Lernfeld 7 - SPS Labor

Lernfeld 7 - SPS Labor

Einführung in die SPS-Programmierung mit dem TIA-Portal 1
Einführungsbeispiel (Schüler:innen) 2
Leitfaden zur Programmerstellung 17
Regeln zur normgerechten SPS-Programmierung 18
Beschriftungvorschriften zur SPS-Programmierung 20
Bausteinauswahl 22
Erstellsprachen 23
Aufgabe des OB1 (Main) 24
Übungsaufgaben: Verknüpfungssteuerungen ohne Speicher 25
Hinweise zur Verwendung von Speichergliedern 26
Übungsaufgaben: Normgerechte Programmierung von VKS mit Speichern 28
Verwendung eines Blinktaktes 29
Hinweise zur Verwendung von Zeiten 30
Übungsaufgaben: Normgerechte Programmierung von VKS mit Speichern und Zeiten 31
Hinweise zur Programmierung von Ablaufsteuerungen 33
Übungsaufgaben: Programmierung von Ablaufsteuerungen 36

Was ist eine SPS?

Bezug zum Theorieunterricht
LF7: LS 10 Aufgaben 5 bis 7

Speicherprogrammierbare Steuerung [1]

Eine speicherprogrammierbare Steuerung (SPS; englisch: programmable logic controller, PLC) ist ein Gerät, das zur Steuerung oder Regelung einer Maschine oder Anlage eingesetzt und auf digitaler Basis programmiert wird. Sie löst die „festverdrahtete“ verbindungsprogrammierte Steuerung in den meisten Bereichen ab.

Hinweis zum Unterrichtsgespräch

Den Schüler:innen sollte bewusst sein, dass die Siemenssteuerungen im Labor und auch das TIA-Portal nur Beispiele eines Herstellers sind. Siemenssteuerungen sind zwar insbesondere auf dem deutschen Markt weit verbreitet, andere Hersteller treten aber auch aufgrund der Preise immer mehr in Konkurrenz. Neben dem TIA-Portal sollte vor allem auch die Software CODESYS erwähnt werden.

Tutorial #2: Einstieg in CoDeSys

CoDeSys ist ein kostenloses Entwicklungssystem für die SPS Programmierung.

Link: https://youtu.be/8VM7qo4WYh4

YouTube-Video

Warum programmieren wir so „umständlich“?

Die Grundlage für die SPS-Programmierung am BSZ Delitzsch ist die Europanorm EN 61131. Wir entnehmen dieser Norm unsere **Regeln zur normgerechten Programmierung**. Den Schüler:innen erscheint diese Form der Programmierung häufig umständlich, da in vielen Betrieben und überbetrieblichen Ausbildungsstätten intuitiv und mit wenigen Regeln programmiert wird. Trotzdem ist es unser Anspruch, den Schüler:innen die Norm nahezubringen. Im Zentrum steht dabei der Aspekt der Bibliotheksfähigkeit der entstehenden Programme.

Die normgerechte Programmierung bringt einige **Vorteile** mit sich:

- Die Strategien zur Programmierung sind softwareunabhängig und können beispielsweise auch auf CoDeSys angewendet werden.
- Die Programme sind hardwareunabhängig und können auf jeder SPS verwendet werden.
- Aufgrund der Bibliotheksfähigkeit können die fertigen Programme auf derselben Steuerung beliebig oft genutzt werden, ohne Änderungen am Programm vornehmen zu müssen. (Siehe Einführungsbeispiel)
- Die Programme sind leichter nachvollziehbar, was sowohl die Fehlersuche als auch die Zusammenarbeit mit anderen Personen vereinfacht.

[1] Quelle: https://de.wikipedia.org/*wiki*/Speicherprogrammierbare_Steuerung Zugrif am 11.04.2024

Um den Einstieg zu erleichtern und die zunächst sehr abstrakten Regeln nachvollziehbarer zu machen, betrachten wir zunächst ein einfaches Beispiel. Hier wird dies anhand der Aufgabe 1 aus den Übungsaufgaben zu den Verknüpfungssteuerungen ohne Speicher vorgeführt.

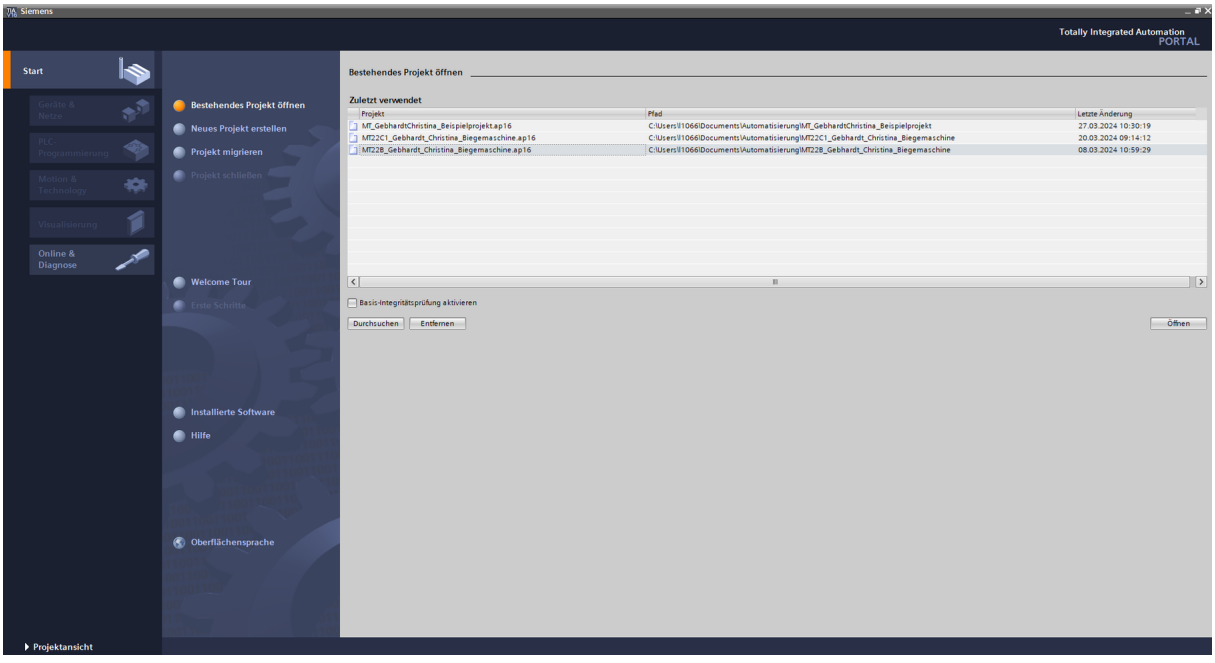
1 **Entladestation Tankwagen**
Für eine Chemieanlage soll die Entladung der Tankwagen, die der Anlage flüssige Rohstoffe liefern, gesteuert werden. Die Entladung kann von zwei Schaltstellen aus über die Taster SF1 und SF2 erfolgen.
Solange genau einer der beiden Taster betätigt ist, wird die Pumpe angesteuert. Wenn der Tank der Anlage den maximalen Füllstand erreicht hat (BG1 = 0), muss die Pumpe sofort stoppen.

TIA-Portal öffnen



Aktuell arbeiten wir mit dem TIA-Portal in der Version 16. Prinzipiell können mit dem TIA-Portal Programme älterer Versionen ebenfalls geöffnet werden. Das TIA-Portal hat eine Desktopverknüpfung auf allen Laborrechnern, kann aber auch über die Suche gefunden werden.

Standardmäßig öffnet sich das TIA-Portal in der **Portalansicht**. In der Portalansicht können vor allem übergeordnete Aufgaben, wie das Hinzufügen von Geräten oder Diagnosen durchgeführt werden. Neben der Portalansicht verfügt das TIA-Portal über die detaillierte **Projektansicht**. Zwischen beiden Ansichten kann jederzeit in der linken unteren Ecke umgeschaltet werden.

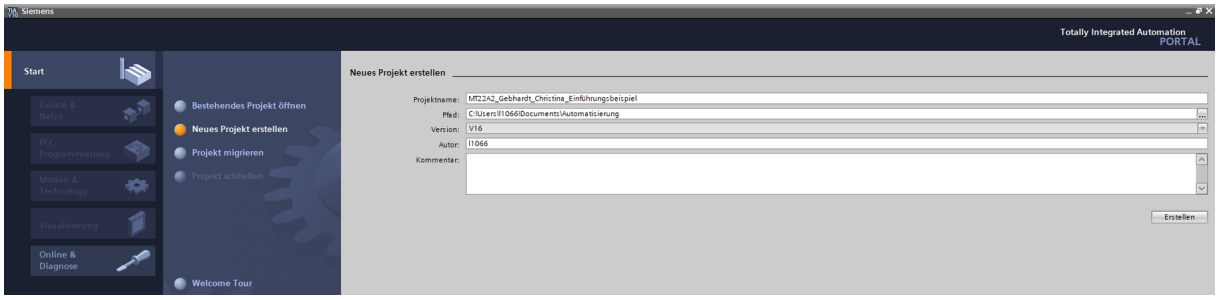


Im Reiter "Start" der Portalansicht können Projekte geöffnet oder angelegt werden.

1. Projekt erstellen und benennen

OPAL Kurs

Unter dem Punkt "Projekt erstellen, kann ein neues Projekt angelegt werden. Die Benennung der Projekte wurde mit dem Muster **Klasse_Name_Vorname_Aufgabenbezeichnung** festgelegt.

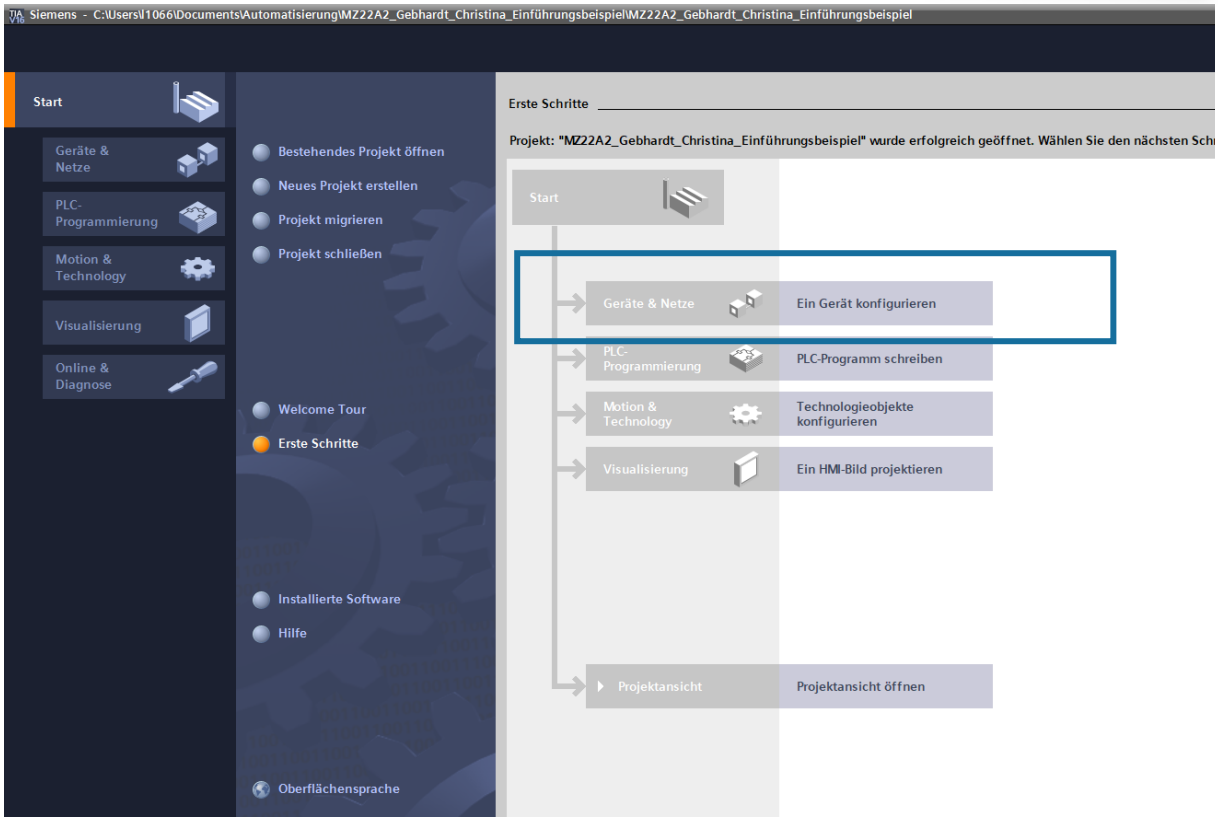


Praxishinweis

Standardmäßig speicher das TIA-Portal die Projekte Lokal in dem Ordner Automatisierung. Es ist empfehlenswert, die Projekte stattdessen im persönlichen Laufwerk zu speichern, damit Sie auch von anderen Rechnern auf Ihre Projekte zugreifen können.

Häufig gibt es im TIA-Portal mehrere Wege. Geräte können auch in der Projektansicht konfiguriert werden.

Nach dem Klick auf "erstellen" öffnet sich in der Portalansicht die Übersicht "Erste Schritte". Beispielsweise von dieser Übersicht gelangt man zum Punkt "**Ein Gerät konfigurieren**", mit dem wir fortfahren.

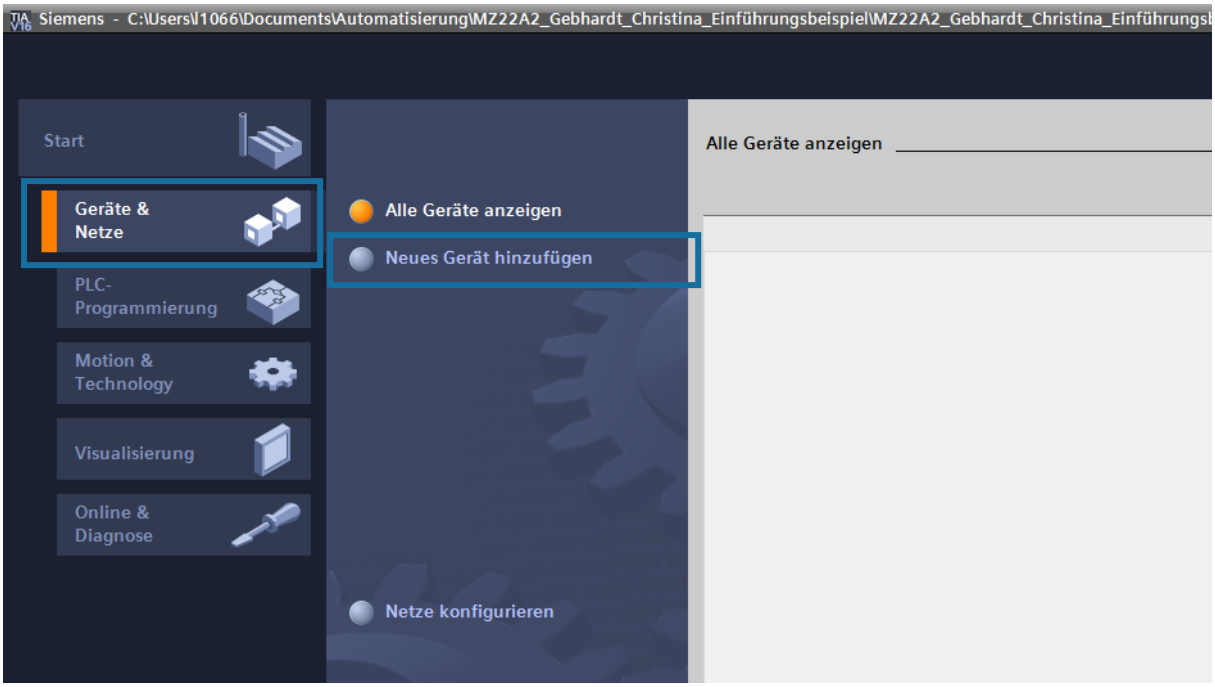


2. Hardware konfigurieren

OPAL Kurs

Bevor mit der eigentlichen Programmierung begonnen wird, wird die Hardware konfiguriert. Das heißt, im TIA-Protal wird das digitale Abbild der tatsächlich verwendeten Hardware zusammengestellt. In unserem Fall ist dies mit sehr wenig Aufwand verbunden, da wir im SPS-Labor mit sogenannten **Compakt-CPU**s arbeiten (1214**C**).

Die CPU kann in der Registerkarte "**Geräte & Netze**" der Portalansicht eingefügt werden.

