Unified Functional Safety Verification Platform for ISO 26262-Compliant Automotive

Joerg Richter
Group Director, R&D, Verification Group

SYNOPSYS®

accellera SYSTEMS INITIATIVE

2019 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE

Impact of Functional Safety

2009–11 Toyota vehicle recalls

From Wikipedia, the free encyclopedia

The **2009-11 Toyota vehicle recalls** involved three separate but related recalls of automobiles by Toyota Motor Corporation occurred at the end of 2009 and start of 2010. Toyota initiated the recalls, the first two with the assistance of the U.S. National Highway Traffic Safety Administration (NHTSA), after reports that several vehicles experienced unintended acceleration. The first recall, on November 2, 2009, was to correct a possible incursion of an incorrect or out-of-place front driver's side floor mat into the foot pedal well, which can cause pedal entrapment. The second recall, on January 21, 2010, was begun after some crashes were shown not to have been caused by floor mat incursion. This latter defect was identified as a possible mechanical sticking of the accelerator pedal causing unintended acceleration, referred to as *Sticking Accelerator Pedal* by Toyota.

"…that Toyota did not follow best practices for real time life critical software, and that a **single bit flip** which can be caused by cosmic rays could cause unintended acceleration."

electrical." This included sticking accelerator pedals, and pedals caught under floor mats.[29]

However, on October 24, 2013, a jury ruled against Toyota and found that unintended acceleration could have been caused due to deficiencies in the drive-by-wire throttle system or Electronic Throttle Control System (ETCS). Michael Barr of the Barr Group testified that NASA had not been able to complete its examination of Toyota's ETCS and that Toyota did not follow best practices for real time life critical software, and that a single bit flip which can be caused by cosmic rays could cause unintended acceleration. As well, the run-time stack of the real-time operating system was not large enough and that it was possible for the stack to grow large enough to overwrite data that could cause unintended acceleration.[30][31] As a result, Toyota has entered into settlement talks with its plaintiffs.[32]

Source: Wikipedia

Unified Functional Safety Verification Platform

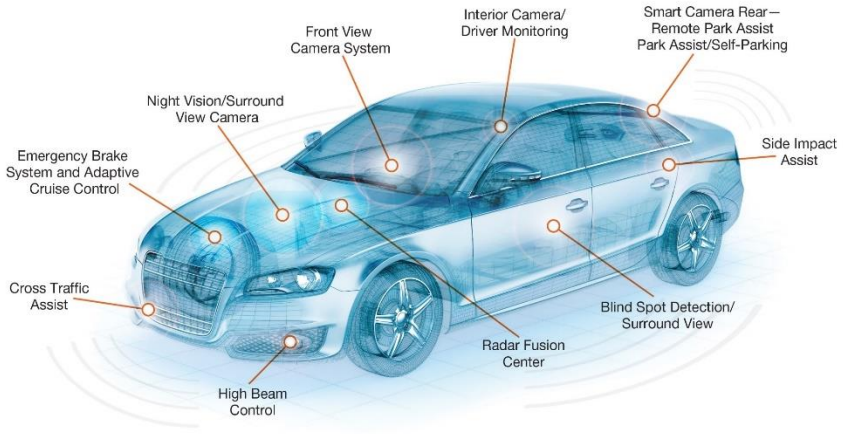
FUNCTIONAL SAFETY OVERVIEW

What is Functional Safety?

- Functional Safety is the "Absence of unreasonable risk due to *hazards* caused by malfunctioning behavior of Electrical/Electronic systems" [ISO 26262]

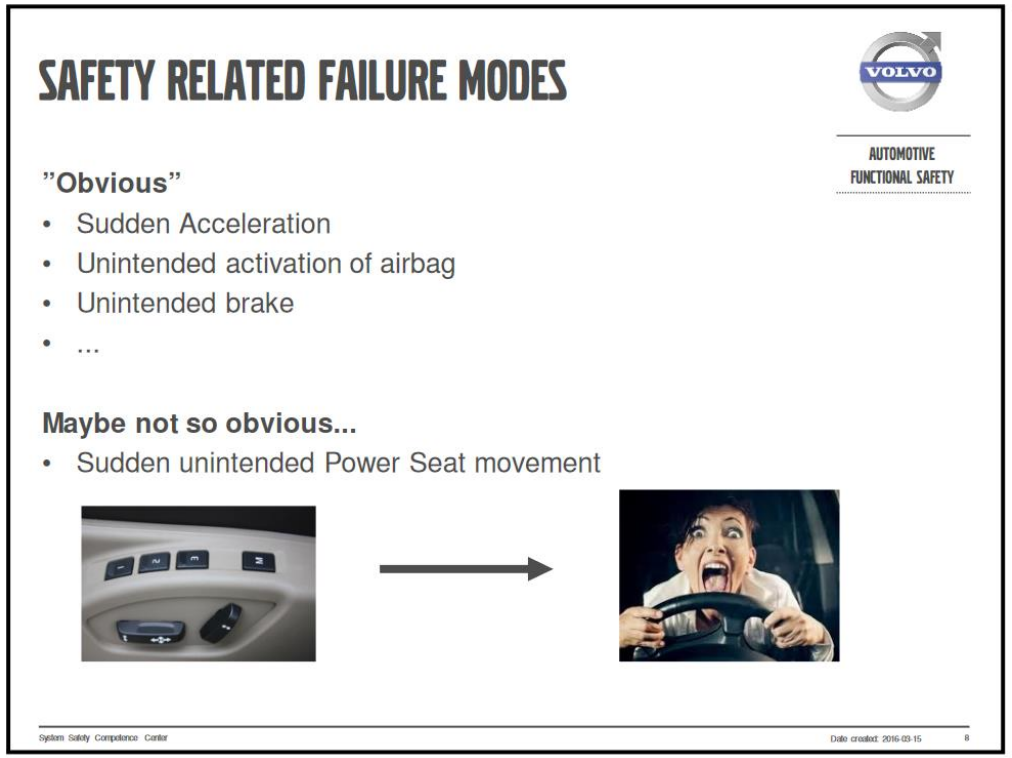
- Functional safety means that potentially dangerous conditions are detected, activating preventative or corrective mechanisms to stop or mitigate the hazardous event

- Functional safety is critical to many markets: Automotive, Aerospace, Medical, etc.



What is Functional Safety?

- Functional Safety is the "Absence of unreasonable risk due to *hazards* caused by malfunctioning behavior of Electrical/Electronic systems" [ISO 26262]



Source: Volvo

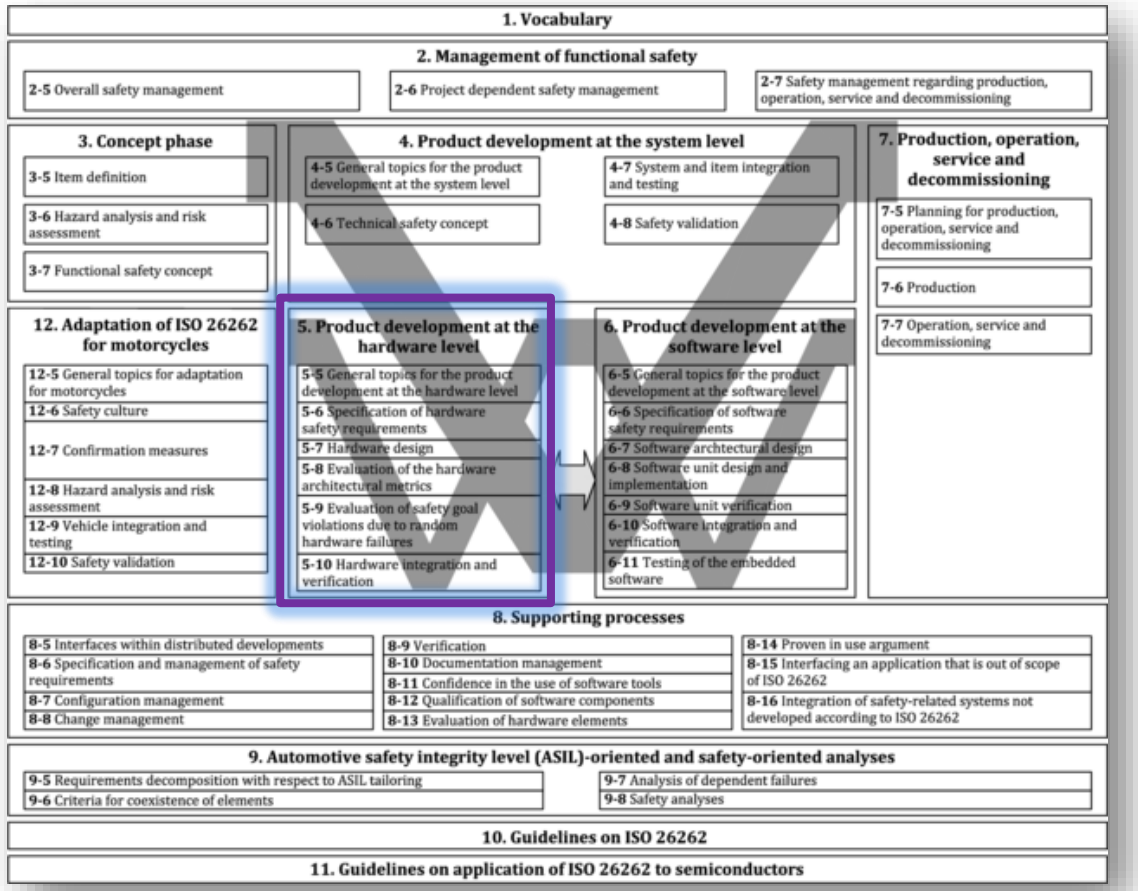
ISO 26262 in numbers

1st edition released in 2011
- 10 parts
- 43 chapters
- 100 work products
- 180 Development methods
- 500 Pages
- 600 Requirements

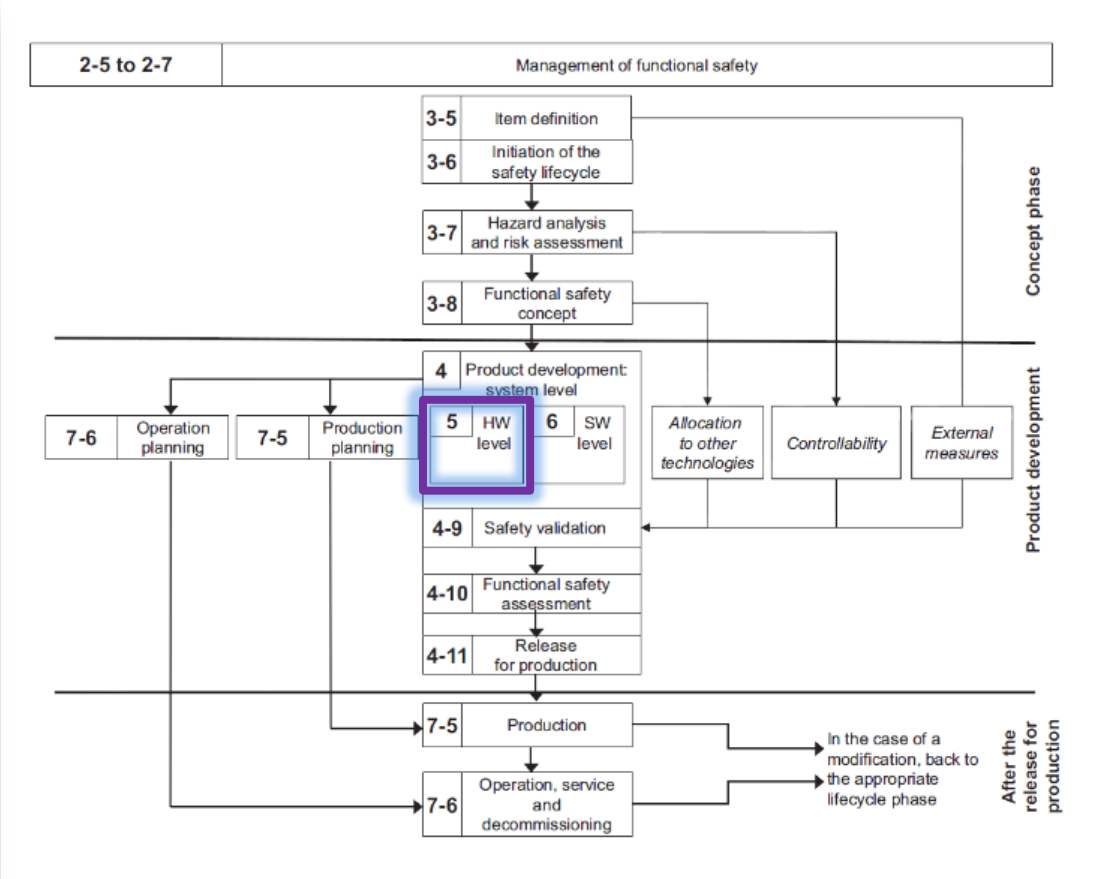
2nd edition released end of 2018
- 12 parts
- …

What is Functional Safety in ISO 26262?

"Absence of unacceptable risk due to hazards caused by malfunctioning behavior of electrical and/or electronic systems."

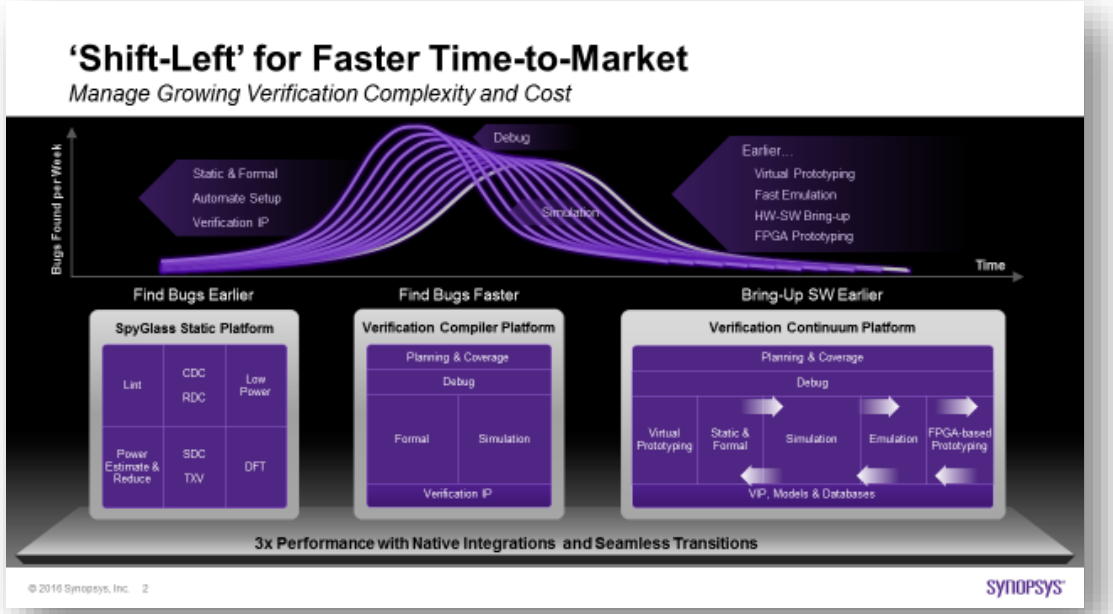


Organization View



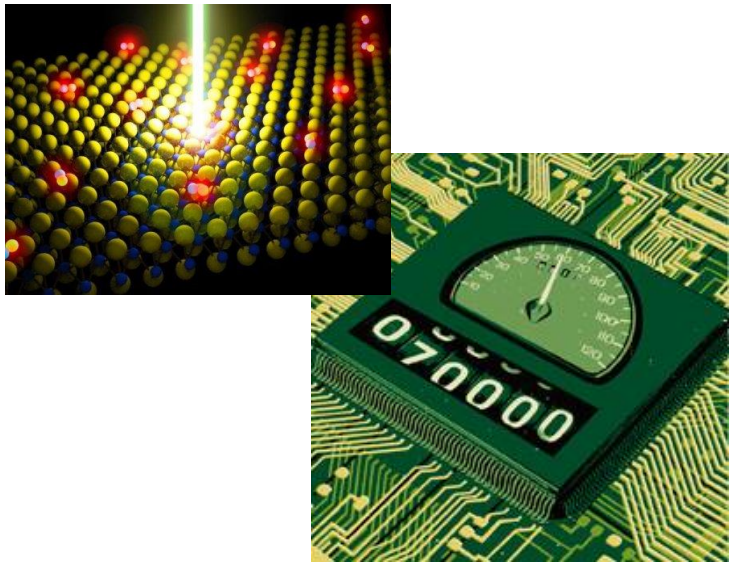
Lifecycle Flow View

Functional Verification
Prevent / Eliminate Bugs



Find and Fix <u>Systematic Faults</u>
Design Bugs that Cause Incorrect Operation
Always Permanent

Functional Safety Verification
Detect / Control Failures



Ensure Proper Handling of <u>Random Faults</u>
Hardware Defects from Aging or Environmental Factors
May be Permanent or Transient

