

Beschreibung des Handling Application Beispieles

Inhalt

1	Einleitung.....	2
1.1	Programmiersprachen.....	3
1.1.1	Instruktionsliste (IL – Editor).....	3
1.1.2	Graftec	3
1.1.3	Funktionsplan (FUPLA – Editor).....	4
1.1.4	FBox Builder.....	4
1.2	Strukturierung.....	5
2	Das Beispielprojekt.....	6
2.1	Modellbeschreibung	6
2.1.1	Schnittstelle zu andern Stationen.....	7
2.2	Installation des Projekts	7
2.3	Bedienung	7
2.3.1	Betriebsarten.....	7
2.3.2	Manuelles Verfahren der Zylinder	7
2.3.3	Reset der Station	8
2.4	Fehlerüberwachungen.....	8
3	Vorgehensweise	9
3.1	Vorbereitungen.....	9
3.1.1	Unterteilung der Anlage in Stationen.....	9
3.1.2	Erstellen der Ein- und Ausgangslisten	10
3.2	Programmierung der SPS	12
3.2.1	Strukturierung der Software	12
3.2.2	Aufteilung der Module	12
3.2.3	Module der Stationen 1 .. n.....	13
3.2.4	Vorteile dieser Art der Strukturierung	14
3.2.5	Nachteile dieser Art der Strukturierung	14
3.2.6	Informationsfluss innerhalb einer Station	15
3.3	Strukturierung des Beispielprojektes	15
3.3.1	Hauptprogramm	15
3.3.2	Übergabemodul Station 1.....	16
3.3.3	Wahl der Werkzeuge.....	16
3.3.4	Codierung: Allgemeines	17
3.3.5	Codierung: Ausgangsbeschaltung	18
3.3.6	Codierung: Bedienung	19
3.3.7	Codierung: Sequenz	21
3.3.8	Codierung: Störauswertung.....	22
3.3.9	Codierung: Störmeldungen aufzeichnen	23
3.3.10	Codierung: Störmeldungen per SMS absetzen	24
3.4	Bedienung mit Web-Editor	25
3.5	Bewegungsabläufe.....	26
3.5.1	Würfel in richtiger Lage	26
3.5.2	Würfel in falscher Lage	27
3.5.3	Würfel in gedrehter Lage.....	28

1 Einleitung

Dieses Dokument beschreibt ein Musterbeispiel einer industriell automatisierten Anlage und soll ebenfalls als Vorlage für größere Projekte dienen.

Ziel dieser Schulungsunterlagen ist nicht die Beschreibung des Programmierwerkzeuges PG5 sondern die Wahl der richtigen Programmiersprache mit den entsprechenden Editoren und eine sinnvolle Strukturierung ihres Projektes. Das hier behandelte Beispiel zeigt, wie eine Übergabestation (eine „Pick and place“-Anlage) programmiert werden kann. Die präsentierte Station greift ein Werkstück, dreht es gegebenenfalls und platziert es auf einem Drehtisch.

Die Sensoren sind alle digital und die Aktuatoren sind pneumatisch realisiert (und können demzufolge digital angesprochen werden).

Da in den wenigsten Fällen eine derartige Hardware zur Verfügung steht, ist auch eine zweite Steuerung in das Beispielprojekt integriert. Diese Steuerung dient als Simulator für die Hardware. So wird der Simulator z.B. als Reaktion auf den gesetzten Eingang für ein Zylinderventil nach einer Verzögerung das Signal des Endschalters zurückgeben. Diese Konstellation ermöglicht es auch, Kabelbrüche zu simulieren und somit das Verhalten des Systems in „Ausnahmesituationen“ zu testen.

1.1 Programmiersprachen

Etwa wie Nägel, die Sinnvollerweise mit einem Hammer versenkt, und Schrauben welche mit Hilfe von Gabelschlüssel oder Schraubendreher fixiert werden, sind im PG5 verschiedene Werkzeuge für unterschiedliche Aufgaben enthalten.

Die Wahl des geeigneten Werkzeuges richtet sich nach den gestellten Aufgaben, der gewünschten Form der Dokumentation und den Stärken und Neigungen des Programmierers.

In diesem Projekt wurde darauf geachtet, jeweils die effizienteste Programmiersprache für die verschiedenen Aufgaben zu wählen. Demzufolge sind relative viele Funktionen in Instruktionsliste (IL) realisiert.

1.1.1 Instruktionsliste (IL – Editor)

Der IL- Editor ist ein sehr effizientes, textbasiertes Werkzeug, das jedoch gewisse Grundkenntnisse über die einzelnen Befehle voraussetzt. Für den Einsteiger werden in der Anfangsphase die Online-Hilfe und die „IL-Instruktionen-Übersicht“ wichtige Hilfsmittel sein. Darin finden sie die Beschreibungen der Befehle, deren Syntax und auch Beispiele.

Die Effizienz der IL Programmierung liegt unter anderem darin begründet, dass dasselbe Instruktions-Set verwendet wird, wie auch tatsächlich von der Steuerung abgearbeitet wird. Somit wird exakt der vom Programmierer erstellte Code von der Steuerung abgearbeitet.

1.1.2 Graftec

GRAFTEC ist ein graphisches Werkzeug zur einfachen Erstellung, Verwaltung und Dokumentation von sequenziellen Abläufen. Die einzelnen Weiterschaltbedingungen (Transitionen) und Aktionen (Steps) können im IL- Editor sowie FUPLA- Editor erstellt werden. Diese Struktur ermöglicht das einfache und übersichtliche „online-Beobachten“ des Programms auf der Steuerung.

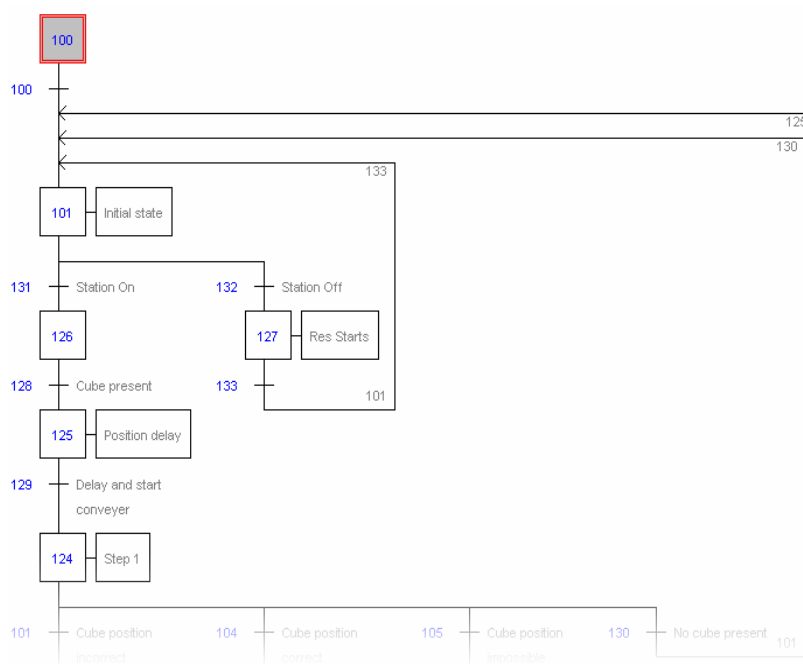


Abbildung 1: Graftec Screenshot

1.1.3 Funktionsplan (FUPLA – Editor)

FUPLA in PG5 ist ein graphisches Programmierwerkzeug. Die Funktionspalette umfasst einfache Elemente für die Kontaktplan- Programmierung, Binär-, Ganzzahl- und Fliesspunkt-Funktionen bis zu sehr komplexen Wandlungs- Kommunikations- und Regelfunktionen.

Da Fupla Programme mit so genannten FBoxen (Funktionsbausteine, welche miteinander verbunden werden) grafisch programmiert werden, ist eine entsprechende Programmseite rasch interpretiert.

Ein weiterer Vorteil an den FBoxen ist, dass mit wenigen (oder sogar nur einer) spezifisch(en) FBox(en) eine sehr komplexe Funktion programmiert werden kann. Je nach Bedarf können spezielle Bibliotheken verwendet werden, so z.B. die HLK (Heizung-Lüftung-Klima-) Bibliothek. Diese Bibliothek ist speziell für die Gebäudeautomation entwickelt worden und umfasst FBoxen für die Regelung, Zeitschaltuhren etc.

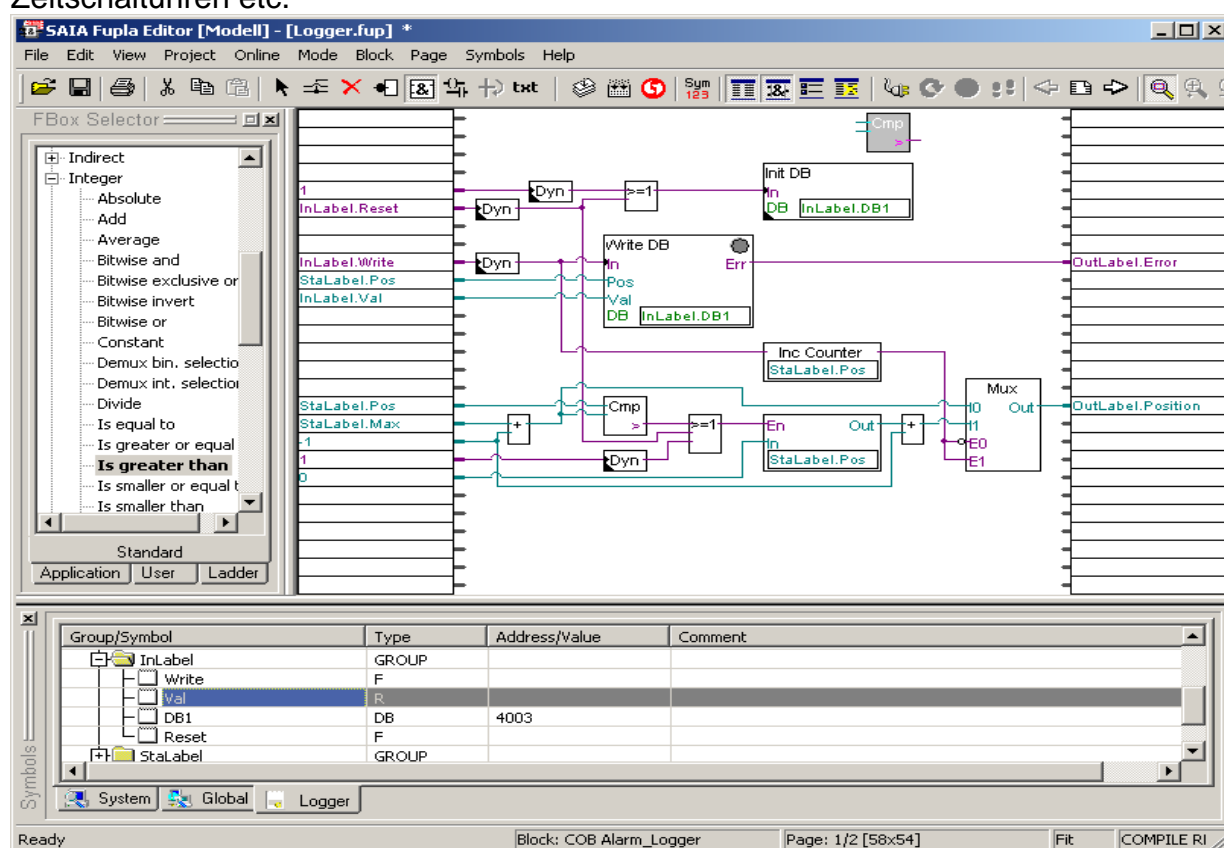


Abbildung 2: Fupla Editor (SFup)

Die Programmdatei Modem_SMS.fup sowie Logger.fup sind mit Fupla realisierte Programmmodule und ist zuständig für das Versenden von SMS Nachrichten im Störfall sowie das Speichern von aufgetretenen Alarmen.

1.1.4 FBox Builder

Jede FBox beinhaltet (für den Benutzer nicht sichtbaren) Code, der die gewünschte Funktion mittels dem IL-Befehlssatz realisiert.

Mit Hilfe des Tools FBox Builder (Standard Version) können in Fupla realisierte Programmteile zu einer neuen FBox zusammengefasst werden. Das erlaubt die einfache Wiederverwendung eines einmal getesteten Programmteils. Da der interne Code jeweils identisch bleibt, werden dadurch Flüchtigkeitsfehler bei der

Ausprogrammierung dieser Unterfunktion ausgeschlossen. Als Beispiel dafür können die FBoxen der Programmdatei Logger.fup betrachtet werden. Die FBoxen mit dem Namen Logger ist die unten ersichtliche, zusammengefasste Fupla Seite. Anstelle der ganzen Seite kann einfach die FBox Logger verwendet werden.

