# Beispielsammlung (1. Beispiel)

Laborübung Sensor/Aktor-Systeme

Wolfgang Kastner

SS 2007 (Vers. 1.5)

## 1 Allgemeines

Dieser erste Teil der Beispielsammlung enthält die Kollektion aller Angaben zum 1. Beispiel, das Sie mit den Grundlagen der SPS-Programmierung mit Kontaktplänen (KOP) vertraut machen soll. Alle diese Beispiele verwenden nur die direkt an die SPS angeschlossenen 8 Schalter Swt7–Swt0 (E0.7–E0.0) und die 8 LEDs LED7–LED0 (A0.7–A0.0) des Bedienpanels. De facto besteht jedes Beispiel aus drei unabhängigen Teilen, die als eigenständige Programm-Files mit Namen `Matrikelnummer_I_{a1,a2,b,c1,c2}` ausgearbeitet werden müssen:

a. *Digital-I/O*

- Bit-Operationen
- Sondermerker

b. *Timer-Programmierung*

- Verwendung der Timer
- Byte/Word/Double-Operationen

c. *Ablaufsteuerung*

- Kettenprogrammierung (SCR's)

Die konkrete Nummer $nr$ des von Ihnen zu lösenden Beispiels ergibt sich zu

$$nr = (E + J) \text{ modulo } 5,$$

wobei $E$ die **Einerstelle** Ihrer Matrikelnummer und $J$ die Einerstelle des aktuellen Jahres bezeichnet. Im Falle von $nr = 3$ müssen Sie z.B. die Beispiele I.a.3, I.b.3 und I.c.3 bearbeiten.

Dokumentieren Sie bitte jede der drei Teillösungen durch kommentierte (!) Netzwerke unter Verwendung einer Symboltabelle (Name, Adresse, Kurzbeschreibung der Variablen). Erstellen Sie dann ein Abgabeprotokoll, das folgende Punkte beinhalten soll:

1. Name, Matrikelnummer und Studienkennzahl.
2. Symboltabelle.
3. Beschreibung eventueller Besonderheiten der Lösung und Modifikationen der Aufgabenstellung des jeweiligen Beispiels.
4. Beantwortung folgender Fragen:

- Wieviel Ausführungszeit benötigt in etwa Ihr *Digital I/O Programm*? Verwenden Sie fr die Abschtzung einerseits Appendix F des S7-200 Systemhandbuchs, andererseits einen Timer.
- Wie werden 1ms, 10ms und 100ms Timer in der S7-214 aktualisiert und worauf ist dabei zu achten?
- Wie verhalten sich Spulen in Ablaufrelais?

5. Probleme, Kritik und Anregungen[1]

   - bei der Beispiellösung
   - im Umgang mit dem Targetsystem und den Entwicklungstools
   - allgemeine Anregungen und Beschwerden bzgl. der Laborübung

Wir bitten Sie, nach Fertigstellung Ihrer Lösung das Laborprotokoll auszudrucken und während der betreuten Übungszeiten Ihrem Betreuer abzugeben und ihm die Lösung der Beispiele zu zeigen.

# 2 Beispielangaben

## Teil a: Digital I/O

Schreiben Sie ein Programm, das ständig den Zustand der Schalter Swt0–7 (hier als $S_0$–$S_7$ bezeichnet) auf LED0–7 anzeigt. Berechnen Sie mit Hilfe von elementaren Bit-Anweisungen unter Verwendung von

