Praktikum Automatisierungstechnik

Wintersemester 2016/2017

Protokoll zum

Versuch 4: Fertigungsanlage

der Gruppe 5

Xiaoyu Xie
Shaochen Qian
Jun Lou

Tag der Versuchsdurchführung: 08 November 2016
15 November 2016

Hiermit versichern wir, dass wir dieses Protokoll selbstständig angefertigt haben.
Karlsruhe,

Inhaltsverzeichnis

1. Einleitung und Versuchsaufbau

Der Praktikumsversuch basiert auf dem Labormodell MoFa France eines flexiblen Fertigungssystems, mit dem die Transport- und Fertigungsvorgänge von industriellen Anlagen untersucht werden können.

Der Versuchsaufbau besteht aus einem Lager mit Werkstücken, einem Portalkran und drei Bearbeitungszentren, die jeweils die Werkstücke transportieren und den Bearbeitungsprozess simulieren können. Alle 12 Werkstücke werden im Lager gesetzt. Der Portalkran befindet sich zu Beginn nach einer Initialisierung an einem Nullpunkt. Die Motoren bewegen den Portalkran entweder vor oder zurück (x-Richtung) sowie die Laufkatze entweder nach links oder nach rechts (y-Richtung), dessen Position durch Programm festgelegt werden kann.

In diesem Praktikumsversuch konzentrieren wir uns auf das Bearbeitungszentrum A und B. Das Bearbeitungszentrum A besteht aus den Werkzeugmaschinen M3 und M4 sowie vier Transportbändern (TB1 bis TB4). Der Portalkran transportiert ein Werkstück auf dem Transportband TB1, anschließend wird das Werkstück vom Transportband TB1 auf das Transportband TB2 übergeben. Die Werkzeugschlitten der Maschine M3 kann sich nach vorn oder hinten bewegen, außerdem kann sich der Werkzeugträger der M3 nach oben oder unten bewegen. Mit Hilfe vier an der Maschine M3 angebrachte Sensoren kann es bestätigt werden, ob sich das Werkzeug über dem Werkstück oder in ihrer Ruheposition befindet, die entsprechende binäre Signale an die SPS weiterleiten. Sowie das Werkstück vor der Maschine M3 still steht, fährt die Maschine gleichzeitig nach vorne und nach unten. Nachdem die Maschine in der Arbeitsposition angekommen ist, fängt der Werkzeugspindel an zu drehen und beginnt die Bearbeitung des Werkstücks. Der Bearbeitungsprozess dauert 5 Sekunden. Nach dem Ende der Bearbeitung fährt die Maschine M3 wieder zurück, sobald die Maschine die Ruheposition erreicht, wird das Werkstück vorwärts transportiert und weiterbearbeitet von der Maschine M4. Der Fertigungsablauf der M4 ist gleich wie der von M3. Das Bearbeitungszentrum B besteht aus einem Drehtisch DT und einer Maschine M1. Erstere wird gedreht anschließend der Annahme des vom Portalkran transportierten Werkstück. Nach einer Drehung von 180° erreicht das Werkstück die Bearbeitungsposition, dabei bewegt sich der Maschine M1 von oben nach unten zur Bearbeitung. Immer wenn der Drehtisch in die Abhol- bzw. Bearbeitungsposition ankommt ist, lässt sich ein binäres Signal erzeugen und weiterleiten. Nachdem die Bearbeitung beendet, fährt die Maschine M1 wieder nach oben zurück und bringt der Drehtisch das Werkstück in die Anfangsposition zurück.

Das Bearbeitungszentrum 3 besteht auch aus einer Zuführeinrichtung (ein Wechseltisch) und einer Werkzeugmaschine (Maschine M2), in dem ein ähnlicher Ablauf besteht.

Das Labormodell besitzt 36 digitale Eingänge und 37 digitale Ausgänge, die mit den E/A-Baugruppen der SPS verbunden. Ein Steuerungsprogramm wird im Praktikum auf einem PC entwickelt und durch WinCC visualisiert. Es gibt 3 Betriebsarten der Anlage, nämlich Tipp-Betrieb, Halbautomatik und Automatik. In der Betriebsart Automatik bestehen 3 Programme. Jedes entspricht dem Vorgang eines Bearbeitungszentrums.

2. Modellierung und Analyse der ereignisdiskreten Systeme

Definition vom Petri-Netz: Ein Petri-Netz ist ein 6-Tupel $N = (S, T, F, C, W, M_0)$mit:

1. der nichtleeren endlichen Stellenmenge $S = \{s_1 \cdots s_n\}$,

2. der nichtleeren endlichen Transitionsmenge $T = \{t_1 \cdots t_n\}$,

3. der nichtleeren endlichen Kantenmengen$F$,welche die Verbindung zwischen Stellen und Transitionen beschreiben,

4. der Menge *C* aller Stellen-Kapazitäten,

5. der Menge *W* aller Kantengewichte,

6. der Anfangsmarkierung $M_0 = (M_0(s_1) \ldots M_0(s_n))$.

Beispiel:

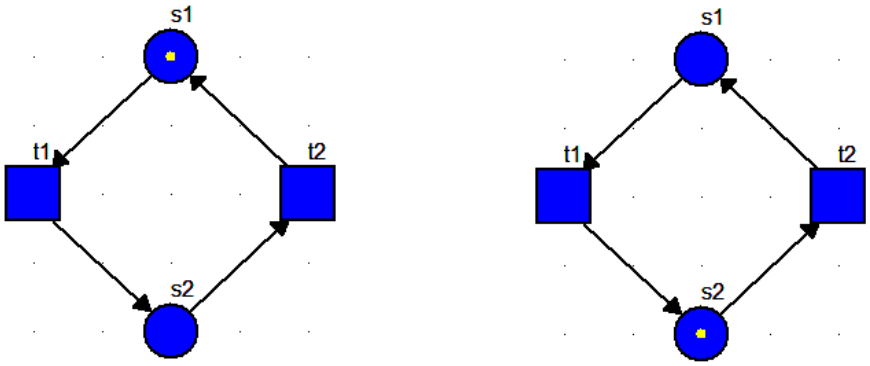

Abbildung 2.1 Zustandsänderungen im Petri-Netz

Die Stellen werden als Kreise gezeichnet, sie repräsentieren die möglichen Maschinenzustände.

Transitionen werden als Rechtecke oder Balken dargestellt und stehen für Zustandsübergänge. Durch das Schalten der Transition kann man die Markierung von einer Stelle zu einer anderen Stelle transportieren.

Schaltvoraussetzung der Transition: Eine Transition ist aktiviert (schaltfähig), wenn gilt:

1.$M(s_i) \geq w(s_i, t_j)$

2.$M(s_i) \leq c(s_i) - w(t_j, s_i)$

indem bedeutet $w(s_i, t_j)$ Markenfluss von Stelle zu Transition, $w(t_j, s_i)$ Markenfluss von Transition zu Stelle.

Algebraische Netzbeschreibung: Ein Petri-Netz heißt reines Petri-Netz, wenn es kein Kreis zwischen einer Stelle und einer Transition gibt.

Zu jeder Transition eines Petri-Netz ist ein Transitionenvektor$t_j = [t_{1j} \cdots t_{nj}]^T$definiert.Die

aus den Transitionenvektoren $\underline{t_j}$ gebildet Matrix $\underline{N} = [\underline{t_1}, \cdots \underline{t_m}]$ heißt Netz-Matrix.Die beschreibt die Verbindung zwischen jeder Transition und alle Stellen.

Die j-te Komponente $v_j(k)$ des Schaltvektors$\underline{v}(k)$gibt an, ob die j-te Transition beim k-ten Schritt schaltet.

Mit Hilfe der Netz-Matrix und des Schaltvektors kann man die Dynamik der Netzmarkierung beschreiben.

$$\underline{m}(k+1) = \underline{m}(k) + \underline{N} \cdot \underline{v}(k+1)$$

Eigenschaften von Petri-Netzen

Konflikt: Zwei aktivierte Transitionen sind miteinander im Konflikt, wenn nach dem Schalten einer Transition die andere Transition nicht mehr aktiviert ist, was für reine Petri-Netze automatisch auch umgekehrt ist.

Kontakt: Das Schalten einer Transition durch die Markierung des Nachbereichs verhindert wird.

Beispiel:

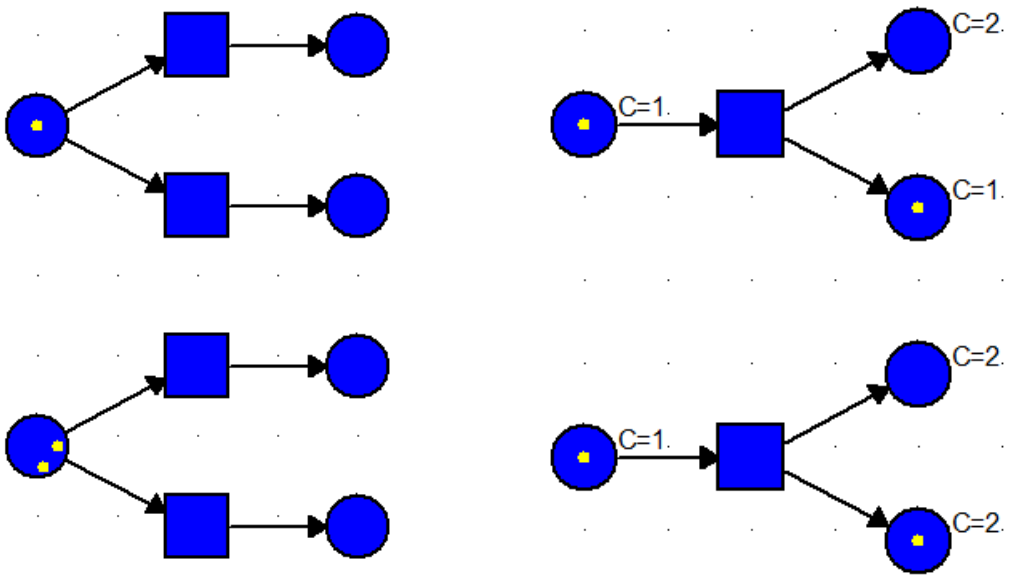

Abbildung 2.2 Konflikt und Kontakt

Erreichbarkeit

1. Eine Markierung M eines Petri-Netzes heißt erreichbar falls eine anwendbare Schaltsequenz existiert, die M_0 in M führt.

2. Die Erreichbarkeitsmenge $R_N(M_0)$ ist die Menge aller von M_0 aus erreichbaren Markierungen, einschließlich M_0 selbst.

Tote Transition: Eine Transition heißt tot unter M_0, wenn die unter M_0 und alle Folgemarkierung von M_0 nicht aktiviert ist.

Tote Markierung: Eine Markierung M heißt tot, wenn unter$M$keine Transition aktiviert ist.

Totale Verklemmung: Es wurde eine tote Markierung erreicht, unter der keine Transition mehr schalten kann.

Partielle Verklemmung: Es wurde eine Markierung erreicht, unter der manche Transitionen tot sind, aber andere noch schalten kann.

Graphische Analyse von Petri-Netzen

Erreichbarkeit: Erreichbarkeitsgraphen E_N einen Knoten *M* besitzt = Die Markierung *M* ist in *N* erreichbar.

Tote Transition: Die Transition tritt an keiner Kante von E_N auf = die Transition ist eine tote Transition.

Totale Verklemmung/ Tote Markierung:E_N besitzt einen Knoten *M* ohne auflaufende Kante = In *N* ist eine totale Verklemmung möglich.

Konflikt: Eine Konfliktsituation kann nur an Knoten *M* auftreten, die mehr als eine auslaufende Kante haben. Dabei handelt es sich nur dann um einen Konflikt, wenn die auslaufenden Kanten der Zielknoten nicht mehr mit den Transitionen beschriftet sind, die schon an den von *M* abgehenden alternativen Kanten standen, ausgenommen der geschalteten Transition.

Ein gerichteter Graph *G* heißt *stark zusammenhängend*, wenn je zwei Knoten von *G* durch einen gerichteten Pfad in beiden Richtungen verbunden sind.

Algebraische Analyse

S-Invarianten: Ein Vektor $\underline{i_s}$ heißt S-Invariant, falls gilt: $\underline{N^T} \cdot \underline{i_s} = \underline{0}$

Die Bedeutung lässt sich anhand der Netzgleichung ableiten:

$$\underline{m(k+1)} = \underline{m(k)} + \underline{N} * \underline{v(k+1)}$$

$$(\underline{N} \cdot \underline{v(k+1)})^T = (\underline{m(k+1)} - \underline{m(k)})^T$$

$$\underline{v^T(k+1)} \cdot \underline{N^T} \cdot \underline{i_s} = \underline{0} = (\underline{m(k+1)} - \underline{m(k)})^T \cdot \underline{i_s}$$

somit gilt:

$$\underline{m(k)}^T \cdot \underline{i_s} = const$$

Diese Eigenschaft kann sowohl für den Entwurf einer Steuerung genutzt werden, bei denen bestimmte Markierungen verhindert werden sollen als auch zur Verifikation.