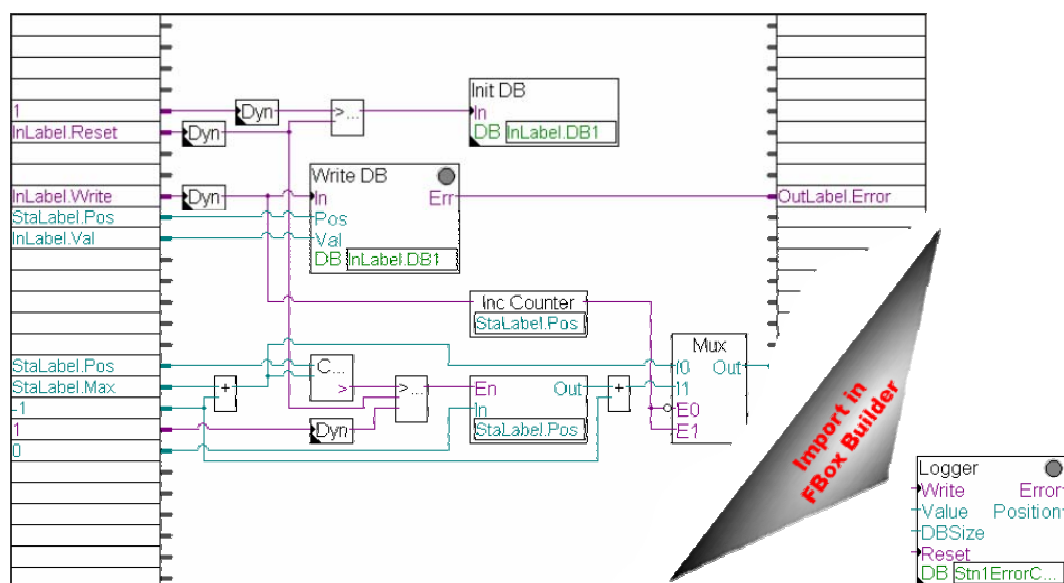


Abbildung 3: FBox kreiert mit FBox Builder

Es ist auch möglich, selbst programmierte IL-Bausteine (Funktionen) in FBoxen zu integrieren, um diese grafisch in ein Programm zu platzieren. Für diese Anwendung ist allerdings eine spezielle Lizenz erforderlich (FBox Builder Full Version)

1.2 Strukturierung

Eine gute Strukturierung der Programme ist sehr wichtig. Dadurch wird ermöglicht, sehr komplexe und umfangreiche Funktionalitäten einer Steuerung in kleinen Schritten zu implementieren und modular zu modifizieren.

Die folgenden Punkte werden durch die Strukturierung maßgeblich beeinflusst:

- Unterteilung der Anlage in kleine, überschaubare Einheiten
- Definition von klaren Schnittstellen zwischen den einzelnen Einheiten
- Verbesserung der Lesbarkeit, Verständlichkeit & Dokumentation von Programmen
- Reduktion von Fehlerquellen
- Vereinfachung bei der Fehlersuche
- Abhängigkeiten sind klar geregelt und übersichtlich
- Wiederverwendbarkeit von Programmcode (Programm-Modulen)

2 Das Beispielprojekt

Ziel dieses Projektes ist die Vorgehensweise bei der Erstellung eines SPS-Projektes zu beschreiben und Lösungsansätze aufzuzeigen. Es soll der Werdegang eines SPS-Projektes von der Aufgabenstellung bis zur Inbetriebnahme der Anlage beschrieben werden. Im Vordergrund steht weniger die Programmierung im Detail, als die Strukturierung des Projektes und die übersichtliche Darstellung der einzelnen Teilaufgaben.

Dazu gehen wir in den nachfolgenden Kapiteln immer vom gleichen, unten beschriebenen Modell aus.

2.1 Modellbeschreibung

Das Modell stellt eine Übergabestation dar, die ein Werkstück greift und lagerichtig auf einen Rundtisch übergibt.

Das Werkstück ist ein Würfel mit einer Bohrung durchs Zentrum. Die Bohrung ist so auszurichten, dass sie gemäß nachfolgender Zeichnung in der Aufnahme des Rundtisches positioniert werden soll. Sollte die Bohrung vertikal verlaufen, ist ein Drehen des Würfels in die richtige Lage nicht möglich und das Teil wird in der Mittelstellung des horizontalen Zylinders abgeworfen.

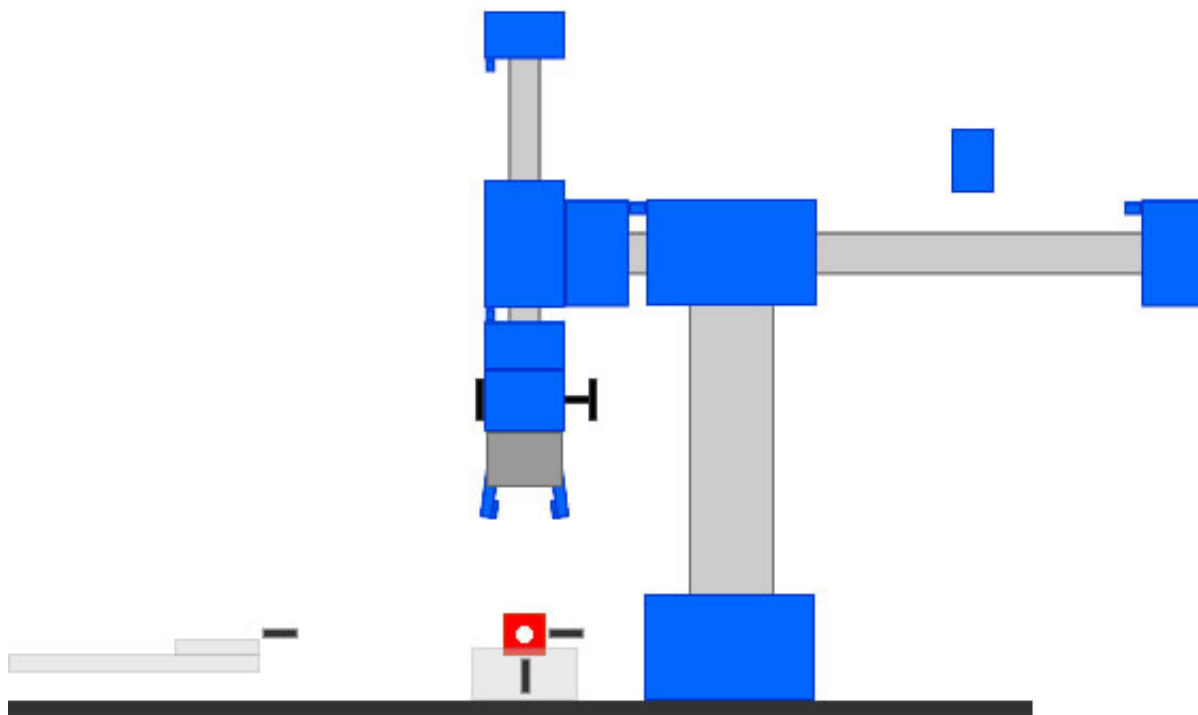


Abbildung 4: Ansicht Übergabestation

2.1.1 Schnittstelle zu andern Stationen

Die Schnittstelle zum Rundtisch wird mit zwei Signalen gemäß folgendem Diagramm realisiert.

Da auch der Rundtisch nur virtuell ist, wird das Signal „Rundtisch drehen“ über das Web Interface der Modellstation realisiert. Die entsprechende Taste (Start von Rundtisch) befindet sich auf der Bedienseite (Operations) des Web Projekts auf der Steuerung.

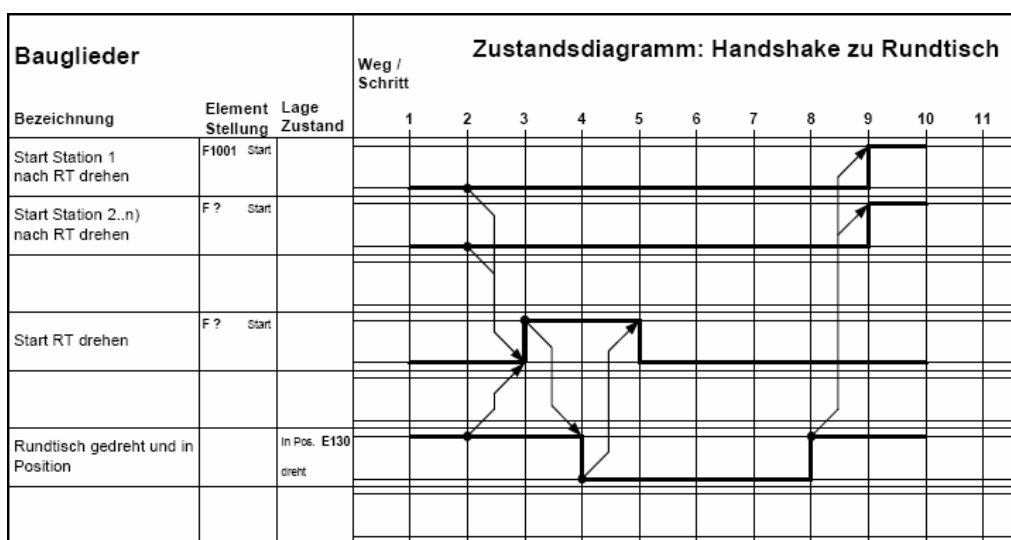


Abbildung 5 Übergabebedingungen

2.2 Installation des Projekts

Bitte beziehen Sie sich für die Installation dieses Projekts auf das beiliegende Dokument „Handling_Application_Installation“. Dieses Dokument beschreibt das Vorgehen für die Installation sowie die Inbetriebnahme dieses Projekts.

2.3 Bedienung

Das Konzept der Bedienung stammt aus der Praxis und hat sich auch an grossen Anlagen mit mehreren Stationen schon bestens bewährt.

2.3.1 Betriebsarten

Die Bedienung der Station ist in folgenden Betriebsarten zu realisieren:

- Manuell (schrittweise auf Tastendruck)
- Zyklus (ein kompletter Ablauf pro Tastendruck)
- Automatik (automatisch starten bei Freigabe vom Rundtisch)

2.3.2 Manuelles Verfahren der Zylinder

Zusätzlich sollen alle Zylinder einzeln in ihre verschiedenen Positionen bewegt werden können, sofern die Mechanik dies erlaubt (Crash-Situationen sind zu verhindern). Die Signale der Endschalter sind dabei zu überwachen und wenn nötig entsprechende Fehlermeldungen auszugeben.

2.3.3 Reset der Station

Mit dem Reset einer Station werden alle zugehörigen Zylinder geordnet in ihre Grundstellung gefahren. Das heißt: Mechanische Kollisionen werden verhindert.

2.4 Fehlerüberwachungen

Jeder Sensor ist zu überwachen und im Fehlerfall ist eine detaillierte Fehlermeldung anzuzeigen. Wenn Fehlermeldungen ausgegeben werden, sind die Abläufe der entsprechenden Station zu stoppen. Ein Wiederanlauf ist erst nach Quittierung des Fehlers zu ermöglichen.

3 Vorgehensweise

3.1 Vorbereitungen

Nachdem wir uns einen groben Überblick über die Aufgabe verschafft haben, stellen wir die Informationen zur Hardware zusammen. Als Grundlage dienen uns vor allem die Elektroschemas, Pneumatikschemas und eventuell vorhandene Weg-Schrittdiagramme.

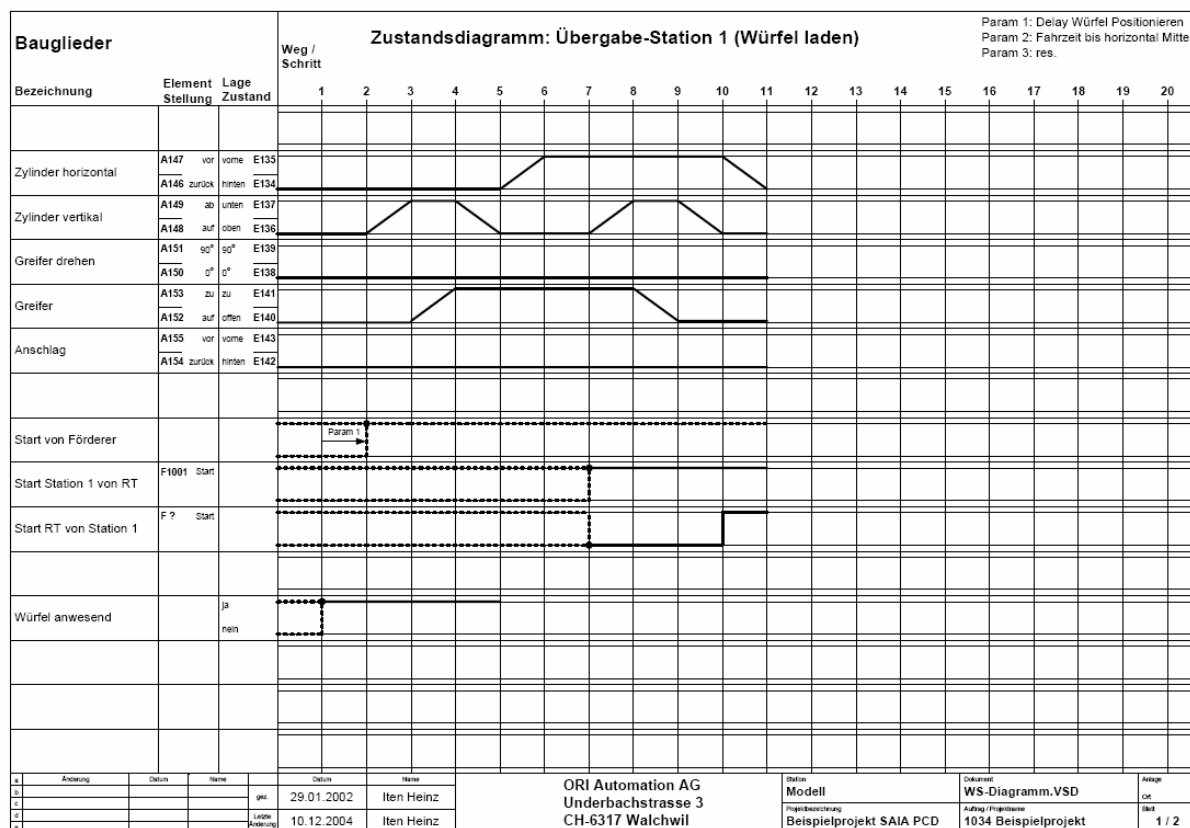


Abbildung 6: Zustandsdiagramm

Eventuell vorhandene Firmenstandards und Vorgaben des Steuerungs- oder Antriebskonzeptes sind ebenfalls zu Beginn der Projektierung einzubringen.