ISO 26262 Requirements – Hardware Development

Show that design functionality is correct, works properly in the context of the system, and is safe

Prevent / Eliminate Bugs

- Validate *Functional Correctness* of the design
- Use best-in-class *Functional Verification* methodology and tools

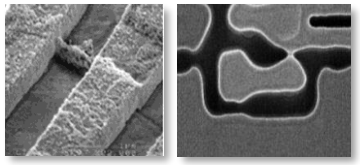
Systematic Faults – Design Bugs

Always permanent

Reduced DPPM

- DFT
- Functional patterns

Random Faults

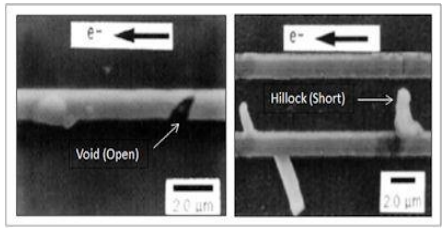


Permanent

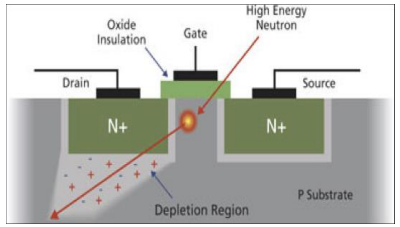
Detect / Control Failures

- Effectiveness of *Safety Mechanisms* to handle faulty behavior
- Assessed by *Functional* ***Safety*** *Verification* methodology and tools

Random Faults – HW Failures



Permanent



Transient

Development → **Manufacturing** → **In Operation**

Lifecycle of Component / System / Automobile

ISO 26262 Requirements – Hardware Development

Show that design functionality is correct, works properly in the context of the system, and is safe

Prevent / Eliminate Bugs

- Validate *Functional Correctness* of the design
- Use best-in-class *Functional Verification* methodology and tools

Systematic Faults – Design Bugs

Always permanent

Reduced DPPM

- DFT
- Functional patterns

Random Faults

Permanent

Detect / Control Failures

- Effectiveness of *Safety Mechanisms* to handle faulty behavior
- Assessed by *Functional **Safety** Verification* methodology and tools

Random Faults – HW Failures

Permanent *Transient*

Development | **Manufacturing** | **In Operation**

Lifecycle of Component / System / Automobile

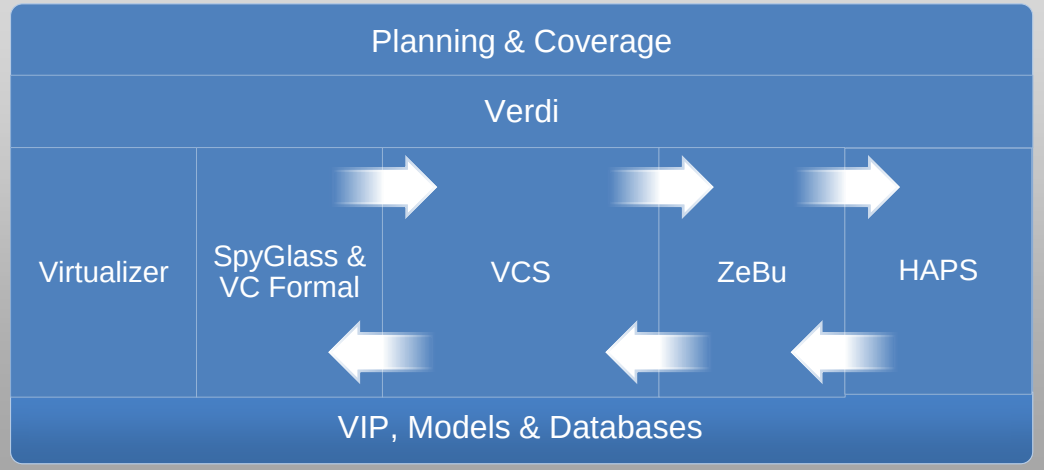
Functional Verification is Essential Starting Point

Prevent / Eliminate Bugs

Avoid Systematic Faults – Design Bugs (Permanent Faults)

Verification & Validation:
Use State of the Art Functional Verification methodology

Verification Continuum Platform

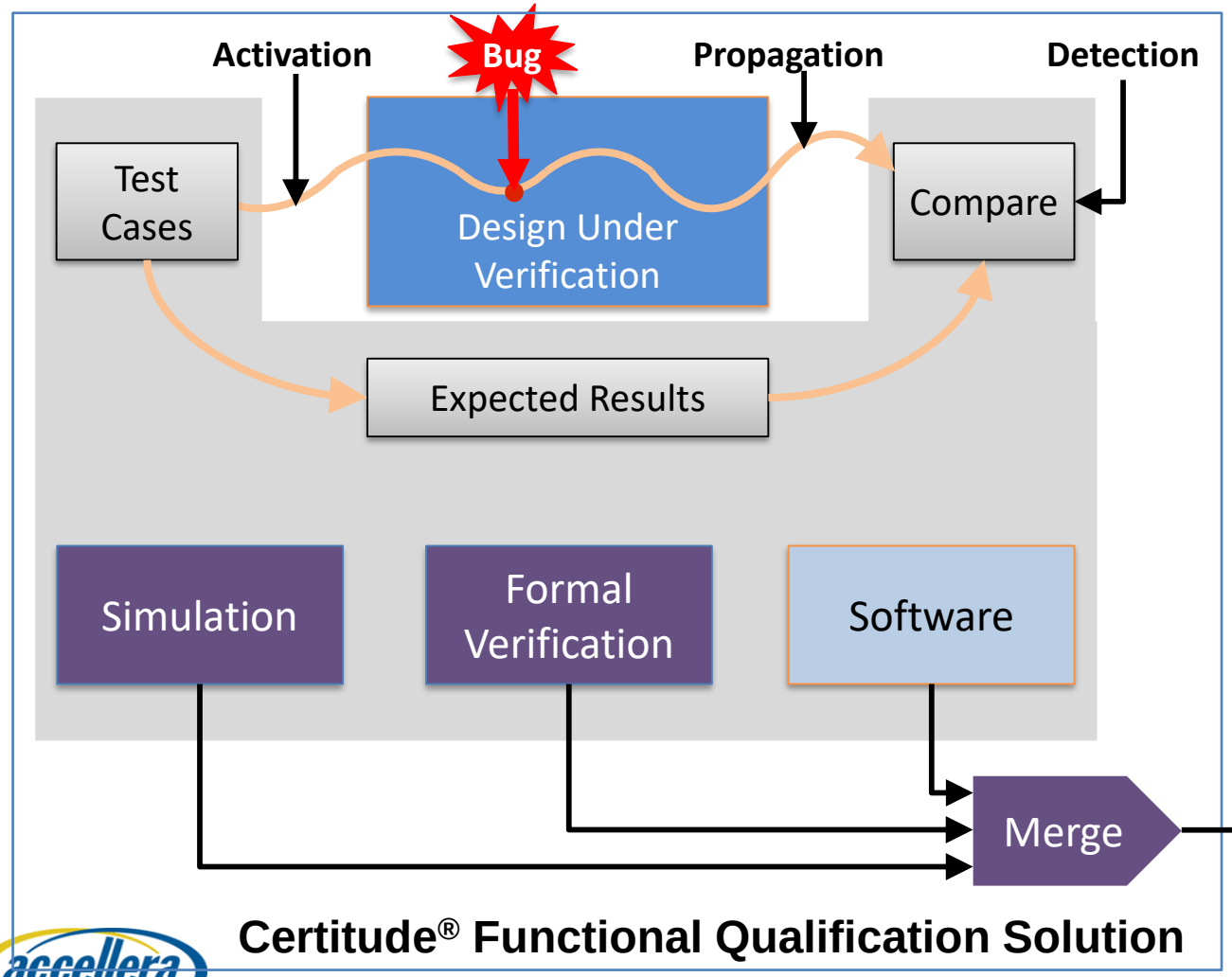


Synopsys Functional Verification Technology Platform

- Many technologies must be used to ensure the highest functional verification quality
- Verification quality analysis provides objective measure of functional verification effectiveness

Demonstrate Verification Flows are Robust

Evidence-based verification quality analysis for ISO 26262 Part 8-9 assessments



*"Risk of **systematic faults** […] is **minimized**"*

Inject and qualify systematic faults at architecture, system, and RT level

Measure verification completeness and functional correctness of design

Natively integrated with VCS and VC Formal, and works with C/C++/SystemC flows

Unified functional verification environment quality metrics

ISO 26262 Requirements – Hardware Development

Show that design functionality is correct, works properly in the context of the system, and is safe

Prevent / Eliminate Bugs

- Validate *Functional Correctness* of the design
- Use best-in-class *Functional Verification* methodology and tools

Systematic Faults – Design Bugs

Always permanent

Reduced DPPM

- DFT
- Functional patterns

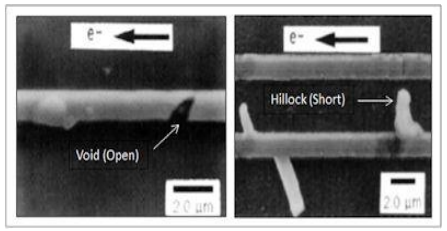
Random Faults

Permanent

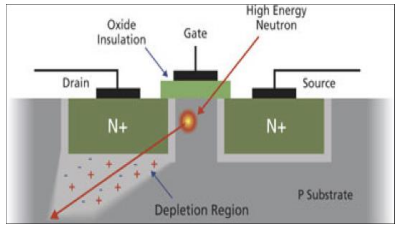
Detect / Control Failures

- Effectiveness of *Safety Mechanisms* to handle faulty behavior
- Assessed by *Functional* ***Safety*** *Verification* methodology and tools

Random Faults – HW Failures



Permanent



Transient

Development → **Manufacturing** → **In Operation**

Lifecycle of Component / System / Automobile

Hazard Analysis and Risk Assessment

Automotive Safety Integrity Level (ASIL)

Probability of Exposure + Controllability by Driver + Severity of Failure = ASIL

E0	Combination of Very low Probabilities
E1	Very Low Probability *(less often than once a year for the great majority of drivers)*
E2	Low Probability *(a few times a year for the great majority of drivers)*
E3	Medium Probability *(once a month or more often for an average driver)*
E4	High Probability *(almost every drive on average)*

C0	Controllable in general
C1	Simply controllable *(99% or more of all drivers are usually able to avoid a harm)*
C2	Normally controllable *(90% or more of all drivers are usually able to avoid a harm)*
C3	Difficult to control or Uncontrollable *(Less than 90% of all drivers are usually able or barely able to avoid a harm)*

S0	No injuries
S1	Light and moderate injuries
S2	Severe and life-threatening injuries (survival possible)
S3	Life threatening injuries (survival uncertain), fatal injuries

ASIL	FIT	
D	< 10	Required
C	< 100	Required
B	< 100	Advised
A	< 1000	informative

Severity	Probability	C1	C2	C3
S1	E0	QM	QM	QM
	E1	QM	QM	QM
	E2	QM	QM	QM
	E3	QM	QM	A
	E4	QM	A	B
S2	E0	QM	QM	QM
	E1	QM	QM	QM
	E2	QM	QM	A
	E3	QM	A	B
	E4	A	B	C
S3	E0	QM	QM	QM
	E1	QM	QM	A
	E2	QM	A	B
	E3	A	B	C
	E4	B	C	D

QM – Quality Management only
Not subject to ASIL requirements

1 FIT – 1 Failure in 10^9 hours

ASIL – Ratings Examples

- ASIL B: Brake lights failure on both sides
- ASIL B: No valid data from rear view camera
- ASIL C: Involuntary braking in cruise control
- ASIL D: Involuntary full power braking
- ASIL D: Involuntary airbag release
- ASIL D: Involuntary acceleration

ISO 26262 Recommendation	ASIL A	ASIL B	ASIL C	ASIL D
Fault injection testing to verify the effectiveness of the Safety Mechanisms	+	+	+ +	+ +

Unified Functional Safety Verification Platform

FMEDA OVERVIEW

FME(D)A - **F**ailure **M**ode **E**ffect (**D**iagnostic) **A**nalysis

Systematic method of failure analysis, for each element

- Identify the manner in which a **failure can occur**
- Identify the **consequences of the failure**
- Identify the **probability/severity of the failure**

➔Define a **Safety Mechanism** to handle the Failure Mode, e.g.

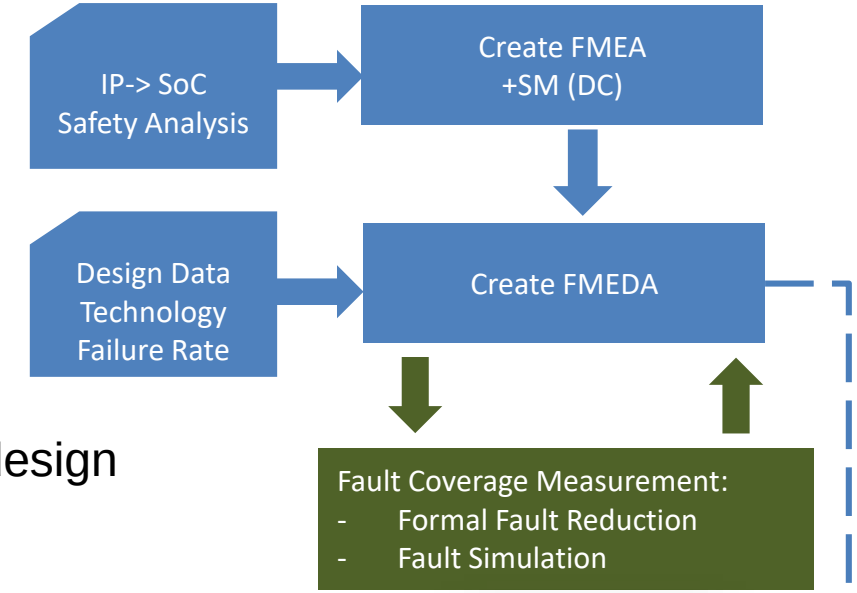
- Dual Core Lockstep with Comparator, ECC, STL (Software Testing Library), Triple Redundancy with Majority Voter

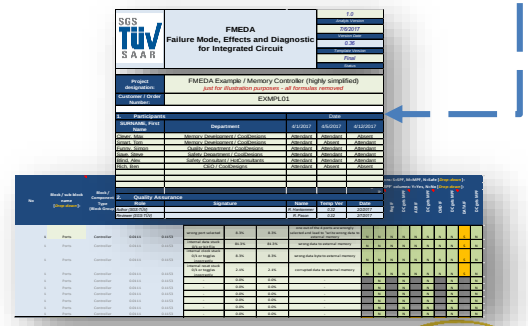
➔**Is the Failure observed? Is the Failure detected?**

FMEA/FMEDA Process – Metric for Random Faults

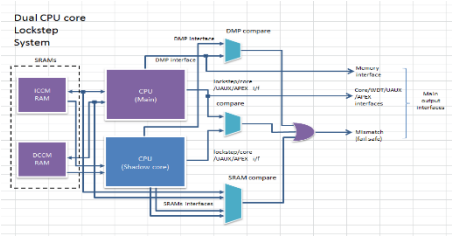
Implement and Confirm Quality of Safety Mechanisms (SM)

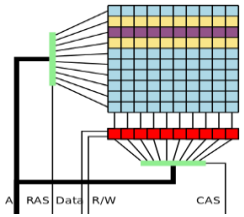
- Identify Failure Mode and Effects Analysis (FMEA) for each IP
- Define Safety Mechanisms to protect against random failures
- Compute **estimated** Safety Metrics with Failure Mode and Effect Diagnostic Analysis (FMEDA)
- Run fault injection to **measure** ISO 26262 metrics on implemented design
- Generate FMEDA report, Safety manual





