# Inhaltsverzeichnis

**SPS / PLC Programmierung** ..... 1
**Sprachelemente** ..... 1
***Struktuierter Text (ST)*** ..... 1
IF ..... 1
FOR ..... 2
CASE ..... 3
WHILE ..... 3
REPEAT ..... 3
RETURN ..... 3
JMP ..... 3
EXIT ..... 3
CONTINUE ..... 4
Aufruf Funktionsbausteine ..... 4
Kommentare ..... 4

# SPS / PLC Programmierung

Eine kurze Zusammenfassung aller Sprachelemente.

Strukturierter Text (ST, auch SCL) ist eine der sechs in IEC 61131-3 festgeschriebenen Programmiersprachen für Automatisierungstechnik. Sie orientiert sich an PASCAL und enthält sowohl Sprachelemente dieser Sprache als auch SPS-typische Elemente. Besonders geeignet ist ST für alle Aufgaben, die sich mit mathematischen Formeln beschreiben lassen, wie die Programmierung komplexer Algorithmen, mathematischer Funktionen und für Rezept- und Datenverwaltung. Solche Programmteile werden mit ST bedeutend vereinfacht.

Typisch für Strukturierten Text sind Anweisungen, die wie in Hochsprachen bedingt (IF..THEN..ELSE) oder in Schleifen (WHILE..DO) ausgeführt werden können. Für SPS-typische Aufgaben wie Timer, Trigger, Counter und RS-FlipFlop kommen auch in ST die Funktionsbausteine der Standardbibliothek zum Einsatz.

---

# Sprachelemente

## Struktuierter Text (ST)

---

### IF

Hier Beckhoff IF

---

ST-Anweisung IF

Die IF-Anweisung verwenden Sie, um eine Bedingung zu prüfen und, abhängig von dieser Bedingung, Anweisungen auszuführen.

Syntax:

IF <boolean expression_1> THEN <IF-instructions> {ELSIF <boolean expression_2> THEN <ELSIF-instruction_1> ELSIF <boolean expression_n> THEN <ELSIF_instruction_n-1> ELSE <ELSE_instructions>} END_IF;

Der Abschnitt innerhalb der geschweiften Klammer {} ist optional.

Wenn <boolean expression_1> TRUE liefert, führt TwinCAT nur die <IF-instructions> und keine der anderen Anweisungen aus.

Ansonsten prüft TwinCAT die booleschen Ausdrücke, beginnend mit <boolean expression_2>,

nacheinander bis ein Ausdruck TRUE liefert. Anschließend wertet TwinCAT alle Anweisungen, die zwischen diesem Ausdruck und vor der nächsten ELSE oder ELSIF -Anweisung stehen, aus und führt sie entsprechend aus.

Wenn keiner der booleschen Ausdrücke TRUE liefert, wertet TwinCAT nur die <ELSE_instructions> aus.

Beispiel:

IF fTemp < 17 THEN

```
  bHeatingOn  := TRUE;
```

ELSIF fTemp > 25 THEN

```
  bOpenWindow := TRUE;
```

ELSE

```
  bHeatingOn  := FALSE;
  bOpenWindow := FALSE;
```

END_IF;

Siehe auch:

```
  ExST-Anweisung CONTINUE
```

—-

## FOR

---

Hier Beckhoff FOR

---

ST-Anweisung FOR

Die FOR-Schleife verwenden Sie, um Anweisungen mit einer bestimmten Anzahl von Wiederholungen auszuführen.

Syntax:

FOR <counter> := <start value> TO <end value> {BY <increment> } DO

```
  <instructions>
```

END_FOR;

Der Abschnitt innerhalb der geschweiften Klammern {} ist optional.

TwinCAT führt die <instructions> solange aus, wie der <counter> nicht größer, oder - bei negativer Schrittgröße increment - kleiner als der <end value> ist. Dies wird vor der Ausführung der <instructions> geprüft.

Immer wenn die Anweisungen <instructions> ausgeführt worden sind, wird der Zähler <counter> automatisch um die Schrittgröße <increment> erhöht. Die Schrittgröße <increment> kann jeden ganzzahligen Wert haben. Wenn Sie keine Schrittgröße angeben, ist die Standard-Schrittgröße 1.

## CASE

## WHILE

## REPEAT

## RETURN

## JMP

## EXIT

## CONTINUE

## Aufruf Funktionsbausteine

## Kommentare

From:
https://jmz-elektronik.ch/dokuwiki/ - **Bücher & Dokumente**

Permanent link:
**https://jmz-elektronik.ch/dokuwiki/doku.php?id=start:sps:programmierung:strukturierter_text&rev=1712313670**

Last update: **2024/04/05 12:41**