Max–Planck–Institut
Dynamik komplexer technischer Systeme
Magdeburg

Studienarbeit

Automatisierung der Versuchsanlage „Membranreaktoren“

Klaus Habermann
geb. am 27. Mai 1982
in Magdeburg
Matr.–Nr. 163 231

verantwortlicher Hochschullehrer

Prof. Dr.–Ing. Achim Kienle

Betreuer

Dr. rer. nat. Frank Klose
Dipl.–Ing. Reiner Könning
Dipl.–Ing. Ákos Tóta

Inhaltsverzeichnis

1 Problemstellung **2**

2 Problemlösung **4**
2.1 Grundgedanke 4
2.2 Konzept und Design 4
2.2.1 Das Auswahlfeld für die MTD 5
2.2.2 Die Ausgabefenster der Sollwerte 8
2.2.3 Die Kontrollbuttons 9
2.2.4 Die Position-and-Run-Funktion 12
2.2.5 Die Verzögerung des Einlesens nach dem Startimpuls des Gaschromatographen 13
2.2.6 Visualisierung des Startsignals des GC 14
2.2.7 Die Einbettung des Bedienbildes für die Automatikfunktion 14
2.3 CFC-Programmierung 16
2.3.1 Verlängerung des Start-Impulses für Visualisierung . . 16
2.3.2 Der 320-Sekunden-Timer 17
2.3.3 Ein RS-Flip-Flop als Trigger 20
2.3.4 Die richtige Ablaufreihenfolge 23
2.3.5 Der AS-OS-Transfer 23
2.4 Das Ablaufscript der Automatik 24
2.4.1 Die Matrix-Text-Datei 24
2.4.2 Die *.pas Datei 25
2.4.3 Der Algorithmus 26
2.4.4 Der Trigger des Automatikscriptes 35

3 Zusammenfassung **36**

4 Schlussbemerkungen **37**

1 Problemstellung

Am Max-Planck-Institut für Dynamik komplexer technischer Systeme in Magdeburg sind zur Untersuchung heterogen katalysierter Reaktionen Versuchsanlagen in Betrieb.
Wesentlich ist hierbei die Nutzung der Gaschromatographie mit Massen–spektrometer-Kopplung (GC-MSD = Gaschromatograph-Massenselektiver Detektor) zur Online-Prozesscharakterisierung, die durch Prozessleittechnikkomponenten der Firma Siemens unterstützt wird.
Die Versuchsanlage kann mit drei verschiedenen Gasarten betrieben werden (standardmäßig Stickstoff, Luft und Kohlenwasserstoff). Hinzu kommt Helium (mindestens 99,999 %) als Betriebsgas für den GC-MSD. Dieser sendet Signale an das Prozessleitsystem (PLS). Das PLS steuert neben den elektronischen Mass Flow Controllern (MFCs), die für die Dosierung der Reaktionsmedien zuständig sind, die gesamte Versuchsanlage. Sie ist mit einer Siemens S7-400 Speicherprogrammierbaren Steuerung ausgerüstet. Die Software PCS7 (Process Control System 7) von Siemens ermöglicht dabei die Kommunikation zwischen den Komponenten Software (WinCC, Step7 mit CFC) und der Steuerungshardware S7-400. Ziel ist, einen eventgesteuerten Betrieb der Versuchsanlage zu ermöglichen, bei der die Ablaufsteuersignale vom GC-MSD kommen.
Durch diese Kombination von Prozessleitsystem und der eigentlichen Versuchsanlage ist es möglich, Messungen durchzuführen, die bestimmten Schemata von Parameterzusammenstellungen (Sollwerte für Temperatur, Durchflussgeschwindigkeiten, Gaszusammensetzungen) folgen.
Die Versuchsanlage besteht aus drei Membranreaktoren, die nacheinander vom Reaktionsgemisch durchströmt werden. Diese Reaktoren sind jeweils von zwei Heizmanschetten ummantelt, deren Temperatur über das PLS gesteuert werden kann. Hinzu kommt ein Kinetikreaktor mit Katalysatorsegment und ein Reaktor zur katalytischen Nachverbrennung der analysierten Reaktionsgemische. Proben der Reaktionsgemische werden über Probenahmekanülen zu einem 8-Port-Multipositionsventil geführt, bei dem über die Software des GC-MSD jeweils ein Kanal ausgewählt werden kann, von dem dann eine Probe analysiert wird. Die Verbindung zwischen dieser Ventil-Box und dem GC-MSD erfolgt über eine bis 300 °C beheizbare Transferleitung. So kann eine Auskondensation von Probenbestandteilen verhindert werden.
Der GC-MSD verfügt zur gaschromatischen Analyse des Produktstromes über eine Gasprobenschleife. Der Probenstrom wird über ein Mass Flow Meter am Auslass dieser Probenschleife überwacht.
Die gewonnenen Daten werden dabei kontinuierlich vom PLS aufgezeichnet und auf Netzlaufwerken sowie auch lokal am Bedienrechner archiviert.

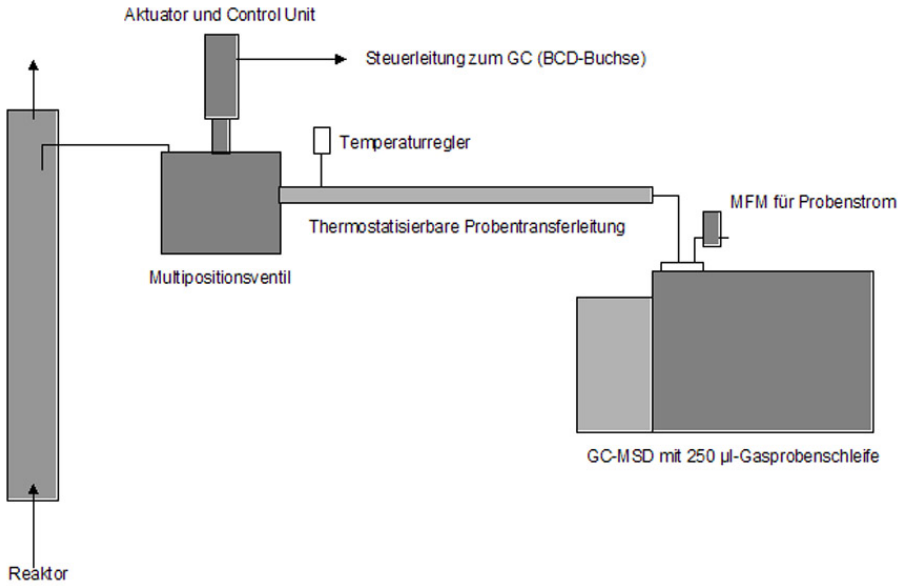


Abbildung 1: Aufbauschema der Versuchsanlagenhardware

Mit Hilfe dieser Versuchsanlage werden folgende Messprinzipien realisiert.

1. Aktivitätstests (konstante Volumenströme und Konzentrationen - Temperatur wird während der Messung erhöht)

2. Kinetikmessungen (Konzentration eines Reaktanden konstant, Temperatur und Volumenstrom konstant - Konzentration des anderen Reaktanden wird während der Messung verändert)

Zudem spielt beim Betrieb der Anlage der Aspekt **Sicherheit** die größte Rolle. Wichtig sind dabei die Einhaltung von Explosionsgrenzen und die Einhaltung von Maximaltemperaturen, um Materialschäden und damit verbundene Gefahren durch den Austritt von gefährlichen Gasen zu unterbinden.

Die Zielstellung dieser Studienarbeit war, es zu ermöglichen, dass solche Messungen **überwachungsfrei** und **vollautomatisch** durchlaufen werden können.

2 Problemlösung

2.1 Grundgedanke

Zur Realisierung der gestellten Aufgabe musste ein Konzept entwickelt werden, das sicherstellt, dass die Automatik erkennen kann, wann genau ein Messzyklus vorbei ist bzw. wann er beginnt.
Ein Messzyklus dauert ca. 13 Minuten.
Hierzu wird ein Signal genutzt, das der Gaschromatograph ausgibt, wenn ein neuer Messzyklus beginnt. Dieses Signal ist sehr kurz und musste mit Hilfe von Impulsverlängerern auf der Hardwareseite verlängert werden, damit sich jedes mal ein stabiles Spannungspotential für eine sichere Auswertung aufbauen kann. Dieses Signal dient als Anstoßmechanismus bzw. als sogenannter Trigger für die Automatik.
Hauptgedanke bei der Entwicklung dieser Automatik war, komplett alle Parametersätze als Zeilen in eine große Matrix zu schreiben, bei der jede Spalte einer solchen Zeile einen bestimmten Sollwert enthält. Die erste Spalte beinhaltet eine laufende Nummer. Die erste Zeile beginnt mit der Nummer 1. Die letzte Zeile beginnt mit der Nummer 0. Welchem Zweck diese Anordnung dient, wird in einem der späteren Abschnitte erklärt.
Diese im Vorfeld einer Messung erstellte Matrix wird in Form einer Textdatei der Form *.csv[1] gespeichert.
Um auf diese Datei zugreifen zu können, bedarf es einer Schnittstelle zwischen Benutzer und dem eigentlichen Geschehen in der Steuerung und im WinCC. Der wesentlichste Bestandteil einer solchen Schnittstelle ist eine passende Bentzeroberfläche.

2.2 Konzept und Design

Zunächst war festzulegen, welche Funktionen solch eine Benutzeroberfläche haben soll. Es zeigte sich als sinnvoll, ein Eingabefeld zu implementieren, wodurch die im Vorfeld der Messung erstellte Matrix-Text-Datei (MTD; Parameterdatei) ausgewählt werden kann. Zur besseren Übersicht über den laufenden Prozess sollten gerade eingelesenen Sollwerte in entsprechenden Ausgabefenstern nach Typ sortiert angezeigt werden.

[1] *.csv Dateien sind Textdateien, die bestimmte Werte durch Kommata voneinander getrennt enthalten und von WinCC besonders gut ausgewertet werden können. Zudem wird bei einer Konvertierung von Exceltabellen in das csv-Format die Anordnung der einzelnen Zellen berücksichtigt und übernommen.

Wichtig war bei der Erstellung der Oberfläche zudem auch der Komfort, den Ablauf der Automatik insofern zu beeinflussen, dass gezielt bestimmte Zeilen bearbeitet werden können.
Letztendlich sollte die Oberfläche zudem noch elementare Funktionsbausteine wie einen Start- und einen Stopbutton für die Automatik sowie die Visualisierung des Triggersignals als Kontrollfunktion enthalten.
Zur Erstellung der Bedienoberfläche wurde der WinCC Graphics Designer, der Bestandteil des PCS7 Pakets ist, verwendet.
In seiner fertigen Gesamtansicht kann man die geordnete und übersichtliche Anordnung der Bedienelemente und Ablesefelder des Automatikbedienbildes erkennen.