3.1.1 Unterteilung der Anlage in Stationen

Als erstes sollte die gesamte Anlage in Stationen unterteilt werden. Als Stationen verstehen wir Einheiten, die einen abgeschlossenen Ablauf ausführen können. Dies sind zum Beispiel:

- Rundtisch (RT)
- Bestückungshandling
- Stapelvorrichtung
- Etikettierstation
- Beschriftungszelle
- Auf- oder Abwickelvorrichtung
- Presse
- Roboterzelle



All diese Stationen sollen nach dem gleichen Bedienungskonzept bedient werden können. Den Stationen werden die für ihren Ablauf benötigten Zylinder und deren entsprechende Endschalter zugeordnet.

Bei der Unterteilung des Projektes in Teilaufgaben kann neben Werkzeugen wie Excel und Word bereits die Entwicklungsumgebung des PG5 hinzugezogen werden. Dieser Teil der Projektierung ist sehr entscheidend und bestimmt oftmals über die spätere Komplexität des Projektes. Aus diesen Unterteilungen resultieren die Schnittstellen zwischen den einzelnen Softwaremodulen. Weiter sind sie mitbestimmend bei der Festlegung der Datenstrukturen.

3.1.2 Erstellen der Ein- und Ausgangslisten

Da die Zuweisungen der Ein- und Ausgänge auch für die Programmierung benötigt werden, sind die Ein- Ausgangslisten möglichst von Anfang an im Programmierwerkzeug einzugeben. Dazu sollen möglichst selbsterklärende Symbole definiert werden, mit aussagekräftigen Kommentaren. Dies erleichtert später deren Verwendung beim Programmieren.

Häufig lassen sich die Kommentare und Elementbezeichnungen mehr oder weniger direkt aus dem Elektro-CAD importieren.

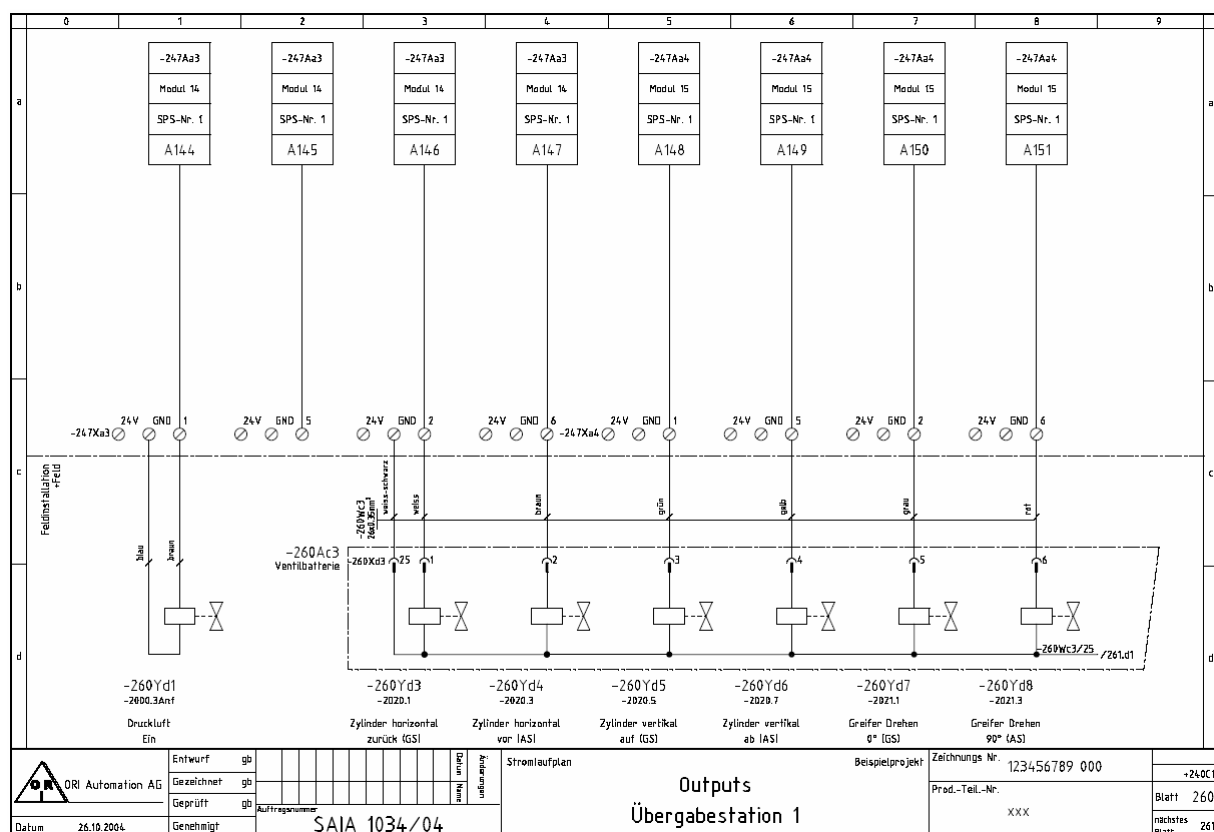


Abbildung 7: CAD Schema



Der Endkunde erhält den größten Nutzen, wenn die Bezeichnungen der SPS-Symbole, die Elektro-Betriebsmittelkennungen und die Klartextanzeigen am Bedienpanel identisch sind.

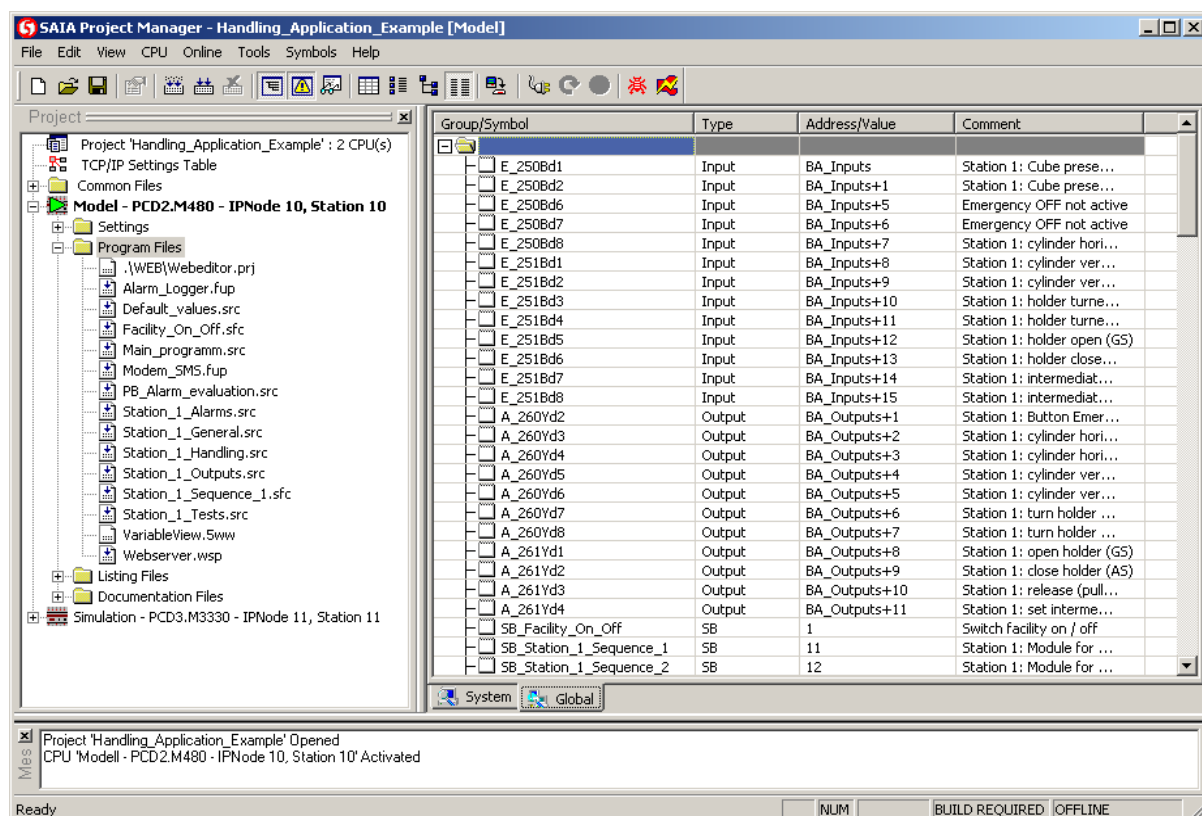


Abbildung 8: PG5 Projekt Manager

Aus der Sicht des Programmierers und der Programmdokumentation kann auch der Symbolname möglichst selbstsprechend gewählt werden. Am Beispiel des Simulators sieht die Symbolliste etwa wie folgt aus:

Group/Symbol	Type	Address/Value	Comment
<input type="checkbox"/> Würfel_anwesend_hinten	Output	16	ES-Sim.: Würfel anwesend von hinten (Bohrung nicht horizontal in Richtung RT)
<input type="checkbox"/> Würfel_anwesend_unten	Output	17	ES-Sim.: Würfel anwesend von unten (Bohrung nicht vertikal)
<input type="checkbox"/> RT_positioniert	Output	18	ES-Sim.: Rundtisch: gedreht und in Position
<input type="checkbox"/> NOT_Aus_inaktiv	Output	21	NOT-Aus OK (nicht betätigt)
<input type="checkbox"/> Horizontal_hinten	Output	22	ES-Simulation: Zylinder horizontal hinten (GS)
<input type="checkbox"/> Horizontal_vorne	Output	23	ES-Simulation: Zylinder horizontal vorne (AS)
<input type="checkbox"/> Vertikal_oben	Output	24	ES-Simulation: Zylinder vertikal oben (GS)
<input type="checkbox"/> Vertikal_unten	Output	25	ES-Simulation: Zylinder vertikal unten (AS)
<input type="checkbox"/> Greifer_gedreht_0_grad	Output	26	ES-Simulation: Greifer gedreht auf 0 Grad (GS)
<input type="checkbox"/> Greifer_gedreht_90_grad	Output	27	ES-Simulation: Greifer gedreht auf 90 Grad (AS)
<input type="checkbox"/> Greifer_offen	Output	28	ES-Simulation: Greifer geöffnet (GS)
<input type="checkbox"/> Greifer_zu	Output	29	ES-Simulation: Greifer geschlossen (AS)
<input type="checkbox"/> Anschlag_eingezogen	Output	30	ES-Simulation: Zwischenanschlag gelöst (GS)
<input type="checkbox"/> Anschlag ausgefahren	Output	31	ES-Simulation: Zwischenanschlag gestellt (AS)
<input type="checkbox"/> B_Not_Aus_Taste_Statio...	Input	1	Taste NOT-Aus bei Station 1 OK
<input type="checkbox"/> Y_horizontal_zurück	Input	2	Ventilstellung: Zylinder horizontal zurück (GS)
<input type="checkbox"/> Y_horizontal_vor	Input	3	Ventilstellung: Zylinder horizontal vor (AS)
<input type="checkbox"/> Y_vertikal_auf	Input	4	Ventilstellung: Zylinder vertikal auf (GS)
<input type="checkbox"/> Y_vertikal_ab	Input	5	Ventilstellung: Zylinder vertikal ab (AS)
<input type="checkbox"/> Y_drehen_0	Input	6	Ventilstellung: Greifer Drehen zu 0 Grad (GS)
<input type="checkbox"/> Y_drehen_90	Input	7	Ventilstellung: Greifer Drehen zu 90 Grad (AS)
<input type="checkbox"/> Y_greifen_auf	Input	8	Ventilstellung: Greifer öffnen (GS)
<input type="checkbox"/> Y_greifen_zu	Input	9	Ventilstellung: Greifer schließen (AS)
<input type="checkbox"/> Y_Anschlag_loesen	Input	10	Ventilstellung: Zwischenanschlag horizontal lösen (GS)
<input type="checkbox"/> Y_Anschlag_stellen	Input	11	Ventilstellung: Zwischenanschlag horizontal stellen (AS)

Abbildung 9: Symbol Table von PG5

3.2 Programmierung der SPS

Bevor mit dem Codieren begonnen werden kann, ist die Strukturierung der Software vorzunehmen. Da sich Herstellerfirmen von Produktionsanlagen meistens in einem speziellen Gebiet betätigen, sind die Aufgaben und Art der Anlagen oft sehr ähnlich. So lassen diese sich häufig mit ein und derselben Grund-Struktur realisieren.



