

Zusammenfassung IAUT

Eric Lindegger – HS25

Danke fürs Herunterladen und Verwenden meiner IAUT Zusammenfassung! Hoffe ihr findet sie nützlich. Falls ihr Fehler findet oder Inputs habt, schreibt mir doch auf eric.lindegger@stud.hslu.ch

Inhaltsverzeichnis

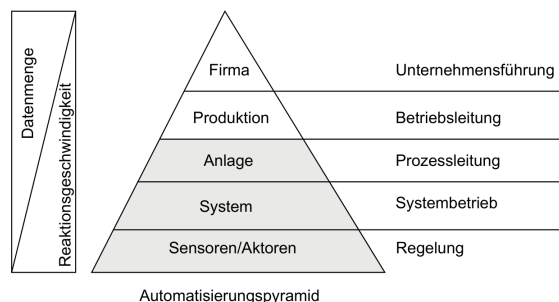
1	Automatisierung	2
1.1	Steuerung - Überblick	2
1.1.1	Klemmen	2
1.2	Allgemeine Aufgaben	2
1.3	Allgemeine Anforderungen	2
2	SPS	3
2.1	Feldbus	3
2.1.1	Spezifische Feldbusse	3
3	Programmierung	3
3.1	EN 61499	3
4	Digitale Regelung	3
5	FSM - Finite State Machine	3
6	ST-Language	3
6.1	Grundaufbau	3
6.2	Variablentypen	3
6.2.1	DUT - Data Unit Type	3
6.2.2	Programm	4
6.2.3	Funktionsblock	4
6.2.4	Funktion	4
6.3	Variablen-Scope	4
6.3.1	Globale Variablen	4
6.4	Conditional Statements	4
6.4.1	If/Else	4
6.4.2	Switch-Case	4
7	TwinCat	4
8	Beispiel Code	4

1 Automatisierung

Definition Automatisierung

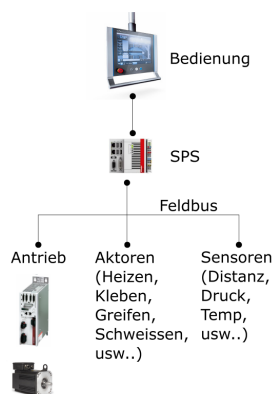
Unter Automatisierung (DIN19222) versteht man das gezielte Ausrüsten einer Einrichtung, so dass sie ganz oder teilweise ohne Mitwirkung des Menschen geschieht und arbeiten kann.

Die Automatisierungspyramide ist der Versuch die gesamte Automatisierungskette auf ein Bild abzubilden:



Die Darstellung ist wie folgt zu interpretieren: Je weiter unten, desto schneller müssen die Komponenten reagieren und je weiter oben, ist die Datenmenge grösser. Die Absicht der Automatisierung ist, dass die automatisierte Anlage bis ins ERP der Firma eingepflegt wird. So muss z.B. keine spezielle Benutzeroberfläche geladen werden um ein neuer Prozess auf der Anlage zu starten.

1.1 Steuerung - Überblick



Die Bedienung ist wo das GUI (Graphical User Interface) dargestellt wird. Hier werden vom Benutzer Eingaben getätigt, wie z.B. Job oder Rezeptwahl.

Der Controller innerhalb der Steuerung/Anlage ist die CPU/SPS. Eine Automatisierung ist häufig eine (PID-)Regelungsaufgabe. Die SPS ist mit Aktoren und Sensoren verbunden:

- Sensoren liefern Informationen
- Aktoren lösen Aktion aus

Die Schnittstellen der Sensoren/Aktoren ist entweder eine Spannung oder einen Strom. Die Spannung ist jedoch eher weniger gängig im Moment, da die Spannung anfälliger auf Störeinflüsse ist als ein Strom. Die gängigsten Schnittstellen sind. Ein weiterer Vorteil der Stromschnittstelle ist, dass auch ein Kabelbruch detektiert werden kann, da der Strom nie 0 wird:

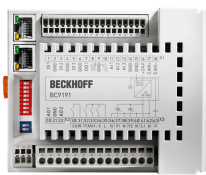
- 0V – 10V
- 4mA – 20mA

1.1.1 Klemmen

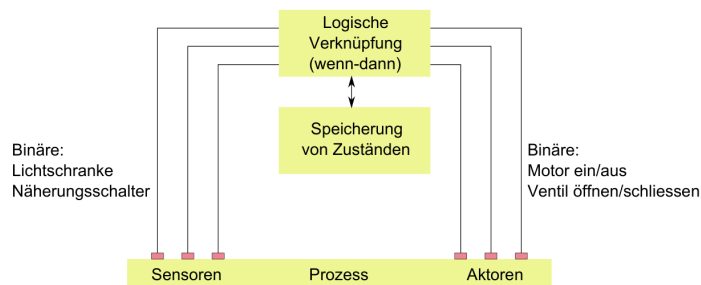
Die Klemmen stellen die Ein- und Ausgänge einer SPS dar und können auf der Feldebene der SPS über ein Bussystem erweitert werden:



2 SPS



Eine SPS ist eine 'Speicherprogrammierbare Steuerung'. Auf Englisch wird sie PLC (Programmable Logic Controller) genannt. Sehr wichtig: die SPS arbeitet konventionell möglichst auf if/else Statements und verarbeitet sämtlicher Code in jedem Zyklus einmal. Das Arbeitsprinzip ist:



2.1 Feldbus

Der Feldbus ist die Kommunikationsschnittstelle der SPS zu den Klemmen und entsprechend Ein- und Ausgängen. Es hängen alle E/A seriell an einem Bus. Wo der Feldbus aktiv ist, wird Feldebene genannt. Der Aufbau des Feldbusses kann wie folgt vorgestellt werden:



Definition Feldbus

Ein Feldbus verbindet in einer Anlage Feldgeräte wie Sensoren und Aktoren zwecks Kommunikation mit einem Steuerungsgerät (SPS).

