FHTW
Fachhochschule
für Technik und Wirtschaft Berlin
-University of Applied Sciences-

UMWELT-
VERFAHRENS-
TECHNIK

Labor Mechatronik Prozessautomatisierung

Versuch V 3

- Programmieren mit SIMATIC STEP 7 –

Inhalt:

1.0 Zielstellung

Die Zielstellung des Versuches besteht im Kennenlernen der Programmiersoftware SIMATIC STEP 7. Weiterhin soll das Arbeiten mit mehreren für die Steuerungstechnik relevanten Verknüpfungsglieder, Speicherfunktionen, Zeiten, Zähler usw. an Beispielen vertieft werden. Für diese Beispiele werden Automatisierungslösungen realisiert.

2.0 Theoretische Grundlagen

STEP 7 ist die Basisprogrammier- und Projektiersoftware für die SIMATIC. Sie setzt sich aus einer Reihe von Applikationen zusammen, mit denen Teillösungen komfortabel realisiert werden können. Mit ihrer Hilfe können Hardware parametriert und konfiguriert, Anwenderprogramme erstellt und getestet und Netzwerke und Verbindungen konfiguriert werden. Es werden unter der grafischen Bedienoberfläche Objekte mit Symbolen dargestellt. Ein Symbol ist einem bestimmten Objekt zugeordnet.
Weiterhin bietet STEP 7 die Möglichkeit, eine Anlage in Projekte zu gliedern. Die Daten für eine Automatisierungslösung werden in einem Projekt (siehe Abb.1) verwaltet. In diesem Projekt liegt die gesamte Automatisierungslösung. Das Projekt enthält z.B. die Station S7-300 sowie das Automatisierungsgerät mit der CPU 313 (Zentralbaugruppe). In dieses Automatisierungsgerät wird das S7-Programm

eingegeben. Das Programm gliedert sich dann in die entsprechenden Bausteine wie z.B. in OB 1 und FC 1.

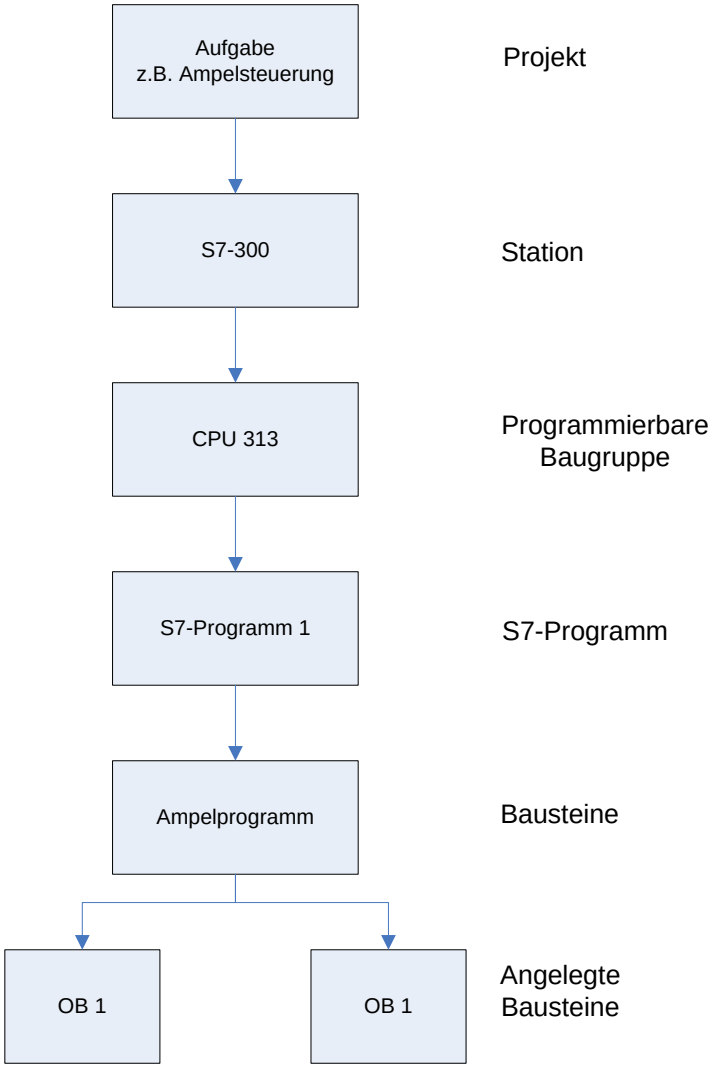

Abbildung 1: Projektstruktur mit SIMATIC S7

Es gibt verschiedene Bausteinarten, die innerhalb eines S7-Anwenderprogramms verwendet werden können:

Organisationsbausteine (OB)

Systemfunktionsbausteine (SFB) und Systemfunktionen (SFC)

Funktionsbausteine (FB)

Funktionen (FC)

Instanz-Datenbausteine (Instanz-DB)

Datenbausteine

Organisationsbausteine legen die Struktur des Anwenderprogramms fest.

Organisationsbausteine bilden die Schnittstelle dem Betriebssystem und dem

Anwenderprogramm. Sie werden von dem Betriebssystem aufgerufen und die zyklische und alarmgesteuerte Programmbearbeitung, das Anlaufverhalten des Automatisierungssystems und die Behandlung von Fehlern. Es können Organisationsbausteine programmiert und so das verhalten der CPU bestimmt werden. Organisationsbausteine bestimmen die Reihenfolge (Startereignisse), in der die einzelnen Programmteile bearbeitet werden.
Funktionen gehören zu den Bausteinen, die selbst programmiert werden können. Eine Funktion ist ein Code-Baustein „ohne Gedächtnis“. Temporäre Variablen der Funktion werden im Lokaldaten-Stack gespeichert. Diese Daten gehen nach der Bearbeitung der FC verloren. Weil eine Funktion keinen zugeordneten Speicher hat, müssen immer Aktualparameter für eine FC angegeben werden. Eine Funktion enthält ein Programm, das immer dann ausgeführt wird, wenn die Funktion von einem anderen Codebaustein aufgerufen wird. Mit einer der zur Verfügung stehenden Programmiersprachen (KOP, FUP, AWL) wird das Programm für die Funktion erstellt.
Der Kontaktplan (KOP) ist eine grafische Programmiersprache. Die Syntax der Anweisungen ähnelt einem Stromlaufplan. KOP ermöglicht eine einfache Verfolgung des Signalflusses zwischen Stromschienen über Kontakte, komplexe Elemente und Spulen.
Die Anweisungsliste (AWL) ist eine textuelle, maschinennahe Programmiersprache. Wird ein Programm in AWL programmiert, so entsprechen die einzelnen Anweisungen weitgehend den Arbeitsschritten, mit denen die CPU das Programm bearbeitet.
Der Funktionsplan (FUP) ist eine grafische Programmiersprache und benutzt zur Darstellung der Logik die von der Booleschen Algebra bekannten logischen Boxen. Außerdem können komplexe Funktionen (z.B. mathematische Funktionen) direkt in Verbindung mit den logischen Boxen dargestellt werden.

Um die Programmierung eines Funktionsplanes zu realisieren ist es nötig hier ein paar grundlegende Operationen zu erläutern. Diese beschränken sich jedoch auf die später zur Lösung der Aufgabenstellung genutzten Operationen.

UND-Verknüpfung

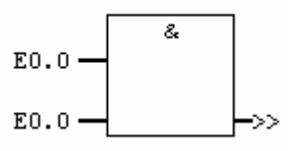

Abbildung 2: Symbol UND-Verknüpfung

Mit der Operation UND können die Signalzustände zweier oder mehr angegebener Operanden an den Eingängen einer UND–Box abfragt werden. Beträgt der Signalzustand aller Operanden "1", so ist die Bedingung erfüllt und die Operation liefert das Ergebnis "1". Beträgt der Signalzustand eines Operanden "0", ist die Bedingung nicht erfüllt und die Operation erzeugt das Ergebnis "0".