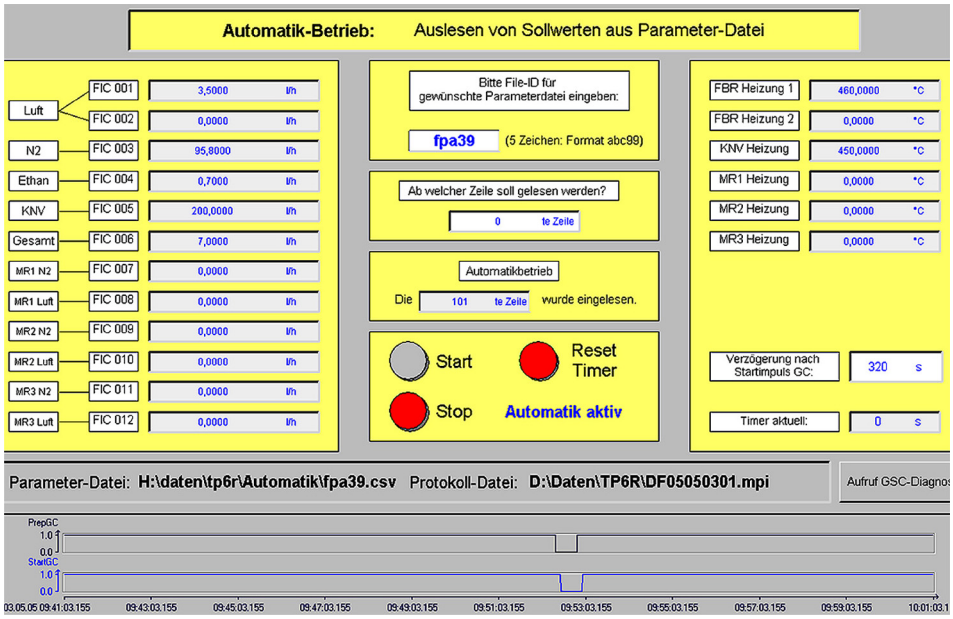


Abbildung 2: Gesamtansicht des Automatikbedienbildes

In den folgenden Kapiteln wird Schritt für Schritt die Entwicklung dieses Bedienbildes beschrieben.

2.2.1 Das Auswahlfeld für die MTD

Das grafische Element zur Auswahl der MTD wurde mit Hilfe von farbig gefüllten Rechtecken erstellt, die Text enthalten. Das eigentliche Eingabefeld, das zugleich auch ein Ausgabefeld ist - ein EA-Feld - wurde mit einem 8 Zeichen langen Ausgabewert versehen und bekam eine Dynamik, d.h. eine

programmierbare Aktion zugewiesen, die bei Änderung des Eingabewertes und durch Drücken der Eingabetaste ausgeführt wird.
Die eingegebende Zeichenfolge, d.h. der Name der MTD, wird dabei auf die im Vorfeld im WinCC erstellte Variable `File-ID` gelegt.

```
strcpy(Nummer, GetTagChar("File-ID"));
strcpy(Dateiname1, "H:\\daten\\tp6r\\Automatik\\");
strcat(Dateiname1, Nummer);
strcat(Dateiname1, ".csv");
SetTagChar("Datafile_1", Dateiname1);

strcpy(Nummer, GetTagChar("File-ID"));
strcpy(Dateiname2, "D:\\Daten\\tp6r\\Automatik\\");
strcat(Dateiname2, Nummer);
strcat(Dateiname2, ".csv");
SetTagChar("Datafile_2", Dateiname2);
```

Abbildung 3: Quelltext der Aktion, die mit der Auswahl der MTD verknüpft ist

Zudem werden durch Drücken der Enter-Taste zwei Variablen im WinCC (`Datafile_1` und `Datafile_2`) beschrieben, die die exakte Position (den Pfad) und den Dateinamen (`File-ID`) im *.csv-Format für die MTD auf dem Netzlaufwerk (`Datafile_1`) und auf dem lokalen Datenträger (`Datafile_2`) enthalten.

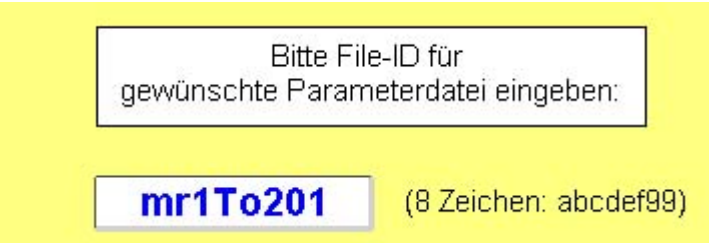


Abbildung 4: Auswahl der Matrix-Text-Datei

Mit dieser Aktion wird erreicht, dass im laufenden Betrieb der Automatik bekannt ist, wo die MTD zum Auslesen der Parameter abgelegt ist.
Dieser Pfad wird während des Automatikbetriebes angezeigt.

Im Falle von Netzausfällen wird die MTD im Vorfeld in den Pfad kopiert, den die Variable `Datafile_2` enthält.
Auf diese Redundanzfunktion wird auf Seite 29 intensiver eingegangen.

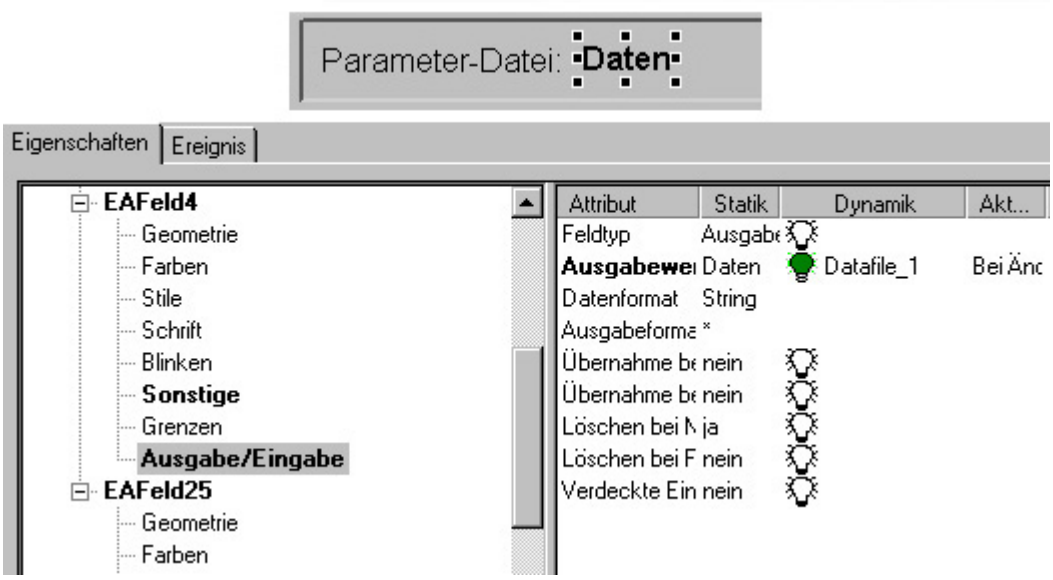


Abbildung 5: Verknüpfung der Variablen `Datafile_1` mit der Pfadanzeige

Zur Übersichtlichkeit während des Messbetriebes wurde ein EA-Feld zur Anzeige der gerade aus der MTD eingelesenen Zeile in das Automatik-Bedienbild eingebettet.

Abbildung 6: Anzeige der gerade aus der MTD eingelesenen Zeile

2.2.2 Die Ausgabefenster der Sollwerte

Das grafische Element zur Anzeige der ausgelesenen Sollwerte aus der MTD besteht jeweils aus einem Ausgabefenster, (einem EA-Feld mit Ausgabefunktion) das bei Änderung des Anzeigewertes aktualisiert wird.
Der Anzeigewert ist jeweils an eine WinCC-externe [2] Variable gekoppelt, die den Sollwert enthält.

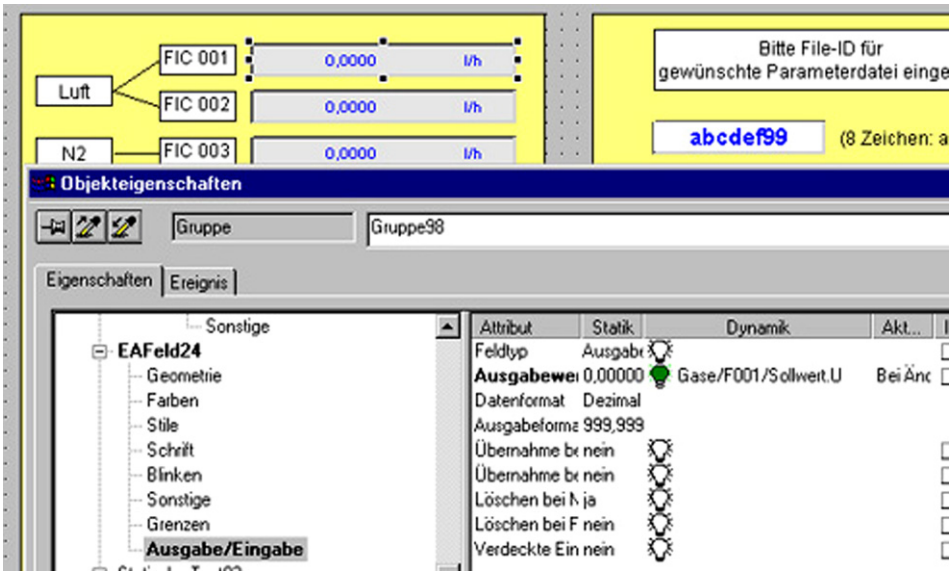


Abbildung 7: Verknüpfung einer Variable mit einem Anzeigeobjekt

Bei dem Flow Indication Controller 001 (FIC 001), der für die Einstellung der Luftströme unter 9 Liter pro Stunde eingesetzt wird, lautet der Pfad zu der Variable, die den Sollwert für diesen FIC enthält: `Gase/F001/Sollwert.U`. Analog dazu sind alle anderen Sollwertausgabefenster aufgebaut, die auf dem Bedienbild für die Automatikfunktion implementiert sind. Zur Identifikation der Flüsse als solche wird zudem bei jedem Ausgabefenster auch die Einheit [l/h] angezeigt - bei den Temperatursollwerten für die Heizungen [°C].