T-Invarianten: Ein Vektor $\underline{i_T}$ heißt T-Invarianten, falls gilt: $\underline{i_T^T} \cdot \underline{N} = \underline{0}$.

Jedes reversible Petri-Netz besitzt eine nicht-negative T-Invariante.

Steuerungsentwurf mit Hilfe von S-Invarianten

Wenn es eine Verbotsspezifikation gibt, kann man mit Hilfe von S-Invarianten Steuerungen entwerfen.

Wir haben jetzt eine Forderung,

$$b \geq \underline{l}^T * \underline{m}(k)$$

Die Elemente des Vektors $\underline{l}^T$ werden zu 1 gewählt, falls die jeweiligen Stellen in diese Forderung auftreten, andernfalls werden sie zu 0 gesetzt.

Es soll nun eine Steuerung berechnet werden, welche diese Forderung. gewährleistet. Hierzu wird zunächst die Ungleichung mit Hilfe einer Schlupfvariable in eine Gleichung umgewandelt:

$$b = \underline{l}^T * \underline{m}(k) + m_c(k), m_c(k) \geq 0$$

Die Schlupfvariable$m_c(k)$kann als Markierung einer zusätzlich dem Netz hinzufügten Stelle $s_c$interpretiert werden. Durch Umformen der obigen Gleichung

$$b = \underline{l}^T * \underline{m}(k) + m_c(k) = [\underline{l}^T 1] \cdot \begin{bmatrix} \underline{m}(k) \\ m_c(k) \end{bmatrix} = const$$

Der Vergleich mit$\underline{m}(k)^T \cdot \underline{i}_s = const$lässt einen Zusammenhang erkennen. Bei dem Vektor $[\underline{l}^T 1]$ handelt sich also offensichtlich um eine S-Invariante des um die Stelle s_c erweiterten Netzes. Daher gilt mit der Netzmatrix $\underline{N}$ des ursprünglichen Netzes und der zu$s_c$gehörenden Zeile $\underline{n}_c^T$ der erweiterten Netzmatrix.

Erhält man

$$\underline{n}_c^T = -\underline{l}^T * \underline{N}$$

die Kanten und ihre Gewichte, mit denen die Stelle$s_c$mit den Transitionen des ursprünglichen Netzes verbunden sein müssen, um das gewünschte Verhalten zu sichern. Zur Berechnung der Anfangsmarkierung der Stelle s_c wird die Gleichung

$$b = \underline{l}^T * \underline{m}(k) + m_c(k)$$

einfach für *k* = 0 ausgewertet. Die so berechnete Steuerung verhindert, dass die Summe der Marken in den durch$\underline{l}^T$gekennzeichneten Stellen des Netzes größer wird als *b*.

3. Implementierung in Speicherprogrammierbaren Steuerungen

Zur Automatisierung ereignisdiskreter Prozesse lassen sich Speicherprogrammierbare Steuerungen (SPS) einsetzten. Das Programm der SPS legt fest, wie die Ausgänge in Abhängigkeit von den Eingängen geschaltet werden sollen.

Die Im Praktikum eingesetzte SPS SIMATIC S7-400, welche modular aufgebaut ist, ist mit einer Stromversorgungsbaugruppe (PS 407 10A), einer CPU (416-2DP) und mit digitalen Ein- und Ausgabebaugruppen (DI32×DC24V und DO32×DC24V/0.5V) ausgerüstet. Die einzelnen Teile des Programms werden modular in verschiedenen Bausteinen abgelegt. Sie beinhalten die Organisationsbausteine (OB), welche zur Steuerung des Programmablaufs dienen, und die Anwendungsbausteine, dazu gehören die Funktion (FC), die Funktionsbausteine (FB) und die Datenbausteine (DB).

Die SPS arbeitet zyklisch. Sie liest die Werte aller Eingänge am Anfang eines Zyklus ein. Dann werden die gespeicherten Programme (die Netzwerke) einmalig ausgeführt und am Ende werden die Ausgänge gesetzt oder rückgesetzt. Dann startet der nächste Zyklus – Die Zeit, die bei einem Durchlauf verstreicht, heißt Zykluszeit. Die Betriebszustände beinhalten RUN, RUN-P und STOP. Mit „Ex.y“, „Ax.y“ bzw. „Mx.y“ lassen sich die Ein-, Ausgänge und die Merker abfragen bzw. setzen.

Mit der Software STEP 7 kann man die SPS konfigurieren und programmieren. Darin sind mehrere Programmiersprachen verfügbar. Sie sind:

1. Anweisungsliste (AWL): textuelle Programmiersprache aufgrund maschinennaher Syntax,

2. Kontaktplan (KOP): Syntax erinnert an einen Stromlaufplan,

3. Funktionsplan (FUP): Verknüpfung von Logikbausteinen.

4. Ablaufsprache (AS): graphische Programmiersprache auf der Basis von Ablaufketten, trägt in STEP 7 den Namen S7-GRAPH,

5. Strukturierter Text (ST): PASCAL-ähnliches Sprachkonzept, heißt in STEP 7 Struktured Control Language (SCL).

Hat man eine Steuerung für einen technischen Prozess entworfen und verifiziert, so kann diese in einer geeigneten Sprache auf einer SPS implementiert werden. Die folgende Voraussetzungen müssen für die Konvertierung einer auf Basis von Petri-Netzen entworfenen Steuerung erfüllt werden: 1. Die Petri-Netze haben die Stellekapazität eins. 2. Konfliktsituationen zwischen Transitionen müssen durch boolesche Schaltausdrücke an den Transitionen aufgelöst werden. 3. Alle Post- und Prekantengewichte sind eins.

Die Schritte von der Konvertierung: 1. Jeder Stelle im Petri-Netz wird eine Variable zugeordnet. 2. Ist eine Stelle belegt, so ist die ihr zugeordnete Variable Eins, sonst Null. 3. Für jede Transition wird ein boolescher Ausdruck gebildet. 4. Bei Ausgangsstellen muss zusätzlich noch eine Verknüpfung zu den Ausgängen hergestellt werden.

4. Aufgaben

Aufgabe 1

Die folgenden AWL-Programme sind fast identisch, lediglich die Reihenfolge der Netzwerke ist vertauscht. Die Zykluszeit der SPS, welche diese Programme auswertet, sei 20 ms. Geben Sie an, zu welchem Zeitpunkt der Ausgang A0.0 gesetzt wird, wenn sich der

Wert des Eingang E0.0 zum Zeitpunkt t = 0 s von Null auf Eins ändert, unter der Annahme, dass alle Merker zu Beginn zurückgesetzt sind.

Antwort:

AWL-Programm 1: zum Zeitpunkt t = 20ms wird der Ausgang A0.0 gesetzt.
Der Merker M0.0 und der Merker M0.1 sowie der Ausgang A0.0 werden in einem Zyklus hintereinander von Null auf Eins geändert.

AWL-Programm 2: zum Zeitpunkt t = 40ms wird der Ausgang A0.0 gesetzt.
Wenn der Merker M0.1 gleich Eins ist, dann soll der Ausgang A0.0 gesetzt werden. Wegen des Netzwerks, in dem der Ausgang A0.0 gesetzt wird, liegt vor dem Netzwerk, in dem der Merker M0.1 gesetzt wird, wird der Ausgang A0.0 in dem zweiten Zyklus von Null auf Eins geändert.

AWL-Programm 3: zum Zeitpunkt t = 40ms wird der Ausgang A0.0 gesetzt.
In dem ersten Zyklus wird der Merker M0.0 gesetzt. Wegen der Reihenfolge der Netzwerke werden der Merker 0.1 und der Ausgang A0.0 in dem zweiten Zyklus auf Eins geändert.

Aufgabe 2

In Abbildung 6.1 ist ein Petri-Netz mit Alternativen dargestellt, dass in ein AWL-Programm übersetzt werden soll. Beantworten Sie zunächst folgende Fragen:

- Aus wie vielen Netzwerken besteht das AWL-Programm?
- Wie kann die Anfangsmarkierung des Petri-Netzes im SPS-Programm berücksichtigt werden?
- Wie werden im Petri-Netz vorhandene Konflikt bei der Ausführung des Programms auf einer SPS gelöst?
- Welche Reihenfolge müssen die Netzwerke haben, damit zur Programmlaufzeit eine Markierung jeweils für eine Zykluszeit der SPS bestehen bleibt?

Übersetzen Sie nun dieses Petri-Netz in ein AWL-Programm, wobei jede Stelle durch einen Merker beschrieben werden soll.

Antwort:

- Aus 5 Netzwerken besteht das AWL-Programm. Jede Transition entspricht ein Netzwerk.
- Die Anfangsmarkierung des Petri-Netzes kann im SPS-Programm als Anfangswerte entsprechender Variablen (Merker, Ein/Ausgang) gespeichert.
- Durch die verschiedene Reihenfolge der Netzwerke im SPS-Programm können die Prioritäten der Transitionen setzen, damit der im Petri-Netz vorhandene Konflikt gelöst werden kann.
- Jede Transition entspricht einem Netzwerk im SPS-Programm. Um zur Programmlaufzeit eine Markierung jeweils für eine Zykluszeit der SPS bestehen zu

bleiben, muss das Netzwerk der letzten Transition auf dem ersten Platz eingestellt werden, danach haben die Netzwerke umgekehrte Reihenfolge, als die Reihenfolge die Transitionen haben.

AWL-Programm:

Netzwerk1: Transition t5 wird aktiviert

```
U    S4
UN   S5
S    S5
R    S4
```

Netzwerk2: Transition t4 wird aktiviert

```
U    S3
UN   S4
S    S4
R    S3
```

Netzwerk3: Transition t3 wird aktiviert

```
U    S1
UN   S3
S    S3
R    S1
```

Netzwerk4: Transition t2 wird aktiviert

```
U    S2
UN   S4
S    S4
R    S1
```

Netzwerk5: Transition t1 wird aktiviert

```
U    S1
UN   S2
S    S2
R    S1
```

Aufgabe 4

Nun soll eine Ablaufsteuerung für die Maschine M3 entworfen werden, welche dann am Versuchstag an der SPS mit S7-Graph programmiert werden soll.

Führen Sie zunächst eine zustandsorientierte Modellierung für das Verhalten der ungesteuerten Maschine M3 gemäß der Beschreibung im Abschnitt 1.1.2 als ein NCE-System durch.

Überlegen Sie sich, auf welche Ein- und Ausgänge der SPS eine Steuerung zugreifen kann bzw. muss. Entwerfen Sie dann eine Ablaufsteuerung und verknüpfen Sie diese mit dem Modell der Maschine unter Verwendung von Condition- und Eventkanten. Überprüfen Sie die Funktionsfähigkeit der Steuerung durch Simulation mit DESSKA.

Antwort:

Für die Maschine M3 beinhalten die Eingänge die Sensorsignale (E 1.2 bis E 1.5), welche die Endposition zeigen. Die Ausgänge beinhalten die Antriebssignale (A 1.4 bis A 1.7 bzw. A 2.0), welche die Aktionen der Aktoren antrieben.

Die vollständige Anbindung der Steuerung an das Streckenmodell ist im Bild 4.1 dargestellt. Hier sind alle Teil-Petri-Netze durch Condition- und Eventkanten verbunden. Z. B. die Sensoren zur Lokalisierung liefern stückweise konstante Werte an die Steuerung, sind die Condition-Kanten benutzt. Für Schaltaktionen, als Ausgangssignaländerung der SPS kommen Eventkanten beim Weiterschalten eines Ablaufschrittes zur Einsatz.

