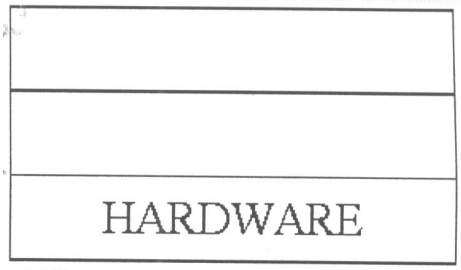
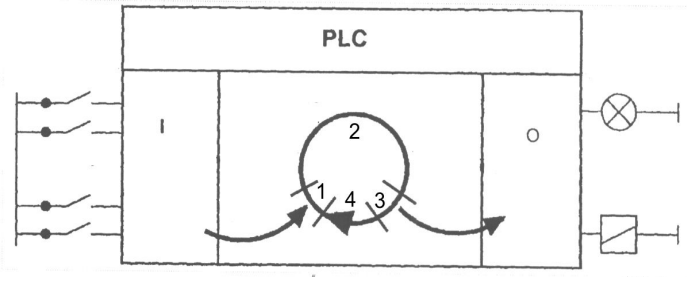
23. Software a programování PA

Vrstvy software	Princip práce PA
	

Vrstvy software:

↔	**Firmware**	↔	#1..... **SW -** má funkci jako OS u PC - spouštění uživatelského programu, obsluha vstupů a výstupů, obsluha chyb, správa paměti
		↔	je závislý na HW (každý výrobce má svůj), výrobce poskytuje pravidelný #2.....
↔	**Programy**	↔	**aplikace - uživatelský SW** – programy pro konkrétní řízení pomocí PA - jsou vytvořené #3.....

- Pozn. **SoftPLC** - software pro #4..... PA na PC (emulace PA - virtuální PA - PC se "tváří" jako PA), slouží k vývoji, ověření funkčnosti a výuce aplikačních programů pro PA

Princip práce PA:

↔	řídící program je zpracováván #5..... („pořád dokola“)
↔	**Perioda programu =** #6..... jednoho pracovního cyklu – řádově v ms (záleží na délce programu, rychlosti CPU)

V každém cyklu probíhá:

1.	**Načtení** aktuálních #7..... **hodnot** do paměti vstupů - analogové a digitální vstupní signály poskytují snímače, tlačítka, spínače
2.	#8..... **hodnot** v paměti podle aktuálního programu
3.	**Nastavení** #9..... (řídících) **hodnot** v paměti výstupů - analogové a digitální výstupní signály pro signální prvky, stykače, elektromagnetické ventily, které řídí akční prvky (motory, válce)
4.	#10..... **PLC** - kontrola chyb, také signál pro obvod #11..... (kontrolující běh programu - při "zaseknutí" programu resetuje PA) + komunikace s ostatními zařízeními

23.1. Programování PA

=	**tvorba** uživatelských **programů** pro požadovanou funkci PA
↔	liší se podle výrobce PA – způsob výrobce popisuje v dokumentaci
↔	programování probíhá většinou na **PC** s **vývojovým softwarem/prostředím** jako tzv. #12..... - např. Siemens Step 7 (TIA portal)

Fáze tvorby programu

1.	**Rozbor zadání** (#13.....)	↔	technologické schéma - náčrt, výkres - fyzické uspořádání řízeného systému
		↔	schéma řízení - vývojový diagram - #14..... řízení
		↔	přiřazení vstupů a výstupů na PA konkrétním zařízením (zapojení snímačů, motorů)

2.	#15..... **programu**	↔	v editoru ve formě grafického nebo textového **programovacího jazyka** (viz další kapitola) + tvorba **dokumentace** programu
3.	**Překlad** (#16.....)	↔	z programovacího jazyka do strojového kódu procesoru PA - vznikne spustitelný kód
4.	**Ladění** (#17.....)	↔	**kontrola výpisu** kritických chyb (#18.....) a varování (#19.....)
		↔	**odstranění** chyb
		↔	#20..... - kontrola správné funkce programu na monitoru PC
5.	#21..... **kódu**	↔	PA musí být se stavu #22..... (viz LED na PA)
		↔	**load** (download/upload) - nahrání přeloženého kódu do operační paměti PA - přes LAN, USB, flash disk
6.	**Testování a** #23.....	↔	po zapojení vstupů a výstupů se PA uvede do stavu #24.....

