GFK-0467L-C

New In Stock!

GE Fanuc Manuals

http://www.pdfsupply.com/automation/ge-fanuc-manuals/series-90-30-9030/GFK-0467L-C

series-90-30-9030

1-919-535-3180

Programovateln r dic systmy

www.pdfsupply.com

Email: sales@pdfsupply.com

GFK-0467L-C

New In Stock!

~~GE Fanuc Manuals~~

series-90-30-9030

1-919-535-3180

Programovateln r dic systmy

GE Fanuc Automation

Programovatelné řídicí systémy

PLC Series 90™-30/20/Micro Instrukční sada CPU

Referenční příručka

GFK-0467L-CZ *červen 1999*

Výstrahy, upozornění a poznámky tak jak jsou používané v této publikaci

Výstraha

Výstražná upozornění se v této publikaci používají ke zdůraznění nebezpečného napětí, proudu, teploty nebo jiných stavů vyskytujících se na tomto zařízení nebo jiných stavů, které by mohly být spojené s jeho používáním a které by mohly způsobit zranění osob.

V situacích, kde by nepozornost mohla způsobit buď zranění osob nebo poškození zařízení, se používá Výstražné upozornění.

Upozornění

Upozornění se používají tam, kde by mohlo dojít k poškození zařízení, pokud by obsluha nedávala pozor.

Poznámka

Poznámky pouze upozorňují na informace, které jsou důležité zejména pro pochopení a obsluhu zařízení.

Tento manuál popisuje činnost systému, zpracování chyb a programovací instrukce Logicmaster 90™ pro programovatelné automaty Series 90™-30, Series 90-20 and Series 90 Micro. PLC Series 90-30 PLCs, Series 90-20 a PLC Series 90 Micro patří do řady programovatelných automatů GE Fanuc Automation Series 90.

Revize tohoto manuálu

- CPU model 364 (verze 9.0 a pozdější) podporuje připojení k síti Ethernet přes jeden ze dvou vestavěných Ethernet portů. Je možno použít porty AAUI a 10BaseT. CPU model 364 je jediné CPU Series 90-30, které podporuje Ethernetová Globální Data (strana 2-39).
- Ruční programovací zařízení neumožňuje změnit čas denních hodin, pokud ochrana klíčkem bude aktivní (strana 2-14).
- Do kapitoly 2 byly přidány hodnoty s vlivem na cyklus pro konfiguraci DSM.
- Pro CPU352 byly stanoveny rozdíly v činnost pro logaritmické/exponenciální funkce (strana 6-12) a funkci celé části (strana 11-11).
- Byla provedena změna popisu sekvenčního záznamníku událostí (SER) k objasnění možností této funkce. Detailní informace o časování pro tuto funkci jsou v Dodatku A. Tuto funkci je možno použít u CPU řady 35x a 36x verze 9.0 a pozdější (strana 12-8).
- Pro CPU řady 35x a 36x verze 9.0 a pozdější (strana 12-63) je možno použít nový požadavek na službu Přeskočení dalšího výstupu a čtení vstupu (SVCREQ #45).
- Byly opraveny seznamy bloku parametrů pro funkce čtení a zápisu požadavku na službu Přístup ke stavu rychlé vnitřní sběrnice (SVCREQ #46) (strana 12-64).
- Ostatní opravy a vysvětlení podle potřeby.

Obsah tohoto manuálu

Kapitola 1 Úvod: uvádí přehled systémů PLC Series 90-30, PLC Series 90-20 PLC a PLC Series 90 Micro a instrukční sady Series 90-30/20/Micro.

Kapitola 2 Činnost systému: popisuje určité systémové činnosti systémů PLC Series 90-30 PLC, PLC Series 90-20 nebo Series 90 Micro. To zahrnuje výklad sekvencí systémového cyklu PLC, sekvence zapínání a vypínání napájení systému, hodiny a časovače, bezpečnost, I/O a zpracování chyb. Také obsahuje všeobecné informace pro základní pochopení programování žebříkové logiky.

Kapitola 3 Výklad a oprava chyb: uvádí informace týkající se lokalizace chyb pro systémy PLC Series 90-30, 90-20 nebo Micro. Vysvětluje popisy chyb v tabulce chyb PLC a kategorie chyb v tabulce chyb I/O.

Kapitoly 4-12. Instruční sady Series 90-30/20/Micro: popisují programovací instrukce, které je možno použít u PLC Series 90-30 PLCs, Series 90-20 PLCs a Series 90 Micro. Tyto kapitoly odpovídají hlavním funkčním programovým skupinám.

Dodatek A. Časování instrukcí: uvádí velikost paměti v bajtech a dobu vykonávání v mikrosekundách pro jednotlivé programovací instrukce. Velikost paměti je počet bajtů, který funkce vyžaduje v aplikačním programu žebříkové logiky.

Dodatek B. Interpretace tabulek chyb:popisuje, jak interpretovat formát struktury hlášení při čtení tabulek chyb pomocí softwaru Logicmaster 90-30/20/Micro.

Dodatek C. Mnemotechnické zkratky instrukcí: uvádí mnemotechnické zkratky, které je možno zapsat k zobrazení programovacích instrukcích během procházení nebo editování programu.

Dodatek D. Funkce tlačítek: uvádí speciální klávesnicová přiřazení používaná v softwaru Logicmaster 90-30/20/Micro. Za tímto dodatkem následuje ***volná informativní karta***, kde jsou uvedené funkce tlačítek a mnemotechnické zkratky.

Dodatek E. Používání čísel s pohyblivou desetinnou čárkou: popisuje zvláštní úvahy při používání matematických operací s plovoucí desetinnou tečkou.

Související publikace

Návod pro použití programovacího softwaru Logicmaster™ 90 Series 90™-30/20/Micro (GFK-0466).

Důležité informace k produktu Logicmaster™ 90 Series 90-30 a 90-20 (GFK-0468).

Návod pro instalaci programovatelného automatu Series 90™-30 (GFK-0356).

Návod pro instalaci programovatelného automatu Series 90™-20 (GFK-0551).

Manuál specifikace I/O modulu Series 90™-30 (GFK-0898).

Návod pro použití modulu programovacího koprocesoru a podpůrného softwaru Series 90™ (GFK-0255).

Návod pro použití PCM vývojového softwaru (PCOP) Series 90™ PCM (GFK-0487).

Návod pro použití alfanumerického zobrazovacího systému CIMPLICITY™ 90-ADS (GFK-0499).

Referenční příručka alfanumerického zobrazovacího systému CIMPLICITY™ 90-ADS (GFK-0641).

Technické údaje modulu koprocesoru alfanumerického displeje (GFK-0521)

Návod pro použití ručního programovacího zařízení PLC Series 90™-30 a 90-20 (GFK-0402).

Návod pro použití Power Mate APM pro PLC Series 90™-30 - Standardní režim (GFK-0840).

Návod pro použití Power Mate APM pro PLC Series 90™-30 - Uživatelská příručka pro vlečný režim (GFK-0781).

Návod pro použití Motion Mate™ DSM302 pro PLC Series 90™-30 (GFK-1464)

Návod pro použití vysokorychlostního čítače Series 90™-30 (GFK-0293).

Návod pro použití Series 90™-30 komunikačního modulu Genius (GFK-0412).

Technické údaje komunikačního modulu Genius (GFK-0272).

Návod pro použití Series 90™-30 řadiče sběrnice Genius™ (GFK-1034).

Návod pro použití řadiče sběrnice FIP Series 90™-70 (GFK-1038).

Návod pro použití FIP vzdáleného I/O Scanneru Series 90™-30 (GFK-1037).

Návod pro použití jednotky rozhraní sběrnice distribuovaného I/O a řídicího systému Field Control™ Genius™ (GFK-0825).

Návod pro použití programovatelného automatu Series 90™ Micro GFK-1065).

Návod pro použití sériové komunikace PLC Series 90™ (GFK-0582).

Budeme rádi za vaše připomínky a návrhy

V GE Fanuc Automation usilujeme o to, abychom vytvořili kvalitní technickou dokumentaci. Po přečtení tohoto manuálu věnujte prosím pár okamžiků na vyplnění a odeslání formuláře připomínek čtenáře, který je na následující stránce.

Libby Allen
Sr. Vedoucí technický redaktor

Kapitola 1 Úvod

PLC Series 90-30, 90-20, a Micro patří do řady programovatelných automatů (PLC) řady GE Fanuc Series 90. Lze je snadno instalovat a nakonfigurovat, mají moderní programovací funkce a jsou kompatibilní s PLC Series 90-70.

PLC Series 90-20 PLC představuje cenově dostupnou platformu pro aplikace s malým počtem I/O. Hlavní cíle PLC Series 90-20 PLC jsou následující:

- Nabídnout malé PLC, které je možno snadno používat, instalovat, modernizovat a udržovat.
- Nabídnout cenově dostupné PLC kompatibilní s ostatními PLC.
- Nabídnout snazší systémovou integraci pomocí standardního komunikačního hardwaru a protokolů.

PLC Series 90 Micro také představuje cenově dostupnou platformu pro palikace s menším počtem I/O. Hlavní cíle Micro PLC jsou stejné jako pro PLC Series 90-20. Kromě toho Micro nabízí následující:

- PLC Micro má CPU, napájení, všechny vstupy a výstupy integrované do jediného malého zařízení.
- Většina modelů má také vysokorychlostní čítač.
- Protože jsou CPU, napájení, vstupy a výstupy integrované v jediném zařízení, je konfigurace snadná.

Struktura software pro PLC Series 90-30 a Series 90-20 PLC typu 341 a nižší používá architekturu, která spravuje paměť a priority vykonávání pomocí mikroprocesoru 80188. PLC 90-30 typu 35x a 36x používají mikroprocesor 80386EX. PLC Series 90 Micro PLC používá mikroprocesor H8. Tato operace podporuje jak vykonávání programu tak i základní úlohy vnitřní správy, jako například diagnostické rutiny, snímače vstupů/výstupů a zpracování alarmů. Systémový software také obsahuje rutiny pro komunikaci s programátorem. Tyto rutiny zajišťují výstup i načtení aplikačních programů, předání stavových informací a řízení PLC.

V PLC Series 90-30 PLC je aplikační program (uživatelská logika), který řídí koncový proces, na který PLC použito, je řízený jednoúčelovým koprocesorem pro řazení instrukcí (Instruction Sequencer Coprocessor - ISCP). ISCP je implementovaný v hardwaru modelu 313 a vyšší a v softwaru systémů typu 311 a v PLC Micro. Mikroprocesor 80188 a ISCP mohou pracovat současně a tak

mikroprocesor může obsluhovat komunikace, zatímco ISCP vykonává hlavní část aplikačního programu; mikroprocesor však musí vykonávat ne-booleovské funkční bloky.

Chyby se u PLC Series 90-30, PLC Series 90-20 a PLC Micro objeví, když nastanou určité poruchy nebo stavy, které budou mít vliv na činnost a výkon systému. Tyto stavy mohou ovlivnit schopnost PLC řídit stroj nebo proces. Jiné stavy mohou působit pouze jako výstraha, jako například signál nízkého napětí baterie jako indikace, že napětí baterie chránící paměť je nízké a je nutno ji vyměnit. Stav nebo porucha se nazývá chyba.

Chyby zpracovává softwarová funkce alarmového procesoru, která poruchy zaznamená buď v tabulce chyb PLC nebo tabulce chyb I/O. (CPU model 331 a vyšší s chybou uvede také časový údaj.) Tyto tabulky je možno zobrazit pomocí programovacího software na obrazovce v Tabulce chyb PLC a v Tabulce chyb I/O v softwaru Logicmaster 90-30/20/Micro použitím řídicích a stavových funkcí.

Poznámka

Funkce plovoucí desetinné tečky podporují ***pouze*** CPU řady 35x a 36x, verze 9 nebo pozdější a všechny verze CPU352.

CPU model 364 (verze 9.10 nebo pozdější) je jediné CPU ze Series 90-30, které podporuje EGD.

Poznámka

Další informace najdete v přílohách na konci tohoto manuálu.

- Příloha A uvádí velikost paměti v bajtech a dobu vykonávání v mikrosekundách pro jednotlivé programovací instrukce.
- Příloha B popisuje, jak interpretovat formát struktury hlášení při čtení tabulek chyb PLC a I/O.
- Příloha C uvádí mnemotechnické zkratky instrukcí pro prohledávání nebo editování programu.
- Příloha D uvádí speciální klávesnicová přiřazení používaná v softwaru Logicmaster 90-30/20/Micro.
- Příloha E popisuje použití matematických operací s plovoucí desetinnou tečkou.

Kapitola 2

Činnost systému

Tato kapitole popisuje některé systémové operace PLC systémů Series 90-30, 90-20 a Micro. Mezi tyto systémové operace patří:

Část 1: Souhrn sekvencí cyklu PLC

Logický program se v PLC Series 90-30, 90-20 a Micro bude vykonávat opakovaně, dokud nebude zastavený povelem programovacího zařízení nebo povelem z jiného zařízení. Sekvence činností potřebná k jednorázovému vykonání programu se nazývá cyklus. Kromě vykonání logického programu cyklus zahrnuje načtení dat ze vstupního zařízení, vyslání dat do výstupního zařízení, provedení interní správy, obsluhu programovacího zařízení a obsluhu ostatních komunikací.

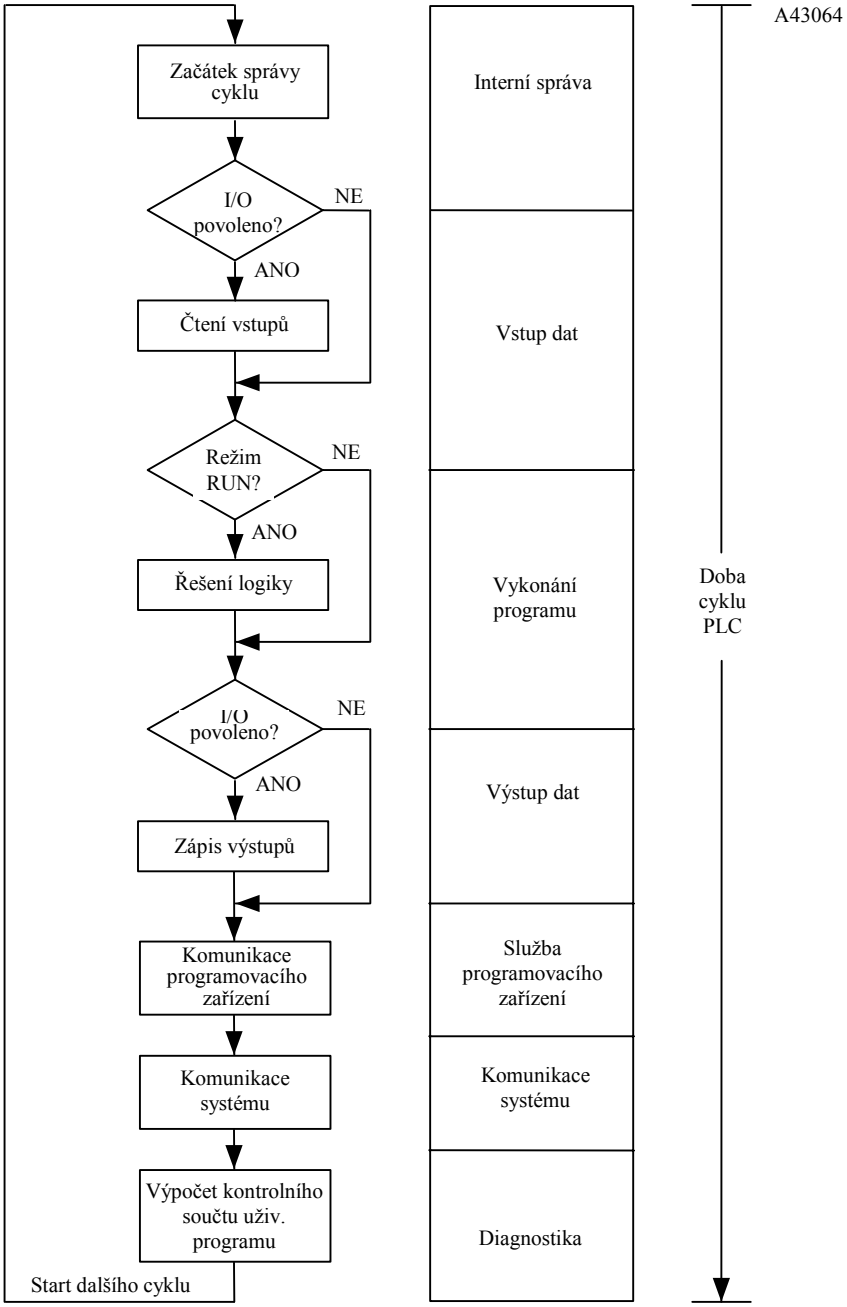
PLC Series 90-30, 90-20 a Micro normálně pracují v režimu **STANDARDNÍHO PROGRAMOVÉHO CYKLU**. Další režimy provozu jsou režim **ZASTAVENÍ SE ZÁKAZEM I/O**, režim **ZASTAVENÍ S POVOLENÍM I/O** a režim **KONSTANTNÍHO CYKLU**. Každý z těchto režimů popsaných v této kapitole je řízený vnějšími jevy a nastavením aplikační konfigurace. PLC provádí rozhodování týkající se svého provozního režimu na začátku každého cyklu.

Standardní programový cyklus

Režim **STANDARDNÍHO PROGRAMOVÉHO CYKLU** normálně běží za všech podmínek. CPU pracuje tak, že vykonává aplikační program, aktualizuje I/O a provádí komunikaci a ostatní úlohy. To se provádí v opakovaném cyklu nazývaném cyklus CPU. Prováděcí sekvence standardního programového cyklu má sedm částí:

1. Správa při spuštění cyklu
2. Snímání vstupů (čtení vstupů)
3. Řešení aplikačního programu logiky
4. Zápis výstupů (aktualizace výstupů)
5. Programátorské služby
6. Jiné než programátorské služby
7. Diagnostika

Všechny tyto kroky vykonávají každý cyklus. I když se okno komunikace programovacího zařízení otevře každý cyklus, programátorské služby se objeví, pouze pokud se zjistí porucha karty nebo pokud programovací zařízení vydá požadavek služby; to znamená, okno komunikace programovacího zařízení nejdříve zkontroluje práci, kterou má vykonat, a pokud žádná není, program ukončí. Sekvence standardního programového cyklu je zobrazena na následujícím obrázku.



Obrázek 2-1. Cyklus PLC

Jak je ukázáno v sekvenci cyklu PLC, je v ní zahrnuto několik položek. Tyto položky se podílejí na celkovém času cyklu, jak je ukázáno v následující tabulce.

Tabulka 2-1. Podíl na času cyklu

Cyklus Prvek	Popis		Časový podíl (ms) [4]
			řada 351, 352, 350 a 36x (časy pro řadu 350 a 36x se předpokládají stejné)
Správa	• Doba výpočtu cyklu. • Plánování startu dalšího cyklu. • Určení režimu dalšího cyklu • Aktualizace referenčních tabulek chyb. • Časovač resetu časovací jednotky		0.279
Zápis dat	Vstupní data se načítají ze vstupních a přídavných modulů		Podíly na času snímání viz tabulka 2-2.
Program Vykonání	Řeší se uživatelská logika		Doba vykonávání závisí na délce programu a typu instrukcí používaných v programu. Doby vykonávání instrukcí jsou uvedené v Příloze A.
Výstup dat	Výstupní data se posílají na výstup a přídavné I/O moduly.		Podíly na času snímání viz tabulka 2-2.
Externí zařízení služby	Zpracovávají se požadavky servisu od programovacích zařízení a inteligentních modulů. [1]	HHP	0.334
		LM-90	0.517
		PCM [2]	0.482
Rekonfigurace	Monitorují se sloty s vadnými moduly a prázdné sloty.		0.319[6]
Diagnostika	Ověřuje se integrita uživatelského programu		0.010 na slovo, pro které se provádí kontrolní součet v každém cyklu [3, 7]

1. Podíl obsluhy externího zařízení na čase cyklu závisí na režimu komunikačního okna, ve kterém se služba zpracovává. Pokud režim okna bude **OMEZENÝ**, během tohoto okna se spotřebuje maximálně 8 milisekund v případě CPU 311, 313, 323, a 331 a 6 milisekund v případě CPU 340 a vyšší. Pokud režim okna bude **ÚPLNÝ**, v tomto okně se spotřebuje maximálně 50 ms v závislosti na počtu požadavků, které se objeví současně.
2. Tato měření se prováděla, když PCM bylo fyzicky přítomno ale nebylo nakonfigurované a na PCM neběžela žádná aplikační úloha.
3. Počet slov, u kterých se v každém cyklu kontroluje součet, je možno změnit pomocí funkčního bloku SVCREQ.
4. Tato měření se prováděla s prázdným programem a výchozí konfigurací. CPU Series 90-30 byly v prázdné sestavě s 10 sloty bez připojených přídavných sestav. Časy v této tabulce také předpokládají, že není aktivní žádná periodický podprogram. pokud bude aktivní periodická podprogram, časy budou delší.
5. Čas načítání dat pro PLC Micro je možno určit následovně: 0.365 ms. (pevné snímání) + 0.036 ms. (doba filtru) x (celková doba cyklu)/0.5 ms.
6. Protože PLC Micro má statický soubor I/O, rekonfigurace není nutná.
7. Protože uživatelský program pro PLC Micro je uložený v paměti Flash, integrita se u něj nekontroluje.

Tabulka 2-2. Podíl času snímání I/O u Series 90-30 35x a 36x (v milisekundách)

Typ modulu		CPU Series 35x a 36x		
		Hlavní sestava	Přídavná sestava	Vzdálená sestava
8-bodový diskrétní vstup		.030	.055	.206
16-bodový diskrétní vstup		.030	.055	.206
32-bodový diskrétní vstup		.043	.073	.269
8-bodový diskrétní výstup		.030	.053	.197
16-bodový diskrétní výstup		.030	.053	.197
32-bodový diskrétní výstup		.042	.070	.259
Kombinace diskrétního vstupu/výstupu		.060	.112	.405
4-kanálový analogový vstup		.075	.105	.396
2-kanálový analogový výstup		.058	.114	.402
16-kanálový analogový vstup (proud nebo napětí)		.978	1.446	3.999
8-kanálový analogový výstup		1.274	1.988	4.472
Kombinace analogového vstupu/výstupu		1.220	1.999	4.338
Vysokorychlostní čítač		1.381	2.106	5.221
I/O procesor		1.574	2.402	6.388
Rozhraní Ethernet (bez připojení)		.038	.041	.053
Power Mate APM (1 osa)		1.527	2.581	6.388
Power Mate APM (2 osy)		1.807	2.864	7.805
DSM 302 *	40 AI, 6 AQ	2.143	3.315	9.527
	50 AI, 9 AQ	2.427	3.732	11.092
	64 AI, 12 AQ	2.864	4.317	13.138
GCM	žádná zařízení	.911	1.637	5.020
	8 zařízení se 64 slovy	8.826	16.932	21.179
GCM+	žádná zařízení	.567	.866	1.830
	32 zařízení se 64 slovy	1.714	2.514	5.783
GBC	žádná zařízení	.798	1.202	2.540
	32 zařízení se 64 slovy	18.382	25.377	70.777
PCM 311	nenakonfigurovaný nebo než aplikační úlohy	.476	Nepoužívá se	Nepoužívá se
	co nejrychlejší čtení 128 %R	.485	Nepoužívá se	Nepoužívá se
ADC (žádná úloha)		.476	Nepoužívá se	Nepoužívá se
I/O Link Master	žádná zařízení	.569	.865	1.932
	16 zařízení se 64 body	4.948	7.003	19.908
I/O Link Slave	32 zařízení se 64 body	.087	.146	.553
	64 zařízení se 64 body	.154	.213	.789

* U aplikací, kde podíl času na snímání DSM má vliv na strojní operaci, může být k přenosu potřebných dat do a z modulu pohybu nutné použít funkční blok Do I/O, a Suspend I/O a požadavek na službu Přístup ke stavu rychlé vnitřní komunikace, aniž by se při každém snímání načítala data. Podrobnosti viz *Návod pro použití Motion Mate DSM302 pro PLC Series 90-30* , GFK1464.

Tabulka 2-3. Podíl času na snímání I/O pro 90-30 Series až do 341 (v milisekundách)

Typ modulu		Model CPU						
		311/313	331			340/341		
			Hlavní rošt	Přídavný rošt	Dislokovaný rošt	Hlavní rošt	Přídavný rošt	Dislokovaný rošt
8-bodový diskrétní vstup		.076	.054	.095	.255	.048	.089	.249
16-bodový diskrétní vstup		.075	.055	.097	.257	.048	.091	.250
32-bodový diskrétní vstup		.094	.094	.126	.335	.073	.115	.321
8-bodový diskrétní výstup		.084	.059	.097	.252	.053	.090	.246
16-bodový diskrétní výstup		.083	.061	.097	.253	.054	.090	.248
32-bodový diskrétní výstup		.109	.075	.129	.333	.079	.114	.320
8-bodová kombinace vstupu/výstupu		.165	.141	.218	.529	.098	.176	.489
4-kanálový analogový výstup		.151	.132	.183	.490	.117	.160	.462
2-kanálový analogový výstup		.161	.138	.182	.428	.099	.148	.392
Vysokorychlostní čítač		2.070	2.190	2.868	5.587	1.580	2.175	4.897
Power Mate APM (1 osa)		2.330	2.460	3.175	6.647	1.750	2.506	5.899
Power Mate APM (2 osy)		3.181	3.647	4.497	9.303	2.154	3.097	7.729
DSM 302	40 AI, 6 AQ	3.613	4.081	5.239	11.430	2.552	3.648	9.697
	50 AI, 9 AQ	4.127	4.611	5.899	13.310	2.911	4.170	11.406
	64 AI, 12 AQ	4.715	5.276	6.759	15.747	3.354	4.840	13.615
GCM	žádná zařízení	.041	.054	.063	.128	.038	.048	.085
	8 zařízení se 64 body	11.420	11.570	13.247	21.288	9.536	10.648	19.485
GCM+	žádná zařízení	.887	.967	1.164	1.920	.666	.901	1.626
	32 zařízení se 64 body	4.120	6.250	8.529	21.352	5.043	7.146	20.052
PCM 311	nenakonfigurovaný nebo bez aplikační úlohy	Nepou-žívá se	3.350	Nepoužívá se	Nepoužívá se	1.684	Nepoužívá se	Nepoužívá se
	co nejrychlejší čtení 128 %R	Nepou-žívá se	4.900	Nepoužívá se	Nepoužívá se	2.052	Nepoužívá se	Nepoužívá se
ADC 311		Nepouž ívá se	3.340	Nepoužívá se	Nepoužívá se	1.678	Nepoužívá se	Nepoužívá se
16-kanálový analogový výstup (proud nebo napětí)		1.370	1.450	1.937	4.186	1.092	1.570	3.796
I/O Link Master	žádná zařízení	1.910	2.030	1.169	1.925	.678	.904	1.628
	Šestnáct zařízení se 64 body	6.020	6.170	8.399	21.291	4.992	6.985	20.010
I/O Link Slave	32 bodů	.206	.222	.289	.689	.146	.226	.636
	64 bodů	.331	.350	.409	1.009	.244	.321	.926

* U aplikací, kde podíl času na snímání DSM má vliv na strojní operaci, může být k přenosu potřebných dat do a z modulu pohybu nutné použít funkční blok Do I/O, a Suspend I/O a požadavek na službu Přístup ke stavu rychlé vnitřní komunikace, aniž by se při každém snímání načítala data. Podrobnosti viz *Návod pro použití Motion Mate DSM302 pro PLC Series 90-30* , GFK1464.

Výpočet doby cyklu

Tabulka 2-1 uvádí seznam sedmi položek, které se podílejí na době cyklu PLC. Doba cyklu se skládá z pevných časů (správa a diagnostika) a proměnných časů. Proměnné časy se liší v závislosti na konfiguraci I/O, velikosti uživatelského programu a typu programovacího zařízení připojeného k PLC.

Příklad výpočtu doby cyklu

V následující tabulce je uvedený příklad výpočtu pro určení doby cyklu pro PLC Series 90-30, model 311.

Moduly a instrukce použité pro tento výpočet jsou následující:

- Vstupní moduly: Pět 16-bodových vstupních modulů Series 90-30.
- Výstupní moduly: Čtyři 16-bodové výstupní moduly Series 90-30.
- Programovací instrukce: Program s 1200 kroky skládající se ze 700 Booleovských instrukcí (LD, AND, OR, atd.), 300 výstupních cívek (OUT, OUTM, atd.) a 200 matematických funkcí (ADD, SUB, atd.).

Správa

Část cyklu týkající se správy vykonává všechny úlohy potřebné k přípravě spuštění cyklu. Pokud PLC bude v režimu **KONSTANTNÍ CYKLUS**, cyklus se bude zobrazovat, dokud neuplyne požadovaná doba cyklu. Jestliže požadovaná doba cyklu již uplynula, nastaví se kontakt OV_SWP %SA0002 a cyklus bude pokračovat bez prodlevy. Dále se hodnoty časovače (setiny, desetiny a sekundy) aktualizují výpočtem zbytku od spuštění předchozího cyklu a času nového cyklu. Aby se udržovala přesnost, skutečný start cyklu se zaznamenává v inkrementech 100 mikrosekund. Každý časovač má pole zbytku, které obsahuje počet inkrementů po 100 mikrosekundách, které se objevily od poslední inkrementace hodnoty časovače.

Čtení vstupů

Čtení vstupů se provádí během části cyklu pro čtení vstupů těsně před řešením logiky. Během této části cyklu se čtou všechny vstupní moduly Series 90-30 a jejich data se uloží podle příslušnosti v paměti %I (diskrétní vstupy) nebo %AI (analogové vstupy). Veškerá globální vstupní data, která přijdou na komunikační modul Genius, rozšířený komunikační modul Genius nebo řadič sběrnice Genius, se uloží v paměti %G.

Moduly se čtou ve vzestupném pořadí adres počínaje komunikačním modulem Genius, pak moduly diskrétních vstupů a nakonec moduly analogových vstupů.

Pokud CPU bude v režimu **STOP** a bude nakonfigurované tak, že v režimu **STOP** se nemá číst I/O, čtení vstupů se přeskočí.

Čtení nebo řešení aplikačního programu logiky

Čtení aplikačního programu logiky znamená, že se právě vykonává aplikační program logiky. Řešení logiky začne vždy první instrukcí v uživatelském aplikačním programu hned, jakmile se dokončí načtení vstupů. Řešení logiky připraví nový soubor výstupů. Řešení logiky skončí, když se vykoná instrukce END (pokud nebudete používat přenosný monitor, instrukce END bude neviditelná).

Aplikační program se vykoná v ISCP v mikroprocesoru 80C188. V případě CPU model 313 a vyšší ISCP bude vykonávat Booleovské instrukce; a 80C188 nebo 80386EX vykonává instrukce časovače, čítače a funkčních bloků. V případě CPU model 311 a 90-20 80C188 bude vykonávat všechny Booleovské instrukce, instrukce časovače, čítače a funkčního bloku. V případě Micro procesor H8 bude vykonávat všechny Booleovské instrukce a instrukce funkčních bloků.

Seznam časů pro vykonávání jednotlivých programových funkcí je uvedený v Příloze A.

Zápis výstupů

Výstupy se zapisují během části cyklu pro zápis výstupů hned po řešení logiky. Výstupy se aktualizují s použitím dat z paměti %Q (pro diskrétní výstupy) a %AQ (pro analogové výstupy). Pokud komunikační modul Genius bude nakonfigurovaný pro přenos globálních dat, pak se data z paměti %G pošlou do GCM, GCM+ nebo GBC. Zápis výstupů u Series 90-20 a Micro zahrnuje pouze diskrétní výstupy.

Během zápisu výstupů se zapisují všechny výstupní moduly Series 90-30 ve vzestupném pořadí adres.

Pokud CPU bude v režimu **STOP** a bude nakonfigurovaná tak, že v režimu **STOP** se nemá číst/zapisovat I/O, zápis výstupů se přeskočí. Zápis výstupů se dokončí, když se všechna výstupní data odešlou do výstupních modulů Series 90-30.

Výpočet kontrolního součtu programu logiky

Výpočet kontrolního součtu se vykoná u uživatelského programu na konci každého cyklu. Protože výpočet kontrolního součtu celého programu může trvat dlouho, na obrazovce detailů CPU můžete zadat počet slov, které se mají zkontrolovat, v rozmezí od 0 do 32.

Pokud vypočítaný kontrolní součet nebude souhlasit s referenčním kontrolním součtem, nastaví se příznak chybného kontrolního součtu programu. To bude mít za následek, že se do tabulky chyb PLC zapíše chyba a režim PLC se změní na **STOP**. Pokud se výpočet kontrolního součtu nedokončí, na okno komunikace programovacího zařízení to nebude mít vliv. Výchozí počet slov, pro které se provádí kontrolní součet, je 8.

Okno komunikace programovacího zařízení

Tato část cyklu je vyhrazena komunikaci s programovacím zařízením. Pokud bude připojený programátor, CPU vykoná okno komunikace programovacího zařízení. Pokud nebude připojeno žádné programovací zařízení a v systému nebude nakonfigurovaná žádná karta, okno komunikace programovacího zařízení se nevykoná. Při každému cyklu se nakonfiguruje pouze jedna karta.

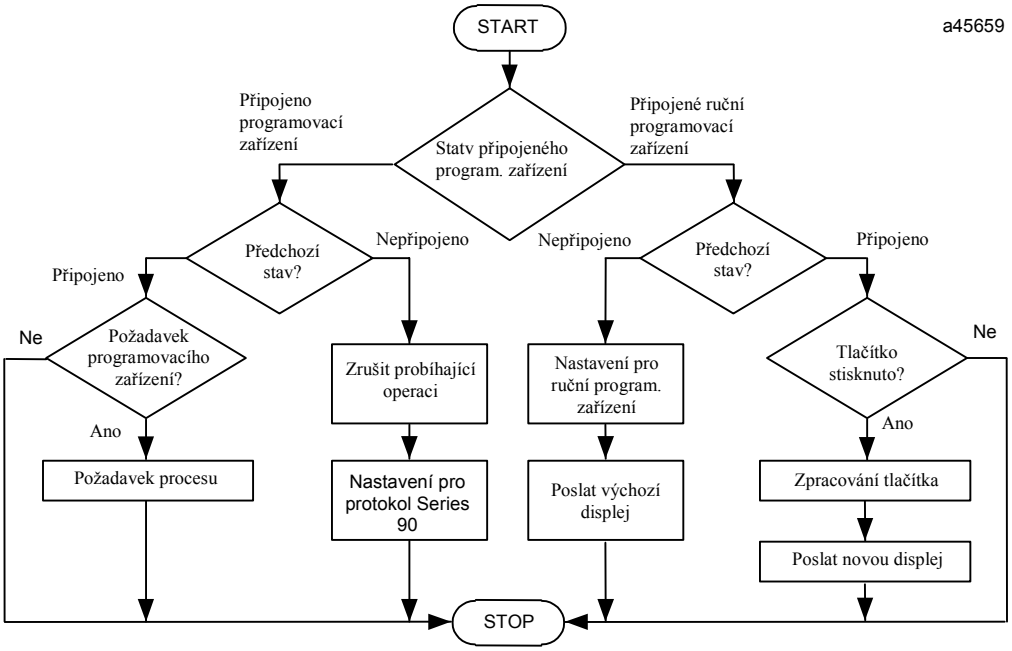
Podporují se přenosné a ostatní programovací zařízení, která se mohou připojit k sériovému portu a používat protokol Series Ninety Protocol (SNP). Podporována je také komunikace programovacího zařízení s inteligentními přídavnými moduly.

Ve výchozím režimu omezeného okna CPU vykoná při každém cyklu jednu operaci na programovací zařízení, to znamená, že akceptuje jeden požadavek nebo odezvu na službu nebo na jedno stisknutí tlačítka. Pokud programovací zařízení vznese požadavek, který vyžaduje více než 6 (nebo 8 v závislosti na CPU – viz poznámka) milisekund na zpracování, zpracování požadavku se rozloží na několik cyklů tak, aby žádný cyklus netrval déle než 6 (nebo 8 v závislosti na CPU – viz poznámka) milisekund.

Poznámka

Časový limit pro okno komunikace je 6 milisekund pro CPU modely 340 a vyšší a 8 milisekund pro modely 311, 313, 323 a 331.

Na následujícím obrázku je vývojový diagram pro část cyklu provádějící komunikaci programovacího zařízení.

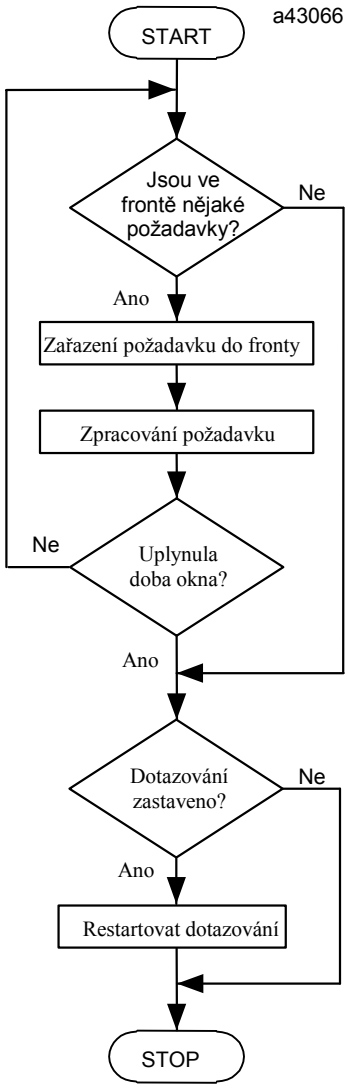


Obrázek 2-2. Vývojový diagram okna komunikace programovacího zařízení

Okno systémové komunikace

Toto je část cyklu, kde se zpracovávají požadavky na komunikaci od inteligentních přídavných modulů, například PCM nebo DSM (viz vývojový diagram). Požadavky se obslouží podle pořadí příchodu. Protože však jsou inteligentní přídavné moduly dotazované cyklickým způsobem, žádný inteligentní přídavný modul nebude mít přednost před jiným modulem.

Ve výchozím režimu **Úplný** je délka okna systémové komunikace omezena na 50 milisekund. Pokud inteligentní přídavný modul vznese požadavek, který vyžaduje více než 50 milisekund na zpracování, zpracování požadavku se rozloží na několik cyklů tak, aby žádný cyklus netrval déle než 50 milisekund.

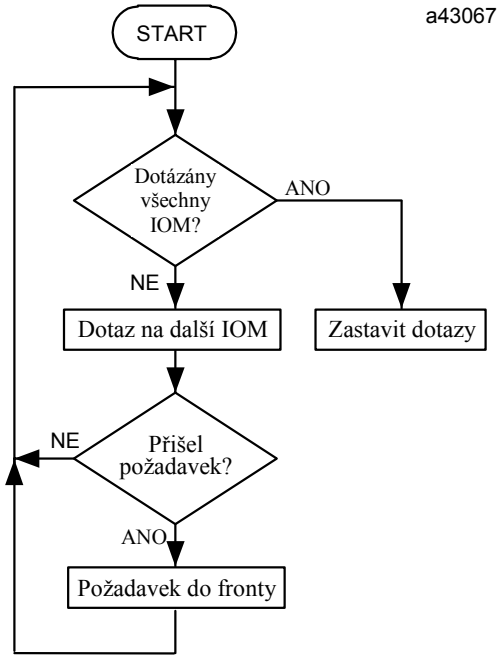


Obrázek 2-3. Vývojový diagram okna systémové komunikace

PCM komunikace s PLC (modely 331 a vyšší)

Inteligentní přídavné moduly (IOM), například PCM, nemají žádnou možnost přerušit CPU, když potřebují obsloužit. CPU se musí každého inteligentního přídavného modulu dotázat, jestli vyžaduje obsloužení. Tento dotaz se během cyklu objevuje asynchronně na pozadí (viz následující vývojový diagram).

Když inteligentní přídavný modul bude dotázán a pošle do CPU požadavek na službu, požadavek se zařadí do fronty na zpracování během okna systémové komunikace.



Obrázek 2-4. Komunikace PCM s PLC

Komunikace DSM s PLC

DSM302 je inteligentní modul pracující asynchronně s CPU modulem Series 90-30. Výměna dat mezi CPU a DSM probíhá automaticky.

DSM je možno nakonfigurovat na tři různé délky dat %AI a %AQ. CPU PLC vyžaduje čas na čtení a zápis dat v interní komunikaci s DSM302. Tabulka 2-2 uvádí délku cyklů při různých konfiguracích dat %AI a %AQ na cyklus. Další úvahy o časování, které platí pro modul DSM302, najdete v *Návodu pro použití Motion Mate DSM302 pro PLC Series 90-30* GFK-1464.

Změny standardního programového cyklu

Kromě normálního vykonávání standardního programového cyklu je možno se setkat nebo si vynutit určité změny. Tyto změny popsané v následujících odstavcích je možno zobrazit a/nebo měnit z programovacího softwaru.

Režim konstantní doby cyklu

U standardního programového cyklu se každý cyklus vykonává pokud možno co nejrychleji, přičemž délka času spotřebovaného na každý cyklus se mění. Opakem k tomuto je režim **KONSTANTNÍ DOBA CYKLU**, kde každý cyklus spotřebuje stejný čas. Toho je možno dosáhnout nastavením nakonfigurovaného konstantního cyklu, který pak bude výchozím režimem cyklu a bude platný vždy, když PLC přejde z režimu **STOP** do režimu **RUN**. Pro časovač konstantního cyklu je možno použít hodnotu v rozsahu 5 až 200 milisekund (nebo až 500 milisekund v případě CPU PLC řady 35x a 36x) (výchozí hodnota je 100 milisekund).

Vzhledem ke změnám doby požadované pro různé části PLC cyklu musí být konstantní doba cyklu nastavená alespoň o 10 milisekund delší než doba cyklu, která se zobrazuje na stavovém řádku, když PLC bude v režimu **NORMÁLNÍ CYKLUS.** Tím se zabrání tomu, aby se vyskytly vedlejší chyby překročení doby cyklu.

Konstantní cyklus použijte, když je nutno se dotazovat na I/O body nebo hodnoty registru s konstantní četností, například algoritmus řízení. Jedním důvodem pro použití režimu **KONSTANTNÍ DOBY CYKLU** může být zajištění, aby se I/O aktualizovaly v konstantních intervalech. Dalším důvodem může být nutnost zajistit, aby se určitá část času spotřebovala mezi zápisem výstupu a dalším čtením vstupu cyklu a vstupy po příchodu dat výstupu z programy se mohly usadit.

Pokud skončí doba konstantního cyklu před jeho dokončením, celý cyklus včetně oken se dokončí. Avšak na začátku dalšího cyklu se zaznamená chyba překročení cyklu.

Poznámka

Na rozdíl od aktivního konstantního cyklu, který je možno editovat jen v režimu **RUN**, režim konfigurovaného konstantního cyklu je možno editovat pouze během režimu **STOP** a než bude změna platná, je nutno “uložit konfiguraci z programovacího zařízení do PLC”. Po uložení tento režim bude jako výchozí režim cyklu.

Cyklus PLC v režimu STOP

Když bude PLC v režimu **STOP**, aplikační program se nevykoná. Komunikace s programovacím zařízením a inteligentními přídavnými moduly bude pokračovat. Kromě toho bude během režimu **STOP** pokračovat vykonávání dotazování chybné desky a rekonfigurace desky. Z důvodu efektivity operační systém používá větší

hodnoty časových úseků, než které se používají v režimu **RUN** (obvykle kolem 50 milisekund na okno). Můžete se rozhodnout, jestli se I/O má nebo nemá číst. Čtení/zápis I/O se může vykonat v režimu **STOP**, pokud parametr ***IOScan-Stop*** na obrazovce CPU detailů bude nastavený na **YES**.

Režimy okna komunikace

Výchozí režim komunikace pro okno komunikace programovacího zařízení je režim "Omezený". To znamená, že pokud zpracování požadavku bude trvat déle než 6 milisekund, zpracuje se ve více cyklech tak, že jeden cyklus nebude trvat déle než 6 milisekund. V případě CPU model 313, 323 a 331 může být délka cyklu během ukládání v režimu **RUN** až 12 milisekund. Režim aktivního okna je možno změnit pomocí obrazovky "Kontrola cyklu" v programu Logicmaster – Instrukce pro změnu režimu aktivního okna najdete v kapitole 5, "Řízení a stav PLC" v *Návodu pro použití programovacího softwaru Logicmaster 90™ Series 90™-30/20/Micro* (GFK-0466).

Poznámka

Pokud se režim systémového okna změní na Omezený, přídavné moduly jako například PCM nebo GBC, které komunikují s PLC pomocí systémového okna, budou mít menší vliv na dobu cyklu, ale odezva na jejich požadavky bude pomalejší.

Klíček u CPU řady 35x a 36x: Změna režimu a ochrana Flash

Každá CPU řady 35x a 36x má na přední části modulu klíček, který umožňuje obsluze ochránit paměť Flash před přepsáním. Když klíček nastavíte do polohy **ON**/**RUN**, nikdo nebude moct změnit paměť Flash, aniž by klíčkem otočil do polohy OFF.

Počínaje verzí 7 CPU modely 351a 352 má klíček jinou funkci: umožňuje přepnout PLC do režimu **STOP**, do režimu **RUN** a vynulovat nezávažné poruchy, jak je popisováno v této části.

Počínaje verzí 8 CPU řady 35x a 36x má klíček rozšířenou funkci ochrany paměti: může se použít k zajištění dvou dalších typů ochrany paměti (viz část "Používání ochrany paměti pro verzi 8 a pozdější").

Pokud klíček bude aktivovaný a v poloze ON/RUN, čas denních hodin je možno změnit pouze pomocí programovacího softwaru. Ruční programovací zařízení neumožňuje změnit čas denních hodin, pokud ochrana klíčkem bude aktivní.

Použití klíčku pro verzi 7 a pozdější

Na rozdíl od možností ochrany paměti Flash v dřívějších verzích, pokud na konfigurační obrazovce CPU klíček nepovolíte pomocí parametru přepínače **RUN**/**STOP**, CPU nebude mít zde popisované rozšířené řízení.

Činnost přepínače má stejná blokování a kontroly před přechodem PLC do režimu **RUN** stejně jako u stávajícího přechodu do režimu **RUN**; to znamená, že PLC nepřejde do režimu **RUN** přes vstup klíčku, pokud PLC bude v režimu **STOP**/**FAULT**. Avšak v režimu **STOP/FAULT** můžete smazat nefatální poruchy a pomocí klíčku převést PLC do režimu **RUN**.

Pokud v tabulce chyb budou chyby, které ***nejsou fatální*** (to znamená, že nebudou mít za následek přechod CPU do režimu **STOP**/**FAULT**), pak CPU přejde do režimu **RUN** při prvním přepnutí ze stavu Stop do Run a tabulka chyb se NEVYNULUJE.

Pokud v tabulce chyb budou chyby, které ***jsou fatální*** (CPU bude v režimu **STOP**/**FAULT**), pak první přepnutí klíčku z polohy **STOP** do polohy **RUN** bude mít za následek, že kontrolka **RUN** na CPU bude blikat rychlostí 2 Hz a spustí se časovač 5 sekund. Blikající kontrolka **RUN** indikuje, že v tabulce chyb jsou závažné chyby. V takovém případě CPU NEPŘEJDE do stavu **RUN**, ani když klíček bude v poloze **RUN**.

Vynulování tabulky chyb pomocí klíčku

Pokud přepnete klíček z polohy **RUN** do polohy **STOP** a nazpět do polohy **RUN** během 5 sekund když kontrolka **RUN** bude blikat, chyby se vynulují a CPU přejde do režimu **RUN**. Kontrolka přestane blikat a zůstane trvale ve stavu **ON**. Klíček musí zůstat buď v poloze **RUN** nebo **STOP** alespoň 1/2 sekundy, než se přepne do opačné polohy.

Poznámka

Pokud necháte uplynout 5 sekund (kontrolka **RUN** přestane blikat), CPU zůstane v původním stavu, režim **STOP**/**FAULT** a chyby v tabulce zůstanou. Pokud klíček v tomto okamžiku přepnete opět z polohy **STOP** do polohy **RUN**, proces se zopakuje s tím, že toto bude jako první přechod.

V následující tabulce je uvedený přehled, jak nastavení těchto dvou parametrů CPU, které mají vliv na klíček (přepínač R/S a IOScan-Stop) a fyzickou polohu klíčku, ovlivní PLC.

Parametr přepínače R/S v konfiguraci CPU	Poloha klíčku	Parametr IOScan-Stop v konfiguraci CPU	Činnost PLC
OFF	X	X	Všechny režimy programovacího zařízení PLC jsou povolené.
ON	ON/RUN	X	Všechny režimy programovacího zařízení PLC jsou povolené.
ON	OFF/STOP	X	PLC nesmí přejít do RUN.
ON	Přepněte přepínač z OFF/STOP do ON/RUN	X	PLC přejde do RUN, pokud se nevyskytly závažné chyby; jinak kontrolka RUN LED bude blikat 5 sekund.
ON	Přepněte přepínač z ON/RUN do OFF/STOP	NE	PLC přejde do STOP–NO IO
ON	Přepněte přepínač z ON/RUN do OFF/STOP	ANO	PLC přejde do STOP–IO

X = Nemá žádný vliv bez ohledu na nastavení

Rozšířená ochrana paměti u CPU verze 8 pozdější

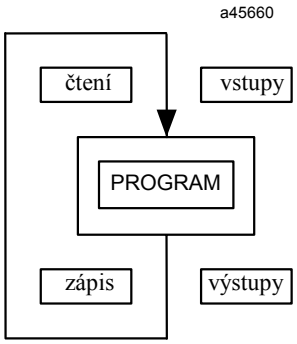
U CPU verze 8 a pozdější má klíček všechny výše uvedené funkce plus při nastavení parametru v programovacím souboru je možno ho použít k ochraně paměti RAM, takže paměť RAM nelze z programovacího softwaru změnit. Když tato ochrana paměti bude povolena, budou zablokované dva typy operací: uživatelský program a konfiguraci nelze změnit a přepis hodnot bodu není povolený. To je aktivováno pomocí pole Mem Protect na obrazovce konfigurace CPU řady 35x nebo 36x v Logicmaster. Výchozí nastavení je Zakázáno.

Část 2: Organizace programu a uživatelské adresy/data

Celková velikost logiky u programovatelných automatů Series 90-30 je uvedena v následující tabulce.

Modely	Paměť uživatelské logiky (Kbajtů)
CPU311	6
CPU313, CPU323	12
CPU331	16
CPU340	32
CPU341	80
CPU350	80 (verze 9 a pozdější) 32 (před verzí 9)
CPU351, CPU352, CPU360, CPU363, CPU364	240 (verze 9 a pozdější) 80 (před verzí 9)

Počínaje CPU verze 9 je možno velikost některých pamětí pro řadu 351, 352 a 36x konfigurovat. (Podrobnější instrukce a popis použitelných velikostí paměti najdete v "Konfigurovatelná paměť pro CPU verze 351 a vyšší" v kapitole 10, Části 3 v *Návodu pro použití programovacího softwaru Logicmaster 90™ Series 90™-30/20/Micro* (GFK-0466K nebo pozdější). Program pro programovatelný automat Series 90-20 může mít pro CPU Model 211 velikost až 2 KB. Uživatelský program obsahuje logiku, která se používá při jeho spouštění. Maximální počet logických příček přípustných na jeden logický blok (hlavní nebo podprogram) je 3000; pro PLC 90-30 je maximální velikost bloku 80 kilobajtů pro bloky C a 16 kilobajtů pro bloky LD a SFC, v bloku SFC se ale část z těchto 16 KB používá pro interní datový blok. PLC vykonává logiku opakovaně.



Seznam velikostí programů a meze adres pro jednotlivé modely CPU najdete v *Návodu pro použití programovatelného automatu Series 90-30* GFK-0356, nebo v *Návodu pro použití programovatelného automatu Series 90-20* GFK-0551.

Všechny programy mají tabulku proměnných, která uvádí popis proměnných a adres, které byly v uživatelském programu přiřazené.

Editor deklarace bloku uvádí bloky podprogramů deklarovaných v hlavním programu.

Bloky podprogramů

Program může při svém vykonávání “vyvolat” bloky podprogramů. Podprogram musí být deklarovaný pomocí editoru deklarace bloku předtím, než je pro tento podprogram možno použít instrukci CALL. V každém bloku logiky v programu je možno použít maximálně 64 deklarací bloků podprogramu a 64 instrukcí CALL. Maximální velikost bloku podprogramu je 16 KB nebo 3000 logických příček, ale hlavní program a všechny podprogramy musí vyhovovat omezením velikosti logiky pro daný model CPU.

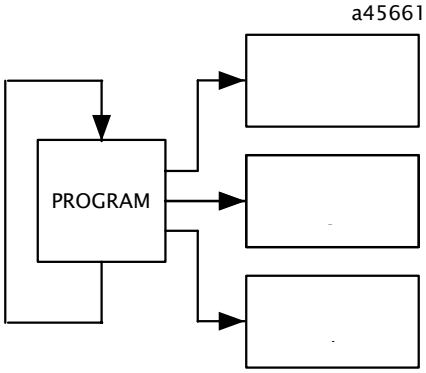
Poznámka

Bloky podprogramů nejsou k dispozici pro PLC Series 90-20 PLC ani pro Micro.

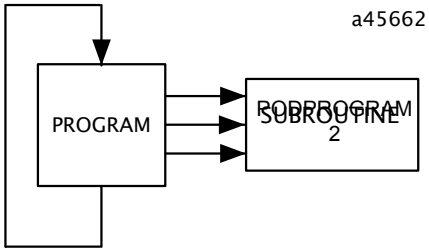
Použití podprogramů je volitelné. Rozdělení programu na menší podprogramy může zjednodušit programování, zlepšit pochopení řídicího algoritmu a zmenšit celkovou velikost logiky potřebnou pro program.

Příklady používání bloků podprogramů

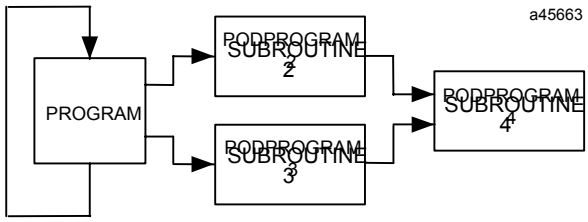
Logiku pro program je možno například rozdělit na tři podprogramy, z nichž každý by byl vyvolávaný z programu podle potřeby. V tomto příkladu blok programu může obsahovat málo logiky a bude sloužit především k řazení bloků podprogramů.



Blok podprogramu je možno použít během vykonávání programu libovolněkrát. Logika, kterou je nutno opakovat v programu vícekrát, by měla být zapsaná jako blok podprogramu. K vyvolání této logiky pak je nutno volat tento blok podprogramu. Tímto způsobem se zmenší velikost programu.



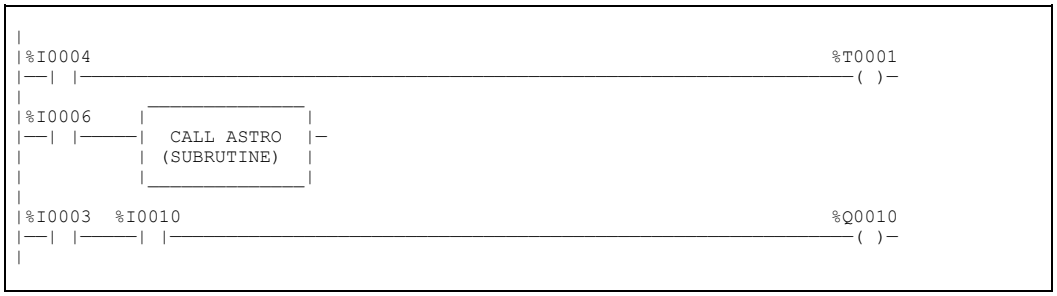
Kromě vyvolání z programu je možno bloky podprogramu také vyvolávat i z jiných bloků podprogramu. Blok podprogramu může dokonce vyvolat sám sebe.



PLC umožní pouze osm vnořovaných volání a pak se zaregistruje chyba "Přetečení zásobníkové paměti aplikace" a PLC přejde do režimu **`STOP/Fault`**. Hlavní program se počítá jako úroveň vnořování 1.

Jak se bloky vyvolávají

Blok podprogramu se vykoná, když bude vyvolaný z programové logiky v programu nebo z jiného bloku.



Tento příklad ukazuje instrukci volání podprogramu CALL tak, jak se objeví ve vyvolávacím bloku.

Periodické podprogramy

Verze 4.20 nebo pozdější CPU 340 a vyšší podporuje periodické podprogramy. Povšimněte si následujících omezení:

1. Funkční bloky časovače (TMR, ONDTR a OFDTR) se v periodickém podprogramu nevykonají správně. Funkční blok DOIO v rámci periodického podprogramu, jehož rozsah adres zahrnuje adresy přiřazené modulu Smart I/O Module (HSC, Power Mate APM, Genius, atd.) způsobí, že CPU ztratí komunikaci s modulem. Kontakty FST_SCN a LST_SCN (%S1 a %S2) budou mít během vykonávání periodického podprogramu nedefinovanou hodnotu. Periodický podprogram nemůže volat ani nemůže být volán jiným podprogramem.
2. Doba vyčkávání pro periodický podprogram (to je maximální interval mezi dobou, kdy se periodický má vykonat, a dobou, kdy se skutečně vykoná) může být kolem 0.35 milisekund, pokud v hlavní sestavě nebude žádný modul PCM, CMM nebo ADC. Pokud v hlavní sestavě bude modul PCM, CMM nebo ADC —i když nebude nakonfigurovaný pro použití —doba vyčkávání může být až 2.25 milisekund. Proto se používání periodických podprogramů s výrobky na bázi PCM se ***nedoporučuje***.

Uživatelské adresy

Data používaná v aplikačním programu se ukládají buď jako registr nebo diskrétní adresa.

Tabulka 2-4. Adresy registrů

Typ	Popis
%R	Předpona %R se používá k přiřazení adresy systémových registrů, což uloží data programu jako výsledek výpočtu.
%AI	Předpona %AI představuje analogový vstupní registr. Za touto předponou následuje adresa registru (například %AI0015). V analogovém vstupním registru je uložena hodnota jednoho analogového vstupu nebo jiná hodnota.
%AQ	Předpona %AQ představuje analogový výstupní registr. Za touto předponou následuje adresa registru (například %AQ56). V analogovém výstupním registru je uložena hodnota jednoho analogového výstupu nebo jiná hodnota.

Poznámka

Všechny adresy registrů během cyklu zapínání CPU zůstávají zachované.

Tabulka 2-5. Diskrétní adresy

Typ	Popis
%I	Předpona %I představuje adresy vstupů. Za touto předponou následuje adresa z tabulky vstupů (například %I00121). Adresy %I se nacházejí v tabulce stavu vstupů, ve které jsou uložené stavy všech vstupů, které přišly ze vstupních modulů během posledního čtení vstupů. Adresa se diskrétním vstupním modulům přiřadí pomocí konfiguračního softwaru nebo pomocí ručního programovacího zařízení. Dokud nebude adresa přiřazena, z modulu není možno přijímat žádná data. Data %I mohou být retentivní nebo neretentivní.
%Q	Předpona %Q představuje adresy fyzických výstupů. Funkce kontroly cívky programu Logicmaster 90-30/20/Micro kontroluje vícenásobné používání adres %Q u cívek relé nebo výstupů funkcí. Počínaje verzí 3 softwaru můžete zvolit úroveň požadované kontroly cívky (`SINGLE`, `WARN MULTIPLE` nebo `MULTIPLE`). Více informací k této funkci najdete v Návodu k používání programovacího softwaru GFK-0466. Za předponou %Q následuje adresa v tabulce výstupů (například %Q00016). Adresy %Q se nacházejí v tabulce stavu výstupů, ve které jsou uložené stavy adres výstupů, jak je naposledy nastavil aplikační program. Tyto hodnoty tabulky stavu výstupů se posílají na výstupní moduly během zápisu na výstupy. Adresa se diskrétním výstupním modulům přiřadí pomocí konfiguračního softwaru nebo pomocí ručního programovacího zařízení. Dokud nebude adresa přiřazena, do modulu není možno odesílat žádná data. Daná adresa %Q může být buď retentivní nebo neretentivní. *
%M	Předpona %M představuje interní adresy. Funkce kontroly cívky zkontroluje vícenásobné použití adres %M u cívek relé nebo výstupů funkcí. Počínaje verzí 3 softwaru můžete zvolit úroveň požadované kontroly cívky (`SINGLE`, `WARN MULTIPLE` nebo `MULTIPLE`). Více informací o této funkci najdete v GFK-0466. Daná adresa %M může být buď retentivní nebo neretentivní. *
%T	Předpona %T představuje přechodné adresy. Protože u těchto adres se nikdy nekontroluje vícenásobné použití cívky, tyto adresy možno je ve stejném programu použít mnohokrát, i když kontrola použití cívky bude povolena. %T je možno použít k zabránění konfliktu použití cívek, když se používají funkce ??cut/paste a file write/include. Protože tato paměť je určena pro přechodné použití, není zajištěna proti ztrátě napětí nebo přechodům **`RUN-TO-STOP-TO-RUN`** a nelze jí proto použít s retentivními cívkami.

* Retentivnost je založena na typu cívky. Více informací najdete v kapitole “Retentivnost dat” na straně 2-21.

Tabulka 2-5. Diskrétní adresy - pokračování

Typ	Popis
%S	Předpona %S představuje adresy stavu systému. Tyto adresy se používají pro přístup ke speciálním PLC datům, jako například časovače, informace čtení a informace o chybách. Systémové adresy zahrnují adresy %S, %SA, %SB a %SC. %S, %SA, %SB a %SC je možno použít na libovolné kontakty. %SA, %SB a %SC je možno použít na retentivní cívky –(M)–. %S je možno použít jako vstupní argumenty na bázi slova nebo bitového řetězce pro funkce nebo funkční bloky. %SA, %SB a %SC je možno použít jako vstupní nebo výstupní argumenty na bázi slova nebo bitového řetězce pro funkce nebo funkční bloky.
%G	Předpona %G představuje adresy globálních dat. Tyto adresy se používají pro přístup k datům sdíleným několika PLC. Adresy %G je možno použít na kontakty a retentivní cívky, protože paměť %G je stále retentivní. %G nelze použít na neretentivní cívky.

Přechody a přepisy

Uživatelské adresy %I, %Q, %M a %G mají související bity přechodu a přepisu. Adresy %T, %S, %SA, %SB a %SC mají bity přechodu ale nemají bity přepisu. CPU využívá bitů přechodu pro čítače a přechodové cívky. Všimněte si, že čítače nepoužívají stejný druh bitů přechodu jako cívky. Bity přechodu pro čítače jsou uložené na lokální adrese.

V CPU modelech 331 a vyšších je možno nastavit bity přepisu. Když bity přepisu budou nastavené, související adresy nelze změnit z programu nebo vstupního zařízení; lze je změnit pouze pomocí povelu z programovacího zařízení. CPU modely 323, 321, 313 a 311 a CPU Micro nepodporují diskrétní adresy pro přepis.

Retentivnost dat

Data se nazývají retentivní, pokud je PLC při zastavení uloží. PLC Series 90 uschová programovou logiku, tabulky chyb a diagnostiku, přepisy a vynucené výstupy, slovní data (%R, %AI, %AQ), bitová data (%I, %SC, %G), chybové bity a vyhrazené bity), %Q a %M data (pokud nejsou používané s neretentivními cívkami) a slovní data uložená v %Q a %M. Data %T se neuloží. I když jak je uvedeno výše, bitová data %SC jsou retentivní, výchozí hodnoty pro %S, %SA a %SB jsou neretentivní.

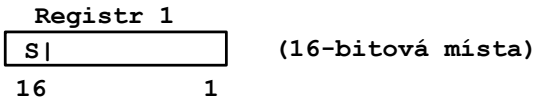
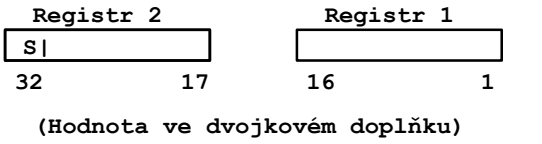
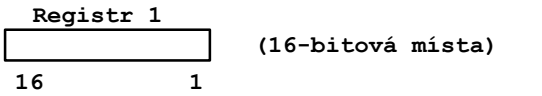
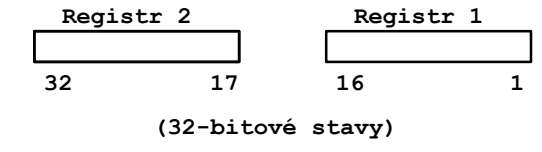
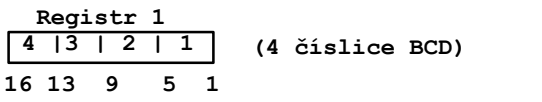
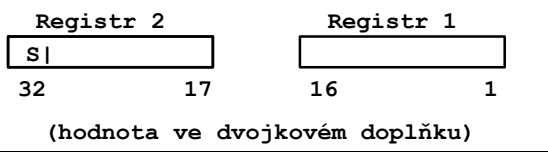
Adresy %Q a %M jsou neretentivní (to znamená, že se po zapnutí napájení vynulují, když PLC přejde z **STOP** do **RUN**) vždy, když se budou používat s neretentivními cívkami. Mezi neretentivní cívky patří cívky —( )—, negované cívky —(/)—, cívky SET —(S)— a cívky RESET —(R)—.

Když se adresy %Q nebo %M použijí s retentivními cívkami nebo se použijí jako výstupy funkčního bloku, obsah se zachová i při ztrátě napětí a přechodech **RUN-TO-STOP-TO-RUN**. Mezi retentivní cívky patří retentivní cívky —(M)—, negované retentivní cívky —(/M)—, retentivní cívky SET —(SM)— a retentivní cívky RESET —(RM)—.

Poslední naprogramování adresy %Q nebo %M pomocí instrukce cívky určuje, jestli adresa %Q nebo %M bude retentivní nebo neretentivní podle typu cívky. Pokud například bylo %Q0001 naposledy naprogramováno jako adresa retentivní cívky, data %Q0001 budou retentivní. Pokud však %Q0001 bylo naposledy naprogramováno jako neretentivní cívka, data %Q0001 budou neretentivní.

Typy dat

Tabulka 2-6. Typy dat

Typ	Název	Popis	Formát dat
INT	Celé číslo se znaménkem	Celá čísla se znaménkem používají 16-bitová paměťová místa a zobrazují se ve tvaru dvojkového doplňku Platný rozsah dat typu INT je –32,768 až +32,767.	
DINT	Celé číslo se znaménkem s dvojnásobnou délkou	Celá čísla se znaménkem s dvojnásobnou přesností se ukládají na 32-bitových paměťových místech (ve skutečnosti to jsou dvě po sobě jdoucí 16-bitová paměťová místa) a zobrazují se ve tvaru dvojkového doplňku. (Bit 32 je znaménkový bit.) Platný rozsah dat typu DINT je –2,147,483,648 až +2,147,483,647.	
BIT	Bit	Data bitového typu jsou nejmenší jednotkou paměti. Mají dva stavy, 1 nebo 0. Bitový řetězec má délku N.	
BYTE	Byte	Data typu Byte mají 8-bitovou hodnotu. Platný rozsah je 0 až 255 (0 až FF v hexadecimálním tvaru).	
WORD	Slovo	Data typu slova používají 16 po sobě jdoucích bitů datové paměti; ale místo bitů v paměti dat představujících číslo bity jsou navzájem nezávislé. Každý bit představuje svůj vlastní binární stav (1 nebo 0) a na bity se nepohlíží tak, že společně tvoří celé číslo. Platný rozsah hodnot slova je 0 až FFFF.	
DWORD	Dvojité slovo	Data typu dvojitého slova mají stejné charakteristiky jako data typu jednoduchého slova ale s tím rozdílem, že používají 32 po sobě jdoucích bitů v paměti dat místo 16 bitů.	
BCD-4	Čtyřmístné dvojkově kódované desítkové číslo	Čtyřmístná BCD čísla používají 16-bitová paměťová místa. Každá BCD číslice používá čtyři bity a může představovat číslo od 0 do 9. Toto BCD kódování 16 bitů má přípustný rozsah hodnot 0 až 9999.	
REAL	Plovoucí tečka	Reálná čísla používají 32 po sobě jdoucích bitů (ve skutečnosti dvě po sobě jdoucí 16-bitová paměťová místa). Rozsah čísel, která je možno uložit v tomto formátu je od ± 1.401298E-45 do ± 3.402823E+38.	

S = Znaménkový bit (0 = kladný, 1 = záporný).

Adresy stavu systému

Adresy stavu systému v PLC Series 90 PLC jsou přiřazené paměti %S, %SA, %SB a %SC. Všechny mají svou zkratku. Příkladem adres časovacích kontaktů může být T_10MS, T_100MS, T_SEC a T_MIN. Příkladem konvenčních adres může být FST_SCN, ALW_ON a ALW_OFF.

Poznámka

Bity %S jsou bity pouze pro čtení; nezapisujte do těchto bitů. Můžete však zapisovat do bitů %SA, %SB a %SC.

V následující tabulce jsou uvedené adresy stavu systému, které je možno použít v aplikačním programu. Když budete zapisovat logiku, můžete použít buď adresu nebo zkratku. Podrobnější popis chyb a informace k jejich opravě najdete v kapitole 3, “Vysvětlivky a oprava chyb”.

Tyto zvláštní názvy nelze použít v jiném kontextu.

Tabulka 2-7. Adresy stavu systému

Adresa	Zkratka	Definice
%S0001	FST_SCN	Nastaveno na 1, když aktuální cyklus je první cyklus.
%S0002	LST_SCN	Resetováno z 1 na 0, když aktuální cyklus je poslední cyklus.
%S0003	T_10MS	Kontakt s časováním 0.01 sekundy
%S0004	T_100MS	kontakt s časováním 0.1 sekundy
%S0005	T_SEC	kontakt s časováním 1.0 sekundy
%S0006	T_MIN	Kontakt s časováním 1.0 minuty
%S0007	ALW_ON	Vždy ve stavu ON.
%S0008	ALW_OFF	Vždy ve stavu OFF.
%S0009	SY_FULL	Nastaví se, když tabulka PLC chyb bude plná. Vynuluje se, když se z PLC tabulky chyb odstraní zápis a když se tabulka PLC chyb smaže.
%S0010	IO_FULL	Nastaví se, když I/O tabulka chyb bude plná. Vynuluje se, když se z I/O tabulky chyb odstraní zápis a když se I/O tabulka chyb smaže.
%S0011	OVR_PRE	Nastaví se, když v paměti %I, %Q, %M nebo %G se bude vyskytovat přepis.
%S0013	PRG_CHK	Nastaví se, když kontrola programu na pozadí bude aktivní.
%S0014	PLC_BAT	U CPU verze 4 nebo pozdější se nastaví jako indikace špatného stavu baterie. Adresa kontaktu se aktualizuje jednou za cyklus.

Tabulka 2-7. Adresy stavu systému – pokračování

Adresa	Název	Definice
%S0017	SNPXACT	SNP-X nadřízený počítač je aktivně připojený k CPU.
%S0018	SNPX_RD	SNP-X nadřízený počítač načetl data z CPU.
%S0019	SNPX_WT	SNP-X nadřízený počítač zapsal data do CPU.
%S0020		Nastaví se do stavu ON, když se relační funkce používající data REAL vykoná úspěšně. Vynuluje se, když některý ze vstupů bude NaN (Ne číslo).
%S0032		Vyhrazeno pro použití programovacím softwarem.
%SA0001	PB_SUM	Nastaví se, když kontrolní součet vypočítaný pro aplikační program nebude souhlasit s referenčním kontrolním součtem. Pokud chyba bude v důsledku přechodné poruchy, diskrétní bit se může vynulovat opakovaným uložením programu do CPU. Pokud chyba bude v důsledku poruchy RAM, je nutno vyměnit CPU.
%SA0002	OV_SWP	Nastaví se, když PLC zjistí, že předchozí cyklus trval déle než čas určený uživatelem. Vynuluje se, když PLC zjistí, že předchozí cyklus netrvá déle, než určený čas. Také se vynuluje během přechodu z režimu **STOP** do režimu **RUN**. Platí pouze, pokud PLC bude v režimu **KONSTANTNÍ CYKLUS** .
%SA0003	APL_FLT	Nastaví se, když se vyskytne chyba aplikace. Vynuluje se, když PLC přejde z režimu **STOP** do režimu **RUN**.
%SA0009	CFG_MM	Nastaví se, když se zjistí nesoulad konfigurace během zapínání systému nebo během ukládání konfigurace. Vynuluje se zapnutím PLC, když se nevyskytne žádný nesoulad nebo během ukládání konfigurace, která je v souladu s hardwarem.
%SA0010	HRD_CPU	Nastaví se, když diagnostika zjistí problém s hardwarem CPU. Vynuluje se vyměněním modulu CPU.
%SA0011	LOW_BAT	Nastaví se, když se vyskytne chyba nízkého napětí baterie. Vynuluje se výměnou baterie a zajištěním, že napájení PLC proběhne bez stavu nízkého napětí baterie.
%SA0014	LOS_IOM	Nastaví se, když I/O modul přestane komunikovat s PLC CPU. Vynuluje se výměnou modulu a vykonáním cyklu zapnutí napájení hlavní sestavy.
%SA0015	LOS_SIO	Nastaví se, když přídavný modul přestane komunikovat s PLC CPU. Vynuluje se výměnou modulu a vykonáním cyklu zapnutí napájení hlavní sestavy.
%SA0019	ADD_IOM	Nastaví se, když se k hlavní sestavě přidá I/O modul. Vynuluje se vykonáním cyklu zapnutí napájení hlavní sestavy a když konfigurace bude souhlasit s hardwarem po uložení.
%SA0020	ADD_SIO	Nastaví se, když se k hlavní sestavě přidá přídavný modul. Vynuluje se vykonáním cyklu zapnutí napájení hlavní sestavy a když konfigurace bude souhlasit s hardwarem po uložení.
%SA0027	HRD_SIO	Nastaví se, když se v přídavném modulu zjistí chyba hardwaru. Vynuluje se výměnou modulu a vykonáním cyklu zapnutí napájení hlavní sestavy.
%SA0031	SFT_SIO	Nastaví se, když se v přídavném modulu zjistí neopravitelná chyba softwaru. Vynuluje se vykonáním cyklu zapnutí napájení hlavní sestavy a když konfigurace bude souhlasit s hardwarem.
%SB0010	BAD_RAM	Nastaví se, když CPU zjistí po zapnutí napájení poškozená data v paměti RAM. Vynuluje se, když CPU zjistí, že paměť RAM je po zapnutí napájení platná.

Tabulka 2-7. Adresy stavu systému – pokračování

Adresa	Zkratka	Definice
%SB0011	BAD_PWD	Nastaví se, když se objeví porušení přístupu přes heslo. Vynuluje se, když se vynuluje tabulka chyb PLC.
%SB0013	SFT_CPU	Nastaví se, když CPU zjistí neopravitelnou chybu softwaru. Vynuluje se vynulováním tabulky chyb PLC.
%SB0014	STOR_ER	Nastaví se, když se během operace ukládání programovacího zařízení objeví chyba. Vynuluje se, když se operace ukládání dokončí úspěšně.
%SC0009	ANY_FLT	Nastaví se, když se vyskytne nějaká chyba. Vynuluje se, když v obou tabulkách chyb nebudou žádné záznamy.
%SC0010	SY_FLT	Nastaví se, když se objeví nějaká chyba, která způsobí, že do PLC tabulky chyb se uloží záznam. Vynuluje se, když v tabulce PLC chyb nebude žádný záznam.
%SC0011	IO_FLT	Nastaví se, když se objeví nějaká chyba, která způsobí, že do I/O tabulky chyb se uloží záznam. Vynuluje se, když v tabulce I/O chyb nebude žádný záznam.
%SC0012	SY_PRES	Nastaví se, pokud v tabulce PLC chyb bude alespoň jeden záznam. Vynuluje se, když v tabulce PLC chyb nebude žádný záznam.
%SC0013	IO_PRES	Nastaví se, pokud v tabulce I/O chyb bude alespoň jeden záznam. Vynuluje se, když v tabulce I/O chyb nebude žádný záznam.
%SC0014	HRD_FLT	Nastaví se, když se vyskytne chyba hardwaru. Vynuluje se, když v obou tabulkách chyb nebudou žádné záznamy.
%SC0015	SFT_FLT	Nastaví se, když se vyskytne chyba softwaru. Vynuluje se, když v obou tabulkách chyb nebudou žádné záznamy.

Poznámka: Každá adresa %S, která zde není uvedená, je vyhrazená a nelze jí v programové logice použít.

Struktura funkčního bloku

Každá logická příčka se skládá z jedné nebo více programových instrukcí. To mohou být jednoduchá relé nebo složitější funkce.

Formát relé žebříkové logiky

Programovací software obsahuje několik typů funkcí relé. Tyto funkce zajišťují základní tok a řízení logiky v programu. Příkladem mohou být rozepnutý kontakt relé a negovaná cívka. Každý z těchto kontaktů relé a cívka mají jeden vstup a jeden výstup. Společně zajišťují logický tok přes kontakt nebo cívku.

Každý kontakt relé nebo cívka musí mít adresu, která se zapíše při volbě relé. U kontaktu adresa představuje umístění v paměti, které určuje tok energie do kontaktu. Pokud v následujícím příkladu adresa %I0122 bude ON, přes kontakt tohoto relé poteče energie.

```
%I0122
 -| |-
```

U cívky adresa představuje umístění v paměti, které je řízeno tokem energie do cívky. Pokud v následujícím příkladu poteče energie do levé strany cívky, adresa %Q0004 se přepne na ON.

```
%Q0004
-( )-
```

Programovací software a ruční programovací zařízení mají funkci kontroly cívky, která kontroluje vícenásobné použití adresy %Q nebo %M u cívek relé nebo výstupů funkcí.

