Verifikation und Test des PROFIsafe-Sicherheitsprofils

Holger Schlingloff [1], Mario Friske [1], Herbert Barthel [2]

[1] Fraunhofer FIRST, Kekuléstr. 7, 12489 Berlin, {holger.schlingloff,mario.friske}@first.fraunhofer.de

[2] Siemens AG, Automation and Drives, Industrial Automation Systems, Research and Development (A&D AS RD DH N 3), Gleiwitzer Str. 555, 90475 Nürnberg, herbert.barthel@siemens.com

Zusammenfassung. PROFIsafe [1] ist ein nach IEC 61508 [6] entwickeltes und geprüftes Sicherheitsprofil für die weit verbreiteten Feldbusprotokolle PROFIBUS [7] und PROFINET [8]. Es wurde von einem Konsortium von mehr als 20 internationalen Herstellern und Endanwendern definiert. Der 2007 publizierte IEC-Standard [9] enthält informelle, semiformale und formale Bestandteile.

In diesem Papier berichten wir über Aktivitäten zur Verifikation und zum automatisierten Test der PROFIsafe-Sicherungsschicht. Aus den Beschreibungen in den PROFIsafe-Spezifikationen wurden von uns formale Modelle (Transitionssysteme bzw. UML-Zustandsmaschinen) erstellt, die als Eingabe für einen Modellprüfer und einen Testgenerator dienten. Durch die unabhängige Modellierung wurden dabei Klarstellungen der Spezifikation erreicht. Mit der Modellprüfung konnten sicherheitsrelevante Eigenschaften der Spezifikation nachgewiesen und mit der Testgenerierung vollständige Testsuiten nach definierten Abdeckungskriterien erstellt werden. Aufgrund der weitgehenden Automatisierung von Modellprüfung und Testgenerierung konnten Erweiterungen des Standards sofort in die Validierung einbezogen werden. Die Ergebnisse der Validierung waren Teil der Zertifizierungsunterlagen zur Zulassung gemäß Sicherheits-Integritätslevel 3 (SIL-3). Die erzeugte Testsuite dient nun zur Überprüfung der Konformität von Endgeräten mit den in den Standardisierungsdokumenten festgelegten Funktionalitäten.

1 Motivation

In der Automatisierungstechnik werden proprietäre und spezialisierte Bussysteme zunehmend durch standardisierte und universelle Lösungen ersetzt. Beispiele sind der 1999 zum IEC-Standard erhobene PROFIBUS (Process Field Bus) [7] und das Ethernet-basierte Echtzeitprotokoll PROFINET [8], das derzeit vor allem im Bereich der Fertigungstechnik eingesetzt wird. Zur *funktionalen Sicherheit*, d. h. zum Schutz vor Gefährdung verursacht durch inkorrekte Funktion von Geräten, fordert die Norm IEC 61508 [6] in diesem Bereich unter anderem eine korrekte und zuverlässige Übertragung von sicherheitsgerichteten Nachrichten. In der Vergangenheit waren Sicherheitseinrichtungen in einer separaten Ebene auf parallel verdrahtete Relaistechnik angewiesen.

PROFIsafe ist ein neues Sicherheitsprotokoll, welches als zusätzliche Schicht auf der Transportschicht von PROFIBUS und PROFINET aufsetzt und eine korrekte Übertragung von sicherheitsrelevanten Nachrichten auch über ein unzuverlässiges Medium garantiert. PROFIsafe-Nachrichten können dabei zusammen mit funktionalen Nachrichten auf demselben Kanal koexistieren, und es ist möglich, Sicherheitssteuerung und funktionale Steuerung auf einem einzigen Gerät zu realisieren. Wie in Abbildung 1 dargestellt, wird aufbauend auf einem Standardprotokoll (PROFIBUS/PROFINET) bei PROFIsafe der Übertragungskanal als „black channel“ betrachtet und durch zusätzliche Maßnahmen gegen Kommunikationsfehler abgesichert. Das Protokoll gewährleistet eine sichere Übertragung auch bei verlorenen, wiederholten oder verfälschten Nachrichten, falscher Reihenfolge, Verzögerungen und wiederkehrenden Speicherfehlern. Als Maßnahmen werden dabei CRC-Signaturen, fortlaufende Nachrichtennummern, Sender/Empfängerauthentifizierung und Zeitüberwachungsmechanismen eingesetzt.

