GE Fanuc Automation

Programovatelné řídicí systémy

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

Referenční příručka

GFK-0467M-CZ *květen 2002*

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 dokument obsahuje informace, které byly k dispozici v době jeho publikování. I když byla věnována maximální snaha přesnosti, cílem zde obsažených informací není zahrnout všechny podrobnosti nebo odchylky v hardwaru nebo softwaru ani postihnout všechny možné souvislosti ve spojitosti s instalací, obsluhou nebo údržbou. Mohou zde být popisované vlastnosti, které se u hardwarových a softwarových systémů nevyskytují. GE Fanuc Automation nepřijímá žádné závazky upozornit majitele této dokumentace na změny provedené později.

GE Fanuc Automation nepřijímá žádné stížnosti ani záruky, přímé nebo zákonné, a nepřebírá žádnou zodpovědnost za přesnost, úplnost, dostatečnost nebo užitečnost zde obsažených informací. Nejsou poskytovány žádné záruky obchodovatelnosti nebo vhodnosti.

Dále uvedené názvy jsou ochrannými známkami společnosti GE Fanuc Automation North America, Inc.

Alarm Master	Genius	PROMACRO	Series Six
CIMPLICITY	Helpmate	PowerMotion	Series Three
CIMPLICITY 90–ADS	Logicmaster	PowerTRAC	VersaMax
CIMSTAR	Modelmaster	Series 90	VersaPro
Field Control	Motion Mate	Series Five	VuMaster
GEnet	ProLoop	Series One	Workmaster

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

Revize tohoto manuálu

- Přidaný model CPU 374, který podporuje připojení k síti Ethernet přes dva vestavěné full-duplexní Ethernet porty 10BaseT/100BaseTx s automatickým nastavením. Modely 364 (verze 9.10 a pozdější) a 374 jsou jediná CPU Series 90-30, která podporují Ethernet Global Data. Všimněte si, že CPU374 podporuje pouze programovací software na bázi Windows®.
- Ostatní opravy a vysvětlení podle potřeby.

Související publikace

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

Uživatelská příručka programovacího softwaru VersaPro™ (GFK-1670)

Krátký úvod do CIMPLICITY® Machine Edition (GFK-1868)

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

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

Manuál specifikací I/O modulů Series 90™-30 (GFK-0898).

Uživatelský manuál modulu programovacího koprocesoru a podpůrného softwaru Series 90™ (GFK-0255).

Uživatelský manuál PCM vývojového softwaru (PCOP) Series 90™ PCM (GFK-0487).

Uživatelský manuál alfanumerického zobrazovacího systému CIMPLICITY™ 90-ADS (GFK-0499).

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

Uživatelský manuál ručního programovacího zařízení PLC Series 90™-30 a 90-20 (GFK-0402).

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

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

Uživatelský manuál Motion Mate™ DSM302 pro PLC Series 90™-30 (GFK-1464)

Uživatelský manuál vysokorychlostního čítače Series 90™-30 (GFK-0293).

Uživatelský manuál komunikačního modulu Genius Series 90™-30 (GFK-0412)

Uživatelský manuál řadiče sběrnice Genius Series 90-30 ™ (GFK-1034).

Uživatelský manuál řadiče sběrnice FIP Series 90™-70 (GFK-1038).

Uživatelský manuál FIP vzdáleného I/O Scanneru Series 90™-30 (GFK-1037).

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

Uživatelský manuál programovatelného automatu Series 90™ Micro GFK-1065).

Uživatelský manuál sériové komunikace PLC Series 90™ (GFK-0582).

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 90-30 model 341 a nižší a PLC Series 90-20 používají mikroprocesor 80188. PLC 90-30 typu 35x a 36x používají mikroprocesor 80386EX. PLC Series 90-30 řady 37x používají mikroprocesor 586. PLC Series 90 Micro používá mikroprocesor H8. Podporuje se jak vykonávání programu tak i základní úlohy vnitřní správy, jako například diagnostické rutiny, snímání vstupů/výstupů a zpracování alarmů. Systémový firmware také obsahuje rutiny pro komunikaci s programovacím zařízením. Tyto rutiny zajišťují výstup i načtení aplikačních programů, předání stavových informací a řízení PLC.

V PLC Series 90-30 je aplikační program (uživatelská logika), který řídí koncový proces, na který je PLC použito, je řízený jednoúčelovým koprocesorem pro řazení instrukcí (Instruction Sequencer Coprocessor - ISCP). ISCP je implementovaný v hardwaru model 313 a vyšší a v softwaru systémů typu 311 a v PLC Micro. Mikroprocesor a hardwarové 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 softwaru 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 pohyblivé desetinné tečky podporují ***pouze*** CPU řady 35x a 36x, verze 9 nebo pozdější a všechny verze CPU352 a CPU 374.

CPU364 (verze 9.10 nebo pozdější) a CPU374 jsou jediná CPU Series 90-30, která podporují Ethernet Global Data (EGD).

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

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

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.

Poznámka pro uživatele programovacího softwaru PLC na bázi Windows

Tento manuál byl napsaný pro uživatele Logicmasteru (programovací software na bázi DOS). Softwarové produkty PLC na bázi Windows, jako například CIMPLICITY® Machine Edition Logic Developer a VersaPro®, obsahují informace o instrukční sadě PLC jako kontextovou nápovědu vestavěnou v softwaru místo v manuálu. Uživatelé programovacího softwaru na bázi Windows si musí uvědomit, že se instrukce objevují odlišně od způsobu, než jak se objevovaly na obrazovce Logicmasteru (v PLC stále pracují stejně). Systém kontextové nápovědy má nejpřesnější informace o používání instrukční sady v programovacím softwaru na bázi Windows. Přehled hlavních rozdílů mezi typy najdete v Dodatku F.

Kapitola 2

Činnost systému

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

Část 1: Souhrn sekvencí PLC cyklu

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, provádění interní správy, obsluhu programovacího zařízení a obsluhu ostatních komunikací.

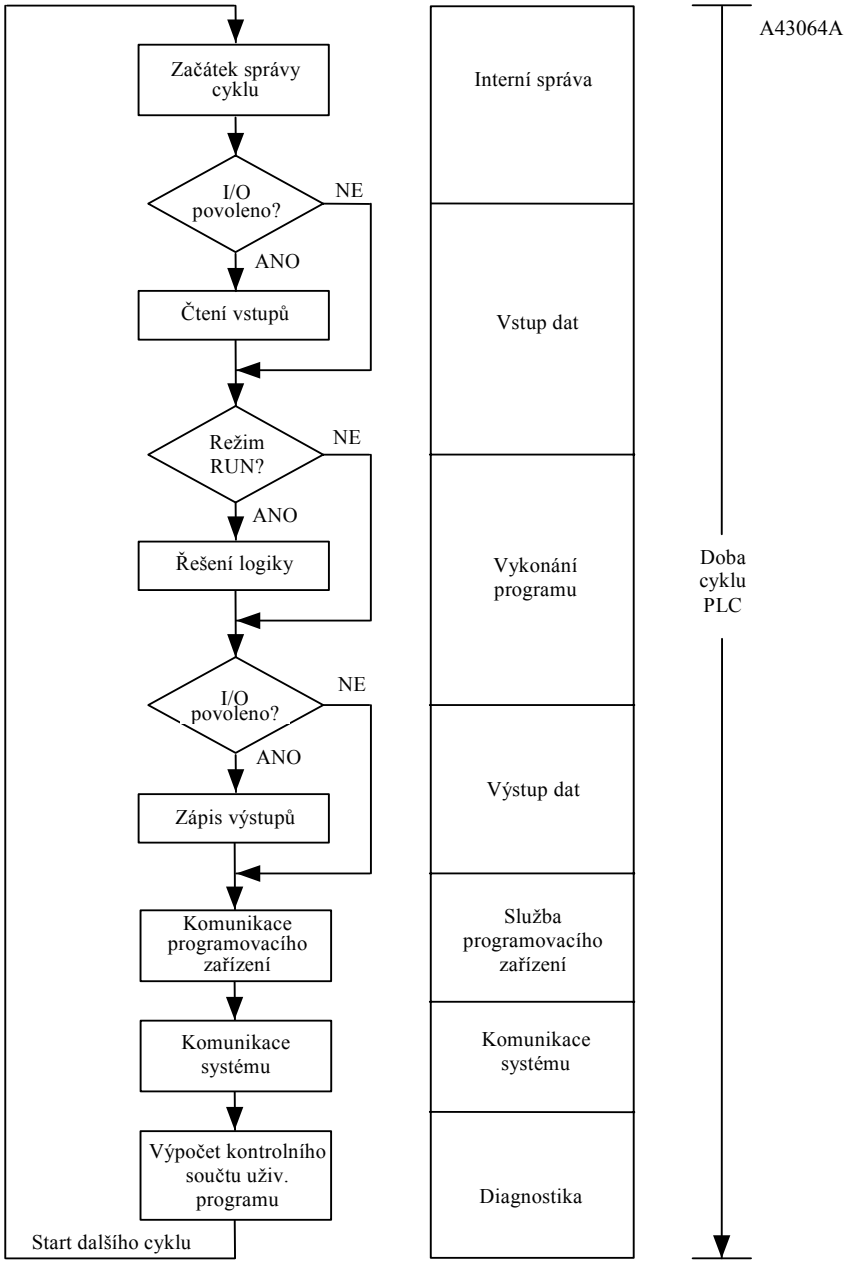
PLC Series 90-30, 90-20 a Micro normálně pracují v režimu **STANDARDNÍHO PROGRAMOVÉHO CYKLU**. Další režimy provozu jsou režim **ZASTAVENÍ SE ZÁKAZEM I/O**, režim **ZASTAVENÍ S POVOLENÍM I/O** a režim **KONSTANTNÍHO CYKLU**. Každý z těchto režimů popsaných v této kapitole je řízený vnějšími jevy a nastavením 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. Komunikace programovacího zařízení
6. Komunikace systému
7. Diagnostika

Všechny tyto kroky se 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. PLC cyklus

Jak je ukázáno v sekvenci PLC cyklu, 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

Prvek cyklu	Popis		Podíl času (v milisekundách) [4]						
			Micro	211	311/313	331	34x	35x/36x	37x
Správa	• Doba výpočtu cyklu • Plánování startu dalšího cyklu • Určení režimu dalšího cyklu • Aktualizace referenčních tabulek chyb • Reset hlídacího časovače		0.368	0.898	0.714	0.705	0.424	0.279	0.027
Zápis dat	Vstupní data se načítají ze vstupních a přídavných modulů		Poznámka 5	Podíly na času zápisu viz tabulka 2-2 a 2-3.					
Vykonávání programu	Řeší se uživatelská logika		Doba vykonávání závisí na délce programu a typech instrukcí používaných v programu. Doby vykonávání instrukcí jsou uvedené v Příloze A.						
Výstup dat	Výstupní data se posílají na výstupní a přídavné moduly.		1.656	Podíly na času zápisu viz tabulka 2-2 a 2-3.					
Komunikace programovacího zařízení a systému	Zpracovávají se požadavky servisu od programovacích zařízení a inteligentních modulů.[1]	HHP	1.93	6.526	4.426	4.524	2.476	0.334	nepoužívá se
		Programovací zařízení	0.380	3.536	2.383	2.454	1.248	0.517	0.026
		PCM[2]	nepoužívá se	nepoužívá se	nepoužívá se	3.337	1.943	0.482	0.029
Rekonfigurace	Monitorují se pozice s vadnými moduly a prázdné pozice.		nepoužívá se[6]	nepoužívá se	0.458	0.639	0.463	0.319	0.243
Diagnostika	Ověření integrity uživatelského programu (podíl času je čas požadovaný na slovo kontrolované při každém cyklu) [3]		nepoužívá se[7]	0.083	0.050	0.048	0.031	0.010	0.022

1. Podíl času obsluhy externího zařízení na době 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 vyšších. 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 žádný periodický podprogram. Pokud bude aktivní periodický podprogram, časy budou delší.
5. Čas načítání dat pro PLC Micro je možno určit následovně: 0.365 ms (pevné snímání) + 0.036 ms (doba filtrace) × (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 na snímání I/O (v milisekundách) pro CPU Series 90-30 35x, 36x a 37x

Typ modulu		CPU Series 35x a 36x			CPU Series 37x		
		Hlavní sestava	Expanzní sestava	Vzdálená sestava	Hlavní sestava	Expanzní sestava	Vzdálená sestava
8-bodový diskrétní vstup		.030	.055	.206	.030	.055	.206
16-bodový diskrétní vstup		.030	.055	.206	.030	.055	.206
32-bodový diskrétní vstup		.043	.073	.269	.048	.075	.272
8-bodový diskrétní výstup		.030	.053	.197	.024	.052	.198
16-bodový diskrétní výstup		.030	.053	.197	.030	.052	.199
32-bodový diskrétní výstup		.042	.070	.259	.047	.069	.258
Kombinace diskrétního vstupu/výstupu		.060	.112	.405	.052	.110	.408
4-kanálový analogový vstup		.075	.105	.396	.085	.109	.403
2-kanálový analogový výstup		.058	.114	.402	.046	.101	.393
16-kanálový analogový vstup (proud nebo napětí)		.978	1.446	3.999	.423	.700	1.741
8-kanálový analogový výstup		1.274	1.988	4.472	.873	1.492	3.635
Kombinace analogového vstupu/výstupu		1.220	1.999	4.338	.862	1.487	4.103
Vysokorychlostní čítač		1.381	2.106	5.221	1.142	1.808	5.234
I/O procesor		1.574	2.402	6.388	1.270	2.125	6.269
Rozhraní Ethernet (bez připojení)		.7129	2.067	3.681	.426	.795	2.302
Power Mate APM (1 osa)		1.527	2.581	6.388	1.236	2.073	6.032
Power Mate APM (2 osy)		1.807	2.864	7.805	1.539	2.439	7.369
DSM 302 *	40 AI, 6 AQ	2.143	3.315	9.527	1.801	2.963	9.275
	50 AI, 9 AQ	2.427	3.732	11.092	2.075	3.373	10.840
	64 AI, 12 AQ	2.864	4.317	13.138	2.441	3.931	12.881
DSM314 *	Nakonfigurovaná 1 osa	1.6	2.6	6.9	1.330	2.337	6.905
	Nakonfigurované 2 osy	2.2	3.8	9.9	1.888	3.148	9.917
	Nakonfigurované 3 osy	2.8	4.3	13.0	2.421	3.953	12.929
	Nakonfigurované 4 osy	3.3	5.2	15.9	2.969	4.761	15.982
GCM	8 32-bitových zařízení	8.826	16.932	21.179	7.386	9.520	20.591
GCM+	žádná zařízení	.567	.866	1.830	.457	.759	1.743
	32 zařízení se 64 slovy	19.497	25.588	80.871	17.036	24.390	80.044
GBC	žádná zařízení	.798	1.202	2.540	.544	.908	2.209
	16 64-slovních zařízení	29.976	40.570	131.702	26.976	38.564	130.639
PCM311	nenakonfigurováno nebo bez aplikační úlohy	.476	nepoužívá se	nepoužívá se	.195	nepoužívá se	nepoužívá se
	vykonávající aplikační program 20 kB	1.746	nepoužívá se	nepoužívá se	.538	nepoužívá se	nepoužívá se
ADC (žádná úloha)		.476	nepoužívá se	nepoužívá se	.193	nepoužívá se	nepoužívá se
I/O Link Master	žádná zařízení	.569	.865	1.932	.996	1.618	3.749
	Šestnáct 64-bodových zařízení	4.948	7.003	19.908	5.924	8.240	26.637
I/O Link Slave	32 bodů	.087	.146	.553	.095	.149	.540
	64 bodů	.154	.213	.789	.165	.219	.803

* 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. V případě DSM302 podrobnosti najdete v *Uživatelském manuálu Motion Mate DSM302 pro PLC Series 90-30* , GFK1464. V případě DSM314 podrobnosti najdete v *Uživatelském manuálu Motion Mate DSM314 pro PLC Series 90-30* , GFK1742. POZNÁMKA: DSM314 bude pracovat pouze s CPU 350, 352, 360, 363, 364 a 374 a pouze s firmwarem CPU verze 10.00 nebo pozdější.

Tabulka 2-3. Podíl času na snímání I/O (v milisekundách) pro CPU Series 90-30 model 311 až 341

Typ modulu		Model CPU						
			331			340/341		
		311/313 /323	Hlavní Sestava	Přídavná sestava	Vzdálená sestava	Hlavní Sestava	Expanzní sestava	Vzdálená sestava
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
DSM302 *	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
PCM311	nenakonfigurováno nebo bez aplikační úlohy	nepoužívá se	3.350	nepoužívá se	nepoužívá se	1.684	nepoužívá se	nepoužívá se
	co nejrychlejší čtení 128 %R	nepoužívá se	4.900	nepoužívá se	nepoužívá se	2.052	nepoužívá se	nepoužívá se
ADC 311		nepoužívá 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 *Uživatelský manuál Motion Mate DSM302 pro PLC Series 90-30*, GFK1464. POZNÁMKA: CPU 311 až 341 nepodporují DSM314.

Výpočet doby cyklu

Tabulka 2-1 uvádí seznam sedmi položek, které se podílejí na době PLC cyklu. 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.).

Část cyklu	Výpočet	Podíl času		
		Bez programovacího zařízení	S HHP	S Logicmaster
Správa	0.705 ms	0.705 ms	0.705 ms	0.705 ms
Vstup dat	0.055 × 5 = 0.275 ms	0.275 ms	0.275 ms	0.275 ms
Vykonávání programu	1000 × 0.4µs* + 200 x 89µs** + 18.2ms	18.2 ms	18.2 ms	18.2 ms
Výstup dat	0.061 × 4 = 0.244 ms	0.244 ms	0.244 ms	0.244 ms
Programátorské služby	0.4 ms + čas na programovací zařízení + 0.6 ms	0 ms	4.524 ms	2.454 ms
Jiné než programátorské služby	Žádné v tomto příkladu	0 ms	0 ms	0 ms
Rekonfigurace	0.639 ms	0.639 ms	0.639 ms	0.639 ms
Diagnostika	0.048 ms	0.048 ms	0.048 ms	0.048 ms
Doba PLC cyklu	Správa + Vstup dat + Výkon programu + Výstup dat + Programátorské služby + Jiné než programátorské služby + Diagnostika	12.611 ms	17.135 ms	15.065 ms

Podrobnosti PLC cyklu

Tato kapitola probírá detaily hlavních částí PLC cyklu:

1. Správa
2. Čtení vstupů
3. Logické skenování aplikačního programu
4. Zápis výstupů
5. Programátorské služby
6. Komunikace systému
7. Rekonfigurace
8. Výpočet kontrolního součtu

1. 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.

2. Čtení vstupu

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 (GCM), rozšířený komunikační modul Genius (GCM+) nebo řadič sběrnice Genius (GBC), se uloží v paměti %G.

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

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

3. Řešení aplikačního programu logiky

Řešení aplikačního programu logiky se provede hned po dokončení čtení vstupů. Řešení aplikačního programu logiky provádí dva hlavní úkoly: (1) řešení/vykonání logiky programu a (2) aktualizace výstupní paměti %Q, %AI a %AQ. (Výstupní moduly se však neaktualizují, dokud se neprovede zápis výstupů.) V zásadě se žebříková logika řeší zleva doprava a shora dolů, i když tento směr toku je možno přechodně změnit vyvoláním podprogramu a pomocí skoků. Řešení logiky skončí, když se zjistí instrukce END nebo když se dosáhne implicitního END OF PROGRAM LOGIC.

CPU 313 a vyšší CPU mají sekvenční koprocesor instrukcí (ISCP), který vykonává Booleovské instrukce, a mikroprocesor 80C188,80386 nebo AMD SC 520 vykonává bloky časovače, čítače a funkční bloky. V případě CPU model 311 a 90-20 bude procesor 80C188 vykonávat všechny Booleovské instrukce, instrukce časovače, čítače a funkčního bloku. V případě Micro bude procesor H8 vykonávat všechny Booleovské instrukce a instrukce funkčních bloků.

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

4. Zápis výstupu

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 budete mít modul komunikace Genius nebo kontrolér sběrnice Genius, který je nakonfigurovaný na přenos globálních dat, pak data z paměti %G se pošlou do GCM, GCM+ nebo GBC. Zápis výstupů u Series 90-20 a Micro zahrnuje pouze diskrétní výstupy.

Během zápisu výstupů se všechny výstupní moduly Series 90-30 zapisují ve vzestupném pořadí adres. Zápis výstupů se dokončí, když se všechna výstupní data odešlou do výstupních modulů Series 90-30.

Pokud CPU bude v režimu **`STOP`** a parametr ***IPScan-Stop*** na obrazovce pro konfiguraci CPU bude nastavený na ***NO***, zápis výstupů se přeskočí.

Upozornění

Pokud parametr *IPScan-Stop* na obrazovce pro konfiguraci CPU bude nastavený na *YES*, skutečné výstupy se mohou nastavit do stavu ON, i když PLC bude v režimu STOP, protože PLC během čtení výstupu zapisuje aktuální hodnoty ve výstupních tabulkách do výstupních modulů.

5. 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 žádný modul, okno komunikace programovacího zařízení se nevykoná. Při každému cyklu se nakonfiguruje pouze jeden modul.

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.

Režimy okna komunikace programovacího zařízení

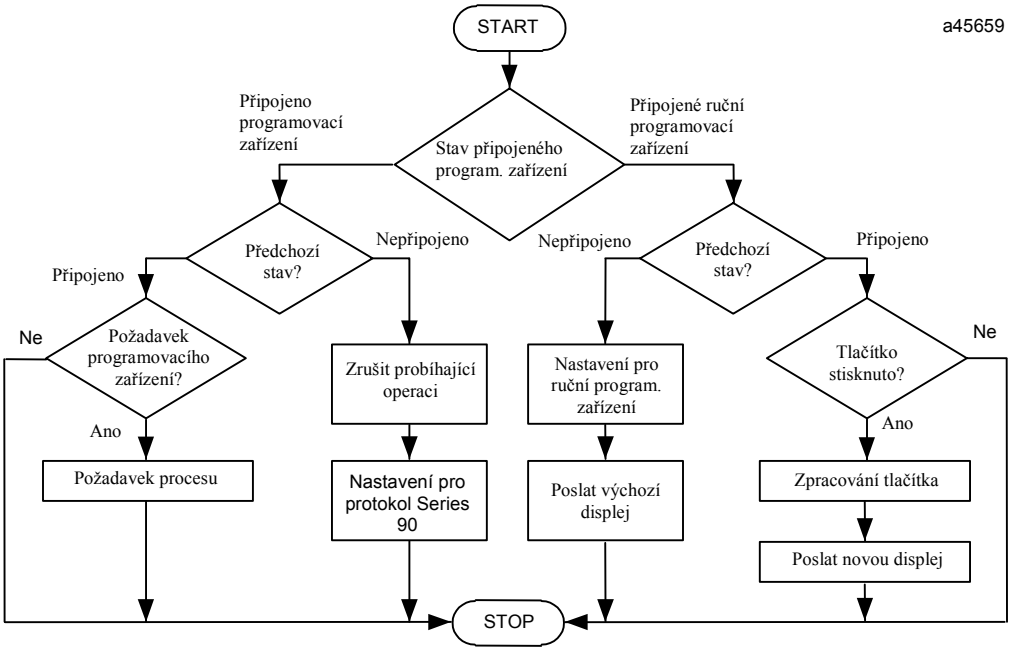
- **Omezený režim**. Ve výchozím režimu omezeného okna CPU vykoná při každém cyklu jednu operaci na programovacím zařízení, to znamená, že akceptuje jeden požadavek nebo odezvu na službu na jedno stisknutí tlačítka. Pokud programovací zařízení vznese požadavek, který vyžaduje více než 6 (nebo 8 v závislosti 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.

- **Úplný režim.** V úplném režimu CPU bude provádět komunikaci programovacího zařízení, dokud nebude dokončena nebo neuplyne 50 milisekund.

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

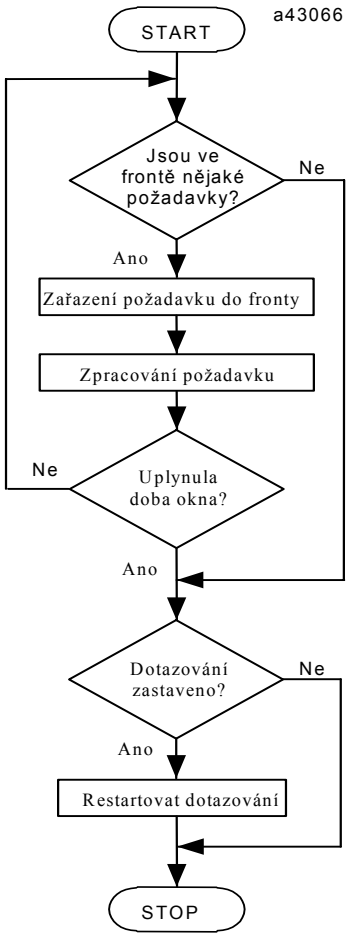


Obrázek 2-2. Vývojový diagram okna komunikace programovacího zařízení

6. Okno komunikace systému (modely 331 a vyšší)

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

7. Rekonfigurace

Během této části cyklu CPU kontroluje skutečnou hardwarovou sestavu v porovnání s nakonfigurovanou hardwarovou sestavou. Pozice, které jsou nakonfigurované pro modul, ale fyzicky jsou prázdné, nebo pozice obsahující vadné moduly CPU nepřečte (tj. CPU z nich nepřečte žádná data a nepošle žádná výstupní data do tohoto modulu nebo na tuto pozici). Pokud během rekonfigurace CPU zjistí, že pozice, která původně obsahovala vadný modul, nyní bude obsahovat modul bez vady nebo že se do PLC fyzicky přidal nakonfigurovaný modul, začne tento modul číst.

Rekonfigurace umožní CPU provádět následující:

- Rozpoznat legitimní změnu, kterou provedete v konfiguraci.
- Ignorovat potenciálně poškozená nebo nepřesná vstupní data z vadných nebo chybějících modulů.
- Zabránit posílání výstupních dat, která by se mohla poškodit vadným výstupním modulem.

8. Výpočet kontrolního součtu

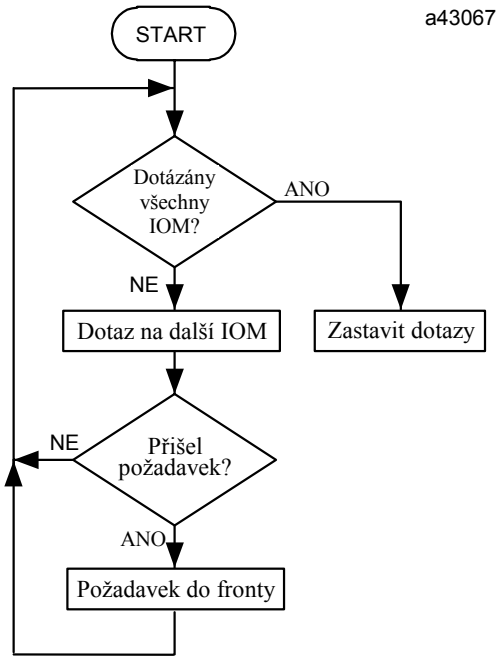
Výpočet kontrolního součtu se vykoná u uživatelského programu na konci každého cyklu. Protože výpočet kontrolního součtu celého programu může trvat dlouho, na obrazovce konfigurace CPU můžete zadat počet slov, které se mají zkontrolovat, v rozmezí od 0 do 32.

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.

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 (periodicky zkontrolovat), 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 modulu digitálního serva (DSM) s PLC

DSM302 a DSM314 jsou inteligentní přídavné moduly, které pracují asynchronně s CPU moduly Series 90-30. Výměna dat mezi CPU a DSM probíhá automaticky přes paměť %Q, %I, %AQ a %AI. CPU PLC vyžaduje čas na čtení a zápis předávaných dat v interní komunikaci s modulem DSM. Tabulka 2-2 uvádí délku cyklů při různých konfiguracích DSM. Další posouzení délky cyklu, které platí pro moduly DSM, najdete v následujících manuálech:

- *Uživatelský manuál Motion Mate DSM302 pro PLC Series 90-30*, GFK-1464.
- *Uživatelský manuál Motion Mate DSM314 pro PLC Series 90-30*, GFK-1742.

Obměny standardního programového cyklu

Kromě normálního vykonávání standardního programového cyklu je možno se setkat nebo si vynutit určité změny. Tyto varianty 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**. Hodnotu režimu **CYKLUS S KONSTANTNÍ DOBOU** je možno nastavit v rozmezí 5 až 200 milisekund pro CPU 311-341 nebo v rozmezí 5 až 500 milisekund pro CPU 350-364 a 374.

Vzhledem ke změnám doby požadované pro různé části PLC cyklu musí být konstantní doba 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.

Režim **CYKLUS S KONSTANTNÍ DOBOU** použijte, když je nutno se dotazovat na I/O body nebo hodnoty registru s konstantní četností, například algoritmus řízení. 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 komunikačních oken se dokončí. Avšak na začátku dalšího cyklu se zaznamená chyba překročení cyklu.

Konfigurace režimu konstantního cyklu

Pro konfigurování režimu konstantního cyklu jsou dva způsoby:

- V konfiguračním softwaru Logicmaster obrazovka konfigurování CPU má konfigurovatelné parametry Režim cyklu a Časovač cyklu. Aby se změny provedly, po provedení volby je nutno konfiguraci uložit z programovacího zařízení do PLC během režimu **STOP**. Po uložení tato konfigurace bude jako výchozí režim cyklu.
- V programovacím softwaru Logicmaster je možno v Tabulce PLC cyklu v menu PLC Control a Status volit parametr Režim cyklu a Časování. Parametry na této obrazovce je možno editovat pouze v režimu **RUN**. Změny provedené na této obrazovce se uloží pouze do PLC, ne do adresáře na vašem PC a budou platné pouze, dokud PLC zůstane v režimu RUN. Jakmile se PLC zastaví, převezme Režim cyklu, který bude platný od doby, kdy PLC přejde do režimu RUN. Tato metoda přechodné konfigurace Režim cyklu je užitečná pro návrh systému a operace ladění.

PLC cyklus 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 chybného modulu a modulu rekonfigurace. 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**.

Upozornění

Pokud parametr *IPScan-Stop* na obrazovce detailů CPU bude nastavený na *YES*, skutečné výstupy se mohou nastavit do stavu ON, i když PLC bude v režimu STOP, protože PLC během čtení výstupu zapisuje aktuální hodnoty ve výstupních tabulkách do výstupních modulů.

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** může být až 12 milisekund. Režim aktivního okna je možno změnit pomocí obrazovky “Kontrola cyklu” v 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, 36x a 37x: Změna režimu a ochrana Flash

Všechna CPU 350-374 mají klíček (CPU 311-341 ho nemají); některé verze firmwaru CPU však všechny funkce klíčku nepodporují. Tato část obsahuje popis těchto rozdílů. Všimněte si, že klíčky u některých těchto CPU jsou označené nápisem ON/RUN a OFF/STOP a u jiných pouze ON a OFF. Bez ohledu na označení všechny tyto klíčky pracují následujícím způsobem:

Ochrana paměti Flash (natvrdo zapojené)

Tuto napevno zapojenou nekonfigurovatelnou funkci je možno použít k ochraně paměti Flash před změněním nepovolanou osobou (osobou bez klíčku). Když klíček bude v poloze ON, paměť Flash nelze změnit (nelze do ní zapisovat). Do paměti Flash je možno zapisovat, když klíček bude v poloze OFF. Tato funkce klíčku bude vždy aktivní bez ohledu na nastavení konfigurovatelných nastavení.

Run/Stop (Konfigurovatelné)

Tato konfigurovatelná funkce byla zavedena u firmwaru CPU verze 7.00. Nastavuje se parametrem **R/S Switch** na obrazovce pro konfiguraci CPU. Parametr **R/S Switch** je implicitně nastavený na *Disabled*. Pokud parametr **R/S Switch** bude nastavený na *Enabled,* můžete PLC zastavit otočením klíčku do polohy OFF a spustit PLC otočením klíčku do polohy ON (pokud se nevyskytnou žádné chyby). Pokud se chyby vyskytnout, stane se následující:

- **Pokud PLC bude obsahovat chybu, ale nikoliv fatální**, otočení klíčku z polohy OFF do polohy ON bude mít za následek, že PLC přejde do režimu běhu a kontrolka RUN bude trvale svítit, ale tabulka chyb se nevynuluje.
- **Pokud PLC bude obsahovat fatální chybu,** otočení klíčku z polohy OFF do polohy ON bude mít za následek, že kontrolka RUN se bude rozsvěcet a zhasínat s periodou pěti sekund a PLC nepřejde do režimu běhu. Tato blikající kontrolka indikuje, že v tabulce chyb je jedna nebo více fatálních chyb. Chyby z chybové tabulky můžete zkusit vynulovat tak, že během periody pěti sekund znovu otočíte klíčkem z polohy OFF do polohy ON. (Pokud perioda pěti sekund uplyne, otočením klíčku z polohy OFF do polohy ON spustíte další tuto periodu.) Jestliže se tímto způsobem chyby neodstraní, musíte příčinu opravit před tím, než budete schopni činnost obnovit. Podrobnosti chyb viz kapitola 3.