1. Kontakten, Spulen
2. Kontakten, Setz- und Rücksetz-Operationen

den logischen Ausdruck

$$\begin{array}{ll}
\text{Beispiel I.a.0:} & X := ((S_0 \wedge \neg S_1) \vee S_2) \wedge ((S_3 \wedge \neg S_4) \vee S_5 \vee (S_6 \wedge \neg S_7)) \\
\text{Beispiel I.a.1:} & X := (\neg S_0 \vee S_1 \vee S_2) \wedge (\neg S_3 \vee \neg S_4 \vee S_5 \vee (S_6 \wedge S_7)) \\
\text{Beispiel I.a.2:} & X := (S_0 \wedge S_1 \wedge S_2) \vee (((\neg S_3 \wedge S_4) \vee S_5) \wedge (\neg S_6 \vee \neg S_7)) \\
\text{Beispiel I.a.3:} & X := ((S_0 \vee S_1) \wedge \neg S_2 \wedge (S_3 \vee \neg S_4) \wedge S_5) \vee (\neg S_6 \wedge S_7) \\
\text{Beispiel I.a.4:} & X := (((S_0 \vee S_1) \wedge S_2) \vee \neg S_3) \wedge (\neg S_4 \vee (S_5 \wedge (\neg S_6 \vee \neg S_7)))
\end{array}$$

und lassen Sie im Falle $X$ = TRUE den momentanen Anzeigewert auf LED0–7 mit 1 Hertz blinken. Beachten Sie, daß —ohne Klammern— der binäre Negationsoperator $\neg$ stärker als die UND-Verknüpfung $\wedge$, und $\wedge$ stärker als die ODER-Verknüpfung $\vee$ bindet.

## Teil b: Timer-Programmierung

Schreiben Sie ein Programm, das wartet, bis auf den Schaltern Swt0–7 (hier wieder als $S_0$–$S_7$ bezeichnet) mindestens $T_s$ Sekunden lang stabil ein Bitmuster $S = S_7 S_6 \ldots S_0$ (interpretiert als unsigned Byte) eingestellt bleibt, das die Bedingung $C(S)$ = TRUE für den in der untenstehenden Tabelle gegebenen C-Ausdruck $C(S)$ erfüllt, und daraufhin folgendes durchführt:

[1] Bitte beachten Sie, daß gerade dieser Abschnitt für die Verbesserung der Laborübung sehr wichtig ist. Wir werten jeweils vor Beginn eines neuen Semesters die Protokolle der Übungsteilnehmer des vorigen Semesters aus, um z.B. jene Dinge zu identifizieren, die unnötige oder unerwartete Schwierigkeiten bereitet haben.

1. Den in der Tabelle angegebenen Wert $I(S)$ in einen Zähler/Rotator/Shifter Funktionsblock $B$ lädt.
2. Den aktuellen Ausgabewert $O$ von $B$ (ggf. dessen least significant Byte) kontinuierlich auf LED7–0 ausgibt.
3. Einen periodischen Timer mit Periodendauer $T_P$ Sekunden startet, der $B$ taktet und schließlich stoppt, wenn die Abbruchbedingung $E(O, S) = \text{TRUE}$ erfüllt ist.

Nach dem Stoppen des periodischen Timers soll die Programmausführung wieder von vorne anfangen; auch die Zeit $T_s$ für den nächsten Durchgang muß ab diesem Zeitpunkt neu zu laufen beginnen. Experimentieren Sie mit den unterschiedlichen Timer-Typen (1ms, 10ms, 100ms) der SPS.

Die folgende Tabelle enthält für jedes Beispiel die konkreten Werte für $T_s$, $C(S)$, $B$ usw.:

| **Bsp.** | $T_s$ [s] | $C(S)$ | $B$ | $I(S)$ | $T_P$ [s] | $E(O, S)$ |
|---|---|---|---|---|---|---|
| I.b.0 | 1.7 | $(S \,\&\, 0\text{xF0}) = 0\text{x50}$ | SHL_B | $S \mid 0\text{x0F}$ | 0.9 | $O = 0$ |
| I.b.1 | 1.3 | $(S \,\hat{}\, 0\text{x55}) \neq 0\text{xFF}$ | CTD | $(S \ll 8) \mid 0\text{xFF}$ | 0.02 | $O = S$ |
| I.b.2 | 2.7 | $(S \,\&\, 0\text{x55}) = 0\text{x55}$ | SHR_W | $(S \ll 8) \mid S$ | 1.3 | $O = 0$ |
| I.b.3 | 1.9 | $(S \,\&\, 0\text{xAA}) \neq 0\text{x00}$ | ROL_B | $S$ | 1.2 | $O = S$ |
| I.b.4 | 1.9 | $((S \,\&\, 0\text{x0F}) = 0\text{x00}$ | CTU | $S$ | 0.03 | $O = (0\text{x100} \mid S)$ |