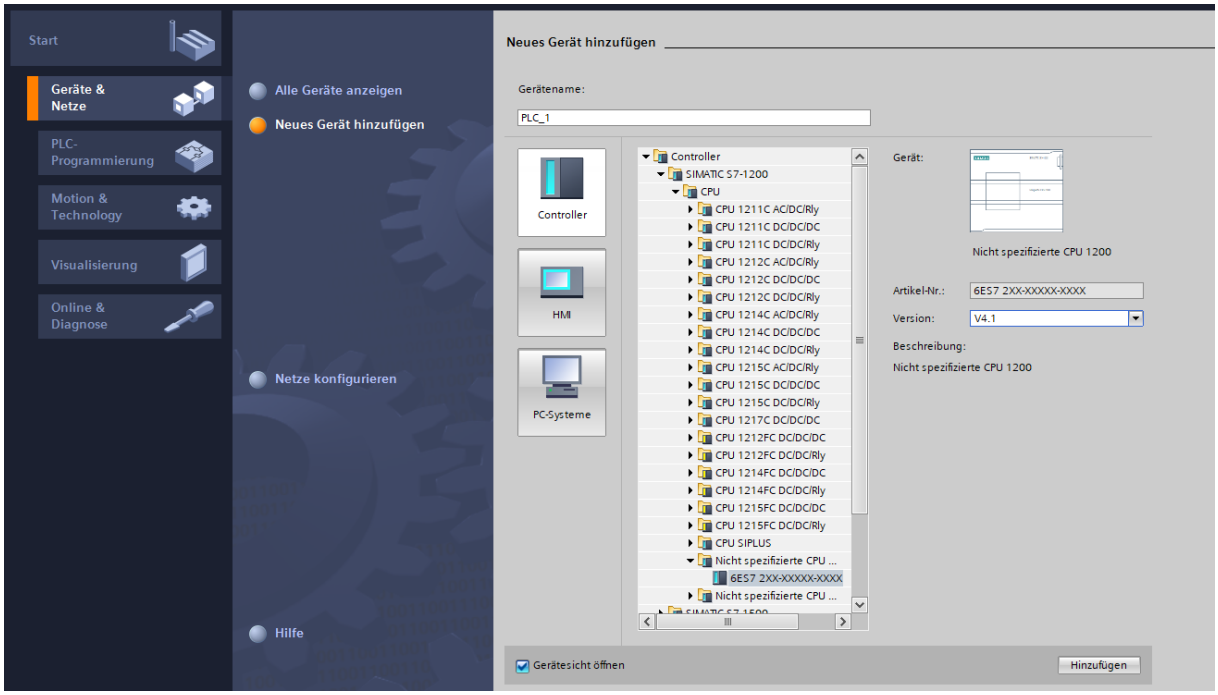


Für das Einfügen der CPU gibt es 2 Möglichkeiten:

- Einlesen der CPU (dauert etwas länger)
- Auswählen der CPU aus dem Katalog (fehleranfälliger, weil beispielsweise die falsche Version ausgewählt werden kann)

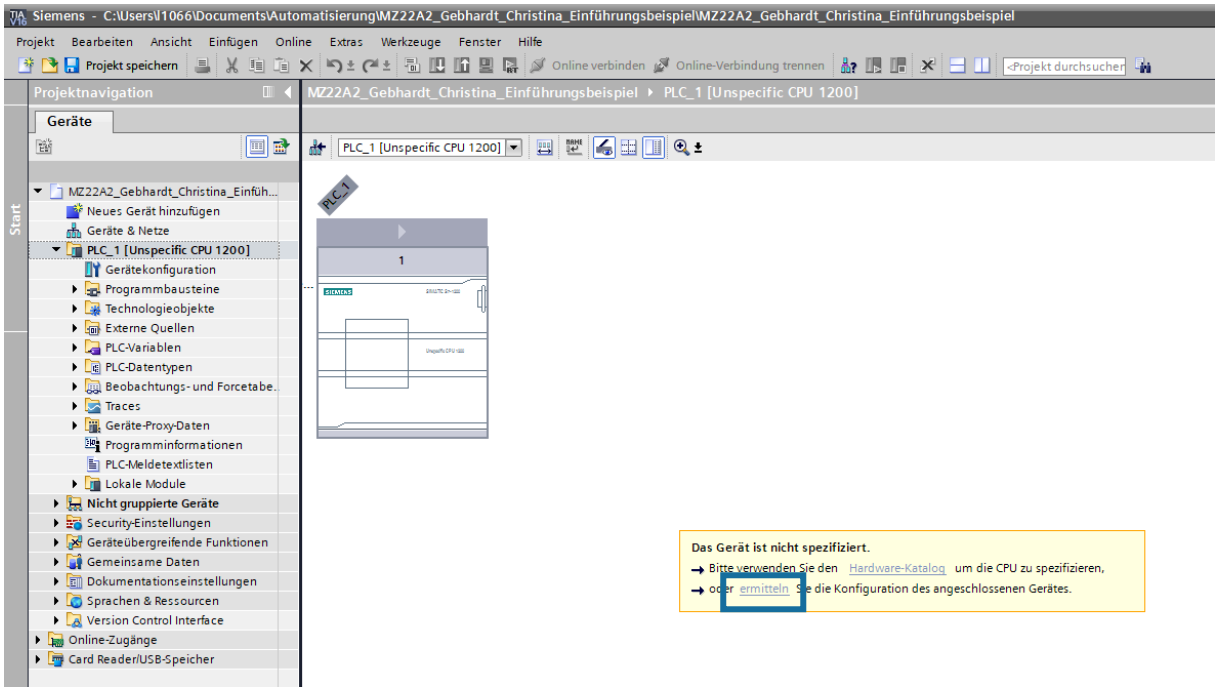
Einlesen der CPU

Um eine CPU einzulesen, muss aus dem Katalog zunächst eine "**Nicht spezifizierte CPU-1200**" ausgewählt werden.



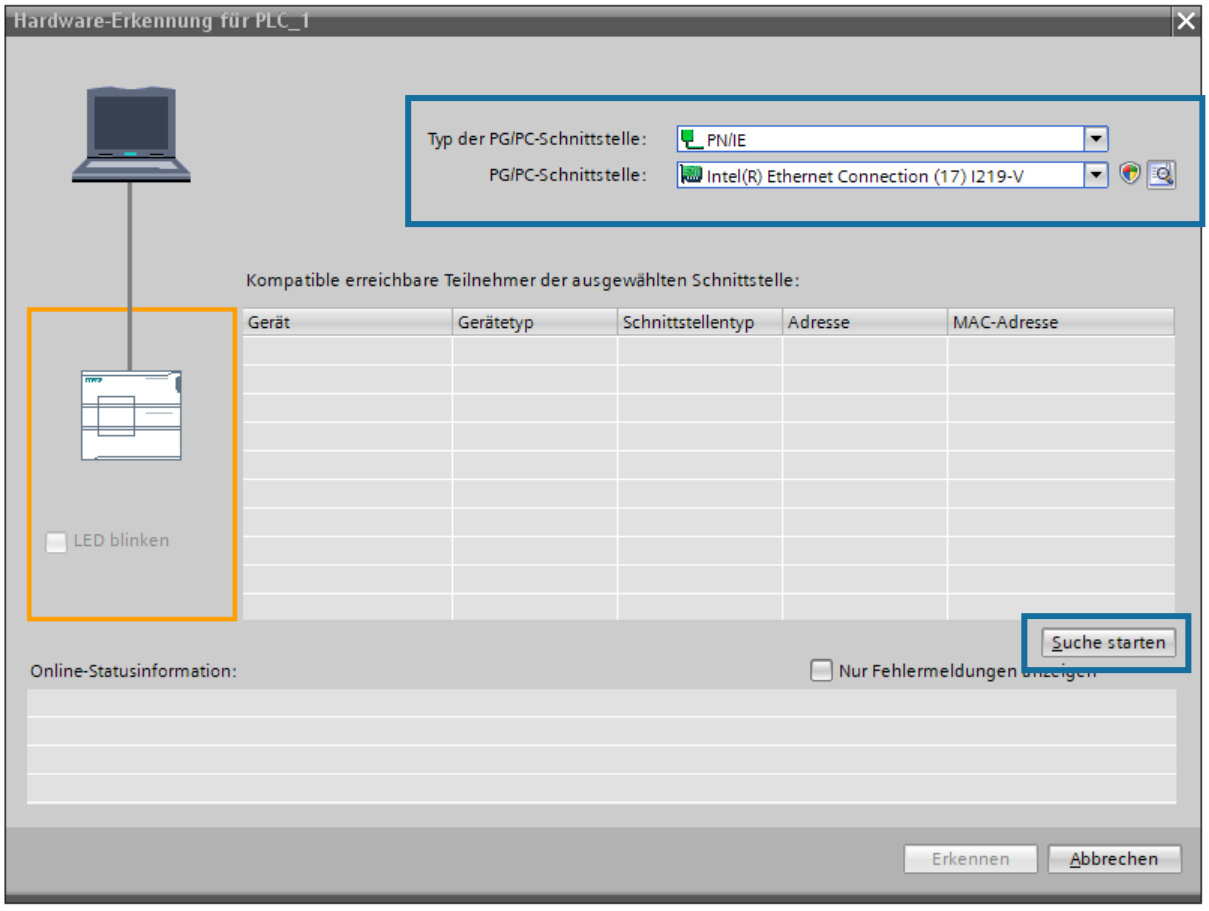
Durch den Klick auf "Hinzufügen" wechselt das TIA-Portal automatisch in die **Projektansicht** und dort in die Ansicht des PLC_1, also der ersten hinzugefügten CPU. Hier erhält man den Hinweis, dass die CPU noch nicht spezifiziert ist und dies über den Katalog geschehen oder die CPU ermittelt werden kann. Für das Auslesen muss "**ermitteln**" angeklickt werden.

PLC:
Programmable
Logic
Controller



Durch den Klick auf ermitteln öffnet sich ein neues Dialogfenster, über das eine Verbindung mit der CPU hergestellt werden kann. Grundvoraussetzung dafür ist natürlich, dass eine SPS am PC angeschlossen ist. Aktuell befindet sich an jedem Arbeitsplatz im SPS-Labor eine CPU.

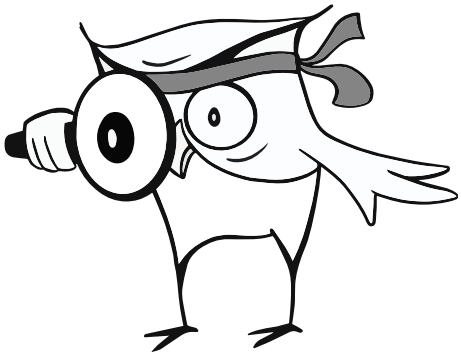
Bei der ersten Verbindung mit der CPU muss zunächst die **Schnittstelle ausgewählt** und anschließend auf "**Suche starten**" geklickt werden.



Praxishinweis - Schnittstelle

Da sowohl die Anbindung an das Schulnetzwerk als auch die Verbindung mit der CPU über **Profinet** eine **RJ45-Schnittstelle** benötigen, verfügen die Rechner im SPS-Labor über **2 Netzwerkkarten**.

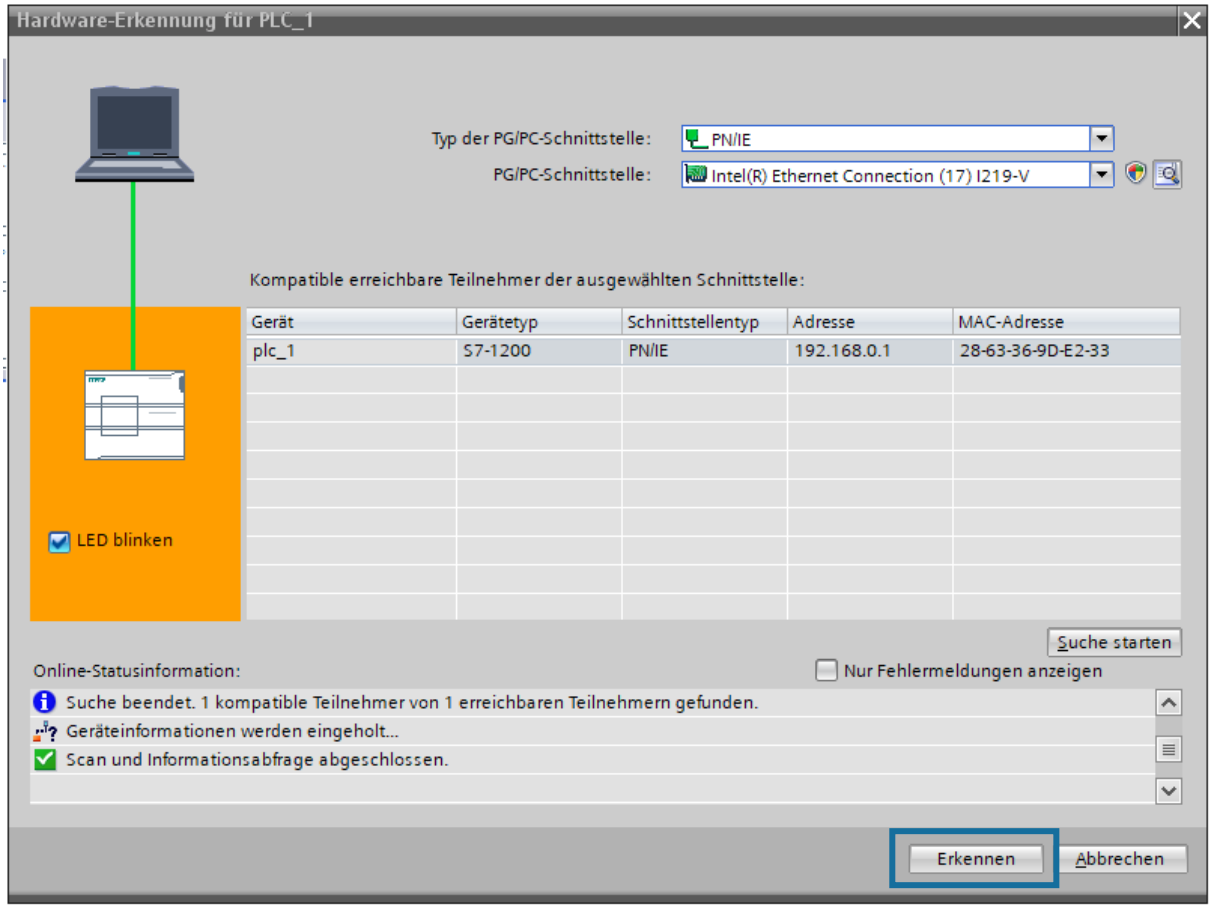
Erhält man bei der Suche die Online-Statusinformation "Gefundenes erreichbares Gerät **rd 120-pc**xx" und "[...] **0 kompatible Teilnehmer** [...] gefunden" hat man über die falsche Netzwerkkarte gesucht.



Praxishinweis - Arbeitsplatz

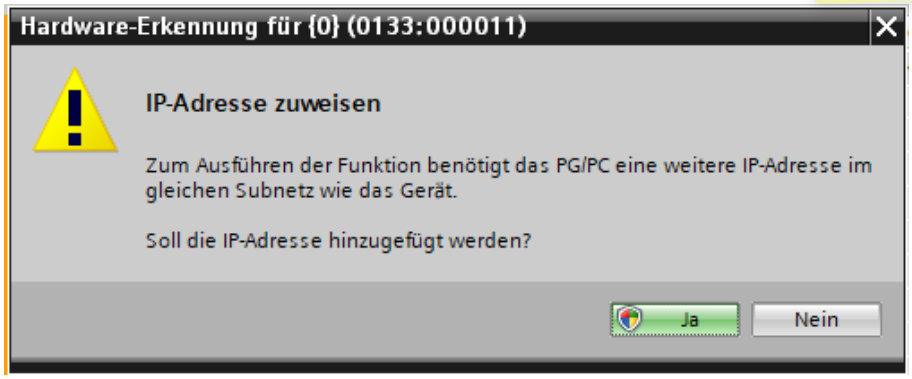
Sollte unklar sein, welche CPU zu welchem Rechner gehört, kann man über "**LED blinken**" die Melde-LEDs der CPU ansteuern.

Wenn die richtige Schnittstelle ausgewählt wurde und eine CPU am PC angeschlossen ist, wird diese als kompatibler Teilnehmer angezeigt. Durch einen Klick auf "**Erkennen**" kann die CPU nun ausgelesen werden.



Um die SPS mit dem PG/PC zu verbinden, müssen beide Geräte über IP-Adressen im gleichen Subnetz verfügen. Da unsere CPUs nicht in einem Netzwerk eingebunden sind, haben sie noch die Standard-IP-Adresse ab Werk. Um das Arbeiten zu vereinfachen, bleibt diese auch bestehen. Die nun erscheinende Meldung "**IP-Adresse zuweisen**" ist also mit "**JA**" zu quittieren.

PG:
Programmier-
gerät

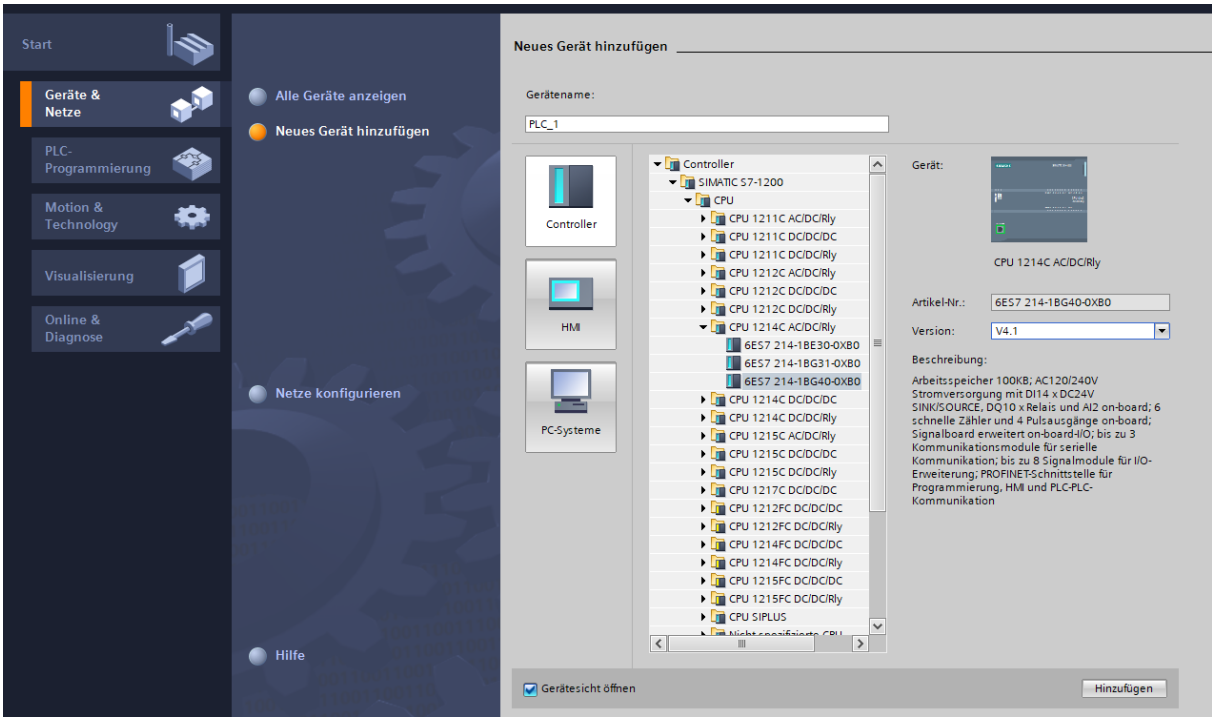


Praxishinweis

Am Ende des Verbindungsaufbaus fragt das TIA-Portal, ob die Einstellungen für die Schnittstelle gespeichert werden sollen. Dies ist zu empfehlen.

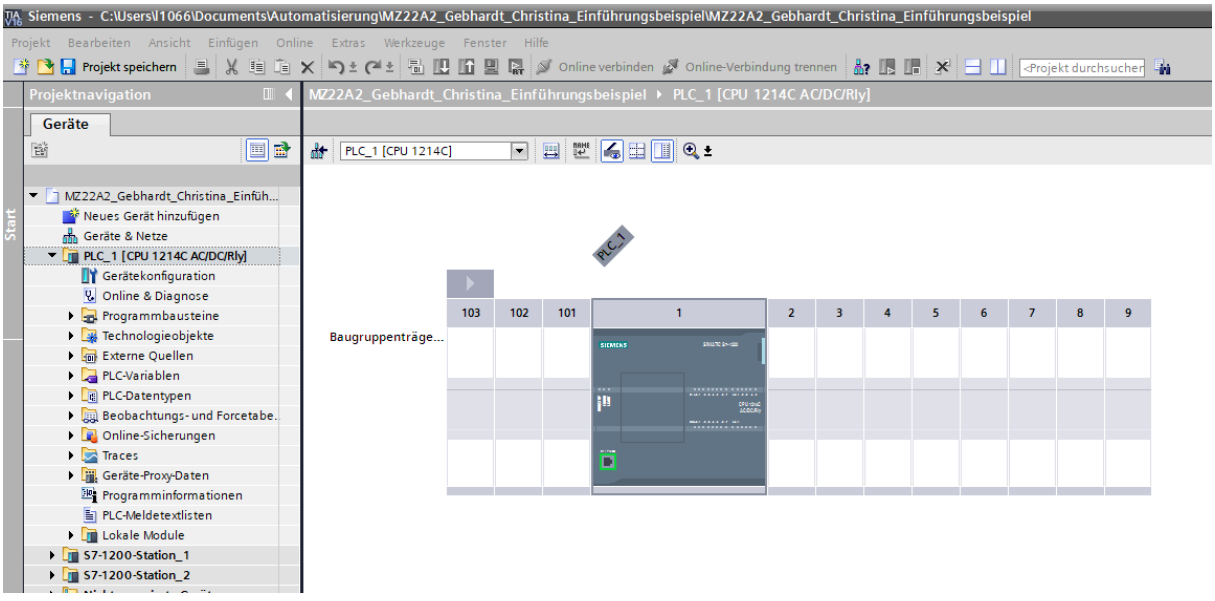
Auswählen der CPU aus dem Katalog

Alternativ zum Einlesen der CPU kann diese auch direkt anhand der **Typenbezeichnung** und der **Bestellnummer** aus dem Katalog ausgewählt werden. Dies kann helfen, wenn nicht an jedem Arbeitsplatz eine funktionierende SPS vorhanden ist. Hierbei ist insbesondere die Auswahl der richtigen Version zu beachten. Die CPUs im SPS-Labor sind alle in der **Version 4.1**.