Další poznámky ke klíčku ohledně Run/Stop

- Pokud parametr **R/S Switch** bude nastavený na *Enabled* a klíček bude v poloze OFF, PLC bude v režimu STOP a programovací software nelze použít k převedení PLC do režimu RUN.
- Pokud parametr **R/S Switch** bude nastavený na *Enabled*, klíček bude v poloze ON a nebudou se vyskytovat žádné fatální chyby, programovací software je možno použít k přepínání PLC mezi režimy RUN a STOP.

- Pokud parametr **R/S Switch** bude nastavený na *Enabled*, klíček bude v poloze ON ale PLC bude zastaveno, PLC můžete nastavit do režimu RUN buď otočením klíčku do polohy OFF a pak zpět do polohy ON nebo pomocí programovacího softwaru.

Ochrana paměti RAM a přepisu (konfigurovatelná)

Tato funkce byla zavedena u firmwaru CPU verze 8.00. Nastavuje se parametrem **Mem Protect** na obrazovce pro konfiguraci CPU. Parametr **Mem Protect** je implicitně nastavený na *Disabled*.

Pokud parametr **Mem. Protect** bude nastavený na *Enabled* a klíček bude v poloze ON, bude platit následující:

- Uživatelskou paměť RAM (programová a konfigurační) nelze měnit.
- Diskrétní body nelze přepisovat.
- Hodiny denního času (TOD) nelze měnit pomocí ručního programovacího zařízení (avšak hodiny TOD lze stále měnit pomocí konfiguračního softwaru).

Zabezpečení klíčků

Každé nové CPU 350-374 se dodává se dvěma klíčky. Pokud budete používat jednu nebo více funkcí ochrany pomocí klíčku uvedené výše, doporučujeme vám tyto klíčky pečlivě chránit. Pokud je ztratíte, necháte na nesprávném místě nebo je někdo ukradne, nebudete moct se svým PLC pracovat a budou k němu mít přístup nepovolané osoby. Náhradní klíčky si můžete zakoupit jako náhradní nebo pro případ, kdy přístup k PLC mají mít více než dvě osoby. U prodejce GE Fanuc si můžete vyžádat sadu se třemi klíčky. Při objednávání požadujte katalogové číslo 44A736756-G01. Všechna CPU 350-374 mají stejný klíček.

Vyřazení funkce klíčku

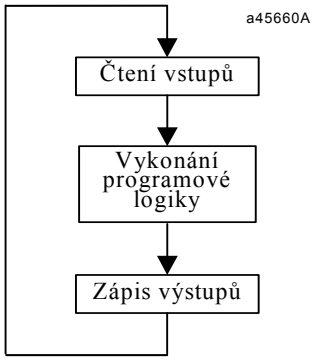
Pokud nebudete chtít používat žádnou funkci ochrany pomocí klíčku, můžete se rozhodnout je všechny vyřadit. To provedete tak, že klíček nastavíte do polohy OFF a parametry **R/S Switch** a **Mem. Protect** nastavíte (jak bylo popsáno výše) do stavu *Disabled* (jejich implicitní nastavení). V tomto stavu budou všechny funkce ochrany pomocí klíčku vyřazené a pro přístup k PLC klíček nebudete potřebovat.

Část 2: Organizace programu a uživatelské adresy/data

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

Velikost uživatelské paměti	
Modely CPU	**Uživatelská paměť (Kbajtů)**
CPU311	6
CPU313, CPU323	12
CPU331	16
CPU340	32
CPU341	80
CPU350	80 (verze 9.00 a pozdější) 32 (před verzí 9.00)
CPU351, CPU352, CPU360, CPU363, CPU364, CPU374	240 (verze 9.00 a pozdější) 80 (před verzí 9.00)

Počínaje firmwarem CPU verze 9.00 jsou velikosti pamětí %R, %AI a %AQ pro CPU 351, 352, 360, 363, 364 a 374 konfigurovatelné. (Více podrobností najdete v *Uživatelském manuálu programovacího softwaru Logicmaster 90™ Series 90™-30/20/Micro*, GFK-0466K nebo pozdější nebo v uživatelském manuálu pro váš programovací software). Program pro programovatelný automat Series 90-20 může mít velikost až 2 KB pro CPU model 211 a maximální počet příček přípustných na blok logiky (hlavního nebo v podprogramu) je 3000. U PLC Series 90-30 maximální velikost je 80 kilobajtů pro bloky C a 16 kilobajtů pro bloky LD a SFC; avšak v bloku SFC se část z 16 KB používá pro interní datový blok. Jak je znázorněno na následujícím obrázku, pokud PLC bude v normálním režimu RUN, uživatelský program logiky se bude vykonávat opakovaně.



Seznam velikostí programů a meze adres pro jednotlivé modely CPU najdete v *Manuálu pro instalaci a popisu hardwaru programovatelného automatu Series 90-30,* GFK-0356, nebo v *Uživatelském manuálu 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 "vyvolávat" 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.

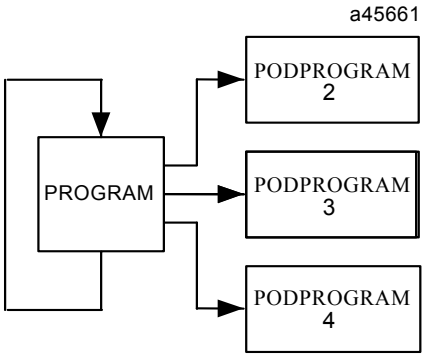
Poznámka

Bloky podprogramů se nepodporují v PLC Series 90-20 PLC nebo Micro.

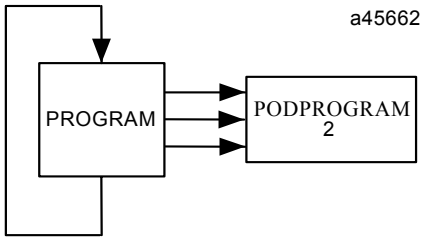
Použití podprogramů je volitelné. Rozdělení programu na menší podprogramy může zjednodušit programování, zlepšit pochopení řídicího algoritmu a případně zmenšit celkovou velikost logiky potřebnou pro program.

Příklady používání bloků podprogramů

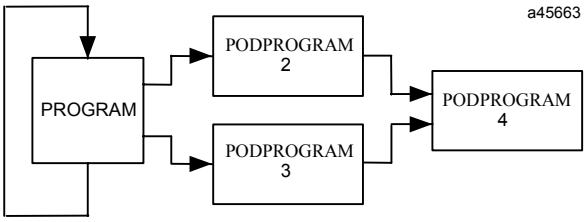
Logiku pro program je možno například rozdělit na tři podprogramy, z nichž každý by byl vyvolávaný z programu podle potřeby. V tomto příkladu blok programu může obsahovat málo logiky a bude sloužit především k řazení bloků podprogramů.



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.



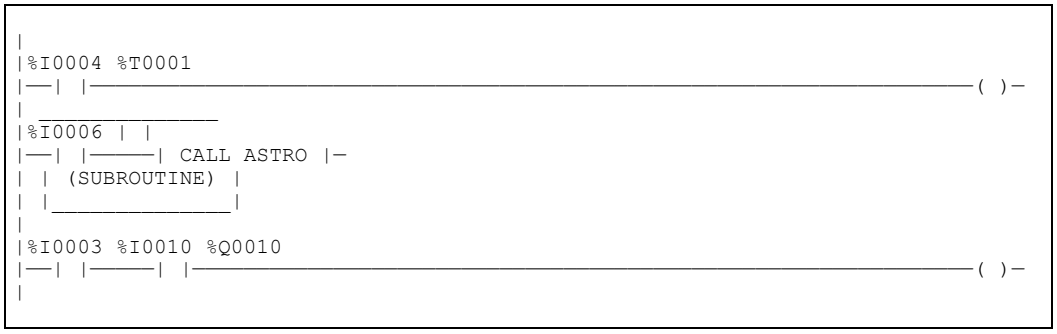
Kromě vyvolání z programu je možno bloky podprogramu také vyvolávat i z jiných bloků podprogramu (“to se nazývá vnořování”). Blok podprogramu může dokonce vyvolat sám sebe.



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

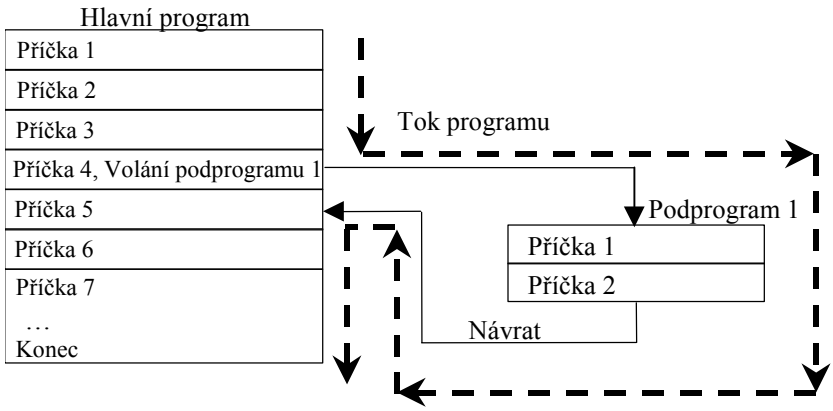
Blok podprogramu se vykoná, když bude vyvolaný z žebříkového programu v programu nebo z jiného bloku podprogramu.



Tento příklad ukazuje instrukci volání podprogramu CALL tak, jak se objeví na obrazovce žebříkové logiky.

Sekvence vyvolávání v programech obsahujících podprogramy

Pokud podprogram bude vyvolaný z jiného programu nebo podprogramu, vyvolaný podprogram se vykoná až do konce a pak vrátí řízení programu nebo podprogramu, který ho vyvolal. Řízení se vrátí na příčku, která následuje za příčkou obsahující volání podprogramu. V následujícím příkladu silná přerušovaná čára ukazuje tok programu (v pořadí, ve kterém se vykonává logiky). V tomto příkladu se vyvolává jednoduchý dvou-příčkový podprogram z Příčky 4 hlavního programu. Po vykonání dvou příček podprogramu se program vrátí do hlavního programu a začne Příčkou 5.



Cyklické 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, APM, DSM, 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 milisekundy, 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 %AQ0056). 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 softwaru 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 vyjmout/vložit a zápis/zahrnutí souboru. 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.
%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.

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

Přezdívky

Uživatel může adresám přiřadit přezdívku. Přezdívka je užitečná, protože uživateli dává informace o účelu nebo funkci adresy. Například v PLC systému nainstalovaném v továrně se výstupní cívka %Q0001 používá k nabuzení relé spouštěče motoru, který řídí fyzické čerpadlo, které zaměstnanci továrny všeobecně nazývají "Čerpadlo číslo 1". Přiřazení přezdívky PUMP1 adrese %Q0001 pomůže zaměstnancům, kteří budou hledat závady na systému, aby poznali účel adresy %Q0001.

Přezdívky musí začínat písmenem a mohou mít délku jeden až sedm znaků. Aby bylo možno rozlišovat mezi adresou paměti (adresou) a přezdívkou, jako první znak paměťové adresy se používá znak procenta (%). Proto například M1 bude PLC pokládat za přezdívku, ale %M1 bude pokládat za paměťovou adresu. Více informací o přezdívkách najdete v manuálu GFK-0466 (uživatelský manuál Logicmaster pro PLC Series 90-30).

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 modely 331 a vyšší je možno Nastaví se 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.

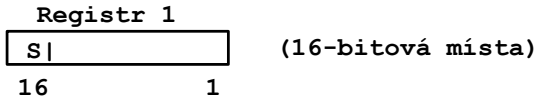
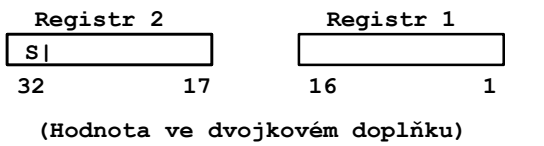
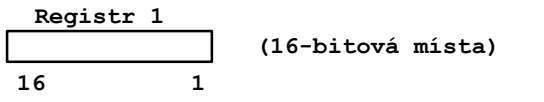
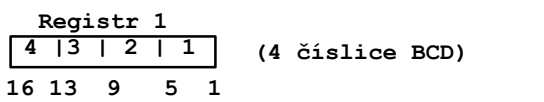
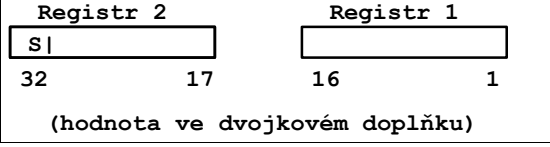
Adresy %Q a %M jsou neretentivní (to znamená, že se po zapnutí napájení vynulují, když PLC přejde z režimu **STOP** do režimu **RUN**) vždy, když se budou používat s neretentivními cívkami. Mezi neretentivní cívky patří cívky —( )—, negované cívky —(/)—, cívky SET —(S)— a cívky RESET —(R)—.

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. (Bit 16 je znaménkový bit.) 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.	
BAJT	Bajt	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. Platný rozsah hodnot dvojitého slova je 0 až FFFFFFFF.	
BCD-4	Čtyřbitové binárně 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	Pohyblivá desetinná 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 jsou přiřazené paměti %S, %SA, %SB a %SC. Všechny mají svou přezdívku. 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". Pro názvy paměťových adres tyto speciální přezdívky nelze použít.

Tabulka 2-7. Adresy stavu systému

Adresa	Přezdívka	Definice
%S0001	FST_SCN	Nastaveno na 1, když je aktuální cyklus prvním cyklem.
%S0002	LST_SCN	Resetuje se z 1 na 0, když je aktuální cyklus posledním cyklem.
%S0003	T_10MS	Kontakt časovače 0,01 sec.
%S0004	T_100MS	Kontakt časovače 0,1 sec.
%S0005	T_SEC	Kontakt časovače 1,0 sec.
%S0006	T_MIN	Kontakt časovače 1,0 minuta.
%S0007	ALW_ON	Vždy ON (zapnuto).
%S0008	ALW_OFF	Vždy OFF (vypnuto).
%S0009	SY_FULL	Nastaví se, když se naplní tabulka chyb PLC. Vynuluje se, když se z PLC tabulky chyb odstraní zápis a když se tabulka PLC chyb smaže.
%S0010	IO_FULL	Nastaví se, když se naplní tabulka chyb I/O. Vynuluje se, když je zadání odstraněno z tabulky chyb I/O a tabulka je smazána.
%S0011	OVR_PRE	Nastaví se, jestliže v paměti %I, %Q, %M nebo %G existuje přepis.
%S0013	PRG_CHK	Nastaví se při aktivní kontrole programu v pozadí.
%S0014	PLC_BAT	U CPU verze 4 nebo pozdější se nastaví jako indikace špatného stavu baterie. Adresa kontaktu se aktualizuje jednou za cyklus.
%S0017	SNPXACT	Nadřízený počítač SNP-X je aktivně připojený k CPU.
%S0018	SNPX_RD	Nadřízený počítač SNP-X načetl data z CPU.
%S0019	SNPX_WT	Nadřízený počítač SNP-X zapsal data do CPU.
%S0020		Nastaví se do stavu ON po úspěšném vykonání relační funkce používající data typu REAL. Vynuluje se, když některý ze vstupů bude NaN (nenumerický).
%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 hardwarové 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 při výskytu chyby aplikace. Vynuluje se, když PLC přejde z režimu **STOP** do režimu **RUN**.

Adresa	Přezdívka	Definice
%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, jestliže diagnostika zjistí problém hardwaru CPU. Vynuluje se vyměněním modulu CPU.
%SA0011	LOW_BAT	Nastaví se při poklesu napětí baterie. Vynuluje se výměnou baterie a zajištěním, že PLC naběhne po zapnutí 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 při zjištění závady některého přídavného modulu. Vynuluje se výměnou modulu a vykonáním cyklu zapnutí napájení hlavní sestavy.
%SA0031	SFT_SIO	Nastaví se po zjištění neopravitelné chyby softwaru v některém volitelném modulu. 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.
%SB0011	BAD_PWD	Nastaví se, když je zjištěno uhrožení heslem chráněného přístupu. 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 při výskytu chyby během operace ukládání programovacím zařízením. Vynuluje se, když se operace ukládání dokončí úspěšně.
%SC0009	ANY_FLT	Nastaví se při výskytu jakékoli chyby. Vynuluje se, když v obou tabulkách chyb nebudou žádné záznamy.
%SC0010	SY_FLT	Nastaví se při výskytu libovolné chyby, jejímž důsledkem je zápis do tabulky chyb PLC. Vynuluje se, když v tabulce PLC chyb nebude žádný záznam.
%SC0011	IO_FLT	Nastaví se při výskytu libovolné chyby, jejímž důsledkem je zápis do tabulky chyb I/O. Vynuluje se, když v tabulce I/O chyb nebude žádný záznam.
%SC0012	SY_PRES	Nastaví se, pokud existuje alespoň jeden záznam v tabulce chyb PLC. Vynuluje se, když v tabulce PLC chyb nebude žádný záznam.
%SC0013	IO_PRES	Nastaví se, pokud existuje alespoň jeden záznam v tabulce chyb I/O. Vynuluje se, když v tabulce I/O chyb nebude žádný záznam.
%SC0014	HRD_FLT	Nastaví se při výskytu chyby hardwaru. Vynuluje se, když v obou tabulkách chyb nebudou žádné záznamy.
%SC0015	SFT_FLT	Nastaví se při výskytu chyby 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 (Instrukce)

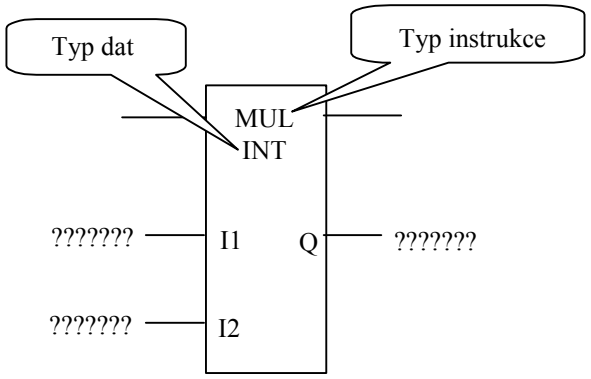
Některé funkce jsou velmi jednoduché, například funkce Hlavního řídicího relé (MCR), která je uvedena v hranatých závorkách jako zkratka názvu funkce:

```
-[ MCR ]-
```

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

Příklad obecného funkčního bloku zobrazený níže je instrukce násobení (MUL). Její části jsou typické pro mnoho funkčních bloků. Avšak počet a typy používaných parametrů se mohou značně lišit podle typu funkčních bloků. Horní část funkčního bloku udává název funkce. Může také udávat typ dat, v tomto případě celé číslo se znaménkem.

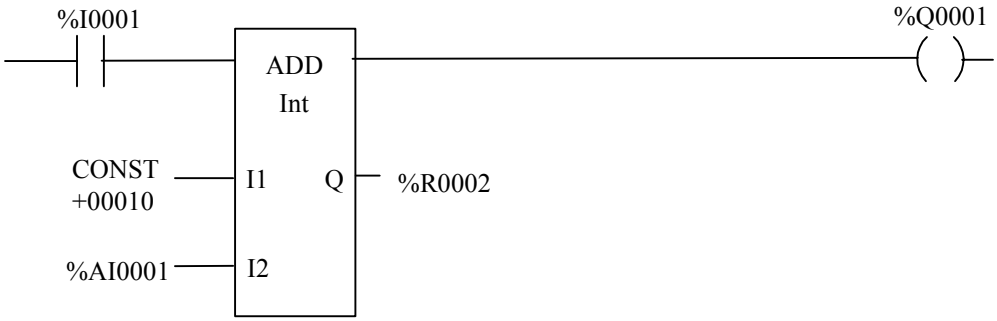
Mnoho programových funkcí (instrukcí) umožňuje zvolit typ dat pro funkci po zvolení samotné funkce. Například typ dat pro funkci MUL se může změnit na celé číslo se znaménkem s dvojnásobnou délkou (D_INT). Další informace o typech dat jsou uvedené na začátku této kapitoly.



Parametry funkčního bloku (instrukce)

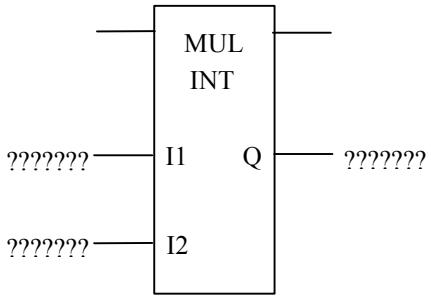
Každý řádek vstupující do levé strany funkčního bloku představuje vstup pro tuto funkci. Existují dvě formy vstupu používaných s funkčními bloky, diskrétní a analogové. Diskrétní vstupy jsou ve stavu ON nebo OFF. Na obrázku níže povolovací kontakt %I0001 je příkladem diskrétního vstupu. Analogové vstupy mohou být buď konstanty nebo adresy. Konstanta je explicitní hodnota. Adresa je paměťová adresa hodnoty. Adresa se v zásadě používá, když se mají změnit vstupní data. Například adresa může být adresou vstupu od analogového měřícího zařízení.

V následujícím příkladu vstupní parametr I1 pro funkční blok ADD je konstanta a vstupní parametr I2 je adresa.



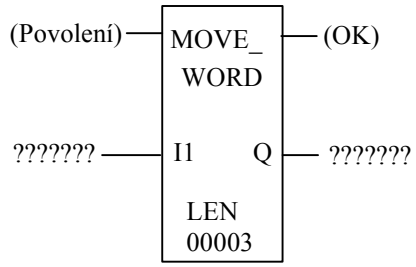
Každý řádek, který vystupuje na pravé straně funkčního bloku, představuje výstup. Výstupy mohou být buď diskrétní nebo analogové. Pokud jsou analogové, hodnota se umístí do registru (adresy). Ve výše uvedeném příkladu je výstup funkčního bloku OK diskrétní a řídí cívku %Q0001. Jeho výstup Q však podrží výslednou hodnotu matematické operace, takže ta se umístí do registru, v tomto případě %R0002.

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.

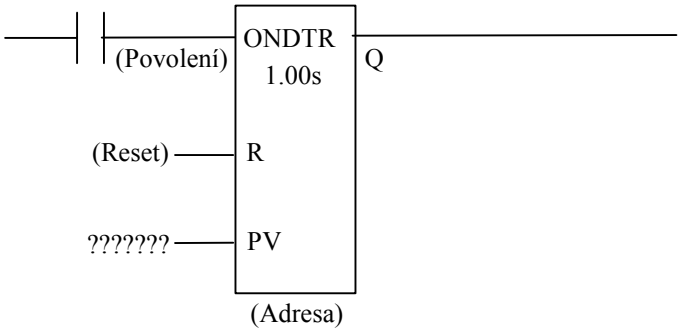


Většina funkčních bloků nemění vstupní data; místo toho při činnosti používají vstupní data a výsledek operace umístí do výstupní adresy.

U funkcí, které pracují s paměťovými adresami, je možno pro funkci zvolit délku. U následujícího funkčního bloku operand LEN zadává počet vstupních slov, které se mají přesunout (v tomto příkladu 3).

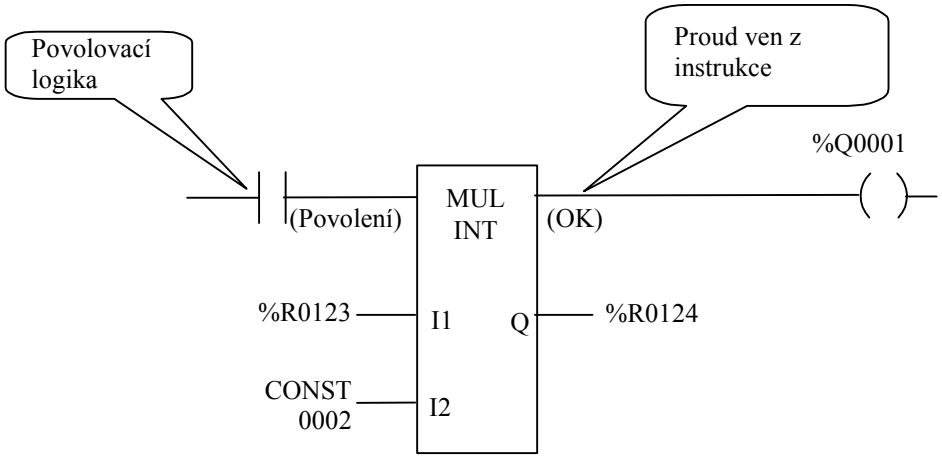


Funkce časovače, čítače, BITSEQ a ID vyžadují adresu pro umístění tří slov (registrů), ve kterých je uložena aktuální hodnota, předvolená hodnota a řídicí slovo nebo "Výskyt" funkce. První slovo tří po sobě jdoucích slov se objeví na obrazovce pod funkčním blokem zobrazeným na následujícím obrázku jako "(Adresa)".



Proud dovnitř a ven z funkce

Proud teče do vstupu funkčního bloku Povolení vlevo nahoře přes povolovací logiku. Většina funkčních bloků má proudový výstup nazývaný výstup "OK". Pokud se funkční blok vykoná správně, výstup OK přejde do jedničky a propustí proud. Pokud k výstupu OK bude připojeno jiné zařízení, například jak je znázorněno níže, toto zařízení bude povoleno. Použití výstupu OK je však pro mnoho funkčních bloků volitelné, protože jejich hlavní účel je na výstupu Q získat výsledek operace (v příkladu níže násobení).



Poznámka

Pokud budete používat programovací software Logicmaster, funkční bloky 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.

Proud 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 pět sekund, použije se sekvence teplého startu.

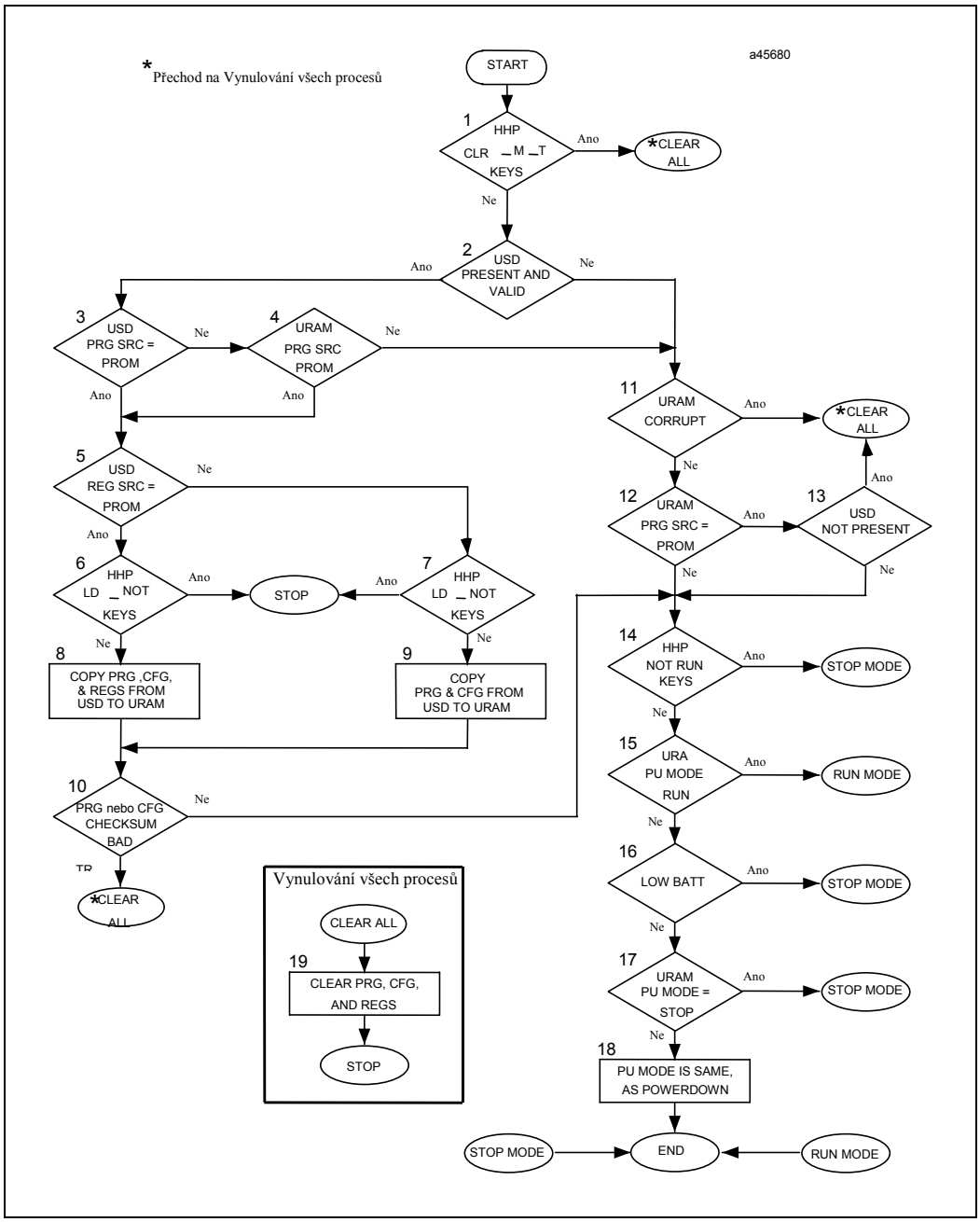
Zapnutí napájení

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

1. CPU provede vlastní diagnostiku. 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. Pokud například v softwarové konfiguraci bude specifikovaný nějaký modul, ale ve skutečné hardwarové konfiguraci bude jiný modul, tento stav bude pokládaný za chybu a bude se generovat chybové hlášení.
5. Pokud softwarová konfigurace nebude existovat, CPU použije vestavěnou 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. 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 (PRG SRC = Zdrojový program)
CFG = Uživatelská konfigurace
REGS = Uživatelské registry (adresy %I, %Q, %M, %G, %R, %AI a %AQ).
USD = Uživatelské paměťové zařízení, buď EPROM, EEPROM nebo flash zařízení.
URAM = Energeticky nezávislá paměť RAM obsahující PRG, CFG a REGS.
HHP = Ruční programovací zařízení
PU = Zapnutí napájení
CLR = Smazání
BATT = Baterie

VYSVĚTLENÍ TEXTU VE VÝVOJOVÉM DIAGRAMU:

(1) Došlo ke stisknutí tlačítek <CLR> a <M_T> na HHP (Ruční programovací zařízení) během zapínání napájení k vynulování celé URAM?
(2) Je nainstalovaná USD (uživatelské paměťové zařízení) a je informace na USD platná?
(3) Je parametr PRG SRC v USD nastavený na PROM, že se má načíst PRG (programová logika) a CFG (konfigurace) 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 (registry) ze zařízení USD?
(6 a 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? To platí pouze pro CPU 311-341. Pro CPU 350-364 a 374 se předpokládá, že je přítomno USD.
(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, 350-374 a 28-bodové 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 systémové 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 ve 28-bodovém Micro a PLC Series 90-30 modely 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 (TOD) 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 a pomocí ručního programovacího zařízení (HHP). Avšak počínaje CPU (350-364) s firmwarem verze 8.00, pokud parametr CPU *Mem. Protect* bude nastavený na *Enabled*, HHP nebude schopný hodiny TOD změnit, pokud klíček CPU bude v poloze ON. Všimněte si, že funkce ochrany pomocí klíčku platí pouze pro CPU 350—374 (ostatní CPU klíček nemají).

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 pro CPU 311-341 a 500 milisekund pro CPU 350-374; 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 nebo vyšší.

Časovač konstantního cyklu

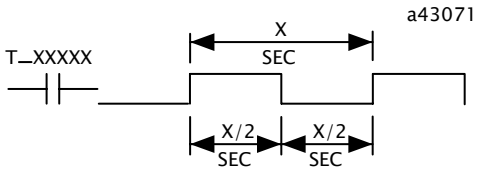
Časovač konstantního cyklu řídí délku programového cyklu, když PLC Series 90-30 bude pracovat v režimu **CYKLUS S KONSTANTNÍ DOBOU**. V tomto režimu činnosti spotřebuje každý cyklus stejně dlouhou dobu. Hodnota časovače konstantního cyklu se nastaví programovacím zařízením a může mít hodnotu od 5 do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

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 PLC cyklu 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 mění pouze během části správy PLC cyklu. 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 k 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.