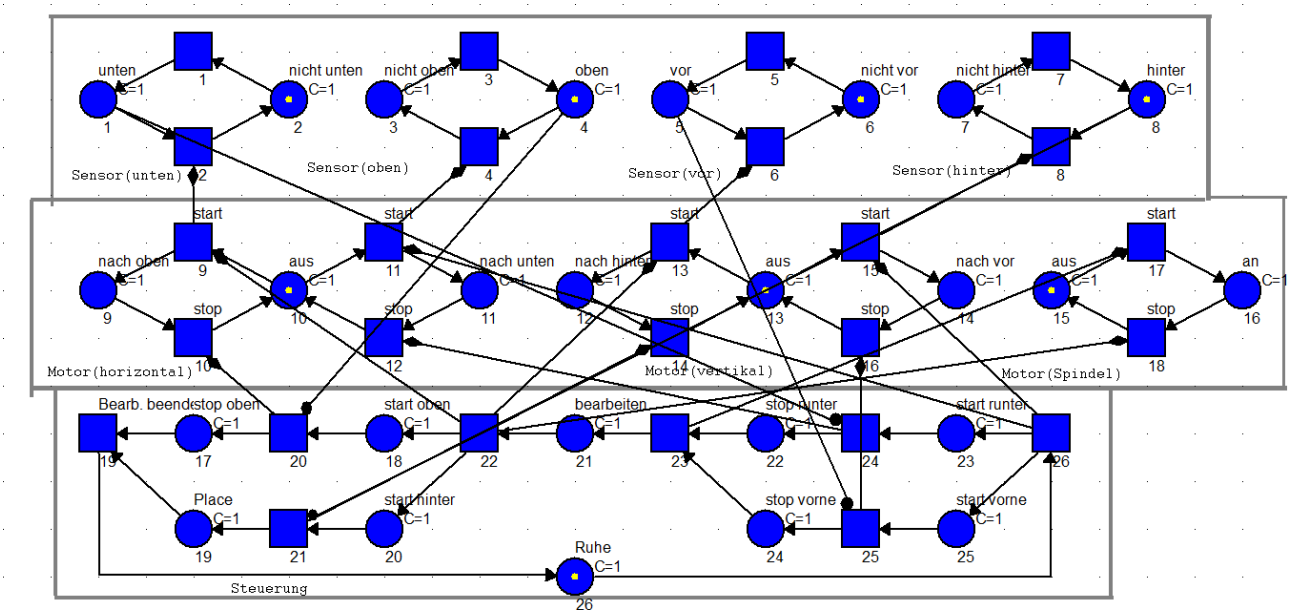

Abbildung 4.1 NCE-System für das gesteuerte Verhalten der Maschine M3

Aufgabe 5

Gegeben sei ein Prozess, welches verhalten durch Petri-Netz (Abbildung 4.2) dargestellt ist. Dabei hat das Holen eines Werkstücks aus dem Lager Priorität vor dem Abholen eines bearbeiteten Werkstücks von TB4. Zum erstens analysieren wir das Petri-Netz bezüglich der toten Markierung durch DESSKA. Danach entwerfen wir eine Steuerung mit Hilfe von S-Invariant.

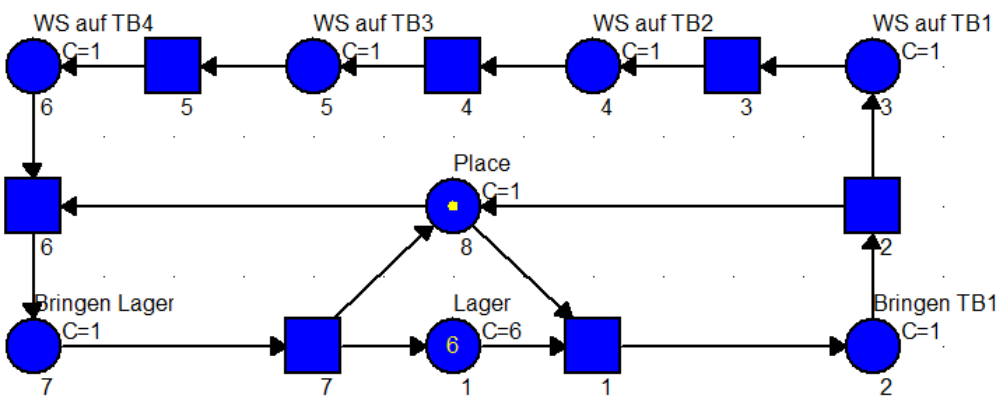

Abbildung 4.2 Petri-Netz für den Prozess

Beladen wir hierzu ungeachtet der Kapazität der Transportbänder immer weiter das TB1. Bekommen wir eine wie Abbildung 4.3 tote Markierung.

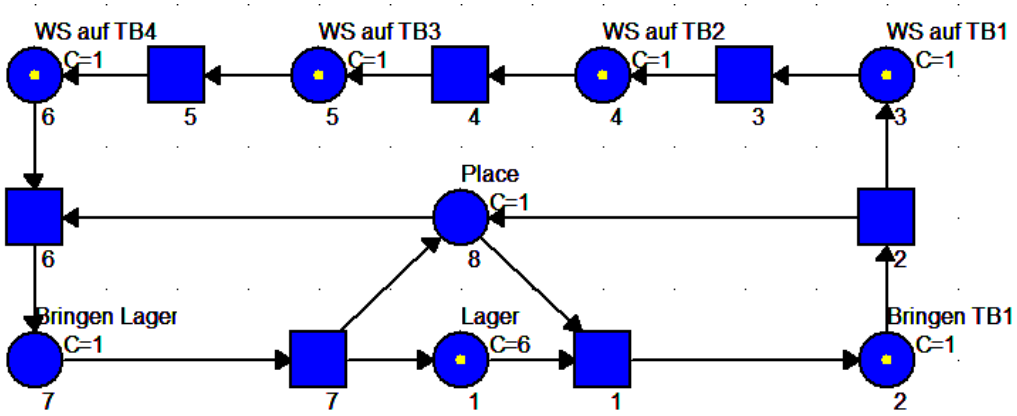

Abbildung 4.3 Tote Markierung

Um diese tote Markierung zu vermeiden, entwerfen wir eine Steuerung mit Hilfe von S-Invarianten. Zum erstens stellen wir eine algebraische Modell von dem Petri-Netz.

Modell ohne Steuerung

$$\underline{m}(k+1) = \underline{m}(k) + \underline{N} * \underline{v}(k+1)$$

mit

$$\underline{N} = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & -1 \\ -1 & 1 & 0 & 0 & 0 & -1 & 1 \end{bmatrix}$$

Die Summe von TB1, TB2, TB3, TB4 und Bringen TB1 muss kleiner als 5 sein.

Forderungsgleichung

$$\underline{b} \geq \underline{l}^T * \underline{m}(k)$$

Das führt sofort zu:

$$5 \geq [0 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 0 \quad 0] * \underline{m}(k)$$

Durch Auflösen der Gleichung:

$$\underline{n_c^T} = -\underline{l}^T * \underline{N}$$

Erhält man die Kanten und ihre Gewichte, mit denen die zusätzliche Stelles_c mit den Transitionen des ursprünglichen Netzes verbunden sein.

$$\underline{n_c^T} = [-1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0]$$

Zur Berechnung der Anfangsmarkierung der Stelle wird durch die Gleichung

$$\underline{m_c}(0) = b - \underline{l}^T * \underline{m}(0)$$

ausgewertet.

$$\underline{m_c}(0) = 4$$

Jetzt fügen wir die Zusätzliche Stelle ein, wie Abbildung 4.4.

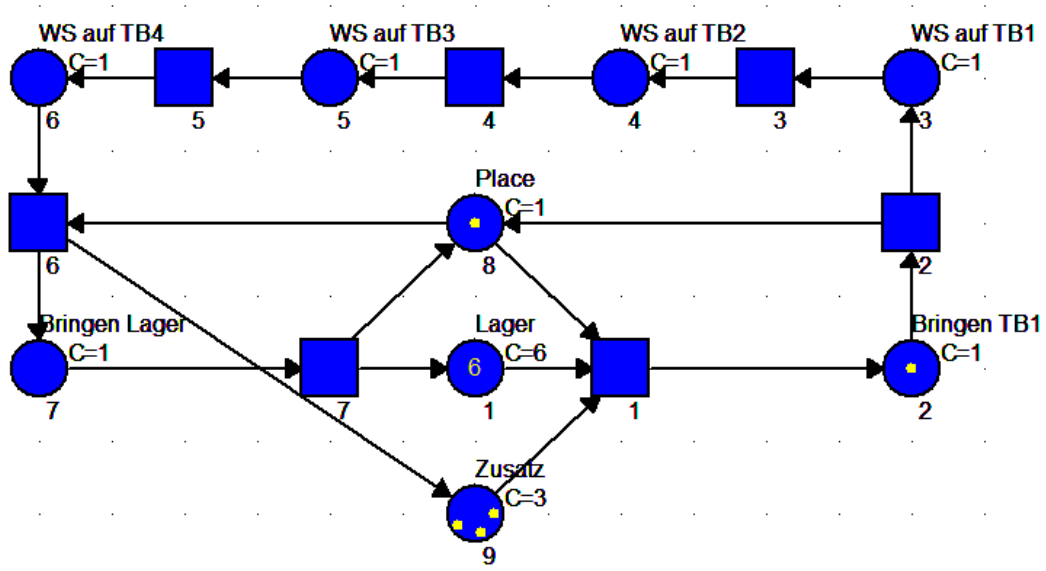

Abbildung 4.4 gesteuertes System mit zusätzliche Stelle

Durch diese neue Stelle ist die oben genannte tote Markierung verboten. Wenn die Summe von TB1-4 4 ist, kann das Bringen TB1 nicht passieren.

Wenn die Sensoren von TB1-4 gleichzeitig 1 sind, setzen wir den Merker M186.2 im Anhang 1 (Anzahl der Werkstücke auf TB1-4 <=3) 1. Damit können wir das Verhalten von Kran beschränken.

Aufgabe 6

Implementieren Sie eine Ablaufsteuerung für einen Bearbeitungszyklus der Maschine M1. Ein Bearbeitungszyklus für die Maschine M1 ist folgendermaßen definiert:

Der Drehtisch wird manuell mit einem Werkstück bestückt. Wird mit einem externen Taster (E4.6) bestätigen, dass ein neues Werkstück eingelegt wurde, soll sich der Drehtisch um 180 Grad(π) drehen. Anschließend soll der Werkzeugträger, wie im NCE-System von Abbildung 5.5, herunterfahren, das Werkzeug für 2 Sekunden bearbeitet werden. Abschließend soll der Werkzeugträger wieder nach oben fahren. Nach dem Einlegen eines neuen Werkstücks kann der Zyklus durch den Taster erneut gestartet werden.

Für die Aufgabe wird ein STEP 7 Projektvorlage bereitgestellt in der die SPS der Versuchsanlage bereits konfiguriert wurde. Verwenden Sie als Programmiersprache KOP oder AWL. Im Programm können Sie beliebige Merker definieren. Implementieren Sie zuerst den Teilzyklus nach Abbildung 5.5 und ergänzen Sie diesen Ablauf anschließend um

den Drehtisch und den Taster. Problematisch ist, dass der Sensor zur Detektion der Endposition des Drehtisches erst verzögert zum Start der Drehbewegung des Tisches seinen Wert ändert. Wie kann dieses Problem gelöst werden?

Antwort: Das Programm besteht aus 6 Netzwerke, Drehtisch drehen, Verzögerung, M1 senken, Spindel drehen, Bearbeitungszeit bestimmen, M1 heben.

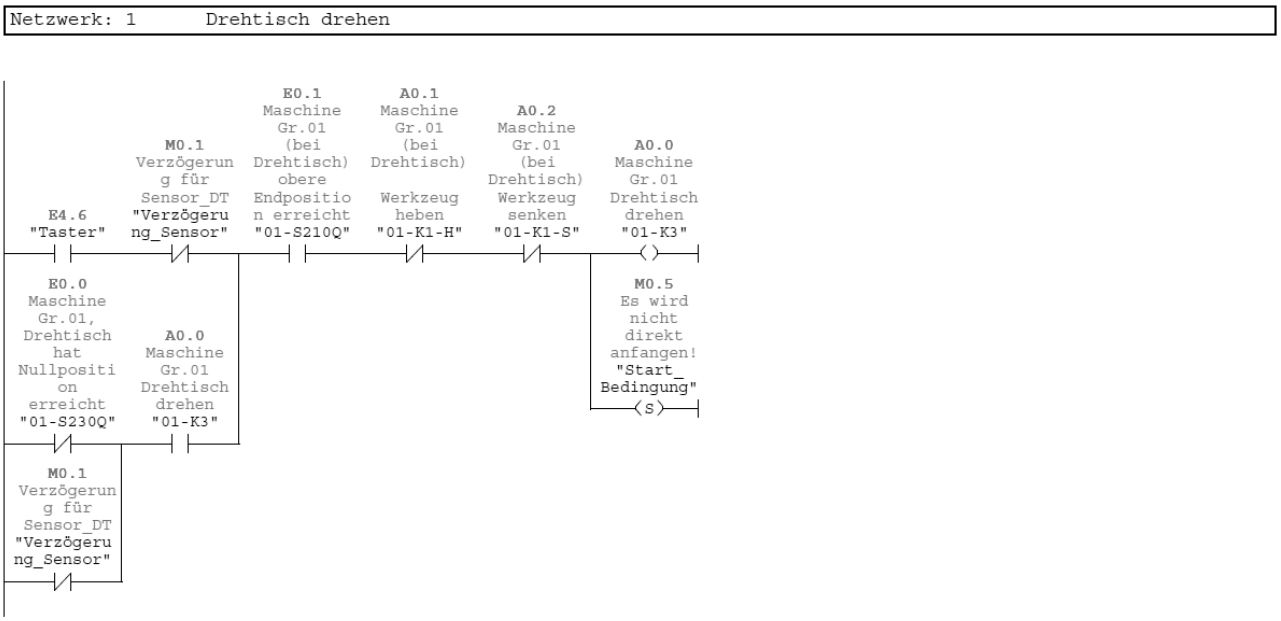

Abbildung 4.5 Netzwerk 1

Netzwerk 1 (Drehtisch drehen): Das Ziel dieses Schritts ist das Drehen des Drehtisches (A0.0 ist TRUE). Die Voraussetzungen dafür beinhalten, dass der Taster gedrückt wird (E4.6 ist TRUE), dass die Maschine M1 oben ist (E0.1 ist TRUE), dass Maschine M1 nicht bewegt (A0.1 ist FALSE und A0.2 ist FALSE). Als Parallelschaltung dient es, dass die Maschine M1 schon bewegt (A0.0 ist TRUE) und dass der Drehtisch die Endposition noch nicht erreicht (E0.0 ist FALSE). Hier mit der zweiten Parallelschaltung kann der Einfluss der Verzögerung des Sensors, welcher die Endposition vom Drehtisch detektiert, simuliert werden. Wenn der Drehtisch anfängt, gibt es eine Verzögerung für 500 ms, d.h. E0.0 wird nicht direkt von TRUE auf FALSE gesetzt und die erste Parallelschaltung ist deshalb offen. Für diese Situation ist M0.1 eine Ergänzung in der Verzögerung.

