sps ipc drives

Elektrische Automatisierung
Systeme und Komponenten
Internationale Fachmesse und Kongress

Nürnberg, 25. – 27.11.2014
mesago.de/sps

25 JAHRE

Answers for automation

mesago
Messe Frankfurt Group

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.ddb.de abrufbar.

Georg Frey, Walter Schumacher, Alexander Verl (Hrsg.):

SPS IPC Drives 2014

1. Auflage, 2014

Apprimus Verlag, Aachen, 2014
Wissenschaftsverlag des Instituts für Industriekommunikation und Fachmedien an der RWTH Aachen
Steinbachstr. 25, 52074 Aachen, Germany
Internet: www.apprimus-verlag.de, E-Mail: info@apprimus-verlag.de

ISBN 978-3-86359-272-1

Analyse von im Kontaktplan implementierten Ablaufsteuerungen

B. Eng. Thomas Kärcher, Hochschule Reutlingen, Fakultät Technik
Prof. Dr.–Ing. Gernot Schullerus, Hochschule Reutlingen, Fakultät Technik

1. Einleitung

Trotz der fortschreitenden Tendenz einer modellbasierten Entwicklung von Applikationssoftware auch im Bereich der speicherprogrammierbaren Steuerungen wird immer noch ein wesentlicher Teil der Applikationen in den nach der IEC 61131–3 genormten Sprachen implementiert. Für die Implementierung von Ablaufsteuerungen ist die Ablaufsprache die erste Wahl, da sie eine direkte graphische Umsetzung des spezifizierten Ablaufs ermöglicht. Dennoch müssen aus Kostengründen oder aufgrund der Anforderungen der Kunden Ablaufsteuerungen auch in anderen Programmiersprachen z.B. im Kontaktplan oder in der Funktionsbausteinsprache implementiert werden. Da in einer in diesen beiden Sprachen programmierten POE der Ablauf nicht direkt ersichtlich ist, muss vor dem Einsatz an der Anlage eine Analyse und ein Test der entwickelten Applikation durchgeführt werden. Ein erster Ansatz dazu wurde bereits in Falcione und Krogh (1993) für im Kontaktplan entwickelte Ablaufsteuerungen, bei denen die Speicherung der Schrittmerker über Selbsthaltungen realisiert wird, vorgestellt. Die im Kontaktplan entwickelte Applikation wird bei diesem Verfahren in eine Darstellung in Ablaufsprache konvertiert, so dass eine weitergehende Analyse möglich wird. Ein komplexer Ansatz, mit dem ein in Anweisungsliste implementiertes SPS–Programm in ein Petri–Netz transformiert und damit einer formalen Analyse zugänglich gemacht wird, wird in Heiner und Menzel (1999) vorgeschlagen. Das Ziel des vorliegenden Beitrags ist es, die in Falcione und Krogh (1993) vorgeschlagene Methode dahingehend zu erweitern, dass auch ohne Selbsthaltungen programmierte Ablaufsteuerungen in ein Petri–Netz transformiert werden können. Gegenüber Heiner und Menzel (1999) wird die Aufgabenstellung so vereinfacht, dass eine handhabbare Darstellung erhalten bleibt. Das Ziel der Transformation ist nicht ein Petri–Netz–Modell mit dem eine vollständige Verifikation möglich ist. Vielmehr soll mit dem Petri–Netz eine Analyse bezüglich bestimmter unerwünschter Zustände oder möglicher Schwachstellen im Ablauf möglich sein. Der Beitrag beschreibt das vorgeschlagene Transformationsverfahren und zeigt dessen Anwendung und aus der Analyse gewonnene Erkenntnisse am Beispiel einer von Studierenden fehlerhaft implementierten Applikation zur Steuerung eines Bohrautomaten im Labor Steuerungstechnik der Fakultät Technik der Hochschule Reutlingen.

2. Grundlagen

Für die Analyse einer im Kontaktplan implementierten Ablaufsteuerung wird in diesem Beitrag das Programm in ein Petri–Netz überführt und dieses mit Hilfe des Erreichbarkeitsgraphen untersucht. Daher werden in den folgenden Abschnitten die Grundlagen von Petri–Netzen und das verwendete Analyseverfahren mit Hilfe des Erreichbarkeitsgraphen entsprechend der Darstellung aus Abel (1990) kurz erläutert.

