
Verifikation und Test des PROFIsafe-Sicherheitsprofils



Mario Friske, Holger Schlingloff

Fraunhofer FIRST

Herbert Barthel

Siemens AG

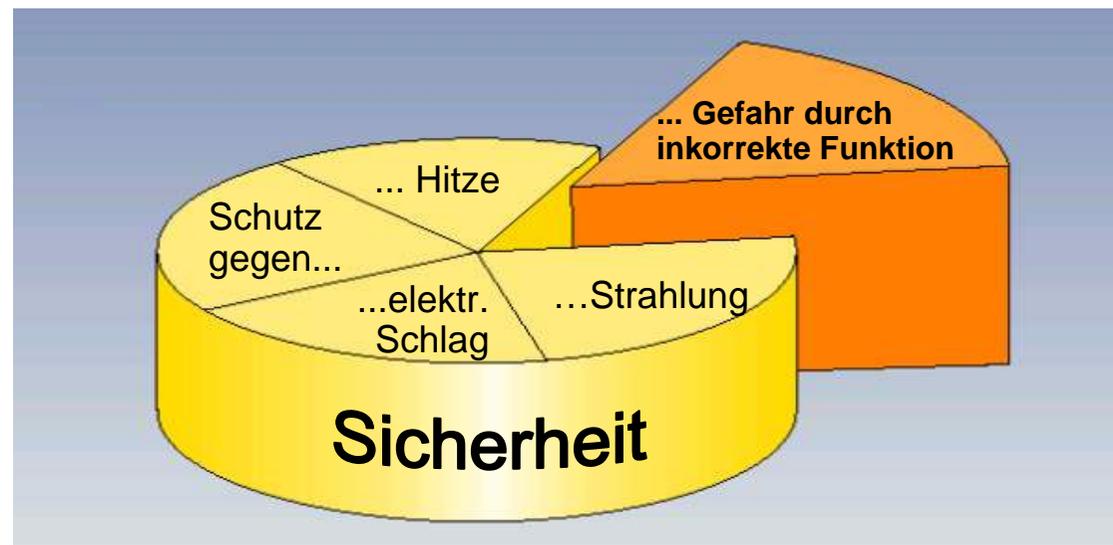
26. Treffen der GI-FG TAV,
Stuttgart, 07.12.2007

Inhalt

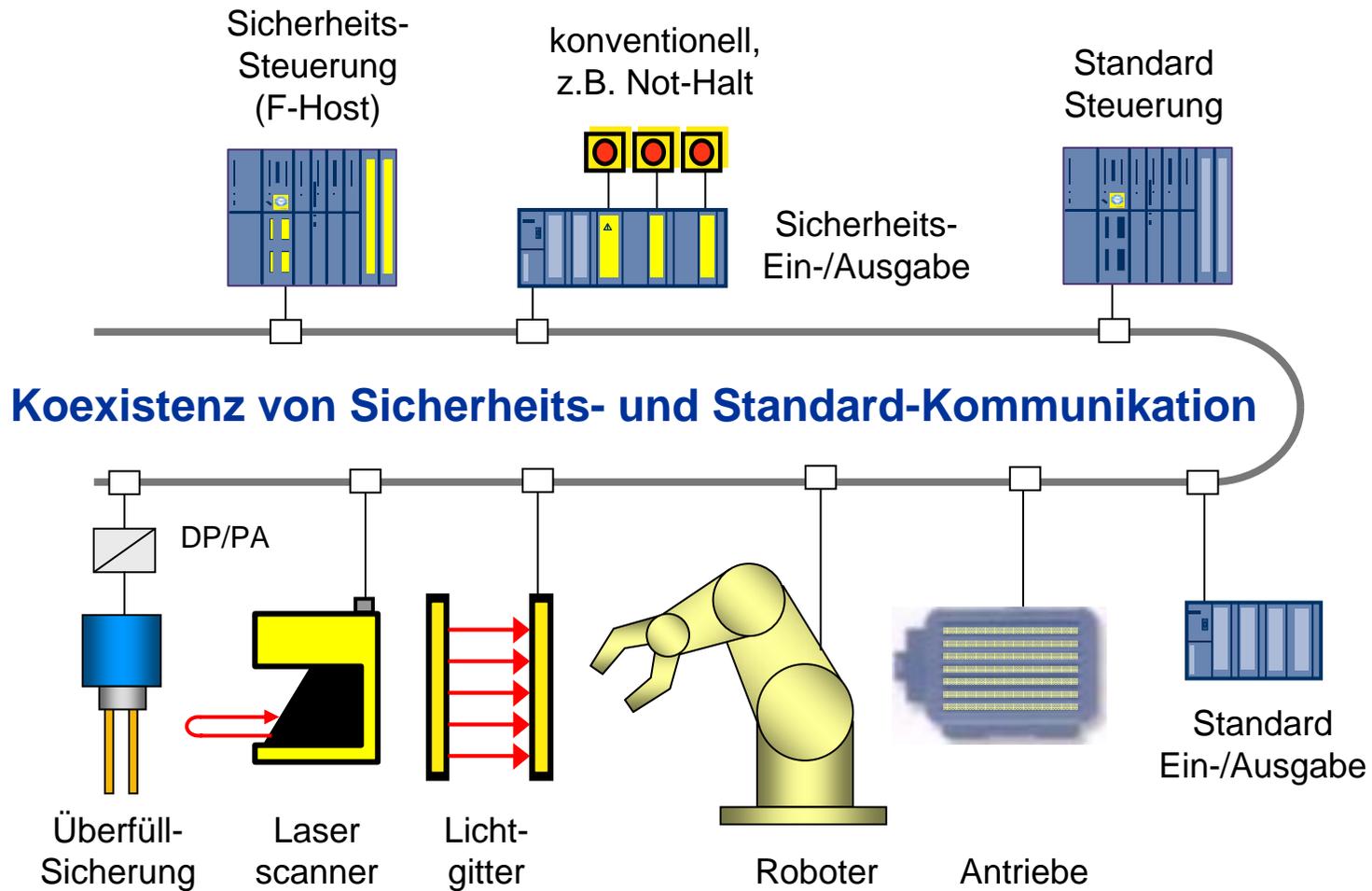
1. Einleitung: PROFIsafe
2. Modellprüfung
3. Testgenerierung aus Zustandsdiagrammen
4. Verbesserung der Testabdeckung
5. Zusammenfassung