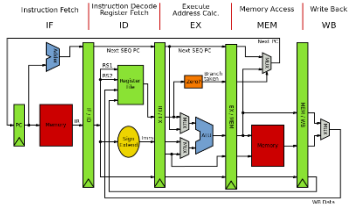
Dual-Core Lockstep





ECC Memory Protection

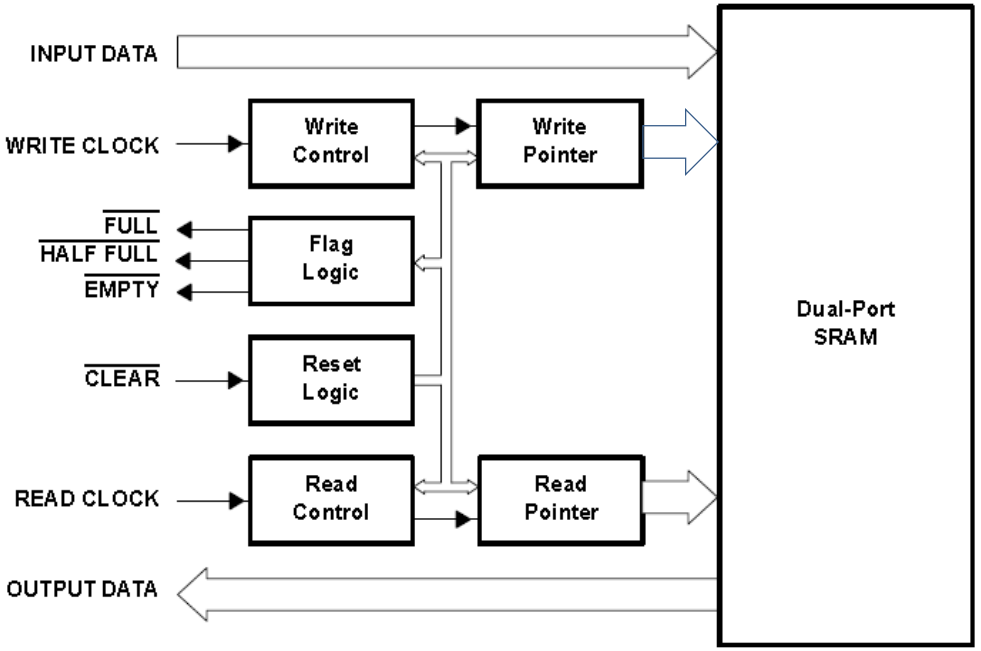
Software Test Libraries



Custom Safety Mechanisms

FMEA Failure Mode Analysis Example

- Failure Mode 1:
  - Failure: incorrect flags indication
    - Effect: Data will be overwritten/ lost
    - Safety Mechanism: Redundant Flag logic
- Failure Mode 2:
    - Failure: Data in SRAM is corrupted
    - Effect: Invalid data
    - Safety Mechanism: ECC
- Address both transient failures and permanent ones



Block Diagram of FIFO with Static Memory

FMEA Creation in VC Functional Safety Manager

SP level Analysis – Main FMEA

Identify Sub-Part Failure Mode

Failure Mode 1:

Failure: Failure: incorrect flags indication
Effect: Data will be overwritten/lost
Safety Mechanism: Redundant read/write control

Define FM in ‘Main FMEA’ tab

Main FMEA

<Filter> Please input text here

ID	Top Design Element	Element 1	Potential Faults	Potential Errors	otential Effect(s of Failure
F001	FIFO	FLAGS	Flag logic is faulty	Incorrect Flags Indication	loss of data
F002	FIFO	FLAGS	Flag logic is faulty	Incorrect Flags Indication	loss of data

Define Safety Mechanism in ‘Primary Safety Mechanisms’ tab

Primary Safety Mechanisms

<Filter> Please input text here

#	ID	Element	Safety Mechanism	Diagnostic or Avoidance?	Requirements ID	Equivalent ISO 26262 Diagnostic
1	SM001	FIFO	Flag Logic Dup	Diagnostic	SoC.Fifo.Req001	Processing units: Registers::HW redundancy (e.g. dual core lockste...

Add SMs to FMs in ‘Main FMEA’

Project IP Report Utilities

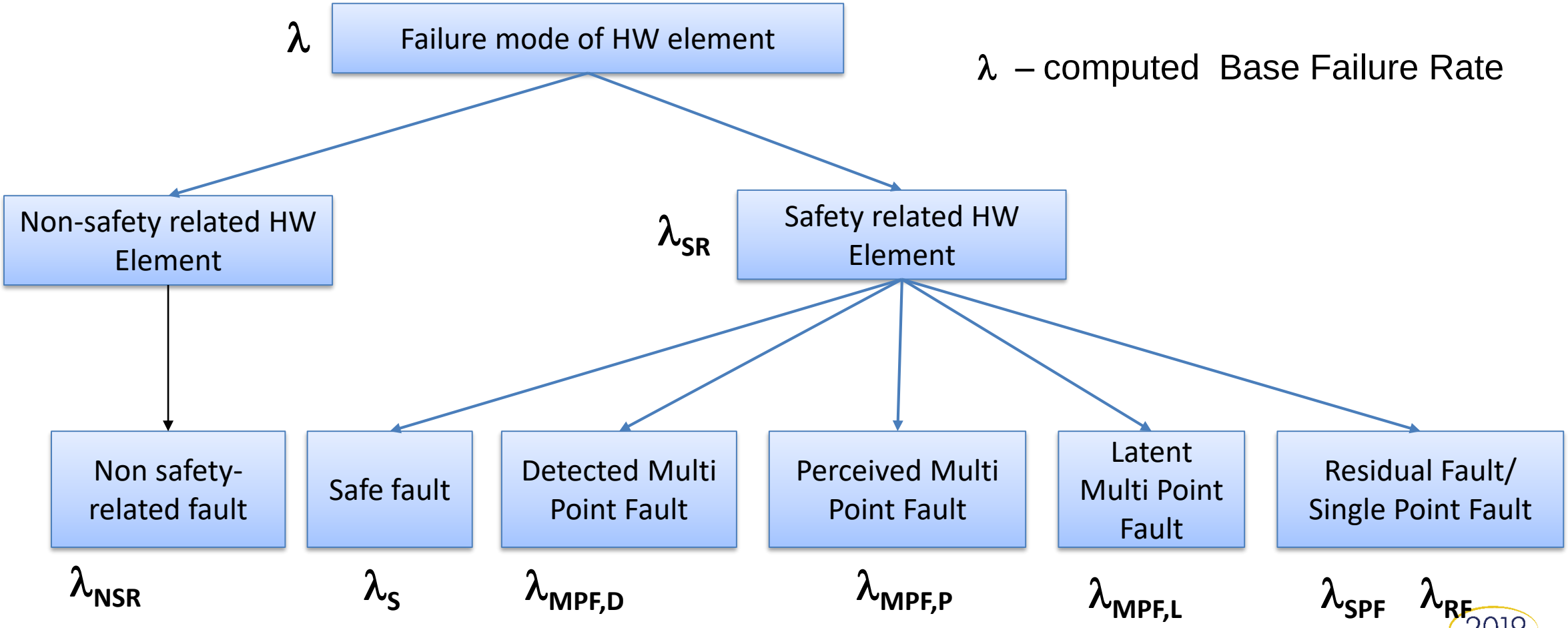
Main FMEA

<Filter> Please input text here

ID	Top Design Element	Element 1	Potential Faults	Potential Errors	otential Effect(s of Failure	ISO 26262 Equivalent Fault/Error/Failure	Systematic or Random Failure?	Permanent or Transient Failure?	Safety Related	Safe Failure	Probability to the Safety Mechanis	Current Primary
F001	FIFO	FLAGS	Flag logic is faulty	Incorrect Flags Indication	loss of data	Processing units: Registers::Stack overflow/underflow	Random	Permanent	true	false	true	SM001
F002	FIFO	FLAGS	Flag logic is flaty	Incorrect Flags Indication	loss of data	Processing units: Registers::Stack overflow/underflow	Random	Transient	true	false	true	SM001

ISO 26262 Metric: Classification of Faults

Part of FMEDA analysis



λ – computed Base Failure Rate

Base Failure Rate Calculation

Tech data + IP design data

- The designers associate the design sub-part relevant for the FM

Main FMEDA

<Filter> Please input text here

	#	FMEA ID	Function Hierarchy
1	9	F001	Associate
2	10	F002	Associate
3	11	F003	Associate
4	12	F004	Associate
5	13	F005	Associate
6	14	F006	Associate
7	15	F007	Associate
8	16	F008	Associate
9	17	F009	Associate
10	18	F010	Associate

Function Hier Association for F001 (Permanent)

List Tree

Filter: Regular Expression Inverse Match

Associate All | Unassociate All | Unselect | << Apply | Overwrite by Copy | From Src Failure Mode: F002

HierScope	Main FMEDAs	SF %	Total SF %	Src
FIFO		0.00	0	
Read Pointer IF	F003	0.00	100	
Read Pointer SM		0.00	100	
Write Pointer IF	F005	0.00	100	
Write Pointer SM		0.00	100	
✓ Flags IF	F001	100	100	✓
Flags SM		0.00	100	
SRAM	F007	0.00	100	

✓ Recursively (Un)Select | Current FM | Assigned | Unassigned | All

Common Scale Factor: 100 % | Total Scale Factor(%): Any

Individual Scale Factor | ✓ Recursively apply

Justification:

Enter justification for chosen scale factor. Leave blank if 100%

Tech Data

<Filter> Please input text here

	#	Type	Permanent Failure Rate (FIT)	Transient Failure Rate (FIT)	Unit	
1	2	Latches	3.4e-6	3.4e-6	FIT per Latch	Latches
2	3	RAM_Bits	1e-7	1e-7	FIT per RAM_Bit	RamBits
3	4	ROM_Bits	1.7e-7	1.7e-7	FIT per ROM_Bit	RomBits
4	5	Digital_Area	3.4e-6	3.4e-6	FIT per Square Micron	LogicArea
5	6	Analog_Area	0.01	0.01	FIT per Square Micron	AnalogArea
6	7	RAM_Equiv_Xtors	2.84e-9	0	FIT per RAM_Equiv_Xtor	RamTransistors
7	8	ROM_Equiv_Xtors	2.36e-5	0	FIT per ROM_Equiv_Xtor	RomTransistors

- The combination of design data and technology data is used for computing base Failure rate **λ**

Design Data

Active Instance: (Base) <Filter>

	#	Simple Hierarchy Name	Real Hierarchy Name	Flops	Latches	RAM Bits	ROM Bits	Digital Area	Analog Area
1	1	FIFO	test.DUT	0	0	0	0	0	0
2	2	Read Pointer IF	test.DUT.RP_IF	3	0	0	0	1776.26	0
3	3	Read Pointer SM	test.DUT.RP_SM	3	0	0	0	1812.05	0
4	4	Write Pointer IF	test.DUT.WP_IF	3	0	0	0	1776.26	0
5	5	Write Pointer SM	test.DUT.WP_SM	3	0	0	0	1812.05	0
6	6	Flags IF	test.DUT.FL_IF	0	0	0	0	4098.44	0
7	7	Flags SM	test.DUT.FL_SM	4	0	0	0	4341.09	0
8	8	SRAM	test.DUT.sdpram_i1	16	0	32	0	827.78	0

Estimated FMEDA Calculation & Report

Fsafe, Fpvsg, Primary SM Specification

- Changing these parameters changes the calc. in the Failure Rates tab below accordingly

Fsafe

Fpvsg

Fper

Krf

Kmpf

Main FMEDA

<Filter> Please input text here | Hide FMEA

	#	FMEA ID	Function Hierarchy	FI Efforts	Estimated F_{safe}	Primary SM Type	Estimated IF DC(K_{RF})	Latent SM Type	Estimated IF Latent DC (K_{MPF})	F_{PVSG}
1	9	F001	Associate		0%	Dup logic and compar...	95%	Dup logic and compar...	99%	100%
2	10	F002	Associate		50%	Dup logic and compar...	90%	No SM assigned	0%	100%
3	11	F003	Associate		0%	Dup logic and compar...	95%	No SM assigned	0%	100%
4	12	F004	Associate		50%	Dup logic and compar...	90%	No SM assigned	0%	100%
5	13	F005	Associate		0%	Dup logic and compar...	95%	No SM assigned	0%	100%
6	14	F006	Associate		50%	Dup logic and compar...	90%	No SM assigned	0%	100%
7	15	F007	Associate		0%	ECC	99%	No SM assigned	0%	100%
8	16	F008	Associate		50%	ECC	99%	No SM assigned	0%	100%
9	17	F009	Associate		0%	No SM assigned	0%	No SM assigned	0%	100%
	18	F010	Associate		50%	No SM assigned	0%	No SM assigned	0%	100%

	#	ID	Top Design Element	Element 1
1	9	F001	FIFO	FLAGS
2	10	F002	FIFO	FLAGS
3	11	F003	FIFO	RD_PTR
4	12	F004	FIFO	RD_PTR
5	13	F005	FIFO	WR_PTR
6	14	F006	FIFO	WR_PTR
7	15	F007	FIFO	SRAM
8	16	F008	FIFO	SRAM
9	17	F009	FIFO	Control
10	18	F010	FIFO	Control

Synthetic Hierarc... | Safety Mechanism Types | Design D... | Main FMEA | Main FMEDA | Primary Safety Mechanisms | Safety Mechanism FMEA | Safety Mechanism FMEDA

Associating an FMEDA with a hierarchical sub-component provides the relevant design data for calculations below

tes

DA #9, F001, Permanent, Safety Related:true, Safe Failure:false

	Flops	Latches	RAM Bits	ROM Bits	Digital Area	Analog Area	RAM Equiv Transistors	ROM Equiv Transistors	FIT	FMD
	0	0	0	0	4098.44	0	0	0	0.013935	24.476816%
Total	32	0	32	0	16711.23	0	0	0	0.05693	100%
Unmapped	0	0	0	0	0	0	0	0	0	0%

Base failure rate

	$\lambda_{intrinsic}$	λ_{nSR}	λ_{SR}	F_{safe}	λ_S	λ_{nS}	$K_{FMC,RF}$	F_{PVSG}	λ_{PVSG}	λ_{SPF}	λ_{RF}
Estimated	0.013935	0	0.013935	0%	0	0.013935	95%	100%	0.013935	0	0.000697
Measured											

	$\lambda_{MPF,PVSG}$	$\lambda_{MPF,FMC,RF}$	$K_{FMC,MPF}$	$\lambda_{MPF,det}$	$\lambda_{MPF,pl}$	F_{per}	$\lambda_{MPF,1}$	SoC built in Diagnostic	Diagnostic ID	SoC built in Coverage
Estimated	0	0.013238	99%	0.013106	0.000132	0%	0.000132	Flag Logic Dup	SM001	95%
Measured										

View the IP level ISO 26262 Metric

Metrics Dashboard Tab

Main FMEDA

<Filter> Please input text here | Hide FMEA

	#	FMEA ID	Function Hierarchy	FI Efforts	Estimated F_{safe}	Primary SM Type	Estimated IF DC(K_{RF})	Latent SM Type	Estimated IF Latent DC (K_{MPF})	F_{PVSG}
1	9	F001	Associate		0%	Dup logic and compar...	95%	Dup logic and compar...	99%	100%
2	10	F002	Associate		50%	Dup logic and compar...	90%	No SM assigned	0%	100%
3	11	F003	Associate		0%	Dup logic and compar...	95%	No SM assigned	0%	100%
4	12	F004	Associate		50%	Dup logic and compar...	90%	No SM assigned	0%	100%
5	13	F005	Associate		0%	Dup logic and compar...	95%	No SM assigned	0%	100%
6	14	F006	Associate		50%	Dup logic and compar...	90%	No SM assigned	0%	100%
7	15	F007	Associate		0%	ECC	99%	No SM assigned	0%	100%
8	16	F008	Associate		50%	ECC	99%	No SM assigned	0%	100%
9	17	F009	Associate		0%	No SM assigned	0%	No SM assigned	0%	100%

Project Management | Synthetic Hierarchy | Safety Mechanism Types | Design Data | Main FMEA | Main FMEDA | Primary Safety Mechanisms | Sa

Metrics Dashboard

	Permanent	**Transient**	Total
Estimated SPFM	96.071764%	96.892655%	96.482209%
Measured SPFM	N/A	N/A	N/A

	SPFM	LFM
ASIL B	>=90%	>=60%
ASIL C	>=97%	>=80%
ASIL D	>=99%	>=90%

	Permanent
Estimated LFM	71.558876%
Measured LFM	N/A
Estimated LFM (for SM part)	95%
Measured LFM (for SM part)	N/A

	Permanent	Transient	Total
Estimated PMHF	0.00223635	0.00176902	0.00400413
Measured PMHF	N/A	N/A	N/A