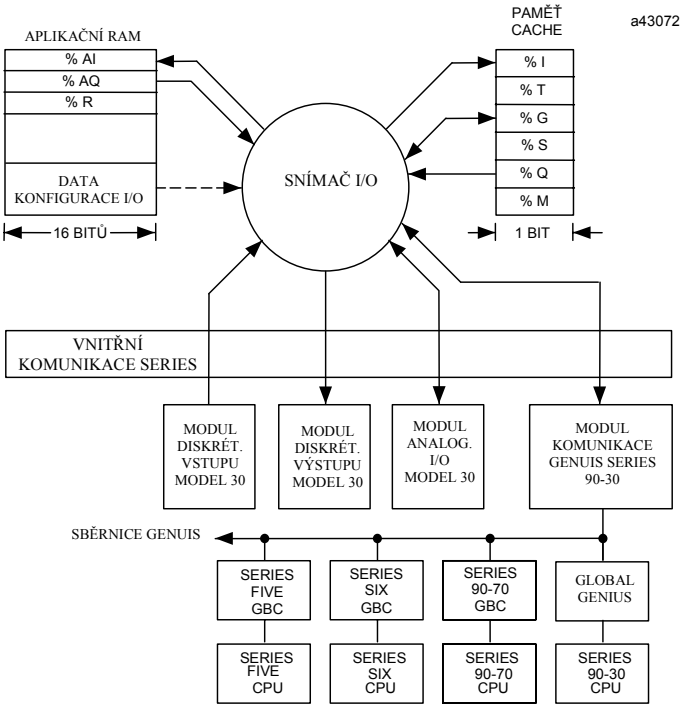
Část 6: I/O Systémy Series 90-30, 90-20 a Micro

I/O systém PLC vytváří rozhraní mezi PLC Series 90-30 a uživatelem dodaným zařízením a vybavením. I/O moduly Series 90-30 se zasunují přímo do pozic v základních deskách Series 90-30. Počet podporovaných I/O modulů Series 90-30 závisí na modelu CPU:

- CPU modely 350-374 podporují až 79 I/O modulů. Tato CPU podporují až osm sestav, což zahrnuje sestavu CPU plus celkem sedm expanzních a/nebo vzdálených sestav.
- CPU modely 331, 340 a 341 podporují až 49 I/O modulů. Tato CPU podporují až pět sestav, což zahrnuje sestavu CPU plus celkem čtyři expanzní a/nebo vzdálené sestavy.
- CPU modely 311 a 313 (základní desky s 5 pozicemi) podporují až 5 I/O modulů Series 90-30. CPU model 323 (základní deska s 10 pozicemi) podporuje až 10 I/O modulů Series 90-30. Tato tři CPU nepodporují expanzní nebo vzdálené sestavy.

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 I/O struktuře v 90-20 najdete v *Návodu pro použití programovatelného řadiče Series 90™-20* (GFK-0551). Další informace o struktuře I/O 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

Ověřte si u svého místního prodejce GE Fanuc, jestli uvedené moduly jsou dostupné. Publikace, které obsahují informace o specifikaci a zapojení jednotlivých I/O modulů Series 90-30 viz sloupec "**Číslo publikace"**.

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 střid.	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, rychlá	GFK-0898
IC693MDL644	16	24 V ss záporná logika, rychlá	GFK-0898
IC693MDL645	16	24 V ss kladná/záporná logika	GFK-0898
IC693MDL646	16	24 V ss kladná/záporná logika, rychlá	GFK-0898
IC693MDL652	32	24 V ss kladná/záporná logika	GFK-0898
IC693MDL653	32	24 V ss kladná/záporná logika, rychlá	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, 2 A	GFK-0898
IC693MDL730	8	12/24 V ss kladná logika, 2 A	GFK-0898
IC693MDL731	8	12/24 VDC záporná kladná, 2 A	GFK-0898
IC693MDL732	8	12/24 V ss kladná logika, 0.5 A	GFK-0898
IC693MDL733	8	12/24 V ss záporná logika, 0.5 A	GFK-0898
IC693MDL734	6	125 V ss kladná/záporná logika, 2 A	GFK-0898
IC693MDL740	16	12/24 V ss kladná logika, 0.5 A	GFK-0898
IC693MDL741	16	12/24 V ss záporná logika, 0.5 A	GFK-0898
IC693MDL742	16	12/24 V ss kladná logika, 1 A	GFK-0898
IC693MDL750	32	12/24 V ss záporná logika	GFK-0898
IC693MDL751	32	12/24 V ss kladná logika, 0.3 A	GFK-0898
IC693MDL752	32	5/24 V ss (TTL) záporná logika, 0.5 A	GFK-0898
IC693MDL753	32	12/24 V ss kladná/záporná logika, 0.5 A	GFK-0898
IC693MDL760	16	11 pneumatických a pět 24 V ss, kladná logika, 0,5 A	GFK-1881
IC693MDL930	8	Relé, N.O., 4 A oddělené	GFK-0898
IC693MDL931	8	Relé, záporná logika a Form C, oddělené	GFK-0898
IC693MDL940	16	Relé, N.O., 2 A	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 IC693APU301	Motion Mate APM Modul, 1 osa – vlečný režim Motion Mate APM Modul, 1 osa – standardní režim	GFK-0781 GFK-0840
IC693APU302 IC693APU302 IC693MCS001/002*	Motion Mate APM Modul, 2 osy – vlečný režim Motion Mate APM Modul, 2 osy – standardní režim Power Mate J Systém řízení pohybu (1 a 2 osy)	GFK-0781 GFK-0840 GFK-1256
IC693DSM302 IC693DSM314	Modul digitálního serva Motion Mate Modul digitálního serva Motion Mate	GFK-1464 GFK-1742
IC693APU305 IC693CMM321	Modul I/O procesoru Modul komunikace Ethernet	GFK-1028 GFK-1541
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
IC693PBM200	Profibus Master Modul	GFK-2121
IC693PBS201	Profibus Slave Modul	GFK-2193
IC693PCM300	PCM, 160 KBajtů (35 KBajtů uživatelský program MegaBasic)	GFK-0255
IC693PCM301	PCM, 192 KBajtů (47 KBajtů uživatelský program MegaBasic)	GFK-0255
IC693PCM311	PCM, 640 KBajtů (190 KBajtů uživatelský program MegaBasic)	GFK-0255
IC693PTM100/101	Power Transducer Modul (PTM)	GFK-1734
IC693TCM302/303	Modul řízení teploty (TCM), osm kanálů	GFK-1466

* Zastaralý. Uvedeno pouze pro informaci.

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. Pro nové instalace doporučujeme tento modul nepoužívat; místo něj použijte modul rozšířené GCM, který má mnohem větší kapacitu.

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 374 (verze 10.0 a pozdější) podporuje připojení k síti Ethernet přes dva vestavěné full-duplexní Ethernet porty 10BaseT/100BaseTx s automatickým nastavením.

CPU364 i CPU374 podporují 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. V Uživatelském manuálu ke svému programovacímu softwaru najdete podrobnosti jak vytvořit, přenášet, editovat a vytisknout programy. 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.

Kapitola 3

Výklad a oprava chyb

Tato kapitola pomáhá při lokalizaci chyb PLC systémů Series 90-30, 90-20 a 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. Nebo hlášený stav může působit pouze jako výstraha, například signál nízkého napětí baterie jako indikace, že je nutno vyměnit baterii pro zálohování paměti. Avšak některé stavy uvedené v chybových tabulkách nejsou hlášením chyb. Pokud například přidáte do PLC nový modul, v tabulce chyb I/O bude záznam "Přidání I/O modulu".

Alarmový procesor

Chyba je stav nebo samotná porucha. Když do CPU přijde a zpracuje se chyba, tento stav se nazývá **alarm.** Firmware v CPU, které tento stav zpracovává, se nazývá alarmový procesor. Uživatelské rozhraní alarmového procesoru 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	Neodpovídající moduly 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
Význam 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	Diagnostika	I/O	io_flt	any_flt	io_pres	los_iom
Ztráta nebo chybějící přídavný modul	Diagnostika	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	Diagnostika	PLC	sy_flt	any_flt	sy_pres	low_bat
Tabulka chyb PLC plná	Diagnostika	—	sy_full			
Tabulka chyb I/O plná	Diagnostika	—	io_full			
Chyba aplikace	Diagnostika	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	Diagnostika	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	Diagnostika	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. Kroky při chybě

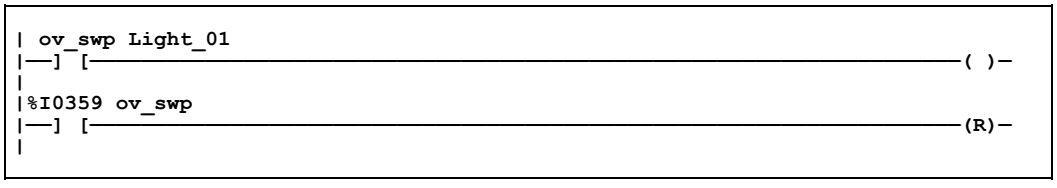
Závažnost chyby	Odezva CPU
Fatální	Záznam chyby do tabulky chyb Nastavení adres chyb Přechod do režimu **STOP**.
Diagnostika	Záznam chyby do tabulky chyb Nastavení adres chyb
Informační	Záznam chyby do tabulky chyb

Když se zjistí chyba, CPU použije krok při chybě odpovídající této chybě. Kroky při chybě nelze u PLC Series 90-30 PLC, Series 90-20 nebo Series 90 Micro nakonfigurovat.

Adresy chyb

Adresy chyb systému v Series 90-30 jsou souhrnné adresy chyb jednoho typu. 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 bit systémové chyby, který se nastaví a pak vynuluje, jak je znázorněno na následujícím obrázku. V tomto příkladu se cívka Light_01 sepne, když se sepne systémový kontakt OV_SWP (%SA0002), který indikuje, že došlo k překročení cyklu. Kontakt OV_SWP a cívka Light_01 se vypnou, když dojde k sepnutí kontaktu %I0359, protože sepnutí kontaktu %I0359 sepne resetovací cívku OV_SWP.



Adresy stavu systému

Procesor alarmu udržuje stavy 128 systémových stavových bitů v paměti %S. Mnoho z těchto stavových adres udává, kde došlo k chybě a o jaký typ chyby jde. Stavové adresy jsou přiřazené pamětem %S, %SA, %SB a %SC a každá z těchto adres má svou přezdívku. Například stavový bit %SA0009 má přezdívku CFG_MM a jeho přechod do jedničky indikuje nesoulad v konfiguraci.

Tyto adresy je možno použít podle potřeby v aplikačním programu. Seznam systémových stavových adres najdete v kapitole 2, “Činnost systému”.

Další důsledky chyb

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

Chyba	Popis vlivu
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 ukládání do PLC dojde k přerušení komunikace mezi PLC a programovacím zařízením nebo dojde k nějaké jiné chybě, která ukončí načítání (ukládání), zaznamená se Porucha sekvence ukládání PLC. Pokud tato chyba bude v systému, PLC nepřejde do režimu **RUN**. Aby se obnovila činnost, je nutno chybu smazat. To lze provést smazáním chyby na příslušné obrazovce tabulky chyb.

Zobrazení tabulky chyb PLC

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

Chyby se ukládají do PLC, takže pokud programovací software bude v režimu **OFFLINE**, v této tabulce chyb se nezobrazí žádné chyby. Pokud programovací software bude buď v režimu **ONLINE** nebo v režimu **MONITOR**, budou se zobrazovat data chyb PLC. V režimu **ONLINE** je možno chyby smazat, i když tato funkce však může být chráněna heslem.

Po smazání se chyby, které přetrvávají, nezapíší znovu do tabulky (s výjimkou chyby "Nízké napětí baterie), pokud by se však neprovedlo vypnutí a zapnutí napájení nebo se neuložila nová konfigurace.

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.

Chyby se ukládají do PLC, takže pokud programovací software bude v režimu **OFFLINE**, v této tabulce chyb se nezobrazí žádné chyby. Pokud programovací software bude buď v režimu **ONLINE** nebo v režimu **MONITOR**, budou se zobrazovat data chyb I/O. V režimu **ONLINE** je možno chyby smazat, i když tato funkce však může být chráněna heslem.

Po smazání se chyby, které přetrvávají, nezapíší znovu do tabulky, pokud by se však neprovedlo vypnutí a zapnutí napájení nebo se neuložila nová konfigurace.

Přístup k doplňkovým informacím

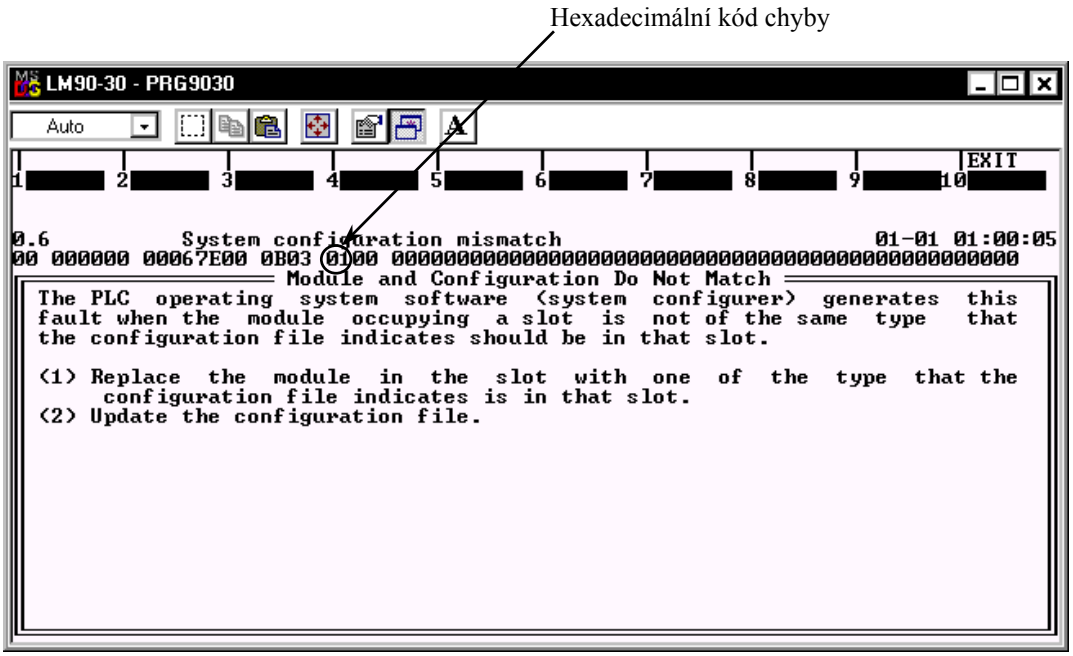
Tabulky chyb obsahují základní informace o dané chybě. Doplňkové informace týkající se jednotlivých chyb je možno zobrazit pomocí programovacího softwaru. Kromě toho, programový software může provést hexadecimální výpis kódu každé chyby.

Poslední údaj "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 chybový kód. Pracovník servisu vám pak poskytne další instrukce ohledně, kroků, které musíte provést.

Následující obrázek obrazovky detailu chyby Logicmaster zobrazuje další informace o chybě a hexadecimální kód chyby popisovaný výše. (Chybový kód jsou první dvě hexadecimální číslice v páté skupině čísel zleva.) Abyste se dostali na tuto obrazovku, pomocí tlačítek se šipkou pro ovládání kurzoru zvolte Tabulku chyb (Ztráta I/O modulu) a pak pomocí klávesy F10 zvolte "zoom". Chcete-li se vrátit na obrazovku tabulky chyb, stiskněte buď klávesu Escape nebo kombinaci kláves Shift a F10.



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 (zleva), 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.

Kroky při chybě

- **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; PLC zůstane v režimu ***RUN***.
- **Informační** chyby se prostě jen zaznamenají do tabulky chyb PLC; PLC zůstane v režimu ***RUN***.

Ztráta nebo chybějící přídavný modul

Chybová skupina **ztráta nebo chybějící přídavný modul** se vyskytne, když přídavný modul nebude 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	Když zkusíte softwarový reset (například stisknutím tlačítka Reset), CPU PLC není schopno obnovit komunikaci s přídavným modulem.
Oprava:	(1) Zopakujte postup softwarového resetu doporučeného pro tento modul. (2) Vyměňte přídavný modul. (3) Vypněte napájení systému. Ověřte, že modul je v sestavě správně zasazený 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 pozici.

Reset, přidání nebo přespočetný přídavný modul

Chybová skupina **Reset, přidání nebo přespočetný přídavný modul** se vyskytne, když přídavný modul (PCM, ADC, atd.) přejde do stavu on-line, bude resetovaný nebo se v sestavě zjistí modul, ale žádný z nich není uvedený v konfiguraci. Závažnost chyby u této skupiny je **Diagnostická**.

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 slutu 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 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 systému s CPU 311, 313 nebo 323 nebo v expanzní nebo vzdálené sestavě se nachází PCM nebo modul typu PCM.
Oprava:	Fyzicky opravte stav odstraněním PCM nebo modulu typu PCM nebo nainstalujte CPU, které podporuje modul. POZNÁMKA: Tyto moduly musí být umístěné pouze v CPU sestavě s CPU, která je podporuje.
Chybový kód:	26
Název:	Modul je zaneprázdněný – modul zatím nepřijal konfiguraci
Popis	Modul není schopný v tomto okamžiku přijmout novou konfiguraci, protože je zaneprázdněný.
Oprava:	Nechejte modul dokončit současnou operaci a konfiguraci uložte znovu.
Chybový kód:	51
Název:	Funkce END se vykonala z akce sekvenčního funkčního grafu 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 (načtené programovacím softwarem). 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) generuje 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 firmware 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 **okamžitě** 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 by se v prodávaných produktech neměly vyskytnout; je možno se s nimi setkat pouze během vývoje firmwaru ve výrobním závodu.
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 (snímač I/O) vygeneruje tuto chybu, když bude požadovat, aby manažer paměti zrušil přiřazení bloku systémové paměti a zrušení přiřazení 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 by se v prodávaném systému neměla vyskytnout; je možno se s ní setkat pouze během vývoje firmwaru ve výrobním závodu.
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 software PLC. Tyto chyby by se v prodávaných produktech neměly vyskytnout; je možno se s nimi setkat pouze během vývoje firmwaru ve výrobním závodu.
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 by se v prodávaných produktech neměly vyskytnout; je možno se s nimi setkat pouze během vývoje firmwaru ve výrobním závodu.
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:	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. 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**. Další informace k této chybě najdete v části "Další důsledky chyb" výše v této kapitole.

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 modelu 30 ve slotu již neodpovídá na povely z CPU PLC nebo když konfigurační soubor bude indikovat, že I/O modul má být v pozici nainstalovaný a že v této pozici 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, není nutno provádět žádné kroky, nebo u vzdálené sestavy bylo provedeno vypnutí a zapnutí napájení. (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) Vyndejte modul, pokud se zde nachází omylem. (2) Aktualizujte a obnovte konfigurační soubor tak, aby zahrnoval další modul, pokud zde má být.

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, například paměťových typů %Q a %M. 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 (kontakt) 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 —| |—

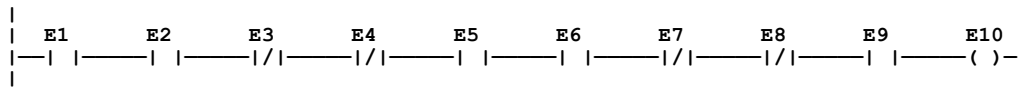
Normální spínací kontakt funguje jako spínač, který propouští proud, pokud související adresa bude ve stavu ON (v logice 1).

Normální rozpínací kontakt —|/|—

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

Příklad

Následující příklad ukazuje příčku s 10 prvky s přezdívkami E1 až E10 (informace o přezdívkách viz kapitola 2). 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.

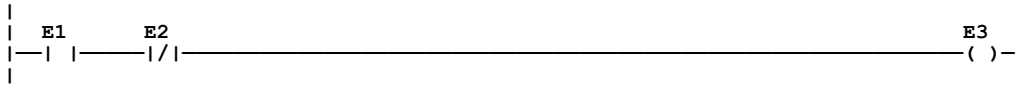


Cívka —( )—

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 globálními adresami Genius (%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.

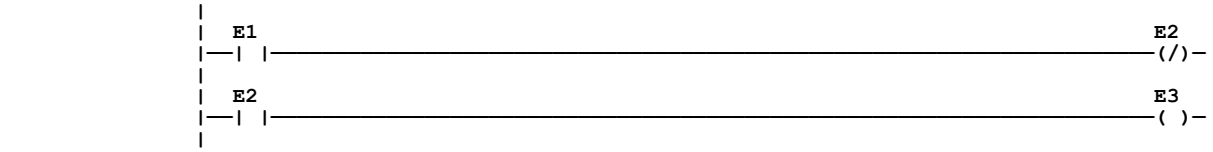


Negovaná cívka —(/)—

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 globálními adresami Genius (%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 —(↓)—

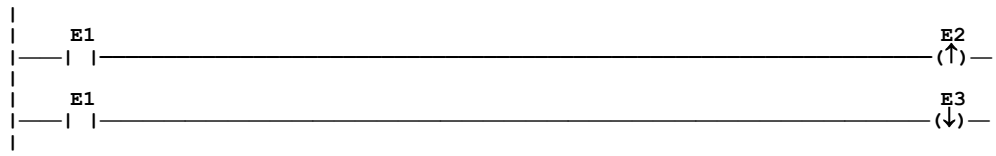
Pokud adresa související s touto cívkou bude ve stavu OFF a když cívkou ***nepoteče*** proud, adresa se nastaví do stavu ON a všechny kontakty související s touto cívkou během jednoho cyklu změní svůj stav.

Adresa použitá s přechodovou cívkou se může použít pouze jako cívka v aplikačním programu, aby se zachovala jednorázová povaha cívky.

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 ve stavu ON nebo ve stavu OFF. Když cívkou SET poteče proud, její adresa zůstane ve stavu ON (bez ohledu na to, jestli proud nadále poteče samotnou cívkou), dokud adresa nebude resetovaná jinou cívkou.

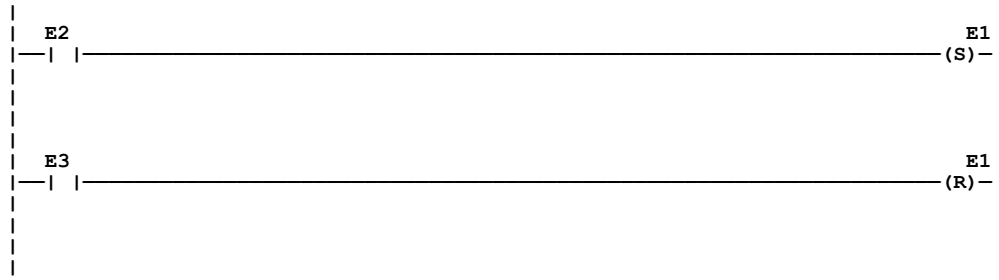
Cívka RESET —(R)—

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.

Příklad

V následujícím příkladu cívka E1(S) se přepne do stavu ON, když adresa E2 bude ve stavu ON. I když E2 přejde do stavu OFF, cívka E1 zůstane ve stavu ON, dokud E3 nenabudí cívku E1(R).

POZNÁMKA: Pokud E2 i E3 budou současně ve stavu ON, cívka E1 bude ve stavu OFF. Je to z toho důvodu, že příčky se čtou shora dolů, takže stav resetovací cívky na druhé příčce bude poslední, která se zapíše do výstupní tabulky. Pokud by pořadí příček bylo obráceno, nastavovací cívka by se četla jako poslední, takže E1 by bylo ve stavu ON, pokud E2 a E3 budou současně 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, které se na obrazovce objeví jako přímé čáry, se používají ke spojení prvků žebříkové logiky mezi funkcemi. Jejich účelem je dokončit tok logiky ("proud") ve schématu logiky zleva doprava.

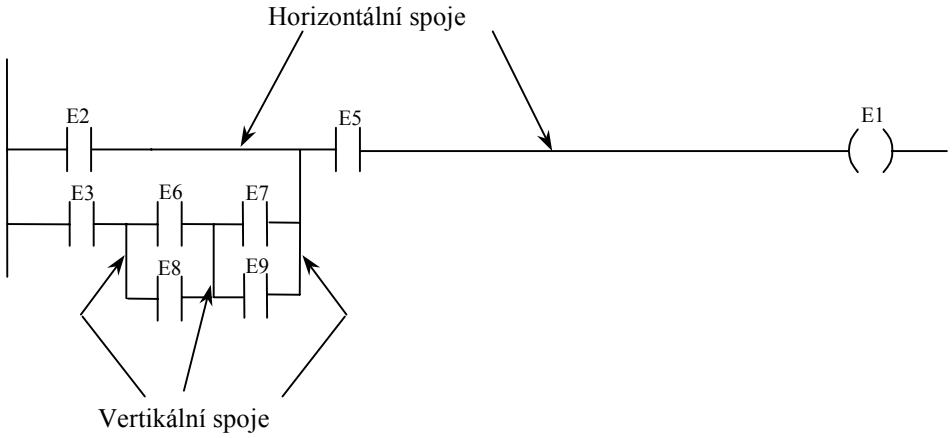
Poznámka

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 používá několik spojů:

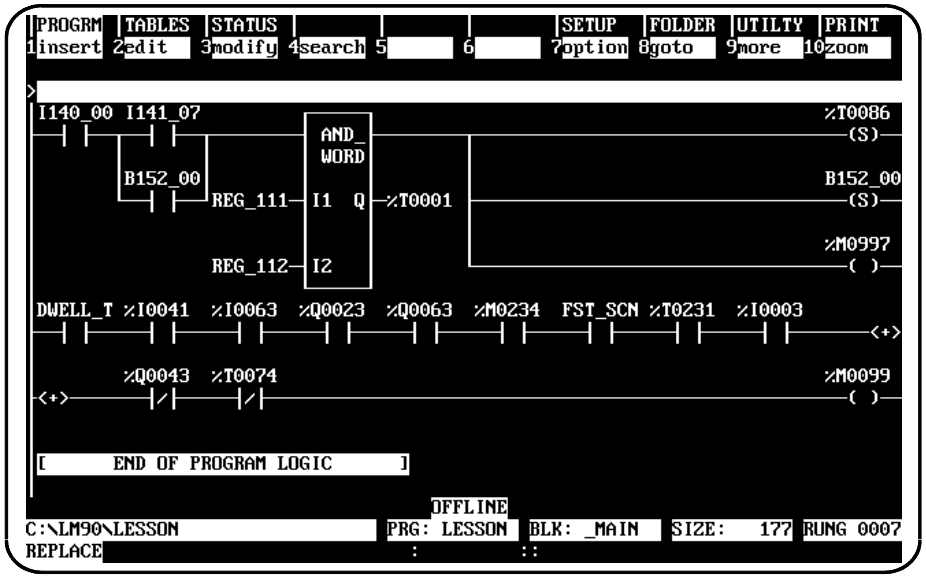
- Horizontální spoje spojují kontakt E2 s kontaktem E5 a kontakt E5 s cívkou E1.
- Vertikální spoje spojují kontakt E8 přes kontakt E6, kontakt E9 přes kontakt E7 a kontakty E7/E9 na pravé straně s křížením kontaktů E2 a E5.



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.

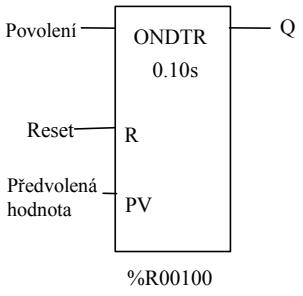
Zkratka	Funkce	Strana
ONDTR	Retentivní časovač zpoždění při zapnutí	5-3
TMR	Jednoduchý časovač zpoždění při zapnutí	5-5
OFDT	Časovač zpoždění při vypnutí	5-8
UPCTR	Vzestupný čítač	5-11
DNCTR	Sestupný čítač	5-13

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

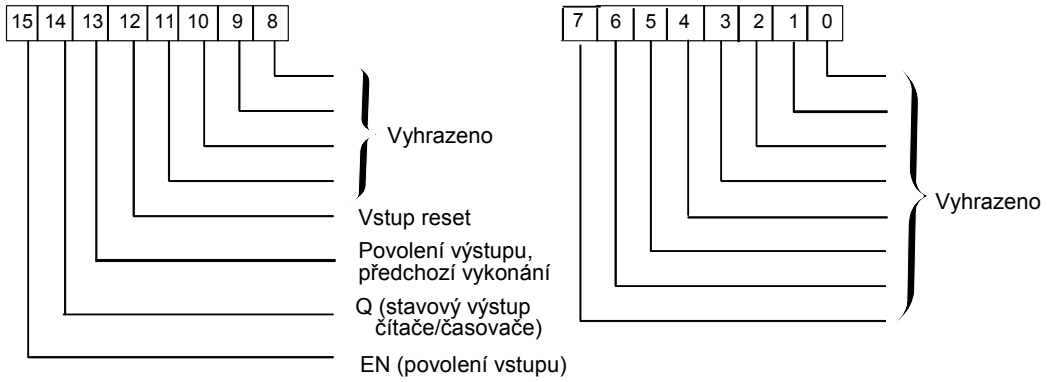
Když budete zapisovat časovač nebo čítač, musíte zapsat počáteční adresu (adresu pro slovo 1) pro tento trojslovní blok přímo pod grafickou reprezentací funkce. V následujícím příkladu tato počáteční adresa je %R00100.



Poznámka

Přesvědčte se, že adresu ve trojslovním bloku nebudete ve svém programu používat nikde jinde (toto zdvojené používání se nazývá "překrývání"). Logicmaster ***nekontroluje ani nevaruje***, jestli se bloky registrů překrývají. Pokud dojde k překrývání trojslovních bloků, časovače a čítače nebudou pracovat správně.

V řídicím slově (třetí slovo v trojslovním bloku) je uložený stav Booleovských vstupů a výstupů souvisejícího funkčního bloku, jak je znázorněno na následujícím obrázku:



Bity 0 až 11 má PLC vyhrazeno pro údržbu přesnosti časovače; tyto bity (0 až 11) se nepoužívají pro funkční bloky čítače.

Poznámka

Dejte pozor, pokud pro vstupní parametr PV (předvolená hodnota) funkce budete používat stejnou adresu jako pro druhé slovo trojslovného bloku. Pokud PV nebude konstanta, vstup PV se normálně nastaví na jiné paměťové místo než druhé slovo. Některá programovací zařízení volí použití adresy druhého slova pro vstup PV, například používají %R0102, když trojslovní 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, protože pokud by se do nich zapsalo, funkce by 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

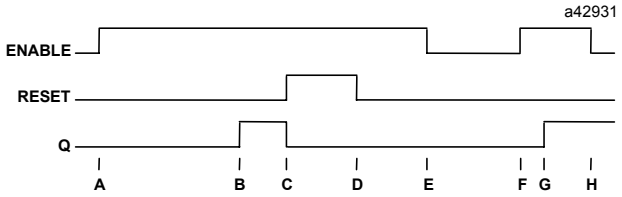
Retentivní časovač zpoždění při zapnutí (ONDTR) bude provádět inkrementaci, když do něj poteče proud, a když se proud zastaví, hodnotu si podrží. Čas se může čítat v desetinách sekundy (výchozí volba), setinách sekundy nebo tisícinách sekundy. Rozsah je 0 až +32,767 časových jednotek. Proto rozsah časování bude 0,001 až 3,276,7 sekund. Stav časovače je při výpadku napájení retentivní; při zapnutí napájení žádná automatická inicializace neproběhne.

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

Poznámka

Pokud během cyklu CPU bude povolený vícenásobný výskyt stejného čítače se stejnou referenční 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.

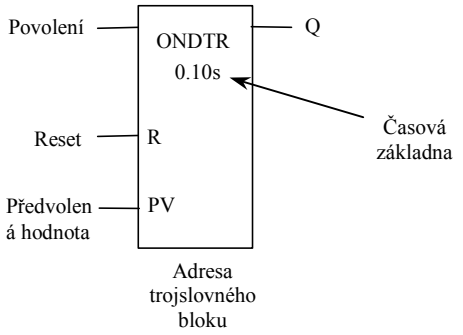
E = ENABLE přejde do nuly; časovač zastaví čítání. Celkový čas zůstane stejný.

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 u PLC řady 35x, 36x a 37x povolení pro ONDTR bude v nule, PV = 0 a do reset R bude téct proud, pak výstup bude v nule. Avšak v případě PLC 311-341 bude za stejných podmínek výstup v jedničce.



Parametry

Parametr	Popis
Adresa trojslovného bloku	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 prvního z těchto tří po sobě jdoucích slov (registrů) přímo pod grafické znázornění funkce (použití druhých dvou slov je samozřejmé). **Upozornění:** Nezapisujte do těchto tří slov pomocí jiných instrukcí. Překrývání těchto adres bude mít za následek chybnou činnost časovače.
Povolení	Když na vstup pro povolení přijde jednička, časovač spustí svou funkci.
R	Vstup resetu. Když na R přijde jednička, provede se reset jeho hodnoty na nulu. Pokud se použije vstup R, musí být připojený jedním nebo více kontakty k napájecí spojnici. To vyžaduje, aby instrukce ONDTR nebyla umístěna v příčce na první pozici (nejvíce vlevo).
PV	Vstup předvolené hodnoty. PV je hodnota, která se zkopíruje do předvolené hodnoty časovače, když se provede povolení nebo reset časovače. Časovač nastaví výstup Q do jedničky, když uplyne hodnota času PV.
Q	Výstup Q bude v jedničce, když aktuální hodnota (CV) bude větší nebo se bude rovnat předvolené hodnotě (PV).
Časová základna	Tento parametr je možno naprogramovat pro časy v inkrementech desetin (0.1), setin (0.01) nebo tisícin (0.001) sekundy. Násobením této hodnoty časové základny číslem ve vstupním parametru předvolená hodnota (PV) se získá skutečná předvolená hodnota.