[2] WinCC-externe Variablen werden durch den AS-OS-Transfer erstellt. Sie repräsentieren die einzelnen Ein- und Ausgänge der für die CFC-Programmierung verwendeten Bausteine. Somit wird eine Kommunikation zwischen WinCC-Operationen und Aktionen im CFC-Plan gewährleistet.

2.2.3 Die Kontrollbuttons

Der Button zum Starten der Automatikfunktion besteht aus einem Kreis, der je nach Betriebszustand seine Füllfarbe ändert und somit signalisiert, ob die Automatikfunktion in Betrieb ist oder nicht.

Abbildung 8: Der Start-Button

Da der Nutzer normalerweise in der Computerwelt nichts anklicken kann, was grau hinterlegt ist, ändert der Start-Button bei laufendem Betrieb der Automatikfuntion seine sonst rote Farbe in grau. Zur Abfrage des Betriebszustandes wurde der Button mit der Variable `Analytik/GC/FreiGCImp.Q0` verknüpft. Bei dieser Verknüpfung handelt es sich um eine Abfrage des Ausgangssignals dieses `FreiGCImp` Bausteins in der Steuerung. Bei einem `High` Signal wird die Farbe des Buttons auf grau geändert - bei `Low` auf rot.
Der Vollständigkeit halber wurde zudem noch eine Bedienfreigabe für diesen Button implementiert, die es bei laufendem Betrieb der Automatikfunktion nicht zulässt, dass sich der Start-Button bedienen lässt.

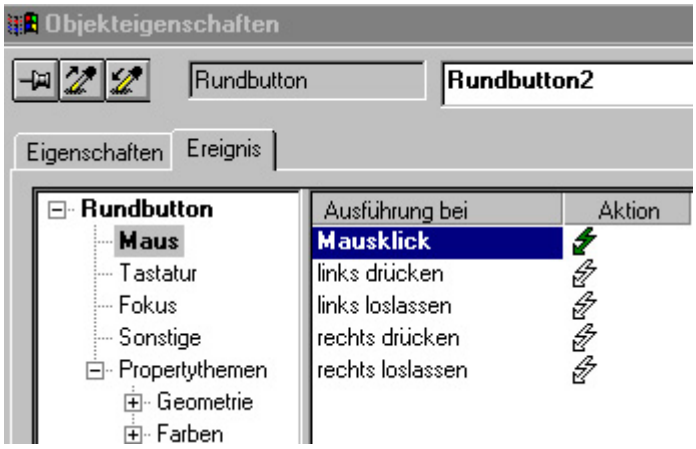


Abbildung 9: Einstellung der Mausklick-Dynamik

Bei einem Mausklick auf den Start-Button wird die Variable `FreiGCImp.I0` auf 1 (`High`) gesetzt.

```
#define TAG0 "Analytik/GC/FreigGCImp.I0"

// next TagID : 1
// WINCC:TAGNAME_SECTION_END

// WINCC:PICNAME_SECTION_START
// syntax: #define PicNameInAction "PictureNa
// next PicID : 1
// WINCC:PICNAME_SECTION_END

SetTagBit(TAG0,1);
```

Abbildung 10: Quelltext für die Mausklick-Dynamik des Start-Buttons

Parallel wird die MTD vom Netzlaufwerk auf das lokal vorhandene Laufwerk kopiert, um ein späteres Auslösen der Redundanzfunktion mit der auf die aktuell laufende Messung angepasste MTD vorzubereiten.
Der Stop-Button ist analog zum Start-Button aufgebaut, jedoch verändert er seine Farbe nicht und kann jederzeit bedient werden. Bei einem Mausklick auf

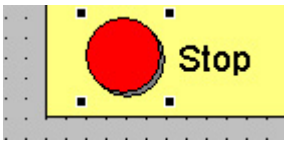


Abbildung 11: Der Stop-Button

den Stop-Button wird der Eingang des CFC-Bausteins `Analytik/GC/FreiGCImp` auf 0 (`Low`) gesetzt und die im WinCC erstellte Variable `ZahlCC` wird zurückgesetzt, bzw. deren Wert wird auf Null geändert. Die Bedeutung dieser Variablen `ZahlCC` wird auf Seite 29 näher erklärt.

```
#define Tag1 "Analytik/GC/FreigGCImp.I0"
#define Tag2 "ZahlCC"
// next TagID : 1
// WINCC:TAGNAME_SECTION_END

// WINCC:PICNAME_SECTION_START
// syntax: #define PicNameInAction "PictureNa
// next PicID : 1
// WINCC:PICNAME_SECTION_END

SetTagBit(Tag1,0);        //Rückgabe-Typ :BO(
SetTagDouble(Tag2,0); //Rückgabe-Typ :BO(
```

Abbildung 12: Die Mausklick-Dynamik des Stop-Buttons

Der Reset Timer-Button (ebenfalls analog zu den anderen beiden Buttons aufgebaut) besitzt die Funktion, die Verzögerung des Triggersignals nach dem Signal des Gaschromatographen zum Start einer neuen Messungssequenz (dem Startimpuls) zurückzusetzen, damit der Timer erneut von 320 Sekunden beginnt, im Sekundentakt abwärts zu zählen. Auf die Bedeutung des Timers wird auf Seite 17 genauer eingegangen.

```
#define Tag0 "Analytik/GC/ResetTimerP.I0"
```

```
SetTagBit(Tag0,1);
```

Abbildung 13: Der Reset Timer-Button und seine Mausklickdynamik

Zur visuellen Unterstützung der Information über den Betriebszustand der Automatikfunktion wurde direkt in unmittelbarer Nähe der Kontrollbuttons ein Textfeld ohne Rahmen implementiert. Es zeigt den Betriebszustand „nicht aktiv“ durch einen statischen Text und den Zustand „aktiv“ durch blinkenden (sich periodisch in der Farbe ändernden) Text an.

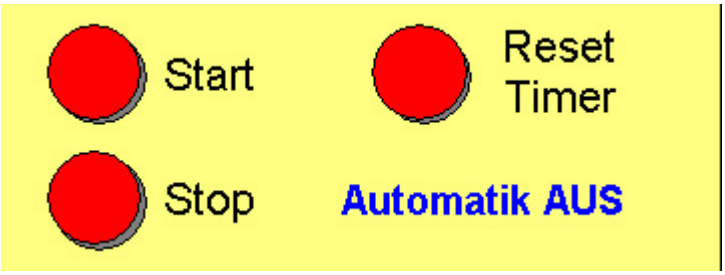


Abbildung 14: Gesamtansicht der Kontrollbuttonanordnung mit Betriebszustandanzeige

2.2.4 Die Position-and-Run-Funktion

Während einer vorgegebenen Messung kann es vorkommen, dass der Bediener plötzlich aus bestimmten Gründen andere Sollwerte für die nachfolgenden Messsequenzen einstellen möchte. Dazu wurde ein Zugang zu einer sogenannten Position-and-Run-Funktion implementiert. Es kann jedoch auch passieren, dass eine MTD beim Entwurf Fehler (z.B. verschobene Dezimalstellen bei den Parametersollwerten) enthält, die erst im laufenden Betrieb der Automatikfunktion erkannt werden. In solch einem Fall trägt die Position-and-Run Funktion wesentlich zur Problemlösung bei. Das heißt genauer, dass man als Nutzer die MTD während des Betriebes verändern kann. Man kann die Elemente der schon eingelesenen Zeile, die die „falschen“ Parameter enthält, korrigieren und das Auslesen an dieser Zeile nochmals ansetzen lassen.

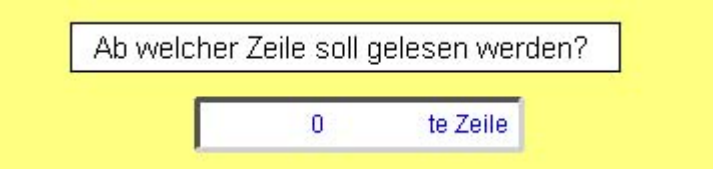


Abbildung 15: Position-and-Run Interface

Das grafische Element dieser Zusatzfunktion besteht neben Textfeldern mit und ohne Rahmen aus einem EA-Feld, das eine Eingabe dieser Zeile ermöglicht. Direkt nach Beendigung der gerade aktuell laufenden Sequenz der Messung springt der Zähler der Automatikfunktion zur eingegebenen Zeile der MTD und setzt den Betrieb an genau dieser Zeile fort.

2.2.5 Die Verzögerung des Einlesens nach dem Startimpuls des Gaschromatographen

Für den Gaschromatographen (GC) der TP6R-Versuchsanlage wurde festgelegt, dass die Verzögerung zwischen dem Startsignal, das von dem Gaschromatographen kommt und der eigentlichen Einstellung der Sollwerte für Flüsse und Temperaturen während dieser neu angelaufenen Messungssequenz 320 Sekunden betragen soll. Erst nach dieser Zeit schaltet ein Probennahmeventil innerhalb des Geräteaufbaus des Gaschromatographen. Während dieser Zeit von 320 Sekunden sollen die Gegebenheiten (Temperaturen und Flüsse) konstant bleiben.
Erst nach Beendigung der Probennahme dürfen die Parameter neu eingelesen und eingestellt werden.
Es wurde im Bedienbild der Automatikfunktion zusätzlich ein EA-Feld zur manuellen Eingabe dieses Zeitraumes (da dieser Zeitraum von GC zu GC unterschiedlich ist und die Automatikfunktion eine universelle Softwareerweiterung darstellen soll) und ein EA-Feld zur Anzeige der Restzeit dieses Timers implementiert. Die Variablen, mit denen diese Visualisierungen realisiert wurden, werden im Kapitel **CFC-Programmierung** näher erläutert.

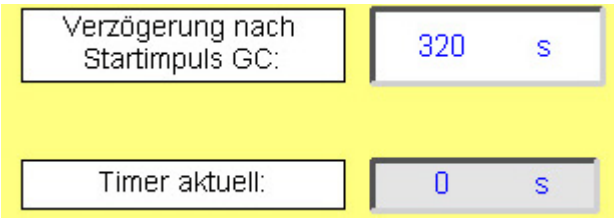


Abbildung 16: Einleseverzögerung und Timerrestzeit

2.2.6 Visualisierung des Startsignals des GC

Um eine schnelle Fehlerdiagnose zu gewährleisten, wurde eine Visualisierung des Startsignals des Gaschromatographen in das Automatikbedienbild eingebunden. Mit dieser kann geprüft werden, ob das besagte Signal überhaupt vom GC abgegeben wurde - und zwar für den Fall, dass eine Zeile der MTD nicht eingelesen wurde oder die Automatikfunktion komplett außer Betrieb gerät.