23.2. Programovací jazyky

↔	jazyk si volí programátor podle typu úlohy a podle svých zkušeností a dovedností

Rozdělení:

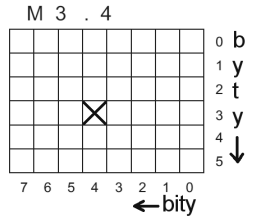
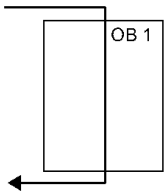
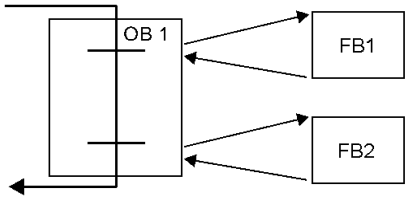
↔	#25..... **programovací jazyky**	↔	program tvoří posloupnost **textových příkazů** (instrukcí - významových slov)		
		↔	rozdělují se na jednodušší **seznam příkazů** (STL - statement list) a **vyšší programovací jazyky** (C, #26.....)		
↔	#27..... **programovací jazyky**	↔	program tvoří posloupnost **grafických symbolů** – jsou přehlednější, druhy:		
			↔	**příčkový diagram** (#28.....)	
			↔	**funkční bloky** (FBD - Function Block Diagram)	
			↔	**sekvenční schéma** (SFC - Sequential Function Chart, #29.....)	

23.2.1. Základní typy proměnných v PA Siemens S7-1200

- **Bit** je #30..... jednotka informace - může mít hodnotu 0 nebo 1 (stav binárního signálu)
- **Byte** je skupina #31... bitů

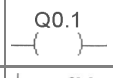
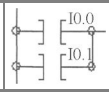
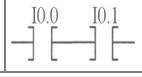
Proměnná	Význam
Ix.y	Hodnota binárního #32....., kde x.y je adresa vstupu - x je pořadové číslo bytu a y je pořadové číslo bitu (počítá se od 0) - např. I0.0 je první vstup (první bit prvního bytu)
Qx.y	Hodnota binárního #33..... - např. Q0.5 je šestý vstup (šestý bit prvního bytu)
Mx.y	Pomocné proměnné (tzv. #34.....) pro ukládání mezivýsledků

- Místo číselných adres lze používat i **slovní jména** (PLC #35.....) pro lepší orientaci v programu - např. Start, Stop

Proměnná v paměti	Hlavní program	Program s podprogramy
		

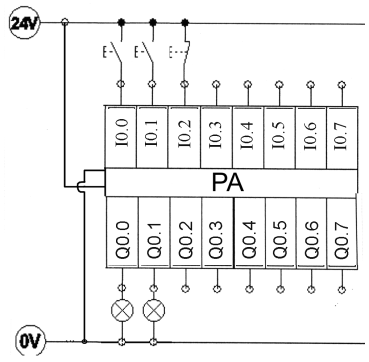
- Program musí obsahovat **hlavní organizační blok** (main OB), který se spouští jako první (z něho se mohou spouštět další bloky - podprogramy)

23.2.2. Příčkový diagram (LAD - Ladder diagram)

↔	program je tvořen posloupností příček #36..... (ladder) - také se říká žebříčkový diagram, příp. kontaktní nebo liniové schéma	
↔	diagram vychází z kontaktního elektrotechnického schématu zapojení vodičů, spínačů a řízeného "spotřebiče":	
↔	vlevo a vpravo jsou #37..... **sběrnice**	
↔	mezi nimi jsou vlevo #38..... v **hranatých** závorkách (spínací NO kontakty)	
↔	nebo rozpínací NC kontakty (mají #39..... hodnoty)	
↔	vpravo jsou **výstupy** v #40..... závorkách (akční prvky)	
↔	spínače svázané **OR** se kreslí #41.....	
↔	spínače svázané **AND** se kreslí #42..... (za sebou)	

Příklady programů:

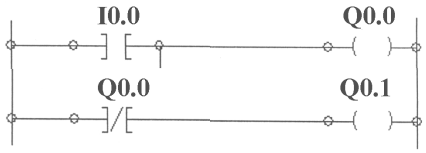
Zapojení vstupů a výstupů



Program 1.1a pro rozsvícení žárovky jedním tlačítkem:

I0.0	Q0.0
0	
1	

Program 1.1b pro negované svícení žárovek:



I0.0	Q0.0	Q0.1
0		
1		

Program 1.2 pro rozsvícení žárovky dvěma tlačítky:

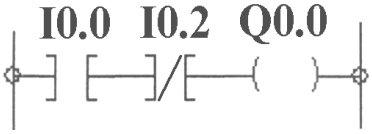
Zapojení s funkcí OR (paralelní)