Durch den Klick auf "Hinzufügen" wechselt das TIA-Portal automatisch in die **Projektansicht** und dort in die Ansicht des PLC_1, also der ersten hinzugefügten CPU.

PLC:
Programmable
Logic
Controller

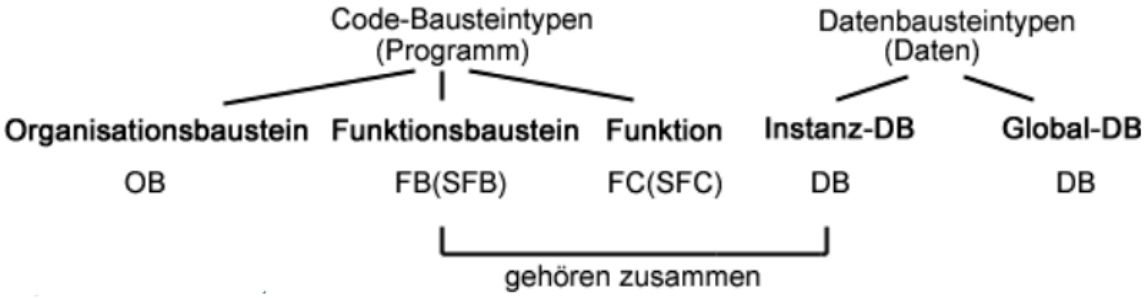


3. Programmbausteine erstellen und benennen

OPAL Kurs

SPS-Programme werden in Bausteine gegliedert. Es gibt 2 grundsätzliche Arten von Bausteinen:

- **Codebausteine** enthalten das Anwenderprogramm oder Aufrufe von anderen Bausteinen
- **Datenbausteine** dienen dem Ablegen und Speichern von Daten



Baustein	Zeichen	Beschreibung
Organisationsbausteine	OB	Diese Bausteine stellen die **Schnittstelle zwischen Anwenderprogramm** und Betriebssystem dar. Sie **reagieren auf Aufrufe aus dem Betriebssystem und können nicht von anderen Bausteinen aufgerufen** werden. Der OB1 (Main) organisiert den zyklischen Programmablauf.
Funktionsbausteine	FB	Ein FB ist ein **Codebaustein**, d.h. er dient der Programmierung des Anwenderprogramms. Zu jedem Aufruf eines Funktionsbaustein gehört ein statischer Speicherbereich (**eigener Instanz-Datenbaustein oder als Multiinstanz**). Dadurch hat dieser Code-Bausteintyp bei jedem Aufruf einen eigenen Speicherbereich und kann somit für **Ablaufsteuerungen** und **Verknüpfungssteuerungen mit Speicher** verwendet werden.
Funktion	FC	Ein FC ist ebenfalls ein **Codebaustein**. Allerdings besitzt er im Gegensatz zum FB kein Gedächtnis. Daher wird er nur für **Verknüpfungssteuerungen ohne Speicher** eingesetzt.
System-funktionen /System-funktionsbausteine	SFC/SFB	SFC und SFB sind **vorgefertigte**, im Betriebssystem der CPU bereits integrierte Code-Bausteine.
Instanzdatenbaustein	DB	Dieser Datenbausteintyp **wird dem Aufruf eines Funktionsbausteins fest zugeordnet** und wird bei der Erstellung eines Funktionsbausteines mit erzeugt.
Globaler Datenbaustein	DB	Dieser Datenbausteintyp **enthält Daten, auf die alle Code-Bausteine des Anwenderprogramms zugreifen können**. Diese Datenbausteine müssen vom Anwender programmiert werden.

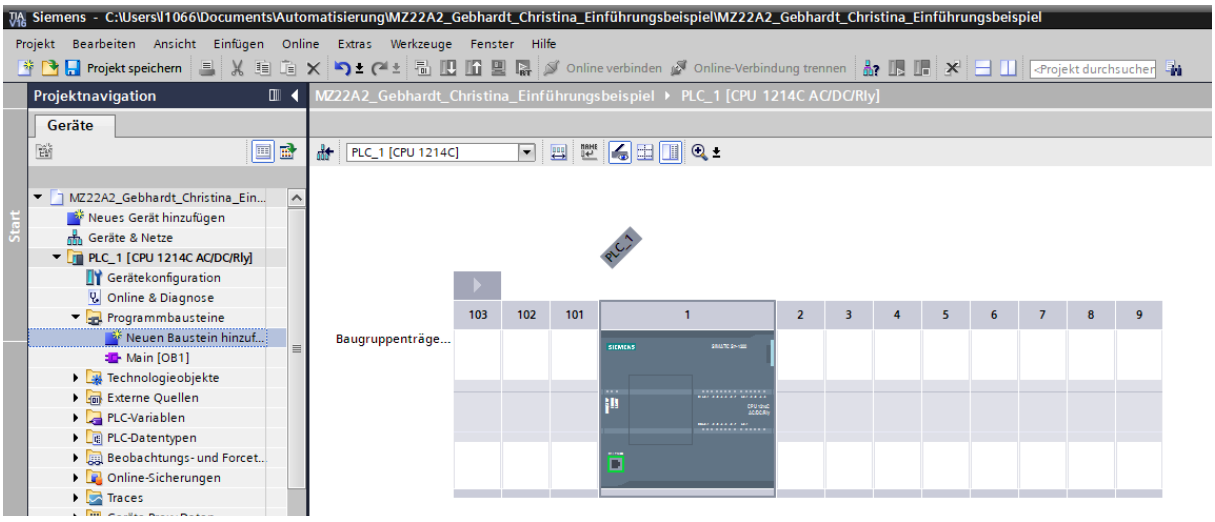
Um den richtigen Baustein für den Programmcode zu wählen, muss zunächst die Aufgabenstellung analysiert werden:

Entladestation Tankwagen
Für eine Chemieanlage soll die Entladung der Tankwagen, die der Anlage flüssige Rohstoffe liefern, gesteuert werden. Die Entladung kann von zwei Schaltstellen aus über die Taster SF1 und SF2 erfolgen.
Solange genau einer der beiden Taster **betätigt ist**, wird die Pumpe angesteuert. Wenn der Tank der Anlage den maximalen Füllstand erreicht hat (BG1 = 0), muss die Pumpe sofort stoppen.

Begriff: Verknüfungssteuerung (aus Arbeitsheft LF7)
Verknüpfungssteuerungen **ordnen** den Signalzuständen der Eingangssignale durch **logische Verknüpfungen** kontinuierlich einen Zustand der Ausgangssignale zu.

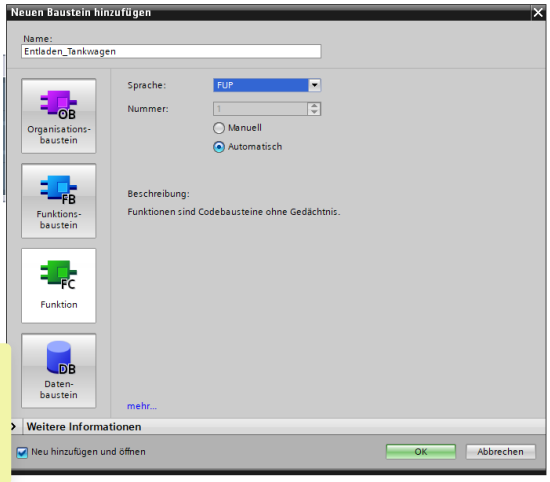
Die beschriebene Steuerung ist als **Verknüpfungssteuerung ohne Speicher** einzuordnen und wird daher in einer **Funktion** programmiert.

Um eine Funktion zu erstellen, klickt man in der **Projektansicht** doppelt in der **Projektnavigation** unter dem Punkt "**Programmbausteine**" auf "**Neuen Baustein hinzufügen**".



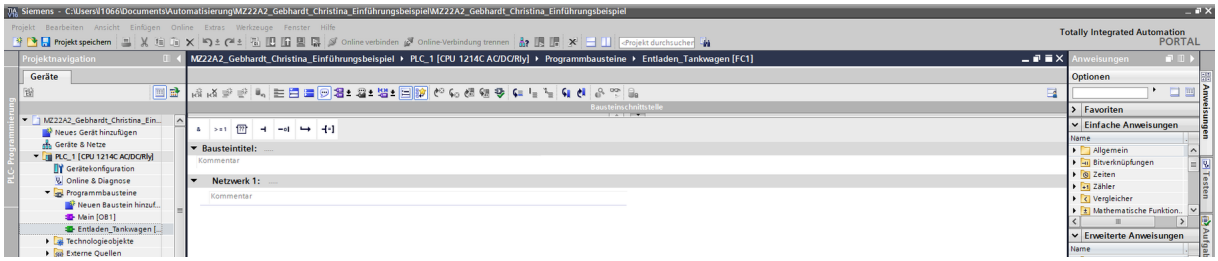
Es öffnet sich ein Dialogfeld, in dem ein neuer Baustein angelegt werden kann. Hier können folgende Einstellungen vorgenommen werden:

- **Namen** des Bausteins eintragen
- **Bausteinart** wählen (hier FC)
- **Erstellsprache** wählen (meist FUP)
- Nummerierung festlegen (kann automatisch bleiben)



Der Bausteinname kann jederzeit durch Rechtsklick in der Projektnavigation auf den Baustein angepasst werden.

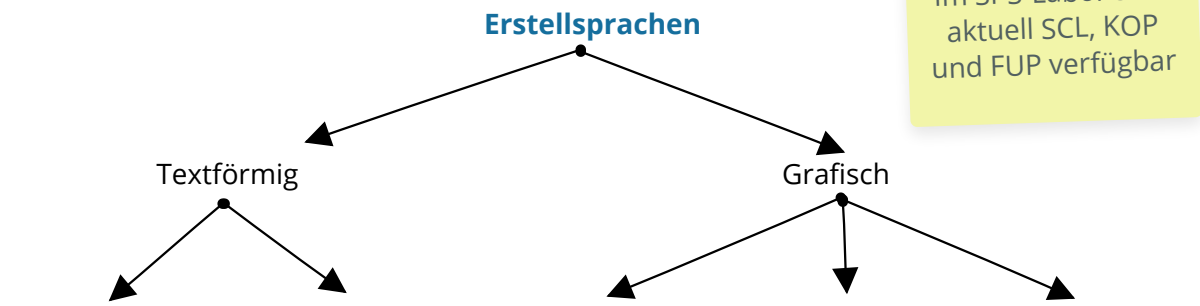
Nach dem Klick auf "OK" öffnet sich automatisch der neu angelegte Baustein im Arbeitsbereich der Projektansicht.



Erstellsprachen

SPS-Programme können in unterschiedlichen Sprachen verfasst werden, diese werden Erstellsprachen genannt. Neben den textförmigen Erstellsprachen, die Ähnlichkeiten zu echten Programmiersprachen aufweisen, gibt es als Vereinfachung für nicht informatische Facharbeiter auch grafische Erstellsprachen. Je grafischer die Erstellsprache, desto geringer der Befehlsvorrat und desto eingeschränkter die Einsatzmöglichkeiten.

Im SPS-Labor sind aktuell SCL, KOP und FUP verfügbar



Anweisungsliste (AWL)	Strukturierter Text (SCL)	Kontaktplan (KOP)	Funktionsplan (FUP)	Ablaufsprache (AB bzw. S7-GRAPH)
Programmbefehle werden durch **Buchstaben** gegeben (z.B. U für UND) und als **Liste untereinander** geschrieben.	Basiert auf der Programmiersprache **Pascal** und enthält den vollen Befehlsvorrat	Basiert auf dem **Stromlaufplan** in aufgelöster Darstellung	Basiert auf den grafischen Symbolen der **Digitaltechnik**	Basiert auf dem **GRAFCET**

4. Lokale Variablen erstellen und programmieren

OPAL Kurs

Ziel im Laborunterricht ist es, **bibliotheksfähige** Programme zu erstellen. Dafür muss die Programmierung schnittstellen- bzw. objektorientiert erfolgen. Innerhalb der FCs und FBs werden also nur **lokale Variablen** verwendet.

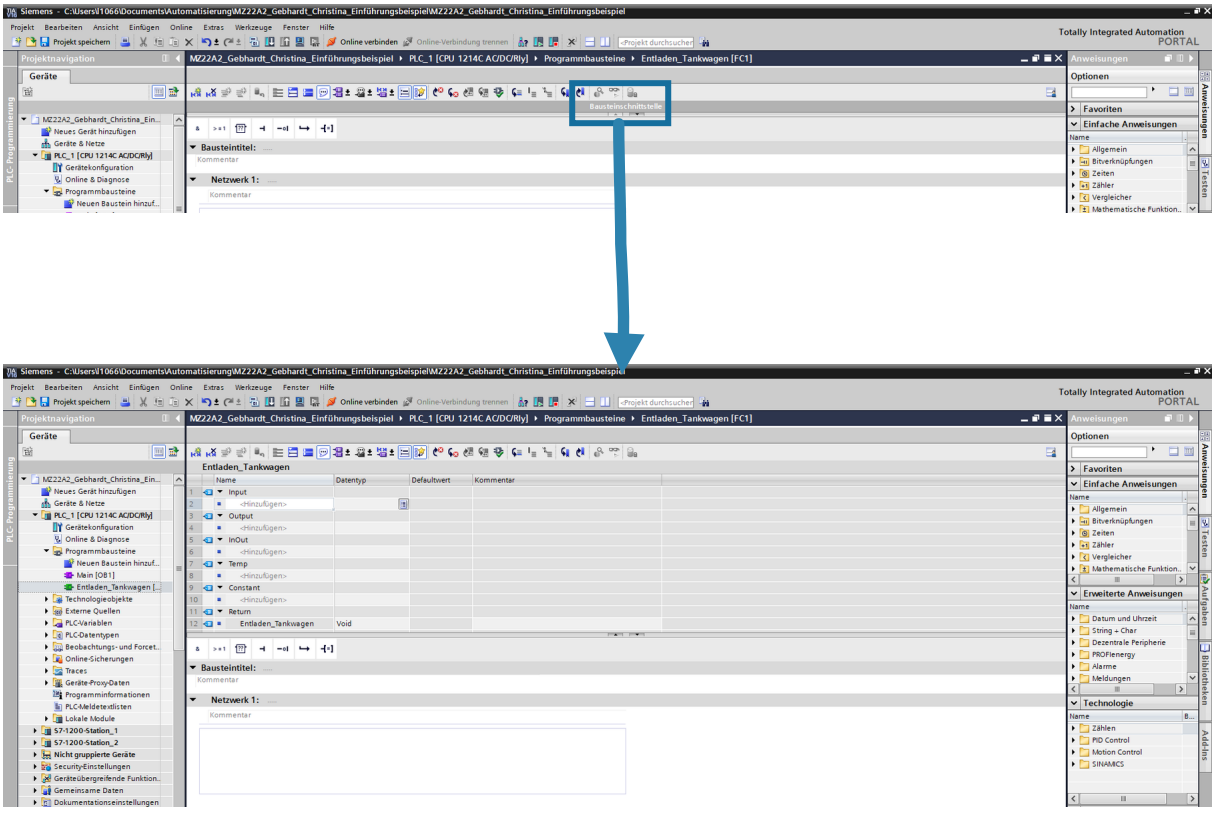
Bibliotheksfähigkeit

Bibliotheksfähigkeit bedeutet, dass das entsprechende Element in einer Bibliothek abgelegt werden kann und aus dieser in jede beliebige Steuerung integriert werden kann, ohne dass Änderungen an dem Element vorgenommen werden müssen. Im Zusammenhang mit der SPS-Programmierung bedeutet Bibliotheksfähigkeit vor allem **Hardwareunabhängigkeit**.

Lokale Variablen

Lokale Variablen sind sogenannte Durchgangsvariablen. Sie dienen als Platzhalter in einem Codebaustein und werden beim Aufruf des Bausteins mit einer globalen Variable (Adresse) verknüpft. Wird ein Codebaustein mehrfach aufgerufen, können die lokalen Variablen bei jedem Aufruf mit anderen globalen Variablen verknüpft werden.

Lokale Variablen werden in der Bausteinschnittstelle im oberen Bereich des Bausteins. Ggf. muss der Bereich für die Bausteinschnittstelle über den kleinen Pfeil geöffnet oder mit der Maus heruntergezogen werden.

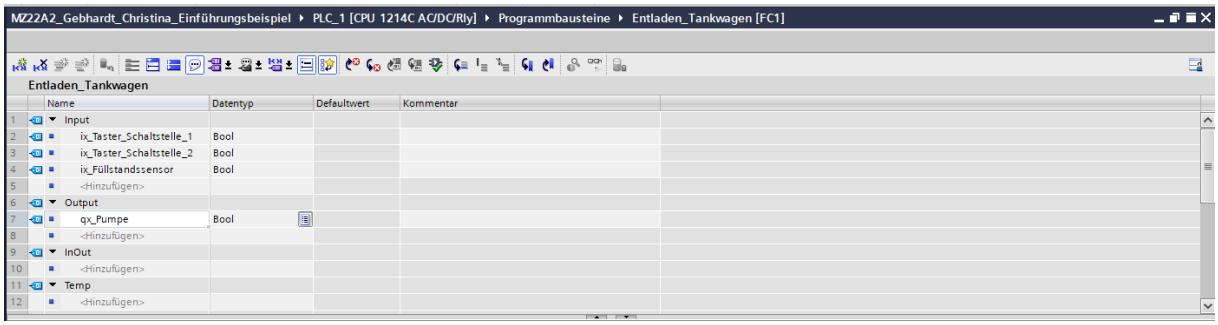


Die Funktionsbeschreibung enthält 3 Input-Variablen und eine Output-Variable. Diese werden gemäß der Beschriftungsvorschrift in der Bausteinschnittstelle angelegt.