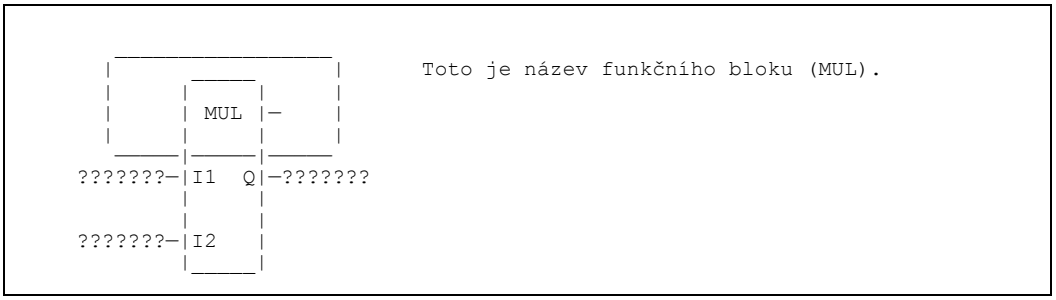
Formát programových funkčních bloků

Některé funkce jsou velmi jednoduché, například funkce MCR, která je uvedena v hranatých závorkách jako zkratka názvu funkce:

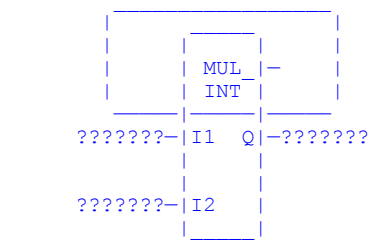
```
-[ MCR ]-
```

Jiné funkce jsou složitější. Ty mohou mít několik míst, kde se zapisují informace, které se mají s funkcí používat.

Obecný funkční blok zobrazený níže je násobení (MUL); parametry se liší v závislosti na typu funkčního bloku. Jeho části jsou typické pro mnoho programových funkcí. Horní část funkčního bloku udává název funkce. Může také udávat typ dat; v tomto případě to je celé číslo se znaménkem.



Toto je název funkčního bloku (MUL).



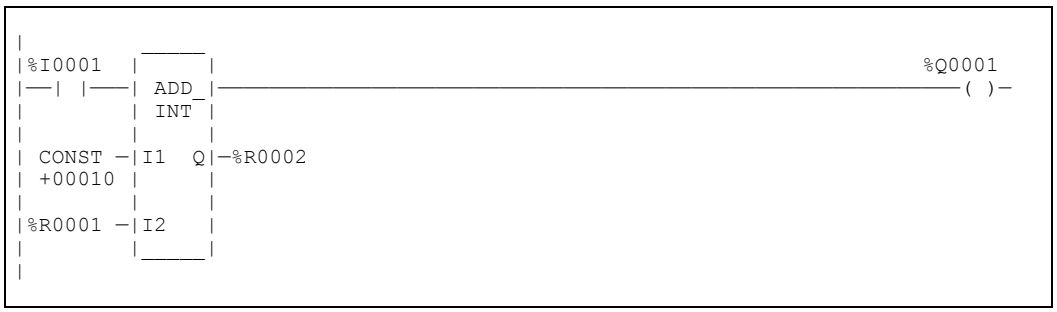
Toto je název funkčního bloku (MUL) a typ dat INT). INT (celé číslo se znaménkem) představuje typ a velikost dat, se kterými se má pracovat.

Mnoho programových funkcí umožňuje zvolit typ dat pro funkci po zvolení samotné funkce. Například typ dat pro funkci MUL se může změnit na celé číslo se znaménkem s dvojnásobnou délkou. Další informace o typech dat jsou uvedené na začátku této kapitoly.

Parametry funkčního bloku

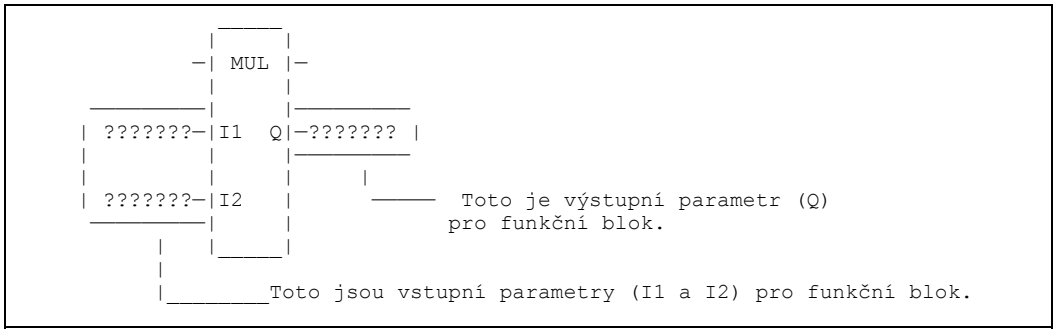
Každý řádek vstupující do levé strany funkčního bloku představuje vstup tohoto bloku. Existují dvě formy vstupu, které je možno předat do funkčního bloku: konstanty a adresy. Konstanta je explicitní hodnota. Adresa je adresa hodnoty.

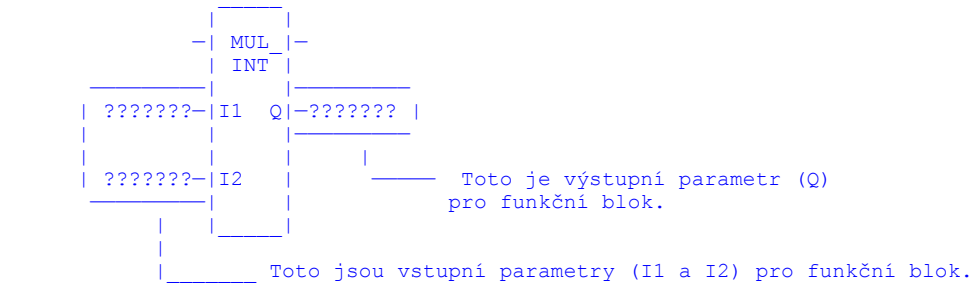
V následujícím příkladu vstupní parametr I1 přichází do funkčního bloku ADD jako konstanta a vstupní parametr I2 přichází jako adresa.



Každý řádek, který vystupuje na pravé straně funkčního bloku, představuje výstup. Existuje pouze jedna forma výstupu z funkčního bloku nebo adresy. Výstupy se nikdy nemohou zapisovat do konstant.

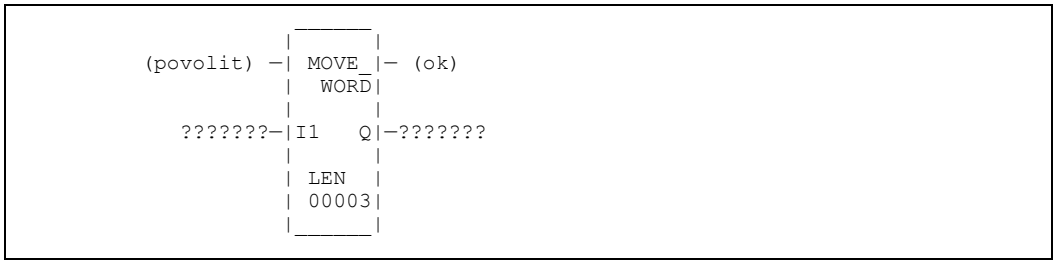
Kde se na levé straně funkčního bloku objeví otazník, musí se zapsat buď samotná data, adresové místo, kde se data nacházejí, nebo proměnná představující adresové místo, kde se data nacházejí. Kde se na pravé straně funkčního bloku objeví otazník, obvykle se zapíše adresa umístění dat, která se mají přenést na výstup funkčního bloku, nebo proměnná, která představuje adresové místo dat, která se mají přenést na výstup funkčního bloku.



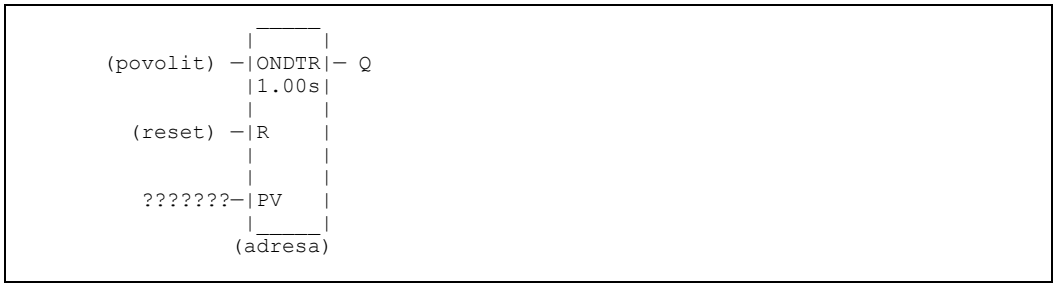


Většina funkčních bloků nemění vstupní data; místo toho umístí výsledek operace do výstupní adresy.

U funkcí, které pracují s tabulkami, je možno pro funkci zvolit délku. U následujícího funkčního bloku operand LEN zadává počet slov, které se mají přesunout.

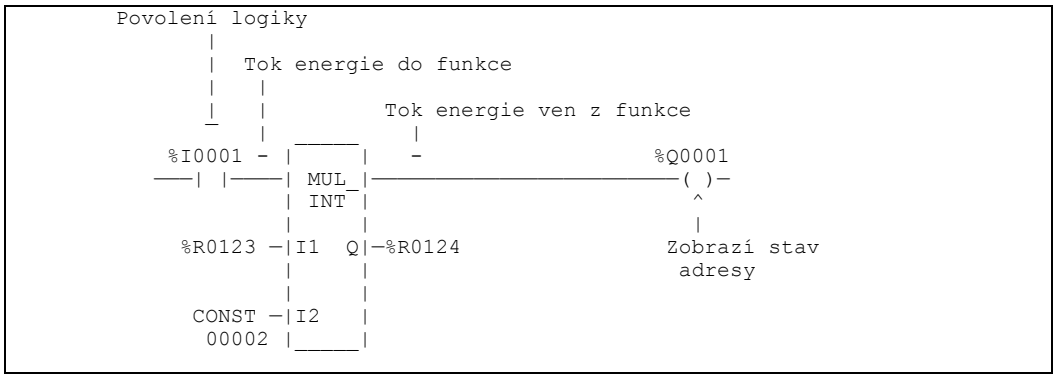


Časovač, čítač, BITSEQ a funkce ID vyžadují adresu pro umístění tří slov (registrů), ve kterých je uložena aktuální hodnota, předvolená hodnota a řídicí slovo nebo "Výskyt" funkce.



Tok energie dovnitř a ven z funkce

Energie teče do funkčního bloku vlevo nahoře. Často se k řízení toku energie do funkčního bloku používá povolení logiky; jinak funkční blok vykoná bezpodmínečně každý cyklus CPU.



Poznámka

Funkční bloky někdy nelze připojit přímo k levé napájecí liště. K vyvolání funkce při každém cyklu je možno použít %S7, bit ALW_ON (vždy v jedničce) s normálně rozpojeným kontaktem připojeným k napájecí liště.

Energie teče ven z funkčního bloku vpravo nahoře. Může se převést do jiné programové logiky nebo na cívku (volitelně). Funkční bloky přenesou energii, když se vykonají úspěšně.

Část 3: Sekvence zapínání a vypínání napájení

U PLC Series 90-30 jsou dvě možné sekvence zapínání napětí; studený a teplý start. CPU normálně používá studený start. Pokud však u PLC systémů model 331 nebo vyšších doba, která uplyne mezi vypnutím a dalším zapnutím napájení, bude kratší než 5 sekund, použije se sekvence teplého startu.

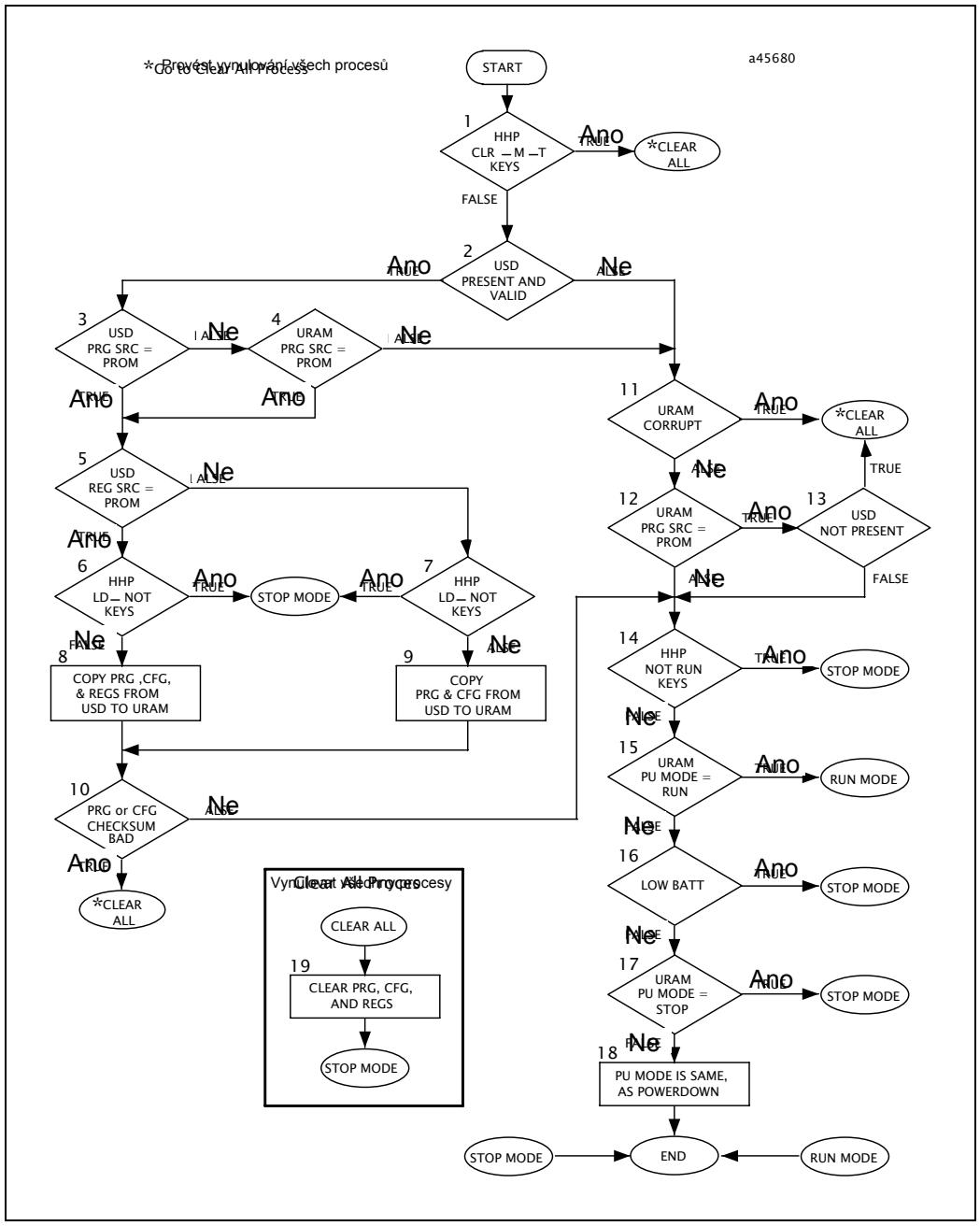
Zapínání

Studený start se skládá z následující posloupnosti dějů. Sekvence teplého startu přeskočí krok 1.

1. CPU provede diagnostiku sebe sama. To zahrnuje kontrolu části baterií zálohované paměti RAM, jestli tato paměť obsahuje platná data.
2. Pokud bude nainstalovaná paměť EPROM, EEPROM nebo Flash a v paměti PROM je nastavena volba, že se při spouštění má použít obsah PROM, obsah PROM se zkopíruje do paměti RAM. Pokud paměť EPROM, EEPROM nebo Flash nebude nainstalovaná, paměť RAM zůstane stejná a obsahem PROM se nepřepíše.
3. CPU prozkoumá každý slot v systému a zjistí, která karta je nainstalovaná.
4. Konfigurace hardwaru se porovná s konfigurací softwaru, aby se zajistilo, že jsou stejné. Jakékoliv zjištěné neshody se budou pokládat za chyby a bude se generovat chybové hlášení. Stejně tak pokud v softwarové konfiguraci bude zadaná karta, ale ve skutečné hardwarové konfiguraci bude nainstalovaný odlišný modul, tento stav bude pokládaný za chybu a bude se generovat chybové hlášení.
5. Pokud softwarová konfigurace nebude existovat, CPU použije výchozí konfiguraci.
6. CPU zavede komunikační kanál mezi sebou a inteligentními moduly.
7. V posledním kroku vykonávání se podle konfigurace CPU určí režim prvního cyklu. V režimu **RUN** cyklus bude pokračovat podle popisu přechodu režimu "**STOP-to-RUN**." Obrázek 2-5 na následující stránce uvádí popis sekvence pro CPU, když rozhoduje, jestli má kopírovat z paměti PROM nebo zapnout napájení v režimu **STOP** nebo **RUN**.

Poznámka

Kroky 2 až 7 výše neplatí pro PLC Series 90 Micro. Informace o sekvenci zapínání a vypínání napájení pro Micro najdete v části Sekvence zapínání a vypínání napájení" v kapitole 5, "Činnost systému," v *Návodu pro použití PLC Series 90 Micro* (GFK-1065).



Obrázek 2-5. Sekvence zapínání napájení

Před příkazem START ve vývojovém diagramu zapínání napájení CPU projde diagnostikou zapínání napájení, která provede test různých periferních zařízení používaných v CPU a otestuje i paměť RAM. Po skončení diagnostiky se provede inicializace interních datových struktur a periferních zařízení, která CPU používá. CPU pak určí, jestli došlo ke zničení dat v uživatelské RAM. Pokud došlo ke zničení dat v uživatelské paměti RAM, uživatelský program a konfigurace se smažou a nastaví na výchozí hodnoty a smažou se i všechny uživatelské registry.

TERMÍNY VE VÝVOJOVÉM DIAGRAMU:

PRG = uživatelský program
CFG = uživatelská konfigurace
REGS = uživatelské registry (adresy %I, %Q, %M, %G, %R, %AI a %AQ).
USD = uživatelské paměťové zařízení, buď EEPROM nebo flash zařízení.
URAM = Energeticky nezávislá paměť RAM obsahující PRG, CFG a REGS.

VYSVĚTLENÍ TEXTU VE VÝVOJOVÉM DIAGRAMU:

(1) Došlo ke stisknutí tlačítek <CLR> a <M_T> na HHP během zapínání napájení k vynulování celé URAM?
(2) Je nainstalovaná USD (může chybět pouze u modelů, které používají zařízení EEPROM) a je informace na USD platná?
(3) Je parametr PRG SRC v USD nastavený na PROM, že se má načíst PRG a CFG ze zařízení USD?
(4) Je parametr PRG SRC v URAM nastavený na PROM, že se má načíst PRG a CFG ze zařízení USD?
(5) Je parametr REG SRC v USD nastavený na PROM, že se má načíst REGS ze zařízení USD?
(6 & 7) Došlo ke stisknutí tlačítek <LD> a <NOT> na HHP během zapínání napájení, aby se PRG, CFG a REGS načetly z USD?
(8) Zkopírovat PRG, CFG a REGS z USD do URAM.
(9) Zkopírovat PRG a CFG z USD do URAM.
(10) Jsou kontrolní součty PRG nebo CFG právě načtené z USD platné?
(11) Je URAM poškozená? Může to být v důsledku výpadku napájení, pokud baterie byla odpojena nebo měla nízké napětí. Také to může být v důsledku aktualizace firmwaru.
(12) Je parametr PRG SRC v URAM nastavený na PROM, že se má načíst PRG a CFG ze zařízení USD?
(13) Je USD nainstalovaná? Platí pouze pro modely, které používají EEPROM zařízení.
(14) Došlo ke stisknutí tlačítek <**NOT**> a <**RUN**> na HHP během zapínání napájení, aby zapínání proběhlo bezpodmínečně v režimu Stop?
(15) Je parametr PWR UP v URAM nastavený na **RUN**?
(16) Je nízké napětí?
(17) Je parametr PWR UP v URAM nastavený na **STOP**?
(18) Nastavte režim zapínání napájení na režim, který byl při vypínání.
(19) Vynulujte PRG, CFG a REGS.

Poznámka

První část tohoto diagramu na předchozí stránce neplatí pro PLC Series 90 Micro. Informace o sekvenci zapínání a vypínání napájení pro Micro najdete v části Sekvence zapínání a vypínání napájení" v kapitole 5, "Činnost systému," v *Návodu pro použití PLC Series 90 Micro* (GFK-1065).

Vypnutí napájení

K vypnutí systému dojde, když se zjistí, že napájecí střídavé napětí pokleslo déle než po dobu jednoho cyklu nebo výstup napájecího zdroje 5 voltů poklesl na hodnotu nižší než 4,9 voltů ss.

Část 4: Hodiny a časovače

Hodiny a časovače, které jsou k dispozici v PLC Series 90-30, zahrnují hodiny uplynulého času, hodiny denního času (modely 331, 340/341, 351/352 a 28-bodový Micro), hlídací časovač a časovač konstantního cyklu. Tři typy časovacích funkčních bloků zahrnují časovač prodlevy při zapnutí, časovač prodlevy při vypnutí a retentivní časovač prodlevy při zapnutí (také nazývaný hlídací časovač). Čtyři časovací kontakty se spínají a rozpínají v intervalech 0,01 sekundy, 0,1 sekundy, 1 sekundy a 1 minuty.

Hodiny uplynulého času

Hodiny uplynulého času ke sledování času uplynulého od zapnutí CPU používají 100 mikrosekundový "takt". Hodiny při výpadku napájení nejsou retentivní; po zapnutí napájení se spustí znovu. Jednou za sekundu hardware přeruší CPU, aby se umožnil záznam počtu sekund. Tento počet sekund se překlopí přibližně za 100 let od začátku čítání času.

Protože hodiny uplynulého času jsou základem pro činnost systémového softwaru a časovacích funkčních bloků, nelze je z uživatelského programu nebo programovacího zařízení resetovat. Avšak aplikační program může přečíst aktuální hodnotu hodin uplynulého času pomocí požadavku službu Service Request 16.

Hodiny denního času

Hodiny denního času u 28-bodového Micro a PLC Series 90-30 model 331 a vyšší udržují hardwarové hodiny denního času. Hodiny denního času vykonávají sedm časových funkcí:

- Rok (dvě číslice)
- Měsíc
- Den v měsíci
- Hodiny
- Minuty
- Sekundy
- Den v týdnu

Hodiny denního času jsou zálohované baterií a udržují svůj stav i při výpadku napájení. Dokud však hodiny nebudou inicializované, jejich hodnota nebude mít význam. Aplikační program může načíst a nastavit hodiny denního času pomocí požadavku službu Service Request #7. Hodiny denního času je také možno načíst a nastavit z konfiguračního softwaru CPU. Všimněte si, že ruční programovací zařízení neumožňuje změnit čas denních hodin, pokud ochrana přepínačem bude aktivní.

Hodiny denního času jsou provedené tak, aby se mohly provádět přechody mezi měsíci a mezi roky. Automaticky se kompenzuje přechodný rok až do roku 2079.

Hlídací časovač

Hlídací časovač u PLC Series 90-30 je provedený tak, aby zachytil katastrofické chybové stavy, které způsobí neobvykle dlouhý cyklus. Hodnota časovače pro hlídací časovač je 200 milisekund (500 milisekund u PLC s CPU řady 35x a 36x); to je pevná hodnota, kterou nelze změnit. Hlídací časovač na začátku každého cyklu vždy začíná od nuly.

Pokud se u CPU 90-30 modely 331 a nižší překročí doba hlídacího obvodu, kontrolka OK LED zhasne; CPU přejde do resetu a úplně se zastaví; a výstupy přejdou do příslušných výchozích stavů. Neprobíhá žádná komunikace a mikroprocesory na všech kartách se zastaví. Provoz se obnoví vykonáním cyklu zapnutí napájení sestavy obsahující CPU. U CPU 90-20, Series 90 Micro a CPU 90-30 model 340 a vyšší bude mít překročení času hlídacího obvodu za následek, že CPU vykoná reset, vykoná svou logiku zapnutí napájení, vygeneruje chybu hlídacího obvodu a přejde do režimu **STOP**.

Časovač doby vypnutí

Časovač doby vypnutí se používá ke stanovení, jak dlouho bylo PLC vypnuté. Když bude PLC vypnuté, resetuje se na 0 a spustí časování. Když se PLC zapne, časování se zastaví a hodnota se zachová, Požadavek na službu Service Request #29 popsaný v kapitole 12 se může použít k načtení hodnoty tohoto časovače.

Poznámka

Tuto funkci je možno použít pouze u CPU Series 90-30 model 311 a vyšší.

Časovač konstantního cyklu

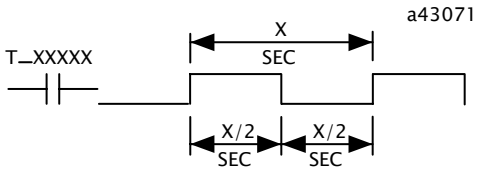
Časovač konstantního cyklu řídí délku programového cyklu, když PLC Series 90-30 bude pracovat v režimu **CYKLUS S KONSTANTNÍ DOBOU**. V tomto režimu činnosti spotřebuje každý cyklus stejně dlouhou dobu. Obvykle většina aplikačních programů, čtení vstupů, čtení logiky aplikačních programů a zápisů na výstupy vyžaduje přesně stejnou dobu vykonávání na každý cyklus. Hodnota časovače konstantního cyklu se nastaví programovacím zařízením a může mít hodnotu od 5 do hodnoty hlídacího obvodu (výchozí hodnota je 100 milisekund).

Pokud doba časovače konstantního cyklu uplyne před dokončením cyklu a nedošlo k překročení předchozího cyklu, PLC uloží do tabulky chyb PLC záznam o překročení cyklu. Na začátku následujícího cyklu PLC nastaví chybový kontakt OV_SWP. Kontakt OV_SWP se resetuje, když PLC nebude v režimu **KONSTANTNÍ DOBA CYKLU** nebo doba posledního cyklu nepřekročila konstantní dobu cyklu.

Časovací kontakty

PLC Series 90 má čtyři časovací kontakty s intervaly 0,01 sekundy, 0,1 sekundy, 1,0 sekunda a 1 minuta. Stav těchto kontaktů se během vykonávání cyklu nemění. Na těchto kontaktech se vytvářejí impulsy, které mají stejnou délku trvání stavu sepnutí a rozepnutí. Tyto kontakty mají adresy T_10MS (0,01 sekundy), T_100MS (0,1 sekundy), T_SEC (1,0 sekunda) a T_MIN (1 minuta).

Následující časový diagram ukazuje dobu sepnutí/rozepnutí těchto kontaktů.



Obrázek 2-6. Časový diagram časových kontaktů

Část 5: Bezpečnost systému

Bezpečnost u PLC Series 90-30, Series 90-20, a Micro je provedena tak, aby se zabránilo neoprávněným změnám obsahu PLC. U PLC jsou čtyři možné úrovně bezpečnosti. První úroveň, která je k dispozici vždy, zajišťuje pouze schopnost číst PLC data; aplikace nemůže provádět žádné změny. Ostatní tři úrovně mají přístup k jednotlivým úrovním chráněný heslem.

Každá vyšší úroveň oprávnění umožňuje provádět větší změny, než úroveň nižší. Úrovně oprávnění se akumulují tak, že oprávnění, které má jedna úroveň, jsou kombinací této úrovně plus všech nižších úrovní. Úrovně a jejich oprávnění jsou:

Úrovně oprávnění	Popis
Úroveň 1	Kromě hesla lze načíst jakákoliv data. To zahrnuje všechny datové paměti (%I, %Q, %AQ, %R, atd.), tabulky chyb a všechny typy programových bloků (data, hodnota a konstanta). V PLC nelze změnit žádné hodnoty.
Úroveň 2	Tato úroveň umožňuje přístup k zápisu do datové paměti. (%I, %R, atd.).
Úroveň 3	Tato úroveň umožňuje přístup k zápisu do aplikačního programu pouze v režimu **STOP**.
Úroveň 4	Toto je výchozí úroveň pro systémy, které nemají nastavené žádné heslo. Výchozí úroveň pro systém s hesly je nejvyšší nechráněná úroveň. Tato nejvyšší úroveň, umožňuje přístup pro čtení a zápis do všech pamětí i hesla v režimu **RUN** i **STOP**. (Data konfigurace nelze měnit v režimu **RUN**.)

Hesla

Pro každou úroveň oprávnění je v systému PLC jedno heslo. (Pro přístup na úrovni 1 není nutno nastavovat žádné heslo.) Každé heslo musí být jedinečné; pro více než jednu úroveň však je možno použít stejné heslo. Hesla mají délku jednoho až čtyř ASCII znaků; zapisovat nebo je měnit je možné pouze pomocí programovacího softwaru nebo pomocí ručního programovacího zařízení.

Změna úrovně oprávnění bude platná, pouze pokud se komunikace mezi PLC a programovacím zařízením nepřeruší. Nemusí probíhat nějaká aktivita, ale komunikační linka se nesmí přerušit. Pokud komunikace nebude probíhat 15 minut, úroveň oprávnění se vrátí na nejvyšší nechráněnou úroveň.

Po připojení PLC bude programovací software od PLC požadovat stav ochrany každé úrovně oprávnění. Programovací software pak bude požadovat, aby PLC přešlo na nejvyšší nechráněnou úroveň a tak dalo programovacímu softwaru přístup k nejvyšší nechráněné úrovni, aniž by musel žádat o nějakou konkrétní úroveň. Když bude k PLC připojené ruční programovací zařízení, PLC se vrátí k nejvyšší nechráněné úrovni.

Požadavky na změnu úrovně oprávnění

Programovací zařízení vyžádá změnu úrovně oprávnění dodáním nové úrovně oprávnění a hesla pro tuto úroveň. Změna úrovně oprávnění se zamítne, když heslo poslané programovacím zařízením nebude souhlasit s heslem uloženým v PLC tabulce přístupových hesel pro požadovanou úroveň. Aktuální úroveň oprávnění zůstane a žádná změna se neprovede. Pokud se budete snažit o přístup nebo budete chtít provést změnu informací v PLC pomocí ručního programovacího zařízení bez správné úrovně oprávnění, ruční programovací zařízení vyšle chybové hlášení, které přístup odmítne.

Uzamknuté/odemknuté podprogramy

Bloky podprogramů je možno uzamknout nebo odemknout pomocí funkce uzamknutí bloku z programovacího softwaru. Je možno použít dva typy uzamknutí:

Typ uzamknutí	Popis
Prohlížení	Po uzamknutí nelze v tomto podprogramu prohlížet podrobnosti.
Editování	Po uzamknutí nelze v podprogramu editovat informace.

Podprogram, který byl dříve uzamknutý pro prohlížení nebo editování, je možno odemknout v editoru deklarace bloku, pokud však nebude trvale uzamknutý pro prohlížení nebo trvale uzamknutý pro editování.

U podprogramů uzamknutých pro prohlížení je možno provádět funkci hledání nebo hledání a záměny. Pokud se v podprogramu uzamknutém pro prohlížení najde hledaný objekt, místo logiky se zobrazí některé z následujících hlášení:

```
Found in locked block <block_name>   (Continue/Quit)
(Nalezeno v uzamknutém bloku <název_bloku> (Pokračovat/konec))
```

nebo

```
Cannot write to locked block <block_name>  (Continue/Quit)
(Nelze zapisovat do uzamknutého bloku <název_bloku> (Pokračovat/konec))
```

Můžete pokračovat v hledání nebo ho ukončit.

Adresáře, které obsahují uzamknuté podprogramy, je možno vyprázdnit nebo odstranit. Pokud adresář bude obsahovat uzamknuté podprogramy, tyto bloky zůstanou uzamknuté, když se použijí funkce programovacího softwaru Copy, Backup a Restore adresáře.

Trvalé uzamknutí podprogramu

Kromě VIEW LOCK a EDIT LOCK existují dva typy trvalého uzamknutí. Pokud bude nastaveno PERMANENT VIEW LOCK, každé detailní prohlížení podprogramu bude odepřeno. Pokud bude nastaveno PERMANENT EDIT LOCK, bude odepřený každý pokus o editování bloku.

Upozornění

Trvalé uzamknutí se liší od normálního VIEW LOCK a EDIT LOCK v tom, že po jeho nastavení ho nelze odstranit.

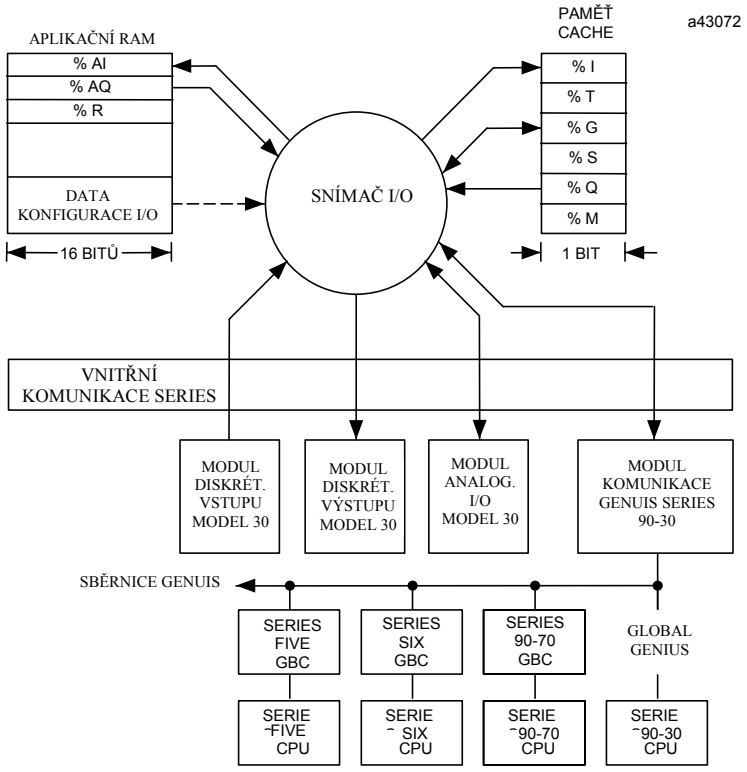
Když bude nastaveno PERMANENT EDIT LOCK, nastavení lze změnit pouze na PERMANENT VIEW LOCK. PERMANENT VIEW LOCK nelze změnit na žádný jiný typ uzamknutí.

Část 6: I/O Systémy Series 90-30, 90-20 a Micro

I/O systém PLC vytváří rozhraní mezi PLC Series 90-30 a uživatelem dodaným zařízením a vybavením. Vstupy a výstupy Series 90-30 se nazývají Series 90-30 I/O. I/O moduly Series 90-30 se zastrkávají přímo do slotů v základní desce CPU nebo do slotů v některé přídavné základní desce pro PLC Series 90-30 modely 331 nebo vyšší. I/O systémy s CPU model 331, 340 a 341 podporují až 49 I/O modulů Series 90-30 (5 sestav). I/O systémy s CPU model 351 a 352 podporují až 79 I/O modulů Series 90-30 (8 sestav). Základní deska s 5 sloty PLC Series 90-30 model 311 nebo model 313 podporuje až 5 I/O modulů Series 90-30; Základní deska s 10 sloty modelu 323 podporuje až 10 I/O modulů Series 90-30.

Struktura I/O pro PLC Series 90-30 je zobrazena na následujícím obrázku.

PLC I/O Systém



Obrázek 2-7. Struktura I/O Series 90-30

Poznámka

Výše uvedený obrázek je charakteristický pro I/O strukturu v 90-30. Inteligentní a přídavné moduly nejsou součástí čtení I/O; používají Okno komunikace systému. Další informace o struktuře I/O 90-20 najdete v *Návodu pro použití programovatelného řadiče Series 90™-20* (GFK-0551). Další informace o I/O struktuře v PLC Micro najdete v *Návodu pro použití PLC Micro Series 90™* (GFK-1065).

I/O moduly Series 90-30

I/O moduly Series 90-30 se dodávají v pěti typech, diskrétní vstup, diskrétní výstup, analogový vstup, analogový výstup a přídavné moduly. V následující tabulce je uvedený seznam I/O modulů Series 90-30 podle katalogových čísel, počtu I/O bodů a krátký popis každého modulu.

Poznámka

Všechny níže uvedené I/O moduly nemusí být v době tisku tohoto manuálu k dispozici. Aktuální dostupnost si zjistěte u svého místního distributora PLC GE Fanuc nebo prodejního zástupce GE Fanuc. Specifikace a informace k zapojení jednotlivých I/O modulů Series 90-30 najdete v *Manuálu specifikací I/O modulů Series 90-30*, GFK-0898.

Obrázek 2-8. I/O moduly Series 90-30

Katalogové číslo	Body	Popis	Číslo publikace
		Diskrétní moduly - vstup	
IC693MDL230	8	120 V stř. oddělený	GFK-0898
IC693MDL231	8	240 V stř. oddělený	GFK-0898
IC693MDL240	16	120 V stříd.	GFK-0898
IC693MDL241	16	24 V stř./ss kladná/záporná logika	GFK-0898
IC693MDL630	8	24 V ss kladná logika	GFK-0898
IC693MDL632	8	125 V ss kladná/záporná logika	GFK-0898
IC693MDL633	8	24 V ss záporná logika	GFK-0898
IC693MDL634	8	24 V ss kladná/záporná logika	GFK-0898
IC693MDL640	16	24 V ss kladná logika	GFK-0898
IC693MDL641	16	24 V ss záporná logika	GFK-0898
IC693MDL643	16	24 V ss kladná logika, rychlé	GFK-0898
IC693MDL644	16	24 V ss záporná logika, rychlé	GFK-0898
IC693MDL645	16	24 V ss kladná/záporná logika	GFK-0898
IC693MDL646	16	24 V ss kladná/záporná logika, rychlé	GFK-0898
IC693MDL652	32	24 V ss kladná/záporná logika	GFK-0898
IC693MDL653	32	24 V ss kladná/záporná logika, rychlé	GFK-0898
IC693MDL654	32	5/12 V ss (TTL) kladná/záporná logika	GFK-0898
IC693MDL655	32	24 V ss kladná/záporná logika	GFK-0898
IC693ACC300	8/16	Simulátor vstupu	GFK-0898

Tabulka 2-8. I/O moduly Series 90-30 – pokračování

Katalogové číslo	Body	Popis	Číslo publikace
		Diskrétní moduly - výstupy	
IC693MDL310	12	120 VAC, 0.5A	GFK-0898
IC693MDL330	8	120/240 VAC, 2A	GFK-0898
IC693MDL340	16	120 VAC, 0.5A	GFK-0898
IC693MDL390	5	120/240 V ss oddělený, 2A	GFK-0898
IC693MDL730	8	12/24 V ss kladná logika, 2A	GFK-0898
IC693MDL731	8	12/24 VDC záporná kladná, 2A	GFK-0898
IC693MDL732	8	12/24 V ss kladná logika, 0.5A	GFK-0898
IC693MDL733	8	12/24 V ss záporná logika, 0.5A	GFK-0898
IC693MDL734	6	125 V ss kladná/záporná logika, 2A	GFK-0898
IC693MDL740	16	12/24 V ss kladná logika, 0.5A	GFK-0898
IC693MDL741	16	12/24 V ss záporná logika, 0.5A	GFK-0898
IC693MDL742	16	12/24 V ss kladná logika, 1A	GFK-0898
IC693MDL750	32	12/24 V ss záporná logika	GFK-0898
IC693MDL751	32	12/24 V ss kladná logika, 0.3A	GFK-0898
IC693MDL752	32	5/24 V ss (TTL) záporná logika, 0.5A	GFK-0898
IC693MDL753	32	12/24 V ss kladná/záporná logika, 0.5A	GFK-0898
IC693MDL930	8	Relé, kladná logika, 4A oddělené	GFK-0898
IC693MDL931	8	Relé, záporná logika a Form C, oddělené	GFK-0898
IC693MDL940	16	Relé, kladná logika, 2A	GFK-0898
		Vstupní/výstupní moduly	
IC693MDR390	8/8	24 V ss vstup, relé výstup	GFK-0898
IC693MAR590	8/8	120 V stř. vstup, relé výstup	GFK-0898
		Analogové moduly	
IC693ALG220	4 kan.	Analogový vstup, napětí	GFK-0898
IC693ALG221	4 kan.	Analogový vstup, proud	GFK-0898
IC693ALG222	16	Analogový vstup, napětí	GFK-0898
IC693ALG223	16	Analogový vstup, proud	GFK-0898
IC693ALG390	2 kan.	Analogový výstup, napětí	GFK-0898
IC693ALG391	2 kan.	Analogový výstup, proud	GFK-0898
IC693ALG392	8 kan.	Analogový výstup, proud/napětí	GFK-0898
IC693ALG442	4/2	Analogový, vstup/výstup kombinace proud/napětí	GFK-0898

Tabulka 2-8. I/O moduly Series 90-30 – pokračování

Katalogové číslo	Popis	Číslo publikace
	Přídavné moduly	
IC693APU300	Vysokorychlostní čítač	GFK-0293
IC693APU301	Power Mate APM Modul, 1 osa – vlečný režim	GFK-0781
IC693APU301	Power Mate APM Modul, 1 osa – standardní režim	GFK-0840
IC693APU302	Power Mate APM Modul, 2 osy – vlečný režim	GFK-0781
IC693APU302	Power Mate APM Modul, 2 osy – standardní režim	GFK-0840
IC693MCS001/2	Power Mate J Systém řízení pohybu (1 a 2 osy)	GFK-1256
IC693APU305	Modul I/O procesoru	GFK-1028
IC693CMM321	Modul komunikace Ethernet	GFK-1084
IC693ADC311	Koprocesor alfanumerického displeje	GFK-0521
IC693BEM331	Řadič sběrnice Genius	GFK-1034
IC693BEM320	Modul I/O Link Interface (slave)	GFK-0631
IC693BEM321	Modul I/O Link Interface (master)	GFK-0823
IC693CMM311	Modul koprocesoru komunikace	GFK-0582
IC693CMM301	Modul komunikace Genius	GFK-0412
IC693CMM302	Modul rozšířené komunikace Genius	GFK-0695
IC693PCM300	PCM, 160KBytů (35KBytů uživatelský program MegaBasic)	GFK-0255
IC693PCM301	PCM, 192KBytů (47KBytů uživatelský program MegaBasic)	GFK-0255
IC693PCM311	PCM, 640KBytů (190KBytů uživatelský program MegaBasic)	GFK-0255

Formáty I/O Dat

Diskrétní vstupy a diskrétní výstupy se ukládají jako bity do bitové rychlé vyrovnávací paměti (tabulka stavů). Data analogového vstupu a analogového výstupu jsou uložena jako slova a jsou uložena v části aplikační paměti RAM vyhrazené pro tento účel.

Výchozí stavy pro výstupní moduly Series 90-30

Při zapínání diskrétní výstupní moduly Series 90-30 přejdou do výchozího stavu vypnutého výstupu. Tento výchozí stav si podrží do doby, než PLC provede první zápis na výstup. Analogové výstupní moduly je možno nakonfigurovat pomocí zkratovací propojky umístěné na demontovatelných svorkovnicích modulů buď na výchozí nulu nebo tak, že si ponechají svůj poslední stav. Analogové výstupní moduly také mohou být napájené z externího zdroje tak, že i když PLC bude bez napájení, analogové výstupní moduly budou i nadále pracovat ve svém zvoleném výchozím stavu.

Diagnostická data

V paměti %S jsou diagnostické bity, které indikují odebrání I/O modulu nebo nesouhlas v konfiguraci I/O. Pro jednotlivé I/O body diagnostické informace nejsou. Více informací o zpracování chyb najdete v kapitole 3, “Vysvětlivky a oprava chyb”.

Globální Data

Globální data Genius

PLC Series 90-30 podporuje velmi rychlé sdílení dat mezi několika CPU pomocí globálních dat Genius. Řadič sběrnice Genius, IC693BEM331 v CPU, verze 5 a pozdější a modul rozšířené komunikace Genius, IC693CMM302, mohou do jiných PLC nebo počítačů vysílat až 128 bytů dat. Mohou přijímat až 128 bytů od každého z až 30 jiných řadičů Genius na síti. Data je možno vysílat z nebo přijímat do libovolného typu paměti nejen jako globální bity %G. Původní modul komunikace Genius, IC693CMM301, je omezený na pevné adresy %G a může předávat pouze 32 bitů po sériové sběrnici na adresách od SBA 16 do 23. Tento modul by se neměl používat, protože rozšířený GCM má schopnost 100-krát větší.

Globální data je možno sdílet mezi PLC Series Five, Series Six a Series 90 připojených ke stejné I/O sběrnici Genius.

Komunikace Ethernet

CPU model 364 (verze 9.0 a pozdější) podporuje připojení k síti Ethernet přes jeden (ale ne oba) ze dvou vestavěných Ethernet portů. Je možno použít porty AAUI a 10BaseT. CPU model 364 (verze 9.10 nebo pozdější) je jediné CPU Series 90-30, které podporuje EGD.

CPU model 364 podporuje globální data Ethernet (EGD), která jsou podobná jako globální data Genius v tom, že umožní, aby jedno zařízení (vysílač) přenášel data do jednoho nebo více zařízení (přijímače) na síti. EGD není podporováno softwarem Logicmaster 90 (vyžaduje programovací zařízení na bázi Windows pro PLC Series 90).

I/O moduly model 20

Pro PLC Series 90-20 je možno použít následující I/O moduly. Každý modul je uvedený podle svého katalogového čísla, počtu I/O míst a krátkého popisu. I/O je společně se zdrojem napájení součástí základní desky. Specifikace a informace k zapojení jednotlivých modelů najdete v kapitole 5 v Uživatelském manuálu programovatelných řídicích automatů *Series 90-20* GFK-0551.

Katalogové číslo	Popis	I/O body
IC692MAA541	I/O a základní modul napájení, 120 V stř. vstup/120 V stř. výstup/120 V stř. napájení	16 vstupů/12 výstupů
IC692MDR541	I/O a základní modul napájení, 24 V ss vstup/Relé výstup/120 V stř. napájení	16 vstupů/12 výstupů
IC692MDR741	I/O a základní modul napájení, 24 V ss vstup/Relé výstup/240 V stř. napájení	16 vstupů/12 výstupů
IC692CPU211	CPU modul, model CPU 211	Nepoužívá se

Konfigurace a programování

Konfigurace je proces přiřazování logických adres a ostatních charakteristik hardwarovým modulům v systému. To je možno provádět buď před nebo po naprogramování pomocí konfiguračního softwaru nebo ručního programovacího zařízení; doporučuje se však nejdříve provést konfiguraci. Pokud to tak neprovedete, podle *Návodu pro použití programovacího softwaru*, GFK-0466, rozhodněte, jestli je lepší začít s programováním nyní.

Programování se skládá z vytvoření aplikačního programu pro PLC. Protože PLC Series 90-30, 90-20 a Series 90 Micro mají společný soubor instrukcí, všechny tři je možno naprogramovat pomocí softwaru Logicmaster 90-30. Kapitoly 4 až 12 popisují programovací instrukce, které je možno použít při vytváření programů žebříkové logiky pro programovatelné automaty Series 90-30 a Series 90-20.

Pokud programovací software Logicmaster 90-30/20/Micro ještě nebude nainstalovaný, postupujte prosím podle instrukcí v *Návodu pro použití programovacího softwaru*, GFK-0466. Návod pro použití vysvětluje, jak programy vytvářet, přenášet, editovat a vytisknout.

Kapitola 3

Výklad a oprava chyb

Tato kapitola pomáhá při lokalizaci chyb PLC systémů Series 90-30, 90-20 and Micro. Uvádí popis chyb, které se objevují v tabulce chyb PLC, a kategorie chyb, které se mohou objevit v tabulce chyb I/O.

Každý výklad chyby v této kapitole uvádí popis chyby pro tabulku chyb PLC nebo kategorii chyby pro tabulku chyb I/O. Popis chyby nebo kategorii chyby odpovídající zápisu najdete v příslušné tabulce chyb zobrazené na obrazovce programovacího zařízení. Pod ní je popis příčiny chyby společně s instrukcemi, jak chybu odstranit.

Kapitola 3 obsahuje následující části:

Část 1: Zpracování chyby

Poznámka

Tyto informace o zpracování chyby platí pro systémy naprogramované pomocí softwaru Logicmaster 90-30/20/Micro.

Chyby se u systému PLC Series 90-30, PLC Series 90-20 nebo PLC Micro objeví, když se vyskytnou určité chyby nebo stavy, které budou mít vliv na činnost a výkon systému. Tyto stavy, jako například odebrání I/O modulu nebo sestavy, mohou mít vliv na schopnost PLC řídit stroj nebo proces. Tyto stavy mohou mít také příznivý vliv, například když se připojí nový modul a bude pak k dispozici pro použití. Nebo tyto stavy mohou působit pouze jako výstraha, například signál nízkého napětí baterie jako indikace, že je nutno vyměnit baterii chránící paměť.

Alarmový procesor

Podmínka nebo porucha samotná se nazývá chyba. Když do CPU přijde a zpracuje se chyba, tento stav se nazývá alarm. Software v CPU, který tento stav zpracovává, se nazývá alarmový procesor. Rozhraní s uživatelem pro alarmový procesor se realizuje přes programovací software. Každá zjištěná chyba se zaznamená do tabulky chyb a zobrazí se buď na obrazovce tabulky chyb PLC případně na obrazovce tabulky chyb I/O.

Třídy chyb

PLC Series 90-30, 90-20 a Micro detekují několik tříd chyb. Jsou to interní poruchy, externí poruchy a provozní poruchy.

Třída chyb	Příklady
Interní poruchy	Moduly neodpovídající Nízké napětí baterie Chyba kontrolního součtu baterie
Externí poruchy I/O	Odebrání sestavy nebo modulu Přidání sestavy nebo modulu
Provozní poruchy	Porucha komunikace Porucha konfigurace Chybně zadané přístupové heslo

Poznámka

Informace týkající se zpracování chyb u PLC Micro najdete v *Návodu pro použití PLC Series 90 Micro* (GFK-1065).

Reakce systému na chyby

Poruchy hardware vyžadují, aby se systém buď zastavil nebo se porucha tolerovala. Poruchy I/O může PLC systém tolerovat, ale nemusí je tolerovat aplikace nebo řízený proces. Provozní poruchy se normálně tolerují. Chyby PLC Series 90-30, 90-20 a Micro mají dva atributy:

Atribut	Popis
Vliv na tabulku chyb	Tabulka chyb I/O Tabulka chyb PLC
Závažnost chyby	Fatální Diagnostická Informační

Tabulky chyb

V PLC se pro záznam chyb udržují dvě tabulky, tabulka chyb I/O pro záznam chyb týkajících se systému I/O a tabulka chyb PLC pro záznam všech ostatních chyb. V následující tabulce jsou uvedené skupiny chyb, závažnost chyby, vliv na tabulku chyb a "název" ovlivněných diskrétních bodů %S.

Tabulka 3-1. Přehled chyb

Chybová skupina	Závažnost chyby	Tabulka chyb	Speciální adresy diskrétních chyb			
Ztráta nebo chybějící I/O modul	Diagnostická	I/O	io_flt	any_flt	io_pres	los_iom
Ztráta nebo chybějící přídavný modul	Diagnostická	PLC	sy_flt	any_flt	sy_pres	los_sio
Nesoulad konfigurace systému	Fatální	PLC	sy_flt	any_flt	sy_pres	cfg_mm
Hardwarová porucha CPU PLC	Fatální	PLC	sy_flt	any_flt	sy_pres	hrd_cpu
Chyba kontrolního součtu programu	Fatální	PLC	sy_flt	any_flt	sy_pres	pb_sum
Nízké napětí baterie	Diagnostická	PLC	sy_flt	any_flt	sy_pres	low_bat
Tabulka chyb PLC plná	Diagnostická	—	sy_full			
Tabulka chyb I/O plná	Diagnostická	—	io_full			
Chyba aplikace	Diagnostická	PLC	sy_flt	any_flt	sy_pres	apl_flt
Chybí uživatelský program	Informační	PLC	sy_flt	any_flt	sy_pres	no_prog
Zničená uživatelská RAM	Fatální	PLC	sy_flt	any_flt	sy_pres	bad_ram
Chyba přístupového hesla	Diagnostická	PLC	sy_flt	any_flt	sy_pres	bad_pwd
Porucha softwaru PLC	Fatální	PLC	sy_flt	any_flt	sy_pres	sft_cpu
Porucha ukládání PLC	Fatální	PLC	sy_flt	any_flt	sy_pres	stor_er
Překročení konstantního času cyklu	Diagnostická	PLC	sy_flt	any_flt	sy_pres	ov_swp
Neznámá chyba PLC	Fatální	PLC	sy_flt	any_flt	sy_pres	
Neznámá chyba I/O	Fatální	I/O	io_flt	any_flt	io_pres	

Závažnost chyby

Chyby mohou být fatální, diagnostické nebo informativní.

Fatální chyby se zaznamenávají do příslušné tabulky, nastaví se všechny diagnostické proměnné a systém se zastaví. Diagnostické chyby se zaznamenají do příslušné tabulky a všechny diagnostické proměnné se nastaví. Informační chyby se pouze zaznamenají do příslušné tabulky.

Možné kroky při chybách jsou uvedené v následující tabulce.

Tabulka 3-2. Chybové akce

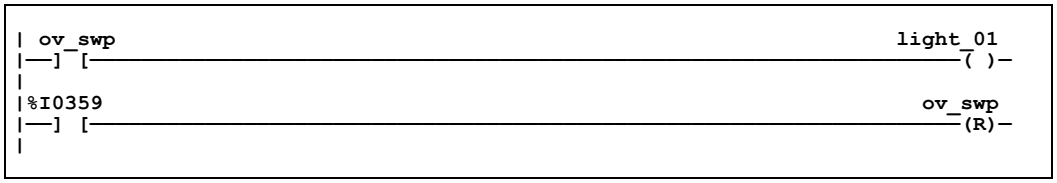
Závažnost chyby	Odezva CPU
Fatální	Záznam chyby do tabulky chyb Nastavení adres chyb Přechod do režimu **STOP**.
Diagnostická	Záznam chyby do tabulky chyb Nastavení adres chyb
Informační	Záznam chyby do tabulky chyb

Když se zjistí chyba, CPU použije chybovou akci odpovídající této chybě. Chybové akce nelze u PLC Series 90-30 PLC, Series 90-20 nebo Series 90 Micro nakonfigurovat.

Adresy chyb

Adresy chyb u Series 90-30 jsou jednoho typu, souhrnné adresy chyb. Souhrnné adresy chyb se nastaví jak indikace, k jaké chybě došlo. Adresa chyby zůstane nastavená, dokud se PLC nevynuluje nebo se nesmaže aplikačním programem.

V následujícím příkladu je uvedený chybový bit, který se nastaví a pak smaže. V tomto příkladu se sepnula cívka light_01, když se vyskytnul stav překročení času cyklu; světlo a kontakt OV_SWP zůstanou sepnuté, dokud se nesepne kontakt %I0359.



Definice adresy chyby

Procesor alarmu udržuje stavy 128 systémových diskrétních bitů v paměti %S. Tyto adresy chyb je možno použít k indikaci, kde došlo k chybě a o jaký typ chyby jde. Adresy chyb jsou přiřazené pamětem %S, %SA, %SB a %SC a každá z těchto pamětí má svou zkratku. Tyto adresy je možno použít podle potřeby v aplikačním programu. Seznam systémových stavových adres najdete v kapitole 2, “Činnost systému”.

Další důsledky chyb

Dvě výše popsané chyby mají další důsledky, které s nimi souvisejí. Jsou popsané v následující tabulce.

Vedlejší vliv	Popis
Porucha softwaru CPU PLC	Když se provede záznam poruchy softwaru CPU PLC, CPU Series 90-30 nebo 90-20 provede okamžitě přechod do režimu **CHYBOVÉHO CYKLU**. V tento režimu není povolena žádná činnost. Jediný způsob, jak tento stav vynulovat, je provést reset PLC vypnutím a zapnutím napájení.
Porucha sekvence ukládání PLC	Pokud během sekvence ukládání (uložení programových bloků a ostatních dat, před kterým byl speciální povel Start-of-Sequence a za kterým následoval povel End-of-Sequence) došlo k přerušení komunikace s programovacím zařízením provádějícím ukládání nebo došlo k jiné poruše, která ukončila načítání, zaznamená se chyba sekvence ukládání PLC. Pokud tato chyba bude v systému, PLC nepřejde do režimu **RUN**.

Zobrazení tabulky chyb PLC

Tabulka chyb PLC zobrazuje chyby PLC, jako například chyba hesla, nesoulad PLC/konfigurace, chyba parity a chyba komunikace.

Programovací software může být v libovolném režimu provozu. Pokud programovací software bude v režimu **OFFLINE**, nebudou se zobrazovat žádné chyby. V režimu **ONLINE** nebo **MONITOR** se budou zobrazovat chyby PLC. V režimu **ONLINE** je možno chyby smazat (to však může být chráněno heslem).

Po smazání se chyby, které přetrvávají, nezapíší znovu do tabulky (s výjimkou chyby "Nízké napětí baterie").

Zobrazení tabulky chyb I/O

Tabulka chyb I/O zobrazuje chyby I/O, jako například chybu obvodu, konflikt adres, vynucené obvody a chyby I/O sběrnice.

Programovací software může být v libovolném režimu provozu. Pokud programovací software bude v režimu **OFFLINE**, nebudou se zobrazovat žádné chyby. V režimu **ONLINE** nebo **MONITOR** se budou zobrazovat chyby I/O. V režimu **ONLINE** je možno chyby smazat (to však může být chráněno heslem).

Po smazání se chyby, které přetrvávají, nezapíší znovu do tabulky.

Přístup k doplňkovým informacím

Tabulky chyb obsahují základní informace o dané chybě. Doplňkové informace týkající se jednotlivých chyb je možno zobrazit pomocí programovacího softwaru. Kromě toho, programový software může provést hexadecimální výpis chyb.

Poslední položka, Oprava, u každého výkladu chyby v této kapitole uvádí krok (kroky), které je nutno provést k odstranění chyby. Všimněte si, že opravný krok u některých chyb zahrnuje příkaz.

Zobrazte tabulku chyb PLC na programovacím zařízení. Spojte se se servisem GE Fanuc a udejte veškeré informace uvedené v popisu chyby.

Tento druhý příkaz znamená, že servisu musíte sdělit jak informaci, kterou si přečtete přímo v tabulce chyb, **tak i** hexadecimální informaci. Pracovník servisu pak vám poskytne další instrukce ohledně, kroků, které musíte provést.

Část 2: Tabulka s výkladem chyb PLC

Každý výklad chyby obsahuje popis chyby a instrukce k jejímu odstranění. Mnoho popisů chyby má více příčin. U těchto příčin se chybový kód zobrazený s doplňkovými informacemi chyby používá k rozlišení různých chybových stavů, které sdílejí stejný popis chyby. Chybový kód jsou první dvě hexadecimální číslice v páté skupině čísel, jak je uvedeno v následujícím příkladu.

```
01   000000   01030100   0902   0200   000000000000
                                 |
                                 |_____  Chybový kód (první dvě hexadecimální
                                             číslice v páté skupině)
```

Některé chyby se mohou vyskytnout proto, že se nepodařil náhodný přístup k paměti v CPU kartě PLC. Tyto stejné chyby se mohou také vyskytnout proto, že systém byl vypnutý a napětí baterie je (nebo bylo) příliš nízké na udržení obsahu paměti. Aby nedocházelo k duplicitě instrukcí, když příčinou chyby může být poškozený obsah paměti, v opravném kroku se prostě sdělí:

Perform the corrections for Corrupted Memory.
(Proveďte opravné kroky pro poškozený obsah paměti.)

To znamená:

1. Pokud systém byl vypnutý, vyměňte baterii. Napětí baterie může být nedostatečné k udržení obsahu paměti.
2. Vyměňte CPU desku PLC. Integrovaný obvod na CPU desce PLC může být vadný.

Následující tabulka vysvětluje, jako rychle najít konkrétní popis PLC chyby v této kapitole. Každá položka je uvedena tak, jak se objeví na obrazovce programovacího zařízení.

Chybové akce

Fatální chyby mají za následek, že PLC přejde do režimu **STOP** na konci cyklu, ve kterém se chyba vyskytla. **Diagnostické** chyby se zaznamenají a nastaví se odpovídající chybové kontakty. **Informační** chyby se prostě jen zaznamenají do tabulky chyb PLC.

Ztráta nebo chybějící přídavný modul

Chybová skupina **Ztráta nebo chybějící přídavný modul** se vyskytne, když PCM, CMM nebo ADC není schopný odpovídat. Porucha se může vyskytnout při zapínání napájení, když modul bude chybět nebo když během operace modul nebude schopný odpovídat. Závažnost chyby u této skupiny je **Diagnostická**.

Chybový kód:	1, 42
Název:	Neprovedl se softwarový reset přídavného modulu
Popis	CPU PLC není schopno po softwarovém resetu obnovit komunikaci s přídavným modulem.
Oprava:	(1) Zkuste provést softwarový reset znovu. (2) Vyměňte přídavný modul. (3) Vypněte napájení systému. Ověřte, že deska PCM je v sestavě správně zasazená a že všechny kabely jsou správně zapojené a usazené. (4) Vyměňte kabely.
Chybový kód:	Všechny ostatní
Název:	Porucha modulu během konfigurace
Popis	Operační software PLC generuje tuto chybu, když modul nebude během zapnutí napájení nebo konfigurace schopný provést uložení.
Oprava:	(1) Vypněte napájení systému. Vyměňte modul umístěný v této sestavě a slotu.

Reset, přidání nebo přespočetný přídavný modul

Chybová skupina **Reset, přidání nebo přespočetný přídavný modul** se vyskytne, když přídavný modul (PCM, ADC, atd.) bude on-line, bude resetovaný nebo se v sestavě zjistí modul, ale v konfiguraci specifikovaný není. Závažnost chyby u této skupiny je **Diagnostická**. Tři byty dat dané chyby uvádějí doplňující informace týkající se chyby.

Oprava:	(1) Proveďte aktualizaci konfiguračního souboru tak, aby zahrnoval modul. (2) Odstraňte modul ze systému.

Nesoulad konfigurace systému

Chybová skupina **Nesoulad konfigurace** se objeví, když modul ve slotu bude jiný než ten, který je specifikovaný v konfiguračním souboru. Závažnost chyby u této skupiny je **Fatální**.

Chybový kód:	1
Název:	Nesoulad konfigurace systému
Popis	Operační software PLC (konfigurátor systému) vygeneruje tuto chybu, když modul ve slotu nebude stejného typu, který by podle konfiguračního souboru měl být, nebo když nakonfigurovaný typ sestavy nebude souhlasit se skutečnou sestavou.
Oprava:	Identifikujte nesoulad a proveďte novou konfiguraci modulu nebo sestavy.
Chybový kód:	6
Název:	Nesoulad konfigurace systému
Popis	To je totéž jako chybový kód 1 v tom, že tato chyba se vyskytne, když modul ve slotu nebude stejného typu, který by podle konfiguračního souboru měl být, nebo když nakonfigurovaný typ sestavy nebude souhlasit se skutečnou sestavou.
Oprava:	Identifikujte nesoulad a proveďte novou konfiguraci modulu nebo sestavy.
Chybový kód:	18
Název:	Nepodporovaný hardware
Popis	V CPU 311, 313 nebo 323 nebo v přípojném roštu se nachází PCM nebo modul typu PCM.
Oprava:	Fyzicky opravte stav odstraněním PCM nebo modulu typu PCM nebo nainstalujte CPU, které podporuje PCM.
Chybový kód:	26
Název:	Modul je zaneprázdněný . modul zatím nepřijal konfiguraci
Popis	Modul není schopný v tomto okamžiku přijmout novou konfiguraci, protože je zaneprázdněný jiným procesem.
Oprava:	Nechejte modul dokončit současnou operaci a konfiguraci uložte znovu.
Chybový kód:	51
Název:	Funkce END se vykonala z akce v SFC
Popis	Tato chyba se vytvoří umístěním funkce END do logiky SFC nebo do logiky vyvolané pomocí SFC.
Oprava:	Odstraňte funkce END z logiky SFC nebo logiky, která je vyvolávána logikou SFC.

Chyba softwaru přídavného modulu

Chybová skupina **Chyba softwaru přídavného modulu** se vyskytne, když v modulu PCM nebo ADC se vyskytne neopravitelná chyba softwaru. Závažnost chyby u této skupiny je **Fatální**.

Chybový kód:	Všechny
Název:	Četnost COMMREQ je příliš vysoká
Popis	Požadavky COMMREQ se do modulu vysílají rychleji, než je stačí zpracovat.
Oprava:	Změňte PLC program tak, aby posílal požadavky COMMREQ do příslušného modulu pomaleji.

Chyba kontrolního součtu programového bloku

Chybová skupina **Chyba kontrolního součtu programového bloku** se vyskytne, když CPU PLC zjistí chybový stav v programových blocích, které přijdou do PLC. Také se objeví, když CPU PLC zjistí chybu kontrolního součtu během ověřování při zapínání napájení nebo během kontroly na pozadí v režimu **RUN**. Závažnost chyby u této skupiny je **Fatální**.

Chybový kód:	Všechny
Název:	Chyba kontrolního součtu programového bloku
Popis	Operační software PLC generuje tuto chybu, když dojde k poškození obsahu programového bloku.
Oprava:	(1) Vynulujte PLC paměť a zkuste uložení znovu. (2) Zobrazte tabulku chyb PLC na programovacím zařízení. Spojte se se servisním střediskem GE Fanuc pro PLC a udejte všechny informace obsažené v popisu chyby.

Signál nízkého napětí baterie

Chybová skupina **Signál nízkého napětí baterie** se vyskytne, když CPU PLC zjistí nízké napětí baterie na napájení PLC nebo modulu, například PCM a bude hlásit stav nízkého napětí baterie. Závažnost chyby u této skupiny je **Diagnostická**.

Chybový kód:	0
Název:	Signál chybné baterie
Popis	Baterie modulu CPU (nebo jiného modulu, který má baterii) je špatná.
Oprava:	Vyměňte baterii. Nevypínejte napájení sestavy.
Chybový kód:	1
Název:	Signál nízkého napětí baterie
Popis	Baterie na CPU nebo jiném modulu má nízké napětí.
Oprava:	Vyměňte baterii. Nevypínejte napájení sestavy.

Překročení konstantní doby cyklu

Chybová skupina **Překročení konstantní doby cyklu** se objeví, když CPU PLC bude pracovat v režimu **KONSTANTNÍ CYKLUS** a zjistí, že cyklus překročil konstantní dobu cyklu. Přídavná data chyby obsahují skutečnou dobu cyklu v prvních dvou bytech a název programu v dalších osmi bytech. Závažnost chyby u této skupiny je **Diagnostická**.

Oprava:	(1) Prodlužte konstantní dobu cyklu.
	(2) Odstraňte logiku z aplikačního programu.

Chyba aplikace

Chybová skupina **Chyba aplikace** se vyskytne, když CPU PLC zjistí chybu v uživatelském programu. Závažnost chyby pro tuto skupinu je **Diagnostická**, s výjimkou když chyba bude Přetečení vyrovnávací paměti volání podprogramu, kdy tato chyba bude **Fatální**.

Chybový kód:	7
Název:	Přetečení vyrovnávací paměti volání podprogramu
Popis	Volání podprogramu je omezeno do hloubky 8. Podprogram může volat jiný podprogram, který opět může volat jiný podprogram, až se dosáhne 8 úrovní volání.
Oprava:	Upravte program tak, aby hloubka volání podprogramu nepřesáhla 8 úrovní.
Chybový kód:	1B
Název:	CommReq se nezpracoval z důvodu omezení PLC paměti
Popis	Požadavky na komunikaci bez čekání je možno do fronty zařazovat rychleji než je možné je zpracovávat, (například jednou za cyklus). V této situaci, když požadavky na komunikaci narostou do takového stavu, že PLC má k použití méně než minimální objem paměti, může požadavek na komunikaci být odmítnutý a nezpracuje se.
Oprava:	Zadávejte méně požadavků na komunikaci, nebo jinak snižte objem zpráv, které se v systému posílají.
Chybový kód:	5A
Název:	Vypnutí požadováno uživatelem
Popis	Operační software PLC (funkční bloky) vygeneruje tento informační alarm, když se v aplikačním programu vykoná požadavek na obsloužení #13.
Oprava:	Nevyžaduje se. Pouze informační alarm.

Chybí uživatelský program

Chybová skupina **Chybí uživatelský program** se vyskytne, když CPU PLC dostane instrukci k přechodu z režimu **STOP** do režimu **RUN** nebo k uložení do PLC a v PLC nebude žádný uživatelský program. CPU PLC zjistí absenci uživatelského programu při zapínání napájení. Závažnost chyby pro tuto skupinu je **Informační**.

Oprava:	Nahrajte aplikační program před tím, než se budete snažit přejít do režimu **RUN**.

Poškozený uživatelský program při zapínání

Chybová skupina **Poškozený uživatelský program při zapínání** se objeví, když CPU PLC zjistí poškození obsahu uživatelské RAM. CPU PLC zůstane v režimu **STOP**, dokud nebude načtený platný program a konfigurační soubor. Závažnost chyby u této skupiny je **Fatální**.

Chybový kód:	1
Název:	Poškozený obsah uživatelské RAM při zapnutí napájení
Popis	Operační software PLC generuje tuto chybu, když při zapnutí zjistí poškození obsahu uživatelské RAM.
Oprava:	(1) Načtěte znovu konfigurační soubor, uživatelský program a adresy (pokud existují). (2) Vyměňte CPU baterii na PLC. (3) Vyměňte kartu přídavné paměti v CPU PLC. (4) Vyměňte CPU PLC.
Chybový kód:	2
Název:	Zjištěný nepřípustný Booleovský operační kód.
Popis	Operační software PLC vygeneruje tuto chybu, když zjistí špatnou instrukci v uživatelském programu.
Oprava:	(1) Obnovte uživatelský program a adresy (pokud existují). (2) Vyměňte kartu přídavné paměti v CPU PLC. (3) Vyměňte CPU PLC.

Chyba přístupového hesla

Chybová skupina **Chyba přístupového hesla** se vyskytne, když CPU PLC obdrží požadavek na změnu úrovně oprávnění a heslo uvedené v požadavku není platné pro tuto úroveň. Závažnost chyby pro tuto skupinu je **Informační**.

Oprava:	Zkuste požadavek znovu se správným heslem.

Chyba systémového softwaru CPU PLC

Chyby v chybové skupině **Chyba systémového softwaru CPU PLC** generuje operační software CPU PLC Series 90-30, 90-20 nebo Micro. Mohou se vyskytnout na mnoha různých místech činnosti systému. Když se vyskytne **Fatální** chyba, CPU PLC **přechodně** přejde do zvláštního režimu **CHYBOVÉHO CYKLU**. Když je PLC v tomto režimu, není přípustná žádná činnost. Jediný způsob, jak tento stav zrušit, je vypnout a zapnout napájení PLC. Závažnost chyby u této skupiny je **Fatální**.

Chybový kód:	1 až B
Název:	Nelze přiřadit uživatelskou paměť
Popis	Operační software PLC (správce paměti) vygeneruje tyto chyby, když software bude požadovat, aby správce paměti přiřadil nebo zrušil přiřazení bloku nebo bloků paměti z uživatelské RAM, které jsou nepřípustné. Tyto chyby se ve výrobním systému *nesmí* vyskytnout.
Oprava:	Zobrazte tabulku chyb PLC na programovacím zařízení. Spojte se se servisním střediskem GE Fanuc pro PLC a udejte všechny informace obsažené v popisu chyby.
Chybový kód:	D
Název:	Systémová paměť není k dispozici
Popis	Operační software PLC (I/O Scanner) vygeneruje tuto chybu, když správce paměti jeho požadavek na blok systémové paměti odmítne, protože je nedostatek systémové paměti. Je to *Informační* chyba, pokud se chyba vyskytne během vykonávání funkčního bloku DO I/O. Je to *Fatální* chyba, pokud se chyba vyskytne během inicializace při zapínání napájení nebo při autokonfiguraci.
Oprava:	Zobrazte tabulku chyb PLC na programovacím zařízení. Spojte se se servisním střediskem GE Fanuc pro PLC a udejte všechny informace obsažené v popisu chyby.
Chybový kód:	E
Název:	Systémovou paměť nelze uvolnit
Popis	Operační software PLC (I/O Scanner) vygeneruje tuto chybu, když bude požadovat, aby správce paměti zrušil přiřazení bloku systémové paměti a zrušení se neprovede. Tato chyba se vyskytne pouze během vykonávání funkčního bloku DO I/O.
Oprava:	(1) Zobrazte tabulku chyb PLC na programovacím zařízení. Spojte se se servisním střediskem GE Fanuc pro PLC a udejte všechny informace obsažené v popisu chyby. (2) Proveďte opravu poškozeného obsahu paměti.
Chybový kód:	10
Název:	Neplatný požadavek I/O Scanneru
Popis	Operační software PLC (I/O Scanner) vygeneruje tuto chybu, když operační systém nebo funkční blok snímání DO I/O nepožaduje ani plné ani částečné snímání I/O. Toto se ve výrobním systému *nesmí* vyskytnout.
Oprava:	Zobrazte tabulku chyb PLC na programovacím zařízení. Spojte se se servisním střediskem GE Fanuc pro PLC a udejte všechny informace obsažené v popisu chyby.
Chybový kód:	13
Název:	Chyba operačního softwaru PLC
Popis	Operační software PLC vygeneruje tuto chybu, když nastanou určité problémy s operačním software PLC. Tato chyba se ve výrobním systému *nesmí* vyskytnout.
Oprava:	(1) Zobrazte tabulku chyb PLC na programovacím zařízení. Spojte se se servisním střediskem GE Fanuc pro PLC a udejte všechny informace obsažené v popisu chyby. (2) Proveďte opravu poškozeného obsahu paměti.

Chybový kód:	14, 27
Název:	Poškozený obsah programové paměti PLC
Popis	Operační software PLC vygeneruje tyto chyby, když se vyskytnou určité problémy s operačním softwarem PLC. Tyto chyby se ve výrobním systému *nesmí* vyskytnout.
Oprava:	(1) Zobrazte tabulku chyb PLC na programovacím zařízení. Spojte se se servisním střediskem GE Fanuc pro PLC a udejte všechny informace obsažené v popisu chyby. (2) Proveďte opravu poškozeného obsahu paměti.
Chybový kód:	27 až 4E
Název:	Chyba operačního softwaru PLC
Popis	Operační software PLC vygeneruje tyto chyby, když se vyskytnou určité problémy s operačním software PLC. Tyto chyby se ve výrobním systému *nesmí* vyskytnout.
Oprava:	Zobrazte tabulku chyb PLC na programovacím zařízení. Spojte se se servisním střediskem GE Fanucpro PLC a udejte všechny informace obsažené v popisu chyby.
Chybový kód:	4F
Název:	Chyba komunikace
Popis	Operační software PLC (procesor požadavku služby) vygeneruje tuto chybu, když se bude snažit vyhovět požadavku, který požaduje k interní komunikaci, a dostane zamítavou odezvu.
Oprava:	(1) Zkontrolujte, jestli na sběrnici neprobíhá neobvyklá činnost. (2) Vyměňte inteligentní přídavný modul, do kterého byl požadavek směřovaný.
Chybový kód:	50, 51, 53
Název:	Chyby systémové paměti
Popis	Operační software PLC vygeneruje tyto chyby, když správce paměti jeho požadavek na blok systémové paměti z důvodu nedostatku systémové paměti odmítne.
Oprava:	(1) Zobrazte tabulku chyb PLC na programovacím zařízení. Spojte se se servisním střediskem GE Fanuc pro PLC a udejte všechny informace obsažené v popisu chyby. (2) Proveďte opravu poškozeného obsahu paměti.
Chybový kód:	52
Název:	Chyba interní komunikace
Popis	Operační software PLC (procesor požadavku služby) vygeneruje tuto chybu, když se bude snažit vyhovět požadavku, který požaduje k interní komunikaci, a dostane zamítavou odezvu.
Oprava:	(1) Zkontrolujte, jestli na sběrnici neprobíhá neobvyklá činnost. (2) Vyměňte inteligentní přídavný modul, do kterého byl požadavek směřovaný. (3) Zkontrolujte paralelní kabel programovacího zařízení, jestli je správně připojený.
Chybový kód:	Všechny ostatní
Název:	Interní systémová chyba CPU PLC
Popis	Vyskytla se interní systémová chyba, která se ve výrobním systému **nesmí** objevit.
Oprava:	Zobrazte tabulku chyb PLC na programovacím zařízení. Spojte se se servisním střediskem GE Fanuc pro PLC a udejte všechny informace obsažené v popisu chyby.

Chyba komunikace během ukládání

Chybová skupina **Chyba komunikace během ukládání** se vyskytne během ukládání programových bloků a jejich dat do PLC. Tok povelů a dat pro uložení programových bloků a dat se spustí speciálním povelem start sekvence a ukončí se povelem konec sekvence. Pokud se komunikace s programovacím zařízením, které provádí ukládání, přeruší nebo se vyskytne jiná chyba, která ukládání přeruší, tato chyba se zaznamená. Dokud se bude tato chyba v systému vyskytovat, řídicí jednotka neprovede přechod do režimu **RUN**.

Tato chyba se *nevynuluje* automaticky při zapnutí napájení; uživatel musí zadat konkrétní stav, který se má smazat. Závažnost chyby u této skupiny je **Fatální**.

Oprava:	Smažte chybu a zkuste program nebo konfigurační soubor načíst znovu.