**Hinweise:**

- Verwenden Sie für $B$ Shift/Rotate/Count-Funktionsblöcke, die um 1 weiterschieben oder -zählen.
- Bei unserer CPU-214 gibt es keinen Rückwärtszähler CTD; verwenden Sie CTUD dafür.
- Beachten Sie, daß Zähler wie etwa CTUD durch eine Word-Zuweisung and den Zähler $Z$ selbst und nicht an $PV$ geladen werden.

## Teil c: Ablaufsteuerung

Schreiben Sie ein Programm, das mit Hilfe von

1. Schrittketten (SCR's)
2. Merkern

die in der untenstehenden Tabelle des jeweiligen Beispiels gegebene State-Machine realisiert. $S \in 0, \ldots, 7$ ist dabei der aktuelle Zustand, $S^+$ der durch das Ereignis $e$ erreichte Folgezustand; $a$ gibt die Aktion an, die beim Zustandswechsel ausgeführt werden soll. Initialer Zustand ist $S = 0$, darin soll LED0 leuchten und alle anderen LED1–7 ausgeschaltet sein.

**Hinweise:**

- Ein '*' im Zustand $S$ bedeutet "jeder mögliche Zustand".
- In der Mehrzahl der Fälle wird es sinnvoll sein, die Aktion $a$ vor dem Verlassen von $S$ auszuführen — im Zustand $S^+$ müßten Sie sich merken, aus welchem Zustand $S$ das Programm gekommen ist.
- Beachten Sie, daß gewöhnliche Spulen ( ) auch in inaktiven Zuständen ausgeführt würden; verwenden Sie daher hier (S) und (R).

**Beispiel I.c.0:**

| $S$ | Ereignis $e$ | $S^+$ | Aktion $a$ |
|---|---|---|---|
| 0 | Swt0 aus und Swt3 ein | 3 | LED0 aus, LED3 ein |
| 3 | Swt3 aus und Swt0 ein | 0 | LED3 aus, LED0 ein |
| 0 | Swt0 ein und Swt3 ein | 4 | LED0 und LED3 mit 1 Hertz blinken |
| 4 | Swt0 aus und Swt3 ein | 3 | LED0 aus, LED3 ein |
| 4 | Swt0 ein und Swt3 aus | 0 | LED0 ein, LED3 aus |

**Beispiel I.c.1:**

| $S$ | Ereignis $e$ | $S^+$ | Aktion $a$ |
|---|---|---|---|
| 0 | Swt1 ein | 1 | LED0 aus, LED1 ein |
| 1 | Swt2 ein und Swt0 aus | 2 | LED1 aus, LED2 ein |
| 1 | Swt2 aus und Swt0 ein | 0 | LED1 aus, LED0 ein |
| 2 | Swt3 ein und Swt1 aus | 3 | LED2 aus, LED3 ein |
| 2 | Swt3 aus und Swt1 ein | 1 | LED2 aus, LED1 ein |
| 3 | Swt4 ein und Swt2 aus | 4 | LED3 aus, LED4 mit 1 Hertz blinken |
| 3 | Swt4 aus und Swt2 ein | 2 | LED3 aus, LED2 ein |

**Beispiel I.c.2:**

| $S$ | Ereignis $e$ | $S^+$ | Aktion $a$ |
|---|---|---|---|
| 0 | Swt1 ein | 1 | LED1 ein |
| 1 | Swt2 ein und Swt0 aus | 2 | LED2 ein |
| 1 | Swt2 aus und Swt0 ein | 0 | LED1 aus |
| 2 | Swt3 ein und Swt1 aus | 3 | LED3 ein |
| 2 | Swt3 aus und Swt1 ein | 1 | LED2 aus |
| 3 | Swt4 ein und Swt2 aus | 4 | LED0–4 mit 1 Hertz blinken |
| 3 | Swt4 aus und Swt2 ein | 2 | LED3 aus |