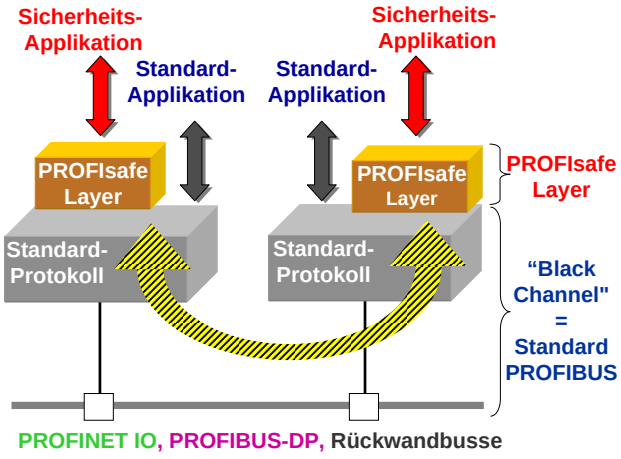

Abbildung 1: Das PROFIsafe-Sicherheitsprofil

PROFIsafe wurde in den Jahren 1999-2005 von einem internationalen Industriekonsortium definiert (standardisiert in IEC 61784-3-3) und nach IEC 61508 für Systeme der Sicherheits-Integritätslevel 3 (SIL-3) zugelassen. Das Protokoll ist dabei weitgehend durch zwei Zustandsmaschinen (*F-Host* und *F-Device*) definiert, wobei ein Host (Steuerrechner) mit beliebig vielen Devices (Endgeräten) kommunizieren kann. Zwischen Host und Device werden Nachrichten in einem definierten Format ausgetauscht. Jede Zustandsmaschine enthält etwa ein Dutzend Zustandsvariablen. Zustandsübergänge werden abhängig vom Wert dieser Zustandsvariablen und von der aktuellen Nach-

richt durchgeführt. Jeder Zustandsübergang löst eine Aktion in dem betreffenden Gerät aus, z. B. Versenden einer Nachricht, Rücksetzen einer Stoppuhr, Aktivierung eines Alarmsignals usw.

Als internationaler Standard ist PROFIsafe offen für Steuergeräte und Anlagen verschiedener Hersteller, die im Betrieb PROFIsafe-gerecht miteinander kommunizieren müssen. Dadurch ergeben sich die beiden folgenden Problemstellungen:

1. Es muss sichergestellt sein, dass Geräte, die den Standard korrekt implementieren, sich nicht gegenseitig beeinträchtigen können. Insbesondere muss ein Host eines Herstellers mit Devices anderer Hersteller zusammenarbeiten können. Es darf zu keinen Verklemmungen und Endlosschleifen kommen.

2. Es muss eine Möglichkeit vorgesehen werden, mit der die Konformität eines Gerätes zum Standard nachgewiesen werden kann. Insbesondere müssen Geräte unabhängig von bestehenden anderen Geräten getestet werden können, damit eine PROFIsafe-gerechte Eingliederung in bestehende Anlagen möglich wird.

Zur Lösung dieser Probleme wurde von Fraunhofer FIRST eine Verifikation des Protokolls durch Modellprüfung durchgeführt, und es wurden Referenz-Testsuiten für neue Hosts und Devices erstellt. Nachfolgend werden die Methoden und Ergebnisse dieser Arbeiten beschrieben, sowie einige dadurch neu aufgeworfene wissenschaftliche und methodische Fragestellungen dargestellt. Die Arbeit ist wie folgt strukturiert: Im Abschnitt 2 beschreiben wir unseren Ansatz zur Verifikation von PROFIsafe. Im Abschnitt 3 stellen wir die Methodik zur automatischen Erzeugung der Referenz-Testsuiten dar. In Abschnitt 4 geben wir einige Beispiele für die von uns erzielten Ergebnisse in Bezug auf die Protokollspezifikation an. In Abschnitt 5 diskutieren wir Möglichkeiten für weitere methodische Verbesserungen und Verallgemeinerungen.