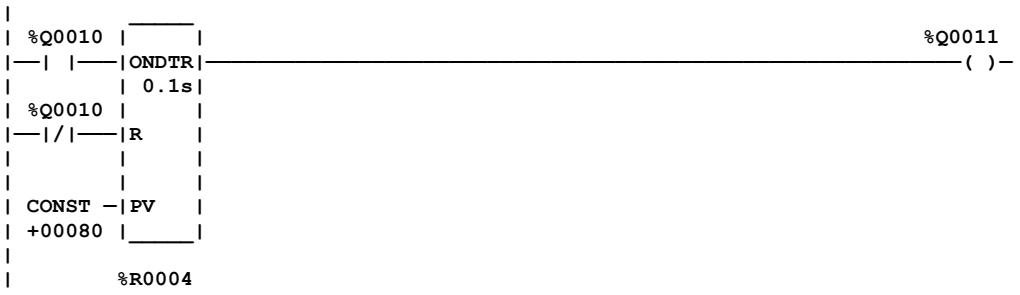
Platné typy paměti

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 nastavení výstupu (%Q0011), který se zapne 8.0 sekundy po zapnutí %Q0010, a vypne se, když se vypne %Q0010. Je to proto, že když se %Q0010 vypne, přes jeho normálně rozpínací kontakt prochází proud na resetovací vstup (R). Časová hodnota 8.0 sekund se získá vynásobením hodnoty PV (80) hodnotou časové základny (0,1s).



TMR

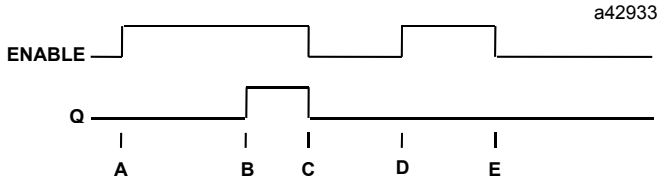
Funkce jednoduchého časovače zpoždění při zapnutí (TMR) bude provádět inkrementaci, dokud skrz ní poteče proud, a provede vynulování, když se proud zastaví. Čas se může čítat v desetinách sekundy (výchozí volba), setinách sekundy nebo tisícinách sekundy. Rozsah je 0 až +32 767 časových jednotek a proto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

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

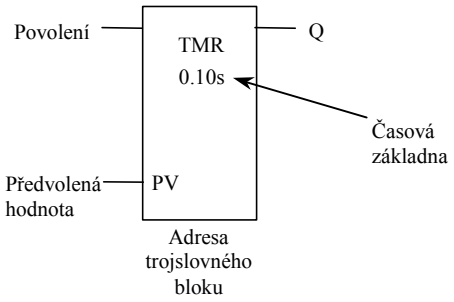
Pokud během cyklu CPU bude povolený vícenásobný výskyt stejného čítače se stejnou referenční adresou, aktuální hodnoty časovačů budou stejné.

Hodnota uplynulého času časovače (CV - aktuální hodnota) bude pokračovat v čítání, dokud logika povolení bude ve stavu ON. Když se aktuální hodnota (CV) 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 (32 767 časových jednotek). Když vstup 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 čítání 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 jedničky před tím, než aktuální hodnota dosáhne předvolené hodnoty PV; Q zůstane v nule; časovač zastaví čítání času a aktuální čas se vynuluje.



Parametry

Parametr	Popis
Adresa trojslovného bloku	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 prvního z těchto tří po sobě jdoucích slov (registrů) přímo pod grafické znázornění funkce (použití druhých dvou slov je samozřejmé). **Upozornění:** Nezapisujte do těchto tří slov pomocí jiných instrukcí. Překrývání těchto adres bude mít za následek chybnou činnost časovače.
Povolení	Když na vstup pro povolení přijde jednička, časovač spustí svou funkci. Když vstup povolení přejde do nuly, aktuální hodnota se resetuje na nulu a Q přejde do nuly.
PV	Vstup předvolené hodnoty. PV je hodnota, která se zkopíruje do předvolené hodnoty časovače, když se provede povolení nebo reset časovače. Časovač nastaví výstup Q do jedničky, když uplyne hodnota času PV.
Q	Výstup Q bude v jedničce, když aktuální hodnota (CV) bude větší nebo se bude rovnat předvolené hodnotě (PV).
Časová základna	Tento parametr je možno naprogramovat pro časy v inkrementech desetin (0.1), setin (0.01) nebo tisícin (0.001) sekundy. Násobením této hodnoty časové základny číslem ve vstupním parametru předvolená hodnota (PV) se získá skutečná předvolená hodnota.

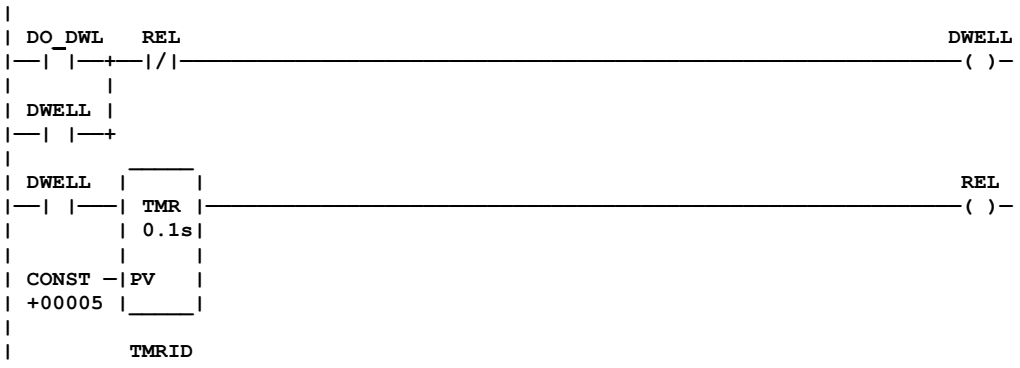
Platné typy paměti

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 TMR používá k řízení doby, po kterou cívka DWELL bude zapnutá. Proces časování začne, když se normální rozpínací (krátkodobý) kontakt DO_DWL sepne, čímž se sepne cívka DWELL. Když kontakt DO_DWL bude rozepnutý, kontakt DWELL udržuje cívku DWELL nabuzenou (“přidrženou”); časovač povolí také další kontakt DWELL. Když časovač dosáhne své předvolené hodnoty poloviny sekundy, cívka REL se nabudí. Normální rozpínací kontakt REL se rozepne, přeruší přidržený stav cívky DWELL, která se rozepne. Kontakt DWELL na vstupu povolení časovače se rozepne, čímž se přeruší proud do časovače, resetuje se aktuální hodnota a odbudí se cívka REL. Obvod pak bude připravený k další aktivaci kontaktu DO_DWL.



OFDT

Načtená hodnota časovače zpoždění při vypnutí (OFDT) se bude inkrementovat, dokud proud nepoteče, a zresetuje se 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, což dává rozsah 0,001 až 3 276,7 sekundy. Stav časovače je při výpadku napájení retentivní; při zapnutí napájení žádná automatická inicializace neproběhne.

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, jeho výstup zůstane přechodně v jedničce a časovač začne načítat čas do aktuální hodnoty; jakmile načtená hodnota dosáhne předvolené hodnoty, výstup přejde do nuly.

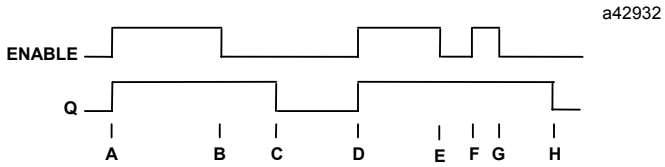
Poznámka

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

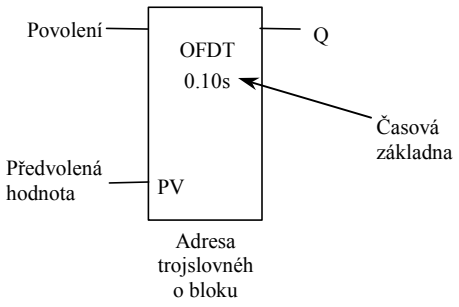
OFDT nebude propouštět proud, pokud předvolená hodnota bude nula nebo záporná hodnota.

Při každém vyvolání funkce nastavením povolovací logiky do stavu OFF (na vstupu povolení) 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).
E = ENABLE přejde do nuly; časovač začne načítat čas.
C = Hodnota CV se rovná hodnotě PV; Q přejde do nuly a časovač zastaví načítání času.
D = ENABLE přejde do jedničky; časovač se resertuje reset (CV = 0), Q přejde do jedničky.
E = ENABLE přejde do nuly; časovač začne načítat čas; Q zůstane v jedničce.
F = ENABLE přejde do jedničky; časovač se resetuje (CV = 0); Q zůstane v jedničce.
G = ENABLE přejde do nuly; časovač začne načítat čas; Q zůstane v jedničce.
H = Hodnota CV se rovná hodnotě PV; Q přejde do nuly a časovač zastaví načítání 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 trojslovného bloku	Časovač 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 prvního z těchto tří po sobě jdoucích slov (registrů) přímo pod grafické znázornění funkce (použití druhých dvou slov je samozřejmé). **Upozornění:** Nezapisujte do těchto tří slov pomocí jiných instrukcí. Překrývání těchto adres bude mít za následek chybnou činnost časovače.
povolení	Když vstup povolení bude na jedničce, výstup Q zůstane v jedničce a aktuální hodnota (CV) bude na nule. Když vstup povolení přejde do nuly, časovač začne čítat čas. Když aktuální hodnota (CV) dosáhne předvolené hodnoty (PV), časovač přestane čítat čas a Q přejde do nuly.
PV	Vstup předvolené hodnoty. PV je hodnota, která se zkopíruje do předvolené hodnoty časovače, když se provede povolení nebo reset časovače. Časovač nastaví výstup Q do nuly, když uplyne hodnota času PV.
Q	Výstup Q bude nabuzený (1), když vstup povolení bude v jedničce a (2), když aktuální hodnota (CV) bude menší než předvolená hodnota (PV) po přechodu vstupu povolení do nuly.
Časová základna	Tento parametr je možno naprogramovat pro časy v inkrementech desetin (0.1), setin (0.01) nebo tisícin (0.001) sekundy. Násobením této hodnoty časové základny číslem ve vstupním parametru předvolená hodnota (PV) se získá skutečná předvolená hodnota.

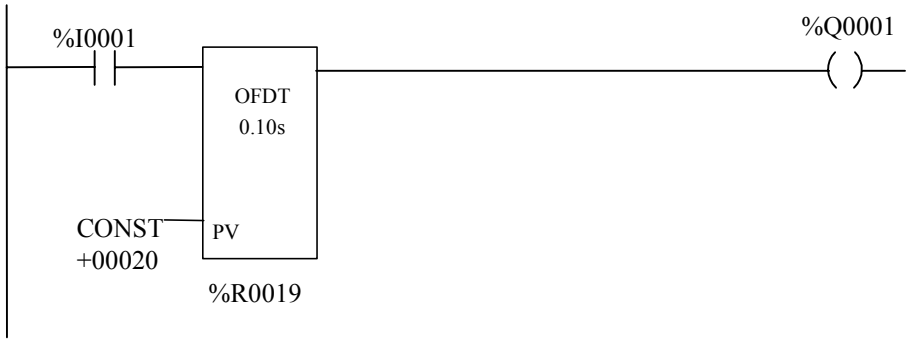
Platné typy paměti

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

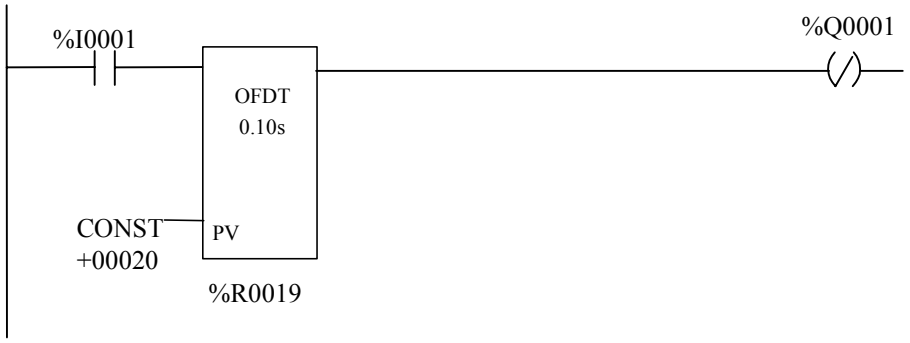
• Platná adresa nebo místo, kde funkcí může téct proud.

Příklady

V následujícím příkladu časovač OFDT zapne cívku %Q0001 při každém sepnutí kontaktu %I0001. Když se %I0001 rozepne, %Q0001 zůstane v jedničce 2 sekundy a pak přejde do nuly.



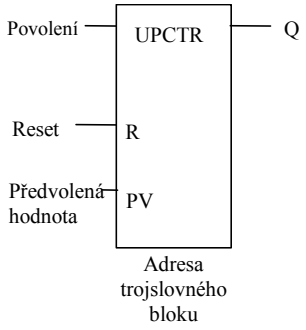
V následujícím příkladu se činnost výstupu obrátí použitím cívky s negovaným výstupem. U tohoto obvodu časovač OFDT vypne negovaný výstup cívky %Q0001 při každém sepnutí kontaktu %I0001. Když se %I0001 rozepne, %Q0001 zůstane v nule 2 sekundy a pak přejde do jedničky.



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 trojslovného bloku	Vzestupný čítač UPCTR používá k uložení následujících informací tři po sobě jdoucí slova 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 prvního z těchto tří po sobě jdoucích slov (registrů) přímo pod grafické znázornění funkce (použití druhých dvou slov je samozřejmé). **Upozornění:** Nezapisujte do těchto tří slov pomocí jiných instrukcí. Překrývání těchto adres bude mít za následek chybnou činnost časovače.
povolení	Při každém přechodu (z nuly do jedničky) na vstupu povolení se aktuální hodnota počtu (CV) inkrementuje o jedničku.
PV	Vstup předvolené hodnoty. PV je hodnota, která se zkopíruje do předvolené hodnoty čítače, když se čítač povolí nebo resetuje. Čítač nastaví výstup Q do jedničky, když načte až do hodnoty PV. Pokud předvolená hodnota bude konstanta, musí to být kladné číslo mezi 0 a 32 767.
Q	Výstup Q bude v jedničce, když aktuální hodnota počtu (CV) bude větší nebo se bude rovnat předvolené hodnotě.
R	Vstup resetu. Když vstup R přejde do jedničky, hodnota aktuálního počtu (CV) se resetuje na nulu.

Platné typy paměti

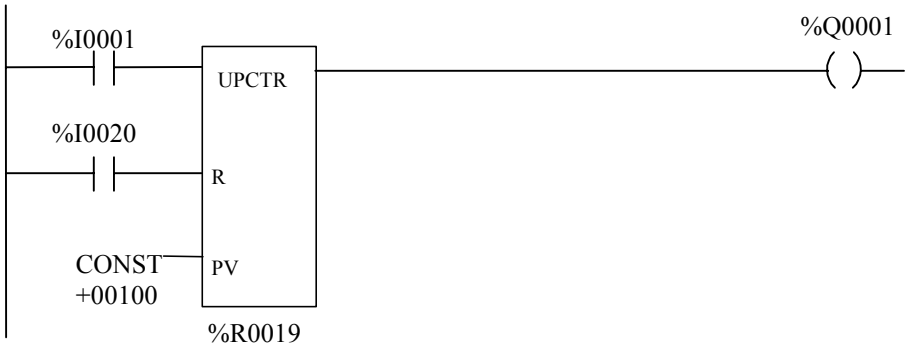
Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
adresa								•				
povolení	•											
R	•											
PV		•	•	•	•		•	•	•	•	•	•
Q	•											•

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

Příklady

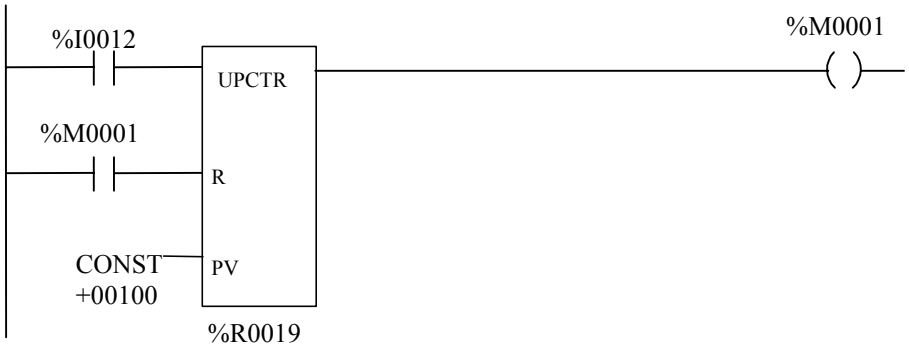
Obvod základního čítače

V následujícím příkladu bude UPCTR inkrementovat svou hodnotu aktuálního počtu (CV) o jedničku při každém přechodu %I0001 z nuly do jedničky. Vstup PV nastaví předvolenou hodnotu na 100 jednotek. Když čítač načítá 100, cívka %Q0001 se sepne. Čítač pak bude pokračovat v čítání přechodů na %I0001 za předvolenou hodnotu (100), dokud buď nedosáhne maximální hodnoty (32 767) nebo dokud se nesepne %I0020 a čítač se resetuje. %Q0001 bude v jedničce vždy, když hodnota CV bude rovna nebo větší než hodnota PV.



Obvod čítače se samočinným resetováním

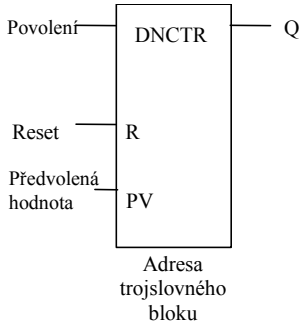
V následujícím příkladu při každém přechodu vstupu %I0012 ze stavu OFF do stavu ON čítač UPCTR přičte 1. Cívka %M0001 bude nabuzená vždy, když se napočítá 100 přechodů %I0012. Jakmile %M0001 přejde do stavu ON, kontakt %M0001 na vstupu R resetuje načtený počet na nulu a %M0001 přejde do nuly.



DNCTR

Funkce sestupného čítače (DNCTR) se používá k odečítání od předvolené hodnoty. Minimální předvolená hodnota je nula; maximální předvolená hodnota je +32 767 pulsů. 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 žádná automatická inicializace neproběhne.



Parametry

Parametr	Popis
Adresa trojslovného bloku	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 prvního z těchto tří po sobě jdoucích slov (registrů) přímo pod grafické znázornění funkce (použití druhých dvou slov je samozřejmé). **Upozornění:** Nezapisujte do těchto tří slov pomocí jiných instrukcí. Překrývání adres bude mít za následek chybnou činnost čítače.
Povolení	Při každém přechodu (z nuly do jedničky) na vstupu povolení se aktuální hodnota počtu (CV) dekrementuje o jedničku.
PV	Vstup předvolené hodnoty. PV je hodnota, která se zkopíruje do předvolené hodnoty čítače (PV) a aktuální hodnoty registru (CV), když se čítač povolí nebo resetuje. Čítač nastaví výstup Q do jedničky, když dokončí odčítání od aktuální hodnoty do nuly.
Q	Výstup Q bude v jedničce, když hodnota aktuálního počtu (CV) bude menší nebo se bude rovnat nule.
R	Vstup resetu. Když vstup R přejde do jedničky, hodnota aktuálního počtu (CV) se resetuje na předvolenou hodnotu (PV) a výstup Q přejde do nuly.

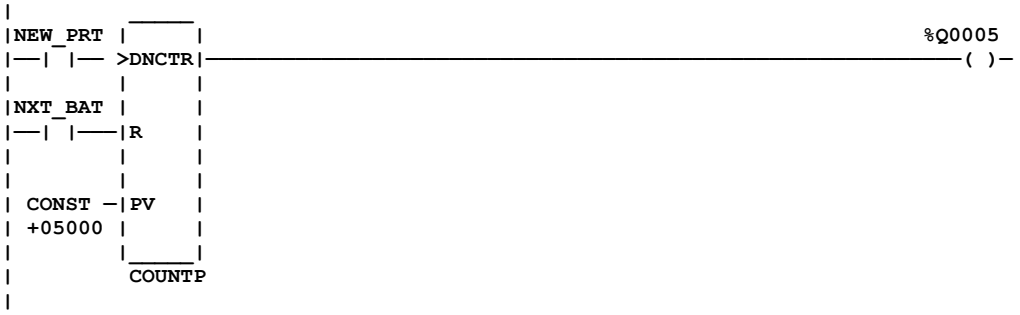
Platné typy paměti

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.

Příklady

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

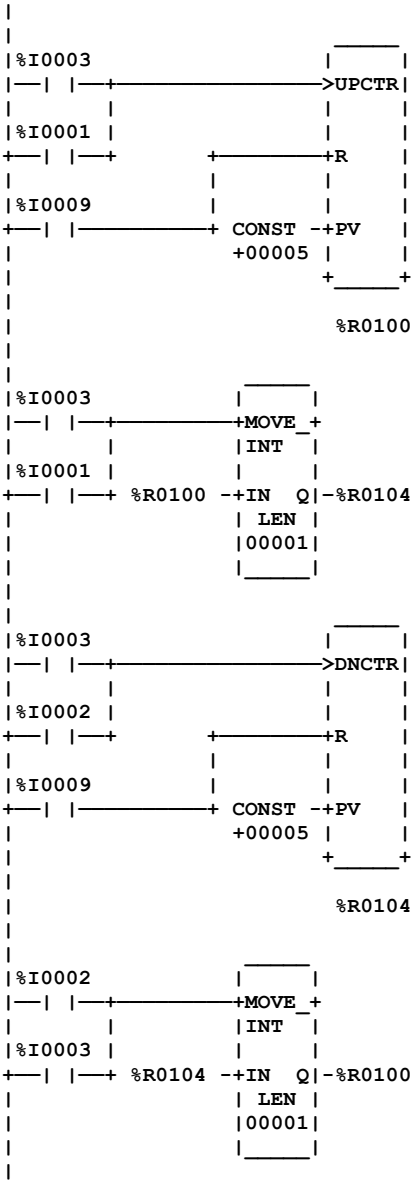


Příklady počtu dílů ve skladu

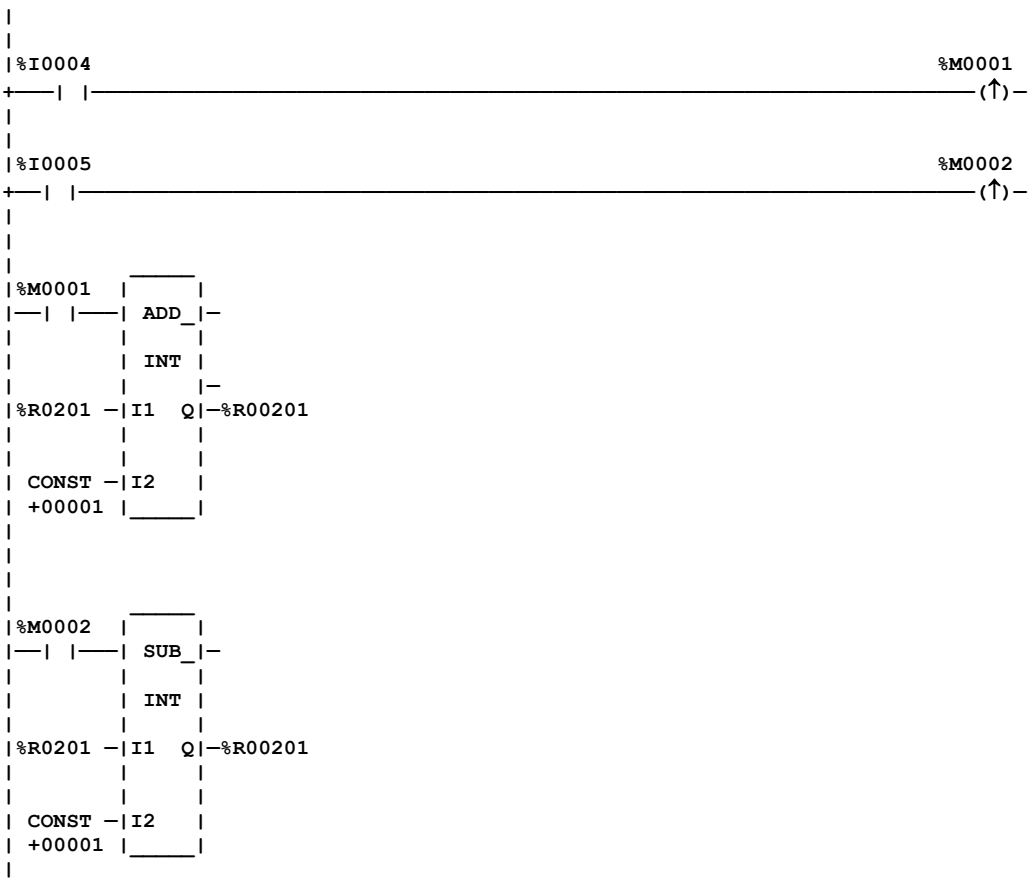
V následujícím příkladu se PLC používá ke sledování počtu dílů uložených v přechodném skladu. Pomocí instrukčního souboru Series 90-30/20/Micro existují dva způsoby, jak dosáhnout této funkce.

První způsob je použít pár vzestupný/sestupný čítač se sdíleným registrem pro celkovou nebo aktuální hodnotu. Když se díly dostanou do skladového prostoru, vzestupný čítač provede inkrementaci o 1 (sepne se %I0001) a zvýší tím aktuální hodnotu počtu dílů ve skladu o 1. Když díl bude ze skladového prostoru odcházet, sestupný čítač provede dekrementaci o 1 (sepne se %I0002) 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 přesune do registru aktuální hodnoty jiného čítače.

V následujícím příkladu %I0001 inkrementuje počet, %I0002 dekrementuje počet, %I0009 resetuje počet na nulu a %I0003, když je v jedničce, podrží počet na aktuální hodnotě bez ohledu na to, co dělají %I0001 a %I0002. Hodnotu počtu je možno načíst z %R0100.



Druhý způsob, jak sledovat pohyb ve skladu, je uvedený níže a používá funkce ADD a SUB, které na svých výstupech sdílejí společný registr %R00201. Když se zvětší počet (sepne se %I0004), instrukce ADD inkrementuje hodnotu v %R00201. Když se počet sníží (sepne se %I0005), instrukce SUB dekrementuje hodnotu v %R00201. V tomto případě se použijí přechodové cívky k vytvoření "jednorázových" vstupů pro funkce ADD a SUB. Pokud by vstupy nebyly jednorázového typu, funkce ADD a SUB by se vykonaly jednou při každém čtení, kdy jsou povolené. (Přechodové cívky nejsou s funkcemi UPDTR a DNCTR zapotřebí, protože jejich vstupy povolení mají vestavěnou funkci přechodu.) Podrobnosti k funkcím ADD a SUB najdete v kapitole 6.



Kapitola 6

Matematické funkce

Tato kapitola popisuje matematické funkce instrukční sady 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.	6-7
SQRT	Druhá odmocnina	Vypočítá druhou odmocninu celého nebo reálného čísla.	6-9
SIN, COS, TAN, ASIN, ACOS, ATAN	Trigonometrické funkce †	Vykoná příslušnou funkci reálného čísla na vstupu IN.	6-11
LOG, LN EXP, EXPT	Logaritmické/exponenciální funkce †	Vykoná příslušnou funkci reálného čísla na vstupu IN.	6-13
RAD, DEG	Převod radiánů †	Vykoná příslušnou funkci reálného čísla na vstupu IN.	6-15

† Trigonometrické funkce, Logaritmické/exponenciální funkce a funkce převodu radiánů je možno použít pouze u CPU modelové řady 35x a 36x verze 9.00 nebo pozdější a u všech verzí CPU352 a CPU37x.

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.

Pravidla pro matematické funkce

Znaménko výsledku	Pro určení znaménka výsledku platí standardní matematická pravidla pro aritmetiku čísel se znaménky.
Sčítání	Instrukce ADD používá vzorec I1 + I2 = Q.
Odečítání	Instrukce SUB používá vzorec I1 – I2 = Q**.**
Násobení	Instrukce MUL používá vzorec I1 × I2 = Q.
Dělení	Instrukce DIV používá vzorec I1 ÷ I2 = Q. **Pro typy INT a DINT**. U typů INT nebo DINT instrukce DIV zaokrouhluje dolů na celočíselný podíl (zbytek se zanedbá); neprovádí zaokrouhlení na nejbližší vyšší celé číslo. Například 53 děleno 5 = 10 (zbytek 3 se zanedbá). **Pro typ REAL**. Pro typ Real instrukce DIV vytvoří výsledek s dekadickým číslem.
Dělení Modulo	Instrukce MOD může použít pouze typy INT a DINT (REAL se nepodporuje). Instrukce MOD používá vzorec I1 ÷ I2 = Q. Instrukce MOD však vytvoří pouze zbytek po operaci dělení a podíl zanedbá. Například 53 děleno 5 = 3 (podíl 10 se zanedbá).

Typy dat pro matematické funkce

Po naprogramování matematické funkce můžete zvolit typ dat. Typ dat se objeví u funkce hned pod jejím názvem (viz příklad na následujícím obrázku). V následující tabulce jsou uvedené tři typy dat, které je možno použít pro matematické funkce:

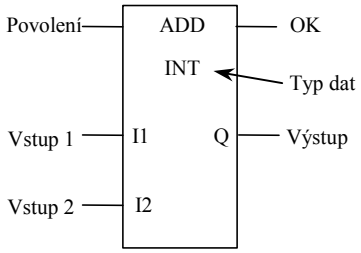
Typ dat	Popis
INT	Celé číslo se znaménkem
DINT	Celé číslo se znaménkem s dvojnásobnou délkou
REAL*	Pohyblivá desetinná tečka

* Data typu REAL je možno použít pouze u CPU řady 35x a 36x, verze firmwaru 9.00 nebo pozdější, nebo u všech verzí CPU352 a CPU37x.

Výchozí typ dat je celé číslo se znaménkem. 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 ok se nastaví do stavu ON; jinak se nastaví do stavu OFF. Pokud se budou používat celá

čísla s jednoduchou nebo dvojnásobnou délkou, znaménko výsledku bude záviset na znaménčích vstupů 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 Pouze platná adresa pro data INT; neplatí pro DINT nebo REAL.

† Když budete používat Logicmaster, pro operace s celočíselnými hodnotami se znaménkem s dvojnásobnou délkou můžete pouze zadat hodnoty v rozmezí –32 768 až +32 767. S VersaPro můžete zadávat úplné hodnoty s dvojitou přesností.

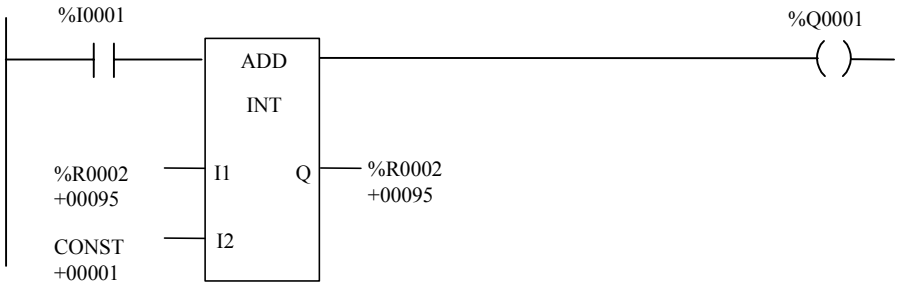
Poznámka

Pro 16-bitové operandy nebo operandy s jedním registrem je výchozí typ INT. V programu Logicmaster stisknutím **F10** změňte volbu typu na DINT, 32-bitové slovo dvojnásobné délky, nebo REAL (pouze pro CPU řady 35x, 36x a 37x). 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 9.00 nebo pozdější) a u všech verzí CPU352 a CPU37x také zabírají 32-bitový dvojitý registr se znaménkem v horním bitu, za kterým následuje exponent a mantisa.

Příklady matematických funkcí

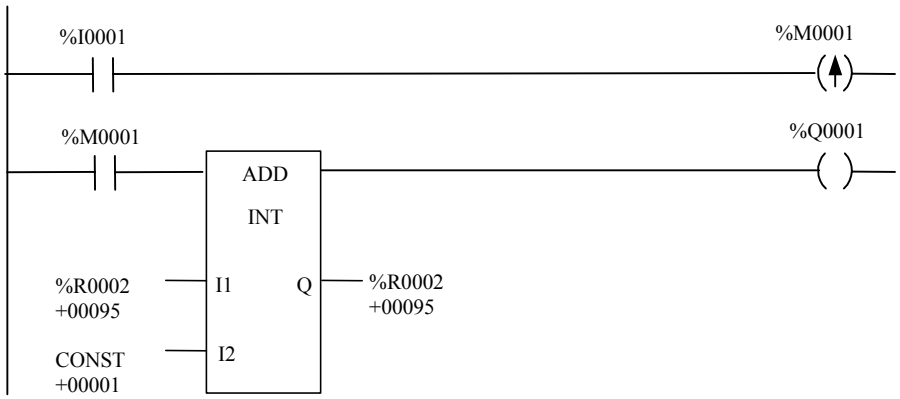
Obvod ADD s problematickou funkcí

V následujícím příkladu se provedl pokus vytvořit obvod čítače, který by čítal počet sepnutí spínače %I0001. Průběžný součet se ukládá do registru %R0002. Cílem tohoto návrhu je, aby když se sepne %I0001, instrukce ADD přidala jedničku k hodnotě v %R0002 (vstup na I2) a umístila novou hodnotu hned zpět do %R0002 (výstup na Q). Problém u tohoto návrhu je, že instrukce ADD se vykoná jednou při každém čtení PLC, když se sepne %I0001. Takže například když %I0001 zůstane sepnuté po dobu pěti čtení, výstup se inkrementuje pětkrát, i když %I0001 se během této doby sepnulo jen jednou. Aby se tento problém opravil, vstup povolení do instrukce ADD musí přicházet z přechodové ("jednorázové") cívky, jak je znázorněno na dalším obrázku.