ODER-Verknüpfung

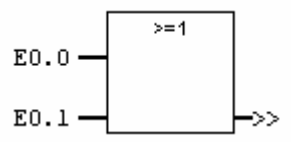

Abbildung 3: Symbol ODER-Verknüpfung

Mit der Operation ODER können die Signalzustände zweier oder mehr angegebener Operanden an den Eingängen einer ODER–Box abfragt werden. Beträgt der Signalzustand eines der Operanden "1", so ist die Bedingung erfüllt und die Operation liefert das Ergebnis "1". Beträgt der Signalzustand aller Operanden "0", ist die Bedingung nicht erfüllt und die Operation erzeugt das Ergebnis "0".

Flipflop (setzen-rücksetzen)

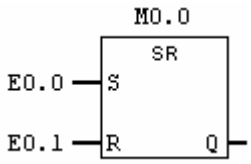

Abbildung 4: Symbol Flipflop

Die Operation Flipflop setzen rücksetzen führt Operationen wie Setzen (S) oder Rücksetzen (R) nur dann aus, wenn das VKE = 1 ist. Ein VKE (Verknüpfungsergebnis) von "0" beeinflusst diese Operationen nicht; der in der Operation angegebene Operand wird nicht verändert. Flipflop setzen rücksetzen wird

gesetzt, wenn der Signalzustand an Eingang S = 1 und an Eingang R = 0 ist. Ist Eingang S = 0 und Eingang R = 1, wird das Flipflop zurückgesetzt. Ist das VKE an beiden Eingängen "1", so wird das Flipflop zurückgesetzt.

Zeit (als Einschaltverzögerung parametrieren und starten)

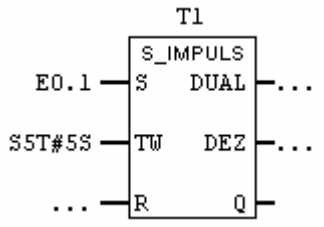

Abbildung 5: Symbol Zeit

Die Operation Zeit als Einschaltverzögerung parametrieren und starten startet die angegebene Zeit, wenn der Starteingang (S) eine steigende Flanke aufweist (d.h. wenn der Signalzustand von "0" auf "1" wechselt). Es ist immer ein Signalwechsel erforderlich, um die Zeit freizugeben. Die Zeit läuft mit dem Wert weiter, der an Eingang TW angegebenen ist, solange der Signalzustand an Eingang S = 1 ist. Eine Signalzustandsabfrage nach "1" an Ausgang Q ergibt "1", wenn die Zeit abgelaufen ist, Eingang S noch immer "1" ist und Rücksetzeingang (R) "0" bleibt. Wechselt der Signalzustand an Eingang S von "1" auf "0", während die Zeit läuft, wird sie angehalten. In diesem Fall ergibt eine Signalzustandsabfrage nach "1" immer "0". Die Zeit wird zurückgesetzt, wenn der Rücksetzeingang (R) von "0" auf "1" wechselt, während die Zeit läuft. Durch diesen Wechsel werden auch der Zeitwert und die Zeitbasis auf Null zurückgesetzt. Die Zeit wird auch dann zurückgesetzt, wenn R = 1 ist, während die Zeit nicht läuft. Der aktuelle Zeitwert kann an den Ausgängen DUAL und DEZ abgefragt werden. Der Zeitwert an Ausgang DUAL ist binär-codiert, der Zeitwert an Ausgang DEZ ist BCD-codiert.

Zähler (parametrieren und vorwärts-/rückwärtszählen)