2.1 Petri–Netze

Ein Petri–Netz wird als ein 6–Tupel

$$PN = (S, T, F, W, K, M_0)$$

mit der Stellenmenge S, der Transitionenmenge T, der Kantenmenge F, der Menge der Kantengewichte W, der Menge der Stellenkapazitäten K sowie der Anfangsmarkierung M_0 definiert. Jede Stelle $s_i \in S$ des Petri–Netzes PN kann eine der jeweiligen Stellenkapazität $k_i \in K$ entsprechende Anzahl von Marken aufnehmen. Damit können Informationen gespeichert bzw. einzelne Zustände in einer Ablaufsteuerung repräsentiert werden. Im Gegensatz zu einem Automaten wird die Information über den Gesamtzustand des durch das Petri–Netz modellierten Prozesses nicht in einem einzigen Zustand gespeichert, sondern durch die aktuelle Anzahl der Marken in den einzelnen Stellen, die sogenannte *Markierung*

$$M = \begin{bmatrix} m_1 & m_2 & \ldots & m_n \end{bmatrix},$$

dargestellt. Dabei bezeichnet m_i die Anzahl der Marken in der Stelle s_i für ein Petri–Netz mit n Stellen. Die Anfangsmarkierung M_0 ist die Markierung des Petri–Netzes zu Beginn der Betrachtung.
Eine Zustandsänderung im Petri–Netz erfolgt durch das Schalten von Transitionen $t_j \in T$, die über Kanten mit den Stellen verbunden sind. Dadurch entsteht ein Markenfluss, der die Markenbelegung der einzelnen Stellen entsprechend der folgenden Schaltregel verändert.

Schaltregel: Beim Schalten einer Transition t_j wird jeder Stelle s_i, von der eine Kante zu dieser Transition führt, eine dem jeweiligen Kantengewicht entsprechende Anzahl w_{ij} an Marken entzogen. Gleichzeitig werden zu jeder Stelle s_l, zu der eine Kante von der Transition t_j aus hinführt, so viele Marken hinzugefügt, wie das entsprechende Kantengewicht w_{jl} angibt.
Aufgrund dieser Schaltregel kann eine Transition nur schalten, wenn in den Stellen s_i, aus denen w_{ij} Marken entnommen werden, mindestens w_{ij} Marken enthalten sind und wenn die Anzahl der Marken in den Stellen s_l, zu denen w_{jl} Marken hinzugefügt werden, nach dem Schaltvorgang nicht größer als ihre jeweilige Stellenkapazität k_l wird.

2.2 Analyse von Petri–Netzen

Eines der Analysewerkzeuge für Petri–Netze ist der Erreichbarkeitsgraph. Ein Erreichbarkeitsgraph enthält alle möglichen Markierungen des Petri–Netzes ausgehend von einer Anfangsmarkierung. Somit können durch eine Analyse dieses Graphen bestimmte Eigenschaften und unerwünschte Zustände eines Petri–Netzes nachgewiesen werden. Die Grundlage für den Erreichbarkeitsgraphen ist die Erreichbarkeitsmenge.

Erreichbarkeitsmenge: Die Erreichbarkeitsmenge enthält alle Markierungen, die von einer gegebenen Anfangsmarkierung M_0 erreicht werden können. Eine Markierung M wird als erreichbar bezeichnet, wenn eine Folge von schaltfähigen Transitionen existiert, die die Anfangsmarkierung M_0 in die Markierung M überführt.
Die graphische Darstellung der Erreichbarkeitsmenge ist der Erreichbarkeitsgraph.

Erreichbarkeitsgraph: Der Erreichbarkeitsgraph stellt alle erreichbaren Markierungen der Erreichbarkeitsmenge, beginnend bei der Anfangsmarkierung, als Knoten dar. Wenn durch das Schalten einer Transition t_k die Markierung M_j von der Markierung M_i aus erreicht wird, enthält der Erreichbarkeitsgraph eine gerichtete Kante mit der Beschriftung t_k vom Knoten i zum Knoten j.

Das folgende Beispiel veranschaulicht diesen Zusammenhang. Im linken Teil der Abbildung 1 ist ein Petri–Netz mit der Stellenmenge $S = \{s_1, s_2, s_3, s_4\}$ und der Transitionenmenge $T = \{t_1, t_2, t_3, t_4\}$ dargestellt. Die Stellen werden in der graphischen Darstellung als Kreise, die Transitionen als Balken und die Marken als Punkte dargestellt. In diesem Beispiel ist die Anfangsmarkierung M_0 = [1 0 0 0]. Das Kantengewicht aller Kanten ist Eins und die Stellenkapazität aller Stellen Zwei. Der rechte Teil der Abbildung 1 zeigt den zugehörigen Erreichbarkeitsgraphen. Die Knoten enthalten die jeweiligen Markierungen der Stellen s_1 bis s_4.