Beschriftungsvorschrift lokale Variablen

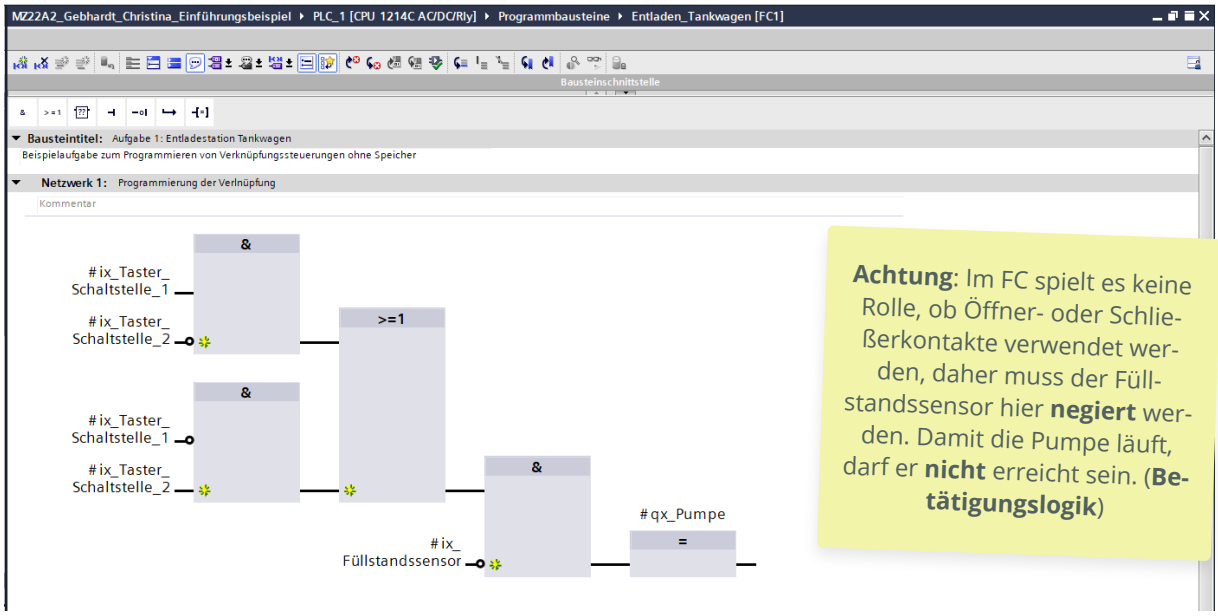
- **Variablenart:** Eingang = i; Ein-/Ausgang = iq; Ausgang = q, statische Variable = s; statische Variable für Multiinstanz = sins, temporäre Variable = t
- **Datentyp**: bool = x; int = i; time = t; etc.
- **Funktion der Variable**

Damit ergeben sich für das Einführungsbeispiel folgende Variablen:



Nach der Festlegung der Variablen kann das eigentliche Programm erstellt werden. Das Programm gliedert sich in Netzwerke. Diese dienen vor allem der Übersichtlichkeit. Die Verknüpfungen können entweder in der Schnellauswahl über den Netzwerken oder im rechten Fenster "Anweisungen" gefunden werden.

OPAL Kurs



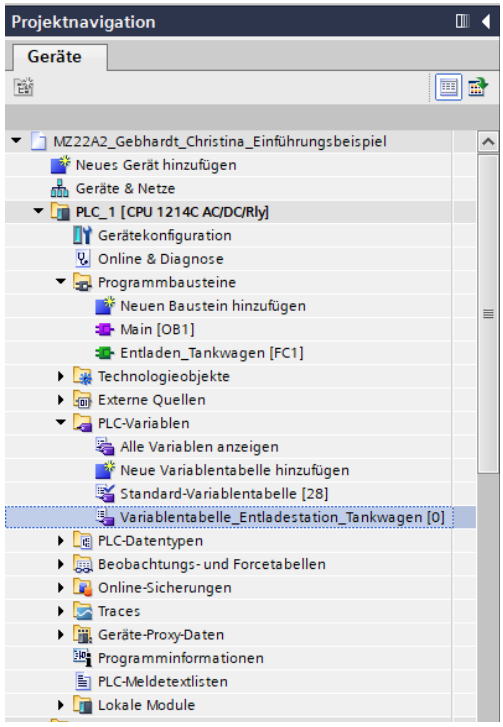
Achtung: Im FC spielt es keine Rolle, ob Öffner- oder Schließerkontakte verwendet werden, daher muss der Füllstandssensor hier **negiert** werden. Damit die Pumpe läuft, darf er **nicht** erreicht sein. (**Betätigungslogik**)

5. Globale Variablen erstellen

Globale Variablen

Globale Variablen sind Symbole für absolute Adressen. Eine globale Variable bezeichnet also in jedem Baustein ein und dasselbe. Globale Variablen können physisch an der SPS vorhandene Ein- bzw. Ausgänge oder Variablen in einem Datenbaustein bzw. Merker, also bestimmte Speicherzellen, bezeichnen. Globale Variablen werden in der PLC-Variablentabelle definiert.

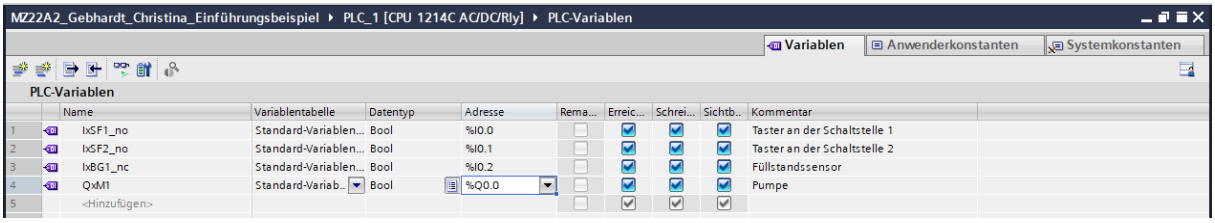
Um die globalen Variablen zu erstellen, wird zunächst eine neue Variablentabelle angelegt. Die entsprechende Schaltfläche findet sich in der Projektansicht in der Projektnavigation unter dem Punkt "PLC-Variablen". Die neue Variablentabelle kann durch einen Rechtsklick auf den Namen umbenannt und durch einen Doppelklick geöffnet werden.
Bei der Erstellung der Variablen ist wiederum die Beschriftungsvorschrift für globale Variablen zu beachten.



Beschriftungsvorschrift globale Variablen

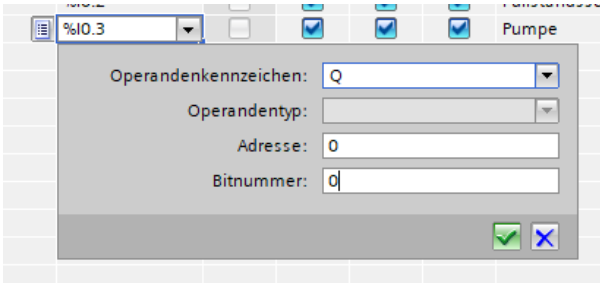
- **Variablenart:** Eingang = I; Ausgang = Q, Merker = M
- **Datentyp**: bool = x; int = i; time = t; etc.
- **Betriebsmittelkennzeichnung gemäß Aufgabenstellung**
- **Kontaktart**: Öffner = _nc; Schließer = _no

Damit ergeben sich für unser Beispiel folgende globale Variablen:



Praxishinweis

Die Autovervollständigung des TIA-Portals trägt automatisch beginnend mit der Eingangsadresse I0.0 bzw. E0.0 (je nach Spracheinstellung) fortlaufende Adressen für die globalen Variablen ein. Bei einem Wechsel der Variablenart muss man die passende Adresse per Hand eingeben.



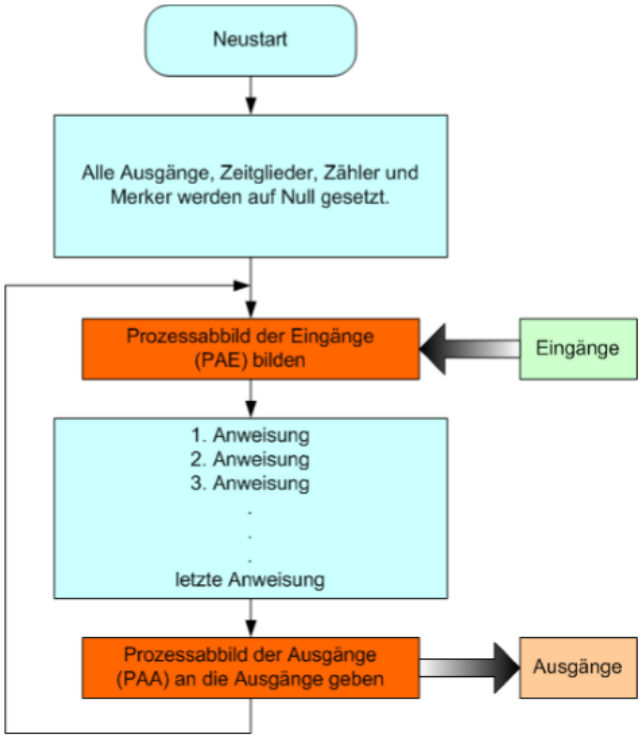
6. Bausteine in Main (OB1) aufrufen und mit globalen Variablen verknüpfen

Im nächsten Schritt erfolgt die **Verknüpfung** des im FC1 geschriebenen Anwenderprogramms **mit dem Betriebssystem** der CPU über den OB1 (Main) und **mit den Ein- und Ausgängen** über die globalen Variablen.

Der Main (OB1) ist ein zyklischer OB, d.h. er wird nach dem Einlesen des Prozessabbilds der Eingänge aufgerufen und abgearbeitet. Da er die Ordnungsnummer 1 hat, hat er die geringste Priorität. Sein Aufruf wird durch OBs höherer Ordnungsnummer, z.B. Fehleralarme, unterbrochen oder verschoben.

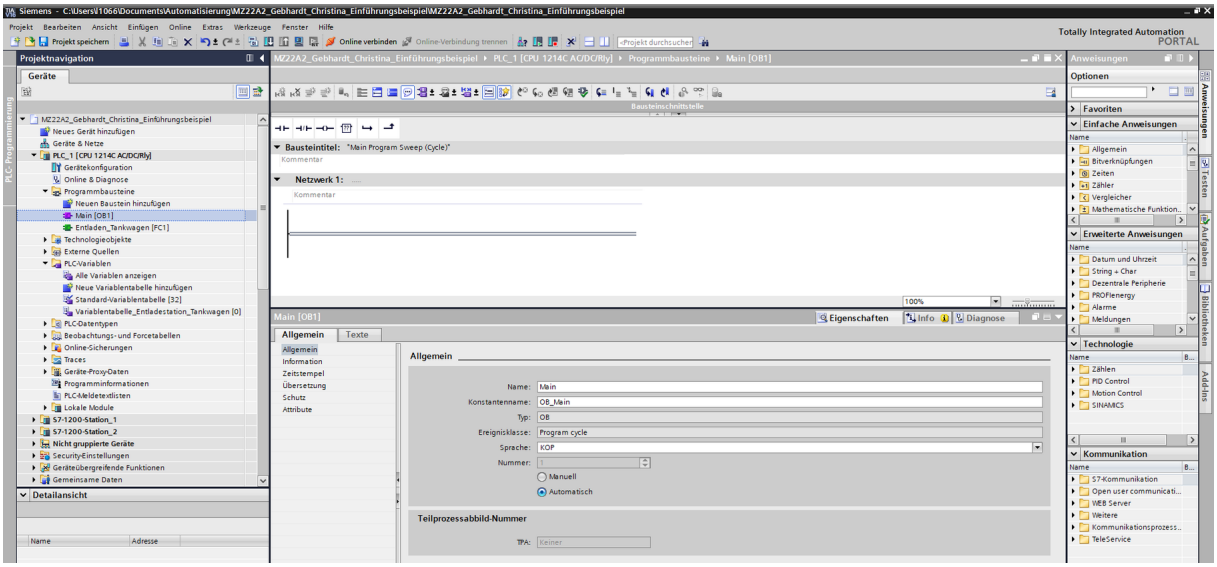
Der OB1 befindet sich in der Projektnavigation unter Programmbausteine. Er wird vom TIA-Portal selbst angelegt und kann durch Doppelklick geöffnet werden.

Standardmäßig ist der OB1 auf KOP eingestellt. Die Erstellsprache kann man in den Eigenschaften des Bausteins ändern.

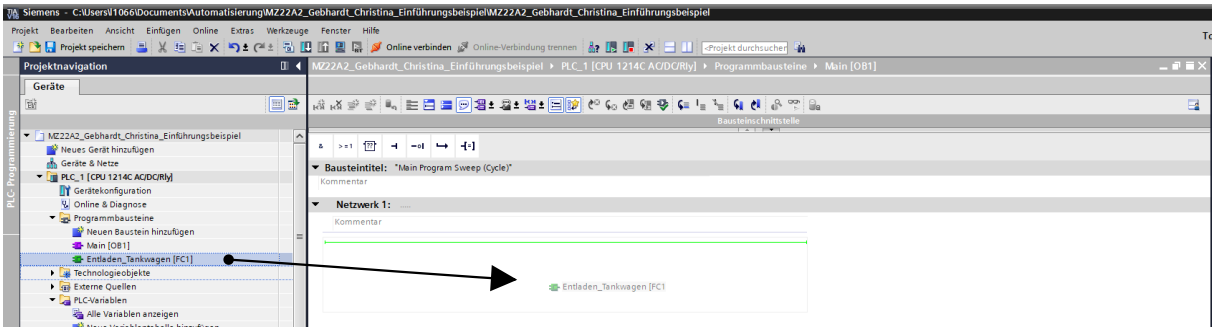


Zykylische Arbeitsweise einer SPS
Entnommen aus Schmid, D.; Baumann, A. et al: Steuern und Regeln – für Maschinenbau und Mechatronik; 2001; Europa Lehrmittel; Haan-Gruiten; 8.Auflage; S.166

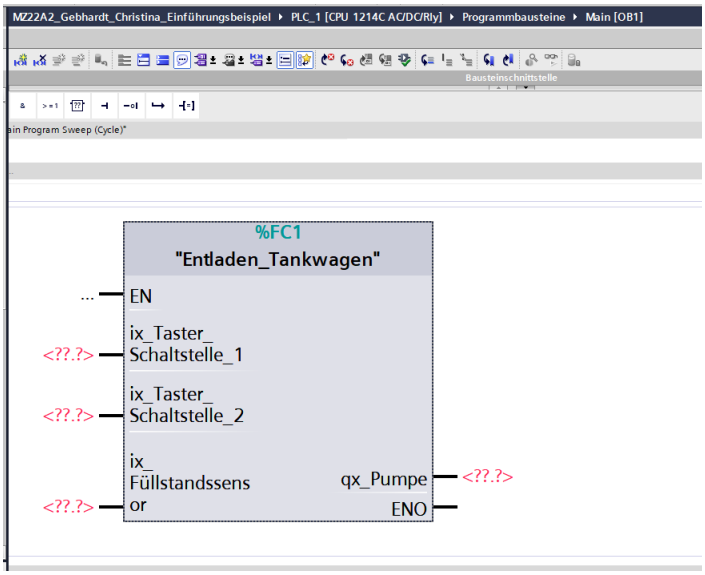
Die Eigenschaften eines Bausteins findet man im unteren Bereich der Arbeitsfläche. Sie können entweder durch den weißen Pfeil in der rechten unteren Ecke geöffnet oder mit der Maus nach oben geschoben werden.



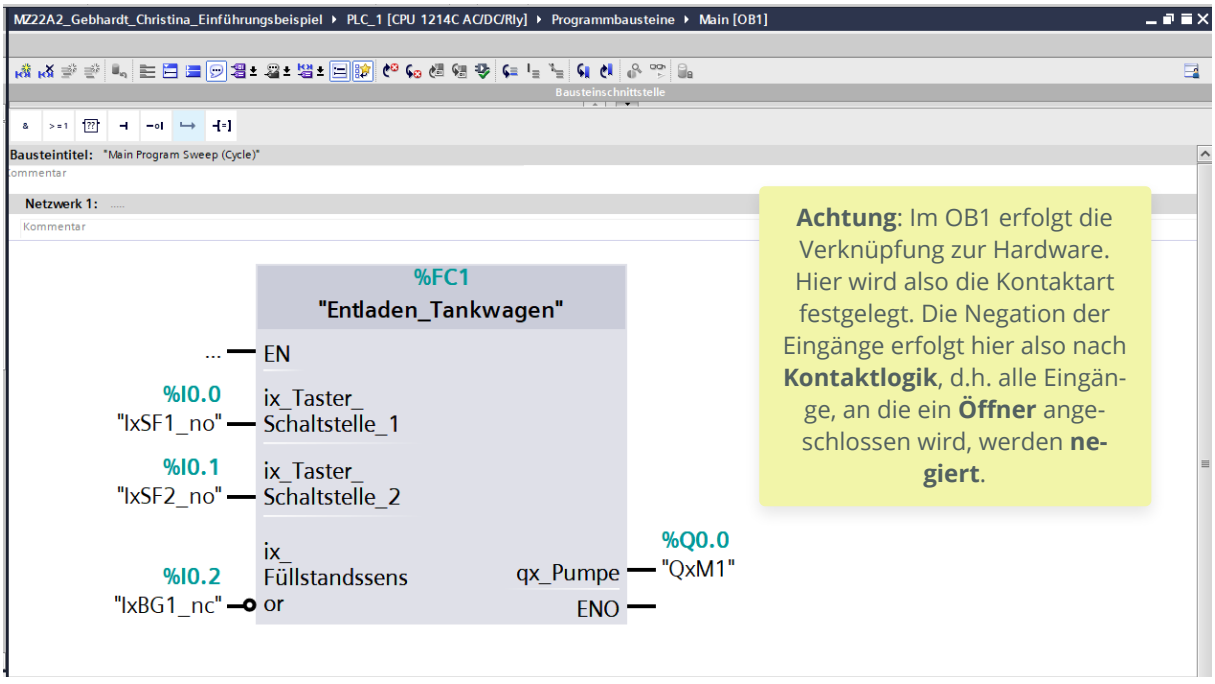
Ein Baustein kann in einem anderen aufgerufen werden, indem er aus der Projektnavigation per Drag-and-drop in das entsprechende Netzwerk gezogen wird. In diesem Beispiel muss der Baustein FC1 in das Netzwerk 1 des OB1 gezogen werden.



Im Netzwerk 1 wird nun der entsprechende Baustein eingefügt. Hat man lokale Variablen verwendet, so sind diese als Ein- bzw. Ausgänge (je nach Variablenart) am Baustein vorhanden.