Eine andere neue Variable(M0.5) ist eingeführt, sodass die Maschine M1 beim Tasten(E4.6) nicht direkt anfangen wird. Der Anfangswert ist FALSE und es dient als eine Voraussetzung für Netzwerk 3. In dem ersten Zyklus wird es auf TRUE gesetzt. Und ab dem nächsten Zyklus wird es vom Speicher gezogen und dann kann Netzwerk 3 gestartet werden.

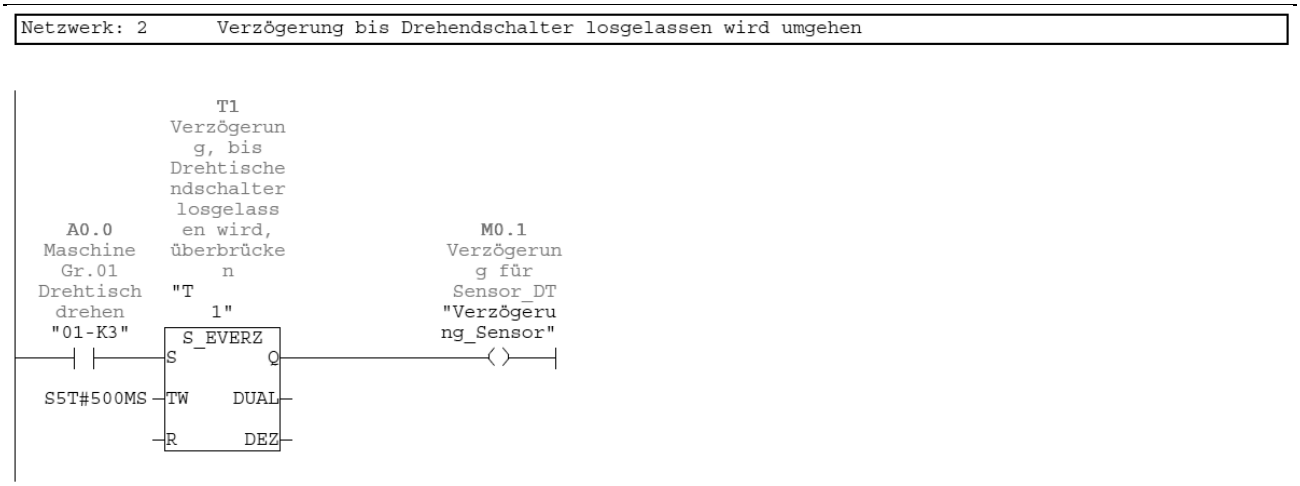

Abbildung 4.6 Netzwerk 2

Netzwerk 2 (Verzögerung): Dieser Schritt simuliert eine Verzögerung des Sensors, welcher die Endposition vom Drehtisch detektiert. Die Verzögerung dauert hier 500 ms. Die Simulation ist schon im Netzwerk 1 erklärt geworden.

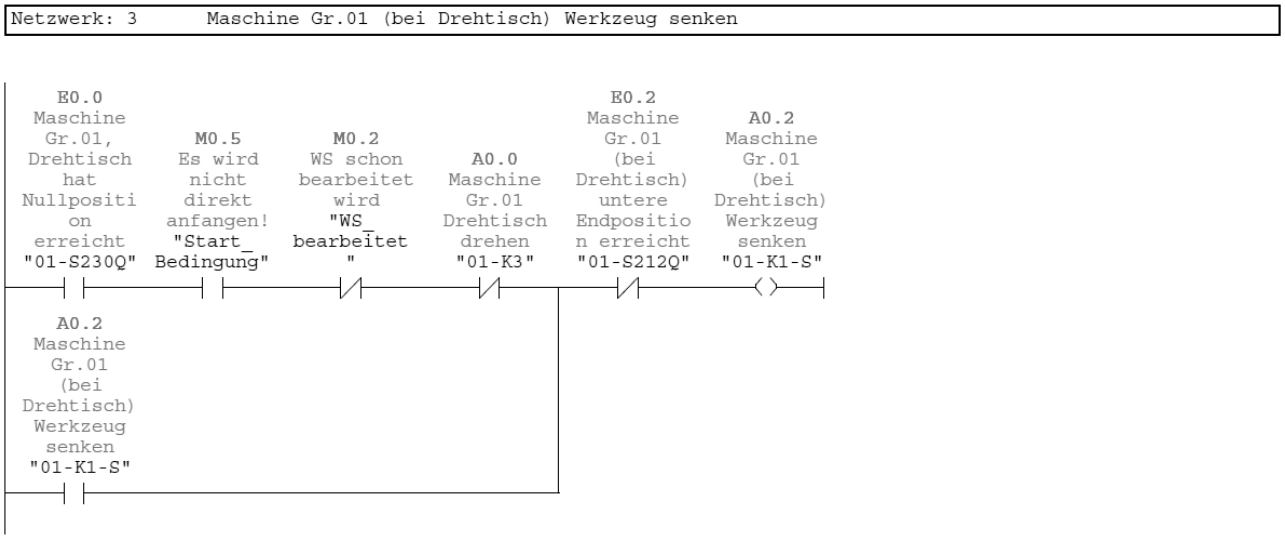

Abbildung 4.7 Netzwerk 3

Netzwerk 3 (M1 senken): Das Ziel dieses Schritts ist, dass die Maschine M1 senkt (A0.2 ist TRUE). Die Voraussetzungen dafür beinhalten, dass der Drehtisch die Endposition erreicht hat (E0.0 ist TRUE), dass die Start Bedingung für Maschine M1 auf TRUE gesetzt wird (M0.5 ist TRUE), dass das Werkstück noch nicht bearbeitet wird(M0.2 ist FALSE), dass der Drehtisch nicht bewegt (A0.0 ist FALSE) sowie dass die Maschine M1 die untere Endposition noch nicht erreicht hat (E0.2 ist FALSE). Als Parallelschaltung dient es, dass die Maschine M1 schon bewegt (A0.2 ist TRUE).

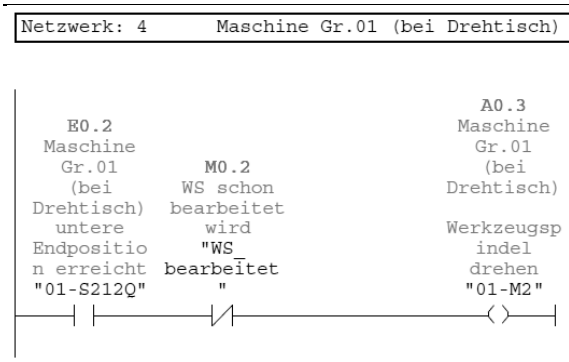

Abbildung 4.8 Netzwerk 4

Netzwerk 4 (Spindel drehen): Das Ziel dieses Schritts ist, dass die Werkzeugspindel dreht (A0.3 ist TRUE). Die Voraussetzungen beinhalten, dass die Maschine M1 die untere Endposition erreicht hat (E0.2 ist TRUE) und das Werkstück noch nicht bearbeitet wird (M0.2 ist FALSE).

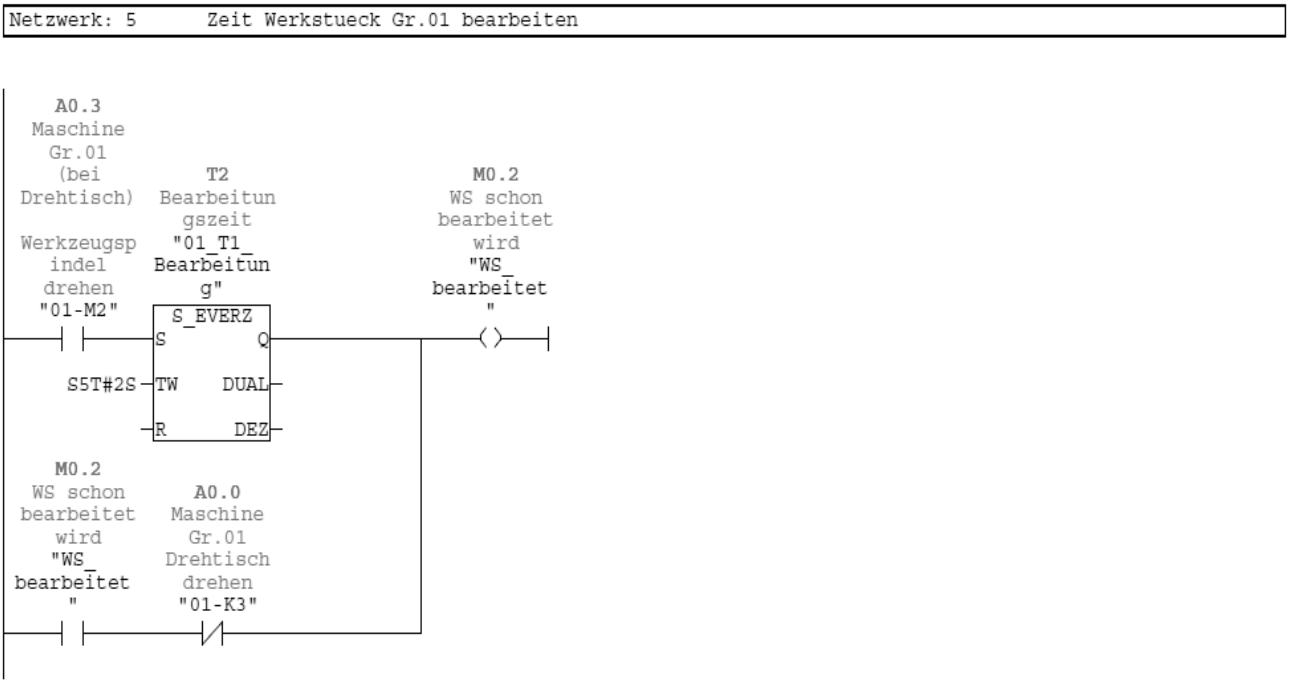

Abbildung 4.9 Netzwerk 5

Netzwerk 5 (Werkstück bearbeiten): Das Ziel dieses Schritts ist, dass der Status der Bearbeitung des Werkstücks auf TRUE gesetzt wird (M0.2 ist TRUE). Die Voraussetzungen dafür beinhalten, dass die Werkzeugspindel dreht (A0.3 ist TRUE), welche dauert 2 s. Als Parallelschaltung dient es, dass das Werkstück schon bearbeitet wird (M0.2 ist TRUE) und dass der Drehtisch nicht bewegt (A0.0 ist FALSE). D.h. der Status bleibt, wenn es auf TRUE gesetzt wird und der Drehtisch nicht bewegt.