Tcl Console | Failure Rates | Metrics Dashboard

View the hierarchical ISO 26262 Metric

Synthetic Hierarchy

Expand All | Collapse All | Report | Open

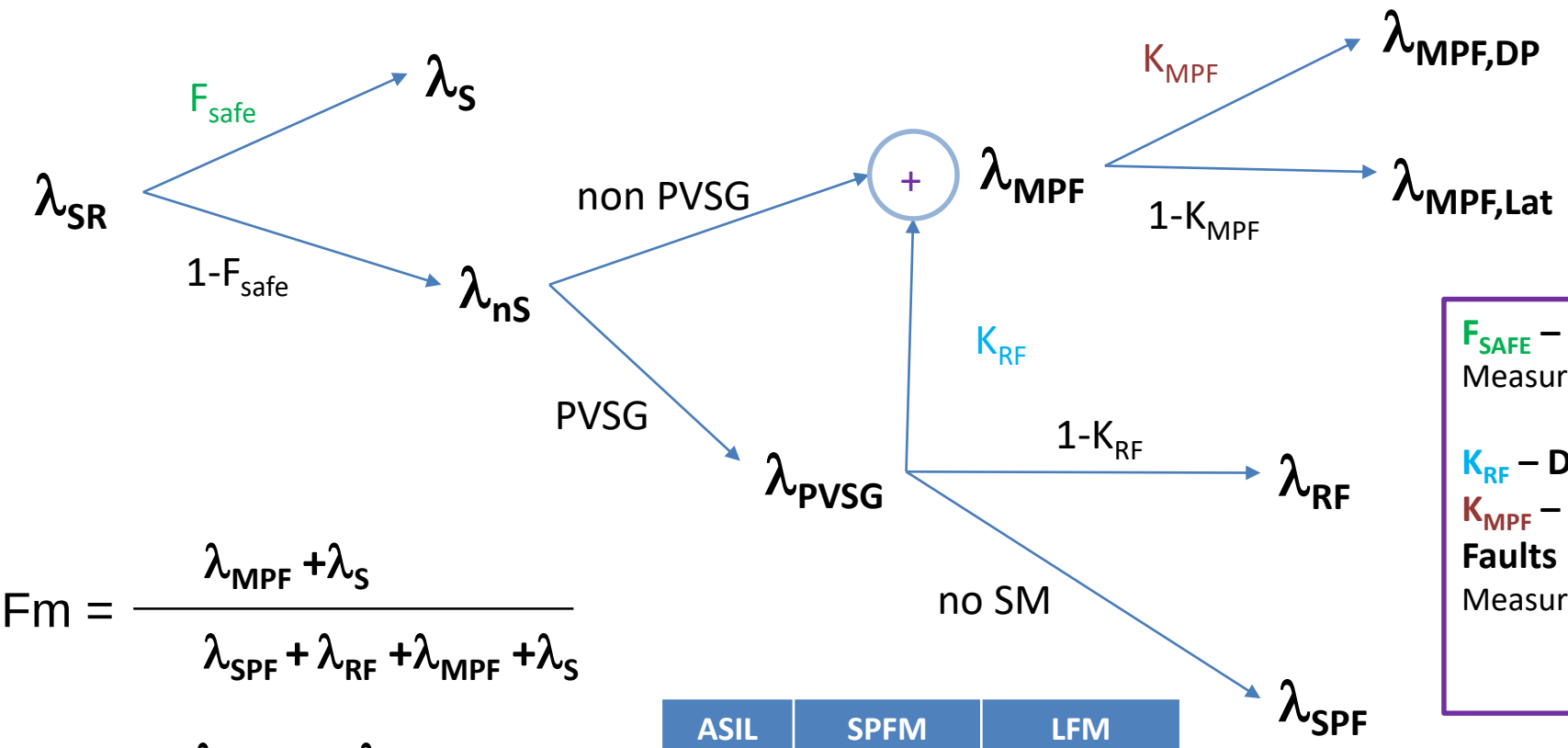
Name	Permanent SPFM	Permanent LFM	Permanent Failure Rate	Permanent Portion of Failure Rate	Transient SPFM	Transient Failure Rate	Transient Portion of Failure Rate
SelfDrvChip	93%	97.5484%	381.642	100%	94.2856%	25.5015	100%
HOST	90%	99%	63.607	16.6667%	94.5187%	4.25025	16.6667%
CPU	93.6%	97.2692%	318.035	83.3333%	94.2389%	21.2513	83.3333%
CPU_Top	90%	90%	63.607	16.6667%	91.5669%	4.25025	16.6667%
ALU (2,R)	99%	99%	127.214	33.3333%	98.6802%	8.50051	33.3333%
DEC (2,M)	90%	99%	127.214	33.3333%	91.1337%	8.50051	33.3333%

Hierarchy Name: SelfDrvChip.HOST

Project Management | Synthetic Hierarchy | Element ID | Main FMEA | Main FMEDA | Safety Mechanism FMEA

ISO 26262 Metric: Formulas for SPFm, LFm

Part of FMEDA analysis



$$SPFm = \frac{\lambda_{MPF} + \lambda_S}{\lambda_{SPF} + \lambda_{RF} + \lambda_{MPF} + \lambda_S}$$

$$LFm = \frac{\lambda_{MPF,DP} + \lambda_S}{\lambda_{MPF,DP} + \lambda_{MPF,L} + \lambda_S}$$

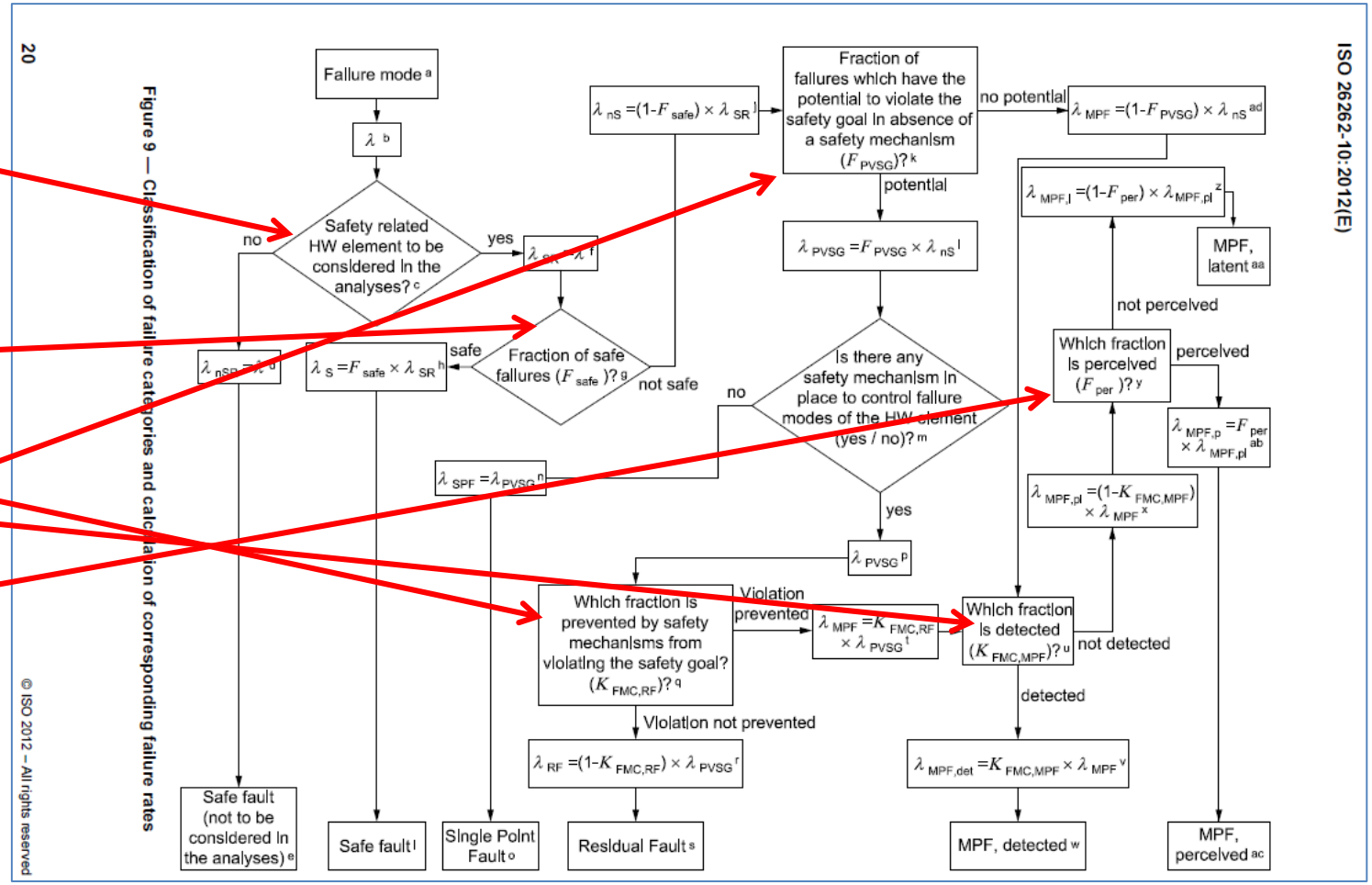
ASIL	SPFM	LFM
B	>= 90%	>= 60%
C	>= 97%	>= 80%
D	>= 99%	>= 90%

F_{SAFE} – Fraction of Safe Faults
Measured by structural analysis, Formal proofs

K_{RF} – Diagnostic Coverage, Residual Faults
K_{MPF} – Diagnostic Coverage, Multi Point Faults
Measured by Fault Injection Simulation

FMEA/FMEDA Columns & ISO 26262

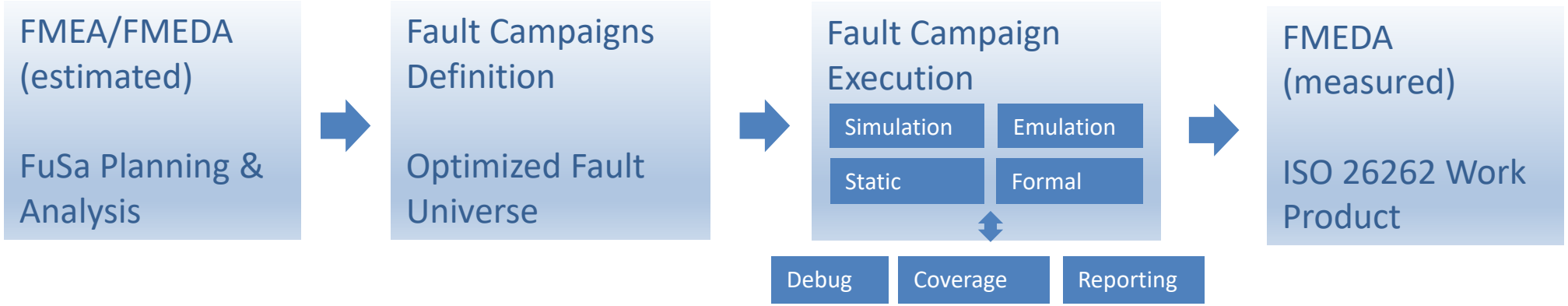
- Main/SM FMEA
  - Safety Related
- Main/SM FMEDA
  - F_{safe}
  - DC (K_{RF})
  - Latent DC (K_{MPF})
  - F_{PVSG}
  - F_{per}



Unified Functional Safety Verification Platform

FAULT INJECTION OVERVIEW

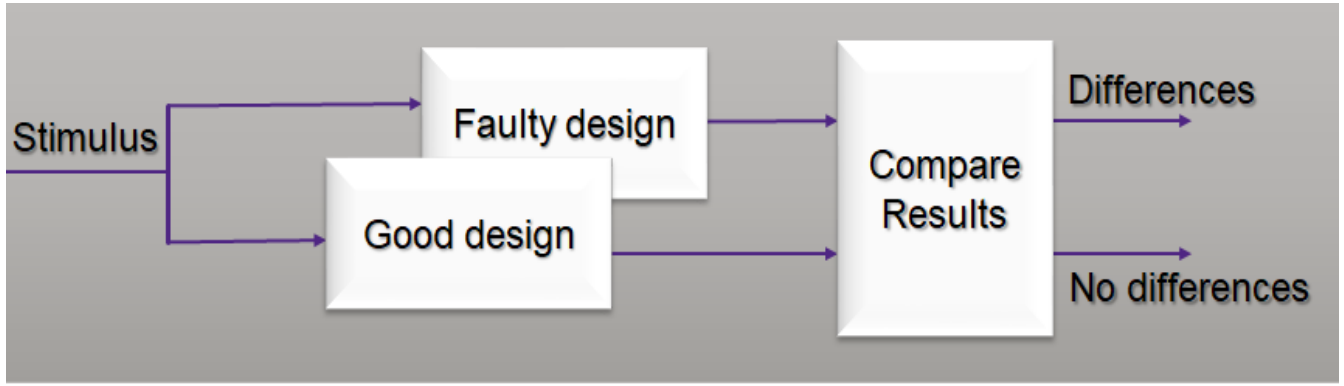
Fault Universe – Fault Campaigns



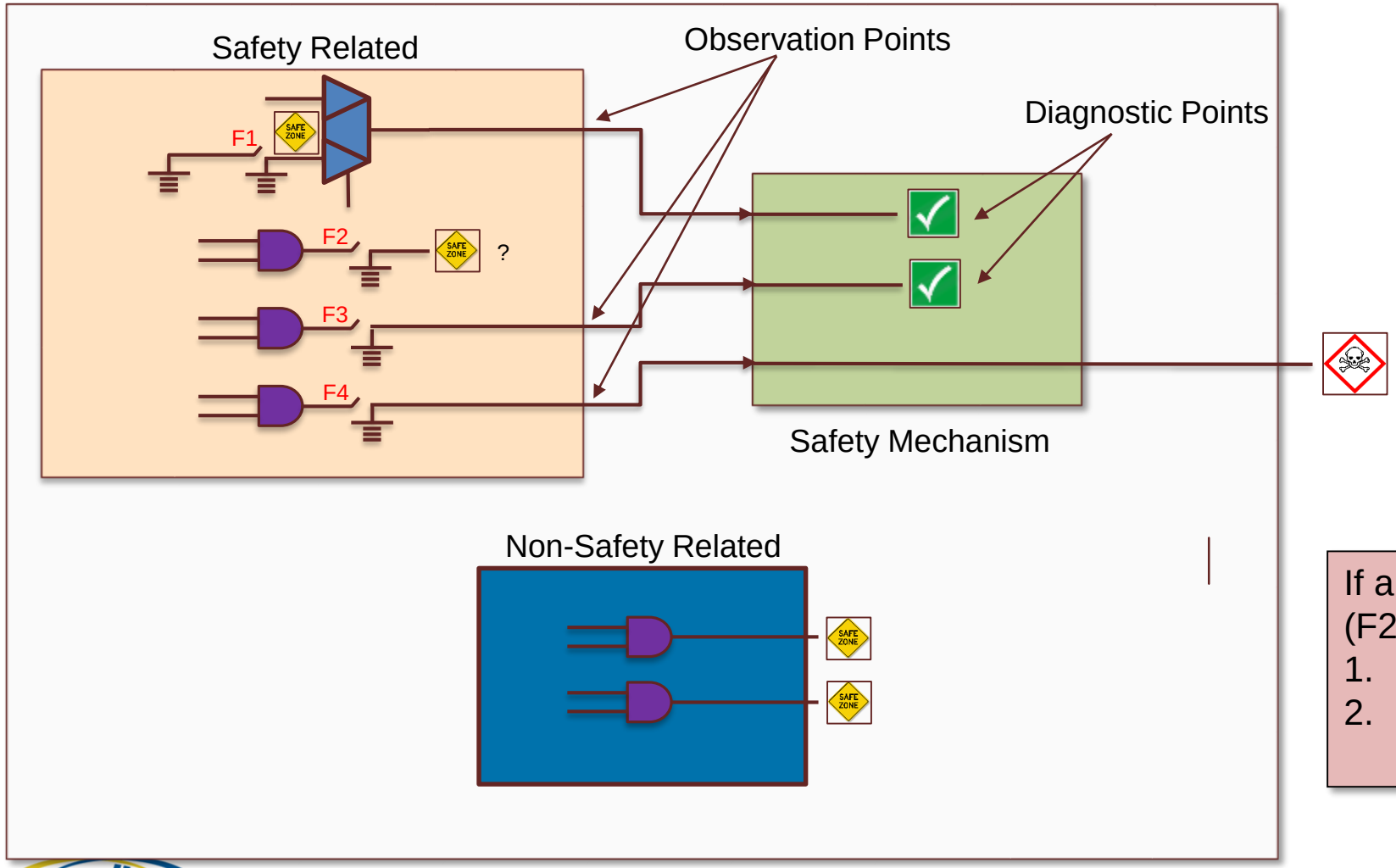
1. Define where to inject what kind of faults, per Failure Mode, add sampling
2. Prune and collapse the fault lists, structural analysis and formal techniques
3. Dynamic testability analysis (of remaining faults to simulate/emulate)
   Which faults can be best classified by which test?
4. Fault simulation/emulation with dynamic adjusting scheduling
5. Formal to (counter)prove “not observed” faults
6. Visualize and debug faults as needed
7. Report fault statistics and Diagnostic Coverage, per Failure Mode

Principles of Fault Injection

- Hypothetical faults are inserted into a design
- Tests are run against the faulty design (also called the Faulty Machine or FM)
- Specific points (detection signals) are compared against the un-faulted network (also called the Good Machine or GM) at designated strobe times
- If the strobing signals show difference between the GM run and the FM run, the fault is said to be detected.