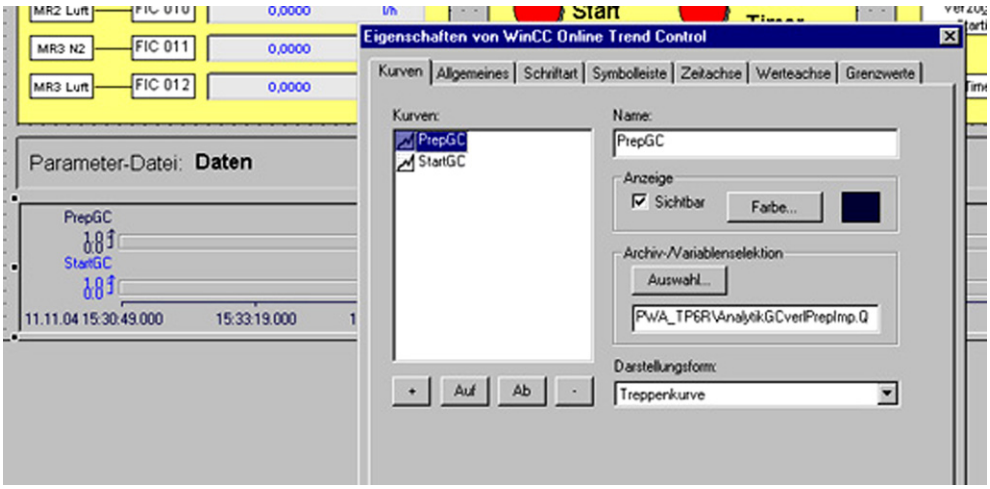


Abbildung 17: WinCC Online Trend Control

Im WinCC gibt es für solche Anzeigen eine Funktion „WinCC Online Trend Control“, bei der man ein Signal gegenüber der Zeit kontinuierlich aufzeichnen und nahezu in Echtzeit ausgeben kann. Natürlich muss man dabei aber auch die Übertragungsgeschwindigkeiten zwischen der SPS[3] und dem Bedien-PC und die im Aufzeichnungs- und Archivierungszyklus des PCs berücksichtigen. Denn dies sind Hauptfaktoren für Verzögerungen, die in solch einem System auftreten.

2.2.7 Die Einbettung des Bedienbildes für die Automatikfunktion

Damit das erstellte Bedienbild für die Automatikfunktion auch in der Runtime-Umgebung neben den schon vorhandenen Bedienbildern für Betriebsarten, Regler, Prozessüberwachung usw. angewählt werden kann, wurde das der Automatikfunktion im sogenannten Picture Tree Manager eingebettet. Dieser ist Bestandteil des WinCC.

[3]SPS S̄peicherprogrammierbare Steuerung - in diesem Fall eine Simatic S7-400

Das Bedienbild für die Automatikfunktion wird im pdl-Format in einen neu erstellten Container mit dem Namen „Automatik“ in die Bilderhierarchie integriert.

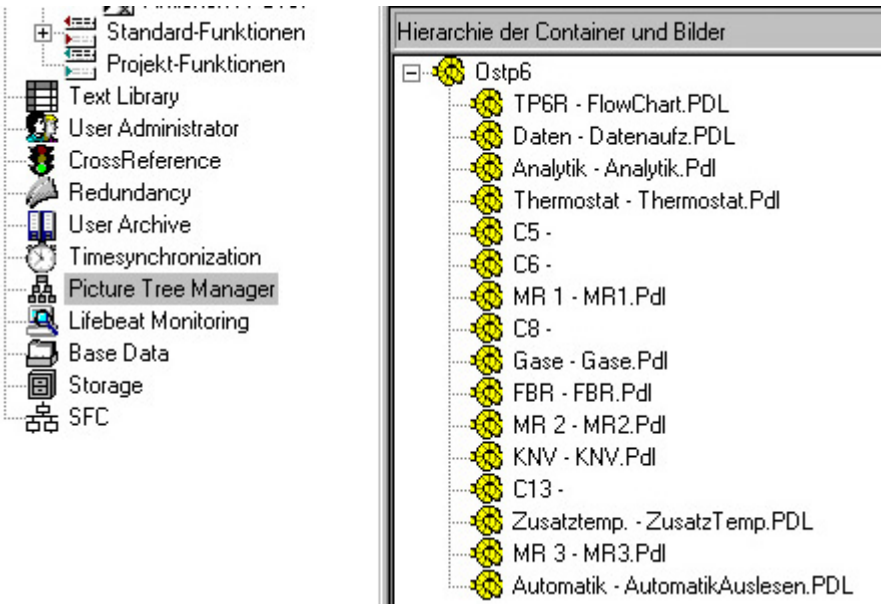


Abbildung 18: Einbettung des Bedienbildes im Picture Tree

Nach dem Einbetten in den Picture Tree konnte die Automatikfiunktion innerhalb der Runtime-Umgebung ausgewählt werden.

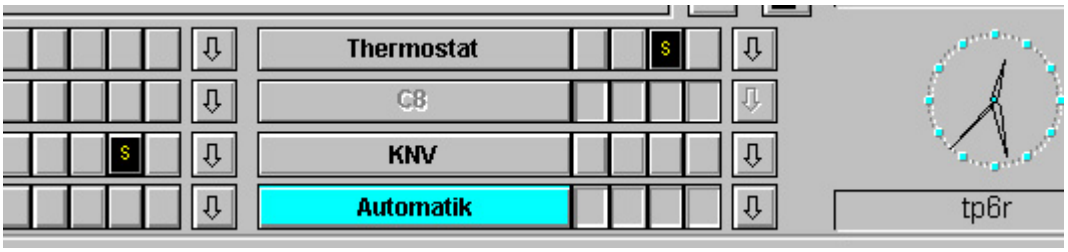


Abbildung 19: Bedienbildauswahl in der WinCC-Runtime-Umgebung

2.3 CFC-Programmierung

Um einen geordneten Ablauf einiger für die Automatikfunktion unentbehrlichen Funktionen auf der Steuerungsebene zu realisieren, wurde der CFC-Editor benutzt. Dieser ist Bestandteil des Siemens Simatic Managers.

2.3.1 Verlängerung des Start-Impulses für Visualisierung

Das Startsignal des Gaschromatographen ist zu kurz, um es auf dem Bedienbild der Automatikfunktion so anzuzeigen, dass ein Bediener erkennen kann, ob das Signal abgegeben wurde. Es war notwendig, dieses Problem für den Zweck der Visualisierung zu lösen.

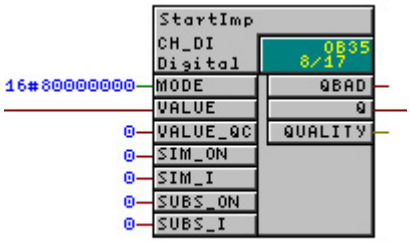


Abbildung 20: `CH_DI` Digital Baustein zur Aufnahme des Startsignals

Dazu wurde das Signal mit Hilfe eines `TIMER_P`-Bausteins der Gruppe Impulsfunktionen aus der Bausteinbibliothek des CFC-Editors verlängert.

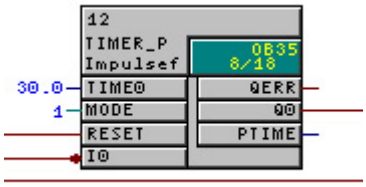


Abbildung 21: `TIMER_P` Impulsef Baustein zur Verlängerung des Startsignals

Dieser Baustein wird dazu im `Mode 1` betrieben, der das anstehende Signal am Eingang `I0` um einen am Eingang `TIME0` festgelegten Wert in Sekunden verlängert.

Um dieses digitale Signal auf Bedienbildern im WinCC anzeigen zu können, benötigt man einen Schnittstellenbaustein - in diesem Falle einen Digital Monitoring Baustein (kurz: `DIG_MON`) der Gruppe Monitor.

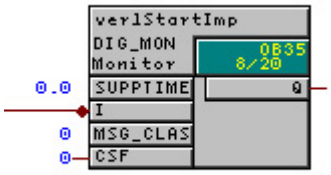


Abbildung 22: `DIG_MON` Monitor Baustein zur Anzeige des verlängerten Startsignals im WinCC

Der Eingang `I` dieses Monitoring-Bausteins wurde nun in der WinCC Online Trend Control Funktion (Abschnitt 2.2.6) benutzt, um das verlängerte Startsignal anzeigen zu können.

2.3.2 Der 320-Sekunden-Timer

Grund für die Implementierung des Timers sind die Schaltverzögerungen bim Öffnen der Ventile des Multipositionsventils.
Vorausgesetzt, der Eingang `IO` des digitalen Operationsbausteins (`OP_D`) `FreigGCImp` ist durch Klicken des Startbuttons (Abschnitt 2.2.3) auf `High` gesetzt und gleichzeitig erfolgt der Startimpuls vom Gaschromatographen, wird ein `TIMER_P`-Baustein durch Schalten auf `High` am Eingang `IO` in der Ablaufkette aktiviert.

Abbildung 23: digitaler `OP_D` Operationsaustein zum An- und Ausschalten der Automatikfunktion

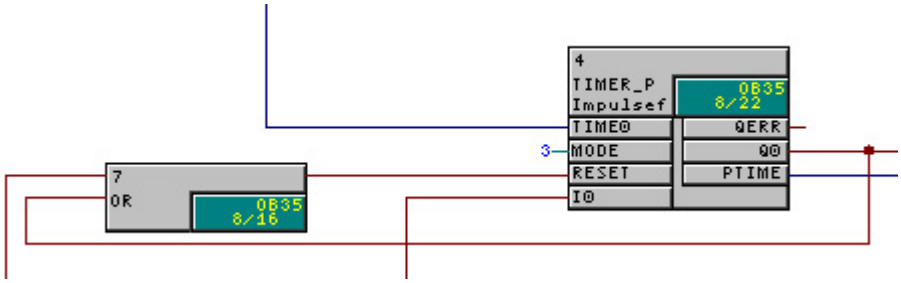


Abbildung 24: TIMER_P-Baustein im Countdown-Modus 3

Dieser Baustein läuft im Mode 3[4] und der Delaywert, von dem der Timer zurückzählen soll, wird durch manuelle Eingabe am Eingang U des analogen Operationsbausteins mit Begrenzung (OP_A_LIM) vorgegeben.