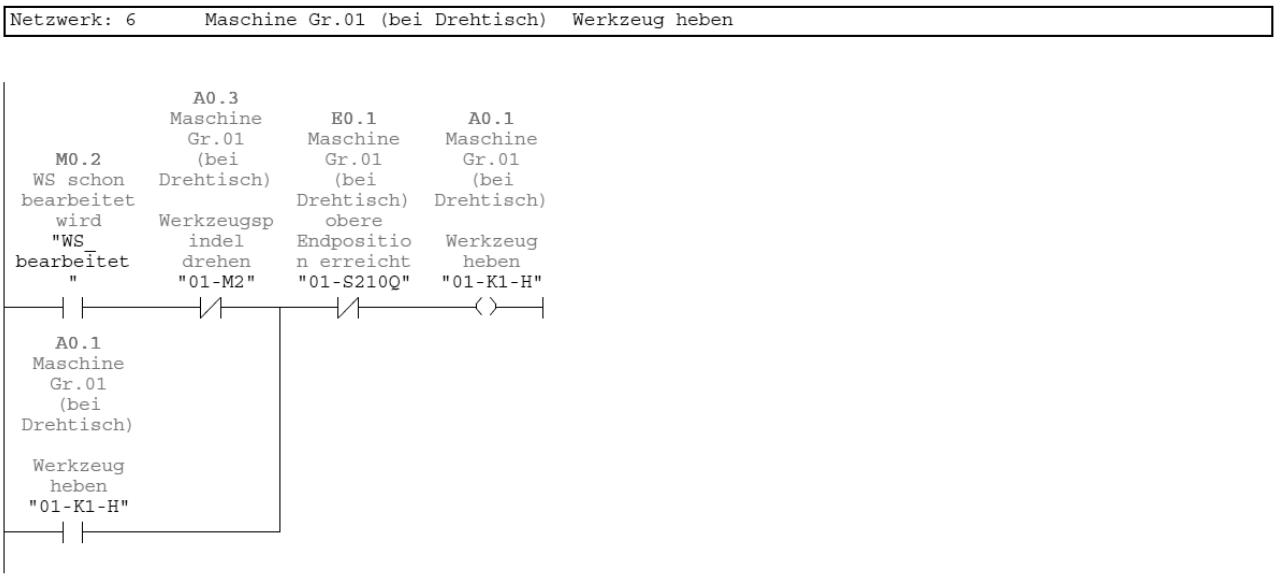

Abbildung 4.10 Netzwerk 6

Netzwerk 6 (M1 heben): Das Ziel dieses Schritts ist, dass die Maschine M1 hebet (A0.1 ist TRUE). Die Voraussetzungen dafür beinhalten, dass das Werkstück schon bearbeitet wird (M0.2 ist TRUE), dass die Werkzeugspindel nicht bewegt (A0.3 ist FALSE) sowie dass die Maschine M1 die obere Endposition noch nicht erreicht hat (E0.1 ist FALSE). Als Parallelschaltung dient es, dass die Maschine M1 schon bewegt (A0.1 ist TRUE).

Aufgabe 7

In den nachfolgenden Aufgaben sollen jeweils ausgewählte Teil des gesamten Steuerprogramms der vollständigen Fertigungsanlage implementiert werden. Für die Aufgaben wird ein STEP 7 Projekt bereitgestellt in dem bereits ein Großteil des Steuerprogramms enthalten ist. Die Struktur des kompletten SPS Programms ist in Anhang A.2 angegeben. Zur Integration der selbst zu erstellenden Programmteile in das bestehende Programm und zur Anbindung an die Visualisierungs- und Kontrolloberfläche, die mit dem Programm WinCC auf dem PC realisiert ist, müssen für die nachfolgenden Aufgaben die bereits definierten Merker aus dem Anhang A.1 verwendet werden. Verwenden Sie für diese und nachfolgende Aufgaben, sofern nicht anders angegeben, die Programmiersprache KOP.

Ergänzen Sie als erstes die Ablaufsteuerung für das Fertigungsprogramm 1 im Baustein FC50 um die in Aufgabe 5 entworfene Verriegelungssteuerung. Verwenden Sie hierfür einen im Anhang A.1 spezifizierten Merker für die Belegung der Transportbänder.

Antwort: Für das Transportband gibt es Forderungen an der Anzahl der Werkstücke auf TB1-4. Deswegen müssen 2 Merker im Programm hinzufügt, nämlich M186.1, welcher zeigt, dass die Anzahl der Werkstücke auf Transportband nicht größer als 3 ist, und M186.2, welcher zeigt, dass die Anzahl der Werkstücke auf Transportband nicht kleiner als 1 ist.

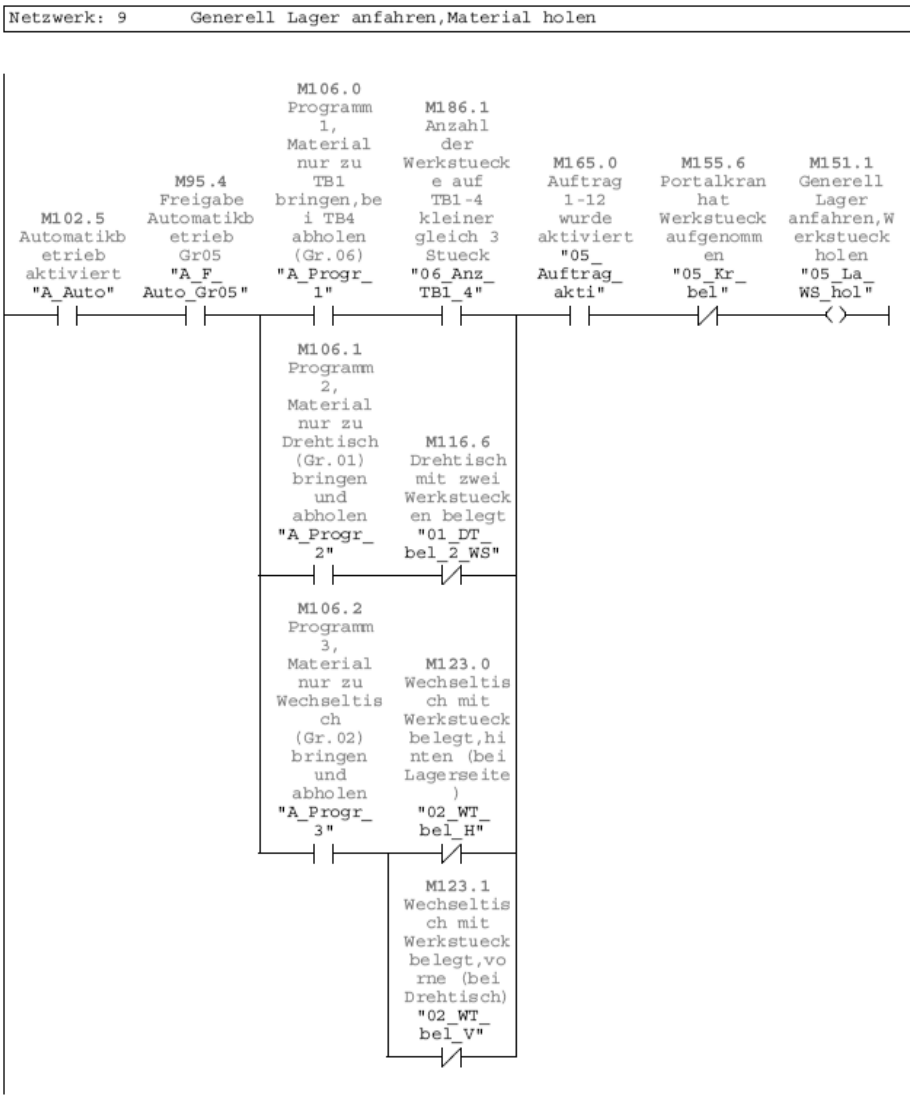

Abbildung 4.11 Netzwerk 9

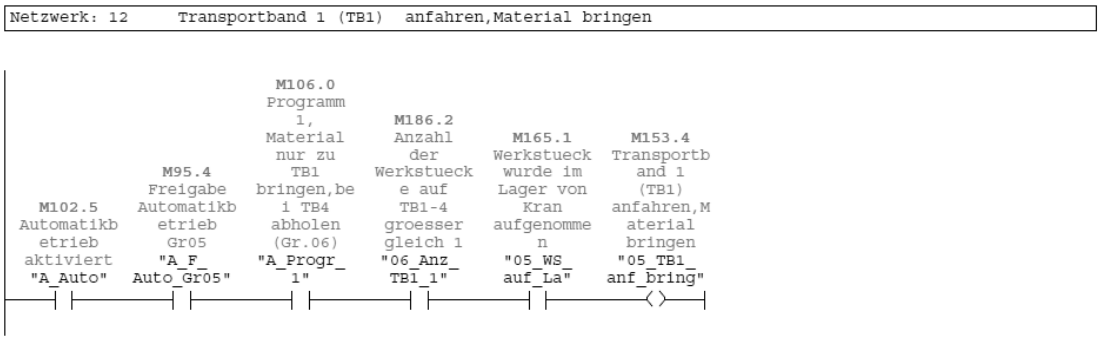

Abbildung 4.12 Netzwerk 12

Aufgabe 8

Implementieren Sie eine Steuerung für die Maschine M3. Gehen Sie dabei wir folgt vor:

1. Implementieren Sie zunächst eine Steuerung für den Tipp-Betrieb im Baustein FC30 (Netzwerke 5,6,8,9 und 12).

2. Fügen Sie dem Steuerungsprogramm einen neuen Funktionsbaustein FB32 hinzu, welcher im OB1 nach dem Baustein FC30 aufgerufen werden soll und implementieren Sie

die in Aufgabe 4 entworfene Steuerung mit Hilfe von S7-Graph. Der Ablauf soll im Halbautomatikbetrieb durch Betätigung des entsprechenden Schalters der Visualisierung oder im Automatikbetrieb bei Vorhandensein eines unbearbeiteten Werkstücks auf TB2 gestartet werden.

Antwort: 1. Die folgenden sind die Netzwerke für den Tipp-Betrieb im Baustein FC30. Das Prinzip vom Senken, Heben, Vorwärtsgang und Rückwärtsgang sind neben der konkreten Signale ähnlich. Hier werden nur Netzwerk 5 und Netzwerk 12 erklärt.

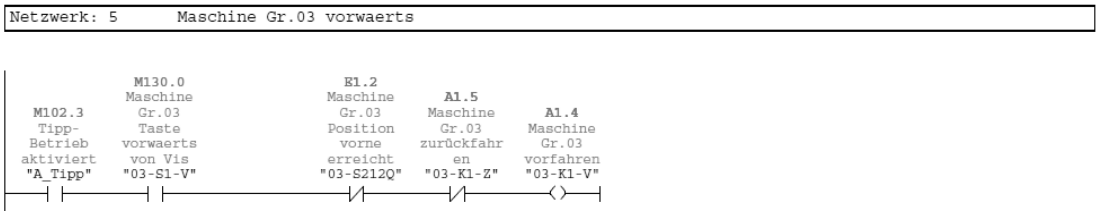

Abbildung 4.13 Netzwerk 5

Netzwerk 5: Der Ausgang ist das Antriebssignal des Vorwärtsgangs der Maschine M3 (A1.4 ist TRUE). Die Voraussetzungen dafür beinhalten, dass der Tipp-Betrieb aktiviert wird (M102.3 ist TRUE), dass der Taster beim WinCC gedrückt wird (M130.0 ist TRUE), dass die Maschine M3 die vorne Endposition nicht erreicht hat (E1.2 ist FALSE), dass Maschine M3 nicht zurückfahrt (A1.5 ist FALSE).

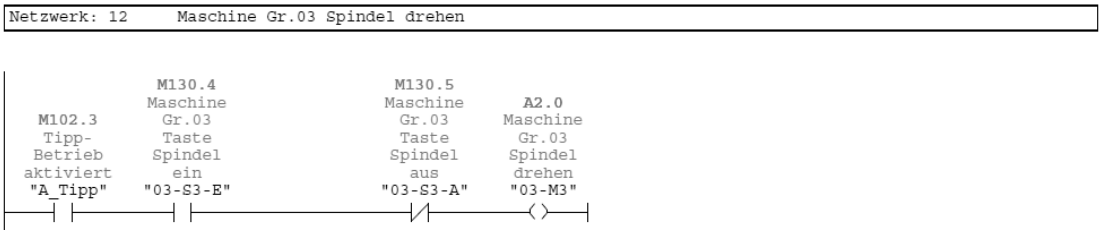

Abbildung 4.14 Netzwerk 12

Netzwerk 12: Der Ausgang ist das Antriebssignal des Drehens des Spindelmotors (A2.0 ist TRUE). Die Voraussetzungen dafür beinhalten, dass der Tipp-Betrieb aktiviert wird (M102.3 ist TRUE), dass der Taster „Ein“ beim WinCC gedrückt wird (M130.4 ist TRUE) und dass der Taster „AUS“ beim WinCC nicht gedrückt wird (M130.5 ist FALSE).

2. Das Steuerungsprogramm basiert auf das Petri-Netz aus Aufgabe 4.

Zu beachten ist der Beginn des Programms. Es gibt 2 parallele Voraussetzungen, nämlich 2 Betriebsarten (Halbautomatik und Automatik). Bei der Halbautomatik beinhalten die Eingänge den Merker (M102.4 "A_Halb_Auto") für Halbautomatischen Betrieb, welcher die Aktivierungstand dieser Betriebsart zeigt, und den Merker (M131.3 "03_WS_Start"), welcher den Zyklus der Maschine M3 startet. Bei der Automatik beinhalten die Eingänge den Merker (M102.5 "A_Auto") für Automatischen Betrieb, welcher die Aktivierungstand dieser Betriebsart zeigt, und die 2 Merker jeweils für eine Anforderung an dem Transportband 2. Der eine ist E2.3 "06-S220Q", welcher zeigt, dass Transportband 2 belegt wird. Der andere ist die Negation vom Merker M182.0 "06_WS_be_TB2", welche zeigt, dass das Werkstück auf Transportband 2 noch nicht bearbeitet wird.

Aufgabe 9

Nun soll eine Steuerung für Maschine M3 implementiert werden, welche robust gegenüber Variationen des Anfangszustands der Maschine ist. Dies ist beispielsweise notwendig, wenn der Automatikbetrieb aktiviert wird, obwohl sich die Maschine noch nicht in der Ruheposition befindet. Implementiert werden soll die Steuerung durch Ergänzung der Netzwerke im Baustein FC30. Löschen Sie daher vor der Programmierung des Bausteins den soeben neu eingefügten Baustein FB32 und den entsprechenden Aufruf im OB1.