Fault Classification by Fault Injection



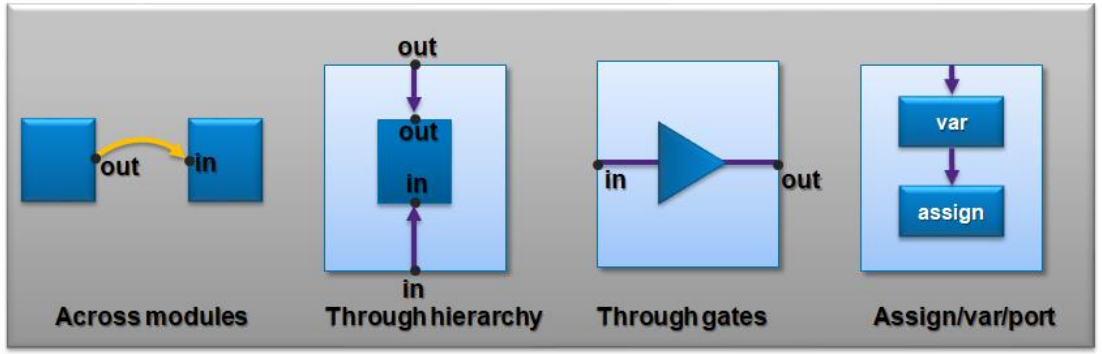
F1 – Safe
F2 – Assumed Dangerous
F3 – Dangerous Detected
F4 – Dangerous Undetected

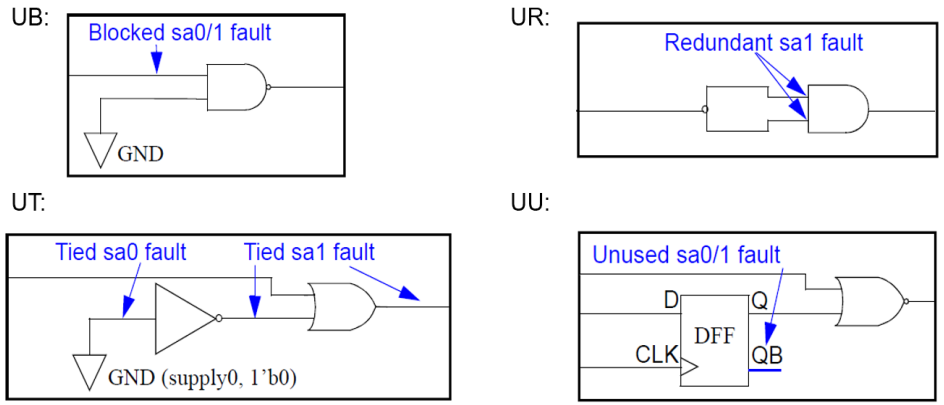
If a fault was not observed and/or detected (F2), it can be:

1. A safe fault
2. A dangerous fault which did not propagate due to insufficient stimulus

Principles of Fault Pruning and Collapsing

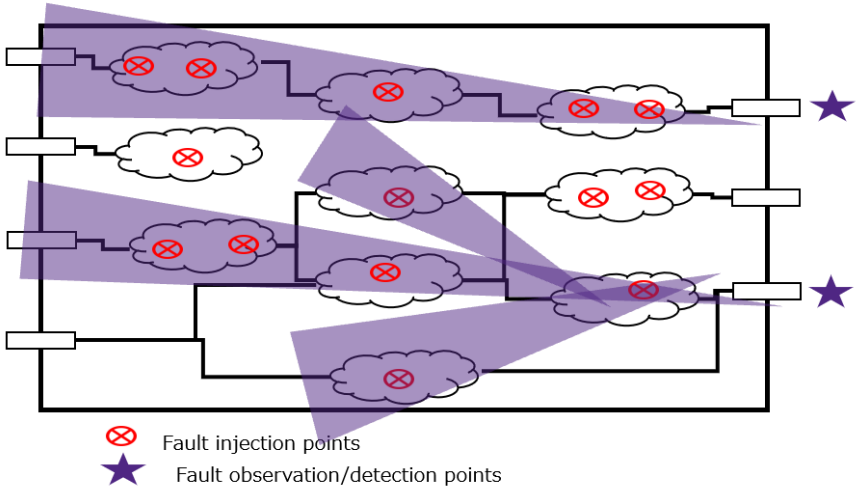
- Fault universe is huge, and in order to make it manageable, following techniques are offered:
  - Fault collapsing
    - Faults are classified as either prime or collapsed
      - A prime fault represents one or more faults
      - A collapsed fault produces the same observable behavior as its equivalent prime fault
    - Only prime faults are simulated
  - Structural fault Pruning
    - Some structural conditions which lead to safe faults are easy to detect, so they can be pruned even in the fault generation step



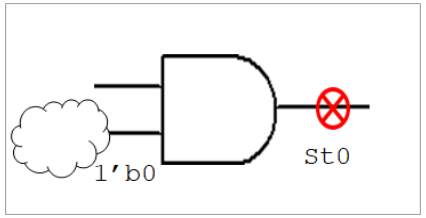


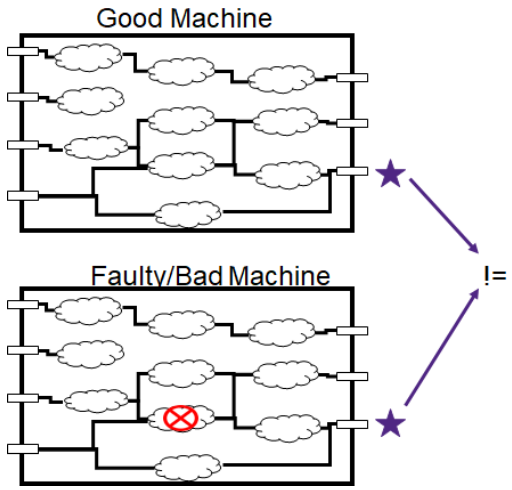
COI, Observability, Controllability Analysis (VC Formal)

- COI determination helps to identify the faults which belong to the failure mode

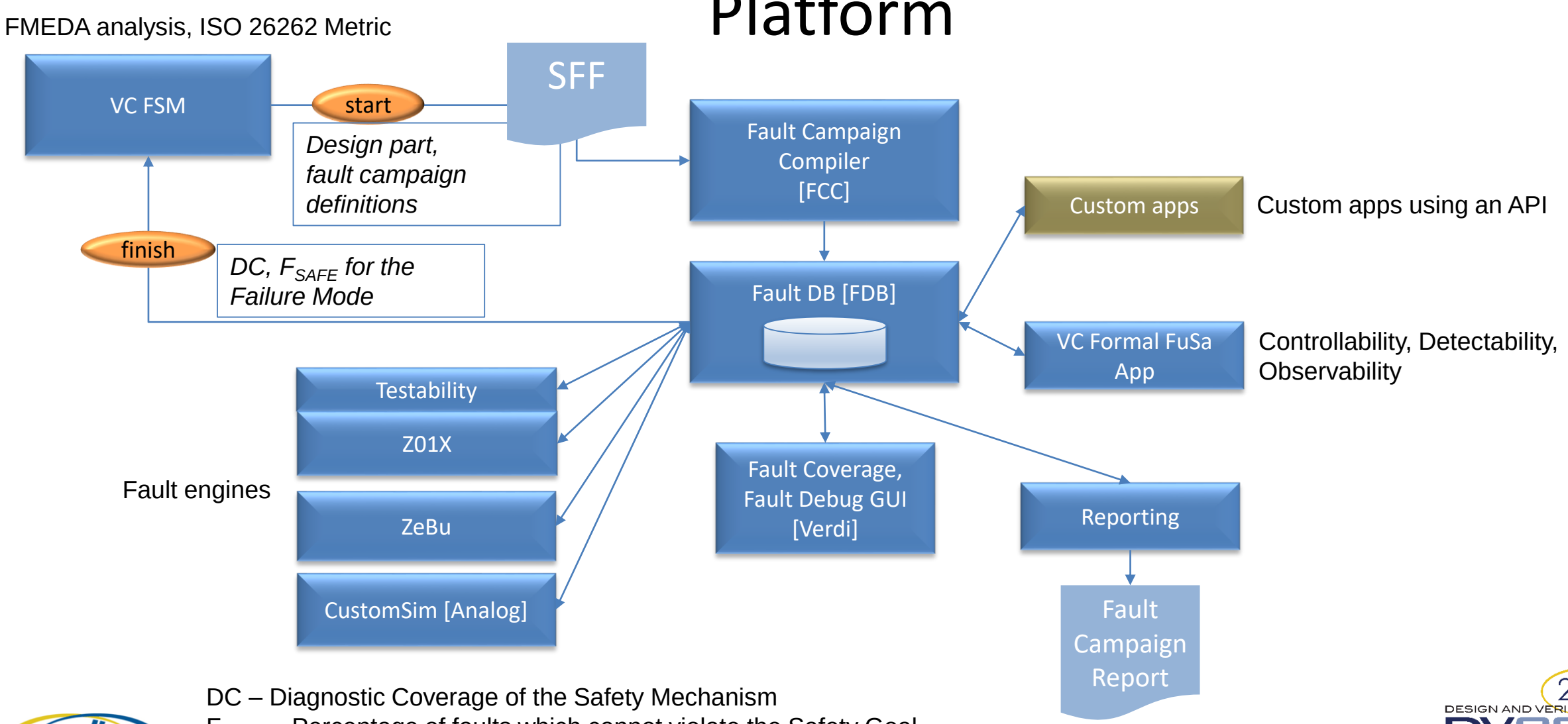


- Controllability and observability analyses help determine which faults are safe





Full Solution for Fault Classification – Unified Platform



DC – Diagnostic Coverage of the Safety Mechanism

F_{SAFE} – Percentage of faults which cannot violate the Safety Goal

Standard Fault Format (SFF) file content

Originally a Z01X feature

- A comprehensive way of defining faults statuses, faults groups and how to resolve types between different tests or even between different tools

```
StatusDefinitions
{
    # Creation of new functional safety definitions
    NN "Not Observed Not Diagnosed";
    NP "Not Observed Potential Diagnosed";
    ND "Not Observed Diagnosed";
    PN "Potential Observed Not Diagnosed";
    OP "Observed Potentially Diagnosed";
    ON "Observed Not Diagnosed";
    OD "Observed Diagnosed";

    # Any fault created and not set by a system task will have this status.
    DefaultStatus (NN)

    # Any fault of this status will be chosen by the simulation for injection
    Selected (NA, NN)
```

```
StatusGroups
{
    SA "Safe" (UT, UB, UR, UU);
    SU "Dangerous Unobserved" (NN, NC, NO, NT);
    DA "Dangerous Assumed" (HA, HM, HT, OA, OZ, IA, IP, IF, IX);
    DN "Dangerous Not Diagnosed" (PN, ON, OP);
    DD "Dangerous Diagnosed" (NP, ND, OD);
}
```

```
# Set fault generation constraints
FaultGenerate
{
    # Create faults on all reg types in hierarchy
    NA [0,1] { VARI "test.DUT.FL_IF.**" }
    # Create faults on all ports in hierarchy
    NA [0,1] { PORT "test.DUT.FL_IF.**" }

    Exclude
    {
        NA [0,1] { VARI "test.DUT.sdpram_i1.L_DataOut" }
        NA [0,1] { VARI "test.DUT.sdp
    }
}
```

```
# Define the merging of faults when multiple tests are run
PromotionTable
{
    StatusLabels (NN,NP,ND,PN,OP,ON,OD)
    #  NN  NP  ND  PN  OP  ON  OD
    [   -   |   |   |  ON   |   |  ;  # NN
        -   -   |   |   |   |   |  ;  # NP
        -   -   -   |  OD   |   |  ;  # ND
        -   -   -   -  ON   |   |  ;  # PN
       ON   -  OD  ON  OD   |   |  ;  # OP
        -   -   -   -   -   -   |  ;  # ON
        -   -   -   -   -   -   -  ;  # OD
    ]
}
```

```
Coverage
{
    "Diagnostic Coverage" = "DD/(SU+DA+DN+DD)";
}
```

New Updates to Standard Fault Format (SFF)

Accommodating Fault Campaign data

- Add information on the related FM, SM, observation and detection points
- Information shall be provided by VC FuSa Manager

```
# Software test library safety mechanism
SafetyMechanism sm_stl {
    Detect { "top.dut.cpu.alarm" }
}
# CPU lock step safety mechanism
SafetyMechanism sm_lockstep {
    Detect { "top.dut.lockstep.mismatch" }
}
FailureMode fm_wrong_register_value {
    Observe { "top.dut.cpu.registers.reg*"
        Exclude { "top.dut.cpu.registers.reg*_shadow" }
    }
    SafetyMechanisms(sm_stl, sm_lockstep)
}
FaultGenerate fm_wrong_register_value {
    NA [0,1] { [PRIM] "top.dut.cpu.**" }
}
```

FMEDA – Failure Mode & Safety Mechanism

→ Is the Failure observed?

→ Is the Failure detected?

Used by "Fault Injection Engines"

- Simulation
- Emulation
- Formal
- Static

to qualify **observed / detected**

Unified Fault Campaign Definition

Compiling Description into Fault Campaign Definition in Fault DB

```
StatusDefinitions
{
    # Creation of new functional sa
    NN "Not Observed Not Diagnosed"
    NP "Not Observed Potential Diag
    ND "Not Observed Diagnosed";
    PN "Potent
    OP "Observe
    ON "Observe
    OD "Observe

    # Any fault
    DefaultStat

    # Any fault
    Selected (N
```

```
StatusGroups
{
    SA "Safe" (UT, UB, UR, UU);
    SU "Dangerous Unobserved" (NN, NC, NO, NT);
                                          IA, IP, IF, IX);
```

```
# Set fault generation constraints
FaultGenerate
{
    # Create faults on all reg types in hierarchy
    NA [0,1] { VARI "test.DUT.FL_IF.**" }
    # Create faults on all ports in hierarchy
    NA [0,1] { PORT "test.DUT.FL_IF.**" }

    Exclude
    {
        NA [0,1] { VARI "test.DUT.sdpram_i
        NA [0,1] { VARI "test.DUT.sdpram_i1.sdpram_i1.L_DataOut" }
    }
}
```

```
Coverage
{
    "Diagnostic Coverage" = "DD/(SU+DA+DN+DD)";
}
```

Textual Description of the Fault Campaign
Independent of who (which Fault Engine) will execute the faults

Expanded and optimized Fault Campaign Definition
Independent of who (which Fault Engine) will execute the faults

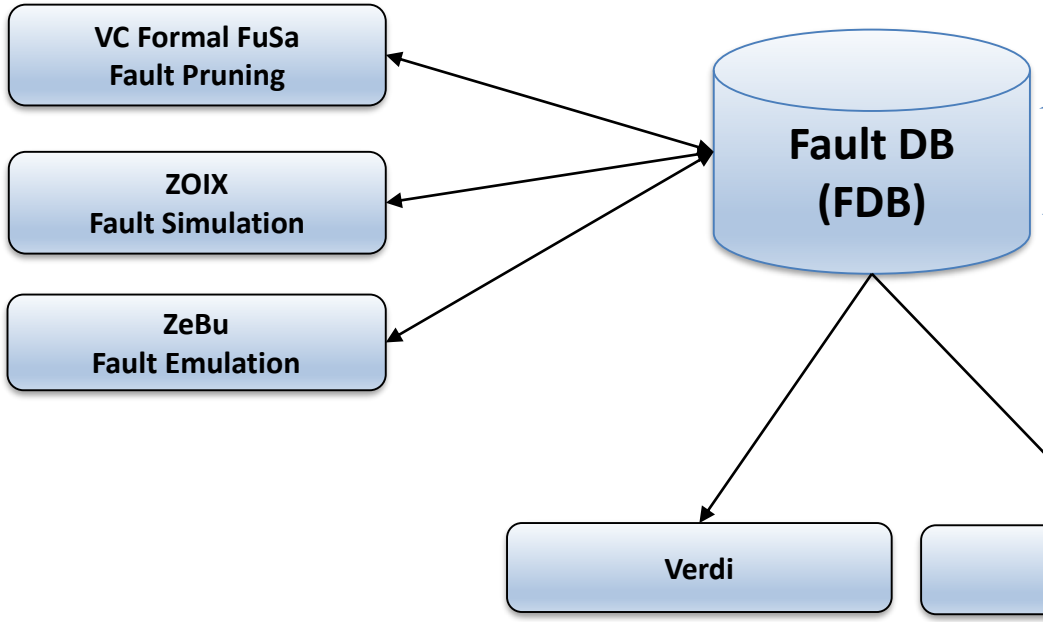
SFF
(Fault
campaign
description)

Fault Campaign Compiler

- Extraction / Expansion from Scope
- Failure Modes with Observation Points
- Safety Mechanisms with Detection Points
- Status Defs, Promotions, Cov Formula, …
- Pruning (e.g. no loads)
- Collapsing (Prime/Equivalent)
- Sampling (Times for transients, Selection in scopes)

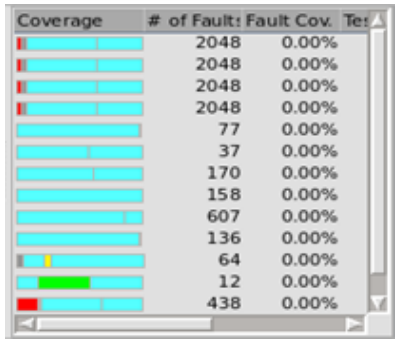
Fault DB

Unified Fault Campaign Execution



Expanded and optimized Fault Campaign Definition
Independent of who (which Fault Engine) will execute the faults