Abbildung 25: analoger OP_A_LIM Operationsaustein mit Begrenzung zur Vorgabe des Delaywertes

Dieser analoge Operationsbaustein ist ein Schnittstellenbaustein, der eine Eingabe aus dem WinCC direkt in den Ablauf innerhalb der Steuerung ermöglicht.
Es wurde in diesem Fall ein Operationsbaustein mit Begrenzung gewählt, um einem Missbrauch sowie Bedienungsfehlern vorzubeugen und somit ge–gebenenfalls entstehenden Schaden an der Masse der Messdaten zu minimieren.
Der Eingang U dieses Bausteins wurde mit dem EA-Feld im WinCC zur Einstellung der Delayzeit nach dem Startsignal vom GC (Abschnitt 2.2.5) verknüpft.
Der Ausgang PTIME des TIMER_P Bausteins gibt die Restzeit bis zum Wert 0 beim Herunterzählen des vorgegebenen Delaywertes an. Um diese Restzeit

[4] Mode 3 eines TIMER_P-Bausteins beinhaltet die Funktion, nach dem Ablauf eines vorgegebenen Wertes (in Sekunden) am Eingang TIME0 den Ausgang Q0 auf High zu setzen - quasi als reiner Countdowntimer zu fungieren

im WinCC anzeigen zu können, wurde dieser Ausgang im CFC-Plan direkt mit dem Eingang eines `MEAS_MON` (Measuring Monitor) Bausteins verbunden, der ebenfalls als Schnittstellenbaustein fungiert und analoge Werte aus der Steuerung erfasst und im WinCC ausgeben kann.

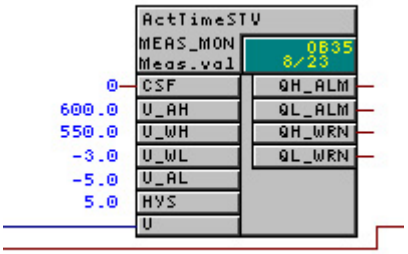


Abbildung 26: analoger `MEAS_MON` Erfassungsbaustein für die Anzeige der Restzeit des Timers im WinCC

Durch Verknüpfung des Einganges dieses Bausteins mit dem EA-Feld (Abschnitt 2.2.5) im Bedienbild für die Automatikfunktion erfolgt die Anzeige der Restzeit des Timers nach dem Startsignal des GC.
Ist der Timer abgelaufen, wird der Ausgang `Q0` dieses `TIMER_P` Bausteins auf 1 gesetzt. Dieses `High` Signal setzt zugleich den Timer zurück, sodass er bei dem nächsten Startsignal vom Gaschromatographen wieder von den am Eingang `TIME0` anliegenden 320 Sekunden abwärts zählen kann. Dazu wurde der Ausgang `Q0` direkt mit dem bausteineigenen Eingang `RESET` dieses `TIMER_P` Bausteins über einen `OR` Logikbausteins verknüpft.
Der Timer kann jedoch auch manuell aus dem WinCC heraus zurückgesetzt werden, indem der Reset Timer-Button angeklickt wird.

Dieser Button ist verknüpft mit dem Eingang `IO` eines digitalen Operationsbausteins (`OP_D`), dessen Ausgang `QO` mit dem zweiten Eingang des besagten `OR` Logikbausteins verbunden ist.

Abbildung 27: digitaler `OP_D` Operationsbaustein zum Zurücksetzen des 320-Sekunden-Timers

Der Ausgang dieses Logikbausteins war mit dem Eingang `RESET` dieses `TIMER_P` Bausteins zu verbinden (siehe Abbildung 2.3.2). Der Operationsbaustein 35 (`OB35` - siehe Abbildung 26) definiert, dass der `High` Wert am Ausgang `QO` des Timers nur 100 Millisekunden lang anliegt. Bei dem nächsten Abtastzyklus der SPS liegt an diesem Ausgang wieder ein `Low` Signal an. Dieses kurze `High` Signal nach Ablauf des Timers wird genutzt, um die bedeutsamste Bausteingruppe in dem Automatik-CFC-Plan zu aktivieren.

2.3.3 Ein RS-Flip-Flop als Trigger

Da der Timer immer nur `High` Signale ausgibt, wenn der Countdown beendet wurde, musste ein Weg gefunden werden, der gewährleistet, dass bei jedem `High` Signal des Timerausganges eine Änderung des für die letztendliche Triggerfunktion genutzten Signals stattfindet. Das Ausgangssignal des Timers wurde dazu mit zwei `UND` Logikbausteinen verknüpft, die wiederum mit einem `RS-Flip Flop` verbunden wurden.

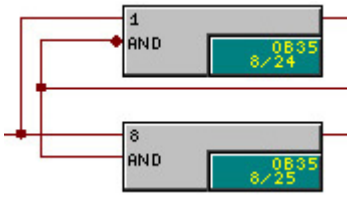


Abbildung 28: 2 `UND` Logikbausteine

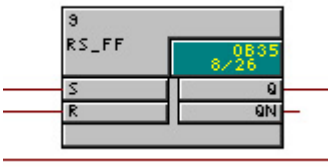


Abbildung 29: Der RS-Flip Flop (`RS_FF`)

Dem Ausgang `Q` des RS-Flip Flops ist ein digitaler Monitoring-Baustein (`DIG_MON`) angefügt, der als Schnittstellenbaustein fungierend den Logikpegel des Ausganges `Q` vom RS-Flip Flop zur Anzeige und Weiterverarbeitung im WinCC aufnimmt.

Abbildung 30: Digitaler Monitoring Baustein (`DIG_MON`)

Das Signal am Eingang `I` dieses `DIG_MON` Bausteines wird als Anstoßsignal des im Abschnitt „Das Ablaufscript der Automatik“ beschriebenen Algorithmus genutzt. Jener Algoritmus wird durch Änderung des besagten Signals - also bei Wechsel zwischen `High` zu `Low` und anders herum - getriggert[5].
Um solch einen Wechsel verwirklichen zu können, wenn der Ausgang des Timers nach Ablauf des Countdowns immer nur `High` Signale liefert, ist der Einsatz des erwähnten RS-Flip Flops in Verbindung mit den Logikbausteinen nötig.
Nachfolgend ist eine Wahrheitstabelle zur Erläuterung der Funktionsweise dieses Flip Flops dargestellt.

S	**R**	**Q**	**Aktion**
1	0	1	Setzen
0	1	0	Rücksetzen

[5]angestoßen bzw. in Gang gebracht

Um einen solchen Wechsel am Ausgang `Q` des RS-Flip Flops (`RS_FF`) zu erzeugen, wird das Ausgangssignal `Q` des `RS_FF` einmal negiert und einmal nicht negiert an jeweils einen Eingang der zwei vorgelagerten `UND` Bausteine geknüpft. Den jeweils anderen Eingang der beiden `UND` Bausteine belegt das Ausgangssignal `Q0` des Timers.
Da das Ausgangssignal des Timers bei Ablauf des Countdowns `High` Level hat, steht an jeweils einem Eingang der zwei `UND` Bausteine am Ende des Countdowns auch ein `High` Signal an. Somit ist für die Betrachtung der Wechselfunktion vom Ausgangssignals `Q` des `RS_FF` nur die Belegung der Eingänge `S` und `R` maßgeblich.

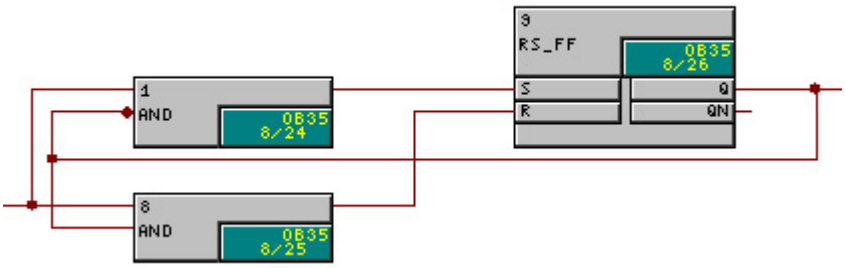


Abbildung 31: Verschaltung von 2 `UND` Bausteinen und einem `RS`-Flip Flop

Die folgende erweiterte Wahrheitstabelle stellt die Funktionsweise und die „Wechselwirkung“ des Zusammenspiels von `UND` Bausteinen und RS-Flip Flop dar, wobei `E1` und `E2` die Eingänge des in Abbildung 31 dargestellten `UND` Bausteins mit dem Namen 1 darstellen und `E3` bzw. `E4` die Eingänge des `UND` Bausteins mit dem Namen 8 sind.

E1	**E2**	**E3**	**E4**	**S**	**R**	**Q**
1	0	1	1	0	1	0
1	1	1	0	1	0	1

Die speichernde Wirkung des RS-Flip Flops wird genutzt, wenn das Ausgangssignal des Timers auf `Low` Level ist und an den Eingängen `S` und `R` des `RS_FF` jeweils ein `Low` Signal ansteht.

2.3.4 Die richtige Ablaufreihenfolge

Bei der Erstellung der CFC-Pläne für die Automatik war es - wie in jedem CFC-Plan - angebracht, die einzelnen Bausteine in eine bestimmte Ablaufreihenfolge zu bringen, damit die logische Abarbeitung der einzelne Bausteine - bedingt durch die zyklische Arbeitsweise der SPS - gewährleistet wird. Andererseits würde es zu Verzögerungen und letztendlich auch Fehlern im Prozessablauf kommen. Diese Ablaufreihenfolge kann entweder direkt beim Erstellen des CFC-Plans realisiert werden oder es ist dafür der Ablaufeditor im CFC-Editor des Simatic Managers zu verwenden, um eventuell nachinstallierte Bausteine in eine schon geordnete Ablaufreihenfolge einzusortieren.

2.3.5 Der AS-OS-Transfer

Der AS-OS-Transfer ist ein Feature von PCS7, das diesem Prozessleitsystem einen sehr entscheidenden Vorteil anderen Prozessleitsystemen gegenüber verschafft. Normalerweise ist es üblich, Variablen und Verbindungen, sowie Meldungen einmal auf der Steuerungsseite und einmal auf der Bedien- und Beobachtungsseite einzurichten. Diese Prozedur ist sehr aufwendig.
Der AS-OS-Transfer von PCS7 führt diesen Abgleich von Verbindungen, Variablen und Meldungen vollautomatisch innerhalb kürzester Zeit durch. Dieser Transfer macht das Prozessleitsystem PCS7 sehr komfortabel gegenüber anderen Lösungen.
Der CFC-Plan für die Automatikfunktion enthält viele Schnittstellenbausteine, die mit dem WinCC im laufenden Betrieb der Automatikfunktion kommunizieren müssen. Um diese Kommunikation und auch eine Einbettung der erstellten Bausteine in das Gesamtumfeld der Steuerung zu realisieren, war der AS-OS-Transfer durchzuführen.
Er macht die Ein- und Ausgänge der Schnittstellenbausteine im WinCC sichtbar und auch bedienbar. Zudem stellt er die Kommunikationsverbindung zwischen Aktionen im WinCC und deren Reaktionen in der Steuerung und umgekehrt her.

