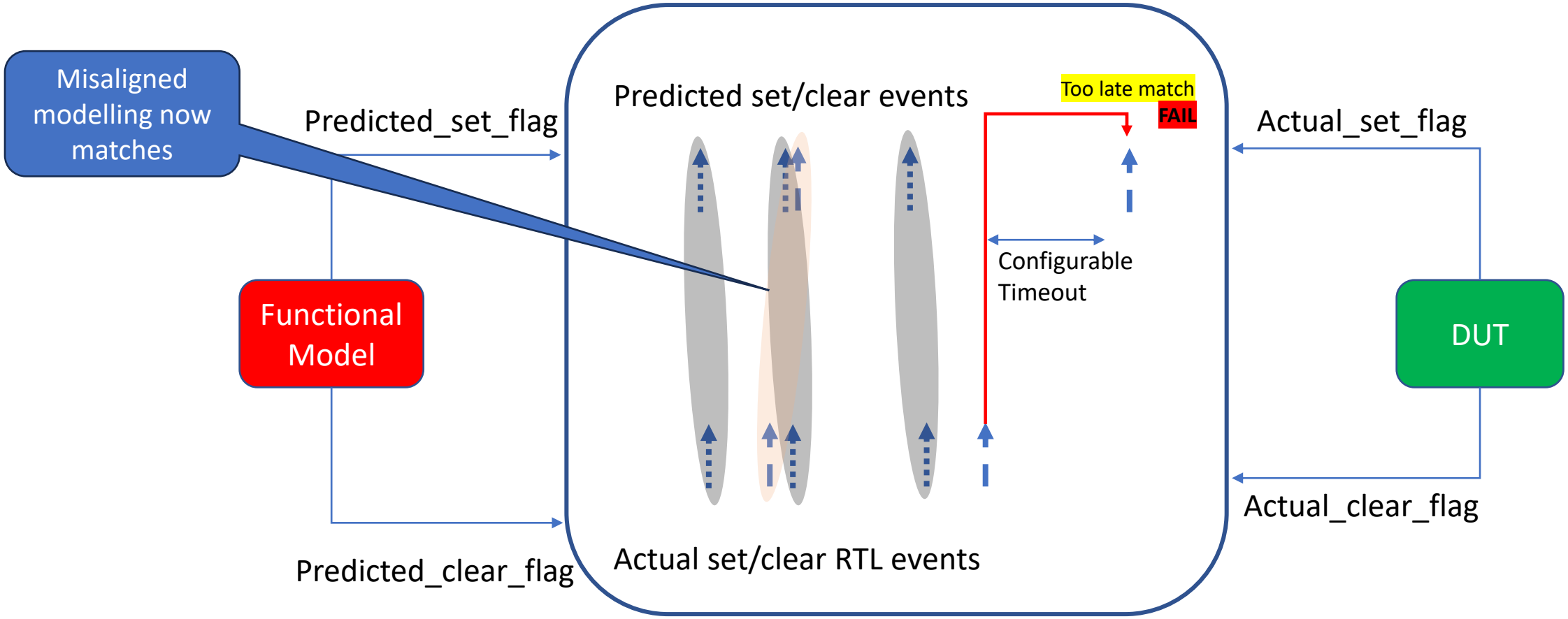
Corner Scenario 1



Corner Scenario 2



Out-of-order Scoreboard with Timeout



Future Development

- Extend the automation flow also to the dynamic VIP code generation.
- Contact EDA Vendors to request native support of the regApp connectivity flow inside the product release of the regApp itself.

Conclusion

A solution for peripheral interrupt line verification

- **Formalizes** an executable description of the interrupt architecture
- **Reduces** the development time
- **Extends** the domain of applicability
- **Reduces** the risks of false failures in dynamic verification
- **Exploits** commercial App register models

Robust

Fully reusable

Widely applicable

Questions