1. Erläutern Sie Aufbau und Wirkungsweise einer SPS! Durch welche Kenngrößen ist sie gekennzeichnet?

Aufbau:

- Eingansbaustein, Zentraleinheit, Ausgangsbaustein sind über Bussystem miteinander verbunden
- Steller nur bei geringem Energiebedarf über Steuerung, sonst nicht in SPS
- Zentraleinheit (CPU, Speicher, E/A- Baugruppe, Zeitbausteine)
    - Sequentielle Verarbeitung einer Befehlsreihenfolge
    - Speicher (enthält Befehlsreihenfolge, Daten)
    - Steuerwerk (Befehlsregister, holt Daten entspr. Adresse aus Speicher, Programmzähler (Adresse des nächsten Befehles))
    - Rechenwerk (Ausführen der Operation im Arithmetik Logik Einheit (ALU), Register für schnelle Datenverarbeitung)
- Eingangsbaugruppe (sichere Signalerkennung, Signalanpassung bzw. Signalumwandlung, Schutz der Elektronik vor Spannung von außen, Entstörung der Signale)
- Ausgangsbaugruppe (Spannungsanpassung von Logik auf Steuerspannung, galvanische Trennung zu Anlage, Leistungsverstärkung um wesentliche Stellglieder ansteuern zu können, Kurzschluss- & Überlastschutz (Relais, Transistorausgang))

Funktionsweise:

- zyklisch = ständige Wiederholung des Programms
- vor Bearbeitung wird der Zustand aller Eingänge im Prozessabbild (RAM) abgelegt (Prozessabbild = eigener Speicherbereich)
- am Ende des Zyklus werden Ausgänge aus Prozessabbild ausgegeben

- kurze Eingangssignale ($\leq t_z$) werden eventuell nicht wahrgenommen
- $t_z \leq t_a \leq 2*t_z$ t_a… Antwortzeit
- Befehlreihenfolge entscheidet Verhalten

2. Beschreiben Sie den inneren Aufbau einer SPS. Welche Speicherarten werden wofür in der SPS eingesetzt?

Siehe 1.
Speicherarten:

- ROM (Betriebssystem, nicht flüchtig, nur einmal beschreibbar)
- EPROM (Programm, nicht flüchtig, löschbar durch UV-Licht)
- EEPROM (Programm, nicht flüchtig, elektrisch löschbar)
- RAM (Daten, flüchtig, ev. Pufferung durch Batterie)

3. Beschreiben Sie den Aufbau der Ein- und Ausgangsbaugruppe!

Siehe 1.

Eingangsbaugruppe:
- Fehlerspannungserkennung
- Signalverzögerung (um Störimpulse zu beseitigen)
- Optokoppler (galvanische Trennung von Steuerung und Anlage)

Ausgangsbaugruppe:
- Optokoppler
- Verstärker (zum Ansteuern von wesentlichen Stellgliedern)
- Kurzschlussüberwachung (Stromerkennung über Leistungswiederstand)

4. Welche Bauformen der SPS kennen Sie? Wie sind diese zu charakterisieren? Wo werden sie eingesetzt?

Klassische SPS:

- Kompakt - SPS (Eingangs-, Ausgangsbaugruppe und Zentraleinheit in einem festen Gehäuse (Klein - SPS))
- Modulare SPS (Zusammenstellen nach Bedarf in Baugruppenträger, Verbindung durch inneres Bussystem, schneller Hochlauf und automatischer Wiederanlauf, Tausch von Baugruppen bei laufendem Betrieb)

Slot – SPS:

- Separate Baugruppe am IPC – Bus; hohe Flexibilität des IPC für Programmierung, Visualisierung und Kommunikation; Karte unabhängig von PC... arbeitet weiter bei Systemabsturz

Soft – SPS:

- PC ist in der Lage, sowohl Steuerungslogik als auch MMI mit Visualisierung parallel zu realisieren (Nachteil: mögliche Systemabstürze, langer Hochlauf, kein automatischer Wiederanlauf, keine remanente Speicherung