Durch den Feldbus entstehen einige wesentliche Vor- und Nachteile:

- + geringer Verkabelungsaufwand
- + Erweiterungen oder Änderungen sind einfach
- Komplexität
- Preis
- Aufwendige Messgeräte (Analyzer)
- Längere Reaktionszeit

2.1.1 Spezifische Feldbusse

Im Gebäude sind die gängigen Feldbusse:

- LON
- EIB
- KNX

Innerhalb der Maschine werden andere Feldbusse verwendet:

- CAN/CANOpen
- Profibus, Profinet
- EtherCAT
- Ethernet/IP

3 Programmierung

Definition SPS-Programmierung

Die EN 61131-3 ist die einzige weltweit gültige Norm für Programmiersprachen von SPS-Steuerungen. Sie definiert die folgenden fünf Sprachen:

- AWL - Anweisungsliste (Assembler)
- KOP - Kontaktplan
- FBS - Funktionsbausteinsprache
- ST - Strukturierte Text
- AS - Ablaufsprache

Die einzige (relevante) Programmiersprache hinsichtlich des IAUT Unterrichts ist ST. Die anderen werden für einfache logische Verknüpfungen gebraucht, oder für z.B. I/O-Zuweisungen. ST wird in Europa oft für die Programmierung von SPS gewählt.

3.1 EN 61499

Diese Norm stellt eine objektorientierte Erweiterung der EN61131 dar.

4 Digitale Regelung

5 FSM - Finite State Machine

6 ST-Language

Dieses Kapitel fasst den Aufbau und Syntax der SPS-Programmiersprache ST zusammen.

6.1 Grundaufbau

Der Aufbau eines Programmes/Funktionsblockes oder Funktion ist immer in zwei Bereiche aufgeteilt: Deklaration und Implementation. Im Deklarationsteil werden alle Inputs, Outputs deklariert und ggf. initialisiert. In der Implementation wird der eigentliche Code geschrieben. Wichtig: die Implementation wird in jedem Clock Zyklus wiederholt.

```

1 // Deklarationsteil
2 VAR_INPUT
3     // input-variablen
4     iVarIn AT %I* : INT;
5 END_VAR
6 VAR_OUTPUT
7     // output-variablen
8     iVarOut AT %Q* : INT;
9 END_VAR
10 VAR_IN_OUT
11     // in/out-variablen
12 END_VAR
13 VAR
14     // lokale variablen
15     tmp : INT;
16 END_VAR
17
18 // Implementationsteil
19 IF iVarIn > 10 THEN
20     tmp := 0;
21 ELSE
22     tmp := iVarIn;
23 END_IF

```

6.2 Variablentypen

6.2.1 DUT - Data Unit Type

Über DUT können benutzerdefinierte Datentypen angelegt werden. Die relevanten sind **STRUCT** und **ENUM**. Durch diese Typen kann ein Programm bedeutend lesbarer gestaltet werden.

6.2.2 Programm

Hallo das ist das erste Mal wo ich von Linux aus das hier mache. So far so good, es geht wirklich gut.

6.2.3 Funktionsblock

6.2.4 Funktion

6.3 Variablen-Scope

6.3.1 Globale Variablen

In einer globalen Variablenliste **GVL** werden global verwendete Variablen deklariert. Sinnvoll ist dies zum Beispiel für: SPS Zykluszeit, Ein-/Ausgänge oder sonstige Konstanten, welche überall im Code wieder verwendet und gegebenenfalls ändern.

6.4 Conditional Statements

6.4.1 If/Else

```

1 IF <cond1> THEN
2     ...
3 ELSIF <cond2> THEN
4     ...
5 ELSE
6     ...
7 END_IF

```

6.4.2 Switch-Case

```

1 CASE iVar OF
2     1:
3         // do something for 1
4     2..5:
5         // do something for range 2 - 5
6 ELSE
7     // do this as default-assignment
8 END_CASE;

```

Die Switch-Bedingung kann auch für Enum-Typen gebraucht werden:

```

1 CASE eState OF
2     Init:
3         // handle init state
4     Running:
5         // handle running state
6     Error:
7         // handle error state
8 ELSE
9     // it is beneficial to always have a default
        statement
10 END_CASE;

```

7 TwinCat

TwinCat ist die Entwicklungsumgebung von Beckoff für deren SPS Systeme. TwinCat liefert diverse Funktionen vorgefertigt für den Benutzer. In diesem Kapitel werden die relevantesten aufgelistet.

8 Beispiel Code