Die lokalen Variablen des Bausteinaufrufs müssen nun mit den globalen Variablen verknüpft werden. Die Verknüpfung erfolgt, indem man die entsprechenden Variablennamen oder Adressen an den Ein- und Ausgängen des Bausteins einträgt.
Zum Schluss werden alle Eingänge, an denen **Öffner** angeschlossen werden, **negiert**.



Achtung: Im OB1 erfolgt die Verknüpfung zur Hardware. Hier wird also die Kontaktart festgelegt. Die Negation der Eingänge erfolgt hier also nach **Kontaktlogik**, d.h. alle Eingänge, an die ein **Öffner** angeschlossen wird, werden **negiert**.

Leitfaden zur Projekterstellung

Um leichter in die Erstellung neuer SPS-Programme zu starten, empfehlen wir Ihnen, nach einer der beiden hier vorgeschlagenen Reihenfolgen vorzugehen:

Variante 1

1. Projekt erstellen und benennen
2. Hardware konfigurieren
3. Alle globalen Variablen (E/A-Variablen) unter PLC-Variablen erstellen inkl. Kommentar
4. Programmbausteine erstellen und benennen
5. Alle lokalen Variablen in FCs/FBs erstellen und Funktion gemäß Aufgabenstellung programmieren
6. Bausteine in Main (OB1) aufrufen und mit globalen Variablen verknüpfen
7. Programm in die SPS laden und testen

Variante 2

1. Projekt erstellen und benennen
2. Hardware konfigurieren
3. Programmbausteine erstellen und benennen
4. Alle lokalen Variablen in FCs/FBs erstellen und Funktion gemäß Aufgabenstellung programmieren
5. Alle globalen Variablen (E/A-Variablen) unter PLC-Variablen erstellen inkl. Kommentar
6. Bausteine in Main (OB1) aufrufen und mit globalen Variablen verknüpfen
7. Programm in die SPS laden und testen

Regeln zur normgerechten SPS-Programmierung

Um Ihr Programmierung in Anlehnung an die **Europäischen Norm EN 61131** normgerecht und übersichtlich zu gestalten, beachten Sie folgende Grundsätze:

1. Regeln zur Benennung

In der SPS-Programmierung vergeben Sie sinnvolle Namen. Benannt werden:

- das SPS-Programm selbst
- die Code-Bausteine
- die Netzwerke
- alle globalen Variablen
- alle lokalen Variablen

Die Beschriftung der einzelnen Elemente wird in der Regel hausintern festgelegt. Die Festlegungen für das BSZ Delitzsch finden Sie unter "Beschriftungsvorschrit SPS-Programmierung"

Anwendungbeispiel

Da die Regeln zur normgerechten Programmierung auf den ersten Blick recht abstrakt wirken, können sie sich am Einstiegsprojekt MT22ASommerSusi_ZweiAusDreiTastern orientieren. In diesem finden Sie Umsetzungsbeispiele für alle Regeln aus den Abschnitten 1 bis 3.

2. Bibliotheksfähigkeit

Bibliotheksfähigkeit

Bibliotheksfähigkeit bedeutet, dass das entsprechende Element in einer Bibliothek abgelegt werden kann und aus dieser in jede beliebige Steuerung integriert werden kann, ohne dass Änderungen an dem Element vorgenommen werden müssen. Im Zusammenhang mit der SPS-Programmierung bedeutet Bibliotheksfähigkeit vor allem **Hardwareunabhängigkeit**.

Bibliotheksfähigkeit erreichen Sie durch die Einhaltung der folgenden Regeln:

1. „Absolute Adressen“ / globale Variablen (z.B. %A2.4) werden nur in Organisationsbausteinen (OB) verwendet.
2. Im Code-Baustein (FC/FB) werden stattdessen „Durchgangsvariablen“ / lokale Variablen erstellt, die als Schnittstelle zwischen den Bausteinen fungieren.
3. Im Code-Baustein werden statt Merkern (globale Variablen) lokale statische Variablen verwendet.
4. Für Programmieraufgaben mit Speicherfunktionen (Flip-Flop, Zeiten, Zähler etc.) wird ein Funktionsbaustein (FB) programmiert.
5. Innerhalb eines FB werden Elemente, die Speicher benötigen (FB, Timer, Zähler) als Multiinstanz aufgerufen.
6. Zeiten und Zähler werden mit IEC-Bausteinen programmiert.
7. Alle Teilfunktionen einer Gesamtfunktion werden in einem gemeinsamen Baustein zusammengefasst, sodass im OB1 nur ein Baustein aufgerufen werden muss.

3. Negationen

1. In einem **FC/FB** immer „positiv denken“, d.h. innerhalb des FCs/FBs erfolgt die Negation eines Eingangs gemäß der Betätigungslogik, d.h. die **Negation** erfolgt dann, wenn ein Signalgeber im jeweiligen Fall **nicht betätigt** sein soll.
2. Beim Aufruf der FCs/FBs **im OB1** erfolgt die **Negation** eines Eingangs gemäß des Kontaktes, d.h. die Negation erfolgt dann, wenn es sich bei dem jeweiligen Signalgeber um einen **Öffner** handelt.

4. Ablaufsteuerungen

In einer Ablaufsteuerung werden die Schrittkette und die Ansteuerung der Aktionen in zwei getrennten Bausteinen programmiert.

> **Hinweis**
> Die Norm ist prinzipiell nicht verbindlich, kann die Programmierung aber erheblich vereinfachen. Der Hauptvorteil besteht darin, dass die so erstellten FCs und FBs systemunabhängig und damit flexibler einsetzbar sind.

Beschriftungsvorschriften zur SPS-Programmierung

Projektname

- Klasse_Name_Vorname_Aufgabenbezeichnung

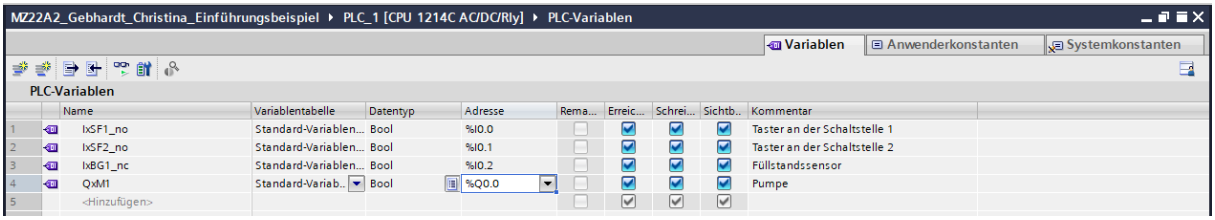
Beispiel: MT22A_Mustermensch_Sascha_Einführungsbeispiel

Globale Variablen

- **Variablenart:** Eingang = I; Ausgang = Q; Merker = M
- **Datentyp**: bool = x; int = i; time = t; etc.
- **Betriebsmittelkennzeichnung gemäß Aufgabenbeschreibung**
- **Kontaktart**: Öffner = _nc; Schließer = _no

Beispiel: IxB1_nc; QxM1

> **Globale Variablen**
>
> Globale Variablen sind Symbole für absolute Adressen. Eine globale Variable bezeichnet also in jedem Baustein ein und dasselbe. Globale Variablen können physisch an der SPS vorhandene Ein- bzw. Ausgänge oder Variablen in einem Datenbaustein bzw. Merker, also bestimmte Speicherzellen, bezeichnen. Globale Variablen werden in der PLC-Variablentabelle definiert.



lokale Variablen

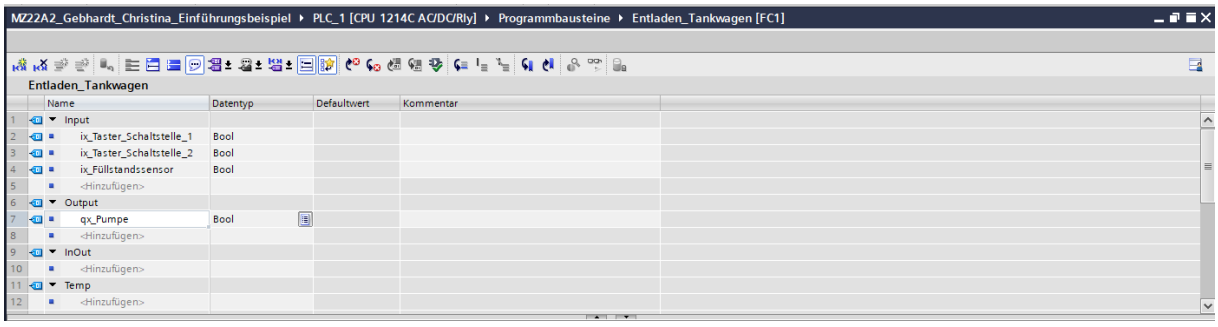
- Variablenart: Eingang = i; Ausgang = q; Ein-/Ausgang = iq; temporäre Varibale = t; statische Variable = s

- **Datentyp**: bool = x; int = i; time = t; etc.

- **Funktion der Variable**

Beispiel: ixFüllstandssensor; qxPumpe

> **Lokale Variablen**
>
> Lokale Variablen sind sogenannte Durchgangsvariablen. Sie dienen als Platzhalter in einem Codebaustein und werden beim Aufruf des Bausteins mit einer globalen Variable (Adresse) verknüpft. Wird ein Codebaustein mehrfach aufgerufen, können die lokalen Variablen bei jedem Aufruf mit anderen globalen Variablen verknüpft werden.



Bausteine der SPS-Programmierung

Es gibt zwei verschiedene Grundarten von STEP 7-Bausteinen. Die Bausteine, die das eigentliche Anwenderprogramm enthalten, nennt man Code-Bausteine. Zur Speicherung von Daten werden die Datenbausteine verwendet.

	Baustein	Kurzzeichen	Beschreibung
Code-Bausteintypen (Programm)	Organisationsbausteine	OB	Diese Bausteine stellen die **Schnittstelle zwischen Anwenderprogramm und Betriebssystem** dar. Sie **reagieren auf Aufrufe aus dem Betriebssystem und können nicht von anderen Bausteinen aufgerufen werden**. Der OB1 (Main) organisiert den zyklischen Programmablauf.
	Funktionsbausteine	FB	Ein FB ist ein Codebaustein, d.h. er dient der Programmierung des Anwenderprogramms. Zu jedem Aufruf eines Funktionsbaustein gehört ein statischer Speicherbereich (eigener **Instanz-Datenbaustein** oder als **Multiinstanz**). Dadurch hat dieser Code-Bausteintyp **bei jedem Aufruf einen eigenen Speicherbereich** und kann somit für **Ablaufsteuerungen** und **Verknüpfungssteuerungen mit Speicher** verwendet werden.
	Funktionen	FC	Ein FC ist ebenfalls ein Codebaustein. Allerdings besitzt er im Gegensatz zum FB kein Gedächtnis. Daher wird er nur für **Verknüpfungssteuerungen ohne Speicher** eingesetzt.
	System-funktionen /-funktions-bausteine	SFC/SFB	SFC und SFB sind **vorgefertigte**, im Betriebssystem der CPU bereits integrierte Code-Bausteine.
Datenbausteintypen (Daten)	Instanzdaten-baustein	DB	Dieser Datenbausteintyp **wird dem Aufruf eines Funktionsbausteins fest zugeordnet** und wird bei der Erstellung eines Funktionsbausteines mit erzeugt.
	Globaler Daten-baustein	DB	Dieser Datenbausteintyp **enthält Daten, auf die alle Code-Bausteine des Anwenderprogramms zugreifen können**. Diese Datenbausteine müssen vom Anwender programmiert werden.

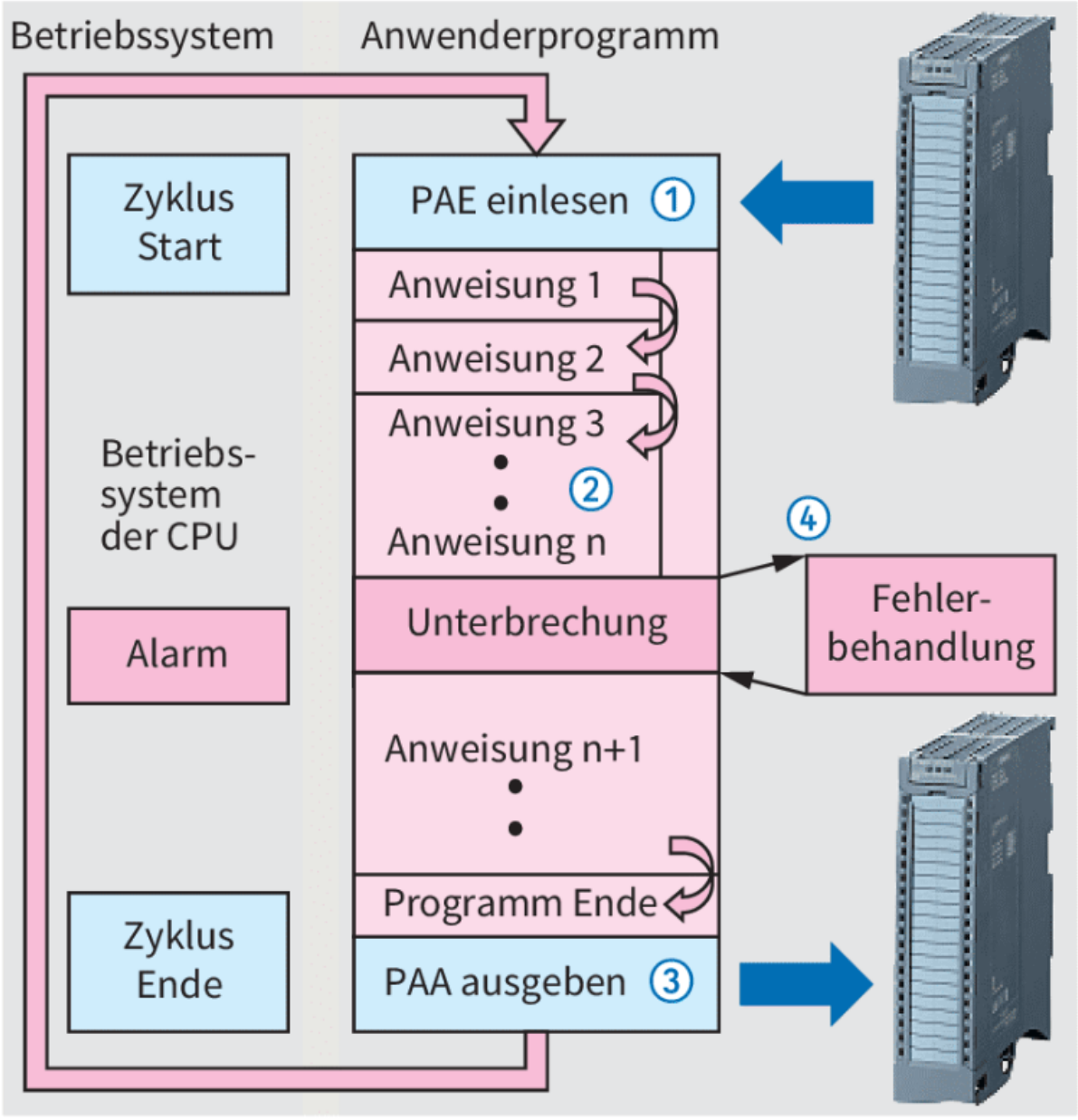
Erstellsprachen in der SPS-Programmierung

Für die SPS-Programmierung existieren 5 Erstellsprachen, von denen Sie drei für die Programmierung einer CPU 1200 zur Auswahl haben.

Erstellsprachen				
Textförmig		**Grafisch**		
Anweisungsliste (AWL)	**Strukturierter Text (SCL)**	**Kontaktplan (KOP)**	**Funktionsplan (FUP)**	Ablaufsprache (AB)
• Einzelne Programmzeilen werden untereinander geschrieben • Befehle beginnen mit einem Kennbuchstaben für die Verknüpfung (z.B. U für UND)	• Basiert auf der Programmiersprache Pascal	• Basiert auf dem Stromlaufplan in aufgelöster Darstellung	• Basiert auf den grafischen Symbolen der Digitaltechnik	• Basiert auf dem GRAFCET

Aufgabe des OB1 (Main)

Der Main (OB1) ist ein zyklischer OB, d.h. er wird nach dem Einlesen des Prozessabbilds der Eingänge aufgerufen und abgearbeitet. Er hat somit die Aufgabe, das Betriebssystem der SPS mit dem Anwenderprogramm zu verbinden und die Bearbeitung des Anwenderprogramms zu ermöglichen. Da er die Ordnungsnummer 1 hat, hat er die geringste Priorität. Sein Aufruf wird durch OBs höhe□rer Ordnungsnummer, z.B. Fehleralarme, unterbrochen oder verschoben.
Der OB1 befindet sich in der Projektnavigation unter Programmbausteine. Er wird vom TIA-Portal selbst angelegt und kann durch Doppelklick geöffnet wer□den.
Standardmäßig ist der OB1 auf KOP ein□gestellt. Die Erstellsprache kann man in den Eigenschaften des Bausteins ändern.



Bildquelle: Günter Aldejohann, Christian Bünz, Michael Krehbiel, Gennadij Petker, Holger Schmidt, Steffen Staus, Roland Stolzenburg, Mike Thielert: Mechatronik Gesamtband; Bildungshaus Schulbuchverlage Westermann, Braunschweig, 2019

Übungsaufgaben: Verknüpfungssteuerungen ohne Speicher

In Anlehnung an die Teilaufgaben 1 und 2 der Leitaufgabe „Fabriktor“ können Sie nun folgende Übungsaufgaben bearbeiten.

① **Entladestation Tankwagen**
Für eine Chemieanlage soll die Entladung der Tankwagen, die der Anlage flüssige Rohstoffe liefern, gesteuert werden. Die Entladung kann von zwei Schaltstellen aus über die Taster SF1 und SF2 erfolgen.
Solange genau einer der beiden Taster betätigt ist, wird die Pumpe angesteuert. Wenn der Tank der Anlage den maximalen Füllstand erreicht hat (BG1 = 0), muss die Pumpe sofort stoppen.

② **Befüllen eines Produktionsbehälters**
Ein Produktionsbehälter besitzt vier Zapfstellen. Die Zapfstellen werden über die Sensoren B01 bis B04 überwacht (1 = Ventil geöffnet). Er wird über zwei Pumpen unterschiedlicher Förderleistung gespeist.
Dabei genügt die Pumpe 1 zum Speisen des Behälters, wenn nur eine Zapfstelle geöffnet ist. Pumpe 2 wird benutzt, wenn zwei und drei Zapfstellen geöffnet sind. Bei gleichzeitiger Entnahme aus allen vier Zapfstellen müssen beide Pumpen den Behälter speisen.

③ **Berieselungsanlage**
In einer Bibliothek befinden sich vier Temperaturmelder. In einem Brandfall sollen diese eine Berieselungsanlage einschalten. Aus sicherheitstechnischen Gründen soll die Anlage erst dann eingeschaltet werden, wenn mindestens drei Temperaturmelder ansprechen.