Část 3: Výklad tabulky chyb I/O

Tabulka chyb I/O řadí data chyb do tří skupin:

- Kategorie chyby.
- Typ chyby.
- Popis chyby.

Chyby popsané na následující stránce mají kategorii chyb, ale nemají typ chyby nebo skupinu chyby.

Každý výklad chyby obsahuje popis chyby a instrukce k jejímu odstranění. Mnoho popisů chyby má více příčin. V těchto případech se chybový kód zobrazený s doplňkovými informacemi chyby získaný stisknutím CTRL-F používá k rozlišení různých chybových stavů, které sdílejí stejný popis chyby. (Více informací o používání CTRL-F najdete v Příloze B, "Interpretace tabulek chyb" v tomto manuálu.) Kategorie chyby jsou první dvě hexadecimální číslice v páté skupině čísel, jak je ukázáno v následujícím příkladu.

```
02  1F0100  00030101FF7F  0302  0200   84000000000003
                                |
                                |_____  Kategorie chyby (první dvě
                                       hexadecimální číslice v páté skupině)
```

Následující tabulka vysvětluje, jak rychle najít konkrétní popis I/O chyby v této kapitole. Každá položka je uvedena tak, jak se objeví na obrazovce programovacího zařízení.

Ztráta I/O modulu

Kategorie chyb **Ztráta I/O modulu** platí pro diskrétní a analogové I/O moduly model 30. S touto kategorií nejsou spojené žádné typy chyb nebo popisy chyb. Závažnost chyby u této skupiny je **Diagnostická**.

Popis	Operační software PLC vygeneruje tuto chybu, když zjistí, že I/O modul model 30 neodpovídá na povely z CPU PLC nebo když konfigurační soubor udává, že I/O modul se má vyskytovat ve slotu a v tomto slotu žádný modul není.
Oprava:	(1) Vyměňte modul. (2) Opravte konfigurační soubor. (3) Zobrazte tabulku chyb PLC na programovacím zařízení. Spojte se se servisním střediskem GE Fanuc pro PLC a udejte všechny informace obsažené v popisu chyby.

Přidání I/O modulu

Kategorie chyb **Přidání I/O modulu** platí pro diskrétní analogové moduly I/O model 30. S touto kategorií nejsou spojené žádné typy chyb nebo popisy chyb. Závažnost chyby u této skupiny je **Diagnostická**.

Popis	Operační software PLC vygeneruje tuto chybu, když se modul, který měl poruchu, vrátí do provozu.
Oprava:	(1) Pokud modul byl vyjmutý a zase vrácený nebo bylo provedeno vypnutí a zapnutí vzdálené sestavy, není nutno provádět žádné kroky. (2) Proveďte aktualizaci konfiguračního souboru nebo modul odstraňte.
Popis	Operační software PLC vygeneruje tuto chybu, když zjistí I/O modul model 30 ve slotu, který by podle konfiguračního souboru měl být prázdný.
Oprava:	(1) Odstraňte modul. (Může být ve špatném slotu.) (2) Aktualizujte a obnovte konfigurační soubor tak, aby zahrnoval další modul

Kapitola 4

Reléové funkce

Tato kapitola vysvětluje použití kontaktů, cívek a spojů v příčkách žebříkové logiky.

Používání kontaktů

Kontakt se používá k monitorování stavu adresy. Na podmínkách nebo stavu monitorované adresy a na typu kontaktu závisí, jestli kontaktem bude protékat proud. Adresa bude ve stavu ON, pokud její stav je 1; nebo bude ve stavu OFF, pokud je její stav 0.

Tabulka 4-1. Typy kontaktů

Typ kontaktu	Zobrazení	Kontakt propouští proud doprava.
Normální spínací	—| |—	Když adresa je ve stavu ON.
Normální rozpínací	—|/|—	Když adresa je ve stavu OFF.
Pokračovací kontakt	<+>———	Když předchozí pokračovací cívka bude ve stavu ON.

Používání cívek

Cívky se používají k řízení diskrétních adres. K řízení proudu cívkou se musí použít podmíněná logika. Cívky vyvolají akci přímo; nepřenášejí proud doprava. Pokud se v důsledku stavu cívky má v programu vykonat další logika, musí se pro tuto cívku použít interní adresa nebo se může použít kombinace pokračovací cívky/kontaktu.

Cívky jsou vždy umístěné na spojnici logiky co nejvíce vpravo. Příčka může obsahovat až osm cívek.

Typ použité cívky bude záviset na typu požadované akce programu. Když se provede vypnutí a zapnutí napájení nebo když PLC přejde z režimu **STOP** do režimu **RUN**, stavy retentivních cívek se uloží. Když se provede vypnutí a zapnutí napájení nebo když PLC přejde z režimu **STOP** do režimu **RUN**, stavy neretentivních cívek se nastaví na nulu.

Tabulka 4-2. Typy cívek

Typ cívky	Zobrazení	Proud do cívky	Výsledek
Normálně rozepnutá	—()—	ON	Nastaví adresu na ON.
		OFF	Nastaví adresu na OFF
Negovaná	—(/)—	ON	Nastaví adresu na OFF
		OFF	Nastaví adresu na ON.
Retentivní	—(M)—	ON	Nastaví adresu na ON, retentivní.
		OFF	Nastaví adresu na OFF, retentivní.
Negovaná retentivní	—(/M)—	ON	Nastaví adresu na OFF, retentivní.
		OFF	Nastaví adresu na ON, retentivní.
Pozitivní přechod	—(↑)—	OFF→ON	Pokud adresa bude OFF, nastaví ji na jeden cyklus na ON.
Negativní přechod	—(↓)—	ON←OFF	Pokud adresa bude OFF, nastaví ji na jeden cyklus na ON.
SET	—(S)—	ON	Nastaví adresu na ON, dokud nebude resetovaná na OFF pomocí —(R)—.
		OFF	Nemění stav cívky.
RESET	—(R)—	ON	Nastaví adresu na OFF, dokud nebude nastavená na ON pomocí —(S)—.
		OFF	Nemění stav cívky.
Retentivní SET	—(SM)—	ON	Nastaví adresu na ON, dokud nebude resetovaná na OFF pomocí —(RM)—, retentivní.
		OFF	Nemění stav cívky.
Retentivní RESET	—(RM)—	ON	Nastaví adresu na OFF, dokud nebude nastavená na ON pomocí —(SM)—, retentivní.
		OFF	Nemění stav cívky.
Pokračovací cívka	——<+>	ON	Nastaví další pokračovací kontakt na ON.
		OFF	Nastaví další pokračovací kontakt na OFF.

Normální spínací kontakt —| |—

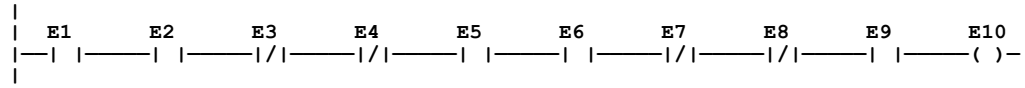
Normální spínací kontakt funguje jako spínač, který propouští proud, pokud související adresa bude ve stavu ON (1).

Normální rozpínací kontakt —|/|—

Normální rozpínací kontakt funguje jako spínač, který propouští proud, pokud související adresa bude ve stavu OFF (0).

Příklad

Následující příklad ukazuje příčku s 10 prvky s názvem E1 až E10. Cívka E10 bude ve stavu ON, když adresy E1, E2, E5, E6 a E9 budou ve stavu ON a adresy E3, E4, E7 a E8 budou ve stavu OFF.

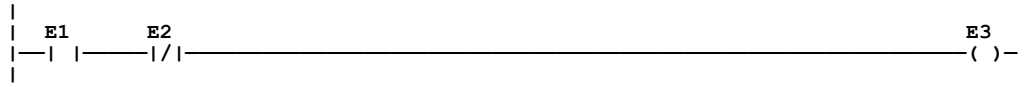


Cívka —( )—

Cívka nastaví diskrétní adresu do stavu ON, když skrz ní teče proud. Je neretentivní; proto jí nelze použít se systémovými stavovými adresami (%SA, %SB, %SC) nebo %G.

Příklad

V následujícím příkladu cívka E3 bude ve stavu ON, pokud adresa E1 bude ve stavu ON a adresa E2 bude ve stavu OFF.

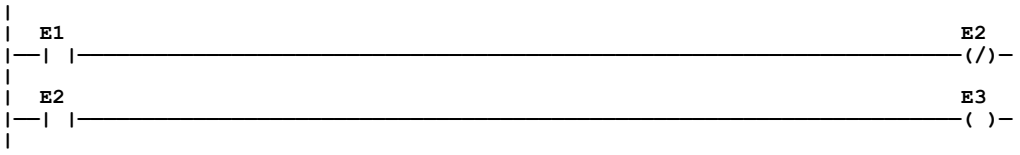


Negovaná cívka —(/)—

Negovaná cívka nastaví diskrétní adresu do stavu ON, pokud skrz ní neteče proud. Je neretentivní; proto jí nelze použít se systémovými stavovými adresami (%SA, %SB, %SC) nebo %G.

Příklad

V následujícím příkladu cívka E3 bude ve stavu ON, pokud adresa E1 bude ve stavu OFF.



Retentivní cívka—(M)—

Stejně jako v případě normálně rozepnuté cívky, retentivní cívka nastaví diskrétní adresu do stavu ON, pokud skrz ní teče proud. Stav retentivní cívky se zachová i při výpadku napájení. Proto ji nelze použít s adresami výhradně z neretentivní paměti (%T).

Negovaná retentivní cívka —(/M)—

Negovaná retentivní cívka nastaví diskrétní adresu do stavu ON, pokud skrz ní neteče proud. Stav negované retentivní cívky se zachová i při výpadku napájení. Proto ji nelze použít s adresami výhradně z neretentivní paměti (%T).

Pozitivní přechodová cívka —(↑)—

Pokud adresa související s pozitivní přechodovou cívkou bude ve stavu OFF a když do cívky poteče proud, nastaví se do stavu ON. Každý kontakt související s touto cívkou změní svůj stav během jednoho čtení (cyklu) PLC. (Pokud příčka obsahující cívku bude při následujících cyklech přeskočena, zůstane ve stavu ON.) Tuto cívku je možno použít jako jednorázovou.

Aby se zachovala jednorázová povaha cívky, každá adresa, pokud je v aplikačním programu, se musí použít jen jako přechodová cívka.

Přechodové cívky je možno použít s adresami z retentivní nebo neretentnivní paměti (%Q, %M, %T, %G, %SA, %SB nebo %SC).

Negativní přechodová cívka —(↓)—

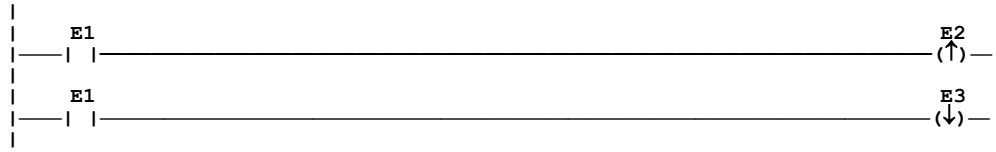
Pokud adresa související s touto cívkou bude ve stavu OFF a když cívkou ***nepoteče*** proud, adresa se nastaví do stavu ON a všechny kontakty související s touto cívkou během jednoho cyklu změní svůj stav.

Aby se zachovala jednorázová povaha cívky, každá adresa, pokud je v aplikačním programu, se musí použít jen jako přechodová cívka.

Přechodové cívky je možno použít s adresami z retentivní nebo neretentnivní paměti (%Q, %M, %T, %G, %SA, %SB nebo %SC).

Příklad

Když v následujícím příkladu, když adresa E1 přejde ze stavu OFF do stavu ON, cívkami E2 a E3 poteče proud a přepne tím E2 do stavu ON po dobu jednoho cyklu. Když E1 přejde ze stavu ON do stavu OFF, proud na E2 a E3 zastaví a po dobu jednoho cyklu se E3 přepne do stavu ON.



Cívka SET —(S) —

SET a RESET jsou neretentivní cívky, které je možno použít k udržení (“zachycení”) stavu adresy (např. E1) ve stavu ON nebo OFF. Když cívkou SET poteče proud, její adresa zůstane ve stavu ON (bez ohledu na to, jestli teče proud samotnou cívkou), dokud adresa nebude resetovaná jinou cívkou.

Cívky SET zapíšou do přechodového bitu pro danou adresu nedefinovaný výsledek. (Viz informace v odstavci “Přechody a přepisy” v kapitole 2, “Činnost systému.”)

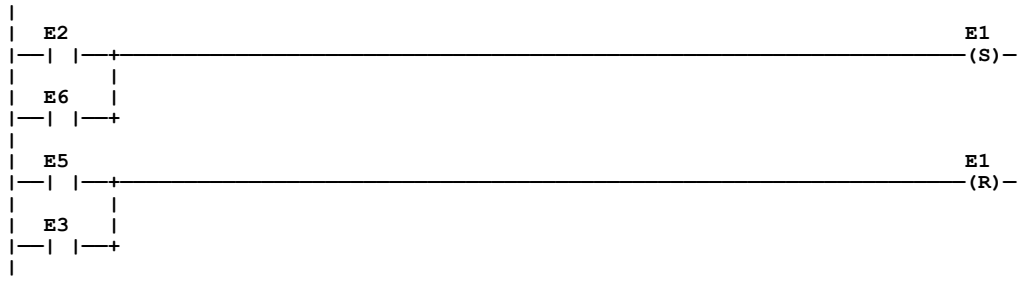
Cívka RESET —(R)—

Když na cívku přijde proud, cívka RESET nastaví diskrétní adresu do stavu OFF. Adresa zůstane ve stavu OFF, dokud adresa nebude resetovaná jinou cívkou. Naposledy nastavená cívka SET nebo cívka RESET páru má přednost.

Cívky RESET zapisují nedefinovaný výsledek do přechodového bitu pro danou adresu. (Viz informace v odstavci “Přechody a přepisy” v kapitole 2, “Činnost systému.”)

Příklad

V následujícím příkladu cívka E1 se přepne do stavu ON vždy, když adresa E2 nebo E6 bude ve stavu ON. Cívka E1 se přepne do stavu OFF vždy, když adresa E5 nebo E3 bude ve stavu ON.



Poznámka

Když úroveň kontroly cívky bude SINGLE, můžete použít specifickou adresu %M nebo %Q pouze s jednou cívkou, ale můžete jí použít současně s jednou cívkou SET a jednou cívkou RESET. Když úroveň kontroly cívky bude WARN MULTIPLE nebo MULTIPLE, pak se každá adresa může použít s více cívkami, cívkami SET a cívkami RESET. V případě vícenásobného použití je možno adresu nastavit do stavu ON buď cívkou SET nebo normální cívkou a přepnout do stavu OFF cívkou RESET nebo normální cívkou.

Retentivní cívka SET —(SM)—

Retentivní cívky SET a RESET jsou podobné cívkám SET a RESET, ale jejich stav zůstává i po vypnutí napájení nebo když PLC přejde z režimu **STOP** do režimu **RUN**. Retentivní cívka SET nastaví diskrétní adresou do stavu ON, pokud skrz ní teče proud. Adresa zůstane ve stavu ON, dokud adresa nebude resetovaná jinou retentivní cívkou RESET.

Retentivní cívky SET zapisují nedefinovaný výsledek do přechodového bitu pro danou adresu. (Viz informace v odstavci “Přechody a přepisy” v kapitole 2, “Činnost systému.”)

Retentivní cívka RESET —(RM)—

Když cívkou poteče proud, tato cívka nastaví diskrétní adresu do stavu OFF. Adresa zůstane ve stavu OFF, dokud adresa nebude nastavená retentivní cívkou SET. Po vypnutí napájení nebo když PLC přejde z režimu **STOP** do režimu **RUN** se stav této cívky zachová.

Retentivní cívky RESET zapisují nedefinovaný výsledek do přechodového bitu pro danou adresu. (Viz informace v odstavci “Přechody a přepisy” v kapitole 2, “Činnost systému.”)

Spoje

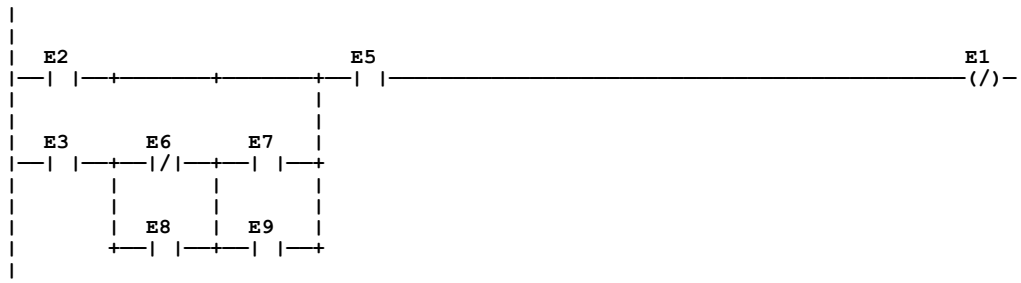
Horizontální a vertikální spoje se používají ke spojení prvků žebříkové logiky mezi funkcemi. Jejich účelem je dokončit tok logiky ("proud") ve schématu logiky zleva doprava.

Poznámka

Horizontální spoj nelze použít k připojení funkce nebo cívky k levé napájecí spojnici. K vyvolání funkce v každém cyklu však je možno použít %S7, systémový bit AWL_ON (vždy v jedničce) s normálně rozepnutým kontaktem připojeným k napájecí sběrnici.

Příklad

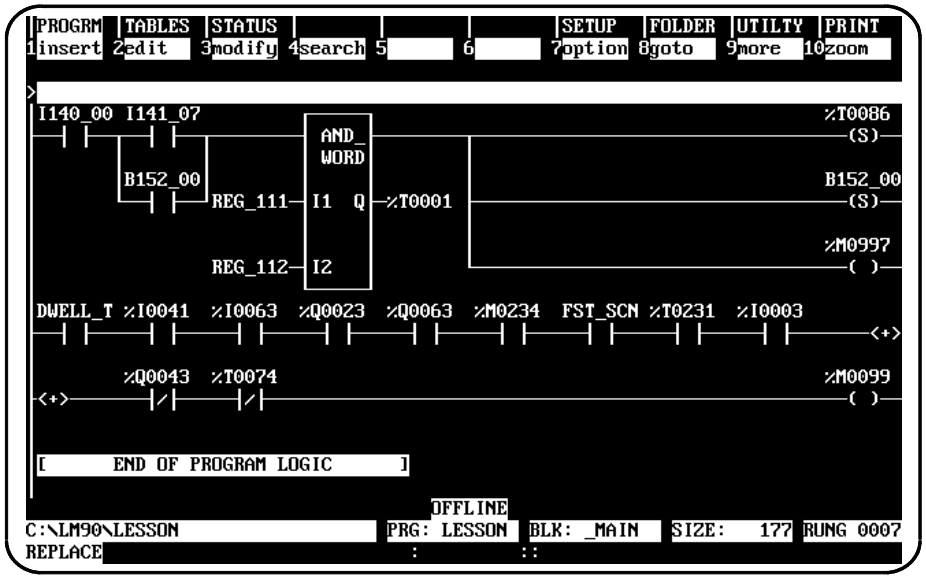
V následujícím příkladu se ke spojení kontaktů E2 a E5 používají dva horizontální spoje. Vertikální spoj se používá k připojení kontaktů E3, E6, E7, E8 a E9 na E2.



Pokračovací cívky (———<+>) a kontakty (<+>———)

Pokračovací cívky (——<+>) a pokračovací kontakty (<+>——–) se používají k pokračování příček reléové žebříkové logiky na více než deseti sloupcích. Stav poslední vykonané pokračovací cívky je stav proudu, který se použije na další vykonávání pokračovacího kontaktu. Před tím, než logika vykoná pokračovací kontakt, musí existovat pokračovací cívka. Stav pokračovacího kontaktu se smaže, když PLC přejde z režimu **Stop** do režimu **Run**, a proud nebude protékat, dokud po přechodu do režimu **Run** nebude nastavena přechodová cívka.

Na jednu příčku může být pouze jedna pokračovací cívka a kontakt; pokračovací kontakt musí být ve sloupci 1 a pokračovací cívka musí být ve sloupci 10. Příklad pokračovací cívky a kontaktu je ukázán na následujícím obrázku:



Kapitola 5

Časovače a čítače

Tato kapitola popisuje, jak použít časovače zpoždění, stopky, vzestupný čítač a sestupný čítač. Data související s touto funkcí jsou při cyklu vypínání a zapínání napájení retentivní.

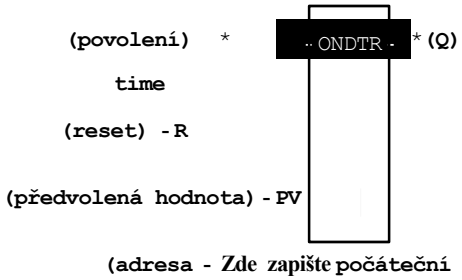
Zkratka	Funkce	Strana
ONDTR	Retentivní časovač zpoždění při zapnutí	5-3
TMR	Jednoduchý časovač zpoždění při zapnutí	5-5
OFDT	Časovač zpoždění při vypnutí	5-8
UPCTR	Vzestupný čítač	5-11
DNCTR	Sestupný čítač	5-12

Data funkčního bloku požadovaná pro časovače a čítače

Každý časovač nebo čítač používá k uložení následujících informací tři slova (registry) paměti %R:

aktuální hodnota (CV)	slovo 1
předvolená hodnota (PV)	slovo 2
řídicí slovo	slovo 3

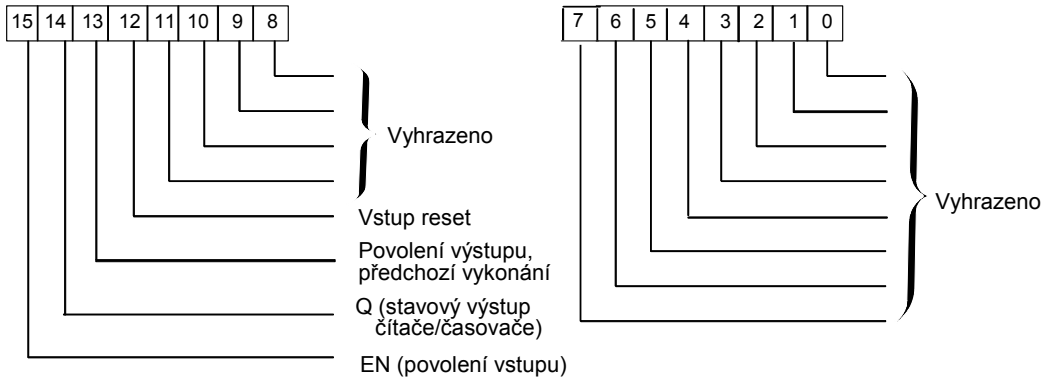
Když budete zapisovat časovač nebo čítač, musíte zapsat počáteční adresu pro tato tři slova (registry) přímo pod grafickou reprezentací funkce. Například:



Poznámka

Nepoužívejte pro tyto trojslovní bloky časovače/čítače po sobě jdoucí registry. Logicmaster ***nekontroluje ani nevaruje***, jestli se bloky registrů překrývají. Pokud umístíte aktuální hodnotu bloku na začátek předvolby předchozího bloku, časovače a čítače nebudou pracovat.

V řídicím slově je uložený stav Booleovských vstupů a výstupů souvisejícího funkčního bloku, jak je znázorněno na následujícím obrázku:



Bity 0 až 11 se používají pro přesnost časovače; bity 0 až 11 se nepoužívají pro čítače.

Poznámka

Dejte pozor, pokud v bloku tří slov budete používat pro PV stejnou adresu jako je druhé slovo bloku. Pokud PV nebude konstanta, PV se normálně nastaví na jiné umístění než druhé slovo. Některé aplikace volí použití adresy druhého slova pro PV, například používají %R0102, když spodní blok dat začíná na %R0101. To umožňuje, aby aplikace změnila PV, když časovač nebo čítač bude běžet. Aplikace může načíst první slovo CV nebo třetí řídicí slovo, ale nemůže do těchto hodnot zapisovat; jinak by funkce nepracovala.

Zvláštní poznámka k některým bitovým operacím

Když budete používat funkci Testovat bit, Nastavit bit, Vynulovat bit nebo Pozice bitu, bity budou očíslované 1 až 16, ***NE*** 0 až 15, jak bylo ukázáno výše.

ONDTR

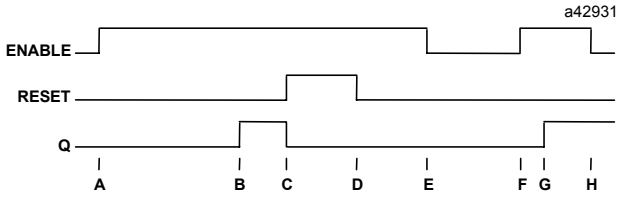
Retentivní časovač zpoždění při zapnutí (ONDTR) bude provádět inkrementaci, když skrz něj poteče proud, a když se proud zastaví, hodnotu si podrží. Čas se může čítat v desetinách sekundy (výchozí volba), setinách sekundy nebo tisícinách sekundy. Rozsah je 0 až +32,767 časových jednotek. Stav časovače je při výpadku napájení retentivní; při zapnutí napájení neproběhne žádná automatická inicializace.

Když skrz ONDTR poteče proud poprvé, začne čítat čas (aktuální hodnotu). Když se v žebříkové logice zjistí tento časovač, provede se aktualizace jeho aktuální hodnoty.

Poznámka

Pokud během cyklu CPU bude povolený vícenásobný výskyt stejného čítače se stejnou adresou, aktuální hodnoty časovačů budou stejné.

Když se aktuální hodnota bude rovnat nebo bude větší než předvolená hodnota PV, výstup Q přejde do jedničky. Dokud časovačem bude téct proud, bude pokračovat v čítání, dokud se nedosáhne maximální hodnoty. Po dosažení maximální hodnoty se tato hodnota podrží a výstup Q zůstane v jedničce bez ohledu na stav vstupu pro povolení.



A = ENABLE přejde do jedničky; časovač začne načítat

B = CV dosáhne PV; Q přejde do jedničky.

C = RESET přejde do jedničky; Q přejde do nuly, celkový čas se resetuje.

D = RESET přejde do nuly; časovač pak začne čítat znovu.

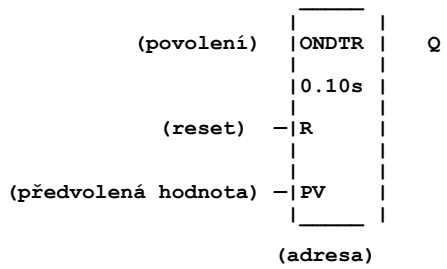
E = ENABLE přejde do nuly; časovač zastaví čítání. Celkový čas zůstane stejný.

F = ENABLE přejde znovu do jedničky; časovač začne čítat čas.

G = CV se rovná PV; Q přejde do jedničky. Časovač pokračuje v čítání času, dokud ENABLE nepřejde do nuly, RESET nepřejde do jedničky nebo CV se nebude rovnat maximálnímu času.

H = ENABLE přejde do nuly; časovač zastaví čítání času.

Když se proud do časovače zastaví, aktuální hodnota se přestane inkrementovat a podrží se. Výstup Q zůstane v jedničce, pokud v ní byl. Když funkcí znovu poteče proud, aktuální hodnota se bude znovu inkrementovat od zapamatované hodnoty. Když na reset R přijde jednička, aktuální hodnota se nastaví zpátky na nulu a výstup Q přejde do nuly. Pokud v případě PLC řady 35x a 36x bude vstup pro povolení na ONDTR v nule, PV = 0 a na reset R přijde jednička, pak výstup bude v nule. Avšak v případě PLC 311–341 bude za stejných podmínek výstup v jedničce.



Parametry

Parametr	Popis
adresa	ONDTR používá k uložení následujících informací tři po sobě jdoucí slova (registry) paměti %R: • Aktuální hodnota (CV) = slovo 1. • Předvolená hodnota (PV) = slovo 2. • Řídicí slovo = slovo 3. Když budete zapisovat ONDTR, musíte zapsat adresu pro umístění těchto tří po sobě jdoucích slov (registrů) přímo pod grafické znázornění funkce. **Poznámka:** Nepoužívejte tuto adresu s jinými instrukcemi. **Upozornění:** Překrývání adres bude mít za následek chybnou činnost časovače.
povolení	Když na vstup pro povolení přijde jednička, provede se inkrementace aktuální hodnoty časovače.
R	Když na R přijde jednička, provede se reset jeho hodnoty na nulu.
PV	PV je hodnota, která se zkopíruje do předvolené hodnoty časovače, když se provede povolení nebo reset časovače.
Q	Výstup Q bude v jedničce, když aktuální hodnota bude větší nebo se bude rovnat předvolené hodnotě.
čas	Časový inkrement je v desetinách (0.1), setinách (0.01) nebo tisícinách (0.001) sekundy pro nejnižší bit předvolené hodnoty PV.

Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
adresa								•				
povolení	•											
R	•											
PV		•	•	•	•		•	•	•	•	•	•
Q	•											•

• Platná adresa nebo místo, kde funkcí může téct proud.

Příklad

V následujícím příkladu se používá retentivní časovač zpoždění při zapnutí k vytvoření signálu (%Q0011), který se zapne 8.0 sekundy po zapnutí %Q0010, a vypne se, když se vypne %Q0010.

```
|
|  %Q0010 |      |                                                    %Q0011
|---| |---|ONDTR|-------------------------------------------------------( )-
|         | 0.1s|
|  %Q0010 |     |
|---|/|---|R    |
|         |     |
|         |     |
|  CONST -|PV   |
|  +00080 |_____|
|
|         %R0004
```

TMR

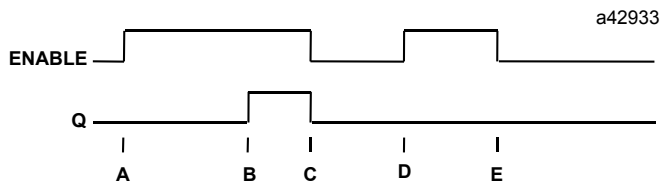
Funkce jednoduchého časovače zpoždění při zapnutí (TMR) bude provádět inkrementaci, dokud skrz ní poteče proud, a provede vynulování, když se proud zastaví. Čas se může čítat v desetinách sekundy (výchozí volba), setinách sekundy nebo tisícinách sekundy. Rozsah je 0 až +32,767 časových jednotek a proto rozsah časování je 0.001 až 3,276.7 sekundy. Stav časovače je při výpadku napájení retentivní; při zapnutí napájení neproběhne žádná automatická inicializace.

Když skrz TMR poteče proud poprvé, časovač začne čítat čas (aktuální hodnotu). Při zjištění v logice se aktuální hodnota se aktualizuje, aby obsahovala celkovou dobu, po kterou funkce časovače byla povolená od posledního resetu.

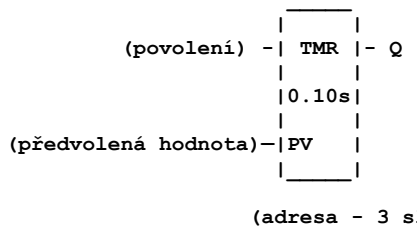
Poznámka

Pokud během cyklu CPU bude povolený vícenásobný výskyt stejného čítače se stejnou referenční adresou, aktuální hodnoty časovačů budou stejné.

Tato aktualizace se bude provádět, dokud povolující logika zůstane ve stavu ON. Když se aktuální hodnota bude rovnat nebo bude větší než předvolená hodnota PV, funkce začne pouštět proud doprava. Časovač bude pokračovat v načítání času, dokud se nedosáhne maximální hodnoty. Když parametr povolení přejde ze stavu ON do stavu OFF, časovač zastaví čítání času a aktuální hodnota se vynuluje.



A = ENABLE přejde do jedničky; časovač začne načítat čas.
B = Aktuální hodnota dosáhne předvolené hodnoty PV; Q přejde do jedničky a časovač bude pokračovat v načítání času.
C = ENABLE přejde do nuly; Q přejde do nuly; časovač zastaví čítání času a aktuální čas se vynuluje.
D = ENABLE přejde do jedničky; časovač začne načítat čas.
E = ENABLE přejde do nuly před dosažením předvolené hodnoty PV; Q zůstane v nule; časovač zastaví načítání času a vynuluje se.



Parametry

Parametr	Popis
adresa	TMR používá k uložení následujících informací tři po sobě jdoucí slova (registry) paměti %R: • Aktuální hodnota (CV) = slovo 1. • Předvolená hodnota (PV) = slovo 2. • Řídicí slovo = slovo 3. Když budete zapisovat TMR, musíte zapsat adresu pro umístění těchto tří po sobě jdoucích slov (registrů) přímo pod grafické znázornění funkce. **Poznámka:** Nepoužívejte tuto adresu s jinými instrukcemi. **Upozornění:** Překrývání adres bude mít za následek chybnou činnost časovače.
povolení	Když na vstup pro povolení přijde jednička, provede se inkrementace aktuální hodnoty časovače. Když TMR nebude povolený, aktuální hodnota se vynuluje a Q přejde do nuly.
PV	PV je hodnota, která se zkopíruje do předvolené hodnoty časovače, když se provede povolení nebo reset časovače.
Q	Výstup Q přejde do jedničky, když TMR bude povolený a aktuální hodnota bude větší nebo se bude rovnat předvolené hodnotě.

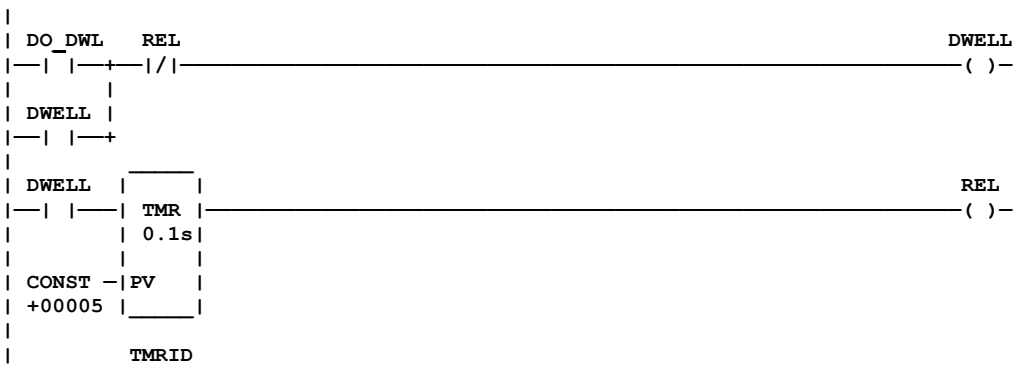
Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
adresa								•				
povolení	•											
PV		•	•	•	•		•	•	•	•	•	•
Q	•											•

• Platná adresa nebo místo, kde funkcí může téct proud.

Příklad

V následujícím příkladu se časovač zpoždění (s adresou) TMRID používá k řízení doby, po kterou cívka DWELL bude zapnutá. Když se normální spínací kontakt DO_DWL (krátkodobě) sepne, cívka DWELL se nabudí. Kontakt cívky DWELL bude držet cívku DWELL nabuzenou (když se kontakt DO_DWL uvolní) a také spustí časovač TMRID. Když TMRID dosáhne předvolené hodnoty půl sekundy, cívka REL se nabudí a přeruší zapamatovaný stav cívky DWELL. Kontakt DWELL přeruší proud do TMRID, provede reset své aktuální hodnoty a odbudí cívku REL. Obvod pak bude připravený k další krátkodobé aktivaci kontaktu DO_DWL.



OFDT

Časovač zpoždění při vypnutí (OFDT) bude provádět inkrementaci, dokud skrz něj proud nepoteče, a provede reset na nulu, když proud poteče. Čas se může čítat v desetinách sekundy (výchozí volba), setinách sekundy nebo tisícinách sekundy. Rozsah je 0 až +32,767 časových jednotek. Stav časovače je při výpadku napájení retentivní; při zapnutí napájení neproběhne žádná automatická inicializace.

Když skrz OFDT poteče proud poprvé, propustí proud doprava a aktuální hodnota (CV) se nastaví na nulu. (OFDT používá slovo 1 [registr] jako paměťové místo pro svou CV – další informace viz část “Parametry” na následující stránce.) Výstup zůstane v jedničce, dokud funkcí bude procházet proud. Pokud do funkce přestane zleva téct proud, energie poteče nadále doprava a časovač začne načítat čas do aktuální hodnoty.

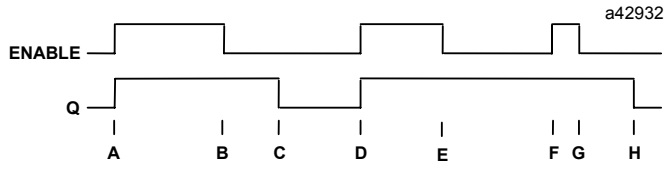
Poznámka

Pokud během cyklu CPU bude povolený vícenásobný výskyt stejného čítače se stejnou referenční adresou, aktuální hodnoty časovačů budou stejné.

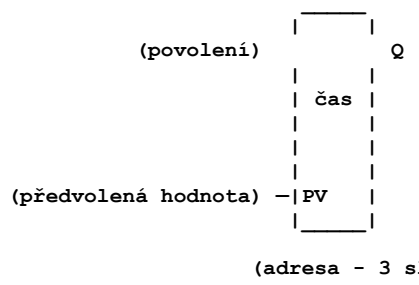
OFDT nebude propouštět proud, pokud předvolená hodnota bude nula nebo záporná hodnota.

Při každém vyvolání funkce, když povolovací logika bude nastavená do stavu OFF, se aktuální hodnota aktualizuje tak, aby obsahovala dobu od vypnutí časovače. Když se aktuální hodnota (CV) bude rovnat předvolené hodnotě (PV), funkce přestane propouštět proud doprava. Když k tomuto dojde, časovač zastaví čítání času – viz část C níže.

Když funkcí znovu poteče proud, aktuální hodnota se vynuluje.



A = ENABLE a Q jsou v jedničce; časovač se resetuje (CV = 0).
B = ENABLE přejde do nuly; časovač začne načítat čas.
C = CV dosáhne PV; Q přejde do nuly a časovač zastaví čítání času.
D = ENABLE přejde do jedničky; časovač se resetuje (CV = 0).
E = ENABLE přejde do nuly; časovač začne načítat čas.
F = ENABLE přejde do jedničky; časovač se resetuje (CV = 0).
G = ENABLE přejde do nuly; časovač začne načítat čas.
H = CV dosáhne PV; Q přejde do nuly a časovač zastaví čítání času.



Když se OFDT použije v bloku programu, který se ***nevyvolává*** každý cyklus, časovač bude načítat čas mezi voláními bloku programu, dokud se neprovede jeho reset. To znamená, že funguje jako časovač, který pracuje v programu s mnohem pomalejším cyklem než časovač v hlavním bloku programu. U bloků programu, které jsou v nečinnosti po delší dobu, je nutno časovač naprogramovat tak, aby umožnil tuto funkci zachycení. Pokud například časovač v bloku programu bude resetovaný a blok programu nebude vyvolaný (nebude aktivní) čtyři minuty, při vyvolání bloku programu již bude celkový načítaný čas čtyři minuty. Pokud se předtím neprovede reset časovače, pro čítač se použije tento čas.

Parametry

Parametr	Popis
adresa	OFDT používá k uložení následujících informací tři po sobě jdoucí slova (registry) paměti %R: • Aktuální hodnota (CV) = slovo 1. • Předvolená hodnota (PV) = slovo 2. • Řídicí slovo = slovo 3. Když budete zapisovat OFDT, musíte zapsat adresu pro umístění těchto tří po sobě jdoucích slov (registrů) přímo pod grafické znázornění funkce. **Poznámka:** Nepoužívejte tuto adresu s jinými instrukcemi. **Upozornění:** Překrývání adres bude mít za následek chybnou operaci časovače.
povolení	Když na vstup pro povolení přijde jednička, provede se inkrementace aktuální hodnoty časovače.
čas	Časový inkrement je v desetinách (0.1), setinách (0.01) nebo tisícinách (0.001) sekundy pro nejnižší bit předvolené hodnoty PV.
PV	PV je hodnota, která se zkopíruje do předvolené hodnoty časovače, když se provede povolení nebo reset časovače.
Q	Výstup Q bude v jedničce, když aktuální hodnota bude menší předvolená hodnota. Stav Q je při výpadku napájení retentivní; při zapnutí napájení se neprovádí žádná automatická inicializace.

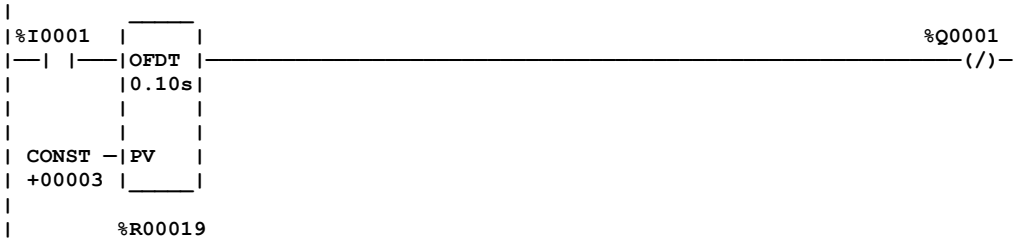
Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
adresa								•				
povolení	•											
PV	•	•	•	•	•		•	•	•	•	•	•
Q	•											•

• Platná adresa nebo místo, kde funkcí může téct proud.

Příklad

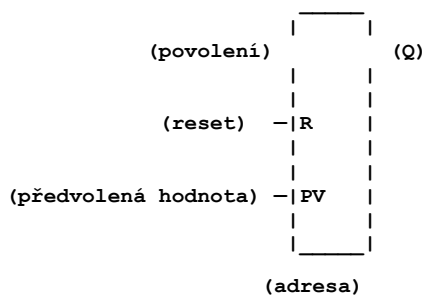
V následujícím příkladu se časovač OFDT používá k vypnutí výstupu (%Q0001) při každém zapnutí vstupu (%I0001). Výstup se sepne opět 0,3 sekundy po vypnutí vstupu.



UPCTR

Funkce vzestupného čítače (UPCTR) se používá k čítání do zadané hodnoty. Rozsah je 0 až +32,767 impulsů. Když reset vzestupného čítače bude ve stavu ON, aktuální hodnota čítače se resetuje na nulu. Při každém přechodu vstupu pro povolení ze stavu OFF do stavu ON se aktuální hodnota inkrementuje o 1. Aktuální hodnota se může inkrementovat dál než na předvolenou hodnotu PV. Výstup bude ve stavu ON vždy, když aktuální hodnota bude větší nebo se bude rovnat předvolené hodnotě.

Stav UPCTR je při výpadku napájení retentivní; při zapnutí napájení žádná automatická inicializace neproběhne.



Parametry

Parametr	Popis
adresa	UPCTR používá k uložení následujících informací tři po sobě jdoucí slova (registry) paměti %R: • Aktuální hodnota (CV) = slovo 1. • Předvolená hodnota (PV) = slovo 2. • Řídicí slovo = slovo 3. Když budete zapisovat UPCTR, musíte zapsat adresu pro umístění těchto tří po sobě jdoucích slov (registrů) přímo pod grafické znázornění funkce. **Poznámka:** Nepoužívejte tuto adresu s jiným vzestupným čítačem, sestupným čítačem nebo jinou instrukcí, jinak se provede nesprávná operace. **Upozornění:** Překrývání adres bude mít za následek chybnou činnost čítače.
povolení	S náběžnou hranou povolení se aktuální počet inkrementuje o jedničku.
R	Když na R přijde jednička, provede se reset jeho hodnoty zpět na nulu.
PV	PV je hodnota, která se zkopíruje do předvolené hodnoty čítače, když se provede povolení nebo reset čítače.
Q	Výstup Q bude v jedničce, když aktuální hodnota bude větší nebo se bude rovnat předvolené hodnotě.

Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
adresa								•				
povolení	•											
R	•											
PV		•	•	•	•		•	•	•	•	•	
Q	•											•

• Platná adresa nebo místo, kde funkcí může téct proud.

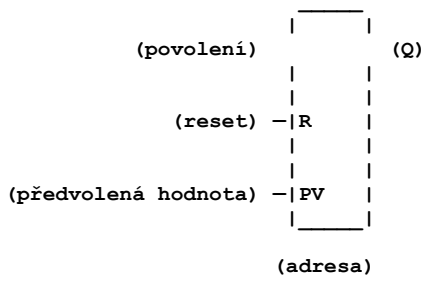
Příklad

V následujícím příkladu při každém přechodu vstupu %I0012 ze stavu OFF do stavu ON vzestupný čítač PRT_CNT přičte 1; interní cívka %M0001 bude nabuzená vždy, když se napočítá 100 dílů. Vždy, když %M0001 bude ve stavu ON, se celkový počet resetuje na nulu.

DNCTR

Funkce sestupného čítače (DNCTR) se používá k odečítání od předvolené hodnoty. Minimální předvolená hodnota je nula; maximální předvolená hodnota je +32,767 impulsů. Minimální aktuální hodnota je –32,768. Po resetování se aktuální hodnota čítače nastaví na předvolenou hodnotu PV. Když se provede přechod vstupu pro povolení ze stavu OFF do stavu ON, aktuální hodnota se dekrementuje o jedničku. Výstup bude ve stavu ON vždy, když aktuální hodnota bude menší nebo se bude rovnat nule.

Aktuální hodnota DNCTR je při výpadku napájení retentivní; při zapnutí napájení neproběhne žádná automatická inicializace.



Parametry

Parametr	Popis
adresa	DNCTR používá k uložení následujících informací tři po sobě jdoucí slova (registry) paměti %R: • Aktuální hodnota (CV) = slovo 1. • Předvolená hodnota (PV) = slovo 2. • Řídicí slovo = slovo 3. Když budete zapisovat DNCTR, musíte zapsat adresu pro umístění těchto tří po sobě jdoucích slov (registrů) přímo pod grafické znázornění funkce. **Poznámka:** Nepoužívejte tuto adresu s jiným sestupným čítačem, vzestupným čítačem nebo jinou instrukcí, jinak se provede nesprávná operace. **Upozornění:** Překrývání adres bude mít za následek chybnou činnost čítače.
povolení	S náběžnou hranou povolení se aktuální hodnota dekrementuje o jedničku.
R	Když na R přijde jednička, provede se reset jeho aktuální hodnoty na předvolenou hodnotu.
PV	PV je hodnota, která se zkopíruje do předvolené hodnoty čítače, když se provede povolení nebo reset čítače.
Q	Výstup Q v jedničce, když aktuální hodnota bude menší nebo se bude rovnat nule.

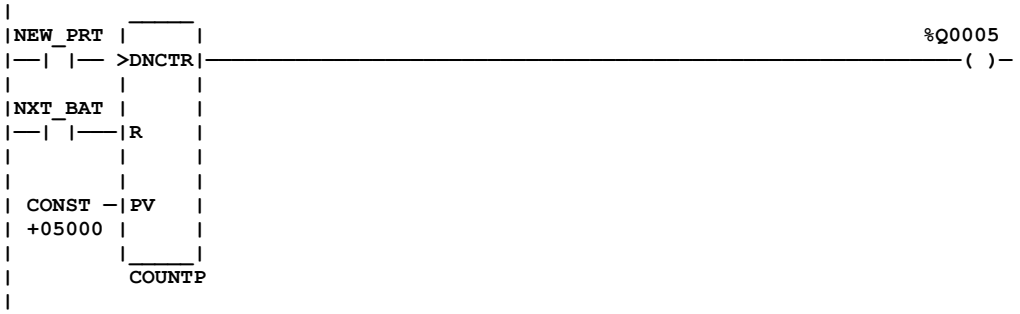
Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
adresa								•				
povolení	•											
R	•											
PV		•	•	•	•		•	•	•	•	•	•
Q	•											•

• Platná adresa nebo místo, kde funkcí může téct proud.

Příklad

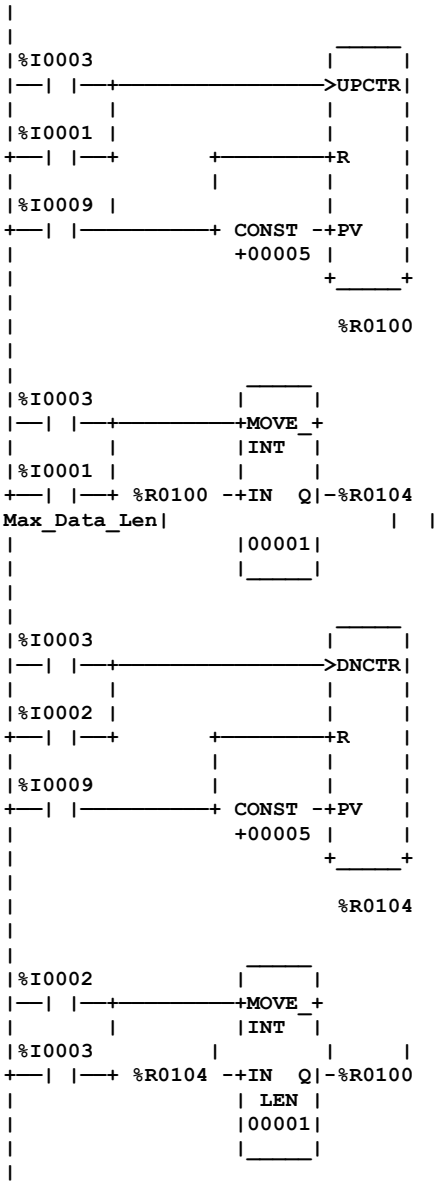
V následujícím příkladu sestupný čítač identifikoval jako COUNTP pulsy 5000 nových dílů před vybuzením výstupu %Q0005.



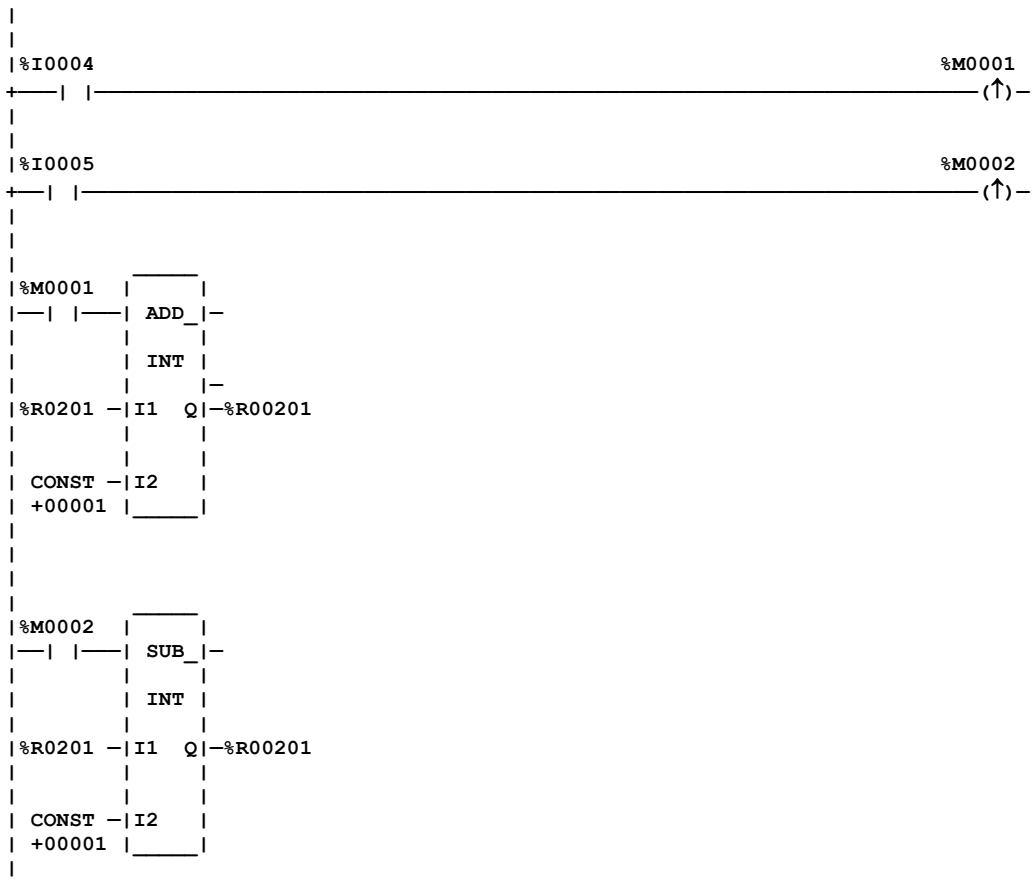
Příklad

V následujícím příkladu se PLC používá ke sledování počtu dílů uložených v přechodném skladu. Pomocí instrukčního souboru Series 90-30/20/Micro existují dva způsoby, jak dosáhnout této funkce.

První způsob je použít vzestupný/sestupný čítač se sdíleným registrem pro celkovou nebo aktuální hodnotu. Když se díly dostanou do skladového prostoru, vzestupný čítač provede inkrementaci o 1 a zvýší tím aktuální hodnotu dílů ve skladu o 1. Když díl bude ze skladového prostoru odcházet, sestupný čítač provede dekrementaci o 1 a tím sníží hodnotu zásob ve skladu o 1. Aby nedocházelo ke konfliktu se sdíleným registrem, oba čítače používají různé adresy registrů. Když se provádí čítání registru, jeho aktuální hodnota se musí přesunout do registru aktuální hodnoty jiného čítače.



Druhý způsob ukázaný níže používá k zajištění sledování stavu skladu funkce ADD a SUB.



Kapitola 6

Matematické funkce

Tato kapitola popisuje matematické funkce instrukčního souboru Series 90-30/20/Micro:

Zkratka	Funkce	Popis	Strana
ADD	Sčítání	Sečte dvě čísla.	6-2
SUB	Odečítání	Odečte jedno číslo od druhého.	6-2
MUL	Násobení	Vynásobí dvě čísla.	6-2
DIV	Dělení	Vydělí jedno číslo jiným číslem, výsledkem je podíl.	6-2
MOD	Dělení Modulo	Vydělí jedno číslo jiným číslem, výsledkem je zbytek.	6-6
SQRT	Druhá odmocnina	Vypočítá druhou odmocninu celého nebo reálného čísla.	6-8
SIN, COS, TAN, ASIN, ACOS, ATAN	Trigonometrické funkce †	Vykoná příslušnou funkci reálného čísla na vstupu IN.	6-10
LOG, LN EXP, EXPT	Logaritmické/exponenciální funkce †	Vykoná příslušnou funkci reálného čísla na vstupu IN.	6-12
RAD, DEG	Převod radiánů †	Vykoná příslušnou funkci reálného čísla na vstupu IN.	6-14

† Trigonometrické funkce, Logaritmické/exponenciální funkce a funkce převodu radiánů lze použít pouze u CPU řady 35x a 36x, verze 9 nebo pozdější a u všech verzí CPU352.

Poznámka

Dělení a dělení Modulo jsou podobné funkce, které se liší ve výstupu; dělení hledá podíl, zatímco dělení Modulo hledá zbytek.

Standardní matematické funkce (ADD, SUB, MUL, DIV)

Matematické funkce zahrnují sčítání, odečítání, násobení a dělení. Když funkcí poteče proud, se vstupními parametry I1 a I2 se vykoná příslušná matematická funkce. Tyto parametry musí být stejného typu dat. Výstup Q bude stejného typu dat jako I1 a I2.

Poznámka

DIV zaokrouhluje dolů; neprovádí zaokrouhlení na nejbližší vyšší celé číslo. (Například 24 DIV 5 = 4.)

Matematické funkce pracují s následujícími typy dat:

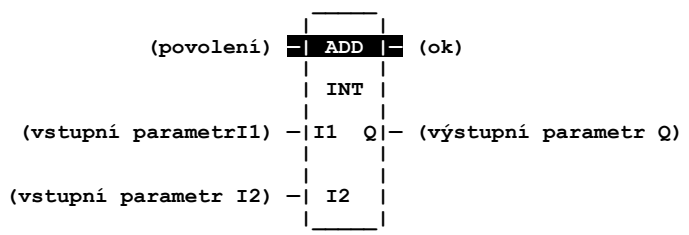
Typ dat	Popis
INT	Celé číslo se znaménkem
DINT	Celé číslo se znaménkem s dvojnásobnou délkou
REAL	Pohyblivá desetinná tečka

Poznámka

Data typu REAL je možno použít pouze u CPU řady 35x a 36x, verze 9 nebo pozdější nebo u všech verzí CPU352.

Výchozí typ dat je celé číslo se znaménkem; po zvolení funkce ho však lze změnit. Více informací o typu dat najdete v kapitole 2, část 2, "Organizace programu a uživatelské adresy/data".

Pokud operace INT nebo DINT povede na přetečení, výstupní adresa se nastaví na nejvyšší možnou hodnotu pro daný typ dat. U čísel se znaménkem se znaménko nastaví tak, aby ukazovalo směr přetečení. Pokud operace nebude mít za následek přetečení (a vstupy budou platná čísla), výstup se nastaví do stavu ON; jinak se nastaví do stavu OFF. Pokud se budou používat celá čísla s jednoduchou nebo dvojitou délkou, znaménko výsledku funkcí DIV a MUL bude záviset na znaménčích I1 a I2.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, operace se provede.
I1	I1 obsahuje konstantu nebo adresu první hodnoty použité v operaci. (I1 je levá strana matematické rovnice, jako v I1 - I2).
I2	I2 obsahuje konstantu nebo adresu druhé hodnoty použité v operaci. (I2 je pravá strana matematické rovnice, jako v I1 - I2).
ok	Výstup ok bude v jedničce, když se funkce vykoná bez přetečení a nevyskytne se neplatná operace.
Q	Vstup Q obsahuje výsledek operace.

Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
I1		o	o	o	o		o	•	•	•	•†	
I2		o	o	o	o		o	•	•	•	•†	
ok	•											•
Q		o	o	o	o		o	•	•	•		

• Platná adresa nebo místo, kde funkcí může téct proud.
o Platná adresa pouze pro data INT; neplatí pro DINT nebo REAL.
† Pro operace s celým číslem se znaménkem s dvojnásobnou délkou jsou konstanty omezené na hodnoty mezi –32,768 a +32,767.

Poznámka

Pro 16-bitové operandy nebo operandy s jedním registrem je výchozí typ INT. Stisknutím **F10** se volba typu změní na DINT, 32-bitové dvojité slovo, nebo REAL (pouze u CPU řady 35x a 36x). PLC hodnoty INT zabírají jeden 16-bitový registr, %R, %AI nebo %AQ. Hodnoty DINT vyžadují dva po sobě jdoucí registry s dolními 16 bity v prvním slově a horními 16 bity se znaménkem v druhém slově. Hodnoty REAL u CPU řady 35x a 36x (verze 9 nebo pozdější) a všechny verze CPU352 také zabírají 32-bitový dvojitý registr se znaménkem v horním bitu, za kterým následuje exponent a mantisa.

Příklad

V následujícím příkladu se při každém nastavení vstupu %I0001 a pokud při odečítání nedojde k přetečení provede dekrementace celého čísla %R0002 o 1 a cívka %Q0001 se sepne.

Matematické funkce a typy dat

Funkce	Operace	Zobrazuje se jako
ADD INT	Q (16 bitů) = I1 (16 bitů) + I2 (16 bitů)	5-místné dekadické číslo se znaménkem
ADD DINT	Q (32 bitů) = I1 (32 bitů) + I2 (32 bitů)	8-místné dekadické číslo se znaménkem
ADD REAL*	Q (32 bitů) = I1 (32 bitů) + I2 (32 bitů)	7-místné dekadické číslo, znaménko a desetinný zlomek
SUB INT	Q (16 bitů) = I1 (16 bitů) - I2 (16 bitů)	5-místné dekadické číslo se znaménkem
SUB DINT	Q (32 bitů) = I1 (32 bitů) - I2 (32 bitů)	8-místné dekadické číslo se znaménkem
SUB REAL*	Q (32 bitů) = I1 (32 bitů) - I2 (32 bitů)	7-místné dekadické číslo, znaménko a desetinný zlomek
MUL INT	Q (16 bitů) = I1 (16 bitů) * I2 (16 bitů)	5-místné dekadické číslo se znaménkem
MUL DINT	Q (32 bitů) = I1 (32 bitů) * I2 (32 bitů)	8-místné dekadické číslo se znaménkem
MUL REAL*	Q (32 bitů) = I1 (32 bitů) * I2 (32 bitů)	7-místné dekadické číslo, znaménko a desetinný zlomek
DIV INT	Q (16 bitů) = I1 (16 bitů) / I2 (16 bitů)	5-místné dekadické číslo se znaménkem
DIV DINT	Q (32 bitů) = I1 (32 bitů) / I2 (32 bitů)	8-místné dekadické číslo se znaménkem
DIV REAL*	Q (32 bitů) = I1 (32 bitů) / I2 (32 bitů)	7-místné dekadické číslo, znaménko a desetinný zlomek

* Pouze CPU řady 35x a 36x, verze 9 nebo pozdější, nebo všechny verze CPU352

Poznámka

Typy vstupních a výstupních dat musí být stejné. Funkce MUL a DIV nepodporují smíšený režim, který podporují PLC 90-70. Například MUL INT dvou 16-bitových vstupů vytvoří 16-bitový součin a ne 32-bitový součin. Použití MUL DINT pro 32-bitový součin vyžaduje, aby oba vstupy byly 32-bitové. Funkce DIV INT provede dělení 16-bitového vstupu I2 se 16-bitovým výsledkem, zatímco DIV DINT provede dělení 32-bitového vstupu I1 32-bitovým vstupem I2 s 32-bitovým výsledkem.

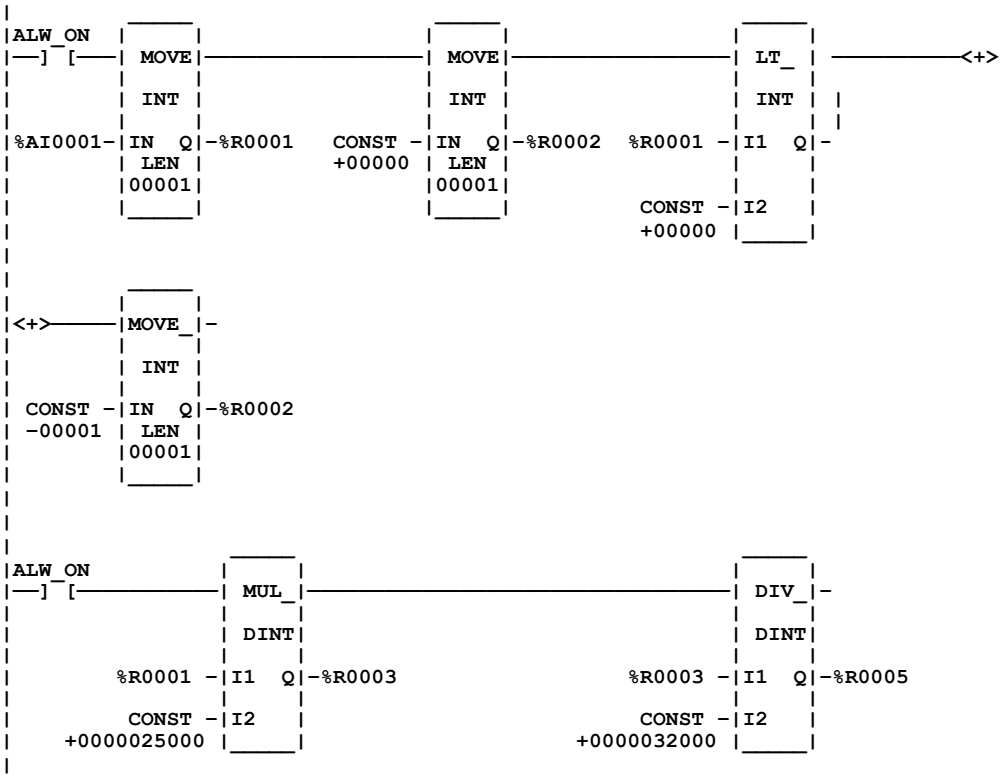
Tyto funkce přenesou proud, pokud nedojde k matematickému přetečení. Pokud dojde k přetečení, výsledek bude největší hodnota s příslušným znaménkem a proud se nepřenese.

Dejte pozor, aby při použití funkcí MUL a DIV nedošlo k přetečení. Pokud budete provádět převod hodnoty INT na hodnotu DINT, pamatujte na to, že CPU používá standardní dvojkový doplněk se znaménkem umístěným v nejvyšším bitu druhého slova. Musíte zkontrolovat znaménko nižšího 16-bitového slova a přemístit ho do druhého 16-bitového slova. Pokud bit s nejvyšší váhou v 16-bitovém slově INT bude 0 (kladné), do druhého slova zapište 0. Pokud bit s nejvyšší váhou v 16-bitovém slově INT bude -1 (záporné), do druhého slova zapište -1 nebo 0FFFFh. Převod DINT na INT je snazší, protože nižší 16-bitové slovo (první registr) je INT část 32-bitového slova DINT. Horních 16 bitů nebo druhé slovo musí být buď hodnota 0 (kladné) nebo –1 (záporné), jinak číslo DINT bude příliš velké na převedení na 16 bitů.

Příklad

Běžnou aplikací je změna analogové vstupní hodnoty operací MUL, za kterou následuje operace DIV a případně operace ADD. Při rozsahu do 32000 a použití MUL INT dojde k přetečení. Použití hodnoty %AI pro MUL DINT také nebude fungovat, protože 32-bitový vstup I1 kombinuje současně 2 analogové vstupy. Musíte přemístit analogový vstup do dolního slova dvojnásobného registru, pak zjistit znaménko a druhý registr nastavit v případě kladného čísla na 0 nebo v případě záporného čísla na -1. Pro následující funkci DIV použijte dvojnásobný registr s MUL DINT pro 32-bitový součin.

Například následující logiku je možno použít k převodu vstupu +/–10 voltů %AI1 na +/– 25000 technických jednotek v %R5.



MOD (INT, DINT)

Funkce Modulo (MOD) se používá k dělení jedné hodnoty jinou hodnotou stejného typu dat s cílem získat zbytek. Znaménko výsledku bude vždy stejné jako znaménko vstupního parametru I1.

Funkce MOD pracuje s následujícími typy dat:

Typ dat	Popis
INT	Celé číslo se znaménkem
DINT	Celé číslo se znaménkem s dvojnásobnou délkou

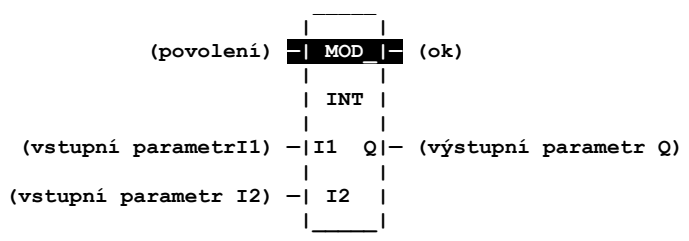
Výchozí typ dat je celé číslo se znaménkem; po zvolení funkce ho však lze změnit. Více informací o typu dat najdete v kapitole 2, část 2, "Organizace programu a uživatelské adresy/data".

Když funkcí bude protékat proud, provede se dělení vstupního parametru I1 vstupním parametrem I2. Tyto parametry musí být stejného typu dat. Výstup Q se vypočítá podle následujícího vztahu:

```
Q = I1 - ((I1 DIV I2) * I2)
```

kde DIV vytvoří celé číslo. Q bude stejného typu dat jako vstupní parametry I1 a I2.

Pokud se nebudete snažit dělit nulou, OK bude vždy ve stavu ON, když do funkce poteče proud. V opačném případě se OK nastaví do stavu OFF.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, operace se provede.
I1	I1 obsahuje konstantu nebo adresu hodnoty, která se má dělit hodnotou I2.
I2	I2 obsahuje konstantu nebo adresu hodnoty, která má dělit hodnotu I1.
ok	Výstup ok bude v jedničce, když se funkce vykoná bez přetečení.
Q	Výstup Q obsahuje zbytek při dělení hodnoty I1 hodnotou I2 .

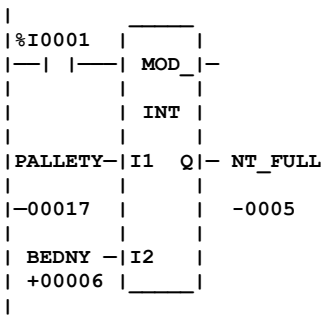
Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
I1		o	o	o	o		o	•	•	•	•†	
I2		o	o	o	o		o	•	•	•	•†	
ok	•											•
Q		o	o	o	o		o	•	•	•		

• Platná adresa nebo místo, kde funkcí může téct proud.
o Pouze platná adresa pro data INT; neplatí pro DINT.
† Pro operace s celým číslem se znaménkem s dvojnásobnou délkou jsou konstanty omezené na hodnoty mezi –32,768 a +32,767.

Příklad

V následujícím příkladu se zbytek celočíselného dělení BEDEN na PALETY umístí do NT_FULL vždy, když %I0001 bude ve stavu ON.



SQRT (INT, DINT, REAL)

Funkce druhé odmocniny (SQRT) se používá k nalezení druhé odmocniny nějaké hodnoty. Když funkcí poteče proud, hodnota výstupu Q se nastaví na celočíselnou část odmocniny vstupu IN. Výstup Q musí být stejného typu dat jako IN.

Funkce SQRT pracuje s následujícími typy dat:

Typ dat	Popis
INT	Celé číslo se znaménkem
DINT	Celé číslo se znaménkem s dvojnásobnou délkou
REAL	Pohyblivá desetinná tečka

Poznámka

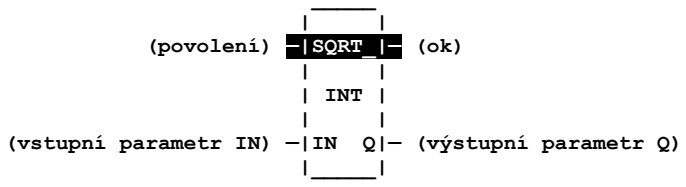
Data typu REAL je možno použít pouze u CPU řady 35x a 36x, verze 9 nebo pozdější nebo u všech verzí CPU352.

Výchozí typ dat je celé číslo se znaménkem; po zvolení funkce ho však lze změnit. Více informací o typu dat najdete v kapitole 2, část 2, "Organizace programu a uživatelské adresy/data".

Pokud se nevyskytne některá z následujících neplatných operací REAL, OK se nastaví do stavu ON, když se funkce vykoná bez přetečení:

- IN < 0.
- IN je NaN (nenumerický).

Jinak se ok nastaví do stavu OFF.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, operace se provede.
IN	IN obsahuje konstantu nebo adresu hodnoty, jejíž odmocnina se má vypočítat. Pokud IN bude menší než nula, funkce proud nepropustí.
ok	Výstup ok bude v jedničce, když se funkce vykoná bez přetečení a nevyskytne se neplatná operace.
Q	Výstup Q obsahuje druhou odmocninu z IN.

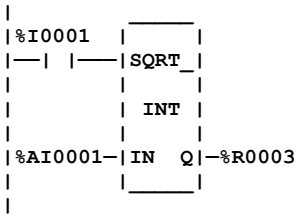
Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN		o	o	o	o		o	•	•	•	•†	
ok	•											•
Q		o	o	o	o		o	•	•	•		

• Platná adresa nebo místo, kde funkcí může téct proud.
o Pouze platná adresa pro data INT; neplatí pro DINT a REAL.
† Pro operace s celým číslem se znaménkem s dvojnásobnou délkou jsou konstanty omezené na hodnoty mezi –32,768 a +32,767.

Příklad

V následujícím příkladu se druhá odmocnina celého čísla na adrese %AI001 umístí do výsledku na adrese %R0003 vždy, když %I0001 bude ve stavu ON.



Trigonometrické funkce (SIN, COS, TAN, ASIN, ACOS, ATAN)

Funkce SIN, COS a TAN se používají k nalezení trigonometrické hodnoty sinus, kosinus nebo tangens vstupní hodnoty. Když některou z těchto funkcí poteče proud, vypočítá se sinus (nebo kosinus nebo tangens) hodnoty IN, jejíž jednotky jsou radiány, a výsledek se uloží do výstupu Q. IN i Q jsou hodnoty s pohyblivou desetinnou tečkou.

Funkce ASIN, ACOS a ATAN se používají k nalezení inverzní trigonometrické hodnoty sinus, kosinus nebo tangens vstupní hodnoty. Když některou z těchto funkcí poteče proud, vypočítá se inverzní sinus (nebo kosinus nebo tangens) hodnoty IN a výsledek se uloží do výstupu Q, jehož jednotky jsou radiány. IN i Q jsou hodnoty s pohyblivou desetinnou tečkou.

Pro funkce SIN, COS a TAN je možno zadat široký vstupních rozsah, kde

$-2^{63} < IN < +2^{63}$, ($2^{63} \approx 9.22 \times 10^{18}$).

Pro funkce ASIN a ACOS je možno zadat úzký rozsah vstupních hodnot, kde $-1 \leq IN \leq 1$. Při dané platné hodnotě pro parametr IN funkce ASIN_REAL vytvoří výsledek Q takový, že:

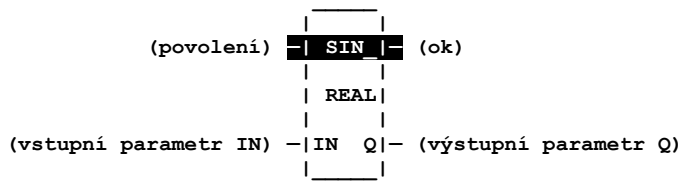
$$\mathbf{ASIN\ (IN)} = -\frac{\pi}{2} \leq Q \leq \frac{\pi}{2}$$

Funkce ACOS_REAL vytvoří výsledek Q takový, že:

$$\mathbf{ACOS\ (IN)} = 0 \leq Q \leq \pi$$

Pro funkci ATAN je možno zadat nejširší rozsah vstupních hodnot, kde $-\infty \leq IN \leq +\infty$. Při dané platné hodnotě pro parametr IN funkce ATAN_REAL vytvoří výsledek Q takový, že:

$$\mathbf{ATAN\ (IN)} = -\frac{\pi}{2} \leq Q \leq \frac{\pi}{2}$$



Poznámka

Funkci TRIG je možno použít pouze u CPU řady 35x a 36x, verze 9 nebo pozdější nebo u všech verzí CPU352.

Parametry

Parametr	Popis
povolení	Když funkce bude povolená, operace se provede.
IN	IN obsahuje konstantu nebo adresu reálné hodnoty použité v operaci.
ok	Výstup ok bude v jedničce, když se funkce vykoná bez přetečení za předpokladu, že se neobjeví neplatná operace a/nebo IN nebude NaN.
Q	Výstup Q obsahuje trigonometrickou hodnotu z IN.

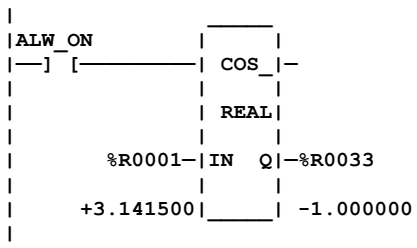
Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN								•	•	•	•	
ok	•											•
Q								•	•	•		

• Platná adresa nebo místo, kde funkcí může téct proud.

Příklad

V následujícím příkladu se COS hodnoty v %R0001 umístí do %R0033.

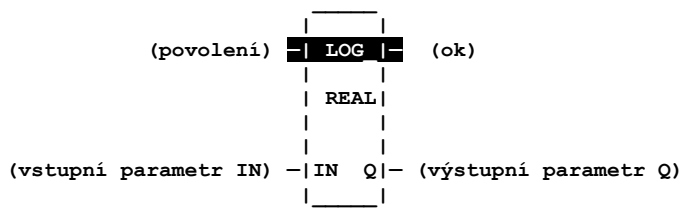


Logaritmické/exponenciální funkce (LOG, LN, EXP, EXPT)

Funkce LOG, LN a EXP mají dva vstupní parametry a dva výstupní parametry. Když funkcí poteče proud, funkce vykoná příslušnou logaritmickou/exponenciální operaci reálné hodnoty na vstupu IN a výsledek umístí na výstup Q.

- Funkce LOG do Q umístí dekadický logaritmus (se základem 10) hodnoty IN .
- Funkce LN do Q umístí přirozený logaritmus hodnoty IN.
- Funkce EXP umocní ***e*** hodnotou zadanou v IN a výsledek se umístí Q.
- Funkce EXPT umocní hodnotu vstupu I1 hodnotou I2 a výsledek umístí do Q. (Funkce EXPT má tři vstupní parametry a dva výstupní parametry.)

Výstupem ok poteče proud, pokud IN nebude NaN (nenumerický) nebo nebude záporný.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, operace se provede.
IN	IN obsahuje reálnou hodnotu použitou v operaci.
ok	Výstup ok bude v jedničce, když se funkce vykoná bez přetečení za předpokladu, že se neobjeví neplatná operace a/nebo IN nebude NaN.
Q	Výstup Q obsahuje logaritmickou/exponenciální hodnotu vstupu IN.

Poznámka

Funkce LOG, LN, EXP a EXPT je možno použít pouze u CPU řady 35x a 36x, verze 9 nebo pozdější, a u všech verzí CPU352.

Poznámka

Když vstupní hodnota IN pro funkci EXP bude záporné nekonečno (-∞), funkce vrátí podle očekávání hodnotu 0. V takovém případě u CPU352 funkce proud *nepropustí*. U všech ostatních CPU 90-30 proud *neproteče*, i když výstup bude 0.

Platné typy paměti

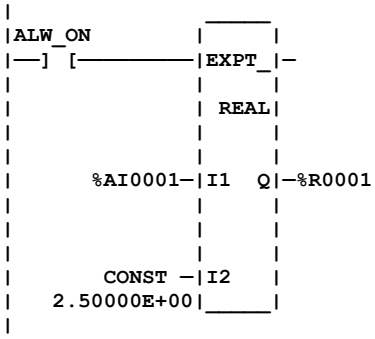
Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN*								•	•	•	•	
ok	•											•
Q								•	•	•		

* U funkce EXPT se vstup IN nahradí vstupními parametry I1 a I2.

