# Was ist eine SPS

eine speicherprogrammierbare Steuerung

Prozessor, RAM, Taktgeber, digitale Ein und Ausgänge (neuere haben mehr Funktionen)

+ sehr robust (IP Klassen Gehäuse), zuverlässig, sehr flexibel auch in größeres System mit Feldbussen einbindbar, Lampen direkt ansteuerbar über 24V Ausgänge, echtzeitfähig.  
- relativ teurer (100 Euro),

Alternativen zur SPS:

1. Generische Elektronikplatine:   
   + billig, wenn man sehr viele gleiche Schaltungen bauen würde  
   - Erweiterungen aufwändig, zB um es in eine übergeordnete Ampelsteuerung einzubinden,
2. Mikrokontrollerboard zB Arduino:  
   + billig, einfach zu programmieren, viele Schnittstellen erhältlich zB Funk GSM  
   - Stromversorgung der Lampen müsste extra erfolgen zB mittels Relais, robustes Gehäuse braucht man extra
3. IndustriePC (gibt es auch in kleinen Gehäuse, ist aber übertrieben für diese Aufgabe, Vorteil Netzwerk wäre schon dabei, Nachteil – digital out)

# SPS Programmierung

Ein- und Ausgänge sind einfache Boolesche Werte, also ein oder aus (5V bzw 24V)

Wird in Umgekehrt Polnischer Notation (UPN) programmiert dh man legt boolesche Werte auf den Stack und verknüpft diese miteinander. Logik reagiert immer auf den obersten Stack-Wert dh ein Ausgang wird geschalten wenn dort wo der Schaltbefehl steht, der oberste Stackwert true ist  
zB

LD E0.0  
UN E0.1  
=A0.0

In C würde man sagen   
if(E0.0 && !E0.1)

{  
 A0.0=true;

}

Programmieren der SPS (zB Siemens Simatic):

Programmierung erfolgt mittels Anweisungsliste (AWL) oder Kontaktplan (sieht aus wie LabView, haben wir aber nicht gemacht). Programmierung erfolgt stackbasierend in umgekehrter polnischer Notation: Werte auf Stack legen und am Stack mit logischen Operatoren verknüpfen. Werte sind als einzelne Bits zu sehen, die true oder false sein können. Es gibt Eingänge zb E0.0 bis E0.7 und Ausgänge A0.0 bis A0.5   
zB   
LD E0.0 //Zustand des ersten Einganges auf den Stack laden, Stack hat Höhe 1  
LD E0.1 //Zustand des zweiten Einganges auf den Stack laden, Stack hat Höhe 2  
ULD //oberste Stackwerte mit Und zu einem Verschmelzen, Stack hat wieder Höhe 1  
= A0.0 //Ausgang erhält den Wert des obersten Stacks

Was haben SPSen noch:  
-Timer (wie Eieruhren, man stellt eine Laufzeit ein, nach der wird der Timerausgang true)  
gibt es in unterschiedlicher Genauigkeit von 1ms, 10ms und 100ms  
  
-Variablen:  
V0.0 … Bit 0 der Variablen V0  
VB0 …VariablenByte 0 also V0.0 bis V0.7, damit kann man also auch Zahlen machen  
VW0 … VariablenWord 16 Bit also die Varaiablen VB0 und VB1 zusammen  
VD0 … VariablenDouble 32 Bit VB0 bis VB3 oder VW0+VW1 (für sehr große Zahlen)  
Setzen von Variablen mit MOVB MOVD usw, vergleichen mit LDB=  
  
-Merker  
M0.0 behalten Infos auch nach Stromausfall

FlipFlops:

S A0.0,1 //Schaltet Ausgang A0.0 auf Dauer true, bis er mit R ausgeschaltet wird  
R A0.0,5 //Schaltet 5 Bits ab dem Ausgang A0.0 auf false, also A0.0, A0.1, …, A0.4 sind dann false

AWL läuft in einer Endlosschleife und in Echtzeit

**Ampel Beispiel:**

Ampelsteuerung Ablauf:  
4 Sekunden Rot  
1 Sekunde Rot und Gelb  
5 Sekunden Grün danach 4 mal 0,5 Sekunden grün blinkend  
3 Sekunden Geld

Könnte man auch mit S und R und Timern programmieren in dem man auf die vorhergehenden Zustand reagiert: zB wenn rot gesetzt ist und genug Zeit vergangen ist kommt orange dazu. Das ist aber extrem kompliziert zB beim Grün Blinken, weil man zusätzliche Merker braucht um zu wissen was vorher war.

Intervall: 0,5 Sekunden (kleinstes Intervall)

Rot: 1-10  
Gelb: 9-10 und 29-35  
Grün: 11-20 und bei 22, 24, 26 und 28

### Ausgänge

A0.0…Rot  
A0.1…Gelb  
A0.2…Grün

## Impulsdiagramm (Zeit-Schalt-Diagramm)

## AWL

NW0 //Takt erzeugen  
 LDN T40 //der oberste Stack ist true, wenn der timer noch nicht abgelaufen ist  
 TON T40, 5 //T4ß ist ein 100ms Timer -> 5 bedeutet 500ms = 0.5s

NW1 //Takt zählen  
 LD T40 //T40 wird alle 500 ms true  
 INCB VB0 //wie VB0++

NW2 //Rot  
 LDB>= 1, VB0   
 LDB<= 10, VB0   
 ULD // if(VB0>=1 && VB0<=10)  
 =A0.0

NW3 //gelb  
 LDB>= 9, VB0  
 LDB<= 10, VB0  
 ULD  
 LDB>= 29, VB0  
 LDB<= 30, VB0  
 ULD  
 OLD  
 =A0.1

NW4 //grün  
 LDB>= 11, VB0  
 LDB<= 20, VB0  
 ULD  
 LDB= 22, VB0  
 LDB= 24, VB0  
 LDB= 26, VB0  
 LDB= 28, VB0  
 OLD  
 OLD  
 OLD  
 OLD  
 =A0.2

NW5 //Zähler zurücksetzen  
 LDB= 36, VB0  
 MOVB 0, VBO