PROFIsafe und “Funktionale Sicherheit”

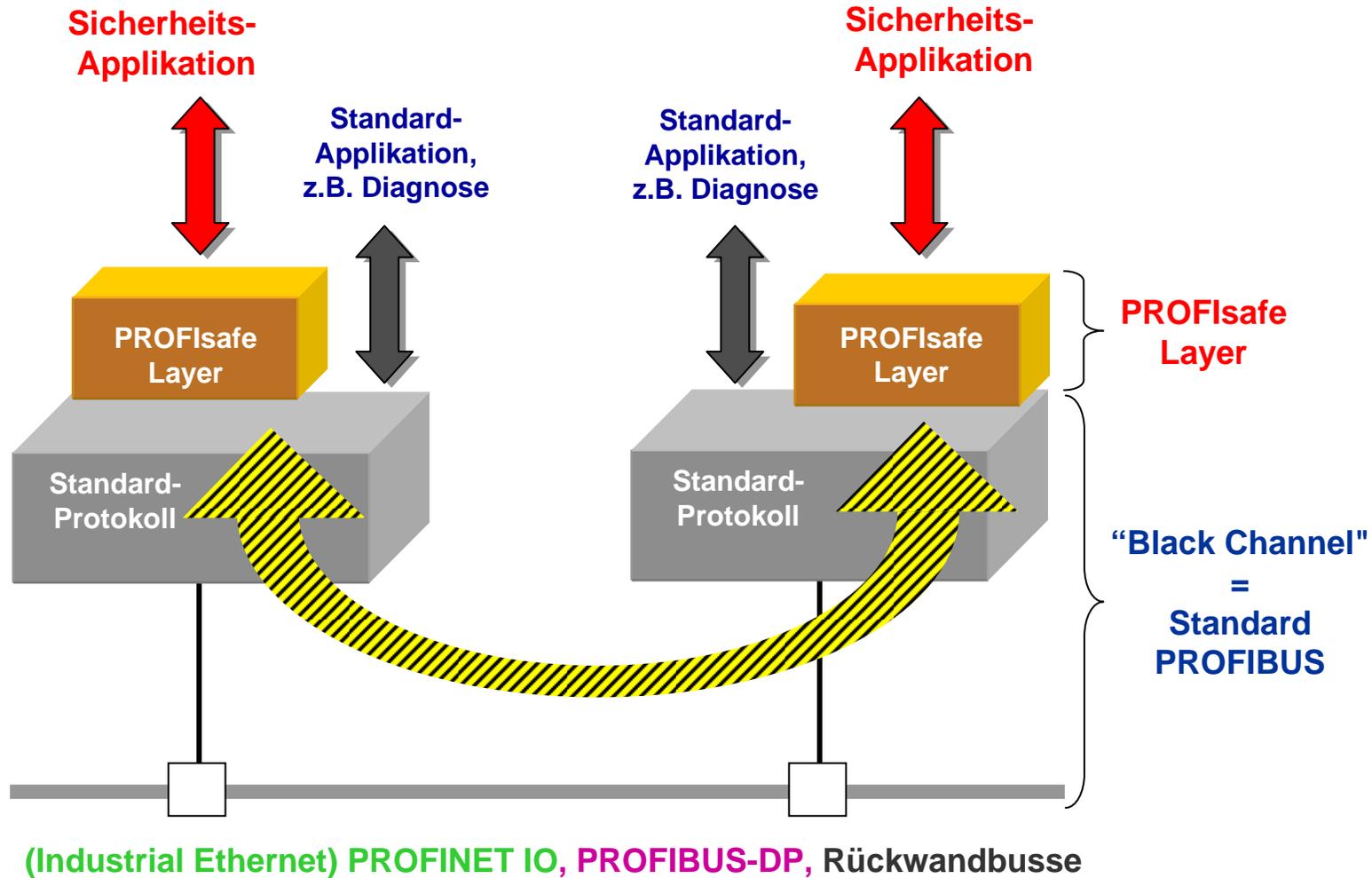
- Sicherheit fordert *Schutz vor Gefährdung* durch verschiedene Ursachen (Bewegung, Hitze, Radioaktivität, el. Schlag, etc.)
- “Funktionale Sicherheit” bedeutet Schutz vor Gefährdung verursacht durch *inkorrekte Funktion*.
- PROFIsafe's Anteil daran ist die *korrekte Übertragung* von Nachrichten
- Zertifizierung gemäß IEC 61508