• Platná adresa nebo místo, kde funkcí může téct proud.

Příklad

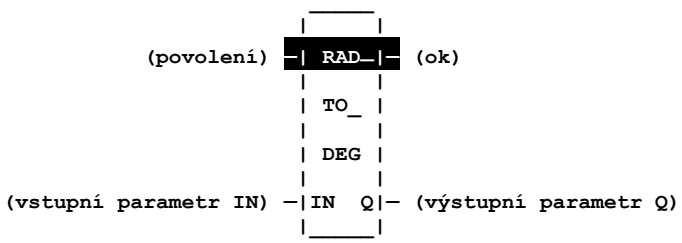
V následujícím příkladu se hodnota %AI0001 umocní hodnotou 2.5 a výsledek se umístí do %R0001.



Převod radiánů (RAD, DEG)

Když funkcí bude protékat proud, provede se příslušný převod (RAD_TO_DEG nebo DEG_TO_RAD, tj. radiány na stupně nebo obráceně) reálné hodnoty na vstupu IN a výsledek se umístí do Q.

Výstupem ok poteče proud, pokud IN nebude NaN (nenumerický) nebo nebude záporný.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, operace se provede.
IN	IN obsahuje reálnou hodnotu použitou v operaci.
ok	Výstup ok bude v jedničce, když se funkce vykoná bez přetečení za předpokladu, že IN nebude NaN.
Q	Výstup Q obsahuje převedenou hodnotu IN.

Poznámka

Funkce převodu radiánů je možno použít pouze u CPU řady 35x a 36x, verze 9 nebo pozdější nebo u všech verzí CPU352.

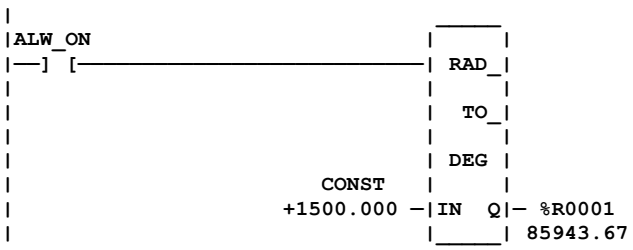
Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN								•	•	•	•	
ok	•											•
Q								•	•	•		

• Platná adresa nebo místo, kde funkcí může téct proud.

Příklad

V následujícím příkladu se +1500 převede na DEG a umístí se na adrese %R0001.



Kapitola 7

Relační funkce

Relační funkce se používají k určení vztahu dvou hodnot. Tato kapitola popisuje následující relační funkce:

Zkratka	Funkce	Popis	Strana
EQ	Rovná se	Test rovnosti dvou čísel.	7-2
NE	Nerovná se	Test nerovnosti dvou čísel.	7-2
GT	Větší než	Test, jestli jedno číslo je větší než druhé.	7-2
GE	Větší nebo rovno	Test, jestli jedno číslo je větší nebo rovno druhému.	7-2
LT	Menší než	Test, jestli jedno číslo je menší než druhé.	7-2
LE	Menší nebo rovno	Test, jestli jedno číslo je menší nebo rovno druhému.	7-2
RANGE	Rozsah	Určí, jestli číslo leží uvnitř předepsaného rozsahu (lze použít u CPU verze 4.5 nebo vyšší).	7-4

Standardní relační funkce (EQ, NE, GT, GE, LT, LE)

Když funkcí bude protékat proud, bude se porovnávat vstupní parametr I1 se vstupním parametrem I2, který musí být stejného typu dat. Relační funkce pracují s následujícími typy dat:

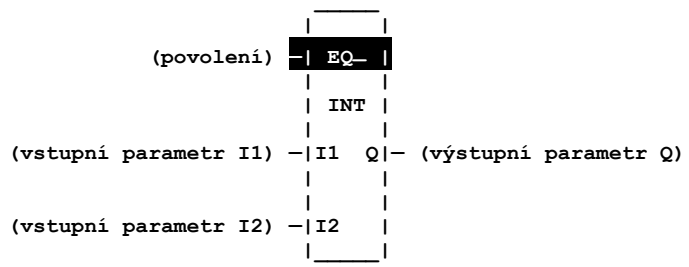
Typ dat	Popis
INT	Celé číslo se znaménkem
DINT	Celé číslo se znaménkem s dvojnásobnou délkou
REAL	Pohyblivá desetinná tečka

Poznámka

Data typu REAL je možno použít pouze u CPU řady 35x a 36x, verze 9 nebo pozdější nebo u všech verzí CPU352. Když se relační funkce používající data REAL vykoná úspěšně, bit %S0020 se nastaví do stavu ON. Vynuluje se, když některý ze vstupů bude NaN (nenumerický). Blok funkce Rozsah nepřijme typ REAL.

Výchozí typ dat je celé číslo se znaménkem. Chcete-li porovnat celá čísla se znaménkem, celá čísla se znaménkem s dvojitou délkou nebo reálná čísla, po zvolení relační funkce zvolte nový typ dat. Chcete-li provést porovnání jiných typů nebo dvou různých typů, nejdříve použijte příslušnou funkci konverze (popsaná v kapitole 11, "Funkce konverze") a změňte data na některý podporovaný typ.

Pokud vstupní parametry I1 a I2 budou souhlasit se zadaným vztahem, výstupem Q bude protékat proud a nastaví se do stavu ON (1); jinak bude nastavený do stavu OFF (0).



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, operace se provede.
I1	I1 obsahuje konstantu nebo adresu hodnoty první porovnávané hodnoty. (I1 je levá strana relační rovnice, jako v I1 < I2).
I2	I2 obsahuje konstantu nebo adresu hodnoty druhé porovnávané hodnoty. (I2 je pravá strana relační rovnice, jako v I1 < I2).
Q	Výstup Q přejde do jedničky, když I1 a I2 budou odpovídat zadanému vztahu.

Poznámka

I1 a I2 musí být platná čísla, tj. nesmí být NaN (nenumerická).

Rozšířený výklad

Funkce	Popis
Rovná se	Pokud se při povolení funkce hodnota na vstupu I1 bude rovnat hodnotě na vstupu I2, výstup Q přejde do jedničky.
Nerovná se	Pokud se při povolení funkce hodnota na vstupu I1 NEBUDE rovnat hodnotě na vstupu I2, výstup Q přejde do jedničky.
Větší než	Pokud při povolení funkce bude hodnota na vstupu I1 větší než hodnota na vstupu I2, výstup Q přejde do jedničky.
Větší nebo rovno	Pokud při povolení funkce bude hodnota na vstupu I1 větší nebo rovna hodnotě na vstupu I2, výstup Q přejde do jedničky.
Menší než	Pokud při povolení funkce bude hodnota na vstupu I1 menší než hodnota na vstupu I2, výstup Q přejde do jedničky.
Menší nebo rovno	Pokud při povolení funkce bude hodnota na vstupu I1 menší nebo rovna hodnotě na vstupu I2, výstup Q přejde do jedničky.

Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
I1		o	o	o	o		o	•	•	•	•†	
I2		o	o	o	o		o	•	•	•	•†	
Q	•											•

• Platná adresa nebo místo, kde funkcí může téct proud.
o Pouze platná adresa pro data INT; neplatí pro DINT nebo REAL.
† Pro operace s celým číslem se znaménkem s dvojnásobnou délkou jsou konstanty omezené na hodnoty celých čísel.

Příklad

V následujícím příkladu se provede porovnání celých čísel se znaménkem s dvojnásobnou délkou PWR_MDE a BIN_FUL, vždy když se nastaví %I0001. Pokud PWR_MDE bude menší nebo rovno BIN_FUL, cívka %Q0002 se zapne.

RANGE (INT, DINT, WORD)

Funkce RANGE se používá ke zjištění, jestli hodnota leží v rozsahu dvou čísel.

Poznámka

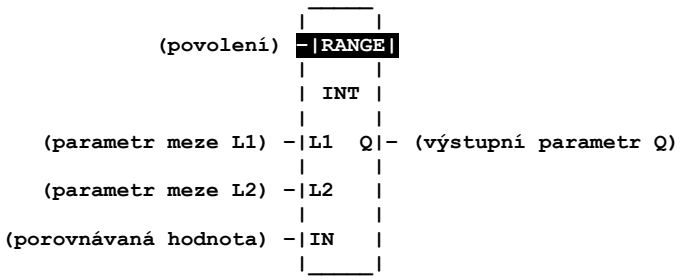
Tuto funkci je možno použít *pouze* u CPU verze 4.41 nebo pozdější.

Funkce RANGE pracuje s následujícími typy dat:

Typ dat	Popis
INT	Celé číslo se znaménkem
DINT	Celé číslo se znaménkem s dvojnásobnou délkou
WORD	Data typu slova

Výchozí typ dat je celé číslo se znaménkem; po zvolení funkce ho však lze změnit. Více informací o typu dat najdete v kapitole 2, část 2, "Organizace programu a uživatelské adresy/data".

Když funkce bude povolena, funkční blok RANGE provede porovnání hodnoty ve vstupním parametru IN s rozsahem zadaným parametry mezí L1 a L2. Když hodnota bude ležet v rozsahu zadaném L1 a L2 včetně, výstupní parametr Q se nastaví do stavu ON (1). Jinak se Q nastaví do stavu OFF (0).



Poznámka

Parametry mezí L1 a L2 představují koncové body rozsahu. K žádnému z těchto parametrů se nevztahuje minimální/maximální nebo vysoká/nízká hodnota. Proto požadovaný rozsah 0 až 100 je možno zadat přiřazením 0 do L1 a 100 do L2 nebo 0 do L2 a 100 do L1.

Parametry

Parametr	Popis
povolení	Když funkce bude povolená, operace se provede.
L1	L1 obsahuje počáteční bod rozsahu.
L2	L2 obsahuje koncový bod rozsahu.
IN	IN obsahuje hodnotu porovnávanou s rozsahem zadaným pomocí L1 a L2.
Q	Výstup Q přejde do jedničky, když hodnota v IN bude ležet uvnitř rozsahu zadaném pomocí L1 a L2 , včetně.

Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
L1		o	o	o	o		o	•	•	•	•‡	
L2		o	o	o	o		o	•	•	•	•‡	
IN		o	o	o	o		o	•	•	•		
Q	•											•

• Platná adresa nebo místo, kde funkcí může téct proud.
o Pouze platná adresa pro data INT nebo WORD; neplatí pro DINT.
† Pro operace s celým číslem se znaménkem s dvojnásobnou délkou jsou konstanty omezené na hodnoty celých čísel.

Příklad 1

V následujícím příkladu se porovnává, jestli %AI0001 leží uvnitř rozsahu zadaného pomocí dvou konstant 0 a 100.

Pravdivostní tabulka funkce RANGE				
Stav povolení %I0001	**Konstanta L1**	**Konstanta L2**	**Hodnota IN %AI0001**	**Stav Q %Q0001**
ON	100	0	< 0	OFF
ON	100	0	0 – 100	ON
ON	100	0	> 100	OFF
OFF	100	0	Nepoužívá se	OFF

Příklad 2

V tomto příkladu se %AI0001 zkontroluje, jestli leží v rozsahu zadaném hodnotami dvou registrů.

Pravdivostní tabulka funkce RANGE				
Stav povolení %I0001	**Hodnota L1 %R**	**Hodnota L2 %R**	**Hodnota IN %AI0001**	**Stav Q %Q0001**
ON	500	0	< 0	OFF
ON	500	0	0 – 500	ON
ON	500	0	> 500	OFF
OFF	500	0	Nepoužívá se	OFF

Kapitola 8

Funkce bitových operací

Funkce bitových operací provádějí porovnání, logické operace a operace přemístění s bitovými řetězci. Funkce AND, OR, XOR a NOT pracují s jedním slovem. Zbývající bitové funkce mohou pracovat s více slovy s maximální délkou řetězce 256 slov. Všechny bitové operace vyžadují data typu WORD.

I když se data musí zadávat v 16-bitových inkrementech, tyto funkce pracují s daty jako souvislým bitovým řetězcem, kde bit 1 prvního slova je bit s nejmenší váhou (LSB). Poslední bit posledního slova je bit s nevyšší váhou (MSB). Pokud například zadáte tři slova dat počínaje adresou %R0100, zpracují se jako 48 souvislých bitů.

%R0100	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1** ← **bit 1 (LSB)**
%R0101	**32**	**31**	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**
%R0102	**48**	**47**	**46**	**45**	**44**	**43**	**42**	**41**	**40**	**39**	**38**	**37**	**36**	**35**	**34**	**33**

↑

(MSB)

Poznámka

Překrývající se rozsahy vstupních a výstupních adres u víceslovních funkcí mohou vytvořit neočekávané výsledky.

V této kapitole jsou popsané následující funkce bitových operací:

Zkratka	Funkce	Popis	Strana
AND	Logický AND	Pokud bit v bitovém řetězci I1 a odpovídající bit v bitovém řetězci I2 budou 1, na odpovídajícím místě ve výstupním řetězci Q se zapíše 1	8-3
OR	Logický OR	Pokud bit v bitovém řetězci I1 a/nebo odpovídající bit v bitovém řetězci I2 budou 1, na odpovídajícím místě ve výstupním řetězci Q se zapíše 1.	8-3
XOR	Logický Exclusive OR	Pokud bit v bitovém řetězci I1 a odpovídající bit v bitovém řetězci I2 budou různé, na odpovídajícím místě ve výstupním řetězci Q se zapíše 1.	8-5
NOT	Logická inverze	Nastaví stav každého bitu ve výstupním bitovém řetězci Q tak, aby byl opačný než odpovídající bit v bitovém řetězci I1.	8-7
SHL	Posunutí doleva	Posune všechny bity slova nebo řetězce slov doleva o zadaný počet míst.	8-8
SHR	Posunutí doprava	Posune všechny bity slova nebo řetězce slov doprava o zadaný počet míst.	8-8
ROL	Rotace doleva	Provede rotaci všech bitů v řetězci o zadaný počet míst doleva.	8-10
ROR	Rotace doprava	Provede rotaci všech bitů v řetězci o zadaný počet míst doprava.	8-10
BTST	Testovat bit	Provede test bitu v bitovém řetězci, jestli je momentálně ve stavu 1 nebo 0.	8-12
BSET	Nastavit bit	Provede nastavení bitu v bitovém řetězci na 1.	8-14
BCLR	Smazat bit	Smaže bit v bitovém řetězci tak, že ho nastaví na 0.	8-14
BPOS	Pozice bitu	Zjistí pozici bitu nastaveného na 1 v bitovém řetězci.	8-16
MSKCMP	Maskované porovnání	Provede porovnání obsahu dvou samostatných bitových řetězců s možností zamaskovat zvolené bity (lze použít u CPU verze 4.5 nebo vyšší).	8-18

AND a OR (WORD)

Při každém čtení, kdy protéká proud, funkce AND nebo OR přezkoumá každý bit v bitovém řetězci I1 a odpovídající bit v bitovém řetězci I2 počínaje bitem s nejnižší váhou každého řetězce.

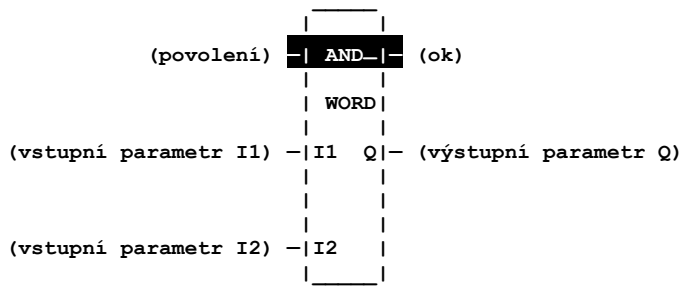
Pokud oba dva bity testované funkcí AND budou v 1, pak se do odpovídajícího místa ve výstupním řetězci Q zapíše 1. Pokud některý nebo oba tyto bity budou 0, pak se do tohoto místa ve výstupním řetězci Q zapíše 0.

Funkce AND je vhodná pro vytvoření masky nebo filtru, kde se propustí jen určité bity (ty, které jsou opačné vzhledem k masce) a všechny ostatní se nastaví na 0. Funkci je také možno použít k vynulování vybrané oblasti paměti slov vykonáním funkce logický AND bitů s jiným bitovým řetězcem, o kterém je známo, že obsahuje samé 0. Zadané bitové řetězce I1 a I2 se mohou překrývat.

Pokud některý nebo oba z těchto dvou bitů testovaných funkcí OR bude v 1, pak se do odpovídajícího místa ve výstupním řetězci Q zapíše 1. Pokud oba tyto bity budou 0, pak se do tohoto místa ve výstupním řetězci Q zapíše 0.

Funkce OR je vhodná pro kombinování řetězců a k řízení mnoha výstupů použitím jednoduché logické struktury. Funkce je ekvivalentem dvou reléových paralelních kontaktů násobených počtem bitů v řetězci. Lze jí použít k řízení kontrolek přímo ze vstupních stavů nebo překopírování stavů blikajících kontrolek.

Funkce propustí proud doprava vždy, když přijde proud.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, operace se provede.
I1	I1 obsahuje konstantu nebo adresu prvního slova prvního řetězce.
I2	I2 obsahuje konstantu nebo adresu prvního slova druhého řetězce.
ok	Výstup ok bude v jedničce, když skrz vstup povolení poteče proud.
Q	Vstup Q obsahuje výsledek operace.

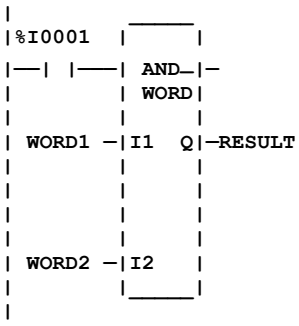
Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
I1		•	•	•	•	•	•	•	•	•	•	
I2		•	•	•	•	•	•	•	•	•	•	
ok	•											•
Q		•	•	•	•	•†	•	•	•	•		

• Platná adresa nebo místo, kde funkcí může téct proud.
† Pouze %SA, %SB nebo %SC; %S nelze použít.

Příklad

V následujícím příkladu se provede testování 16-bitových řetězců představovaných názvy WORD1 a WORD2 vždy, když bude nastavený vstup %I0001. Výsledek logického AND se zapíše do výstupního řetězce RESULT.



WORD1	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**1**	**0**	**0**	**0**
WORD2	**1**	**1**	**0**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**
RESULT	**0**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**

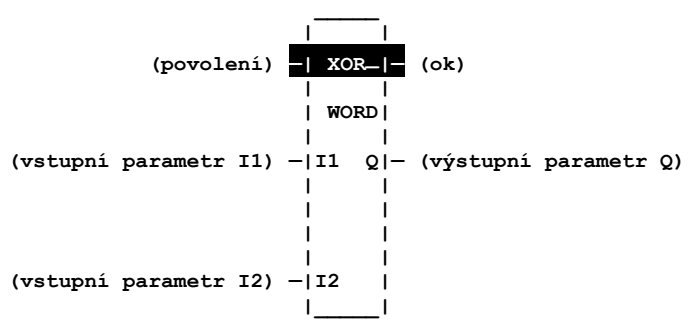
XOR (WORD)

Funkce Exclusive OR (XOR) se používá k porovnání jednotlivých bitů v bitovém řetězci I1 s odpovídajícími bity v řetězci I2. Pokud bity budou různé, na odpovídající místo ve výstupním bitovém řetězci se zapíše 1.

Při každém čtení, kdy protéká proud, funkce provede test každého bitu v bitovém řetězci I1 a odpovídající bit v bitovém řetězci I2 počínaje bitem s nejnižší váhou každého řetězce. Pokud u těchto dvou testovaných bitů bude pouze jeden z nich v 1, pak se do odpovídajícího místa ve výstupním řetězci Q zapíše 1. Funkce XOR propustí proud doprava vždy, když přijde proud.

Pokud řetězec I2 a výstupní řetězec Q budou začínat na stejné adrese, 1 umístěná v řetězci I1 bude mít za následek, že odpovídající bit v řetězci I2 se bude střídat mezi 0 a 1 a bude měnit stav při každém čtení, dokud bude procházet proud. Delší cykly je možno naprogramovat pulsním modulováním proudu dvojnásobku požadované rychlosti blikání; puls proudu musí být dlouhý jedno čtení (jednorázová cívka nebo samočinně nulovaný časovač).

Funkce XOR je vhodná pro rychlé porovnání dvou bitových řetězců nebo k blokování skupiny bitů rychlostí jednoho stavu ON na dvě čtení.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, operace se provede.
I1	I1 obsahuje konstantu nebo adresu hodnoty prvního slova pro vykonání funkce XOR.
I2	I2 obsahuje konstantu nebo adresu hodnoty druhého slova vykonání funkce XOR.
ok	Výstup ok bude v jedničce, když skrz vstup povolení poteče proud.
Q	Výstup Q obsahuje výsledek funkce XOR mezi I1 a I2.

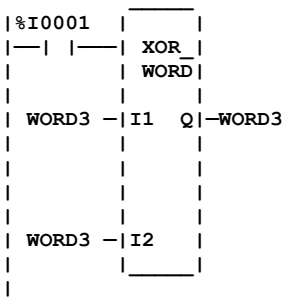
Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
I1		•	•	•	•	•	•	•	•	•	•	
I2		•	•	•	•	•	•	•	•	•	•	
ok	•											•
Q		•	•	•	•	•†	•	•	•	•		

• Platná adresa nebo místo, kde funkcí může téct proud.
† Pouze %SA, %SB nebo %SC; %S nelze použít.

Příklad

V následujícím příkladu se vynuluje bitový řetězec představovaný názvem WORD3 vždy, když bude nastavený vstup %I0001.



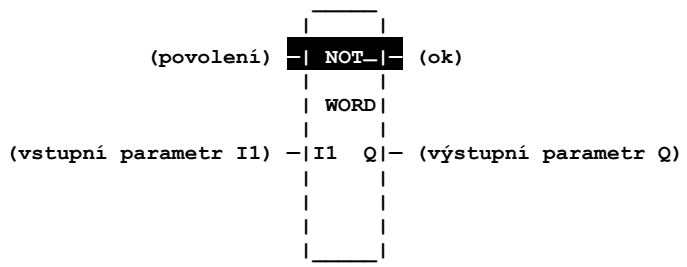
I1 (WORD3)	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**1**	**0**	**0**	**0**
I2 (WORD3)	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**1**	**0**	**0**	**0**

Q (WORD3)	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

NOT (WORD)

Funkce NOT se používá k nastavení stavu jednotlivých bitů výstupního bitového řetězce Q do opačného stavu než odpovídající bit v bitovém řetězci I1.

Při každém čtení, kdy bude procházet proud, se všechny bity změní a tak vytvoří ve výstupním bitovém řetězci Q logický komplement k I1. Funkce propustí proud doprava vždy, když přijde proud.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, operace se provede.
I1	I1 obsahuje konstantu nebo adresu hodnoty slova, které se má negovat.
ok	Výstup ok bude v jedničce, když skrz vstup povolení poteče proud.
Q	Výstup Q obsahuje operace NOT (negace) s řetězcem I1.

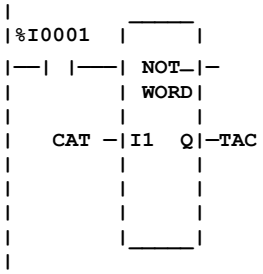
Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
I1		•	•	•	•	•	•	•	•	•	•	
ok	•											•
Q		•	•	•	•	•†	•	•	•	•		

• Platná adresa nebo místo, kde funkcí může téct proud. † Pouze %SA, %SB nebo %SC; %S nelze použít.

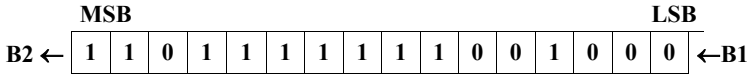
Příklad

V následujícím příkladu se při každém nastavení vstupu %I0001 bitový řetězce představovaný názvem TAC nastaví na převrácený stav bitového řetězce CAT.

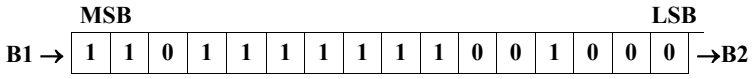


SHL a SHR (WORD)

Funkce posunutí doleva (SHL) se používá k posunutí všech bitů ve slově nebo skupiny slov doleva o zadaný počet míst. Když se vyskytne posunutí, zadaný počet bitů se posune ven z výstupního řetězce doleva. Jak se bity posunou na vyšším konci řetězce ven, současně se stejný počet bitů na druhém konci posune dovnitř.



Funkce posunutí doprava (SHR) se používá k posunutí všech bitů slova nebo skupiny slov doprava o zadaný počet míst. Když se vyskytne posunutí, zadaný počet bitů se posune ven z výstupního řetězce doprava. Jak se bity posunou na nižším konci řetězce ven, současně se stejný počet bitů na druhém konci posune dovnitř.



Pro obě funkce je možno zvolit řetězec s délkou 1 až 256 slov.

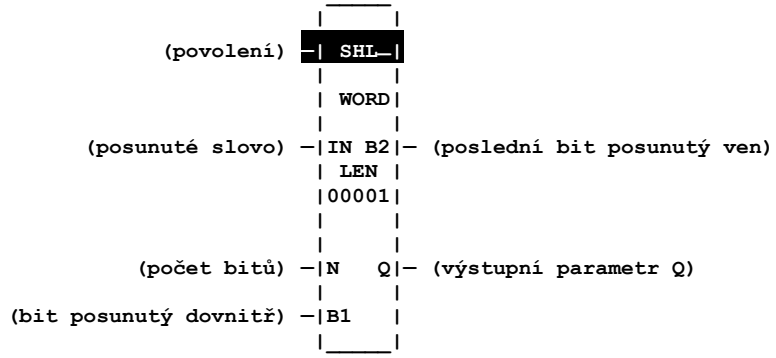
Pokud počet bitů (N), o které se má provést posunutí, bude větší než počet bitů pole (LEN) * 16 nebo pokud počet bitů, o které se má provést posunutí, bude nula, pak se pole (Q) zaplní kopiemi vstupního bitu (B1) a vstupní bit se zkopíruje do výstupního proudu (B2). Pokud počet bitů posunutí bude nula, pak se neprovede žádné posunutí; vstupní pole se zkopíruje do výstupního pole; a vstupní bit (B1) se zkopíruje do proudu.

Bity, které se posouvají dovnitř na začátku řetězce, se definují pomocí vstupního parametru B1. Pokud jako počet posouvaných bitů bude zadaná délka větší než 1, každý bit se zaplní stejnou hodnotou (0 nebo 1). To může být:

- Booleovský výstup jiné programové funkce.
- Samé 1. K tomu účelu jako povolení pro vstup B1 použijte speciální adresu z názvem ALW_ON.
- Samé 0. K tomu účelu jako povolení pro vstup B1 použijte speciální adresu z názvem ALW_OFF.

Pokud počet bitů zadaný pro posunutí nebude nula, funkce SHL nebo SHR budou přenášet proud doprava.

Výstup Q bude posunutou kopií vstupního řetězce. Pokud chcete posunout vstupní řetězec, výstupní parametr Q2 musí používat stejné paměťové místo jako vstupní parametr IN. Celý posunutý řetězec se zapíše při každém čtení, kdy bude protékat proud. Výstup B2 je poslední bit posunutý ven. Pokud například byly posunuté čtyři bity, B2 bude čtvrtý bit posunutý ven.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, provede se posunutí.
IN	IN obsahuje první posouvané slovo.
N	N obsahuje počet míst (bitů), o které se pole má posunout
B1	B1 obsahuje hodnotu bitu, který se má posunout do pole.
B2	B2 obsahuje hodnotu posledního bitu, který se má z pole posunout ven.
Q	Výstup Q obsahuje první slovo posunutého pole.
LEN	LEN je počet slov v poli, které se má posunout.

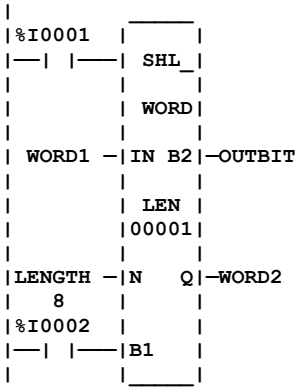
Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN		•	•	•	•	•	•	•	•	•		
N		•	•	•	•		•	•	•	•	•	
B1	•											
B2	•											•
Q		•	•	•	•	•†	•	•	•	•		

• Platná adresa nebo místo, kde funkcí může téct proud.
† Pouze %SA, %SB nebo %SC; %S nelze použít.

Příklad

V následujícím příkladu se při každém nastavení vstupu %I0001 ve výstupním bitovém řetězci s názvem WORD2 vytvoří kopie slova WORD1 posunutého doleva o počet bitů představovaných názvem LENGTH. Výsledné otevřené bity na začátku vstupního řetězce se nastaví na hodnotu %I0002.



ROL a ROR (WORD)

Funkce rotace doleva (ROL) se používá k rotaci bitů v řetězci o zadaný počet míst doleva. Když se vyskytne rotace, provede se rotace vstupního řetězce o zadaný počet bitů doleva a zpět do řetězce zprava.

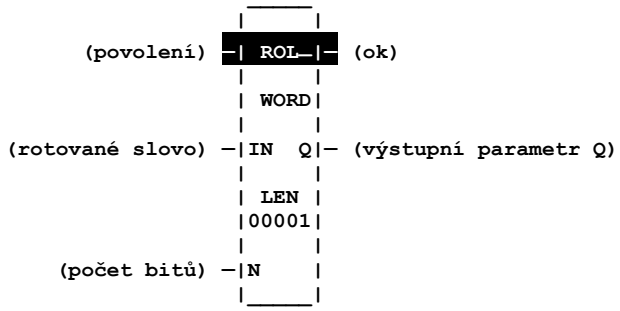
Funkce rotace doprava (ROR) provede rotaci bitů v řetězci doprava. Když se vyskytne rotace, provede se rotace vstupního řetězce o zadaný počet bitů doprava a zpět do řetězce zleva.

Pro obě funkce je možno zvolit řetězec s délkou 1 až 256 slov.

Počet míst zadaných pro rotaci musí být větší než nula a menší než počet bitů v řetězci. Jinak se nevykoná žádný pohyb a nebude se generovat žádný proud.

Pokud počet bitů zadaných pro rotaci nebude větší než celková délka řetězce a nebude menší než nula, funkce ROL nebo ROR přenesou proud doprava.

Výsledek se zapíše do výstupního řetězce Q. Pokud chcete provést rotaci vstupního řetězce, výstupní parametr Q2 musí používat stejné paměťové místo jako vstupní parametr IN. Celý rotovaný řetězec se zapíše při každém čtení, kdy bude protékat proud.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, provede se rotace.
IN	IN obsahuje první rotované slovo.
N	N obsahuje počet míst (bitů), o kolik se má provést rotace pole.
ok	Výstup ok bude v jedničce, když funkcí rotace poteče proud a délka rotace nebude větší než velikost pole.
Q	Výstup Q obsahuje první slovo rotovaného pole.
LEN	LEN je počet slov v poli, které se má rotovat.

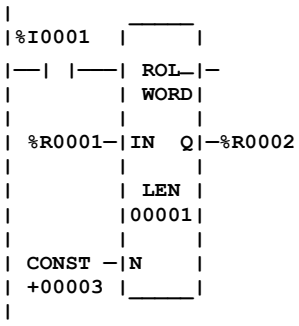
Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN		•	•	•	•	•	•	•	•	•		
N		•	•	•	•		•	•	•	•	•	
ok	•											•
Q		•	•	•	•	•†	•	•	•	•		

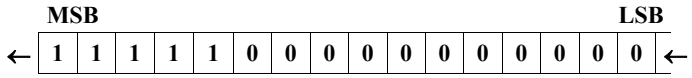
• Platná adresa nebo místo, kde funkcí může téct proud.
† Pouze %SA, %SB nebo %SC; %S nelze použít.

Příklad

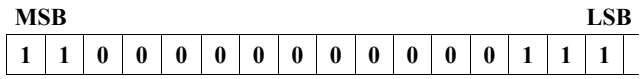
V následujícím příkladu se při každém nastavení vstupu %I000 provede rotace vstupního bitového řetězce %R0001 o 3 bity a výsledek se zapíše do %R0002. Po vykonání této funkce vstupní bitový řetězec %R0001 zůstane nezměněný. Pokud se pro IN a Q použije stejná adresa, provede se rotace na místě.



%R0001:



%R0002 (po nastavení %I0001):

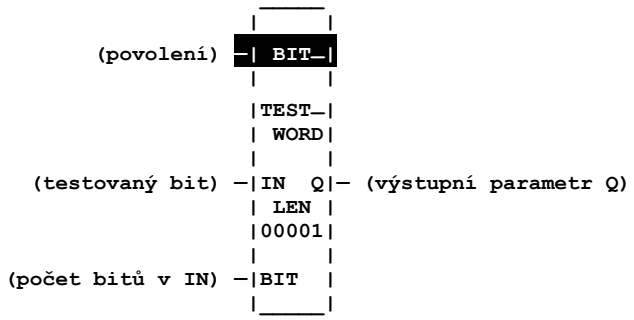


BTST (WORD)

Funkce testování bitu (BTST) se používá k testování bitu uvnitř bitového řetězce, jestli se právě nachází ve stavu 1 nebo 0. Výsledek testu se zapíše do výstupu Q.

Při každém cyklu, kdy prochází proud, funkce BTST nastaví svůj výstup do stejného stavu jako zadaný bit. Pokud se k zadání čísla bitu použije registr místo konstanty, stejný funkční blok může v po sobě jdoucích cyklech testovat různé bity. Pokud hodnota BIT bude mimo rozsah (1 ≤ BIT ≤ (16 * LEN)), pak se Q nastaví do stavu OFF.

Je možno zvolit řetězec s délkou 1 až 256 slov.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, provede se testování bitu.
IN	IN obsahuje první slovo dat použitých v operaci.
BIT	BIT obsahuje číslo bitu v IN, který se má testovat. Platný rozsah je (1 ≤ BIT ≤ (16 * LEN)).
Q	Výstup Q bude v jedničce, pokud testovaný bit byl 1.
LEN	LEN je počet slov v testovaném řetězci.

Poznámka

Když budete používat funkci Testování bitu, Nastavení bitu, Smazání bitu nebo Pozice bitu, bity budou očíslované 1 až 16, ***NE*** 0 až 15, jak bylo ukázáno výše.

Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN		•	•	•	•	•	•	•	•	•		
BIT		•	•	•	•		•	•	•	•	•	
Q	•											•

• Platná adresa nebo místo, kde funkcí může téct proud.

Příklad

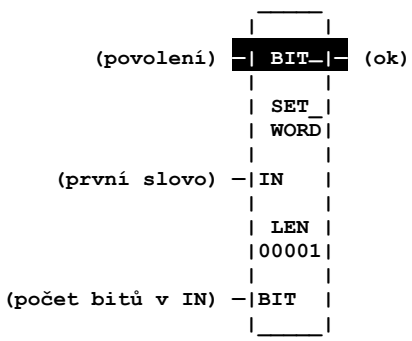
V následujícím příkladu se při každém nastavení vstupu %I0001 provede testování bitu na místě nacházejícím se na adrese PICKBIT. Bit je součástí řetězce PRD_CDE. Pokud bude ve stavu 1, výstup Q přenese proud a cívka %Q0001 se zapne.

BSET a BCLR (WORD)

Funkce nastavení bitu (BSET) se používá k nastavení bitu v bitovém řetězci na 1. Funkce smazání bitu (BCLR) se používá ke smazání bitu v řetězci nastavením tohoto bitu na 0.

Při každém cyklu, kdy bude procházet proud, funkce nastaví zadaný bit na 1 v případě funkce BSET nebo na 0 v případě funkce BCLR. Pokud se k zadání čísla bitu použije proměnná (registr) místo konstanty, stejný funkční blok může v po sobě jdoucích cyklech nastavovat různé bity.

Je možno zvolit řetězec s délkou 1 až 256 slov. Pokud hodnota pro BIT nebude mimo rozsah ($1 \leq BIT \leq (16 * LEN)$), funkce bude přenášet proud doprava. Jinak výstup ok bude nastavený do stavu OFF.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, bitová operace se provede.
IN	IN obsahuje první slovo dat použitých v operaci.
BIT	BIT obsahuje číslo bitu v IN, který se má nastavit nebo vynulovat. Platný rozsah je ($1 \leq BIT \leq (16 * LEN)$).
ok	Výstup ok bude v jedničce, když skrz vstup povolení poteče proud.
LEN	LEN je počet slov v bitovém řetězci.

Poznámka

Když budete používat funkci Testování bitu, Nastavení bitu, Smazání bitu nebo Pozice bitu, bity budou očíslované 1 až 16, ***NE*** 0 až 15, jak bylo ukázáno výše.

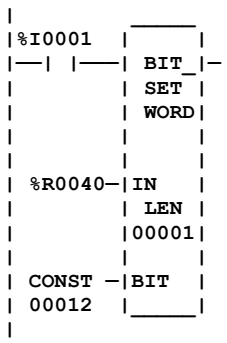
Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN		•	•	•	•	†	•	•	•	•		
BIT		•	•	•	•		•	•	•	•	•	
ok	•											•

• Platná adresa nebo místo, kde funkcí může téct proud.
† Pouze %SA, %SB nebo %SC; %S nelze použít.

Příklad

V následujícím příkladu se při každém nastavení vstupu %I0001 bit 12 řetězce začínajícího na adrese %R0040 nastaví na 1.



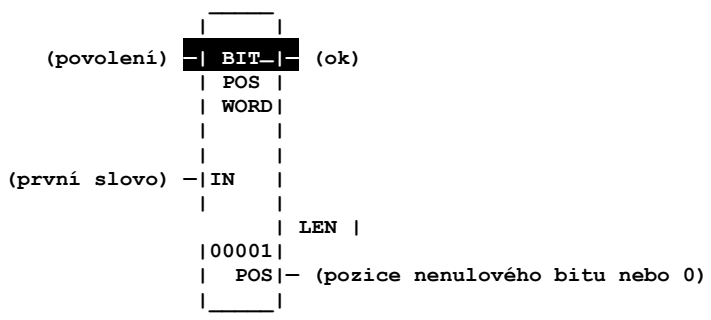
BPOS (WORD)

Funkce pozice bitu (BPOS) se používá k nalezení pozice bitu nastaveného na 1 v bitovém řetězci.

Při každém cyklu, kdy funkcí bude procházet proud, funkce provede čtení bitového řetězce začínajícího v IN. Když funkce zastaví čtení, buď byl nalezen bit rovnající se 1 nebo se provedlo čtení celého řetězce.

POS bude nastaveno na pozici v bitovém řetězci prvního nenulového bitu; pokud se nenajde žádný nenulový bit, POS se nastaví na nulu.

Je možno zvolit řetězec s délkou 1 až 256 slov. Funkce propustí proud doprava vždy, když vstup povolení bude ve stavu ON.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, provede se operace hledání bitu.
IN	IN obsahuje první slovo dat použitých v operaci.
ok	Výstup ok bude v jedničce, když skrz vstup povolení poteče proud.
POS	Pozice prvního nalezeného nenulového bitu nebo nula, pokud se nenalezne žádný nenulový bit.
LEN	LEN je počet slov v bitovém řetězci.

Poznámka

Když budete používat funkci Testování bitu, Nastavení bitu, Smazání bitu nebo Pozice bitu, bity budou očíslované 1 až 16, ***NE*** 0 až 15, jak bylo ukázáno výše.

Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN		•	•	•	•	•	•	•	•	•		
POS		•	•	•	•		•	•	•	•		
ok	•											•

• Platná adresa nebo místo, kde funkcí může téct proud.

Příklad

V následujícím příkladu se při nastaveném vstupu %I0001 bude prohledávat bitový řetězec začínající na %M0001, dokud se nenalezne bit rovnající se 1 nebo se neprohledá 6 slov. Cívka %Q0001 se zapne. Pokud se najde bit rovnající se 1, jeho pozice v bitovém řetězci se zapíše do %AQ001. Pokud %I0001 bude nastaveno, bit %M0001 bude 0 a bit %M0002 bude 1, pak hodnota zapsaná do %AQ001 bude 2.

MSKCMP (WORD, DWORD)

Funkce maskovaného porovnání (MSKCMP) (*k dispozici u CPU verze 4.41 nebo pozdější*) se používá k porovnání obsahu dvou samostatných bitových řetězců s možností zamaskovat zvolené bity. Délka porovnávaného bitového řetězce se zadává v parametru LEN (kde hodnota LEN udává počet 16-bitových slov pro funkci MSKCMPW a 32-bitových slov pro funkci MSKCMPD).

Když logika řídicí vstup pro povolení do funkce propustí proud na vstup povolení (EN), funkce začne porovnávat bity v prvním řetězci s odpovídajícími bity v druhém řetězci. Porovnávání bude pokračovat, dokud se nezjistí neshoda nebo dokud se nedosáhne konce řetězce.

Vstup BIT se používá k uložení čísla bitu, kde má začít další porovnávání (přičemž *0* udává první bit v řetězci). Výstup BN se používá k uložení čísla bitu, kde se vyskytlo poslední porovnávání (přičemž *1* udává první bit v řetězci). Použití stejné adresy pro BIT a BN bude mít za následek, že porovnávání začne na pozici dalšího bitu po neshodě; nebo pokud budou při dalším vyvolání funkčního bloku všechny bity porovnané úspěšně, porovnávání začne na začátku.

Pokud budete chtít začít další porovnávání na některé jiné pozici v řetězci, můžete do BIT a BN zapsat různé adresy. Pokud hodnota BIT bude pozice, která je za koncem řetězce, BIT se před začátkem dalšího porovnávání resetuje na 0.

Když všechny bity v I1 a I2 jsou stejné

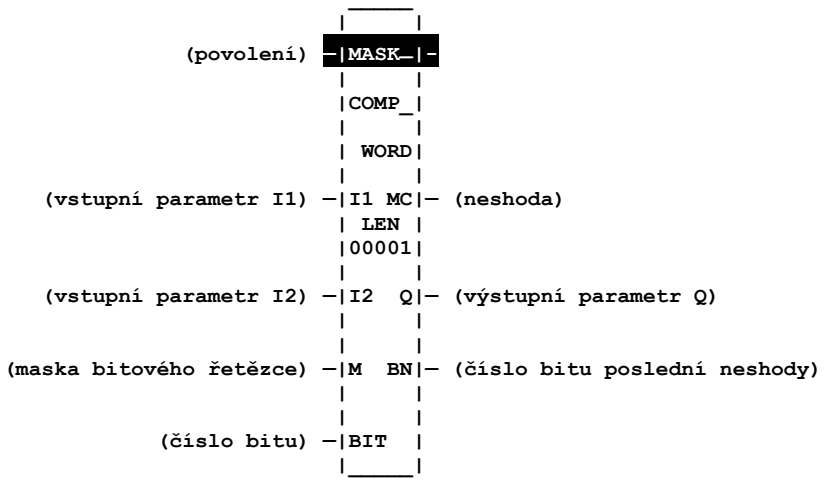
Pokud všechny odpovídající bity v řetězcích I1 a I2 budou souhlasit, funkce nastaví výstup MC "neshoda" na 0 a BN na nejvyšší číslo bitu ve vstupním řetězci. Porovnávání se pak zastaví. Při dalším vyvolání MSKCMPW se resetuje na 0.

Když se zjistí neshoda

Když dva aktuálně porovnávané bity nebudou stejné, funkce zkontroluje patřičně očíslovaný bit v řetězci M (masce). Pokud bit masky bude *1*, porovnávání bude pokračovat, dokud nedosáhne další neshody nebo konce vstupních řetězců.

Pokud se zjistí neshoda a odpovídající bit masky bude 0, funkce provede následující:

1. Nastaví odpovídající bit masky v M na 1.
2. Nastaví výstup neshody (MC) na 1.
3. Aktualizuje výstupní bitový řetězec Q tak, aby souhlasil s novým obsahem řetězce masky M.
4. Nastaví výstup čísla bitu (BN) na číslo neshodného bitu.
5. Zastaví porovnávání.



Parametry

Parametr	Popis
povolení	Povolovací logika k povolení funkce.
I1	Adresa prvního bitového řetězce, který se má porovnat.
I2	Adresa druhého bitového řetězce, který se má porovnat.
M	Adresa masky bitového řetězce.
BIT	Adresa čísla bitu, kde má začít další porovnávání.
MC	Uživatelská logika ke zjištění, jestli se vyskytla neshoda.
Q	Výstupní kopie bitového řetězce masky (M).
BN	Číslo bitu, kde se vyskytla poslední neshoda.
LEN	LEN je počet slov v bitovém řetězci.

Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
I1		o	o	o	o	o	o	•	•	•		
I2		o	o	o	o	o	o	•	•	•		
M		o	o	o	o	o†	o	•	•	•		
BIT		•	•	•	•	•	•	•	•	•	•	
LEN											•‡	
MC	•											•
Q		o	o	o	o	o†	o	•	•	•		
BN		•	•	•	•	•	•	•	•	•		

• Platná adresa nebo místo, kde funkcí může téct proud.
o Pouze platná adresa pro data WORD; neplatí pro DWORD.
† Pouze %SA, %SB %SC; %S nelze použít.
‡ Maximální konstantní hodnota 4095 pro WORD a 2047 pro DWORD.

Příklad

V následujícím příkladu se po prvním čtení vykoná funkční blok MSKCMPW. Řetězce %M0001 až %M0016 se porovnají s řetězci %M0017 až %M0032. Řetězce %M0033 až %M0048 obsahují hodnotu masky. Hodnota v řetězci %R0001 určuje, na které pozici má porovnávání v těchto dvou vstupních řetězcích začít. Obsah výše uvedených adres před vykonáním funkčního bloku je následující:

(I1) – %M0001 = 6C6Ch =

0	1	1	0	1	1	0	0	0	1	1	0	1	1	0	0

(I2) – %M0017 = 606Fh =

0	1	1	0	1	1	0	1	0	1	1	0	1	1	1	1

(M/Q) – %M0033 = 000Fh =

0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

(BIT/BN) – %R0001 = 0
(MC) – %Q0001 = OFF

Obsah těchto adres po vykonání funkčního bloku bude následující:

(I1) – %M0001 =

0	1	1	0	1	1	0	0	0	1	1	0	1	1	0	0

(I2) – %M0017 =

0	1	1	0	1	1	0	1	0	1	1	0	1	1	1	1

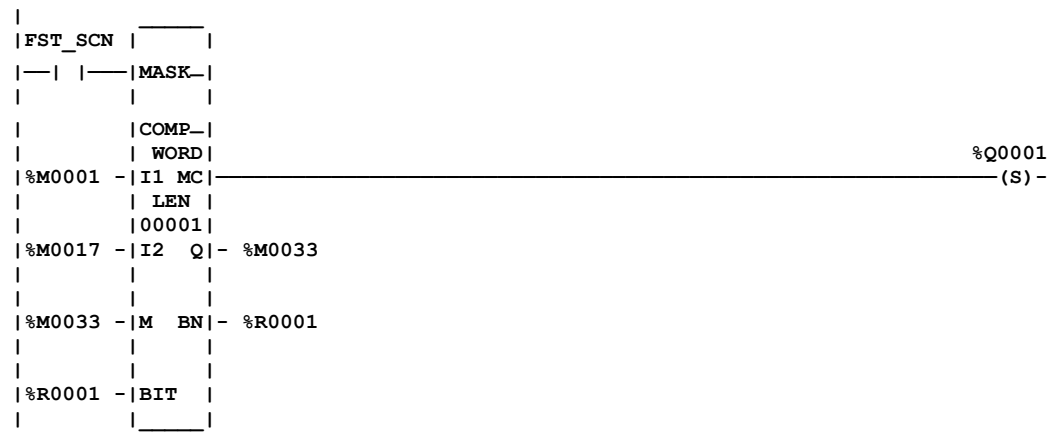
(M/Q) – %M0033 =

0	0	0	0	0	0	0	1	0	0	0	0	1	1	1	1

(BIT/BN) – %R0001 = 8

(MC) – %Q0001 = ON

Reprezentace žebříkového schématu



Všimněte si, že ve výše uvedeném příkladu jsme k vynucení jednoho a právě jednoho vykonání použili kontakt FST_SCN; jinak by se maskované porovnávání opakovalo a nemuselo by se dosáhnout požadovaných výsledků.

Kapitola 9

Funkce přesunu dat

Funkce přesunu dat zajišťují základní funkce přesunu dat. Tato kapitola popisuje následující funkce přesunu dat:

Zkratka	Funkce	Popis	Strana
MOVE	Přesunutí	Zkopíruje data po jednotlivých bitech. Maximální přípustná délka je 256 slov s výjimkou MOVE_BIT, kdy délka je 256 bitů. Data je možno přesouvat do různých typů dat bez předchozí konverze.	9-2
BLKMOV	Přesunutí bloku	Zkopíruje blok sedmi konstant do zadaného paměťového místa. Konstanty se zapisují jako součást funkce.	9-5
BLKCLR	Smazání bloku	Nahradí obsah bloku dat samými nulami. Tuto funkci je možno použít ke smazání oblasti bitové paměti (%I, %Q, %M, %G nebo %T) nebo slovní paměti (%R, %AI nebo %AQ). Maximální přípustná délka je 256 slov.	9-7
SHFR	Posunutí registru	Posune jedno nebo více datových slov do tabulky. Maximální přípustná délka je 256 slov.	9-8
BITSEQ	Bitový sekvenční přepínač	Vykoná posunutí bitové sekvence v poli bitů. Maximální přípustná délka je 256 slov.	9-11
COMMREQ	Požadavek na komunikaci.	Umožní, aby program komunikoval s inteligentním modulem, například komunikačním modulem Genius nebo programovatelným koprocesorovým modulem.	9-14

MOVE (BIT, INT, WORD, REAL)

Funkce MOVE se používá ke zkopírování dat (jako jednotlivé bity) z jednoho místa na jiné. Protože se data kopírují v bitovém formátu, nové místo nemusí být stejného typu dat jako původní místo.

Funkce MOVE má dva vstupní parametry a dva výstupní parametry. Když funkcí bude protékat proud, funkce provede kopírování dat ze vstupního parametru IN do výstupního parametru Q bitově. Pokud se provádí přesunutí z jednoho místa diskrétní paměti do jiného (například z paměti %I do paměti %T), přechodová informace spojená s diskrétními paměťovými členy se aktualizuje tak, aby indikovala, jestli operace MOVE u některých diskrétních paměťových členů způsobí změnu stavu. Data ve vstupním parametru se nezmění, pokud se ve zdrojovém a cílovém umístění nebude vyskytovat překrytí.

U typu BIT je situace jiná. Pokud pole BIT zadané v parametru Q nebude zahrnovat všechny bity v bajtu, přechodové bity spojené s tímto bajtem (které nejsou v tomto poli) se vynulují, když funkcí MOVE_BIT bude protékat proud.

Vstup IN může být buď adresa přesunovaných dat nebo konstanta. Pokud bude zadána konstanta, pak se do místa zadaného výstupní adresou zapíše konstanta. Pokud například konstanta zadaná v IN bude 4, pak se do paměťového místa zadaného v Q zapíše 4. Pokud délka bude větší než 1 a než zadaná konstanta, pak se konstanta zapíše do paměťového místa zadaného v Q a následujících míst až po zadanou délku. Pokud například konstanta zadaná v IN bude 9 a délka bude 4, pak se do paměťového místa zadaného v Q a třech následujících míst zapíše 9.

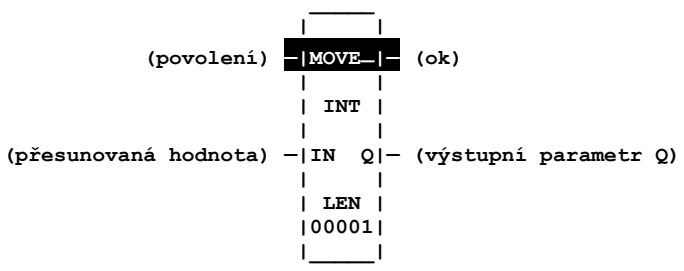
Operand LEN udává počet:

- Slov, které se mají přesunout pomocí MOVE_INT a MOVE_WORD.
- Bitů, které se mají přesunout pomocí MOVE_BIT.
- Reálných čísel, která se mají přesunout pomocí MOVE_REAL.

Poznámka

Data typu REAL je možno použít pouze u CPU řady 35x a 36x, verze 9 nebo pozdější, nebo u všech verzí CPU352.

Funkce propustí proud doprava vždy, když přijde proud.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, provede se přesunutí.
IN	IN obsahuje přesouvanou hodnotu. V případě MOVE_BIT je možno použít libovolnou diskrétní adresu; nemusí být spojená s bajtem. Avšak 16 bitů počínaje zadanou adresou se zobrazí přímo.
ok	Výstup ok bude v jedničce vždy, když funkce bude povolená.
Q	Když se provede přesunutí, do Q se zapíše hodnota z IN. V případě MOVE_BIT je možno použít libovolnou diskrétní adresu; nemusí být spojená s bajtem. Avšak 16 bitů počínaje zadanou adresou se zobrazí přímo.
LEN	LEN udává počet slov nebo bitů, které se mají přesunout. V případě MOVE_WORD a MOVE_INT LEN musí být 1 až 256 slov. V případě MOVE_BIT, když IN je konstanta, LEN musí být 1 až 16 bitů; jinak LEN musí být 1 až 256.

Poznámka

U CPU řady 351, 352 a 36x funkce MOVE_INT a MOVE_WORD nepodporují překrývání parametrů IN a Q, kde adresa IN je menší než adresa Q. Například s hodnotami IN=%R0001, Q=%R0004, LEN=5 (slov), bude obsah %R0007 a %R0008 neurčitý; pokud se však použijí hodnoty Q=%R0001, IN=%R0004, LEN=5 (slov), obsah bude platný.

Všimněte si také, že pouze CPU řady 35x a 36x (verze 9 a pozdější plus všechny verze CPU352) mají v současné době možnost pohyblivé desetinné tečky a proto jsou jediné, které mohou vykonat funkci MOVE_REAL.

Platné typy paměti

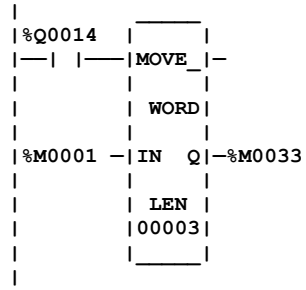
Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN		•	•	•	•	o	•	•	•	•	•	
ok	•											•
Q		•	•	•	•	o†	•	•	•	•		

Poznámka: U dat typu REAL jsou jediné platné typy %R, %AI a %AQ.

- • Platná adresa pro data typu BIT, INT nebo WORD nebo místo, kudy proud může téct skrz funkci. U funkce MOVE_BIT diskrétní uživatelské adresy %I, %Q, %M a %T nemusí být spojené s bajtem.
- o Platná adresa pouze pro data typu BIT nebo WORD; neplatí pro INT.
- † Pouze %SA, %SB %SC; %S nelze použít.

Příklad 1

Když vstup pro povolení %Q0014 bude ve stavu ON, z paměťového místa %M0001 se přesune 48 bitů do paměťového místa %M0033. I když cílové umístění překrývá zdrojové umístění o 16 bitů, přemístění se provede správně (s výjimkou CPU 351 a 352, jak bylo uvedeno dříve).



Před použitím funkce přesunutí:

INPUT (%M0001 až %M0048)

1

%M	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0
%M	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
%M	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Po použití funkce přesunutí:

INPUT (%M0033 až %M0080)

33

%M	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0
%M	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
%M	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Příklad 2

V tomto příkladu se při každém nastavení %I0001 tři bity %M0001, %M0002 a %M0003 přesunou do %M0100, %M0101 a %M0102. Cívka %Q0001 se zapne.

BLKMOV (INT, WORD, REAL)

Funkci přesunutí bloku (BLKMOV) použijte pro zkopírování bloku sedmi konstant na zadané místo.

Poznámka

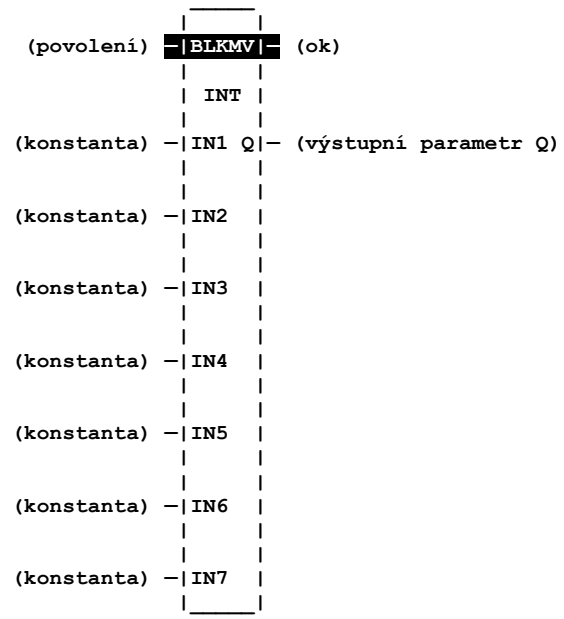
Data typu REAL je možno použít pouze u CPU řady 35x a 36x, verze 9 nebo pozdější, nebo u všech verzí CPU352.

Funkce BLKMOV má osm vstupních parametrů a dva výstupní parametry. Když funkcí poteče proud, funkce zkopíruje konstanty do po sobě jdoucích míst počínaje umístěním zadaným ve výstupu Q. Výstup Q nelze přivést jako vstup do další programové funkce.

Poznámka

U funkce BLKMOV_INT se hodnoty IN1 – IN7 zobrazí jako dekadická čísla se znaménkem. U funkce BLKMOV_WORD se hodnoty IN1 – IN7 zobrazí v hexadecimálním tvaru. U funkce BLKMOV_REAL se hodnoty IN1 – IN7 zobrazí ve formátu REAL.

Funkce propustí proud doprava vždy, když přijde proud.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, provede se přesunutí bloku.
IN1— IN7	IN1 až IN7 obsahuje sedm konstant.
ok	Výstup ok bude v jedničce vždy, když funkce bude povolená.
Q	Výstup Q obsahuje první celé číslo přesunovaného pole. IN1 se přesune do Q.

Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN1 — IN7											•	
ok	•											•
Q		•	•	•	•	o†	•	•	•	•		

Poznámka: U dat typu REAL jsou jediné platné typy %R, %AI a %AQ.

• Platná adresa pro místo, kudy proud může téct skrz funkci.

o Pouze platná adresa pro data WORD; neplatí pro INT nebo REAL.

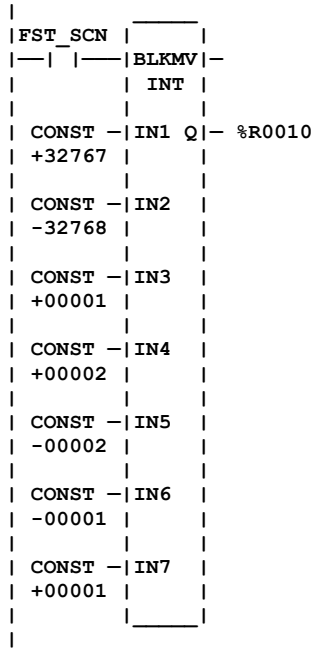
† Pouze %SA, %SB %SC; %S nelze použít.

Poznámka

Funkce plovoucí desetinné tečky existuje pouze CPU řady 35x a 36x, verze 9 nebo pozdější nebo u všech verzí CPU352. Tyto CPU 90-30 jsou jediné, které jsou schopné vykonat BLKMV_REAL.

Příklad

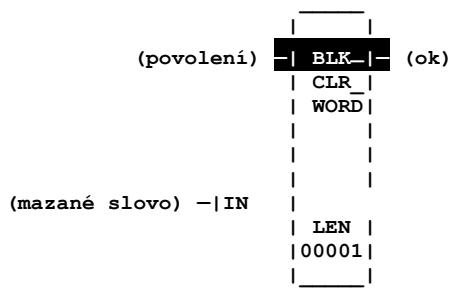
V následujícím příkladu, když vstup pro povolení s názvem FST_SCN bude ve stavu ON, funkce BLKMOV provede zkopírování sedmi vstupních konstant do paměťových míst %R0010 až %R0016.



BLKCLR (WORD)

Funkce smazání bloku (BLKCLR) se používá k vyplnění zadaného bloku dat samými nulami.

Funkce BLKCLR má dva vstupní parametry a jeden výstupní parametr. Když funkcí bud procházet proud, funkce zapíše nuly do paměťových míst počínaje adresou zadanou v IN. Pokud data, která se mají smazat, budou z diskrétní paměti (%I, %Q, %M, %G nebo %T), přechodová informace spojená s těmito adresami se také smaže. Funkce přenáší proud doprava.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, pole se smaže.
IN	IN obsahuje první slovo pole, které se má smazat.
ok	Výstup ok bude v jedničce vždy, když funkce bude povolená.
LEN	LEN musí být 1 až 256 slov.

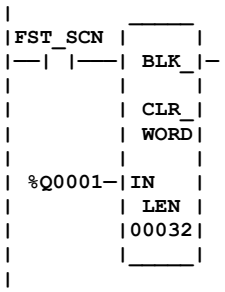
Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN		•	•	•	•	•†	•	•	•	•		
ok	•											•

• Platná adresa nebo místo, kde funkcí může téct proud. † Pouze %SA, %SB %SC; %S nelze použít.

Příklad

V uvedeném příkladu se při zapnutí napájení 32 slov paměti %Q (512 bodů) počínaje od %Q0001 vyplní nulami.



SHFR (BIT, WORD)

Funkce posunutí registru (SHFR) se používá k posunutí jednoho nebo více datových slov nebo datových bitů z adresového místa do zadané paměťové oblasti. Například jedno slovo se může posunout do paměťové oblasti se zadanou délkou pěti slov. Výsledkem tohoto posunutí bude, že jiné slovo dat se posune ven z konce paměťové oblasti.

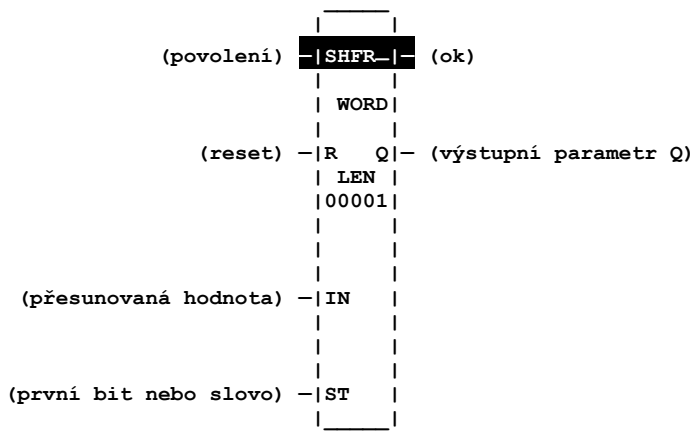
Poznámka

Při přiřazování adres mohou překrývající se rozsahy vstupních a výstupních adres u víceslovních funkcí vytvářet neočekávané výsledky.

Funkce SHFR má čtyři vstupní parametry a dva výstupní parametry. Resetovací vstup (R) bude mít přednost před vstupem pro povolení funkce. Když reset bude aktivní, všechny adresy začínající v registru posunutí (ST) až do délky zadané v LEN se vyplní nulami.

Pokud funkcí poteče proud a reset nebude aktivní, každý bit nebo slovo registru posunutí se přesune na další vyšší adresu. Poslední člen registru posunutí se posune do Q. Nejvyšší adresa členu registru posunutí v IN se posune do uvolněného členu začínajícího na ST. Obsah registru posunutí je přístupný přes program, protože v adresovatelné paměti logiky je nastavený na absolutní místa.

Funkce propustí proud doprava vždy, když bude procházet proud přes logiku pro povolení.



Parametry

Parametr	Popis
povolení	Když vstup povolení bude v jedničce a R ne, posunutí se provede.
R	Když R bude v jedničce, registr posunutí umístěný na ST se vyplní nulami.
IN	IN obsahuje hodnoty, které se mají posunout do prvního bitu nebo slova registru posunutí. V případě SHFR_BIT je možno použít libovolnou diskrétní adresu; nemusí být spojená s bajtem. Avšak 16 bitů počínaje zadanou adresou se zobrazí přímo.
ST	ST obsahuje první bit nebo slovo registru posunutí. V případě SHFR_BIT je možno použít libovolnou diskrétní adresu; nemusí být spojená s bajtem. Avšak 16 bitů počínaje zadanou adresou se zobrazí přímo.
ok	Výstup ok bude v jedničce vždy, když funkce bude povolená a R nebude povolené.
Q	Výstup Q obsahuje bit nebo slovo posunuté ven z registru posunutí. V případě SHFR_BIT je možno použít libovolnou diskrétní adresu; nemusí být spojená s bajtem. Avšak 16 bitů počínaje zadanou adresou se zobrazí přímo.
LEN	LEN určuje délku registru posunutí. V případě SHFR_WORD LEN musí být 1 až 256 slov. V případě SHFR_BIT, LEN musí být 1 až 256 bitů.

Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
R	•											
IN		•	•	•	•	•	•	•	•	•	•	
ST		•	•	•	•	•†	•	•	•	•		
ok	•											•
Q		•	•	•	•	•†	•	•	•	•		

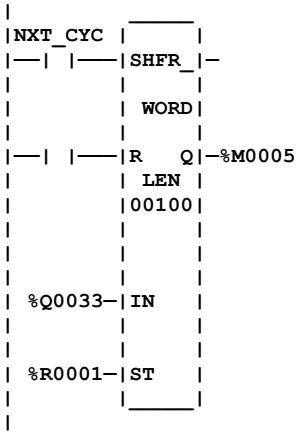
• Platná adresa pro data typu BIT, nebo WORD nebo místo, kudy proud může téct skrz funkci.
U funkce SHFR_BIT diskrétní uživatelské adresy %I, %Q, %M a %T nemusí být spojené s bajtem.

† Pouze %SA, %SB %SC; %S nelze použít.

Příklad 1

V následujícím příkladu registr posunutí pracuje s registrovými paměťovými místy %R0001 až %R0100. Když resetovací adresa CLEAR bude aktivní, slova registru posunutí se nastaví na nulu.

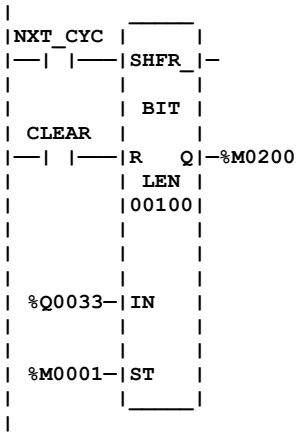
Když adresa NXT_CYC bude aktivní a CLEAR neaktivní, slovo z výstupní stavové tabulky na adrese %Q0033 se přesune do registru posunutí na adrese %R0001. Slovo posunuté ven z registru na adrese %R0100 se uloží do výstupu %M0005.



Příklad 2

V tomto příkladu registr posunutí pracuje s paměťovými místy %M0001 až %M0100. Když resetovací adresa CLEAR bude aktivní, funkce SHFR vyplní %M0001 až %M0100 nulami.

Když NXT_CYC bude aktivní a CLEAR ne, funkce SHFR posune data v %M0001 až %M0100 dolů o jeden bit. Bit v %Q0033 se posune do %M0001, přičemž bit posunutý ven z %M0100 se zapíše do %M0200.



BITSEQ (BIT)

Funkce bitového sekvenčního přepínače (BITSEQ) provádí posunutí bitové sekvence v poli bitů. Funkce BITSEQ má pět vstupních parametrů a jeden výstupní parametr. Činnost funkce závisí na předchozí hodnotě parametru EN, jak je znázorněno v následující tabulce.

Současné vykonání R	Předchozí vykonání EN	Současné vykonání EN	Vykonání bitového sekvenčního přepínače
OFF	OFF	OFF	Bitový sekvenční přepínač se nevykoná.
OFF	OFF	ON	Bitový sekvenční přepínač provádí inkrementaci/dekrementaci po 1.
OFF	ON	OFF	Bitový sekvenční přepínač se nevykoná.
OFF	ON	ON	Bitový sekvenční přepínač se nevykoná.
ON	ON/OFF	ON/OFF	Bitový sekvenční přepínač provede reset.

Resetovací vstup (R) přepíše povolení (EN) a vždy resetuje sekvenční přepínač. Když R bude aktivní, aktuální číslo kroku se nastaví na hodnotu předanou přes parametr čísla kroku. Pokud se nepředá žádné číslo kroku, krok se nastaví na 1. Všechny bity v sekvenčním přepínači se nastaví na 0 kromě bitu označeného aktuálním krokem, který se nastaví na 1.

Když EN bude aktivní a R bude neaktivní, bit označený aktuálním číslem kroku se smaže. Aktuální číslo kroku se buď inkrementuje nebo dekrementuje podle směru parametru. Pak se bit označený číslem nového kroku nastaví na 1.

- Když se číslo kroku bude inkrementovat a dostane se mimo rozsah (1 ≤ číslo kroku ≤ LEN), vrátí se zpět na 1.
- Když se číslo kroku bude dekrementovat a dostane se mimo rozsah (1 ≤ číslo kroku ≤ LEN), vrátí se zpět na LEN.

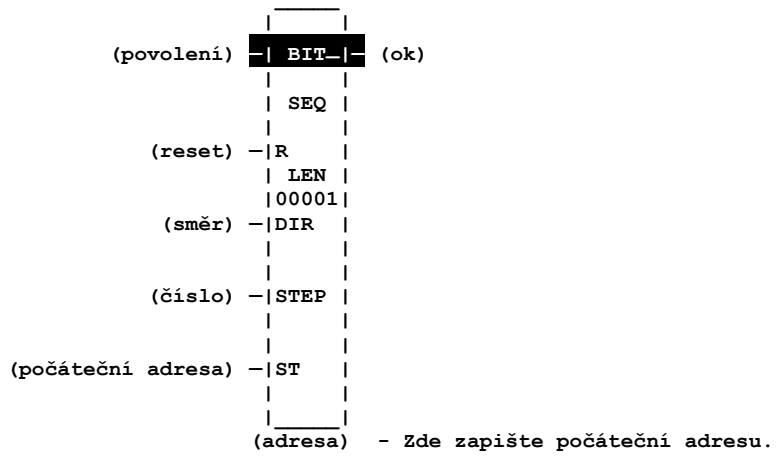
Parametr ST je volitelný. Pokud se nepoužije, BITSEQ bude pracovat výše popsaným způsobem s tou výjimkou, že se žádné bity nenastaví ani nevynulují. V zásadě BITSEQ pak provádí cyklování aktuálního čísla kroku uvnitř přípustného rozsahu.

Paměť požadovaná pro bitový sekvenční přepínač

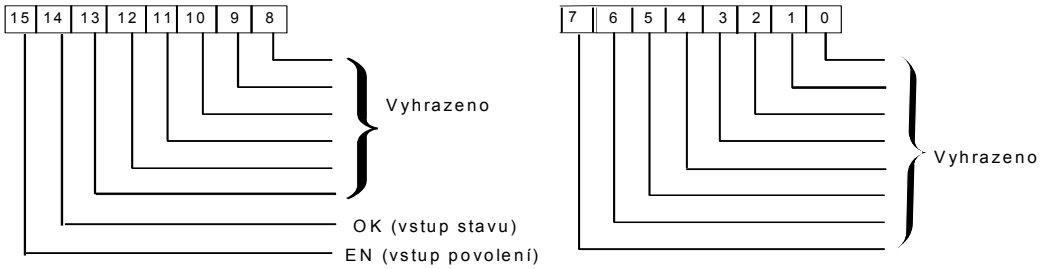
Každý bitový sekvenční přepínač používá k uložení následujících informací tři slova (registry) paměti %R:

aktuální číslo kroku	slovo 1
délka sekvence (v bitech)	slovo 2
řídicí slovo	slovo 3

Když budete zapisovat bitový sekvenční přepínač, musíte zapsat počáteční adresu pro tato tři slova (registry) přímo pod grafickou reprezentací funkce (viz příklad na následující stránce).



V řídicím slově je uložený stav Booleovských vstupů a výstupů souvisejícího funkčního bloku, jak je znázorněno na následujícím obrázku:



Poznámka

Bity 0 až 13 se nepoužívají. Všimněte si také, že bity musí být v parametru STEP zapsané jako 1 až 16, ***NE*** 0 až 15.

Parametry

Parametr	Popis
adresa	Adresa je umístění aktuálního kroku bitového sekvenčního přepínače, délky a posledního stavu povolení a ok.
povolení	Když funkce bude povolená, pokud nebyla povolená v předchozím cyklu, a pokud R nebude v jedničce, provede se posunutí bitové sekvence.
R	Když R bude v jedničce, číslo kroku bitového sekvenčního přepínače se nastaví na hodnotu v STEP (výchozí = 1) a bitový sekvenční přepínač se vyplní nulami s výjimkou bitu aktuálního čísla kroku.
DIR	Když DIR bude v jedničce, číslo kroku bitového sekvenčního přepínače se před posunutím inkrementuje. Jinak se bude dekrementovat.
STEP	Když R bude v jedničce, číslo kroku se nastaví do této hodnoty.
ST	ST obsahuje první slovo bitového sekvenčního přepínače.
ok	Výstup ok bude v jedničce vždy, když funkce bude povolená.
LEN	LEN musí být v rozmezí 1 až 256 bitů.

Poznámka

Kontrola cívky u funkce BITSEQ zkontroluje 16 bitů z parametru ST, i když LEN bude menší než 16.

Platné typy paměti

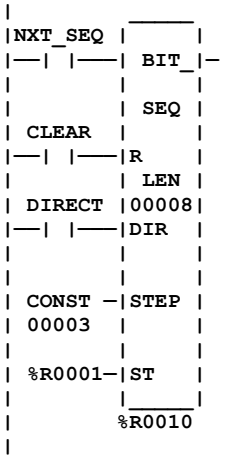
Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
adresa								•				
povolení	•											
R	•											
DIR	•											
STEP		•	•	•	•		•	•	•	•	•	•
ST		•	•	•	•	•†	•	•	•	•		•
ok	•											•

• Platná adresa nebo místo, kde funkcí může téct proud.
† Pouze %SA, %SB %SC; %S nelze použít.

Příklad

V následujícím příkladu sekvenční přepínač pracuje s pamětí registru %R0001. Jeho statická data jsou uložena v registrech %R0010, %R0011 a %R0012. Když CLEAR bude aktivní, sekvenční přepínač se resetuje a aktuální krok se nastaví na krok číslo 3. Prvních 8 bitů z %R0001 se nastaví na nulu.

Když NXT_SEQ bude aktivní a CLEAR bude neaktivní, bit pro krok číslo 3 se vynuluje a bit pro krok číslo 2 nebo 4 (v závislosti na tom, jestli DIR bude v jedničce) se nastaví.



COMMREQ

Funkci požadavku komunikace (COMMREQ) použijte, když program bude potřebovat komunikovat s inteligentním modulem, například komunikačním modulem Genius nebo programovatelným koprocesorovým modulem.

Poznámka

Informace na následujících stránkách uvádějí formát funkce COMMREQ. K naprogramování COMMREQ pro jednotlivé typy zařízení budete potřebovat další informace. Požadavky na programování pro jednotlivé moduly, které používají funkci COMMREQ, jsou popsané v dokumentaci k modulu.

Funkce COMMREQ má tři vstupní parametry a jeden výstupní parametr. Když funkcí COMMREQ bude procházet proud, do inteligentního modulu se pošle povelový blok dat. Povelový blok začíná na adrese zadané pomocí parametru IN. Sestava a číslo slotu inteligentního modulu jsou zadané v SYSID.

COMMREQ může buď poslat zprávu a čekat na odpověď nebo poslat zprávu a pokračovat bez čekání na odpověď. Pokud povelový blok bude určovat, že program nebude čekat na odpověď, obsah povelového bloku se pošle do přijímacího zařízení a vykonávání programu bude pokračovat okamžitě. (Hodnota časové prodlevy se ignoruje.) Tento režim se nazývá režim **NOWAIT**.

Pokud povelový blok bude určovat, že program bude čekat na odpověď, obsah povelového bloku se pošle do přijímacího zařízení a CPU bude čekat na odpověď. Maximální doba, po kterou PLC bude čekat, než zařízení odpoví, je zadaná v povelovém bloku. Pokud zařízení neodpoví během této doby, vykonávání programu bude pokračovat. Tento režim se nazývá režim **WAIT**.

Výstup Chyba funkce (FT) se může nastavit do stavu ON, když:

1. Zadaná cílová adresa nebude existovat (SYSID).
2. Zadaná úloha nebude platná pro zařízení (TASK).
3. Délka dat bude 0.
4. Adresa ukazatele stavu zařízení (část povelového bloku) neexistuje. To může být v důsledku nesprávné volby typu paměti nebo když adresa uvnitř typu paměti je mimo rozsah.

Povelový blok

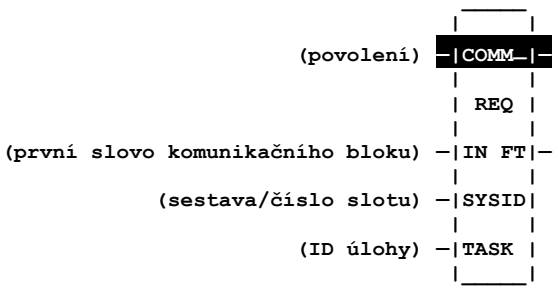
Povelový blok předává informace inteligentnímu modulu o povelu, který se má vykonat.

Adresa povelového bloku pro funkci COMMREQ je zadána ve vstupu IN. Tato adresa může být slovně orientovaná oblast paměti (%R, %AI nebo %AQ). Délka povelového bloku závisí na množství dat posílaných do zařízení.

Povelový blok má následující strukturu:

Délka (ve slovech)	adresa
Příznak Čekat/Nečekat	adresa + 1
Paměť ukazatele stavu	adresa + 2
Offset ukazatele stavu	adresa + 3
Hodnota prodlevy při nečinnosti	adresa + 4
Maximální doba komunikace	adresa + 5
Blok dat	adresa + 6 až adresa + 133

Informace požadované pro komunikační blok je možno umístit do určené oblasti paměti pomocí příslušné programovací funkce.