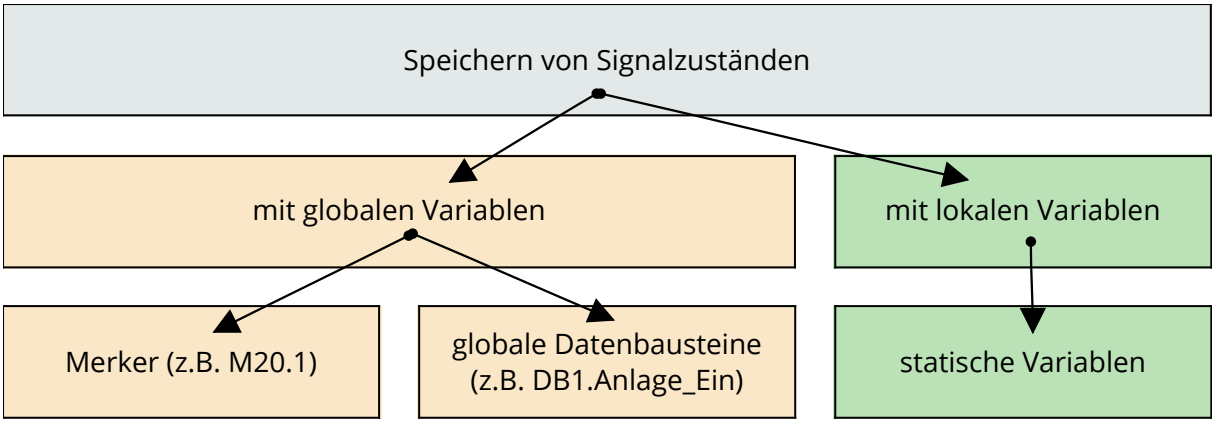
④ **Alarmanlage**
Bei einer Alarmanlage kann von drei verschiedenen Stellen aus Alarm ausgelöst werden: durch Schließen des Infrarotsensors B01 oder durch Öffnen eines Türkontaktes B02 oder durch Öffnen eines Fensterkontaktes B03. Die zugehörige Alarmhupe wird in Betrieb gesetzt, wenn am Ausgang der SPS ein 1-Signal liegt. Die Alarmanlage wird über den Schalter S1 ein- und ausgeschaltet.

⑤ **Helligkeitssteuerung**
In einem Arbeitsraum wird an vier Stellen die Helligkeit gemessen. Die Beleuchtung soll automatisch eingeschaltet werden, wenn Sensor B2 und B3 oder die Sensoren B1 und B2 und B4 oder die Sensoren B1 und B3 und B4 ansprechen.

⑥ **Rauchmeldeanlage**
Eine Lagerhalle wird durch vier Rauchsensoren (B1 bis B4) überwacht. Bei Rauch geben sie ein 0-Signal, sonst ein 1-Signal. Wenn ein Sensor Rauch erkennt, leuchtet eine Warnleuchte (P1). Bei mindestens zwei gleichzeitigen Meldungen ertönt die Sirene (P2).
Geben vier Rauchmelder einen Alarm, so wird ein automatisches Wählsystem eingeschaltet, dass die Feuerwehr alarmiert. Das Wählgerät (P3) wird über einen Ausgang der SPS angesteuert (Ausgang auf „1“).

Hinweise zur Verwendung von Speichergliedern

Um Signalzustände in einer S7 zu speichern gibt es grundsätzlich 3 Möglichkeiten. Dabei verwenden 2 Möglichkeiten globale und eine lokale Variablen:



Wann verwende ich was?

Merker
Im Rahmen der normgerechten Programmierung im SPS-Labor verzichten wir weitestgehend auf die Verwendung von Merkern. Sie werden **lediglich als System- bzw. Taktmerker** für Funktionen der CPU genutzt.

Globale Datenbausteine
Globale Datenbausteine werden vor allem für die **Kommunikation** der CPU **mit anderen Geräten** benötigt. Über einen globalen Datenbaustein können z.B. Informationen mit einem HMI (Panel) ausgetauscht werden.

Statische Variablen
Ein Ziel der **normgerechten Programmierung** sind **bibliotheksfähige**, beliebig oft aufrufbare Bausteine. Um dies zu erreichen, darf innerhalb eines FCs oder FBs ausschließlich mit lokalen Variablen gearbeitet werden. Um Signale zu speichern werden statische Variablen verwendet, diese stehen ausschließlich im FB zur Verfügung. Wird ein FB aufgerufen, wird diesem Aufruf ein Instanz-Datenbaustein zugeordnet. In diesem werden die Zustände der statischen Variablen abgespeichert. Da bei jedem Aufruf des FB ein neuer Instanz-Datenbaustein zugeordnet wird, wird die jeweilige statische Variable für jeden Aufruf mit einer neuen globalen Variable im jeweiligen DB verknüpft. Die einzelnen Aufrufe können so ohne Änderungen am Programm unabhängig voneinander bearbeitet werden.

Instanz-Datenbausteine

Damit ein Programm von einer CPU bearbeitet werden kann, müssen die meisten lokalen Variablen mit globalen Variablen verknüpft werden, dies gilt insbesondere auch für speichernde lokale Variablen (statische Variablen).
Verfügt ein Baustein über **statische Variablen** muss er daher **beim Aufruf mit einem Instanz-Datenbaustein verknüpft** werden. Dies gilt auch für Bausteine, die nicht selbst programmiert sondern aus der Bibliothek eingefügt werden. Auch **Zeitbausteine** benötigen intern statische Variablen und müssen daher mit einem Instanz-Datenbaustein verknüpft werden.

Multiinstanzen

Multiinstanzen werden verwendet, wenn Bausteine, die einen Instanz-Datenbaustein benötigen, in einem FB aufgerufen werden.

Würde man in diesem Fall für den Aufruf einen eigenen Instanz-Datenbaustein vergeben, wäre der übergeordnete FB nicht mehr bibliotheksfähig, da in ihm globale Variablen im Datenbaustein verwendet werden.

Um dieses Problem zu vermeiden, erhält der Aufruf des Bausteins keinen eigenen Instanz-Datenbaustein vergeben, sondern der Instanz-Datenbaustein des übergeordneten FB mitverwendet. Da so, der im FB aufgerufene Baustein bei jedem Aufruf des FB einem neuen Instanz-Datenbaustein zugeordnet werden kann, bezeichnet man ihn als Multiinstanz.

Für eine Multiinstanz wird eine statische Variable im übergeordneten FB benötigt. Diese legt das TIA-Portal automatisch an.

Bei jedem Aufruf eines Bausteins mit Speicherfunktionen öffnet sich das Dialogfenster "Aufrufoptionen". Erfolgt der Aufruf in einem FB wird die Option Multiinstanz angeboten.

Praxishinweis

"Wenn du **Multiinstanz** auswählen kannst, dann tu das! Nimm Einzelinstanz nur, wenn nichts anderes geht."

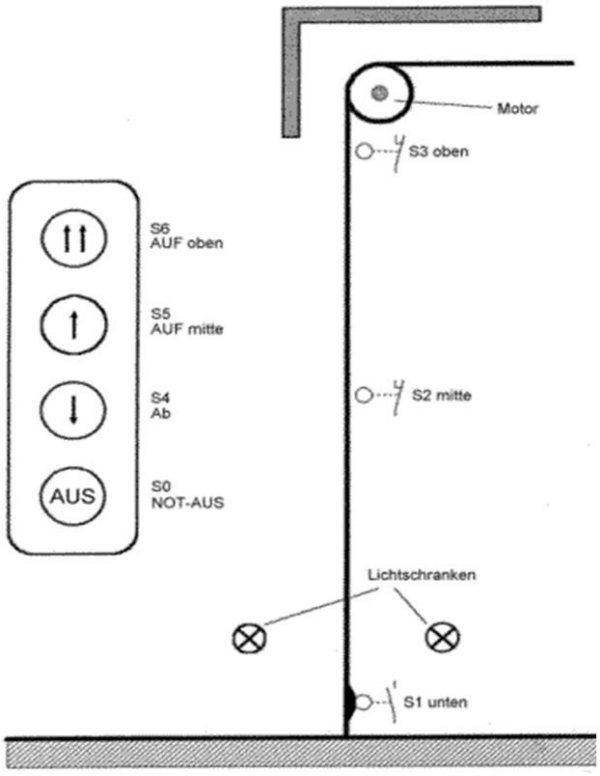


Übungsaufgaben: Normgerechte Programmierung von Verknüpfungssteuerungen mit Speichern

In Anlehnung an die Teilaufgaben 3 und 4 der Leitaufgabe „Fabriktor“ können Sie nun folgende Übungsaufgaben bearbeiten.

⑦ **Rolltor**

Ein Rolltor soll für Pkw- bzw. Lkw-Verkehr unterschiedlich weit aufgefahren werden. Wird bei geschlossenem Tor der Taster „AUF Mitte“ (S5) betätigt, so fährt das Tor bis in die mittlere Stellung auf. Bei Betätigung des Tasters „AUF oben“ (S6) fährt das Tor bis in die obere Stellung. Wird der NOT-AUS-Taster (S0) betätigt, schaltet die Anlage sofort ab. Mit der Taste „Ab“ (S4) wird das Tor zugefahren. Aus Sicherheitsgründen ist vor bzw. hinter dem Tor je eine Lichtschranke montiert. Wird bei der Abwärtsbewegung des Tores eine der beiden Lichtschranken unterbrochen, muss der Motor sofort abgeschaltet werden und danach in die obere Endlage fahren. Als zusätzliche Sicherheit werden vor und hinter dem Tor noch Warnlampen installiert, die bei jeder Bewegung des Tores blinken sollen.



⑧ **T-Kippglied**

Programmieren Sie folgende Funktion: Wird der Taster S1 betätigt, so wird das Schütz K1 angesteuert. Wird der Taster S1 erneut betätigt, so fällt das Schütz wieder ab.

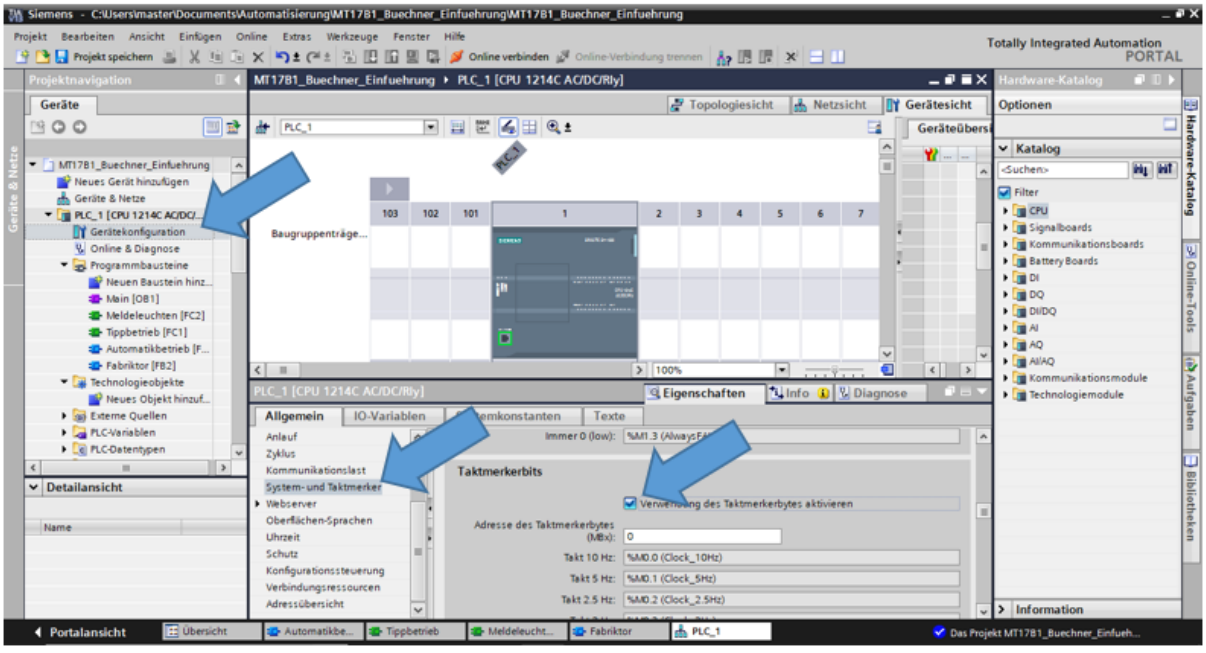
⑨ **Wendeschaltung Band**

Der Antrieb eines Förderbandes erfolgt durch einen Drehstrommotor. Bei Betätigung des Tasters „S1_vor“ wird durch Linkslauf des Motors das Band vorwärts bewegt. Bei Betätigung des Tasters „S3_zurück“ wird durch Rechtslauf des Motors das Band rückwärts bewegt. Bei Betätigung des Tasters „S2_Aus“ soll der Motor stehen bleiben. Eine Umkehr der Bewegungsrichtung soll durch Betätigung des Tasters für die Gegenrichtung direkt möglich sein. Eine gleichzeitige Betätigung der Taster „S1_vor“ und „S3_zurück“ bewirken Motorstillstand, desgleichen wenn ein Taster „S1_vor“ oder „S3_zurück“ und der Taster „S2_Aus“ gleichzeitig gedrückt werden. Waren die Taster „S1_vor“ und „S3_zurück“ gleichzeitig gedrückt, müssen erst beide wieder losgelassen werden, bevor ein Neustart in der gewünschten Richtung wirksam wird. Ist der Motor in Betrieb, wird dies durch eine Meldeleuchte angezeigt.

Verwendung eines Blinktakts

Blinktakte können durch eine Kombination von Timern erzeugt werden. S7 Steuerungen stellen allerdings systemseitig einen Blinktakt zur Verfügung. Im Folgenden wird der systemeigene Blinktakt verwendet. Die CPU stellt 8 verschiedene Taktfrequenzen zur Verfügung, die auf den 8 Bit des Taktmerkers liegen. Um den Taktmerker zu verwenden, muss dieser in den Geräteeinstellungen aktiviert werden.

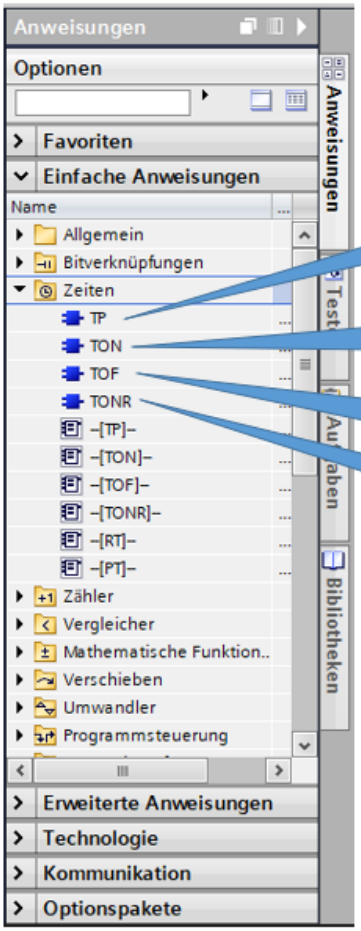
Gehen Sie in die **Projektansicht** und öffnen Sie die **Gerätekonfiguration**. Markieren Sie die CPU und schieben Sie das **Eigenschaftsfenster** (in der Mitte unten) etwas nach oben. In den Eigenschaften der CPU finden Sie den Unterpunkt **System- und Taktmerker**. Setzen Sie hier den Haken bei Taktmerker, um diesen zu verwenden und speichern Sie die Einstellung.



Nach Standardeinstellung liegen die verschiedenen Takte nun auf dem Merkerbyte 0. Da gemäß Norm in den Codebausteinen keine globalen Variablen wie Merker verwendet werden sollen, wird der Blinkttakt bis zum OB1 als lokale Inputvariable definiert.

Hinweise zur Verwendung von Zeiten

In vielen Programmen werden Zeitverzögerungen benötigt. Diese werden in der SPS Programmierung mittels **TIMERN** realisiert. **TIMER** benötigen einen **Datenbaustein**, der ihnen Speicherplatz zuweist. Für eine normgerechte Programmierung muss der TIMER im FB als **Multiinstanz** aufgerufen werden, d.h. der TIMER verwendet den Datenbaustein, der dem FB im OB1 zugewiesen wird.



Die Zeitfunktionen finden Sie im Fenster „Anweisungen“ unter dem Punkt Zeiten. Für eine normgerechte Programmierung verwenden Sie die TIMER mit den blauen Symbolen.

TP – Impulstimer: erzeugt ein Signal von definierter Länge

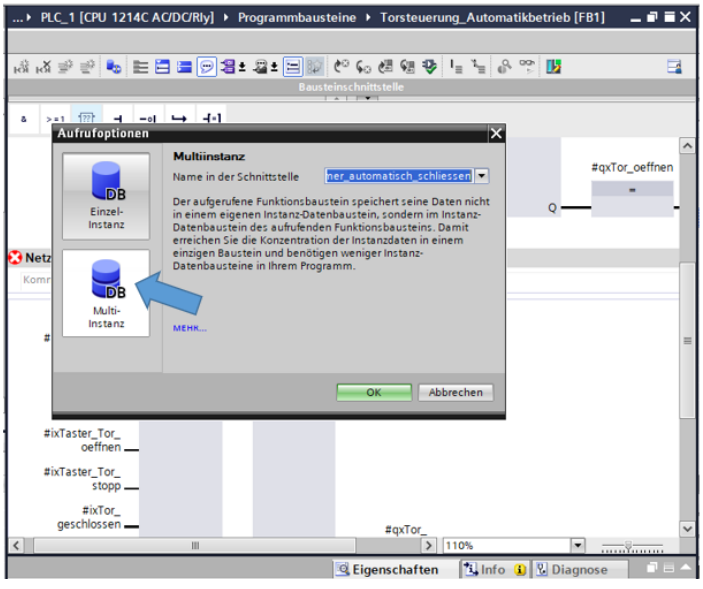
TON – Einschaltverzögerung: der Ausgang schaltet erst auf „1“, wenn der Eingang bereits die eigestellte Zeit lang eine „1“ führt.

TOF – Ausschaltverzögerung: der Ausgang bleibt die eigestellte Zeit lang noch auf „1“, wenn der Eingang bereits eine „0“ führt.

TONR – Einschaltverzögerung mit Rücksetzfunktion

Eine ausführlichere Beschreibung zu den einzelnen Zeitbausteinen erhalten Sie in der Hilfe des TIA Programms. Ziehen Sie dazu den Baustein in die Programmierfläche, markieren Sie ihn und drücken Sie F1.