Aus der Praxis zeigt sich, dass durch neue Anforderungen und Funktionalitäten die Strukturen gepflegt und eventuell sogar erweitert werden müssen. Dies erfordert eine saubere Verwaltung der verschiedenen Fortschrittsstände.

3.2.1 Strukturierung der Software

Aus der Strukturierung der Software ergeben sich automatisch Schnittstellen zwischen den einzelnen Softwaremodulen. Das Ziel bei einer guten Strukturierung liegt darin, die Aufgaben in kleine, überschaubare Module zu unterteilen und dabei möglichst verständliche, einfache Schnittstellen zu erhalten.



Hierbei ist zu beachten, dass zu detaillierte Strukturierung und Unterteilung die Verständlichkeit der Programme nicht verbessert.

Aus der Praxis hat sich gezeigt, dass folgende Aufgaben in separate Module gepackt werden können, ohne allzu aufwändige Schnittstellen zu erhalten.

3.2.2 Aufteilung der Module

In diesem Beispiel sind folgend Module erzeugt worden. Einige der Funktionen können mit verschiedenen Werkzeugen erstellt werden. Bei diesen Dateien ist die Wahl des Werkzeugs hauptsächlich durch die persönlichen Präferenzen des Programmierers beeinflusst.

Aufgabe	Werkzeug	Beschreibung
Hauptprogramm (Main_Program.src)	IL	Allgemeine Funktionen wie Blinkerflag, etc.
Start der Anlage (Facility_On_Off.sfc)	Graftec	Aufstarten und stoppen der gesamten Anlage. Das beinhaltet z. B.: Druckluft einschalten - Druckaufbau abwarten - Hydraulikaggregat, Transportbänder und andere Motoren gestaffelt einschalten und überwachen - geordnetes Abschalten der Anlage etc.
Bedienung allgemein (nicht verwendet)	IL	Spezielle Funktionen der Bedienung für die gesamte Anlage wie z.B.: Editieren der Systemzeit am OP
Analoge Eingänge (nicht verwendet)	IL oder Fupla	Zyklisches lesen und skalieren der analogen Eingänge
Analoge Ausgänge (nicht verwendet)	IL oder Fupla	Zyklisches wandeln der Stellwerte und Ausgeben an die analogen Ausgänge
Störsammenfassung (nicht verwendet)	IL oder FUPLA	Zusammenfassen der Störungen aller Stationen
Störüberwachungen	IL oder	Störüberwachung mit zeitlicher

PB_Alarm_evaluation.src	AWL	Verzögerung setzen gemäss Bits der einzelnen Stationen
Alarm Logging auf SPS (Alarm_Logger.fup)	Fupla	Speichern von Alarmen mit Zeitstempel
Alarmierung per SMS (Modem_SMS.fup)	Fupla	Senden von Störmeldungen per SMS an Operator
Funktionen 1..n	Diverse	Verschiedene Funktionen die mit unterschiedlichen Parametern von verschiedenen Stationen aufgerufen werden können

3.2.3 Module der Stationen 1 .. n

Aufgabe	Werkzeug	3.2.3.1 Beschreibung
Station_1_General	AWL	Aufrufen der verschiedenen Funktionen dieser Station wie nachfolgend beschrieben. <ul style="list-style-type: none"> • Tages- und Totalzähler bearbeiten. • Grundstellung kontrollieren. • Kritische Zustände kontrollieren wie z.B.: Freigabe Rundtisch von Station, Freigabe WT von Station
Station_1_Handling	AWL	Bedienung der Station, setzen der Betriebsart, Weiterschaltbedingung für die Stations-Sequenzen
Station_1_Sequence_1	Graftec	1. Ablaufsequenz der Station
Station_1_Sequence_n	Graftec	Weitere Ablaufsequenzen
Station_1_Outputs	AWL	Ausgänge der Station anhand der Betriebsart und des Arbeitsfortschrittes ansteuern
Station_1_Alarms	AWL	Störüberwachung mit eventueller zeitlicher Verzögerung setzen für alle Betriebsarten

Beim genaueren Betrachten der obigen Tabelle fällt Ihnen vielleicht auf, dass die Ausgänge und Störungen in separaten Modulen untergebracht sind und für alle Betriebsarten gelten sollen.

Sie fragen sich vielleicht:



Werden so die Schnittstellen zwischen der Bedienung, den Sequenzen, der Ausgangsbeschaltung und den Störungen nicht viel zu komplex?



Nein:

Die Bedienung am OP oder Terminal setzt in der SPS nur Bedienungsbits (Flags) und keine Ausgänge. Der Entscheid, ob Ausgänge geschaltet werden dürfen, liegt immer in der Gewalt der SPS.

Die Sequenzen steuern Statusbits (Flags) in der SPS, die den gewünschten Status des Ausganges darstellen und nicht direkt den Ausgang schalten.

Im Modul „Station_x_Outputs“ werden die Ausgänge nicht nur anhand des

gewünschten Status aus den Sequenzen gesetzt. Darin können auch mechanische Kollisionen während dem manuellen Zylinderfahren oder einem geordneten Reset verhindert werden.



Die Störungen werden losgelöst vom Ablauf erfasst. Im Allgemeinen sind Störungen falsche Sensorsignale bei bestimmten Aktorzuständen und können unabhängig von der Betriebsart kontrolliert werden. Das heisst: Wenn ein Vertikalmodul (z.B. Zylinder) nach oben angesteuert wird, sollte zeitverzögert der untere Endschalter lösen und der obere Endschalter ansprechen. Ist dies nach einer definierten Zeit nicht der Fall, soll eine entsprechende Störungsanzeige ausgegeben werden (da offenbar das mechanische Bewegen des Zylinders verhindert wurde).

3.2.4 Vorteile dieser Art der Strukturierung

- Die Aufgaben werden ganz klar gewissen Modulen zugeordnet und somit nur an einer Stelle behandelt.
- Wer diese Struktur kennt, findet sich im Programm schneller zurecht.
- Erweiterungen und Korrekturen sind nicht an mehreren Stellen nötig (Fehlerquellen werden minimiert).
- Es gibt nur eine Ablaufsequenz für alle Betriebsarten.
- Bei der Inbetriebnahme muss nicht jede Betriebsart separat getestet werden.
- Fehlersuche wird vereinfacht, wenn die Ausgänge nur an einer Stelle gesetzt werden.
- Die Sicherheit von kritischen Situationen wie z.B. Drehen des Rundtisches kann durch zusätzliche Abfrage der Merker „Grundstellung_RT“ aller beteiligten Stationen erhöht werden. Diese werden zyklisch und sequenzunabhängig aufbereitet. So wird auch ein Druckverlust und Verlassen des Endschalters, nach korrekter Beendigung der Sequenz, erkannt.

3.2.5 Nachteile dieser Art der Strukturierung

- Bei sehr kleinen, einfachen Aufgaben ist der minimale Aufwand grösser.

3.2.6 Informationsfluss innerhalb einer Station

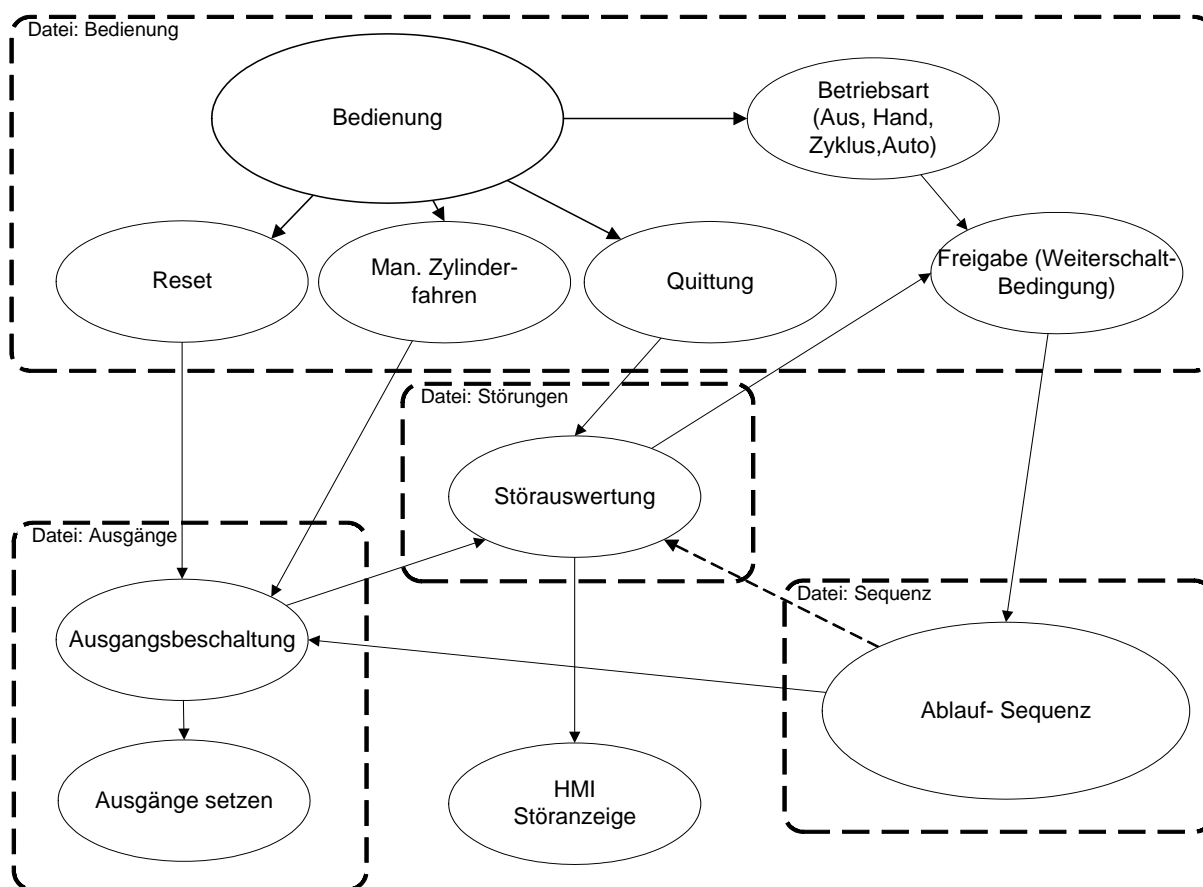


Abbildung 10: Informationsfluss

Aus dem oben abgebildeten Flussdiagramm können speziell folgende Informationen entnommen werden:

- Die Ausgänge werden weder vom HMI noch von der Sequenz direkt gesetzt.
- Die Störauswertung bezieht sich hauptsächlich aus der Ausgangsbeschaltung und nur in speziellen Fällen aus der Ablaufsequenz.
- In der Ablaufsequenz wird auf das Freigabesignal (Weiter) und den Zustand der Anlage (Sensoren) und nicht auf die einzelnen Betriebsarten zugegriffen.