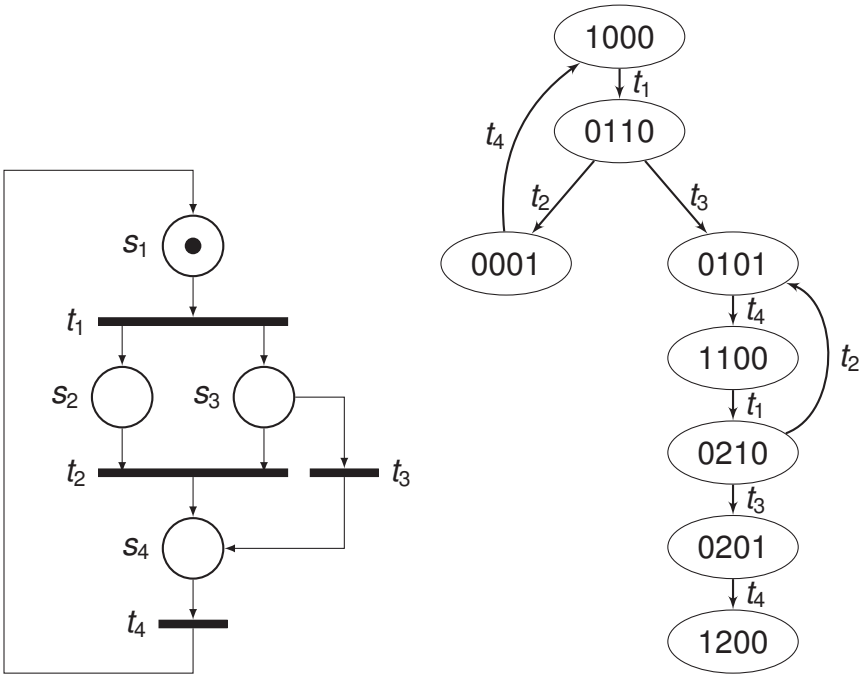

Abbildung 1: Petri–Netz mit zugehörigem Erreichbarkeitsgraphen

Aus der Analyse dieses Graphen wird deutlich, dass aus dem Knoten mit der Markierung M = [1 2 0 0] keine Kante wegführt, dass also keine Transition mehr schaltfähig ist. Dies entspricht einer *Verklemmung* und wäre demnach ein unerwünschter Zustand.

3. Analyse des Programms

Für die in diesem Beitrag vorgeschlagene Analyse einer im Kontaktplan (KOP) programmierten Ablaufsteuerung wird folgende Vorgehensweise verwendet:

1. Transformation des Programms in KOP in ein äquivalentes Petri–Netz.
2. Konstruktion des Erreichbarkeitsgraphen des Petri–Netzes.
3. Analyse des Erreichbarkeitsgraphen.

In diesem Abschnitt wird die Transformation der Ablaufsteuerung in KOP in ein Petri–Netz beschrieben. Das Anwenden des bereits bekannten Analyseverfahrens mit Hilfe des Erreichbarkeitsgraphen wird im Abschnitt 4 an einem Beispiel gezeigt.

3.1 Voraussetzung

Für die Transformation des Programms in ein Petri–Netz muss sowohl der Aufbau jedes einzelnen Netzwerks der Ablaufsteuerung in KOP als auch die Variablendeklaration mit

Initialisierung bekannt sein. Alle booleschen Variablen, denen in mindestens einem Netzwerk ein Berechnungsergebnis zugewiesen wird, werden als Schrittmerker aufgefasst. Die übrigen booleschen Variablen werden als Übergangsbedingungen betrachtet. Prinzipiell könnten diese Übergangsbedingungen durch Verwendung eines *signalinterpretierten Petri–Netzes* (vgl. Frey (2002)) auch im Modell berücksichtigt werden. Dies ist jedoch für das hier vorgestellte Verfahren nicht erforderlich. Die Vorteile, die sich aus der Verwendung eines solchen Modells ergeben, werden in zukünftigen Arbeiten untersucht.

3.2 Transformationsregeln

Gemäß (International Electrotechnical Comission, 2014, 351–26–53) ist eine Ablaufsteuerung eine „Steuerung mit schrittweisem Ablauf, bei der der Übergang von einem Schritt auf den folgenden programmgemäß entsprechend den vorgegebenen Übergangsbedingungen erfolgt".
Bei einer Programmierung in KOP werden die Schritte des Ablaufs typischerweise über Schrittmerker verwaltet. In den Netzwerken werden die Übergänge zwischen den Schritten abhängig von den aktiven Schritten und den Übergangsbedingungen programmiert. Das hier vorgestellte Verfahren beschränkt sich auf den Fall, dass für die Zuweisung der Schrittmerker ausschließlich *speichernd setzende* bzw. *speichernd rücksetzende* Spulen verwendet werden. Unter dieser Voraussetzung wurde für die Transformation ein umfangreiches Regelwerk entwickelt.
In diesem Beitrag wird nur ein für das grundsätzliche Verständnis des Verfahrens notwendiger Ausschnitt aus diesen Regeln dargestellt und anhand des Beispiels aus der Abbildung 2 erläutert. Dieses Beispiel zeigt zwei typische Implementierungsvarianten eines Übergangs von einem Vorgängerschritt in einen Nachfolgeschritt, dargestellt durch die Schrittmerker `S_VOR` und `S_NACH`, wenn die Übergangsbedingung `B_1` den Wert `TRUE` erhält. Die Auswertung der aktiven Schrittmerker und Übergangsbedingungen erfolgt im *Berechnungsteil* des Netzwerks, das damit berechnete Ergebnis wird im *Zuweisungsteil* entsprechend der dort eingesetzten Spulen den Schrittmerkern zugewiesen.