5. Erläutern Sie am Beispiel der Programmierung des Fertigungssystems die Entwicklungsphasen eines SPS Programms!

Spezifikation:

- Beschreibung mit Struktur und Lageplan

Entwurf:

- Beschreibung nach FUP
- Weg – Schritt – Diagramm, Ablaufbeschreibung
- Entwerfen der Programmmodule
- Schaltplan, Stückliste

Realisierung:

- Ablaufsprache für Ablaufsteuerung
- Einfache Verknüpfungssteuerung
- Steuerungsmoduln (AWL, KOP, FBS)

Inbetriebnahme / Programmtest:

- Test einzelner Module, Verknüpfung dieser Module, Gesamtprogramm, Zeitregime
- Test auf Fehlbedienung, Grenzfälle, Zeit, Sensor- und Aktorausfall

Dokumentation:

- Aufgabenbeschreibung
- Lageplan, Technologieschema
- Schaltplan, Signalplan, Klemmenanschlussplan
- Zuordnungsliste der Ein- & Ausgänge
- Kommentiertes Programm
- Hinweise für Inbetriebnahme und Bedienung
- Fehlerliste

6. Erläutern Sie die Konfigurierungs- und Strukturierungsmöglichkeiten eines SPS-Programms nach IEC 61131-3!

Strukturierung auf Konfigurationsebene

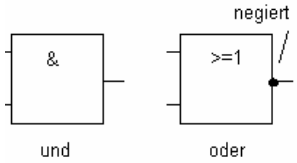
- Konfiguration stellt Automatisierungssystem dar
- Ressource stellt Prozess dar
- Prozess sind ein / mehrere Programme (Task) zugeordnet
- Ressource besitzt Steuerelemente für zeitliche Steuerung der Programme
- Task legt fest, ob Programm zyklisch oder einmalig abläuft

Strukturierung auf Programmebene:

- Programm (äußere Schale) kann von keinem anderen Programmorganisationsbaustein aufgerufen werden
- Funktionsbausteine und Funktionen können von einem Programm oder Funktionsbausteinen aufgerufen werden

7. Nennen und kennzeichnen Sie die Möglichkeiten der Programmierung nach IEC 61131-3!

unterschiedliche Programmierung in ST/AWL/FUP/KOP und Taskkonzepte: zyklischer Ablauf oder Interrupt

Programmierart	**Bausteine**	**Vorteile**	**Nachteile**
KOP: Kontaktplan [Verknüpfungsprogr.]	-[ ]- Schließer -[/]- Öffner -( )- Merker/Ausgang	Günstig für Entwickler von konventionellen Relais- und Schützensteuerungen	Aus Plan/Listing schwierig die Wirkungsweise der Steuerung zu erkennen, Problematische Realisierung von Funktionsbausteinen
FUP: Funktionsplan [Verknüpfungsprogr.]	Logik-, Speicher-, Zähl-, Zeitbausteinen 	Direkte Umsetzung der logischen Zusammenhänge (Bool), Bausteindarstellung möglich, direkte Übereinstimmung mit AWL	Wirkungsweise der Steuerung nicht erkennbar, aufwendige Ablaufsteuerung, ermöglicht nur einfache und/oder Verknüpfungen
AWL: Anweisungsliste [Verknüpfungsprogr.]	Log. Verknüpfungen, nur eine Anweisung/Zeile Form: [Marke]-[Operator]-[Operant]-[Kommentar] *Z1 L A1 <Lampe ein>*	Direkte Umsetzung der Boolschen Algebra, Programmierung des Programmablaufes möglich	Wirkungsweise der Steuerung nicht erkennbar, Ablaufsteuerung aufwendig zu programmieren
AS: Ablaufsprache (Ablaufprogrammierung)	Aufteilen des Prozesses in Prozessschritte und Transitionen Weiterschaltung nur, wenn Vorhergehender Prozessschritt eingeschaltet, nachfolgender Schritt nicht eingeschaltet und Transition erfüllt ist, Übergangsbedingung ist gleichzeitig Endebedingung (Probl. mit Alternativ/Parallelverzw)s	2 Ebenen: <u>Graph:</u> Prozessschritte und Transition <u>Lupenebene:</u> Beschreibg. Der logischen Zusammenhänge (über AWL,FUT,KOP)	Vorteile: Trennung von Prozesszustand und Transition Nachteile: Sprünge und Schleifen möglich
ST: Strukturierter Text	Pascalähnl. Hochsprache, Ausdruck: Konstrukt, der nach seiner Abarbeitung einen Wert zurückliefert IF....THEN....ELSE mögl.	Gute Strukturierungsmöglichkeiten, verständlicher als reiner Bool, mehr Freiraum bei Variblennamen, geeignet für komplexe Steuerung	Keine reinlogische Steuerung. Komplexer