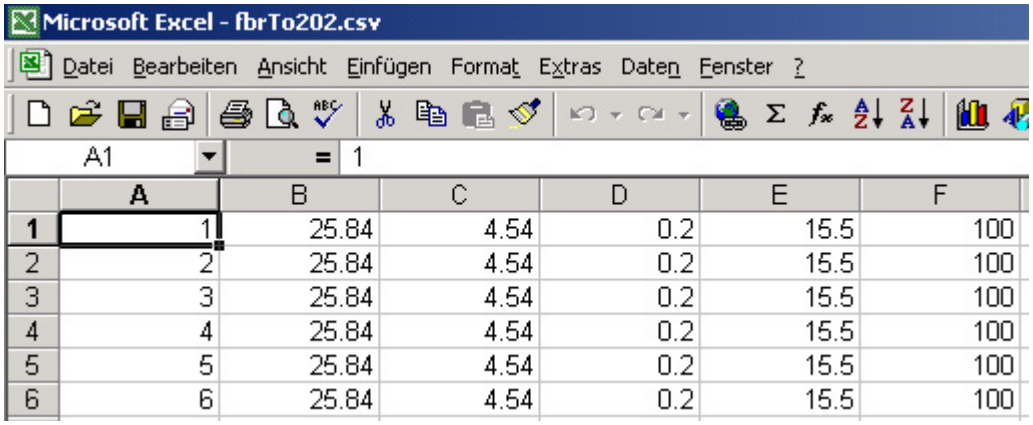
Abbildung 32: Der AS-OS-Transfer macht die Kommunikation zwischen Steuerung und WinCC möglich

2.4 Das Ablaufscript der Automatik

Damit ein vorgegebener zyklischer Ablauf der sogenannten Automatikfunktion realisiert werden konnte, war im Global Script Editor, der Bestandteil des WinCC ist, ein Algorithmus mit Hilfe von Elementen der Standard-C Programmiersprache zu entwerfen, der zyklisch getriggert werden kann und den reibungslosen Ablauf der Automatik unterstützt.

2.4.1 Die Matrix-Text-Datei

In diesem Abschnitt soll erläutert werden, warum die Matrix-Text-Datei unbedingt in Matrixform erstellt werden muss.
Diese Matrixform ist sowohl aus praktischer als auch aus funktioneller Sicht allen anderen Formen vorzuziehen.
Zum einen wird für den Benutzer auf einen Blick klar, welcher Wert in dieser großen Tabelle für welchen Sollwert in der Steuerung steht und für welchen Messzyklus innerhalb einer Messung er relevant ist.



	A	B	C	D	E	F
1	1	25.84	4.54	0.2	15.5	100
2	2	25.84	4.54	0.2	15.5	100
3	3	25.84	4.54	0.2	15.5	100
4	4	25.84	4.54	0.2	15.5	100
5	5	25.84	4.54	0.2	15.5	100
6	6	25.84	4.54	0.2	15.5	100

Abbildung 33: Auszug aus der MTD mit laufender Zeilennummer von 1 beginnend

Zum anderen wird im Ablaufscript, das zyklisch während einer Messung immer wieder abgearbeitet wird, jedes mal die aktuelle Zeile mitgezählt, die gerade bearbeitet wird. Die Anzeige der gerade eingelesenen Zeile und die Position-and-Run Funktion profitieren von dieser Funktion. Da die MTD die Form einer Matrix hat und in der ersten Spalte jeder Zeile eine mit größerer Zeilenanzahl sich um 1 vergrößernde Zahl (mit der Zahl 1 in der ersten Zeile beginnend) steht, war es besonders einfach, einen Algorithmus zu entwerfen, der diese Zahl ausliest und zur weiteren Verarbeitung verarbeitet.

51	18.28	12.1	0.2	15.5	100
52	18.28	12.1	0.2	15.5	100
53	18.28	12.1	0.2	15.5	100
54	18.28	12.1	0.2	15.5	100
55	20	0	0	15	50
0	0	20	0	15	50

Abbildung 34: Die letzte Zeile der MTD beginnt mit einer Null

In der ersten Spalte der letzten Zeile der MTD steht eine Null, deren Funktion auf Seite 32 näher erläutert wird.

2.4.2 Die *.pas Datei

Der Ablaufalgorithmus wird in einer Datei mit der Endung *.pas gespeichert. Als Grundprogrammiersprache wird hierbei Standard-C verwendet.

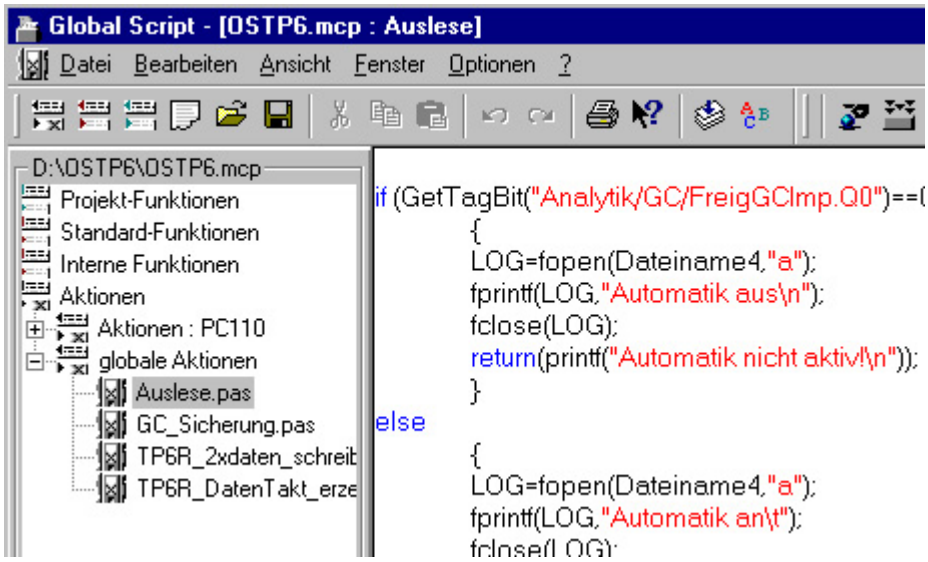


Abbildung 35: Die Datei Auslese.pas wird im Global Script Editor bearbeitet

Im WinCC ist ein dazu benötigter Compiler mit einer zusätzlichen Befehlsbibliothek für WinCC-Funktionen integriert, der ein angenehmes und schnelles Arbeiten ermöglicht.

2.4.3 Der Algorithmus

Zu Beginn des Scriptes waren Dateizeiger und Variablen zu deklarieren, die für den Ablauf der Automatik von großer Bedeutung sind.
Die Variablen `Dateiname1` und `Dateiname2` sind wichtig für die Funktion, die in Abschnitt 2.2.1 schon erläutert wurde. Für diese Variablen wurden jeweils 60 Zeichen reserviert. Die Bedeutungen der `copystring2`-Variable sowie die Variable `auswahl` werden auf den Seiten 28 bzw 29 später näher erläutert.
`Nummer` enthält den in Absatz 2.2.1 schon erwähnten Zeichensatz aus `File-ID` - quasi den Namen der MTD. Alle oben genannten Variablen sind vom Typ `char`, das heißt, sie können Zeichenketten aus Buchstaben, Sonderzeichen und Ziffern bis zu einer in eckigen Klammern angegebenen Länge enthalten. Darüber hinaus wurden Laufvariablen vom Typ `int`[6] und Variablen im `double` Format, in denen diverse Zahlenwerte, die u.a. Auskunft über die aktuell bearbeitete Zeile (siehe Abschnitt 2.4.1) geben, gespeichert werden.
Die Variablen `LNR`, `fdata1`, `fdata2` usw. sind Variablen, die bei dem Auslesen der Sollwerte aus der MTD die Funktion der Zwischenspeicherung erfüllen, bevor die Werte auf die entsprechenden Sollwerteingänge der dafür vorgesehenen Bausteine in der Steuerung geschrieben werden. `LNR, fdata1,fdata2, ... , fdata18` enthalten dabei die Daten der Spaltenelemente der gerade bearbeiteten Zeile in der MTD, wobei in `LNR` die laufende Nummer zur Anzeige der Zeile zwischengespeichert wird. Diese Zwischenspeichervariablen haben `float`[7] Format, da Sollwerte für Flüsse sehr oft mit mehreren Nachkommastellen angegeben werden.

[6] Integervariablen können nur natürliche Zahlen enthalten.
[7] Floatvariablen sind Gleitkommazahlen.

```
{

FILE *Daten;
FILE *Daten2;
FILE *Lauf;                    //Datei gespeichert auf Festplatte

char Dateiname1[60];
char Dateiname2[60];
char copystring2[130];
char auswahl[31];
char Nummer[9];
int Auslese, i,k=1,stepist,stopc;
double stepc,posnrun,Zahl;
float LNR,fdata1,fdata2,fdata3,fdata4,fdata5,fdata6,fdata7,fdata8,fdata9,fda
```

Abbildung 36: Deklarieren der Variablen am Anfang des Algorithmus

Zum Auslesen einer binären Variable aus einem Baustein in der Steuerung wird der `GetTagBit` Befehl genutzt. Diesen Befehl kann man direkt mit einer Abfrage verbinden und dann gegebenenfalls sofort eine Maßnahme ergreifen lassen, die auf eine positive Abfrage dieser Variable, die in diesem Fall entweder einen `High` oder `Low` Zustand enthält, folgen soll.
Diese Abfrage stellt fest, ob der Startbutton im Bedienbild der Automatikfunktion tatsächlich angeklickt wurde bzw. ob die Automatikfunktion iniziiert wurde.

```
if (GetTagBit("Analytik/GC/FreigGCImp.Q0")==0) return(printf("Automatik nicht aktiv!\n"));
```

Abbildung 37: Abfrage des Zustandes des für die Iniziierung der Automatikfunktion wichtigen Bausteinausganges

Wenn der Ausgang `Q0` dieses `FreiGCImp` Bausteins `Low` level besitzt, soll die weitere Bearbeitung Datei `Auslese.pas` mit der Ausgabe eines entsprechenden Warntextes im „GSC-Diagnose-Fenster“[8] gestoppt und abgebrochen werden.
Wenn der Ausgang `Q0` den Zustand `High` besitzt, wird das Script weiter ausgeführt.