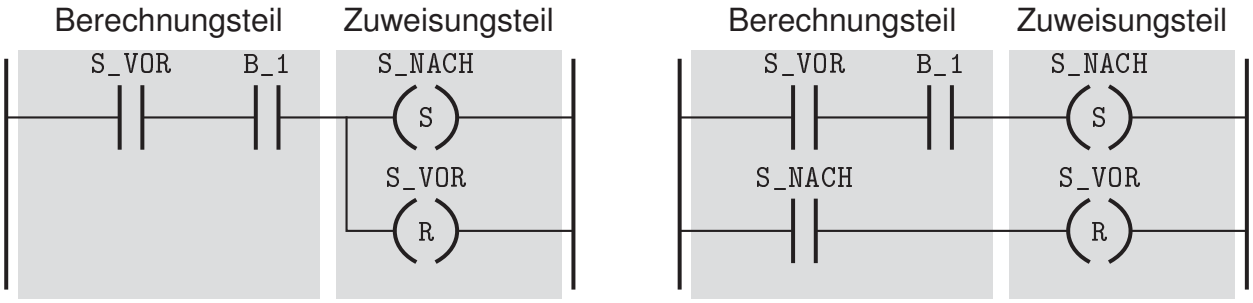

Abbildung 2: Zwei Varianten von Zustandsübergängen in KOP

Die Transformation einer Struktur entsprechend dem linken Teil der Abbildung 2 in ein Petri–Netz erfolgt nach folgenden Regeln:

1. Jeder Schrittmerker j in KOP entspricht einer Stelle s_j im Petri–Netz.

2. Für ein Netzwerk i in KOP mit einem Zustandsübergang nach Abbildung 2 links gelten folgende Regeln zur Kanten- und Transitionenbildung:

   (a) Eine Transition t_i im Petri–Netz wird angelegt.

(b) Für jeden als Schließer verdrahteten Schrittmerker j im Berechnungsteil des Netzwerks wird im Petri–Netz eine Kante von der Stelle s_j zur Transition t_i eingefügt.

(c) Für jeden Schrittmerker k, der über eine speichernd setzende Spule im Zuweisungsteil des Netzwerks verdrahtet ist, wird eine Kante von der Transition t_i zur Stelle s_k eingefügt.

(d) Existiert für den im Berechnungsteil als Schließer verdrahteten Schrittmerker j keine mit diesem Schrittmerker verknüpfte rücksetzende Spule im Zuweisungsteil, so wird im Petri–Netz eine Kante von der Transition t_i zur Stelle s_j eingefügt.

(e) Existiert für einen im Zuweisungsteil als rücksetzende Spule verdrahteten Schrittmerker j im Berechnungsteil kein Schließer, der mit dem Schrittmerker j verknüpft ist, wird eine zusätzliche Transition t_{i_j} und eine Kante von s_j nach t_{i_j} eingefügt. Die Transition t_{i_j} schaltet gleichzeitig mit t_i sofern sie schaltfähig ist.

3. Alle durch die Regel 2 entstandenen Kanten erhalten ein Kantengewicht von Eins.

4. Die Stellenkapazitäten aller Stellen im Petri–Netz sind gleich Zwei.

5. In der Anfangsmarkierung erhalten diejenigen Stellen eine Marke, bei denen der entsprechende Schrittmerker mit `TRUE` initialisiert wurde.