Die zu implementierende Steuerung soll die Vorgaben der Spezifikation vom Abschnitt 1.2 erfüllen. Folgende Hinweise sollen bei der Lösung der Aufgabe hilfreich sein:

- Die Motoren für die Bewegung der Maschine sollen im Halbautomatikbetrieb nach dem Betätigung des entsprechenden Knopfs in der Visualisierung solang aktiv sein, bei der jeweilige Endschalter erreicht worden ist.
- Im Tipp- und Halbautomatikbetrieb soll die Spindel durch den entsprechenden Knopf in der Visualisierung eingeschaltet werden können. Dies soll jedoch nicht erlaubt sein, wenn der Ausknopf gedrückt wird, im Halbautomatikbetrieb nur dann, wenn kein Zyklus ausgelöst worden war und im Tippbetrieb nur solange der Knopf gedrückt gehalten wird.
- Im Halbautomatikbetrieb soll durch einen entsprechenden Knopf in der Visualisierung ein kompletter Zyklus ausgelöst werden können.
- Befindet sich ein zu bearbeitendes Werkstück im Automatikbetrieb an der Maschine oder wurde im Halbautomatikbetrieb ein kompletter Zyklus ausgelöst, so soll sich die Maschine vorne unten befinden, solange der Bearbeitungsvorgang nicht abgeschlossen wurde, andernfalls soll sie hinten oben sein. Ein Werkstück gilt als bearbeitet, nachdem der Timer für die Spindel abgelaufen ist.
- Werden die Förderbänder betätigt, soll der Zyklus abgebrochen werden und die Maschine in Ruheposition fahren.

Antwort: Im Baustein FC30 werden 9 Netzwerke ergänzt, um die Steuerung für Maschine M3 im Tipp-, Halbautomatik- und Automatikbetrieb durchzuführen. In der Steuerung vom Halbautomatikbetrieb bestehen 2 Arten. Einerseits können die Bewegung der Maschine M3 durch den entsprechenden Knopf in der Visualisierung ausgewählt und ausgeführt werden. Anderseits kann ein Zyklus durch die Taste Start in der Visualisierung ausgelöst werden.

Hier werden die ergänzten Netzwerke erklärt.

Antwort:

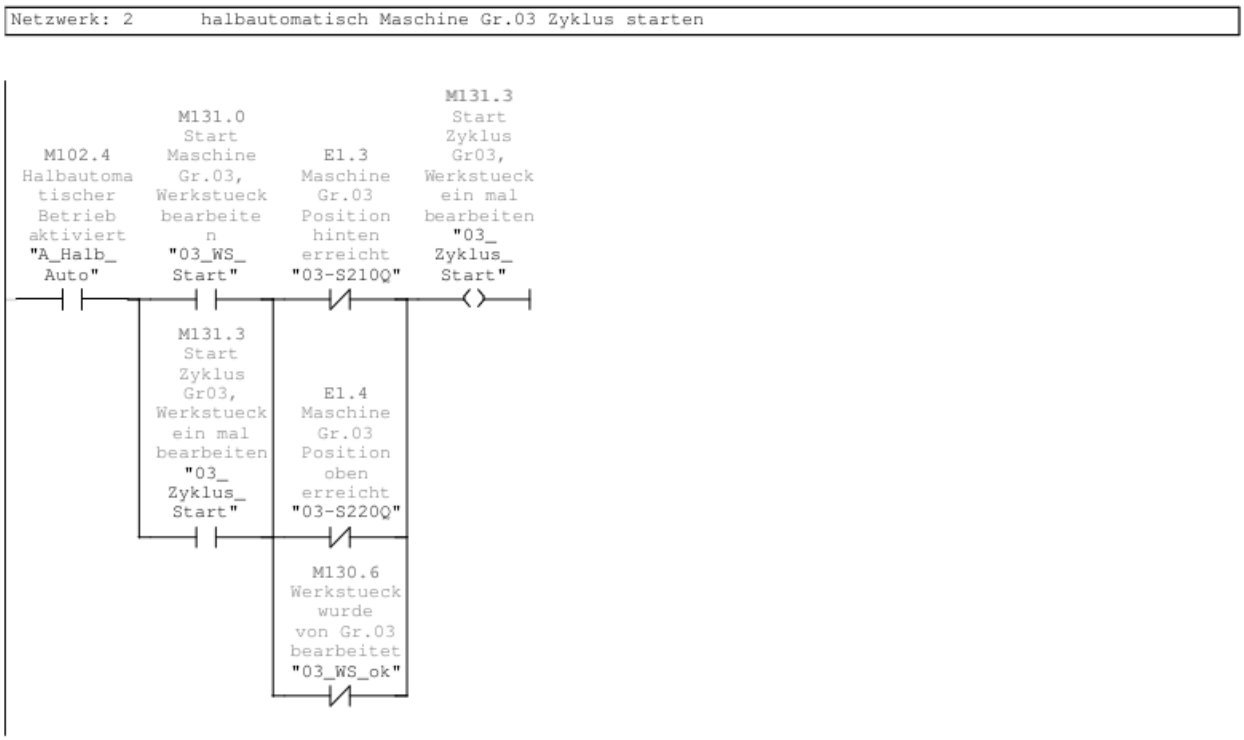

Abbildung 4.15 Netzwerk 2

Netzwerk 2: weil im Halbautomatikbetrieb ein kompletter Zyklus ausgelöst werden kann, muss es beurteilt werden, ob die Maschine M3 im Zyklus ist. Um das Zyklus im Halbautomatikbetrieb zu starten, müssen die Taste Halbautomatikbetrieb und Taste Maschine M3 Zyklus Start im WinCC betätigt werden, was dazu führt, dass der Merker M102.4 und M131.0 auf Eins gesetzt wird. Wenn die Maschine M3 die Ruheposition (hintere und obere Endposition) nicht erreicht oder ein Werkstück nicht fertig bearbeitet ist, ist der Vorgang noch in einem Zyklus, das heißt, der Merker M131.3 ist gleich Eins. Im Programm zeigt der Merker M130.6 “03_WS_ok“ an, ob der Bearbeitungsvorgang eines Werkstücks abgeschlossen ist, dessen Zustand im Netzwerk 4 bestätigt wird.

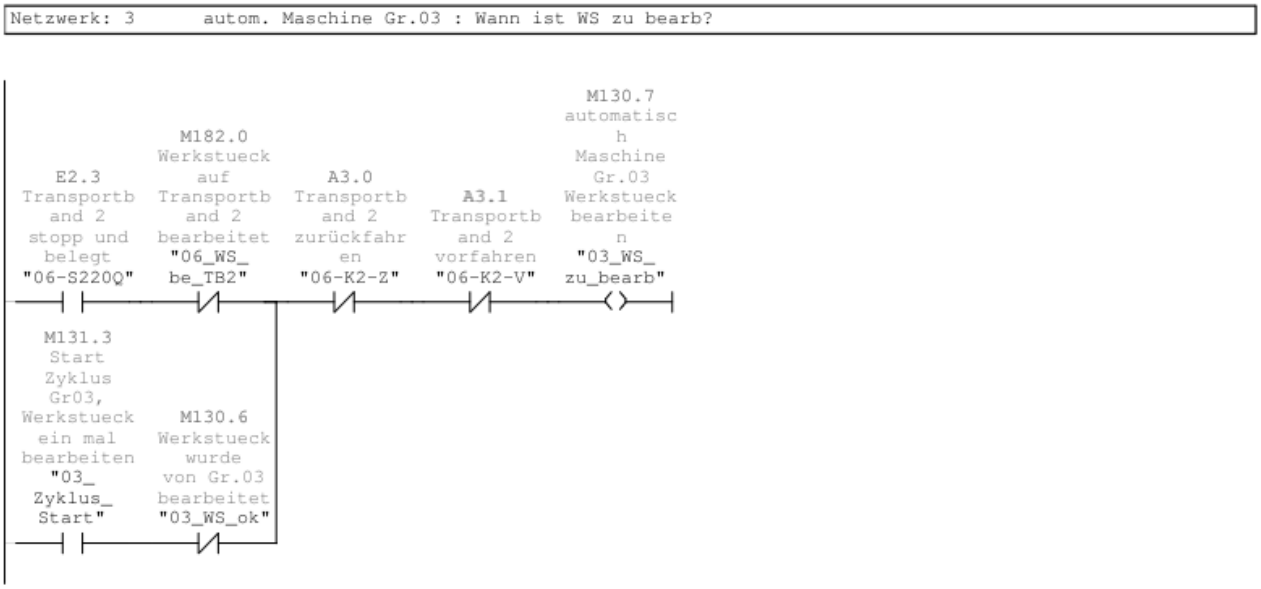

Abbildung 4.16 Netzwerk 3

Netzwerk3: in diesem Netzwerk wird der Wert vom Merker M130.7 anhand des Betriebszustands geändert. Der Merker M130.7 zeigt an, ob ein Werkstück noch bearbeitet werden muss. Wenn der Merker M131.3 gleich Eins ist, nämlich ist die Maschine M3 im Zyklus, und der Werkstück nicht von der Maschine M3 bearbeitet wurde sowie das Transportband 2 nicht vorfährt und zurückfährt, wird der Merker M130.7 auf Eins geändert. Falls die Maschine M3 nicht im Zyklus ist, dann muss das Transportband 2 stoppen und belegt werden, was den Eingang E2.3 entspricht. Außerdem wird der Merker M182.0 gebraucht, der im Netzwerk 13 vom Baustein FC60 kontrolliert wird und zeigt, ob das Werkstück auf TB 2 bearbeitet ist.

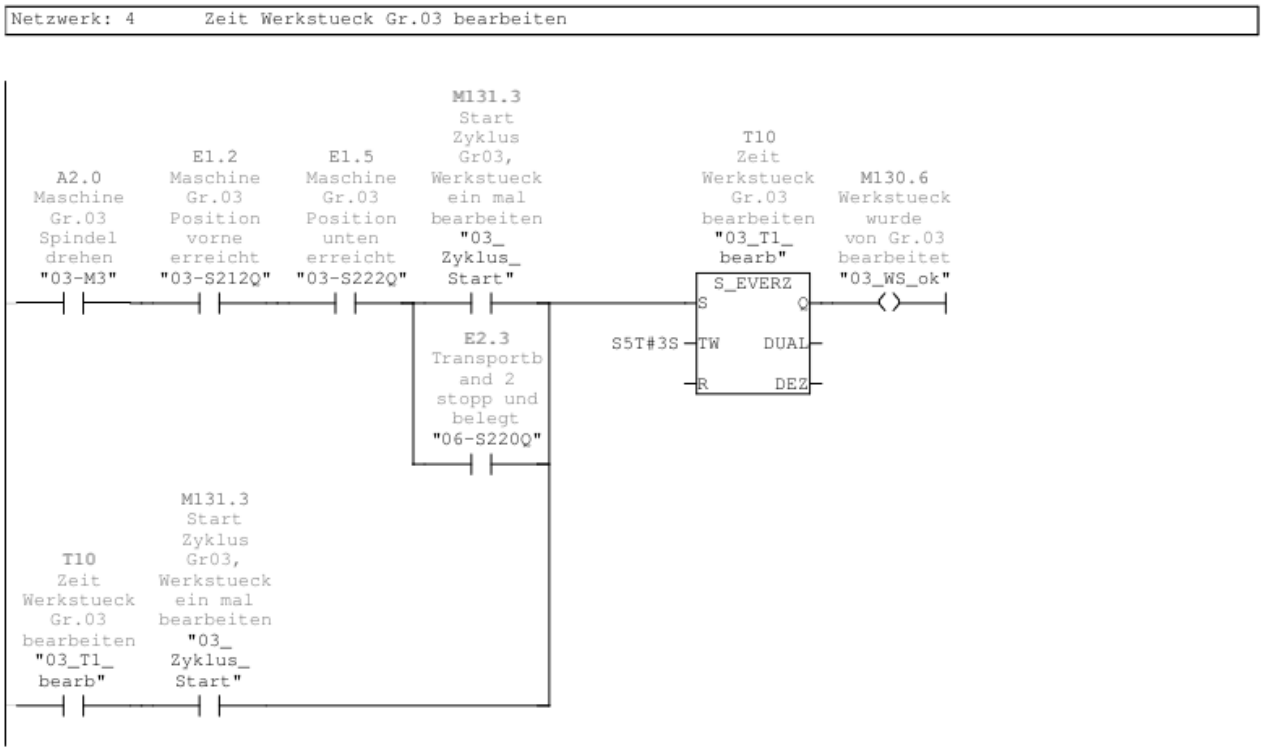

Abbildung 4.17 Netzwerk 4

Netzwerk4: nachdem die Spindel der Maschine M3 dreht, muss die Bearbeitungszeit für 3s gesichert werden. Hier wird ein Timer T10 verwendet. Nachdem der Timer für die Spindel abgelaufen ist, ändert sich der Merker M130.6 auf Eins und gilt ein Werkstück als bearbeitet.

