Technische Universität Dortmund

Fakultät Maschinenbau

Fachgebiet IT in Produktion und Logistik

Univ.-Prof. Dr.-Ing. Markus Rabe

Projektarbeit

Entwicklung neuer Übungsaufgaben auf dem Gebiet der grafischen Programmiersprachen für speicherprogrammierbare Steuerungen zum Einsatz im Lehrbetrieb

Name:	Daniel Darmograj, Lucas Hupe
Matrikelnummer:	184457, 185433
Ausgegeben am:	09.07.2020
Eingereicht am:	11.02.2021
Betreuer:	Dr.-Ing. Dipl.-Inform. Anna Antonia Scheidler, M.Sc. Florian Hochkamp

Inhaltsverzeichnis

Abkürzungsverzeichnis

AS Ablaufsprache

AWL Anweisungsliste

FBS Funktionsbausteinsprache

KOP Kontaktplan

SPS Speicherprogrammierbare Steuerung

ST Strukturierter Text

Abbildungsverzeichnis

1 Einleitung

In den letzten 30 Jahren hat die Automatisierung von Produktionsanlagen zunehmend an Bedeutung gewonnen [Bernstein 2018]. Durch den Einsatz von speicherprogrammierbaren Steuerungen (SPS) wurde eine niedrige Einstiegsschwelle, mithilfe einer grafischen Darstellung mit Kontaktsymbolen, für das vorhandene Fachpersonal ermöglicht, um den rasch steigenden Automatisierungsgrad zu erfüllen [Bernstein 2018]. Auch in der heutigen Zeit begegnet man SPS, zum Teil ohne es zu bemerken, im öffentlichen Leben, wie z.B. bei der Ampel- und der Fahrstuhlsteuerung [Bauernhansl et al. 2017]. Weiterhin bleibt SPS innerhalb des Zukunftsprojekts Industrie 4.0 ein wichtiger Bestandteil, welcher nun mit dem Gebiet der Informatationstechnik verbunden werden muss [Brecht et al. 2018].

Ziel der fachwissenschaftlichen Projektarbeit ist die Erstellung von Übungsaufgaben zum Thema der grafischen Programmiersprachen innerhalb der SPS zum Einsatz im Lehrbetrieb. Diese Übungsaufgaben sollen die grafischen Programmiersprachen, Kontakt- und Funktionsplan, abdecken. Darüber hinaus sollen, mithilfe einer Moodle-Lerneinheit, die Grundlagen der grafischen Programmiersprachen vermittelt und das Wissen, durch zusätzliche Aufgaben, vertieft werden.

Die Projektarbeit ist in mehrere Abschnitte unterteilt. Zunächst findet im zweiten Kapitel eine begriffliche Einordnung der SPS statt. Zusätzlich werden die grafischen Programmiersprachen von den weiteren Teilbereichen, der textlichen Sprachen und der Mischform, abgegrenzt. Im dritten Kapitel werden bereits bestehende Fallbeispiele aus der Literatur beschrieben und mithilfe von aufgestellten Kriterien bewertet. Dadurch lässt sich im weiteren Verlauf beurteilen, welche der vorgestellten Übungsaufgaben sich für den Einsatz im Lehrbetrieb bzw. in der Moodle-Lerneinheit eignen. Darauf aufbauend werden im vierten Kapitel die Vorgehensweise und die Lösung der geeigneten Übungsaufgaben aus dem dritten Kapitel vorgestellt und eigene Übungsaufgaben erarbeitet. Ergänzend zu den Übungsaufgaben für den Einsatz im Lehrbetrieb wird im fünften Kapitel die Moodle-Lerneinheit beschrieben, welche den Studierenden eine Möglichkeit zum interaktiven Lernen bietet und zum besseren Verständnis der grafischen Programmiersprachen beiträgt. Die Projektarbeit endet mit einer Zusammenfassung der erarbeiteten Inhalte sowie einem kurzen Ausblick.

2 Stand der Technik auf dem Gebiet der speicherprogrammierbaren Steuerung [LH/DD]

In diesem Kapitel werden zuerst die Grundlagen für die Projektarbeit gelegt und die Definition der SPS erläutert. Außerdem werden auf die Veränderungen, die aufgrund der SPS-Norm entstanden sind, eingegangen und die Auswirkungen auf die Programmierungen erläutert. Anschließend werden die verschiedenen Programmiermöglichkeiten innerhalb der SPS erläutert.

2.1 Definition der speicherprogrammierbaren Steuerung [LH]

Eine speicherprogrammierbare Steuerung ist ein digital arbeitendes elektronisches System, welches meist für den Einsatz in industriellen Umgebungen verwendet wird. [Benda 1986]. Dabei wird durch einen internen programmierbaren Speicher die anwenderorientierte Steuerungsanweisungen zur Implementierung spezifischer Funktionen, angewandt, um durch digitale oder analoge Eingangs- und Ausgangssignale verschiedenen Arten von Maschinen und Prozessen steuern zu können [IEC 61131-3 2003]. Dabei benötigt man eine elektronische Steuerung mit einer internen Verdrahtung (Hardware) und ein, an die zu steuernde Maschine oder Anlage, angepasstes Programm, welches den gewünschten Ablauf festlegt (Software) [Benda 1986]. Weiterhin muss es ein Ein- und Ausgabegerät geben [Benda 1986]. Als Eingabegeräte zählen Grenztaster, Befehlsgeräte oder auch analoge Stromgeber [Wellenreuther & Zastrow 2015]. Als Ausgabegerät kann man unter anderem Kupplungen oder Lampen wählen [Benda 1986]. Eine SPS wird erst durch das Zusammenspiel des Programms mit den Ein- und Ausgabegeräten zu einer funktionalen Steuerung [Benda 1986]. Da das Programmieren von Steuerungen und Regelungen ein sehr individueller Prozess, abhängig vom Anwendungsbereich, ist, sollte diesem die SPS-Programmierungsnorm DIN EN 6113-3 zu Grunde liegen [Wellenreuther & Zastrow 2015]. Die Programmierungsnorm hat eine rationale und herstellungsunabhängige Programmerstellung zum Ziel [Wellenreuther & Zastrow 2015]. Dabei soll besonders auf die Wiederverwendbarkeit geachtet werden, so dass bei zunehmender Komplexität der Automatisierungsaufgabe die Entwicklungskosten der Software nicht unverhältnismäßig steigen [Wellenreuther & Zastrow 2015]. Daher empfehlen Wellenreuther und Zastrow, dass es in erster Linie darum geht, eine bibliotheksfähige Programmlösung zu finden, welche anschließend auch weiterverwendet werden kann. Dafür dürfen die zur Lösung programmierten SPS-Bausteine

soweit wie möglich keine anlagenspezifischen Bedingungen besitzen, wie zum Beispiel SPS-Operanden für Ausgänge, Eingänge oder Zeitglieder [Wellenreuther & Zastrow 2015]. Vielmehr sollen erst bei der Zusammenfügung zu einem Hauptprogramm, die anlagenspezifischen SPS-Operanden eingegeben werden [Wellenreuther & Zastrow 2015]. Die SPS-Norm DIN EN 61131-3 hat einige grundlegende Konzepte, wie beispielsweise das Programmorganisations- oder das Taskkonzept infolge des neuen Programmier-Standards beeinflusst [Wellenreuther & Zastrow 2015]. Auch das Datentyp- und Variablenkonzept ist durch die verbindlichen Definitionen für Datentypen, Konstanten und Variablen einheitlicher [IEC 61131-3 2003]. Da man in der klassischen SPS-Programmierung vorher ungebundenes operieren gewohnt war, brachte die Vereinheitlichung große Veränderungen in der Arbeitsweise mit sich [Wellenreuther & Zastrow 2015]. Das bereits zuvor erwähnte Programmorganisationskonzept ist ein hierarchisch gegliedertes System, bestehend aus dem Hauptprogrammtyp (PROGRAMM) und zwei Unterprogrammtypen [Plaßmann & Schulz 2016]. Die Hauptprogramme besitzen dabei Zugriffsmöglichkeiten auf E/A-Operanden. Ein Unterprogramm besitzt die Gedächtnisfunktion (FUNCTION_BLOCK), das andere Unterprogramm nicht (FUNCTION) [Wellenreuther & Zastrow 2015]. Eine Gedächtnisfunktion beinhaltet die Möglichkeit über einen Programmzyklus hinweg Deklarationen von internen Zustandsvariablen speichern zu können [Wellenreuther & Zastrow 2015]. Die bereits zuvor beschriebene Datentyp- und Variablenkonzeption in Verbindung mit dem Programmorganisationskonzept ermöglicht die Erhöhung der Wiederverwendbarkeit der Bausteine für eine spätere Anwendung, da diese eine bibliotheksfertige Programmlösung darstellen [Wellenreuther & Zastrow 2015]. Daher ist die neue Anwendungsprogrammierung anspruchsvoller als die herkömmliche SPS-Programmierung [Plaßmann & Schulz 2016]. Des Weiteren wurde das Konzept der Tasks durch die SPS-Programmierungsnorm stark beeinflusst. Durch diese Tasks erfolgt die Ausführungssteuerung der Programmorganisationseinheit [Wellenreuther & Zastrow 2015]. Das Taskkonzept muss über die Fähigkeit einer priorisierbaren, zeitzyklischen und ereignisorientierten Programmsteuerung verfügen [Wellenreuther & Zastrow 2015]. Dadurch kann der Programmierer nun eine zyklisch arbeitende Steuerung und ein getaktetes Regelungsprogramm mit konstanter Abtaktzeit entwerfen [Wellenreuther & Zastrow 2015]. Die SPS-Norm beeinflusste außerdem das Fachsprachenkonzept, welches aus fünf Sprachen besteht [Wellenreuther & Zastrow 2015].

Im Folgenden werden die fünf Sprachen detailliert erläutert, um die Tragweite der Standarisierung aufzuzeigen.

2.2 Programmiermöglichkeiten innerhalb der speicherprogrammierbaren Steuerung [DD]

Im Allgemeinen bietet die SPS nach der Programmiernorm DIN EN 61131-3 fünf Programmiersprachen. Diese lassen sich, wie in Abbildung 1 dargestellt, in Textsprachen und grafischen Sprachen unterteilen. Während sich innerhalb der Textsprachen die Anweisungsliste (AWL) und der strukturierte Text (ST) einordnen lassen, zählen die Funktionsbausteinsprache (FBS) und der Kontaktplan (KOP) zu den grafischen Sprachen. Die Ablaufsprache (AS) kann als Mischform der Sprachen gesehen werden, da diese sowohl grafische als auch textliche Elemente beinhaltet (siehe Abbildung 1).

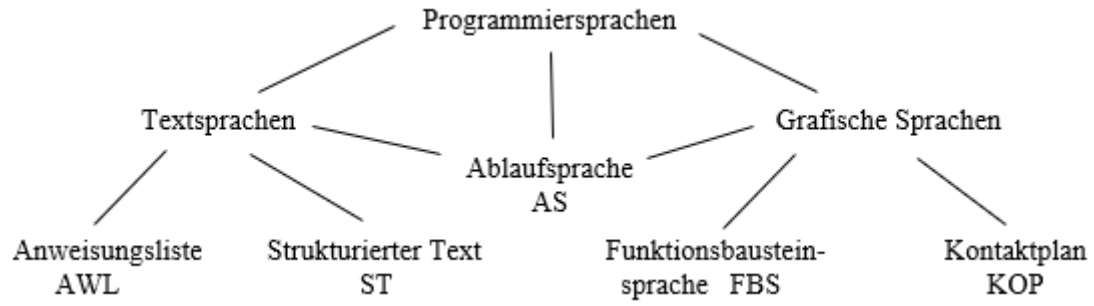


Abbildung 1: Programmiersprachen gemäß DIN EN 61131-3 [Wellenreuther & Zastrow 2015, S. 19]

Im Folgenden wird auf die Unterschiede der einzelnen Sprachen untereinander eingegangen.

2.2.1 Textsprachen

Die Textsprachen setzen sich aus einer Aufreihung von Anweisungen zusammen [John & Tiegelkamp 2009]. Besonders hervorzuheben ist, dass sie eine hohe Programmierfreiheit zur Lösung der Aufgaben bieten, was einen großen Vorteil gegenüber den grafischen Sprachen darstellt [ten Hompel 2018]. Allerdings ist nachteilig zu erwähnen, dass die jeweiligen Textsprachen gegenüber den grafischen Sprachen eine gewisse Komplexität aufweisen können, wodurch sie eher von erfahreneren Programmierern genutzt werden [Rabe 2019b]. Die Anwender haben hier die Wahl zwischen der AWL und dem ST.

Die AWL ist eine maschinennahe Programmiersprache und kann nach der Norm DIN EN 61131-3 universell eingesetzt werden [Pickhardt 2000]. Dadurch wird die AWL häufig als Zwischensprache verwendet, um die restlichen vier Sprachen abzubilden [Pickhardt 2000]. Die

AWL beinhaltet eine zeilenorientierte Sprache [Lerch 2016]. Dies bedeutet, dass eine Anweisung genau in einer Zeile vorzufinden ist und das Programm nach der Reihenfolge der Anweisungen gelesen und bearbeitet wird [John & Tiegelkamp 2009]. Neben den zuvor erwähnten Vor- und Nachteilen der Textsprachen besitzt die AWL spezifische Vor- und Nachteile gegenüber dem folgenden ST. Während der geringere Ressourcenverbrauch nach der Kompilierung des Programms gegenüber dem ST einen Vorteil darstellt, ist die Unübersichtlichkeit des Programms mit zunehmender Aufgabenkomplexität nachteilig zu erwähnen [ten Hompel 2018]. In Abbildung 2 ist beispielhaft der Aufbau einer AWL-Anweisung abgebildet, welche im Folgenden erläutert wird.

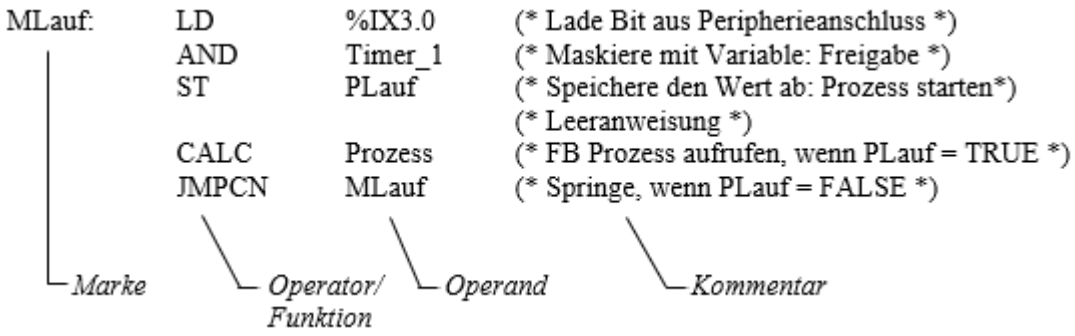


Abbildung 2: Beispiel für eine AWL-Anweisung [John & Tiegelkamp 2009, S. 104]

Wie in Abbildung 2 ersichtlich, werden (Sprung-) Marken verwendet, wenn die Anweisungszeile von einer anderen Anweisung aufgegriffen wird. Damit muss diese Anweisung nicht wiederholt werden, sondern kann mit einer Sprungmarke auf die jeweilige Anweisungszeile springen [John & Tiegelkamp 2009]. Operatoren beschreiben die jeweilige Prozedur, welche in der Anweisung ausgeführt werden soll (siehe Abbildung 2). Beispiele für Operatoren anhand der booleschen Wahrheitstabelle sind: AND, OR, XOR, NOT. Operanden sind vom Programmierer definierte Variablen und geben an was die Operatoren verwenden sollen, was man aus Abbildung 2 entnehmen kann. Ein Beispiel für einen Operanden ist, wie in Abbildung 2 abgebildet, PLauf. Dieser Operand beschreibt das Starten des Prozesses. Kommentare dienen dem Anwender lediglich zum Verständnis der Anweisung (siehe Abbildung 2). Die Bausteine, Sprungmarke und Kommentar, sind optional anzusehen und können deshalb entfallen [Pickhardt 2000].

Der ST ist eine weitere textuelle Programmiersprache und wird auch als Hochsprache bezeichnet, da dieser die meisten Ähnlichkeiten zu bekannten Programmiersprachen, wie beispielsweise PASCAL oder C, aufweist [John & Tiegelkamp 2009]. Im Gegensatz zur AWL kann eine Anweisung über mehrere Zeilen bzw. mehrere Anweisungen innerhalb einer Zeile

beschrieben werden [John & Tiegelkamp 2009]. Dadurch ist es möglich das Programm mithilfe von Anweisungsblöcken übersichtlich aufzubauen, was einen erheblichen Vorteil gegenüber der AWL darstellt [Pickhardt 2000]. Allerdings ist der zuvor erwähnte Nachteil, des höheren Ressourcenverbrauchs gegenüber der AWL, nicht zu vernachlässigen [ten Hompel 2018]. Zwar entfallen die optionalen Sprunganweisungen der AWL in diesem Sinne beim ST, allerdings können diese durch IF-Anweisungen dargestellt werden, was dementsprechend zu keinem Nachteil auf der Programmierebene führt [Pickhardt 2000]. Wie man der Abbildung 3 entnehmen kann, weist ein ST-Programm große Ähnlichkeiten zum AWL-Programm auf. Ein Ausdruck des ST-Programms entspricht sinngemäß einer AWL-Anweisung. Der Unterschied zueinander liegt darin, dass eine ST-Anweisung mehrere Ausdrücke enthalten kann und ein ST-Programm aus mehreren ST-Anweisungen besteht.