Die Struktur im rechten Teil der Abbildung 2 weist das gleiche Verhalten wie die Struktur im linken Teil der Abbildung 2 auf, wenn die beiden rechts dargestellten Netzwerke unmittelbar aufeinander folgen. Obwohl das Netzwerk im linken Teil der Abbildung 2 die kompaktere Form der Implementierung darstellt, ist in einigen Programmierwerkzeugen nur eine Implementierung eines Zustandsübergangs wie im rechten Teil der Abbildung 2 möglich. Die Regeln für die Transformation dieser Netzwerke sind ähnlich der oben angegebenen Regel 2.
Mit der Regel 4 wird der Tatsache Rechnung getragen, dass in einer Struktur aus der Abbildung 2 die Anweisungen auch dann ausgeführt werden, wenn der Schrittmerker `S_NACH` bereits gesetzt ist. Der entsprechende Schaltvorgang ist im Petri–Netz nur dann möglich, wenn die Stellenkapazität mindestens Zwei beträgt. Durch zusätzliche Kanten und Transitionen wird die zweite Marke aus der entsprechenden Stelle wieder entfernt, so dass der Zustand des Petri–Netzes wieder den Zustand der Schrittmerker widerspiegelt.
Durch weitere Netzelemente wird sichergestellt, dass das Schalten der den Netzwerken entsprechenden Transitionen in der Reihenfolge der Auswertung der Netzwerke erfolgt. Ebenso enthält das Verfahren Regeln zur Transformation von Öffnern sowie weiteren Strukturen, die bei der Implementierung von Ablaufsteuerungen in KOP auftreten können. Die Übergangsbedingungen werden bei der Konstruktion des Petri–Netzes nicht berücksichtigt, können aber für die Interpretation der Analyseergebnisse des Erreichbarkeitsgraphen herangezogen werden. Eine explizite Berücksichtigung der Übergangsbedingungen könnte zukünftig durch eine Erweiterung auf signalinterpretierte Petri–Netze erfolgen.

3.3 Analyse des transformierten Petri–Netzes

Das mit den Transformationsregeln erzeugte Petri–Netz beschreibt das Verhalten der Ablaufsteuerung in KOP. Aus diesem Petri–Netz wird ein Erreichbarkeitsgraph konstruiert und analysiert. Mit dieser Analyse kann die Ablaufsteuerung auf verbotene und unerwünschte Zustände untersucht werden. Dieser Beitrag beschränkt sich auf folgende Zustände:

- Zustände, die nicht mehr verlassen werden können. Ein solcher Zustand liegt dann vor, wenn im Erreichbarkeitsgraphen mindestens ein Knoten existiert, von dem keine Kanten zu anderen Knoten führen.
- Verbotene Zustände, d.h. Zustände, bei denen Schritte gleichzeitig aktiviert sind, die jedoch nicht gleichzeitig aktiviert sein dürfen.

Die verbotenen Zustände werden im Erreichbarkeitsgraphen durch die einzelnen Markierungen, die die Anzahl der Marken in den einzelnen Stellen angeben, erkannt.

4. Anwendung des Verfahrens

Die im Abschnitt 3 beschriebene Vorgehensweise wird nun am Beispiel einer Ablaufsteuerung eines Bohrautomaten aus dem Labor Steuerungstechnik an der Fakultät Technik der Hochschule Reutlingen veranschaulicht. Die wesentlichen Komponenten des in der Abbildung 3 dargestellten Bohrautomaten sind drei pneumatische Zylinder Z1, Z2 und Z3, ein Bohrer und ein Werkstückmagazin.

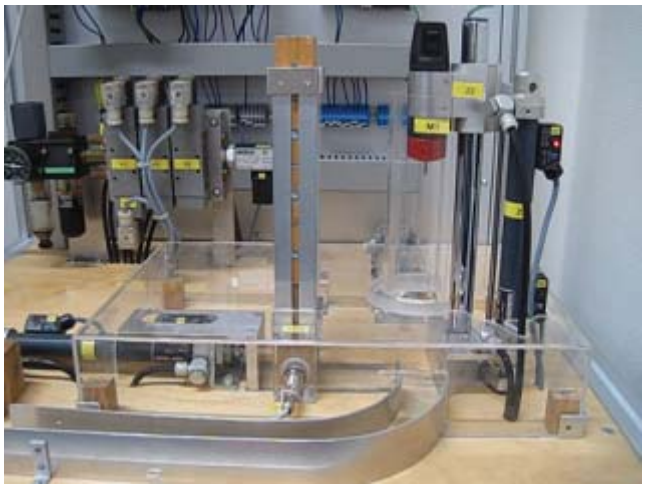

Abbildung 3: Aufbau des Bohrautomaten

Der Ablauf mit den Schritten gemäß der Tabelle 1, der an diesem Bohrautomaten implementiert werden soll, wird im Folgendem beschrieben.

Schritt	Beschreibung	Aktion
1	Z1 ausfahren	Z1 fährt nach vorne
2	Z2 ausfahren	Z2 fährt nach unten
3	Z2 einfahren	Z2 fährt nach oben (Z2 in Grundstellung)
4	Z1 einfahren	Z1 fährt nach hinten (Z1 in Grundstellung)
5	Z3 ausfahren	Z3 fährt nach vorne
6	Z3 einfahren	Z3 fährt nach hinten (Z3 in Grundstellung)

Tabelle 1: Beschreibung der einzelnen Schritte für den Ablauf des Bohrautomaten

Zu Beginn des Ablaufs befinden sich die drei Zylinder in Grundstellung. Nach dem Start des Automatikbetriebs am Bedienpult der Anlage fährt der Zylinder Z1 im Schritt 1 aus seiner Grundstellung nach vorne, schiebt dadurch ein Werkstück aus dem Magazin vor und spannt es ein. Nach dem Erreichen eines Endlagekontakts wird im Schritt 2 der Zylinder Z2 mit eingeschalteten Bohrer nach unten bewegt und das Werkstück bearbeitet.