**Beispiel I.c.3:**

| $S$ | Ereignis $e$ | $S^+$ | Aktion $a$ |
|---|---|---|---|
| 0 | Swt1 Wechsel aus → ein | 1 | LED1 ein |
| 1 | Swt1 Wechsel aus → ein | 2 | LED2 ein |
| 1 | Swt7 Wechsel aus → ein | 0 | LED1 aus |
| 2 | Swt1 Wechsel aus → ein | 3 | LED3 ein |
| 2 | Swt7 Wechsel aus → ein | 1 | LED2 aus |
| 3 | Swt1 Wechsel aus → ein | 4 | LED0–4 mit 1 Hertz blinken |
| 3 | Swt7 Wechsel aus → ein | 2 | LED3 aus |
| $*$ | Swt6 ein | 0 | LED0 ein, LED1–7 aus |

| **Beispiel I.c.4:** | | | |
|---|---|---|---|
| $S$ | Ereignis $e$ | $S^+$ | Aktion $a$ |
| 0 | Swt0 aus | 4 | LED4 ein |
| 0 | Swt1 ein | 1 | LED1 ein |
| 4 | Swt1 ein | 1 | LED1 ein |
| 1 | Swt1 aus | 5 | LED5 ein |
| 1 | Swt2 ein | 2 | LED2 ein |
| 5 | Swt2 ein | 2 | LED2 ein |
| 2 | Swt2 aus | 6 | LED6 ein |
| 2 | Swt3 ein | 3 | LED3 ein |
| 6 | Swt3 ein | 3 | LED3 ein |
| 3 | Swt3 aus | 7 | LED7 mit 1 Hertz blinken |

**Hinweis für die Verwendung von steigenden Flanken in Ablaufsteuerungen:**

- Problem: n Zustände einer Ablaufsteuerung sollen mit einem Schalter Swt1 in aufsteigender, mit einem Schalter Swt2 in absteigender Richtung durchgeschaltet werden (jeweils mit steigender Flanke). Die naheliegende Realisierung sieht man in Abbildung 1 links.

  Dabei tritt folgendes Problem auf (wir starten in SCR 0): jede Betätigung (und wieder Wegnehmen) von Swt1 schaltet in den nächsten Zustand. Wird in SCR $i$ aber Swt2 betätigt, so geht das Programm (durch alle Zustände durchlaufend) sofort wieder in SCR 0. Wird jetzt wieder Swt1 betätigt, so werden sofort alle Zustände bis SCR $i+1$ durchlaufen.

- Begründung: Im folgenden bedeutet $|P_{\mathsf{Swt1}}|_k$ den aktuellen Signalzustand für das $|P|$ in SCR $k$, das von Schalter Swt1 getriggert wird, und $|P'_{\mathsf{Swt1}}|_k$ den alten gemerkten Zustand für dieses $|P|$.

  Die SPS merkt sich für jedes $|P|$ den alten Zustand des Signals. Dieser wird aber innerhalb einer Ablaufsteuerung nur aktualisiert, wenn die jeweilige SCR aktiv ist. Wird das Netzwerk mit dem $|P|$ exekutiert, dann wird mit dem alten Zustand verglichen und das SCRT nur dann ausgeführt, wenn der alte Zustand 0 und der aktuelle 1 ist. Den neuen Zustand merkt sich die SPS wieder.

  Daher passiert in obigem Beispiel folgendes: Wir starten mit dem Defaultzustand 1 für alle $|P|$'s, also $|P'_{\mathsf{Swt1}}|_k = |P'_{\mathsf{Swt2}}|_k = 1$. Wird SCR $k$ aktiv und ist Swt1 nicht aktiv, so wird $|P'_{\mathsf{Swt1}}|_k = 0$ gesetzt. Mit dem Wechsel von Swt1 wird $|P_{\mathsf{Swt1}}|_k = 1$, eine Flanke wird erkannt und SCRT $k+1$ ausgeführt. Für $|P'_{\mathsf{Swt1}}|_k$ wird der Zustand 1 gemerkt. In SCR $k+1$ ist $|P'_{\mathsf{Swt1}}|_{k+1} = |P_{\mathsf{Swt1}}|_{k+1} = 1$ (keine Flanke), das Programm bleibt also in diesem Zustand stehen. Daher ist für alle $0 \leq i \leq k+1$ beim Raufschalten $|P'_{\mathsf{Swt1}}|_i = 1$, wenn wir im Zustand SCR $k+1$ sind. Weiters ist $|P'_{\mathsf{Swt2}}|_i = 0$.