V následujícím zlepšeném obvodu vstupní spínač %I0001 řídí přechodovou ("jednorázovou") cívku, jejíž kontakt se sepne na vstupu povolení funkce ADD pouze po jedno čtení při každém sepnutí kontaktu %I0001. Aby se kontakt %M0001 sepnul znovu, kontakt %I0001 se musí rozepnout a znovu sepnout.

Opravený návrh obvodu ADD



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, a všechny verze CPU352 a CPU37x.

Poznámka

Typy vstupních a výstupních dat pro matematické funkce musí být stejné. Funkce MUL a DIV nepodporují smíšený režim, který podporují PLC Series 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 čísla I1 16-bitovým číslem I2 a vznikne 16-bitový výsledek, zatímco DIV DINT provede dělení 32-bitového čísla I1 32-bitovým číslem I1 a vznikne 32-bitový výsledek.

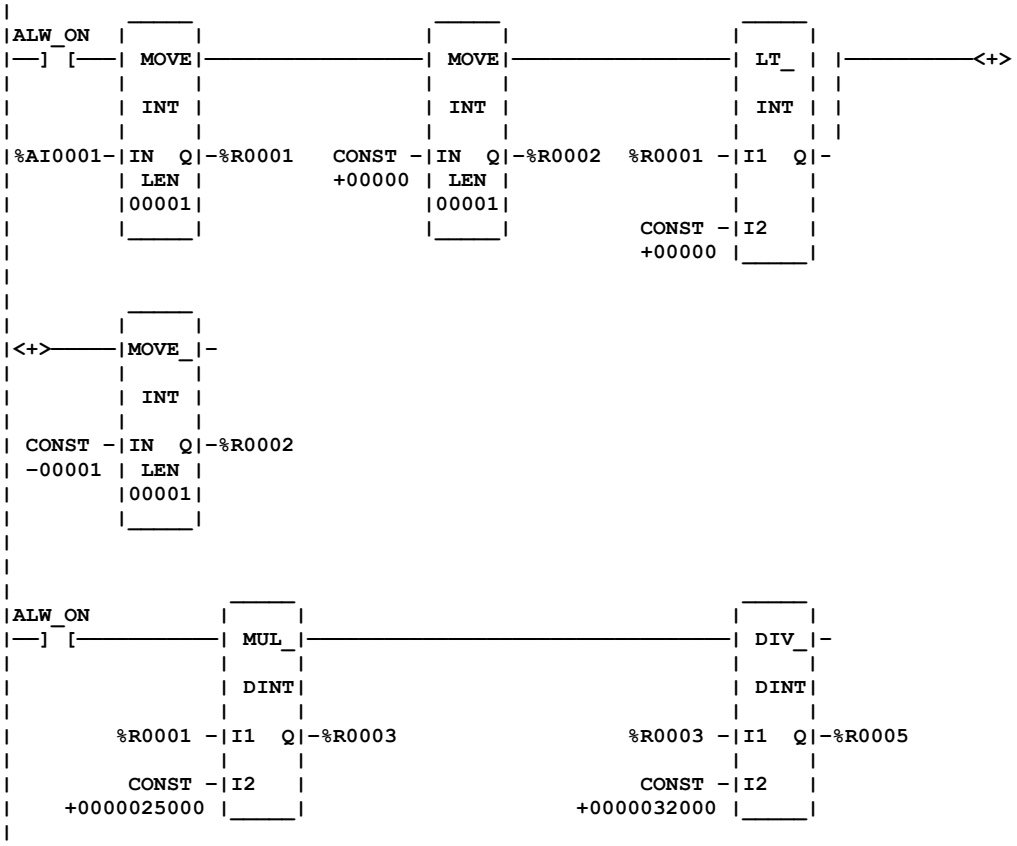
Pokud jsou povolené, tyto funkce přenesou proud, pokud nedojde k matematickému přetečení. Pokud dojde k přetečení, výsledek bude největší hodnota s příslušným znaménkem a proud se nepřenese.

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 (nejvýznamnějšího) 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 (bude indikovat kladnou hodnotu), do druhého slova zapište 0. Pokud bit s nejvyšší váhou v 16-bitovém slově INT bude -1 (bude indikovat zápornou hodnotu), 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. Vstupní analogové napětí 0 až ±10 voltů vloží hodnoty 0 až ±32 000 do odpovídajícího vstupního registru %AI. Násobením vstupního registru pomocí funkce INT MUL dojde k přetečení, protože instrukce typu INT má vstupní a výstupní rozsah 32 767 až -32 768. Použití hodnoty %AI jako vstup pro MUL DINT také nebude fungovat, protože 32-bitové I1 zkombinuje současně 2 analogové vstupy. Aby se vyřešil tento problém, můžete přemístit analogový vstup do dolního slova dvojitého registru, pak otestovat znaménko a nastavit druhé slovo na 0, pokud znaménko při testu bude kladné, nebo na -1, pokud bude záporné. Pak použijte dvojitý registr vytvořený instrukcí MUL DINT, který vytvoří 32-bitový výsledek a který je možno použít s následující funkcí DINT DIV.

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



Jiný ale méně přesný způsob naprogramování tohoto obvodu pomocí instrukce INT je, že se nejdříve umístí instrukce DIV a pak instrukce MUL. Hodnota I2 pro instrukci DIV bude 32 a hodnota I2 pro instrukci MUL bude 25. Tím se zachová proporce měřítka výše uvedeného obvodu a hodnoty zůstanou v pracovním rozsahu instrukcí typu INT. Avšak instrukce DIV vždy zanedbá jakoukoliv hodnotu zbytku, takže když se výstup DIV bude násobit instrukcí MUL, bude se násobit chyba zavedená zanedbáním zbytku. Procento chyby je v celém rozsahu vstupních hodnot nelineární a při nižších vstupních hodnotách je větší.

Naproti tomu výsledky v příkladu výše jsou přesnější, protože operace DIV se provede jako poslední, takže se nenásobí zanedbaný zbytek. Pokud se vyžaduje větší přesnost, použijte matematickou instrukci typu REAL v tomto příkladu tak, aby se zbytek nezanedbal.

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

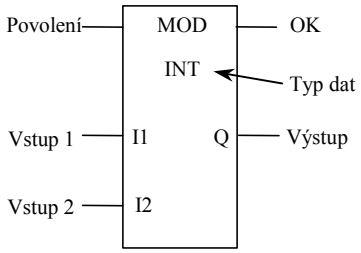
Výchozí typ dat je celé číslo se znaménkem; typ dat však je možno změnit po zvolení konkrétní datové tabulkové funkce. Více informací o typu dat najdete v kapitole 2, část 2, "Organizace programu a uživatelské adresy/data".

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, pokud existuje, který je výsledkem dělení hodnoty I1 hodnotou I2. Pokud hodnota I1 bude přesně celočíselný násobek I2, výstup Q bude nula, což znamená, že není žádný zbytek.

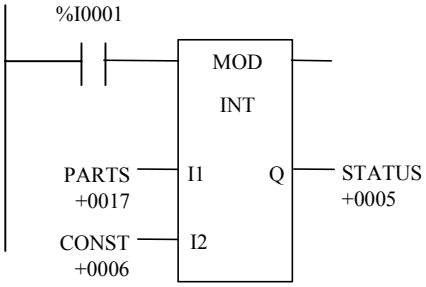
Platné typy paměti

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	•	•	•		

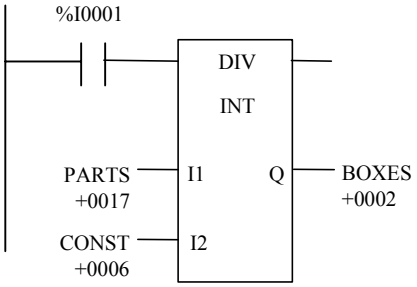
• Platná adresa nebo místo, kde funkcí může téct proud..
o Pouze platná adresa pro data INT; neplatí pro DINT.
† Pro operace s celým číslem se znaménkem s dvojnásobnou délkou jsou konstanty omezené na hodnoty v rozmezí –32 768 a +32 767.

Příklad

V následujícím příkladu se kontejnery automaticky plní díly. V jednom kontejneru může být šest dílů. Tento obvod určí stav aktuálně plněného kontejneru dělením modulo. Když bude funkce MOD povolená, provede dělení registru (PARTS) obsahujícího počet dílů šesti. Výstup (STATUS) instrukce MOD indikuje, kolik dílů (mezi 1 a 5) bylo uloženo do aktuálního kontejneru. Když bude aktuální kontejner plný, výstup Q se bude rovnat nule; pokud aktuální kontejner bude zaplněný jen částečně, výstup bude indikovat počet dílů, které jsou již v kontejneru. Hodnoty v příkladu ukazují, že se vyrobilo celkem 17 dílů a že v aktuálním kontejneru je pět dílů. (Ostatních 12 dílů je v jiných dvou kontejnerech.)



Ke stanovení počtu naplněných kontejnerů můžete použít instrukci DIV v následujícím obvodu.



Jeden možný problém u těchto obvodů je, že v registru s přezdívkou PARTS může být uložený maximální počet 32 767 kusů. Pokud budete potřebovat větší počet, bude zapotřebí další logika k (1) resetování registru PARTS před tím, než dosáhne maximum, (2) k zachycení počtu kontejnerů naplněných před resetováním registru PARTS a (3) k resetování registru PARTS, když registr STATUS bude na nule, aby počet zůstal přesný.

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

U dat typu INT a DINT se na výstup přenese pouze celočíselná část druhé odmocniny. Zlomek se zanedbá. Například druhá odmocnina ze 2 nebo 3 bude 1 a odmocnina z 5, 6, 7 nebo 8 bude 2.

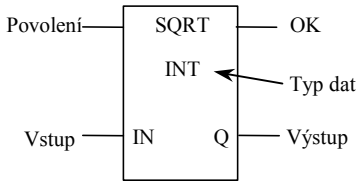
Poznámka

Data typu REAL je možno použít pouze u CPU řady 35x a 36x, verze 9.00 nebo pozdější a u všech CPU352 a CPU37x.

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

Výstup OK bude v jedničce, když se funkce vykoná bez přetečení. Pokud se objeví některá z následujících neplatných operací, OK se nastaví do nuly:

- IN < 0
- IN je NaN (nenumerický).



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. Avšak u dat typu INT a DINT se ponechá pouze celočíselná část druhé odmocniny; jakákoliv zlomková část se zanedbá.

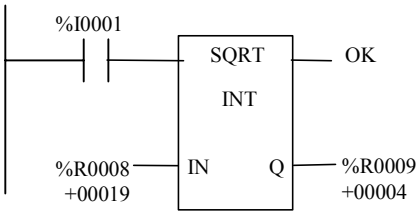
Platné typy paměti

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	•	•	•		

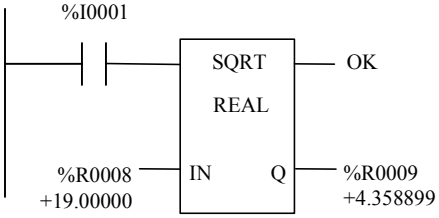
• Platná adresa nebo místo, kde funkcí může téct proud.
o Pouze platná adresa pro data INT; neplatí pro DINT a REAL.
† Pro operace s celým číslem se znaménkem s dvojnásobnou délkou jsou konstanty omezené na hodnoty v rozmezí –32 768 a +32 767.

Příklady

V následujícím příkladu se druhá odmocnina celého čísla na adrese %R0008 umístí do výsledku na adrese %R0009 vždy, když %I0001 bude ve stavu ON.



Jinou alternativou k předchozímu příkladu je, že se stejná funkce vykoná pomocí instrukce SQRT typu REAL, což povede na přesnější výsledek, jak je znázorněno na následujícím obrázku.



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 určená funkce hodnoty na vstupu IN a výsledek se uloží do výstupu Q, jehož jednotky jsou radiány. IN i Q jsou hodnoty s pohyblivou desetinnou tečkou.

Funkce SIN, COS a TAN akceptují široký rozsah vstupních hodnoty, 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:

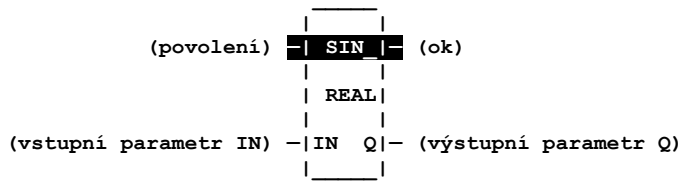
$$\mathbf{ASIN\ (IN)} = -\frac{\pi}{2} \leq Q \leq \frac{\pi}{2}$$

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 a u všech verzí CPU352 a CPU37x.

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, jinak se objeví neplatná operace a/nebo IN je NaN (nenumerická).
Q	Výstup Q obsahuje trigonometrickou hodnotu z IN.

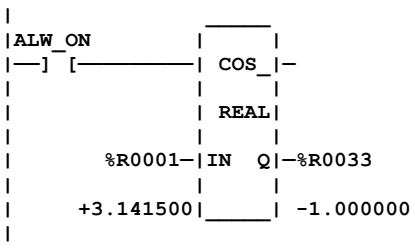
Platné typy paměti

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.

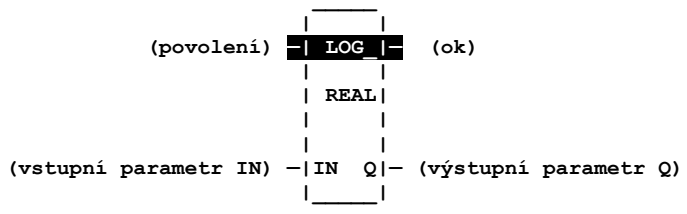


Logaritmické/exponenciální funkce (LOG, LN, EXP, EXPT)

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. (POZNÁMKA: ***e*** je konstanta používaná při logaritmických výpočtech. Její přibližná hodnota je 2,71828.)
- Funkce EXPT umocní hodnotu vstupu I1 hodnotou I2 a výsledek se 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	Pokud se neobjeví neplatná operace a/nebo IN je NaN nebo není záporné, výstup ok bude v jedničce, když se funkce vykoná bez přetečení.
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 a CPU37x.

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 funkce proud *nepropustí*, i když výstup bude 0. (Výsledkem dělení záporné hodnoty nulou bude hodnota -∞. Na obrazovce Logicmaster se objeví jako –OVERFLOW.)

Platné typy paměti

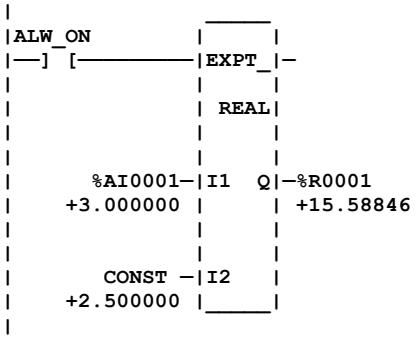
Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN*								•	•	•	•	
ok	•											•
Q								•	•	•		
I1*								•	•	•	•	
I2*								•	•	•	•	

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

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

Příklad

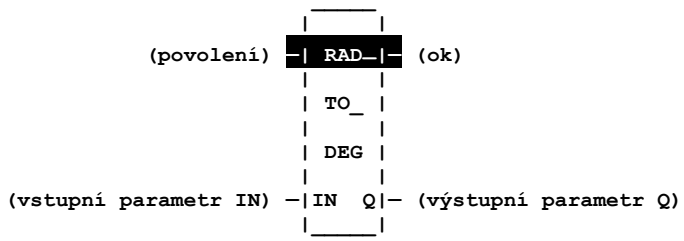
V následujícím příkladu se hodnota %AI0001, +3.000000, umocní hodnotou +2.500000 a výsledek, +15.58846, se umístí do %R0001.



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 a CPU37x.

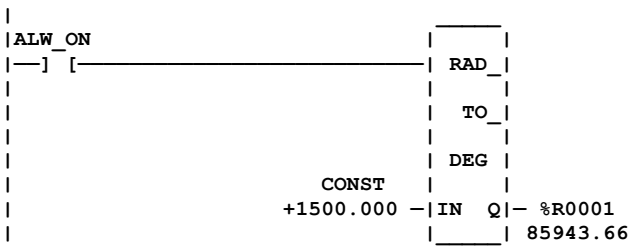
Platné typy paměti

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 než nebo Rovná se	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 než nebo Rovná se	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:

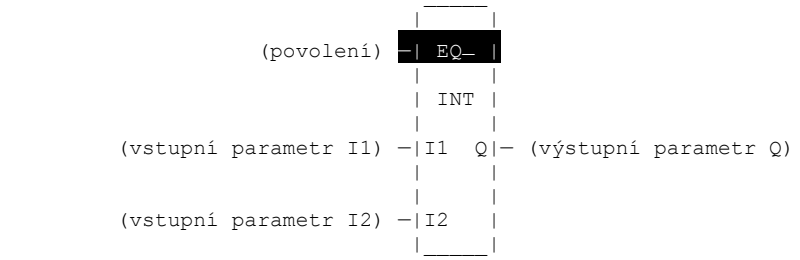
Typ dat	Popis
INT	Celé číslo se znaménkem
DINT	Celé číslo se znaménkem s dvojnásobnou délkou
REAL	Pohyblivá desetinná tečka (není k dispozici pro funkci RANGE)

Poznámka

Data typu REAL je možno použít pouze u CPU řady 35x a 36x, verze 9 nebo pozdější a u všech verzí CPU352 a CPU37x. Když se relační funkce používající data REAL vykoná úspěšně, systémový 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 než nebo rovná se	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 než nebo rovná se	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.

† Pokud budete programovat s PLC softwarem Logicmaster, pro operace s celým číslem se znaménkem s dvojnásobnou délkou budou konstanty omezené na hodnoty celých čísel (+32 767 až –32 768). Pokud budete programovat se softwarem VersaPro, jsou přípustná celá čísla se znaménkem s dvojnásobnou délkou.

Příklad

V následujícím příkladu se provede porovnání celých čísel se znaménkem s dvojnásobnou délkou %R00100/101 a %R00102/103 vždy, když kontakt povolení %I001 bude ve stavu ON. Pokud hodnota na vstupu I1 bude menší nebo rovna hodnotě na vstupu I2, cívka %Q00002 se sepne. V následujícím příkladu cívka %Q00002 je vypnutá, protože I1 je větší než I2.

RANGE (INT, DINT, WORD)

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

Poznámka

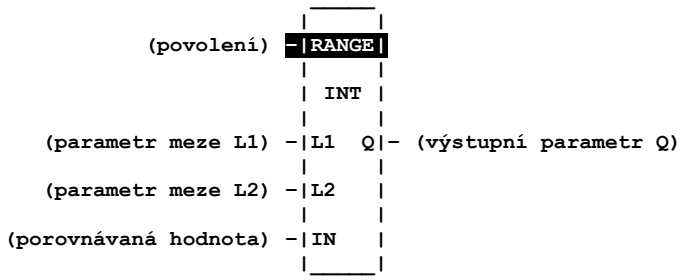
Tuto funkci je možno použít *pouze* u CPU verze 4.41 nebo pozdější.

Funkce RANGE pracuje s těmito typy dat (typ REAL není u funkce RANGE podporovaný):

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; typ dat je možno změnit po zvolení konkrétní datové tabulkové funkce. 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 pro příklad 1				
Stav povolení %I0001	**Konstantní hodnota L1**	**Konstantní hodnota 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	Jakákoliv hodnota	OFF

Příklad 2

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

Pravdivostní tabulka funkce RANGE pro příklad 2				
Stav povolení %I0001	**Hodnota L1 %R0001**	**Hodnota L2 %R0002**	**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	Jakákoliv hodnota	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 jsou omezené na práci 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.	8-57
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, které je povolené, funkce AND nebo OR provedou porovnání stavu jednotlivých bitů v bitovém řetězci I1 s odpovídajícím bitem v bitovém řetězci I2 počínaje nejméně významným bitem v každém řetězci.

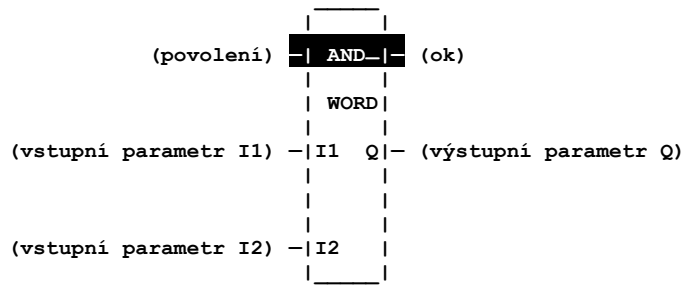
Pokud oba dva bity testované funkcí AND budou v 1, pak se do odpovídajícího místa ve výstupním řetězci Q zapíše 1. Pokud některý nebo oba tyto bity budou 0, pak se do tohoto místa ve výstupním řetězci Q zapíše 0.

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 jednoduchého funkčního bloku. Funkce je ekvivalentem dvou reléových paralelních kontaktů pro každou pozici bitu 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.

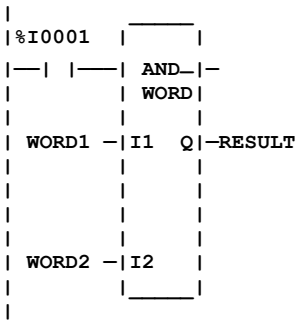
Platné typy paměti

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 (I1)	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**1**	**0**	**0**	**0**
WORD2 (I2)	**1**	**1**	**0**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**

RESULT (Q)	**0**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**

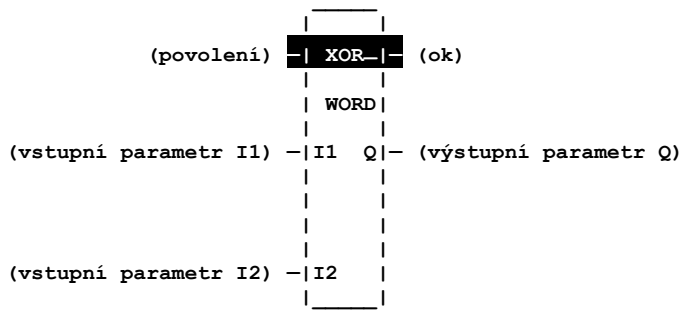
XOR (WORD)

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

Funkce XOR je vhodná pro porovnání dvou bitových řetězců nebo k překlopení skupiny bitů do jedničky a do nuly rychlostí jednoho stavu ON na dvě čtení.

Při každém čtení, kdy protéká proud, funkce provede porovnání každého bitu v řetězci I1 s odpovídajícím bitem v řetězci I2 počínaje nejméně významným bitem v každém řetězci. Pokud při porovnávání bude pouze jediný bit v logické 1, pak se do odpovídajícího bitu řetězce Q umístí 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 přepínáním povolení vstupu do funkce dvojnásobkem požadované rychlosti blikání; u této aplikace vstup povolení musí přejít do jedničky na dobu jednoho čtení (použijte kontakt cívky jednorázového typu nebo obvod se samočinným nulováním časovače).



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 pro 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

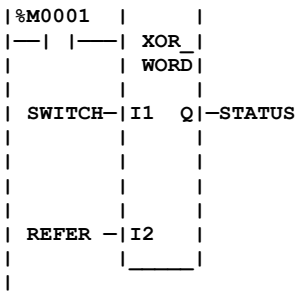
Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
I1		•	•	•	•	•	•	•	•	•	•	
I2		•	•	•	•	•	•	•	•	•	•	
ok	•											•
Q		•	•	•	•	•†	•	•	•	•		

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

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

Příklad alarmového obvodu s použitím funkce XOR

Kdykoliv v následujícím příkladu bude kontakt povolení %M0001 sepnutý, 16-bitový řetězec s názvem SWITCH se porovná s referenčním bitovým řetězcem s názvem REFER. Bitový řetězec SWITCH je skupina bitů, které představují stav jedničky/nuly kontaktů alarmového spínače. Bitový řetězec REFER představuje normální nebo nealarmový stav těchto bitů. Pokud se stav některého bitu v řetězci SWITCH bude lišit od svého odpovídajícího bitu v řetězci REFER, jejich odpovídající výstup Q přejde do logické 1. Za normálního (nealarmového) stavu hodnota slova s názvem STATUS bude nula.



Pozice bitu	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
I1 (SWITCH)	0	1	0	1	1	1	1	0	1	1	0	0	0	0	0	0
I2 (REFER)	0	0	0	1	1	1	1	1	1	1	0	0	1	0	0	0
Q (STATUS)	0	1	0	0	0	0	0	1	0	0	0	0	1	0	0	0

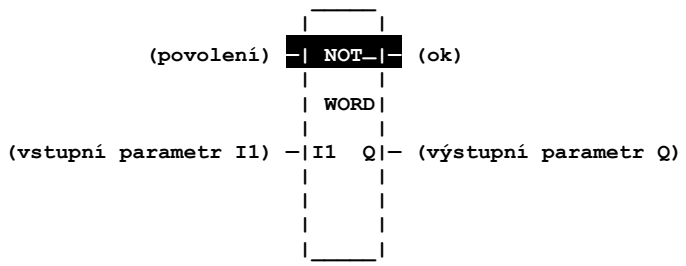
Data na výstupu STATUS je možno použít jako vstup do funkce Nerovná se (NE), která provede porovnání slova s názvem STATUS s konstantou nula. Pokud se STATUS nebude rovnat nule, NE nastaví svůj výstup do jedničky a bude indikovat přítomnost alarmu.

Bity na výstupu STATUS, které se nerovnají logické 1, je možno identifikovat pomocí funkce BPOS (Pozice bitu), která vyhledá bity ve STATUS a nahlásí pozici (číslo mezi 1 a 16) prvního bitu (počínaje bitem 1), u kterého zjistí, že je v logické 1. Ve výše uvedeném příkladu funkce BPOS předá na výstup číslo 4 jako indikaci, že čtvrtý bit je v logické 1. Chcete-li testovat více než jeden bit, můžete uložit záznam bitu 4, pomocí funkce BCLR (Smazat bit) smazat bit 4 a pak zopakovat test BPOS a najít další bit, který se rovná logické 1 (bit 9 v příkladu výše). Tento proces je možno opakovat, dokud budete nacházet další nenulové bity. Všimněte si, že funkce BCLR a BPOS jsou detailně popsané na jiném místě v této kapitole.

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.

Funkce NOT vykoná a propustí proud při každém čtení, při kterém vstup povolení bude v jedničce.



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.

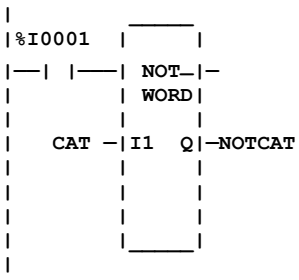
Platné typy paměti

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

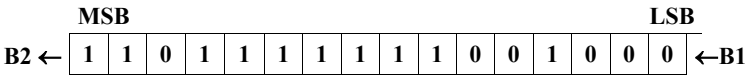
V následujícím příkladu se při každém nastavení vstupu %I0001 bitový řetězec s názvem NOTCAT nastaví na převrácený stav bitového řetězce CAT, jak je znázorněno v pravdivostní tabulce níže.



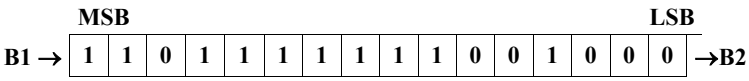
CAT	1	1	0	1	0	1	0	0	1	1	0	1	0	0	0	1
NOTCAT	0	0	1	0	1	0	1	1	0	0	1	0	1	1	1	0

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.

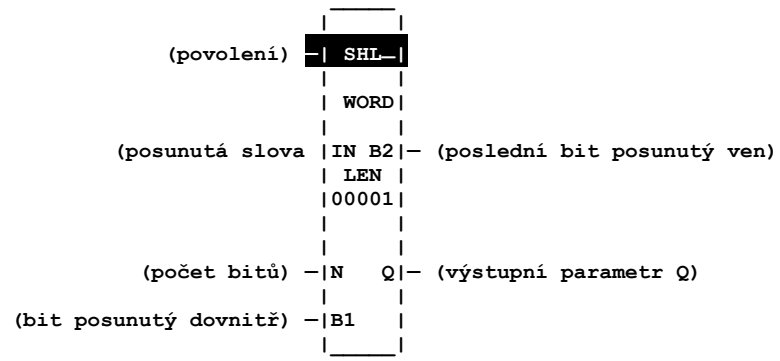
Pokud počet bitů (N), o které se má provést posunutí, bude větší než počet bitů pole (LEN) * 16 nebo pokud počet bitů, o které se má provést posunutí, bude nula, pak se pole (Q) zaplní kopiemi vstupního bitu (B1) a vstupní bit se zkopíruje do výstupu (B2). Pokud počet bitů posunutí bude nula, pak se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

Bity, které se posouvají dovnitř na začátku řetězce, se definují pomocí vstupního parametru B1, který vyžaduje kontakt s napájecí spojnicí. Pokud jako počet posouvaných bitů bude zadaná délka větší než 1, každý bit se zaplní stejnou hodnotou (0 nebo 1) B1. Vstup B1 je možno řídit pomocí

- kontaktu ALW_ON (%S07), který drží B1 trvale na logické 1.
- kontaktu ALW_OFF (%S06), který drží B1 trvale na logické 0.
- kontaktu od interní cívky, například %M nebo %Q, který umožňuje měnit hodnotu.
- kontaktu %I, který umožňuje měnit hodnotu ze vstupního kontaktu.

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 Q musí používat stejné paměťové místo jako vstupní parametr IN. Instrukce SHL/SHR se vykonají každé čtení, u kterého bude vstup povolení v jedničce. Výstup B2 obsahuje hodnotu posledního bitu vysunutého ven; pokud se například posunuly čtyři bity, B2 bude obsahovat hodnotu (1 nebo 0) čtvrtého bitu vysunutého ven.



Parametry

Parametr	Popis
povolení	Když logika povolení bude v 1, provede se posunutí.
IN	IN obsahuje adresu prvního posouvaného slova.
N	N obsahuje počet míst (pozic bitů), o které se pole má posunout.
B1	B1 obsahuje hodnotu bitu (0 nebo 1), který se má posunout do pole.
B2	B2 obsahuje hodnotu posledního bitu (0 nebo 1), který se má z pole posunout ven.
Q	Výstup Q obsahuje první slovo posunutého pole.
LEN	LEN je počet slov (1 - 256) v poli, které se má posunout.

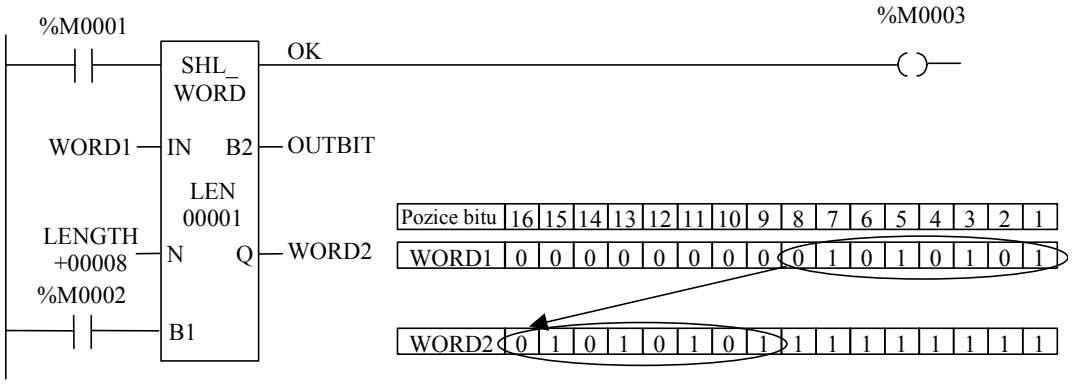
Platné typy paměti

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

Když v následujícím příkladu bude vstup %M0001 v jedničce, SHL vytvoří kopii bitového řetězce v IN (s názvem WORD1). Pak v této kopii posune všechny bity doleva o 8 bitových pozic (určených hodnotou v N). Bity od pozic 9 - 16 se vysunou ven (vyřadí se) a bity, které byly na pozici 1 – 8, nyní zaujmou pozice 9 – 16. Bity na pozici 1 – 8, které se "uprázdnily" při vysunutí bitů 1 – 8, se zaplní jedničkami, protože v tomto příkladu je kontakt %M0002 sepnutý a nastavuje vstup B1 na logickou 1. Nakonec se posunuté/naplněné slovo zapíše do adresy na výstup Q (s názvem WORD2). Původní WORD1 na IN se nezmění. Výstup B2 se rovná nule, protože poslední bit vysunutý ven byla logická nula (bit, který byl na pozici 9) a cívka %M0003 bude sepnutá, protože funkce pracovala správně a proto na výstupu OK bude propouštět proud.