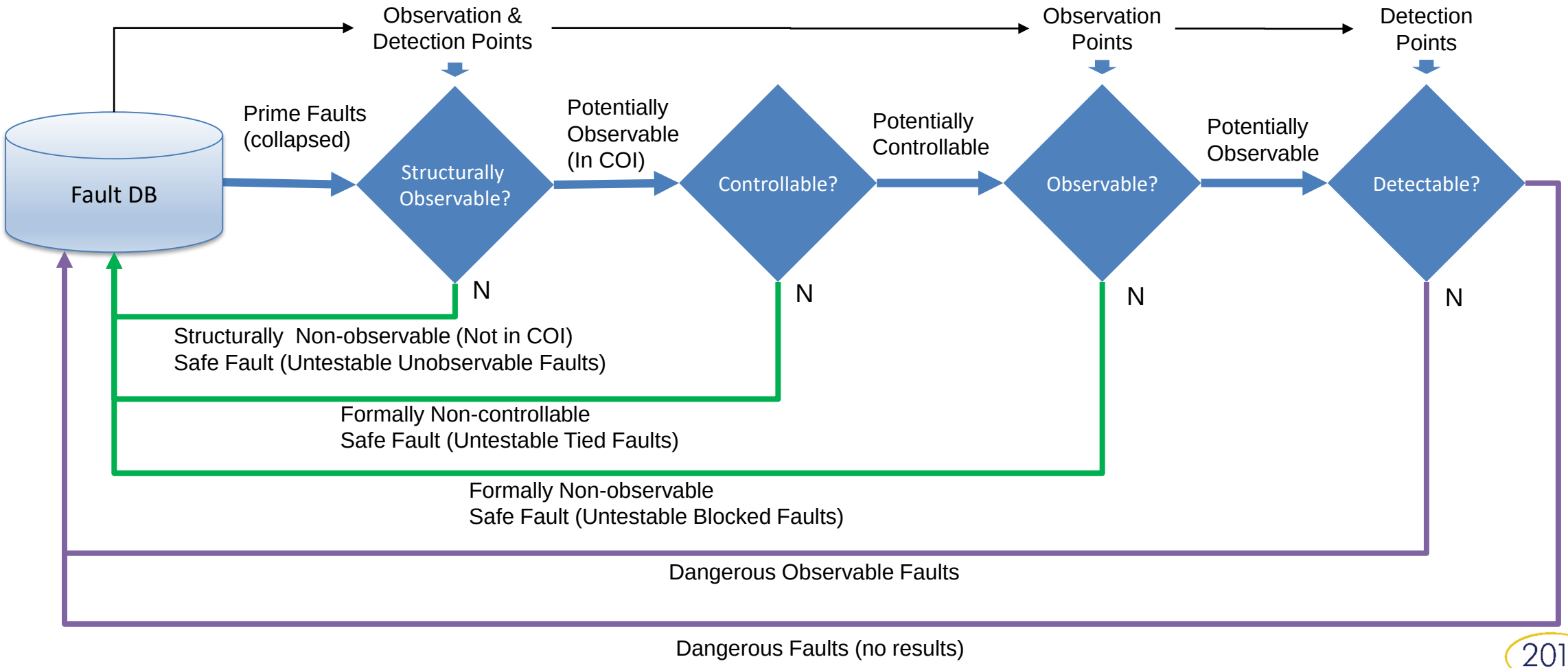
Fault Engine specific results, with
- Single understanding of a fault and its status
- Status promotion within tool and across tools
- Single representation and reporting



Coverage	# of Fault:	Fault Cov.	Te:
	2048	0.00%	
	2048	0.00%	
	2048	0.00%	
	2048	0.00%	
	77	0.00%	
	37	0.00%	
	170	0.00%	
	158	0.00%	
	607	0.00%	
	136	0.00%	
	64	0.00%	
	12	0.00%	
	438	0.00%	


```
#-------------------------------------------------------
# Fault Coverage Summary
#-------------------------------------------------------
# Total Faults:                                414
#
# Detected $finish/$stop            DF          1     0.42%
# Not Tested                        NT          1     0.84%
# Not Observed                      NO          5     3.78%
# Not Controlled                    NC         12     7.56%
# Not Observed Not Diagnosed        NN         28    11.76%
# Not Observed Diagnosed            ND          4     2.10%
# Observed Not Diagnosed            ON         20    11.34%
# Observed Diagnosed                OD        105    57.98%
# Observed Formal                   OO          1     0.84%
#
# Untestable Unused                 UU        117    62.18%
# Untestable Blocked                UB        148    84.87%
#
# Safe                              SA        265
# Dangerous Unobserved              SU         46
# Dangerous Assumed                 DA          0
# Dangerous Not Diagnosed           DN         27
# Dangerous Diagnosed               DD        109
```

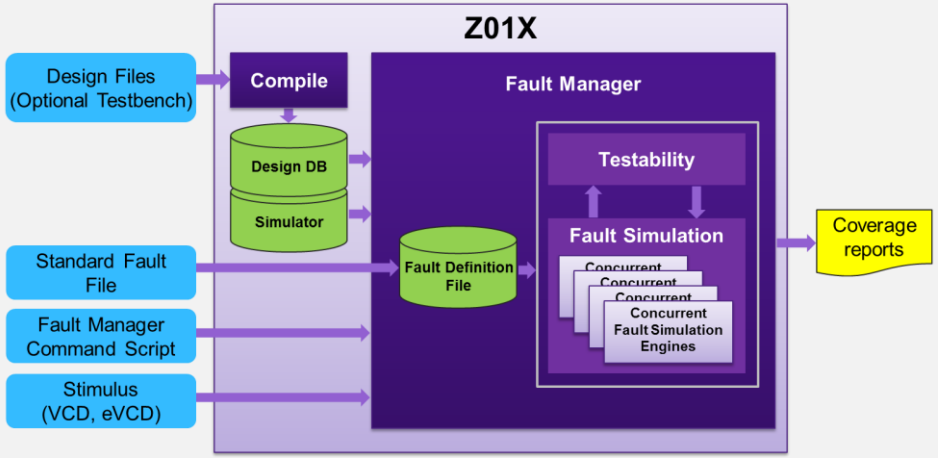
VC Formal FuSa - Fault Pruning



Fault Injection Campaign – Z01X Functional Safety

Highest performance fault simulation solution for ISO 26262 compliance requirements

- **Z01X Key Features**
  - Compatibility with ISO 26262 requirements and functional verification environments
  - Flexible fault management and testability-based fault optimization
  - Support for RTL and gate-level fault simulation
  - State-of-the-art concurrent fault simulation algorithm
  - TAT for very large designs and fault lists



Z01X is in use at major automotive semiconductor suppliers worldwide

Z01X is the fastest and most production-proven functional safety fault simulator in the industry

Synopsys Accelerates Development of Safety-Critical Products with Design Solutions for ARM Cortex-R52

High speed Z01X and Certitude fault simulation help assure functional safety for automotive safety standards

Sep 19, 2016

Mobileye Adopts Key Synopsys Automotive Functional Safety Verification Solution to Enable ISO 26262 Compliance of its Next-Generation ADAS SoCs

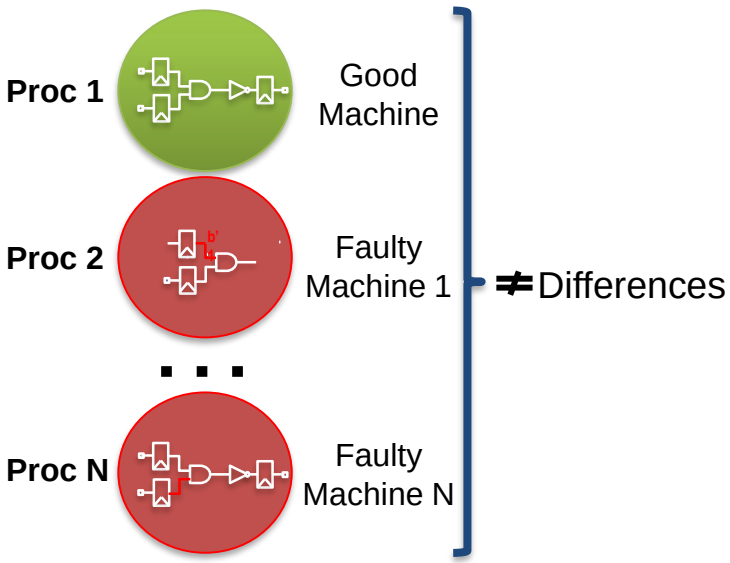
Mobileye Adopts Z01X Functional Safety for EyeQ4

Nov 21, 2016

Concurrent Fault Simulation

Parallel Simulation Technology

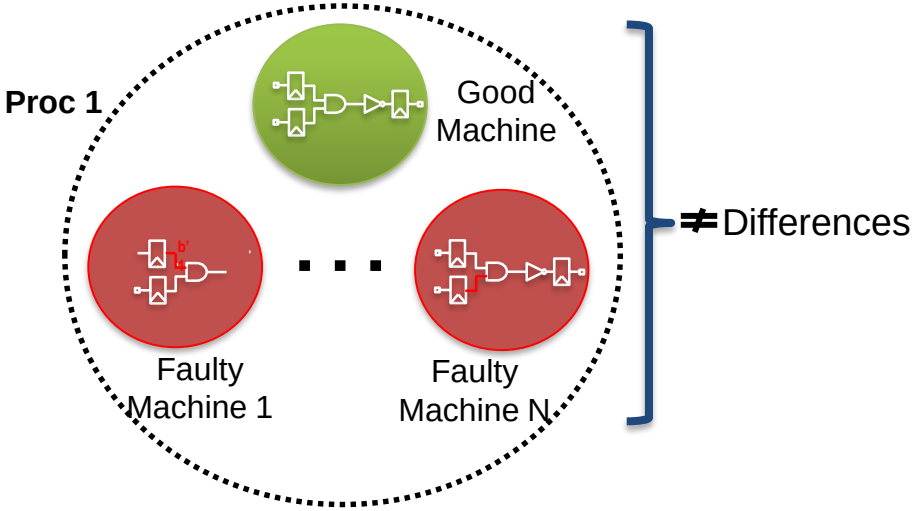
One fault per simulation



$$\frac{(\textit{Logic Sim Runtime}) \times \#\textit{Faults}}{\#\textit{Processors}}$$

Z01X Concurrent Simulation Technology

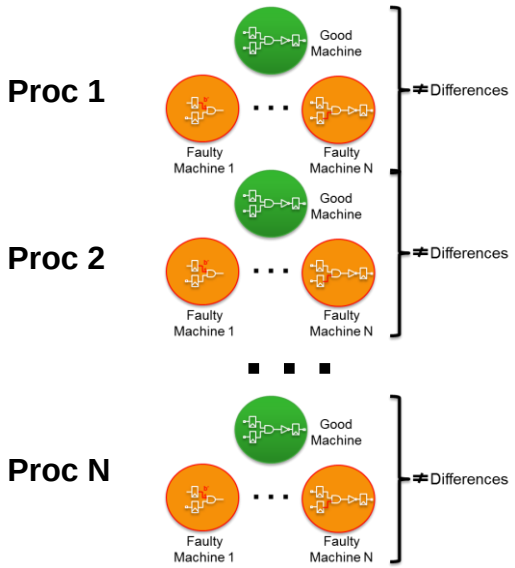
Thousands of faults per single simulation



$$\frac{(\sim 3\ X\ \textit{Logic Sim Runtime}) \times \#\textit{Faults}}{\textit{FPP}}$$

Distributed Z01X Simulations

Parallelize Z01X simulations via LSF/SGE…

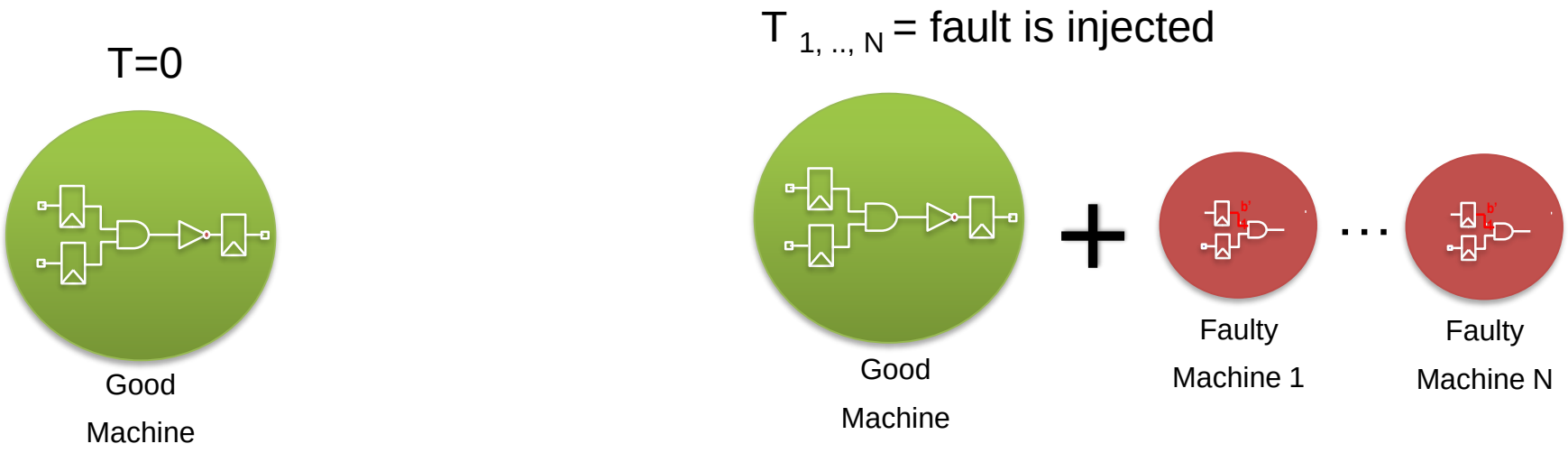


$$\frac{(\sim 3\ X\ \textit{Logic Sim Runtime}) \times \#\textit{Faults}}{\textit{FPP} \times \#\textit{Processors}}$$

ZOIX Concurrent Fault Simulator

- The design is “diverged” whenever the GM and FM values are different
  - FM copy of the design is created
- The diverged part of the design is simulated “concurrently” with the GM and other FM’s
- The diverged part is “converged” when the GM and FM values are the same

➔ **Significantly faster simulation of faults by using concurrency**



ZOIX Testability Analysis COATS

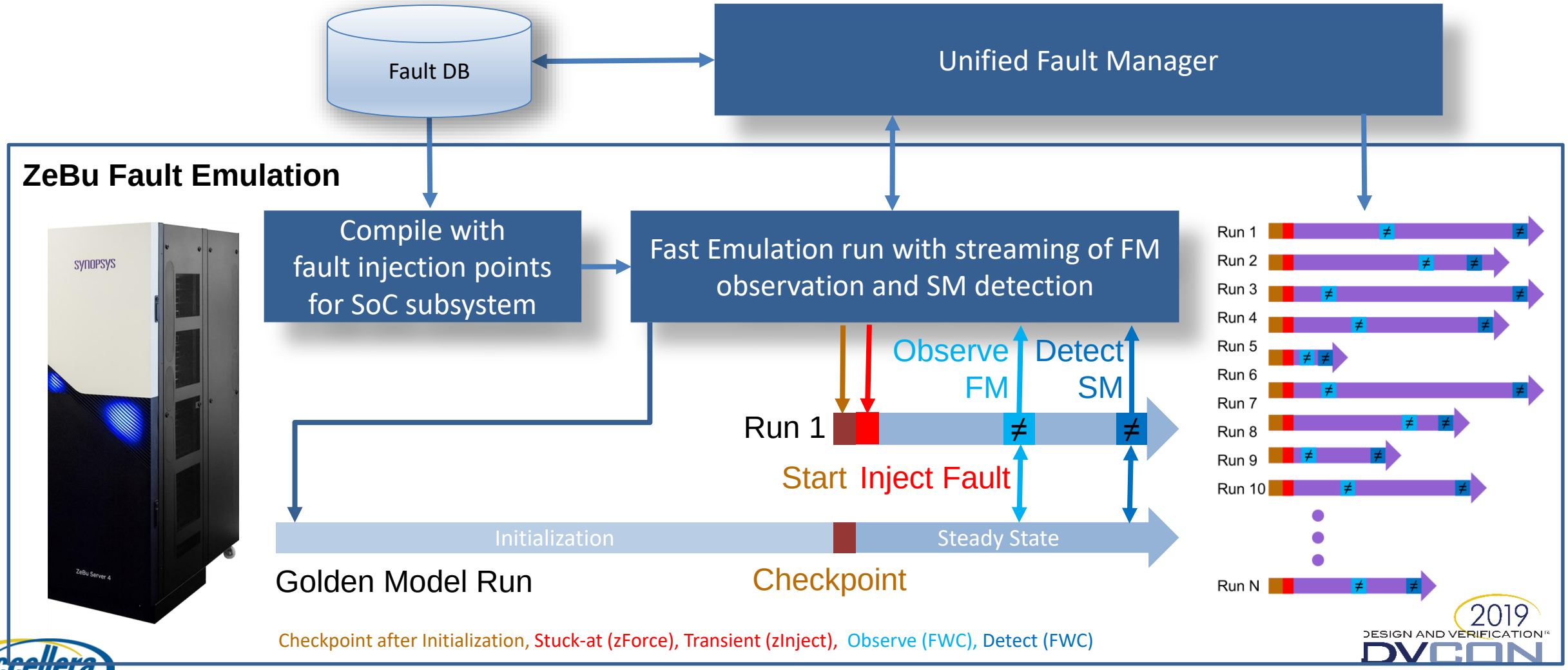
Controllability Observability and Testability System