Parametry

Parametr	Popis
povolení	Když je funkce povolená, vykoná se komunikace.
IN	IN obsahuje první slovo komunikačního bloku.
SYSID	SYSID obsahuje číslo sestavy (bajt s nejvyšší váhou) a číslo slotu (bajt s nejnižší váhou) cílového zařízení.
TASK	TASK obsahuje ID úlohy procesu na cílovém zařízení.
FT	FT bude v jedničce, pokud se při zpracování COMMREQ zjistí chyba.

Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN								•	•	•		
SYSID		•	•	•	•		•	•	•	•	•	
TASK								•	•	•	•	
FT	•											•

• Platná adresa nebo místo, kde funkcí může téct proud.

Příklad

V následujícím příkladu, když vstup pro povolení %M0020 bude ve stavu ON, povelový blok, jehož umístění začíná na %R0016, se pošle do komunikační úlohy 1 v zařízení umístěném v sestavě 1, slot 2 daného PLC. Pokud při zpracování COMMREQ dojde k chybě, nastaví se %Q0100.

Poznámka

U systémů, které nemají přídavné sestavy, SYSID musí být pro hlavní sestavu nula.

Kapitola 10

Tabulkové funkce

Tabulkové funkce se používají k vykonávání následujících funkcí:

Zkratka	Funkce	Popis	Strana
ARRAY_MOVE	Přesunutí pole	Zkopíruje zadaný počet datových prvků ze zdrojového do cílového pole.	10-2
SRCH_EQ	Hledat shodné	Vyhledá všechny hodnoty v poli, které se rovnají zadané hodnotě.	10-6
SRCH_NE	Hledat neshodné	Vyhledá všechny hodnoty v poli, které se nerovnají zadané hodnotě.	10-6
SRCH_GT	Hledat větší než	Vyhledá všechny hodnoty v poli, které jsou větší než zadaná hodnota.	10-6
SRCH_GE	Hledat větší nebo shodné	Vyhledá všechny hodnoty v poli, které jsou větší nebo se rovnají zadané hodnotě.	10-6
SRCH_LT	Hledat menší než	Vyhledá všechny hodnoty v poli, které jsou menší než zadaná hodnota.	10-6
SRCH_LE	Hledat menší nebo shodné	Vyhledá všechny hodnoty v poli, které jsou menší nebo se rovnají zadané hodnotě.	10-6

Maximální délka přípustná pro tyto funkce je 32,767 bajtů nebo slov nebo 262,136 bitů (bity je možno použít pouze pro ARRAY_MOVE).

Tabulkové funkce pracují s následujícími typy dat:

Typ dat	Popis
INT	Celé číslo se znaménkem
DINT	Celé číslo se znaménkem s dvojnásobnou délkou
BIT *	Data typu bit
BYTE	Data typu bajt
WORD	Data typu slovo

* Možno použít pouze ve spojení s ARRAY_MOVE.

Výchozí typ dat je celé číslo se znaménkem. Typ dat je možno změnit po zvolení konkrétní datové tabulkové funkce. Chcete-li provést porovnání jiných typů nebo dvou různých typů, nejdříve použijte příslušnou funkci konverze (popsaná v kapitole 11, "Funkce konverze") a změňte data na některý výše uvedený typ.

ARRAY_MOVE (INT, DINT, BIT, BYTE, WORD)

Funkce přesunutí pole (ARRAY_MOVE) použijte ke zkopírování zadaného počtu datových prvků ze zdrojového pole do cílového pole.

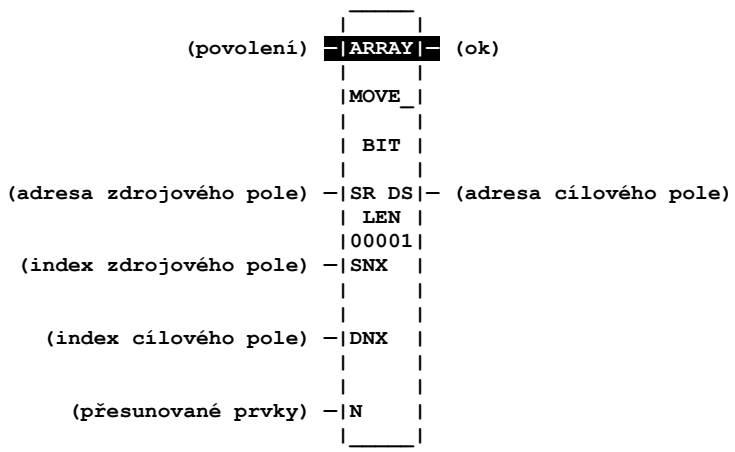
Funkce ARRAY_MOVE má pět vstupních parametrů a dva výstupní parametry. Když funkcí bude procházet proud, počet datových prvků v ukazateli počtu (N) se vyjme ze vstupního pole počínaje indexovaným paměťovým místem (SR + SNX - 1). Datové prvky se zapíšou do výstupního pole počínaje indexovaným paměťovým místem (DS + DNX - 1). Operand LEN udává počet prvků, které tvoří každé pole.

V případě ARRAY_MOVE_BIT, kdy jako parametry pro začáteční adresu zdrojového pole a/nebo cílového pole bude zvolena slovně orientovaná paměť, bit s nejnižší váhou zadaného slova bude první bit pole. Zobrazená hodnota bude obsahovat 16 bitů bez ohledu na délku pole.

Indexy v instrukci ARRAY_MOVE jsou na bázi jedniček. Při použití ARRAY_MOVE nelze adresovat žádný prvek mimo zdrojové nebo cílové pole (tak jak jsou zadané jejich začáteční adresou a délkou).

Pokud se nevyskytne některá z následujících podmínek, výstupem ok poteče proud:

- Vstup pro povolení bude ve stavu OFF.
- (N + SNX – 1) je větší než LEN.
- (N + DNX – 1) je větší než LEN.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, operace se provede.
SR	SR obsahuje začáteční adresu zdrojového pole. V případě ARRAY_MOVE_BIT je možno použít libovolnou adresu; nemusí být spojená s bajtem. Avšak 16 bitů počínaje zadanou adresou se zobrazí přímo.
SNX	SNX obsahuje index zdrojového pole.
DNX	DNX obsahuje index cílového pole.
N	N udává indikátor počtu.
ok	Výstup ok bude v jedničce, když vstupem pro povolení poteče proud.
DS	DS obsahuje začáteční adresu cílového pole. V případě ARRAY_MOVE_BIT je možno použít libovolnou adresu; nemusí být spojená s bajtem. Avšak 16 bitů počínaje zadanou adresou se zobrazí přímo.
LEN	LEN udává počet prvků začínajících na SR a DS, které tvoří každé pole.

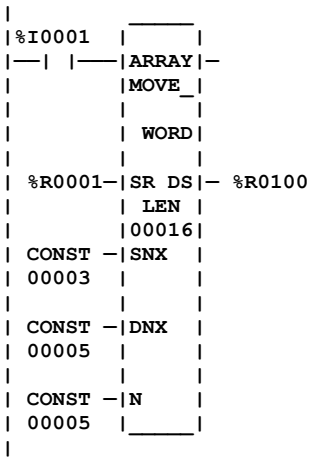
Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
SR		o	o	o	o	Δ†	o	•	•	•		
SNX		•	•	•	•		•	•	•	•	•	
DNX		•	•	•	•		•	•	•	•	•	
N		•	•	•	•		•	•	•	•	•	
ok	•											•
DS		o	o	o	o	†	o	•	•	•		

- • Platná adresa nebo místo, kde funkcí může téct proud.
  U funkce ARRAY_MOVE_BIT diskrétní uživatelské adresy %I, %Q, %M a %T nemusí být spojené s bajtem.
- o Platná adresa pouze pro data typu INT, BIT, BYTE nebo WORD; neplatí pro DINT.
- Δ Pouze platný typ dat BIT, BYTE nebo WORD; neplatí pro INT nebo DINT.
- † Pouze %SA, %SB %SC; %S nelze použít.

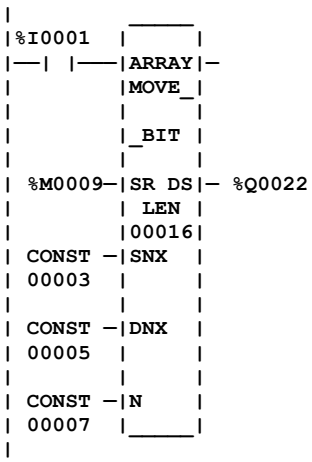
Příklad 1

V tomto příkladu se přečtou adresy %R0003 – %R0007 z pole %R0001 – %R0016 a pak se zapíšou do adres %R0104 – %R0108 z pole %R0100 – %R0115.



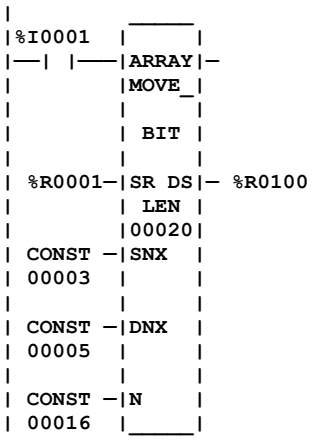
Příklad 2

Pomocí bitové paměti pro SR a DS se přečtou adresy %M0011– %M0017 z pole %M009 – %M0024 a zapíšou se do adres %Q0026 – %Q0032 z pole %Q0022 – %Q0037.



Příklad 3

Pomocí slovní paměti pro SR a DS se přečte třetí bit s nejnižší váhou adresy %R0001 až druhý bit s nejnižší váhou adresy %R0002 z pole obsahujícího všech 16 bitů z adresy %R0001 a čtyři bity z adresy %R0002 a pak se zapíšou do pátého bitu s nejnižší váhou z adresy %R0100 až čtvrtého bitu s nejnižší váhou z adresy %R0101 z pole obsahujícího všech 16 bitů z adresy %R0100 a čtyři bity z adresy %R0101.



Funkce vyhledávání

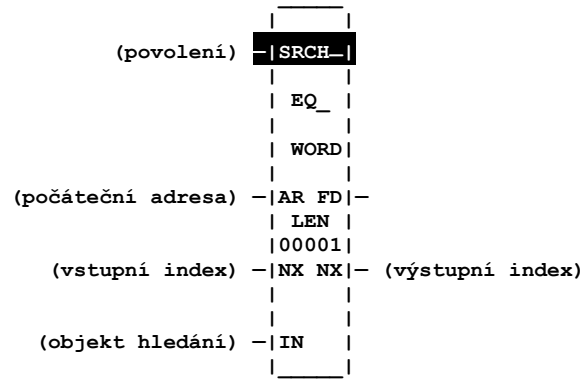
K prohledání všech hodnot pole pro tuto konkrétní operaci použijte příslušnou funkci vyhledávání uvedenou níže.

Zkratka	Funkce	Popis
SRCH_EQ	Hledat shodné	Vyhledá všechny hodnoty v poli, které se rovnají zadané hodnotě.
SRCH_NE	Hledat neshodné	Vyhledá všechny hodnoty v poli, které se nerovnají zadané hodnotě.
SRCH_GT	Hledat větší než	Vyhledá všechny hodnoty v poli, které jsou větší než zadaná hodnota.
SRCH_GE	Hledat větší nebo shodné	Vyhledá všechny hodnoty v poli, které jsou větší nebo se rovnají zadané hodnotě.
SRCH_LT	Hledat menší než	Vyhledá všechny hodnoty v poli, které jsou menší než zadaná hodnota.
SRCH_LE	Hledat menší nebo shodné	Vyhledá všechny hodnoty v poli, které jsou menší nebo se rovnají zadané hodnotě.

Každá funkce má čtyři vstupní parametry a dva výstupní parametry. Když funkcí bude protékat proud, pole se bude prohledávat počínaje od (AR + vstup NX). To je počáteční adresa pole (AR) plus index do tohoto pole (vstup NX).

Hledání bude pokračovat, dokud se nenajde prvek pole hledaného objektu (IN) nebo dokud se nedosáhne konce pole. Pokud se prvek pole najde, výstupní parametr (FD) se nastaví do stavu ON a výstupní parametr (výstup NX) se nastaví do relativní polohy tohoto prvku uvnitř pole. Pokud se nenajde žádný prvek před dosažením konce pole, výstupní parametr (FD) se nastaví do stavu OFF a výstupní parametr (výstup NX) se nastaví do nuly.

Platné hodnoty pro vstup NX jsou 0 až LEN - 1. Aby se mohlo začít hledání na prvním prvku, NX musí být nastaveno na nulu. Tato hodnota se při každém vykonání inkrementuje o jedničku. Proto výstup NX bude nabývat hodnot 1 až LEN. Pokud hodnota vstupu NX bude mimo rozsah (< 0 nebo ≥ LEN), jeho hodnota se nastaví na výchozí nulovou hodnotu.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, operace se provede.
AR	AR obsahuje počáteční adresu prohledávaného pole.
Vstup NX	Vstup NX obsahuje index pole, kde má začít hledání.
IN	IN obsahuje hledaný objekt.
Výstup NX	Výstup NX obsahuje polohu v poli hledaného cíle.
FD	FD udává, že byl nalezený prvek pole a funkce byla úspěšná.
LEN	LEN udává počet prvků začínajících na AR, které tvoří pole. Může být 1 až 32,767 bajtů nebo slov.

Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
AR		o	o	o	o	Δ	o	•	•	•		
NX in		•	•	•	•		•	•	•	•	•	
IN		o	o	o	o	Δ	o	•	•	•	•	
NX out		•	•	•	•		•	•	•	•		
FD	•											•

• Platná adresa nebo místo, kde funkcí může téct proud.
o Platná adresa pouze pro data typu INT, BYTE nebo WORD; neplatí pro DINT.
Δ Platná adresa pouze pro typ dat BYTE nebo WORD; neplatí pro INT nebo DINT.

Příklad 1

Pole AR je definováno jako paměťové adresy %R0001 – %R0005. Když EN bude ve stavu ON, v části pole mezi adresami %R0004 a %R0005 se bude hledat prvek, jehož hodnota se bude rovnat IN. Pokud %R0001 = 7, %R0002 = 9, %R0003 = 6, %R0004 = 7, %R0005 = 7 a %R0100 = 7, pak hledání začne na adrese %R0004 a skončí na adrese %R0004, přičemž FD bude nastaveno na ON a do %R0101 bude zapsaná 4.

Příklad 2

Pole AR je definováno jako paměťové adresy %AI0001 – %AI0016. Hodnoty prvků pole jsou 100, 20, 0, 5, 90, 200, 0, 79, 102, 80, 24, 34, 987, 8, 0 a 500. Na začátku bude v %AQ0001 hodnota 5. Když EN bude ve stavu ON, při každém cyklu se bude prohledávat pole a bude se hledat shoda hodnoty IN a nuly. První cyklus začne prohledávání na %AI0006 a najde shodu na %AI0007, takže FD se nastaví do stavu ON a v %AQ0001 bude 7. Druhý cyklus začne prohledávání na %AI0008 a najde shodu na %AI0015, takže FD zůstane ve stavu ON a v %AQ0001 bude 15. Další cyklus začne na %AI0016. Protože se dosáhne konce pole bez nalezení shody, FD se nastaví do stavu OFF a %AQ0001 se nastaví na nulu. Další cyklus začne prohledávání na začátku pole.

Kapitola 11

Převodní funkce

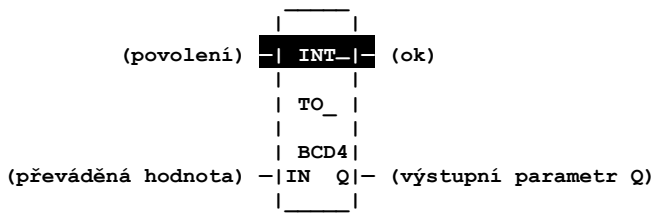
Převodní funkce se používají k převádění datových položek z jednoho číselného typu na jiný. Mnoho programovacích instrukcí, například matematické funkce, se musí používat s daty jednoho typu. Tato kapitola popisuje následující převodní funkce:

Zkratka	Funkce	Popis	Strana
BCD-4	Převod na BCD-4	Převede celé číslo se znaménkem na 4-místný formát BCD.	11-2
INT	Převod na celé číslo se znaménkem	Převede BCD-4 nebo REAL na celé číslo se znaménkem	11-3
DINT	Převod na celé číslo se znaménkem s dvojnásobnou délkou	Převede REAL na formát celého čísla se znaménkem s dvojnásobnou délkou	11-5
REAL	Převod na REAL	Převede INT, DINT, BCD-4 nebo WORD na REAL.	11-7
WORD	Převod na WORD	Převede REAL na formát WORD.	11-9
TRUN	Celá část	Zaokrouhlí reálné číslo směrem k nule.	11-11

—>BCD-4 (INT)

Funkce převodu na BCD-4 se používá k vytvoření 4-místného BCD ekvivalentu celého čísla se znaménkem. Tato funkce původní data nemění. Data je možno převést na formát BCD tak, aby bylo možno řídit LED kódované v kódu BCD nebo provést předvolbu externího zařízení, například vysokorychlostního čítače.

Když funkcí bude procházet proud, vykoná se převod a výsledek bude k dispozici na výstupu Q. Pokud výsledkem zadaného převodu nebude hodnota, která bude v rozsahu 0 až 9999, funkce propustí proud, když se na ní proud objeví.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, provede se převod.
IN	IN obsahuje adresu celočíselné hodnoty, která má být převedena na BCD-4.
ok	Výstup ok bude v jedničce, když se funkce vykoná bez chyby.
Q	Výstup Q obsahuje BCD-4 tvar původní hodnoty v IN.

Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN		•	•	•	•		•	•	•	•	•	
ok	•											•
Q		•	•	•	•		•	•	•	•		

- Platná adresa nebo místo, kde funkcí může téct proud.

Příklad

V následujícím příkladu, když bude nastavený vstup %I0002 a nevyskytnou se žádné chyby, celé číslo na adrese %I0017 až %I0032 se převede na čtyři BCD číslice a výsledek se uloží na paměťovém místě %Q0033 až %Q0048. K ověření úspěšného převodu se používá cívka %Q1432.

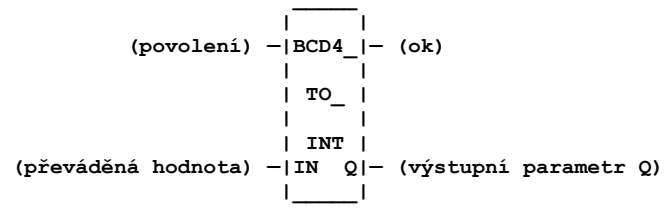
—>INT (BCD-4, REAL)

Funkce převodu na celé číslo se znaménkem se používá k vytvoření celočíselného ekvivalentu dat BCD-4 nebo REAL. Tato funkce původní data nemění.

Poznámka

Data typu REAL je možno použít pouze u CPU řady 35x a 36x, verze 9 nebo pozdější nebo u všech verzí CPU352.

Když funkcí bude procházet proud, vykoná se převod a výsledek bude k dispozici na výstupu Q. Pokud data nebudou mimo rozsah, funkce bude vždy přenášet proud, pokud se na ní proud přivede.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, provede se převod.
IN	IN obsahuje adresu hodnoty BCD-4, REAL nebo konstanty, která se má převést na celé číslo.
ok	Pokud data nebudou mimo rozsah nebo typu Nan (nenumerické), výstup ok bude v jedničce vždy, když vstup povolení bude v jedničce.
Q	Výstup Q obsahuje celočíselný tvar původní hodnoty v IN.

Platné typy paměti

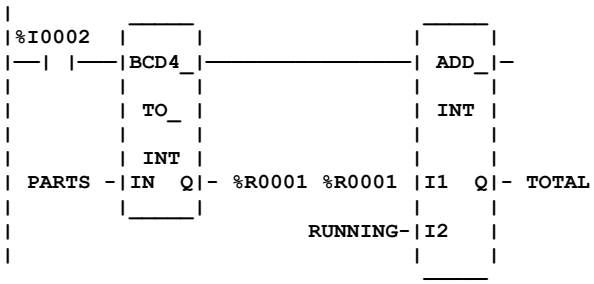
Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN		•	•	•	•		•	•	•	•	•	
ok	•											•
Q		•	•	•	•		•	•	•	•		

Poznámka: U dat typu REAL jsou jediné platné typy %R, %AI a %AQ.

• Platná adresa nebo místo, kde funkcí může téct proud.

Příklad

V následujícím příkladu při každém nastavení %I0002 se BCD-4 hodnota v PARTS převede na celé číslo se znaménkem a předá se do funkce ADD, kde se přičte k celočíselné hodnotě se znaménkem představované adresou RUNNING. Součet vytvořený funkcí ADD se zapíše na adresu TOTAL.



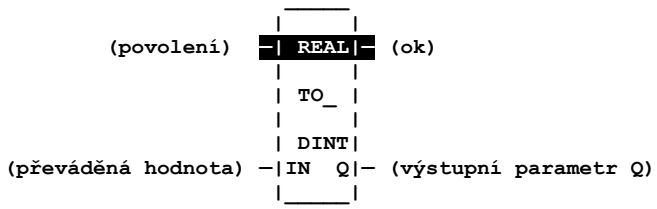
—>DINT (REAL)

Funkce převodu na celé číslo s dvojnásobnou délkou se znaménkem se používá k vytvoření celočíselného ekvivalentu dat celého čísla se znaménkem s dvojnásobnou délkou nebo reálných dat. Tato funkce původní data nemění.

Poznámka

Data typu REAL je možno použít pouze u CPU řady 35x a 36x, verze 9 nebo pozdější nebo u všech verzí CPU352.

Když funkcí bude procházet proud, vykoná se převod a výsledek bude k dispozici na výstupu Q. Pokud reálná hodnota není mimo rozsah, funkce bude vždy přenášet proud, pokud se na ní proud přivede.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, provede se převod.
IN	IN obsahuje adresu hodnoty BCD-4, REAL nebo konstanty, která se má převést na celé číslo s dvojnásobnou délkou.
ok	Pokud reálná hodnota mimo rozsah, výstup ok bude v jedničce vždy, když vstup povolení bude v jedničce.
Q	Q obsahuje původní hodnotu v IN ve tvaru celého čísla se znaménkem s dvojnásobnou délkou.

Poznámka

Když se bude provádět převod z REAL na DINT, může dojít ke ztrátě přesnosti, protože REAL má 24 významných bitů.

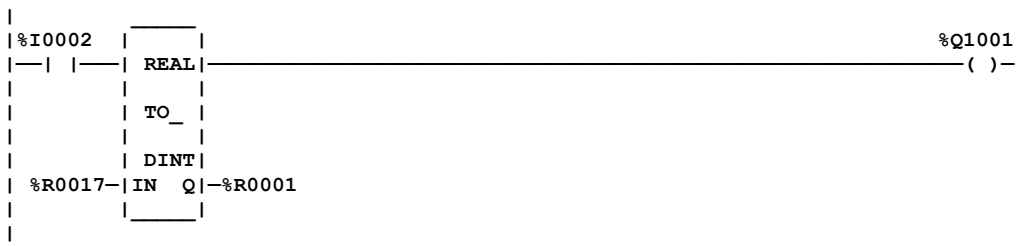
Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN		o	o	o	o		o	•	•	•	•	
ok	•											•
Q								•	•	•		

- Platná adresa nebo místo, kde funkcí může téct proud.

Příklad

V následujícím příkladu, když bude nastavený %I0002, reálná hodnota na vstupní adrese %R0017 se převede na celé číslo se znaménkem s dvojnásobnou délkou a výsledek se zapíše na adresu %R0001. Výstup %Q1001 se nastaví vždy, když se funkce vykoná úspěšně.



—>REAL (INT, DINT, BCD-4, WORD)

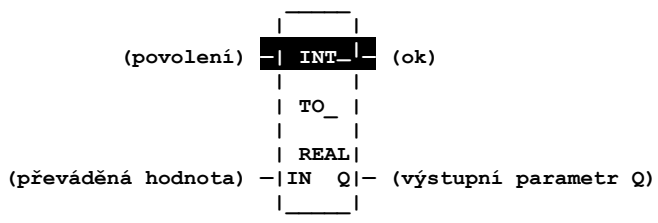
Funkce převodu na reálné číslo se používá k vytvoření reálné hodnoty vstupních dat. Tato funkce původní data nemění.

Když funkcí bude procházet proud, vykoná se převod a výsledek bude k dispozici na výstupu Q. Pokud výsledkem zadaného převodu nebude hodnota, která bude mimo rozsah, funkce propustí proud, když se na ní proud přivede.

Když se bude provádět převod z DINT na REAL, může dojít ke ztrátě přesnosti, protože počet významných bitů se sníží na 24.

Poznámka

Tuto funkci je možno použít pouze u CPU řady 35x a 36x, verze 9 nebo pozdější nebo u všech verzí CPU352.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, provede se převod.
IN	IN obsahuje adresu celočíselné hodnoty, která má být převedena na REAL.
ok	Výstup ok bude v jedničce, když se funkce vykoná bez chyby.
Q	Q obsahuje REAL tvar původní hodnoty v IN.

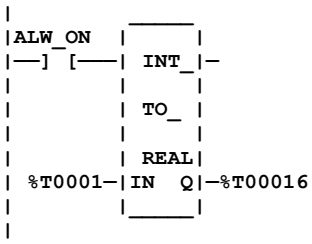
Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN		o	o	o	o		o	•	•	•	•	
ok	•											•
Q								•	•	•		

• Platná adresa nebo místo, kde funkcí může téct proud.
o Neplatí pro DINT_TO_REAL.

Příklad

V následujícím příkladu celočíselná hodnota na vstupu IN je 678. Výsledná hodnota zapsaná do %T0016 bude 678.000.



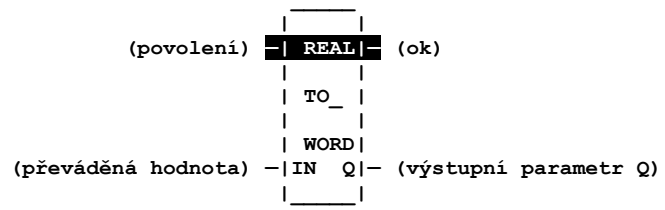
—>WORD (REAL)

Funkce převodu na WORD se používá k vytvoření WORD ekvivalentu reálných dat. Tato funkce původní data nemění.

Poznámka

Tuto funkci je možno použít pouze u CPU řady 35x a 36x.

Když funkcí bude procházet proud, vykoná se převod a výsledek bude k dispozici na výstupu Q. Pokud výsledkem zadaného převodu bude hodnota v rozsahu 0 až FFFFh, funkce propustí proud, když se na ní proud přivede.



Parametry

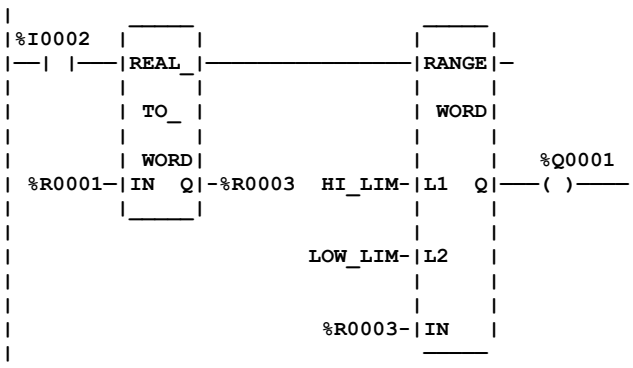
Parametr	Popis
povolení	Když funkce bude povolená, provede se převod.
IN	IN obsahuje adresu hodnoty, která se má převést na WORD.
ok	Výstup ok bude v jedničce, když se funkce vykoná bez chyby.
Q	Q obsahuje původní hodnotu v IN ve tvaru celého čísla bez znaménka.

Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN								•	•	•	•	
ok	•											•
Q		•	•	•	•		•	•	•	•		

• Platná adresa nebo místo, kde funkcí může téct proud.

Příklad


%I0002
REAL_
TO_
WORD
IN
Q
%R0001
%R0003
HI_LIM
LOW_LIM
L1
L2
IN
RANGE
WORD
Q
%Q0001

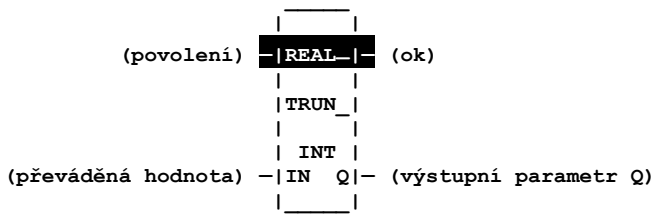
TRUN (INT, DINT)

Funkce Celá část se používá k zaokrouhlení reálného čísla směrem k nule. Tato funkce původní data nemění.

Poznámka

CPU řady 35x a 36x, (verze 9 nebo pozdější a všechny verze CPU352) jsou jediné CPU Series 90-30 s funkcí plovoucí desetinné tečky. Proto funkci TRUN nemá smysl používat u jiných CPU 90-30.

Když funkcí bude procházet proud, vykoná se převod a výsledek bude k dispozici na výstupu Q. Pokud výsledkem zadaného převodu nebude hodnota mimo rozsah nebo IN nebude typu NaN (nenumerické), CPU 352 funkce propustí proud při příchodu proudu. U všech ostatních CPU řady 35x a 36x funkce proud *nepropustí*.



Parametry

Parametr	Popis
Povolení	Když funkce bude povolená, provede se převod.
IN	IN obsahuje adresu reálné hodnoty, která se má zaokrouhlit.
Ok	Výstup ok bude v jedničce, když se funkce vykoná bez chyby za předpokladu, že hodnota nebude mimo rozsah nebo IN nebude NaN.
Q	Q obsahuje celou část původní hodnoty IN ve tvaru INT nebo DINT.

Poznámka

Když se bude provádět převod z REAL na DINT, může dojít ke ztrátě přesnosti, protože REAL má 24 významných bitů.

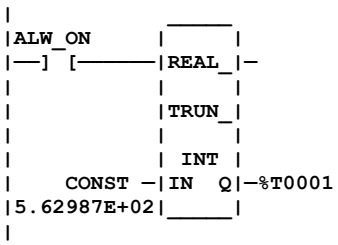
Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN								•	•	•	•	
ok	•											•
Q		o	o	o	o		o	•	•	•		

• Platná adresa nebo místo, kde funkcí může téct proud.
o Platí pouze pro REAL_TRUN_INT.

Příklad

V následujícím příkladu se zobrazená konstanta zaokrouhlí a celočíselný výsledek 562 se zapíše do %T0001.



Kapitola 12

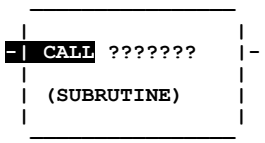
Řídicí funkce

Tato kapitola popisuje řídicí funkce, které je možno použít k omezení vykonávání programu a ke změnění způsobu, jakým CPU vykonává aplikační program. (Viz kapitola 2, část 1, “Přehled cyklů PLC”, kde jsou uvedené informace o cyklu CPU.

Funkce	Popis	Strana
CALL	Vykonávání programu přejde na zadaný blok podprogramu.	12-2
DOIO	Po dobu jednoho cyklu okamžitě obslouží zadaný rozsah vstupů nebo výstupů. (Všechny vstupy nebo výstupy na modulu se obslouží, pokud adresová místa na tomto modulu budou součástí funkce DO I/O. Částečné aktualizace I/O modulu se neprovádějí.) Nebo do interní paměti se místo reálných vstupních bodů uloží načtené I/O.	12-3
SER	Sekvenční záznamník událostí - shromažďuje řadu vzorků. Řídicí blok funkce obsahuje uživatelem dodanou konfiguraci vykonávání funkčního bloku, konfiguraci vzorků a parametry operace.	12-8
END	Vykoná přechodný konec logiky. Program se zpracuje od první příčky k poslední příčce nebo k instrukci END (podle toho, co je zjištěno dříve). Tato instrukce je vhodná pro účely ladění, ale není přípustná v programování SFC (viz poznámka na straně 12-8).	12-21
MCR a MCRN	Naprogramuje funkci Hlavní řídicí relé (Master Control Relay). MCR způsobí, že všechny příčky mezi MCR a následným ENDMCR se zpracují bez proudu. Program Logicmaster 90-30/20/Micro podporuje dva způsoby funkce MCR, vnořovaný způsob (MCRN) a nevnořovaný způsob (MCR).	12-22
ENDMCR a ENDMCRN	Udává, že následující logika se má vykonat s normálním proudem. Program Logicmaster 90-30/20/Micro podporuje dva způsoby funkce ENDMCR, vnořovaný způsob (ENDMCRN) a nevnořovaný způsob (ENDMCR).	12-25
JUMP a JUMPN	Vykonávání logiky odskočí na zadané místo (udané návěštím LABEL, viz níže) uvnitř logiky. Program Logicmaster 90-30/20/Micro podporuje dva způsoby funkce JUMP, nevnořovaný způsob (JUMP) a vnořovaný způsob (JUMPN).	12-26
LABEL a LABELN	Udává cílové místo instrukce JUMP. Program Logicmaster 90-30/20/Micro podporuje dva způsoby funkce LABEL, nevnořovaný způsob (LABEL) a vnořovaný způsob (LABELN).	12-28
COMMENT	Umístí do programu komentář (vysvětlení příček). Po naprogramování instrukce je možno text zapsat "nakouknutím" do instrukce.	12-29
SVCREQ	Požaduje speciální službu PLC. (Viz seznam služeb na straně 12-30.)	12-30
PID	Vytváří dva PID (proporcionálně/integračně/derivační) řídicí algoritmy s uzavřenou smyčkou. • Standardní ISA PID algoritmus (PIDISA). • Algoritmus s nezávislým členem (PIDIND).	12-71

CALL

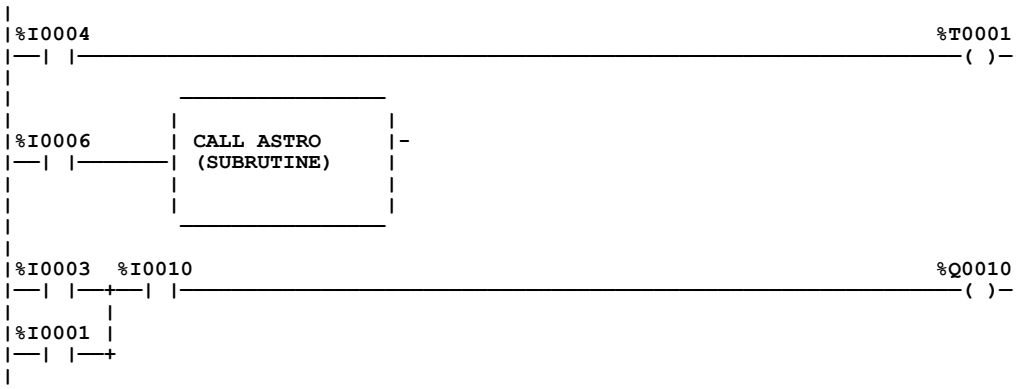
Použití funkce CALL provede během vykonávání programu odskok do určitého podprogramu.



Když funkcí CALL bude protékat proud, čtení okamžitě přejde do zadaného bloku podprogramu a vykoná ho. Když se dokončí vykonání bloku podprogramu, řízení se vrátí do bodu v logice hned za instrukci CALL.

Příklad

Následující příklad ukazuje podprogram instrukce CALL tak, jak se objeví v bloku volání. Po nastavení kurzoru na instrukci můžete stisknutím **F10** nahlédnout do podprogramu.



Poznámka

PLC Series 90 Micro nepodporují podprogramy; proto funkce CALL nejsou pro PLC Series 90 Micro vhodné.

DOIO

Funkce DO I/O (DOIO) se používá k aktualizaci vstupů nebo výstupů na jedno čtení v průběhu vykonávání programu. Funkci DOIO je možno kromě normálního čtení I/O také použít k aktualizaci zvolených I/O během vykonávání programu.

Pokud budou zadané vstupní adresy, funkce umožní zjistit pro programovou logiku nejnovější hodnoty vstupů. Pokud vstupní adresy budou zadané, DO I/O provede aktualizaci výstupů podle nejnovějších hodnot vstupů uložených v I/O paměti. I/O se obsluhuje v inkrementech celých I/O modulů; PLC v případě potřeby pozmění adresy během vykonávání funkce.

Funkce DOIO má čtyři vstupní parametry a jeden výstupní parametr. Když funkcí bude protékat proud a vstupní adresy budou zadané, provede se čtení vstupních bodů od počáteční adresy (ST) až po konečnou adresu END. Pokud bude zadaná adresa v ALT, do paměti se uloží kopie vstupních hodnot počínaje od této adresy a aktualizace skutečných vstupních bodů se neprovede. ALT musí mít stejnou velikost jako typ načítané adresy. Pokud se pro ST a END použije diskrétní adresa, pak ALT musí být také diskrétní. Pokud v ALT nebude zadána žádná adresa, aktualizace skutečných vstupních bodů se provede.

Když funkcí DOIO bude protékat proud a budou zadané výstupní adresy, výstupní body od počáteční adresy (ST) až po konečnou adresu (END) se zapíšou do výstupních modulů. Pokud se výstupy mají zapsat do výstupních modulů z jiné interní paměti než %Q nebo %AQ, v ALT je možno zadat počáteční adresu. Rozsah výstupů zapisovaných do výstupních modulů je zadaný počáteční adresou (ST) a koncovou adresou (END).

Vykonávání funkce bude pokračovat, dokud nebudou načtené všechny vstupy zvoleného rozsahu nebo nebudou obsloužené všechny výstupy na kartě I/O. Vykonávání programu se pak vrátí na další funkce, která následuje za DO I/O.

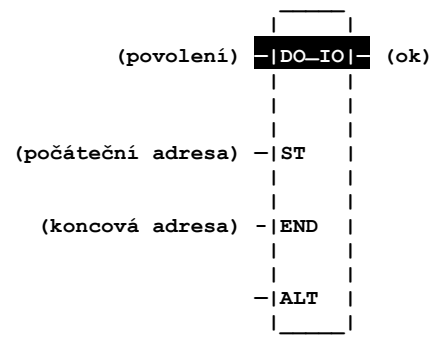
Pokud rozsah adres bude zahrnovat přídavný modul (HSC, APM, atd.), pak se načtou všechna vstupní data (%I a %AI) nebo všechna výstupní data (%Q a %AQ) pro tento modul. Parametr ALT se během čtení přídavných modulů ignoruje. Rozsah adres nesmí zahrnovat rozšířený GCM modul (viz poznámka níže).

Poznámka

U CPU verze 9.0 a pozdější funkci DOIO ***je*** možno použít s rozšířeným GCM modulem.

Funkce propustí proud doprava vždy, když přijde proud přes logiku pro povolení, pokud:

- Ve zvoleném rozsahu budou všechny adresy zadaného typu.
- CPU bude schopna řádně zpracovávat přechodný seznam I/O vytvořený funkcí.
- Zadaný rozsah bude obsahovat I/O, které souvisejí s chybou “Ztráta I/O”.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, provede se omezené čtení vstupů nebo zápis výstupů.
ST	ST je počáteční adresa nebo soubor vstupních nebo výstupních bodů nebo slov, které se mají obsloužit.
END	END je koncová adresa nebo soubor vstupních nebo výstupních bodů nebo slov, které se mají obsloužit.
ALT	U čtení vstupů ALT udává adresu pro uložení načtených hodnot vstupních bodů/slov. U zápisu výstupů ALT udává adresu, ze které se mají získat hodnoty výstupního bodu/slova pro odeslání do I/O modulu. U CPU model 331 a pozdější parametr ALT může mít vliv na rychlost vykonávání funkčního bloku DOIO (viz poznámka níže a kapitola o rozšířené funkci DO I/O pro CPU 331 a pozdější na straně 12-4).
ok	Výstup ok bude v jedničce, když se čtení vstupu nebo zápis na výstup dokončí normálně.

Poznámka

U CPU model 331 a pozdější se parametr ALT funkčního bloku DOIO může použít k zápisu pozice jednoho modulu v hlavní sestavě. Když se toto provede, funkční blok DOIO se vykoná během 80 mikrosekund místo 236 mikrosekund, potřebných když se funkční blok vykoná bez parametru ALT. K zamezení překrývání adres nebo nesouladu typů modulů se neprovádí žádná kontrola chyb.

Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
ST		•	•						•	•		
END		•	•						•	•		
ALT		•	•	•	•		•	•	•	•		•
ok	•											•

• Platná adresa nebo místo, kde funkcí může téct proud.

Příklad vstupu 1

V následujícím příkladu, když vstup pro povolení %I0001 bude ve stavu ON, načtou se adresy %I0001 až %I0064 a %Q0001 se sepne. Kopie načtených vstupů se uloží do interní paměti od adresy %M0001 do %M0064. Aktualizace skutečných vstupních bodů se neprovede. Tento tvar funkce je možno použít k porovnání aktuálních hodnot vstupních bodů s hodnotami vstupních bodů na začátku čtení.

Příklad vstupu 2

V následujícím příkladu, když vstup pro povolení %I0001 bude ve stavu ON, načtou se adresy %I0001 až %I0064 a %Q0001 se sepne. Načtené vstupy se uloží do paměti stavu vstupů od adresy %I0001 do %I0064. Tento tvar funkce umožňuje načíst vstupní body jednou nebo vícekrát během výkonné části programu cyklu CPU.

Příklad výstupu 1

V následujícím příkladu, když vstup pro povolení %I0001 bude ve stavu ON, hodnoty analogových výstupních kanálů %AQ001 až %AQ004 se zapíšou do adres %R0001 až %R0004 a %Q0001 se sepne. Hodnoty na adresách %AQ001 až %AQ004 se do analogových výstupních modulů nezapíšou.

Příklad výstupu 2

V následujícím příkladu, když vstup pro povolení %I0001 bude ve stavu ON, hodnoty na adresách %AQ001 až %AQ004 se zapíšou do analogových výstupních kanálů %AQ001 až %AQ004 a %Q0001 se sepne.

Rozšířená funkce DO I/O pro CPU 331 a pozdější

Upozornění

Pokud se rozšířená funkce DO I/O použije v programu, program by se neměl načíst verzí software Logicmaster 90-30/20 software starší než 4.01.

U verze 4.20 nebo pozdější CPU model 331 a pozdější je možno použít rozšířenou verzi funkce DO I/O (DOIO) . Tuto rozšířenou verzi funkce DOIO je možno použít pouze na jednom modulu s diskrétními vstupy nebo diskrétními výstupy s 8, 16 nebo 32 body.

Parametr ALT udává pozici v hlavní sestavě, ve které je modul umístěný. Například konstanta 2 v tomto parametru bude informovat CPU, že má vykonat rozšířenou verzi funkčního bloku DOIO pro modul v pozici 2.

Poznámka

Jediná kontrola, kterou provádí blok rozšířené funkce DOIO, je kontrola stavu modulu v zadané pozici, jestli tento modul je v pořádku.

Rozšířená funkce DOIO platí pouze pro moduly umístěné v hlavní sestavě. Proto parametr ALT musí mít pro sestavu s 5 pozicemi hodnotu mezi 2 a 5 nebo mezi 2 a 10 pro sestavu s 10 pozicemi.

Počáteční a koncová adresa musí být buď %I nebo %Q. Tyto adresy udávají první a poslední adresu, na kterou je modul nakonfigurovaný. Pokud například vstupní modul se 16 body bude nakonfigurovaný na adresu %I0001 až %I0016 v pozici 10 hlavní sestavy s 10 pozicemi, parametr ST musí být %I0001, parametr END musí být %I0016 a parametr ALT musí být 10, jak je znázorněno na následujícím obrázku:

Následující tabulka porovnává doby vykonávání normálního funkčního bloku DOIO pro vstupní/výstupní modul s 8, 16 nebo 32 body s dobou vykonávání rozšířeného funkčního bloku DOIO.

Modul	Doba vykonávání normálního DOIO	Doba vykonávání rozšířeného DOIO
Diskrétní vstupní modul s 8 body Diskrétní výstupní modul s 8 body	224 mikrosekund 208 mikrosekund	67 mikrosekund 48 mikrosekund
Diskrétní vstupní modul s 16 body Diskrétní výstupní modul s 16 body	224 mikrosekund 211 mikrosekund	68 mikrosekund 47 mikrosekund
Diskrétní vstupní modul s 32 body Diskrétní výstupní modul s 32 body	247 mikrosekund 226 mikrosekund	91 mikrosekund 50 mikrosekund

SER

Vlastnosti

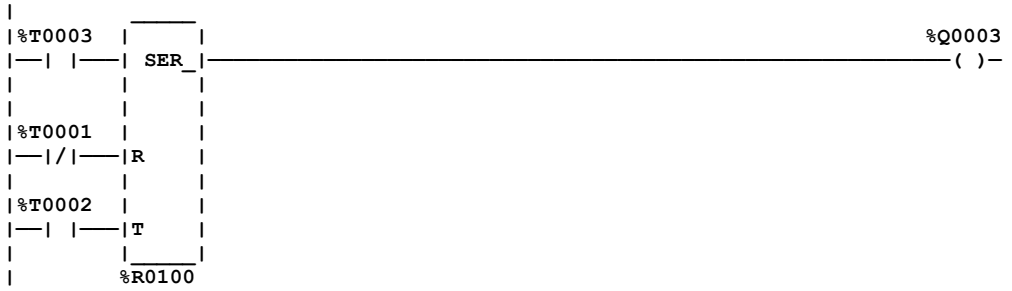
- Funkční blok SER (sekvenční záznamník událostí) shromažďuje soubor vzorků. Když vstupem Povolení bude protékat proud, funkční blok SER načte až 32 souvislých nebo nesouvislých bitů na vzorek.
- Každý SER může zachytit až 1024 vzorků.
- Pokud funkční blok SER bude vložený do periodického podprogramu, četnost vzorkování bude určena četností vykonávání periodického podprogramu.
- Časovou značkou může být označen pouze vzorek s aktivačním signálem. Vzorek s aktivačním signálem může být označený časovým údajem ve formátu BCD (maximální rozlišitelnost je 1 s) nebo ve formátu POSIX (maximální rozlišitelnost je 10 ms). Časová značka se umístí pouze ve spouštěcím bodě. SER nepodporuje více než jeden časový údaj na záznam.
- SER je možno nakonfigurovat pro režimy před aktivací, během aktivace a po aktivaci. (Viz stránka 12-14.)
- Operace SER se nakonfiguruje řídicím blokem funkce, který můžete vytvořit pomocí povelů Přesunutí bloku (BLKMV). (Viz stránka 12-10)

Poznámka

Synchronizace mezi PLC se nepodporuje.

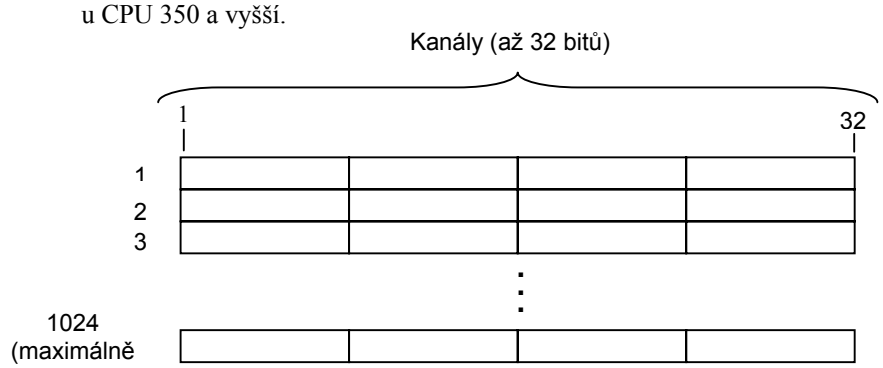
Funkční blok má jeden vstup a tři výstupy: povolení, reset a spouštění.

Příklad funkčního bloku SER



Poznámka

Tato funkce vyžaduje firmware CPU verze 9.00 nebo vyšší a je k dispozici ***pouze*** u CPU 350 a vyšší.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená a vstup reset bude na nule, funkční blok SER načte jeden vzorek ze všech nakonfigurovaných kanálů.
R	Když vstupem reset bude protékat proud, funkce SER se resetuje bez ohledu na stav vstupu pro povolení. Údaje Zásobník vzorků, Offset vzorku s aktivačním signálem, Aktivační čas a Offset aktuálního vzorku se vynulují. Funkční blok zůstane v resetovaném stavu, dokud vstupem pro reset nepřestane protékat proud. Ve stavu reset výstup OK přejde do nuly. Když vstupem pro reset přestane protékat proud, vzorkování se obnoví.
T	Pokud bude zvolený režim Aktivační vstup, funkční blok bude povolený a aktivační vstup přejde do jedničky, SER přejde do aktivovaného stavu. Provede se záznam Aktivačního času, Offsetu vzorku s aktivačním signálem a vzorku. Vzorek s aktivačním signálem se zaznamená bez ohledu na počet sejmutých vzorků. Po aktivaci záznamník dějů bude pokračovat ve vzorkování, dokud se neprovede Počet vzorků po aktivaci, kdy se načítání vzorků zastaví, dokud vstupem pro reset nepoteče proud. Pokud režim aktivace bude nastavený na Plný zásobník, aktivační signál se bude ignorovat. Informace o konfiguraci režimu aktivace najdete v odstavci “Řídicí blok funkce” na straně 12-10.
Počáteční adresa	Na této adrese začíná pole řídicího bloku funkce s délkou 78 slov. Řídicí blok funkce definuje vykonávání funkčního bloku, konfiguraci vzorků a parametry operace. Podrobnosti viz “Řídicí blok funkce” na straně 12-10.
ok	Výstup ok bude v jedničce, když budou splněné podmínky aktivace (zadané parametrem Aktivační režim) a vzorkování bude dokončeno. Výstupem bude procházet proud bez ohledu na stav vstupu pro povolení, dokud vstupem pro reset bude procházet proud.

Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
Řídicí blok								•				
R	•											
T	•											
ok	•											•

• Platná adresa nebo místo, kde funkcí může téct proud.

Řídicí blok funkce

Řídicí blok funkce je pole se 78 slovy, které definuje informace o zachycení dat a mechanismus aktivace pro funkci SER. V konkrétním programu je možno s jedním funkčním povelovým blokem a blokem dat spojit pouze jeden funkční blok sekvenčního záznamníku událostí.

Chcete-li nakonfigurovat parametry pro funkční blok SER, proveďte následující kroky:

1. Nastavte uložené hodnoty polí, jak je definováno v následující tabulce. K inicializaci registrů můžete použít přesun bloků nebo inicializovat data v tabulce a tabulku uložit před aktivací funkce SER.
2. Do žebříkové logiky přidejte funkční blok SER.

Poznámka

Pokud budete vyžadovat x kanálů, kde x se nerovná 8, 16, 24, ale je menší než 32, musíte zvolit počet kanálů, který je větší než x a násobkem 8, a u nepoužitých kanálů vyplnit jejich neexistující popis. Neexistující popis kanálu má volič segmentu 0xFFh, parametr délky rovnající se počtu nepoužitých kanálů a offset 0.

Slovo	Parametr	Popis
0 (počáteční adresa)	Stav	Čte se pouze proměnná, která udává aktuální stav funkčního bloku SER. Další informace jsou v Přídavných stavových datech (Slovo 1). **Poznámka:** Pokud se v řídicím bloku zjistí chyba, stav se nastaví na 6, výstup OK se vynuluje a nevykoná se žádná akce. Nastavení stavu může být: 0 = Reset 1 = Neaktivní 2 = Aktivní 3 = Aktivovaný (signálem) 4 = Dokončený 5 = Chyba přejetí 6 = Chyba parametru
1	Přídavná stavová data	Proměnná určená pouze pro čtení, která obsahuje přídavné stavové informace o funkci SER. Nastavení tohoto parametru viz “Přídavná stavová data” na straně 12-12.
2	Režim aktivace	Definuje podmínky funkčního bloku SER pro přechod do stavu aktivace. Platná nastavení jsou: 0 = Režim Aktivačního vstupu 1 = Režim Plného zásobníku Pokud v režimu Aktivačního vstupu bude funkční blok povolený, časová značka se vygeneruje, když bude aktivovaný signál Aktivace. Vzorkování bude pokračovat, dokud nebude splněný Počet vzorků po aktivaci. Když k tomu dojde, výstup OK přejde do jedničky. V režimu Plného zásobníku se signál Aktivace ignoruje. Když bude funkční blok povolený, vzorkování bude pokračovat, dokud nebude zásobník plný. Když k tomu dojde, výstup OK přejde do jedničky. Parametr Počet vzorků určuje velikost zásobníku.
3	Formát Aktivačního času	Určuje, jak se Aktivační čas bude zobrazovat. V případě zobrazování ve formátu BCD nastavte tento parametr na 0. V případě zobrazování ve formátu POSIX nastavte tento parametr na 1. (Podrobnosti viz strana 12-20.)
4—7	Vyhrazeno	Slova 4 až 7 jsou vyhrazena a musí být nastavena na nulu.

Slovo	Parametr	Popis
8	Počet kanálů (bitů na vzorek)	Udává počet bitů dat, které se vzorkují a ukládají se do zásobníku vzorků při každém vykonání funkčního bloku. Platné hodnoty jsou 8, 16, 24 nebo 32 bitů. Inkrement je velikost bajtu (8 bitů) a všechny nepoužité kanály musí být nakonfigurované s neexistujícím popisem kanálu. (Viz slova 14 až 77.)
9	Počet vzorků	Udává velikost zásobníku vzorků. Platné hodnoty jsou 1 až 1024 vzorků. (Aktuální velikost zásobníku v bitech je Počet vzorků krát Počet kanálů.)
10	Počet vzorků po aktivaci	Udává počet vzorků, které se nashromáždí po té, co bude splněna podmínka aktivace. Tento parametr je možno nastavit na hodnotu v rozmezí 0 až (Počet vzorků - 1). Tento parametr je platný, pouze když režim aktivace bude nastavený na Aktivační vstup (0).
11	Pozice vstupního modulu	Udává umístění vstupního modulu pro vzorkování dat (pozice v hlavní sestavě). Pokud hodnota bude 0, čtení vstupního modulu bude zakázáno. Když se bude provádět čtení vstupního modulu, jeho hodnoty se uloží lokálně a na hodnoty adres nakonfigurovaných pro modul to nebude mít vliv. K uložení hodnot z načteného vstupního modulu do zásobníku vzorků bloku dat musí být určený popis kanálu. Pokud modul nebude existovat nebo bude vadný, data, která přijdou v okamžiku čtení, budou nula. Pokud k tomuto dojde, chyba se do tabulky chyb neuloží; indikace chyby zůstane na IO skenneru.
12	Volič typu bloku dat	Udává typ dat přiřazený Bloku dat. Pokud například budete chtít začít na %R0100, do tohoto parametru musíte zapsat 08. Platná nastavení tohoto parametru: %R (08h), %AI (0Ah), %AQ (0Ch). Podrobnosti k bloku dat viz strana 12-13.
13	Offset bloku dat	Udává počáteční adresu Bloku dat. Tento parametr vychází z nuly. Pokud například budete chtít začít na %R0100, do tohoto parametru musíte zapsat 99. Přesvědčte se, že pro celý blok dat budete mít dost paměti.
14-77	Popis kanálu	Udává adresu umístění (Volič typu, Délka a Offset) spojené s tímto konkrétním kanálem. V závislosti na počtu vzorkovaných kanálů a délce dat může být od 1 do 32 popisů kanálu. Data se vracejí v pořadí definovaném v této části.
	Volič/délka typu kanálu	Zapisuje se jako hexadecimální hodnota; toto slovo definuje volič typu a délku dat v (bitech). MSB = Volič typu. LSB = Délka dat. Délka dat se používá pro vzorky, které jsou sousedící. Volič typu je možno použít na libovolný typ diskrétních dat: %I (46h), %Q (48h), %M (4Ch), %T (4Ah), %G (56h), %S (54h), %SA (4Eh), %SB (50h), %SC (52h), Prázdný volič (FFh) a Volič vstupního modulu (00h).
		Délka parametru může být v rozsahu 1 - 32, ale součet všech délek nesmí být větší než parametr Počet kanálů. Délka větší než 1 umožňuje nakonfigurovat několik sousedících kanálů s popisem jediného kanálu.
	Offset kanálu	Zapisuje se jako hexadecimální hodnota; toto slovo definuje offset BIT pro typ dat nebo vstupní modul zadaný ve Voliči typu. Tento ofset začíná od nuly. Rozsah tohoto parametru se liší v závislosti na Voliči typu (typ a délka dat). Offset udává umístění v datové tabulce nebo vstupním modulu, kde se má provádět vzorkování.

Stavy přídavných stavových dat

Přídavná stavová data (slovo 1 v řídicím bloku funkce) udávají přídavné stavové informace pro funkci SER.

Hodnota	Stav	Popis
0	Stav Reset	Na vstup Reset přichází proud. Údaje Zásobník vzorků, Offset vzorku s aktivačním signálem, Aktivační čas a Offset aktuálního vzorku se vynulují. Výstup zůstává v nule. Přechod do ***Neaktivního stavu*** se provede, když se přeruší proud do vstupu Reset. Přídavná stavová data nemají žádný význam a vynulují se.
1	Neaktivní	Stav mezi stavem Reset a stavem Aktivní. V tomto stavu se neprovádějí žádné operace. Výstup SER zůstává v nule. Přechod do aktivního stavu se provede, když funkčním blokem bude téct proud.
2	Aktivní	Vstupem Povolení prochází proud, ale funkční blok není resetovaný, je v chybovém stavu nebo je aktivovaný. Když funkční blok bude povolený, na každé vykonání se zaznamená jeden vzorek. Výstup zůstává v nule. Snímá se stav Aktivace (zadaný parametrem Režim aktivace) a pokud podmínky budou splněné, provede se přechod do Stavu aktivace. Pokud bylo načteno více než "Počet vzorků", přídavná stavová data se nastaví na 0x10, v opačném případě se nastaví na 0x00.
3	Aktivovaný	Stav, kdy podmínka aktivace definovaná režimem Aktivace bude splněná. Další vzorky se načtou v závislosti na režimu aktivace a nastavení parametru. Výstup zůstává v nule. Přechod do stavu Dokončeno se provede, když bude dokončeno veškeré vzorkování. Pokud bylo načteno více než "Počet vzorků", přídavná stavová data se nastaví na 0x10, v opačném případě se nastaví na 0x00.
4	Dokončeno	Všechno vzorkování je dokončeno. Výstupem protéká proud. Je přípustný pouze přechod do stavu Reset. Pokud bylo načteno více než "Počet vzorků", přídavná stavová data se nastaví na 0x10, v opačném případě se nastaví na 0x00.
5	Chyba přejetí	Blok řízení/dat překročil konec typu paměti. Výstup zůstává v nule. Je přípustný pouze přechod do stavu Reset. Přídavná stavová data nemají žádný význam a vynulují se.
6	Chyba parametru	V řídicím bloku funkce nebo jiných operačních parametrech je chyba. Výstup zůstává v nule. Je přípustný pouze přechod do stavu Reset. Slovo Přídavná stavová data obsahuje offset do řídicího bloku, kde se vyskytla chyba parametru.
7	Chyba stavu	Parametr Stav je neplatný. Výstup zůstává v nule. Je přípustný pouze přechod do stavu Reset. Neplatná hodnota stavu se uloží v řídicím bloku na adrese Přídavná stavová data.

Formát bloku dat SER

Blok dat SER obsahuje zásobník vzorků, offset vzorků a informace o aktivaci. Tuto informaci předává CPU a z této paměťové oblasti je možno ji pouze číst. Přiřazení dostatečného místa pro registr bloku dat je věcí uživatele. Formát bloku je následující:

Slovo*	Popis parametru
0	Číslo offsetu aktuálního vzorku. Udává adresu, kde je umístěný poslední vzorek. Tento parametr vychází z nuly. Platný rozsah je -1 až 1023. Umístění registru vzorku = (Počet bajtů na vzorek) * (Parametr offsetu)/2 + (Počáteční registr zásobníku vzorků). **Poznámka:**Tato hodnota nebude platit, dokud se neprovede vzorkování. Když se provede reset funkce SER přes vstup Reset, tato hodnota se nastaví na -1.
1	Číslo offsetu vzorku s aktivačním signálem. Adresy umístění získaného vzorku, když stav aktivace přejde do stavu Splněno. Tento parametr vychází z nuly. Platný rozsah je -0 až 1023. Umístění registru vzorku = (Počet bajtů na vzorek) * (Parametr offsetu)/2 + (Počáteční registr zásobníku vzorků). **Poznámka:**Tato hodnota nebude platit, dokud nebude splněná podmínka aktivace. Když se provede reset funkce SER (přes vstup Reset), tato hodnota se nastaví na 0.
2 až 5	Aktivační čas: Udává čas podle denních hodin v PLC, kdy ve funkčním bloku stav aktivace přešel do stavu Splněno. Časová hodnota se zobrazí ve formátu BCD (výchozí) nebo formátu POSIX. Formát určuje parametr *Formát Aktivačního času* v řídicím bloku. Po aktivaci vstupu Reset se tato hodnota nastaví na nulu.
6 až konec zásobníku vzorků	Zásobník vzorků. Oblast paměti, ve které jsou uložena data vzorků. Když parametr resetování bude v jedničce, tato oblast bude nastavena na nulu. Velikost zásobníku vzorků se liší v závislosti na počtu kanálů a velikosti vzorku. Zásobník vzorků je kruhový zásobník - když se provede zápis do posledního místa, následující vzorek přepíše vzorek v prvním registru. Konec zásobníku vzorků = 5 + ({[(# odečtených vzorků) * (# vzorkovaných kanálů/ 8)] +1} / 2

*Offset od počáteční adresy definovaný Voličem typu bloku dat (slovo 12) a Offsetem bloku dat (slovo 13) v Řídicím bloku funkce.

Operace SER

Pokud při čtení bude SER povoleno, načtou se nakonfigurované vzorkovací body a umístí se do kruhového seznamu. Po načtení nakonfigurovaného počtu vzorků přejde výstup do jedničky. Přechod výstupu je možno použít k zaznamenání času odečtu posledního vzorku nebo ke spuštění dalšího vzorkování. (Viz "Režimy vzorkování.")

Před začátkem vzorkování se musí funkční blok SER resetovat (povolit proud vstupem Reset). Resetováním se provede inicializace oblasti bloku dat. Pokud stav funkčního bloku nebude resetovaný, vykoná operaci s aktuálními hodnotami v bloku dat, což bude mít za následek, že offset vzorků bude nesprávný a data v bloku dat budou nesprávná.

Řídicí blok funkčního bloku SER se čte při každém vykonání funkčního bloku ve stavu Reset, Aktivní nebo Aktivovaný. Pokud během vykonávání programu v Řídicím bloku změníte parametr, změna bude platná při dalším čtení funkčního bloku SER souvisejícího s Řídicím blokem. Pokud se zjistí chyba, operace se zastaví a funkční blok přejde do příslušného chybového stavu. Aby bylo

možno vzorkování spustit znovu, chybu je nutno odstranit a provést reset funkčního bloku (povolit proud vstupem Reset).

Pokud zvolíte čtení vstupního modulu, PLC *neověří*, že modul je diskrétní vstupní modul nebo že Popisy kanálu související s modulem mají platnou délku a offset podle velikosti modulu. Vzorkování vstupního modulu musíte nastavit správně. I když na vstupní modul může ukazovat několik popisů kanálu, modul se během vykonávání funkčního bloku bude číst vždy jen jednou.

Funkční blok SER je možno umístit do normálního uživatelského programu logiky nebo do periodického podprogramu. Pokud bude umístěný do uživatelského programu logiky, rozlišitelnost intervalu mezi čteními bude rozlišitelnost času čtení, která se může měnit v závislosti na počtu a typu funkcí aktivních při konkrétním čtení. Pokud funkční blok bude umístěný do podprogramu přerušení, interval je možno nastavit dokonce až na 1 ms a rozlišitelnost bude mít při 1 ms vysokou opakovatelnost s malým kolísáním.

Doba vykonávání jednoho funkčního bloku v případě periodického podprogram s 1ms může spotřebovat až 50% zdrojů CPU. Vykonávání více než dvou funkcí SER s periodickým podprogramem s 1 ms byste proto neměli plánovat.

Režimy vzorkování

Režim vzorkování SER určuje parametry Režim aktivace (slovo 2 v řídicím bloku funkce) a Počet vzorků po aktivaci (slovo 10). Obsah zásobníku vzorků je nutno interpretovat podle toho, jak jsou tyto parametry nakonfigurované.

Aktivací řízené vzorkování

Chcete-li nakonfigurovat režim vzorkování před aktivací, během aktivace a po aktivaci, je nutno zvolit Režim aktivace (slovo 2 = 0). Režim vzorkování se řídí Počtem vzorků po aktivaci (slovo 10). Ve všech případech se vzorkování spustí, když signál Povolení přejde do jedničky. Když signál Aktivace přejde do jedničky, vzorkování bude pokračovat, dokud se nenačte Počet vzorků po aktivaci. Když se vzorkování dokončí, signál Výstup funkčního bloku přejde do jedničky.

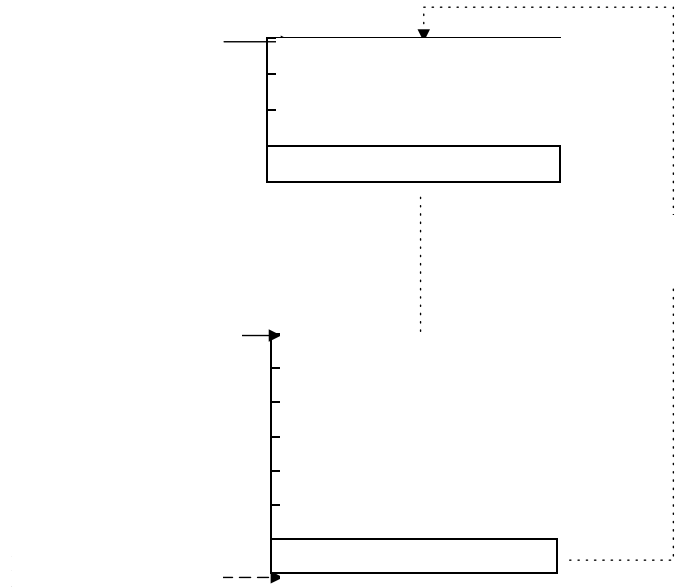
Pokud se před splněním podmínky Počet vzorků po aktivaci načte více než Počet vzorků (slovo 9), zásobník se “protočí”, to znamená, že SER se vrátí na začátek zásobníku a přepíše počáteční vzorky.

Když aktivace poprvé přejde ze stavu OFF do stavu ON, do nakonfigurovaného místa se uloží aktivační čas.

Před aktivací

Bude souvisle načítat vzorky, dokud se bude detekovat aktivace.

Chcete-li nakonfigurovat tento režim, nastavte slovo 10 na hodnotu 0 tak, aby se při signálu aktivace v jedničce vzorkování zastavilo a vygenerovala se časová značka. (Všechny vzorky načtené před aktivací).

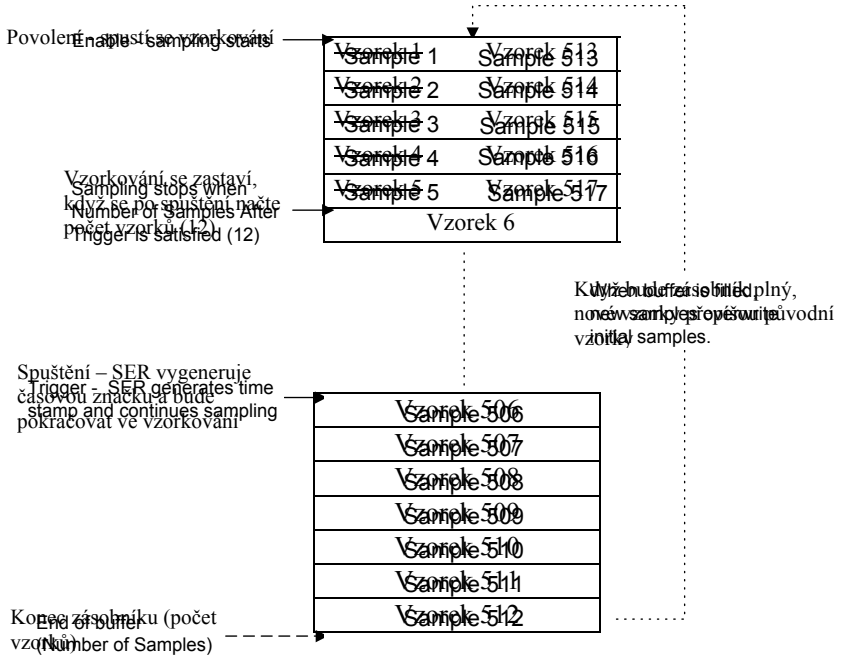


Obrázek 12-1. Příklad vzorkování SER před aktivací

Během aktivace

Provádí se souvislé načítání vzorků, dokud se nenačte Počet vzorků po aktivaci.

Chcete-li nakonfigurovat tento režim, nastavte slovo 10 na hodnotu 1 až Počet vzorků (slovo 9). Když signál aktivace bude v jedničce, vzorkování bude pokračovat, dokud se nenačte nastavený počet vzorků. V následujícím příkladu je Počet vzorků po aktivaci nastavený na 12. Když se vzorkování dokončí, zásobník bude obsahovat 500 vzorků před aktivací a 12 vzorků po aktivaci.

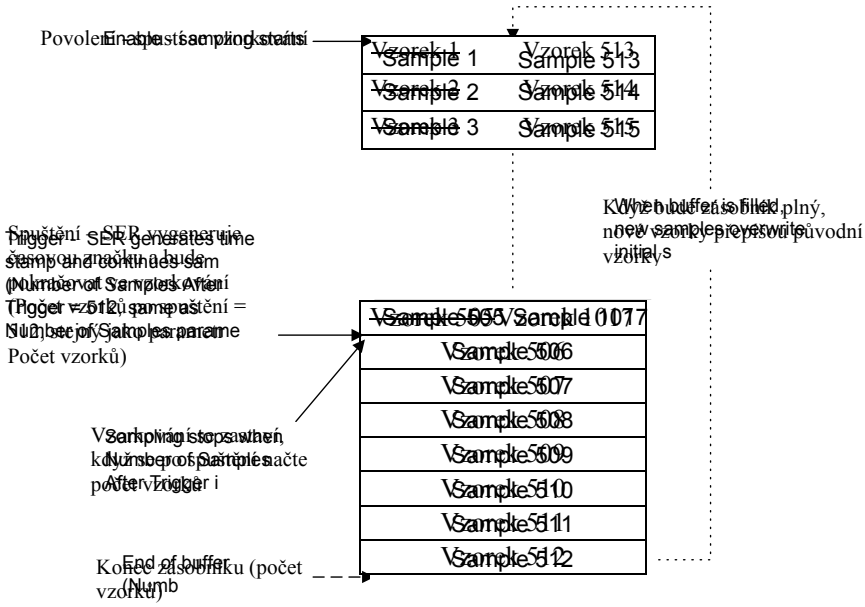


Obrázek 12-2. Příklad vzorkování SER během aktivace

Po aktivaci

Provádí se souvislé vzorkování, dokud se nenačte Počet vzorků.

Chcete-li nakonfigurovat tento režim, nastavte slovo 10 na hodnotu rovnající se Počtu vzorků (slovo 9). Když signál aktivace bude v jedničce, vzorkování bude pokračovat, dokud se nenačte nastavený počet vzorků. (Všechny vzorky načtené po aktivací).



Obrázek 12-3. Vzorkování SER po aktivaci

Plný zásobník (Aktivace neřídí vzorkování)

Pokud Režim aktivace bude nastavený na 1, parametr Počet vzorků po aktivaci (slovo 10) se bude ignorovat a vstupní signál Aktivace nebude mít na činnost funkčního bloku žádný vliv. Když bude funkční blok povolený, vzorkování bude pokračovat, dokud se nenačte Počet vzorků (slovo 8) a zásobník vzorků se nezaplní. Když zásobník vzorků bude plný, vygeneruje se značka Aktivační čas a výstup OK funkčního bloku přejde do jedničky.

Příklad SER

V následujícím příkladu byl sestavený funkční blok podle popisu v tabulce 12-1.

```
|                 _____
|%T0003  |       |                                                    %Q0003
|--| |---| SER_|-------------------------------------------------------( )-
|        |       |
|        |  BIT  |
|%T0001  |       |
|--|/|---|R      |
|        |       |
|%T0002  |       |
|--| |---|T      |
|        |_______|
|        %R0100
```

Příklad řídicího bloku funkce

V tomto příkladu systém má diskrétní vstupní modul se 16 body v sestavě 0 v pozici 4 a vykonával se dostatečně dlouho k načtení 572 vzorků (512 + 60). Vstupem Povolení protéká proud, ale vstupem reset a Aktivace ne.

Tabulka 12-1. Řídicí blok funkce pro příklad SER

Slovo	Registr	Parametr	Hodnota (dek)	Hodnota (hex)	Popis
0	%R0100	Stav	2	0002	Funkční blok je ve stavu Aktivní. To znamená, že funkční blok se vykonává normálně a při každém zjištění funkčního bloku v programu logiky provede načtení vzorku.
1	101	Přídavná stavová data	1	0001	Přídavná stavová data udávají, že se provedlo načtení více než 512 vzorků a že zásobník vzorků se již alespoň jednou protočil kolem dokola.
2	102	Režim aktivace	0	0000	Záznamník událostí je nakonfigurovaný k aktivaci podle vstupu Aktivace.
3	103	Formát Aktivačního času	0	0000	BCD
4	104	Vyhrazeno	0	0000	Vyhrazené parametry musí být vždy nastavené na 0.
5	105	Vyhrazeno	0	0000	
6	106	Vyhrazeno	0	0000	
7	107	Vyhrazeno	0	0000	
8	108	Počet kanálů	24	0018	Sestava vzorků se skládá z 24 bitů dat.
9	109	Počet vzorků, které se mají odečíst	512	0200	Velikost zásobníku je 512 vzorků. Všimněte si, že velikost zásobníku vzorků není 512 bajtů. Je to 512 x (24/8) = 1536 bajtů nebo 768 slov. (Každý vzorek má délku 3 bajty.)
10	110	Počet vzorků po aktivaci	12	000C	Počet vzorků, které se mají nashromáždit po aktivaci, je 12.
11	111	Pozice vstupního modulu	4	0004	Vstupní modul v sestavě 0 v pozici 4 se přečte tak, aby jeho hodnoty byly k dispozici pro vzorkování.
12	112	Volič typu bloku dat	8	0008	Datový typ je 0x08 (%R).
13	113	Offset bloku dat	200	00C8	Offset je 200, což umístí začátek bloku dat na %R0201. Offset je hodnota vycházející z nuly, ale tabulky registrů začínají na %R0001. Proto počáteční bod bloku dat je %R0001 + 200 = %R0201.

Slovo	Registr	Parametr	Hodnota (dek)	Hodnota (hex)	Popis
Popis kanálu		Zbývající slova obsahují popisy kanálů. V tomto příkladu bylo definováno šest popisů kanálů.			
14	114	Typ: Délka	17921	4601	Popis kanálu 1: Popis prvního kanálu zvolí typ %I s délkou 1 a offsetem 0. Tím se jako kanál 1 zvolí %I0001.
15	115	Offset	0	0000	
16	116	Typ : Délka	-253	FF03	Popis kanálu 2: Popis druhého kanálu zvolí volič NULL s délkou 3 a offsetem 0. Volič NULL má za následek, že kanály 2 - 4 se budou ignorovat nebo se "přeskočí". Tyto kanály budou vždy obsahovat hodnotu vzorku Nula.
17	117	Offset	0	0000	
18	118	Typ: Délka	3	0003	Popis kanálu 3: Popis třetího kanálu zvolí volič vstupního modulu s délkou 3 a offsetem 12. Volič vstupního modulu má za následek, že se vzorky odečítají ze vstupního modulu. Tento popis kanálu zvolí hodnoty v bodech 13, 14 a 15 vstupního modulu pro kanály 5 - 7.
19	119	Offset	12	0012	
20	120	Typ : Délka	18434	4802	Popis kanálu 4: Popis čtvrtého kanálu zvolí typ %Q s délkou 2 a offsetem 8. Tím se pro kanály 8 a 9 zvolí %Q0009 a %Q0010.
21	121	Offset	8	0008	
22	122	Typ: Délka	8	0008	Popis kanálu 5: Popis pátého kanálu je další volič vstupního modulu. Má délku 8 a offset 0. Má za následek, že hodnoty pro body 1 až 8 vstupního modulu se umístí do kanálu 10 - 17.
23	123	Offset	0	0000	
24	124	Typ: Délka	-249	FF07	Popis kanálu 6: Popis šestého kanálu je další volič NULL. Má délku 7 a offset 0. Popis kanálu NULL má za následek, že kanály 18 - 24 se vyplní nulami. Tento poslední popis kanálu je nutný k vyplnění zásobníku vzorků na 24 bitů předepsaných v parametru počtu kanálů. Protože je nakonfigurovaných všech 24 kanálů, další popisy kanálů nejsou zapotřebí.
25	125	Offset	0	0000	

Obsah vzorků

Tabulka 12-2 uvádí přehled hodnot obsažených v jednotlivých vzorcích podle popisů kanálů v řídicím bloku vzorku.

Tabulka 12-2. Obsah vzorků pro příklad SER

Číslo kanálu	Obsah kanálu
1	%I0001
2 - 4	Nuly
5	Bod 13 vstupního modulu
6	Bod 14 vstupního modulu
7	Bod 15 vstupního modulu
8	%Q0009
9	%Q0010
10 - 17	Body 1 - 8 vstupního modulu
18 - 24	Nuly

Blok dat pro vzorový řídicí blok

Tabulka 12-3 uvádí formát dat bloku, který je výsledkem vzorového řídicího bloku na stránce 12-17. Všimněte si, že začíná registrem 201, jak je popsáno v řídicím bloku parametrem offsetu typu (slova 12 a 13).