  Wird nun Swt2 in SCR $k+1$ betätigt, dann geht $|P_{\mathsf{Swt2}}|_{k+1}$ auf 1 und wir schalten in den vorigen Zustand. Weiters wird $|P'_{\mathsf{Swt2}}|_{k+1} = 1$ gesetzt. War in SCR $k+1$ Swt1 nicht aktiv, dann ist $|P'_{Swt1}|_{k+1} = 0$. Wir stehen nun in SCR $k$, und da Swt1 immer noch nicht aktiv ist, wird nun auch $|P'_{\mathsf{Swt1}}|_k = 0$ gesetzt. Swt2 ist immer noch aktiv (da wir den Schalter nicht innerhalb eines Zyklus wieder wegnehmen können), daher ist nun $|P_{\mathsf{Swt2}}|_k$ auf 1. Da $|P'_{\mathsf{Swt2}}|_k = 0$ war, wird ein Flankenwechsel detektiert und es wird sofort in SCR $k-1$ gesprungen sowie $|P'_{\mathsf{Swt2}}|_k = 1$ gesetzt. Das geht weiter bis wir wieder bei SCR 0

sind. In diesem Moment ist $|P'_{\mathsf{Swt1}}|_i = 0$ für alle $0 \leq i \leq k+1$ und $|P'_{\mathsf{Swt2}}|_i = 1$ für alle $0 \leq i \leq k+1$. Wenn wir nun endlich Swt2 wieder zurücknehmen, wird $|P'_{\mathsf{Swt2}}|_0 = 0$.

Wenn wir jetzt Swt1 betätigen, passiert dasselbe wie vorher mit Swt2: da $|P'_{\mathsf{Swt1}}|_i = 0$ ist und Swt1 aktiv ist, wird eine Flanke detektiert, $|P'_{\mathsf{Swt1}}|_i = 1$ gesetzt und in SCR $i+1$ gesprungen. Das endet erst im Zustand SCR $k+2$, in dem erstmals $|P'_{\mathsf{Swt1}}|_{k+2} = 1$ ist und keine Flanke mehr erkannt wird. Dort bleibt das Programm also stehen und von dort kann man wieder normal weiter raufschalten.

Wenn man in SCR $k+1$ Swt1 aktiv läßt und Swt2 betätigt (was bedeutet, daß $|P'_{\mathsf{Swt1}}|_i = 1$ bleibt für alle $0 \leq i \leq k+1$, dann kann man auch von SCR 0 wieder normal in SCR 1 schalten. Umgekehrt, wenn man beim Raufschalten Swt2 aktiv läßt (und daher $|P'_{\mathsf{Swt2}}|_i = 1$ wird), dann kann man auch normal von SCR $k+1$ auf SCR $k$ runterschalten.

- Lösungsvorschlag: Wenn man in mehreren Teilen einer Ablaufkette auf dieselbe Flanke eines Eingangs reagieren muß, geht man beispielsweise folgendermaßen vor:

  Zu Beginn (außerhalb der Ablaufsteuerung) speichert man in einem separaten Netzwerk die Zustandsänderung des Eingangs unter Zuhilfenahme eines Merkers.

  In der Schrittkette überprüft man dann den Zustand des Merkers (der aktiv ist, wenn eine Flanke am Eingang passiert ist). Ist das Merkerbit 1 verzweigt man in den nächsten Teil der Ablaufsteuerung und setzt den entsprechenden Merker rück (vgl. Abbildung 1 rechts).