Nach dem Betätigen eines weiteren Endlagekontakts fährt der Bohrer im Schritt 3 wieder nach oben und erreicht seine Grundstellung. Anschließend wird im Schritt 4 das Werkstück entspannt, indem der Zylinder Z1 in seine Grundstellung zurück fährt. Schließlich wird im Schritt 5 durch das Ausfahren des Zylinders Z3 aus seiner Grundstellung das Werkstück ausgeworfen. Nach Betätigung eines Endlagekontakts fährt der Zylinder Z3 im Schritt 6 in seine Grundstellung zurück. Nach dem Erreichen der Grundstellung von Z3 beginnt die Bearbeitung des nächsten Werkstücks mit Schritt 1.

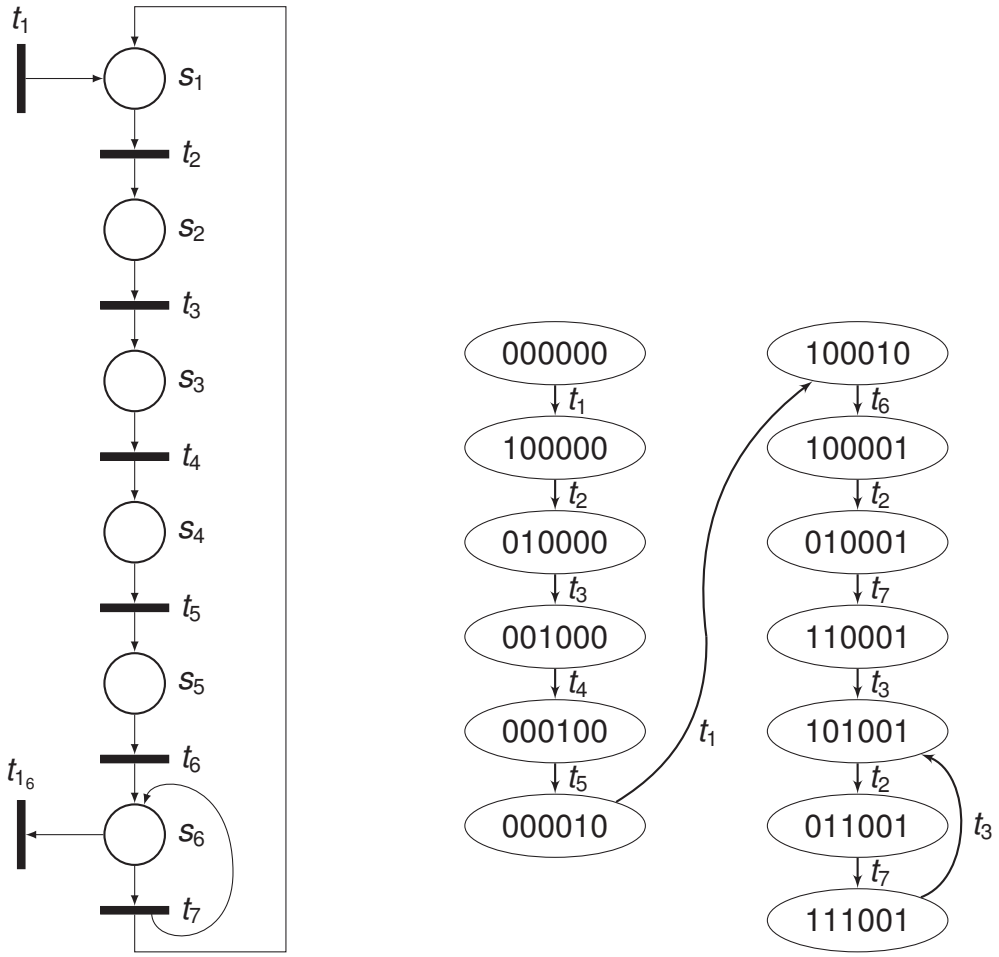

Abbildung 4: Transformiertes Petri–Netz mit Ausschnitt aus dem zugehörigem Erreichbarkeitsgraphen