Ziel von PROFIsafe



Kommunikation über „Black Channel“



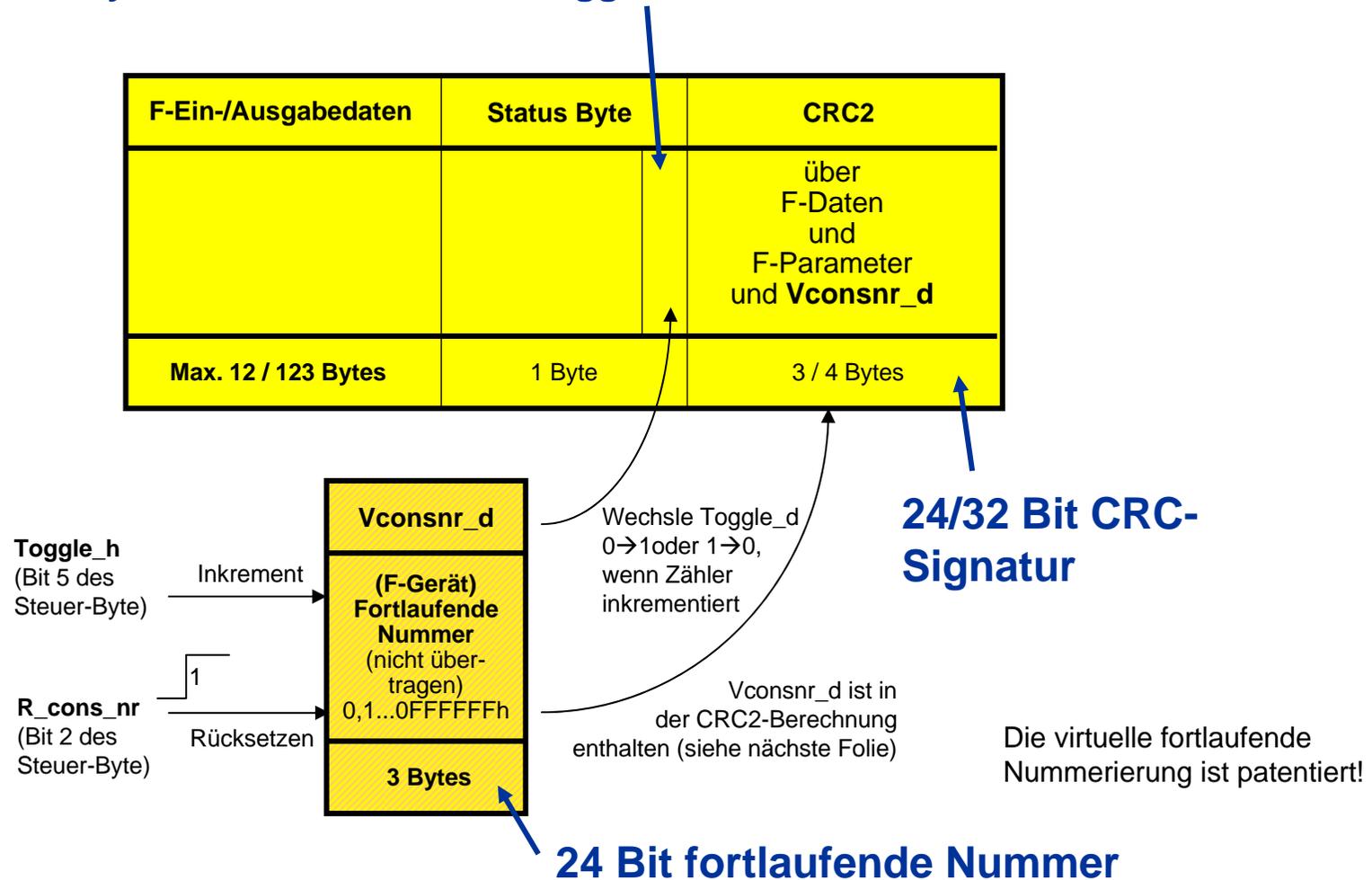
Robustheit gegenüber Kommunikationsfehlern

Abhilfe durch PROFIsafe: Fehlertyp:	(Virtuelle) Fortlaufende Nummerierung	Zeitüberwachung mit Quittung	Codename für Sender und Empfänger	Daten- Konsistenz- Überwachung
Wiederholung	✓			
Verlust	✓	✓		
Einfügung	✓	✓	✓	
Falsche Reihenfolge	✓			
Datenverfälschung				✓
Verzögerung		✓		
Maskerade (Standard Nachricht "spielt" sicher)		✓	✓	✓
Wiederkehrende Spei- cherfehler in Switches	✓			