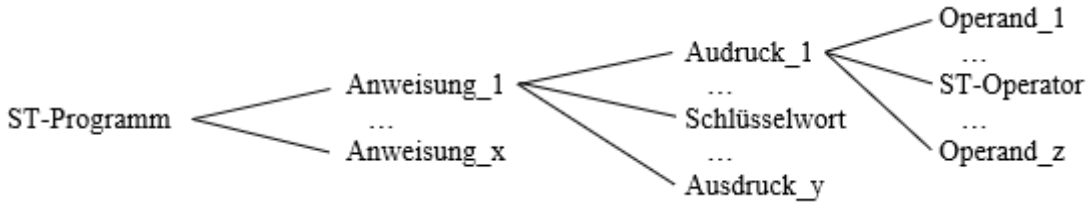


Abbildung 3: Aufbau eines ST-Programms [John & Tiegelkamp 2009, S. 121]

2.2.2 Grafische Sprachen

Die grafischen Sprachen stellen mittels vordefinierter grafischer Elemente eine Möglichkeit zur Lösung der Programmieraufgabe dar. Aufgrund der geringeren Komplexität gegenüber der Textsprachen werden sie häufiger von weniger erfahrenen Programmierern verwendet [Rabe 2019b]. Allerdings besitzen die grafischen Sprachen nur einen eingeschränkten Funktionsumfang, da man nur die vordefinierten Elemente nutzen kann, was problematisch bei komplexeren Programmieraufgaben sein kann [ten Hompel 2018]. Die Unterteilung der grafischen Sprachen erfolgt in die Funktionsbausteinsprache (FBS) und den Kontaktplan (KOP).

Die FBS besitzt ihren Ursprung in der Signalverarbeitung und ist prinzipiell durch ihre grafische Darstellung ohne weiteres nachvollziehbar [Pickhardt 2000]. Grundsätzlich besteht die FBS, wie man der Abbildung 4 entnehmen kann, aus Funktionen, Funktionsblöcken und -bausteinen, welche durch Rechtecke dargestellt werden. Die Verbindungen der Funktionen untereinander, also die Weitergabe der Variablen, werden mittels horizontaler und vertikaler Linien veranschaulicht (siehe Abbildung 4).

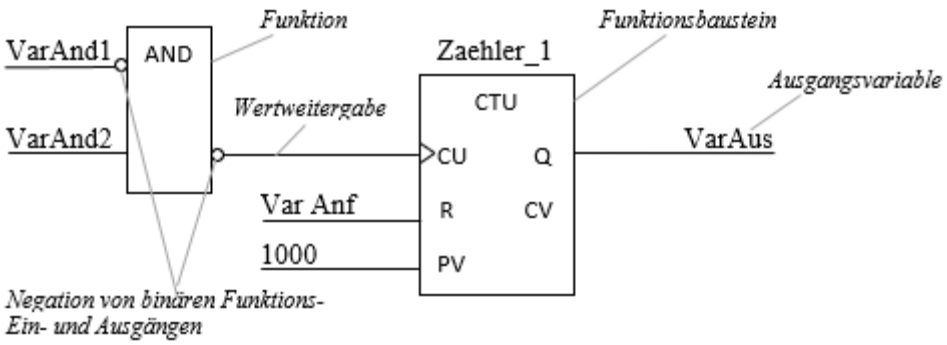


Abbildung 4: Elemente eines FBS-Netzwerks [John & Tiegelkamp 2009, S. 141]

Der Vorteil der gut nachvollziehbaren grafischen Darstellung der FBS kann bei komplizierten Programmieraufgaben schwinden, da es durch die Vielzahl der Funktionen und Verbindungen unübersichtlich werden kann [Rabe 2019b].

Der KOP dagegen ähnelt, wie man den Abbildungen 5 und 6 entnehmen kann, den Stromlaufplänen, welche aus dem Gebiet der elektromechanischen Relaissysteme stammen [Pickhardt 2000]. Prinzipiell bearbeitet der KOP boolesche Signale, was die Komplexität reduziert und wodurch die Funktionalität auch von unerfahreneren Programmierern erfasst werden kann [John & Tiegelkamp 2009].

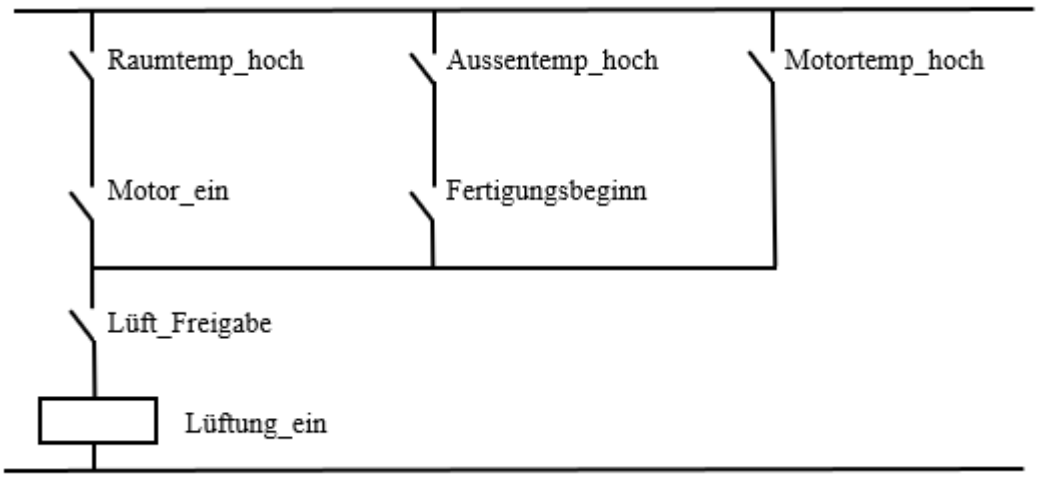


Abbildung 5: Beispiel für einen Stromlaufplan [Grötsch 2004, S. 38]

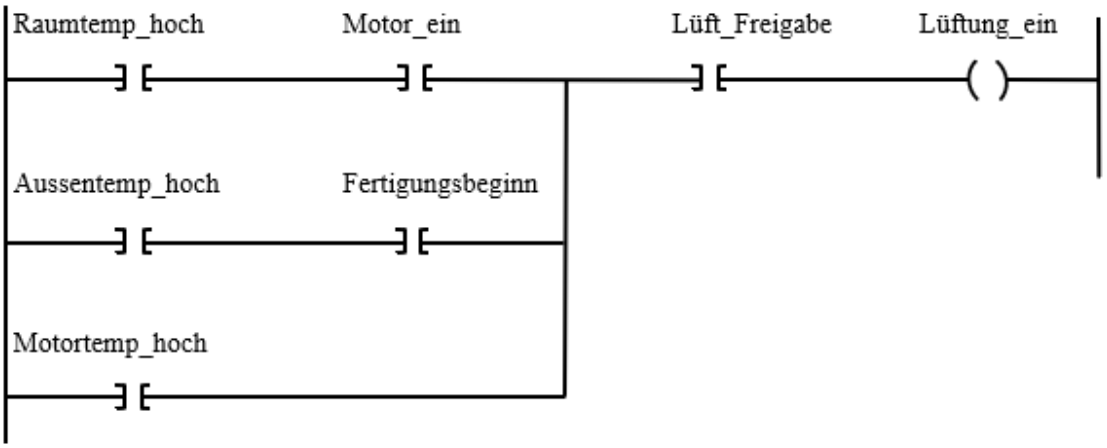


Abbildung 6: Kontaktplan zum Stromlaufplan [Grötsch 2004, S. 39]

Wie man den Abbildungen 5 und 6 entnehmen kann, wird ein KOP an seinen Rändern durch sogenannte Stromschienen begrenzt, wobei der Strom von links nach rechts fließt. Er besteht aus fünf Elementen [Pickhardt 2000]. Das erste Element beschreibt die Verbindungen [[John & Tiegelkamp 2009]. Während horizontale Linien den Wert FALSE oder TRUE weiterreichen, verknüpfen vertikale Linien die horizontalen Linien mit einem OR und reichen diesen Wert wiederum an folgende horizontale Linien weiter [John & Tiegelkamp 2009]. Kontakte stellen das zweite Element dar [John & Tiegelkamp 2009]. Sie verknüpfen den Wert der eingehenden Verbindung mit dem Wert der zugewiesenen Variablen und geben das Ergebnis weiter [Pickhardt 2000]. Spulen dienen als drittes Element der Zuweisung von Werten an Variablen vom Datentyp BOOL [Pickhardt 2000]. Folgend gilt die Ausführungssteuerung als viertes Element, mit dem man Sprungbefehle realisieren kann [Pickhardt 2000]. Zuletzt zählt der Aufruf von Funktionen und Funktionsbausteinen als fünftes Element, welche ebenfalls in einem KOP-Programm aufgerufen werden können [Pickhardt 2000].

2.2.3 Mischform der Sprachen: Ablaufsprache

Die Ablaufsprache (AS) ist eine Mischform aus den textuellen und grafischen Programmiersprachen. Im Allgemeinen nutzt der Anwender die grafische Variante der AS, da somit die Zusammenhänge deutlicher werden [John & Tiegelkamp 2009]. Die Grundidee der AS liegt darin den zu automatisierenden Prozess in einzelne Schritte zu unterteilen, sodass ein schrittweises Zustandsverhalten entsteht [Pickhardt 2000]. Dementsprechend eignen sich Prozesse, wie zum Beispiel die Steuerung einer Waschmaschine, Ampel oder eines chemischen Mischprozesses besonders gut, da innerhalb dieser Prozesse die Schritte der einzelnen Phasen klar

bestimmt sind [John & Tiegelkamp 2009]. Der Abbildung 7 ist zu entnehmen, dass eine AS aus zwei Grundelementen besteht. Das erste Element ist der Schritt, welcher durch ein Rechteck dargestellt wird [John & Tiegelkamp 2009]. Jeder Schritt erhält automatisch einen Merker, um den Zustand des Schritts zu beschreiben und einen Timer, um die verstrichene Zeit seit der Aktivierung zu liefern [Pickhardt 2000]. Die *Rotphase* ist beispielsweise ein Schritt, welcher den Merker *signal.rot* und den Timer *Rotphase.T* enthält. Das zweite Element ist die Transition, welche durch eine horizontale Linie dargestellt wird [Pickhardt 2000]. Jede Transition enthält eine Transitionsbedingung, die auf drei Arten angegeben werden kann [Pickhardt 2000]. Sie kann durch Konnektoren, Transitionsnamen oder auch direkt angegeben werden [Pickhardt 2000]. Die Verbindung zwischen Rot- und Rotgelbphase ist beispielsweise eine Transition mit der Transitionsbedingung, dass der Timer der Rotphase *Rotphase.T* größer oder gleich der *rotzeit* ist. Ein großer Vorteil der AS liegt darin, dass alle Sprachen nach der Norm DIN EN 61131-3 genutzt werden können, um die Transitionsbedingung aufzustellen [Pickhardt 2000].

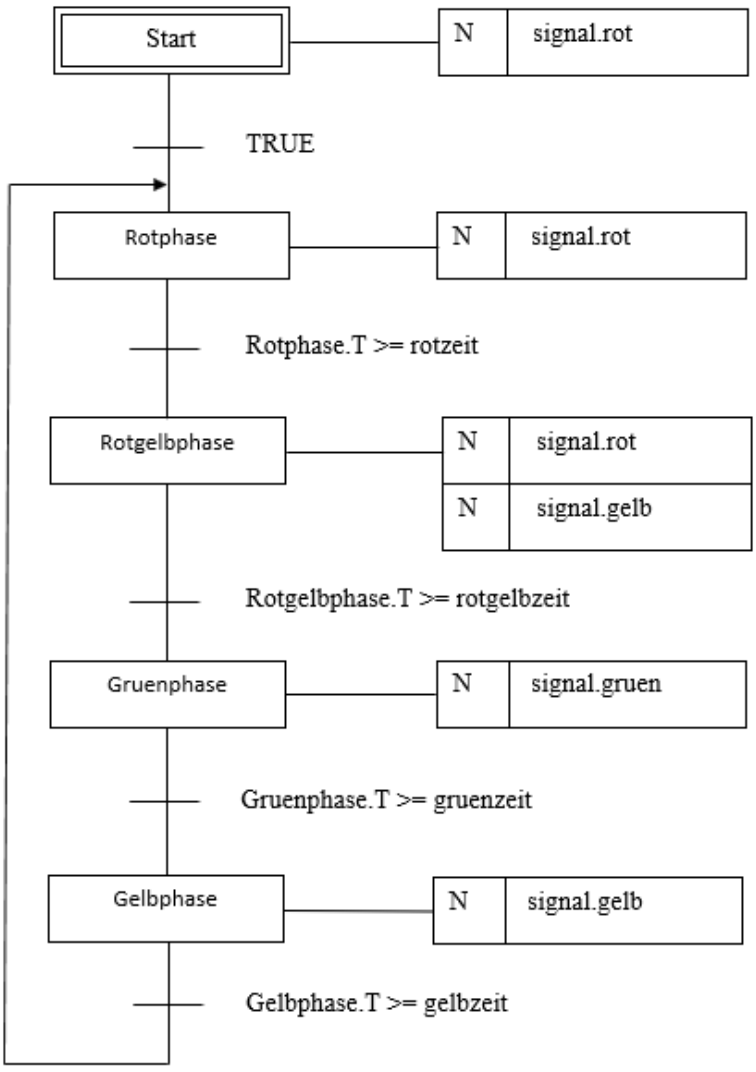


Abbildung 7: AS-Programm für eine Verkehrsampel [Pickhardt 2000, S. 157]

3 Bestehende Übungsaufgaben aus der Literatur [DD/LH]

Im Folgenden werden verschiedene Übungsaufgaben aus der Literatur vorgestellt. Diese umfassen zum einen die Aufgabenstellung und zum anderen die Lösung der jeweiligen Übungsaufgabe. Im Rahmen dieser Projektarbeit wird die Aufgabenstellung so angepasst, dass nur die grafischen Programmiersprachen betrachtet werden. Die vorgestellten Übungsaufgaben werden anschließend, anhand der aufgestellten Kriterien aus Kapitel 3.2, im Kapitel 3.3 bewertet.

3.1 Beschreibung bereits bestehender Übungsaufgaben [DD]

Übungsaufgabe 1: Wahlfreie Schaltstellen [Wellenreuther & Zastrow 2007]

Diese Aufgabe beschäftigt sich mit einem Ablassventil eines Silos, welches von drei Schaltstellen aus wahlweise geöffnet bzw. geschlossen werden kann.

Aufgabenstellung:

Das Ablassventil eines Silos soll von drei Schaltstellen aus (S1, S2 u. S3) über ein 24V-Elektromagnetventil Y wahlweise geöffnet bzw. geschlossen werden können (Wechselschaltungsverhalten von drei Schaltstellen aus). An den Schaltstellen werden einpolige Schalter verwendet.

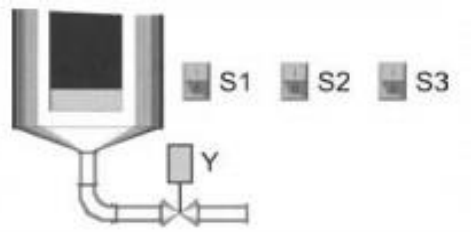


Abbildung 8: Wahlfreie Schaltstellen [Wellenreuther & Zastrow 2007, S.12]

1. Bestimmen Sie mit einer Wahrheitstabelle den Zusammenhang zwischen den Eingängen S1, S2, S3 und dem Ausgang Y.
2. Bestimmen Sie mithilfe der Wahrheitstabelle den Funktionsplan.

Lösung zu Übungsaufgabe 1:

S3	S2	S1	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

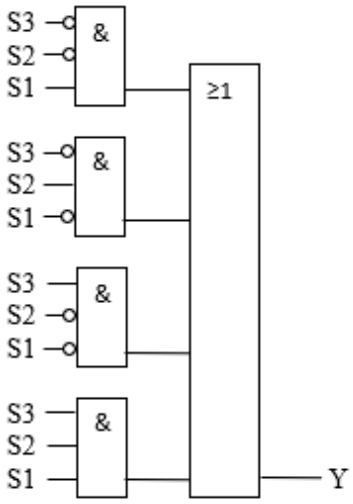


Abbildung 9: Lösung 1.1 (links) und 1.2 (rechts) [Wellenreuther & Zastrow 2007, S.143]

Übungsaufgabe 2: Siloentleerung [Wellenreuther & Zastrow 2007]

Bei dieser Aufgabe soll ein Silo über zwei Pumpen entleert werden. Um die Pumpen zu steuern, wird die Höhe des Füllstands über drei verschiedene Sensoren bestimmt.

Aufgabenstellung:

Der Inhalt eines Silos kann über die Pumpen P1 und P2 entleert werden. Welche der beiden Pumpen bei der Entleerung des Silos eingeschaltet sind, ist abhängig vom Silofüllstand. Befindet sich der Füllstand unterhalb von Sensor S2, ist Pumpe P1 einzuschalten. Liegt der Füllstand zwischen Sensor S2 und Sensor S3, wird die Pumpe P2 eingeschaltet. Bei Füllstand oberhalb von S3 laufen beide Pumpen. Die Entleerung des Silos wird mit dem Schalter S4 ein- und ausgeschaltet. Beim Auftreten eines Sensorfehlers (z.B. S3 meldet und S2 meldet nicht) werden beide Pumpen und eine Störungsanzeige H1 eingeschaltet.