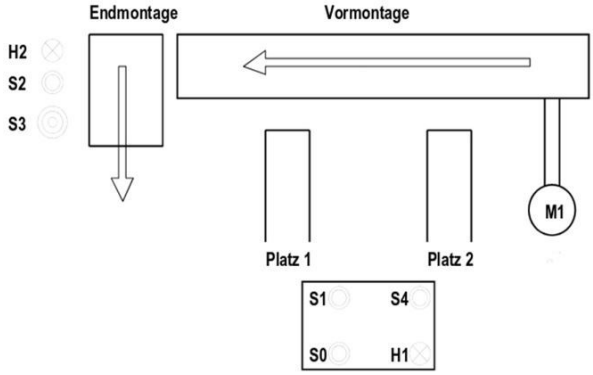
Wenn Sie einen Zeitbaustein in einen FB einfügen, öffnet sich folgendes Dialogfenster. In diesem Dialogfenster können Sie entscheiden, ob dem Zeitbaustein ein eigener Datenbaustein zugeordnet werden soll (Einzelinstanz) oder ob er den Datenbaustein des FBs mitverwenden soll (Multiinstanz). Wählen Sie **Multiinstanz** aus und geben Sie dem Timer einen Namen.
Das TIA Portal legt nun automatisch eine statische Variable für den Timer an.



Übungsaufgaben: Normgerechte Programmierung von Verknüpfungssteuerungen mit Speichern und Zeiten

In Anlehnung an die Teilaufgaben 5, 6 und 7 der Leitaufgabe „Fabriktor“ können Sie nun folgende Übungsaufgaben bearbeiten.

(10) Förderanlage



Ein Förderband transportiert die an zwei Montageplätzen vorgefertigten Montageelemente gemeinsam zur Endmontage. Die Steuerung wird mit dem Taster S1 eingeschaltet. Im EIN- Zustand leuchtet die Lampe H1. Wurde die Steuerung vor Ablauf der Förderzeit von Hand oder durch Netzausfall ausgeschaltet, so darf das Band nach dem Wiedereinschalten nicht selbsttätig anlaufen. Mit dem Taster S4 kann der Motor M1 im Tippbetrieb eingeschaltet werden, um Bauteile vom Montageplatz 2 zum Montageplatz 1 zu transportieren.
Nach dem Auflegen des letzten Bauteils wird der Bandmotor M1 mit dem Taster S2 eingeschaltet. Seine Einschaltdauer ist durch die Transportzeit zwischen Platz 2 und der Endmontage bestimmt; die Zeit wird bei der Inbetriebnahme eingestellt. Sobald das Band anläuft, leuchtet an der Endmontage die Lampe H2. Sollte während das Band läuft ein gefährlicher Zustand auftreten, kann die Steuerung über den Taster S3, einem NOT-AUS- Schalter, der am Platz der Endmontage angebracht ist, sofort still gesetzt werden. Die Steuerung wird mit dem Taster S0 ausgeschaltet.

(11) Lüftersteuerung

Der Lüfter eines innenliegenden Badezimmers soll gesteuert werden. 15 Sekunden nach Einschalten des Badezimmerlichtes über den Schalter S1 läuft der Lüfter an. Der Lüfter soll nach Ausschalten des Lichtes noch 20 Sekunden lang eingeschaltet bleiben. Außerdem ist es möglich, den Lüfter über den Schalter S2 direkt ein- und auszuschalten.

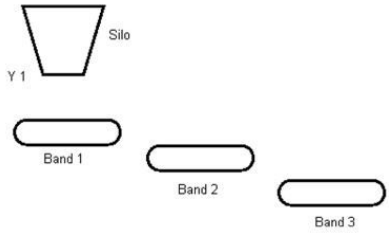
(12) Temperaturüberwachung

Wird von einem Temperatursensor eine erhöhte Temperatur gemeldet (1-Signal), dann wird nach 5s die Kontrolllampe P1 dies anzeigen, aber nur wenn die Temperatur nach dieser Zeit noch erhöht ist. Eine weitere Lampe P2 zeigt die Erhöhung der Temperatur ebenfalls nach 5s an, aber im Unterschied zur Lampe P1 auch dann, wenn die Temperaturerhöhung innerhalb der 5s wieder abgeklungen ist. Eine Quittiertaste S1 schaltet die Lampe P2 mit einer Zeitverzögerung von 10s aus.

(13) Verpackungsmaschine

Eine Verpackungsmaschine soll mit einem „Eintaster“ ein- und mit einem „Austaster“ ausgeschaltet werden. An der Anlage befinden sich zwei Vorratsbehälter. Ist einer der beiden Vorratsbehälter leer (Teile „S 2“; Kartons „S 3“) wird über eine zugehörige Meldeleuchte (blinkend) ein Signal gegeben. Wird der Behälter nicht innerhalb von 5 Sekunden wieder gefüllt, wird über eine Hupe und eine weitere Meldeleuchte ein Signal gegeben. Die Hupe lässt sich durch den Taster „S 4“ quittieren die Meldeleuchte nur durch einen Schlüsseltaster im Meisterbüro. Die Maschine läuft nur, wenn Material vorhanden ist und keine Meldung vorliegt.

(14) Siloanlage



Entwickeln Sie für die Siloanlage eine sinnvolle Einschaltreihenfolge und schreiben Sie ein SPS-Programm mit automatischer Einschaltung der 3 Bänder und des Siloventils im Abstand von jeweils 5 Sekunden. Der Hochlauf soll durch P2 blinkend (2Hz) und der Dauerbetrieb durch P3 mit Dauerlicht angezeigt werden. Eine Störung wird durch P1 (1Hz-Blinklicht) signalisiert. Mit dem Taster S1 wird die Steuerung ein und mit dem Taster S0 ausgeschaltet. Die Motoren werden durch je ein Motorschutzrelais geschützt. Die Schütze sind nach sicherheitstechnischen Belangen zu verriegeln.

(15) Schiebetür

Die Schiebetür in einem Kaufhaus soll sich automatisch öffnen und schließen. Zum Öffnen der Tür muss die Fußmatte betreten werden. Mit dem Betreten der Matte wird der Sensor B01 aktiviert, der die Tür öffnet. Nach dem Verlassen der Matte schließt die Tür nach einer Zeitverzögerung von 5 Sekunden.
Wird innerhalb der Verzögerungszeit die Matte erneut betreten, dann wird die Zeit angehalten und nach Verlassen der Matte erneut gestartet. Wird während des Schließvorgangs die Matte erneut betreten, dann wird die Tür sofort wieder geöffnet. Aus Sicherheitsgründen verhindert eine Lichtschranke B02 das Schließen der Tür, solange der Lichtstrahl unterbrochen ist. Mit einem Hauptschalter S01 kann die Automatik ausgeschaltet werde. In diesem Fall schließt die Tür sofort. Ein Notaustaster S0 setzt die Anlage sofort still.

(16) Stern-Dreieck-Anlauf

Bei der Steuerung eines Drehstrommotors soll der Motor in Sternschaltung anlaufen. Nach einer Anlaufzeit von 5 Sekunden soll der Motor automatisch in den Dreieckbetrieb umgeschaltet werden. Mit dem Taster „S2“ werden das Sternschütz „Q1“ und das Netzschütz „Q2“ eingeschaltet. Es ist zu beachten, dass das Netzschütz aber erst dann eingeschaltet werden darf, wenn das Sternschütz eingeschaltet ist. Nach der Hochlaufzeit von 5 Sekunden wird automatisch das Sternschütz abgeschaltet und das Dreieckschütz eingeschaltet. Der Sternbetrieb soll durch „P2“ blinkend (2Hz) und der Dauerbetrieb durch „P2“ mit Dauerlicht angezeigt werden. Eine Störung wird durch „P1“ (1Hz-Blinklicht) signalisiert.
Mit dem Taster „S1“ wird die Steuerung ausgeschaltet. Der Motor wird durch ein Motorschutzrelais geschützt. Die Schütze sind nach sicherheitstechnischen Belangen zu verriegeln.

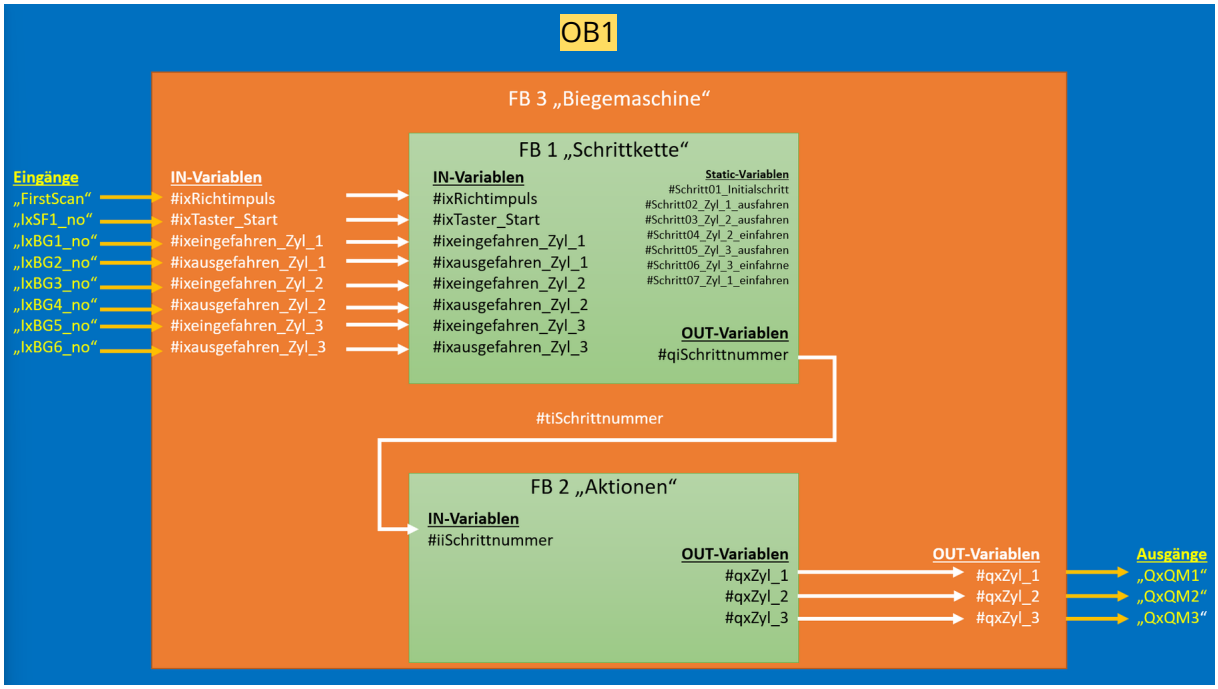
(17) Rolltreppe

Eine Rolltreppe wird über einen Taster „Betriebsbereit“ in den Bereitschaftszustand versetzt. Der Anlauf erfolgt, wenn die Lichtschranke von Personen unterbrochen wird (Unterbrechung = 1-Signal). Der Antrieb soll dann jeweils 20 Sekunden eingeschaltet bleiben. Dabei muss berücksichtigt werden: wenn mehrere Personen hintereinander die Rolltreppe benutzen, muss auch die letzte Person die vollen 20 Sekunden Laufzeit erhalten. Bei Betätigung eines Austasters wird der Bereitschaftszustand verlassen; dabei muss gegebenenfalls der aktuelle Betrieb beendet werden. Bei Betätigung des Sofortaustasters oder bei Ansprechen des Thermofühlers des Antriebsmotors muss die Anlage sofort angehalten werden und über Blinktakt (4Hz) eine Störmeldung ausgegeben werden. Gemeldet werden: „Betriebsbereit“, „Aus“, „Störung“.

Hinweise zur Programmierung von Ablaufsteuerungen

Programmstruktur

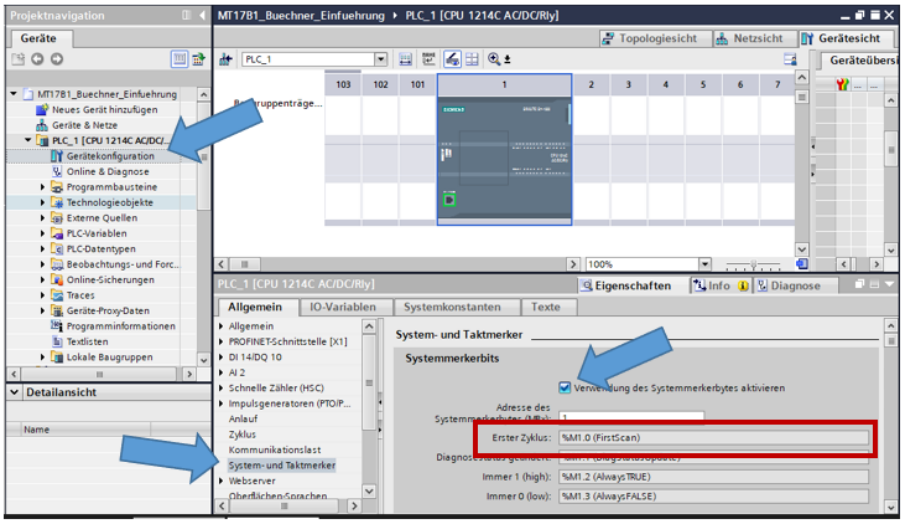
Laut Norm müssen die Schrittkette und die Aktionen einer Ablaufsteuerung in zwei getrennten Bausteinen programmiert werden. Aufgrund der Bibliotheksfähigkeit müssen diese beiden Bausteine in einem 3. Baustein zusammengefasst werden. Daher ergibt sich folgende Programmstruktur (anhand der Leitaufgabe):



Richtimpuls

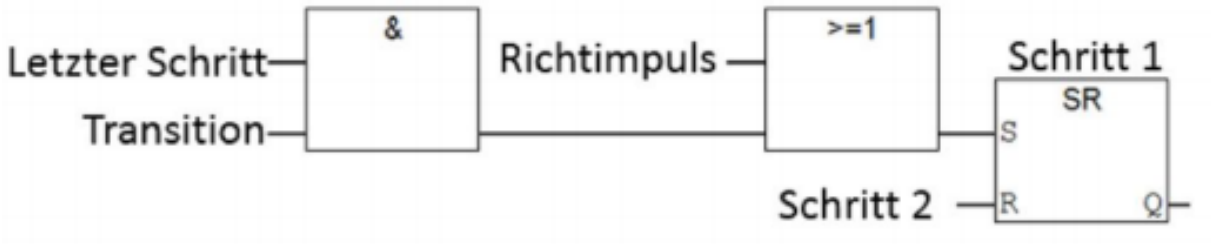
Gemäß der Regeln für eine Ablaufkette (siehe Arbeitsheft LF7), muss der erste Schritt einer Ablaufkette, also der Initialschritt, nach dem Einschalten der SPS ohne eine weitere Bedingung aktiv werden. Um dies zu erreichen wird ein Signal benötig, das nur für die Dauer des ersten Zyklus eine „1“ führt und die restliche Bearbeitungszeit auf „0“ bleibt. Dieses Signal wird als Richtimpuls bezeichnet.

Im **TIA-Portal** kann für den Richtimpuls des **Systemmerkerbit 0 „FirstScan“** genutzt werden. Die Systemmerker werden in den Einstellungen der CPU aktiviert und liegen nach Standardeinstellungen auf dem Merkerbyte 1.

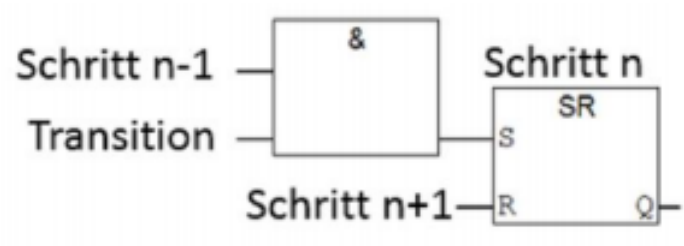


Programmierung der Schrittkette

Wie bereits erwähnt setzt der Richtimpuls den Initialschritt. Zudem gilt für den Initialschritt, wie für jeden anderen Schritt der Kette auch, dass er von dem vorangegangenen Schritt und der entsprechenden Transition gesetzt und von dem folgenden Schritt rückgesetzt wird. Da eine Ablaufsteuerung immer einem geschlossenen Kreislauf folgt, ist der vorangegangene Schritt im Fall des Initialschrittes der letzte Schritt der Kette. Für den Initialschritt gilt daher folgende Grundstruktur:



Der **Schritt** selbst ist eine **statische Variable**. Statische Variablen können nur innerhalb eines Bausteins verwendet werden, Der aktive Schritt muss aber aus der Schrittkette an die Aktionen übergeben werden. Hierfür gibt es verschiedene Realisierungsmöglichkeiten. Beispielsweise kann für jeden Schritt eine **OUT-Variable** angelegt werden, die jeweils an den Ausgang Q des SR-Gliedes angeschlossen wird. Eine elegantere Variante ist es, den aktiven Schritt als Schrittnummer (**OUT-Variable**) an den Baustein Aktionen zu übergeben (siehe Beispiel).
Bei den **Transitionen** handelt es sich bezogen auf die Schrittkette hingegen um Eingangssignale. Daher müssen die Transitionen als lokale **IN-Variablen** festgelegt werden. Alternativ können Transitionen auch zeitabhängig sein. In diesem Fall werden **Zeitglieder** als **Multiinstanz** eingesetzt.
Die folgenden Funktionsschritte ähneln dem Initialschritt, werden aber nicht mehr durch den Richtimpuls gesetzt, sodass sich für sie die folgende Struktur ergibt:



Programmierung der Aktionen

Im zweiten Baustein „Aktionen“ wird die Ansteuerung der späteren Ausgänge programmiert. Um Doppelzuweisungen zu vermeiden, sollte sich die Programmierung der Aktionen nicht an der Schrittkette, sondern an den Ausgängen orientieren. D.h. pro Ausgang wird ein Netzwerk angelegt. Die späteren **Ausgangssignale** werden dabei als lokale **OUT-Variablen** deklariert. Da die Ansteuerung der Ausgänge in Abhängigkeit von den Schritten erfolgt, muss der Zustand der Schritte an den Baustein Aktionen übergeben werden. Bezogen auf die Aktionen sind die **Schritte** bzw. die **Schrittnummer** Eingangssignale und müssen demzufolge als lokale **IN-Variablen** definiert werden.
Die Entscheidung, ob die Aktionen in einem Funktionsbaustein oder einer Funktion programmiert werden, ist abhängig vom GRAFCET. Enthält der GRAFCET **ausschließlich kontinuierlich wirkende Aktionen** ohne zeitabhängige Zuweisungsbedingungen, genügt eine **Funktion (FC)**. Enthält der GRAFCET aber **speichernde Aktionen oder zeitabhängige Zuweisungsbedingungen,** muss ein **Funktionsbauseiten (FB)** verwendet werden.
Sieht man von möglichen Zuweisungsbedingungen ab, gibt es prinzipiell 3 mögliche Arten von Aktionen bzw. Zuweisungen an den Ausgang.

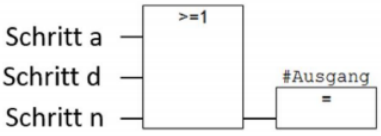
1. Möglichkeit:

Der Ausgang wird nur in einer einzigen kontinuierlich wirkenden Aktion verwendet, d.h. er ist nur genau in einem Schritt aktiv. Dabei handelt es sich um die einfachste Variante. Der jeweilige Schritt wird dem Ausgang direkt zugewiesen.