2 Verifikation

Bei der Validierung sicherheitsrelevanter Applikationen ist es wichtig, auf betriebsbewährte Werkzeuge zurückzugreifen. Zur Verifikation wurde daher der seit 1998 verfügbare Modellprüfer *nuSMV* (v 2.2.5) [11] eingesetzt, der auf dem klassischen *Symbolic Model Verifier (SMV)* aufbaut. nuSMV ist ein symbolischer Modellprüfer, der eine vollständige Zustandsraumsuche für verteilte Transitionssysteme realisiert. Die Eingabesprache *SMV Language (SMVL)* erlaubt eine hierarchische Modellierung der Kommunikationssysteme mit aussagenlogischen Ausdrücken. Besonderheiten sind die automatische Analyse von Invarianten, die konjunktive und disjunktive Partitionierung von Modellen sowie die Möglichkeit der automatischen Variablenanordnung. Die zu verifizierenden Eigenschaften können in verzweigter Temporallogik, *Computational Tree Logic (CTL)*, oder linearer Temporallogik, *Linear Temporal Logic (LTL)*, formuliert werden. nuSMV ist ein robustes, gut dokumentiertes System, welches in einer Vielzahl von universitären und industriellen Projekten eingesetzt wurde.

Verifikation durch automatische Modellprüfung ist eine im Hardwareentwurf etablierte Standardmethode. Beispielsweise wird eine Variante von SMV zur Verifikation von Schaltungen auf Registerebene seit geraumer Zeit vom Hersteller Cadence Design Systems, Inc. [4] vertrieben. Zur Modellprüfung wurden die Zustandsmaschinen des PROFIsafe-Protokolls in die Modellierungssprache SMVL übertragen. Ein Problem beim sogenannten *„Software Model Checking“* ist die relativ große Zahl von zu codierenden Zustandsvariablen. Diese führt bei der Übersetzung des Modells in die Internrepräsentation oder bei der Berechnung der erreichbaren Zustände oft zu einem nicht realisierbaren Speicherbedarf. Im vorliegenden Fall konnte das Problem durch eine geschickte Variablenanordnung, durch die Reduktion der Breite von Zählvariablen und durch spezielle Abstraktions- und Partitionierungstechniken [12] gelöst werden. Das resultierende Verifikationsmodell bestand aus vier hierarchisch strukturierten Komponenten, hatte etwa $8 * 10^{13}$ mögliche und 10^7 erreichbare Zustände. Ein typischer Übersetzungsvorgang dauerte weniger als eine Minute, die Verifikation gewisser Eigenschaften dauerte bis zu einigen Stunden.

Ziel der Verifikation war der Nachweis von globalen Kommunikationseigenschaften, etwa der Verklemmungsfreiheit, Erreichbarkeit, Lebendigkeit und von allgemeinen Sicherheitsinvarianten. Die Formulierung dieser Eigenschaften in CTL erfordert ein breites Spektrum an Fertigkeiten: einige Eigenschaften wie z. B. die Prüfung auf Verklemmungsfreiheit und Lebendigkeit sind in nuSMV bereits „fest eingebaut“. Andere Eigenschaften wie z. B. die Erreichbarkeit gewisser Zustände lassen sich mit etwas Übung leicht hinschreiben. Wieder andere erfordern eine gewisse Erfahrung im Umgang mit temporalen Logiken und bei der Transkription. Als Beispiel betrachten wir die folgende Invariante:

> *The PROFIsafe F-host always activates failsafe values when recognizing a watchdog timeout.*

Diese Eigenschaft kann wie folgt formalisiert werden:

> *Whenever the F-host receives a watchdog timeout, then from the next step onward, the signal app_FV_active provided to the application process cannot be read as false, until an operator acknowledgement is received.*

Die so formalisierte Eigenschaft kann dann 1:1 in SMVL übersetzt werden.