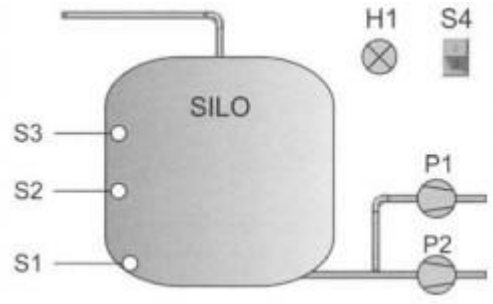


Abbildung 10: Siloentleerung [Wellenreuther & Zastrow 2007, S.12]

1. Bestimmen Sie mit einer Wahrheitstabelle den Zusammenhang zwischen den Eingängen S1, S2, S3 und den Ausgängen PI und P2.
2. Zeichnen Sie mithilfe der Wahrheitstabelle den dazugehörigen Funktionsplan unter Berücksichtigung des Schalters S4.

Lösung Übungsaufgabe 2:

S3	S2	S1	P1	P2	H1	Bemerkungen
0	0	0	1	0	0	
0	0	1	1	0	0	
0	1	0	1	1	1	Sensorfehler
0	1	1	0	1	0	
1	0	0	1	1	1	Sensorfehler
1	0	1	1	1	1	Sensorfehler
1	1	0	1	1	1	Sensorfehler
1	1	1	1	1	0	

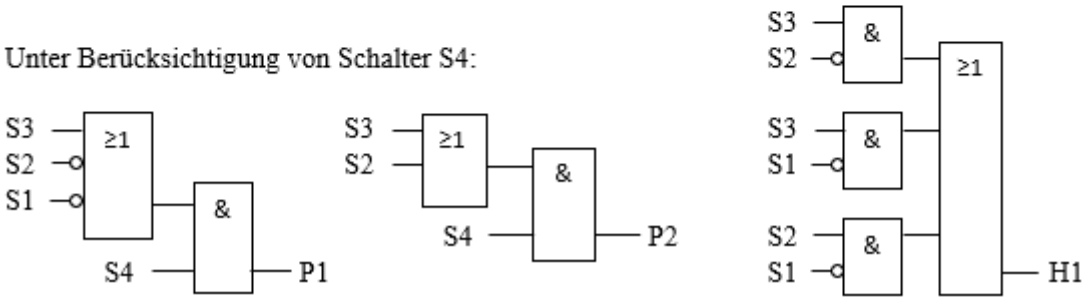


Abbildung 11: Lösung 2.1 (oben) und 2.2 (unten) [Wellenreuther & Zastrow 2007, S.144]

Übungsaufgabe 3: Selbsthaltung [Kaftan 1991]

Bei dieser Aufgabe geht es um die Selbsthaltung in Schützsteuerungen für eine Speicherfunktion. Je nachdem, ob das Ein- oder das Ausschalten vorrangig ist, gibt es zwei Varianten das Schütz auszuschalten.

Aufgabenstellung:

Die in Schützsteuerungen für eine Speicherfunktion übliche Schaltung ist die Selbsthaltung. Parallel zum EIN-Taster ist ein Schließer des Schützes geschaltet, über den nach dem Einschalten Haltestrom für die Schützspule fließt. Der Haltestrom wird unterbrochen und das Schütz ausgeschaltet, sobald der AUS-Taster (Öffner) betätigt wird. Für das Ausschalten des Schützes sind zwei Varianten möglich, je nachdem, ob das Einschalten oder das Ausschalten vorrangig ist.

Schaltung A: Ausschalten hat Vorrang

Für das Ausschalten wird als Geber ein Öffner verwendet (Drahtbruchsicherheit). Kurzzeitiges 1-Signal am Eingang E 1.3 (S1) setzt Ausgang A 3.6 (K1) auf 1-Signal. Schütz K1 zieht an. Kurzzeitiges Signal am Eingang E 1.4 (S2) setzt Ausgang A 3.6 auf 0-Signal zurück. Schütz K1 fällt ab. Für das Setzen müssen beide Eingänge auf 1-Signal abgefragt werden. Im Kontaktplan und im Funktionsplan werden daher E 1.3 und E 1.4 als Schließer programmiert. Es Wird erst die ODER-Bedingung, dann die UND-Bedingung programmiert. Bei gleichzeitiger Betätigung beider Taster ist A 3.6 ausgeschaltet, weil die UND-Bedingung nicht erfüllt ist.

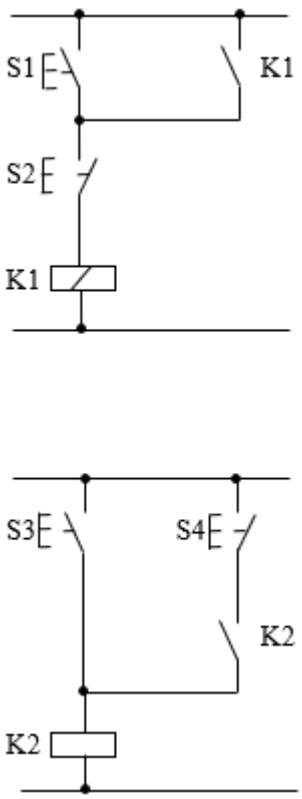


Abbildung 12: Stromlaufplan A (oben) und Stromlaufplan B (unten) [Kaftan 1991, S. 67]

Schaltung B: Einschalten hat Vorrang

Für das Ausschalten wird wieder ein Öffner als Geber verwendet. Kurzzeitiges 1-Signal am Eingang E 1.5 (S3) setzt Ausgang A 3.7 (K2). Schütz K2 zieht an. Kurzzeitiges 0-Signal am Eingang E 1.6 (S4) setzt Ausgang A 3.7 zurück. Schütz K2 fällt ab. Es wird erst die UND-Bedingung, dann die ODER-Bedingung programmiert. Bei gleichzeitiger Betätigung beider Tasten ist der Ausgang A 3.7 eingeschaltet, weil die ODER-Bedingung erfüllt ist.

1. Nach den Stromlaufplänen (Abbildung 12) die jeweiligen Funktions- und Kontaktpläne zeichnen.

Lösung Übungsaufgabe 3:

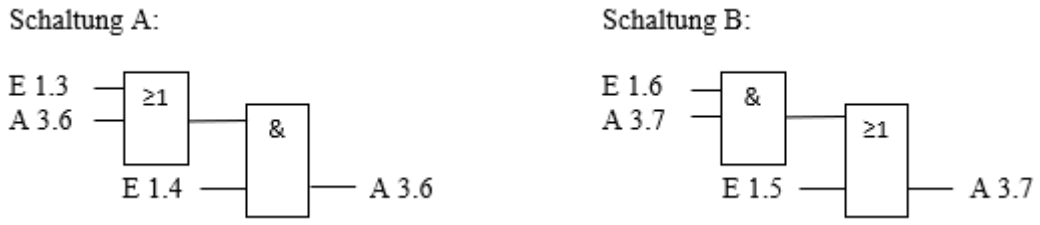
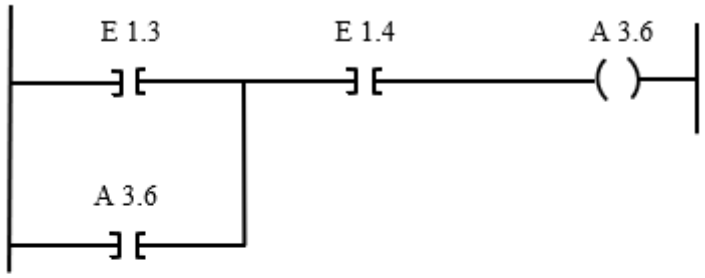


Abbildung 13: Lösung 3.1 Funktionsplan [Kaftan 1991, S. 245]

Schaltung A:



Schaltung B:

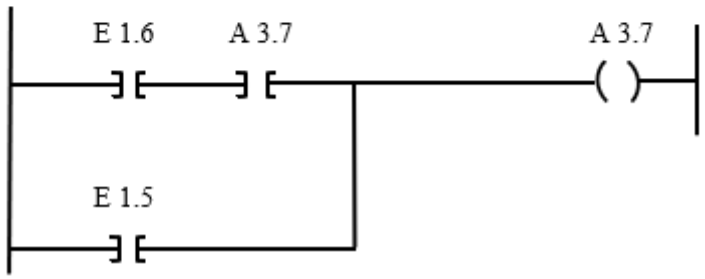


Abbildung 14: Lösung 3.1 Kontaktplan [Kaftan 1991, S. 245]

Übungsaufgabe 4: Funktionsschaltungen [Rabe 2019a]

Bei dieser Aufgabe werden drei verschiedene Systeme vorgestellt, welche man jeweils mit einem Funktionsplan darstellen soll.

Aufgabenstellung:

Erstellen Sie einen Funktionsplan für folgende Systeme

- Eine Maschine soll einen Fräsvorgang ausführen, wenn der Starttaster (S1) betätigt wird und die Schutzvorrichtung (S2) geschlossen ist.
- Die Signalleuchte einer Maschine soll blinken, wenn eine bestimmte Drehzahl n unterschritten (S1 = 0) oder die Lagertemperatur T zu hoch (S2 = 0) oder der Kühltank K leer (S3 = 1) ist.
- In einer Behörde steht für Kunden ein Ticketautomat bereit. Kunden, die nur etwas abholen wollen, drücken den Schalter 1. Kunden, die ein Anliegen haben, drücken Schalter 2. Der Automat erstellt die entsprechenden Tickets, auf denen die Wartezone vermerkt ist. Die Arbeitsweise des Automaten wird aus Sicherheitsgründen durch zwei separate Sensoren erfasst. Nur wenn beide Schalter unterschiedliche Signalzustände haben, ist die Funktionsweise gewährleistet.

Lösung Übungsaufgabe 4:

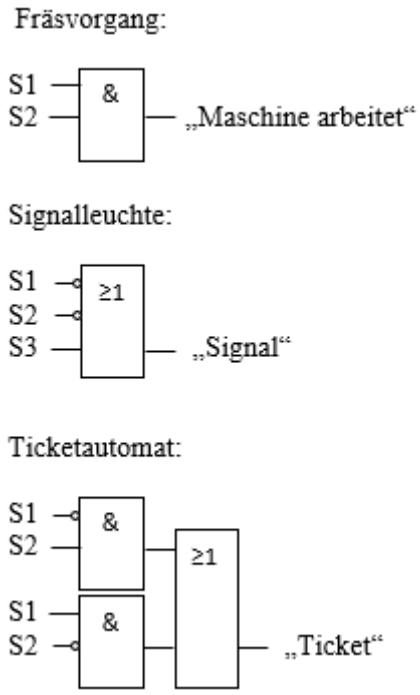


Abbildung 15: Funktionsplan Fräsvorgang (oben), Signalleuchte (mittig) und Ticketautomaten (unten) [Rabe 2019a, S. 24-26]

Übungsaufgabe 5: Lüfterüberwachung [Wellenreuther & Zastrow 2007]

Bei dieser Aufgabe sollen die vier Lüfter einer Tiefgarage überwacht werden und der Status mit einer Ampelanzeige ausgegeben werden. Wenn mehr als zwei Lüfter arbeiten, leuchtet die Ampel grün und wenn weniger als zwei Lüfter arbeiten, leuchtet sie rot. Wenn genau zwei Lüfter arbeiten soll die Ampel gelb leuchten.

Aufgabenstellung:

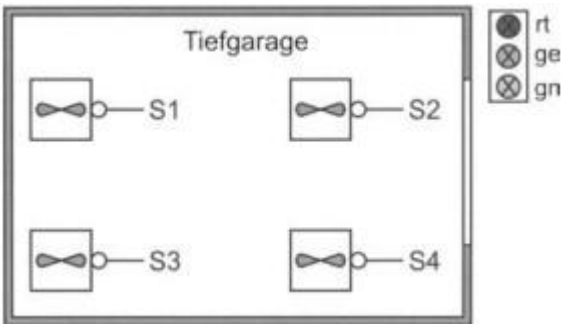


Abbildung 16: Lüfterüberwachung [Wellenreuther & Zastrow 2007]

In einer Tiefgarage sind vier Lüfter installiert. Die Funktionsüberwachung der Lüfter erfolgt durch je einen Luftströmungswächter. An der Einfahrt der Tiefgarage ist eine Ampel angebracht. Sind alle vier Lüfter oder drei Lüfter in Betrieb, ist für eine ausreichende Belüftung gesorgt und die Ampel zeigt Grün. Bei Betrieb von nur zwei Lüftern schaltet die Ampel auf Gelb. Es dürfen nur Fahrzeuge ausfahren. Sind weniger als zwei Lüfter in Betrieb, muss die Ampel Rot anzeigen.

1. Bestimmen Sie mit einer Wahrheitstabelle den Zusammenhang zwischen den Eingängen S1 bis S4 und den Ausgängen rt, ge und gn.
2. Zeichnen Sie den Funktionsplan zur Ansteuerung der drei Ausgänge.

Lösung Übungsaufgabe 5:

S4	S3	S2	S1	gn	ge	rt
0	0	0	0	0	0	1
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	1	0
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	0	1	0
0	1	1	1	1	0	0
1	0	0	0	0	0	1
1	0	0	1	0	1	0
1	0	1	0	0	1	0
1	0	1	1	1	0	0
1	1	0	0	0	1	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	1	0	0

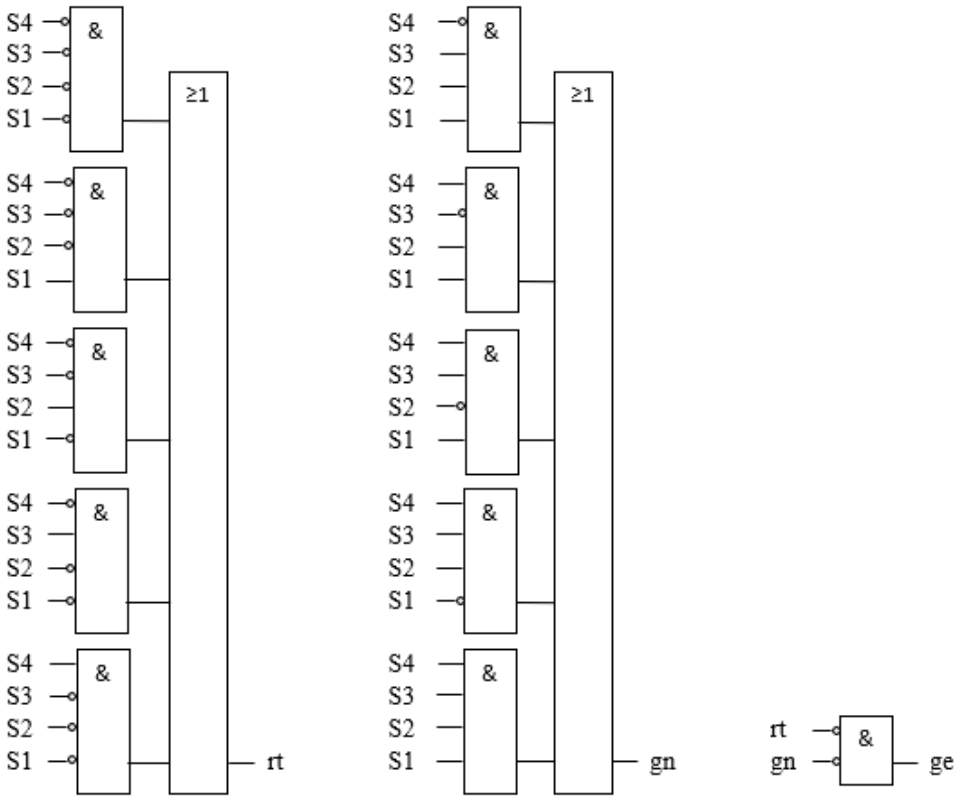


Abbildung 17: Lösung 5.1 (oben) und 5.2 (unten) [Wellenreuther & Zastrow 2007, S. 145]

Übungsaufgabe 6: NOR-Verknüpfung [Benda 1986, S.33]

Bei dieser Aufgabe soll eine Wahrheitstabelle und ein Funktionsplan anhand eines vorgegebenen Stromablaufplans erstellt werden.

Aufgabenstellung:

1. Entwerfen Sie zur folgenden Steuerschaltung (Stromablaufplan) die Wahrheitstabelle und zeichnen Sie den Funktionsplan.

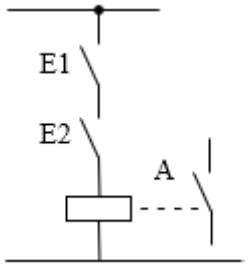


Abbildung 18: Steuerschaltung zur NOR-Verknüpfung [Benda 1986, S.33]

Lösung Übungsaufgabe 6:

E1	E2	A
0	0	0
1	0	0
1	1	0
1	1	1

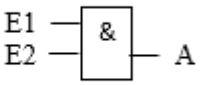


Abbildung 19: Lösung 6.1 [Benda 1986, S.105]

3.2 Aufstellung von Kriterien zur Bewertung der Übungsaufgaben [LH]

Im folgenden Kapitel werden nun die Kriterien zur Bewertung der im vorherigen Kapitel vorgestellten Aufgaben beschrieben und erläutert. Die aufgestellten Kriterien werden im Laufe der Arbeit zur Bewertung der Aufgaben genutzt.