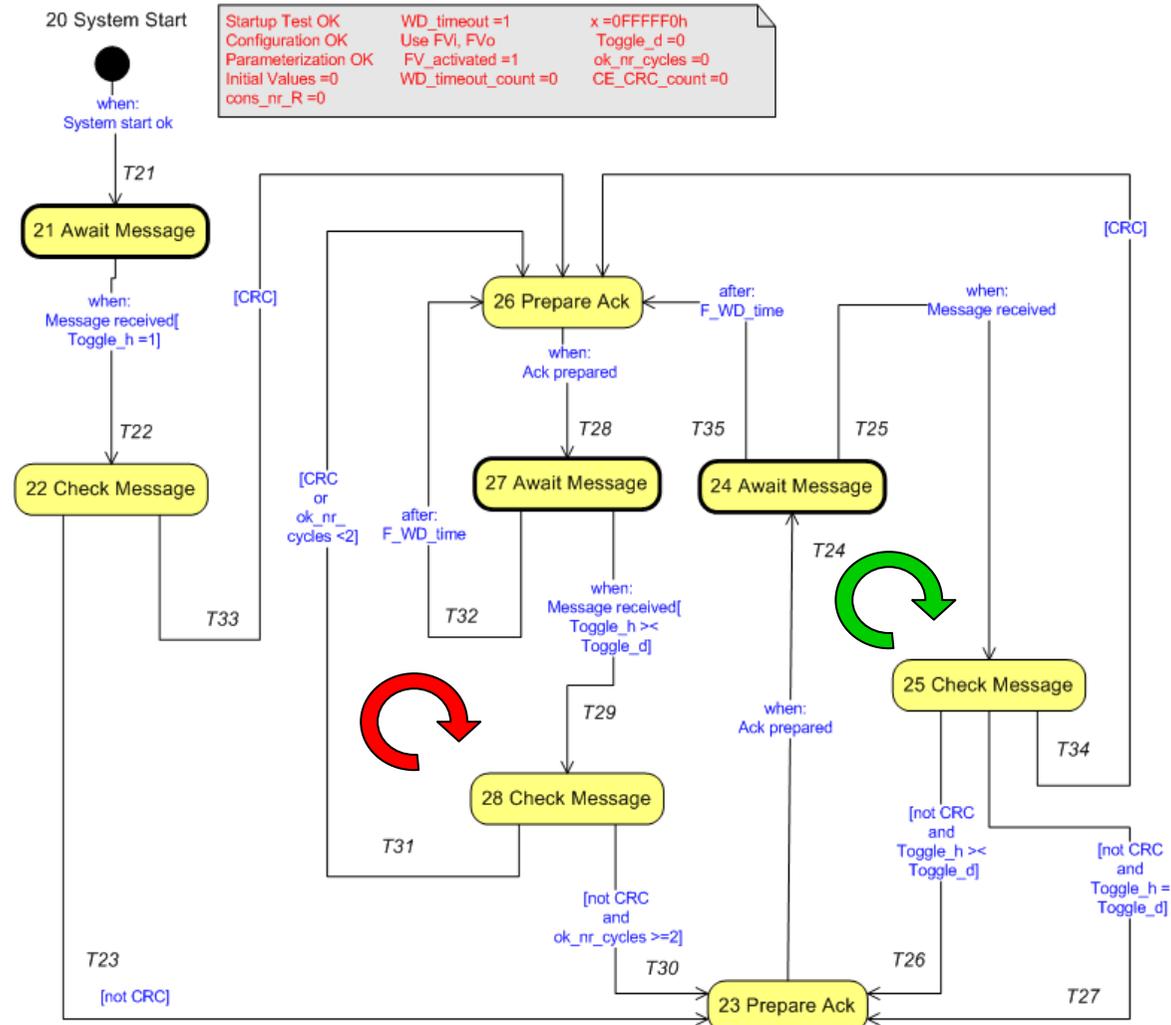
- Fortlaufende Nummerierung (Consecutive Number)
- Zeitüberwachung (Watchdog)
- Authentizität (Codename)
- Datenintegrität (Cyclic Redundancy Check = CRC)

Aufbau eines PROFIsafe-Telegramms

Synchronisation durch "Toggle Bit"



Protokollspezifikation durch Zustandsmaschinen



Zugehörige Transitionsbeschreibungen

TRAN-SITION	SOURCE STATE	TARGET STATE	ACTION
T1	1	2	use FV, activate_FV =1, FV_activated_S =1 Toggle_h =1
T2	2	3	send safety PDU
T3	3	4	restart host-timer
T4	4	5	old_x =x, x =x+1, if x =01000000h then x =1 Toggle_h = not Toggle_h, if FV_activated =1 or activate_FV_C =1 or Device_Fault =1 then use FVi, FV_activated_S =1 else use PVi, FV_activated_S =0 if activate_FV_C =1 or Device_Fault =1 then use FVo, activate_FV =1 else use PVo, activate_FV =0 iPar_OK_S =iPar_OK
T5	5	6	send safety PDU
T6	6	4	restart host-timer
T7	6	7	-
T8	7	5	if FV_activated =1 or activate_FV_C =1 or Device_Fault =1 then use FVi, FV_activated_S =1 else use PVi, FV_activated_S =0