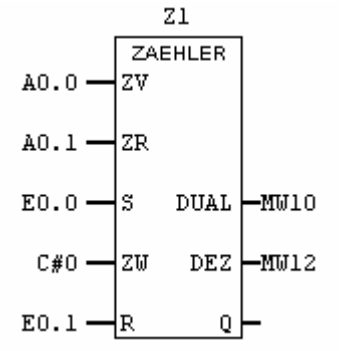

Abbildung 6: Symbol Zähler

Durch einen Flankenwechsel von "0" auf "1" am Eingang S der Operation Parametrieren und vorwärts–/rückwärtszählen wird der Zähler mit dem Zählwert ZW vorbesetzt. Der Wert des Zählers wird bei steigender Flanke am Eingang ZV um "1" erhöht, wenn der Zählwert kleiner als 999 ist. Der Wert des Zählers wird bei steigender Flanke am Eingang ZR um "1" vermindert, wenn der Zählwert größer als "0" ist. Haben beide Zähleingänge eine steigende Flanke, werden beide Operationen bearbeitet und der Zählwert bleibt unverändert. Wird der Zähler gesetzt und ist an den Eingängen ZV/ZR das VKE = 1, so zählt der Zähler entsprechend im nächsten Zyklus, auch wenn kein Flankenwechsel gegeben war. Der Zähler wird zurückgesetzt, wenn am Eingang R eine 1 anliegt. Das Rücksetzen des Zählers setzt den Zählwert auf "0". Eine Signalzustandsabfrage nach "1" an Ausgang Q ergibt "1", wenn der Zählwert größer als "0" ist. Die Abfrage ergibt "0", wenn der Zählwert gleich "0" ist.

Vergleicher (CMP/ ganze Zahlen vergleichen)

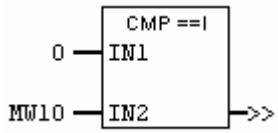

Abbildung 7: Symbol Vergleicher

Die Operation Ganze Zahlen vergleichen (16 Bit) führt eine Vergleichsoperation auf Basis 16 Bit-Festpunktzahlen aus. Verglichen werden die Eingänge IN1 und IN2 entsprechend der Vergleichsart, die ausgewählt wurde.

3.0 Versuchsdurchführung

3.1 Versuchsaufbau

Der Versuchsaufbau ist im Labor durchgeführt. Ziel ist es, die Programme für den Prozessteilbereich Ofentür und den Prozess Ampelsteuerung oder den Prozess Tiefgarage zu projektieren und zur Funktion zu beringen. Die Prozessabläufe werden im Folgenden dargestellt.

3.2 Aufgabenstellung und Versuchsdurchführung

Für die folgenden Beispiele sollen Automatisierungslösungen realisiert werden. Wählen Sie geeignete Operanden zur Lösung dieser Aufgaben aus.

3.2.1 Ampelsteuerung

Zur Einarbeitung in die SIMATIC Software soll eine einfache Ampelsteuerung programmiert werden. Dabei soll der Verkehr sowohl für die Fußgänger, als auch für die Autos geregelt werden. Für die Autos sind die Ampeln mit roten, gelben und grünen Lampen für beide Fahrtrichtungen ausgestattet. Für die Fußgänger gelten rote und grüne Lampen. Dazu kommt ein Schalter für die Fußgänger um die Ampel auf grün umzuschalten.