Laut dem Landesinstitut für Schule sollen ausgewählte Übungen die Studierenden in ihrem individuellen Lernprozess unterstützen und auf eine eigenständige, produktive Lösung abzielen. Dabei können grundsätzlich zwei verschiedene Funktionen von Aufgaben unterschieden werden [Landesinstitut für Schule (2005)]. Zum einen die Überprüfung von Leistungen und zum anderen zur Unterstützung fachlicher Lernprozesse [Landesinstitut für Schule 2005]. Dabei gibt es mehrere Kriterien, die eine gute Lernaufgabe auszeichnen [Kraus & Nieweler 2011].

Eine gute Lernaufgabe soll problemorientiertes Arbeiten ermöglichen und zum anderen Bedingungen für ein erfolgreiches Lernen fördern [Landesinstitut für Schule 2005]. Der Lernende soll durch die Lernaufgabe nicht nur ein passives Wissen, wie zum Beispiel durch auswendig lernen erwerben, sondern soll die Aufgabe den Erwerb intelligenten Wissens fördern [Landesinstitut für Schule 2005]. Laut dem Landesinstituts soll der Lernende mehrere aufeinander aufbauende Aufgabenteile bearbeiten, um so sein Wissen weiterentwickeln zu können. Im Rahmen dieser Arbeit werden, wie vom Landesinstitut empfohlen, folgende selbst gewählte Ausprägungen dabei gewählt. Es wird unterschieden zwischen Aufgaben, die nur zwei aufeinander aufbauende Teilaufgaben haben, Aufgaben, die zwischen drei und fünf Teilaufgaben besitzen und Aufgaben mit sechs oder mehr als sechs Teilaufgaben. Die Ausprägungen werden in dieser Arbeit so gewählt, da Aufgaben mit nur zwei Teilaufgaben kaum aufeinander aufbauende Lernmöglichkeiten ergeben [Wolf & Biehler 2004]. Aufgaben mit über sechs Aufgabenteilen werden laut Wolf und Biehler (2004) meist als überladen angesehen und sind daher auch nur bedingt nutzbar

Die Verzahnung der Lebenswelt der Lernenden mit der Aufgabenstellung spielt in der Diskussion über Bildungsstandards eine wichtige Rolle [Maier et al. 2010]. Laut Maier et al. gibt es vier zu unterscheidenden Ausprägungen in dieser Relation. Zum einen gibt es die Aufgaben ohne Lebensweltbezug. In dieser Ausprägung wird keine Verknüpfung zwischen dem Fachwissen und der Erfahrungswelt der Lernenden vorgegeben oder gefordert. Eine weitere Ausprägung befasst sich mit dem konstruierten Lebensweltbezug. Es gibt bei dieser Aufgabenstellung eine Verknüpfung zwischen Fachwissen und der Lebenswelt, jedoch entspricht diese

Lebenswelt nicht den Erfahrungen der Lernenden und eine Analogie zu den Erfahrungen ist kaum erkennbar. Die dritte Ausprägung beinhaltet laut den Verfassern Aufgaben mit konstruiertem aber authentisch wirkendem Lebensweltbezug. Bei dieser Aufgabenart ergibt der Zusammenhang zwischen Fachwissen und dem Lebensweltbezug Sinn, jedoch ist er immer noch erkennbar konstruiert. Die vierte und letzte Ausprägung laut Maier et al. befasst sich mit dem realem Lebensweltbezug. Bei dieser Ausprägung geht die Differenz zwischen der Lebenswelt des Lernenden und der Aufgabenstellung gegen null [Maier et al. 2010]. Auch in dieser Dimension wird, im Rahmen der Projektarbeit, die vierte Ausprägung nicht berücksichtigt, da es für Studierende, außer in seltenen Ausnahmen, keinen realen Lebensweltbezug geben kann bei der Programmierung einer SPS.

Lernaufgaben lassen sich auch in die Dimension der Komplexität unterteilen [Maier et al. 2010]. Das Landesinstitut für Schule gibt jedoch auch zu bedenken, dass die Ausprägungen immer auf die Übungsaufgaben angepasst und für jeden Aufgabentyp neu aufgestellt werden müssen Zur besseren Einschätzung der Übungsaufgabe wird, im Rahmen dieser Projektarbeit, nun die Anzahl an Schalter und Ausgängen zur Lösung des Problems als Ausprägungen gewählt. Die Dimension der Komplexität wird in niedrig, mittel und hoch unterteilt. Eine niedrige Komplexität beinhaltet Aufgaben mit bis zu drei Schaltern und einer Ausgabe. Diese Ausprägung wird so gewählt, da diese Aufgaben nur eine kleine Anzahl an möglichen Lösungen gibt. Die Ausprägung mittlere Komplexität befasst sich mit SPS-Aufgaben mit drei Schaltern und mindestens zwei Ausgaben. Durch die erhöhte Vielfalt der möglichen Kombinationen der Elemente wird diese Ausprägung gewählt. Aufgaben mit hoher Komplexität besitzen, in dieser Projektarbeit, mehr als vier Schalter und mehr als drei Ausgängen. Durch die große Anzahl an möglichen Kombinationen der vier Schalter und der Ausgabe sind diese Aufgaben deutlich komplexer im Vergleich zu den anderen Aufgaben. Ersichtlich ist dies auch an der Größe der Wahrheitstabelle. Die Tabelle besitz mit 16 möglichen Varianten eine Vielzahl verschiedener Möglichkeiten und ist daher komplexer.

Dimensionen	Ausprägungen		
Strategielernen	Maximal zwei Teilaufgaben	Drei bis fünf Teilaufgaben	≥ sechs Teilaufgaben
Lebensweltbezug	Kein	Konstruiert	Authentisch
Komplexität	Niedrig	Mittlere	Hohe

Abbildung 20:Kriterien zur Bewertung von Aufgaben

Mit den nun aufgestellten Dimensionen und dazugehörigen Ausprägungen werden im nächsten Kapitel die bereits beschriebenen Aufgaben bewertet.

3.3 Bewertung der Übungsaufgaben anhand aufgestellter Kriterien [DD]

Nachdem in den Kapiteln 3.1 und 3.2 einige Übungsaufgaben und Kriterien zur Bewertung dieser Übungsaufgaben vorgestellt wurden, folgt nun die Überprüfung auf die Eignung der Übungsaufgaben zum Einsatz im Lehrbetrieb.

Zuerst wird die in Kapitel 3.1 vorgestellte „Übungsaufgabe 1: Wahlfreie Schaltstellen“ betrachtet. Die Übungsaufgabe befasst sich mit dem Ablassventil eines Silos, welches von drei Schaltstellen aus wahlweise geöffnet bzw. geschlossen werden kann. Sie besitzt zwei Teilaufgaben, was kaum Raum für aufeinander aufbauende Lernmöglichkeiten bietet [Wolf & Biehler 2004]. Zur Erweiterung der Dimension des Strategielernens könnten bei dieser Aufgabe Teilaufgaben hinzufügt werden. Beispielsweise wäre eine passende zusätzliche Teilaufgabe, dass man nicht nur den Funktionsplan mithilfe der Wahrheitstabelle aufstellt, sondern auch den Kontaktplan. Eine weitere Möglichkeit wäre den Funktionsplan aus der zweiten Teilaufgabe in einen Kontaktplan zu überführen. Zur zweiten Dimension, dem Lebensweltbezug, lässt sich sagen, dass die Übungsaufgabe einen konstruierten Lebensweltbezug besitzt. Der Lernende kennt aus seiner Erfahrungswelt ein Silo, sowie die Funktionsweise eines Ventils und dadurch erscheint es ihm auch logisch, dass das Ventil von drei Schaltstellen aus geöffnet bzw. geschlossen werden kann. Da der Großteil der Lernenden wahrscheinlich nicht selbst in direktem Kontakt mit Silos steht, fehlt hier der authentische Lebensweltbezug. Zur dritten Dimension, der Komplexität, lässt sich die Aufgabe laut Kapitel 3.2 als Aufgabe mit niedriger Komplexität beschreiben, da nur drei Schalter und eine Ausgabe vorliegen. Zusammenfassend lässt sich sagen, dass die „Übungsaufgabe 1: Wahlfreie Schaltstellen“ im Allgemeinen eine gute Übungsaufgabe zum Einstieg in die grafischen Programmiersprachen für SPS darstellt. Sie beinhaltet nur zwei Teilaufgaben, was den Studierenden am Anfang nicht überfordert und kann optional erweitert werden, damit sein Wissen weiterentwickelt werden kann. Prinzipiell ist ein hoher Lebensweltbezug anzustreben, da dieser das Interesse des Studierenden am ehesten weckt [Maier et al. 2010]. Ein weiteres Argument für die Eignung als Übungsaufgabe zum Einstieg beschreibt die niedrige Komplexität der Aufgabe.

Die zweite vorgestellte „Übungsaufgabe 2: Siloentleerung“ befasst sich, wie die erste Übungsaufgabe, mit einem Silo. Bei dieser Aufgabe soll das Silo über zwei Pumpen entleert werden, welche über die jeweilige Füllstandshöhe im Silo ein- bzw. ausgeschaltet werden. Die Bewertung dieser Übungsaufgabe folgt analog zur „Übungsaufgabe 1: Wahlfreie Schnittstellen“. Die Übungsaufgabe enthält ebenfalls zwei Teilaufgaben, welche optional mit den zuvor erwähnten

Möglichkeiten zur Erweiterung der Aufgabe ergänzt werden können. Die Aufgabe besitzt auch hier einen konstruierten Lebensweltbezug, was auf dieselbe Begründung wie bei der ersten Übungsaufgabe zurückzuführen ist. Im Unterschied zur ersten Übungsaufgabe ist die Komplexität höher, was man unter anderem daran erkennt, dass hier drei Schalter mit zwei Ausgaben vorzufinden sind. Daher wird diese Übungsaufgabe als Aufgabe mit mittlerer Komplexität bewertet. Abschließend kommt man zu einem äquivalenten Ergebnis der Bewertung gegenüber der ersten Übungsaufgabe mit dem einzigen Unterschied, dass die Komplexität der zweiten Übungsaufgabe etwas höher ist. Dementsprechend bietet die „Übungsaufgabe 2: Siloentleerung“ Potenzial nach der „Übungsaufgabe 1: Wahlfreie Schaltstellen“ angeboten zu werden, denn beide Aufgaben befassen sich mit demselben Thema und durch die steigende Komplexität erhöht sich der Lerneffekt für die Studierenden [Landesinstitut für Schule 2005].

Die dritte „Übungsaufgabe 3: Selbsthaltung“ beschäftigt sich mit der üblichen Schaltung, der Selbsthaltung, in Schützsteuerungen für eine Speicherfunktion. Bei dieser Übungsaufgabe soll anhand des jeweiligen Stromlaufplans und Aufgabentexts ein Funktions- und Kontaktplan für das entsprechende Netzwerk erstellt werden. Die Übungsaufgabe hat zwar nur einen Aufgabenteil, aber theoretisch gesehen könnte man diesen auch auf zwei Aufgabenteile aufteilen, da für jede Schaltung jeweils ein Funktions- und ein Kontaktplan erstellt werden soll. Die Aufgabe besitzt keine aufeinander aufbauenden Aufgabenteile, um sein Wissen weiterentwickeln zu können [Wolf & Biehler 2004]. Problematisch an dieser Übungsaufgabe ist, dass aus dem Stromlaufplan direkt der Funktions- und Kontaktplan erstellt werden soll, wodurch ein gewisses Vorwissen vorausgesetzt wird. Hier könnte optional, ähnlich zu den vorherigen Übungsaufgaben, die Aufgabe erweitert und somit vereinfacht werden, indem man zuerst eine Wahrheitstabelle zu der jeweiligen Aufgabe erstellt. Des Weiteren liegt hier ein konstruierter Lebensweltbezug vor, da die Studierenden zwar Erfahrungen mit Stromlaufplänen und Speicherfunktionen innerhalb ihres Studiums gemacht haben, aber die Analogie zur Lebenswelt eher gering ist. Es liegt eine niedrige Komplexität der Aufgabe vor, da bei beiden Netzwerken drei Eingaben und eine Ausgabe vorzufinden sind. Folglich unterscheidet sich die „Übungsaufgabe 3: Selbsthaltung“ im Vergleich zu den beiden zuvor vorgestellten Übungsaufgaben hauptsächlich darin, dass keine Wahrheitstabelle erstellt werden soll, wodurch ein gewisses Vorwissen und ein Gefühl zur Lösung der Übungsaufgabe von den Studierenden vorausgesetzt wird. Daher sollte diese Übungsaufgabe erst nach den ersten beiden Übungsaufgaben angeboten werden, um einen größeren Lerneffekt für den Studierenden zu erzielen.

Die vierte vorgestellte „Übungsaufgabe 4: Funktionsschaltungen“ beschreibt mehrere unterschiedliche Systeme und fordert die Erstellung eines Funktionsplans für jedes dieser Systeme. Bei dieser Übungsaufgabe gibt es eine Teilaufgabe für alle Systeme, da für jedes der drei vorgestellten Systeme jeweils ein Funktionsplan erstellt werden soll. Der Lebensweltbezug ist systemabhängig und hier liegt sowohl ein konstruierter als auch ein authentischer Lebensweltbezug vor. Während die ersten beiden Systeme einen konstruierten Lebensweltbezug besitzen, bietet das dritte System einen authentischen Lebensweltbezug, da alle Studierenden in ihrer eigenen Lebenswelt mit Ticketautomaten in Verbindung stehen. Alle Systeme dieser Übungsaufgabe besitzen eine niedrige Komplexität, was man laut Kapitel 3.2 an der Anzahl der Schalter und Ausgängen erkennen kann. Zusammenfassend bietet die „Übungsaufgabe 4: Funktionsschaltungen“ einen guten Einstieg in die grafische Programmiersprache Funktionsplan für SPS. Optional könnte man die Übungsaufgabe dahingehend erweitern, dass nicht nur der Funktionsplan, sondern auch der Kontaktplan abgefragt wird. Dadurch würden beide grafischen Programmiersprachen abgefragt. Ebenfalls ist zu erwähnen, dass der Studierende durch die Bearbeitung der verschiedenen Systeme eine gute Übungsaufgabe mit niedriger Komplexität erhält.

Die fünfte „Übungsaufgabe 5: Lüfterüberwachung“ beschreibt eine Überwachung von vier Lüftern einer Tiefgarage. Eine Ampel gibt den aktuellen Status wieder. Diese Übungsaufgabe enthält eine äquivalente Aufgabenstellung wie die ersten beiden vorgestellten Übungsaufgaben. Es wird zuerst eine Wahrheitstabelle und dann ein Funktionsplan für das beschriebene System gefordert. Wegen der Äquivalenz zu den ersten beiden vorgestellten Übungsaufgaben, könnte die Übungsaufgabe auch hier optional dadurch erweitert werden, dass nicht nur der Funktionsplan erstellt werden soll, sondern auch der Kontaktplan. Besonders hervorzuheben ist der authentische Lebensweltbezug, da das Thema dieser Übungsaufgabe für die Studierenden eine Analogie zu ihren eigenen Erfahrungen darstellt und dadurch auch greifbar wirkt. Durch den authentischen Lebensweltbezug wird das Interesse der Studierenden eher geweckt als bei den zuvor vorgestellten Übungsaufgaben. Zur Komplexität der Aufgabe lässt sich sagen, dass sie eine hohe Komplexität besitzt, was man anhand der Lösung erkennen kann. Es sind nämlich vier Schalter und drei Ausgänge vorzufinden. Zusammenfassend lässt sich sagen, dass die „Übungsaufgabe 5: Lüfterüberwachung“, vor allem durch den authentischen Lebensweltbezug und der hohen Komplexität, eine gute Übungsaufgabe für den Einsatz im Lehrbetrieb bietet.

Die „Übungsaufgabe 6: NOR-Verknüpfung“ unterscheidet sich von den bisher vorgestellten Übungsaufgaben dahingehend, dass sie keinen beschreibenden Aufgabentext enthält. Man soll aus einem gegebenen Stromablaufplan eine Wahrheitstabelle und einen Funktionsplan

erstellen. Aus der Aufgabenstellung geht hervor, dass man die Übungsaufgabe der ersten Ausprägung des Strategielernens zuordnen kann. Der größte Unterschied zu den bisher aufgestellten Übungsaufgaben liegt darin, dass hier kein Lebensweltbezug vorliegt. Es wird ein beliebiger Stromlaufplan gegeben, um daraus eine Wahrheitstabelle und einen Funktionsplan zu erstellen. Durch den fehlenden Lebensweltbezug ist die Übungsaufgabe kritisch anzusehen, da das Interesse des Studierenden definitiv nicht geweckt wird. Weiter liegt eine niedrige Komplexität vor, da es nur zwei Eingänge und einen Ausgang gibt. Schließlich kommt man zu dem Entschluss, dass die Aufgabe vor allem wegen des fehlenden Lebensweltbezugs problematisch sein könnte. Prinzipiell ist es aber durchaus denkbar diese Übungsaufgabe zur vertiefenden Übung anzubieten.