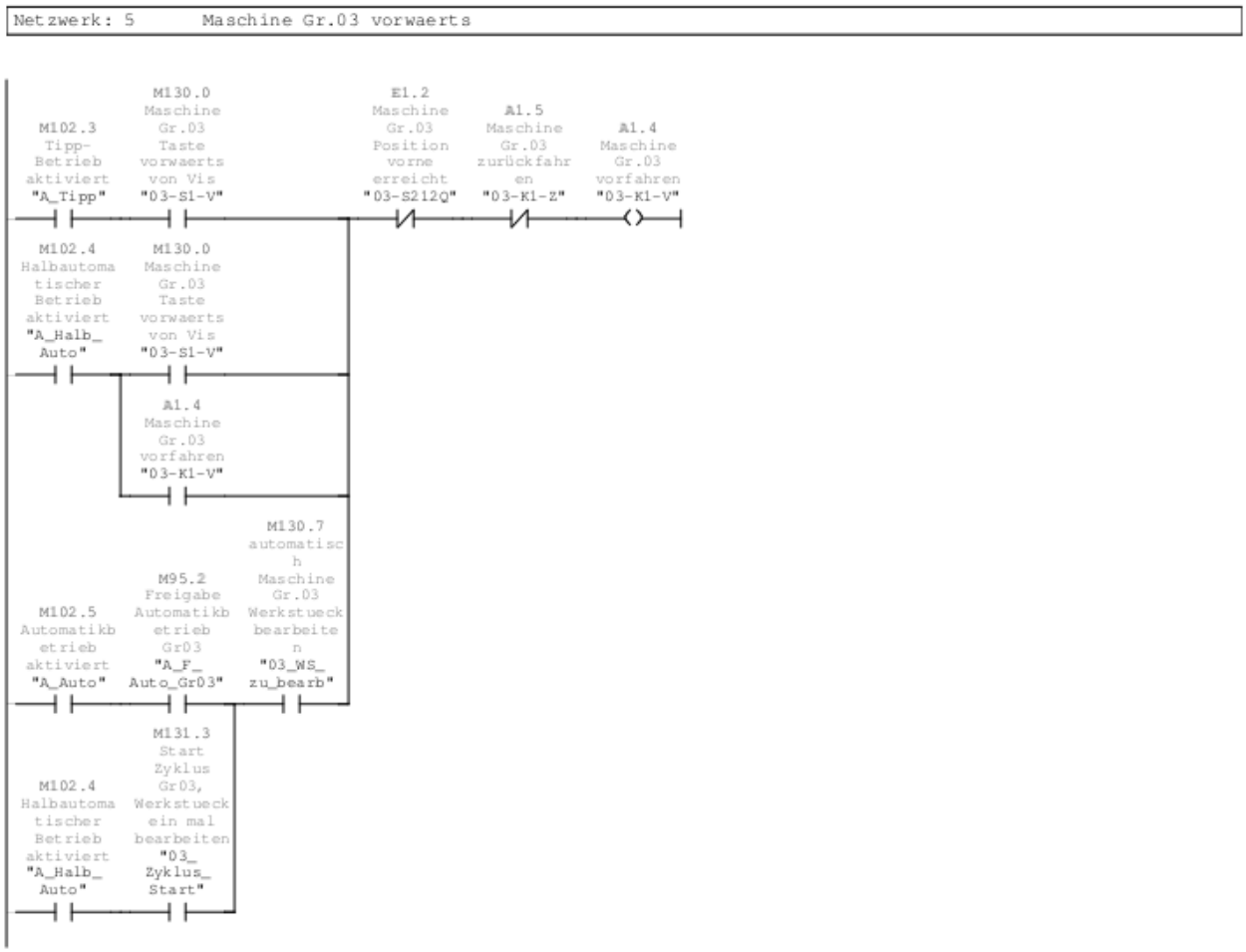

Abbildung 4.19Netzwerk 5

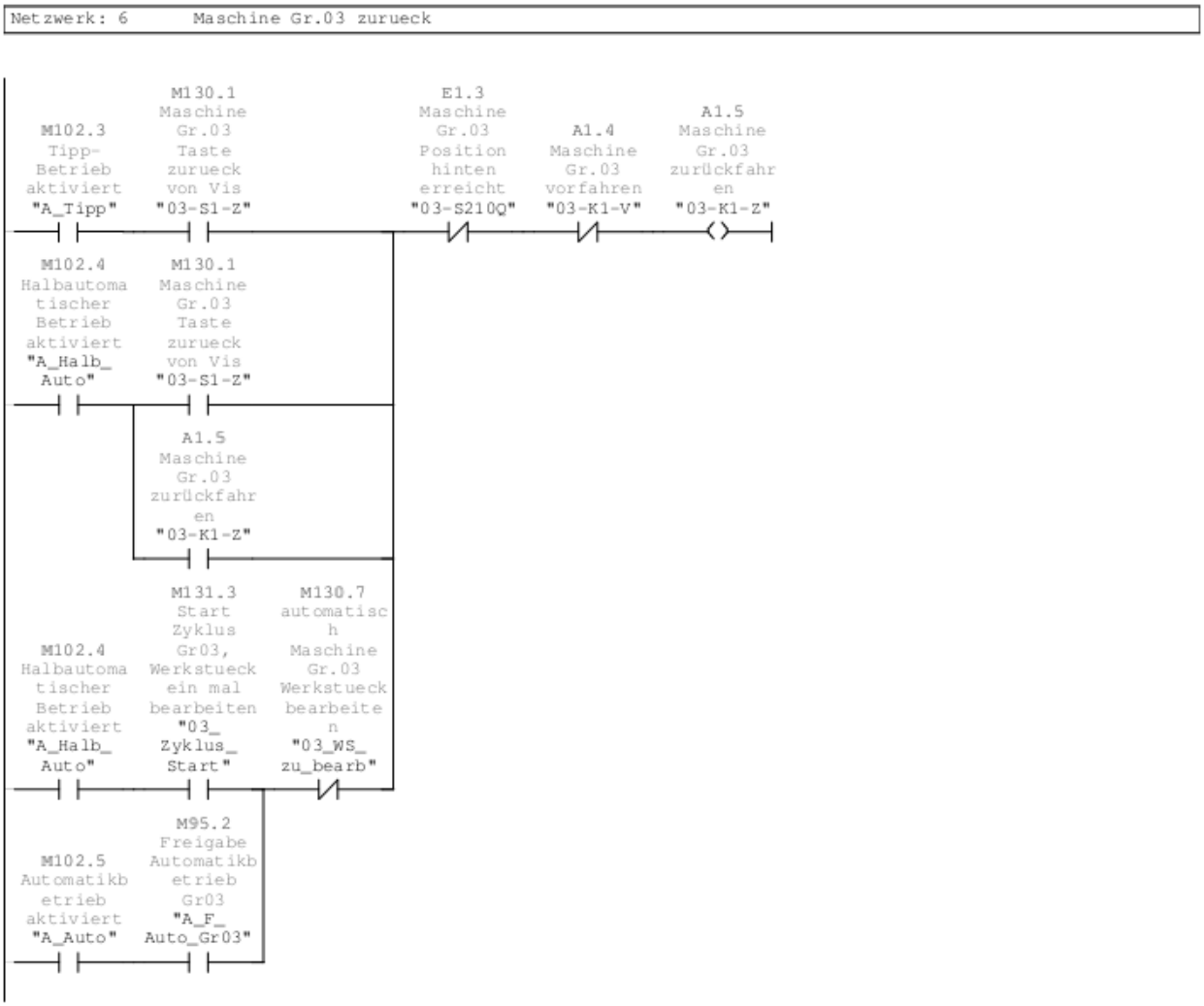

Abbildung 4.18 Netzwerk 6

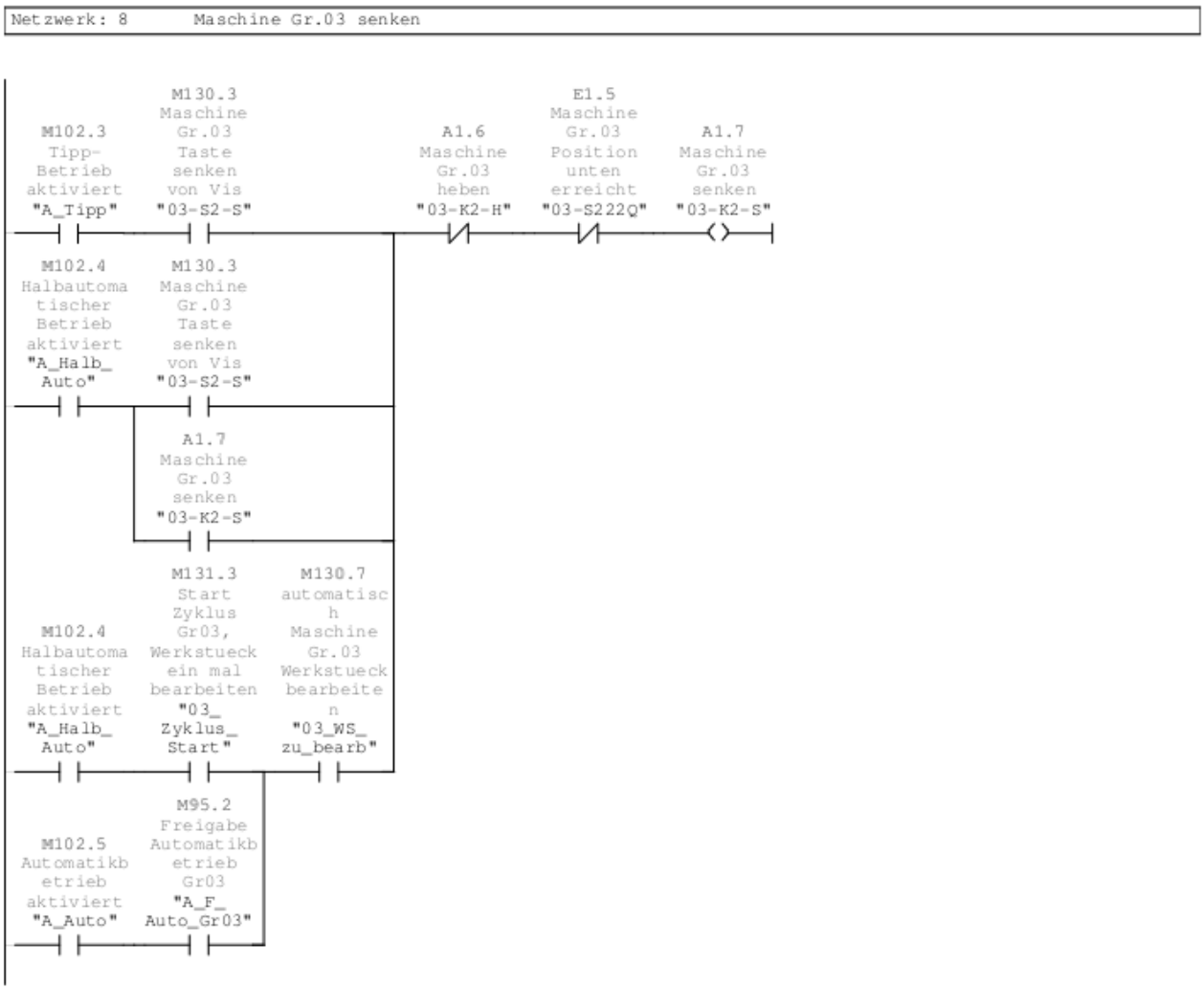

Abbildung 4.19 Netzwerk 8

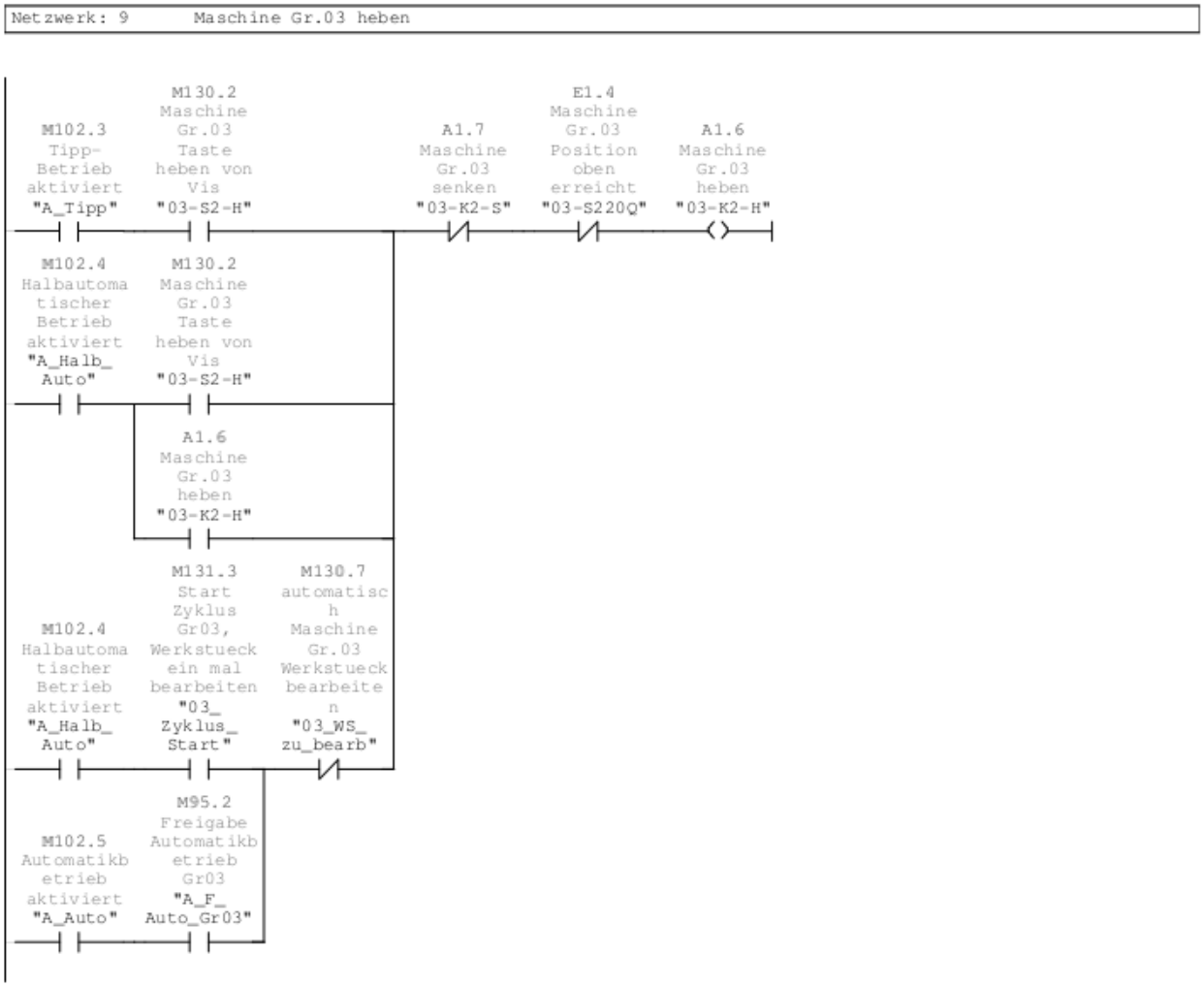

Abbildung 4.20 Netzwerk 9

Netzwerk5 – Netzwerk6, Netzwerk8 – Netzwerk9: die Bewegung der Maschine M3 wird in diesen vier Netzwerke gesteuert. Was zu beachten ist, dass im Automatikbetrieb eine Taste “Freigabe Automatikbetrieb M3“ in der Visualisierung eingeschaltet wird muss vor der Bewegung der Maschine M3. Im Programm benutzen wir den entsprechenden Merker M95.2.

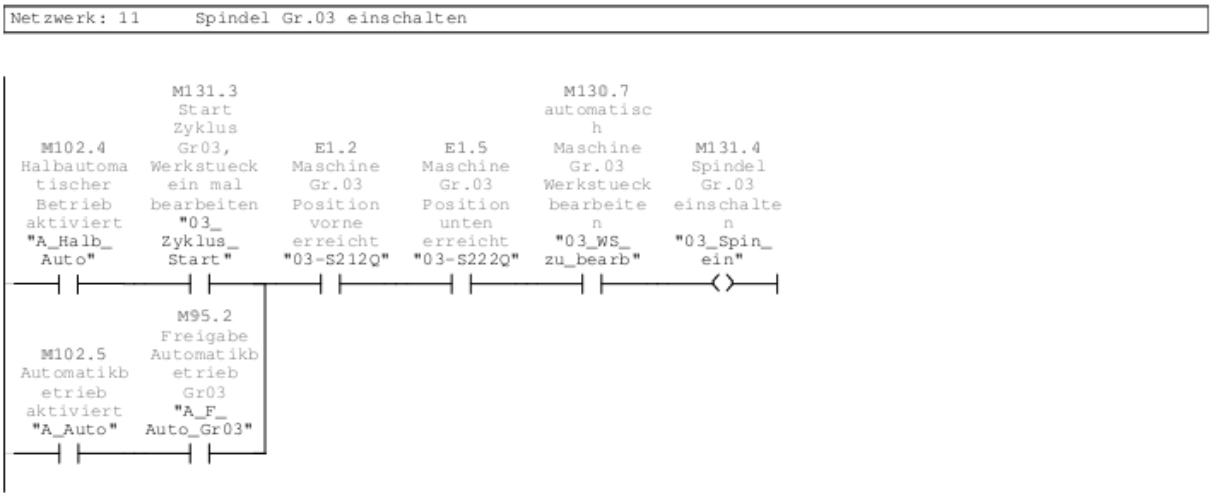

Abbildung 4.21 Netzwerk 11

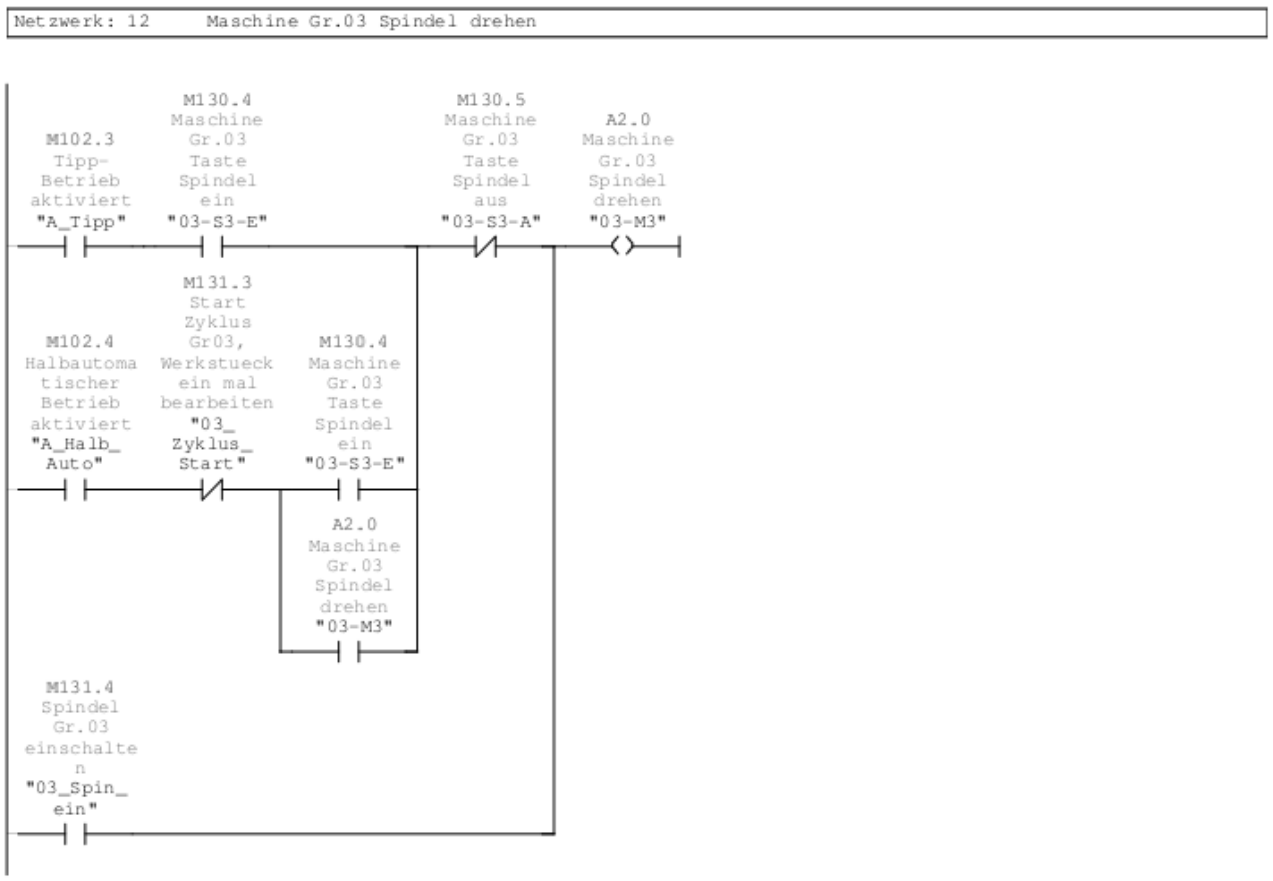

Abbildung 4.22 Netzwerk 12

Netzwerk11 – Netzwerk12: die zwei Netzwerke betreffen die Steuerung der Spindel der Maschine M3. Im Netzwerk 11 wird der Merker M131.4, der besonders in der Steuerung für das Zyklus verwendet wird, auf Eins gesetzt nachdem alle Voraussetzungen erfüllt sind. Die Voraussetzungen dafür sind, der Ablauf im Zyklus des Halbautomatikbetrieb ist oder im Automatikbetrieb mit Freigabe ist, dabei befindet sich die Maschine M3 in der Bearbeitungsposition und ist der Merker M130.7 “03_WS_zu_bearb“ gleich Eins. Der Merker M131.4 wird dann später im Netzwerk 12 ausgewertet und sich so auf die Belegung des Ausgangs A2.0 “03-M3“ auswirkt.