- Uses controllability (toggle) and observability (backtrace) algorithms
- Provides early identification of untested areas
- Dynamic test ordering according to the test quality of faults selected for that test
- Elimination of redundant tests
- Only simulates detectable faults
- Calculates “Tenacity” value

➔ Optimizes the fault campaign orchestration

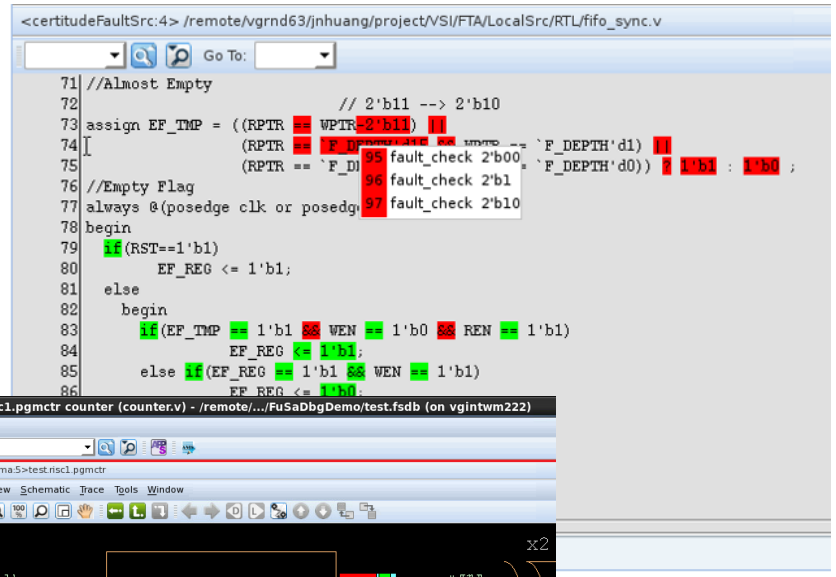
Fault Injection in Emulation

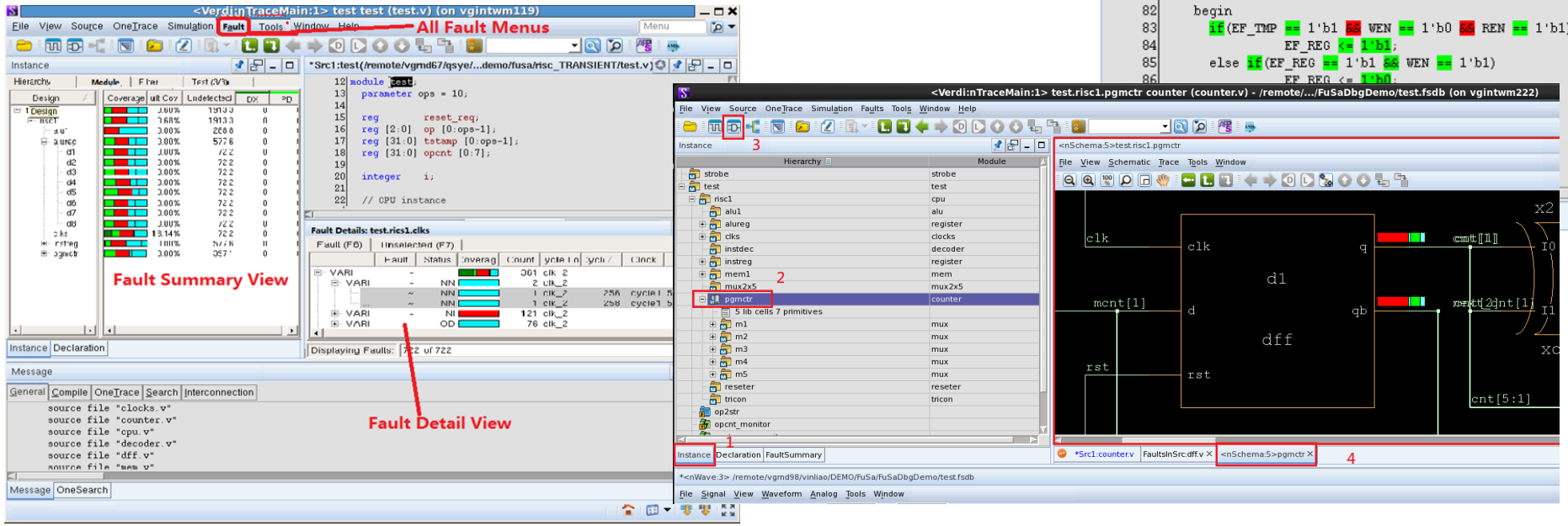
ZeBu Fast Fault Emulation Technology



Verdi Integration

- Advanced fault debug/coverage features for the Unified Functional Safety Platform
  - Annotate fault info in Verdi Schematic/Source views
  - Enable waveform mismatch debug between GM and FM
  - Support trace functions for mismatched waveforms
  - Display coverage information by hierarchy, fault type, etc.





Fault Campaign Back-annotated Results

Fault Injection Campaign Results to Calculate FMEDA Metrics

VC Functional Safety Manager:HOST

Project IP Report Utilities Help

Main FMEDA

<Filter> Please input text here | Show FMEA

	#	FMEA ID	Function Hierarchy	FI Efforts	Estimated F_{safe}	Measured F_{safe}	Primary SM Type	Estimated IF DC(K_{RF})	Measured IF DC(K_{RF})	Latent SM Type	Estimated IF Latent DC(K_{MPF})	Measured IF Latent DC(K_{MPF})	F_{PVSG}	F_{per}
1	1	HOST_FM_1	Associate	FIE_1, FIE_3, FI...	0%	60%	SM type not specif...	90%	88%	No SM assigned	0%	98%	100%	0%
2	2	HOST_FM_2	Associate	FIE_1, FIE_2	50%	80%	SM type not specif...	90%	0%	No SM assigned	0%	0%	100%	0%
3	3	HOST_FM_3	Associate	FIE_1, FIE_3	0%	0%	SM type not specif...	90%	70%	No SM assigned	0%	0%	100%	0%
4	4	HOST_FM_4	Associate	FIE_1, FIE_4	50%	0%	SM type not specif...	90%	96%	No SM assigned	0%	0%	100%	0%
5	5	HOST_FM_5	Associate	FIE_1, FIE_5	0%	0%	SM type not specif...	90%	0%	No SM assigned	0%	70%	100%	0%
6	6	HOST_FM_6	Associate	FIE_1, FIE_6	50%	0%	SM type not specif...	90%	0%	No SM assigned	0%	96%	100%	0%
7	7	HOST_FM_7	Associate	FIE_1, FIE_2	0%		SM type not specif...	90%		No SM assigned	0%		100%	0%
8	8	HOST_FM_8	Associate	FIE_1, FIE_3	50%		SM type not specif...	90%		No SM assigned	0%		100%	0%

Main FMEA | Main FMEDA | Primary Safety Mechanisms | Safety Mechanism FMEA | Latent Safety Mechanisms

Failure Rates

Main FMEDA #1, HOST_FM_1, Permanent, Safety Related:true, Safe Failure:false

	Flops	Latches	RAM Bits	ROM Bits	Digital Area	Analog Area	RAM Equiv Transistors	ROM Equiv Transistors	FIT	D_{FMi}
Function	611	41	25856	0	3218.4344	155136	25856	0	8.300039	13.04893%
Total	26430	1152	198144	0	38415.4101	1188864	198144	0	63.607048	100%
Unmapped	32	1	0	0	730.8403	0	0	0	0.000043	0.000068%

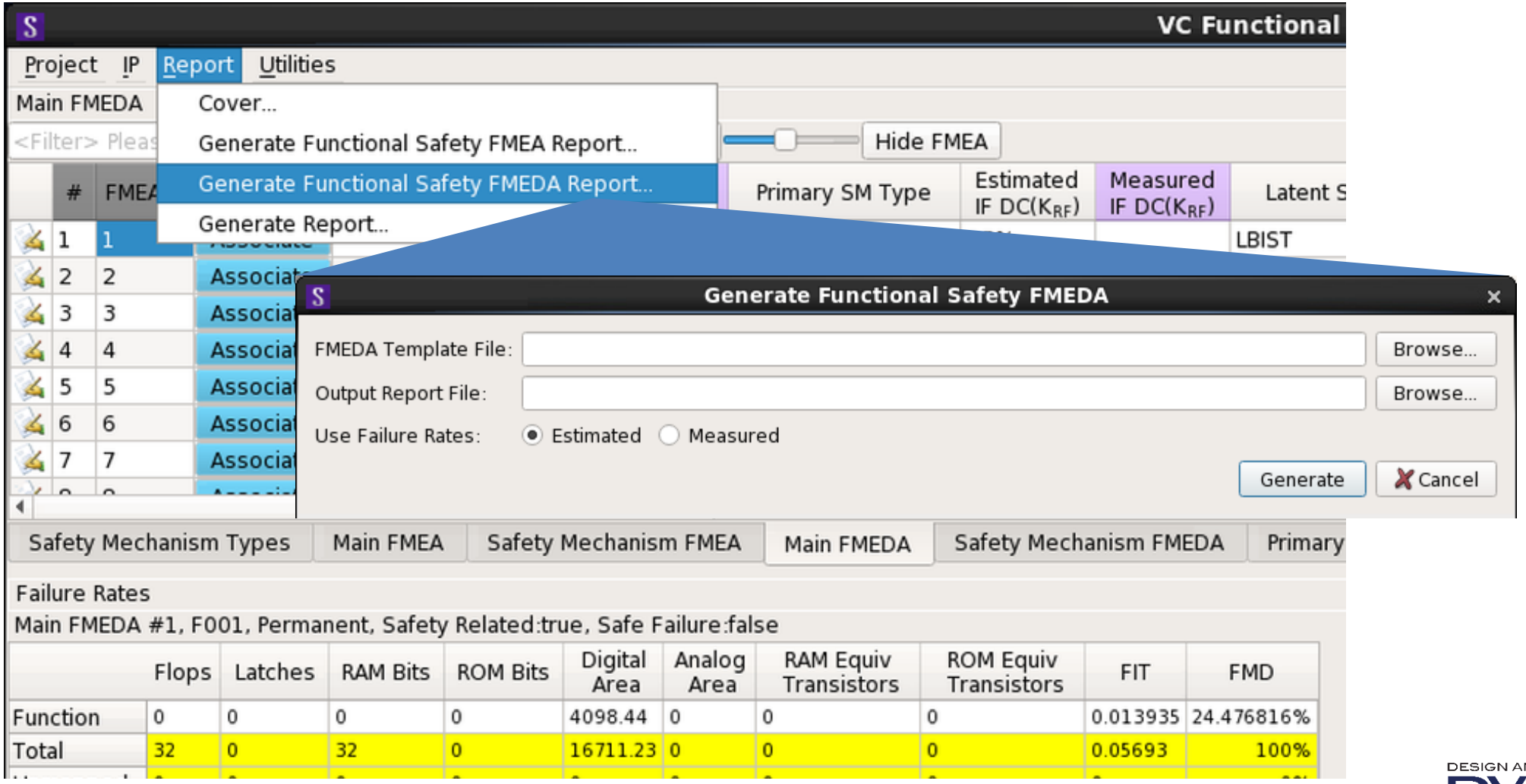
	$\lambda_{intrinsic}$	λ_{nSR}	λ_{SR}	F_{safe}	λ_S	λ_{nS}	$K_{FMC,RF}$	F_{PVSG}	λ_{PVSG}	λ_{SPF}	λ_{RF}
Estimated	8.300039	0	8.300039	0%	0	8.300039	90%	100%	8.300039	0	0.830004
Measured	8.300039	0	8.300039	60%	4.980023	3.320016	88%	100%	3.320016	0	0.398402

	$\lambda_{MPF,PVSG}$	$\lambda_{MPF,FMC,RF}$	$K_{FMC,MPF}$	$\lambda_{MPF,det}$	$\lambda_{MPF,pl}$	F_{per}	$\lambda_{MPF,1}$	SoC built in Diagnostic	Diagnostic ID	SoC built in Coverage
Estimated	0	7.470035	0%	0	7.470035	0%	7.470035	Host Safety 2	HOST_PSM_2	90%
Measured	0	2.921614	98%	2.863181	0.058432	0%	0.058432			

Tcl Console | Metadata | Failure Rates

Server: demo Project: SelfDrvChip IP: HOST User: user1

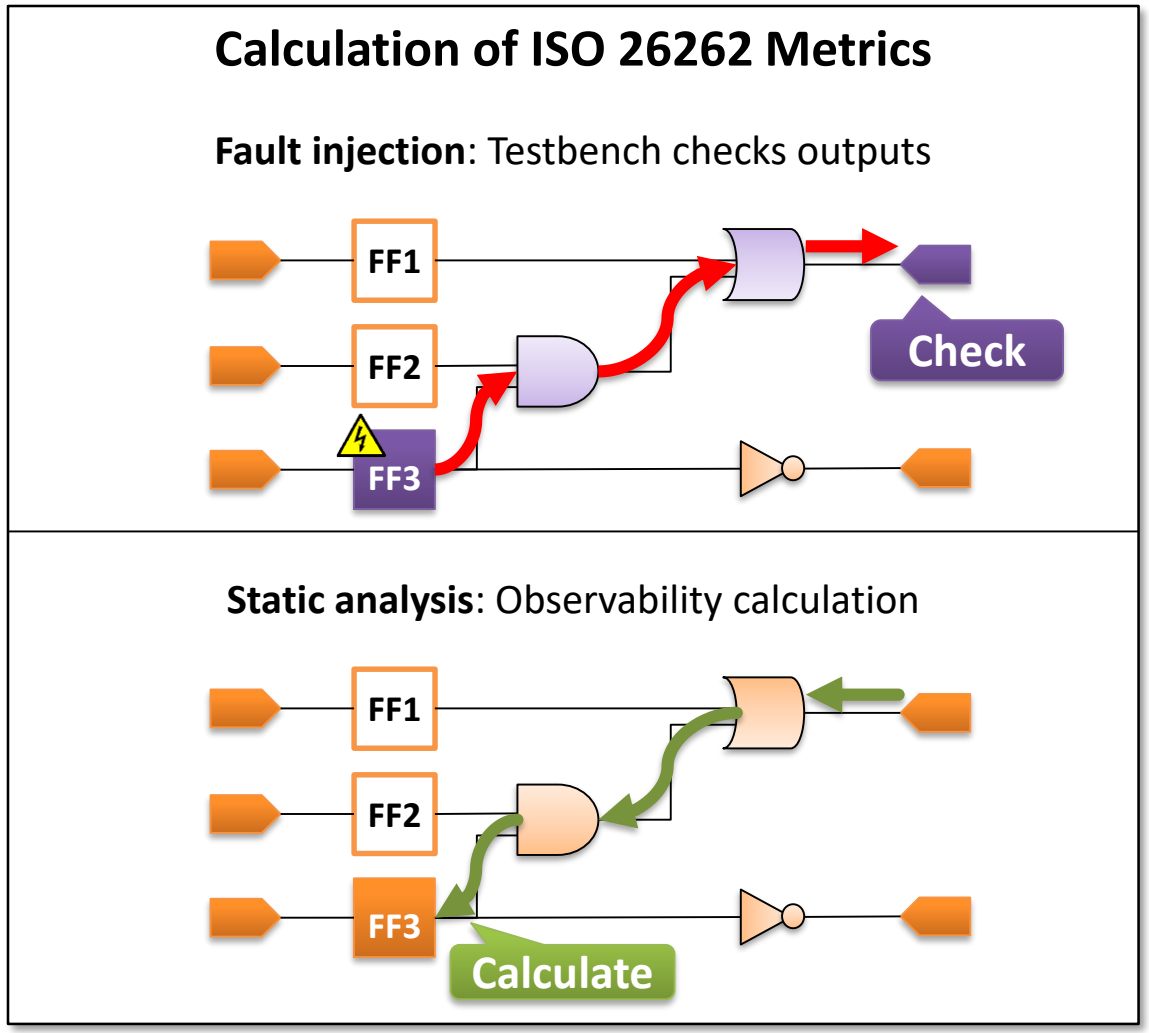
Generating FMEDA Reports



Early Soft Error Analysis for ISO 26262

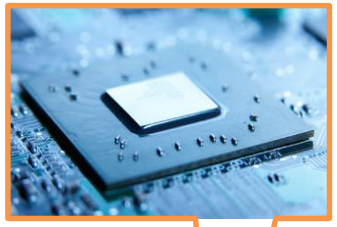
Using Static Analysis **(TestMAX FuSa)**