Schlussendlich können alle vorgestellten Übungsaufgaben eine Möglichkeit für den Einsatz im Lehrbetrieb bieten. Allerdings ist auch zu erwähnen, dass jede Aufgabe Potenzial zur Optimierung nach den vorgestellten Kriterien aus Kapitel 3.2 bietet. Des Weiteren bietet es sich an einige Übungsaufgaben als Abfrage in der Moodle-Lerneinheit, welche im Kapitel 5 beschrieben wird, zu nutzen, da sie unabhängig voneinander sind und sich somit ideal zur Vertiefung des Wissens der Studierenden eignen. Im Rahmen dieser Projektarbeit und anhand der aufgestellten Kriterien aus dem dritten Kapitel lässt sich sagen, dass sich für die Übungsaufgaben zum Einsatz innerhalb der Präsenzübung eher Übungsaufgaben mit einem hohen Lebensweltbezug eignen, da dadurch das Interesse der Studierenden geweckt wird und sie die Inhalte direkt mit den eigenen Erfahrungen verknüpfen können [Maier et al. 2010]. Aus den vorgestellten Übungsaufgaben erfüllt nur die fünfte Übungsaufgabe diese Eigenschaft. Daher wird die fünfte Übungsaufgabe im folgenden Kapitel aufgegriffen und weitere Übungsaufgaben für die Präsenzübung erarbeitet, welche anhand der aufgestellten Kriterien aus Kapitel 3.2 geeignete Übungsaufgaben darstellen. Damit können die restlichen vorgestellten Übungsaufgaben in die Moodle-Lerneinheit, siehe Kapitel 5, zur vertiefenden Übung einbezogen werden.

4 Geeignete Übungsaufgaben für die Präsenzübung [DD]

In diesem Kapitel werden, im Hinblick der aufgestellten Kriterien aus Kapitel 3.2, geeignete Übungsaufgaben auf dem Gebiet der grafischen Programmiersprachen für SPS zum Einsatz im Lehrbetrieb vorgestellt. Da innerhalb dieser Projektarbeit die zwei grafischen Programmiersprachen Funktionsbausteinsprache und Kontaktplan betrachtet wurden, bietet es sich an zu beiden Programmiersprachen jeweils eine Übungsaufgabe bereitzustellen. Als dritte und letzte Übungsaufgabe werden die grafischen Sprachen ineinander überführt, um das Verständnis der Studierenden zu festigen.

In Bezug auf die vorgestellten Kriterien aus Kapitel 3.2 ist zu erwähnen, dass die Dimension des Strategielernens eher eine untergeordnete Rolle einnimmt, da es aufgrund des spezifischen Themengebiets wenig Spielraum für viele unterschiedliche Teilaufgaben gibt. Besonders hervorzuheben ist, dass der Fokus geeigneter Übungsaufgaben im Rahmen dieser Projektarbeit und der aufgestellten Kriterien im dritten Kapitel auf einen authentischen Lebensweltbezug gelegt wird, da dieser das Interesse der Studierenden am ehesten weckt und dadurch ein größerer Lerneffekt entsteht [Maier et al. 2010]. Ebenfalls ist nicht zu vernachlässigen, dass die folgenden Übungsaufgaben eine mittlere bzw. hohe Komplexität beinhalten sollten, da die Moodle-Lerneinheit, siehe Kapitel 5, unter anderem die Grundlagen der jeweiligen Programmiersprache vermittelt und Übungsaufgaben mit niedriger bzw. mittlerer Komplexität zum Einstieg in die grafischen Programmiersprachen für SPS beinhaltet. Demnach können die beschriebenen Übungsaufgaben in diesem Kapitel, auf Grundlage des vermittelten Wissens aus der Moodle-Lerneinheit, zur Festigung und Erweiterung des Verständnisses der Studierenden folgen.

4.1 Beschreibung der Übungsaufgaben [DD]

Übungsaufgabe 1 (KOP): Fahrstuhlüberprüfung

Bei der ersten geeigneten Übungsaufgabe wird der Kontaktplan bearbeitet. Der Fahrstuhl eignet sich besonders gut als Übungsaufgabe für die grafischen Programmiersprachen, da er einen sehr hohen Lebensweltbezug für die Studierenden bietet. Aufgrund dieser Eigenschaft können diese die Thematik direkt mit ihren eigenen Erfahrungen verknüpfen und ihr Wissen weiter ausbauen [Maier et al. 2010]. Zur Komplexität der Übungsaufgabe lässt sich sagen, dass sie nach Kapitel 3.2 eine hohe Komplexität besitzt, da sie mehr als vier Eingänge und drei Ausgänge besitzt. Daher eignet sie sich ebenfalls als Übungsaufgabe für die Präsenzübung.

Aufgabenstellung:

Ein Fahrstuhl einer behindertengerechten Schule kann sich zwischen dem Erdgeschoss und dem zweiten Stock bewegen. Jede Etage (E0, E1, E2) besitzt einen Sensor zur Überprüfung, ob der Fahrstuhl sich auf der jeweiligen Etage befindet. Des Weiteren gibt es im Erdgeschoss einen Schalter um den Fahrstuhl nach unten (Anfrage_Runter) und im zweiten Stock einen Schalter um den Fahrstuhl nach oben (Anfrage_Hoch) zu schicken. Im ersten Stock gibt es beide Schalter. Durch eine digitale Anzeige weiß man welcher Schalter im ersten Stock betätigt werden muss, um den Fahrstuhl herbeizurufen. Falls der Fahrstuhl bereits im Erdgeschoss ist und der Schalter zum Runterfahren betätigt wird, soll der Fahrstuhl auf der Etage bleiben (Bleibe_Stehen). Analoges gilt für die Betätigung des Schalters zum Hochfahren, wenn der Fahrstuhl sich bereits im zweiten Stock befindet.

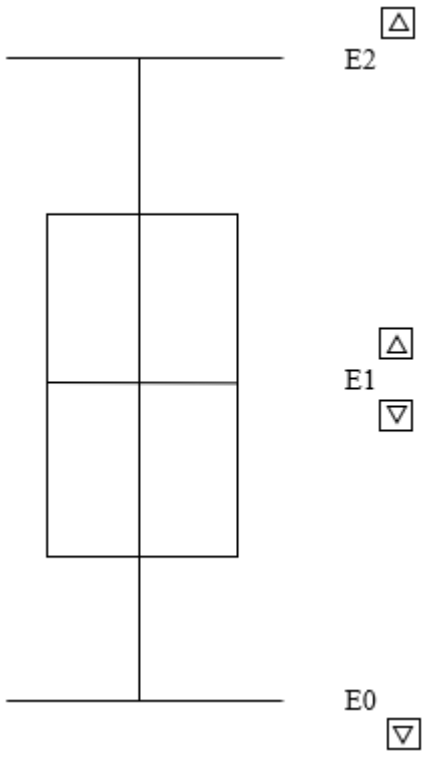


Abbildung 21: Fahrstuhl

1. Bestimmen Sie mit einer Wahrheitstabelle den Zusammenhang zwischen den gegebenen Eingängen (Anfrage_Hoch, Anfrage_Runter, E0, E1, E2) und den Ausgängen (Fahre_Hoch, Fahre_Runter, Bleibe_Stehen).
2. Erstellen Sie mithilfe der Wahrheitstabelle aus Aufgabenteil 1 einen Kontaktplan.

Übungsaufgabe 2 (FBS): Lüfterüberwachung [Wellenreuther & Zastrow 2007]

Bei der zweiten geeigneten Übungsaufgabe wird die Funktionsbausteinsprache betrachtet. Hier kann man die in Kapitel 3.1 vorgestellte „Übungsaufgabe 5: Lüfterüberwachung“ verwenden. Wie man der Bewertung aus Kapitel 3.3 entnehmen kann, eignet sich diese Übungsaufgabe besonders gut für die Präsenzübung. Sie besitzt einen authentischen Lebensweltbezug, da jeder Studierende mit dem Thema der Tiefgarage in seinem eigenen Leben konfrontiert wurde und dadurch das Interesse geweckt wird [Maier et al. 2010]. Zusätzlich ist die Komplexität der Übungsaufgabe, nach Kapitel 3.2, höher gegenüber der ersten geeigneten Übungsaufgabe und dadurch lernen die Studenten auch größere Systeme mit der Funktionsbausteinsprache darzustellen. Im Folgenden wird die Aufgabenstellung, analog wie in Kapitel 3.2, vorgestellt und die dazugehörige Lösung wird in Kapitel 4.2 beschrieben.

Aufgabenstellung:

In einer Tiefgarage sind vier Lüfter installiert. Die Funktionsüberwachung der Lüfter erfolgt durch je einen Luftströmungswächter. An der Einfahrt der Tiefgarage ist eine Ampel angebracht. Sind alle vier Lüfter oder drei Lüfter in Betrieb, ist für eine ausreichende Belüftung gesorgt und die Ampel zeigt Grün. Bei Betrieb von nur zwei Lüftern schaltet die Ampel auf Gelb. Es dürfen nur Fahrzeuge ausfahren. Sind weniger als zwei Lüfter in Betrieb, muss die Ampel Rot anzeigen.

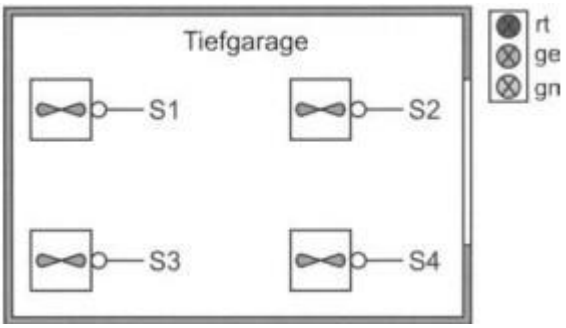


Abbildung 22: Lüfterüberwachung [Wellenreuther & Zastrow 2007]

1. Bestimmen Sie mit einer Wahrheitstabelle den Zusammenhang zwischen den Eingängen S1 bis S4 und den Ausgängen rt, ge und gn.
2. Zeichnen Sie den Funktionsplan zur Ansteuerung der drei Ausgänge.

Übungsaufgabe 3: Transferaufgabe

Die dritte und letzte geeignete Übungsaufgabe dient zur abschließenden Kontrolle des Verständnisses der Studierenden für die grafischen Programmiersprachen in SPS. Dazu wird der Funktions- bzw. Kontaktplan nicht wie zuvor anhand einer Systembeschreibung und anschließend erstellten Wahrheitstabelle aufgestellt, sondern direkt aus der anderen grafischen Programmiersprache. Die Studierenden sind mit den zuvor bearbeiteten Übungsaufgaben vertraut und können daher direkt ihre eigene Lösung in die andere Programmiersprache umwandeln. Zur Selbstkontrolle können sie im Anschluss die ebenfalls in der vorherigen Aufgabe erstellte Wahrheitstabelle hinzuziehen.

Aufgabenstellung:

1. Erstellen Sie mithilfe des Kontaktplans aus „Übungsaufgabe 1 (KOP): Fahrstuhlüberprüfung“ einen Funktionsplan.
2. Erstellen Sie mithilfe des Funktionsplans aus „Übungsaufgabe 2 (FBS): Lüfterüberwachung“ einen Kontaktplan.

4.2 Vorgehensweise und Lösungen der Übungsaufgaben [DD]

Übungsaufgabe 1 (KOP): Fahrstuhlüberprüfung

Zuerst wird die Erstellung einer Wahrheitstabelle zu den gegeben Ein- und Ausgängen abgefragt, welche durch Abbildung 23 dargestellt wird. Es gibt zwei verschiedene Anfragen und jeweils einen Sensor pro Etage. Wenn sich der Fahrstuhl bereits in Bewegung befindet und eine Anfrage durch einen Schalter getätigt wird soll nichts passieren, da der Fahrstuhl sonst möglicherweise während der Fahrt einen Richtungswechsel vornehmen würde. Da nichts passieren soll, ist dieser Fall auch nicht in der Wahrheitstabelle vorzufinden. Es gibt drei verschiedene Fälle pro Anfrage, da der Fahrstuhl entweder die Anfrage erfüllt, weil er sich auf einer der beiden anderen Etagen befindet oder stehen bleibt, weil er sich bereits auf der angefragten Etage befindet.

Anfrage_Hoch	Anfrage_Runter	E0	E1	E2	Fahre_Hoch	Fahre_Runter	Bleibe_Stehen
1	0	1	0	0	1	0	0
1	0	0	1	0	1	0	0
1	0	0	0	1	0	0	1
0	1	1	0	0	0	0	1
0	1	0	1	0	0	1	0
0	1	0	0	1	0	1	0

Abbildung 23: Lösung 1.1

Die zweite Teilaufgabe fordert die Erstellung eines Kontaktplans mithilfe der vorher erstellten Wahrheitstabelle, welchen man der Abbildung 24 entnehmen kann. Es entstehen drei Netzwerke, da es drei Ausgaben gibt. Für den Fall, dass man den Schalter zum Runterfahren betätigt fährt der Fahrstuhl runter, wenn er sich auf der ersten oder zweiten Etage befindet. Analog gilt für das zweite Netzwerk, dass der Fahrstuhl bei Betätigung des Schalters zum Hochfahren hochfährt, wenn er sich im Erdgeschoss oder auf der ersten Etage befindet. Für den Fall, dass der Fahrstuhl weder runter oder hoch fährt, bleibt der Fahrstuhl stehen.

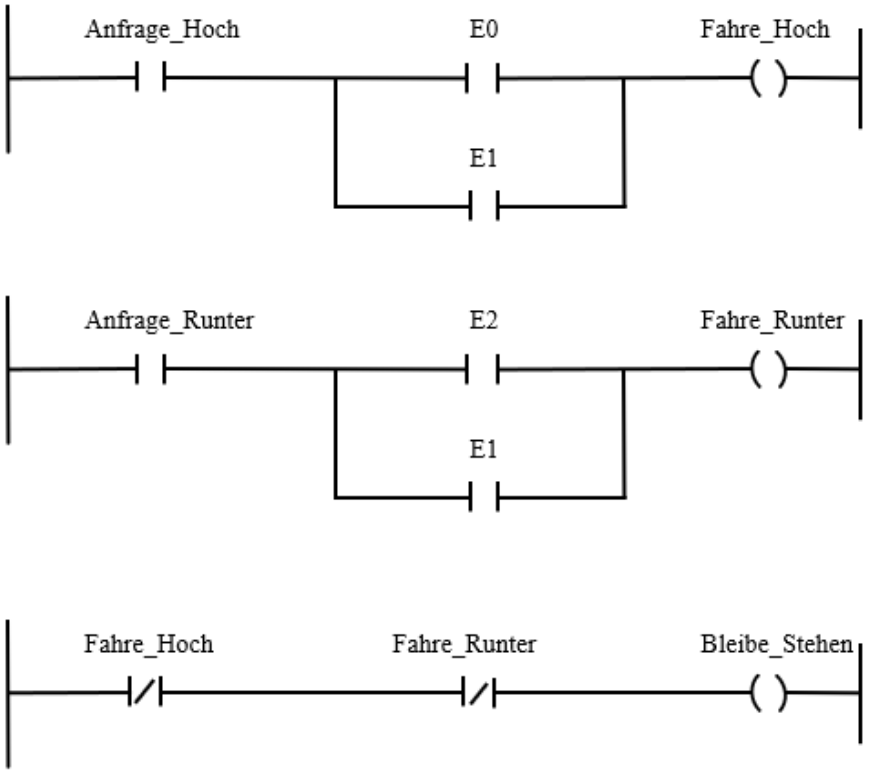


Abbildung 24: Lösung 1.2

Übungsaufgabe 2 (FBS): Lüfterüberwachung [Wellenreuther & Zastrow 2007]

Zu Anfang soll man bei der ersten Teilaufgabe eine Wahrheitstabelle erstellen (siehe Abbildung 25). Im ersten Block der Wahrheitstabelle sind die Lüfter S4 und S3 nicht im Betrieb und es werden alle Kombinationen für die Lüfter S2 und S1 betrachtet. Der Unterschied zwischen dem ersten und dem zweiten Block liegt darin, dass nun der Lüfter S3 immer in Betrieb ist. Wohingegen im dritten Block der Lüfter S4 statt S3 immer in Betrieb ist. Der letzte Block beschreibt die Zustände, wenn die Lüfter S3 und S4 immer in Betrieb sind. Somit sind alle möglichen Kombinationen der Lüfter abgearbeitet.

Bei der zweiten Teilaufgabe soll das vorgestellte System mit der Funktionsbausteinsprache dargestellt werden, was man ebenfalls der Abbildung 25 entnehmen kann. Es entstehen drei Netzwerke, da es drei Ausgaben gibt. Das erste Netzwerk beschreibt alle Möglichkeiten für den Fall, dass die Ampel rot ist, also maximal ein Lüfter in Betrieb ist. Dazu verwendet man für die Lüfter untereinander die UND-Bedingung (&) und für die fünf verschiedenen Kombinationen die ODER-Bedingung (≥1). Das zweite Netzwerk beschreibt den Zustand, dass die Ampel grün ist, also mindestens drei Lüfter arbeiten. Hier geht man analog zum ersten Netzwerk vor. Der letzte Fall beschreibt die gelbe Ampel, also genau zwei Lüfter in Betrieb sind. Theoretisch

gesehen könnte man wie bei den vorherigen Netzwerken jeden einzelnen Fall betrachten. Allerdings ist eine elegantere Lösung, wenn man eine UND-Bedingung in Zusammenhang mit den Ausgaben rot und grün erstellt. Dies bedeutet also, wenn die Ampel nicht rot und auch nicht grün ist, muss sie gelb sein.