8. Mit welchen Datentypen und Grundbefehlsarten wird bei der SPS-Programmierung gearbeitet?

a) Datentypen:

Bool: Boolsche Logische Zahl, welche 0 oder 1 sein kann
SINT: ShortInteger, kurze ganze Zahl zw. 0-255
INT: Integer, ganze Zahl zwischen -2^{15} und $2^{15}-1$ (wegen der 0)
DINT: DoubleInteger, ganze Zahl zwischen -2^{31} und $2^{31}-1$ (wegen der 0)
Real: Gleitpunktzahl zwischen 2.9E-39 und 2.9E38
TIME: Zeitdauer, je nach Angabe in ms, s...
String: Zeichenfolge
Byte: 8 Bit Bitfolge (kein Wert: nur Folge von Einzelbits)
Word: 16 Bit Bitfolge

Variabelendeklaration mit:

```
VAR_XXXXX
    Druck: INT
END_VAR;
```

b) Grundbefehlsarten:

AND	&	und
OR	>=1	oder
XOR	=2k+1	
NOT	0	nicht (als Zusatz)
GT	>	größer
GE	>=	größer oder gleich
EQ	=	gleich
NE	<>	ungleich
ADD	+	addition
MUL	*	multiplikation
SUB	-	subtraktion
DIV	/	division
MOVE	:=	Eingangswert auf Ausgangswert zuweisen

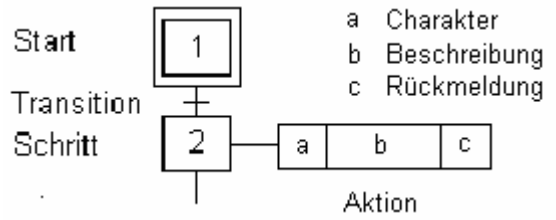
9. Erläuterns Sie Aufbau und Einsatzmöglichkeiten der Ablaufsprache zur SPS-Programmierung und vergleichen Sie die Bestandteile mit der AWL-Programmierung am Beispiel des Fertigungssystems!

Ablaufsprache: ist wesentliche Programmierwerkszeug der IEC 61131
Dient der Gliederung und Strukturierung von Programmen
Hilfe um logische Zusammenhänge zu erfassen: davon einfach in AWL umschreibbar
Grafische oder tabellarische Verdeutlichung/Visualisierung von SPS-Programmen

Tabellarisch::

Schritt	*Name*	*Anf.-Bed.*	*Aktion*	*End-Bed.*	*Charakt.*

Grafisch:



Bestandteile: Schritt: Darstellung von Aktionsreihenfolge der Anlage
Nur hier Setzen / Rücksetzen von Ausgängen
Kann in Reihenfolge aktiv oder passiv sein
bestehend aus Aktion und Flag (Anz., ob Schritt aktiv)
Falls Flag aktiv: Aktion einmal pro Zyklus ausgeführt
Jedem Schritt werden Aktionen zugeordnet:

Schrittname

Aktion: legt fest, welche Befehle ausgeführt werden
innerhalb des Funktionsblocks formuliert, der gleich dem Ablaufplan ist > Instruktion in AWL

Befehle: verschiedene Ausführungscharakt.: n...nicht gespeichert (aktion nur so lange, wie schritt aktiv), s...gespeichert

Initialschritt: erster Schritt in Ablaufsprache bzw. erster aktiver Schritt

Transition: Bedingung unter der von einem Schritt auf einen oder mehrer folgender Schritte abgegeben wird
Bedingung bestehend aus Boolsche Bedingung

Struktur: Parallelverzweig: jede PV beginnt und endet mit Schritt
Wenn PV-vorliegender Schritt=1 > beide ersten PV-Schritte=1
Parallele Abarbeitung der Schritte in den Zweigen
Schritt nach PV aktiv, wenn alle Schritte in den PV-Zweigen abgearbeitet und Transition vor Schritt 1 ist

Alternativverzw: jeder Alternativzweig beginnt und endet mit Tansition
Der Schritt wo Transition zuerst 1 ist, wird abgearbeitet

Sprung: Verbindung zu Schritt unter Sprungsymbols

AWL: textuelle assemblerähnliche Programmiersprache: Folge von Anweisungen, je Eine/Zeile
Dient zur rein-logischen Verknüpfung von Steuerungsein- und –ausgängen
Ein Operator hat nur einen Operant
Geringe Strukturierungsmöglichkeiten

Anweisung bestehend aus:

Marke	Operator	Operand	Kommentar
Start	*LD*	*%IX1.2*	*(Teil da)*
Z1	*L*	*SM1*	*(Schrittmerker1)*

Operatoren haben keine Priorität: Reihenfolge kann durch Klammerung geändert werden
Nur noch für einfachere Steuerungen verwendet

10. Kennzeichnen Sie die Programmierart „Strukturierter Text“ nach IEC 61131-3 und leiten Sie den Einsatzbereich daraus ab!

Pascal-ähnliche Hochsprache, welche eine höhere Strukturierung von Anweisungen ermöglicht, als AWL und daher AWL mehr und mehr ablöst
Variablennamen besser deklarierbar, nicht Reinlogische Programmiersprache
Ausdruck: Konstrukt, welches nach seiner Auswertung einen Wert zurückliefert

- bestehend aus Operatoren (+,-,MOD,OR) und Operanten (Variable, Zahl, Ausdruck)
- Auswertung durch Abarbeitung der Operatoren nach Priorität, von hoher zu niedriger
  Hohe Priorität: Klammern/Funktionsaufruf Niedrige: AND, XOR, OR

Beispiel:
```
IF (MASCHINE_EINGESCHALTET = TRUE) THEN
    SOLLPOSITION := SOLLPOSITION + 100;
    AUSGANG1 := EINGANG1 AND EINGANG2;
ELSE
    AUSGANG1 := FALSE;
END_IF;
```

11. Charakterisieren Sie mögliche Betriebsarten. Wie sind diese zu realisieren?

Realisierung allgemein für mehrere Betriebsarten:
Im übergeordneten Programmbaustein (ORG) werden die einzelnen Betriebsarten ermittelt und je ein individueller Merker gesetzt (M_Schritt, M_Tipp).
Einführen eines Aktionsfreigabemerkers, der Voraussetzung zum Sprung zu nächsten Schritt ist
Ebenfalls im ORG werden die einzelnen Betriebsarten wie folgt definiert:

Automatik: Selbstständiger Ablauf des Prozesses mit Wiederholung
Realisierung:
Schrittmerker für jeden Schritt einführen: Schrittmerker wirkt als Anker
Nach Schrittende wird aktiver Schrittmerker rückgesetzt und nächster gesetzt
Definierte Sprünge durch eindeutige Schrittmerker für Schritte möglich
Wiederholung durch Setzen des ersten Schrittmerkers im letzten Schritt