3.3 Strukturierung des Beispielprojektes

Für das Beispielprojekt wird folgende Strukturierung vorgeschlagen:

3.3.1 Hauptprogramm

Im Hauptprogramm (COB0) werden alle allgemeinen Aufgaben behandelt. Diese sind zum Beispiel:

- Defaultwerte laden
- Taktgeber aufbereiten
- Aufstarten der Anlage

3.3.2 Übergabemodul Station 1

Das komplette Übergabemodul wird im COB 1 behandelt. Dazu gehören folgende Funktionen, die in separaten Dateien untergebracht sind:

- Allgemeines mit: Überwachungen, Grundstellung, Aufrufe verschiedener Stationsbausteine
- Bedienung der Station
- Störungen der Station aufschalten
- Ausgangsbeschaltung der Station
- Ablaufsequenz des Übergabemoduls

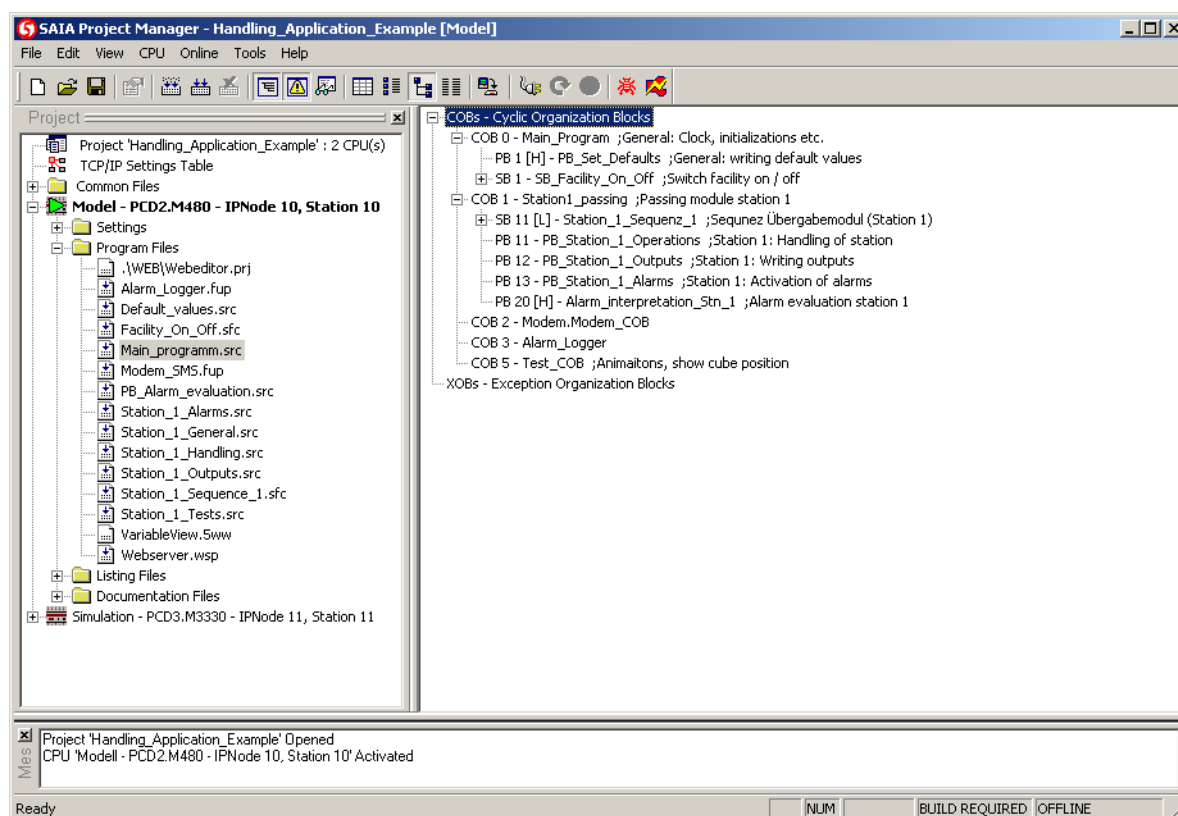


Abbildung 11: PG5 Project Manager mit Programmstruktur

Im COB 5 werden für Testzwecke die Bildnummern für das animierte Modell und die Lage des Würfels aufbereitet.

3.3.3 Wahl der Werkzeuge

Aus der Praxis hat sich gezeigt, dass die Wahl der Werkzeuge von unterschiedlichen Faktoren beeinflusst wird. So wird oftmals vom Kunden eine spezielle Art der Dokumentation bevorzugt, oder auf die Programmierkenntnisse des Servicepersonals Rücksicht genommen. Der Programmierer bringt Vorlieben für das eine oder andere Werkzeug mit, oder er stellt die Effizienz in den Vordergrund.

In unserem Beispielprojekt setzen wir die den Aufgaben am besten entsprechenden und in der Praxis am häufigsten verwendeten Werkzeuge ein.

3.3.4 Codierung: Allgemeines

In dieser Datei werden folgende Aufgaben ausgeführt:

- Verschiedene Initialisierungen der Station
- Verschiedene Überwachungen (Schutztüre, Druckluft etc.)
- Auswertung der Grundstellung der Station
- Auswertung der Freigaben (Rundtisch drehen, Zentrierung des Werkstückträgers etc.)
- Aufruf der Sequenzen
- Aufruf verschiedener Programmblöcke (Bedienung, Ausgänge, Störungen etc.)
- Störauswertung

```

SAIA IL Editor [Model] - [Station_1_General.src*]
File Edit Search View Project Online Tools Window Help

; STATION 1 : Passing module
; -----
COB    Station1_passing           ;Passing module station 1
      0

; Initialisations
; -----
; Protection door
; -----
ACC    H
OUT    Station_1.Protection_Door_OK ;Protection door ok (closed or bypassed witht the key)

; Air pressure
; -----
ACC    H
OUT    Station_1.Air_Pressure_OK    ;Air pressure OK

; Toggle Emergency Off station 1 (simulation)
; -----
STH    Toggle_Emergency_OFF_Stn_1   ;Toggle emergency OFF of station 1
DYN    Dyn_Emergency_OFF_Stn_1      ;Dynamize Emergency OFF of stn. 1
COM    A_260Yd2                     ;Station 1: Button Emergency off is OK

; Init Emergency Off to 0 on startup
$INIT
ACC    H
SET    A_260Yd2                     ;Station 1: Button Emergency off is OK
$ENDINIT

; Emergency Off OK
; -----
STH    F 250Rd6                     ;Emergency OFF not active
  
```

Abbildung 12 SAIA IL Editor (SEdit)



Wichtig: Um nicht für jede Station separate Statusflags für die Zylinderbeschaltung reservieren zu müssen, werden für alle Stationen die gleichen Flags verwendet. Diese Zylinderstatus für 16 Zylinder) werden in einem Register pro Station gespeichert und müssen vor deren Verwendung geladen und nachher wieder gesichert werden. Dieses Vorgehen erlaubt eine Art Multi-Instanz-Programmierung.

3.3.5 Codierung: Ausgangsbeschaltung

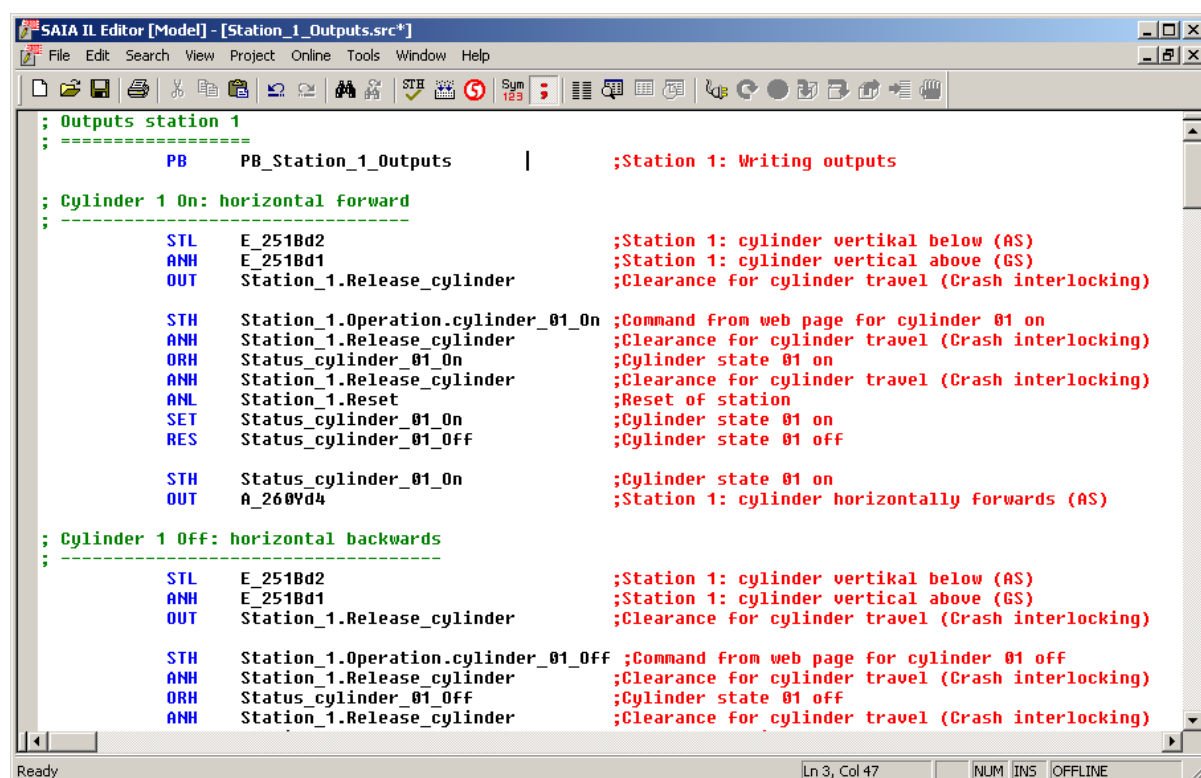


Abbildung 13: SEdit

In der Ausgangsbeschaltung werden möglichst alle der Station zugeordneten Ausgänge ein- oder ausgeschaltet. Wir betrachten exemplarisch den ersten Zylinder unserer Übergabestation.

In unserm Fall wird ein bipolares Ventil eingesetzt.

Das heisst: Das Ventil wird mit je einer Spule in die „Ein“- oder „Aus“-Stellung bewegt. Solche Ventile gibt es in den Ausführungen monostabil (z.B.: 5/2-Wege Ventil) und bistabil (5/3-Wege Ventil mit Mittelstellung). Die Ansteuerung in der Ausgangsbeschaltung ändert nicht. (Bistabile Ventile werden z.B. verwendet, wenn man mit einem Lichtvorhang hardwaremäßig die Bewegung unterbrechen will.)

Bei der Verwendung von unipolaren Ventilen existiert nur eine Spule für die „Ein“-Stellung. Diese wird eingeschaltet (Befehl SET) wenn der „Status_Zylinder_01_Ein“ gesetzt ist und ausgeschaltet (Befehl RES) wenn der „Status_Zylinder_01_Aus“ gesetzt ist.

Für jede manuelle Bewegung, also „vor“ und „zurück“, kann eine separate Crashverriegelung realisiert werden. In unserm Fall wird für beide Bewegungen die Zylinderfreigabe nur gegeben, wenn der vertikale Zylinder oben ist (E251Bd2=low, E251Bd1=high). Diese Crashverriegelung wird auch im Reset-Fall berücksichtigt. Bei der Programmierung von Ablaufsequenzen werden direkt die Statusbits manipuliert, das heisst: das Verhindern von Kollisionen liegt hier in der Verantwortung des Programmierers. Einerseits besteht so zwar die Möglichkeit, in der Ablaufsequenz eine Kollision zu programmieren. Diese ist aber spätestens nach erfolgreicher Inbetriebnahme im Handbetriebsmodus korrigiert. Andererseits lassen

sich so auch spezielle Ansteuerungen wie Drucklosschalten oder beidseitiges Druckbeaufschlagen von Zylindern realisieren.



Vorsicht: Für alle Ausgangsbeschaltungen innerhalb einer Station wird das gleiche Flag für die Zylinderfreigabe verwendet.

Die Bedienung der einzelnen Zylinder (manuelles Verfahren der Zylinder) erfolgt über die Flags „Station_1.Bedienung.Zylinder_01_Ein“ oder „Station_1.Bedienung.Zylinder_01_Aus“. Diese werden in unserm Fall mit der im Web-Editor erstellten Bedienung manipuliert. Die Flags werden solange gesetzt, wie die entsprechenden Tasten gedrückt werden.