ROL a ROR (WORD)

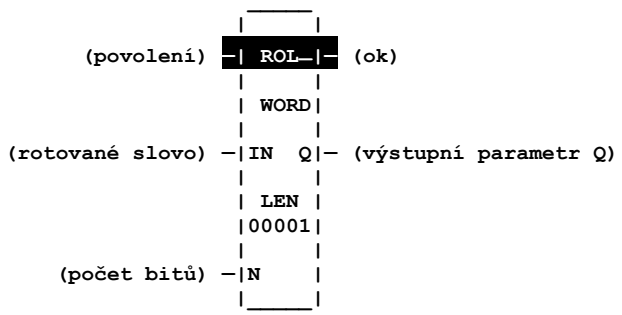
Funkce rotace doleva (ROL) provede rotaci všech bitů v řetězci o zadaný počet míst doleva. Když se vyskytne rotace, provede se rotace vstupního řetězce o zadaný počet bitů doleva a zpět do řetězce zprava.

Funkce posunutí doprava (SHR) provede posunutí všech bitů slova doprava o zadaný počet míst. Když se vyskytne rotace, provede se rotace zadaného počtu bitů ven ze vstupního řetězce doprava a zpět do řetězce zleva.

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

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

Výsledek rotace se zapíše do výstupního řetězce Q. Pokud chcete provést rotaci vstupního řetězce, výstupní parametr Q2 musí používat stejné paměťové místo jako vstupní parametr IN. Funkce rotace se vykoná při každém čtení, při kterém logika povolení bude v jedničce.



Parametry

Parametr	Popis
povolení	Když vstup povolení bude v jedničce, provede se rotace.
IN	IN obsahuje první rotované slovo.
N	N obsahuje počet míst (pozic bitů), o kolik se má provést rotace pole.
ok	Výstup ok bude v jedničce, když funkce rotace bude povolená a délka rotace (na N) bude větší než nula, ale nebude větší než je velikost pole.
Q	Výstup Q obsahuje první slovo rotovaného pole.
LEN	LEN je počet slov v poli (1 - 256), které se má rotovat.

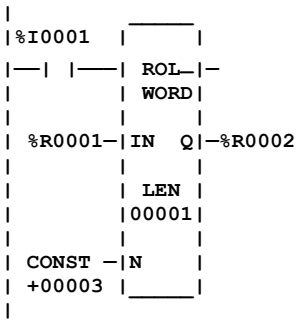
Platné typy paměti

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

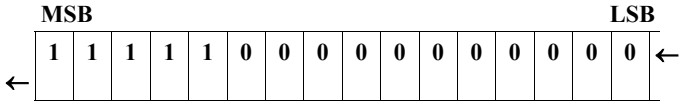
• Platná adresa nebo místo, kde funkcí může téct proud..
† Pouze %SA, %SB nebo %SC; %S nelze použít.

Příklad

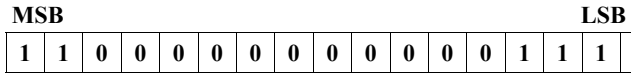
V následujícím příkladu funkce ROL při každém povolení vstupu %I0001 vytvoří kopii vstupního řetězce v IN. Pak v kopii provede rotaci vstupního bitového řetězce o 3 bity (zadané hodnotou na vstupu N) a výsledek umístí do %R0002. Po vykonání této funkce vstupní bitový řetězec %R0001 zůstane nezměněný. Pokud však budete chtít provést rotaci vstupního řetězce, pro IN a Q použijte stejnou adresu.



%R0001:



%R0002 (po provedení rotace):



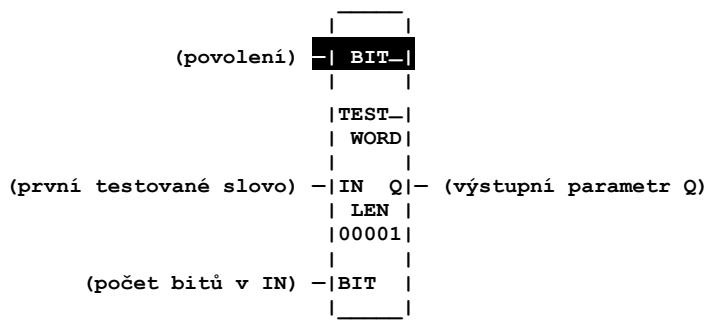
BTST (WORD)

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 zadaný následujícím vztahem, pak se Q nastaví do stavu OFF.

Vztah: 1 ≤ BIT ≤ (16 * LEN)

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.

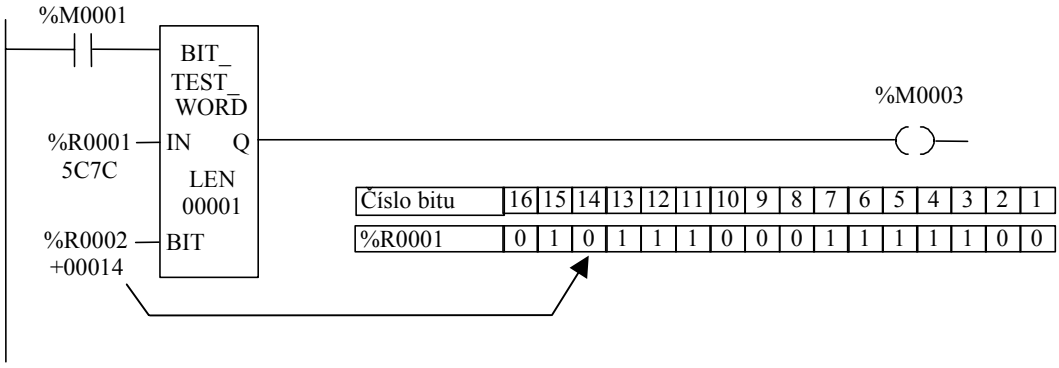
Platné typy paměti

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

Když v následujícím příkladu bude vstup povolení %M0001 v jedničce, provede se test bitu 14 ve slově %R0001 (bit 14 je zadaný hodnotou v %R0002). Protože bit 14 v hodnotě uvedené pro %R0001 (5C7C) je nula, výstup Q se nenastaví do jedničky. Všimněte si, že tato funkce může být pouze typu WORD; proto paměťové adresy používané na IN se objeví na obrazovce Logicmasteru v hexadecimálním formátu. Avšak hodnota na BIT se objeví v celočíselném formátu bez ohledu na to, jestli se používá konstanta nebo paměťová adresa.

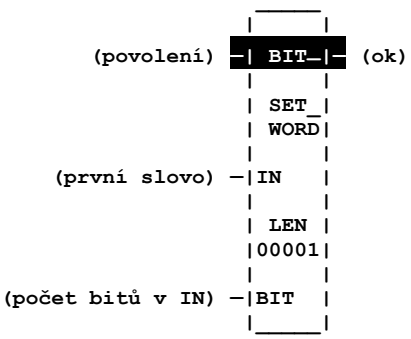


BSET a BCLR (WORD)

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 funkcí 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. Pokud například LEN bude nastaveno na 1, pak délka testovaného bitového řetězce bude 16. Pokud v tomto případě číslo na BIT bylo 17 nebo větší, bude mimo rozsah, takže výstup ok se nenastaví do jedničky.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, bitová operace se provede.
IN	IN obsahuje adresu prvního slova bitového řetězce použitého 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	Pokud reálná hodnota na vstupu BIT nebude mimo platný rozsah, výstup ok bude v jedničce vždy, když vstup povolení bude v jedničce.
LEN	LEN je počet slov v bitovém řetězci, jehož počáteční adresa je nakonfigurovaná na IN.

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

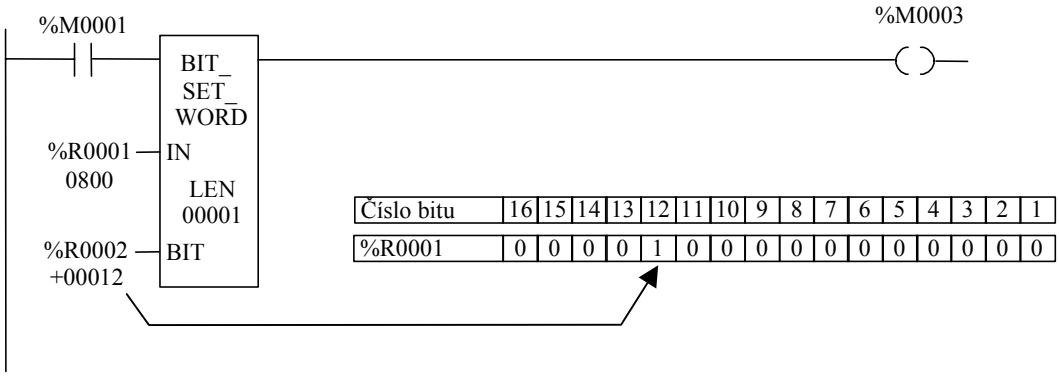
Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN		•	•	•	•	†	•	•	•	•		
BIT		•	•	•	•		•	•	•	•	•	
ok	•											•

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

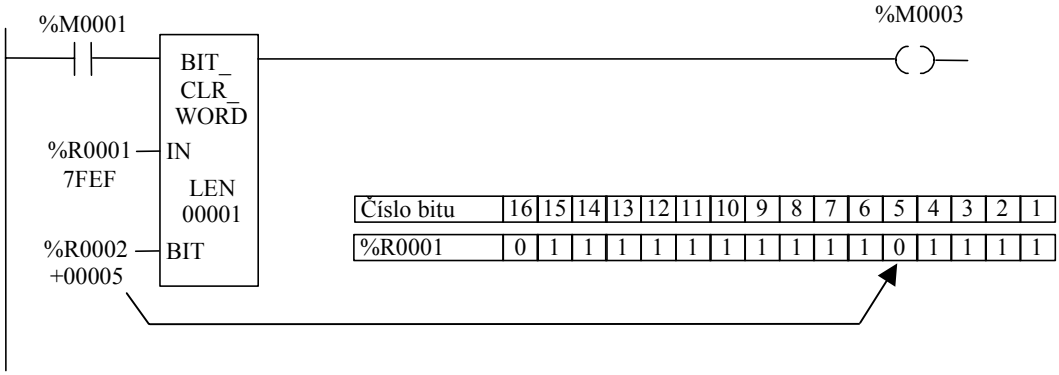
Příklady

Všimněte si, že funkce Nastavit bit a Smazat bit mohou být pouze typu WORD; proto paměťové adresy používané na IN se objeví na obrazovce Logicmasteru v hexadecimálním formátu. Avšak když se používá konstanta nebo paměťová adresa, hodnota na BIT se objeví v celočíselném formátu.

Když v následujícím příkladu vstup %M0001 bude v jedničce, bit 12 (zadaný vstupem BIT) řetězce začínajícího na adrese %R0001 (adresa na vstupu IN) bude nastavený na 1.



Když v následujícím příkladu vstup %M0001 bude v jedničce, bit 5 (hodnota vstupu BIT) řetězce začínajícího na adrese %R0001 (adresa na vstupu IN) se nastaví na 0 (vynuluje se).



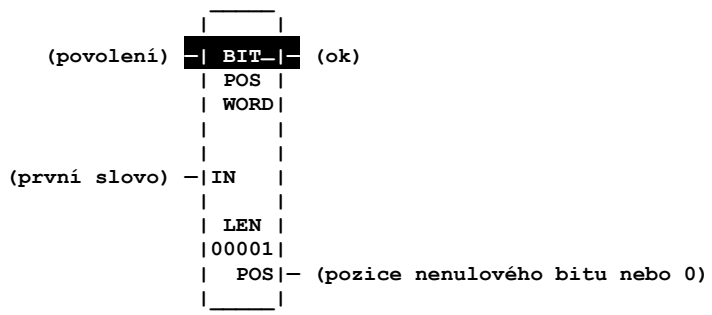
BPOS (WORD)

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

Při každém cyklu, kdy funkce bude povolená, se provede čtení bitového řetězce začínajícího na 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 bitového řetězce použitého 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

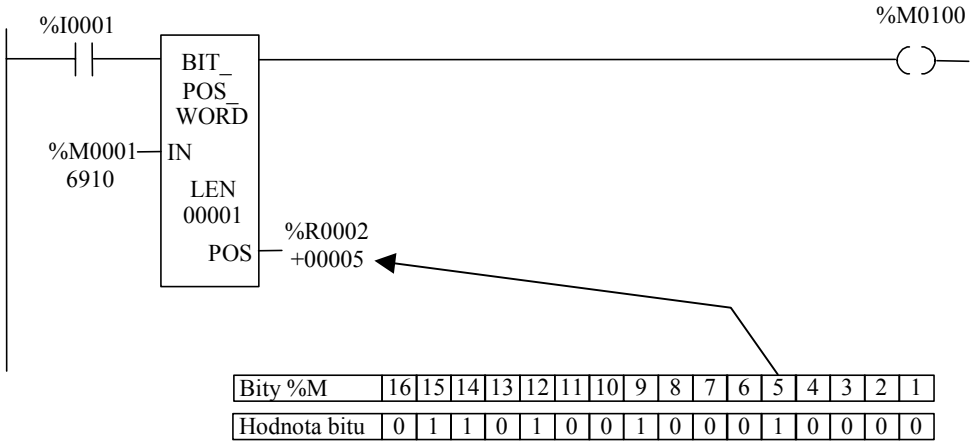
Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN		•	•	•	•	•	•	•	•	•		
POS		•	•	•	•		•	•	•	•		
ok	•											•

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

Příklad

Všimněte si, že funkce Pozice bitu může být pouze typu WORD; proto paměťové adresy používané na IN se objeví na obrazovce Logicmasteru v hexadecimálním formátu. Hodnota na POS se však objeví v celočíselném formátu. Logicmaster zobrazí prvních 16 bitů na IN v hexadecimálním formátu.

V následujícím příkladu se při nastaveném vstupu %I0001 bude prohledávat bitový řetězec začínající na %M0001, dokud se nenalezne bit rovnající se 1 nebo dokud se neprohledá celý řetězec. Cívka %M0100 se zapne. Pokud se najde bit rovnající se 1, jeho pozice v bitovém řetězci se zapíše do %R0002; jinak se do %R0002 zapíše hodnota 0. V ukázaném příkladu bit 5 je první logická 1 nalezená hledáním (které začíná bitem 1), takže hodnota zapsaná do %R0002 je 5.



Bity %M	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
Hodnota bitu	0	1	1	0	1	0	0	1	0	0	0	1	0	0	0	0

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 MSKCMP typu jednoduchého slova nebo 32-bitových slov pro funkci MSKCMP typu dvojitého slova).

Když jeho povolení vstupu bude v jedničce, funkce provede porovnání bitů v prvním řetězci s odpovídajícími bity v druhém řetězci. Porovnávání bude pokračovat, dokud se nezjistí neshoda nebo dokud se nedosáhne konce řetězce. Funkce vykoná každé čtení, u kterého vstup povolení bude v jedničce, takže pro mnoho aplikací se pro povolení vstupu použije "jednorázový" kontakt.

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

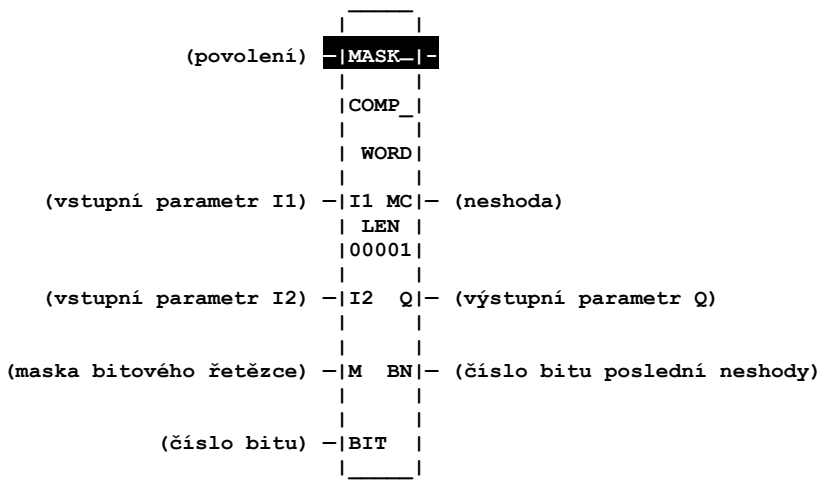
Pokud všechny odpovídající bity v řetězcích I1 a I2 budou souhlasit, funkce nastaví výstup MC “neshoda” na 0 a BN na nejvyšší číslo bitu ve vstupním řetězci. Porovnávání se pak zastaví. Při dalším vyvolání MSKCMP se BN resetuje na 0.

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*, neshoda se bude ignorovat a 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	Přejde do logické jedničky na jedno čtení, pokud se objeví neshoda. Pokud se požaduje "zachycení" výstupu po jednom čtení, na tomto výstupu je možno použít nastavovací cívku.
Q	Výstupní kopie bitového řetězce masky (M).
BN	Číslo bitu, kde se vyskytla poslední neshoda.
LEN	LEN je počet slov v bitovém řetězci.

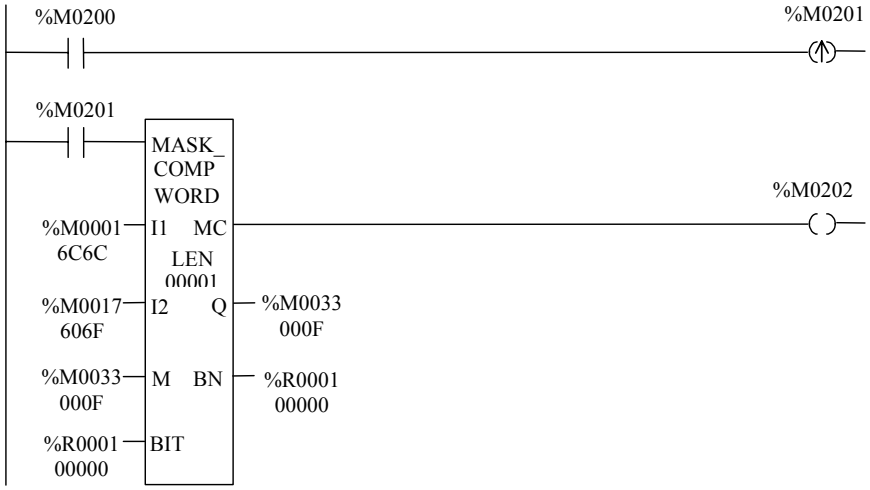
Platné typy paměti

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 1 – Instrukce MSKCMP

Když dojde k sepnutí %M0200, kontakt přechodové cívky %M0201 se sepne na jedno čtení, což umožní, aby se funkce MSKCMP vykonala jednou. %M0001 až %M0016 (I1) se porovnají s %M0017 až %M0032 (I2). %M0033 až %M0048 (M) obsahují hodnotu masky. Hodnota v %R0001 (BIT) určuje, na které pozici bitu (0) má porovnávání v těchto dvou vstupních řetězcích I1 a I2 začít.



Stav před prvním vykonání MSKCMP

Obsah vstupních adres před tím, než se vykoná MSKCMP, je následující:

%M bity	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
Vstup 1(I1)	0	1	1	0	1	1	0	0	0	1	1	0	1	1	0	0

%M bity	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
Vstup 2(I2)	0	1	1	0	1	1	0	1	0	1	1	0	1	1	1	1

%M bity	48	47	46	45	44	43	42	41	40	39	38	37	36	35	35	33
Maska (M/Q)	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

BIT/BN (%R0001) = 0
MC (%M0202) = OFF

Stav po prvním vykonání MSKCMP

Následující tabulka ukazuje obsah adres Maska (M/Q) po jednom vykonání MSKCMP. (I1 a I2 jsou stále na výše uvedených hodnotách.) Protože devátý bit vytvořil neshodu, devátý bit (%M0041) v řetězci Maska bude nastavený na logickou 1, BIT/BN bude obsahovat hodnotu 9 a výstup MC na jedno čtení přejde do jedničky. I když bity na první a druhé pozici nejsou stejné, nevytvoří neshodu, protože bity masky pro tyto pozice jsou v 1.

%M bity	48	47	46	45	44	43	42	41	40	39	38	37	36	35	35	33
Maska (M/Q)	0	0	0	0	0	0	0	1	0	0	0	0	1	1	1	1

BIT/BN (%R0001) = 9
MC (%M0202) = ON (na jedno čtení)

Příklad 2 – Detekce chyby pomocí funkce maskovaného porovnání

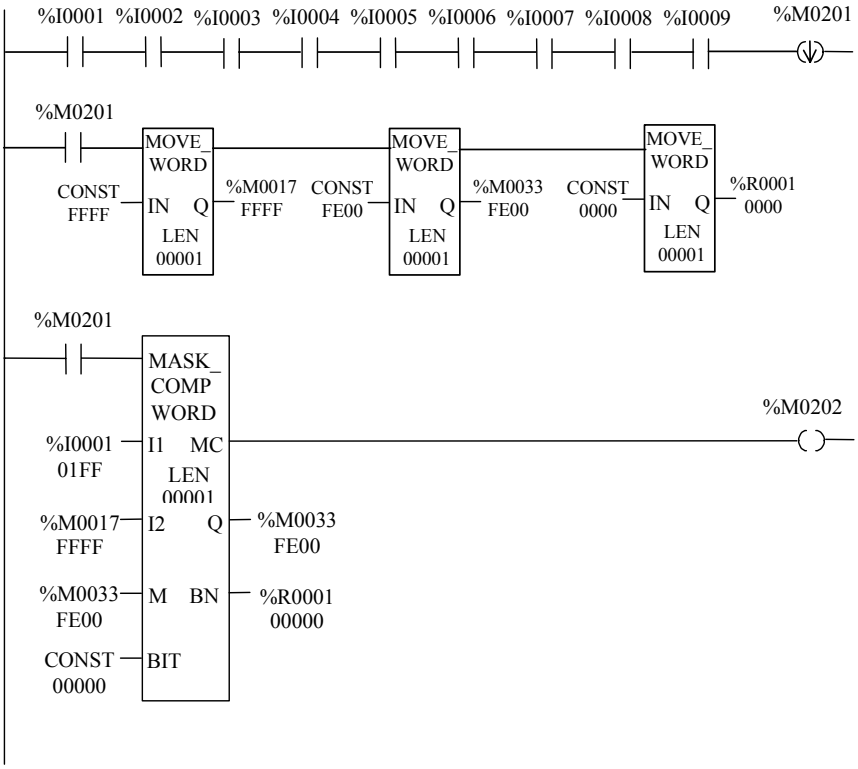
Náhodné chyby je obtížné zachytit. Příkladem může být, když několik spínačů bude sériově zapojeno v obvodu, který napájí chybové relé. Za normálních podmínek budou všechny spínače sepnuté a chybové relé se bude napájet ("bezchybné" uspořádání). Když se objeví chyba, jeden kontakt se rozpojí a chybové relé odpadne. Pokud vadný kontakt zůstane rozpojený, je snadné určit, který spínač chybu způsobil. Někdy se však kontakt rozpojí na krátký okamžik, možná na méně než jednu sekundu, a pak se zase sepne. To způsobí, že chybové relé krátce odpadne a uzavře proces. Protože se kontakt zase sepne, všechno se zdá normální.

Aby bylo možno řešit takový problém, následující obvod funguje jako "lapač chyby" tak, že zjistí, který kontakt se rozpojil, a uloží jeho číslo do registru. Kontakty od vstupních spínačů na první příčce, které jsou zapojené k bodu vstupního modulu (%I1 – %I9), jsou naprogramované sériově tak, že napájejí %M0021, negativní přechodovou cívku.

Druhá příčka inicializuje MSKCMP tak, aby byla připravená zachytit chybu. První instrukce Přesunutí zapíše do vstupu I2 funkce MSKCMP samé jedničky. Druhá instrukce Přesunutí zapíše hodnoty 1 do bitů 10-16 slova masky (takže tyto bity se ignorují), protože pro spínače %I0001-%I0009 je zapotřebí pouze prvních devět bitů porovnávaných slov (MSKCMP používá celá slova). Třetí instrukce Přesunutí vynuluje výstupní registr %R0001 tak, aby byl připravený hlásit nejnovější chybu.

Během normální činnosti bude prvních devět bitů na vstupu I1 funkce MSKCMP v logické 1, protože všechny přepínače jsou sepnuté. Na vstup I2 se zapíšou samé logické 1, protože to je normální stav, se kterým se vstupní spínače porovnávají. Maska má 1 v bitech 10-16, protože tyto bity se nepoužívají, neboť je pouze devět vstupních spínačů. Když se spínač rozepne, kontakt %M0201 se sepne na jedno čtení. To bude mít za následek, že se na druhé příčce objeví inicializační přesunutí a na třetí příčce se povolí MSKCMP. MSKCMP provede porovnání vstupních spínačů s logickými 1 na vstupu I2, identifikuje, který spínač je v logické 0 (rozpojený) a zapíše číslo bitu rozpojeného spínače na výstup BN (%R0001). Bity jsou číslované od 1-9 počínaje %I1. Pokud se například měl rozpojit %I4, %R0001 bude obsahovat číslo 4.

Všimněte si, že pokud se v tomto obvodu spínač rozepne a zase sepne, cívka %M0201 odpadne a znovu sepne, ale číslo spínače, který se rozpojil, bude uloženo v %R0001. Pokud se však spínač například rozepne znovu, obsluha stroje stiskne tlačítko nouzového zastavení nebo otevře bezpečnostní zábranu, maskované porovnání se aktivuje znovu a zapíše číslo posledního rozepnutí spínače do %R0001. To znamená, že po výskytu chyby je nutno se zařízení nedotýkat, dokud nebude možno hodnotu v %R0001 zkontrolovat. Pokud by to nebylo praktické, je možno použít další instrukci Přesunutí.


%I0001 %I0002 %I0003 %I0004 %I0005 %I0006 %I0007 %I0008 %I0009
%M0201
%M0201
MOVE_ WORD
CONST FFFF
IN Q
LEN 00001
%M0017 FFFF
MOVE_ WORD
CONST FE00
IN Q
LEN 00001
%M0033 FE00
MOVE_ WORD
CONST 0000
IN Q
LEN 00001
%R0001 0000
%M0201
MASK_ COMP WORD
%I0001 01FF
I1 MC
LEN 00001
%M0017 FFFF
I2 Q
%M0033 FE00
%M0033 FE00
M BN
%R0001 00000
CONST 00000
BIT
%M0202

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	Move	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 bitů (%I, %Q, %M, %G nebo %T) nebo paměti slov (%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.	9-15

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 funkce bude povolena, provede kopírování dat ze vstupního parametru IN do výstupního parametru Q po bitech. 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. Pokud se na adresách vstupu a výstupu nebude vyskytovat překrytí, data ve vstupním parametru se nezmění.

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 mít hodnotu 4 a délka (LEN) bude 1, 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 se bude rovnat 4, pak se do paměťového místa zadaného v Q a také do třech následujících míst zapíše 9.

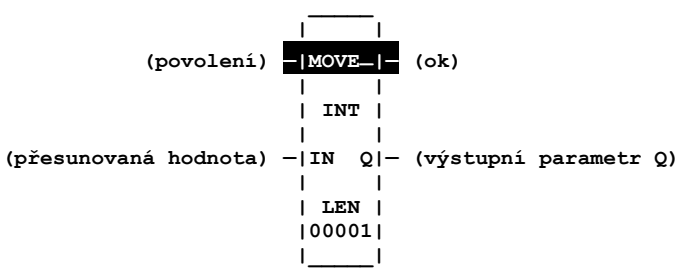
Operand LEN udává počet:

- Slov, které se mají přesunout pomocí MOVE_INT a MOVE_WORD.
- Bitů, které se mají přesunout pomocí MOVE_BIT.
- Reálná čísla, 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 a všech verzí CPU352 a 37x.

Funkce propustí proud doprava vždy, když přijde proud.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, provede se přesunutí.
IN	IN obsahuje kopírovanou (přesouvanou) hodnotu. V případě MOVE_BIT je možno použít libovolnou diskrétní adresu; nemusí být spojená s bajtem. Avšak 16-bitová hodnota začínající zadanou adresou se zobrazí na obrazovce Logicmasteru.
ok	Výstup ok bude v jedničce vždy, když funkce bude povolená.
Q	Když se provede přesunutí, do Q se zkopíruje hodnota z IN. V případě MOVE_BIT je možno použít libovolnou diskrétní adresu; nemusí být spojená s bajtem. Avšak 16-bitová hodnota začínající zadanou adresou se zobrazí na obrazovce Logicmasteru.
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, 36x a 37x funkce MOVE_INT a MOVE_WORD nepodporují překrývání parametrů IN a Q, kde by adresa IN byla 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 9.00 a pozdější) a všechny verze CPU352 a 37x mají funkci pohyblivé desetinné tečky a proto jsou jedinými CPU Series 90-30, které mohou vykonat MOVE_REAL.

Platné typy paměti

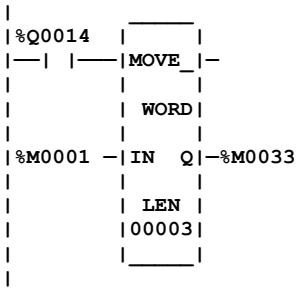
Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN		•	•	•	•	o	•	•	•	•	•	
ok	•											•
Q		•	•	•	•	o†	•	•	•	•		

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 – Překrývání adres (pouze pro CPU 311-341)

Když kontakt povolení vstupu %Q0014 bude ve stavu ON, z paměťového místa %M0001 se přemístí 48 bitů do paměťového místa %M0033. I když cílové umístění překrývá zdrojové umístění o 16 bitů, přemístění se provede správně (s výjimkou CPU 35x a 36x, jak bylo uvedeno dříve).



Před použitím funkce Přesunutí:

INPUT (%M0001 až %M0048)

															1
%M0016	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0
%M0032	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
%M0048	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Po použití funkce Přesunutí:

OUTPUT (%M0033 až %M0080)

															33
%M0048	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0
%M0064	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
%M0080	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Příklad 2 – pro všechna CPU

V tomto příkladu se při každém nastavení %I0003 hodnoty tří bitů %M0001, %M0002 a %M0003 přesunou do %M0100, %M0101 a %M0102 a cívka %Q0001 se sepne.

BLKMOV (INT, WORD, REAL)

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

Poznámka

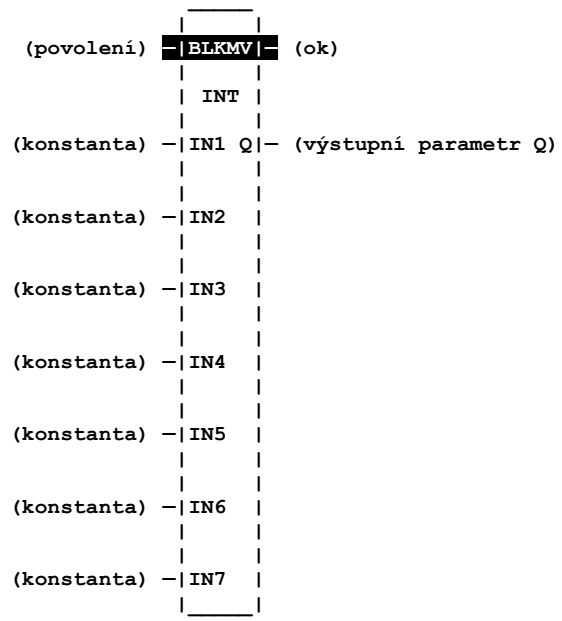
Data typu REAL je možno použít pouze u CPU řady 35x a 36x, verze 9 nebo pozdější, a všech verzí CPU352 a 37x.

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 je povolená.



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.

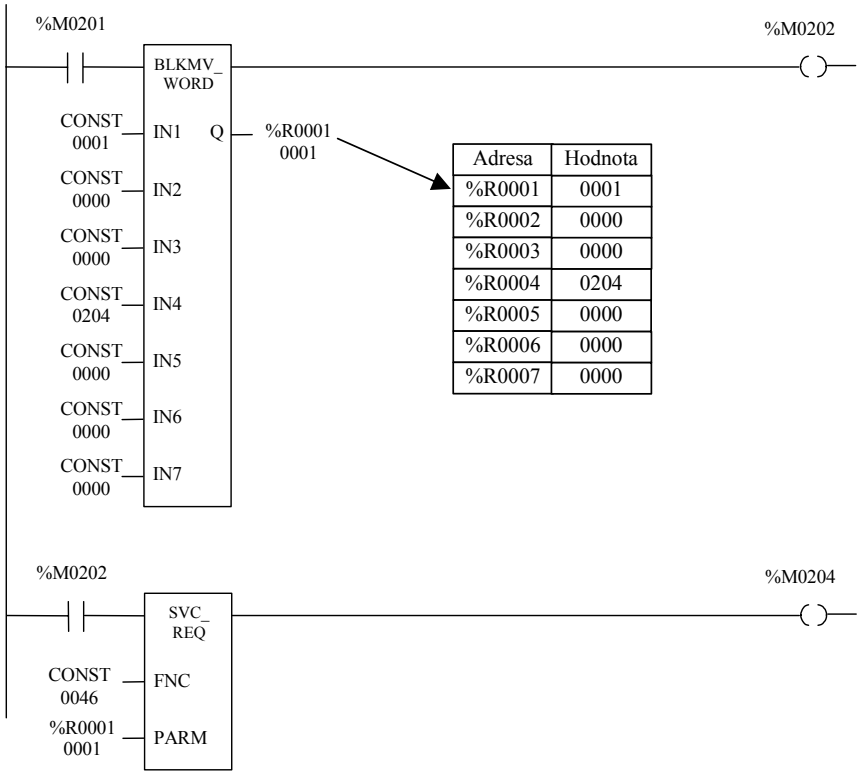
† Pouze %SA, %SB %SC; %S nelze použít.