Tabulka 12-3. Blok dat pro vzorový řídicí blok SER

Offset	Registr	Popis parametru	Hodnota (dek)	Hodnota (hex)
0	%R0201	Číslo offsetu aktuálního vzorku	59	003B
1	202	Číslo offsetu vzorku s aktivačním signálem	0	0000
2 - 5	203 – 206	Aktivační čas (BCD)	0 0 0 0	0000 0000 0000 0000
6 - 768	207 – 975	Zásobník vzorků	Data vzorku	Data vzorku

Aktuální offset vzorku je 59, což znamená, že 59. vzorek je poslední vzorek umístěný do zásobníku vzorků (ne 59 registrů). Se 3 bajty na vzorek aktuální offset bude na 59 * 3 = 177 bajtů nebo nejvyšším bajtu 89. registru. Protože podmínky aktivace nebyly splněné, vzorek s aktivačním signálem a aktivační čas budou 0 a výstup se do jedničky nenastaví. Zásobník vzorků bude obsahovat 512 vzorků, kde 59 bude poslední vzorek a 60 bude nejstarší vzorek.

Formáty časových značek pro spuštění funkčního bloku SER

Příklad času spuštění 3. listopadu 1998 v 8:34:05:16.

BCD formát:

```
struct time_of_day_clk_rec {
        unsigned char  seconds;
        unsigned char  minutes;
        unsigned char  hours;
        unsigned char  day_of_month;
        unsigned char  month;
        unsigned char  year;
};
```

Registr	Parametr	Hodnota (dek)	Hodnota (hex)
%R203	Minuty/sekundy	13317	3405
%R204	Den v měsíci/hodiny	776	0308
%R205	Rok/měsíc	-26607	9811
%R206	Nepoužívá se	0	0

Formát POSIX::

```
struct timespec {
        long    tv_sec; /* Number of seconds since January 1, 1970 */
        long    tv_nsec;/* Number of nanoseconds into next seconds */
};
```

Registr	Parametr	Hodnota (dek)	Hodnota (hex)
%R203	Sekundy dolní slovo	-7811	e17d
%R204	Sekundy horní slovo	13845	3615
%R205	Nanosekundy dolní slovo	26624	6800
%R206	Nanosekundy horní slovo	2441	0989

END

Funkce END provádí přechodné ukončení logiky. Program se vykoná od první příčky k poslední příčce nebo k funkci END, podle toho, co je zjištěno dříve.

Funkce END bezpodmínečně ukončí vykonávání programu. Za funkcí ukončení v příčce již nesmí být nic. Za funkcí END se nevykoná žádná logika a řízení se předá na začátek programu pro další cyklus.

Funkce END je vhodná pro účely ladění, protože zabrání vykonání každé logiky, která by následovala.

Programovací software Logicmaster zařadí značku [END OF PROGRAM LOGIC], která indikuje konec vykonávání programu. Tato značka se používá, když se do logiky nenaprogramuje žádná funkce END.

Příklad

V následujícím příkladu je naprogramováno END k ukončení aktuálního cyklu.

```
|
|  STOP
|
|-[  END  ]
|
```

Poznámka

Umístěním funkce END do logiky SFC nebo logiky, kterou SFC vyvolává, u CPU verze 7 a pozdějších způsobí chybu “Funkce END vykonána z akce SFC”. (U CPU před verzí 7 funkce nepracovala správně, ale negenerovala se žádná chyba.) Informace o této chybě najdete v části “Nesoulad konfigurace systému” v kapitole 3, části 2.

MCRN/MCR

Funkce Hlavní řídicí relé (Master Control Relay - MCR) se musí používat s funkcí Konec hlavního řídicího relé (End Master Control Relay - ENDMCR). Funkce musí mít stejný název. Všechny příčky mezi aktivní funkcí MCR a odpovídající funkcí ENDMCR se vykonají, aniž by do cívek tekl proud. Funkce ENDMCR související s MCR bude mít za následek, že se obnoví vykonávání normálního programu. Na rozdíl od instrukce JUMP se funkce MCR mohou vyskytnout pouze v dopředném směru. Instrukce ENDMCR se v programu musí objevit až po její odpovídající instrukci MCR.

Pro MCR platí následující pravidla:

- Časovače neprovádějí inkrementaci nebo dekrementaci. Typy TMR se resetují. U funkčního bloku ONDTR je v registru uložena jeho hodnota.
- Normální výstupy jsou v nule; negované výstupy jsou v jedničce.

Poznámka

Když skrz MCR bude procházet proud, logika, kterou řídí, se přečte a zobrazí se stav kontaktů, ale žádný výstup nepřejde do jedničky. Pokud nebudete mít na paměti, že MCR řídí sledovanou logiku, může se vám zdát, že je v chybovém stavu. Aby se indikovalo, že MCR řídí žebříkovou logiku, software bude na obrazovce zobrazovat dvojitou napájecí lištu.

Software Logicmaster 90-30/20/Micro podporuje dvě formy funkce MCR, nevnořovanou a vnořovanou formu.

Kompatibilita CPU

Typ CPU	Instrukce hlavního řídicího relé
CPU Series 90 35x a 36x (verze 2 a pozdější)	Používejte pouze vnořovanou formu (MCRN)
CPU Series 90 verze 1	Používejte pouze nevnořovanou formu (MCR)

Operace MCRN

Funkci MCRN je možno umístit kdekoliv v programu, pokud bude řádně vnořená vzhledem k ostatním MCRN a nebude se vyskytovat uvnitř nevnořované MCR nebo nevnořovaného JUMP.

Pokud pár MCRN/ENDMCRN bude vnořený uvnitř jiného páru MCRN/ENDMCRN, musí být v tomto jiném páru obsažený celý. Je přípustné vnořování až do osmi úrovní. Příklad viz strana 12-24.

Poznámka

U CPU řady 35x a 36x použijte vždy jen jeden MCRN na každý ENDMCRN.

Jednomu ENDMCRN může odpovídat několik funkcí MCRN (s výjimkou CPU řady 35x a 36x, jak bylo uvedeno výše). To je analogické vnořovanému JUMP, kde ke stejnému návěští LABEL

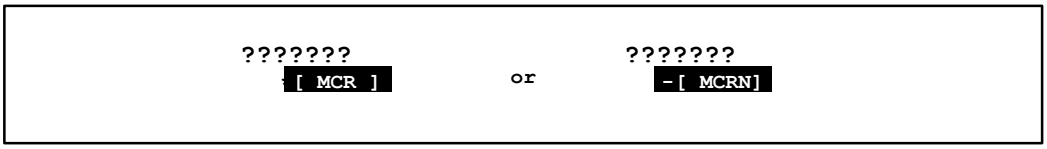
může existovat více instrukcí JUMP. Rozdíly mezi funkcí JUMP funkcí MCR najdete v odstavci “Rozdíly mezi MCR a skoky” na straně 12-23.

Operace MCR

Na každou instrukci ENDMCR může být pouze jedna instrukce MCR. Nevnořované MCR a ENDMCR se nesmí překrývat ani nesmí obsahovat žádný jiný pár instrukcí MCR/ENDMCR nebo pár instrukcí JUMP/LABEL. Nevnořované MCR se nesmí vyskytovat v rozsahu žádného páru JUMP/LABEL.

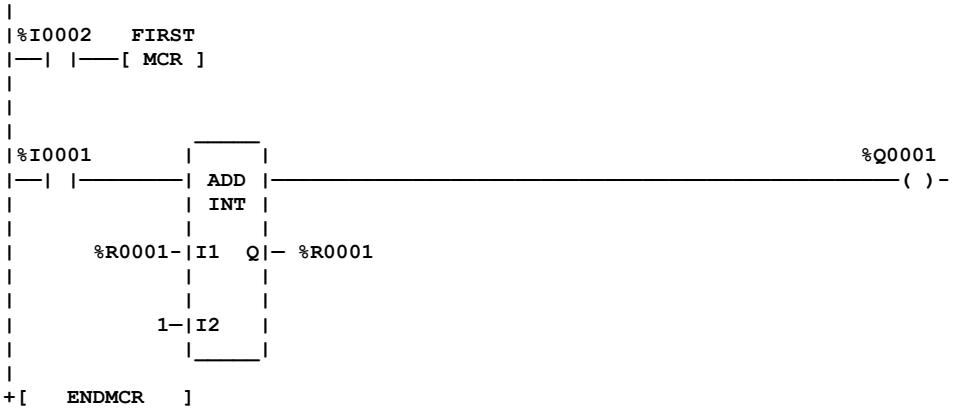
Parametry

Obě formy funkce MCR mají stejné parametry. Obě mají povolení Booleovského vstupu EN a název, který identifikuje MCR. Tento název se používá opět s instrukcí ENDMCR. Ani funkce MCR ani MCRN nemají žádný výstup; za MCR v příčce nesmí být nic.

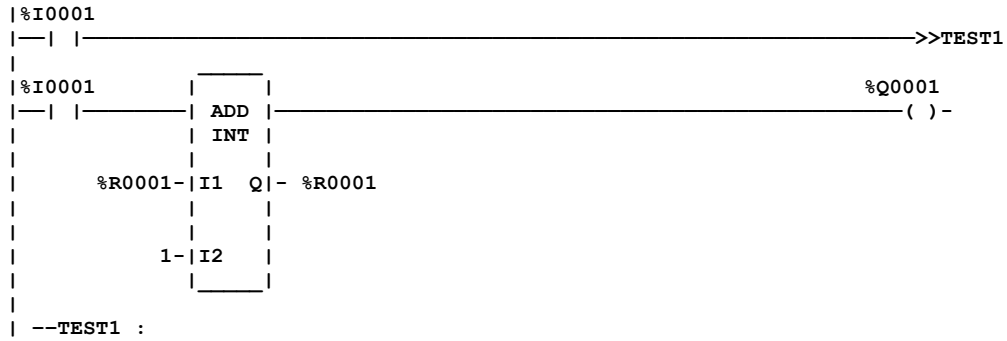


Rozdíly mezi MCR a JUMP

U funkce MCR se funkční bloky uvnitř MCR vykonají ***bez průtoku proud*** a cívky ***jsou vypnuté***. V následujícím příkladu bude MCR povoleno, když %I0002 bude ve stavu ON. Když bude MCR povoleno – i když %I0001 bude ve stavu ON – funkční blok ADD se vykoná ***bez*** průtoku proudu (tj. nepřipočítá se 1 k %R0001) a %Q0001 přejde do stavu OFF.



U funkce JUMP se mezi JUMP a LABEL ***nevykonají*** žádné funkční bloky a cívky ***zůstanou beze změny***. V následujícím příkladu, když %I0002 bude ve stavu ON, provede se JUMP. Protože logika mezi JUMP a LABEL se přeskočí, %Q0001 zůstane beze změny (tj., pokud bylo ve stavu ON, zůstane ve stavu ON; pokud bylo ve stavu OFF, zůstane ve stavu OFF).



Příklad

Následující příklad ukazuje MCRN s názvem "Second" vnořený uvnitř MCRN s názvem "First". Vždy když %I0002 umožní průtok proudu do funkce MCRN, vykonávání programu bude pokračovat bez průtoku proudu do cívek, dokud se nenarazí na odpovídající ENDMCRN. Pokud %I0001 a %I0003 budou ve stavu ON, %Q0001 přejde do stavu OFF a %Q0003 zůstane ve stavu ON.

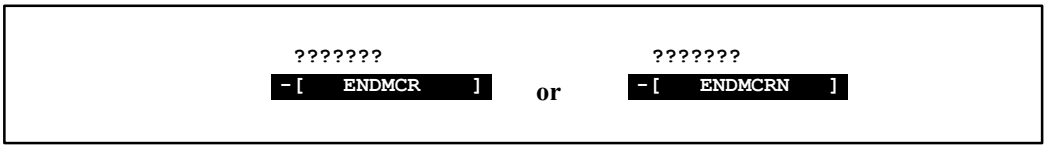
K usnadnění lokalizace chyb žebříkových programů dvojitá napájecí lišta identifikuje logiku, kterou řídí MCR.

ENDMCRN/ENDMCR

Funkce Konec hlavního řídicího relé (End Master Control Relay - ENDMCR) se používá k obnovení normálního vykonávání programu po funkci MCR. Když funkce MCR spojená s ENDMCR bude aktivní, ENDMCR vrátí vykonávání programu na normální průtok proudu. Když funkce MCR spojená s ENDMCR bude neaktivní, ENDMCR nebude mít žádný vliv.

Software Logicmaster 90-30/20/Micro podporuje dvě formy funkce ENDMCR, nevnořovanou a vnořovanou formu. Nevnořovaná forma, ENDMCR, se musí použít s nevnořovanou funkcí MCR, MCR. Vnořovaná forma, ENDMCRN, se musí použít se vnořovanou funkcí MCR, MCRN.

Funkce ENDMCR má negovaný Booleovský vstup EN. Povolení instrukce musí zajistit napájecí lišta; vykonávání nesmí být podmíněné. Funkce ENDMCR má také název, který ENDMCR identifikuje a spojuje jí s odpovídajícími MCR. Funkce ENDMCR nemá žádné výstupy; před ani za instrukcí ENDMCR v příčce nesmí nic být.



Příklad

V následujících příkladech je naprogramovaná instrukce ENDMCR k ukončení rozsahu MCR “clear.”

Příklad nevnořeného ENDMCR

```
|
|  CLEAR
|
|-[   ENDMCR    ]
|
```

Příklad vnořeného ENDMCR

```
|
|  CLEAR
|
|-[   ENDMCRN   ]
|
```

JUMP

Instrukce JUMP se používá k přeskočení části programu logiky. Vykonávání programu bude pokračovat v místě zadaném návěštím LABEL. Když JUMP bude aktivní, všechny cívky uvnitř jejího rozsahu zůstanou v původních stavech. To zahrnuje cívky související s časovači, čítači, přídržemi a relé.

Software Logicmaster 90-30/20/Micro podporuje dvě formy instrukce JUMP, nevnořovanou a vnořovanou formu. Nevnořovaná forma byla k dispozici u softwaru od verze 1 a má tvar ————————>>LABEL01, kde LABEL01 je název odpovídající nevnořené instrukce LABEL.

U nevnořovaných instrukcí JUMP, může ke každé instrukci LABEL existovat pouze jedna instrukce JUMP. JUMP může být JUMP směrem dopředu nebo dozadu.

Nevnořované JUMP a LABEL se nesmí překrývat ani nesmí obsahovat žádný jiný pár instrukcí JUMP/LABEL nebo pár instrukcí MCR/ENDMCR. Nevnořované instrukce JUMP a odpovídající LABEL nemohou být uvnitř jiného páru JMP/LABEL nebo jiného páru MCR/ENDMCR. Kromě toho každý pár MCR/ENDMCR nebo jiný pár JUMP/LABEL nesmí být uvnitř nevnořeného páru JUMP/LABEL.

Poznámka

Nevnořovaná forma instrukce JUMP je jediná instrukce JUMP, kterou je možno použít u PLC Series 90-30 verze 1. Vnořovanou funkci JUMP je možno použít (a je pro použití doporučená) pro všechny nové aplikace.

Všimněte si také, že CPU řady 35x a 36x podporují pouze vnořované skoky. CPU řady 35x a 36x nevnořované skoky nepodporují.

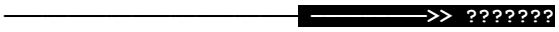
Vnořovaná forma instrukce JUMP má tvar ———N——>>LABEL01, kde LABEL01 je název odpovídající vnořené instrukci LABEL. Je možno jí použít u softwaru Logicmaster 90-30/20/Micro a firmwaru PLC verze 2 a pozdější.

Vnořovanou instrukci JUMP je možno do programu umístit kamkoliv za předpokladu, že se nebude vyskytovat uvnitř některé nevnořené instrukce MCR nebo nevnořené instrukce JUMP.

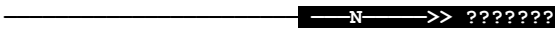
K jednomu vnořenému návěští LABEL může existovat několik vnořených instrukcí JUMP. Vnořené instrukce JUMP mohou být skoky směrem dopředu nebo dozadu.

Obě formy instrukce JUMP se v aktuální příčce vždy umisťují do sloupců 9 a 10; po instrukci JUMP v příčce již nic být nesmí. Průtok proudu přeskočí přímo z instrukce na příčku s názvem návěští.

Nevnořovaný JUMP:



Vnořovaný JUMP:



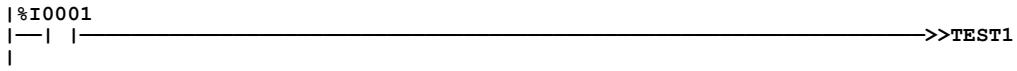
Upozornění

Aby u instrukcí JUMP směrem dopředu a dozadu nedošlo k vytvoření nekonečné smyčky, JUMP směrem dozadu musí obsahovat způsob, jak do instrukce zařadit podmínku.

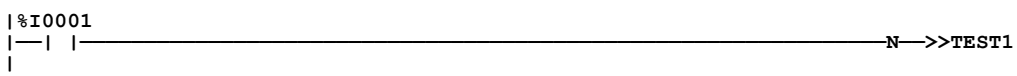
Příklady

V následujících příkladech vždy, když JUMP TEST1 bude aktivní, se tok proudu přenese na LABEL TEST1.

Příklad nevnořeného JUMP:



Příklad vnořeného JUMP:



LABEL

Instrukce LABEL funguje jako cílové místo pro JUMP. Instrukce LABEL se používá k obnovení normálního vykonávání programu po instrukci JUMP.

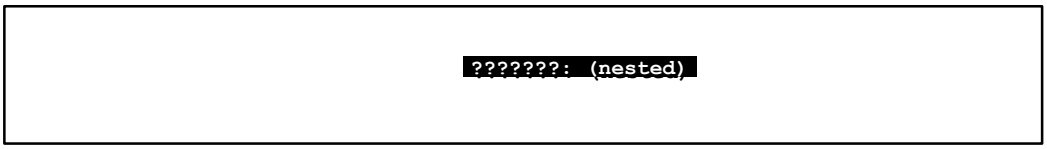
V programu smí být ve spojení s konkrétním názvem návěští pouze jedna instrukce LABEL. Programy bez odpovídajícího páru JUMP/LABEL lze v PLC vytvořit a uložit, ale nelze je vykonat.

Software Logicmaster 90-30/20/Micro podporuje dvě formy funkce LABEL, nevnořovanou a vnořovanou formu. Nevnořovaná forma, LABEL01:, se musí použít s nevnořovanou funkcí JUMP, ------>>LABEL01. Vnořovaná forma, LABEL01: (vnoření), se musí použít s vnořovanou funkcí JUMP, ———N——>>LABEL01.

Instrukce LABEL nemá žádné vstupy a žádné výstupy; před ani za instrukcí LABEL v příčce nesmí nic být.

Nevnořovaný LABEL:

Vnořovaný LABEL:



Příklad

V následujících příkladech se průtok proudu z JUMP TEST1 obnoví počínaje od LABEL TEST1.

Příklad nevnořeného LABEL:

```
|
| TEST1   :
|
```

Příklad vnořeného LABEL:

```
|
| TEST1   :(nested)
|
```

POZNÁMKA

Funkce COMMENT se používá k zapsání poznámky (vysvětlení příčky) do programu. Poznámka se může skládat až z 2048 znaků textu. V žebříkové logice je představována následovně:

Text je možno číst nebo editovat nastavením kurzoru na (* COMMENT *) po najetí na příčku a zvolení Zoom (F10). Text poznámky je možno také vytisknout.

Delší text je možno do výtisku zařadit pomocí anotačního textového souboru, jak je uvedeno níže:

1. Vytvoření poznámky:
   A. Zapište text do místa, kde má začínat text z jiného souboru.
   B. Nastavte kurzor na začátek nové řádky a zapište **\I** nebo **\i**, za jednotkou musí následovat dvojtečka, podadresář nebo adresář a název souboru, jak je uvedeno v následujícím příkladu.

      ```
      \I d:\text\commnt1
      ```

      Označení jednotky není nutné, pokud soubor bude umístěný na stejné jednotce jako adresář programu.
   C. Pokračujte v editování programu nebo přejděte do MS-DOS.
2. Po ukončení práce v programovacím zařízení vytvořte textový soubor pomocí libovolného programového balíku kompatibilního s MS-DOS. Souboru dejte název, který je zapsaný v poznámce, a umístěte ho do jednotky uvedené v poznámce.

SVCREQ

Funkce Service Request (SVCREQ) se používá k vyžádání některé z následujících speciálních služeb PLC:

Tabulka 12-4. Funkce Service Request

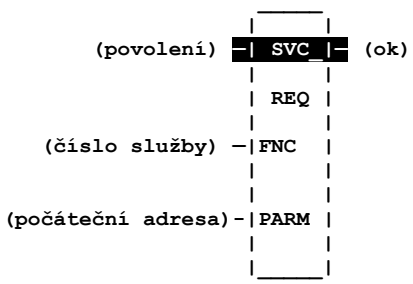
Funkce	Popis
1	Změna/čtení časovače konstantního cyklu
2	Čtení hodnot okna
3	Změna režimu okna komunikace programovacího zařízení a hodnoty časovače
4	Změna režimu okna komunikace systému a hodnoty časovače
6	Změna/čtení stavu kontrolního součtu úlohy a počtu slov pro kontrolní součet
7	Změna/čtení hodin denního času
8	Reset hlídacího časovače.
9	Čtení doby cyklu od začátku cyklu
10	Čtení názvu programu
11	Čtení PLC ID
12	Čtení stavu běhu PLC
13	Zastavení PLC
14	Vymazání tabulek chyb
15	Čtení posledního záznamu v tabulce chyb
16	Čtení hodin uplynulého času
18	Čtení stavu přepisu I/O
23	Čtení hlavního kontrolního součtu
26/30	Dotaz na I/O.
29	Čtení uplynulého času od vypnutí
45	Přeskočení dalšího výstupu a čtení vstupu (Pozastavení I/O.)
46	Přístup ke stavu rychlé vnitřní sběrnice

Přehled SVC REQ

Funkce SVCREQ má tři vstupní parametry a jeden výstupní parametr. Když funkcí SVCREQ bude protékat proud, PLC má vykonat uvedenou funkci FNC. Parametry pro funkci začínají na adrese dané pro PARM. Pokud nebude zadáno nesprávné číslo funkce, nesprávné parametry nebo adresy nebudou mimo rozsah, funkce SVCREQ proud propustí. Další příčiny neprovedení funkce jsou popsané na následujících stránkách.

Adresa uvedená pro PARM může reprezentovat každý typ slovní paměti (%R, %AI nebo %AQ). Tato adresa je první ze skupiny, která tvoří "blok parametrů" pro funkci. Přídavné parametry jsou uložené v 16 po sobě jdoucích místech. Celkový počet vyžadovaných adres závisí na typu použité funkce SVCREQ.

Bloky parametrů je možno použít jako vstupy funkce i jako místa, kam se po vykonání funkce zapíšou data. Proto přístup k datům, která funkce vygeneruje, je na stejném místě, které je zadáno pro PARM.



Parametry

Parametr	Popis
povolení	Když povolení bude v jedničce, požadavek na službu se vykoná.
FNC	FNC obsahuje konstantu nebo adresu pro požadavek služby.
PARM	PARM obsahuje počáteční adresu bloku parametrů pro požadovanou službu.
ok	Výstup ok bude v jedničce, když se funkce vykoná bez chyby.

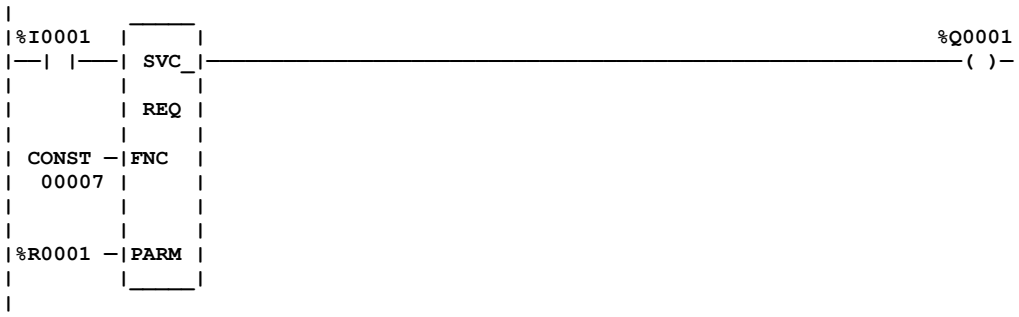
Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
FNC		•	•	•	•		•	•	•	•	•	
PARM		•	•	•	•		•	•	•	•		
ok	•											•

• Platná adresa nebo místo, kde funkcí může téct proud.

Příklad

V následujícím příkladu, když vstup povolení %I0001 bude ve stavu ON, vyvolá se číslo funkce SVCREQ 7 a blok parametrů bude umístěný počínaje od %R0001. Pokud se operace provede správně, výstupní cívka %Q0001 se nastaví do stavu ON.



SVCREQ #1: Změna/čtení časovače konstantního cyklu

Počínaje CPU 90-30 verze 8 se funkce SVCREQ #1 používá k:

- Zakázání režimu **konstantního cyklu**
- Povolení režimu **konstantního cyklu** a k použití staré hodnoty časovače
- Povolení režimu **konstantního cyklu** a použití nové hodnoty časovače
- Pouze k nastavení nového hodnoty časovače
- Čtení stavu režimu **konstantního cyklu** a hodnoty časovače

Poznámka

Ze všech CPU probíraných v tomto manuálu se Service Request 1 podporuje *pouze* u CPU 90-30 počínaje verzí 8.0.

Blok parametrů má délkou dvou slov.

Chcete-li režim **konstantního cyklu** zakázat, zapište funkci SVCREQ #1 s tímto blokem parametrů:

0	adresa
ignorováno	adresa + 1

Chcete-li režim **konstantního cyklu** povolit, zapište funkci SVCREQ #1 s tímto blokem parametrů:

1	adresa
0 nebo hodnota časovače	adresa + 1

Poznámka

Pokud má časovač použít novou hodnotu, zapište jí do druhého slova. Pokud se hodnota časovače měnit nemá, do druhého slova zapište 0. Pokud hodnota časovače již neexistuje, zápis 0 bude mít za následek, že funkce nastaví výstup OK do stavu OFF.

Chcete-li změnit hodnotu časovače **bez** změny volby stavu režimu cyklu, zapište funkci SVCREQ #1 s tímto blokem parametrů:

2	adresa
nová hodnota časovače	adresa + 1

Chcete-li načíst aktuální stav časovače a hodnotu bez změny jednoho nebo druhého, zapište funkci SVCREQ #1 s tímto blokem parametrů:

3	adresa
ignorováno	adresa + 1

Poznámka

Po použití funkce SVCREQ #1 s blokem parametrů uvedeným na předchozí stránce CPU verze 8 a vyšší předá hodnotu 0 v případě normálního cyklu a 1 v případě konstantního cyklu. Nezaměňujte je se *vstupními* hodnotami uvedenými níže.

Úspěšné vykonání se provede, pokud:

1. Jako požadovaná operace nebude zapsáno jiné číslo než 0, 1, 2 nebo 3:

0	Zakázání režimu **KONSTANTNÍHO CYKLU**
1	Povolení režimu **KONSTANTNÍHO CYKLU**
2	Nastavení pouze novou hodnotu časovače
3	Čtení režimu **KONSTANTNÍHO CYKLU** a hodnoty časovače (Viz poznámka výše.)

2. Časová hodnota nebude větší než 2550 ms (2.55 sekundy).
3. Nebude povolena konstantní doba cyklu bez naprogramované hodnoty časovače nebo se starou hodnotou časovače 0.

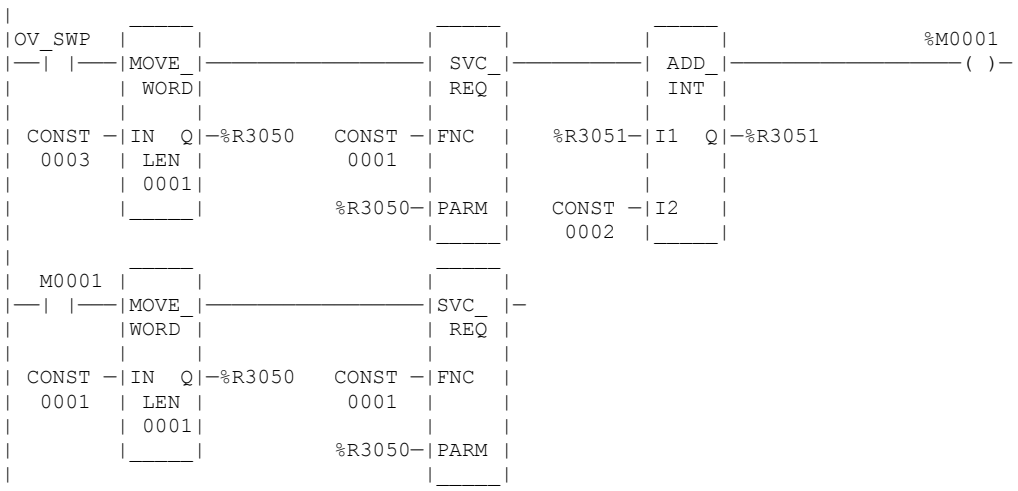
Po vykonání funkce předá stav časovače a hodnotu na stejných adresách bloků parametrů:

0 = zakázáno	
1 = povoleno	adresa
aktuální hodnota časovače	adresa + 1

Pokud slovo adresa + 1 bude obsahovat hexadecimální hodnotu FFFF, žádná hodnota časovače nebyla nikdy naprogramována.

Příklad

Tento příklad ukazuje logiku v programovém bloku. Když bude nastavený kontakt povolení OV_SWP, přečte se časovač konstantního cyklu, časovač se inkrementuje o dvě milisekundy a nová hodnota časovače se odešle zpět do PLC. Blok parametrů je lokální paměť na adrese %R3050. Protože funkce MOVE a ADD vyžadují tři horizontální polohy kontaktů, logika v příkladu používá diskrétní interní cívku %M0001 jako přechodnou adresu pro uložení úspěšného výsledku první příčky. V každém cyklu, kdy OV_SWP nebude nastaveno, se %M0001 nastaví do stavu OFF.



SVCREQ #2: Čtení hodnot okna

Funkce SVCREQ #2 se používá k získání hodnoty času aktuálního režimu okna pro okno programovacího zařízení, okno komunikace systému a okno úlohy na pozadí.

Poznámka

Ze všech CPU probíraných v tomto manuálu se Service Request 2 podporuje pouze u CPU 90-30 počínaje verzí 8.0.

Pro každé okno existují tři režimy:

Název režimu	Hodnota	Popis
Omezený režim	0	Doba vykonání okna je omezena na odpovídající výchozí hodnotu nebo hodnotu definovanou pomocí funkce SVCREQ #3 pro okno komunikace programovacího zařízení nebo funkci SVCREQ #4 pro okno systémové komunikace. Okno se ukončí, pokud již nebudou žádné další úlohy k vykonání.
Konstantní režim	1	Každé okno bude pracovat v **ÚPLNÉM** režimu a PLC bude přecházet mezi těmito třemi okny po dobu rovnající se součtu hodnot času jednotlivých oken. Pokud jedno okno bude v **KONSTANTNÍM** režimu, zbývající dvě okna automaticky přejdou do **KONSTANTNÍHO** režimu. Pokud PLC bude pracovat v režimu **KONSTANTNÍHO OKNA** a doba vykonávání konkrétního okna nebude definována pomocí odpovídající funkce SVCREQ, pro výpočet konstantní doby okna se použije výchozí hodnota doby pro toto okno.
Úplný režim	2	Bez ohledu na dobu okna související s konkrétním oknem, jestli doba bude výchozí nebo definovaná pomocí funkce požadavku služby, okno poběží, dokud se nedokončí všechny úlohy tohoto okna.

Okno se zakáže, když hodnota času bude nula.

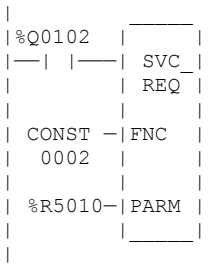
Blok parametrů má délkou tří slov. :

	Horní bajt	Dolní bajt	
Okno programovacího zařízení	Režim	Hodnota v ms	adresa
Okno komunikace systému	Režim	Hodnota v ms	adresa + 1
Okno pozadí			adresa + 2

Všechny parametry jsou výstupní parametry. K naprogramování této funkce není nutno do bloku parametrů hodnoty zapsat. Výstupní hodnoty pro všechna tři okna jsou uvedena v milisekundách.

Příklad

V následujícím příkladu, když bude nastavený vstup povolení %Q0102, operační systém PLC uloží aktuální hodnoty času těchto tří oken do bloku parametrů počínaje adresou %R5010. V následujících třech popisech funkcí SYS REQ jsou uvedené další příklady ukazující funkci Čtení hodnot okna.



SVCREQ #3: Změna režimu okna komunikace programovacího zařízení a hodnoty časovače

Funkce SVCREQ #3 se používá ke změně režimu okna komunikace programovacího zařízení a hodnoty časovače. Změna se provede v cyklu CPU následujícím po cyklu, kdy se funkce vyvolala.

Poznámka

Ze všech CPU probíraných v tomto manuálu se Service Request 3 podporuje pouze u CPU 90-30 počínaje verzí 8.0.

Funkce SVCREQ #3 bude přenášet proud doprava, pokud nebude zvolený jiný režim než 0 (Omezený), 1 (Konstantní) nebo 2 (Úplný).

Blok parametrů má délkou jednoho slova.

Chcete-li okno programovacího zařízení zakázat, zapište funkci SVCREQ #3 s tímto blokem parametrů:

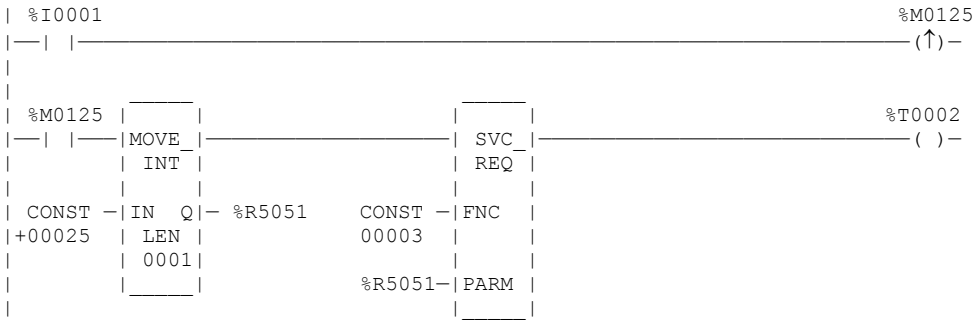
Horní bajt	Dolní bajt	
0	0	adresa

Chcete-li okno programovacího zařízení povolit, zapište funkci SVCREQ #3 s tímto blokem parametrů:

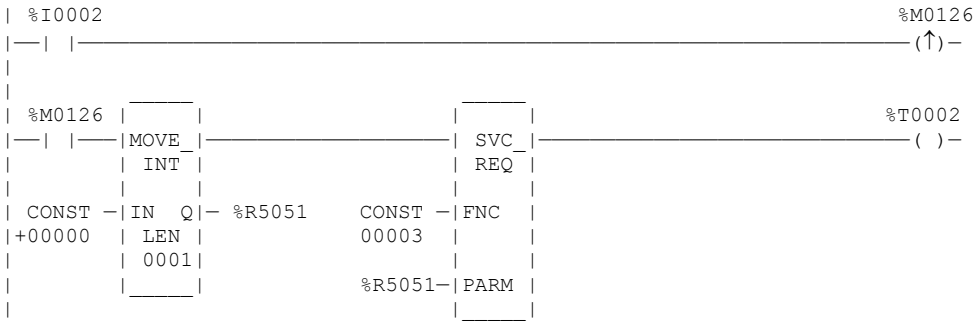
Horní bajt	Dolní bajt	
Režim	Hodnota od 1 do 255 ms	adresa

Příklad

V následujícím příkladu, když %M0125 přejde do stavu ON, okno komunikace programovacího zařízení bude povoleno a bude přiřazena hodnota 25 ms. Blok parametrů je v paměti na adrese %R5051.



Chcete-li okno komunikace programovacího zařízení zakázat, k přiřazení hodnoty nula (0) použijte Service Request 3. V tomto příkladu, když %M0126 přejde do stavu ON, okno komunikace programovacího zařízení bude povoleno a bude přiřazena hodnota 0 ms. Blok parametrů je v paměti na adrese %R5051.



SVCREQ #4: Změna režimu okna komunikace systému a hodnoty časovače

Funkce SVCREQ #4 se používá ke změně režimu okna komunikace systému a hodnoty časovače. Změna se provede v cyklu CPU následujícím po cyklu, kdy se funkce vyvolala.

Poznámka

Ze všech CPU probíraných v tomto manuálu se Service Request 4 podporuje pouze u CPU 90-30 počínaje verzí 8.0.

Funkce SVCREQ #4 bude přenášet proud doprava, pokud nebude zvolený jiný režim než 0 (Omezený), 1 (Konstantní) nebo 2 (Úplný).

Blok parametrů má délkou jednoho slova.

Chcete-li okno komunikace systému zakázat, zapište funkci SVCREQ #4 s tímto blokem parametrů:

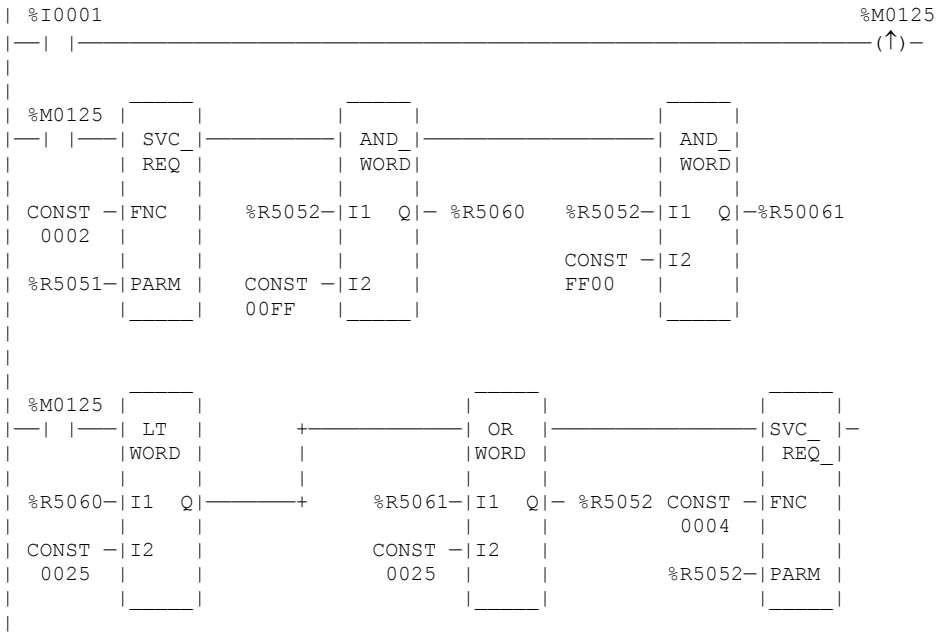
Horní bajt	Dolní bajt	
0	0	adresa

Chcete-li okno komunikace systému povolit, zapište funkci SVCREQ #4 s tímto blokem parametrů:

Horní bajt	Dolní bajt	
Režim	Hodnota od 1 do 255 ms	adresa

Příklad

V následujícím příkladu, když %M0125 přejde do stavu ON, načte se režim a hodnota časovače okna komunikace systému. Pokud hodnota časovače bude větší nebo se bude rovnat 25 ms, hodnota se nezmění. Pokud bude menší než 25 ms, hodnota se změní na 25 ms. V obou případech se okno povolí, když se vykonání příčky dokončí. Blok parametrů pro všechna tři okna je na adrese %R5051. Protože režim a hodnota časovače pro okno komunikace systému je druhá hodnota v bloku parametrů, kterou funkce Čtení hodnot okna (funkce #2) vygeneruje, adresa času existujícího okna pro okno komunikace systému bude v dolním bajtu adresy %R5052.



SVCREQ #6: Změna/čtení stavu kontrolního počtu slov pro kontrolní součet

Funkce SVCREQ se používá s funkcí číslo 6 k:

- Přečtení aktuálního počtu slov
- Nastavení nového počtu slov

Úspěšné vykonání se provede, pokud jako požadovaná operace (viz níže) nebude zadáno jiné číslo než 0 nebo 1.

Pro funkce úlohy kontrolního součtu má blok parametrů délku dvou slov.

Chcete-li přečíst aktuální počet slov:

Zapište funkci SVCREQ 6 s tímto blokem parametrů:

0	adresa
ignorováno	adresa + 1

Po vykonání funkce se předá aktuální kontrolní součet v druhém slově bloku parametrů. Pro funkci čtení se nezadává žádný rozsah; předaná hodnota je počet slov, u kterých se provedl kontrolní součet.

0	adresa
aktuální počet slov	adresa + 1

Chcete-li nastavit nový počet slov:

Zapište funkci SVCREQ 6 s tímto blokem parametrů:

1	adresa
nový počet slov	adresa + 1

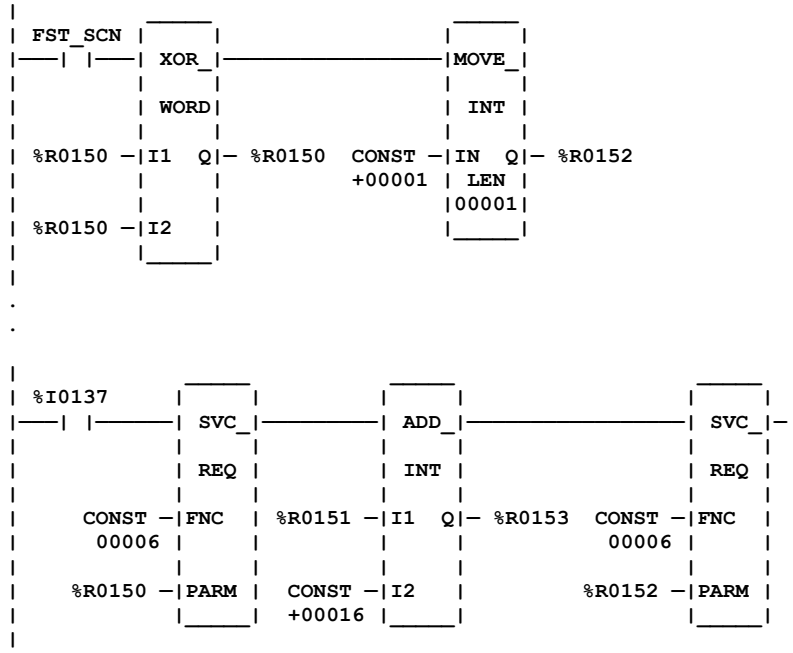
Zapsání 1 bude mít za následek, že PLC provede úpravu počtu slov, se kterými se má provést kontrolní součet, na hodnotu uvedenou v druhém slově bloku parametrů. V případě CPU 331 nebo 311 počet může být 0 nebo 32; v případě CPU 211 hodnota může být 0 nebo 4.

Poznámka

Tento Service Request nelze použít u PLC Series90 Micro.

Příklad

V následujícím příkladu, když bude nastavený kontakt povolení FST_SCN, vytvoří se bloky parametrů pro funkci úlohy kontrolního součtu. Dále, když v programu vstup %I0137 přejde do stavu ON, z operačního systému PLC se načte počet slov, se kterými se má provést kontrolní součet. Tento počet se zvýší o 16 a výsledky funkce ADD_INT se uloží do parametru "nové nastavení počtu". Druhý blok požadavku služby požaduje, aby PLC nastavilo nový počet slov.



Bloky parametrů v příkladu jsou umístěné na adrese %R0150. Mají následující obsah:

0 = načíst aktuální počet	%R0150
uchovat aktuální počet	%R0151

1 = nastavit aktuální počet	%R0152
uchovat nový počet pro nastavení	%R0153

SVCREQ #7: Změna/čtení hodin denního času

Funkce SVCREQ se používá se funkcí číslo 7 ke čtení a k nastavení hodin denního času v PLC.

Poznámka

Tuto funkci je možno použít pouze s CPU 90-30 verze 331 nebo vyšší a s CPU PLC Series 90 Micro s 28 body (to je, IC693UDR005, IC693UAA007 a IC693UDR010) a CPU PLC Series 90 Micro s 23 body (IC693UAL006).

Úspěšné vykonání se provede, pokud:

1. Jako požadovaná operace (viz níže) nebude zapsáno jiné číslo než 0 nebo 1.
2. Nebudou zadána neplatná data.
3. Předaná data budou v očekávaném formátu.

U funkcí datumu/času délka bloku parametrů závisí na formátu dat. BCD formát vyžaduje 6 slov; komprimovaný ASCII vyžaduje 12 slov.

0 = čtení času a datumu 1 = nastavení času a datumu	adresa
1 = BCD formát 3 = komprimovaný ASCII formát	adresa + 1
data	adresa + 2 do konce

Ve slově 1 zadejte, jestli funkce má provést čtení nebo změnu hodnot.

```
0    =    čtení
1    =    změna
```

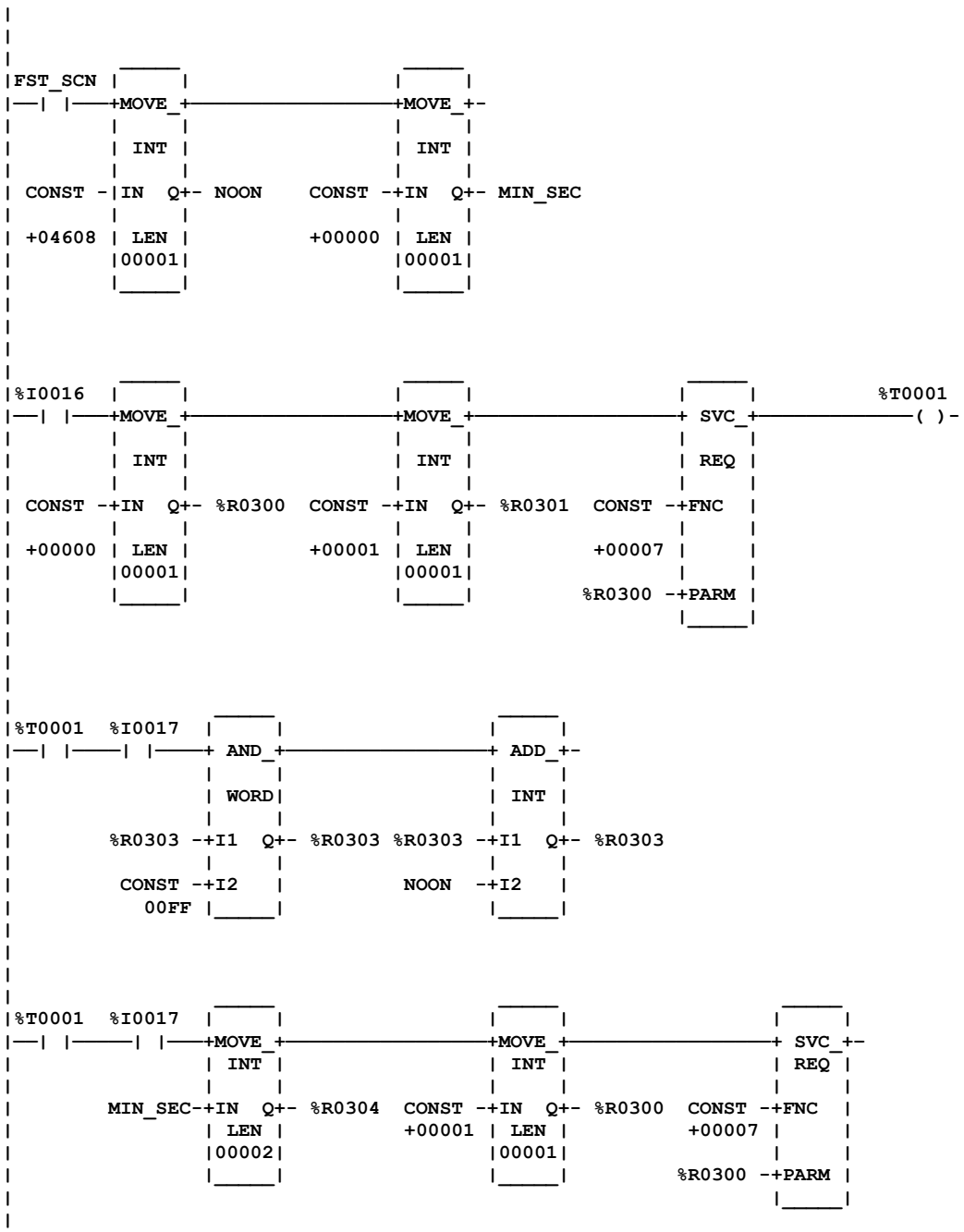
Ve slově 2 zadejte formát dat.

```
1    =    BCD
3    =    komprimovaný ASCII s vloženými mezerami a dvojtečkami
```

Slova 3 až konec bloku parametrů obsahují výstupní data předaná funkcí čtení nebo nová data poskytovaná funkcí změny. V obou případech je formát těchto datových slov stejný. Když se provádí čtení datumu a času, slova (adresa + 2) až (adresa + 8) bloku parametrů se na vstupu budou ignorovat.

Příklad

V následujícím příkladu, když se vyvolá předchozí logika, blok parametrů pro hodiny denního času se vytvoří a nejdříve se vyžádá aktuální datum a čas a pak se hodiny nastaví na 12 hodin v poledne ve formátu BCD. Blok parametrů je umístěný na adrese globálních dat %R0300. Pole NOON může být v programu nastaveno kdekoliv tak, aby obsahovalo hodnoty 12, 0 a 0. (Pole NOON musí také obsahovat data na %R0300.) BCD formát vyžaduje pro blok parametrů šest po sobě jdoucích paměťových míst.



Obsah bloku parametrů

Obsah bloku parametrů pro různé formáty dat je uvedený na následujících stránkách. Pro oba formáty dat platí, že:

- Hodiny jsou uložené ve 24-hodinovém formátu.
- Den v týdnu je číselná hodnota:

Hodnota	Den v týdnu
1	Neděle
2	Pondělí
3	Úterý
4	Středa
5	Čtvrtek
6	Pátek
7	Sobota

Změna/načtení datumu a času s použitím BCD formátu:

Ve formátu BCD každá položka datumu a času zabere jeden bajt. Tento formát vyžaduje šest slov. Poslední bajt šestého slova se nepoužívá. Když se nastavuje datum a čas, tento bajt se ignoruje; když se čte datum a čas, funkce předá prázdný znak (00).

Horní bajt	Dolní bajt		Příklad výstupního bloku parametrů: Načtení datumu a času v BCD formátu (Neděle, 3. června, 1988 v 2:45:30 odpoledne)	
1 = změna nebo	0 = čtení	adresa	0	
1		adresa + 1	1	
měsíc	rok	adresa + 2	07	88
hodiny	den v měsíci	adresa + 3	14	03
sekundy	minuty	adresa + 4	30	45
(prázdný)	den v týdnu	adresa + 5	00	01

Změna/načtení datumu a času s použitím komprimovaného formátu ASCII s vloženými dvojtečkami

V případě komprimovaného formátu ASCII každá číslice položky času a datumu je jeden formátovaný bajt ASCII. Kromě toho se do data vloží mezera a dvojtečky, aby se datum mohlo přenést beze změny na tiskové nebo zobrazovací zařízení. Tento formát vyžaduje 12 slov.

Horní bajt	Dolní bajt		Příklad výstupního bloku parametrů: Načtení datumu a času v komprimovaném ASCII formátu (Pondělí, 2. října 1989 ve 23:13:00)	
1 = změna nebo	0 = čtení	adresa	0	
3		adresa + 1	3	
rok	rok	adresa + 2	39	38
měsíc	(mezera)	adresa + 3	31	20
(mezera)	měsíc	adresa + 4	20	30
den v měsíci	den v měsíci	adresa + 5	32	30
hodiny	(mezera)	adresa + 6	32	20
:	hodiny	adresa + 7	3A	33
minuty	minuty	adresa + 8	33	31
sekundy	:	adresa + 9	30	3A
(mezera)	sekundy	adresa + 10	20	30
den v týdnu	den v týdnu	adresa + 11	32	30

SVCREQ #8: Reset hlídacího časovače

Funkce SVCREQ #8 se používá k resetování hlídacího časovače během cyklu.

Poznámka

Ze všech CPU probíraných v tomto manuálu se Service Request 8 podporuje pouze u CPU 90-30 počínaje verzí 8.0.

Když uplyne doba hlídacího časovače, PLC ukončí činnost bez varování. Tato funkce umožňuje, aby časovač pokračoval v činnosti během úlohy náročné na čas (například když čeká na odezvu od komunikační linky).

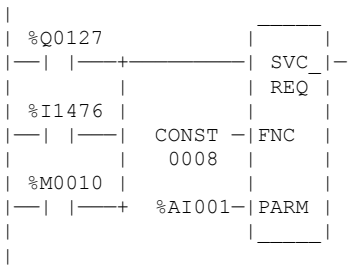
Upozornění

Přesvědčte se, že restartování hlídacího časovače nebude mít nepříznivý vliv na řízený proces.

Tato funkce není spojena s žádným blokem parametrů; avšak programovací software vyžaduje, aby byl provedený zápis do PARM. Zapište sem příslušnou adresu; nebude se používat.

Příklad

V následujícím příkladu, když bude nastavený výstup povolení %Q0127 nebo vstup povolení %I1476 nebo interní cívky %M0010, provede se reset hlídacího časovače.



SVCREQ #9: Čtení doby cyklu od začátku cyklu

Funkce SVCREQ #9 se používá k načtení času v milisekundách od začátku cyklu. Data jsou v 16-bitovém slovním formátu.

Poznámka

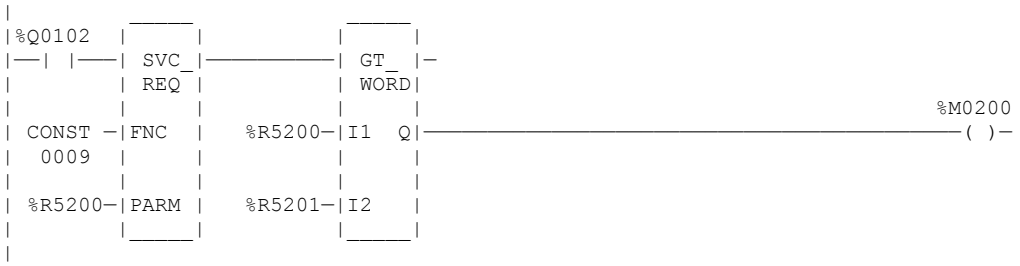
Ze všech CPU probíraných v tomto manuálu se Service Request 9 podporuje pouze u CPU 90-30 počínaje verzí 8.0.

Blok parametrů je pouze výstupní blok parametrů; má délku jednoho slova.

doba od začátku cyklu	adresa

Příklad

V následujícím příkladu se doba uplynulá od začátku cyklu vždy načte do adresy %R5200. Pokud bude větší než hodnota v %R5201, interní cívka %M0200 přejde do stavu ON.



SVCREQ #10: Čtení názvu programu

Funkce SVCREQ #10 se používá k načtení názvu aktuálně vykonávaného programu.

Poznámka

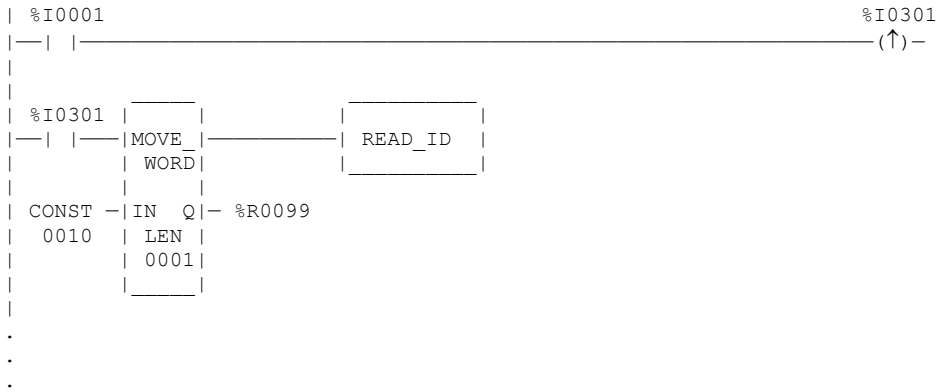
Ze všech CPU probíraných v tomto manuálu se Service Request 10 podporuje *pouze* u CPU 90-30 počínaje verzí 8.0.

Výstupní blok parametrů má délku čtyř slov. Předá osm znaků ASCII; poslední je prázdný znak (00h). Pokud název programu bude mít méně než sedm znaků, na konec se připojí prázdné znaky.

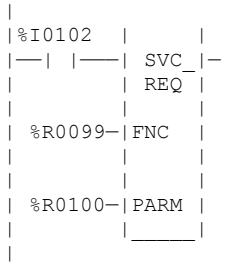
Dolní bajt	Horní bajt	
znak 1	znak 2	adresa
znak 3	znak 4	adresa + 1
znak 5	znak 6	adresa + 2
znak 7	00	adresa + 3

Příklad

V následujícím příkladu, když vstup pro povolení %I0301 přejde do stavu OFF, adresa registru %R0099 se načte s hodnotou 10, což je kód pro funkci Čtení názvu programu. Pak se vyvolá programový blok READ_ID a název programuse obnoví. Blok parametrů je umístěný na adrese %R0100. READ_ID se také používá v následujícím příkladu.



Programový blok READ_ID



SVCREQ #11: Čtení PLC ID

Funkce SVCREQ #11 se používá k načtení názvu PLC Series 90 vykonávajícího program.

Poznámka

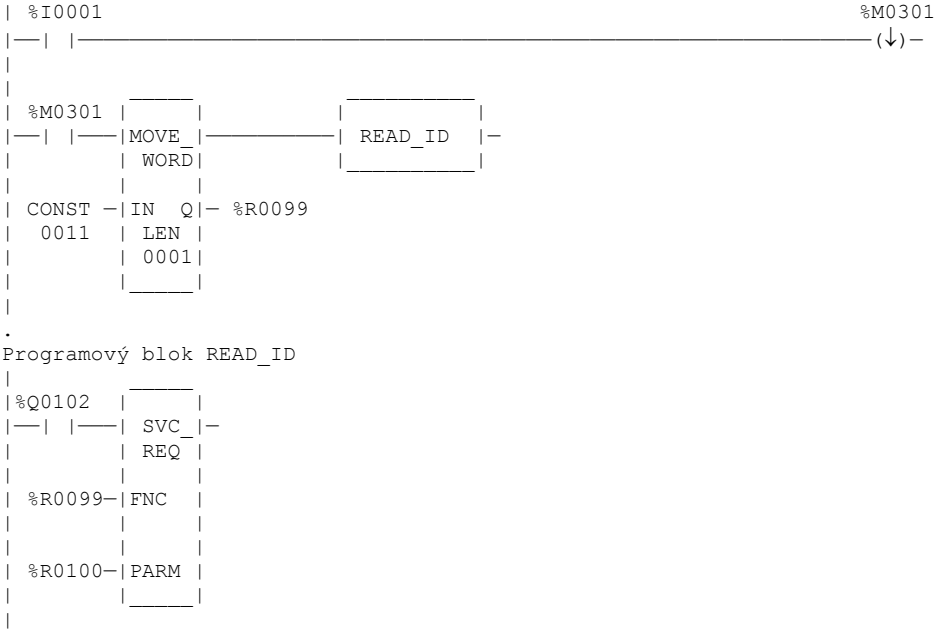
Ze všech CPU probíraných v tomto manuálu se Service Request 11 podporuje *pouze* u CPU 90-30 počínaje verzí 8.0.

Výstupní blok parametrů má délku čtyř slov. Předá osm znaků ASCII; poslední je prázdný znak (00h). Pokud PLC ID bude mít méně než sedm znaků, na konec se připojí prázdné znaky.

Dolní bajt	Horní bajt	
znak 1	znak 2	adresa
znak 3	znak 4	adresa + 1
znak 5	znak 6	adresa + 2
znak 7	00	adresa + 3

Příklad

V následujícím příkladu, když vstup pro povolení %I0001 přejde do stavu OFF, adresa registru %R0099 se načte s hodnotou 11, což je kód PLC ID funkce pro funkci Čtení PLC ID. Pak se vyvolá programový blok READ_ID a ID se obnoví. Blok parametrů je umístěný na adrese %R0100. S výjimkou kontaktu povolení a čísla funkce toto je stejný kód použitý v předchozím příkladu.



SVCREQ #12: Čtení stavu běhu PLC

Funkce SVCREQ #12 se používá k načtení aktuálního stavu chodu CPU PLC.

Poznámka

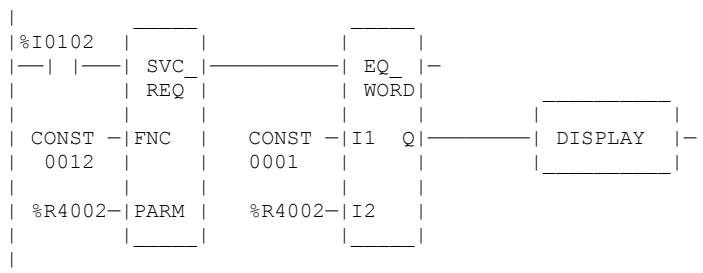
Ze všech CPU probíraných v tomto manuálu se Service Request 12 podporuje *pouze* u CPU 90-30 počínaje verzí 8.0.

Blok parametrů je pouze výstupní blok parametrů; má délku jednoho slova.

1 = běh/zakázáno	adresa
2 = běh/povoleno	

Příklad

V následujícím příkladu se stav běhu PLC vždy načte do adresy %R4002. Pokud stav bude Běh/Zakázáno (Run/Disabled), funkce CALL vyvolá programový blok DISPLAY.



SVCREQ #13: Zastavení PLC

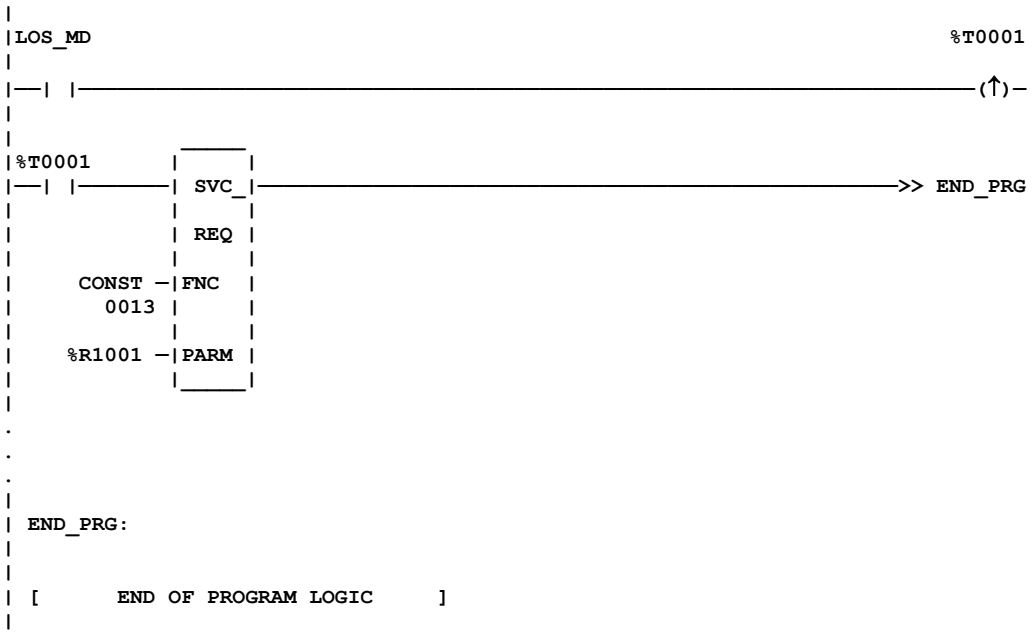
Funkce SVCREQ #13 se používá k zastavení PLC ***na konci následujícího cyklu***. Všechny výstupy přejdou do příslušných výchozích stavů na začátku následujícího cyklu PLC. Do tabulky chyb PLC se vloží informativní chyba a nevykoná se nic kromě funkčního bloku "SHUT DOWN PLC". Čtení I/O bude pokračovat podle konfigurace.

Tato funkce nemá žádný blok parametrů.

Příklad

V následujícím příkladu, když se vyskytne chyba “Ztráta I/O modulu”, se vykoná funkce SVCREQ #13. Protože není zapotřebí žádný blok parametrů, vstup PARM se nepoužije; avšak programovací software vyžaduje, aby byl provedený zápis do PARM.

Tento příklad používá JUMP na konci programu a vynutí jeho uzavření, když se funkce Zastavení PLC vykoná úspěšně. Tento JUMP a LABEL jsou zapotřebí, protože přechod do režimu **STOP** se provede až na konci cyklu, ve kterém se funkce vykoná.



Poznámka

Aby bylo zaručeno, že kontakt %S0002 LST_SCN bude pracovat správně, PLC vykoná jeden cyklus navíc po cyklu, ve kterém se vykonala funkce SVCREQ #13.

SVCREQ #14: Vymazání tabulek chyb

Funkce SVCREQ #14 se používá k vymazání tabulky chyb PLC nebo tabulky chyb I/O. Pokud jako požadovaná operace (viz níže) nebude zadáno jiné číslo než 0 nebo 1, výstup SVCREQ se nastaví do stavu ON.

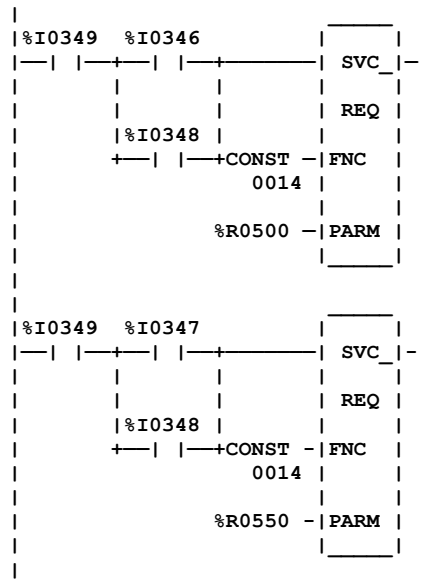
U této funkce blok parametrů má délku 1 slovo. Je to pouze blok vstupního parametru.

0 = smazat tabulku chyb PLC 1 = smazat tabulku chyb I/O	adresa

Příklad

V následujícím příkladu, když vstup %I0346 bude ve stavu ON a vstup %I0349 bude ve stavu ON, smaže se tabulka chyb PLC. Když vstup %I0347 bude ve stavu ON a vstup %I0349 bude ve stavu ON, smaže se tabulka chyb I/O. Když bude vstup %I0348 ve stavu ON a vstup %I0349 ve stavu ON, smažou se obě tabulky.

Blok parametrů pro tabulku chyb PLC je na adrese %R0500; blok parametrů pro tabulku chyb I/O je na adrese %R0550. Oba bloky parametrů je možno umístit kdekoliv v programu.



SVCREQ #15: Čtení posledního záznamu v tabulce chyb

Funkce SVCREQ #15 se používá k přečtení posledního záznamu buď v tabulce chyb PLC nebo tabulce chyb I/O. Pokud jako požadovaná operace (viz níže) nebude zadáno jiné číslo než 0 nebo 1 nebo tabulka chyb nebude prázdná, výstup SVCREQ se nastaví do stavu ON. (Další informace o záznamech v tabulce chyb najdete v kapitole 3, "Výklad a oprava chyb.")

U této funkce blok parametrů má délku 22 slov. Blok vstupního parametru má následující formát:

0 = Načtení tabulky chyb PLC 1 = Načtení tabulky chyb I/O	adresa

Formát bloku výstupního parametru závisí na tom, jestli funkce potřebuje číst data z tabulky chyb PLC nebo tabulky chyb I/O.

Výstupní formát tabulky chyb PLC

Dolní bajt	Horní bajt	
0		
dlouhý/krátký		adresa + 1
volný		adresa + 2
adresa chyby PLC		adresa + 3
		adresa + 4
skupina chyby a akce		adresa + 5
chybový kód		adresa + 6
		adresa + 7
		adresa + 8
		adresa + 9
		adresa + 10
		adresa + 11
specifická data chyby		adresa + 12
		adresa + 13
		adresa + 14
		adresa + 15
		adresa + 16
		adresa + 17
		adresa + 18
		adresa + 19
časová značka		adresa + 20
		adresa + 21

Výstupní formát tabulky chyb I/O

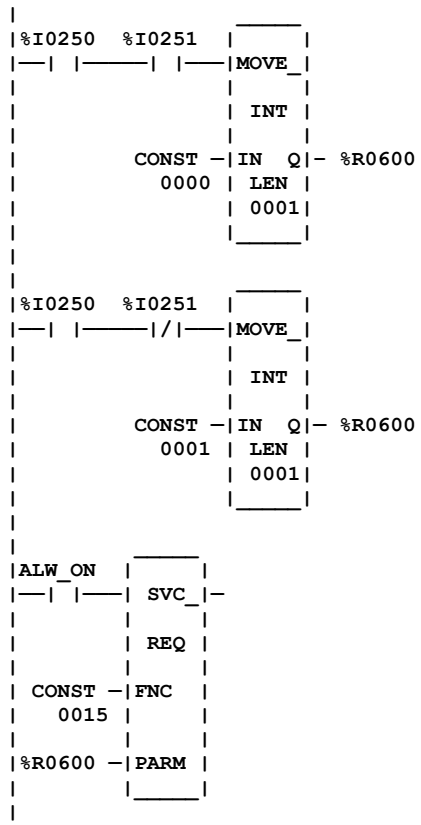
Dolní bajt	Horní bajt
1	
dlouhý/krátký	
adresa	
adresa chyby I/O	
skupina chyby a akce	
kategorie chyby	typ chyby
popis chyby	
specifická data chyby	
časová značka	

V první bajtu adresy slova + 1 indikátor Dlouhý/krátký definuje množství specifických dat chyby, které jsou v záznamu chyb. Může to být:

```
Tabulka chyb PLC:  00 = -8 bajtů (krátký)
                   01 = 24 bajtů (dlouhý)
Tabulka chyb I/O:  02 = —5 bajtů (krátký)
                   03 = 21 bajtů (dlouhý)
```

Příklad 1

V následujícím příkladu, když vstup %I0251 bude ve stavu ON a vstup %I0250 bude ve stavu ON, do bloku parametrů se načte poslední záznam v tabulce chyb PLC. Když vstup %I0251 bude ve stavu ON a vstup %I0250 bude ve stavu ON, do bloku parametrů se načte poslední záznam v tabulce chyb I/O. Blok parametrů je umístěný na adrese %R0600.



Příklad 2

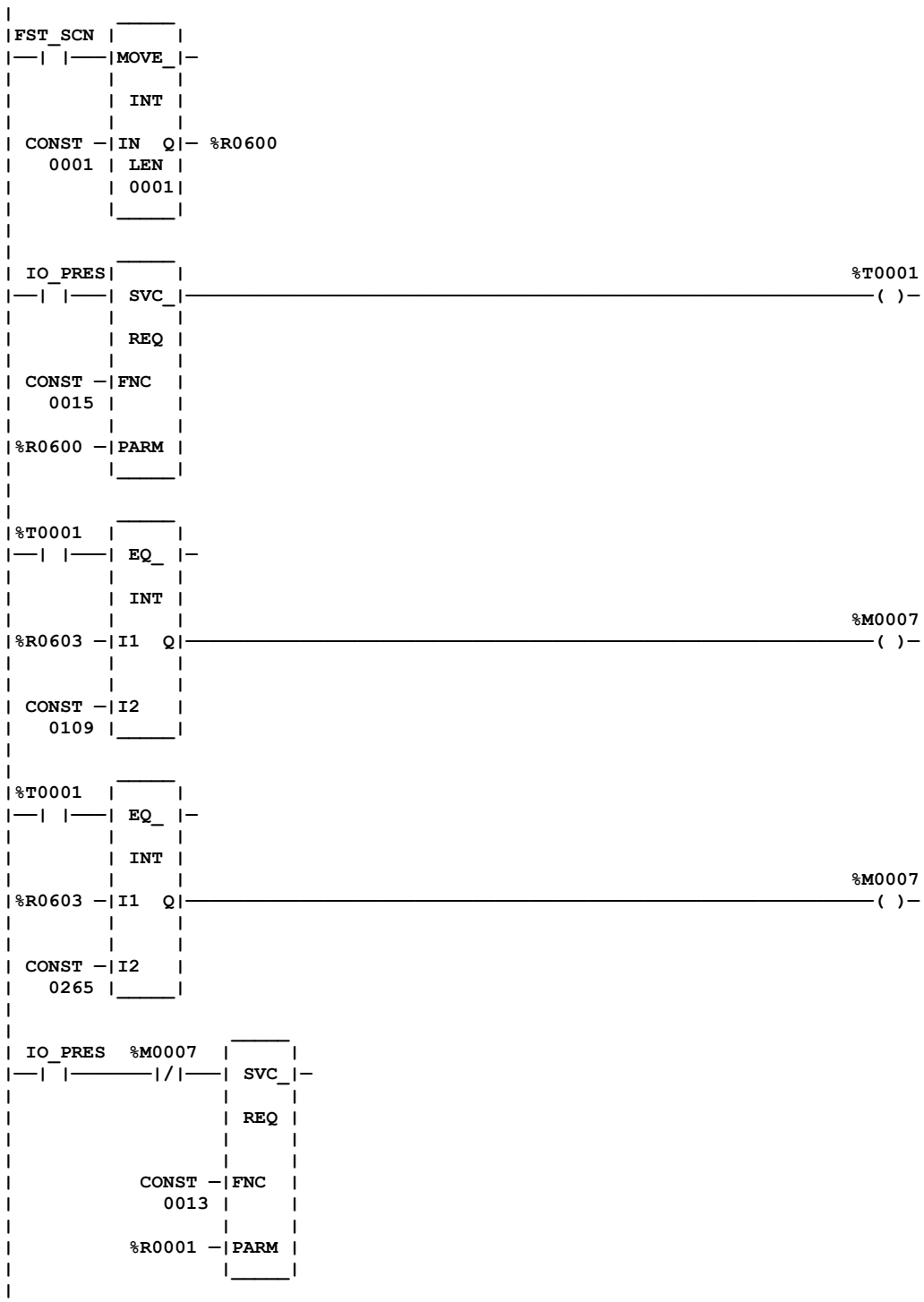
V následujícím příkladu PLC ukončí činnost, když se v modulu I/O vyskytne nějaká chyba s výjimkou, když se chyba vyskytne v modulech v sestavě 0, pozici 9 a v sestavě 1, pozici 9. Pokud se chyba vyskytne na těchto modulech, systém poběží dál. Parametr pro "typ tabulky" se nastaví při prvním cyklu. Když bude kontakt IO_PRES nastavený, indikuje to, že tabulka chyb I/O obsahuje záznam. CPU PLC nastaví normálně rozpojený kontakt v cyklu po tom, co chybová logika provede záznam chyby do tabulky. Pokud se chyby do tabulky umístí během dvou po sobě následujících cyklů, normálně rozpojený kontakt se nastaví po dobu dvou po sobě následujících cyklů.

Příklad používá blok parametrů umístěný na adrese %R0600. Po vykonání funkce SVCREQ bude čtvrté, páté a šesté slovo obsahovat adresu modulu I/O, který měl chybu:

1		%R0600
dlouhý/krátký		%R0601
adresa		%R0602
číslo sestavy	číslo pozice	%R0603
číslo I/O sběrnice	adresa sběrnice	%R0604
adresa bodu		%R0605

data chyby

V programu bloky EQ_INT provádějí porovnávání adresy sestavy/pozici v tabulce s hexadecimálními konstantami. Interní cívka se sepne, když sestava/pozice, kde se vyskytla chyba, bude splňovat výše uvedená kritéria. Pokud %M0007 bude ve stavu ON, jeho normálně sepnutý kontakt se rozpojí a zabrání tak ukončení provozu. Obráceně, pokud %M0007 bude ve stavu OFF, protože se chyba objevila na jiném modulu, normálně sepnutý kontakt bude sepnutý a ukončení provozu se vykoná.


FST_SCN
MOVE_
INT
CONST 0001
IN
LEN 0001
%R0600
IO_PRES
SVC_
REQ
CONST 0015
FNC
%R0600
PARM
%T0001
EQ_
INT
%R0603
I1
CONST 0109
I2
%T0001
EQ_
INT
%R0603
I1
CONST 0265
I2
%T0001
%M0007
%M0007
IO_PRES
%M0007
SVC_
REQ
CONST 0013
FNC
%R0001
PARM

SVCREQ #16: Čtení hodin uplynulého času

Funkce SVCREQ se používá s funkcí číslo 16 k načtení hodnoty hodin uplynulého času systému. Tyto hodiny sledují uplynulý čas v sekundách od zapnutí napájení PLC. Časovač se přetočí dokola přibližně každých 100 let.

Tato funkce má pouze blok výstupních parametrů. Blok parametrů má délku 3 slova.