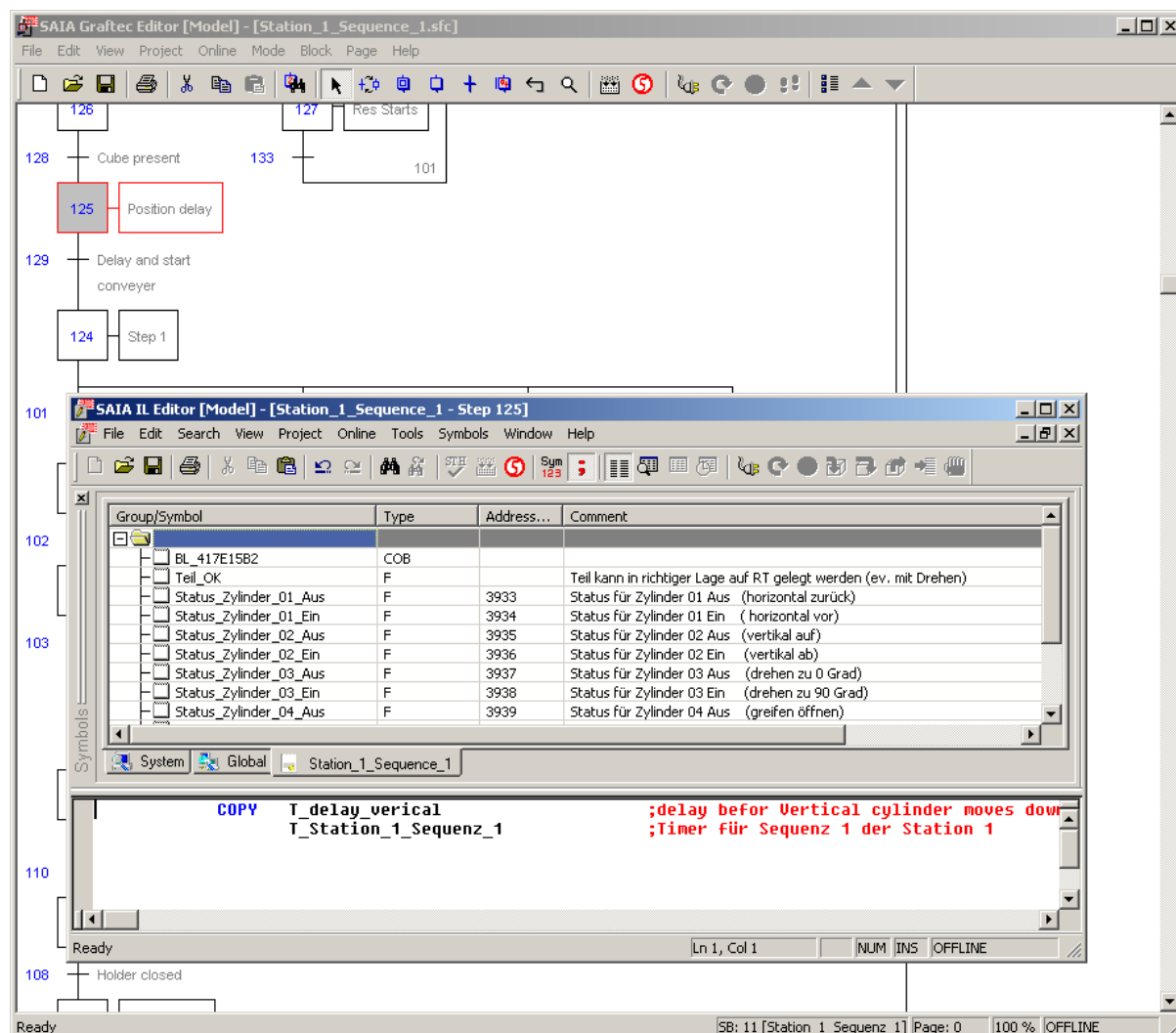


Abbildung 14: Graftec Editor (SGraf) mit Sedit und Symbol Table

Aus der Ablaufsequenz werden die Bewegungen durch das Manipulieren der Flags „Status_Zylinder_01_Ein“ und „Status_Zylinder_01_Aus“ ausgelöst. In unserm Fall sind die globalen Symbole auch lokal noch einmal definiert und mit Informationen zur Bewegung ergänzt worden. Dadurch wird der Code verständlicher und mögliche Flüchtigkeitsfehler fallen schneller auf.

3.3.6 Codierung: Bedienung

Die Bedienung ist vorwiegend über Flags realisiert. Für jede Station ist eine Gruppe mit den gleichnamigen Bedienflags definiert.

Beim Einsatz eines Bedienpanels oder dem Web-Editor werden diese Flags direkt gesetzt, solange die entsprechenden Bedienelemente betätigt werden. Ist die Bedienung mit mechanischen Tasten und Schaltern realisiert, so müssen vor der Auswertung der Bedienung, die Flags anhand der digitalen Eingänge gesetzt und rückgesetzt werden.

3.3.7 Codierung: Sequenz

Die Ablaufsequenzen programmieren wir mit GRAFTEC, einem graphischen Werkzeug zur einfachen Erstellung, Verwaltung und Dokumentation von sequenziellen Abläufen.

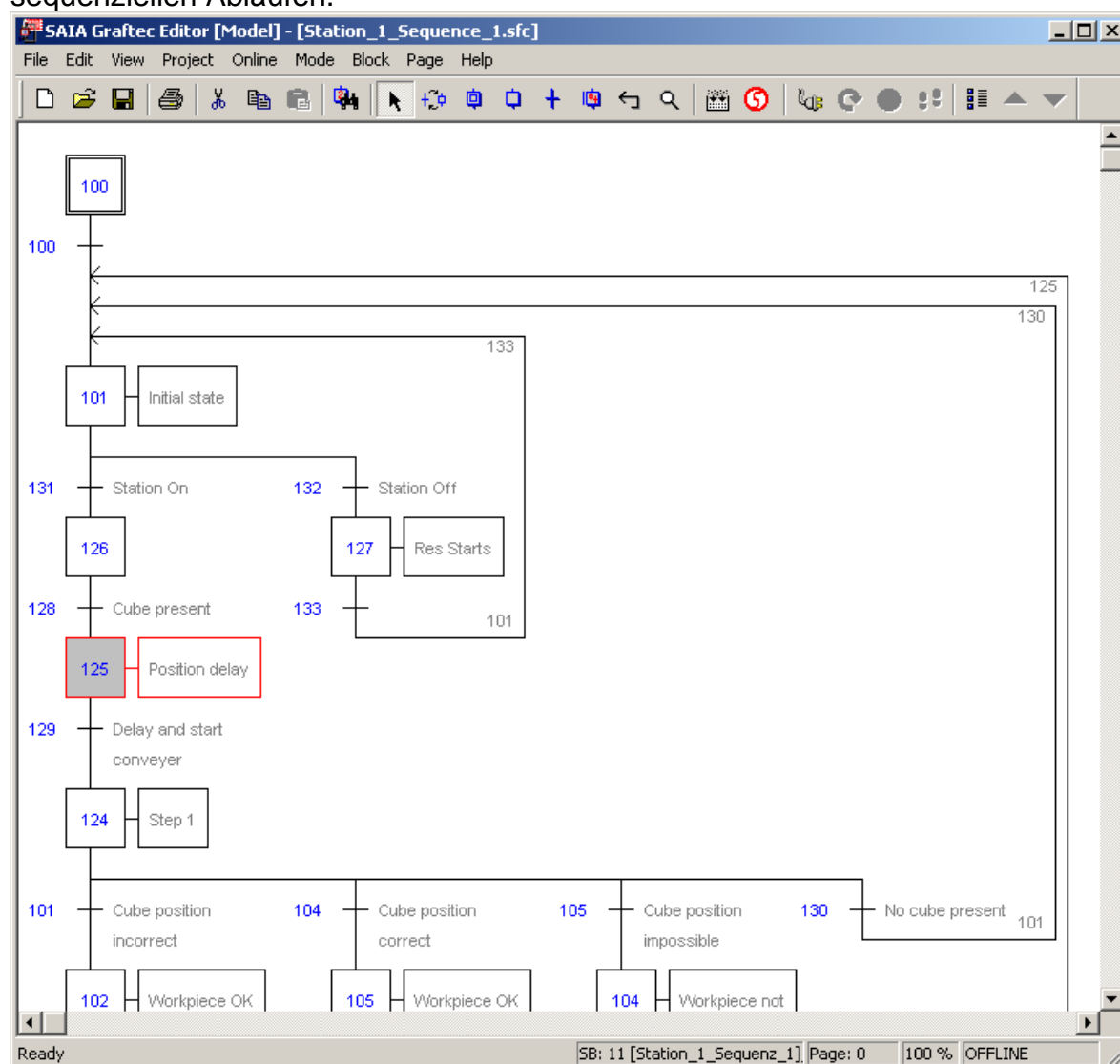


Abbildung 15: Graftec Editor (SGraf)

Die Programmierung von Ablaufsequenzen beginnt mit der grafischen Konstruktion der Ablaufstruktur. Ausgangslage ist die Grundstellung der Station, denn auch ein Reset der Station bleibt solange aktiv, bis die Grundstellung der Station wieder erreicht ist.

Diesen sequenziellen Block (SB) rufen wir nur auf, wenn kein Reset ansteht. Im Reset-Fall wird der SB, für den Neustart der Sequenz, auf seinen Initial-Step gesetzt. Untenstehend der Entsprechende Code von der Datei Station1_General.src:

```
; Reset Graftec-sequence
STH Station_1.Reset ;Reset of station
RSB H SB_Station_1_Sequence_1 ;Station 1: Module for passing forward (sequence 1)
100
CSB L SB_Station_1_Sequence_1 ;Station 1: Module for passing forward (sequence 1)
```

Abbildung 16: Reset und Aufruf der Graftec Sequenz

3.3.8 Codierung: Störauswertung

Bevor die Störungen einer Station ausgewertet werden können, müssen zuerst die einzelnen Störungsbits gesetzt werden. Dies geschieht im Modul „Station_1_Störungen.src“.

```

; Error message 10: Intermediate stop not set
STH    A_261Yd4                ;Station 1: set intermediate stop horizontal (AS)
ANL    E_251Bd8                ;Station 1: intermediate stop set (AS)
ORH    A_261Yd4                ;Station 1: set intermediate stop horizontal (AS)
ANH    E_251Bd7                ;Station 1: intermediate stop open (GS)
OUT    Temp_element_F0+10      ;Temporary element 0 [to 64]; type flag

; Error message 11: Emergency OFF active (-E250Bd6)
STL    E_250Bd6                ;Emergency OFF not active
OUT    Temp_element_F0+11      ;Temporary element 0 [to 64]; type flag
  
```

Abbildung 17: Setzen von aktiven Störungen

Sobald ein Eingangssignal nicht der erwarteten Lage entspricht, wird ein temporäres Störungsbit gesetzt. Dies ist z.B. der Fall, wenn eine Bewegung eines Zylinders gestartet wird, bis dessen Endlage erreicht ist. 32 solcher temporären Flags werden am Ende des Moduls zum Register „aktuelle_Störungen“ zusammengefasst.

Die Störauswertung wird einmal pro Sekunde oder durch die Quittierung der Störungen aufgerufen (in der Datei Station1_General.src).

```

; Alarm check
STH    F Pulse_1_Sek           ;Clock; high for 1 program cycle every second
ORH    Station_1.Acknowledge_Alarm ;Acknowledge alarm
CPB    H Alarm_interpretation_Stn_1 ;Alarm evaluation station 1
  
```

Abbildung 18: Aufruf der Störauswertung

Das Bitmuster im Register „aktuelle_Störungen“ wird regelmässig auf Veränderungen kontrolliert. Bei jeder Veränderung im Bitmuster wird die einstellbare Verzögerungszeit für Störanzeigen neu gestartet. Erst wenn diese Verzögerungszeit abgelaufen und ein Störungsbit gesetzt ist, wird die entsprechende Störung angezeigt und die Station auf den Status „Störung_aktiv“ gesetzt. Anhand des gesetzten Störungsbit im Register „angezeigte_Störungen“ kann nun eine Klartextanzeige erfolgen.

Der Status „Störung_aktiv“ der Station bleibt auch nach Behebung der Störung erhalten, bis diese quittiert wurde, um ein unbeabsichtigtes Weiterlaufen der Sequenz zu verhindern.



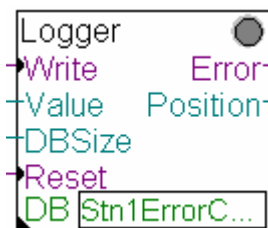
Vorteil: Diese Sequenz-unabhängige Störauswertung ist allgemein gültig und führt in jedem Fall, auch im manuellen Zylinderfahren, zu einer korrekten Störanzeige.

3.3.9 Codierung: Störmeldungen aufzeichnen

Jedes Mal, wenn eine Störung auftritt, wird durch das Programmmodul **Alarm_Logger.fup** einen entsprechenden Eintrag in einem Alarmbuffer gemacht. Dies geschieht mittels speziell für dieses Projekt erstellte FBoxen mit dem Namen Logger.

Diese nicht ganz triviale Aufgabe wird durch diese FBoxen sehr komfortabel realisiert:

- Eine Ansteigende Flanke an dem Input „Write“ bewirkt, dass der an dem Eingang „Value“ anstehende Wert (z.B. der Fehler Code) in den im Feld „DB“ spezifizierte Datenblock eingetragen wird.
- Der an dem Eingang „DBSize“ anlegende Wert bestimmt, wieviele Werte maximal in den Datenblock (DB) geschrieben werden sollen.
- Eine ansteigende Flanke am Eingang Reset setzt den Buffer zurück.
- Die Position des zuletzt eingetragenen Wertes wird an dem Ausgang Position ausgegeben.



Die von diesem Fupla Programm beschriebenen DBs (je einer für den Fehlercode, die Zeit und das Datum, jeweils gleichzeitig beschreiben) werden in dem Web Interface der Steuerung ausgelesen.



Die oben beschriebene FBox ist keine standardmässig installierte FBox! Die entsprechende Bibliothek muss zuerst installiert werden. Bitte führen Sie hierzu das in dem Ordner „Common Files“ in dem PG5 Project manager liegende Installationsprogramm „FBox_Installer.exe“ aus.

3.3.10 Codierung: Störmeldungen per SMS absetzen

Jedes Mal, wenn eine Störung auftritt, kann mit diesem Programmmodul ein SMS auf ein Handy abgesetzt werden.