Im Zuge der Verifikation konnten sämtliche geforderten Eigenschaften nachgewiesen werden. In einigen Fällen waren gewisse Eigenschaften zunächst

nicht erfüllt. Eine Analyse der Ursachen ergab in jedem Fall, dass dies auf Mehrdeutigkeiten oder nicht vollständig definierte Anfangspositionen in der Spezifikation zurückzuführen war. Beispielsweise betrachtet der Modellprüfer, falls für eine bestimmte Zustandsvariable kein Startwert vorgegeben wird, *alle* möglichen Belegungen dieser Variable. Falls auch nur eine einzige davon zu einem Fehler führen könnte, wird der entsprechende Ablauf ausgegeben. Die Abbildung der Verifikationsergebnisse auf die ursprüngliche Spezifikation erfolgte durch den Verifizierer. Generell konnte also durch die Verifikation nicht nur der Nachweis der Korrektheit von PROFIsafe erbracht werden, sondern es konnten auch Präzisierungen in der Spezifikation erzielt werden.

3 Testgenerierung

Die Testgenerierung aus Zustandsmaschinen ist seit längerer Zeit Gegenstand von Forschung und Entwicklung, wobei verschiedenste Ansätze entwickelt wurden. Oftmals wird die Zustandsmaschine „flachgeklopft", d. h. Parallelität und Hierarchien werden aufgelöst, und anschließend eine Tiefensuche durchgeführt. Eine andere Alternative ist die Modellprüfung nach vorhergehender Fehlerinjektion, wobei gefundene Gegenbeispiele als Testfälle dienen. Es sind unzählige experimentelle, jedoch nur sehr wenige kommerzielle Werkzeuge verfügbar. Typischerweise erreichte Abdeckungskriterien sind *All-States*, *All-Transitions* und *Modified Condition / Decision Coverage (MCDC)*. Im Hinblick auf eine PNO-Zertifizierung[1] durch ein PI-Prüflabor[2] [2] erweisen sich *zielorientierte Ansätze* als vorteilhaft. Bei diesen wird anhand der Abdeckungskriterien zunächst eine Menge von zu erfüllenden *Testzielen* berechnet.

Zur Testgenerierung für das PROFIsafe-Sicherheitsprofil erfolgte zunächst eine 1:1-Übertragung der Zustandsdiagramme in ein professionelles UML-Modellierungswerkzeug (Telelogic Rhapsody in C++). Anschließend wurde ein zielorientierter Generator [10] zur automatischen Erzeugung der Testsuite eingesetzt. Dann wurden die generierten Testfälle in das Zielformat der bereits bestehenden Ausführungsumgebung konvertiert und durch den zielorientierten Ansatz bedingte doppelt vorhandene Testfälle mithilfe eines Post-Processors entfernt.

Danach wurde die Abdeckung gemäß den definierten Kriterien überprüft. Um nachweislich 100% Testabdeckung für die Standardkriterien *MCDC*, *All-Transitions*, *All-Events* und *All-Local-States* zu erreichen, wurden sämtliche erforderlichen Testziele detailliert betrachtet. Über 80% der Testziele wurden vom Testgenerator sofort automatisch erreicht. Für jedes Testziel, welches vom Generator nicht selbständig erfüllt werden konnte, gab es zwei Alternativen: Entweder wurden dafür weitere Testfälle mittels Simulation der Zustandsmaschine manuell erzeugt, oder es wurde die Unerfüllbarkeit (z. B. bedingt durch Nichterreichbarkeit im generierten Code) nachgewiesen und das Testziel aus dem Überdeckungskriterium entfernt. Alle unerfüllbaren Testziele waren nachweislich sicherheitstechnisch irrelevant.