Zertifizierung

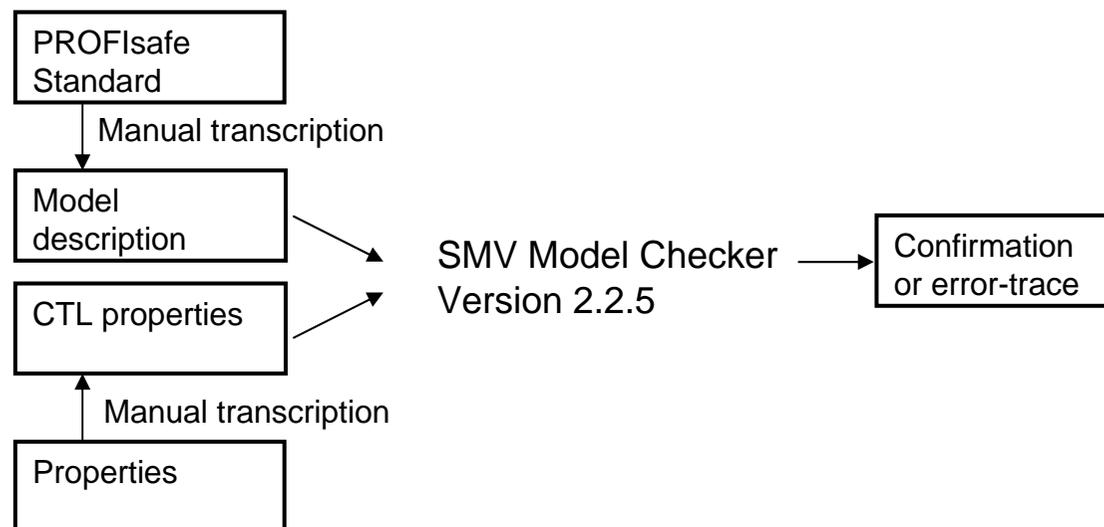
- Zulassung des Protokolls auch für sicherheitskritische Anwendungen (TÜV und BGIA)
- IEC 61508 empfiehlt formale Verifikation der wesentlichen Sicherheitseigenschaften ab Sicherheitsanforderungsstufe 2 (SIL 2-4)
- Problem: Identifikation und Nachweis relevanter Eigenschaften
- IEC 61508 fordert funktionale und Black-Box-Tests für alle Sicherheitsanforderungsstufen (SIL 1-4) („highly recommended“)
- Problem: Erstellung einer Testsuite für zukünftige PROFIsafe-Implementierungen

Modellprüfung

- Zielsetzung: Verifikation von globalen Kommunikationseigenschaften
 - Verklemmungsfreiheit, Lebendigkeit, Sicherheit
- Standardtechnologie im Hardware-Design
 - Kommerzielle und nichtkommerzielle Werkzeuge verfügbar
- Problematik hier: relativ viele Zustandsvariablen
 - „Software Model Checking“

Vorgehensweise

- Übertragung der Host- und Slave/Device-Zustandsmaschinen in die Modellierungssprache SMVL (Symbolic Model Verification Language)
- Übertragung von Eigenschaften in die Beschreibungssprache CTL (Computational Tree Logic)
- Geeigneter Aufruf von SMV
- Vergleich der Gegenbeispiele mit der Spezifikation



Details und Ergebnisse zu Verifikation

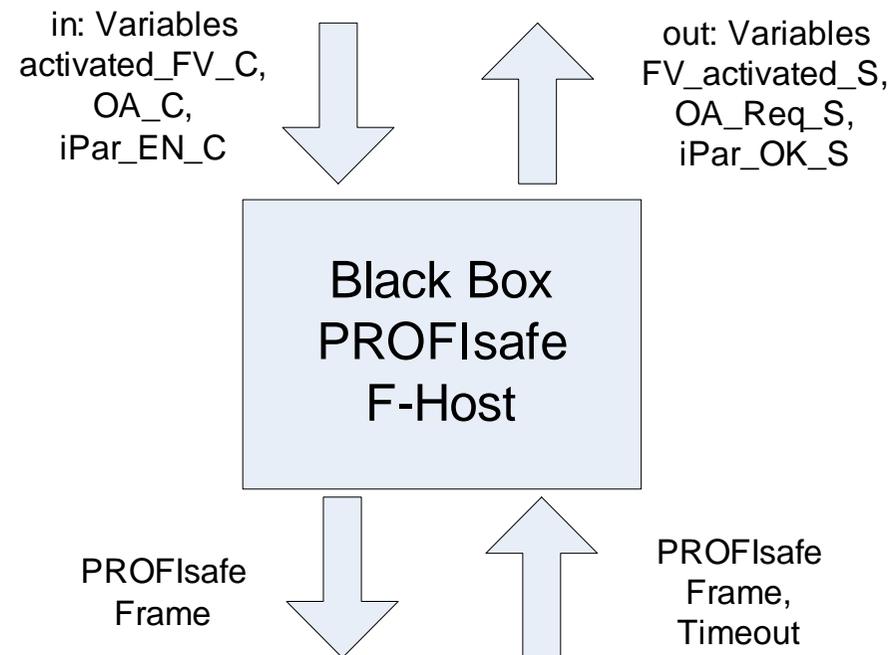
- Beherrschung der Zustandsraumexplosion durch geschickte Variablenordnung
- Abstraktion von Zustandskomponenten
- Reduktion der Breite von Zählvariablen

- Nachweis aller geforderten Eigenschaften erfolgt
- Zustandsraum: $8 \cdot 10^{13}$ erreichbare Zustände

```
NuSMV > check_spec -p "(AG EF Host.state=Await_Device_Ack_6_A &
AG EF Host.state=Await_Device_Ack_9_A &
AG EF Host.state=Check_Device_Ack_4_A &
AG EF Host.state=Check_Device_Ack_7_A &
AG EF Host.state=Check_Device_Ack_10_A &
AG EF Host.state=Prepare_Message_5 &
AG EF Host.state=Prepare_Message_8 &
AG EF Host.state=wait_delay_time_11)"
NuSMV > -- specification ((((((AG EF Host.state = Await_Device_Ack_6_A & AG EF Host.state = Await_Device_Ack_9_A) & AG
EF Host.state = Check_Device_Ack_4_A) & AG EF Host.state = Check_Device_Ack_7_A) & AG EF Host.state =
Check_Device_Ack_10_A) & AG EF Host.state = Prepare_Message_5) & AG EF Host.state = Prepare_Message_8) & AG EF
Host.state = wait_delay_time_11) is true
```