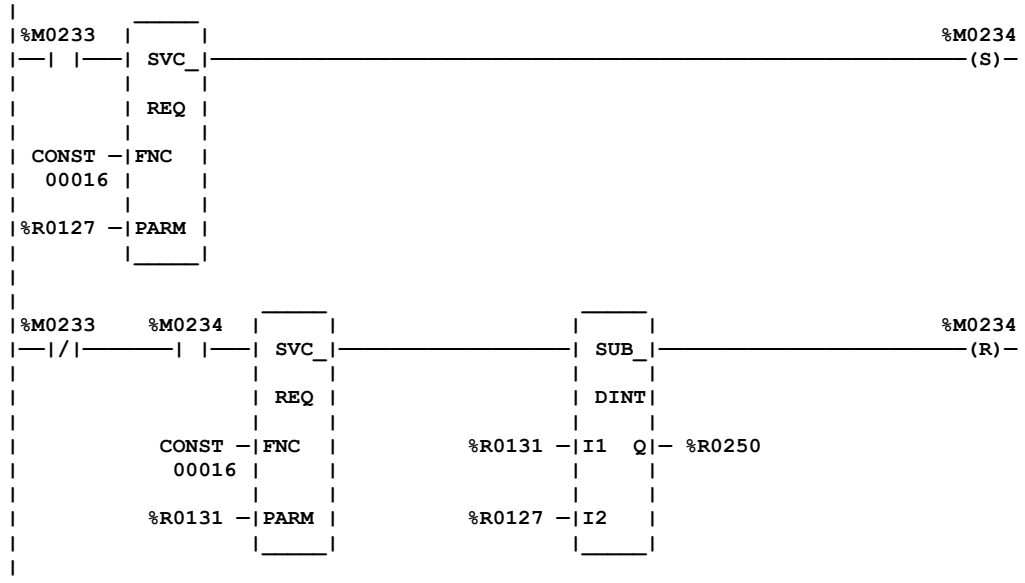
sekundy od zapnutí napájení (nižší řád)	adresa
sekundy od zapnutí napájení (vyšší řád)	adresa + 1
Časové intervaly 100 mikrosekund	adresa + 2

První dvě slova jsou uplynulý čas v sekundách. Poslední slovo je počet časových intervalů 100 mikrosekund v aktuální sekundě.

Příklad

V následujícím příkladu, když interní cívka %M0233 bude ve stavu ON, přečte se hodnota hodin uplynulého času a cívka %M0234 se nastaví. Když bude ve stavu OFF, hodnota se přečte znovu. Pak se vypočítá rozdíl mezi hodnotami a výsledek se uloží do paměťového registru na adrese %R0250.

Blok parametrů pro první čtení je na adrese %R0127; pro druhé čtení na adrese %R0131. Výpočet ignoruje počet 100-mikrosekundových časových intervalů a skutečnost, že typ DINT je ve skutečnosti hodnota se znaménkem. Výpočet bude správný, dokud čas od zapnutí napájení nedosáhne přibližně 50 let.



SVCREQ #18: Čtení stavu přepisu I/O

Funkce SVCREQ #18 se používá k načtení aktuálního stavu přepisu v CPU.

Poznámka

Tuto vlastnost je možno použít ***pouze*** u CPU 331 a vyšších.

U této funkce blok parametrů má délku 1 slovo. Je to pouze blok výstupního parametru.

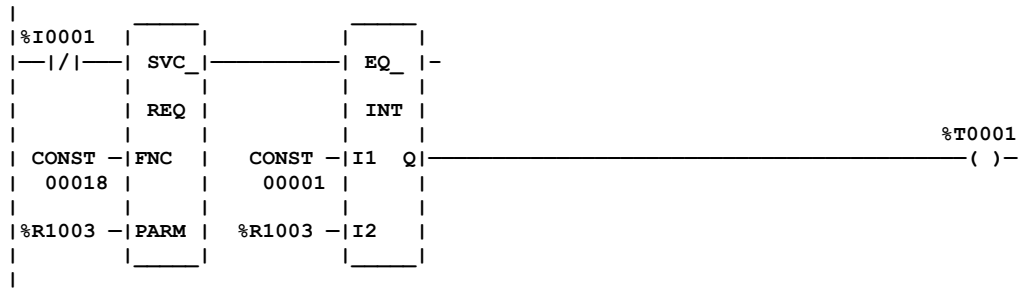
0 = Nejsou nastavené žádné přepisy 1 = Jsou nastavené přepisy	adresa

Poznámka

SVCREQ #18 hlásí pouze přepisy adresy %I a %Q.

Příklad

V následujícím příkladu se přepisy I/O vždy načtou do adresy %R1003. Pokud budou existovat nějaké přepisy, výstup %T0001 přejde do jedničky.



SVCREQ #23: Čtení hlavního kontrolního součtu

Funkce SVCREQ #23 se používá k načtení hlavního kontrolního součtu uživatelského programu a konfigurace. Výstup SVCREQ bude vždy nastavený do stavu ON, pokud funkce budou povolené a výstupní blok informací (viz níže) bude začínat na adrese dané parametrem 3 (PARM) funkce SVCREQ.

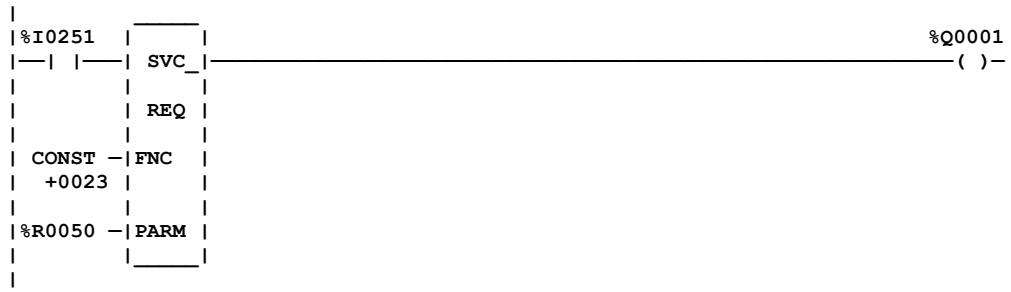
Když **RUN MODE STORE (ukládání za chodu programu)** bude aktivní, kontrolní součet programu nemusí být platný, dokud se uložení nedokončí. Proto na začátku bloku výstupních parametrů jsou dva příznaky, které udávají, kdy bude kontrolní součet programu a konfigurace platný.

U této funkce blok výstupních parametrů má délku 12 slov s následujícím formátem.

Hlavní kontrolní součet programu platí (0 = neplatí, 1 = platí)	adresa
Hlavní kontrolní součet konfigurace platí (0 = neplatí, 1 = platí)	adresa + 1
Počet programových bloků (včetně _MAIN)	adresa + 2
Velikost uživatelského programu v bajtech (data typu DWORD)	adresa + 3
Přídavný kontrolní součet programu	adresa + 5
CRC kontrolní součet programu (data typu DWORD)	adresa + 6
Velikost konfiguračních dat v bajtech	adresa + 8
Přídavný kontrolní součet konfigurace	adresa + 9
Kontrolní součet konfigurace (data typu DWORD)	adresa + 10

Příklad

V následujícím příkladu, když vstup %I0251 bude ve stavu ON, informace hlavního kontrolního součtu se uloží do bloku parametrů a výstupní cívka (%Q0001) se sepne. Blok parametrů je na adrese %R0050.



SVCREQ #26/30: Dotaz na I/O

Funkce SVCREQ #26 (nebo #30 – jsou identické; to znamená, že k vykonání stejné věcí můžete použít kterékoliv číslo) umožňuje dotázat se na aktuální nainstalované moduly a porovnat je s konfigurací sestavy/pozice a generovat alarmy přidání, ztráty a neshody, jako když se vykoná uložení konfigurace. Tato funkce SVCREQ provede záznam chyby v závislosti na typu chyby do tabulky chyb PLC nebo tabulky chyb I/O.

Tato funkce nemá žádný blok parametrů a vždy skrz ní protéká proud.

Poznámka

Doba pro vykonání této funkce SVCREQ závisí na tom, kolik chyb se vyskytne. Proto doba vykonání této funkce SVCREQ bude větší v situacích, kdy se více modulů bude nacházet v chybovém stavu.

Příklad

V následujícím příkladu, když vstup %I0251 bude ve stavu ON, se provede dotaz na aktuální moduly a ty se porovnají s konfigurací sestavy/pozice. Výstup %Q0001 přejde do jedničky po dokončení SVCREQ.

Poznámka

Tento Service Request nelze použít u PLC Micro.

SVCREQ #29: Čtení uplynulého času od vypnutí

Funkce SVCREQ #29 se používá k načtení času uplynulého mezi posledním vypnutím a posledním zapnutím napájení. Výstup SVCREQ bude vždy nastavený do stavu ON a výstupní blok informací (viz níže) bude začínat na adrese dané parametrem 3 (PARM) funkce SVCREQ.

Poznámka

Tuto funkci je možno použít pouze u CPU 311 a vyšších.

Tato funkce má pouze blok výstupních parametrů. Blok parametrů má délku 3 slova.

Doba uplynulá od vypnutí napájení (nižší řád)	adresa
Doba uplynulá od vypnutí napájení (vyšší řád)	adresa + 1
Časové intervaly 100 mikrosekund	adresa + 2

První dvě slova jsou uplynulý čas od vypnutí napájení v sekundách. Poslední slovo je zbytek uplynulého času od vypnutí napájení v časových intervalech 100 mikrosekund (které je vždy 0). Když PLC nebude schopné správně vypočítat dobu uplynulou od vypnutí napájení, doba se nastaví na 0. K tomu může dojít, když PLC bude zapnuté a na HPP bude stisknuto CLR M/T. K tomu může také dojít, když se překročí doba hlídacího časovače před zapnutím.

Příklad

V následujícím příkladu, když vstup %I0251 bude ve stavu ON, se doba uplynulá od vypnutí napájení uloží do bloku parametrů a výstupní cívka (%Q0001) se sepne. Blok parametrů je na adrese %R0050.

SVCREQ #45: Přeskočení dalšího zápisu výstupu a čtení vstupu

(Pozastavení I/O) Funkce SVCREQ #45 se používá k přeskočení dalšího zápisu na výstupy a čtení vstupů. Žádné změny v tabulkách výstupních adres během cyklu, ve kterém se vykonala funkce SVCREQ #45, nebudou mít vliv na fyzické výstupy odpovídajících modulů. Žádné změny fyzických vstupních dat na modulu nebudou mít vliv na odpovídající vstupní adresy během cyklu, po tom, ve kterém se vykonala funkce SVCREQ #45.

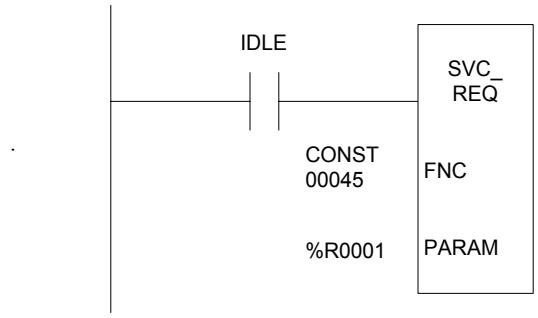
Tato funkce nemá žádný blok parametrů.

Poznámka

Použití SVCREQ #45 nemá vliv na funkční blok DOIO. Pokud bude použitý ve stejném programu logiky jako SVCREQ #45, funkce bude provádět inkrementaci I/O.

Příklad

V následujícím příkladu se přeskočí další zápis na výstupy a čtení vstupů, když “Klidový” kontakt bude propouštět proud.



SVCREQ #46: Přístup ke stavu rychlé vnitřní sběrnice

Funkce SVCREQ #46 se používá k vykonání jedné z následujících funkcí přístupu k rychlé vnitřní sběrnici:

1. Čtení slova přídavných stavových dat z jednoho nebo několika předepsaných inteligentních modulů.
2. Zápis slova přídavných stavových dat z jednoho nebo několika předepsaných inteligentních modulů.
3. Čtení/zápis: Čtení slova přídavných stavových dat z jednoho nebo několika speciálních modulů a zápis datové hodnoty 0 až 15 do stejného modulu, všechno během jediné operace.

Poznámky

Tento Service Request je možno použít pouze s moduly, které ho podporují. V současné době jediný modul, který tuto funkci podporuje, je DSM (Digital Servo Module) verze 312, který v době publikování tohoto manuálu nebyl k dispozici. Jeho uvedení na trh se však plánuje brzy.

Funkční blok COMM_REQ nebo DOIO by se neměl vykonávat se zadanými moduly během stejného cyklu logiky, během kterého se vykonává některá funkce zápisu dat, protože mohou způsobit ztrátu zapisovaných dat.

Během stejného cyklu logiky by se neměly vykonávat dvě funkce, které provádějí zápis (Zápis nebo Čtení/Zápis) do stejného modulu, protože mohou způsobit ztrátu prvních zapisovaných dat.

Tento Service Request má proměnnou délku, jak je popsáno níže. První slovo bloku parametrů určuje, která funkce se má použít, a má následující formát:

1 = Čtení přídavných dat 2 = Zápis přídavných dat 3 = Čtení/zápis přídavných dat	adresa

Čtení přídavných stavových dat (Funkce #1)

Funkce Čtení přídavných stavových dat načte slovo přídavných stavových dat z každého modulu určeného seznamem v bloku parametrů a uloží hodnoty stavových dat do bloku parametrů. Blok parametrů vyžaduje (N + 4) slov adresové paměti, kde N je počet modulů, do kterých se budou zapisovat data.

K interpretaci výstupních hodnot použijte tabulku na následující straně.

Tabulka 12-5. Výstupní hodnoty pro funkci Čtení přídavných dat

Umístění	Pole	Význam
Adresa	Funkce	1 = čtení přídavných stavových dat
Adresa + 1	Chybový kód	Sem se uloží chybový kód, pokud se funkce nevykoná správně z důvodu, že nějaký modul bude chybět, nebude vhodný nebo nebude pracovat. Podrobnosti viz “Chybové kódy” na straně 12-69.
Adresa + 2	Chyba sestavy a pozice	Číslo sestavy a pozice, kde došlo k chybě
Adresa + 3	První sestava a pozice	Číslo sestavy a pozice (ve tvaru RRSS hexadecimálně, kde RR je číslo sestavy a SS je číslo pozice) prvního modulu, ze kterého se budou číst data
Adresa + 4	Čtení dat z prvního modulu	Sem se uloží data načtená z prvního modulu
Adresa + 5	Druhá sestava s pozice	Číslo sestavy a pozice (ve tvaru RRSS hexadecimálně, kde RR je číslo sestavy a SS je číslo pozice) druhého modulu, ze kterého se budou číst data
Adresa + 6	Čtení dat z druhého modulu	Sem se uloží data načtená z druhého modulu
Adresa + (I * 2) + 1	I-tá sestava a pozice	Číslo sestavy a pozice (ve tvaru RRSS hexadecimálně, kde RR je číslo sestavy a SS je číslo pozice) I-tého modulu, ze kterého se budou číst data
Adresa + (I * 2) + 2	Čtení dat z I-tého modulu	Sem se uloží data načtená z I-tého modulu
Adresa + (N * 2) + 1	Poslední sestava a pozice	Číslo sestavy a pozice (ve tvaru RRSS hexadecimálně, kde RR je číslo sestavy a SS je číslo pozice) posledního modulu, ze kterého se budou číst data
Adresa + (N * 2) + 2	Čtení dat z posledního modulu	Sem se uloží data načtená z posledního modulu
Adresa + (N * 2) + 3	Konec indikátoru seznamu	Nula v tomto slově označuje konec seznamu modulů.

Zápis dat (Funkce #2)

Funkce zápisu dat zapíše datovou hodnotu 0 až 15 z bloku parametrů do jednoho nebo více modulů určených seznamem v bloku parametrů. Blok parametrů vyžaduje (N + 4) slov adresové paměti, kde N je počet modulů, do kterých se budou zapisovat data.

Tabulka 12-6. Výstupní hodnoty pro funkci Zápis dat

Umístění	Pole	Význam
Adresa	Funkce	2 = Zápis dat
Adresa + 1	Chybový kód	Sem se uloží chybový kód, pokud se funkce nevykoná správně z důvodu, že nějaký modul bude chybět, nebude vhodný nebo nebude pracovat. Pokud se funkce vykoná, ale na některý z modulů data nepřijdou správně, nenastaví se žádný chybový kód. Podrobnosti viz "Chybové kódy" na straně 12-69.
Adresa + 2	Chyba sestavy a pozice	Číslo sestavy a pozice, kde došlo k chybě
Adresa + 3	První sestava a pozice	Číslo sestavy a pozice (ve tvaru RRSS hexadecimálně, kde RR je číslo sestavy a SS je číslo pozice) prvního modulu, do kterého se budou posílat data
Adresa + 4	Zápis dat pro první modul	Tato datová hodnota se zapíše do prvního modulu
Adresa + 5	Druhá sestava s pozice	Číslo sestavy a pozice (ve tvaru RRSS hexadecimálně, kde RR je číslo sestavy a SS je číslo pozice) druhého modulu, do kterého se budou posílat data
Adresa + 6	Zápis dat pro druhý modul	Tato datová hodnota se zapíše do druhého modulu
Adresa + (I * 1) + 1	I-tá sestava a pozice	Číslo sestavy a pozice (ve tvaru RRSS hexadecimálně, kde RR je číslo sestavy a SS je číslo pozice) I-tého modulu, do kterého se budou posílat data
Adresa + (I * 2) + 2	Zápis dat pro I-tý modul	Tato datová hodnota se zapíše do I-tého modulu
Adresa + (N * 2) + 1	Poslední sestava a pozice	Číslo sestavy a pozice (ve tvaru RRSS hexadecimálně, kde RR je číslo sestavy a SS je číslo pozice) posledního modulu, do kterého se budou posílat data
Adresa + (N * 2) + 2	Zápis dat pro poslední modul	Tato datová hodnota se zapíše do posledního modulu
Adresa + (N * 2) + 3	Konec indikátoru seznamu	Nula v tomto slově označuje konec seznamu modulů.

Čtení/Zápis dat (Funkce #3)

Funkce čtení/zápis načte slovo přídavných stavových dat z modulu určeného seznamem v bloku parametrů a pak uloží datovou hodnotu 0 až 15 z bloku parametrů do tohoto modulu. Tento proces čtení a zápisu se opakuje pro všechny moduly ze seznamu v bloku parametrů. Blok parametrů vyžaduje (N * 3) + 3 slov adresové paměti, kde N je počet modulů, se kterými se provádí výměna dat.

Tabulka 12-7. Výstupní hodnoty pro funkci Čtení/Zápis dat

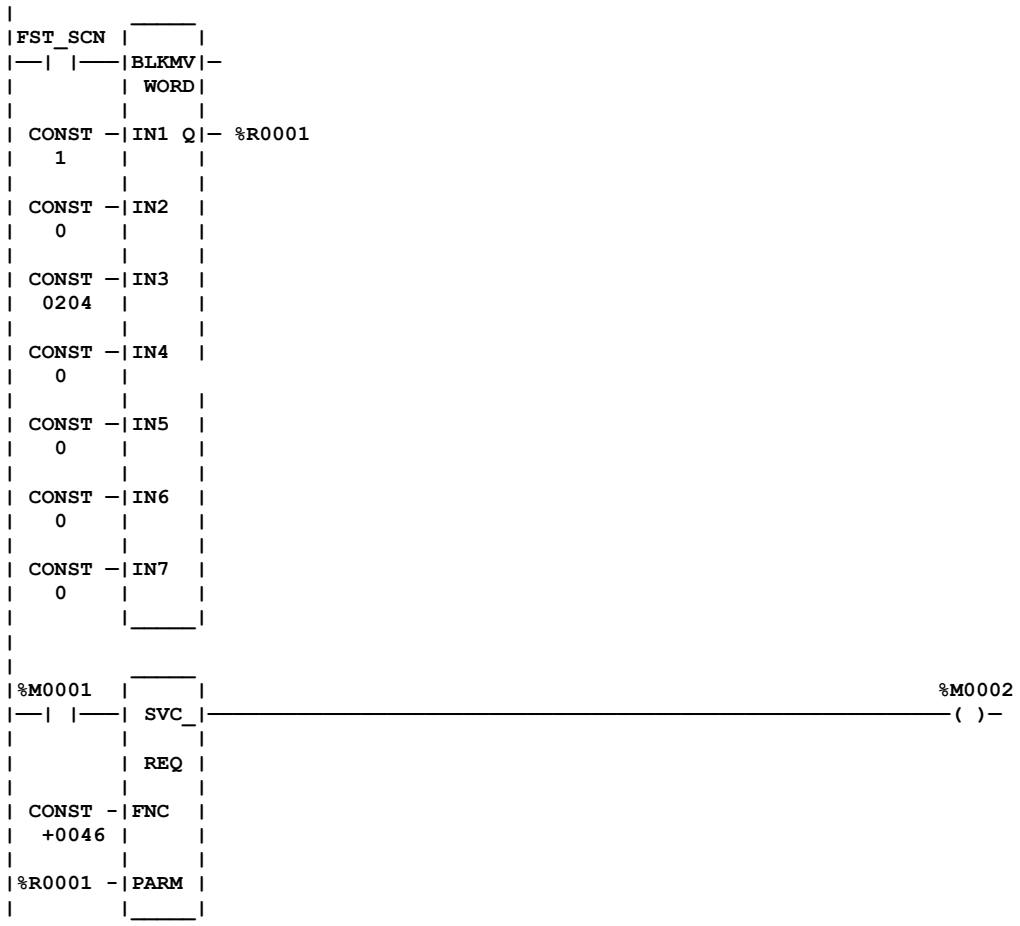
Umístění	Pole	Význam
Adresa	Funkce	3 = čtení/zápis
Adresa + 1	Chybový kód	Sem se uloží chybový kód, pokud se funkce nevykoná správně z důvodu, že nějaký modul bude chybět, nebude vhodný nebo nebude pracovat. Pokud se funkce vykoná, ale na některý z modulů data nepřijdou správně, nenastaví se žádný chybový kód. Podrobnosti viz "Chybové kódy" na straně 12-69.
Adresa + 2	Chyba sestavy a pozice	Číslo sestavy a pozice, kde došlo k chybě
Adresa + 3	První sestava a pozice	Číslo sestavy a pozice (ve tvaru RRSS hexadecimálně, kde RR je číslo sestavy a SS je číslo pozice) prvního modulu, se kterými se provádí výměna dat.
Adresa + 4	Čtení dat z prvního modulu	Sem se uloží data načtená z prvního modulu
Adresa + 5	Zápis dat pro první modul	Tato datová hodnota se zapíše do prvního modulu
Adresa + 6	Druhá sestava s pozice	Číslo sestavy a pozice (ve tvaru RRSS hexadecimálně, kde RR je číslo sestavy a SS je číslo pozice) druhého modulu, se kterými se provádí výměna dat.
Adresa + 7	Čtení dat z druhého modulu	Sem se uloží data načtená z druhého modulu
Adresa + 8	Zápis dat pro druhý modul	Tato datová hodnota se zapíše do druhého modulu
Adresa + ((I-1) * 3) + 3	I-tá sestava a pozice	Číslo sestavy a pozice (ve tvaru RRSS hexadecimálně, kde RR je číslo sestavy a SS je číslo pozice) I-tého modulu, se kterými se provádí výměna dat.
Adresa + ((I-1) * 3) + 4	Čtení dat z I-tého modulu	Sem se uloží data načtená z I-tého modulu
Adresa + ((I-1) * 3) + 5	Zápis dat pro I-tý modul	Tato datová hodnota se zapíše do I-tého modulu
Adresa + ((N-1) * 3) + 3	Poslední sestava a pozice	Číslo sestavy a pozice (ve tvaru RRSS hexadecimálně, kde RR je číslo sestavy a SS je číslo pozice) posledního modulu, se kterými se provádí výměna dat.
Adresa + ((N-1) * 3) + 4	Čtení dat z posledního modulu	Sem se uloží data načtená z posledního modulu
Adresa + ((N-1) * 3) + 5	Zápis dat pro poslední modul	Tato datová hodnota se zapíše do posledního modulu
Adresa + (N * 3) + 3	Konec indikátoru seznamu	Nula v tomto slově označuje konec seznamu modulů.

Chybové kódy

Hodnota	Popis
1	Úspěch – funkce se vykonala normálně.
-1	Modul není v předepsaném pozici.
-2	Nesprávný modul – modul v předepsaném pozici není inteligentní modul nebo nepodporuje tuto funkci.
-3	Modul nepracuje - modul v předepsaném pozici nekomunikuje správně s CPU.
-4	Chyba parity načtených dat – během operace čtení z přídavné vzdálené sestavy se vyskytla chyba parity.
-5	V povelovém bloku byla zadaná neplatná funkce.

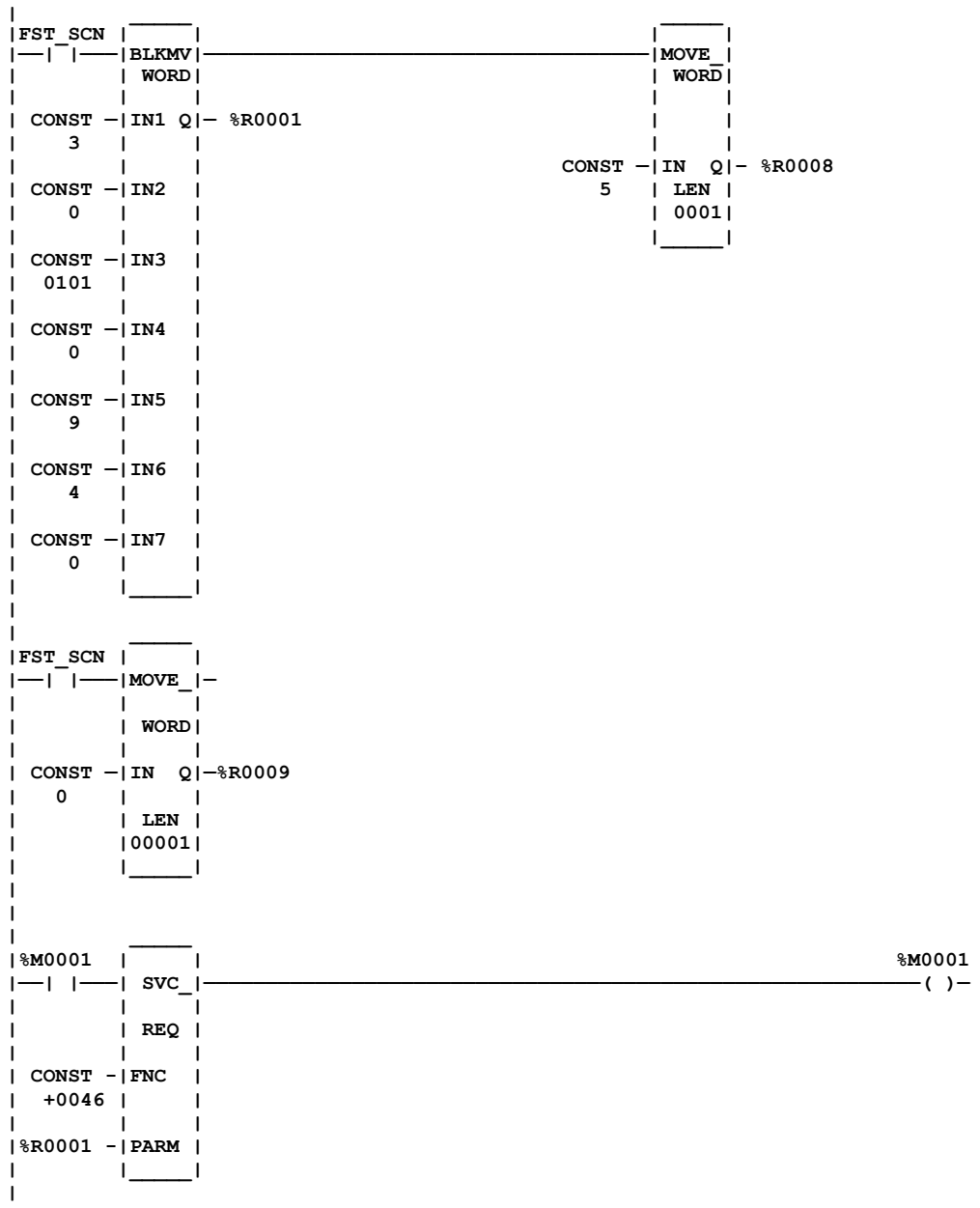
Příklad 1

Následující příklad ukazuje Čtení jednoho modulu v sestavě 2, pozice 4. IN4 a IN5 musí být nastavené na nulu (0). IN6 a IN7 v tomto příkladu nejsou důležité. Pokud se funkce dokončí úspěšně, data budou v %R0004.



Příklad 2

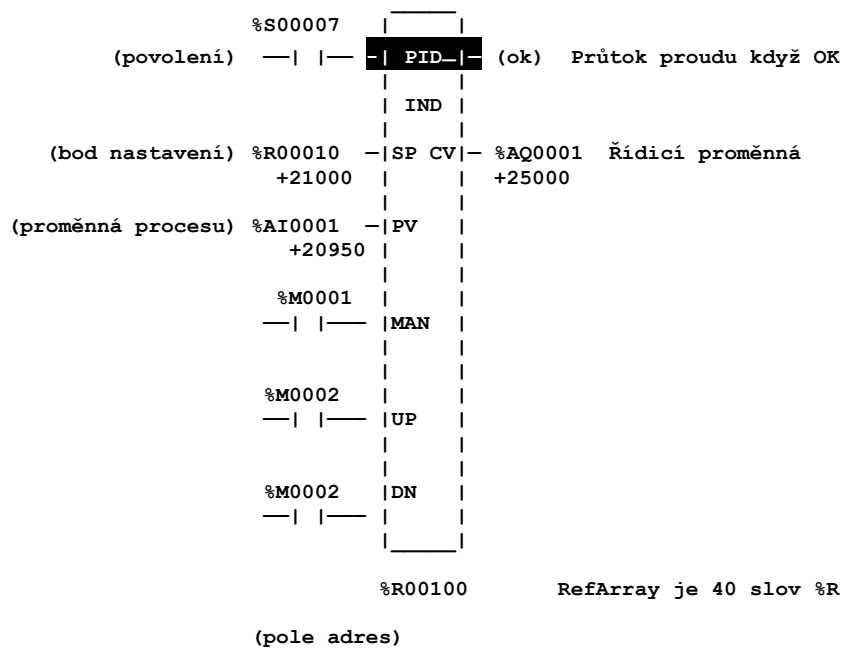
Tento příklad provádí načtení přídavných stavových dat z modulu v sestavě 0, pozice 4 a z modulu v sestavě 1, pozice 1. Zapíše 5 do prvního modulu a 9 do druhého modulu. Všimněte si, že moduly nemusí být uvedené v pořadí podle čísla pozice. Data načtená z modulu v sestavě 0, pozice 4 se uloží do %R0007. Data načtená z modulu v sestavě 1, pozice 1 se uloží do %R0004.



PID

Proporcionální plus integrální plus derivační (PID) řídicí funkce je nejlepší známý univerzální algoritmus pro řídicí procesy v uzavřené smyčce. PID řídicí blok PLC Series 90 provádí porovnávání Proměnné procesu (PV) ve zpětné vazbě s požadovaným Bodem nastavení procesu (SP) a na základě odchylky provede aktualizaci výstupu Řídicí hodnoty (CV).

Blok používá zisk PID smyčky a ostatní parametry uložené v poli čtyřiceti 16-bitových slov (výklad viz strana 12-73) k řešení PID algoritmu v požadovaném časovém intervalu. Pro kompatibilitu s 16-bitovými analogovými proměnnými procesu jsou všechny parametry 16-bitová celočíselná slova. To umožňuje použít paměť %AI pro vstupní Proměnné procesu a paměť %AQ použít pro výstupní Řídicí proměnné. Níže uvedený příklad uvádí typické vstupy.



Protože 16 čísel s celočíselnou hodnotou je ve stanoveném měřítku, mnoho parametrů musí být definováno v počtech nebo jednotkách parametru PV nebo počtech nebo jednotkách parametru CV. Například vstup SP musí mít měřítko stanovené ve stejném rozsahu jako PV, protože PID blok vypočítává odchylku z rozdílu těchto dvou vstupů. Hodnoty PV a CV mohou být –32000 nebo 0 až 32000 v souladu s analogovou stupnicí nebo od 0 do 10000 a zobrazovat proměnné jako 0.00% až 100.00%. Hodnoty PV a CV nemusí být ve stejném měřítku, kdy pak koeficient měřítka bude součástí zisku PID.

Poznámka

PID se nevykoná častěji než jednou za každých 10 milisekund. Pokud funkci nastavíte tak, aby se vykonával každý cyklus a cyklus bude kratší než 10 milisekund, mohl by se tím změnit výsledek. V takovém případě se funkce PID nespustí, dokud se neobjeví dostatečný počet cyklů, aby celkový uplynulý čas byl 10 milisekund. Pokud například doba cyklu bude 9 milisekund, funkce PID se vykoná každý druhý cyklus a při každém vykonání uplyne doba 18 milisekund.

Parametry

Parametr	Popis
povolení	Pokud kontakt funkci povolí, PID funkce se vykoná.
SP	SP je Bod nastavení řídicí smyčky nebo procesu. Po nastavení v jednotkách PV provede PID úpravu výstupu CV tak, aby PV souhlasilo s SP (nulová odchylka).
PV	Proměnná procesu přivedená z řízeného procesu, často vstup %AI.
MAN	Když bude nastavený na 1 (přes kontakt), blok PID bude v **RUČNÍM** režimu. Pokud tento parametr nebude nastavený (0), blok PID bude v automatickém režimu.
UP	Pokud bude nastavený do jedničky společně se vstupem MAN, provede inkrementaci CV o 1 CV na jedno řešení. *
DN	Pokud bude nastavený do jedničky společně se vstupem MAN, provede dekrementaci CV o jedničku na jedno řešení. *
Adresa RefArray	Adresa je umístění informace řídicího bloku PID (uživatelské a interní parametry). Používá 40 slov %R, která nelze sdílet.
ok	Výstup ok bude v jedničce, když se funkce vykoná bez chyby. Pokud bude existovat chyba, bude v nule.
CV	CV je výstup řídicí proměnné do procesu, často analogový výstup %AQ.

*Provede inkrementaci (parametr UP) nebo dekrementaci (parametr DN) o 1 na jeden přístup funkce PID.

Platné typy paměti

Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
SP		•	•	•	•		•	•	•	•	•	
PV		•	•	•	•		•	•	•	•		
MAN	•											
UP	•											
DN	•											
adresa								•				
ok	•											•
CV		•	•	•	•		•	•	•	•		

• Platná adresa nebo místo, kde funkcí může téct proud.

Blok parametrů PID

Kromě 2 vstupních slov a 3 ručních řídicích kontaktů blok PID používá parametry v poli RefArray. Tyto parametry musí být nastavené před vyvoláním bloku. Ostatní parametry používá PLC a nejsou konfigurovatelné. %Ref uvedená v následující tabulce je stejná adresa jako RefArray na konci bloku PID. Číslo za znaménkem plus je offset v poli. Pokud například RefArray bude začínat na %R100, %R113 bude obsahovat Ruční povel používaný k nastavení Řídicí proměnné a integrátoru v Ručním režimu.

Tabulka 12-8. Přehled parametrů PID

Registr	Parametr	Jednotky dolního bitu	Rozsah hodnot
%Ref+0000	Číslo smyčky	Celé číslo	0 až 255 (pouze pro uživatelskou displej)
%Ref+0001	Algoritmus	N/A; nastavuje a spravuje PLC	Nenastavuje se
%Ref+0002	Perioda vzorkování	10 milisekund	0 (každý cyklus) až 65535 (10.9 minut). Pro PLC 90-30 použijte alespoň 10 (viz poznámka na straně 12-71).
%Ref+0003	Pásmo necitlivosti +	Jednotky PV	0 až 32000 (nikdy záporné)
%Ref+0004	Pásmo necitlivosti -	Jednotky PV	–32000 až 0 (nikdy kladné)
%Ref+0005	Proporcionální zisk – Kp	0.01 CV%/PV%	0,327 až 67,28
%Ref+0006	Derivační zisk – Kd	0.01 sekundy	0 až 327.67 sekundy
%Ref+0007	Integrační zisk – Ki	Opakování/1000 sekund	0 až 32.767 opakování/sekundu
%Ref+0008	Posunutí CV/Výstupní Offset	Jednotky CV	–32000 až 32000 (přičíst k výstupu integrátoru)
%Ref+0009	Horní omezovač	Jednotky CV	–32000 až 32000(>%Ref+10) výstupní limit
%Ref+0010	Dolní Omezovač	Jednotky CV	–32000 až 32000(<%Ref+09) výstupní limit
%Ref+0011	Minimální doba odezvy	Sekunda/plný zdvih	0 (žádná) až 32000 sekund k přesunutí 32000 CV
%Ref+0012	Konfigurační slovo	Používá se dolních 5 bitů	Bit 0 až 2 pro +/– odchylky, výstupní polaritu, derivaci
%Ref+0013	Ruční povel	Jednotky CV	Sleduje CV v Auto nebo nastaví CV v Ručním
%Ref+0014	Řídicí slovo	***Pokud*** bit 1 nebude nastavený, spravuje PLC.	Pokud nebude nastaveno jinak, spravuje PLC: pokud 1, dolní bit nastaví Override (viz popis v tabulce “Detaily parametrů PID” na straně 12-76)
%Ref+0015	Interní SP	N/A; nastavuje a spravuje PLC	Nenastavuje se
%Ref+0016	Interní CV	N/A; nastavuje a spravuje PLC	Nenastavuje se
%Ref+0017	Interní PV	N/A; nastavuje a spravuje PLC	Nenastavuje se
%Ref+0018	Výstup	N/A; nastavuje a spravuje PLC	Nenastavuje se

Tabulka 12-8. Přehled parametrů PID – pokračování

Registr	Parametr	Jednotky dolního bitu	Rozsah hodnot
%Ref+0019	Paměť diferenciálních hodnot	N/A; nastavuje a spravuje PLC	Nenastavuje se
%Ref+0020 a %Ref+0021	Paměť integrálních hodnot	N/A; nastavuje a spravuje PLC	Nenastavuje se
%Ref+0022	Paměť odezvových hodnot	N/A; nastavuje a spravuje PLC	Nenastavuje se
%Ref+0023	Hodiny	N/A; nastavuje a spravuje PLC	Nenastavuje se
%Ref+0024			
%Ref+0025	(čas posledního vykonání)		
%Ref+0026	Uložení zbytku Y	N/A; nastavuje a spravuje PLC	Nenastavuje se
%Ref+0027	Dolní rozsah pro SP, PV	Jednotky PV	–32000 až 32000 (>%Ref+28) pro zobrazení
%Ref+0028	Horní rozsah pro SP, PV	Jednotky PV	–32000 až 32000 (<%Ref+27) pro zobrazení
%Ref+0029 +• %Ref+0034	Vyhrazeno pro interní použití	N/A	Nenastavuje se
%Ref+0035 • %Ref+0039	Vyhrazeno pro externí použití	N/A	Nenastavuje se

Pole RefArray u PLC 90-30 musí být registry %R. Všimněte si, že každé volání bloku PID musí používat jiné 40-slovní pole, i když všech 13 uživatelských parametrů bude stejných, protože ostatní slova v poli se používají pro interní uložení dat PID. Přesvědčte se, že pole nesahá za konec paměti.

Chcete-li nakonfigurovat operační parametry, zvolte funkci PID a stisknutím tlačítka **F10** proveďte zoom do obrazovky zobrazující Uživatelské parametry; pak pomocí tlačítek se šipkami zvolte pole a zapište požadované hodnoty. Pro většinu výchozích hodnot můžete použít 0 s výjimkou Horního omezovače CV, který musí být větší než Dolní omezovač CV, aby blok PID mohl pracovat. Všimněte si že, blok PID **nepropustí** proud, pokud v Uživatelských parametrech bude chyba, proto během úpravy dat provádějte monitorování s pomocí dočasné cívky.

Jakmile budou zvolené vhodné hodnoty PID, je nutno je definovat jako konstanty v BLKMOV, aby bylo možno je v případě potřeby použít k novému načtení uživatelských parametrů PID.

Činnost instrukce PID

Normální automatická činnost vyvolá PID blok každý cyklus, když skrz vstupní kontakt Povolení poteče proud a skrz Manual proud nepoteče. Blok porovná hodiny aktuálního uplynulého času PLC s posledním časem řešení PID uloženým v interním RefArray. Pokud rozdíl bude větší než perioda vzorkování definovaná ve třetím slově (%Ref+2) pole RefArray, algoritmus PID se vyřeší s použitím rozdílu a provede se aktualizace posledního času řešení a výstupu Řídicí proměnné. V automatickém režimu se výstup Řídicí proměnné zapíše do parametru Ruční povel %Ref+13.

Pokud proud poteče do vstupních kontaktů Povolení a Ruční, blok PID přejde do Ručního režimu a výstup Řídicí proměnné se nastaví z parametru Ruční povel %Ref+13. Pokud proud poteče buď vstupem UP nebo DN, slovo Ručního povelu inkrementuje nebo dekrementuje CV o jedničku při každém řešení PID. Pro rychlejší ruční změny výstupu Řídicí proměnné je také možno přičíst nebo odečíst libovolnou hodnotu CV přímo k/od slova Ručního povelu.

Blok PID používá parametry Horního a Dolního omezovače CV k omezení výstupu CV. Pokud bude definovaná minimální doba odezvy, použije se k omezení velikosti změny výstupu CV. Pokud dojde k překročení buď amplitudy CV nebo velikosti omezení, hodnota uložená v integrátoru se změní tak, aby CV bylo na mezi. Tato vlastnost, která zabraňuje resetu při přetočení, (definovaná na stránce 12-78) znamená, že i když se odchylka bude po delší dobu snažit dostat CV nad (nebo pod) omezovače, výstup CV se od omezovače odpoutá, jakmile hodnota odchylky změní znaménko.

Tato činnost s Ručním povelem sledujícím CV v Automatickém režimu a nastavením CV v Ručním režimu zajistí přechod bez skoků mezi Automatickým a Ručním režimem. Horní a Dolní omezovače CV a Minimální doba přejezdu se stále přivádějí na výstup CV v Ručním režimu a interní hodnota uložená v integrátoru se aktualizuje. To znamená, že když budete krokovat Ruční povel v Ručním režimu, výstup CV se nebude měnit rychleji, než je definováno Minimální dobou odezvy, a nedostane se nad nebo pod hranice Horního a Dolního omezovače CV.

Poznámka

Konkrétní funkce PID by se během jednoho cyklu neměla vyvolat více než jednou.

Následující tabulka uvádí více podrobností o parametrech popisovaných krátce v tabulce 12-3. Číslo v závorkách po názvu parametru je offset v poli RefArray.

Tabulka 12-9. Detaily parametrů PID

Datový údaj	Popis
Číslo smyčky(00)	Je to volitelný parametr používaný k identifikaci PID bloku. Je to celé číslo bez znaménka, které umožňuje společnou identifikaci v PLC s číslem smyčky definovaným zařízením rozhraní obsluhy. Číslo smyčky se zobrazuje pod adresou bloku, když se logika monitoruje ze softwaru Logicmaster 90-30/20/Micro.
Algoritmus(01)	Celé číslo bez znaménka, které je nastaveno v PLC jako identifikace, který algoritmus funkční blok používá. Algoritmus ISA je definovaný jako algoritmus 1 a nezávislý algoritmus je identifikovaný jako algoritmus 2.
Perioda vzorkování (02)	Nejkratší doba v inkrementech 10 milisekund mezi řešeními PID algoritmu. Například pro periodu vzorkování 100 milisekund se použije 10. Pokud bude 0, algoritmus se bude řešit vždy, když se blok vyvolá (viz odstavec níže o plánování bloku PID). PID algoritmus se řeší pouze, když aktuální uplynulý čas PLC se bude rovnat nebo bude větší než čas posledního řešení PID plus tato perioda vzorkování. Nezapomeňte, že 90-30 nebude používat dobu řešení kratší než 10 milisekund (viz poznámka na straně 12-71); takže v případě kratších časů se cykly přeskočí. Tato funkce kompenzuje aktuální uplynulý čas od posledního vykonání v rozmezí 100 mikrosekund. Pokud tato hodnota bude nastavena na 0, funkce se vykoná vždy, když bude povolena; je však omezena na minimum 10 milisekund, jak bylo uvedeno výše.
Pásmo necitlivosti (+/-) (03/04)	Celočíselné hodnoty definující horní (+) a dolní (–) mez pásma necitlivosti v jednotkách PV. Pokud se nevyžaduje žádné pásmo necitlivosti, tyto hodnoty musí být 0. Pokud Odchylka PID (SP – PV) nebo (PV – SP) bude vyšší než hodnota (–) a nižší než hodnota (+), výpočty PID se provedou s odchylkou nastavenou na 0. Pokud odchylky budou nenulové, hodnota (+) musí být větší než 0 a hodnota (–) musí být menší než 0, jinak blok PID nebude fungovat. *Musí zůstat na 0, dokud se neprovede nastavení nebo optimalizace zisku PID smyčky*. Potom může být nutné přičíst pásmo necitlivosti, aby nedocházelo k malým změnám na výstupu CV v důsledku malých změn odchylky, třeba pro eliminaci mechanického opotřebení.
Proporcionální zisk – Kp (05)	Toto celé číslo, nazývané Zisk regulátoru Kc, ve verzi ISA určuje změnu CV (v jednotkách CV) při změně hodnoty odchylky o 100 jednotek PV. Zobrazuje se jako 0.00 %/% s přesností na 2 desetinná místa. Například Kp zapsané jako 450 se zobrazí jako 4.50 a bude se podílet na výstupu PID hodnotou Kp*Odchylka/100 nebo 450*Odchylka/100. Kp je v zásadě první zisk nastavený při seřizování smyčky PID.
Derivační zisk – Kd (06)	Toto celé číslo určuje změnu CV (v jednotkách CV), pokud se Odchylka nebo PV (v jednotkách PV) změní o jedničku každých 10 milisekund. Je zapsaný jako čas s nejnižším bitem indikujícím 10 milisekund a zobrazuje se jako 0.00 sekund s přesností na 2 desetinná místa. Například Kd s hodnotou 120 se zobrazí jako 1.20 sekundy a bude se podílet na PID výstupu hodnotou Kd*změna odchylky/změna času. Když se odchylka změní o 4 jednotky PV během každých 30 milisekund, podíl na PID výstupu bude 120*4/3. Kd je možno použít ke zrychlení pomalé odezvy smyčky, ale je velmi citlivé na vstupní šum PV.
Integrační zisk - Ki (07)	Toto celé číslo určuje změnu CV (v jednotkách CV), pokud Odchylka bude mít konstantní hodnotu 1 (v jednotkách PV). Zobrazuje se jako 0.000 Opakování/sekundu s přesností na 3 desetinná místa. Například Ki s hodnotou 1400 se zobrazí jako 1.400 Opakování/sekundu a bude se podílet na PID výstupu hodnotou Ki * Odchylka*dt. Když se odchylka změní o 20 jednotek PV a doba cyklu PLC bude 50 milisekund (perioda vzorkování 0), podíl na PID výstupu bude 1400* 20 * 50/1000. Ki je obvykle druhý zisk, který se nastavuje po Kp.
Posunutí CV/Výstupní Offset (08)	Celočíselná hodnota v jednotkách CV přičtená k výstupu PID před omezením rychlosti a amplitudy. Může se použít k nastavení nenulové hodnoty CV, když se používá pouze proporcionální zisk Kp, nebo pro dopředné řízení výstupu této PID smyčky z jiné řídicí smyčky.

Tabulka 12-9. Detaily parametrů PID – pokračování

Datový údaj	Popis
Horní a Dolní omezovače CV (09/10)	Celočíselná hodnota v jednotkách CV, která definuje nejvyšší a nejnižší hodnotu CV. Tyto hodnoty se vyžadují a Horní omezovač musí mít kladnější hodnotu než Dolní omezovač, jinak PID blok nebude pracovat. Ty se obvykle používají k definování mezí na základě fyzických omezení pro výstup CV. Také se používají ke změně měřítka zobrazení Sloupcového diagramu CV pro displej LM90 nebo ADS PID. Blok zabraňuje resetu při přetočení pro změnu hodnoty integrátoru, když se dosáhne hodnoty omezovače CV.
Minimální doba přejezdu (11)	Kladná hodnota k definování minimálního počtu sekund pro výstup CV k posunutí z 0 do plného zdvihu 100% nebo 32000 jednotek CV. Je to inverzní hodnota meze, jak rychle se může výstup CV změnit. Pokud hodnota bude kladná, CV se nemůže změnit více než o 32000 jednotek CV krát rozdíl času (v sekundách) děleno Minimální dobou přejezdu. Pokud například Perioda vzorkování byla 2.5 sekundy a Minimální doba přejezdu bude 500 sekund, CV se nemůže změnit o více než 32000*2.5/500 neboli 160 jednotek CV během jednoho řešení PID. Stejně jako u CV omezovačů, i zde je vlastnost proti přetočení, která upraví hodnotu integrátoru, když dojde k překročení hodnoty meze CV. Pokud Minimální doba přejezdu bude 0, žádná hodnota meze CV nebude. Když budete provádět optimalizaci nebo seřizování zisků PID smyčky, přesvědčte se, že Minimální doba přejezdu je nastavená na 0.
Konfigurační slovo	Dolních 5 bitů tohoto slova se používá k úpravě tří standardních nastavení PID. Ostatní bity musí být nastavené na 0. Dolní bit nastavte na 1, chcete-li změnit standardní hodnotu odchylky PID z normální (SP - PV) na (PV - SP) a při tom obrátit znaménko hodnoty zpětné vazby. To je pro opačně působící členy, kde CV musí klesat, když PV roste. Druhý bit nastavte na 1, chcete-li obrátit polaritu výstupu tak, že CV bude záporná hodnota výstupu PID místo normální kladné hodnoty. Čtvrtý bit nastavte na 1, chcete-li změnit derivační funkci z používání normální změny hodnoty odchylky na změnu hodnoty zpětné vazby PV. Dolních 5 bitů Konfiguračního slova má následující detailní popis: **Bit 0** = Hodnota odchylky. Když tento bit bude nastavený na 0, hodnota odchylky bude SP — PV. Když tento bit bude nastavený na 1, hodnota odchylky bude PV — SP. **Bit 1** = Polarita výstupu. Když je tento bit nastaven na 0, výstup CV bude představovat výstup výpočtu PID. Když je tento bit nastaven na 1, výstup CV bude představovat negaci výstupu výpočtu PID. **Bit 2** = Derivační funkce na PV. Když tento bit bude nastavený na 0, derivační funkce se použije na hodnotu odchylky. Když tento bit bude nastavený na 1, derivační funkce se použije na hodnotu PV. Všechny zbývající bity musí být nastavené na nulu. **Bit 3** = Funkce pásma necitlivosti. Když bit pásma necitlivosti bude nastavený na nulu, pak není zvolena žádná funkce pásma necitlivosti. Pokud odchylka bude ležet uvnitř mezí pásma necitlivosti, pak odchylka bude tvrdě nastavena na nulu. V opačném případě odchylka mezemi pásma necitlivosti nebude ovlivněna. Když bit pásma necitlivosti bude nastavený na jedničku, pak není je zvolena funkce pásma necitlivosti. Pokud odchylka bude ležet uvnitř mezí pásma necitlivosti, pak odchylka bude tvrdě nastavena na nulu. Pokud však odchylka bude mimo pásmo necitlivosti, pak se odchylka sníží o meze pásma necitlivosti (odchylka = odchylka – mez odchylka pásma necitlivosti). **Bit 4** = Funkce zabraňující resetu při přetočení. Když tento bit bude nastavený na nulu, funkce zabraňující resetu při přetočení použije zpětný výpočet při resetu. Když výstup bude omezený, nahradí se akumulovaná hodnota zbytku Y (definovaná na straně 12-78) jakoukoliv hodnotou potřebnou k vytvoření přesně omezeného výstupu. Když tento bit bude nastavený na jedničku, nahradí se akumulovaná hodnota Y hodnotou Y na začátku výpočtu. Tímto způsobem se předem omezená hodnota Y přidrží tak dlouho, dokud nebude omezený výstup. **POZNÁMKA:** Bit funkce zabraňující resetu při přetočení je možno použít pouze u CPU 90-30 verze 6.50 nebo pozdější. Pamatujte, že bity se nastavují jako mocniny 2. Chcete-li například nastavit Konfigurační slovo na 0 pro výchozí konfiguraci PID, musíte přičíst 1 a změnit hodnotu Odchylky z SP–PV na PV–SP nebo přičíst 2 a změnit Polaritu výstupu z CV = výstup PID na CV = – výstup PID nebo přičíst 4 a změnit Derivační funkci z hodnoty změny Odchylky na hodnotu změny PV atd.

Tabulka 12-9. Detaily parametrů PID – pokračování

<table>
<tr><th>Datový údaj</th><th>Popis</th></tr>
<tr><td>Ruční povel (13)</td><td>Toto je celočíselná hodnota nastavená na aktuální výstup CV, když blok PID bude v Automatickém režimu. Když blok bude přepnutý do Ručního režimu, tato hodnota se použije k nastavení výstupu CV a interní hodnoty integrátoru v rozmezí Dolního a Horního omezovače a Doby přejezdu.</td></tr>
<tr><td>Řídicí slovo (14)</td><td>Toto je interní parametr, který normálně zůstává na 0.

Pokud dolní bit Přepisu bude nastavený na 1, pro vzdálenou činnost tohoto bloku PID (viz níže) se musí použít toto slovo a další interní parametry SP, PV a CV. To umožňuje, aby vzdálená zařízení rozhraní obsluhy, například počítač, mohla převzít řízení mimo program PLC. Pozor: pokud nebudete chtít, aby k tomuto došlo, nastavte Řídicí slovo na nulu. Pokud dolní bit bude 0, další čtyři bity je možno načíst a sledovat stav vstupních kontaktů PID, dokud kontaktem Povolení PID nebude protékat proud. Struktura diskrétních dat s pozicemi prvních pěti bitů má následující formát:

<table>
<tr><th>Bit:</th><th>Hodnota slova:</th><th>Funkce:</th><th>Stav nebo Externí funkce, když bit Přepisu bude nastavený na 1:</th></tr>
<tr><td>0</td><td>1</td><td>Přepis</td><td>Je-li 0, monitorují se kontakty bloku níže. Je-li 1, nastaví se kontakty externě.</td></tr>
<tr><td>1</td><td>2</td><td>Ručně/ Auto</td><td>Je-li 1, blok bude v Ručním režimu; s jinými čísly bude v Automatickém režimu.</td></tr>
<tr><td>2</td><td>4</td><td>Povolení</td><td>Normálně musí být 1; jinak by se blok nikdy nevyvolal</td></tr>
<tr><td>3</td><td>8</td><td>UP/roste</td><td>Je-li 1 a Ručně (bit 1) bude v 1, CV se bude inkrementovat při každém řešení.</td></tr>
<tr><td>4</td><td>16</td><td>DN/klesá</td><td>Je-li 1 a Ručně (bit 1) bude v 1, CV se bude dekrementovat při každém řešení.</td></tr>
</table>
</td></tr>
<tr><td>SP (15)</td><td>(Nenastavuje se – nastavuje a spravuje PLC) Sleduje vstup SP; musí být nastaveno externě, je-li Přepis = 1.</td></tr>
<tr><td>CV (16)</td><td>(Nenastavuje se – nastavuje a spravuje PLC) Sleduje výstup CV.</td></tr>
<tr><td>PV (17)</td><td>(Nenastavuje se – nastavuje a spravuje PLC) Sleduje vstup PV; musí být nastaveno externě, je-li bit Přepis = 1.</td></tr>
<tr><td>Výstup (18)</td><td>(Nenastavuje se - nastavuje a spravuje PLC) Je to hodnota slova se znaménkem představující výstup funkčního bloku před aplikací volitelné inverze. Pokud nebude nakonfigurovaná žádná inverze a bit polarity výstupu v řídicím slově bude nastavený na 0, tato hodnota se bude rovnat výstupu CV. Pokud bude zvolena inverze a bit polarity výstupu bude nastavený na 1, tato hodnota se bude rovnat záporné hodnotě výstupu CV.</td></tr>
<tr><td>Paměť diferenciálních hodnot (19)</td><td>Používá se interně pro uložení přechodných hodnot. Nezapisujte do tohoto místa.</td></tr>
<tr><td>Paměť integrálních hodnot (20/21)</td><td>Používá se interně pro uložení přechodných hodnot. Nezapisujte do tohoto místa.</td></tr>
<tr><td>Paměť odezvových hodnot (22)</td><td>Používá se interně pro uložení přechodných hodnot. Nezapisujte do tohoto místa.</td></tr>
<tr><td>Hodiny (23–25)</td><td>Paměť hodin uplynulého času (čas posledního vykonání PID). Nezapisujte do těchto míst.</td></tr>
<tr><td>Zbytek Y (26)</td><td>Pamatuje si zbytek při dělení integrátoru pro nulovou odchylku v ustáleném stavu.</td></tr>
<tr><td>Dolní a Horní rozsah (27/28)</td><td>Volitelné celočíselné hodnoty v jednotkách PV, které definují nejvyšší a nejnižší zobrazovanou hodnotu horizontálního sloupcového diagramu SP a PV Logicmasteru (vyvoláno tlačítkem Zoom) a zobrazování ADS PID.</td></tr>
<tr><td>Vyhrazeno (29–34 a 35–39)</td><td>29–34 jsou vyhrazené pro interní použití; 35–39 jsou vyhrazené pro externí použití. Jsou vyhrazené pro použití GE Fanuc a nelze je použít pro jiné účely.</td></tr>
</table>

Interní parametry v RefArray

Jak bylo popsáno v tabulce 12-3 na předchozích stránkách, blok PID čte 13 uživatelských parametrů a používá zbylých 40 slov z RefArray pro interní ukládání PID. Normálně byste nikdy neměli potřebovat žádné z těchto hodnot měnit. Pokud vyvoláte blok PID v Automatickém režimu, možná budete chtít použít SVC_REQ #16 k načtení aktuálních hodin uplynulého času PLC do %Ref+23 a aktualizovat poslední čas řešení PID, aby na integrátoru nedošlo ke skokové změně. Pokud dolní bit Přepis Řídicího slova (%Ref+14) bude nastavený na 1, další čtyři bity Řídicího slova musí být nastavené tak, aby se řídily vstupní kontakty bloku PID (podle popisu v tabulce 12-3 na předchozích stranách), a interní SP a PV musí být nastavené, jako by z žebříkové logiky řízení bloku PID byly vyjmuté.

Volba algoritmu PID (PIDISA nebo PIDIND) a zisky

Blok PID je možno naprogramovat tak, že se zvolí buď algoritmus PID verze s Nezávislou (PID_IND) hodnotou nebo se standardním ISA (PID_ISA). Jediný rozdíl v algoritmu je, jak jsou definované integrační a derivační zisky. Abychom pochopili rozdíl, je nutno porozumět následujícímu:

Oby typy PID vypočítávají hodnotu Odchylky jako SP – PV, kterou je možno změnit na Opačně pracující režim PV – SP, když Hodnota Odchylky (dolní bit 0 v konfiguračním slově %Ref+12) bude nastavená na 1. Opačně fungující režim je možno použít, když chcete, aby se výstup CV pohyboval v opačném směru změn na vstupu PV (CV klesá při rostoucím PV) místo normálního růstu CV při rostoucím PV.

Odchylka = (SP – PV) nebo (PV – SP), pokud dolní bit Konfiguračního slova bude nastavený na 1

Derivace se normálně počítá ze změny hodnoty Odchylky od posledního řešení PID, což může vyvolat velkou změnu na výstupu, když se změní hodnota SP. Pokud toto není žádoucí, třetí bit Konfiguračního slova je možno nastavit na 1 a vypočítat derivaci ze změny PV. Hodnota dt (nebo rozdíl času) je určený odečtením poslední doby hodin PID pro tento blok od hodin aktuálního uplynulého času PLC.

dt = Hodiny aktuálního uplynulého času PLC – Hodiny aktuálního uplynulého času při posledním řešení PID

Derivace = (Odchylka – předchozí Odchylka)/dt nebo (PV – předchozí PV)/dt, když 3. bit Konfiguračního slova bude nastavený na 1

Algoritmus s nezávislou hodnotou PID (PID_IND) vypočítá výstup jako:

Výstup PID = Kp * Odchylka + Ki * Odchylka* dt + Kd * Derivace + Posunutí CV

Algoritmus standardního ISA (PID_ISA) má odlišný tvar:

Výstup PID = Kc * (Odchylka + Odchylka * dt/Ti + Td * Derivace) + Posunutí CV

kde Kc ke zisk regulátoru a Ti je Integrační doba a Td je doba Derivace. Výhodou ISA je, že změna Kc vyvolá změnu podílu integrální a derivační hodnoty i proporcionální, což může vést na snazší optimalizaci smyčky. Pokud budete mít zisky PID hodnot nebo Ti a Td, použijte

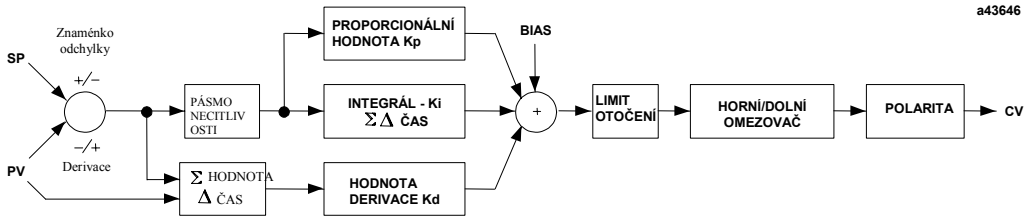
Kp = Kc Ki = Kc/Ti a Kd = Kc/Td

k jejich převedení a k použití jako vstupy uživatelských parametrů PID.

Výše uvedené Posunutí hodnoty CV je přídavná hodnota oddělená od složek PID. To může být nutné, když budete používat pouze proporcionální zisk Kp a budete chtít, aby CV mělo nenulovou hodnotu, když se PV bude rovnat SP a Odchylka bude 0. V takovém případě nastavte Posunutí CV na požadovanou hodnotu CV, když PV bude mít hodnotu SP. Posunutí CV je také možno použít pro dopředné řízení, kde se používá jiná smyčka PID nebo řídicí algoritmus k nastavení výstupu CV této smyčky PID.

Pokud se použije integrální zisk Ki, Posunutí CV bude normálně 0, protože integrátor působí jako automatické posunutí. K nastavení integrátoru na požadovanou hodnotu CV spusťte Ruční režim a použijte slovo Ručního povelu (%Ref+13), pak přejděte do Automatického režimu. To také funguje, když Ki bude 0, ale s tou výjimkou, že integrátor po přechodu do Automatického režimu nebude nastavený podle Odchylky.

Následující schéma ukazuje, jak algoritmus PID pracuje:



Obrázek 12-4. Algoritmus nezávislé hodnoty (PIDIND)

Algoritmus ISA (PIDISA) je podobný ale s tou výjimkou, že zisk Kp se vypočítá z Ki a Kd, takže integrální zisk bude Kp * Ki a derivační zisk bude Kp * Kd. Znaménko odchylky, derivace a polarita se nastavují pomocí bitů v uživatelském parametru Konfiguračního slova.

Amplituda CV a meze rychlosti

Blok neposílá vypočítaný výstup PID přímo na CV. Oba algoritmy PID mohou na výstupu Řídicí proměnné vyvolat omezení změny amplitudy a rychlosti. Maximální rychlost změny je určena vydělením maximální 100% hodnoty CV (32000) Minimální dobou přejezdu, pokud bude zadaná větší než 0. Pokud například Minimální doba přejezdu bude 100 sekund, omezení rychlosti bude 320 jednotek CV za sekundu. Pokud dt doby řešení bylo 50 milisekund, nový výstup CV se nemůže změnit o více než 320*50/1000 nebo 16 jednotek CV od předchozího výstupu CV.

Výstup CV se pak porovná s hodnotami Horního a Dolního omezovače CV. Pokud dojde k překročení některé meze, výstup CV se nastaví na omezenou hodnotu. Pokud se překročí meze rychlosti nebo amplitudy a změní se CV, interní hodnota integrátoru se nastaví tak, aby souhlasila s omezenou hodnotou a nedocházelo k přetočení resetem.

Nakonec blok zkontroluje Polaritu výstupu (2. bit Konfiguračního slova %Ref+12) a pokud tento bit bude na 1, změní znaménko výstupu.

CV = Omezený výstup PID nebo –Omezený výstup PID, když je nastavený bit Polarita výstupu

Pokud blok bude v Automatickém režimu, výsledné CV se uloží do Ručního povelu %Ref+13. Pokud blok bude v Ručním režimu, rovnice PID se přeskočí, protože CV je nastaveno Ručním povelem, ale stále se budou kontrolovat všechna omezení rychlosti a amplitudy. To znamená, že Ruční povel nemůže změnit výstup nad Horní omezovač CV nebo pod Dolní omezovač CV a výstup se nemůže měnit rychleji než je přípustná Minimální doba přejezdu.

Perioda vzorkování a plánování bloku PID

Blok PID je digitální implementace analogové řídicí funkce, takže časový vzorek dt na výstupu rovnice PID není nekonečně malý časový vzorek použitelný při analogovém řízení. Většinu řízených procesů je možno aproximovat jako zisk se zpožděním prvního nebo druhého řádu, možná s čistým časovým zpožděním. Blok PID nastaví výstup CV pro proces a používá zpětnou vazbu procesu PV ke zjištění Odchylky a k upravení následujícího výstupu CV. Klíčový parametr procesu je celková časová konstanta, což je rychlost, kterou PV odpoví na změnu CV. Jak bylo popsáno v předchozím odstavci Nastavení zisků smyčky, celková časová konstanta Tp+Tc pro systém prvního řádu je čas požadovaný k tomu, aby PV dosáhlo 63% své konečné hodnoty, když se změní CV. Pokud Perioda vzorkování nebude značně kratší než celková časová konstanta, blok PID nebude schopný řídit proces. Větší Periody vzorkování vytvoří nestabilní blok.

Perioda vzorkování nesmí být větší než celková časová konstanta dělená 10 (nebo v nejhorším případě 5). Pokud se například bude zdát, že PV dosáhne asi 2/3 své konečné hodnoty během 2 sekund, Perioda vzorkování musí být menší než 0,2 sekundy nebo v nejhorším případě 0,4 sekundy. Na druhé straně Perioda vzorkování nesmí být příliš malá, například menší než celková časová konstanta dělená 1000, jinak by hodnota Ki*Odchylka*dt pro integrátor PID provedla zaokrouhlení na 0. Například velmi pomalý proces, který k dosažení úrovně 63% potřebuje 10 hodin čili 36000 sekund, musí mít Periodu vzorkování 40 sekund nebo delší.

Pokud proces nebude velmi rychlý, není obvykle nutné použít Periodu vzorkování 0 pro řešení algoritmu v každém cyklu. Pokud se používá hodně PID smyček s Periodou vzorkování větší než doba cyklu, může docházet k velkému kolísání doby cyklu PLC, když mnoho smyček bude končit řešením algoritmu ve stejnou dobu. Jednoduchým řešením je seřadit jeden nebo více bitů do pole bitů nastavených na 0, které se používá k povolení průtoku proudu do jednotlivých bloků PID.

Určení charakteristik procesu

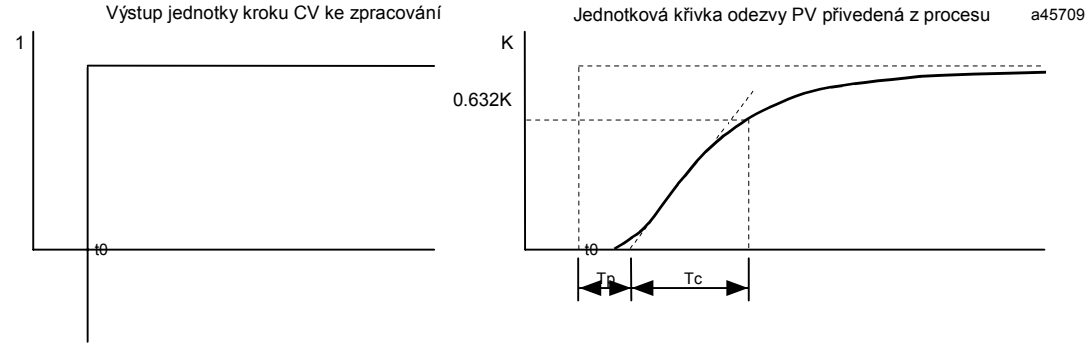
Zisky smyčky PID, Kp, Ki a Kd, jsou určené charakteristikami řízeného procesu. Dvě klíčové otázky pro nastavování smyčky PID jsou:

1. Jak velká bude změna na PV, když se CV změní o pevnou velikost, neboli jaký je zisk otevřené smyčky.
2. Jak rychle systém odpoví neboli jak rychle se změní PV po skokové změně výstupu CV?

Mnoho procesů je možno aproximovat ziskem procesu, zpožděním prvního nebo druhého řádu a čistým časovým zpožděním. Ve frekvenční oblasti funkce přenosu pro zpoždění prvního řádu s čistým časovým zpožděním bude:

PV(s)/CV(s) = G(s) = K * e **(–Tp s)/(1 + Tc s)

Grafické znázornění skokové změny v čase t0 v časové oblasti vytvoří křivku odezvy s otevřenou smyčkou:



Parametry následujícího modelu procesu je možno určit z křivky odezvy jednotky PV:

K	Zisk otevřené smyčky procesu = výsledná změna na PV/změna na CV v čase t0 (Žádný index u K)
Tp	Časové zpoždění nebo doba nečinnosti procesu nebo potrubí po t0 před tím, než se výstup procesu PV se začne hýbat.
Tc	Časová konstanta procesu prvního řádu, čas požadovaný po Tp, aby PV dosáhlo 63.2% cílové hodnoty PV

Obvykle nejrychlejší způsob, jak tyto parametry změřit, je přepnout blok PID do Ručního režimu a provádět malé kroky na výstupu CV tak, že se bude měnit Ruční povel %Ref+13 a do grafu vynášet odezvy PV v závislosti na čase. U pomalých procesů to lze provádět ručně, ale u rychlejších procesů bude lepší použít zapisovač nebo počítačový program pro záznam grafických dat. Velikost kroku CV musí být dostatečně velká, aby vyvolala pozorovatelnou změnu na PV, ale ne zase tak velká, aby se přerušil měřený proces. Dobrá velikost může být v rozmezí od 2 do 10% rozdílu mezi hodnotou Horního a Dolního omezovače CV.

Nastavení uživatelských parametrů včetně optimalizace zisků smyčky

Protože všechny parametry PID zcela závisí nařízeném procesu, neexistují předem dané hodnoty, které by fungovaly, avšak nalezení vhodného zisku smyčky je obvykle otázka jednoduché iterační metody.

1. Nastavte všechny parametry funkčního bloku na 0, pak nastavte Horní a Dolní omezovač CV na nejvyšší a nejnižší očekávanou hodnotu CV. Periodu vzorkování nastavte na odhadovanou časovou konstantu procesu (viz výše)/10 až 100.

2. Blok přepněte do Ručního režimu a nastavte Ruční povel (%Ref+13) na různé hodnoty, aby se zjistilo, jestli se CV může dostat na Horní a Dolní omezovač. Zaznamenejte hodnotu PV v některém bodě CV a uložte jí do SP.

3. Nastavte malý zisk, například 100*Maximum CV/Maximum PV, do Kp a ukončete Ruční režim. Krokujte SP v inkrementech 2 až 10% maximálního rozsahu PV a sledujte odezvu PV. Kp zvětšete, když kroková odezva PV bude příliš pomalá, nebo Kp zmenšete, když PV překmitne a bude oscilovat, aniž by se dosáhla ustálená hodnota.

4. Jakmile zjistíte Kp, začněte zvyšovat Ki, aby se dosáhlo překmitnutí, které se utlumí na ustálené hodnotě během 2 až 3 cyklů. To může vyžadovat zmenšení Kp. Také zkuste různé velikosti kroků a pracovní body CV.

5. Po nalezení vhodné hodnoty zisků Kp a Ki zkuste přidat Kd, aby se dosáhlo rychlejší odezvy na vstupní změny za předpokladu, že to nezpůsobí oscilace. Kd často není nutné a se zašuměným PV nebude fungovat.

6. Zkontrolujte zisky v různých pracovních bodech SP a v případě potřeby přičtěte pásmo necitlivosti a minimální dobu přejezdu. Některé opačně pracující procesy mohou potřebovat nastavení znaménka Odchylky v Konfiguračním slově nebo bitů polarity.

Nastavení zisků smyčky —Ziegler-Nicholsova optimalizační metoda

Jakmile budou určené modelové parametry procesu K, Tp a Tc, je možno je použít k odhadů zisků smyčky PID. Následující metoda, kterou vyvinuli Ziegler a Nichols ve 40. letech 20. století, je určena k vytvoření dobré odezvy na vzruchy systému se ziskem vytvářejícím poměr amplitud 1/4. Poměr amplitud je poměr druhé špičky k první špičce v odezvě uzavřené smyčky.

1. Vypočtěte rychlost reakce:

   R = K/Tc

2. Pouze pro proporcionální řízení vypočtěte Kp jako

   Kp = 1/(R * Tp) = Tc/(K * Tp)

3. Pro proporcionální a integrační řízení použijte

   Kp = 0.9/(R * Tp) = 0.9 * Tc/(K * Tp)
   Ki = 0.3 * Kp/Tp

4. Pro proporcionální, integrační a derivační řízení použijte

   Kp = G/(R * Tp) kde G je od 1.2 do 2.0
   Ki = 0.5 * Kp/Tp
   Kd = 0.5 * Kp * Tp

5. Zkontrolujte, že Perioda vzorkování je v rozsahu (Tp + Tc)/10 až (Tp + Tc)/1000.

Jiná metoda, postup “Ideální optimalizace”, je navržena tak, aby zajistila nejlepší odezvu na změny SP, která bude zpožděná pouze o zpoždění procesu Tp o dobu nečinnosti.

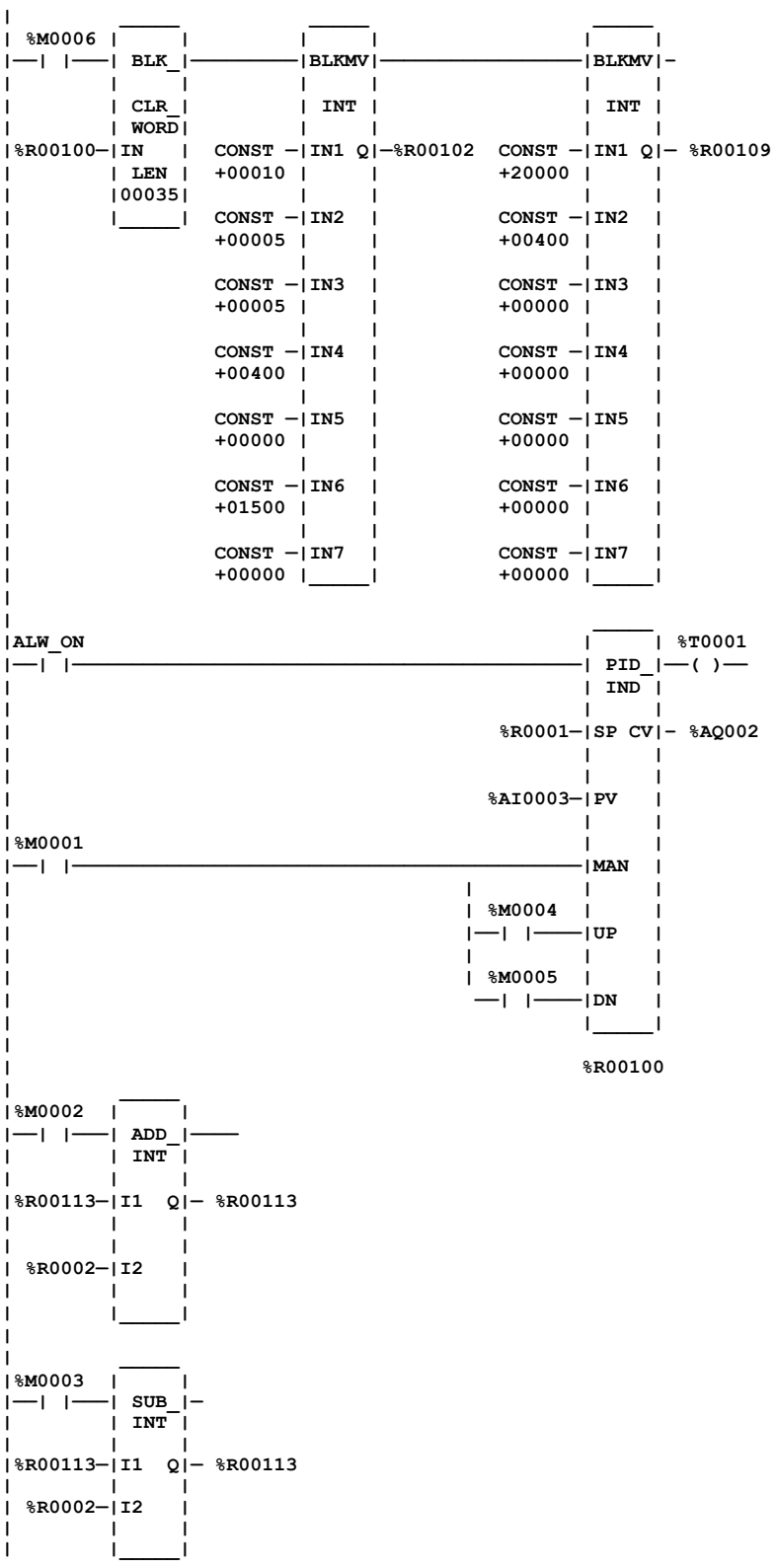
Kp = 2 * Tc/(3 * K * Tp)
Ki = Tc
Kd = Ki/4 pokud se použije derivační hodnota

Jakmile budou určené počáteční zisky, je nutno je převést na celočíselné Uživatelské parametry. Aby se zabránilo problémům s měřítkem, zisk procesu K se musí vypočítat jako změna v jednotkách vstupu PV dělená krokem změny výstupu v jednotkách CV a ne v technických jednotkách PV nebo CV. Všechny časy musí být udávané v sekundách. Jakmile bude určeno Kp, Ki a Kd, hodnoty Kp a Kd je nutno násobit 100 a zapsat jako celá čísla, zatímco Ki je nutno násobit 1000 a zapsat do Uživatelského parametru %RefArray.