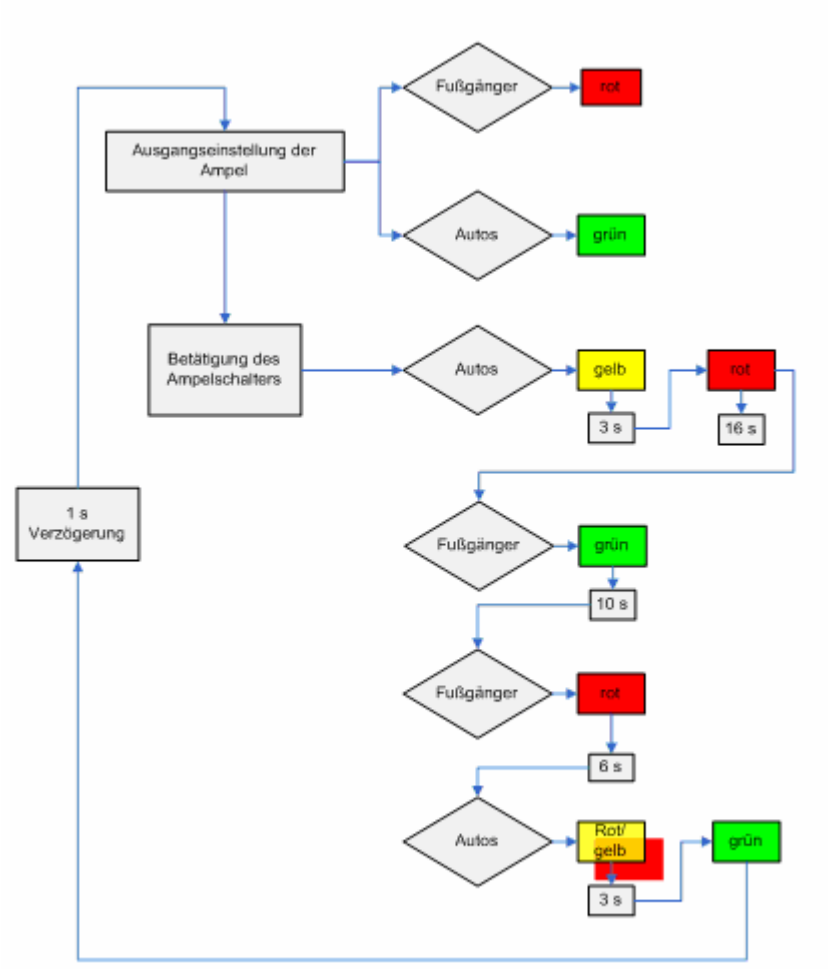

Abbildung 8: Ablauf der Ampelsteuerung

Die Zuordnung der Eingangs- und Ausgangsvariablen ist wie folgt:

Tabelle 1: Bsp. Zuordnung Ampel

Eingangsvariable	**Betriebsmittel-Kennzeichen**	**Zuordnung**	**logische Zuordnung**
Schalter 1	S0	E 4.0	betätigt S0=1
Schalter 2	S1	E 4.1	betätigt S1=1
Ausgangsvariable			
Fußgänger	ROT GRÜN	A 8.0 A 8.1	leuchtet=1 leuchtet=1
Auto	ROT GELB GRÜN	A 8.5 A 8.6 A 8.7	leuchtet=1 leuchtet=1 leuchtet=1

3.2.2 Pufferspeicher

In einer Montagestraße (Transportbandmotor) befindet sich ein Pufferspeicher mit Bildröhren. Der Zu- sowie der Abgang der Einheiten werden über Lichtschranken kontrolliert, deren Impulse einem Zähler zugeführt werden. Steigt der Bestand auf den oberen Grenzwert von 30, dann soll der Transportbandmotor sowie die Lichtschranke abgeschaltet werden. In einer Zeit von 60sec werden dann die Bildröhren in diesem Speicher behandelt. Anschließend werden diese mit dem Transportbandmotor wieder aus dem Speicher hinausbefördert. Nachdem der untere Grenzwert von 0 erreicht ist, wird dies durch eine Meldelampe angezeigt. Die Eichung des Zählers erfolgt nach jedem Vorgang über einen Tastschalter. Der Vorgang erfolgt diskontinuierlich.
Die Zuordnung der Eingangs- und Ausgangsvariablen ist wie folgt:

Tabelle 2: Bsp. Zuordnung Pufferspeicher

Eingangsvariable	**Betriebsmittel-Kennzeichen**	**Zuordnung**	**logische Zuordnung**
Start	S0	E 1.0	betätigt S0=1
Eichen	S1	E 1.1	gedrückt S1=1
Ausgangsvariable			
Motor	M	A 9.0	leuchtet=1
Lichtschranke 1	LI1	A 9.1	leuchtet=1
Lichtschranke 2	LI2	A 9.2	leuchtet=1
Meldeleuchte	H	A 9.3	leuchtet=1