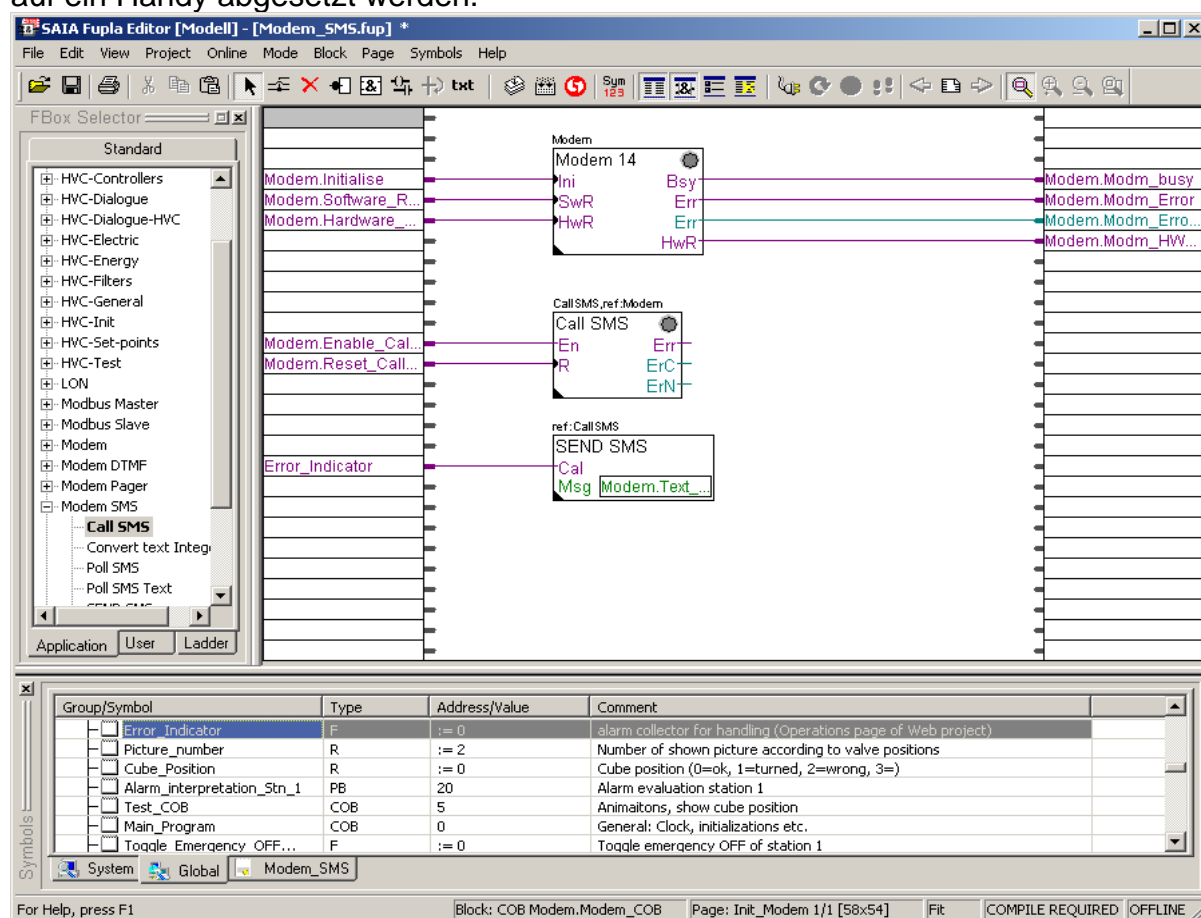


Abbildung 19: Screenshot von Fupla Programm Modem_SMS.fup

Voraussetzung für die Funktionalität dieses Programmteils sind die folgenden:

- Ein analoges Modem PCD2.T813, verbunden mit dem TTL Stecker (Port 1) über Steckplatz 2 der PCD2.M480
- Eine analoge Telefonleitung
- Die Nummer des SMS Service Centers muss korrekt eingetragen sein in der FBox Call SMS
- Die Natelnummer des Empfängers muss korrekt eingetragen sein in der FBox Call SMS
- Der Button „Enable SMS“ in dem Web Interface (Seite Settings) muss aktiviert worden sein (Wird bei jedem Neustart wieder deaktiviert).

Weitere Informationen bezüglich dem Absetzen und Empfangen von SMS Meldungen sowie Onlineverbindungen über ein Modem finden sie in dem Beispielprojekt „Sending and receiving SMS“ auf der Supportseite unter: <http://www.sbc-support.ch/GettingStarted/examples/examples.htm>.

3.4 Bedienung mit Web-Editor

In der verwendeten PCD2.M480 Steuerung ist ein Webserver integriert. Auf dessen Basis kann einfach und kostengünstig eine einfache Bedienung oder Konfigurationsoberfläche realisiert werden.

Im PG5 Programmierwerkzeug ist (optional) ein Web Editor integriert, der speziell die Anbindung an das SPS-Programm über die Symboltabellen erleichtert. Die Funktionalitäten des Web-Editors sind nicht Bestandteil dieser Dokumentation und entnehmen Sie bitte aus den entsprechenden Handbüchern.

Mit dem WEB-Builder werden die im Web-Editor erstellten HTML-Seiten und sonstige Dateien in SPS-kompatible Datenblöcke zusammengefasst.

Bitte beziehen Sie sich für die Anleitung für die Bedienung dieses Projekts auf das beiliegende Dokument „Handling_Application_Installation“. Dieses Dokument beschreibt neben dem Vorgehen für die Installation sowie die Inbetriebnahme dieses Projekts auch die Bedienung der Web Pages auf den Stationen.

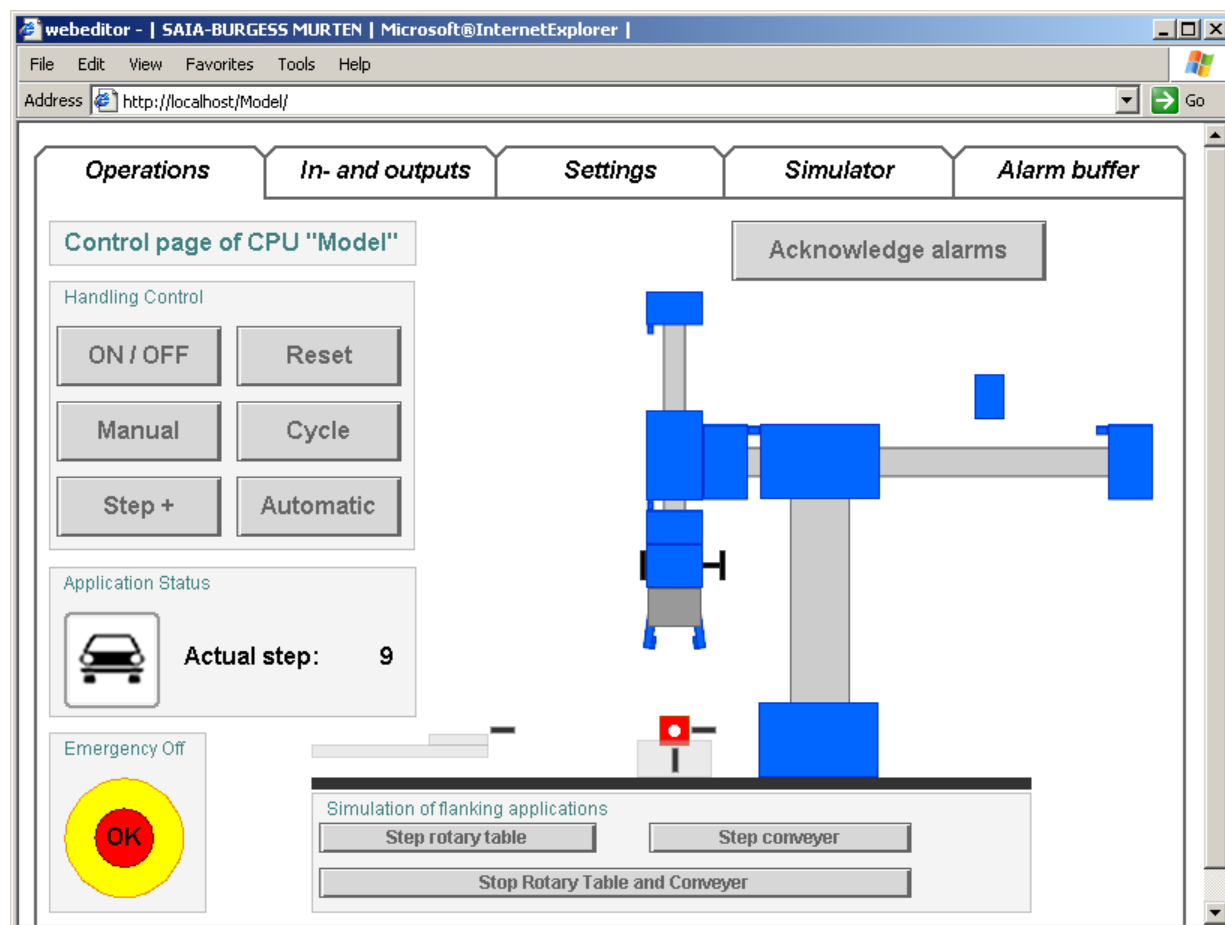
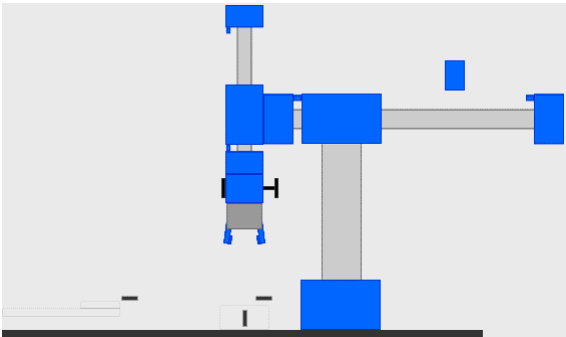
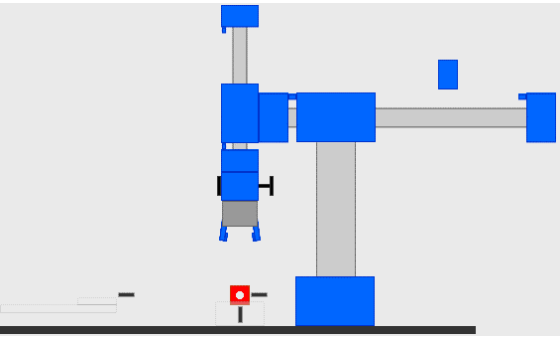
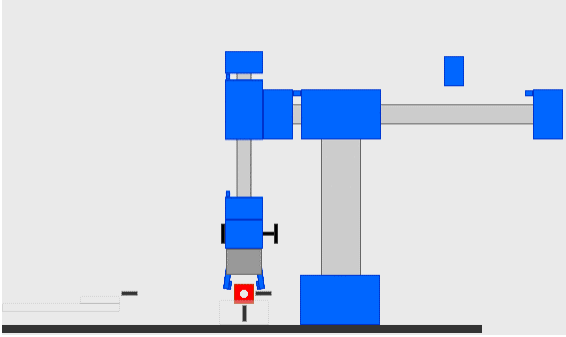
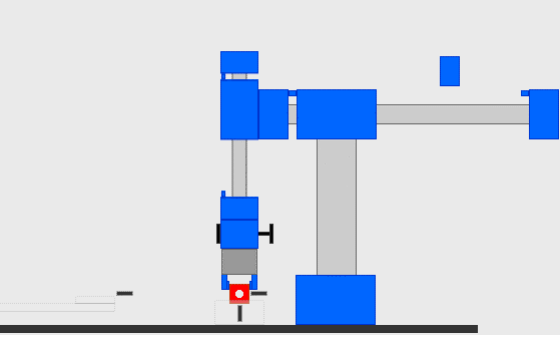
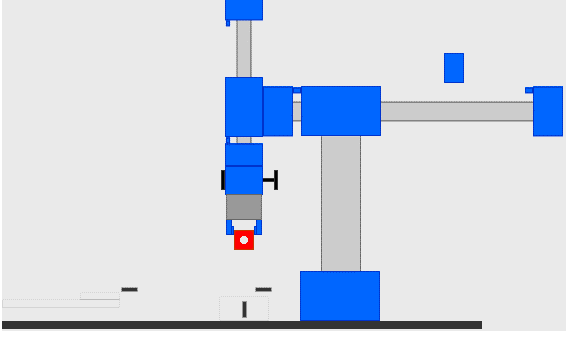
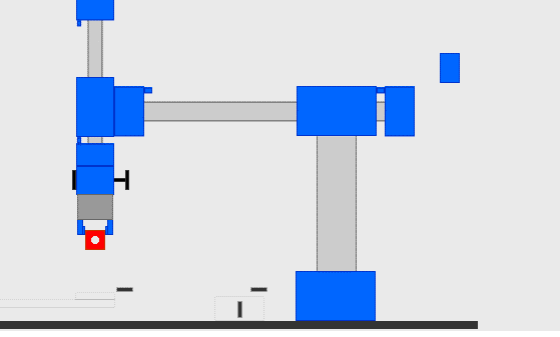
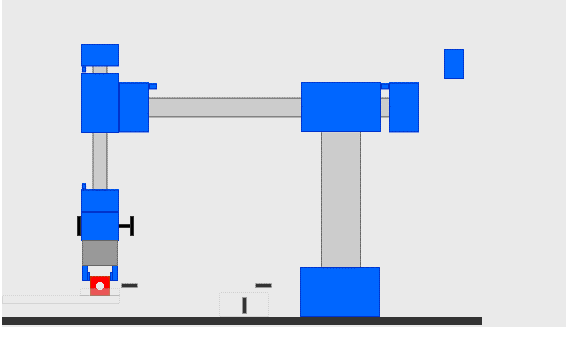
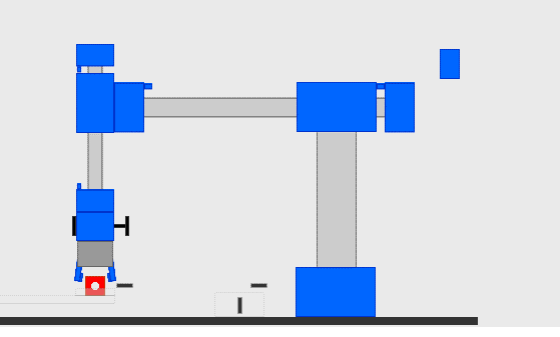