Příklad volání PID

V následujícím příkladu je Perioda vzorkování 100 milisekund a zisk Kp 4.00 a zisk Ki 1.500. Bod nastavení je uložený v %R1, výstup Řízené proměnné v %AQ2 a výstup Proměnné procesu v %AI3. Horní a Dolní omezovač CV musí být nastavený, v tomto případě na 20000 a 400, a bylo zahrnuto malé pásmo necitlivosti +5 a –5. 40 slov RefArray začíná na %R100. Uživatelské parametry se normálně nastaví v RefArray pomocí PID Zoom tlačítka **F10**, ale %M6 je možno inicializovat těchto 14 slov začínajících na %R102 (%Ref+2) od konstant uložených v logice.

Blok je možno přepnout do Ručního režimu pomocí %M1, aby bylo možno nastavit Ruční povel %R113. Bit %M4 nebo %M5 je možno použít ke zvětšení nebo zmenšení %R113 a CV PID a integrátor o 1 při řešení každých 100 milisekund. Pro rychlejší ruční obsluhu je možno použít bity %M2 a %M3 k přičtení nebo odečtení hodnoty v %R2 k/od %R113 při každém cyklu PLC. Výstup %T1 bude v jedničce, když PID bude OK.


%M0006
BLK_
CLR_
WORD
%R00100
IN
LEN
00035
BLKMV
INT
CONST +00010
IN1 Q
%R00102
IN2
CONST +00005
IN3
CONST +00005
IN4
CONST +00400
IN5
CONST +00000
IN6
CONST +01500
IN7
CONST +00000
CONST +20000
CONST +00400
CONST +00000
CONST +00000
CONST +00000
CONST +00000
%R00109
ALW_ON
PID_
IND
%T0001
%R0001
SP CV
%AQ002
%AI0003
PV
%M0001
MAN
%M0004
UP
%M0005
DN
%R00100
%M0002
ADD_
INT
%R00113
I1
Q
I2
%R0002
%M0003
SUB_
INT

Dodatek A

Časování instrukcí

PLC Series 90-30, 90-20 a Micro podporují mnoho různých funkcí a funkčních bloků. Tento dodatek obsahuje tabulky uvádějící velikost paměti v bajtech a dobu vykonávání jednotlivých funkcí v mikrosekundách. Velikost paměti je počet bajtů, který vyžaduje funkce v aplikačním programu žebříkové logiky.

U každé funkce jsou uvedené dvě doby vykonávání:

Doba vykonávání	Popis
Povoleno	Doba vyžadovaná k vykonání funkce nebo funkčního bloku, když do funkce nebo ven z funkce bude protékat proud. Nejlepší doby se obvykle dosáhne, když data, která blok používají, budou uložena v uživatelské RAM (slovně orientované paměti) a ne v diskrétní paměti.
Zakázáno	Doba vyžadovaná k vykonání funkce, když proud bude téct do funkce nebo do funkčního bloku; je to však neaktivní stav, jako když časovač bude držený ve stavu resetu.

Poznámka

Časovače a čítače se aktualizují při každém zjištění v logice; časovače o dobu spotřebovanou posledním cyklem a čítače o jeden krok.

Poznámka

V případě CPU 350, 351, 352 a 360 jsou doby shodné s výjimkou instrukce MOVE, kde doba v případě CPU 350 je jiná - viz poznámka na konci tabulky na straně A-6.

Tabulka A-1. Časování instrukce, standardní modely

Skupina funkcí	Funkce	Povoleno				Zakázáno				Inkrement				Velikost
		311	313	331	340/41	311	313	331	340/41	311	313	331	340/41	
Časovače	Časovač zpoždění při zapnutí	146	81	80	42	105	39	38	21	–	–	–	–	15
	Časovač zpoždění při vypnutí	98	47	44	23	116	63	58	32	–	–	–	–	9
	Časovač	122	76	75	40	103	54	53	30	–	–	–	–	15
Čítače	Vzestupný čítač	137	70	69	36	130	63	62	33	–	–	–	–	11
	Sestupný čítač	136	70	69	37	127	61	61	31	–	–	–	–	11
Matematické	Sčítání (INT)	76	47	46	24	41	0	1	0	–	–	–	–	13
	Sčítání (DINT)	90	60	60	34	41	1	0	0	–	–	–	–	13
	Odečítání (INT)	75	46	45	25	41	0	1	0	–	–	–	–	13
	Odečítání (DINT)	92	62	62	34	41	1	0	0	–	–	–	–	13
	Násobení (INT)	79	49	50	28	41	0	1	0	–	–	–	–	13
	Násobení (DINT)	108	80	101	43	41	1	0	0	–	–	–	–	13
	Dělení (INT)	79	51	50	27	41	0	1	0	–	–	–	–	13
	Dělení (DINT)	375	346	348	175	41	1	0	0	–	–	–	–	13
	Dělení modulo (INT)	78	51	49	27	41	0	1	0	–	–	–	–	13
	Dělení modulo (DINT)	134	103	107	54	41	1	0	0	–	–	–	–	13
	Druhá odmocnina (INT)	153	124	123	65	42	0	1	0	–	–	–	–	9
	Druhá odmocnina (DINT)	268	239	241	120	42	0	0	1	–	–	–	–	9
Relační	Rovno (INT)	66	35	36	19	41	1	1	0	–	–	–	–	9
	Rovno (DINT)	86	56	54	29	41	1	0	0	–	–	–	–	9
	Nerovno (INT)	67	39	35	22	41	1	1	0	–	–	–	–	9
	Nerovno (DINT)	81	51	51	28	41	1	0	0	–	–	–	–	9
	Větší než (INT)	64	33	35	20	41	1	1	0	–	–	–	–	9
	Větší než (DINT)	89	59	58	32	41	1	0	0	–	–	–	–	9
	Větší než / Rovno (INT)	64	36	34	19	41	1	1	0	–	–	–	–	9
	Větší než / Rovno (DINT)	87	58	57	30	41	1	0	0	–	–	–	–	9
	Menší než (INT)	66	35		19	41	1	1	0	–	–	–	–	9
	Menší než (DINT)	87	57		30	41	1	1	0	–	–	–	–	9
	Menší než / Rovno (INT)	66	36	34	21	41	1	1	0	–	–	–	–	9
	Menší než / Rovno (DINT)	86	57	56	31	41	1	1	0	–	–	–	–	9
	Rozsah (INT)	92	58	54	29	46	1	0	1	–	–	–	–	15
	Rozsah (DINT)	106	75	57	37	45	0	0	0	–	–	–	–	15
	Rozsah (WORD)	93	60	54	29	0	0	0	0	–	–	–	–	15

Poznámky:

1. Doba (v mikrosekundách) platí pro verzi 5.01 softwaru Logicmaster 90-30/20 pro CPU model 311, 313, 340 a 341 (verze 7 pro 331).
2. U tabulkových funkcí jsou inkrementy v jednotkách zadané délky; u funkcí s bitovými operacemi v mikrosekundách/bit; u funkcí přesunu dat v mikrosekundách/počet bitů nebo slov.
3. Doba povolení pro jednotky s jednoduchou délkou typu %R, %AI a %AQ.
4. Doba COMMREQ se měřila mezi CPU a HSC.
5. DOIO je doba k vygenerování hodnoty na výstupu diskrétního výstupního modulu.
6. Kde je více možností, výše uvedená doba představuje nejhorší možný případ.

Tabulka A-1. Časování instrukce, standardní modely – pokračování

Skupina funkcí	Funkční	Povoleno				Zakázáno				Inkrement				Velikost
		311	313	331	340/41	311	313	331	340/41	311	313	331	340/41	
Bitové operace	Logický AND	67	37	37	22	42	0	0	1	–	–	–	–	13
	Logický OR	68	38	38	21	42	0	0	1	–	–	–	–	13
	Logický Exclusive OR	66	38	37	20	42	0	1	1	–	–	–	–	13
	Logická inverze, NOT	62	32	31	17	42	0	1	1	–	–	–	–	9
	Posunutí bitu doleva	139	89	90	47	74	26	23	13	11.61	11.61	12.04	6.29	15
	Posunutí bitu doprava	135	87	85	45	75	26	24	13	11.63	11.62	12.02	6.33	15
	Rotace bitu doleva	156	127	126	65	42	1	1	0	11.70	11.78	12.17	6.33	15
	Rotace bitu doprava	146	116	116	62	42	1	1	0	11.74	11.74	12.13	6.27	15
	Pozice bitu	102	72	49	38	42	1	0	0	–	–	–	–	13
	Smazání bitu	68	38	35	21	42	1	1	1	–	–	–	–	13
	Testování bitu	79	49	51	28	41	0	0	1	–	–	–	–	13
	Nastavení bitu	67	37	37	20	42	0	0	0	–	–	–	–	13
	Maskované porovnání (WORD)	217	154	141	74	107	44	39	21	–	–	–	–	25
	Maskované porovnání (DWORD)	232	169	156	83	108	44	39	22	–	–	–	–	25
Přesunutí dat	Přesunutí (INT)	68	37	39	20	43	0	0	0	1.62	1.62	5.25	1.31	13
	Přesunutí (BIT)	94	62	64	35	42	0	0	0	12.61	12.64	12.59	6.33	13
	Přesunutí (WORD)	67	37	40	20	41	0	0	0	1.62	1.63	5.25	1.31	13
	Přesun bloku (INT)	76	48	50	28	59	30	30	16	–	–	–	–	27
	Přesun bloku (WORD)	76	48	49	29	59	29	28	15	–	–	–	–	27
	Smazání bloku	56	28	27	14	43	0	0	0	1.35	1.29	1.40	0.78	9
	Posunutí registru (BIT)	201	153	153	79	85	36	34	18	0.69	0.68	0.71	0.37	15
	Posunutí registru (WORD)	103	53	52	29	73	25	23	12	1.62	1.62	2.03	1.31	15
	Bitový sekvenční přepínač	165	101	99	53	96	31	29	16	0.07	0.07	0.08	0.05	15
	COMM_REQ	1317	1272	1489	884	41	2	0	0	–	–	–	–	13
Tabulkové	Přesunutí pole													
	INT	230	201	177	104	72	41	40	20	1.29	1.15	10.56	2.06	21
	DINT	231	202	181	105	74	44	42	23	3.24	3.24	10.53	2.61	21
	BIT	290	261	229	135	74	43	42	23	–.03	–.03	-0.01	0.79	21
	BYTE	228	198	176	104	74	42	42	23	0.81	0.82	8.51	1.25	21
	WORD	230	201	177	104	72	41	40	20	1.29	1.15	10.56	2.06	21
	Hledat shodné													
	INT	197	158	123	82	78	39	37	20	1.93	1.97	2.55	1.55	19
	DINT	206	166	135	87	79	38	36	21	4.33	4.34	4.55	2.44	19
	BYTE	179	141	117	74	78	38	36	21	1.53	1.49	1.83	1.03	19
	WORD	197	158	123	82	78	39	37	20	1.93	1.97	2.55	1.55	19

Poznámky:
1. Doba (v mikrosekundách) platí pro verzi 5.01 softwaru Logicmaster 90-30/20 pro CPU model 311, 313, 340 a 341 (verze 7 pro 331).
2. U tabulkových funkcí jsou inkrementy v jednotkách zadané délky; u funkcí s bitovými operacemi v mikrosekundách/bit; u funkcí přesunu dat v mikrosekundách/počet bitů nebo slov.
3. Doba povolení pro jednotky s jednoduchou délkou typu %R, %AI a %AQ.
4. Doba COMMREQ se měřila mezi CPU a HSC.
5. DOIO je doba k vygenerování hodnoty na výstupu diskrétního výstupního modulu.
6. Kde je více možností, výše uvedená doba představuje nejhorší možný případ.
7. U instrukcí, které mají hodnotu inkrementu, násobte inkrement (délkou –1) a přičtěte tuto hodnotu k základnímu času.

Tabulka A-1. Časování instrukce, standardní modely – pokračování

Skupina funkcí	Funkce	Povoleno				Zakázáno				Inkrement				Velikost
		311	313	331	340/41	311	313	331	340/41	311	313	331	340/41	
	Hledat neshodné													
	INT	198	159	124	83	79	39	36	21	1.93	1.93	2.48	1.52	19
	DINT	201	163	132	84	79	37	35	21	6.49	6.47	6.88	3.82	19
	BYTE	179	141	117	73	79	38	36	19	1.54	1.51	1.85	1.05	19
	WORD	198	159	124	83	79	39	36	21	1.93	1.93	2.48	1.52	19
	Hledat větší než													
	INT	198	160	125	82	79	37	38	19	3.83	3.83	4.41	2.59	19
	DINT	206	167	135	88	78	38	36	20	8.61	8.61	9.03	4.88	19
	BYTE	181	143	118	73	79	37	36	19	3.44	3.44	3.75	2.03	19
	WORD	198	160	125	82	79	37	38	19	3.83	3.83	4.41	2.59	19
	Hledat větší než / Rovno													
	INT	197	160	124	83	77	38	36	20	3.86	3.83	4.45	2.52	19
	DINT	205	167	136	87	80	39	36	21	8.62	8.61	9.02	4.87	19
	BYTE	180	142	118	75	79	37	37	20	3.47	3.44	3.73	2.00	19
	WORD	197	160	124	83	77	38	36	20	3.86	3.83	4.45	2.52	19
	Hledat menší než													
	INT	199	159	124	84	78	38	36	20	3.83	3.86	4.48	2.48	19
	DINT	206	168	135	87	79	38	38	19	8.62	8.60	-1.36	4.88	19
	BYTE	181	143	119	75	80	38	37	20	3.44	3.44	3.75	2.00	19
	WORD	199	159	124	84	78	38	36	20	3.83	3.86	4.45	2.48	19
	Hledat menší než / Rovno													
	INT	200	158	124	82	79	38	37	21	3.79	3.90	4.45	2.55	19
	DINT	207	167	137	88	78	39	37	19	8.60	8.61	9.01	4.86	19
	BYTE	180	143	119	74	78	40	37	19	3.46	3.44	3.73	2.02	19
	WORD	200	158	124	82	79	38	37	21	3.79	3.90	4.45	2.55	19
Převod	Převod na INT	74	46	39	25	42	1	1	1	–	–	–	–	9
	Převod na BCD–4	77	50	34	25	42	1	1	1	–	–	–	–	9

Poznámky: 1. Doba (v mikrosekundách) platí pro verzi 5.01 softwaru Logicmaster 90-30/20 pro CPU model 311, 313, 340 a 341 (verze 7 pro 331).

2. U tabulkových funkcí jsou inkrementy v jednotkách zadané délky; u funkcí s bitovými operacemi v mikrosekundách/bit; u funkcí přesunu dat v mikrosekundách/počet bitů nebo slov.
3. Doba povolení pro jednotky s jednoduchou délkou typu %R, %AI a %AQ.
4. Doba COMMREQ se měřila mezi CPU a HSC.
5. DOIO je doba k vygenerování hodnoty na výstupu diskrétního výstupního modulu.
6. Kde je více možností, výše uvedená doba představuje nejhorší možný případ.
7. U instrukcí, které mají hodnotu inkrementu, násobte inkrement (délkou –1) a přičtěte tuto hodnotu k základnímu času.

Tabulka A-1. Časování instrukce, standardní modely – pokračování

Skupina funkcí	Funkce	Povoleno				Zakázáno				Inkrement				Velikost
		311	313	331	340/41	311	313	331	340/41	311	313	331	340/41	
Řízení	Volání podprogramu	155	93	192	85	41	0	0	0	–	–	–	–	7
	Do I/O	309	278	323	177	38	1	0	0	–	–	–	–	12
	Algoritmus PID – ISA	1870	1827	1812	929	91	56	82	30	–	–	–	–	15
	Algoritmus PID – IND	2047	2007	2002	1017	91	56	82	30	–	–	–	–	15
	Instrukce End	–	–	–	–	–	–	–	–	–	–	–	–	–
	Service Request													
	# 6	93	54	63	45	41	2	0	0	–	–	–	–	9
	# 7 (Čtení)	–	37	309	161	–	2	0	0	–	–	–	–	9
	# 7 (Nastavení)	–	37	309	161	–	2	0	0	–	–	–	–	9
	#14	447	418	483	244	41	2	0	0	–	–	–	–	9
	#15	281	243	165	139	41	2	0	0	–	–	–	–	9
	#16	131	104	115	69	41	2	0	0	–	–	–	–	9
	#18	–	56	300	180	–	2	0	0	–	–	–	–	9
	#23	1689	1663	1591	939	43	1	0	0	–	–	–	–	9
	#26//30*	1268	1354	6680	3538	42	0	0	0	–	–	–	–	9
	#29	–	–	55	41	–	–	1	0	–	–	–	–	9
	Vnořené MCR/ENDMCR kombinované	135	73	68	39	75	25	21	12	–	–	–	–	8

***** Service request #26/30 byl měřený s použitím vysokorychlostního čítače, 16-bodového výstupu v sestavě s 5 pozicemi.

Poznámky: 1. Doba (v mikrosekundách) platí pro verzi 5.01 softwaru Logicmaster 90-30/20 pro CPU model 311, 313, 340 a 341 (verze 7 pro 331).

2. U tabulkových funkcí jsou inkrementy v jednotkách zadané délky; u funkcí s bitovými operacemi v mikrosekundách/bit; u funkcí přesunu dat v mikrosekundách/počet bitů nebo slov.
3. Doba povolení pro jednotky s jednoduchou délkou typu %R, %AI a %AQ.
4. Doba COMMREQ se měřila mezi CPU a HSC.
5. DOIO je doba k vygenerování hodnoty na výstupu diskrétního výstupního modulu.
6. Kde je více možností, výše uvedená doba představuje nejhorší možný případ.
7. U instrukcí, které mají hodnotu inkrementu, násobte inkrement (délkou –1) a přičtěte tuto hodnotu k základnímu času.

Tabulka A-2. Časování instrukcí, modely s vysokým výkonem

Skupina funkcí	Funkce	Povoleno 350/351/36x	Zakázáno 350/351/36x	Inkrement 350/351/36x	Povoleno 352	Zakázáno 352	Inkrement 352	Velikost
Časovače	Časovač zpoždění při zapnutí	4	6	–	4	5	–	15
	Časovač	3	3	–	2	2	–	15
	Časovač zpoždění při vypnutí	3	3	–	3	2	–	15
Čítače	Vzestupný čítač	1	3	–	2	2	–	13
	Sestupný čítač	3	3	–	1	2	–	13
Matematické	Sčítání (INT)	2	0	–	1	0	–	13
	Sčítání (DINT)	2	0	–	2	0	–	19
	Sčítání (REAL)	52	0	–	33	0	–	17
	Odečítání (INT)	2	0	–	1	0	–	13
	Odečítání (DINT)	2	0	–	2	0	–	19
	Odečítání (REAL)	53	0	–	34	0	–	17
	Násobení (INT)	21	0	–	21	0	–	13
	Násobení (DINT)	24	0	–	24	0	–	19
	Násobení (REAL)	68	1	–	38	1	–	17
	Dělení (INT)	22	0	–	22	0	–	13
	Dělení (DINT)	25	0	–	25	0	–	19
	Dělení (REAL)	82	2	–	36	2	–	17
	Dělení modulo (INT)	21	0	–	21	0	–	13
	Dělení modulo (DINT)	25	0	–	25	0	–	19
	Druhá odmocnina (INT)	42	1	–	41	1	–	10
	Druhá odmocnina (DINT)	70	0	–	70	0	–	13
	Druhá odmocnina (REAL)	137	0	–	35	0	–	11
Trigonometrické	SIN (REAL)	360	0	–	32	0	–	11
	COS (REAL)	319	0	–	29	0	–	11
	TAN (REAL)	510	1	–	32	1	–	11
	ASIN (REAL)	440	0	–	45	0	–	11
	ACOS (REAL)	683	0	–	63	0	–	11
	ATAN (REAL)	264	1	–	33	1	–	11
Logaritmické	LOG (REAL)	469	0	–	32	0	–	11
	LN (REAL)	437	0	–	32	0	–	11
Exponenciální	EXP	639	0	–	42	0	–	11
	EXPT	89	1	–	54	1	–	17
Převod radiánů	Převod RAD na DEG	65	1	–	32	1	–	11
	Převod DEG na RAD	59	0		32	0		11

Poznámky:

1. Doba (v mikrosekundách) platí pro verzi 7 softwaru Logicmaster 90-30/20/Micro pro CPU Model 351 a 352.
2. U tabulkových funkcí jsou inkrementy v jednotkách zadané délky; u funkcí s bitovými operacemi v mikrosekundách/bit; u funkcí přesunu dat v ikrosekundách/početbitů nebo slov.
3. Doba povolení pro jednotky s jednoduchou délkou typu %R, %AI a %AQ.
4. Doba COMMREQ se měřila mezi CPU a HSC.
5. DOIO je doba k vygenerování hodnoty na výstupu diskrétního výstupního modulu.
6. Kde je více možností, výše uvedená doba představuje nejhorší možný případ.

Tabulka A-2. Časování instrukcí, modely s vysokým výkonem – pokračování

Skupina funkcí	Funkce	Povoleno 350/351/36x	Zakázáno 350/351/36x	Inkrement 350/351/36x	Povoleno 352	Zakázáno 352	Inkrement 352	Velikost
Relační	Rovno (INT)	1	0	–	1	0	–	10
	Rovno (DINT)	2	0	–	2	0	–	16
	Rovno (REAL)	57	0	–	28	0	–	14
	Nerovno (INT)	1	0	–	1	0	–	10
	Nerovno (DINT)	1	0	–	1	0	–	16
	Nerovno (REAL)	62	0	–	31	0	–	14
	Větší než (INT)	1	0	–	1	0	–	10
	Větší než (DINT)	1	0	–	1	0	–	16
	Větší než (REAL)	57	0	–	32	0	–	14
	Větší než / Rovno (INT)	1	0	–	1	0	–	10
	Větší než / Rovno (DINT)	1	0	–	1	0	–	10
	Větší než / Rovno (REAL)	57	1	–	31	1	–	14
	Menší než (INT)	1	0	–	1	0	–	10
	Menší než (DINT)	1	0	–	1	0	–	16
	Menší než (REAL)	58	1	–	36	1	–	14
	Menší než / Rovno (INT)	1	0	–	1	0	–	10
	Menší než / Rovno (DINT)	3	0	–	3	0	–	16
	Menší než / Rovno (REAL)	37	0	–	37	0	–	14
	Rozsah (INT)	2	1	–	2	1	–	13
	Rozsah (DINT)	2	1	–	2	1	–	22
	Rozsah (WORD)	1	0	–	1	0	–	13
Bitové operace	Logický AND	2	0	–	2	0	–	13
	Logický OR	2	0	–	2	0	–	13
	Logický Exclusive OR	1	0	–	1	0	–	13
	Logická inverze, NOT	1	0	–	1	0	–	10
	Posunutí bitu doleva	31	1	1.37	31	1	1.37	16
	Posunutí bitu doprava	28	0	3.03	28	0	3.03	16
	Rotace bitu doleva	25	0	3.12	25	0	3.12	16
	Rotace bitu doprava	25	0	4.14	25	0	4.14	16
	Pozice bitu	20	1	–	20	1	–	13
	Smazání bit	20	0	–	20	0	–	13
	Testování bitu	20	0	–	20	0	–	13
	Nastavení bitu	19	1	–	19	1	–	13
	Maskované porovnání (WORD)	52	0	–	52	0	–	25
	Maskované porovnání (DWORD)	50	0	–	49	0	–	25

Poznámky:

1. Doba (v mikrosekundách) platí pro verzi 7 softwaru Logicmaster 90-30/20/Micro pro CPU Model 351 a 352.
2. U tabulkových funkcí jsou inkrementy v jednotkách zadané délky; u funkcí s bitovými operacemi v mikrosekundách/bit; u funkcí přesunu dat v mikrosekundách/počet bitů nebo slov.
3. Doba povolení pro jednotky s jednoduchou délkou typu %R, %AI a %AQ.
4. Doba COMMREQ se měřila mezi CPU a HSC.
5. DOIO je doba k vygenerování hodnoty na výstupu diskrétního výstupního modulu.
6. Kde je více možností, výše uvedená doba představuje nejhorší možný případ.
7. U instrukcí, které mají hodnotu inkrementu, násobte inkrement (délkou –1) a přičtěte tuto hodnotu k základnímu času.

Tabulka A-2. Časování instrukcí, modely s vysokým výkonem – pokračování

Skupina funkcí	Funkce	Povoleno 350/351/36X	Zakázáno 350/351/36X	Inkrement 350/351/36X	Povoleno 352	Zakázáno 352	Inkrement 352	Velikost
Přesunutí dat	Přesunutí (INT)	2	0	0.41	2	0	0.41	10
	Přesunutí (BIT)	28	0	4.98	28	0	4.98	13
	Přesunutí (WORD)	2	0	0.41	2	0	0.41	10
	Přesunutí (REAL)	24	1	0.82	24	1	0.82	13
	Přesun bloku (INT)	2	0	–	2	0	–	28
	Přesun bloku (WORD)	4	4	–	3	0	–	28
	Přesun bloku (REAL)	41	0	–	41	0	–	13
	Smazání bloku	1	0	0.24	1	0	0.24	11
	Posunutí registru (BIT)	49	0	0.23	46	0	0.23	16
	Posunutí registru (WORD)	27	0	0.41	27	0	0.41	16
	Bitový sekvenční přepínač	38	22	0.02	38	22	0.02	16
	COMM_REQ	765	0	–	765	0	–	13
Tabulka	Přesunutí pole							
	INT	54	0	0.97	54	0	0.97	22
	DINT	54	0	0.81	54	0	0.81	22
	BIT	69	0	0.36	69	0	0.36	22
	BYTE	54	1	0.64	54	1	0.64	22
	WORD	54	0	0.97	54	0	0.97	22
	Hledat shodné							
	INT	37	0	0.62	37	0	0.62	19
	DINT	41	1	1.38	41	1	1.38	22
	BYTE	35	0	0.46	35	0	0.46	19
	WORD	37	0	0.62	37	0	0.62	19
	Hledat neshodné							
	INT	37	0	0.62	37	0	0.62	19
	DINT	38	0	2.14	38	0	2.14	22
	BYTE	37	0	0.47	37	0	0.47	19
	WORD	37	0	0.62	37	0	0.62	19
	Hledat větší než							
	INT	37	0	1.52	37	0	1.52	19
	DINT	39	0	2.26	39	0	2.26	22
	BYTE	36	1	1.24	36	1	1.24	19
	WORD	37	0	1.52	37	0	1.52	19
	Hledat větší než / Rovno							
	INT	37	0	1.48	37	0	1.48	19
	DINT	39	0	2.33	39	0	2.33	22
	BYTE	37	1	1.34	37	1	1.34	19
	WORD	37	0	1.48	37	0	1.48	19

Poznámky:

1. Doba (v mikrosekundách) platí pro verzi 7 softwaru Logicmaster 90-30/20/Micro pro CPU model 350 a 360.
2. U tabulkových funkcí jsou inkrementy v jednotkách zadané délky; u funkcí s bitovými operacemi v mikrosekundách/bit; u funkcí přesunu dat v mikrosekundách/počet bitů nebo slov.
3. Doba povolení pro jednotky s jednoduchou délkou typu %R, %AI a %AQ.
4. Doba COMMREQ se měřila mezi CPU a HSC.
5. DOIO je doba k vygenerování hodnoty na výstupu diskrétního výstupního modulu
6. Kde je více možností, výše uvedená doba představuje nejhorší možný případ.
7. U instrukcí, které mají hodnotu inkrementu, násobte inkrement (délkou –1) a přičtěte tuto hodnotu k základnímu času.

Tabulka A-2. Časování instrukcí, modely s vysokým výkonem – pokračování

Skupina funkcí	Funkce	Povoleno 350/351/36x	Zakázáno 350/351/36x	Inkrement 350/351/36x	Povoleno 352	Zakázáno 352	Inkrement 352	Velikost
	Hledat menší než							
	INT	37	0	1.52	37	0	1.52	19
	DINT	41	1	2.27	41	1	2.27	22
	BYTE	37	0	1.41	37	0	1.41	19
	WORD	37	0	1.52	37	0	1.52	19
	Hledat menší než / Rovno							
	INT	38	0	1.48	38	0	1.48	19
	DINT	40	1	2.30	40	1	2.30	22
	BYTE	37	0	1.24	37	0	1.24	19
	WORD	38	0	1.48	38	0	1.48	19
Převod	Převod na INT	19	1	–	19	1	–	10
	Převod na BCD–4	21	1	–	21	1	–	10
	Převod na Real	27	0	–	21	0	–	8
	Převod na WORD	28	1	–	30	1	–	11
	Celá část INT	32	0	–	32	0	–	11
	Celá část DINT	63	0	–	31	0	–	11
Řízení	Volání podprogramu	72	1	–	73	1	–	7
	Do I/O	114	1	–	115	1	–	13
	Algoritmus PID – ISA *	162	34	–	162	34	–	16
	Algoritmus PID – IND *	146	34	–	146	34	–	16
	Instrukce End	–	–	–	–	–	–	–
	Service Request							
	#6	22	1	–	22	1	–	10
	# 7 (Čtení)	75	1	–	75	1	–	10
	# 7 (Nastavení)	75	1	–	75	1	–	10
	#14	121	1	–	121	1	–	10
	#15	46	1	–	46	1	–	10
	#16	36	1	–	36	1	–	10
	#18	261	1	–	261	1	–	10
	#23	426	0	–	426	0	–	10
	#26//30**	2260	1	–	2260	1	–	10
	#29	20	0	–	20	0	–	10
	#43							
	Vnořené MCR/ENDMCR Kombinované	1	1	–	1	1	–	4
	Sekvenční záznamník událostí (SER)	Viz tabulka A-3	26.50	Viz tabulka A-3				

*Výše uvedené doby PID platí pro verzi 6.5 CPU 351.

*Service request #26/30 byl měřený s použitím vysokorychlostního čítače, 16-bodového výstupu v sestavě s 5 pozicemi.

Poznámky:
1. Doba (v mikrosekundách) platí pro verzi 7 softwaru Logicmaster 90-30/20/Micro pro CPU model 350 a 360.
2. U tabulkových funkcí jsou inkrementy v jednotkách zadané délky; u funkcí s bitovými operacemi v mikrosekundách/bit; u funkcí přesunu dat v mikrosekundách/počet bitů nebo slov.
3. Doba povolení pro jednotky s jednoduchou délkou typu %R, %AI a %AQ.
4. Doba COMMREQ se měřila mezi CPU a HSC.
5. DOIO je doba k vygenerování hodnoty na výstupu diskrétního výstupního modulu.
6. Kde je více možností, výše uvedená doba představuje nejhorší možný případ.
7. U instrukcí, které mají hodnotu inkrementu, násobte inkrement (délkou –1) a přičtěte tuto hodnotu k základnímu času.

Tabulka A-3. Časování funkčního bloku SER

Konfigurace	Příklad	Čas (µsec)
Bez proudu (zakázáno)	—	26.50
Sousedící		
8 kanálů	%I1—8	79.94
16 kanálů	%I1—16	80.58
24 kanálů	%I1—24	81.56
32 kanálů	%I1—32	81.73
8 + 8 sousedících kanálů	%I1—8 a %Q1—8	111.03
8 + 8 + 8 sousedících kanálů	%I1—8, %Q1—8 a %M1—8	143.38
8 + 8 + 8 + 8 sousedících kanálů	%I1—8, %Q1—8 a %M1—8 a %T1—8	175.79
Nesousedící	%I1, %M10, %Q3, atd.	
8 kanálů		299.64
16 kanálů		552.83
24 kanálů		806.35
32 kanálů		1059.85
Reset		
s 8 kanály	—	162.63
s 16 kanály	—	267.51
s 24 kanály	—	372.73
s 32 kanály	—	477.95

Poznámky: Když bude zadaná pozice se vstupním modulem, ke každému **sousedícímu** a **nesousedícímu** časování připočtěte 46 µsec.

Když se objeví spuštění, přičtěte dalších 29 µsec v případě použití formátu BCD nebo 148 usec v případě formátu Posix.

Doby uvedené pro reset jsou pro maximální velikost bufferu s 1024 vzorky. (Reset vynuluje všechny vzorky v bufferu vzorků.)

Velikosti instrukcí pro vysokovýkonná CPU

Velikost paměti je počet bajtů, který vyžaduje instrukce v aplikačním programu žebříkové logiky. CPU model 351 a 352 pro nejběžnější Booleovské funkce vyžadují tři bajty – viz tabulka A-3.

Tabulka A-4. Velikost instrukcí pro CPU 350—352, 360, 363 a 364

Funkce	Velikost
Bez operace	1
Výsledek funkce AND s obsahem zásobníku vložit na začátek zásobníku	1
Výsledek funkce OR s obsahem zásobníku vložit na začátek zásobníku	1
Zkopírovat začátek zásobníku	1
Vyjmout zásobník	1
Inicializace zásobníku	1
Návěští	5
Skok	5
Všechny ostatní instrukce	3
Funkční bloky viz tabulka A-2	–

Doby vykonávání Booleovských funkcí

Následující tabulka uvádí doby vykonávání cívek a kontaktů pro moduly CPU Series 90-30.

Tabulka A-5. Doby vykonávání Booleovských funkcí

Model CPU	Doba vykonávání 1,000 Booleovských kontaktů/cívek
Series model 350 ad 360	0.22 milisekund
Model 340E	0.3 milisekund
Model 331	0.4 milisekund
Model 313/323	0.6 milisekund
Model 311	18.0 milisekund

Dodatek

B

Interpretace chybových tabulek

PLC Series 90-30, Series 90-20 a Series 90 Micro udržují dvě tabulky, tabulku chyb I/O pro chyby generované v I/O zařízeních (včetně I/O regulátorů) a tabulku chyb PLC pro interní chyby PLC. Informace v této příloze umožňují při čtení těchto tabulek interpretovat formát struktury hlášení. Obě tabulky obsahují podobné informace.

- Tabulka chyb PLC obsahuje:
  - Lokalizaci chyby
  - Popis chyby
  - Datum a čas chyby
- Tabulka chyb I/O obsahuje:
  - Lokalizaci chyby
  - Adresu
  - Kategorii chyby
  - Typ chyby
  - Datum a čas chyby

Tabulka chyb PLC

Přístup do tabulky chyb PLC je přes programovací software. Informace o přístupu do tabulky chyb najdete v kontextové nápovědě, *v Návodu k používání programovacího softwaru Logicmaster 90 Series 90-30/20/Micro*, GFK-0466.

V následujícím schématu jsou vyznačena jednotlivá pole záznamu chyby pro výše zobrazenou chybu nesouladu konfigurace systému.

Zápis chyby nesouladu konfigurace systému vypadá následovně. (Všechna data jsou v hexadecimálním tvaru.)

Pole	Hodnota	Popis
Dlouhý/krátký	00	Tato chyba obsahuje 8 bajtů přídavných dat chyby.
Sestava	00	Hlavní sestava (sestava 0)
Pozice	03	Pozice 3
Úloha	44	
Chybová skupina	0B	Chyba nesouladu konfigurace systému
Chybová akce	03	FATÁLNÍ chyba
Chybový kód	01	

V následujících odstavcích je uvedený popis jednotlivých polí záznamu chyby. Jsou zde zahrnuté tabulky obsahují rozsah hodnot jednotlivých polí.

Indikátor Dlouhý/krátký

Tento bajt indikuje, jestli chyba obsahuje 8 bajtů nebo 24 bajtů přídavných dat chyby.

Typ	Kód	Přídavná data chyby
Krátký	00	8 bajtů
Dlouhý	01	24 bajtů

Doplňkové

Těchto šest bajtů jsou doplňkové bajty, které se používají k tomu, aby zápis v tabulce chyb PLC měl přesně stejnou délku jako zápis v tabulce chyb I/O.

Sestava

Číslo sestavy může být v rozsahu 0 až 7. Nula je číslo hlavní sestavy obsahující PLC. Sestavy 1 až 7 jsou přídavné sestavy připojené k PLC přes prodlužovací kabel.

Pozice

Číslo pozice může být v rozsahu 0 až 9. CPU PLC je vždy umístěno v pozici 1 hlavní sestavy (sestava 0).

Úloha

Číslo úlohy může být v rozsahu 0 až +65,535. Některé číslo úlohy technikovi PLC poskytuje další informace; úlohu je možno obvykle vynechat.

Chybová skupina PLC

Chybová skupina je nejvyšší klasifikace chyby. Identifikuje obecnou kategorii chyby. Text s popisem chyby, který zobrazuje software Logicmaster 90-30/20/Micro, je založený na chybové skupině a chybových kódech.

Tabulka B-1 uvádí seznam možných chybových skupin v tabulce chyb PLC.

Poslední nemaskovatelná chybová skupina, ***Doplňkové chybové kódy PLC***, byla deklarovaná pro zpracování nových chybových stavů v systému, aniž by PLC muselo specificky chybové kódy znát. Do této skupiny patří všechny neznámé kódy alarmu typu PLC.

Tabulka B-1. Chybové skupiny PLC

Číslo skupiny		Název skupiny	Význam chyby
Dekadicky	**Hexadecimálně**		
1	1	Ztráta nebo chybějící sestava	Fatální
4	4	Ztráta nebo chybějící přídavný modul	Diagnostická
5	5	Přidání nebo přespočetná sestava	Diagnostická
8	8	Přidání nebo přespočetný přídavný modul	Diagnostická
11	B	Nesoulad konfigurace systému	Fatální
12	C	Chyba systémové sběrnice	Diagnostická
13	D	Hardwarová porucha CPU PLC	Fatální
14	E	Nefatální hardwarová chyba modulu	Diagnostická
16	10	Chyba softwaru přídavného modulu	Diagnostická
17	11	Chyba kontrolního součtu programového bloku	Fatální
18	12	Signál nízkého napětí baterie	Diagnostická
19	13	Překročení konstantního času cyklu	Diagnostická
20	14	Systémová tabulka chyb PLC plná	Diagnostická
21	15	Tabulka chyb I/O plná	Diagnostická
22	16	Chyba uživatelské aplikace	Diagnostická
–	–	Doplňkové kódy chyb PLC	Podle specifikace
128	80	Chyba systémové sběrnice	Fatální
129	81	Po zapnutí napájení není uživatelský program	Informační
130	82	Zjištěno poškození uživatelské RAM	Fatální
132	84	Chyba přístupového hesla	Informační
135	87	Porucha softwaru CPU PLC	Fatální
137	89	Porucha sekvence ukládání PLC	Fatální

Chybová akce

S každou chybou může být spojena jedna ze tří činností. Tyto chybové akce jsou pevně spojené s PLC Series 90-30 a nelze je uživatelsky měnit.

Tabulka B-2. Chybové akce PLC

Závažnost chyby	Kroky, které provede CPU	Kód
Informační	Záznam chyby do tabulky chyb	1
Diagnostická	Záznam chyby do tabulky chyb Nastavení adres chyb	2
Fatální	Záznam chyby do tabulky chyb Nastavení adres chyb Přechod do režimu STOP	3

Chybový kód

Chybový kód popisuje chybu podrobněji. Každá chybová skupina má svůj soubor chybových kódů. Tabulka B-3 uvádí chybové kódy pro chybovou skupinu softwaru PLC (skupina 87H).

Tabulka B-3. Chybové kódy alarmu pro chyby softwaru CPU PLC

Dekadicky	Hexadecimálně	Název
20	14	Poškozený obsah programové paměti PLC
39	27	Poškozený obsah programové paměti PLC
82	52	Chyba interní komunikace
90	5A	Vypnutí požadováno uživatelem
Všechny ostatní		Interní systémová chyba CPU PLC

Tabulka B-4 uvádí chybové kódy pro všechny ostatní chybové skupiny.

Tabulka B-4. Chybové kódy alarmu pro chyby PLC

Dekadicky	Hexadecimálně	Název
Chybový kód PLC ztráty skupiny přídavného modulu (4)		
44	C	Neprovedl se softwarový reset přídavného modulu
45	D	Neprovedl se softwarový reset přídavného modulu
255	FF	Neprovedla se komunikace přídavného modulu
79	4F	Ztráta dceřinné karty
Chybové kódy skupiny resetu, přidání přídavného modulu nebo přespočetného přídavného modulu (8)		
2	2	Restart modulu dokončený
04	4	Přidání dceřinné karty
05	5	Reset dceřinné karty
	Všechny ostatní	Reset, přidání přídavného modulu nebo přespočetný přídavný modul
Chybové kódy skupiny chyb softwaru přídavného modulu (10 hex)		
1	1	Nepodporovaný typ karty
2	2	COMREQ – plná schránka pro výstupní zprávy začínající na COMREQ
3	3	COMREQ – schránka pro odezvu je plná
5	5	Interní komunikace s PLC; Ztráta požadavku
11	B	Chyba zdroje (přiřazení, přetečení tabulky, atd.)
13	D	Chyba uživatelského programu
401	191	Poškození softwaru modulu; Nutné nové načtení
Chybové kódy skupiny nesouladu konfigurace systému (B hex)		
8	8	Nesoulad analogového rozšíření
10	A	Nepodporovaná funkce
23	17	Program překračuje meze paměti
58	3A	Nesoulad dceřinné karty
Chybové kódy skupiny chyb systémové sběrnice (C hex)		
Všechny ostatní		Chyba systémové sběrnice
Chybové kódy skupiny kontrolního součtu bloku programu (11 hex)		
3	3	Chyba kontrolního součtu programu nebo programového bloku
Chybové kódy signálu nízkého napětí baterie		
0	0	Závada baterie na CPU PLC nebo na jiném modulu
1	1	Nízké napětí baterie na CPU PLC nebo na jiném modulu
Chybové kódy skupiny chyb uživatelské aplikace (16 hex)		
2	2	Hlídací časovač PLC překročil čas
5	5	COMREQ – Režim ČEKÁNÍ není u tohoto povelu k dispozici
6	6	COMREQ – Chybné ID úlohy
7	7	Přetečení zásobníkové paměti aplikace
Chybové kódy skupiny chyb systémové sběrnice (80 hex)		
1	1	Operační systém
Chybové kódy skupiny poškození obsahu RAM při zapnutí napájení (82 hex)		
1	1	Poškozený obsah uživatelské RAM při zapnutí napájení
2	2	Zjištěný nepřípustný Booleovský operační kód.
3	3	PLC_ISCP_PC_OVERFLOW
4	4	PRG_SYNTAX_ERR
Chybové kódy chyb hardwaru CPU PLC (D hex)		
Všechny kódy		Hardwarová porucha CPU PLC

Přídavná data chyby

Toto pole obsahuje podrobnosti záznamu chyby. Mohou se vyskytovat například následující data:

Skupina poškození obsahu RAM:

Čtyři chybové kódy ve skupině nesouladu konfigurace systému dávají dodatečná data chyby:

Tabulka B-5. Chybová data PLC – Zjištěný nepřípustný Booleovský operační kód

Přídavná data chyby	Nesoulad čísla modelu
[0]	Obsah registru chyb ISCP
[1]	Chybný OPCODE
[2,3]	Programový čítač ISCP
[4,5]	Číslo funkce

V případě chyby RAM na CPU PLC (jedna z chyb se hlásí jako hardwarová chyba CPU PLC) se adresa chyby uloží v prvních čtyřech bajtech pole.

Hardwarová porucha CPU PLC (chyba RAM):

Časová značka chyby PLC

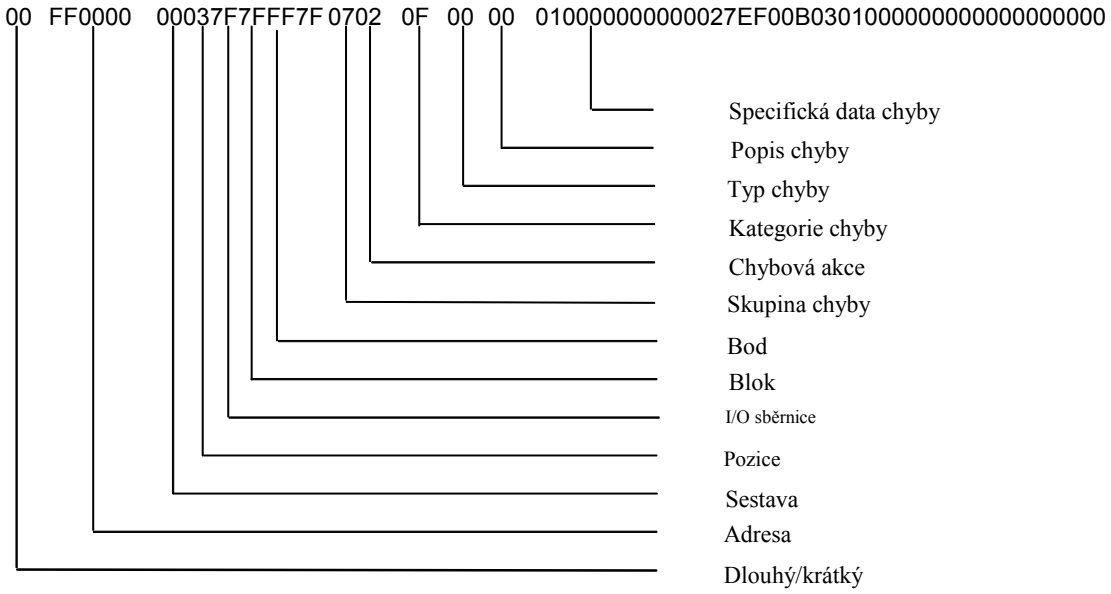
Šestibajtová časová značka je hodnota systémových hodin, kdy CPU PLC provedlo zaznamenání chyby. (Hodnoty jsou kódované ve formátu BCD.)

Tabulka B-6. Časová značka chyby PLC

Číslo bajtu	Popis
1	Sekundy
2	Minuty
3	Hodiny
4	Den v měsíci
5	Měsíc
6	Rok

Tabulka chyb I/O

V následujícím schématu jsou vyznačeny hexadecimální informace zobrazené v jednotlivých polích záznamu chyby.



V následující odstavcích je uvedený popis jednotlivých polí v tabulce chyb I/O. Jsou zde zahrnuté tabulky obsahují rozsah hodnot jednotlivých polí.

Indikátor Dlouhý/krátký

Tento bajt indikuje, jestli chyba obsahuje 5 bajtů nebo 21 bajtů specifických dat chyby.

Tabulka B-7. Bajt indikace formátu tabulky chyb I/O

Typ	Kód	Specifická data chyby
Krátký	02	5 bajtů
Dlouhý	03	21 bajtů

Adresa

Adresa je tří-bajtová adresa obsahující typ a umístění (nebo offset) paměti I/O v této paměti, která odpovídá bodu, kde se vyskytla chyba. Nebo když se vyskytne chyba bloku Genius nebo integrálního analogového bloku, adresa bude ukazovat na první bod bloku, kde se chyba vyskytla.

Tabulka B-8. Adresa I/O

Byte	Popis	Rozsah
0	Typ paměti	0 – FF
1–2	Offset	0 – FF

Bajt typu paměti může být některá z následujících hodnot.

Tabulka B-9. Typ paměti adresy I/O

Název	Hodnota (Hexadecimálně)
Analogový vstup	0A
Analogový výstup	0C
Analogová skupina	0D
Diskrétní vstup	10 nebo 46
Diskrétní výstup	12 nebo 48
Diskrétní skupina	1F

Adresa chyby I/O

Adresa chyby I/O je šesti-bajtová adresa obsahující adresu sestavy, pozice, sběrnice a bodu I/O, který vygeneroval chybu. Adresa bodu je slovo; všechny ostatní adresy mají délku jeden bajt. Těchto pět hodnot při chybě nemusí existovat.

Když se objeví adresa chyby I/O, která nebude obsahovat všech pět adres, na adrese se objeví 7F hex jako indikace, kde končí platnost. Pokud se například 7F objeví v bajtu sběrnice, pak chyba bude chyba modulu. Význam mají pouze hodnoty sestavy a pozice.

Sestava

Číslo sestavy může být v rozsahu 0 až 7. Nula je číslo hlavní sestavy, tedy té, která obsahuje PLC. Sestavy 1 až 7 jsou přídavné sestavy.

Pozice

Číslo pozice může být v rozsahu 0 až 9. CPU PLC je vždy umístěný v pozici 1 hlavní sestavy (sestava 0).

Bod

Bod může být v rozsahu 1 až 1024 (dekadicky). Informuje, na kterém bodu v bloku se vyskytla chyba, když chyba bude typu vztahující se k bodu.

Chybová skupina I/O

Chybová skupina je nejvyšší klasifikace chyby. Identifikuje obecnou kategorii chyby. Text s popisem chyby, který zobrazuje software Logicmaster 90-30/20/Micro, je založený na chybové skupině a chybových kódech.

Tabulka B-10 uvádí seznam možných chybových skupin v tabulce chyb I/O. Čísla skupiny menší než 80 (Hex) jsou maskovatelné chyby.

Poslední nemaskovatelná chybová skupina, ***Doplňkové chybové kódy I/O***, byla vytvořena pro zpracování nových chybových stavů v systému, aniž by PLC muselo specificky chybové kódy znát. Všechny neznámé kódy alarmu typu I/O patří do této skupiny.

Tabulka B-10. Chybové skupiny I/O

Číslo skupiny	Název skupiny	Chybová akce
3	Ztráta nebo chybějící modul I/O	Diagnostická
7	Přidání nebo přespočetný modul I/O	Diagnostická
9	Chyba IOC nebo I/O sběrnice	Diagnostická
A	Chyba I/O modulu	Diagnostická
–	Doplňkové kódy chyb I/O	Podle specifikace

Chybové akce I/O

Chybová akce udává, jaký krok má CPU PLC provést, když se vyskytne chyba. Tabulka B-1 uvádí seznam možných chybových akcí.

Tabulka B-11. Chybové akce I/O

Závažnost chyby	Kroky, které provede CPU	Kód
Informační	Záznam chyby do tabulky chyb	1
Diagnostická	Záznam chyby do tabulky chyb Nastavení adres chyb	2
Fatální	Záznam chyby do tabulky chyb Nastavení adres chyb Přechod do režimu STOP	3

Specifická data chyby I/O

Záznam v tabulce chyb I/O může obsahovat až 5 bajtů specifických dat chyby I/O.

Symbolická specifická data chyby

Tabulka B-12 uvádí data, která jsou zapotřebí pro konfiguraci obvodu bloku.

Tabulka B-12. Specifická data chyby I/O

Dekadické číslo	Hexadecimální kód	Popis
Konfigurace obvodu		
	1 2 3	Obvod je vstup - třístavový Obvod je vstup Obvod je výstup

Kroky při specifické chybě

Vynucené/nevynucené chyby obvodu se hlásí jako informativní chyby. Všechny ostatní jsou diagnostické nebo fatální.

Chyby nesouladu čísla modelu, nesouladu typu I/O a neexistujícího I/O modulu se hlásí do tabulky chyb PLC pod skupinou Nesoulad konfigurace systému. Nehlásí se do tabulky chyb I/O.

Časová značka chyby I/O

Šestibajtová časová značka je hodnota systémových hodin, kdy CPU PLC provedlo zaznamenání chyby. Hodnoty jsou kódované ve formátu BCD.

Tabulka B-13. Časová značka chyby I/O

Číslo bajtu	Popis
1	Sekundy
2	Minuty
3	Hodiny
4	Den v měsíci
5	Měsíc
6	Rok

Dodatek C

Mnemotechnické zkratky instrukcí

V režimu Zobrazení programu/Editování můžete programovací instrukci rychle zapsat nebo vyhledat tak, že napíšete znak & a za ním mnemotechnickou zkratku instrukce. U některých instrukcí můžete také zadat adresu nebo zkratku, návěští nebo lokální adresu.

Tento dodatek uvádí mnemotechnické zkratky programovacích instrukcí v programovacím softwaru Logicmaster 90-30/20/Micro. Úplné mnemotechnické zkratky jsou uvedené ve sloupci 3 této tabulky a nejkratší způsob zápisu, který můžete provést, je uvedený ve sloupci 4.

Nápovědu na obrazovce pro tyto mnemotechnické zkratky můžete zobrazit kdykoliv během programování stisknutím tlačítek ALT a I.

Funkční skupina	**Instrukce**	**Mnemotechnická zkratka**						
		Všechny	**INT**	**DINT**	**BIT**	**BYTE**	**WORD**	**REAL**
Kontakty	Každý kontakt	&CON	&CON					
	Normálně rozepnutý kontakt	&NOCON	&NOCON					
	Normálně sepnutý kontakt	&NCCON	&NCCON					
	Pokračovací kontakt	&CONC	&CONC					
Cívky	Každá cívka	&COI	&COI					
	Normálně rozepnutá cívka	&NOCOI	&NOCOI					
	Negovaná cívka	&NCCOI	&NCCOI					
	Cívka s kladným přechodem	&PCOI	&PCOI					
	Cívka se záporným přechodem	&NCOI	&NCOI					
	Cívka SET	&SL	&SL					
	Cívka RESET	&RL	&RL					
	Retentivní cívka SET	&SM	&SM					
	Retentivní cívka RESET	&RM	&RM					
	Retentivní cívka	&NOM	&NOM					
	Negovaná retentivní cívka	&NCM	&NCM					
	Pokračovací cívka	&COILC	&COILC					
Spoje	Horizontální spoj	&HO	&HO					
	Vertikální spoj	&VE	&VE					
Časovače	Časovač zpoždění při zapnutí	&ON	&ON					
	Časovač uplynulého času	&TM	&TM					
	Časovač zpoždění při vypnutí	&OF	&OF					
Čítače	Vzestupný čítač	&UP	&UP					
	Sestupný čítač	&DN	&DN					

Funkční skupina	Instrukce	Mnemotechnická zkratka							
		Všechny	**BCD-4**	**INT**	**DINT**	**BIT**	**BYTE**	**WORD**	**REAL**
Matematické	Sčítání	&AD		&AD_I	&AD_DI				&AD_R
	Odečítání	&SUB		&SUB_I	&SUB_DI				&SUB_R
	Násobení	&MUL		&MUL_I	&MUL_DI				&MUL_R
	Dělení	&DIV		&DIV_I	&DIV_DI				&DIV_R
	Modulo	&MOD		&MOD_I	&MOD_DI				&MOD_R&SQ_R
	Druhá odmocnina	&SQ		&SQ_I	&SQ_DI				
	Sinus	&SIN							
	Kosinus	&COS							
	Tangens	&TAN							
	Inverzní sinus	&ASIN							
	Inverzní kosinus	&ACOS							
	Inverzní tangens	&ATAN							
	Dekadický logaritmus	&LOG							
	Přirozený logaritmus	&LN							
	Mocnica e	&EXP							
	Mocnina x	&EXPT							
Relační	Rovná se	&EQ		&EQ_I	&EQ_DI				&EQ_R
	Nerovná se	&NE		&NE_I	&NE_DI				&NE_R
	Větší než	&GT		&GT_I	&GT_DI				&GT_R
	Větší nebo rovno	&GE		&GE_I	&GE_DI				&GE_R
	Menší než	&LT		&LT_I	&LT_DI				&LT_R
	Menší nebo rovno	&LE		&LE_I	&LE_DI				&LE_R
Bitové operace	AND	&AN						&AN_W	
	OR	&OR						&OR_W	
	Exclusive OR	&XO						&XO_W	
	NOT	&NOT						&NOT_W	
	Posunutí doleva	&SHL						&SHL_W	
	Posunutí doprava	&SHR						&SHR_W	
	Rotace doleva	&ROL						&ROL_W	
	Rotace doprava	&ROR						&ROR_W	
	Testování bitu	&BT						&BT_W	
	Nastavení bitu	&BS						&BS_W	
	Smazání bit	&BCL						&BCL_W	
	Pozice bitu	&BP						&BP_W	
	Maskované porovnání	&MCMP						&MCM_W	
Převod	Převod na celé číslo	&TO_INT	&TO_INT_BCD4	&MOV					
	Převod na celé číslo s dvojnásobnou délkou	&TO_DINT		&BLKM					&BCD4_R
	Převod na BCD-4	&BCD4		&BLKC	&TO_REAL_DI			&TO_REAL_W	
	Převod na REAL	&TO_REAL		&SHF					
	Převod na WORD	&TO_W		&BI					
	Celá část celého čísla	&TRINT		&COMMR					
	Celá část celého čísla s dvojitou délkou	&TRDINT							

Funkční skupina	Instrukce	Mnemotechnická zkratka						
		Všechny	INT	DINT	BIT	BYTE	WORD	REAL
Přesunutí dat	Přesunutí Přesunutí bloku Smazání bloku Posunutí registr Bitový sekvenční přepínač Požadavek na komunikaci	&MOV &BLKM &BLKC &SHF &BI &COMMR	&MOV_I &BLKM_I		&MOV_BI &SHF_BI		&MOV_W &BLKM_W &AR_W	&MOV_R &BLKM_R
Tabulka	Přesunout pole Hledat shodné Hledat neshodné Hledat větší než Hledat větší nebo shodné Hledat menší než Hledat menší nebo shodné	&AR &SRCHE &SRCHN &SRCHGT &SRCHGE &SRCHLT &SRCHLE	&AR_I &SRCHE_I &SRCHN_I &SRCHGT_I &SRCHGE_I &SRCHLT_I &SRCHLE_I	&AR_DI &SRCHE_DI &SRCHN_DI &SRCHGT_DI &SRCHGE_DI &SRCHLT_DI &SRCHLE_DI	&AR_BI	&AR_BY &SRCHE_BY &SRCHN_BY &SRCHGT_BY &SRCHGE_BY &SRCHLT_BY &SRCHLE_BY	&AR_W &SRCHE_W &SRCHN_W &SRCHGT_W &SRCHGE_W &SRCHLT_W &SRCHLE_W_	
Řízení	Volání podprogramu Do I/O SER Algoritmus PID – ISA Algoritmus PID – IND SFC Reset Konec Výklad logické příčky Systémový Services Request Hlavní řídicí relé Konec hlavního řídicího relé Vnořované hlavní řídicí relé Vnořovaný konec nadřízeného řídicího relé Skok Vnořovaný skok Návěští Vnořované návěští	&CA &DO &SER &PIDIS &PIDIN &SFCR &END &COMME &SV &MCR &ENDMCR &MCRN &ENDMCRN &JUMP &JUMPN &LABEL &LABELN							

Dodatek D

Funkce tlačítek

Tento dodatek uvádí seznam funkcí tlačítek, která jsou aktivní v prostředí softwaru. Chcete-li zobrazit tuto informaci na obrazovce programovacího zařízení, stisknutím tlačítka ALT-K vyvoláte nápovědu tlačítek.

Sekvence tlačítek	Popis	Sekvence tlačítek	Popis
Tlačítka, která je možno použít v celém softwaru			
ALT-A	Zrušení	CTRL-Break	Ukončení programu
ALT-C	Vymazání pole	Esc	Návrat z vnoření
ALT-M	Změna režimu programu	CTRL-Home	Obsah předchozího povelového řádku
ALT-R	Změna stavu PLC Run/Stop .	CTRL-End	Obsah následujícího povelového řádku
ALT-E	Přepínání stavové oblasti	CTRL- ←	Kurzor doleva uvnitř pole
ALT-J	Přepínání povelového řádku	CTRL-→	Kurzor doprava uvnitř pole
ALT-L	Uvést seznam souborů v adresáři	CTRL-D	Dekrementace adresy
ALT-P	Vytisknout obrazovku	CTRL-U	Inkrementace adresy
ALT-H	Nápověda	Tab	Změna/inkrementace obsahu pole
ALT-K	Nápověda k tlačítku	Shift-Tab	Změna/dekrementace obsahu pole
ALT-I	Nápověda k mnemotechnické zkratce instrukce	Enter	Přijmout obsah pole
ALT-N	Přepínání voleb zobrazení	CTRL-E	Zobrazení poslední systémové chyby
ALT-T	Spuštění režimu Teach	F12 nebo Keypad -	Přepínání diskrétní adresy
ALT-Q	Zastavení režimu Teach	F11 nebo Keypad *	Přepsání diskrétní adresy
ALT-n	Playback souboru n (n = 0 až 9)		
Tlačítka, která jsou k dispozici pouze v editoru programu			
ALT-B	Přepínání zvonku textového editoru	Keypad +	Přijmout příčku
ALT-D	Smazat prvek příčky/Smazat příčku	Enter	Přijmout příčku
ALT-S	Uložit blok do PLC a na disk	CTRL-PgUp	Předchozí příčka
ALT-X	Zobrazit úroveň vnoření	CTRL-PgDn	Následující příčka
ALT-U	Aktualizovat disk	~	Horizontální můstek
ALT-V	Okno proměnné tabulky	|	Vertikální můstek
ALT-F2	Přechod na referenční tabulku operandu	Tab	Přechod na další pole operandu
Speciální tlačítka			
ALT-O	Přepis hesla Možno použít pouze na obrazovce Heslo v konfiguračním softwaru.		

Karta Nápovědy na následující stránce obsahuje seznam nápověd ke tlačítkům a také text nápovědy mnemotechnických zkratek k instrukcím pro software Logicmaster 90-30/20/Micro. Tato karta je vytisknuta trojmo a pro snazší vyjmutí z manuálu je perforovaná.

Na této stránce vytiskněte 1. stranu GFJ-055D.

Na této stránce vytiskněte 2. stranu GFJ-055D.

Dodatek

E

Používání čísel s pohyblivou desetinnou tečkou

Při používání čísel s pohyblivou desetinnou tečkou je nutno mít na zřeteli několik věcí. Tyto obecné úvahy jsou popisované v první části. Instrukce pro zápis a zobrazení čísel s pohyblivou desetinnou tečkou najdete na straně E-5 a následujících.

Poznámka

Funkce pohyblivé desetinné tečky podporují ***pouze*** CPU řady 35x a 36x, verze 9 nebo pozdější a všechny verze CPU352.

Čísla s pohyblivou desetinnou tečkou

Programovací software umožňuje editování, zobrazování, ukládání a vyvolávání čísel s reálnou hodnotou. Některé funkce pracují s čísly s pohyblivou desetinnou tečkou. Chcete-li však v programovacím softwaru použít čísla s pohyblivou desetinnou tečkou, musíte mít CPU řady 35x nebo 36x (viz poznámka výše). Čísla s pohyblivou desetinnou tečkou jsou reprezentována v dekadickém vědeckém formátu zápisu a zobrazují se na šest platných číslic.

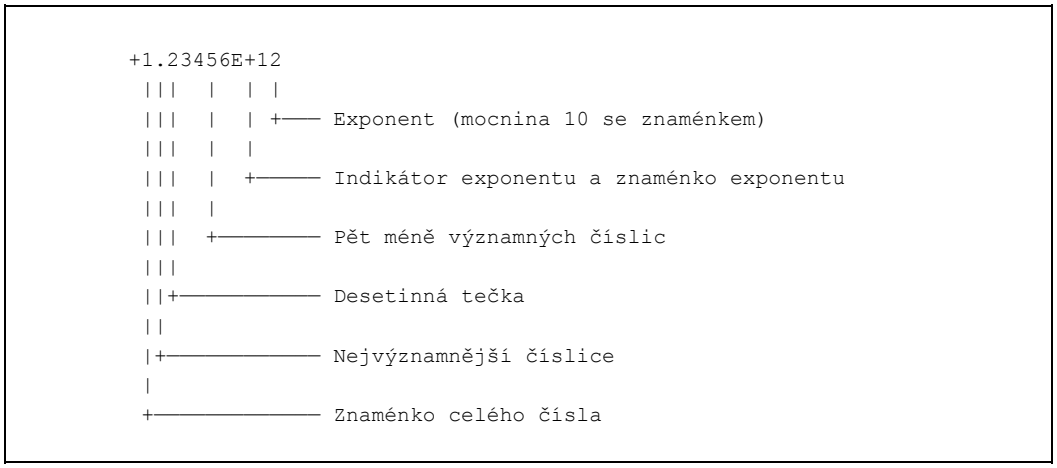
Poznámka

V tomto manuálu se termín "plovoucí desetinná tečka" používá univerzálně k popisu vlastnosti zobrazení/zápisu čísla s plovoucí desetinnou tečkou programovacího softwaru.

Používá se následující formát. Pro čísla v rozsahu 9999999 až 0.0001 zobrazení nemá žádný exponent a má až šest nebo sedm platných číslic. Například:

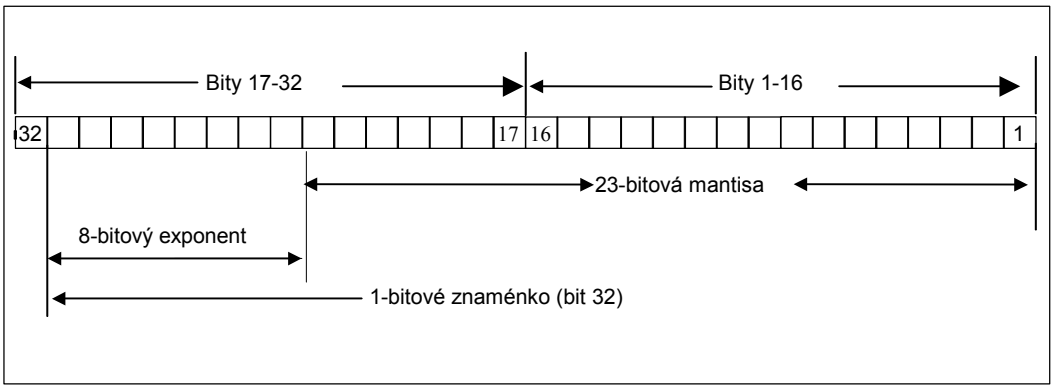
Zapsáno	Zobrazeno	Popis
.000123456789	+.0001234567	Deset číslic, šest nebo sedm platných
–12.345e-2	–.1234500	Sedm číslic, šest nebo sedm platných
1234	+1234.000	Sedm číslic, šest nebo sedm platných

Mimo výše uvedený rozsah se zobrazí pouze šest platných číslic a zobrazení bude vypadat následovně:

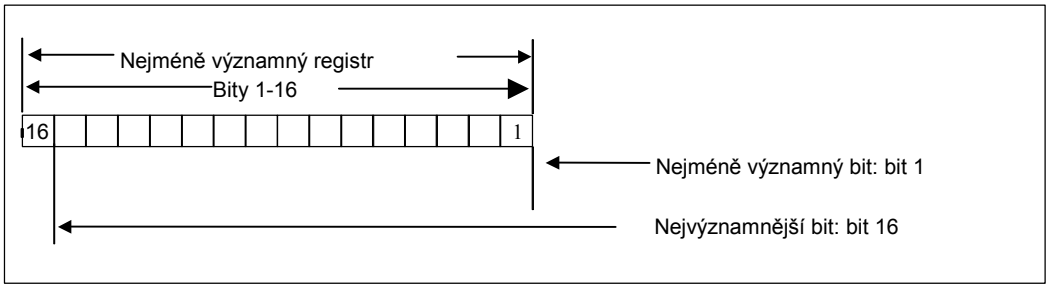


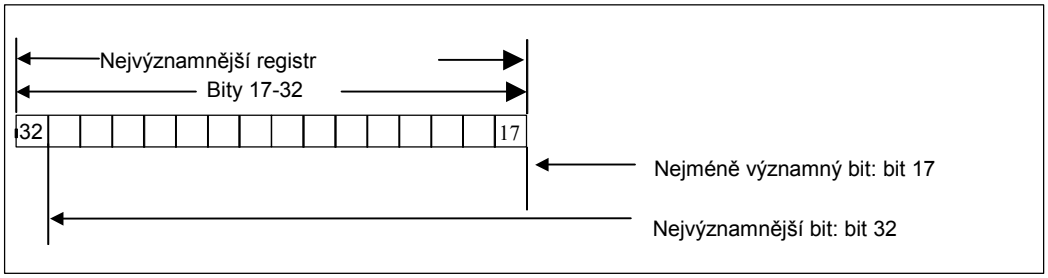
Interní formát čísel s pohyblivou desetinnou tečkou

Čísla v pohyblivé řádové tečce se ukládají ve standardním formátu IEEE s jednoduchou přesností. Tento formát vyžaduje 32 bitů, které obsadí dva sousedící 16-bitové registry PLC. Kódování bitů je uvedeno v následujícím diagramu.



Použití registru jedním číslem s pohyblivou desetinnou tečkou je uvedeno v následujícím diagramu. Pokud v tomto diagramu bude číslo s pohyblivou desetinnou tečkou zabírat například registry R5 a R6, R5 bude nejméně významný registr a R6 nejvýznamnější registr.





Hodnoty čísel s pohyblivou desetinnou tečkou

K vypočítání hodnoty čísla s pohyblivou desetinnou tečkou z binárního čísla uloženého ve dvou registrech použijte následující tabulku.

Exponent (e)	Mantisa (f)	Hodnota čísla s plovoucí desetinnou tečkou
255	Nenulová	Není platné číslo (NaN)
255	0	$-1^s * \infty$
0 < e < 255	Jakákoliv hodnota	$-1^s * 2^{e-127} * 1.f$
0	Nenulová	$-1^s * 2^{-126} * 0.f$
0	0	0

f = mantisa. Mantisa je binární zlomek.
e = exponent. Exponent je celé číslo E takové, že E+127 je mocnina 2, kterou se mantisa musí násobit, aby se získala hodnota s pohyblivou desetinnou tečkou.
s = znaménkový bit.
* = operátor násobení

Uvažujme například číslo s pohyblivou desetinnou tečkou 12,5. Binární reprezentace čísla s pohyblivou tečkou IEEE bude:

```
01000001 01001000 00000000 00000000
```

nebo 41480000 hex. Nejvýznamnější bit (znaménkový bit) je nula (s=0). Dalších osm nejvýznamnějších bitů jsou 10000010 nebo 130 dekadicky (e=130).

Mantisa je uložená jako desetinné binární číslo s desetinnou tečkou před 23 nejvýznamnějšími bity. Proto nejvýznamnější bit v mantise je násobek 2^{-1}, další nejvýznamnější bit je násobek 2^{-2}, a tak dále až nejméně významný bit, který je násobek 2^{-23}. Konečných 23 bitů (mantisa) bude:

```
1001000 00000000 00000000
```

Hodnota mantisy pak bude .5625 (to je , $2^{-1} + 2^{-4}$).

Protože e > 0 a e < 255, v tabulce výše použijeme třetí vztah:

$$\begin{aligned} \text{číslo} &= -1^s * 2^{e-127} * 1.f \\ &= -1^0 * 2^{130-127} * 1.5625 \\ &= 1 * 2^3 * 1.5625 \\ &= 8 * 1.5625 \\ &= 12.5 \end{aligned}$$

Jak vidíte, výše uvedená binární reprezentace je správná.

Rozsah čísel, která je možno uložit v tomto formátu je od ± 1.401298E–45 až ± 3.402823E+38 a číslo nula.

Zápis a zobrazení čísel s pohyblivou desetinnou tečkou

Do mantisy je možno zapsat a uložit až šest nebo sedm platných číslic; programovací software však zobrazí pouze prvních šest těchto číslic. Před mantisou může být kladné nebo záporné znaménko. Pokud nebude zapsáno žádné znaménko, předpokládá se, že číslo s pohyblivou desetinnou tečkou je kladné.

Pokud bude zapsaný exponent, musí před ním být písmeno **E** nebo **e** a mantisa musí obsahovat desetinnou tečku, aby nedošlo k záměně s hexadecimálním číslem. Před exponentem může být znaménko; pokud však žádné znaménko nebude, bude se předpokládat, že je kladné. Pokud nebude zapsaný žádný exponent, bude se předpokládat, že je nula. V číslech s pohyblivou desetinnou tečkou nesmí být žádná mezera.

Pro snadné používání je na povelovém řádku a v zápisu do pole dat přípustných několik formátů. Tyto formáty zahrnují celé číslo, dekadické číslo nebo dekadické číslo následované exponentem. Jakmile se data zapíšou a stiskne se tlačítko **Enter**, tato čísla se převedou na standardní tvar pro zobrazení.

Příklady platných zápisů čísla s pohyblivou desetinnou tečkou jsou uvedena následující tabulce.

Zapsáno	Zobrazeno
250	+250,0000
+4	+4.000000
–2383019	–2383019.
34.	+34.00000
–.0036209	–.003620900
12.E+9	+1.20000E+10
–.0004E–11	–4.00000E–15
731.0388	+731.0388
99.20003e–29	+9.92000E–28

Příklady neplatných zápisů čísla s pohyblivou desetinnou tečkou můžou být:

Neplatný zápis	Výklad
–433E23	Chybí desetinná tečka
10e-19	Chybí desetinná tečka
10.e19	Mantisa nesmí obsahovat mezery mezi číslicemi nebo znaky. Tento zápis se přijme jako 10.10 a zobrazí se chyba.
4.1e19	Exponent nesmí obsahovat mezery mezi číslicemi nebo znaky. Tento zápis se přijme jako 4.1e0 a zobrazí se chyba.

Chyby v číslech s pohyblivou desetinnou tečkou a operace

Pokud funkce REAL vygeneruje číslo větší než 3.402823E+38 nebo menší než -3.402823E+38, u CPU 352 dojde k přetečení. U všech ostatních modelů 90-30, které podporují operace s pohyblivou desetinnou tečkou, je rozsah větší 2^{16} nebo menší než $-2^{16.}$ Pokud číslo bude rozsah přesahovat, výstup ok funkce se nastaví do stavu OFF; a výsledek se nastaví na kladné nekonečno (u čísla většího než 3.402823E+38 v případě CPU 352 nebo 2^{16} u všech ostatních modelů) nebo na záporné nekonečno (u čísla menšího než –3.402823E+38 nebo -2^{16} u všech ostatních modelů). Místo výskytu lze určit testováním ok výstupu.

POS_INF	= 7F800000h	– IEEE hexadecimální reprezentace kladného nekonečna.
NEG_INF	= FF800000h	– IEEE hexadecimální reprezentace záporného nekonečna.

Poznámka

Pokud budete používat softwarově pohyblivou desetinnou tečku (všechny modely s možností operace s pohyblivou desetinnou tečkou kromě CPU 352), při ±1.175494E–38 se čísla zaokrouhlí na nulu (0).

Pokud se nekonečno vzniklé přetečením použije jako operand v jiné funkci REAL, může nastat nedefinovatelný výsledek. Tento nedefinovaný výsledek se nazývá NaN (nenumerická hodnota). Například výsledek součtu kladného a záporného nekonečna není definován. Když se vyvolá funkce ADD_REAL s kladným a záporným nekonečnem jako jejími operandy, vytvoří jako výsledek NaN.

U CPU 352 každá funkce REAL, která je schopná vytvořit NaN, vytvoří speciální NaN, které identifikuje funkci:

NaN_SW	= FFFFFFFFh	– Softwarový NaN s pohyblivou desetinnou tečkou
NaN_ADD.	= 7F81FFFFh	– Hodnota chyby přičtení reálného čísla v hexadecimálním tvaru
NaN_SUB	= 7F81FFFFh	– Hodnota chyby odečtení reálného čísla v hexadecimálním tvaru
NaN_MUL	= 7F82FFFFh	– Hodnota chyby násobení reálného čísla v hexadecimálním tvaru
NaN_DIV	= 7F83FFFFh	– Hodnota chyby dělení reálného čísla v hexadecimálním tvaru
NaN_SQRT	= 7F84FFFFh	– Hodnota chyby druhé odmocniny reálného čísla v hexadecimálním tvaru
NaN_LOG	= 7F85FFFFh	– Hodnota chyby logaritmu reálného čísla v hexadecimálním tvaru
NaN_POW0	= 7F86FFFFh	– Hodnota chyby exponentu reálného čísla v hexadecimálním tvaru
NaN_SIN	= 7F87FFFFh	– Hodnota chyby sinu reálného čísla v hexadecimálním tvaru
NaN_COS	= 7F88FFFFh	– Hodnota chyby cosinu reálného čísla v hexadecimálním tvaru
NaN_TAN	= 7F89FFFFh	– Hodnota chyby tangensu reálného čísla v hexadecimálním tvaru
NaN_ASIN	= 7F8AFFFFh	– Hodnota chyby inverzního sinu reálného čísla v hexadecimálním tvaru
NaN_ACOS	= 7F8BFFFFh	– Hodnota chyby inverzního cosinu reálného čísla v hexadecimálním tvaru
NaN_BCD	= 7F8CFFFFh	– Chyba převodu BCD-4 na reálné číslo.
REAL_INDEF	= FFC00000h	– Reálné nekonečno, chyba dělení nuly nulou.

Všechna ostatní CPU, která podporují operace s pohyblivou desetinnou tečkou vytvoří výstup NaN: FFFF FFFF.

Pokud do jiné funkce bude přivedený výsledek NaN, projde skrz na výsledek. Pokud například NaN_ADD bude první operand pro funkci SUB_REAL, výsledek funkce SUB_REAL bude NaN_ADD. Pokud oba operandy funkce budou NaN, první operand projde skrz. Díky této vlastnosti předávání NaN skrz funkce je možno identifikovat funkci, kde NaN bylo jako první.

Poznámka

V případě NaN výstup ok bude ve stavu OFF (v nule).

Následující tabulka vysvětluje, kdy proud bude nebo nebude procházet, když se při binárních operací jako Sčítání, Násobení, atd. bude pracovat s čísly, která se budou pokládat nebo se budou rovnat nekonečnu. Jak bylo ukázáno dříve, výstupy přesahující kladné nebo záporné meze se budou pokládat za POS_INF respektive NEG_INF.

Tabulka E-1. Obecný případ průtoku proudu pro operace s pohyblivou desetinnou tečkou

Operace	Vstup 1	Vstup 2	Výstup	Proud
Všechny	Číslo	Číslo	Kladné nebo záporné nekonečno	Ne
Všechny kromě dělení	Nekonečno	Číslo	Nekonečno	Ano
Všechny	Číslo	Nekonečno	Nekonečno	Ano
Dělení	Nekonečno	Číslo	Nekonečno	Ne
Všechny	Číslo	Číslo	NaN	Ne

:

A

B

C

Č

D

E

F

G

H

Ch

R

Ř

S

T

U

V

W

X

Z