[8]Im Bedienbild der Automatikfunktion ist ein Button implementiert, der zur Fehlerdiagnose und Step-by-Step Fehlersuche ein WinCC-internes Fenster aufruft, das mit dem Globel-Script-Editor verknüpft ist und im WinCC normalerweise nicht sichtbare `printf` Befehle ausgeben kann

Zunächst wird getestet, ob die MTD auf der lokalen Festplatte vorhanden ist. Dies geschieht durch den `Daten2=fopen` Befehl, dem eine Abfrage folgt, die bei gescheitertem Öffnen der Datei die WinCC-interne binäre Variable `lokal_vorhanden` entsprechend mit einer Null versieht. Ist die MTD lokal vorhanden und lässt sich öffnen, wird `lokal_vorhanden` mit einer 1 beschrieben und die Datei wird wieder geschlossen.

```
Daten2=fopen(Dateiname2,"a");

if(Daten2==NULL)        SetTagBit("lokal_vorhanden",0);
else                    {SetTagBit("lokal_vorhanden",1);
                        fclose(Daten2);}
```

Abbildung 38: Abfrage, ob MTD lokal vorhanden ist

Falls die MTD lokal nicht vorhanden ist, wird die MTD vom Netzlaufwerk, auf dem sie vor jeder Messung abgespeichert wird und von dem die Automatikfunktion im Normalbetrieb die Sollwertdaten ausliest, auf das lokale Laufwerk kopiert. Dadurch kann bei Netzwerkausfällen der Automatikbetrieb durch Auslese der MTD von lokal weitergeführt werden.
Für diese Aktion war ein DOS-Befehl für Kopiervorgänge zu implementieren, der im Hintergrund den gesamten Inhalt eines Pfades in einen anderen Pfad kopiert. Die Befehle `xcopy` und `WinExec` sowie `SW_HIDE` bewirken, dass die MTD von dem Netzlaufwerk im Hintergrund (also nicht sichtbar) auf das lokale Laufwerk kopiert wird. Die Variable `copystring2` enthält auf bis zu 130 Zeichen die Zeichenketten mit den Pfadangaben der beiden Variablen `Dateiname1` und `Dateiname2`,

```
sprintf(copystring2, "xcopy %s %s", Dateiname1,Dateiname2);
printf("Copystring: %s \n", copystring2);
WinExec(copystring2, SW_HIDE);
```

Abbildung 39: Kopieren der MTD vom Netz auf das lokale Laufwerk

wobei diese beiden Variablen ihren Inhalt vorher aus den WinCC-externen Variablen[9] `Datafile_1` und `Datafile_2` mit dem `GetTagChar` Befehl erhalten haben.

[9]Ihr Wert wird automatisch aus der Steuerung über eine Netzverbindung geholt.

```
strcpy(Dateiname1, GetTagChar("Datafile_1"));
strcpy(Dateiname2, GetTagChar("Datafile_2"));
```

Abbildung 40: Kopieren der Pfadangaben von externen WinCC Variablen zu Scriptvariablen

Im Normalbetrieb der Automatikfunktion werden die Sollwerte aus der MTD ausgelesen, die auf dem Netzlaufwerk gespeichert wird. Im weiteren Verlauf des Algorithmus für die Automatikfunktion wird zunächst getestet, ob sich die MTD zur Auslese auf dem Netzlaufwerk befindet. In Abhängigkeit davon, ob der Zugriff auf das Netzlaufwerk besteht, wird die Variable `auswahl` entweder mit der Pfadangabe der MTD auf dem lokalen Laufwerk oder der gleichen MTD auf dem Netzlaufwerk beschrieben.
Im Script wird also davon ausgegangen, dass es einen Netzfehler gibt, sobald die MTD auf dem Netzlaufwerk nicht geöffnet werden kann. Dann wird die Variable `auswahl` wird mit der Pfadangabe der MTD auf dem lokalen Laufwerk (`Dateiname2`) beschrieben.
Kann die MTD auf dem Netzlaufwerk jedoch geöffnet werden, bekommt die Variable `auswahl` den Inhalt der Variable `Dateiname1`. Die Datei wird wieder geschlossen.

```
Daten2=fopen(Dateiname1,"a");

if(Daten2==NULL)
        {
        SetTagBit("vorhanden",0);
        strcpy(auswahl,Dateiname2);
        }
else    {SetTagBit("vorhanden",1);strcpy(auswahl,Dateiname1);
        fclose(Daten2);}
```

Abbildung 41: Test, ob Netzfehler vorliegt oder nicht

Die WinCC-interne Variable `ZahlCC` spielt eine große Rolle bei dem korrekten Erkennen der gerade eingelesenen Zeile der MTD. Sie wurde als 32Bit Gleitkommavariable erstellt, die vor jedem Start der Automatikfunktion den Startwert 0 besitzt. So wird von Anfang an ein richtiges Aufwärtszählen pro Aufruf der Datei `Auslese.pas`[10] gewährleistet.

[10] bei zyklischer Triggerung durch die Änderung des Ausganges `Q` des Bausteins `Start+5min` (siehe Abschnitt 2.3.3)

Der Scriptinternen Variable `Zahl` wird im nächsten Schritt der Wert dieser WinCC-internen Variable `ZahlCC` zugewiesen, um mit diesem Wert innerhalb des Scriptablaufs arbeiten zu können.

```
Zahl = (GetTagDouble("ZahlCC"));
```

Abbildung 42: Überschreiben der Information über die Zeile vom WinCC zum Script

Die Realisierung der Position-and-Run Funktion innerhalb des Scriptablaufs wird im folgenden Abschnitt beschrieben.
Die WinCC-interne Variable `stepsoll` enthält die jeweilig auszulesende Zeile durch das Ausfüllen des EA-Feldes für die Position-and-Run Funktion (Abschnitt 2.2.4) im Automatikbedienbild. Zunächst wird der Wert dieser Variable zur Weiterverarbeitung im Scriptablauf auf die Variable `posnrun` geschrieben. Danach wird überprüft, ob diese Variable einen Wert ungleich 0 hat. Ist dem so, wird die Variable `Zahl`, die die aktuell zu bearbeitende Zeilenzahl enthält, mit dem um 1 kleineren Wert der Variable `posnrun` beschrieben. Der Wert muss um 1 kleiner sein, da bei dem nächsten Durchlauf des Scriptes (bei einer neuen Messequenz) die Variable `ZahlCC` um 1 erhöht wird. Sonst würde die Zeile bearbeitet werden, die in der MTD unter der steht, die eigentlich bearbeitet werden soll. Gleichzeitig wird der Wert der Variable `posnrun` auf 0 gesetzt - nur dadurch kann gewährleistet werden, dass beim nächsten Durchlauf des Scriptes die Position-and-Run Funktion nicht iniziiert wird. Die Variable `stepsoll` im WinCC wird ebenfalls auf 0 gesetzt.

```
posnrun = (GetTagDouble("stepsoll"));

if (posnrun!=0)   {Zahl = posnrun-1;
                   posnrun=0;
                   SetTagDouble("stepsoll",0);
                   };
```

Abbildung 43: Quelltext der Position-and-Run Funktion

Direkt nach der Position-and-Run Abfrage wird die Variable `Zahl` um 1 vergrößert. Dies bewirkt das auf Seite 29 beschriebene Aufwärtszählen bei jedem Aufruf der Scriptdatei.

```
++Zahl;
stepc = Zahl;
```

Abbildung 44: `Zahl` wird um 1 vergrößert und auf `stepc` überschrieben

Danach wird der Wert der „neuen“ Variable `Zahl` auf die Variable `stepc` überschrieben, die in der Ausleseaktion der MTD ihre Verwendung findet.

Der eigentliche Auslesevorgang findet erst jetzt statt. Dazu wird die MTD mit dem Attribut „+r“ zum Lesen geöffnet - im Normalbetrieb vom Netzlaufwerk, bei Netzausfällen vom lokalen Laufwerk. Kann die Datei in beiden Fällen nicht geöffnet werden, wird die weitere Bearbeitung der Scriptdatei abgebrochen und in dem GSC-Diagnose-Fenster erscheint dann der Rückgabewert „-2“[11].
Der Ausleseprozess ist durch eine Do-While-Schleife realisiert. Die geöffnete MTD wird nun spaltenweise ausgelesen - Element für Element. Zuerst wird die laufende Nummer am Anfang einer jeden Zeile ausgelesen, danach die 17 Sollwerte für Temperaturen und Flüsse.

```
        Daten=fopen(auswahl,"r+");
        if(Daten==NULL) return (-2);

do{     fscanf(Daten, "\n%f\r", &LNR);
        fscanf(Daten, "%f\r", &fdata1);
        //fscanf(Daten, "%f\r", &fdata2);
        fscanf(Daten, "%f\r", &fdata3);
        fscanf(Daten, "%f\r", &fdata4);
        fscanf(Daten, "%f\r", &fdata5);
        fscanf(Daten, "%f\r", &fdata6);
        fscanf(Daten, "%f\r", &fdata7);
```

Abbildung 45: Der Beginn der Do-While-Schleife für das Auslesen der MTD

[11] In diesem Fall werden diese Rückgabewerte als Fehlercodes genutzt. Fehlercodes sind Abkürzungen für Ausgaben, die bei bestimmten im Scriptablauf auftretenden Fehlern im GSC-Diagnose-Fenster ausgegeben werden. Sie werden vom Programmierer festgelegt, damit dieser weiß, an welcher Stelle des Scriptes der jeweilige Fehler aufgetreten ist. Für die Fehlersuche spielt dieser Rückgabewert eine wichtige Rolle.