Poznámka

Funkce pohyblivé desetinné tečky podporují pouze CPU řady 35x a 36x, verze 9 nebo pozdější a všechny verze CPU352 a CPU 37x. Tyto CPU 90-30 jsou jediné, které jsou schopné vykonat BLKMV_REAL.

Příklad

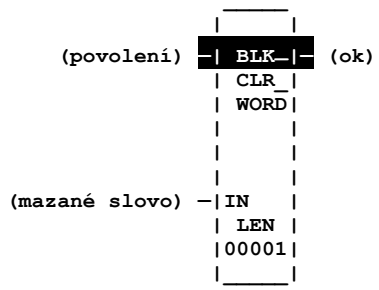
V následujícím příkladu, když kontakt pro povolení vstupu %M0201 bude ve stavu ON, funkce BLKMOV provede zkopírování sedmi vstupních konstant do paměťových míst %R0001 až %R0007 (zadaných na výstupu Q). Pokud se BLKMV vykoná úspěšně, výstup OK se nastaví do jedničky, což nabudí %M0202. Tím kontakt %M0202 povolí funkci Service Request na následující příčce, která jako blok parametrů používá %R0001 až %R0007. (Více informací o instrukci Service Request najdete v kapitole 12.)



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.

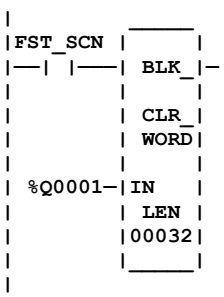
Platné typy paměti

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 a %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

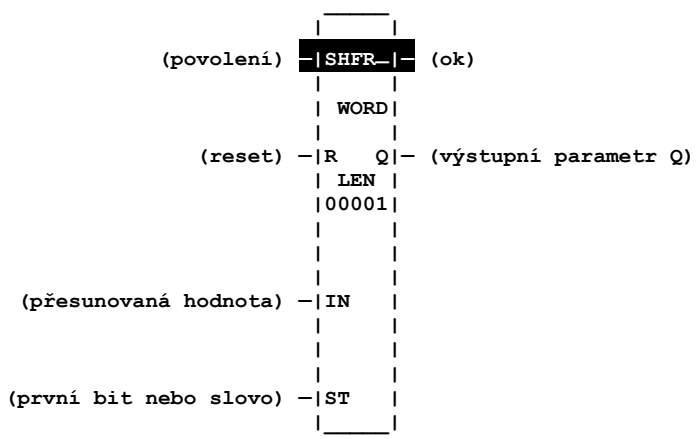
Při přiřazování adres mohou překrývající se rozsahy vstupních a výstupních adres u víceslovních funkcí vytvářet neočekávané výsledky.

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. Pokud Q bude mít jedinečnou adresu, data vysunutá ven z Q se ztratí. Pokud však IN a Q budou míst stejnou adresu, data se v posuvném registru zacyklí. Nejvyšší adresa členu registru posunutí v IN se posune do uvolněného členu začínajícího na ST. Obsah registru posunutí je přístupný přes program logiky, protože se celý nachází v adresovatelné paměti.

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

Funkce se vykoná jednou při každém čtení, když je povolená; takže pokud budete chtít posunutí provést pouze jednou na dané sepnutí kontaktu, může být užitečné použít "jednorázový" typ kontaktu povolení z přechodové cívky.



Parametry

Parametr	Popis
povolení	Když vstup povolení bude v jedničce a vstup R bude v nule, posunutí se vykoná. Všimněte si, že SHFR se vykoná jednou na každý cyklus, který je povolený.
R	Když vstup 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 vstup bude povolený a vstup R bude v nule.
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		•	•	•	•	•†	•	•	•	•		

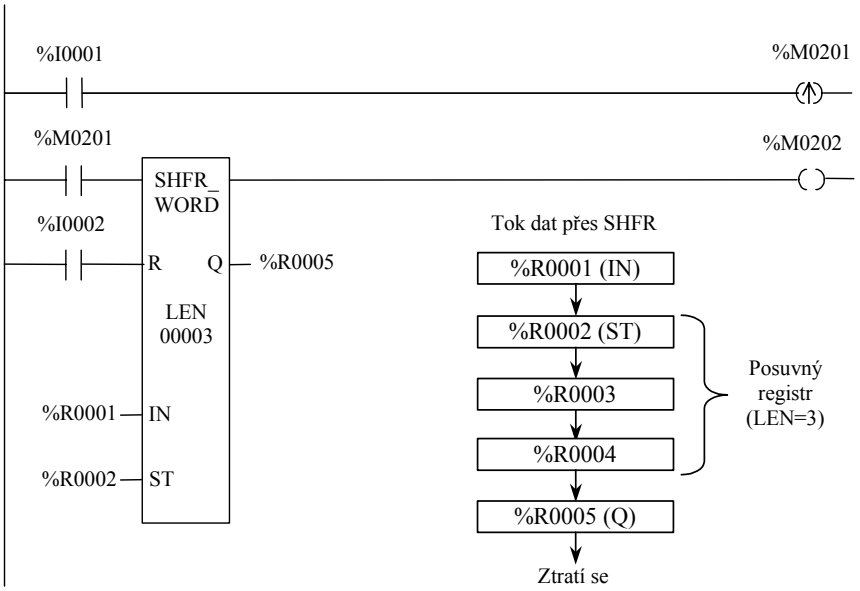
• Platná adresa pro data typu BIT, nebo WORD nebo místo, kudy proud může téct skrz funkci.
U funkce SHFR_BIT diskrétní uživatelské adresy %I, %Q, %M a %T nemusí být spojené s bajtem.

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

Příklad 1

V tomto příkladu registr posunutí pracuje se třemi (LEN=3) paměťovými místy %R0002 až %R0004. Když resetovací kontakt %I0002 bude v jedničce, tři slova registru posunutí se nastaví na nulu.

Když se kontakt %I0001 sepne, kontakt %M0201 na vstupu povolení SHFR se sepne na dobu jednoho cyklu. Tím se data v %R0004 posunou na výstupní adresu Q, %R0005 (data, která byla v %R0005, se ztratí). Data v %R0003 se posunou do %R0004; data v %R0002 se posunou do %R0003 a data v %R0001 (IN) se posunou do %R0002 (ST). Tok dat je znázorněný na obrázku níže. Pokud budete chtít, data je možno zacyklit pomocí stejné adresy na IN a Q.



Příklad 2

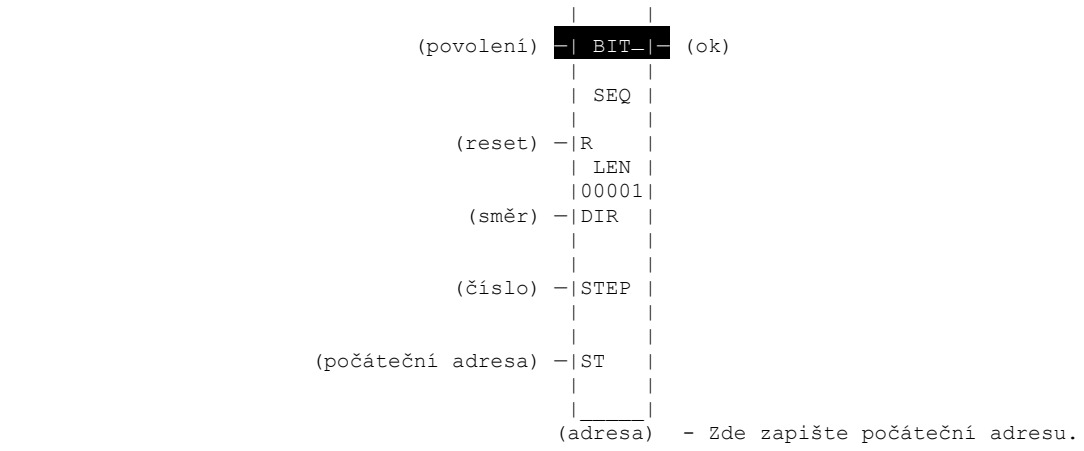
V příkladu 2 je posuvný registr typu BIT. Při LEN nastaveném na 100 bude pracovat s paměťovými místy %M0001 až %M0100. Když resetovací adresa CLEAR bude aktivní, funkce SHFR vyplní %M0001 až %M0100 nulami.

Když NXT_CYC (“jednorázový” kontakt přechodové cívky) bude v jedničce a CLEAR bude v nule, funkce SHFR vykoná posunutí dat v %M0001 až %M0100 nahoru o jeden bit. Bit v %Q0033 se posune do %M0001, přičemž bit posunutý ven z %M0100 se zapíše do Q (%M0200). Předchozí hodnota Q se ztratí.

```
|                _____
|NXT_CYC |      |
|—|  |——|SHFR_|—
|         |      |
|         | BIT  |
| CLEAR   |      |
|—|  |——|R   Q |—%M0200
|         | LEN  |
|         |00100 |
|         |      |
|         |      |
|         |      |
| %Q0033—|IN    |
|         |      |
|         |      |
| %M0001—|ST    |
|         |_____|
|
```

BITSEQ (BIT)

Funkce Bitového sekvenčního přepínače (BITSEQ) provede posunutí jedné logické jedničky sekvenčně po kruhové dráze přes pole bitů. Když se vykoná posunutí bitu na konec pole, provede se přetočení na druhý konec pole na další posunutí a pokračuje se odsud. Funkce BITSEQ má pět vstupních parametrů a jeden výstupní parametr.



Požadavek povolení vstupu

Aby funkce mohla vykonat jedno posunutí, povolení vstupu bitového sekvenčního přepínače vyžaduje přechod z logické nuly do logické jedničky, a nevykoná se znovu, dokud na vstup povolení nepřijde další přechod s náběžnou hranou. Proto používání kontaktu pozitivní přechodové cívky na vstupu povolení není nutné.

Vstup R (Reset)

Když tento vstup bude v jedničce, bitový sekvenční přepínač se nevykoná.

Resetový vstup (R) přepíše povolení (EN) a vždy resetuje sekvenční přepínač. Když R bude aktivní, aktuální číslo kroku se nastaví na hodnotu zadanou v číselném parametru STEP a všechny ostatní bity se nastaví na 0. Pokud nebude zadáno žádné číslo STEP (STEP=0), krok se nastaví na bit 1 a všechny ostatní bity se nastaví na 0.

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

Vstup STEP

- 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 (zůstane na výchozí hodnotě nula), 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.

Vstup DIR (Směr)

Směr rotace bitu je možno měnit nastavením vstupu DIR na jedničku nebo na nulu. Pokud bude v jedničce, bit se bude v poli inkrementovat. Pokud bude v nule, bit se bude dekrementovat.

Vstup ST (Počáteční adresa) a parametr LEN (Délka)

Vstup ST obsahuje paměťové místo počáteční adresy pole sekvenčního přepínače. Délka pole v bitech je nastavena v parametru LEN. Pokud například ST bude %M0001 a LEN se bude rovnat 16, pole se bude skládat z %M0001 až %M0016. Pokud ST bude adresa %R, pak LEN bude určovat, kolik po sobě jdoucích bitů v paměti %R bude patřit do pole. Pokud například ST bude %R0004 a LEN se bude rovnat osmi, v poli se použije pouze prvních osm bitů registru %R; posledních osm bitů %R0004 bitový sekvenční přepínač bude ignorovat.

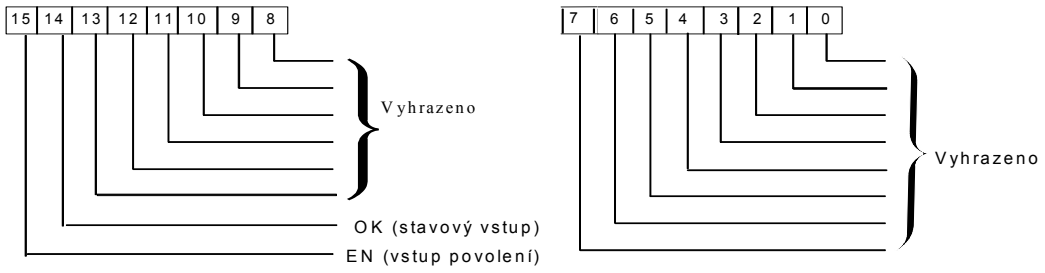
Řídicí blok paměti vyžadovaný pro bitový sekvenční přepínač

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 naprogramujete bitový sekvenční přepínač pomocí Logicmasteru, 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 v řídicím bloku 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

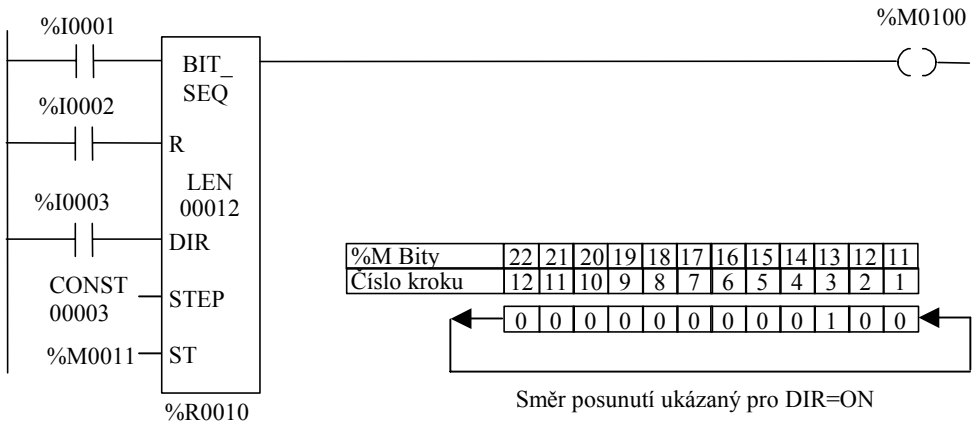
Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
adresa								•				
povolení	•											
R	•											
DIR	•											
STEP		•	•	•	•		•	•	•	•	•	•
ST		•	•	•	•	•†	•	•	•	•		•
ok	•											•

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

Příklad

V následujícím příkladu bitový sekvenční přepínač pracuje s bity %M0011 (zadáno na vstupu ST) až %M0022 (protože LEN se rovná dvanácti). Jeho řídicí blok tří slov je uložený v registrech %R0010, %R0011 a %R0012. Když %I0002 (na vstupu R) bude v jedničce, sekvenční přepínač se resetuje, což znamená, že bit pro krok tři (zadáno na vstupu STEP) se nastaví na logickou jedničku a všechny ostatní bity se nastaví na nulu.

Když %I0001 přejde do logické 1 (při %I0002 v nule), bit pro krok číslo 3 se vynuluje a buď se nastaví do jedničky bit pro krok číslo 4, pokud DIR bude v jedničce, nebo se nastaví do jedničky bit pro krok číslo 2, pokud DIR bude v nule.



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 obecný 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. Zadaný cíl (SYSID) se na tomto místě nenachází.
2. Zadané číslo úlohy (TASK) nebude pro cílové zařízení platné.
3. Délka dat je 0 (v povelovém bloku).
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

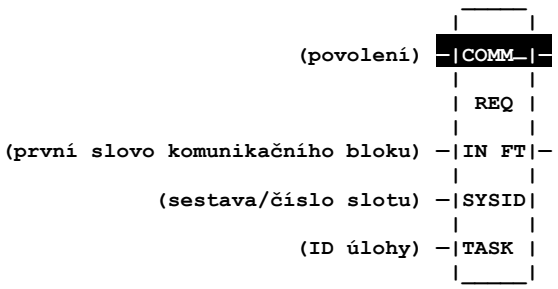
Povelový blok předává informace cílovému inteligentnímu modulu. Obsahuje číslo povelu, který se má vykonat, a veškerá přenášená data.

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 typu modulu adresovaného v COMMREQ a na množství posílaných dat.

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 povelový blok je možno umístit do určené oblasti paměti pomocí příslušné programovací funkce, například Přesunutí bloku nebo řada Posunutí.



Parametry

Parametr	Popis
povolení	Když vstup povolení bude v jedničce, požadavek na komunikaci se vykoná jednou během čtení. Pokud není žádoucí posílat COMMREQ několikrát, vstup povolení musí být kontakt přechodové cívky.
IN	IN obsahuje počáteční adresu prvního slova povelového bloku.
SYSID	SYSID obsahuje číslo sestavy (bajt s nejvyšší váhou) a číslo pozice (bajt s nejnižší váhou) cílového modulu.
TASK	TASK obsahuje ID úlohy procesu na cílovém modulu.
FT	Výstup FT (chyba) bude v jedničce, pokud se při zpracování COMMREQ zjistí chyba.

Poznámka

COMMREQ Series 90-30 **nemá** výstup OK.

Platné typy paměti

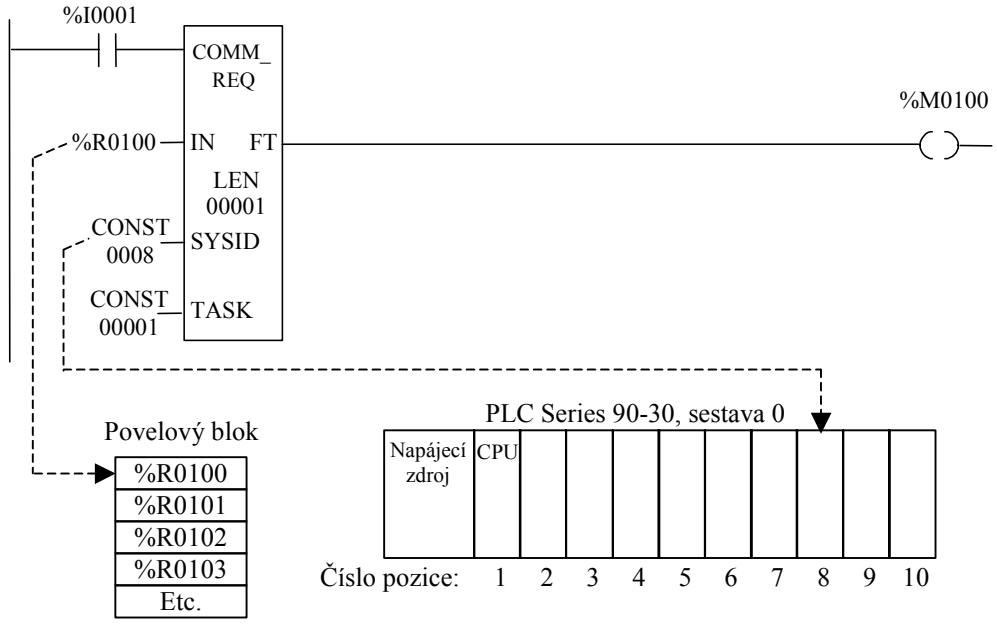
Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN								•	•	•		
SYSID		•	•	•	•		•	•	•	•	•	
TASK								•	•	•	•	
FT	•											•

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

Příklad

Když v následujícím příkladu vstup povolení %I0001 bude v jedničce, povelový blok začínající na %R0100 (zadáno na vstupu IN) se pošle do komunikační úlohy 1 (vstup TASK = 1) v modulu umístěného v PLC v sestavě 0, pozici 8 (SYSID=0008). Pokud se během zpracování COMMREQ vyskytne chyba, výstup Fault (FT) přejde do jedničky, čímž se sepne %M0100.

Všimněte si, že adresa na vstupu IN udává počáteční adresu povelového bloku. Také hexadecimální číslo na SYSID udává číslo sestavy a pozice cílového modulu; vyšší bajt se vztahuje k číslu sestavy a nižší bajt se vztahuje k číslu pozice. Proto SYSID s hodnotou 0008 v tomto příkladu znamená sestavu 00 a pozici 08. Sestava 0 (nula) se vždy vztahuje k hlavní nebo CPU sestavě, takže pokud cílový modul byl v expanzní nebo vzdálené sestavě, vyšší bajt SYSID bude obsahovat nenulové číslo, které bude odpovídat číslu nakonfigurované sestavy, kde je umístěný cílový modul.



Kapitola 10

Tabulkové funkce

Tabulkové instrukce se používají k vykonávání následujících funkcí:

Zkratka	Funkce	Popis	Strana
ARRAY_MOVE	Přesunout 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.	10-7
SRCH_NE	Hledat neshodné	Vyhledá všechny hodnoty v poli, které se nerovnají zadané hodnotě.	10-7
SRCH_GT	Hledat větší než	Vyhledá všechny hodnoty v poli, které jsou větší než zadaná hodnota.	10-7
SRCH_GE	Hledat větší nebo shodné	Vyhledá všechny hodnoty v poli, které jsou větší nebo se rovnají zadané hodnotě.	10-7
SRCH_LT	Hledat menší než	Vyhledá všechny hodnoty v poli, které jsou menší než zadaná hodnota.	10-7
SRCH_LE	Hledat menší nebo shodné	Vyhledá všechny hodnoty v poli, které jsou menší nebo se rovnají zadané hodnotě.	10-7

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 slova

* 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 v softwaru žebříkové logiky. 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)

Definovaná pole a datové prvky

Pro účely tohoto výkladu, **pole** je seskupení souvislé adresovatelné paměti PLC, například %R0100 až %R0120. **Datový prvek** jsou data uložená v jedné jednotce paměti pole. Pokud například pole bude typu Bit, pak každý datový prvek bude uložená v jediném bitu paměti, například %M0001 (nebo to může být jeden bit v paměti typu registr). Nebo pokud pole bude typu Word, pak každý datový prvek bude uložený jako 16-bitové slovo paměti, například %R0100 (nebo to může být 16 po sobě jdoucích bitů %I). Více informací k tomuto najdete v tabulce "Platné typy paměti".

Indexová čísla

Každý datový prvek pole má referenční číslo nazývané **indexové** číslo, které automaticky přiřazuje PLC. Indexové číslo indikuje pozici datového prvku v poli. Datové prvky jsou číslované ve vzestupném pořadí počínaje nejnižší paměťovou adresou v poli, která má přiřazené indexové číslo jedna.

Například následující pole typu Word má začáteční adresu %R0105. Má deset datových prvků, jejichž indexová čísla jsou 1 až 10.

Adresa	Index. číslo
%R0105	1
%R0106	2
%R0107	3
%R0108	4
%R0109	5
%R0110	6
%R0111	7
%R0112	8
%R0113	9
%R0114	10

Instrukce Přesunout pole

Funkci Přesunout pole použijte ke zkopírování zadaného počtu datových prvků ze zdrojového pole do cílového pole. Každé pole adresované instrukcí Přesunout pole má shodný počet datových prvků. Funkce Přesunout pole umožňuje, aby mezi zdrojovým a cílovým polem relativní pozice použité v přesunutí byly odlišné. Například tři datové prvky začínající na indexu 5 ve zdrojovém poli je možno zkopírovat do tří datových prvků v cílovém poli začínající na indexu 7.

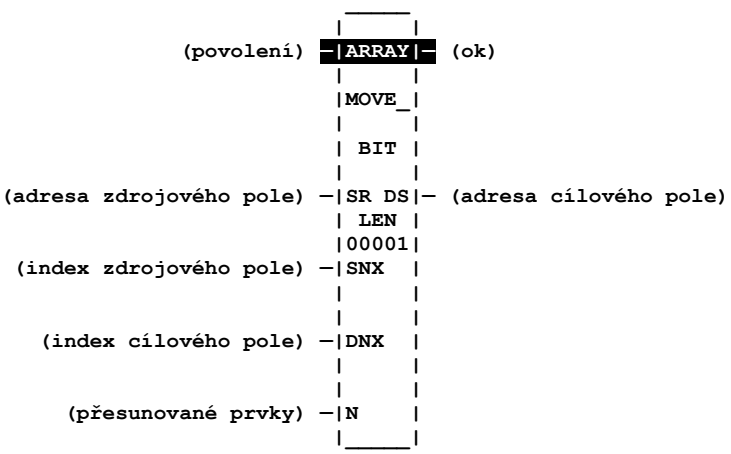
Funkce ARRAY_MOVE má pět vstupních parametrů a dva výstupní parametry. Když bude funkce povolená, počet datových prvků v indikátoru počtu (N) se zkopíruje ze vstupního pole počínaje indexovanou pozicí zadanou na vstupu SNX. Datové prvky se zapíšou do výstupního pole počínaje indexovanou pozici zadanou na DNX. 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. Hodnota zobrazená na obrazovce Logicmaster 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. Tento vztah používá PLC k zajištění, že se bude adresovat prvek mimo zdrojové pole.
- (N + DNX – 1) je větší než LEN. Tento vztah používá PLC k zajištění, že se bude adresovat prvek mimo cílové pole.
- SNX nebo DXN = 0.



Parametry

Parametr	Popis
povolení	Když funkce bude povolená, operace Přesunout pole 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 na obrazovce Logicmaster.
SNX	SNX obsahuje indexové číslo ve zdrojovém poli prvního kopírovaného datového prvku.
DNX	DNX obsahuje indexové číslo v cílovém poli prvního datového prvku, do kterého se má kopírovat.
N	Počet datových prvků, které se mají kopírovat.
ok	Výstup ok bude v jedničce, když skrz vstup 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 datových 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.

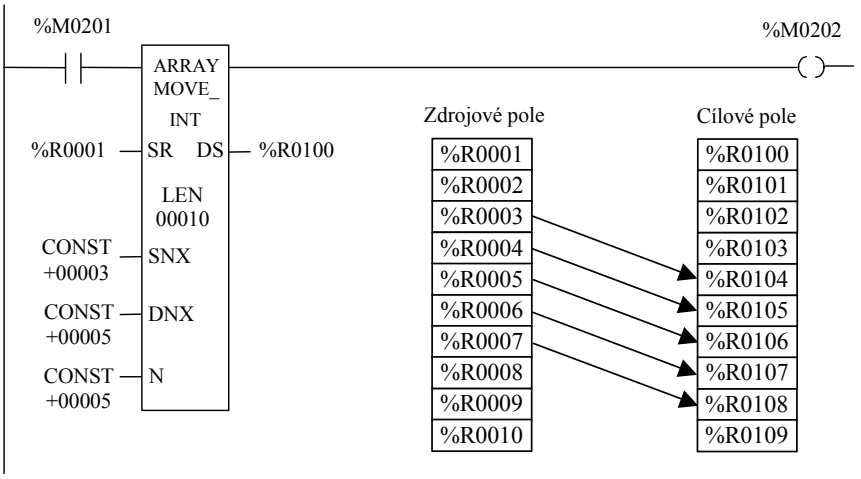
o Platná adresa pouze pro data typu INT, BIT, BYTE nebo WORD; neplatí pro DINT.

Δ Pouze platný typ dat BIT, BYTE nebo WORD; neplatí pro INT nebo DINT.

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

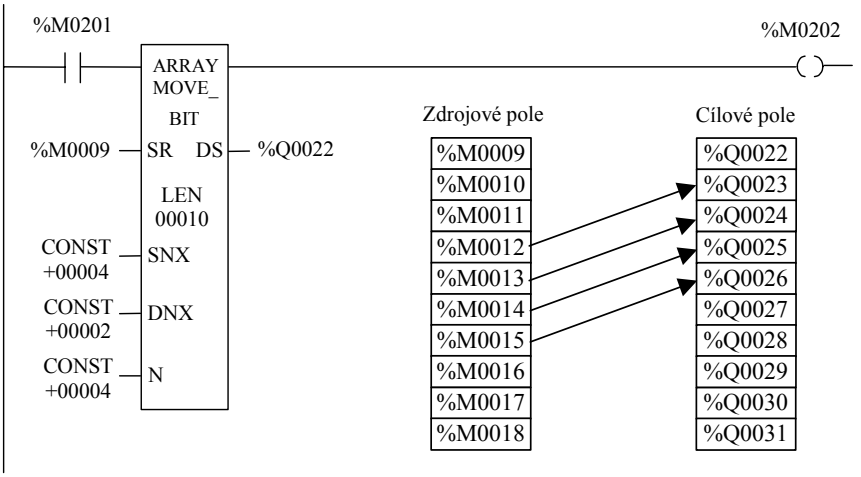
Příklad 1

V tomto příkladu jsou obě pole typu INT, mají délku 10 prvků (celá čísla) zadanou parametrem LEN=10. Jejich počáteční adresy jsou zadané pomocí SR a DS. Když kontakt povolení %M0201 bude sepnutý, ze zdrojového pole do cílového pole se zkopíruje pět datových prvků (zadáno pomocí N=5). Pět kopírovaných datových prvků zdrojového pole začíná indexovým číslem 3, protože SNX=3. Místa kopírovaná do cílového pole začínají indexovým číslem 5, protože DNX=5. Takže se přečtou %R0003 až %R0007 zdrojového pole a pak se zkopírují do %R0104 až %R0108 cílového pole.



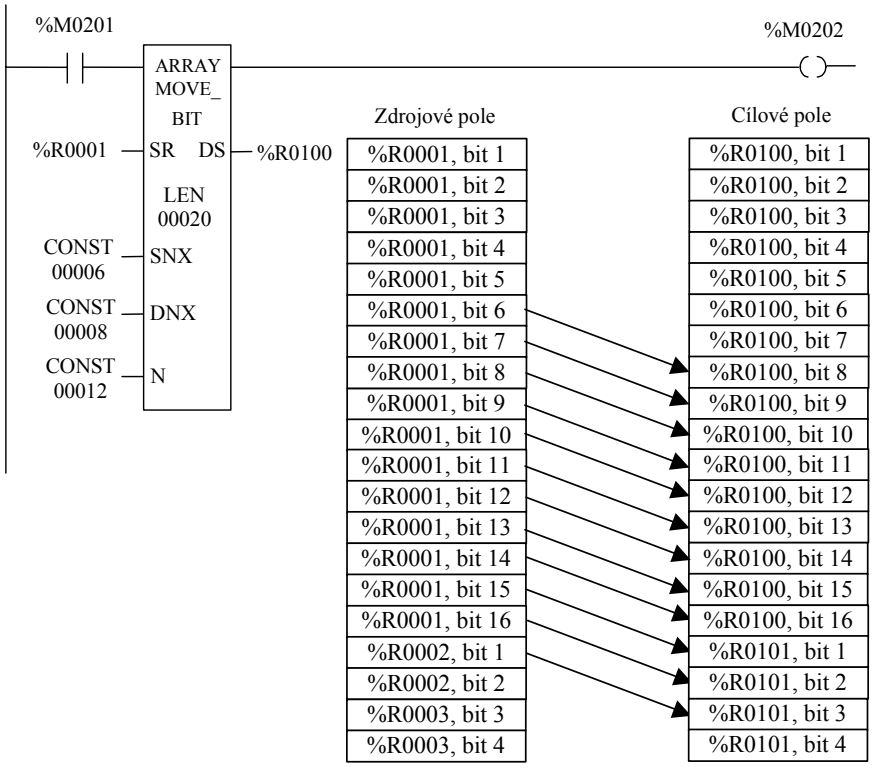
Příklad 2

V tomto příkladu jsou obě pole typu BIT, mají délku 10 prvků (bitů) zadanou parametrem LEN=10. Jejich počáteční adresy jsou zadané pomocí SR a DS. Když kontakt povolení %M0201 bude sepnutý, ze zdrojového pole do cílového pole se zkopírují čtyři prvky (zadáno pomocí N=4). Čtyři kopírované datové prvky zdrojového pole začínají indexovým číslem 4, protože SNX=4. Místa kopírovaná do cílového pole začínají indexovým číslem 2, protože DNX=2. Takže se přečtou %M0012 až %M0015 zdrojového pole a pak se zkopírují do %Q0023 až %Q0026 cílového pole.



Příklad 3

V tomto příkladu jsou obě pole typu BIT, mají délku 20 prvků (bitů), zadanou parametrem LEN=20. Jejich počáteční adresy jsou zadané pomocí SR a DS. Když kontakt povolení %M0201 bude sepnutý, ze zdrojového pole do cílového pole se zkopírují čtyři prvky (zadáno pomocí N=12). 12 kopírovaných datových prvků zdrojového pole začíná indexovým číslem 6, protože SNX=6. Místa kopírovaná do cílového pole začínají indexovým číslem 8, protože DNX=8. Takže se přečtou %R0001, bit 6 až %R0002, bit 1 zdrojového pole a pak se zkopírují do %R0100, bit 8 až %R0101, bit 3 cílového pole.



Funkce vyhledávání

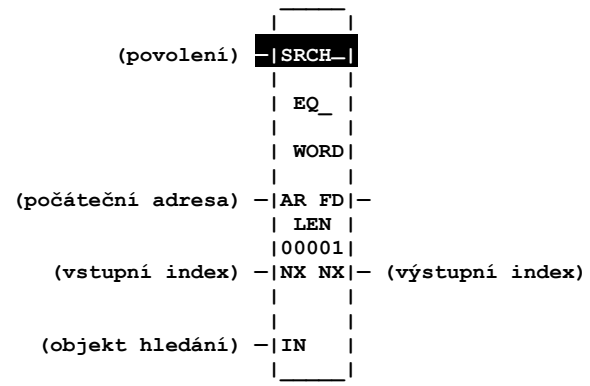
K prohledání všech hodnot pole pro tuto konkrétní operaci použijte příslušnou funkci vyhledávání uvedenou níže.