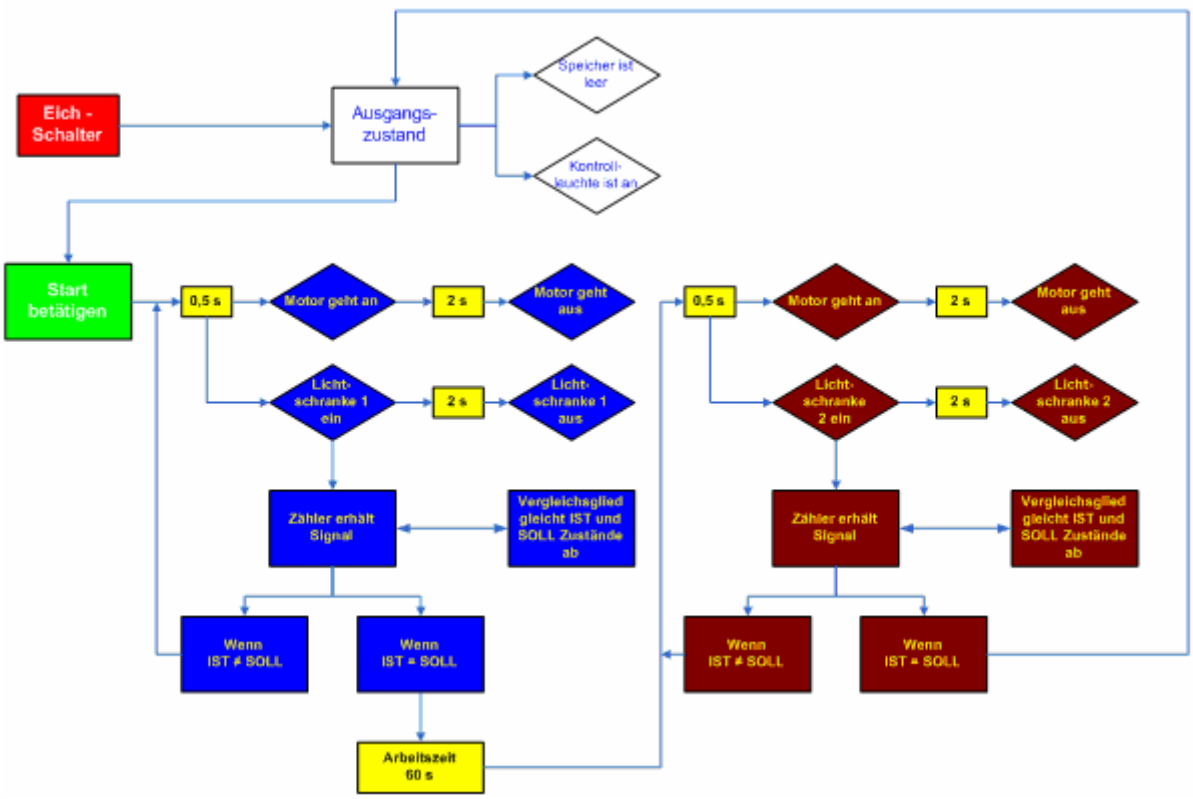

Abbildung 9: Ablauf des Pufferspeichers

Ofentürsteuerung

Eine Ofentür mit den Funktionen „Öffnen, Schließen und Stillstand" wird durch einen Zylinder gesteuert. In der Grundstellung ist die Ofentür geschlossen. Wenn „Öffnen“ gedrückt wird soll die Ofentür innerhalb von 6s öffnen und offen bleiben. Es soll jedoch auch möglich sein die Ofentür vorzeitig zu „schließen“. Der Schließvorgang soll 3s dauern. Der Schließvorgang kann jedoch durch die Funktionen „Lichtschranke“ und „Halt“ unterbrochen werden und wieder öffnen. Dies wird als Störung eingebaut. Nach dem „Schließen“ beginnt der Ausgangszustand wieder.