Blackbox-Test von Host und Slave/Device

Application = Upper Tester



Profibus = Lower Tester

Testgenerierung aus Zustandsmaschinen

- Seit längerem Gegenstand von Forschung und Entwicklung
- Zustandsmaschine wird häufig „flachgeklopft“ mit anschließender Tiefensuche
- Alternative: Modellprüfung mit Fehlerinjektion, Gegenbeispiele dienen als Testfälle
- Kommerzielle und experimentelle Werkzeuge verfügbar

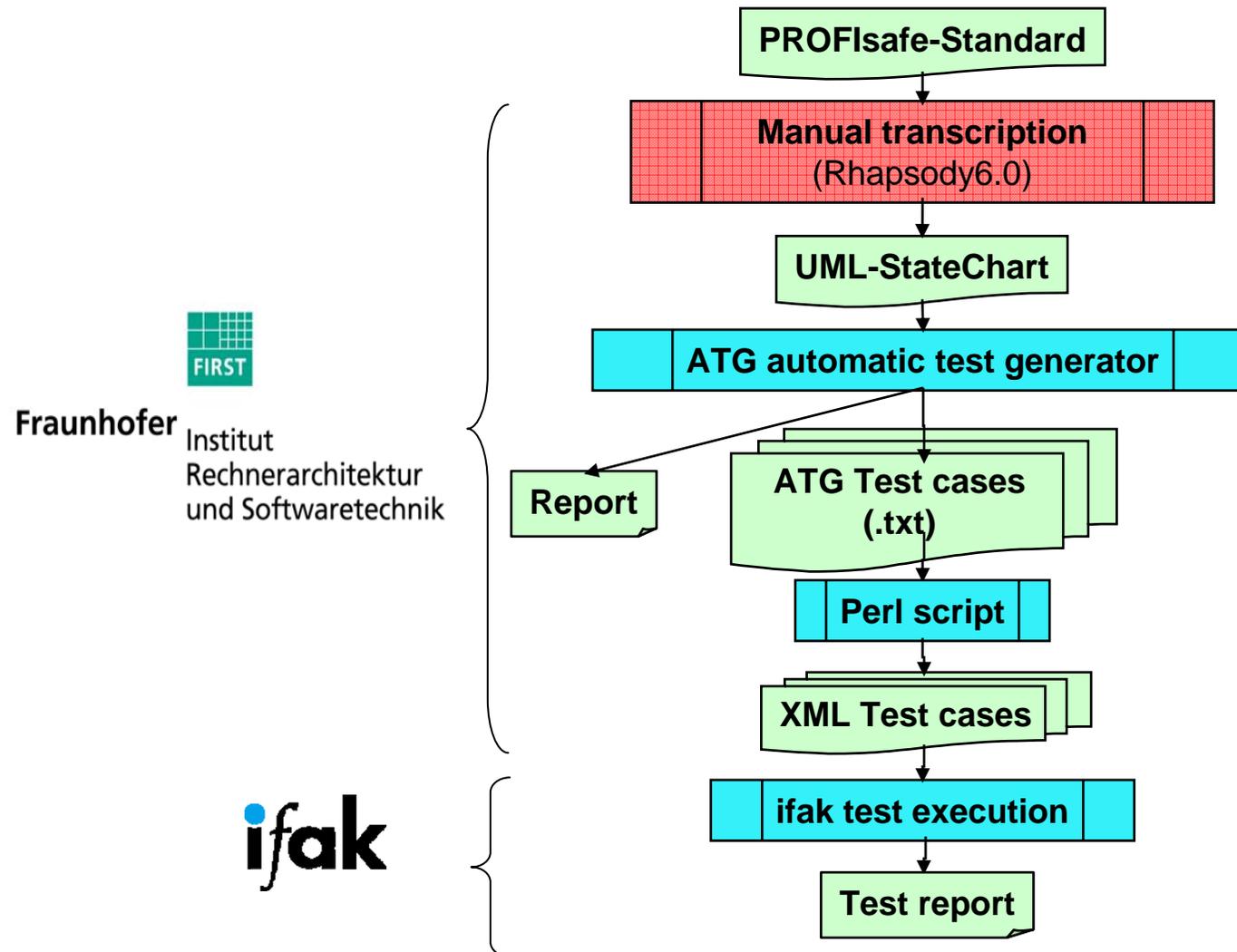
The logo for Telelogic, featuring the word "Telelogic" in a bold, italicized, blue sans-serif font.The logo for Reactis, featuring the word "Reactis" in a red sans-serif font with a registered trademark symbol (®) to the upper right.The logo for CONFORMIQ, featuring the word "CONFORMIQ" in a bold, black sans-serif font, with the letter "Q" in green.