Dabei wird jedes mal am Ende des Auslesens jeder Zeile eine zusätzliche Laufvariable um 1 erhöht. In diesem Prozess werden die Zwischenspeichervariablen jedes mal wieder mit den gerade ausgelesenen Sollwerten überschrieben. Sobald der Wert dieser Laufvariable den um 1 veringerten Wert der Variable `stepc` erreicht hat, wird der Auslesevorgang beendet und die Zwischenspeichervariablen behalten den „aktuellen“ Sollwertstand.
Die MTD wird geschlossen.

```
        fscanf(Daten, "%f\r", &fdata15);
        fscanf(Daten, "%f\r", &fdata16);
        fscanf(Daten, "%f\r", &fdata17);
        fscanf(Daten, "%f\r", &fdata18);

        SetTagFloat("step",k);
        ++k;

        }
while(k<stepc+1);

        fclose(Daten);
```

Abbildung 46: Ende der Do-While-Schleife für das Auslesen der MTD

Wenn die letzte Zeile der MTD ausgelesen wurde, ist die Zwischenspeichervariable `LNR` mit einer 0 belegt. In der fortlaufenden Bearbeitung des Automatikscriptes wird nun abgefragt, ob die Variable `LNR` mit einer 0 belegt ist. Ist dies der Fall, wird die Automatik ausgeschaltet, die scriptinterne Variable für die Nummer der ausgelesenen Zeile `Zahl` auf 0 gesetzt und die schon in das System implementierte Datenaufzeichnung zur Erfassung der Messwerte während einer Messung ausgeschaltet.
Im weiteren Verlauf des Automatikscriptes werden nun die Werte der Zwischenspeichervariablen an die Sollwerteingänge der jeweiligen Bausteine in der Steuerung für Flüsse und Temperaturen übergeben.
Bei dieser Prozedur erfolgt an einer Stelle jedoch nochmals eine Abfrage. Der Zuluftstrom kann auf einen Wert kleiner als 9 Liter pro Minute eingestellt werden. Massflowcontroller (MFC) sind jeweils in einem bestimmten Bereich von Flüssen einsetzbar. Für Flüsse unterhalb der 9 l/min - Grenze muss ein anderer MFC eingesetzt werden, als für Flüsse oberhalb dieser Grenze.

```
if (LNR==0)
        {
        SetTagBit("Analytik/GC/FreigGClmp.I0",0);
        Zahl=0;
        SetTagBit("DatenAufzEin",0);
        SetTagBit("DatenAufzEin_1",0);
        SetTagBit("AufzZeile1",0);
        SetTagBit("DatenAufzFehler1",0);
        SetTagBit("DatenAufzFehler2",0);
        SetTagBit("DateiVorh",0);
        };
```

Abbildung 47: Abfrage, ob letzte Zeile der MTD eingelesen wurde

```
if(fdata3<=9)
{
SetTagFloat("Gase/F001/Sollwert.U",fdata3);
SetTagFloat("Gase/F002/Sollwert.U",0);
};
if(fdata3>9)
{
SetTagFloat("Gase/F001/Sollwert.U",0);
SetTagFloat("Gase/F002/Sollwert.U",fdata3);
};
```

Abbildung 48: Separation der Sollwertzuweisung für Zuluftstrom

Aus diesem Grund ist in dem Automatiscript eine Abfrage implementiert, die überwacht, welche Größe der angegebene Zuluftstrom haben soll. Diese Afrage wird mit Hilfe einer einfachen `If`-Abfrage realisiert, die je nach Sollwert für den Zuluftstrom den Wert auf den Eingang des einen für Flüsse unter oder gleich 9 l/min zulässigen oder auf den Eingang des anderen für Flüsse über 9 l/min zulässigen MFCs schreibt.
Die Sollwertübergabe von den Zwischenspeichervariablen auf die Eingänge der Sollwertbausteine[12] war mit dem Befehl `SetTagFloat` zu realisieren.

[12]Die Bausteine für die jeweiligen Sollwerte für Temperaturen und Flüsse sind analoge Operationsbausteine vom Typ `OP_A_LIM`, die zugleich eine obere Sollwertgrenze besitzen, um durch Fehleingaben hervorgerufene Störungen und Gefahren zu vermeiden.

```
SetTagFloat("Gase/F003/Sollwert.U",fdata1);
SetTagFloat("Gase/F004/Sollwert.U",fdata4);
SetTagFloat("Gase/F005/Sollwert.U",fdata6);
SetTagFloat("Gase/F006/Sollwert.U",fdata5);
SetTagFloat("Gase/F007/Sollwert.U",fdata7);
SetTagFloat("Gase/F008/Sollwert.U",fdata8);
SetTagFloat("Gase/F009/Sollwert.U",fdata9);
SetTagFloat("Gase/F010/Sollwert.U",fdata10);
SetTagFloat("Gase/F011/Sollwert.U",fdata11);
SetTagFloat("Gase/F012/Sollwert.U",fdata12);
```

Abbildung 49: Überschreiben der Sollwerte in die Steuerung

Beim Durchlauf des Automatikscriptes wurde der Wert der Variable `Zahl` um 1 erhöht. Der neue Wert dieser Variable wir mit dem Befehl `SetTagDouble` am Ende des Scriptes auf die im WinCC angelegte Variable `ZahlCC` gelegt, damit diese den Wert behalten kann und bim erneuten Durchlauf des Automatikscriptes auf den gespeicherten Wert zurückgegriffen werden kann, um die nächste Zeile aus der MTD zu bearbeiten.

```
        SetTagDouble("ZahlCC",Zahl);

return (0);
}
```

Abbildung 50: Übergabe der Zeilenzahl am Ende des Scriptes

2.4.4 Der Trigger des Automatikscriptes

Als Anstoßmechanismus (Trigger) des im vorherigen Abschnitt beschriebenen Algorithmus zum kompletten Auslesen aller Zeilen der Matrix-Text-Datei wurde der immer wiederkehrende Wechsel des Zustandes am Ausgang des Bausteines `Start+5min` verwendet.

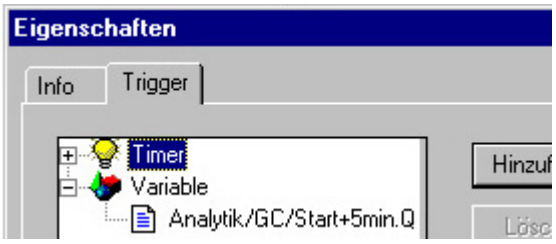


Abbildung 51: Trigger des Automatikscriptes

Dies bedeutet, dass bei Flankenwechsel `High-Low` oder `Low-High` am Ausgang dieses `OP_D` Bausteins das Automatikscript dazu gebracht wird, durchzulaufen. Es läuft so oft durch, bis die Zeile in der MTD erreicht wird, die mit 0 beginnt.
Diese „Nullzeile“ wird ebenfalls bearbeitet. Nachdem die Nullzeile eingelesen wurde und das Script das letzte mal abgearbeitet wurde, wird die Automatik abgeschaltet. Dies wird auf dem Bedienbild der Automatikfunktion durch Ausgabe der Nachricht `Automatik AUS` angezeigt. Darüber hinaus wird nach der Messung die Datenaufzeichnung deaktiviert.
Nach Eingeben einer neuen MTD-ID im Auswahlfeld kann eine neue Messung gestartet werden.
Um zu vermeiden, dass vom Bediener dabei vergessen wird, die Datenaufzeichnung ebenfalls zu starten, ist im Bedienbild ein blinkendes Ausgabefenster implementiert. Dies soll ihn daran erinnern, die Datenaufzeichnung zu starten.

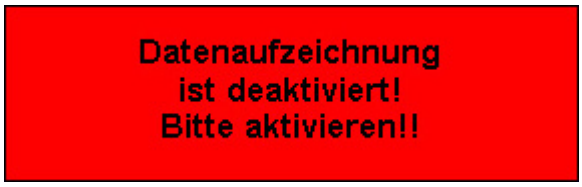


Abbildung 52: Hinweis über den Betriebszustand der Datenaufzeichnung

3 Zusammenfassung

Ziel dieser Studienarbeit war, den Forschern unter Herrn Dr. rer. nat. Frank Klose im Rahmen der DFG Forschergruppe „Membranunterstützte Reaktionsführung“ den Messbetrieb der betrachteten Anlage insofern zu vereinfachen, dass neben der laufenden Messung ungestört auch andere Arbeiten durchgeführt werden können und die Zeit während der Messung effektiv sinnvoll genutzt wird.
Da die Messungen schon vor dieser Aufgabenstellung von Speicherprogrammierbaren Steuerungen unterstützt wurden und ein komplett eingerichtetes Prozessleitsystem zur Verfügung stand, konnte ausgehend von der schon vorhandenen Basis eine Funktion implementiert werden, ohne die Struktur dieser Prozessleitsoftware maßgeblich zu verändern.
Zur Lösung der gestellten Aufgabe war lediglich eine weitere grafische Schnittstelle zu implementieren, die die Kommunikation zwischen dem Benutzer der Anlage und den sich in der Prozessleitsoftware und der Steuerung befindlichen Komponenten zur Realisierung einer solchen Automatikfunktion ermöglichen konnte.
Ferner war innerhalb der Steuerung ein Ablauf zu integrieren, der die vom Gaschromatographen ausgegebenen digitalen Signale aufnehmen und weiterverarbeiten konnte.
Mit Hilfe dieser Steuerungsfunktionen und der verarbeiteten digitalen Signale war es möglich, in der Prozessleitsoftware einen Algorithmus zu entwerfen, der das Kernstück der Automatikfunktion darstellt.

4 Schlussbemerkungen

Innerhalb der Einarbeitungsphase habe ich viel über die Funktionsweise des Prozessleitsystems PCS7 von Siemens gelernt. Wichtige Zusammenhänge zwischen Hardware und Software sind mir klar geworden. Bei der Lösung dieser Aufgabe konnte ich eigene Ideen entwickeln und auch umsetzen, was diese Studienarbeit zu einem großen Erlebnis gemacht hat.
Ich bedanke mich beim Max-Planck-Institut für Dynamik komplexer technischer Systeme Magdeburg für die Bereitstellung der Versuchsanlage als Testobjekt und Entwicklungsumgebung.
Außerdem bedanke ich mich bei den Mitarbeitern des Max-Planck-Instituts Magdeburg für die freundliche und kompetente Beratung und Unterstützung während meiner Arbeiten.
Nicht nur ich habe von dieser Studienarbeit profitiert.
Der Messdurchsatz hat sich vervielfacht; ganze Scharen von Messungen können im Gegensatz zu vorher mehrere Tage lang durchlaufen. Die Mitarbeiter der Forschergruppe können sich zusätzlich auf andere Aufgaben konzentrieren, während die Messungen laufen. Lediglich die große Menge an Auswertungen, die erledigt werden müssen, dürfte negativ ins Gewicht fallen. Auf der anderen Seite werden die Mitarbeiter mit einer großen Fülle von verwendbaren Messwerten belohnt.
Diese Automatikfunktion ist universell einsetzbar und könnte durchaus in noch mehr Versuchsständen implementiert werden, um auch für andere Mitarbeiter des Max-Planck-Instituts für Dynamik komplexer technischer Systeme Magdeburg mehr Zeit für andere Tätigkeitsfelder während des vollautomatischen Messbetriebes zu gewinnen.