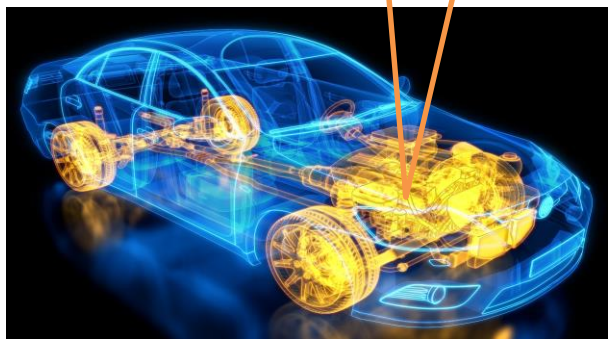
- Propagation based on probabilities
  - Can be applied in RTL or gates
  - Fast runtime
- Does not require testbenches
  - Ability to identify and address hotspots early in the design cycle
  - Measure impact of implemented safety mechanisms
- Can be used in conjunction with fault injection later in the design cycle
  - Minimizes iterations



Digital and Analog Fault Simulations

Z01X and CustomFault

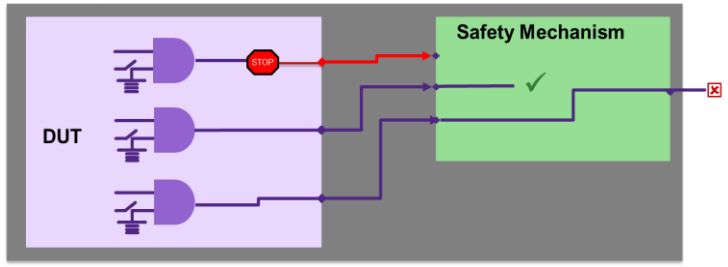




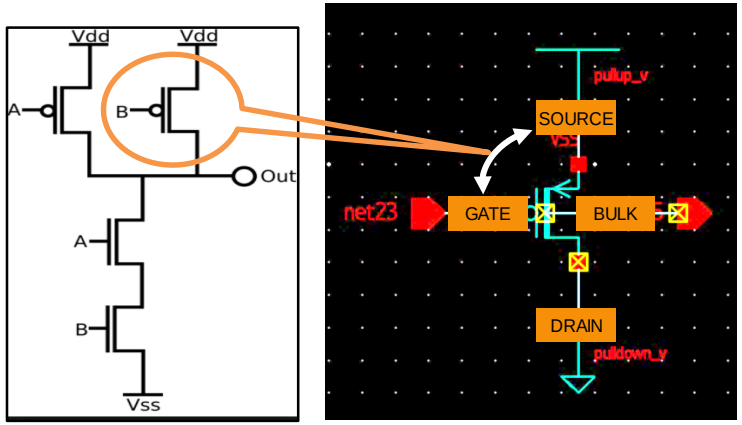
RTL Faults

```
$fs_default_status("SF");
// Check observation points for dangerous faults
always @(negedge clk)
  begin
    int compare = $fs_compare(sig1,sig2,sig3);
    if (compare)
      $fs_set_status("DF");
  end
```

Gate-level Faults



Transistor-level Defects



Digital Fault Simulation

Analog Fault Simulation

Unified Functional Safety Verification Platform

SUMMARY

Press Release – October 7th, 2019

Synopsys Announces Industry-First Unified Functional Safety Verification Solution to Accelerate Time-to-Certification for IPs and SoCs

VC Functional Safety Manager Reduces ISO 26262 FMEA/FMEDA and Fault Classification Effort by Up to 50 Percent

MOUNTAIN VIEW, Calif., Oct. 7, 2019 /PRNewswire/ --

Highlights:

- Automation is required to address the challenging certification requirements and increased efforts associated with new automotive IPs and SoCs
- Industry's first and most comprehensive functional safety verification solution includes unified FMEA/FMEDA and fault classification automation, powerful verification engines, ISO 26262-certified tools, and expert services
- Anticipated increase in effort from functional safety verification can be reduced by up to 50 percent using this new unified solution

Synopsys, Inc. (Nasdaq: SNPS) today announced the industry's first and most comprehensive unified functional safety verification solution to accelerate time to ISO 26262 certification for automotive IP and semiconductor companies targeting the highest Automotive Safety Integrity Levels (ASIL D). As part of the solution, Synopsys introduced VC Functional Safety Manager, a FMEA/FMEDA and fault classification automation technology enabling architects, IP designers, and verification engineers to accelerate their functional safety verification with productivity gains up to 50 percent compared to traditional manual and error-prone functional safety verification point tools.

"Arm strongly believes safety will be critical to the successful deployment of advanced ADAS and autonomous solutions," said Neil Stroud, senior director of technology strategy, Automotive and IoT Line of Business, Arm. "With ISO 26262 compliance and functional safety verification requirements increasing for

SYNOPSYS®

DATASHEET

VC Functional Safety Manager

Accelerate functional safety certification of IP and SoC with comprehensive FMEA/FMEDA and fault campaign management tool

Overview

Synopsys® VC Functional Safety Manager provides a comprehensive tool for IP and semiconductor groups targeting functional safety certification for ISO 26262, IEC 61508 and other functional safety standards. It serves the needs of IP and SoC architects, IP designers and verification engineers by providing a scalable and automated solution for Failure Modes and Effect Analysis (FMEA), unified fault campaigns management, annotation and calculation of metrics for the Failure Modes, Effects and Diagnostic Analysis (FMEDA), and configuration of work products for delivery to assessors and customers.

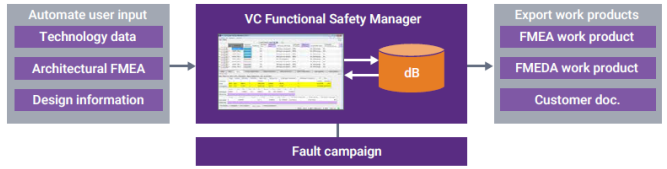
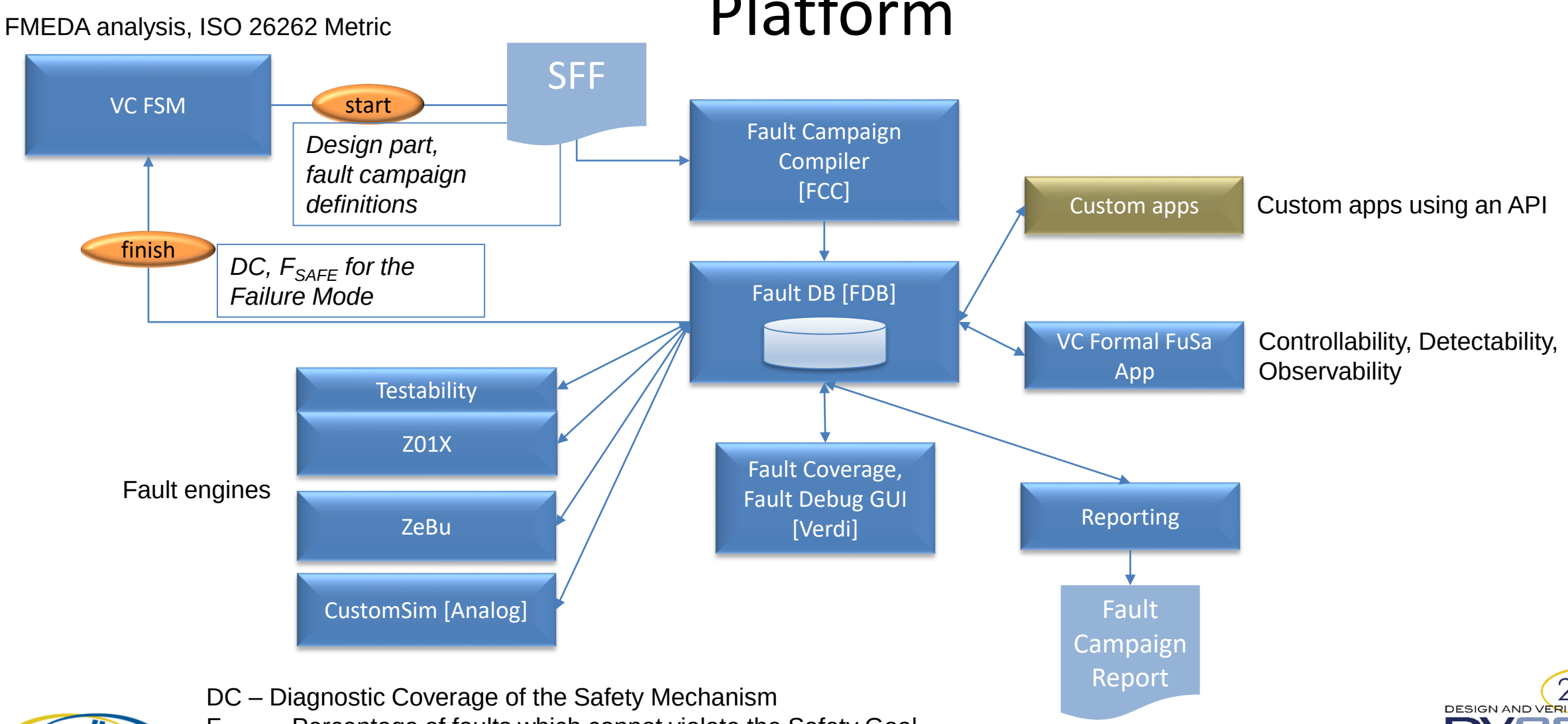


Figure 1: VC Functional Safety Manager Automates Functional Safety Verification tasks

Scalable and Collaborative

VC Functional Safety Manager delivers a scalable and collaborative FMEA/FMEDA solution. Existing tool do not scale with data, design and team size, are subject to

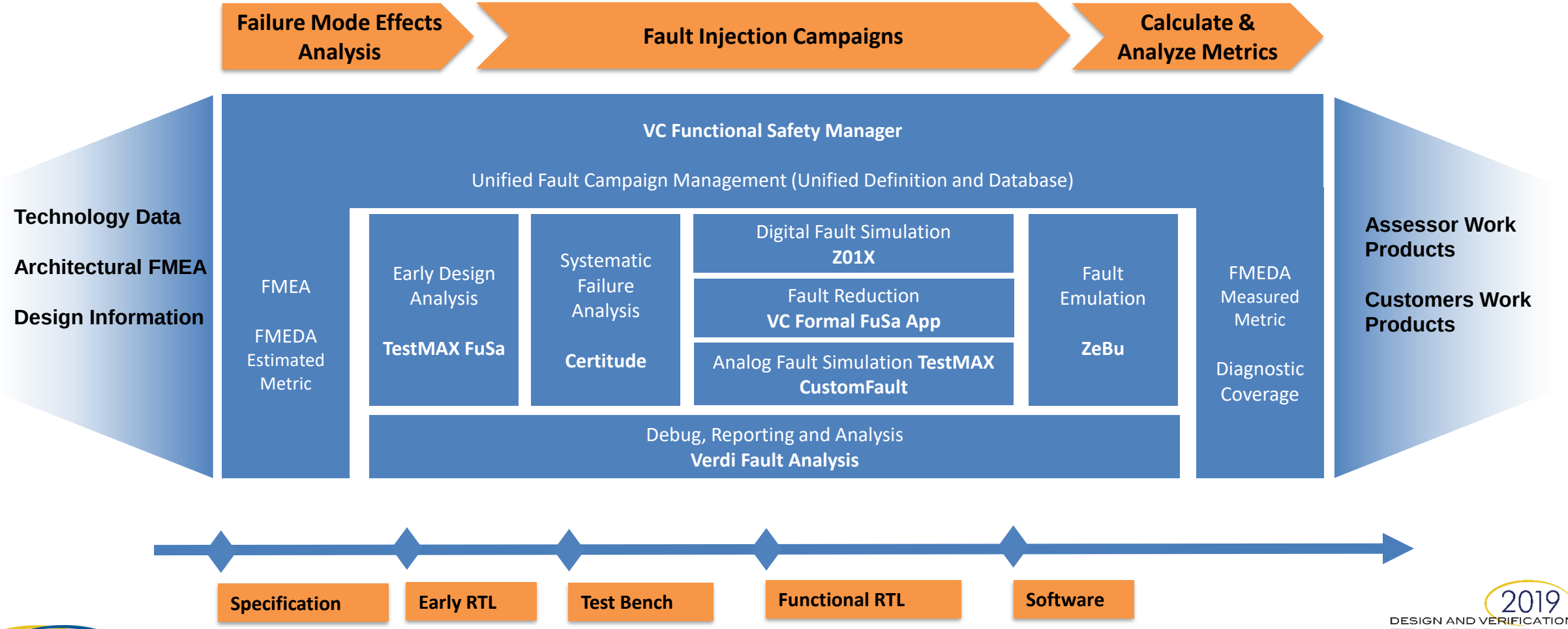
Full Solution for Fault Classification – Unified Platform



DC – Diagnostic Coverage of the Safety Mechanism

F_{SAFE} – Percentage of faults which cannot violate the Safety Goal

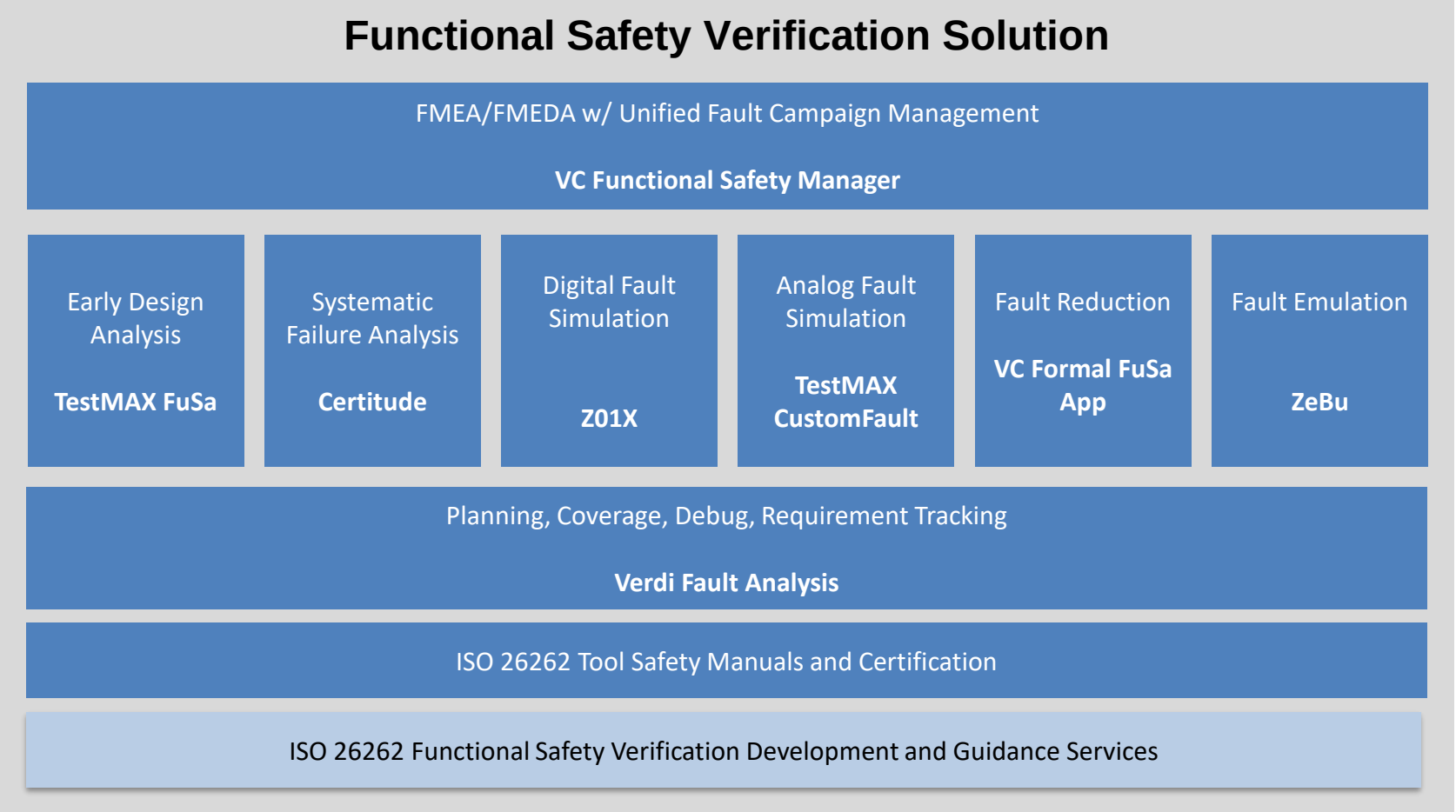
Unified Fault Campaign Ensures Efficiency and Consistency



Comprehensive Functional Safety Verification Solution

Highest Productivity to Accelerate Time to Compliance

- Unified FMEA/FMEDA and fault campaign automation
- Fastest fault campaign engines with unified debug and reporting
- Tool chain certification
- Expert guidance based on proven hands-on experience



Questions