S4	S3	S2	S1	gn	ge	rt
0	0	0	0	0	0	1
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	1	0
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	0	1	0
0	1	1	1	1	0	0
1	0	0	0	0	0	1
1	0	0	1	0	1	0
1	0	1	0	0	1	0
1	0	1	1	1	0	0
1	1	0	0	0	1	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	1	0	0

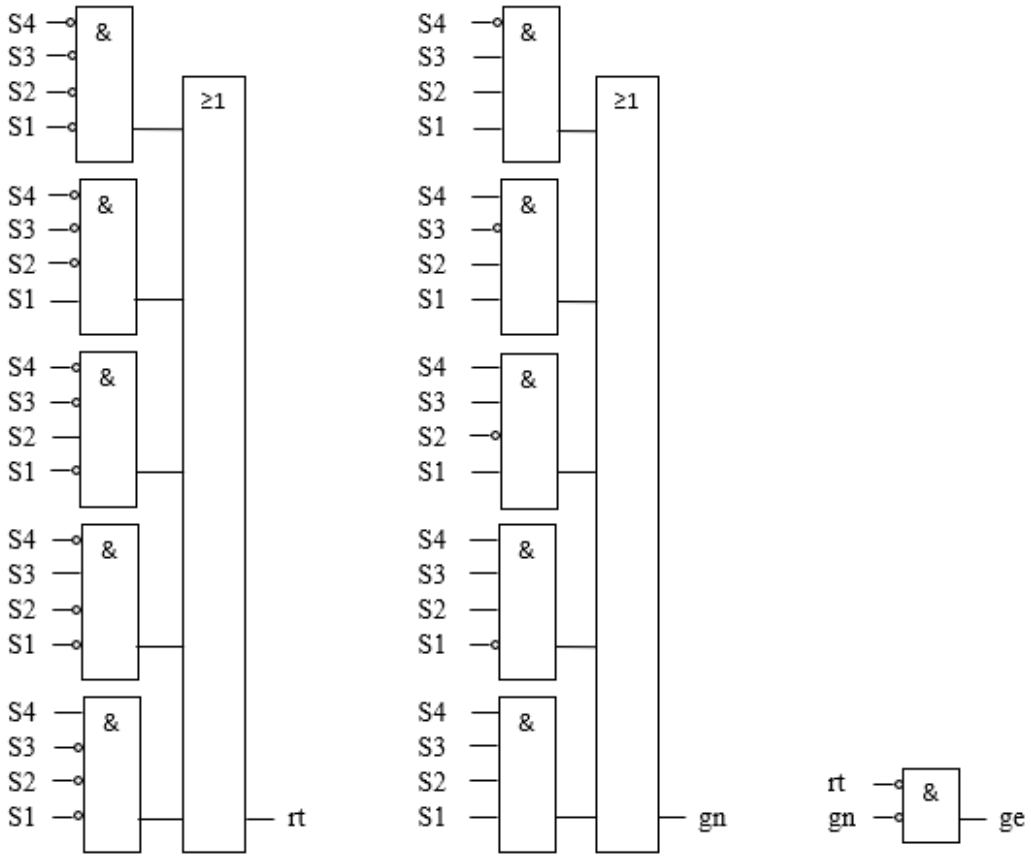


Abbildung 25: Lösung 2.1 (oben) und 2.2 (unten) [Wellenreuther & Zastrow 2007, S. 145]

Übungsaufgabe 3: Transferaufgabe

Bei der ersten Teilaufgabe betrachtet man den Kontaktplan aus „Übungsaufgabe 1 (KOP): Fahrstuhlüberprüfung“ und wandelt ihn in einen Funktionsplan um. Dazu erstellt für jedes der drei Netzwerke einen Funktionsplan. Man übersetzt den Kontaktplan in einen Funktionsplan, indem man die Grundelemente des Funktionsplans benutzt. Die Lösung der Aufgabe kann man Abbildung 26 entnehmen.

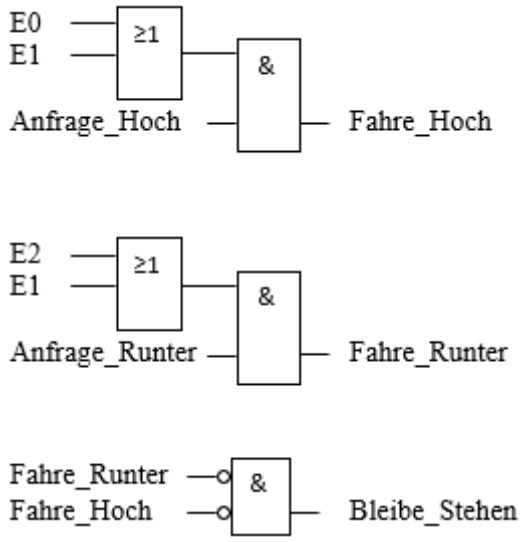


Abbildung 26: Lösung 3.1

Die zweite Teilaufgabe beinhaltet die Übersetzung des Funktionsplans aus „Übungsaufgabe 2 (FBS): Lüfterüberwachung“ in einen Kontaktplan. Hier geht man analog zur ersten Teilaufgabe vor mit dem Unterschied, dass man nun den Kontaktplan mit den Grundelementen des Kontaktplans darstellt. Die Lösung dieser Teilaufgabe kann man Abbildung 27 entnehmen.

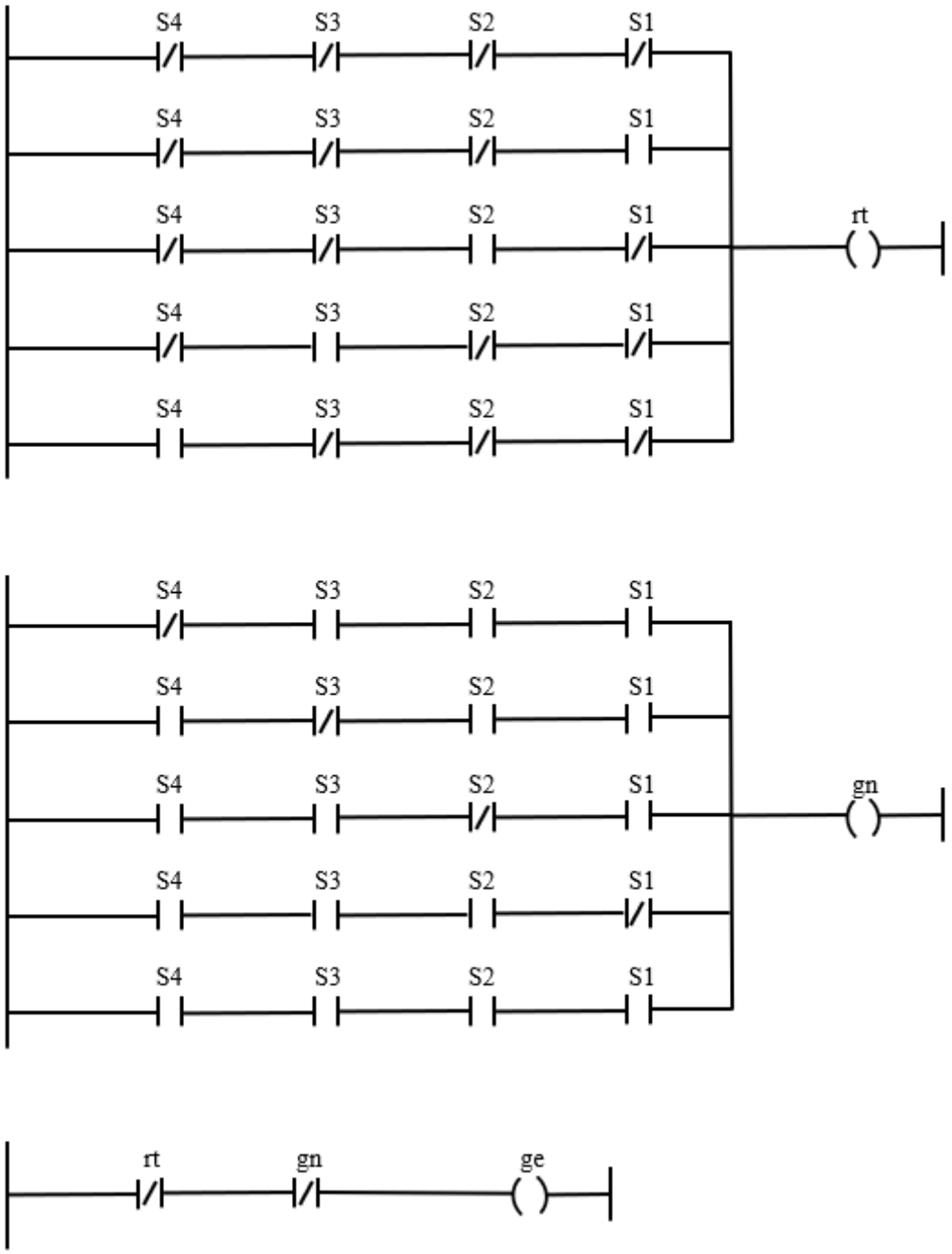


Abbildung 27: Lösung 3.2

5 Moodle-Lerneinheit [LH]

Dieses Kapitel beschäftigt sich mit der Einbettung der vorgestellten Übungsaufgaben in die Moodle-Umgebung und weiteren Übungsaufgaben, die in der Moodle-Lerneinheit implementiert sind. Des Weiteren findet eine Evaluation statt, um bestimmen zu können, ob das Erlernen in der Moodle-Umgebung eine gute Ergänzung zu den Präsenzübungen und Tutorien sind. Die Moodle-Lerneinheit trägt dabei den Namen Testkurs SPS. Die Teilnehmer der Moodle-Lerneinheit mit dem Status Lehrende werden als Ersteller der Übungsaufgaben im folgenden auch Trainer genannt.

5.1 Übertragung von Übungsaufgaben in Moodle-Lerneinheit [LH]

Nachdem im Kapitel 4 die endgültigen Übungsaufgaben für die Präsenzübung vorgestellt wurden, wird nun in diesem Kapitel über die verschiedenen Möglichkeiten der Darstellung von Übungsaufgaben in Moodle diskutiert. Die folgenden Informationen der möglichen Aktivitäten entstammen aus der Beschreibung der Aktivitäten in der Moodle-Version 3.7.

Ein Teil der möglichen Aktivitäten, die die Moodle-Lernplattform anbietet, sind für die Darstellung der Übungsaufgaben dieser Arbeit ungeeignet. Diese werden im Folgenden nicht weiter betrachtet, da diese beispielsweise als Datenbank, Umfragen, Fragenforum oder Besprechungsforum genutzt werden können. Weitere Aktivitäten, die eine kollaborative Zusammenarbeit zwischen den Studierenden fördern soll oder Aktivitäten, die eine gegenseitige Beurteilung bedingen, werden im Rahmen dieser Projektarbeit ebenso nicht weiter betrachtet.

Die folgenden Aktivitäten könnten zwar zur Darstellung von Übungsaufgaben genutzt werden, werden jedoch aus individuellen Gründen ebenfalls ausgeschlossen. Die Aktivität Lektion könnte zur Aufbereitung des bekannten Wissens aus den Vorlesungen genutzt werden., Denn erst mit Bestehen der abschließenden Testfrage, würde der Teilnehmer Zugriff auf die nächste Lektion bekommen. Die Aktivität würde den Vorteil bringen, dass sich der Teilnehmer intensiv mit der Lektion vertraut machen muss, um den Test zu bestehen und so weitere Lektionen frei zuschalten. Jedoch ist diese Aktivität eher bei einem Selbststudium einsetzbar. In einer weiteren Aktivität, dem Lernpaket, werden Inhalte auf mehreren Seiten bereitgestellt. Außerdem können Testfragen in ein Lernpaket direkt integriert werden. Für die zuvor erarbeiteten Übungsaufgaben sollte diese Aktivität jedoch nicht gewählt werden, da sie dazu dient Informationen an die

Teilnehmenden zu übermitteln und eher eine Vorlesung ersetzt als eine Übung von gelerntem Wissen bietet.

Nun werden die weiteren Aktivitäten vorgestellt, die zur Darstellung der Übungsaufgaben gewählt werden mit einer Beschreibung, welche Elemente man für eine Überführung der Aufgaben benötigt. Durch die H5P-Aktivität ist es beispielsweise möglich, interaktive Videos, Drag-and-Drop-Fragen und Multiple-Choice-Fragen zu erstellen.

Die bereits zur Darstellung von Übungsaufgaben ausgeschlossene Aktivität Feedback wird in der Projektarbeit für die Evaluation benutzt.

Im Rahmen dieser Projektarbeit wurde die Möglichkeit die Drag-and-Drop-Abfrage genutzt, um ausgewählte Übungsaufgaben darzustellen. Um eine Übungsaufgabe zu erstellen, benötigt man zuerst ein Bild, welches man als „Hintergrundbild“ einfügt (siehe Abbildung 28). Anschließend wird die Ablagezone definiert, in der die Antwortelemente gezogen werden können, die über den Reiter Text oder Bild eingefügt werden können (siehe Abbildung 29).

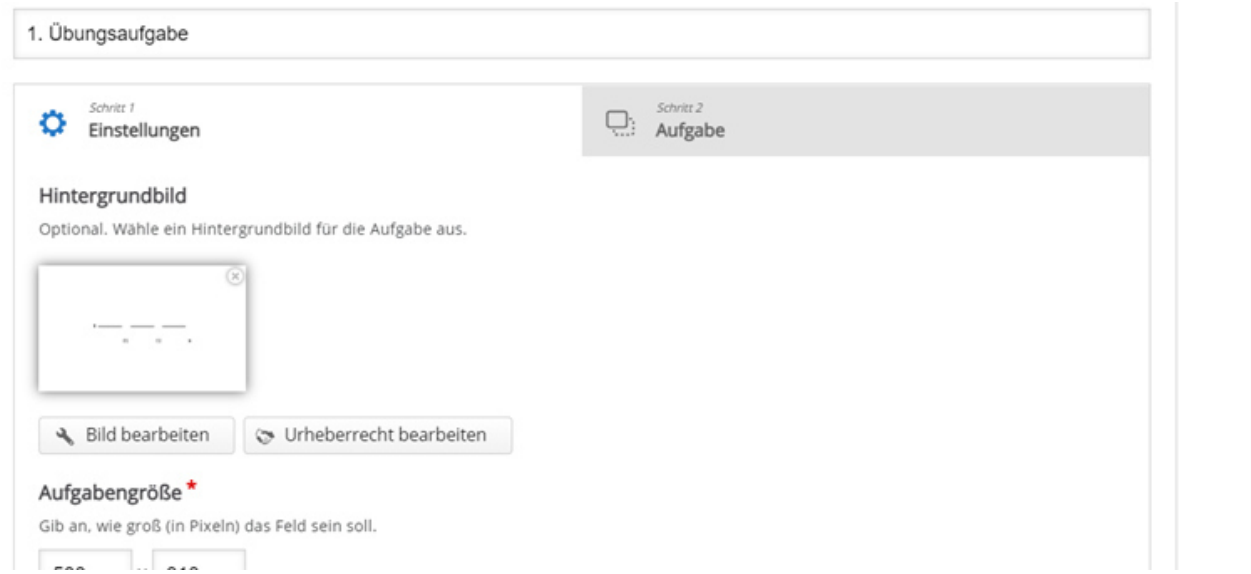


Abbildung 28: Einfügen des Hintergrundbildes

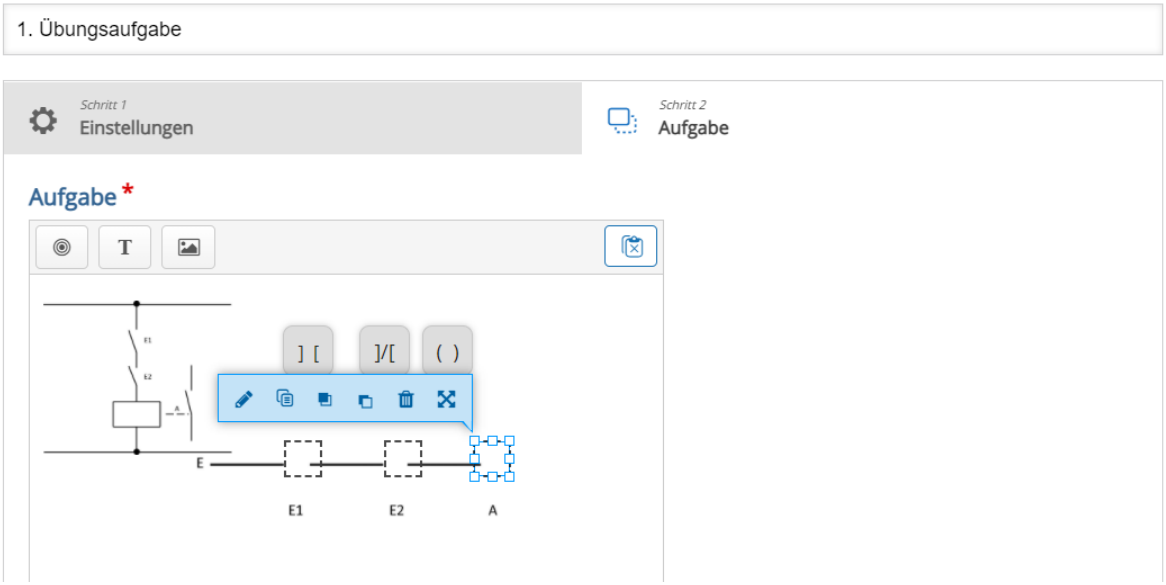


Abbildung 29: Einfügen der Ablagezone und der Antwortelemente

Ein Problem, welches sich bei dieser Möglichkeit ergibt, ist das jegliche Bildrechte an den Hintergrundbildern oder den Bildelementen die als Antwortmöglichkeiten dienen, besessen werden müssen. Eine mögliche Lösung ist es, die Bilder und Graphiken selbst mithilfe von Power-Point zu erstellen. Ein weiteres Problem bilden die Negation-Bausteine. Diese können nicht als Textelement dargestellt werden. Diese Bausteine müssen erneut in Form eines selbsterstellten Bildes eingefügt werden.