- Typische Abdeckungskriterien: All-States, All-Transitions, Modified Condition / Decision Coverage (MCDC)
- Zielorientierte Ansätze: vorteilhaft für Zertifizierung => Anwendung für den Test des PROFIsafe-Sicherheitsprofils

Verfahren zur Testgenerierung für PROFIsafe

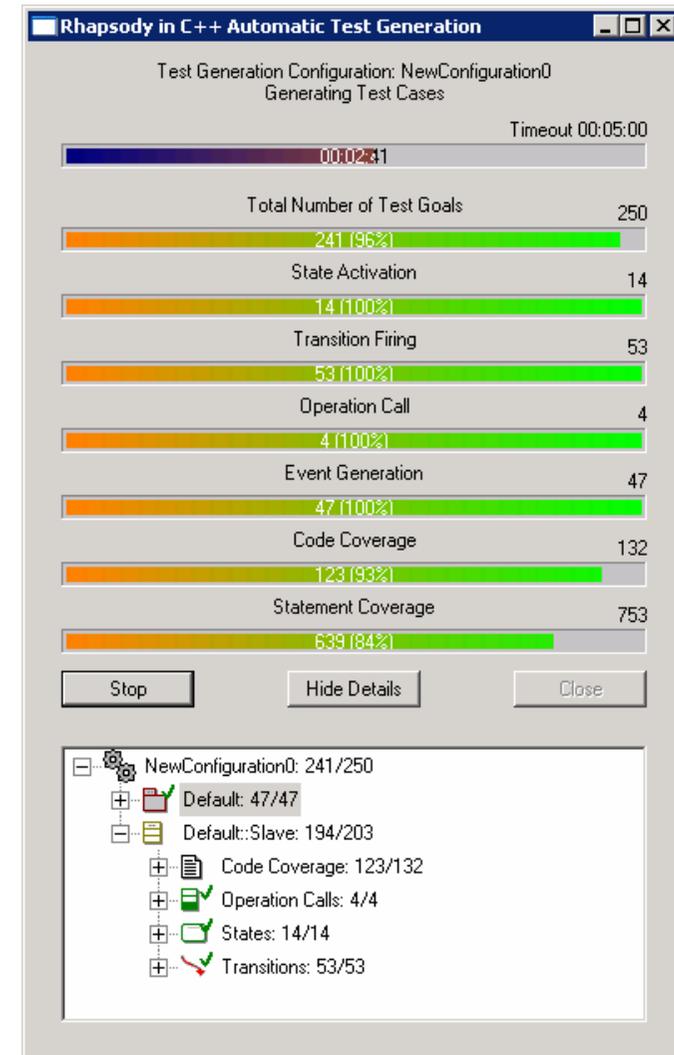
- 1:1-Umsetzung der Zustandsmaschinen in UML-Zustandsdiagramme (Telelogic Rhapsody in C++)
- Bestimmung von Schnittstellen des Typs „required“ and „provided“
- Verwendung eines Generators, z.B. ATG (Automatic Test Generator) zur automatischen Erzeugung der Testsuite
- Aufbereitung der Ergebnisse:
 - Formatkonvertierungen
 - Entfernung doppelt vorhandener Testfälle
- Vervollständigung der Abdeckung zu 100% gemäß der definierten Kriterien, siehe All-3-Transitions-Abdeckung

Werkzeugkette zum automatisierten Test



Erzielbare Abdeckung

- ATG erstrebt MCDC auf dem generierten Code:
 - es war zu zeigen, dass jede atomare Teilentscheidung einer Bedingung den Wert der Gesamtentscheidung unabhängig beeinflusst
 - Zustandsmaschinen werden in Case-Anweisungen übersetzt
 - Subsumiert All-Transitions, All-Events und All-Local-States
 - C++-Code in Actions wird unverändert hinzugefügt
 - Zielorientierter Ansatz
 - Vervollständigung der Abdeckung zu 100% gem. Kriterien



Erreichte Ziele bei PROFIsafe

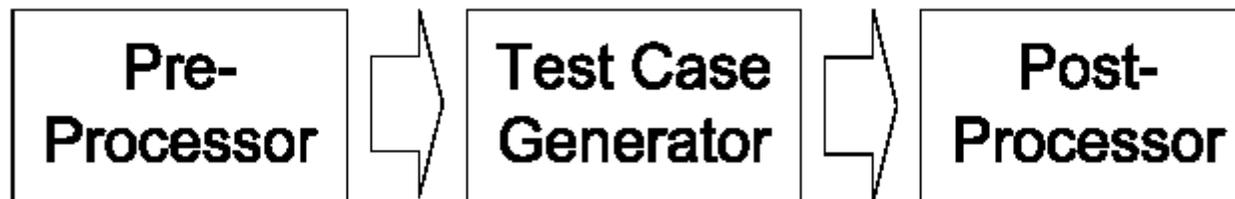
- 100% Testabdeckung gem. Standardkriterien (MCDC, All-Transitions, All-Events und All-Local-States)
- Hinreichend für Sicherheitsanwendung bis SIL3
- Nachweislich vollständiges Erreichen der Abdeckung durch detaillierte Betrachtung aller erforderlichen Testziele:
 1. Vervollständigung der Testsuite durch Testerzeugung mittels Simulation der Zustandmaschine, oder
 2. ggf. detaillierter Nachweis der Unerfüllbarkeit einzelner Testziele, z.B. bedingt durch Nichterreichbarkeit im generierten Code (Diese unerfüllten Testziele sind nachweislich sicherheitstechnisch irrelevant.)

Verbesserungsmöglichkeiten?