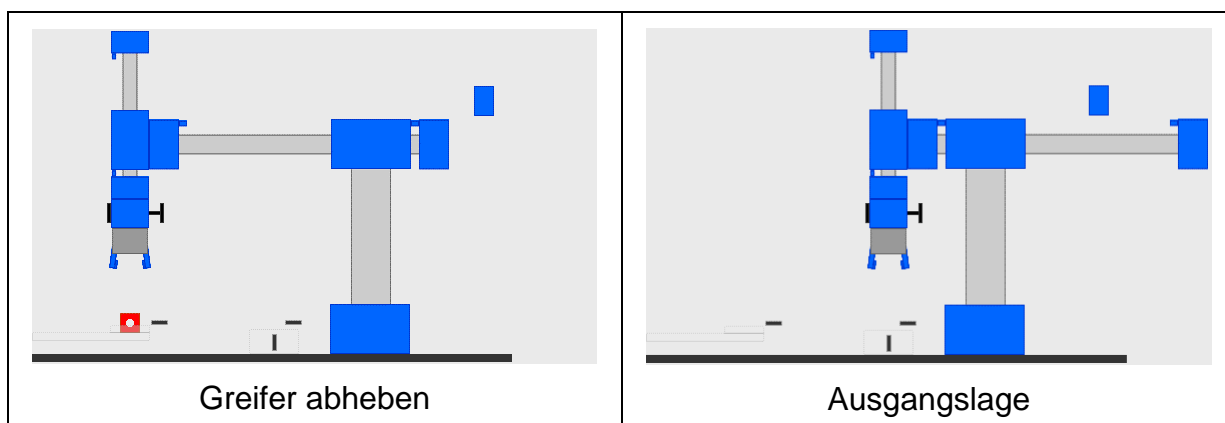
Abbildung 20: Hauptseite der webbasierten Bedienung

3.5 Bewegungsabläufe

In den folgenden Kapiteln wird der Bewegungsablauf der Stationen in Bildform dargestellt. Abhängig von der Lage des Würfels variiert der Bewegungsablauf der Übergabestation.

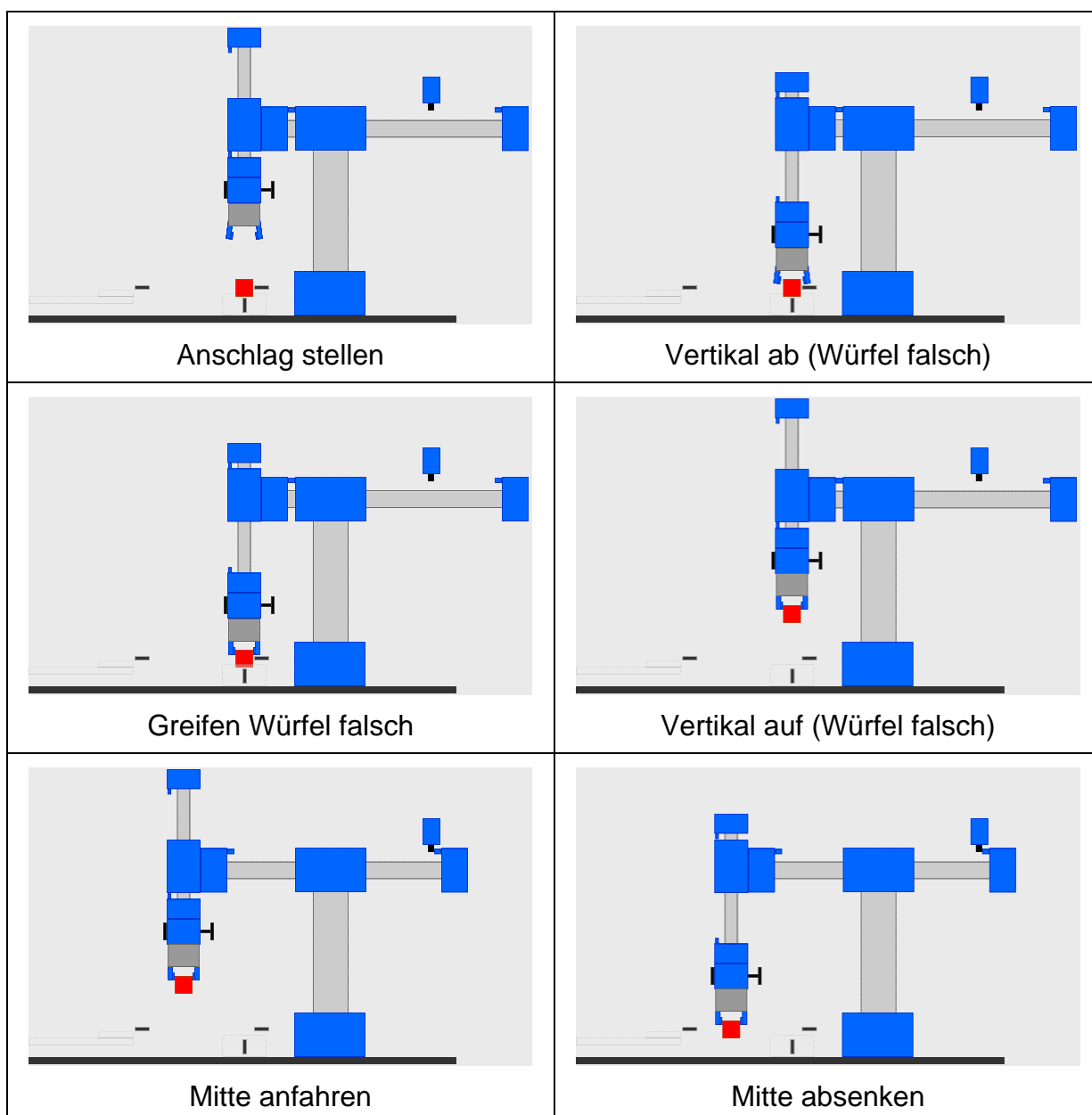
3.5.1 Würfel in richtiger Lage

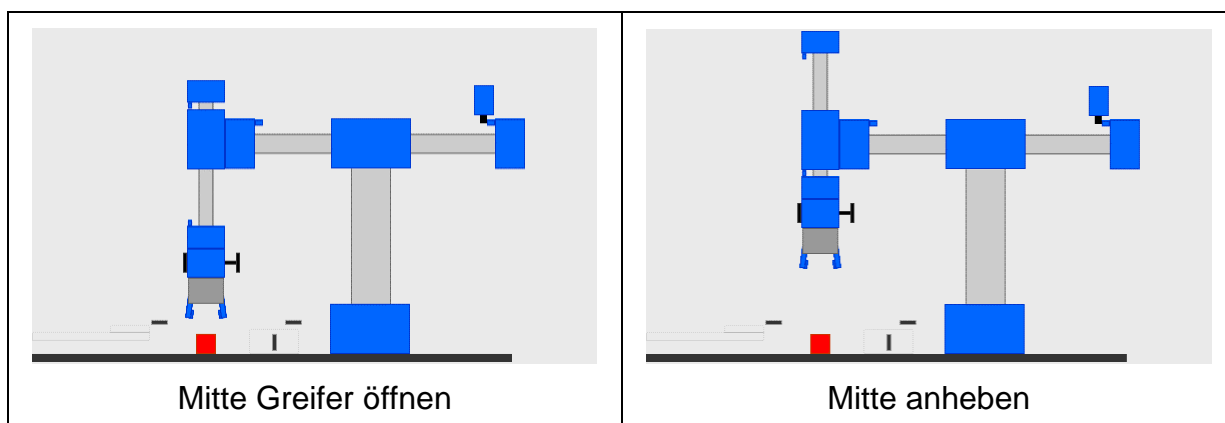
 <p>Grundstellung</p>	 <p>Grundstellung mit Würfel (Lage OK)</p>
 <p>Vertikal ab</p>	 <p>Würfel greifen</p>
 <p>Würfel heben</p>	 <p>Horizontal vor</p>
 <p>Würfel auf Rundtisch</p>	 <p>Greifer öffnen</p>



3.5.2 Würfel in falscher Lage

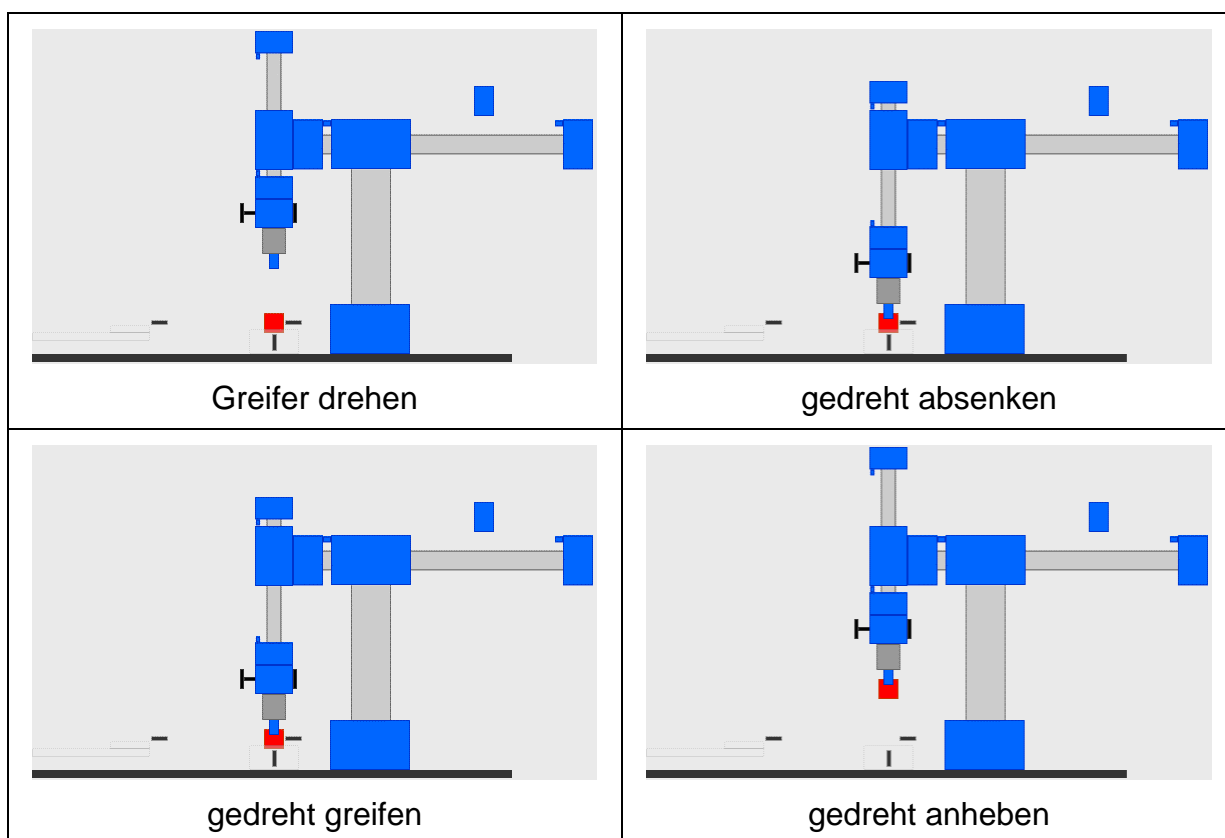
Der Würfel kann nicht in die richtige Lage gebracht werden. Er wird entnommen und in der Ausschussposition abgeworfen.





3.5.3 Würfel in gedrehter Lage

Der Würfel muss mit gedrehtem Greifer gefasst werden, damit er lagerichtig auf den Rundtisch gelegt werden kann.



Ab dieser Position entspricht der weitere Ablauf der Sequenz mit richtiger Würfellage ab Bild „Würfel heben“.