Eine weitere Aktivität, neben der Verwendung von Drag-and-Drop, die in dieser Projektarbeit zur Darstellung von Aufgaben benutzt wird, ist der Test. Innerhalb dieser Aktivität gibt es mehrere Fragentypen zur Auswahl. Zur Darstellung der Übungsaufgaben in dieser Arbeit wird der Fragentyp Multiple-Choice ausgewählt. Die Fragestellung kann bei einer Multiple-Choice-Aufgabe in den Fragetext eingefügt werden. Um das Bild der Steuerschaltung einzufügen, muss der Ersteller über einen Reiter die Grafik hochladen (siehe Abbildung 30)

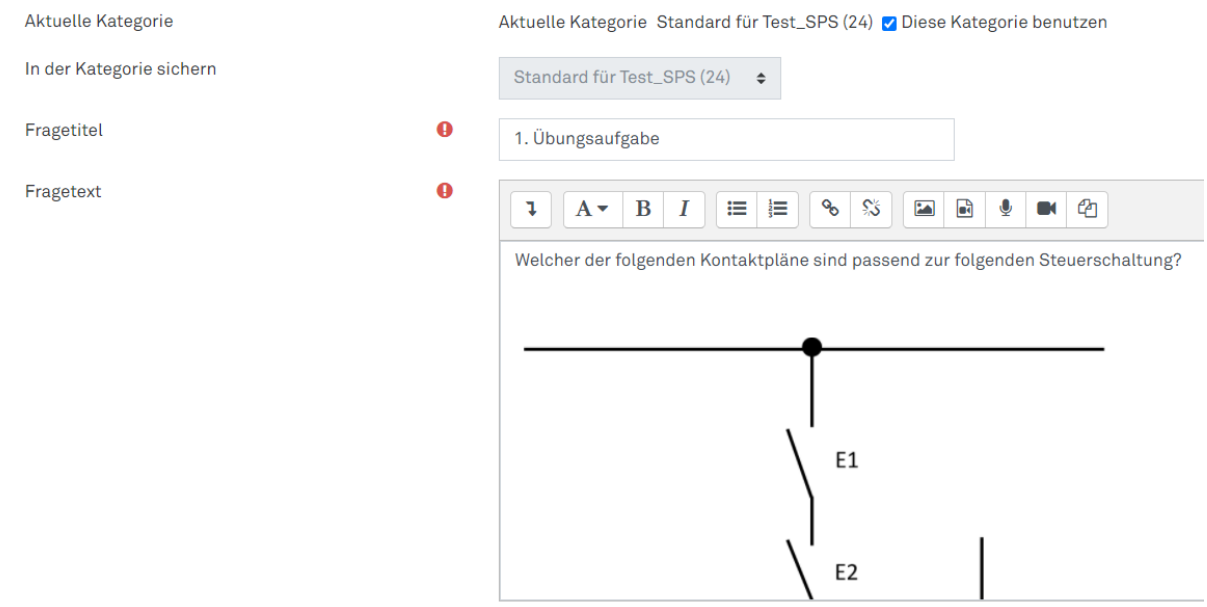


Abbildung 30: Erstellung der Multiple-Choice-Frage

Die auf der Moodle-Plattform von einem Trainer gestellten Antwortmöglichkeiten zu der gestellten Frage können, wie die Aufgabenstellung, mitvielen verschiedenen Darstellungsformen angegeben werden. In dieser Projektarbeit werden Bilder als Antwortmöglichkeit gewählt. Die Bilder müssen jedoch zuvor, wie bereits erwähnt, selbst über ein Tool, beispielsweise PowerPoint erstellt werden und können dann als Antwortmöglichkeiten hochgeladen werden.

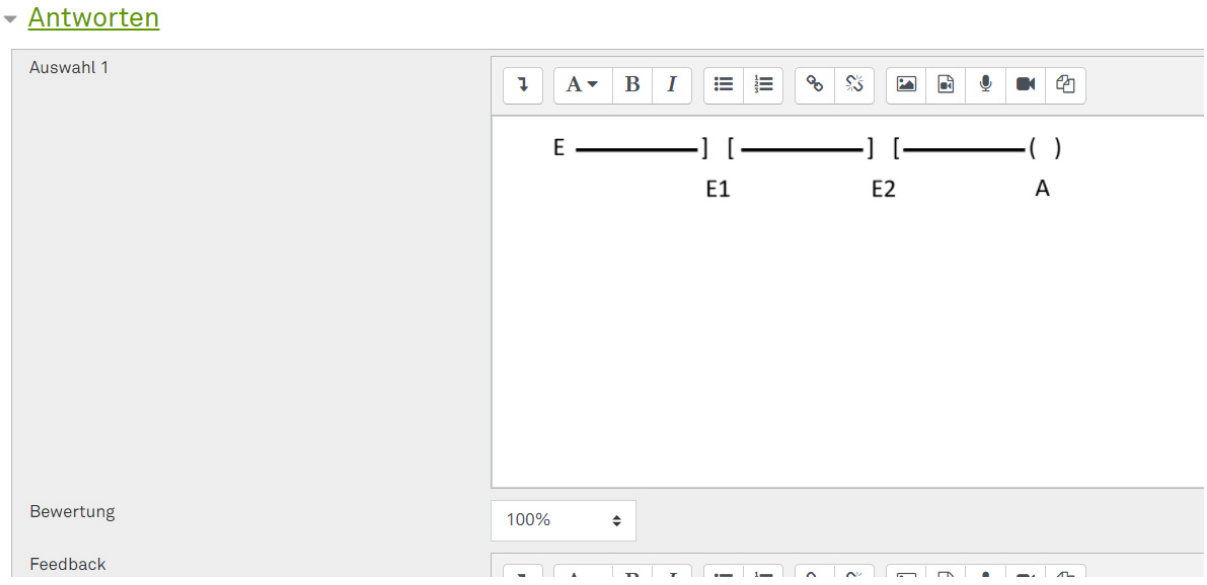


Abbildung 31: Einfügen der Antwortmöglichkeiten

Ob die Darstellungsformen des Drag-and-Drop sowie des Multiple-Choice sinnvoll gewählt wurden zur Darstellung von Übungsaufgaben von SPS, wird nun im nächsten Kapitel durch eine Evaluation bestimmt.

5.2 Aufbau der Moodle-Lerneinheit [LH]

Die selbsterstellte Moodle-Lerneinheit ist in vier Bereiche unterteilt. Die ersten beiden Bereiche FBS und KOP, beinhalten jeweils Aufgaben mit den Darstellungsformen des Multiple-Choices und des Drag-and-Drops. Im dritten Bereich befinden sich einige der bereits vorgestellten Aufgaben aus Kapitel 3.1 als Zusatzaufgaben. Den Abschluss der Moodle-Lerneinheit bildet die Evaluation, auf die in Kapitel 5.3 näher eingegangen wird.

Bei den verschiedenen Darstellungsformen der Übungsaufgaben werden zum einem die verschiedenen Grundelemente der SPS abgefragt und zum anderen sollen die Teilnehmer vier weitere Übungsaufgaben bearbeiten. Die Aufgaben der Programmiersprachen FBS und KOP sind in der Darstellungsformen Multiple-Choice und Drag-and-Drop jeweils identisch. Bei den Übungsaufgaben in der Moodle-Lerneinheit wird aufgrund der beschränkten Möglichkeiten kein Text zu den Aufgaben hinzugefügt, wodurch die Dimension des Lebensweltbezugs vernachlässigt wird. Die vier Übungsaufgaben pro Darstellungsform werden in der Moodle-Lerneinheit als eine Aufgabe definiert. Durch diese Definition können die zusammengesetzten Aufgaben der mittleren Ausprägung der Dimension des Strategielernens zugeordnet werden. Die Teilaufgaben sind weiterhin so aufgebaut, dass sie aufeinander aufbauen. In den ersten beiden Teilaufgaben wird zunächst eine Steuerschaltung dargestellt, die nur eine UND- oder eine ODER-Verknüpfung benötigen. In der dritten Teilaufgabe wird eine Kombination der beiden Verknüpfungen abgefragt. Die vierte Aufgabe bringt dem Teilnehmer ein neues Grundelement der SPS nahe. Die Teilaufgaben sollen in dieser Reihenfolge bearbeitet werden, da durch steigende Komplexität weiterführenden Verständnis angeregt wird [Petschenka et al. 2004].

Die erste Übungsaufgabe im Bereich FBS entspricht der bereits vorgestellten Übungsaufgabe 6 aus Kapitel 3.1 und wird daher nicht mehr gesondert vorgestellt.

Die zweite Übungsaufgabe in der Moodle-Lerneinheit zum Thema FBS befasst sich nun mit der ODER-Verbindung (siehe Abbildung 32). Die Steuerschaltung besitzt drei Schalter, die nur an ein Ausgangssignal gekoppelt sind. Die Komplexität ist aufgrund der Schalter und der Ausgangssignale als gering definiert (siehe Abbildung 33).

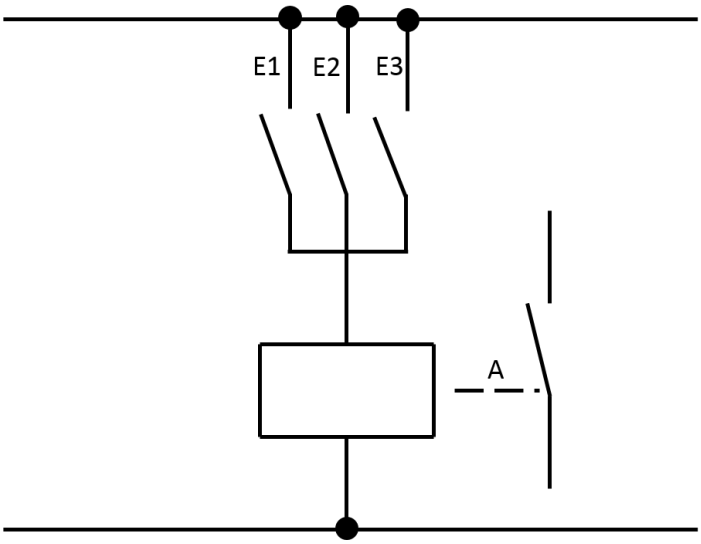


Abbildung 32: Moodle-Aufgabenstellung Nr. 2 FBS

E1	E2	E3	A
0	0	0	0
1	0	0	1
0	1	0	1
1	1	0	1
0	0	1	1
1	0	1	1
0	1	1	1
1	1	1	1

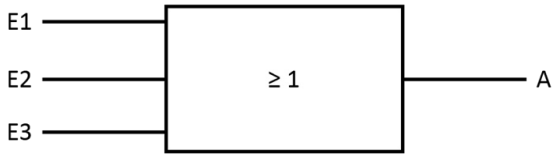


Abbildung 33:Moodle-Lösung Nr. 2 FBS

In der dritten Übungsaufgabe der FBS-Einheit wird nun eine Kombination einer UND- und ODER-Verbindung benötigt. Diese Aufgabe besitzt auch weiterhin nur eine geringe Komplexität, aufgrund der geringen Ausgangssignale (siehe Abbildung 34).

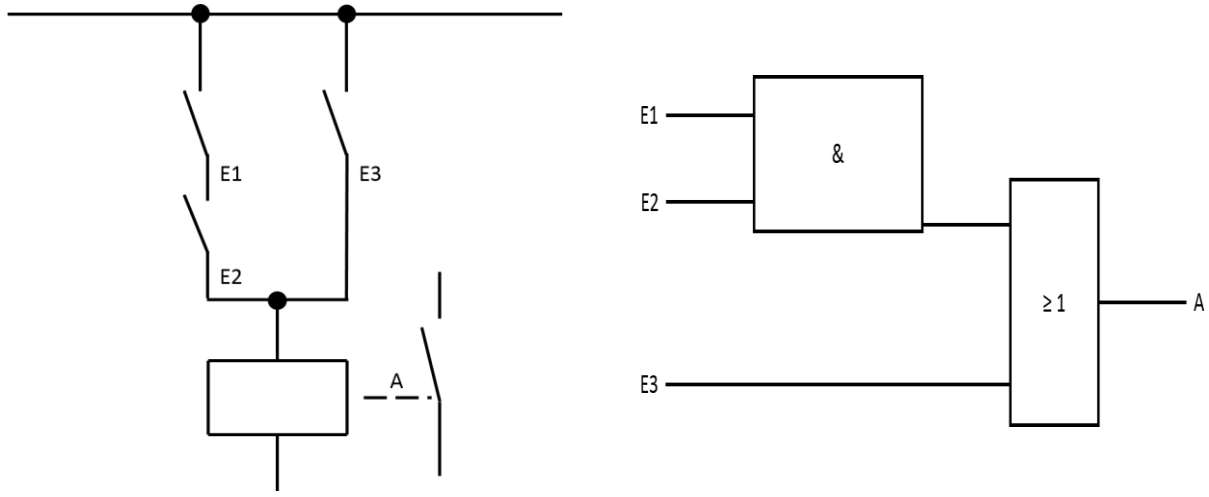


Abbildung 34: Moodle-Aufgabenstellung und Lösungen Nr. 3 FBS

Bei der letzten Übungsaufgabe zum Thema FBS soll der Lernende nun sein bereits erlerntes Wissen nutzen, um eine Transferleistung zu erbringen. Der Lernende soll nun aus dem Funktionsplan die benötigte Wahrheitstabelle ableiten (siehe Abbildung 35 und 36) [Benda 1983].

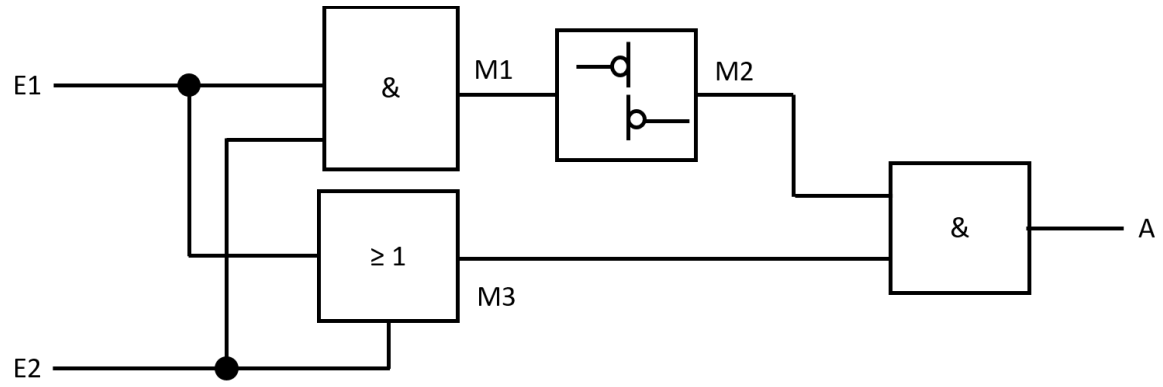


Abbildung 35: Moodle-Aufgabenstellung Nr. 4 FBS

E1	E2	M1	M2	M3	A
0	0	0	1	0	0
1	0	0	1	1	1
0	1	0	1	1	1
1	1	1	0	1	0

Abbildung 36: Moodle-Lösung Nr. 4 FBS

Die Übungsaufgaben zum Thema KOP sind den FBS-Übungen ähnlich. Zuerst werden die verschiedenen Grundelemente der FBS abgefragt, um darauf aufbauend ausgewählte Übungsaufgabe zu zur Verfügung zu stellen. Dabei ist der erste Aufgabenteil wieder die bereits vorgestellte Übungsaufgabe aus Kapitel 3.1. Die zweite Teilaufgabe ist ähnlich, zu der FBS-Übungsaufgabe (siehe Abbildung 37). Bei dieser Teilaufgabe gibt es zwei Eingangssignale und ein Ausgangssignal [Kaftan 1991].

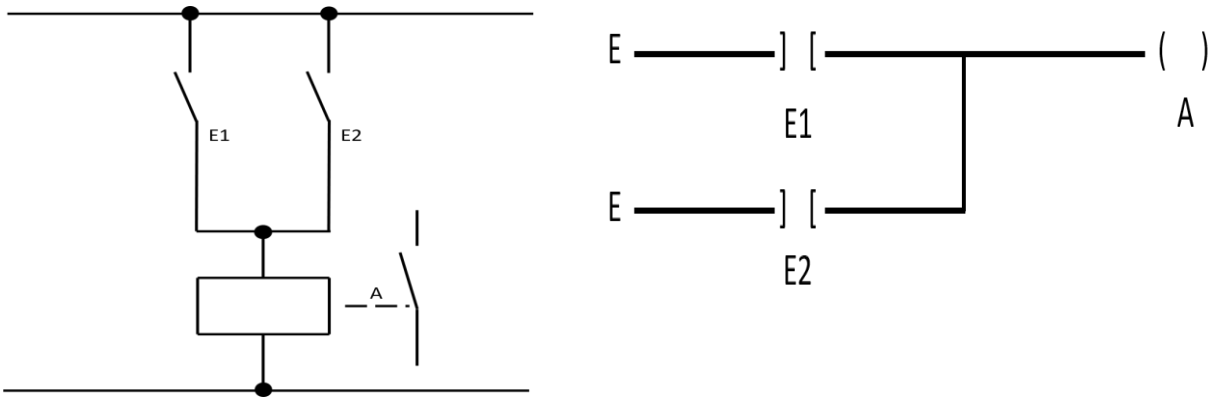


Abbildung 37: Moodle-Übungsaufgabe und Lösungen Nr. 2 KOP

Wie in Abbildung 38 dargestellt, soll die dritte Übungsaufgabe für den Bereich KOP, ähnlich der Übungsaufgabe 3 aus dem Bereich FBS, wieder eine Kombination der vorher bereits bearbeiteten Übungsaufgaben gestellt werden [Kaftan 1991].