Über die zum Erreichen der Sicherheitsziele notwendigen Abdeckungskriterien hinaus gibt es eine Reihe weiterer möglicher Kriterien, die in der wissenschaftlichen Literatur vorgeschlagen werden. Um mit einem betriebsbewährten kommerziellen Testgenerator Testsuiten zu erzeugen, die auch gemäß dieser weitergehenden Kriterien vollständig sind, wurden Abdeckungsbegriffe als Produkt von *Generator Coverage Function* und *Enhancement Function* definiert. Durch geeignete Anreicherung des Modells mittels Pre-Processing und anschließende Kombination und Filterung der Testfälle mittels Post-Processing gelang es, prinzipiell jedes beliebige transitionssequenzbasierte Kriterium zu erreichen. Durch zusätzliche Variablen und Bedingungen an Transitionen können zusätzliche Testziele für den Testfallgenerator definiert werden, wobei die ursprüngliche Zustandsmaschine erhalten bleibt. Durch zusätzliche Ereignisse an Transitionen werden Informationen wie z. B. Variablenwerte an den Post-Processor übergeben. Die beiden resultierenden Methoden sind *Transparent Transition Instrumentation* und *Extended Transition Instrumentation*; für eine detailliertere Darstellung siehe [5]. Bei PROFIsafe wurde das Kriterium *All-3-Transitions* [3] auf einer Abstraktion der Zustandsmachine verwendet, durch die das Sicherheitsprofil definiert ist.

Die so erzeugte Testsuite wurde zum automatisierten Test der Referenzimplementierung von PROFIsafe verwendet. Weiterhin soll sie der Konformitätsprüfung zukünftiger Implementierungen von PROFIsafe-Geräten dienen. Durch die modellbasierte Testgenerierung gelang es, eine Testsuite in einem Umfang, in einer Qualität und einer Geschwindigkeit zu erzeugen, wie es manuell nicht möglich gewesen wäre. Dadurch konnte extrem schnell auf notwendige Anpassungen in Spezifikation, Testsystem und Referenzimplementierung reagiert werden.

4 Ergebnisse

Konkrete Ergebnisse der Validierungsaktivitäten sind die Präzisierung der Spezifikation, die Optimierung der Referenzimplementierung und die Entwicklung einer automatisierten Testausführungsumgebung einschließlich zugehöriger Testsuite mit definierter Testüberdeckung. Diese deckt auch pathologische Fälle ab, die bei einer manuell erstellten Testsuite eventuell nicht berücksichtigt worden wären.

Ein wesentliches Ergebnis ist die Verbesserung der Spezifikation hinsichtlich Testbarkeit, Vollständigkeit, Funktionalität und Verständlichkeit. Nachfolgend wird jeweils ein Beispiel angegeben.

[1] PNO: PROFIBUS-Nutzerorganisation
[2] PI: PROFIBUS & PROFINET International

Testbarkeit: Im F-Host und F-Device wird jeweils eine als 24-Bit-Zahl implementierte laufende Nummer hochgezählt. Nach dem Erreichen des Maximums FFFFFFh wird diese wieder auf den Wert 1 zurückgesetzt. (Der Wert 0 ist für den Start und Fehlerzustände reserviert.) Um die korrekte Implementierung des Rücksetzens zu validieren, ist es notwendig, zu testen, ob der Zähler korrekt auf den Wert 1 gesetzt wird, nachdem der Maximalwert erreicht wurde. In der initialen Version der Spezifikation wurde die laufende Nummer mit dem Wert 0 initialisiert. In einem reinen Black-Box-Test des F-Device besteht keine andere Möglichkeit, diesen Wert zu erreichen, als die komplette Sequenz zu durchlaufen. Bei einer angenommenen Zykluszeit von 40 ms wären also zum Erreichen des Überlaufs 186 h erforderlich. In der geänderten Spezifikation wird nun der Zähler initial auf den Wert FFFFF0h gesetzt, sodass das Rücksetzen bereits nach 16 Zyklen erfolgt.

Vollständigkeit: In der initialen Spezifikation waren einige Variablen nicht explizit initialisiert; folglich wurden diese im Modell auch nicht initialisiert. Bei Verifikation und Testgenerierung wurden die betroffenen Variablen nicht auf den Wert 0 gesetzt. Dadurch kam es zu Abweichungen vom gewünschten Verhalten. Die Spezifikation wurde entsprechend vervollständigt.