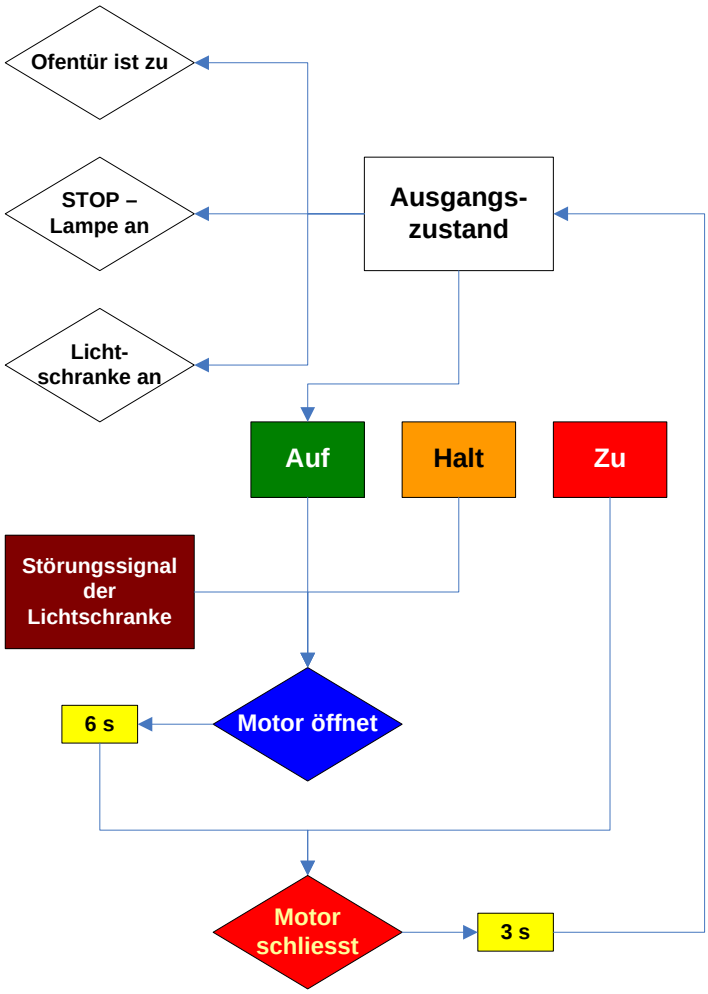

Abbildung 10: Ablauf der Steuerung “Ofentür“

Tabelle 3: Bsp. Zuordnung Ofentür

Eingangsvariable	**Betriebsmittel-Kennzeichen**	**Zuordnung**	**logische Zuordnung**
Öffnen	S1	E 0.0	gedrückt S1=1
Schließen	S2	E 0.1	gedrückt S1=1
Halt	S3	E 0.2	gedrückt S1=0
Lichtschranke	LI	E 0.5	gedrückt S1=0
Ausgangsvariable			
Zylinder "Tür auf"	Y1	A 13.0	leuchtet=1
Zylinder "Tür zu"	Y2	A 13.1	leuchtet=1

4.0 Versuchsprotokoll

Zum Versuch ist ein ausführliches Protokoll anzufertigen. Das Protokoll ist in folgende Teile zu gliedern:

4.1. Allgemeine Angaben:

- Datum des Versuches
- Versuch
- Praktikumsgruppe
- Teilnehmer am Versuch

4.2. Versuchsauswertung

- Für alle Teilversuche sind die in jeweiliger Programmiersprache erstellten Programme darzustellen.
- Die Automatisierungslösungen der einzelnen Teilversuche sind zu dokumentieren und zu diskutieren. Für die vollständige Dokumentation sind die Quellen mit zu liefern. Dazu sind die nach DIN EN 6.1131 – 3 genormten Programmiersprachen zu verwenden.

Hinweise:

- Die Ausdrucke sind als Anlage zum Protokoll mit abzugeben.
- Das Versuchsprotokoll ist von allen Versuchsteilnehmern zu unterschreiben und spätestens eine Woche nach dem Versuch abzugeben.

4.3. Vorbereitungsfragen

1.) Skizzieren Sie den Aufbau eines Steuerrechners SIMATIC S7.
2.) Wie werden Projekte in der SIMATIC S7 gegliedert (Projektstruktur)?
3.) Geben Sie die Aufgaben der OB`s an und diskutieren Sie deren Aufgaben in Hinblick auf die gestellten Automatisierungsaufgaben.
4.) Beschreiben Sie die Unterschiede von FUP, AWL und SCL?
5.) Was sind Funktionsbausteine?
6.) Was sind Datenbausteine?
7.) Welche Möglichkeiten der Darstellung von Daten gibt es in der SPS SIMATIC S7?
8.) Was sind Adressen und welche Möglichkeiten der Adressierung gibt es in der SIMATIC S7?
9.) Welche Aufgabe haben Merker innerhalb eines Programms?

5.0 Literatur

[1] Kaftan, Jürgen: SPS-Grundkurs mit SIMATIC S7, Vogel Buchverlag, Würzburg 1998
[2] Wellenreuther,G.; Zastrow,D.: Automatisieren mit SPS Theorie und Praxis, Vieweg, Braunschweig 2002
[3] Berger, Hans: Automatisieren mit STEP 7 in AWL und SCL, Siemens AG; Publicis Corporate Publishing ; Erlangen 2004
[4] Dokument XSPS_v1.pdf: Nachfrage Laborbetreuer/ Internetseite