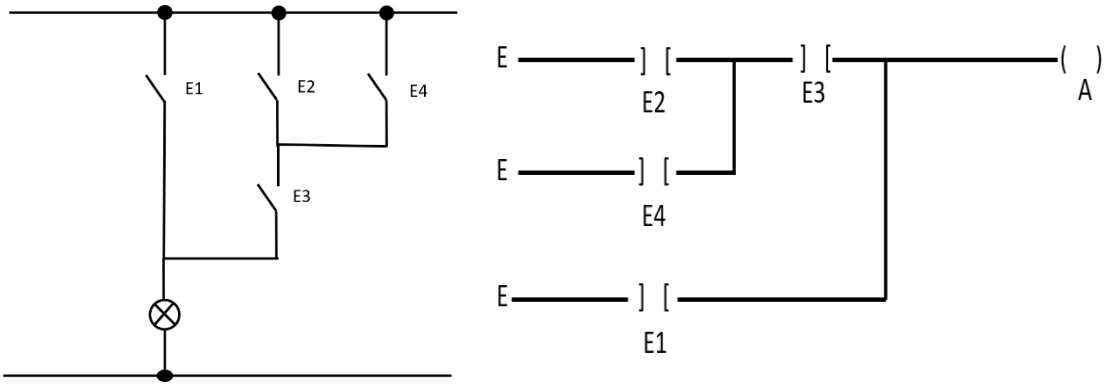


Abbildung 38: Moodle-Übungsaufgabe und Lösung Nr. 3 KOP

Die vierte und letzte Übungsaufgabe für den Bereich KOP in der Moodle-Lerneinheit führt wiederum ein neues Element ein (siehe Abbildung 39) [Kaftan (1991), S. 55]. Das neue Element ist, wie in der FBS-Übungsaufgabe, die Negation.

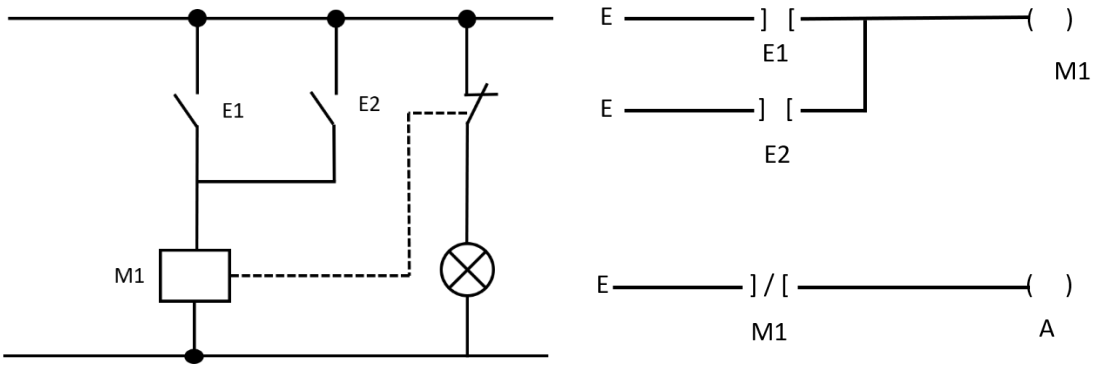


Abbildung 39: Moodle-Übungsaufgabe und Lösung Nr. 4 KOP

5.3 Evaluation der Moodle-Lerneinheit [LH]

Zur Evaluation der Übungsaufgaben der Moodle-Lerneinheit wurde ein Fragebogen entwickelt. Dieser Fragebogen konnte ausschließlich von Studierendender Technischen Universität Dortmund durchgeführt werden, da hierzu ein Moodle-Account obligatorisch war. Um die Testphase zu beginnen, wurde auf verschiedenen sozialen Medien wie WhatsApp oder studentischen Plattformen nach Probanden gesucht. Für die Evaluation wurden neun Probanden gefunden. Diese wurden anschließend den Moodle-Kurs hinzugefügt. Um die Evaluation durchzuführen, sollten die Probanden die verschiedenen Übungsaufgaben bearbeiten. Die Evaluation begann mit der Einführung des Probanden in das Skalenniveau der einzelnen Übungsaufgaben. Der Proband hatte die Möglichkeit die folgenden Fragen mit 1 bis 3 zu beantworten, wobei 1 als nicht zustimmend und 3 als zustimmend interpretiert wird. Ziel der Evaluation war es, unterschiedliche Meinungen zu den möglichen Darstellungsformen der Aufgaben zu bekommen.

Die Probanden wurden dabei nach ihren Einschätzungen zu den Themen Wiederholung des Erlernten, Übersichtlichkeit und Komplexität der Übungsaufgaben befragt. Die Probanden wurden dabei erst nach ihrer Einschätzung zum Fragentyp Multiple-Choice und anschließend zum Fragentyp Drag-and-Drop befragt.

Die ersten beiden Fragen haben das Ziel die Fragentypen auf die Möglichkeiten der Wiederholung des Erlernten zu testen. Aufgaben mit dem Typ Multiple-Choice wurden durchschnittlich mit 2,00 bewertet und die Drag-and-Drop-Aufgabe mit 2,89. Die Evaluation zeigt, dass die Drag-and-Drop-Aufgaben mit Ausnahme von einer Bewertung mit der maximalen Punktzahl bewertet wurde. Durch die beiden Werte kann darauf geschlossen werden, dass die Probanden Drag-and-Drop-Aufgaben zur Wiederholung von Erlerntem Wissen favorisieren. Die nächsten beiden Fragen behandelten das Thema Übersichtlichkeit der beiden Darstellungsformen. Die Evaluation ergab auch in Bezug auf die Übersichtlichkeit, dass der Fragentyp Drag-and-Drop die Probanden mehr angesprochen hat als die Multiple-Choice-Aufgaben. Die Drag-and-Drop-Aufgaben wurden im Durchschnitt mit 2,67 bewertet. Die Multiple-Choice-Aufgaben wiederum wurden mit einem Wert von 2,0 bewertet. Die fünfte und sechste Frage befasste sich mit der Komplexität der Fragentypen. Die Probanden empfanden die Multiple-Choice-Aufgaben dabei als komplexer als die Drag-and-Drop-Aufgaben, obwohl die Aufgabenstellungen immer identisch waren. Die Komplexität der Multiple-Choice-Aufgaben wurde durch die Probanden mit 2,00 bewertet, die Komplexität der Drag-and-Drop-Aufgaben mit 1,67. Einige Probanden haben zusätzlich die Möglichkeit genutzt Ergänzungsvorschläge oder Auffälligkeiten

anzumerken. Einem Probanden ist die aufbauende Komplexität der Übungsaufgaben aufgefallen. Der Proband merkte an, dass die ersten beiden Übungsaufgaben eine gute Einführung wären. Dies ist wie vorher bereits beschrieben, laut Patschenass et al. ein guter Aufbau für Übungsaufgaben [Patschenass et al. 2004]. Ein weiterer Proband merkte an, dass die bereits durchgeführten Testergebnisse weiterhin angezeigt werden sollen, um den Lernfortschritt verfolgen zu können. Laut Haake et al. ist Feedback und die Sichtbarkeit eines Fortschrittes beim Lernen ein wichtiger Bestandteil zum Motivieren der Lernenden [Haake et al. 2020]. In Online-Kursen ist es gerade entscheidend Feedback zu integrieren [Hartung 2017], daher wird diese Idee als sinnig angesehen und sollte übernommen werden.

Insgesamt kann nach Auswertung der Mittelwerte der Evaluationsfragen angenommen werden, dass die Probanden die Darstellungsform der Drag-and-Drop-Aufgaben als eine bessere Darstellungsform für SPS-Übungen ansehen. Aufgrund der Einschätzungen der Probanden wurden Drag-and-Drop-Aufgaben als weniger komplex, übersichtlicher und als eine gute Möglichkeit erlerntes Wissen zu wiederholen eingeschätzt. Aus diesem Grund wird die interaktive Darstellungsform Drag-and-Drop für weitere Moodle-Umgebungen mit Übungsaufgaben vorgeschlagen.

6 Zusammenfassung

Mit dieser Projektarbeit sollten Übungsaufgaben zum Thema der grafischen Programmiersprachen innerhalb der SPS zum Einsatz im Lehrbetrieb erstellt werden. Zusätzlich sollte eine Moodle-Lerneinheit geschaffen werden, wodurch die Grundlagen der grafischen Programmiersprachen vermittelt und das Wissen der Studierenden, durch zusätzliche Aufgaben, vertieft werden kann. Mithilfe von aufgestellten Kriterien wurden einige bereits bestehende Übungsaufgaben aus der Literatur bewertet und neue geeignete Übungsaufgaben für die Präsenzübung erarbeitet. Bei der Moodle-Lerneinheit haben sich die zwei Möglichkeiten Drag-and-Drop und Multiple-Choice zur Abfrage der Studierenden herauskristallisiert. Anhand einer Evaluation mit mehreren Studierenden der TU Dortmund konnte eine Empfehlung zu einer der beiden Möglichkeiten gegeben werden.

Um geeignete Übungsaufgaben auf dem Gebiet der grafischen Programmiersprachen innerhalb von SPS zu erstellen, war es im Vorhinein wichtig die SPS im zweiten Kapitel zu definieren und die grafischen Programmiersprachen von den textuellen Sprachen und der Mischform abzugrenzen. Im Anschluss wurden im dritten Kapitel bereits bestehende Übungsaufgaben aus der Literatur vorgestellt und Kriterien zur Bewertung der vorgestellten Übungsaufgaben erarbeitet. Anhand dieser Kriterien wurde jede vorgestellte Übungsaufgabe auf Eignung zum Einsatz im Lehrbetrieb bzw. in der Moodle-Lerneinheit geprüft. Nachdem aus den vorherigen Kapiteln alle wichtigen Vorbereitungen abgeschlossen wurden, konnten im vierten Kapitel die Vorgehensweise und die Lösung der geeigneten Übungsaufgabe aus der Literatur und der eigenen erarbeiteten Übungsaufgaben für die Präsenzübung vorgestellt werden. Abschließend wurden im fünften Kapitel die Möglichkeiten zur Übertragung von Übungsaufgaben in die Moodle-Lerneinheit aufgezeigt und der Aufbau, sowie die eingebetteten Aufgaben vorgestellt. Da sich zwei Möglichkeiten, Drag-and-Drop und Multiple-Choice, zur Darstellung der Übungsaufgaben innerhalb von Moodle ergeben haben, wurde mithilfe einer Evaluation mit mehreren Studierenden von der TU Dortmund eine Empfehlung ausgesprochen, welche Möglichkeit der Darstellung sich eher eignet.

Es ist zu erwähnen, dass die Übungsaufgaben aus dem vierten Kapitel sich nur anhand der aufgestellten Kriterien aus dem dritten Kapitel besonders gut eignen. Durch eine Änderung oder Erweiterung der Kriterien könnte der Fall eintreten, dass die Übungsaufgaben nicht mehr geeignet wären bzw. abgeändert werden müssten. Zur Moodle-Lerneinheit lässt sich sagen, dass nur die Darstellungsmöglichkeiten Drag-and-Drop und Multiple-Choice betrachtet wurden und

prinzipiell weitere Darstellungsmöglichkeiten, neben Drag-and-Drop und Multiple-Choice, vorstellbar wären. Des Weiteren ergab sich bei der Evaluation zwar eindeutig, dass Drag-and-Drop als Darstellungsmöglichkeit von den Studierenden bevorzugt wurde, allerdings wurde die Evaluation mit nur neun Personen durchgeführt, wodurch die Aussagekraft geschwächt ist. Um die Aussagekraft der Evaluation zu stärken, wäre eine weitere Evaluation mit einer größeren Anzahl an Teilnehmern sinnvoll. Ebenfalls könnte man prüfen, ob eine Kombination der Darstellungsmöglichkeiten in Moodle eine bessere Alternative darstellen würde.

Literaturverzeichnis

Bauernhansl, T.; ten Hompel, M.; Vogel-Heuser, B. (2017): *Handbuch Industrie 4.0*, 2. Auflage, Berlin: Springer Vieweg.

Benda, D. (1986): *Speicherprogrammierbare Steuerung – SPS für Praktiker*, 1. Auflage, Sindelfingen: expertverlag.

Bernstein, H. (2018): *Speicherprogrammierbare Steuerung – SPS*, 1. Auflage, Berlin/Bosten: Walter de Gruyter GmbH.

Greefrath, G. (2004): *Offene Aufgaben mit Realitätsbezug – Eine Übersicht mit Beispielen und erste Ergebnisse aus Fallstudien,* in: *mathematica didactica*, Jg. 2, Nr. 27, S. 16-38.

Grötsch, E. (2004): *SPS - Speicherprogrammierbare Steuerungen als Bausteine verteilter Automatisierung*, 5. Auflage, München: Oldenbourg Industrieverlag GmbH.

Hartung, S. (2017): *Lernförderliches Feedback in der Online-Lehre gestalten*, in: Griesehop, H. R. ; Bauer, E. (Hrsg.), *Lehren und Lernen online*, Wiesbaden: Springer, S. 199–217.

IEC 61131-3 (2003): *Speicherprogrammierbare Steuerungen-Teil 3: Programmiersprachen*, Deutsche Fassung EN 61131-3:2003.

John, K. H.; Tiegelkamp, M. (2009): *SPS-Programmierung mit IEC 61131-3*, 4. Auflage, Berlin/Heidelberg: Springer Verlag.

Kaftan, J. (1991): *SPS-Grundkurs: Anleitungen, Übungen, Lösungen*, 3. Auflage, Würzburg: Vogel.

Kraus, A.; Nieweler, A. (2011): *La tâche: von der Übung zur Aufgabe – Kompetenzentwicklung und Aufgabenorientierung,* in: *Der fremdsprachige Unterricht Französisch*, Jg. 45, Nr. 112, S. 5.

Landesinstitut für Schule (2005): *Standardorientierte Unterrichtsentwicklung – Moderatorenmanual Deutsch – Modul 2: Aufgaben konstruieren*, 1. Auflage, Soest: Landesinstitut für Schule/Qualitätsagentur.

Lerch, R. (2016): *Elektrische Messtechnik*, 7. Auflage, Berlin/Heidelberg: Springer Verlag.

Meier, U.; Kleinknecht, M.; Metz, K.; Bohl, T. (2010): *Ein allgemeindidaktisches Kategoriesystem zur Analyse des kognitiven Potenzials von Aufgaben*, in: *Beiträge zur Lehrerinnen- und Lehrerbildung*, Jg. 28, Nr. 1, S. 84-96.

Petschenka, A.; Kerres, N.; Kerres, M. (2004): *Lernaufgaben beim E-Learning,* in: Hohenstein, A.; Wilbers, K. (Hrsg.), *Handbuch E-Learning,* Köln: Fachverlag Deutscher Wirtschaftsdienst, Kapitel 4.19.

Pickhardt, R. (2000): *Grundlagen und Anwendung der Steuerungstechnik*, 1. Auflage, Braunschweig/Wiesbaden: Vieweg.

Plaßmann, W.; Schulz, D. (2016): *Handbuch Elektrotechnik – Grundlagen und Anwendungen für Elektrotechniker*, 7. Auflage, Wiesbaden: Springer Fachmedien.

Rabe, M. (2019a): *SPS-Programmierung – Informationsaustausch produzierender Unternehmen – Übung 1*, Fachgebiet IT in Produktion und Logistik, TU Dortmund.

Rabe, M. (2019b): *SPS-Programmierung – Informationsaustausch produzierender Unternehmen – Vorlesung 5 – SPS-Programmierung*, Fachgebiet IT in Produktion und Logistik, TU Dortmund.

Ten Hompel, M. (2018): *Speicherprogrammierbare Steuerung* – IT-Systeme in Produktion und Logistik 1, Lehrstuhl für Förder- und Lagerwesen, TU Dortmund.

Wellenreuther, G.; Zastrow, D. (2015): *Automatisieren mit SPS – Theorie und Praxis*, 6. Auflage, Wiesbaden: Springer Fachmedien.

Wellenreuther, G.; Zastrow, D. (2007): *Automatisieren mit SPS – Übersichten und Übungsaufgaben*, 3. Auflage, Wiesbaden: Vieweg & Sohn Verlag.

Wolf, P; Biehler, R. (2014): *Entwicklung und Erprobung anwendungsorientierter Aufgaben für Ingenieurstudienanfänger/innen,* in: Zeitschrift für Hochschulentwicklung, Jg. 9, 2014, Nr. 4, S. 169-190.

Anhang

Antwort	Empfanden Sie die Multiple-Choice-Aufgabe als gute Möglichkeit das Erlernte zu wiederholen?	Empfanden Sie die Multiple-Choice-Aufgabe als gute Möglichkeit das Erlernte zu wiederholen?
1	3	3
2	1	3
3	1	3
4	1	3
5	2	3
6	2	3
7	3	3
8	2	2
9	3	3

Antwort	Empfanden Sie die Multiple-Choice-Aufgabe als übersichtlich?	Empfanden Sie die Multiple-Choice-Aufgabe als übersichtlich?
1	3	1
2	1	3
3	1	3
4	1	3
5	3	3
6	3	2
7	2	3
8	1	3
9	3	3

Antwort	Empfanden Sie die Multiple-Choice-Aufgabe als komplex?	Empfanden Sie die Multiple-Choice-Aufgabe als komplex?
1	1	1
2	2	1
3	3	1
4	3	2
5	1	2
6	2	3
7	2	2
8	2	1
9	2	2