Aufgrund einer fehlerhaften Implementierung dieses Ablaufs während des Praktikums Steuerungstechnik wurde ein Zustand erreicht, bei dem der Bearbeitungszyklus des ersten Werkstücks fehlerfrei ablief, wo aber bei der Bearbeitung des zweiten Werkstücks der Bohrautomat zur Laufzeit in einen Zustand gerät, bei dem sich der Zylinder Z2 mit dem Bohrer ständig auf und ab bewegt. Dieses fehlerhafte Verhalten zur Laufzeit soll durch das in diesem Beitrag vorgestellte Verfahren vor dem Test des Programms an der Anlage erkannt werden.
Mit den im Abschnitt 3 vorgestellten Regeln wird das entsprechende Petri–Netz für die fehlerhafte Ablaufsteuerung erstellt und der zugehörige Erreichbarkeitsgraph konstruiert. Für das Petri–Netz wird im linken Teil der Abbildung 4 aus Gründen der Übersichtlichkeit nur der Teil mit den Stellen, die den Schrittmerkern entsprechen, dargestellt. Ebenso wird für den Erreichbarkeitsgraphen im rechten Teil der Abbildung 4 nur ein Ausschnitt mit den

für die hier betrachtete Analyse wesentlichen Markierungen und Kanten abgebildet.
Bei der Betrachtung des durch die Transformation gewonnenen Petri–Netzes wird der Ablauf im Petri–Netz vom Schritt 1 bis Schritt 6 bzw. von der Stelle s_1 zur Stelle s_6 deutlich. Beim Übergang von der Stelle s_6 zur Stelle s_1 ist jedoch ein unerwünschtes Verhalten dieser Ablaufsteuerung zu erkennen, da beim Schalten der Transition t_7 nicht nur der Nachfolgeschritt 1 aktiviert wird, sondern auch der Vorgängerschritt 6 über die zusätzliche Kante aktiviert bleibt. Darüber hinaus kann die Marke in der Stelle s_6 über eine zusätzliche Transition t_{16} entnommen und die Stelle s_1 ohne weitere durch Schrittmerker gegebene Vorbedingungen über die Transition t_1 markiert werden. Die Netzwerke 1 und 7, aus denen dieses Verhalten resultiert, sind in den Abbildungen 5 und 6 dargestellt. Die Transformation dieser Netzwerke wird im Folgendem anhand der im Abschnitt 3 vorgestellten Regeln beschrieben.