(Aktionsfreigabe wird in ORG immer wieder gesetzt, sodass auf Rücksetzen des Aktionsfreigabemerkers sofort das Setzen folgt)

Zyklus: Selbsständiger Ablauf des Prozesses mit Halt am Prozessende
Realisierung:
Wie Automatikbetrieb, jedoch ohne Setzen des ersten Schrittmerkers im letzten Schritt > sondern setzen auf Schritt 0: Warten auf START

Schritt: Ablauf eines Prozessschrittes nach START und Anhalten nach Prozessschritt
Realisierung:
Einführen eines globalen Aktionsfreigabemerker in jedem Teilprozess
In ORG: Setzen des Aktionsfreigabemerkers nur bei Drücken von START
Rücksetzen des Freigabemerkers nach jedem Schritt

Tipp: Ablauf des Prozesses, solange Starttaste gedrückt, weiterfahren über Schrittende hinaus
Realisierung:
Aktionsfreigabemerker wird bei Drücken der Taste „START“ auf 1 gesetzt, sonst immer auf 0 rückgesetzt (global im ORG)

12. Durch welche Maßnahme kann in der SPS der Prozesszustand eindeutig verfolgt werden? Was ist dabei nach einer NOT-AUS-Situation zu berücksichtigen?

Individuelle Merker für jeden Prozessschritt können eingeführt werden. Es muss dafür gesorgt werden, dass jeweils nur ein Schrittmerkeraktiv ist. D.h. es muss am Schrittende der aktive Schrittmerker rückgesetzt und der folgende Schrittmerker gesetzt werden. Die Betriebszustände können auch durch Merker festgelegt werden.
Bei NOT-AUS liegt im Allgemeinen eine Gefahrensituation vor. Daher muss die gesamte Anlage manuell geräumt werden. Da das NOT-AUS-Signal jederzeit ausgelöst werden kann, ist es nötig, um einen sicheren Startpunkt für die gesamte Anlage finden zu können, die Anlage neu zu RICHTEN. Dabei müssen zunächst sämtliche Aktionsfreigabemerker, Schrittmerker und Betriebsartenmerker (außer einem Richtenmerker)auf 0 gesetzt werden, die Readysignale für folgende Stationen rückgesetzt werden, Anzeigen gelöscht und die Anlage auf einen definierten Anfangszustand gefahren werden. Erst danach kann mittels START der gesamte Prozess vom Prozessanfang anlaufen.

13. Wie ist HALT und NOT-AUS definiert und wie können Sie mittels einer SPS realisiert werden?

HALT: bei Automatik: anhalten des Prozesses am Zyklusende
Bei Zyklus: anhalten am Schrittende
Schritt/Tipp: ohne Wirkung, da sowieso Befehl „Start“ benötigt
Realisierung:
Im Übergeordneten Programmbaustein (ORG) wird, bei Signal HALT, ein „Halt-Merker“ gesetzt, der den Prozessablauf definiert:
In „Betrieb Halt“ wird der im Ablauf eingebundene globale Aktionsfreigabemerker auf 0 gesetzt, Setzen des Aktionsfreigabemerkers auf 1 durch Drücken von START mit Gleichzeitigem Rücksetzen des „Halt-Merkers“

NOT-AUS: NOT-AUS ist negiertes Signal: immer „1“: wenn NOT-AUS, dann „0“
sofortiges Anhalten des Prozesses (Aktoren drucklos, Motoren aus)
Beseitigung der Gefährdung (manuell!)
Richten unbedingt erforderlich
Start mit Neuanlauf
Realisierung:
Wenn NOT-AUS Signal 0 ist, dann Stopp-Merker im übergeordneten Baustein setzen mit Stoppmerker: Rücksetzen des Aktionsfreigabemerkers und Abarbeiten der NOT-AUS Routine: Abschalten von Motoren
Weiterführen des Prozesses nur nach Richten (erreichen der Ausgangsstellung) möglich