Im Netzwerk12 wird die Belegung des Ausgangs A2.0 unter 3 möglichen Voraussetzungen berücksichtigt, nämlich im Tipp-Betrieb, im Zyklus des Halbautomatikbetriebs und des Automatikbetriebs, sowie im einfachen Halbautomatikbetrieb. Im Zyklus wird der Ausgang A2.0 von den Merker M131.4 gesteuert. Im einfachen Halbautomatikbetrieb oder Tipp-Betrieb wird der Ausgang nicht auf null geändert solange die Taste “Spindel aus“ nicht gedrückt wird. Außerdem soll die Taste “Spindel ein“ betätigt werden.

5. Auftretende Probleme und Diskussion

Bei der Aufgabe 6 tritt ein Problem auf, welches uns für eine ganze Woche verwirrt hat. Nachdem wir die KOP geladen, funktioniert das Programm nur für einen Zyklus. Wenn der Taster (E4.6) zum zweiten Mal gedruckt wird, bewegen nur der Drehtisch und die Maschine M1 nicht. Wir haben vielmals die Logik des ganzen Programms analysiert und sind davon überzeugt, dass alle Ein-/Ausgänge sowie deren Zusammenhang richtig sind. Schließlich

finden wir die Ursache, dass wir S_SEVERZ anstatt S_EVERZ als das Timer gewählt haben. Das erste wird den Ausgang auf TRUE setzen. Dann wird M0.2 immer auf TRUE gesetzt und die Maschine M1 kann nicht bewegen.

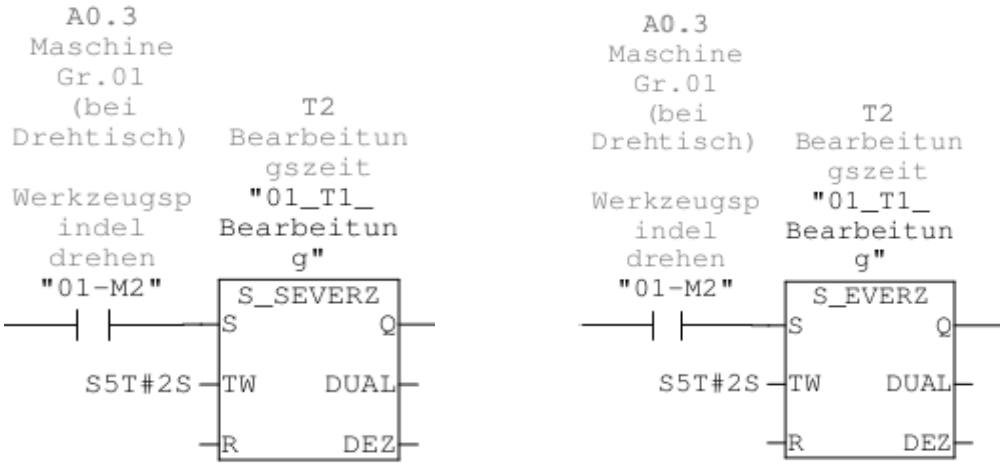

Abbildung 6.1 falsches und richtiges Timer

Im SPS Programm werden normalerweise einige Merker ergänzt, um die Sicherheit des ganzen Systems zu gewährleisten. Zum Beispiel im Netzwerk5 des Bausteins FC30 von der Aufgabe 9, der Ausgang A1.5 werden gebraucht um zu sichern, dass die Maschine M3 nicht fährt zurück. Das kann auch den möglichen Konflikt vermeiden.

6. Literatur

[1] Folien und Beiblätter „Automatisierung ereignisdiskreter und hybrider Systeme“