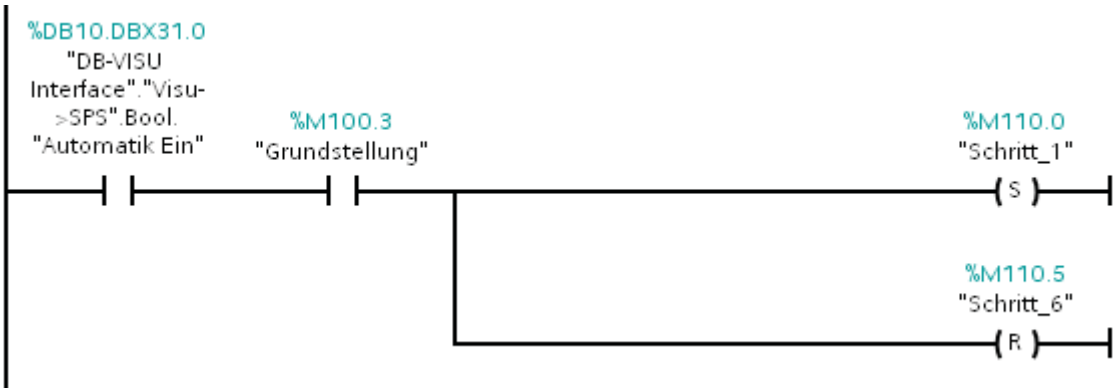

Abbildung 5: Netzwerk 1 der fehlerhaften Ablaufsteuerung

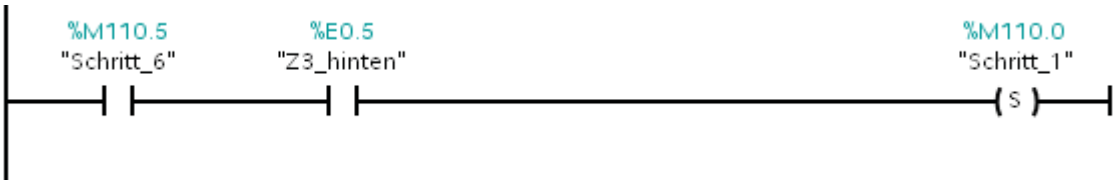

Abbildung 6: Netzwerk 7 der fehlerhaften Ablaufsteuerung

Nach der Regel 1 werden für die Schrittmerker `Schritt_1` und `Schritt_6` die Stellen s_1 und s_6 angelegt. Das Netzwerk 1 wird nach den Regeln 2a, 2c und 2e transformiert. Die beiden Übergangsbedingungen im Berechnungsteil des Netzwerks kennzeichnen den Zustand der Schaltfläche für den Start des Ablaufs in der Bedienoberfläche bzw. die Information, dass alle Zylinder ihre Grundstellung erreicht haben.
Für das Netzwerk 7 wird durch Regel 2a eine Transition t_7 angelegt. Nach Regel 2b wird eine Kante von s_6 nach t_7 und nach Regel 2c eine Kante von der Transition t_7 zur Stelle s_1 erzeugt. Da bei diesem Netzwerk das Rücksetzen des Vorgängerschritts `Schritt_6` fehlt, wird nach Regel 2d eine Kante von t_7 nach s_6 erzeugt.
Die nun folgende Analyse des Erreichbarkeitsgraphen zeigt, dass mehrere Knoten existieren, bei denen zwei oder mehrere aufeinanderfolgende Stellen markiert sind. Dazu zählt der Knoten mit der Markierung M = [1 1 0 0 0 1]. Bei dieser Markierung sind die Stellen s_1, s_2 und s_6 gleichzeitig markiert. Damit sind im Ablauf drei aufeinanderfolgende Schritte gleichzeitig aktiviert. Dieses Verhalten entspricht einer unsicheren Ablaufsteuerung und stellt somit einen unerwünschten Zustand dar. Dies gilt auch für die Markierung

M = [1 0 0 0 1 0]. Erreicht die Anlage den dieser Markierung entsprechenden Zustand, wären die Schritte 1 (Einspannen) und 5 (Auswerfen) gleichzeitig aktiviert.
Die ab der Bearbeitung des zweiten Werkstücks beobachtete Endlosschleife bei der Ansteuerung des Bohrers wird ebenfalls aus dem dargestellten Ausschnitt des Erreichbarkeitsgraphen deutlich. Ausgehend von der Markierung M = [1 0 1 0 0 1] erhält man über die Schaltfolge der Transitionen t_2, t_7 und t_3 eine Schleife, bei der der Schritt 2 abwechselnd aktiv und nicht aktiv ist. Aufgrund des bei dieser Anlage verwendeten Ventils für den Zylinder Z2 fährt dieser im Schritt 2 nach unten. Ist Schritt 2 nicht aktiv, fährt der Zylinder nach oben, so dass aus dieser Schleife das beobachtete Verhalten deutlich wird.
Dieses Anwendungsbeispiel zeigt somit, dass mit der vorgestellten Methode tatsächlich auftretende Probleme bereits bei der Analyse erkannt werden können.

5. Zusammenfassung

In diesem Beitrag wurde gezeigt, wie mit Hilfe von Verfahren zur Analyse von Petri–Netzen ein in der Programmiersprache *Kontaktplan* erstelltes SPS–Programm analysiert werden kann. Das Ziel des Verfahrens ist dabei nicht eine Verifikation im eigentlichen Sinne sondern das Aufdecken von verbotenen oder unerwünschten Zuständen. Im Beitrag wurden Regeln zur Transformation des im Kontaktplan erstellten Ablaufs in ein Petri–Netz angegeben und anhand der Analyse eines fehlerhaft implementierten Ablaufs die Leistungsfähigkeit des Ansatzes vorgestellt. Das Beispiel zeigt, dass Programmfehler bereits vor einem Test an der realen Anlage erkannt werden können.
Bei der weiteren Entwicklung des Verfahrens liegt ein Schwerpunkt auf der Verallgemeinerung auf im Kontaktplan entwickelte Programmorganisationseinheiten, die nicht nur reine Abläufe implementieren. Ein weiterer wichtiger Entwicklungsschritt ist die graphische Unterstützung der Fehlersuche im Erreichbarkeitsgraphen, so dass insgesamt ein leistungsfähiges Werkzeug zur Unterstützung der Implementierung von Ablaufsteuerungen im Kontaktplan zur Verfügung steht.

Literatur

Dirk Abel. *Petri–Netze für Ingenieure: Modellbildung und Analyse diskret gesteuerter Systeme*. Springer Verlag, Berlin, 1990.

Albert Falcione und Bruce H. Krogh. Design Recovery for Relay Ladder Logic. *IEEE Control Systems Magazine*, 13(2):90–98, 1993.

Georg Frey. *Design and Formal Analysis of Petri Net Based Logic Control Algorithms*. Shaker Verlag, Aachen, 2002.

Monika Heiner und Thomas Menzel. Modellierung und Analyse von SPS–Anwenderprogrammen mit Petri–Netzen. In *Tagungsband, 6. Fachtagung Entwurf komplexer Automatisierungssysteme*, Seiten 247–265, 1999.

International Electrotechnical Comission. Deutsche Online–Ausgabe des International Electrotechnical Vocabulary, 2014. URL http://www.dke.de/de/Online-Service/DKE-IEV/Seiten/IEV-Woerterbuch.aspx.