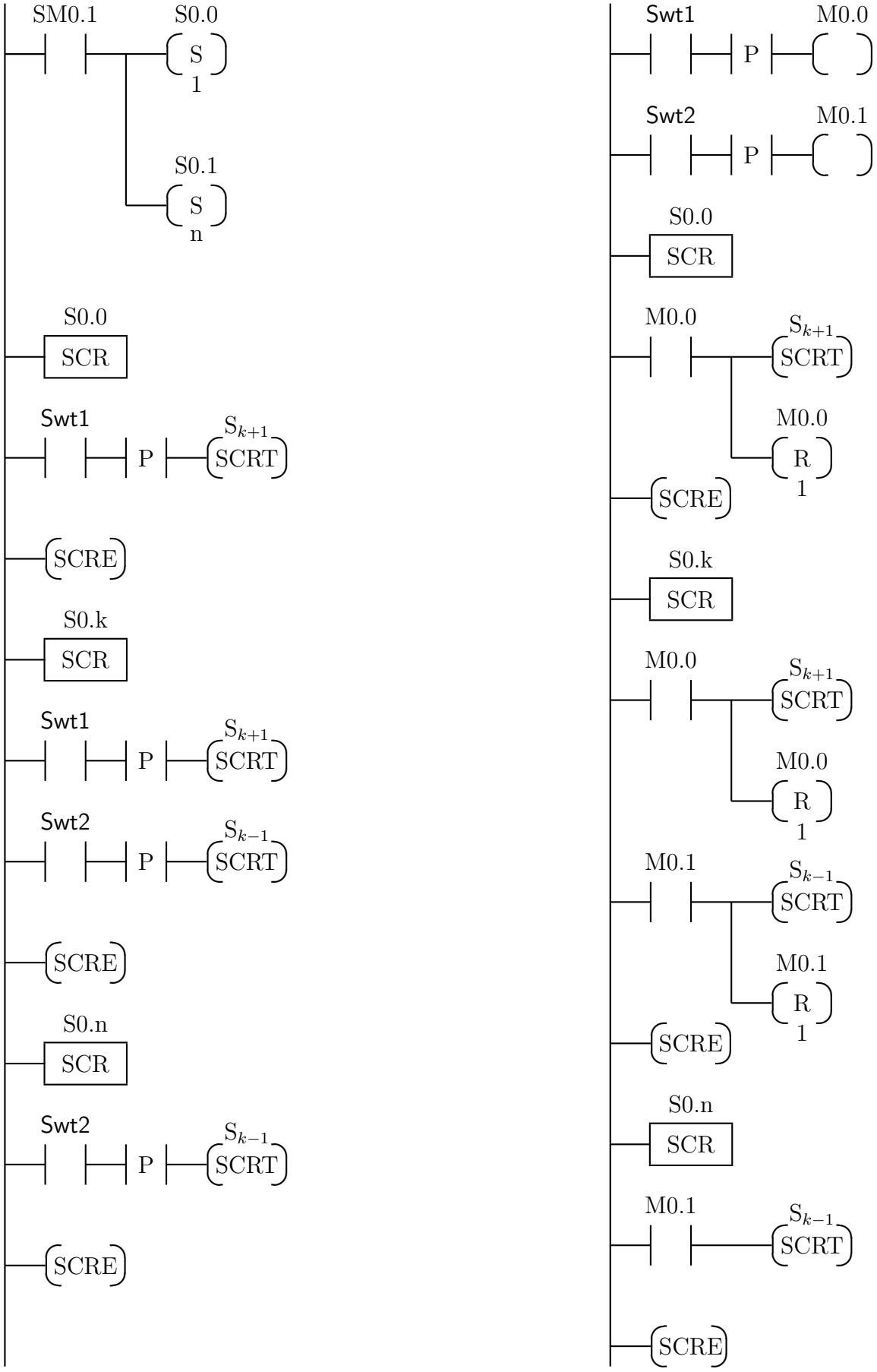


Abbildung 1: Probleme bei steigenden Flanken in Ablaufsteuerungen