I0.0	I0.1	Q0.0

Zapojení s funkcí AND (sériové)

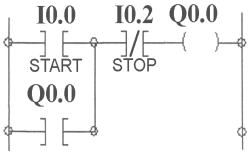
I0.0	I0.1	Q0.1

Program 1.3 pro zhasnutí žárovky rozpínacím kontaktem I0.2



I0.0	I0.2	Q0.0

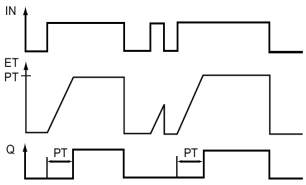
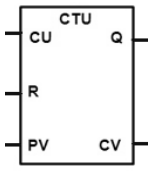
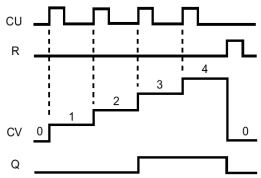
Program 1.4 pro svícení žárovky i po uvolnění tlačítka I0.0



23.2.3. Funkční bloky

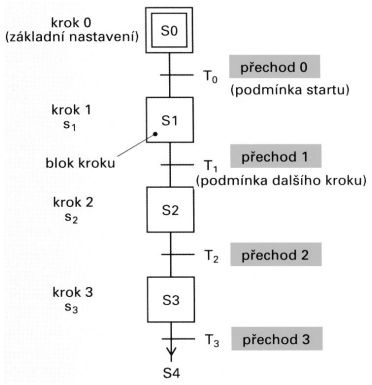
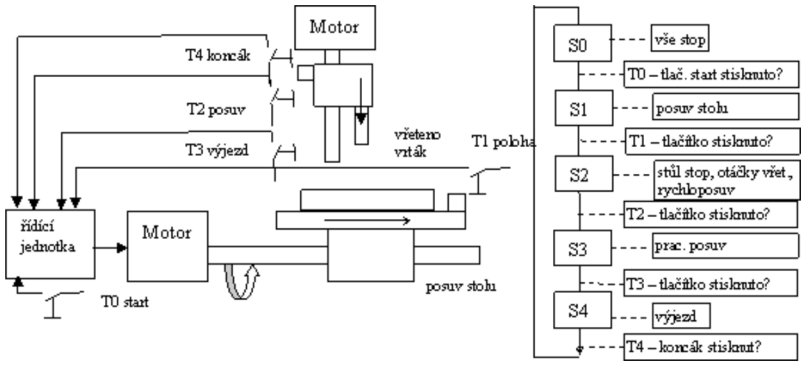
↔	obdélníkový **blok** symbolizuje určitou složitější **funkci** - např. čítač nebo #43.....
↔	zleva bloku jsou **vstupy**, vpravo #44.....
↔	programátor může naprogramovat nové funkční bloky podle potřeby

Příklady funkčních bloků:

Časovač TON (timer on delay)		**Čítač CTU (counter up)**	
			
Časovač se spustí po zapnutí IN a po nastaveném čase PT (preset time) nastaví výstup Q na 1 (ET je uběhnutý čas)		Každý náběh CU navýší aktuální počet CV, po dosažení PV se Q zapne, R slouží k vynulování CV	

23.2.4. Sekvenční schéma

↔	používá se pro grafický popis **časové posloupnosti** jednotlivých akcí v rámci sekvenčního řízení strojů - také Sequential Function Chart, GRAFCET, GRAPH
↔	je tvořeno svislou #45..... kroků (posloupností - steps) - graficky v obdélnících
↔	přechod na následující krok (transition) je podmíněn splněním #46..... - graficky vodorovnými čarami

Sekvenční schéma obecně	**Příklad řízení vrtačky**
	

Opakování software a programování PA

1	nejnižší vrstva software u PA (systémový software)	
2	software pro simulaci PA na PC	
3	doba jednoho pracovního cyklu programu u PA	
4	ladění programu anglicky (odstraňování chyb)	
5	překlad programu z programovacího jazyka do strojového kódu u PA	
6	kontaktní schéma pro programování PA připomíná žebřík neboli anglicky	
7	pro vstupy se v kontaktním schématu používají závorky	
8	sekvenční schéma pro PA je tvořeno sekvencí kroků neboli jejich	
9	u sekvenčního schématu pro PA je přechod na následující krok podmíněn splněním	
10	poslední čtvrtá fáze pracovního cyklu programu u PA	