Funktionalität: In der generierten Testsuite waren auch pathologische Testfälle enthalten, in welchen mehrere Fehlerzustände gleichzeitig bzw. unmittelbar hintereinander auftreten. Dabei handelt es sich um Situationen, die zwar theoretisch möglich, jedoch in der Praxis höchst unwahrscheinlich sind. Um einen fehlerfreien Betrieb auch in diesen Extremsituationen sicherzustellen, wurde das Statusbyte einer Nachricht geeignet modifiziert. Damit wird eine Synchronisation von F-Host und F-Device nach dem gleichzeitigen Auftreten mehrerer Fehler vereinfacht.

Verständlichkeit: In der initialen Spezifikation war eine Teilbedingung für einen Zustandsübergang in der Form „1→0 edge activate_FV“ angegeben. Beschrieben werden sollte damit das Auftreten einer fallenden Flanke des Kontrollbits „activate_FV“. Die Diskussion zeigte, dass auf diese Art und Weise ein Zähler spezifiziert werden sollte. Als Ergebnis der Modellierung wurde derselbe Sachverhalt in leichter verständlicher Form deklarativ spezifiziert, indem eine neue Zählvariable „ok_nr_cycles“ eingeführt und die ursprüngliche Bedingung durch „ok_nr_cycles≥2“ ersetzt wurde.

5 Weitergehende Fragestellungen

Die durchgeführten Aktivitäten führen zu einigen aus praktischer und wissenschaftlicher Sicht sehr interessanten Fragestellungen. Aus praktischer Sicht ist eine bessere Kopplung von Verifikations- und Testaktivitäten wünschenswert, was eine bessere Integration der Methoden und Werkzeuge erfordert. Aus wissenschaftlicher Sicht ist insbesondere die Betrachtung von Abdeckung als Produkt mehrerer Funktionen interessant. Hier stellen sich u. a. die Fragen nach theoretischen Grundlagen, nach dem Fehleraufdeckungspotential und nach Implikationen für Sicherheitsstandards.

Literatur

[1] http://www.profisafe.net/.

[2] http://www.profibus.com/pi/support/ptls/.

[3] Robert V. Binder: *Testing Object-Oriented Systems: Models, Patterns, and Tools*. Object Technology Series. Addison Wesley, 1999.

[4] Cadence Design Systems. http://www.cadence.com/.

[5] Mario Friske und Bernd-Holger Schlingloff: *Improving Test Coverage for UML State Machines using Transition Instrumentation*. In: *Computer Safety, Reliability, and Security – 26th International Conference, SAFECOMP 2007, Nuremberg, Germany, September 18-21, 2007. Proceedings*, LNCS 4680, S. 301–314, 2007.

[6] *IEC 61508 (all parts), Functional safety of electrical / electronic / programmable electronic safety-related systems*, 2000.

[7] *IEC 61158 / 61784-1, Industrial communications - Fieldbus profile - Part 1: Profile sets for continuous and discrete manufacturing relative to fieldbus use in industrial control systems; communication profile family 3*.

[8] *IEC 61158 / 61784-2, Industrial communications - Fieldbus profile - Part 2: Additional profiles for ISO/IEC 8802 3 based communication networks in real-time applications; communication profile family 10*.

[9] *IEC 61784-3-3, Industrial communication networks - Profiles - Part 3-3: Functional safety fieldbuses - Additional specifications for CPF 3*.

[10] I-Logix Inc.: *Automatic Test Generation User Guide. Release 2.3*, 2004.

[11] *NuSMV Homepage*. http://nusmv.irst.itc.it/.

[12] R. K. Ranjan, A. Aziz, B. Plessier, C. Pixley, und R. K. Brayton: *Efficient BDD algorithms for FSM synthesis and verification*. In IEEE/ACM Proceedings International Workshop on Logic Synthesis, Lake Tahoe (NV), 1995.