- Konformitätstest
 - Prüft die Relation zwischen Zustandsmaschinen (nicht anwendbar)
 - sichere Zustandsidentifikation nicht garantiert
 - Rücksetzen des SUT oft aufwendig oder nicht möglich
- Spezifikationsbasierter Test
 - Vollständigkeit und Minimalität der Testsuite im Allgemeinen nicht vorrangig
 - ein Testfall pro Anwendungsfall unzureichend
 - anpassbare Abdeckungskriterien erwünscht

Erhöhung der Abdeckung

- Möglichst Verwendung eines kommerziellen Testgenerators:
 - Betriebsbewährtheit
 - Grundlage zur Erhöhung der Abdeckung
 - Pre-Processing zur automatisierten Anreicherung des Modells
 - Post-Processing zur Kombination und Filterung der Testfälle
- Betrachtung der Abdeckung als Produkt einer *Generator Coverage Function* und einer *Enhancement Function*



Transition Instrumentation

- Möglichkeiten zur Anreicherung des Modells
 - Zusätzliche Zustände (und Transitionen)
 - vergrößert das Modell, aber nicht unbedingt die Abdeckung
 - Zusätzliche Variablen and Bedingungen an Transitionen
 - ursprüngliche Zustandsmaschine bleibt erhalten
 - anwendbar zur Definition zusätzlicher Testziele für den Testfallgenerator (“user defined test goals”)
 - Zusätzliche Ereignisse an Transitionen
 - anwendbar um zusätzliche Information an den Post-Processor zu geben, z.B., Zustandsidentifikation, Variablenwerte etc.
- Entwicklung zweier Methoden:
 1. Transparent Transition Instrumentation
 2. Extended Transition Instrumentation

Transparent Transition Instrumentation

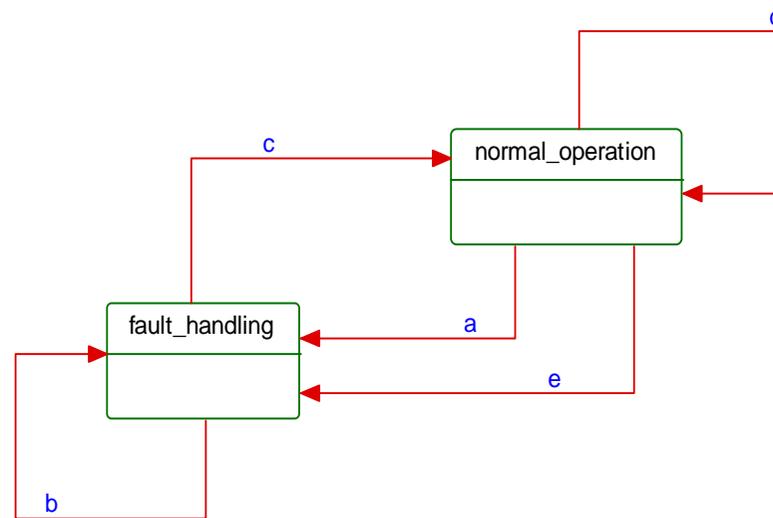
- Einfügen einer Bedingung mit einer zusätzlichen C++-Variablen in den Action-Code führt zum Generieren von zwei Testfällen
- z.B. um Transitionssequenz $t_1 \rightarrow t_2 \rightarrow t_3$ als Testziel zu erreichen, Hinzufügen folgendes Action-Codes:

```
if(counter==1){counter++;} to t1  
if(counter==2){counter++;} to t2  
if(counter==3){counter++;} to t3
```

Erzwingt Generierung der gewünschten Transitionssequenz $t_1 \rightarrow t_2 \rightarrow t_3$
- Instrumentierung des Modell per Hand oder Skript
- Technik eignet sich zur Realisierung aller sequenzbasierten Abdeckungskriterien

Beispiel: All-3-Transitions-Abdeckung

- Berechnung aller Sequenzen der Länge 3 aus einer Abstraktion der Zustandsmaschine
- Instrumentierung der Zustandsmaschine; Testgenerator bestimmt weitere notwendige Transitionen
- Hinzufügen von Zählvariablen für folgende Sequenzen: abc, aca, acd, bca, bcd, cab, cac, cda, dab, dac, ebc, ece, ecd, bce, ceb, cec, cde, deb, dec, ace, eca, abb, ebb, bbb, bbc.



Extended Transition Instrumentation

- Transparent Transition Instrumentation:
 - definierte Abdeckung, z.B. All 3-Transitions + MCDC
 - flexible Unterstützung üblicher Abdeckungskriterien
 - erzeugt viele kurze Testfälle
 - Prüfung des Testobjekts in der Breite, jedoch nicht Tiefe
- Nachbereitung zur Kombination von kurzen Testfällen zu längeren Testsequenzen
 - Erhaltung der Abdeckung
(e.g. adaeafdbdcecffgghdagfebeahfhebfhhgecgdcdbgdbech)
 - Sicherstellung, dass Testsequenzen "zusammenpassen"
- Realisierung durch zusätzliche "notifier events"
 - Anzeige des Zustands von globalen Variablen
 - Anzeige enthaltener Transitionssequenzen
 - Berücksichtigung bei Kombination durch Post-Prozessor

Zusammenfassung

- Verifikation und Test eines sicherheitsgerichteten Kommunikationsprofils
- Nachweisliches Erzielen von 100% Abdeckung gemäß definierter Kriterien, Zertifizierung nach SIL 3
- Verbesserung und Vervollständigung der Spezifikation
- Entwicklung einer allgemeinen Methode und Werkzeugkette zur Erhöhung der Testabdeckung

Weitergehende Fragestellungen:

- Bessere Kopplung von Verifikation und Test?
- Theorie spezifikationsbasierter Abdeckungskombination?
 - Fehleraufdeckungspotential
 - Implikationen für Sicherheitsstandards