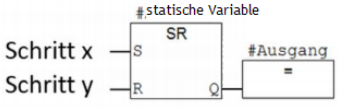
2. Möglichkeit:

Der jeweilige Ausgang wird in mehreren kontinuierlich wirkenden Aktionen verwendet, d.h. er ist in mehreren, nicht aufeinander folgenden Schritten aktiv. Da die jeweiligen Schritte den Ausgang unabhängig voneinander ansteuern können, werden sie dem Ausgang über eine Oder-Verknüpfung zugewiesen.



3. Möglichkeit:

Der Ausgang wird in einer speichernden Aktion verwendet, d.h. er ist über mehrere, aufeinander folgende Schritte aktiv, bzw. wird von einem Schritt ein- und von einem anderen wieder ausgeschaltet.
Diese Problematik kann natürlich auch mit der Möglichkeit 2 gelöst werden, würde so aber bei sehr vielen aufeinander folgenden Schritten sehr aufwendig. Wesentlich kürzer ist die Verwendung eines SR-Gliedes, das von dem einschaltenden Schritt gesetzt und von dem ausschaltenden Schritt rückgesetzt wird.

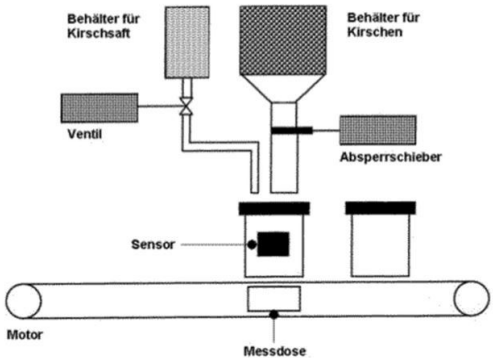


Übungsaufgaben: Programmierung von Ablaufsteuerungen

In Anlehnung an die Leitaufgabe „Prägemaschine“ können Sie nun folgende Übungsaufgaben bearbeiten.

(18) Kirschabfüllanlage

Auf einem Transportband, das durch den Motor M1 angetrieben wird, werden Gläser unter einem Behälter bewegt. Der Sensor B01 meldet ein vorhandenes Glas und
schaltet den Bandmotor ab. Gleichzeitig wird der Absperrschieber Y1 geöffnet. Dadurch fallen Kirschen in das Glas. Das Gewicht des Glases wird mit einer Messdose erfasst.
Der Bereich des Spannungsausganges der Messdose liegt bei 0V..10V. Dieser Spannungsbereich

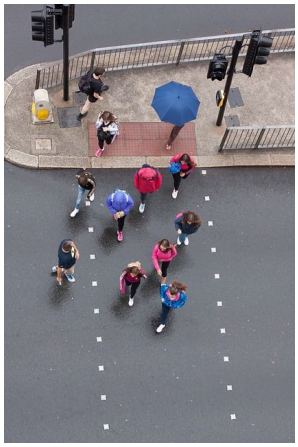


Kirschabfüllanlage

entspricht dem Gewicht von 0g..500g. Wird die Gewichtsmarke von 250g erreicht, soll der Absperrschieber geschlossen und das Ventil Y2 geöffnet werden. Dadurch gelangt Kirschsaft in das mit 250g Kirschen gefüllte Glas. Das Ventil soll schließen, wenn die zweite Gewichtsmarke von 450g erreicht ist. Zur Vereinfachung können die Gewichtsmarken als zwei boolesche Eingänge der SPS angenommen werden (Schwellwertschalter B002 und B003). Das gefüllte Glas soll nach einer Wartezeit von 750ms durch Transportband zur Verpackung weiter transportiert werden. Der Ablauf für dieses Glas endet, wenn die Lichtschranke B004 unterbrochen wird (0-Signal). Die gesamte Anlage wird durch den Austaster S2 abgeschaltet und durch einen Eintaster S1 eingeschaltet. Nach dem Ausschalten der Anlage wird das letzte Glas zu Ende befüllt und abzutransportieren. Eine Betriebsanzeige P1 soll den Einschaltzustand anzeigen.

(19) Fußgängerampel

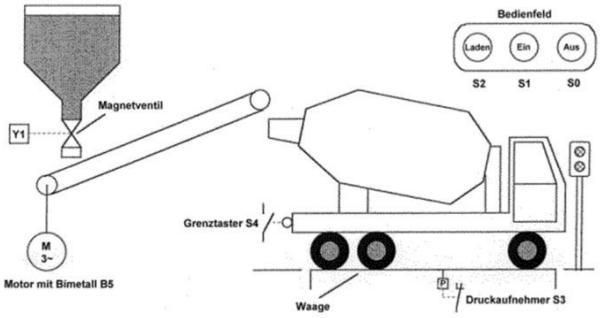
Gesteuert werden soll eine Fußgängerampel vor einer Schule. Die Ampel schaltet nur auf Bedarf für Fußgänger grün. Mit dem Taster S1 wird die Ampelanlage eingeschalten. Nun leuchten die Fußgängerampeln beide rot (P1 und P3). Die Verkehrsampeln leuchten beide grün (P7 und P10). Mit dem Taster S0 wird die Ampelanlage abgeschaltet.
Wird nun ein Taster der Fußgängerampeln (S2 oder S3) betätigt, schaltet die Ampelanlage in folgender Reihenfolge um:


Fußgängerampel

- 3 Sekunden nach Tasterbetätigung schalten die Verkehrsampeln von Grün nach Gelb (P7 und P10 → P6 und P9).
- Nach einer Gelbphase von 1,8 Sekunden wechseln die Verkehrsampeln auf Rot (P6 und P9 → P5 und P8).
- Nach einer zusätzlichen Sicherheitswartepause von 2 Sekunden schalten nun die Fußgängerampeln für 5 Sekunden von Rot nach Grün (P1 und P3 → P2 und P4)
- Nach weiteren 2 Sekunden schalten die Verkehrsampeln mit einer Umschaltdauer von 1,5 Sekunden von
  Rot über Rot/Gelb nach Grün zurück (P5 und P8 → P5, P8, P6 und P9 → P7 und P10)

(20) Betonmisch- und Verladeanlage

Die Hauptbestandteile der Anlage sind: das Magnetventil des Siloschiebers Y1, das Förderband und die Waage. Zur Orientierung für den Lkw-Fahrer steht noch eine Ampel zur Verfügung.
Mit dem Taster S1 wird die Anlage betriebsbereit geschaltet, d. h. die Waage ist betriebsbereit und die Ampel leuchtet „Grün“ (P1). Der Taster S0 (NOT-AUS) und das Auslösen des Bimetalls B5 schaltet



Betonmisch- und Verladeanlage

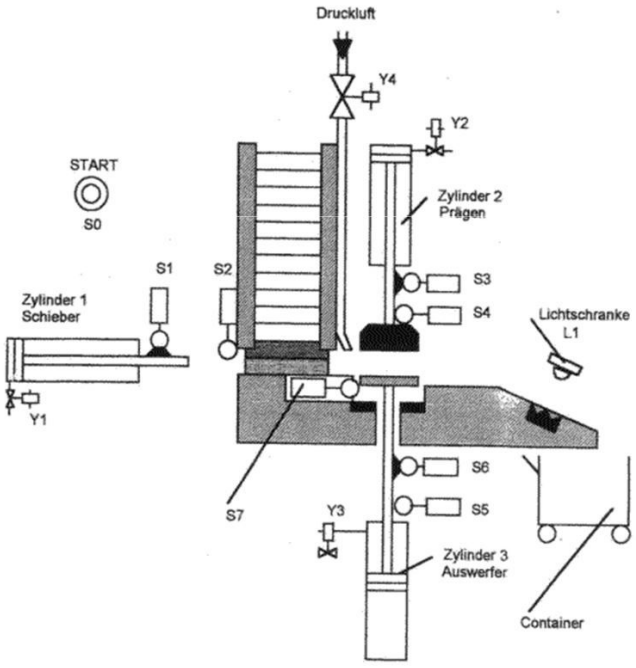
die gesamte Anlage aus. Fährt ein LKW auf die Waage und der Grenztaster S4 wird angefahren, schaltet die Ampel auf „Rot“ (P2) und es kann mit dem Ladetaster S2 der Ladevorgang eingeleitet werden. Nach Betätigung von S2 schaltet zuerst das Förderband M1 ein. Nach 3 Sekunden öffnet der Schieber Y1. Spricht der Druckschalter S3 der Waage an, schließt der Schieber Y1 und nach
einer Verzögerung von 5 Sekunden schaltet das Band ab, die Ampel schaltet auf „Grün“. Der LKW kann die Waage verlassen.
Ein neuer Ladevorgang kann unter den bekannten Bedingungen eingeleitet werden.

(21) Prägemaschine

Mit einer Prägemaschine soll auf Werkstücken eine Kennzeichnung eingeprägt werden.

Funktionsablauf:

Nach Betätigung der Starttaste (S0) schiebt ein Schieber (Y1) ein Werkstück aus dem Magazin auf die Platte des Auswerfers. Der Auswerfer (Y3) senkt ab. Wenn die Prägeform belegt ist (S7=1), stößt der Prägestempel (Y2) abwärts. Nach einer Wartezeit von 2 Sekunden geht der Prägestempel wieder in seine Ausgangsstellung. Nach dem Prägevorgang stößt der Auswerfer (Y3) das fertige Teil aus der Form. Ist der Auswerfer ausgefahren (S6=1), dann öffnet Y4. Mit dem Luftstrom aus

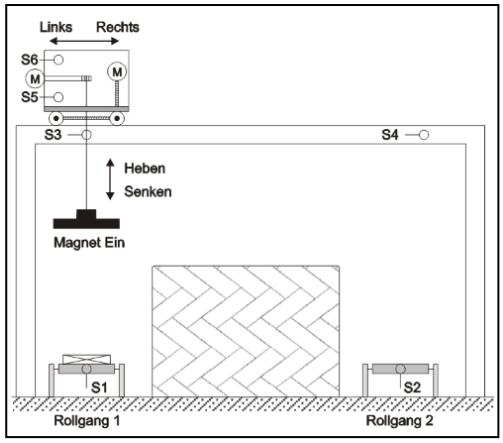


Prägemaschine

der Luftdüse (Y4) wird das Werkstück in den Auffangbehälter geblasen. Eine Lichtschranke L1 spricht an, wenn das Teil in den Auffangbehälter fällt und schließt die Luftdüse (Y4). Anschließend kann der nächste Programmvorgang beginnen.
Anmerkung: Es findet keine automatische Wiederholung des Vorgangs statt.

(22) Umsatzbühne

Mit Hilfe einer Umsetzbühne sollen Werkstücke vom Rollgang 1 auf den Rollgang 2 umgesetzt werden. Wenn ein Werkstück auf Rollgang 1 eingelaufen ist, wird dies durch S1 erfasst. Daraufhin muss der Magnet abgesenkt werden. Der Sensor S5 meldet die untere Magnetposition. Dann kann der Magnet eingeschaltet und nach einer Verzögerungszeit von 2 Sekunden mit dem Werkstück bis in die obere Hubposition (S6) angehoben werden. Anschließend wird der Kran zur Rollenbahn 2 verfahren. Nach Erreichen der rechten Endlage (S4) wird der Magnet gesenkt. Das Erreichen der unteren Magnetposition wird wieder durch S5 gemeldet. Der Magnet wird abgeschaltet. Wenn der Sensor S2 „Werkstück abgesetzt“ meldet, kann der Magnet nach einer Verzögerungszeit von 2 Sekunden bis zur oberen Hubposition angehoben werden.



Anschließend wird die Laufkatze nach links bis zur Endlage (S3) verfahren. Dieser Steuerungsablauf soll nur ablaufen, wenn der Kran in der Grundstellung steht. Dies ist gegeben, wenn die Laufkatze links steht, der Magnet oben und ausgeschaltet ist. Der Antrieb für den Hub wird über die zwei Ausgänge "Heben" und "Senken" angesteuert. Der Längsfahrantrieb wird über die zwei Ausgänge "Rechtslauf" und "Linkslauf" angesteuert. Der Hubmagnet wird über einen Ausgang "Magnet Ein" aktiviert.

(23) Bohrmaschine

Aufbau:

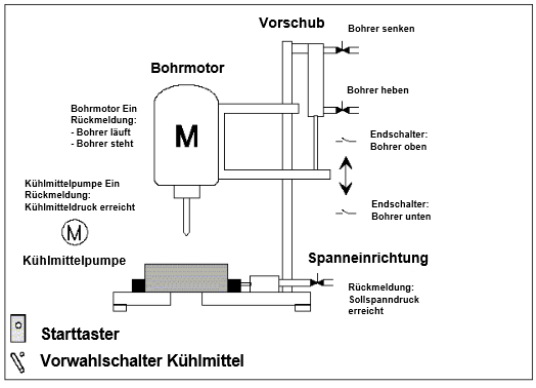
- Bohrmotor (M1) verfügt über eine Drehrichtung
- zwei getrennte Rückmeldungen für Bohrmotor: Bohrer läuft (B1) , Bohrer steht (B2) (jeweils 1-Signal)
- Starttaster (S1) für automatischen Bohrvorgang
- Vorwahlschalter (S2) mit/ohne Kühlmittel (mit Kühlmittel =1-Signal)
- Kühlmittelpumpe (M2) mit Rückmeldung (B3): Kühlmitteldruck erreicht = 1-Signal
- hydraulische Spanneinrichtung (Y1) mit Rückmeldung (B4): Sollspanndruck erreicht = 1-Signal (der Öldruck muss während des gesamten Bohrvorganges anstehen)
- hydraulischer Vorschub Bohrer heben (Y2) / senken (Y3)
- zwei Rückmeldungen für Bohrerposition: Bohrer oben (B5), Bohrer unten (B6) (jeweils 1-Signal)

Grundzustand:

- Bohrmotor und Kühlmittelpumpe aus
- Vorschub für Bohrer befindet sich in oberer Position
- Kein Werkstück gespannt Bohrvorgang

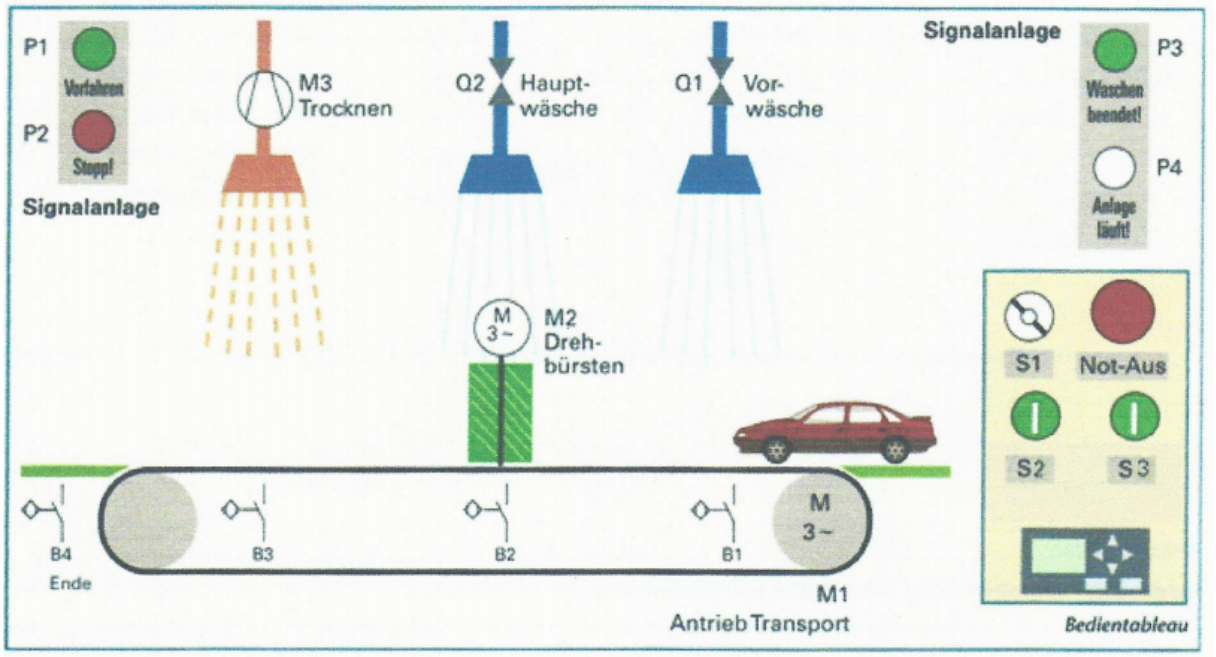
Bohrvorgang:

- Werkstück einlegen (manuell)
- materialabhängig Vorwahlschalter für Kühlmittelzuschaltung betätigen (manuell)
- automatischer Bohrvorgang wird manuell über Starttaster gestartet
- Werkstück mit Sollspanndruck einspannen
- Kühlmittelpumpe zuschalten (je nach Vorwahl)
- Bohrer über Vorschub bis zur unteren Sollposition absenken (bohren)
- in unterer Sollposition weitere 0,5s bohren (Ölsäule hält die Bohrmaschine in Position)
- Bohrer über Vorschub bis zur oberen Sollposition anheben
- Werkstück lösen, Bohrmotor und Kühlmittelpumpe ausschalten
- Werkstück entnehmen (manuell)



(24) Waschanlage

Technologieschema



Funktionsbeschreibung

Über den Schlüsselschalter S1 wird die Anlage betriebsbereit geschaltet. Die Lampe P1 „Vorfahren“ signalisiert, dass das Auto weiter vorgefahren werden kann. Wird der Sensor B1 erreicht, wird dem Fahrer durch P2 „Stopp“ signalisiert, dass die Startposition erreicht ist. Durch die Taster S2 wird der Waschvorgang gestartet.

Der Antrieb M1 zieht das Auto kontinuierlich durch die Waschanlage. Die Sensoren B1 bis B3 (Schließer) starten jeweils die Programmpunkte Vorwäsche, Hauptwäsche und Trocknen. Während des gesamten Waschvorgangs leuchtet P4 „Anlage läuft“ im Sekundentakt. P2 leuchtet ebenfalls während des gesamten Waschvorganges.
Wird der Sensor B4 erreicht, signalisiert P3, dass der Waschvorgang beendet ist. P1 leuchtet wieder auf.

Löst einer der Motorschutzschalter (Öffner) der Motoren M1 bis M3 aus, so wird die Anlage gestoppt und die Lampen P2 und P3 blinken im Sekundentakt. Dasselbe passiert, wenn der Not-Aus-Schalter S0 betätigt wird. Um die Anlage wieder in Betrieb zu nehmen, müssen die Taster S2 und S3 gleichzeitig betätigt werden. Dadurch wird das Fahrzeug an das Ende der Waschanlage vorgefahren, ohne dass die jeweiligen Programmpunkte abgearbeitet werden.
Ein neuer Waschvorgang kann immer nur dann starten, wenn der Sensor B4 signalisiert, dass das Fahrzeug aus der Waschanlage gefahren wurde.