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, nastaví se výstupní parametr (FD) 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 hodnotu nula.



Parametry

Parametr	Popis
povolení	Když vstup povolení bude v jedničce, operace se provede.
AR	AR obsahuje počáteční adresu prohledávaného pole (cílového pole).
Vstup NX	Vstup NX obsahuje indexové číslo (v cílovém poli), kde má začít hledání.
IN	IN obsahuje hledaný objekt.
Výstup NX	Pokud se hledaný objekt najde, jeho pozice v poli (jeho indexové číslo) se zapíše sem.
FD	Tento výstup přejde do jedničky jako indikace, že hledaný objekt byl v poli nalezen.
LEN	LEN udává počet prvků začínajících na AR, které tvoří pole. Může být 1 až 32,767 bajtů nebo slov.

Platné typy paměti

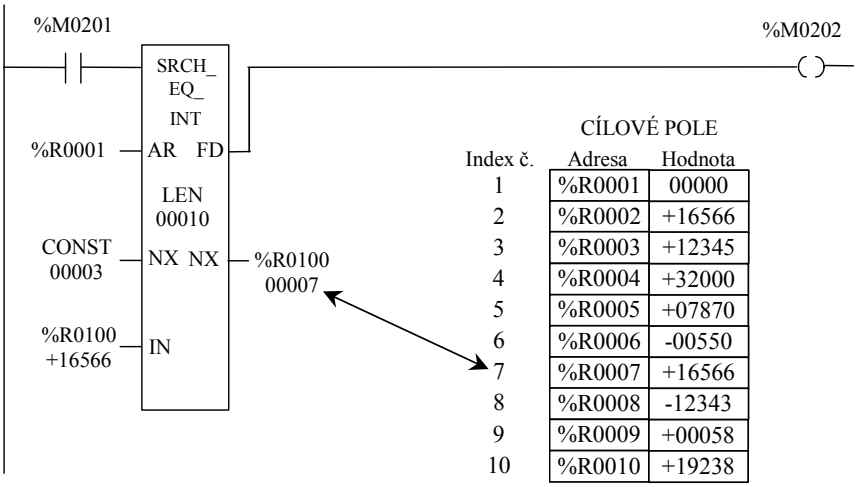
Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
AR		o	o	o	o	Δ	o	•	•	•		
NX in		•	•	•	•		•	•	•	•	•	
IN		o	o	o	o	Δ	o	•	•	•	•	
NX out		•	•	•	•		•	•	•	•		
FD	•											•

• 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 data typu BYTE nebo WORD; neplatí pro INT nebo DINT.

Příklad 1

Funkce SRCH_EQ (typu INT) v tomto příkladu prohledává blok paměti, který začíná na %R0001 (zadáno v AR) a pokračuje až do %R0010 (LEN=10). Hodnota, která je hledaná, je definována na IN a je +16566. Vstup NX s hodnotou 3 indikuje, že hledání má začít na čtvrtém datovém prvku, protože hodnota NX se při vykonání funkce inkrementuje o 1.

Když kontakt povolení %M0201 bude sepnutý, funkce SRCH_EQ bude prohledávat zadané pole počínaje indexovým číslem 4 a bude hledat hodnotu rovnající se hodnotě v IN, tedy +16566. Pokud tuto hodnotu najde v %R0007, který má indexové číslo 7, zapíše číslo 7 do výstupu NX na %R0100. Také nastaví výstup FD do jedničky, což znamená, že hledaný objekt byl v poli nalezen. Všimněte si, že i když adresa %R0002 také obsahuje hledanou hodnotu +16566, tento datový prvek nebyl do hledání zahrnutý, protože vstupní parametr NX s hodnotou 3 určuje, že hledání má začít na čtvrtém datovém prvku, což je %R0004.



CÍLOVÉ POLE

Index č.	Adresa	Hodnota
1	%R0001	00000
2	%R0002	+16566
3	%R0003	+12345
4	%R0004	+32000
5	%R0005	+07870
6	%R0006	-00550
7	%R0007	+16566
8	%R0008	-12343
9	%R0009	+00058
10	%R0010	+19238

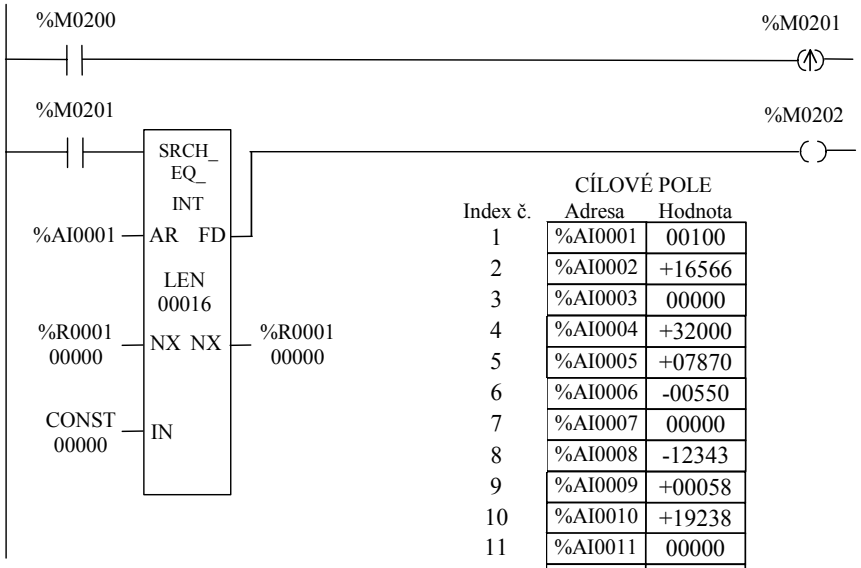
Příklad 2

Pole v tomto příkladu začíná na %AI0001 (zadáno na AR) a pokračuje do %AI0016 (LEN=16). Hodnota, která je hledaná, je definována na IN a je +16566. Vstup NX s počáteční hodnotou 0 indikuje, že hledání má začít na prvním datovém prvku v poli, protože hodnota NX se při vykonání funkce inkrementuje o 1.

Když se %M0200 sepne poprvé, funkce vykoná první hledání počínaje datovým prvkem 1 a bude hledat hodnotu rovnající se hodnotě na IN, tedy hodnotu 00000. Pokud tuto hodnotu najde v %AI0003, který má indexové číslo 3, zapíše číslo 3 do výstupu NX a vstupu NX, protože oba mají adresu %R0001. Také nastaví výstup FD do jedničky, což znamená, že hledaný objekt byl v poli nalezen.

Když se %M0200 sepne podruhé, vstupní hodnota NX, která je nyní nastavena na 3, se inkrementuje o 1, takže druhé hledání začne na čtvrtém prvku pole, %AI0004. Cílová hodnota 00000 se nyní nalezla v %AI0007, sedmém datovém prvku, takže do %R0001 se zapíše číslo 7. Podle této předlohy bude postupovat každé další prohledávání až do pátého prohledávání, ve kterém se nenajde žádný cíl. Protože se nenalezl žádný cíl, do %R0001 se zapíše 0, což zajistí, že když se spustí prohledávání znovu, začne na začátku pole.

Číslo prohledávání	Prohledávání začne na datovém prvku	Výsledek prohledávání (v %R0001)
1	1	3
2	4	7
3	8	11
4	12	15
5	16	0



CÍLOVÉ POLE

Index č.	Adresa	Hodnota
1	%AI0001	00100
2	%AI0002	+16566
3	%AI0003	00000
4	%AI0004	+32000
5	%AI0005	+07870
6	%AI0006	-00550
7	%AI0007	00000
8	%AI0008	-12343
9	%AI0009	+00058
10	%AI0010	+19238
11	%AI0011	00000
12	%AI0012	+16566
13	%AI0013	+12345
14	%AI0014	+32000
15	%AI0015	00000
16	%AI0016	-00550

Kapitola 11

Převodní funkce

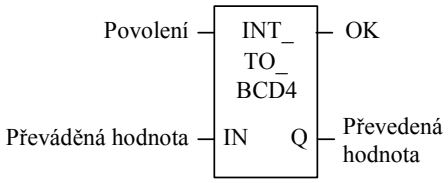
Převodní funkce se používají k převádění datových položek z jednoho číselného typu na jiný. Mnoho programovacích instrukcí, například matematické funkce, se musí používat s daty jednoho typu. Tato kapitola popisuje následující převodní funkce:

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 %M0017 až %M0032 se převede na čtyři BCD číslice a výsledek se uloží na paměťovém místě %Q0033 až %Q0048. K ověření úspěšného převodu se sepne cívka %M0032.

```
|                 _____
|%I0002  |       |                                                    %M0032
|—| |———| INT_|————————————————————————————————————————————————————( )—
|         | TO_ |
|         |     |
|         | BCD4|
| %M0017—|IN  Q|—%Q0033
|         |_____|
|
```

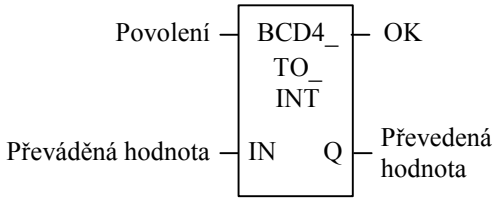
—>INT (BCD-4, REAL)

Funkce převodu na celé číslo se znaménkem se používá ke generování 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 a u všech verzí CPU352 a CPU37x.

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

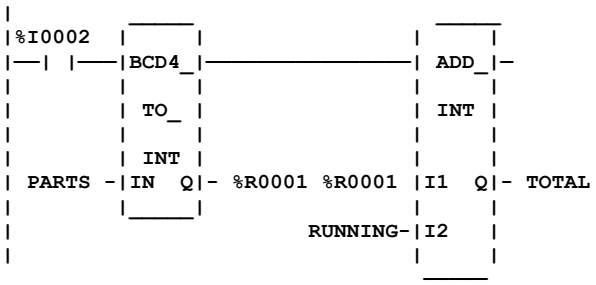
Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN		•	•	•	•		•	•	•	•	•	
ok	•											•
Q		•	•	•	•		•	•	•	•		

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

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

Příklad 1 – BCD4 na celé číslo

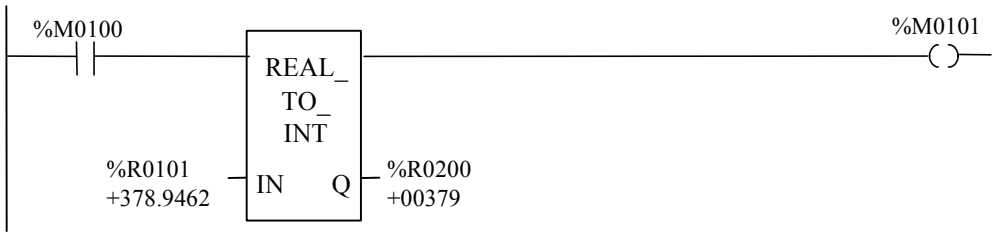
V následujícím příkladu, když vstup %I0002 bude v jedničce, hodnota BCD-4 v PARTS se převede na celé číslo se znaménkem a výsledek se zapíše na adresu %R0001. V následující funkci ADD se %R0001 přičte k hodnotě celého čísla se znaménkem představovaného adresou RUNNING. Součet vytvořený funkcí ADD se zapíše na adresu TOTAL.



Příklad 2 – Reálné číslo na celé číslo

Tento příklad ukazuje převod reálného čísla na adrese %R0101 na celé číslo na adrese %R0200. Když kontakt povolení vstupu %M0100 bude v jedničce, provede se převod. Všimněte si, že během převodu se reálné číslo zaokrouhlí na nejbližší celé číslo. Pokud desetinná část reálného čísla bude 0,5 nebo větší, výsledné celé číslo se zaokrouhlí nahoru. Pokud desetinná část reálného čísla bude menší než 0,5, výsledné celé číslo se zaokrouhlí dolů. V následujícím příkladu se reálná hodnota 378,9462 zaokrouhlí na celočíselnou hodnotu 379.

Pokud zaokrouhlení je nežádoucí, použijte funkci REAL_TRUN_INT, která během převodu ořízne desetinnou část reálného čísla bez ohledu na jeho hodnotu.



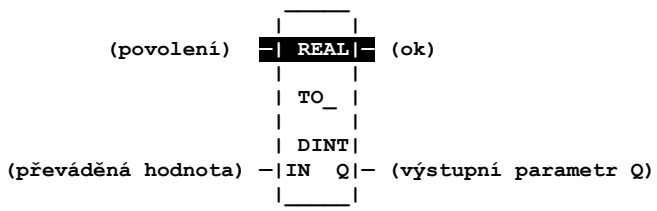
—>DINT (REAL)

Funkce převodu na celé číslo s dvojnásobnou délkou se znaménkem se používá ke generování 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 a u všech verzí CPU352 a CPU37x.

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

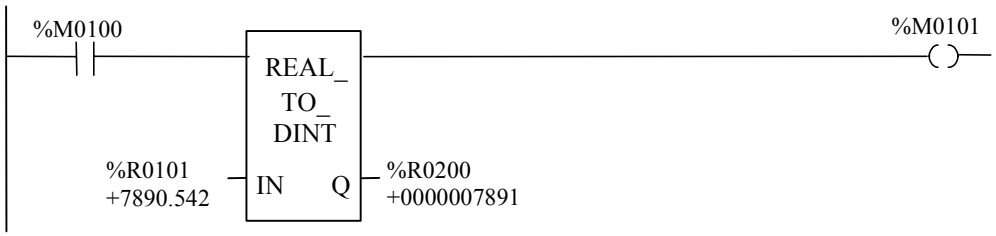
Parametr	proud	%I	%Q	%M	%T	%S	%G	%R	%AI	%AQ	konst.	žádný
povolení	•											
IN		o	o	o	o		o	•	•	•	•	
ok	•											•
Q								•	•	•		

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

Příklad

V následujícím příkladu, když vstup povolení %M0100 bude v jedničce, reálná hodnota na vstupní adrese %R0101 se převede na celé číslo se znaménkem s dvojnásobnou délkou a výsledek se zapíše na adresu %R0200. Všimněte si, že během převodu se reálné číslo zaokrouhlí na nejbližší celé číslo. Pokud desetinná část reálného čísla bude 0,5 nebo větší, výsledné celé číslo se zaokrouhlí nahoru. Pokud desetinná část reálného čísla bude menší než 0,5, výsledné celé číslo se zaokrouhlí dolů. V následujícím příkladu se reálná hodnota 7890,542 zaokrouhlí na celočíselnou hodnotu s dvojnásobnou délkou 7891.

Pokud zaokrouhlení je nežádoucí, použijte funkci REAL_TRUN_DINT, která během převodu ořízne desetinnou část reálného čísla bez ohledu na jeho hodnotu.



—>REAL (INT, DINT, BCD-4, WORD)

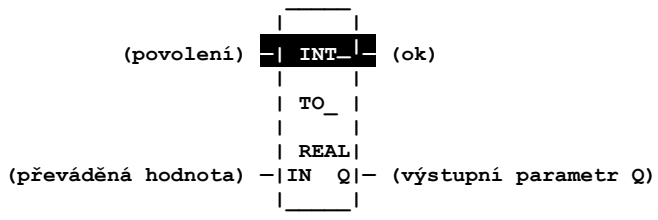
Funkce převodu na reálné číslo se používá ke generování reálné hodnoty vstupních dat. Tato funkce původní data nemění.

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 a u všech verzí CPU352 a CPU37x.



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.

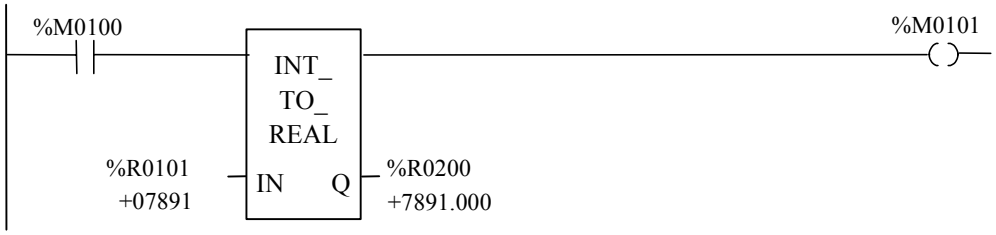
Platné typy paměti

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.

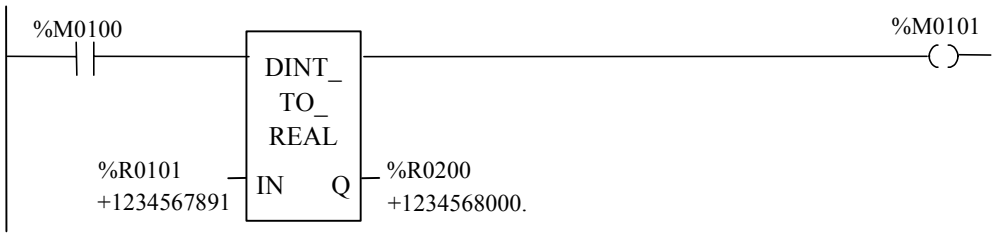
Příklad 1 - Převod celého čísla na reálné číslo

V následujícím příkladu celočíselná hodnota na vstupu IN je +07891. Výsledná hodnota umístěná do %R0200 po převodu do formátu reálného čísla je +7891,000.



Příklad 2 – Převod celého čísla s dvojnásobnou délkou na reálné číslo

V následujícím příkladu celočíselná hodnota s dvojnásobnou délkou na vstupu IN je +1234567891. Výsledná hodnota umístěná do %R0200 po převodu do formátu reálného čísla je +1234568000. Všimněte si, že celé číslo s dvojnásobnou délkou má 10 významných míst, ale reálné číslo má pouze 7 významných míst; proto se celé číslo během převodu na reálné číslo zaokrouhlí na 7 významných míst. V uvedeném příkladu se čtyři nejméně významné číslice 7891 celého čísla s dvojnásobnou délkou zaokrouhlí na 8000 ve čtyřech nejméně významných místech reálného čísla.



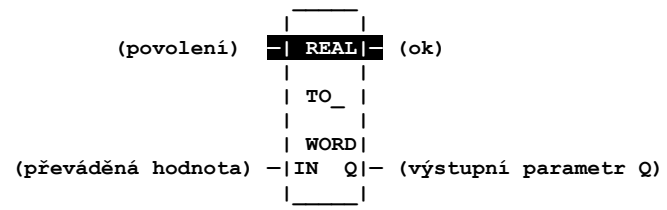
—>WORD (REAL)

Funkce převodu na WORD se používá ke generování WORD ekvivalentu reálných dat. Tato funkce původní data nemění.

Poznámka

Tuto funkci je možno použít pouze u CPU řady 35x, 36x a 37x.

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		•	•	•	•		•	•	•	•		

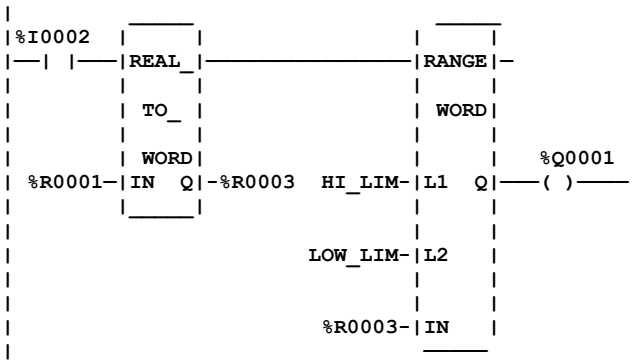
• Platná adresa nebo místo, kde funkcí může téct proud.

Příklad – Převod reálného čísla na slovo

Protože v tomto příkladu funkce RANGE není k dispozici jako typ REAL, reálná hodnota v %R0001 se nejdříve převede na hodnotu slova (v %R0003), která se pak použije jako vstup pro následující funkci RANGE WORD.

Následující tabulka ukazuje hodnoty na různých vstupech a výstupech pro následující obrázek.

Údaj	Hodnota nebo stav
%R	15767,83
%R0003	3A89h (15 768 dekadicky)
HI LIM	4E20h (20 000 dekadicky)
LOW LIM	2710h (10 000 dekadicky)
Q1	ON



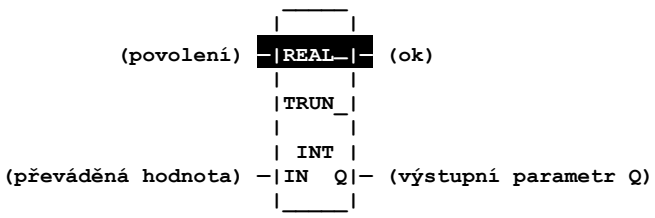
TRUN (INT, DINT)

Funkce Celá část se používá k zaokrouhlení reálného čísla směrem k nule. Během převodu se všechna čísla ve výstupním číslu napravo od desetinné tečky zanedbají. Tato funkce původní číslo nemění.

Poznámka

CPU řady 35x a 36x (verze 9.00 nebo pozdější a všechny verze CPU352) a 37x jsou jediná 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/37x 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 INT nebo DINT.

Poznámka

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

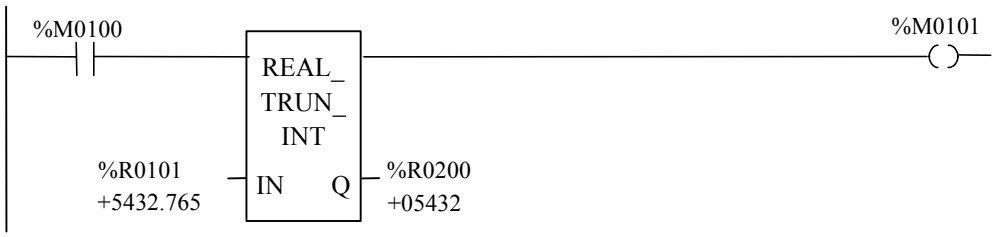
Platné typy paměti

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.

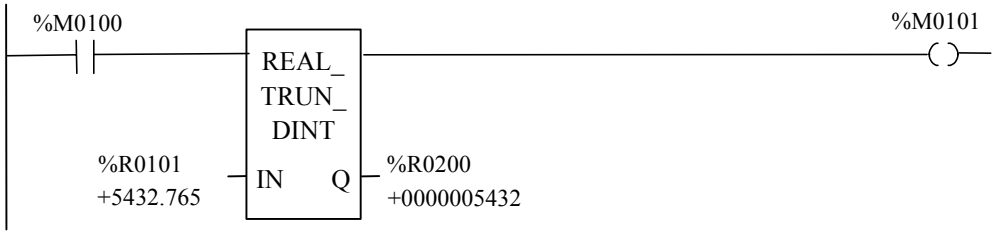
Příklad 1 – Zaokrouhlení reálného čísla na celé číslo s výstupní cívkou pro CPU352

V následujícím příkladu se hodnota %R0101 zaokrouhlí (desetinná část se zanedbá) a výsledná celočíselná hodnota +05432 se uloží do %R0200. Pokud se použije CPU352, %M0101 přejde do jedničky a bude indikovat úspěšný převod. Pokud se použije některé jiné CPU řady 35x, 36x nebo 37x, na výstupu OK se nebude generovat žádný proud, takže se nenaprogramuje žádná cívka.



Příklad 2 – Zaokrouhlení reálného čísla s dvojnásobnou délkou na celé číslo s výstupní cívkou pro CPU352

V následujícím příkladu se hodnota %R0101 zaokrouhlí (desetinná část se zanedbá) a výsledná celočíselná hodnota s dvojnásobnou délkou +0000005432 se uloží do %R0200. Pokud se použije CPU352, %M0101 přejde do jedničky a bude indikovat úspěšný převod. Pokud se použije některé jiné CPU řady 35x, 36x nebo 37x, na výstupu OK se nebude generovat žádný proud, takže se nenaprogramuje žádná cívka.



Kapitola 12

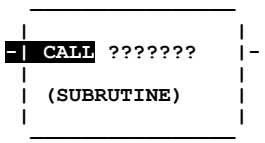
Řídicí funkce

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 buď 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).	12-23
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).	12-24
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).	12-30
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).	12-31
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).	12-33
COMMENT	Umístí do programu komentář (vysvětlení příček). Po naprogramování instrukce je možno text zapsat "nakouknutím" do instrukce (k vyvolání zoom použijte klávesu F10).	12-34
SVCREQ	Požaduje speciální službu PLC. (Viz seznam služeb na straně 12-35.)	12-35
PID	Vytváří dva PID (proporcionálně/integračně/derivační) řídicí algoritmy s uzavřenou smyčkou. • Standardní ISA PID algoritmus (PIDISA). • Algoritmus s nezávislým členem (PIDIND).	12-70

CALL

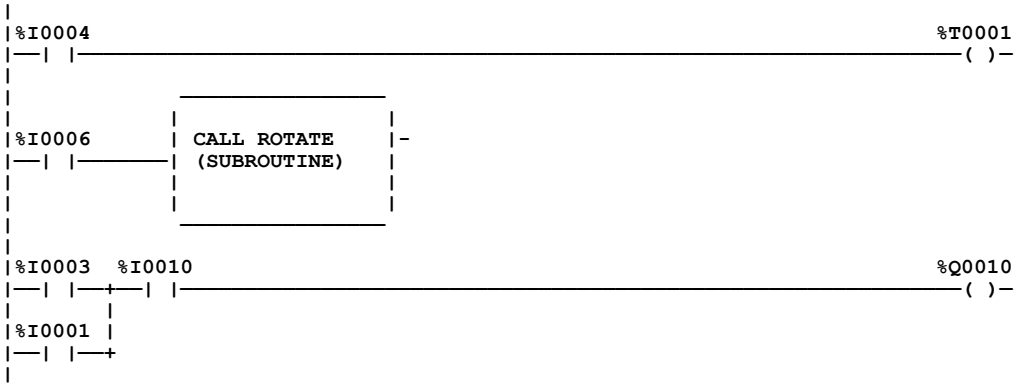
Použití funkce CALL provede během vykonávání programu odskok do určitého podprogramu.



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 na příčku v logice hned za instrukci CALL.

Příklad

V následujícím příkladu je naprogramovaná instrukce CALL k vyvolání podprogramu s názvem ROTATE, když bude sepnutý kontakt %I0006. (Všimněte si, že než můžete v instrukci CALL zapsat název podprogramu, tento název podprogramu již musí existovat v tabulce Deklarace bloků.) Nastavením kurzoru na v instrukci CALL můžete stisknutím **F10** nahlédnout do podprogramu a prohlédnout si logiku podprogramu. Jakmile bude podprogram vyvolaný, vykonávání programu odskočí na podprogram, který se vykoná až do konce, pak se vykonávání programu předá na příčku následující za příčkou volání. V následujícím příkladu se volá podprogram z druhé příčky, takže když se dokončí vykonávání podprogramu, čtení programu bude pokračovat třetí příčkou.



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 zadaných 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. Za normálních okolností se vstupní tabulka aktualizuje během části PLC cyklu, kdy se čtou vstupy, a neaktualizuje se znovu až do dalšího cyklu. Výstupní tabulky se aktualizují během části cyklu PLC, kdy probíhá řešení logiky, ale aktualizace výstupních modulů se neprovede, dokud se nedokončí části řešení logiky. S funkcí DO I/O je možno vynutit aktualizaci vstupních tabulek a výstupních modulů během části čtení, kdy probíhá řešení logiky. Tato schopnost vám umožní načíst změny vstupů a zapisovat na výstupy rychleji, než by bylo možno s normálním čtením PLC. Více informací o cyklu PLC najdete v kapitole 2.

Pokud budou zadané vstupní adresy, funkce umožní zjistit programové logice nejnovější hodnoty vstupů (zapsané ve vstupních tabulkách). Pokud vstupní adresy budou zadané, DO I/O provede aktualizaci výstupních modulů podle nejnovějších hodnot vstupů uložených v I/O paměti. I/O se obsluhuje v inkrementech celých I/O modulů; PLC v případě potřeby během vykonávání funkce upraví adresy.

Použití se vstupními moduly

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 příslušných vstupních tabulek 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 příslušných vstupních tabulek se provede.

Použití s výstupními moduly

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 I/O modulech. Vykonávání programu se pak vrátí na další funkce, která následuje za DO I/O.

Použití s přídavnými moduly

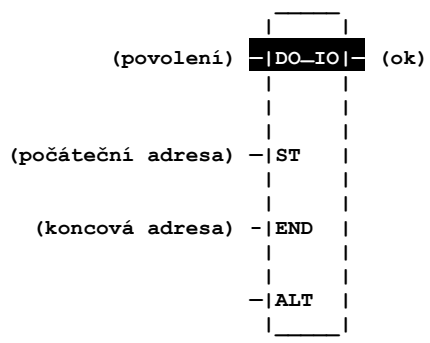
Pokud rozsah adres bude zahrnovat přídavný modul (HSC, APM, atd.), pak se načtou všechna vstupní data (%I a %AI) nebo všechna výstupní data (%Q a %AQ) pro tento modul. Parametr ALT se během čtení přídavných modulů ignoruje. Pokud se také požaduje použít DOIO s modulem rozšířené GCM (IC693CMM302), musí být splněný požadavek následující poznámky.

Poznámka

Funkci DOIO je možno ***použít pouze*** s rozšířeným GCM modulem (IC693CMM302) v systémech s CPU verze 9.0 a pozdější.

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 vstup povolení bude v jedničce, provede se omezené čtení vstupů nebo výstupů.
ST	ST je počáteční adresa skupiny vstupních nebo výstupních bodů nebo slov, které se mají obsloužit.
END	END je koncová adresa skupiny 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 dále v této kapitole). **Pokud se funkce ALT nepoužije, tento vstup musí být ponechaný prázdný; pokud se pro ALT naprogramuje hodnota 0, na CPU se může objevit chyba překročení doby hlídacího obvodu.**
ok	Výstup ok bude v jedničce, když se čtení vstupu nebo zápis na výstup dokončí normálně.

Poznámka

Funkce rozšířeného DOIO je možno použít u CPU model 331 a pozdější. V rozšířeném DOIO se může parametr ALT použít k zadání čísla pozice jednoho diskrétního vstupního nebo výstupního modulu v hlavní sestavě. Tato funkce rozšířeného DOIO se vykoná během 80 mikrosekund místo 236 mikrosekund, potřebných, když funkce DOIO bude naprogramovaná bez parametru ALT. K zamezení překrývání adres nebo nesouladu typů modulů se neprovádí žádná kontrola chyb. Podrobnosti viz odstavec "Funkce rozšířeného DO I/O" dále v této kapitole.

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 povolení %M0001 bude ve stavu ON, načtou se adresy %I0001 (zadáno v ST) až %I0064 (zadáno v END) a %Q0001 přejde do jedničky. Kopie načtených vstupů se uloží do interní paměti od adresy %M0001 (zadáno v ALT) do %M0064. Protože v ALT bylo zadáno alternativní umístění, DO_IO neprovede aktualizaci vstupní tabulky %I. Tento tvar funkce je možno použít k porovnání aktuálních hodnot vstupních bodů s jejich předchozími hodnotami (tj. jejich hodnotami na začátku čtení řešení logiky).

Příklad vstupu 2

V následujícím příkladu, když vstup povolení %M0001 bude ve stavu ON, načtou se adresy %I0001 (zadáno v ST) až %I0064 (zadáno v END) a %Q0001 přejde do jedničky. Protože v ALT bylo zadáno alternativní paměťové místo, funkce DO_IO použije načtené vstupní hodnoty k aktualizaci vstupní tabulky od adresy %I0001 do %I0064. Tento tvar funkce umožňuje načíst a aktualizovat vstupní body jednou nebo vícekrát během části cyklu CPU, kdy se vykonává řešení logiky. Všimněte si, že když se nepoužije vstup ALT, musí zůstat prázdný, jak je znázorněno. Nezadávejte na vstup ALT nulu, protože to bude mít za následek chybu hlídacího časovače.

Příklad výstupu 1

V následujícím příkladu, když vstup povolení %M0001 bude ve stavu ON, hodnoty analogových výstupních kanálů %AQ001 (zadáno v ST) až %AQ004 (zadáno v END) se zapíšou do adres %R0001 (zadáno v ALT) až %R0004 a %Q0001 přejde do jedničky. Protože v ALT bylo zadáno alternativní paměťové místo %R0001, DO_IO nezapíše hodnoty v %AQ001 až %AQ004 do analogových výstupních modulů.

Příklad výstupu 2

V následujícím příkladu, když vstup povolení %M0001 bude ve stavu ON, hodnoty na adresách %AQ001 až %AQ004 se zapíšou do analogových výstupních kanálů %AQ001 až %AQ004 na příslušných analogových výstupních modulech a %Q0001 přejde do jedničky. DO_IO provede aktualizaci analogových výstupních modulů, protože v ALT nebylo zadáno žádné alternativní paměťové místo. Všimněte si, že když se nepoužije vstup ALT, musí zůstat prázdný, jak je znázorněno. Nezadávejte na vstup ALT nulu, protože to bude mít za následek chybu hlídacího časovače.

Rozšířená funkce DO I/O pro CPU 331 a pozdější

Upozornění

Program obsahující funkci rozšířeného DO I/O se nesmí načíst verzí softwaru Logicmaster 90-30/20 software starší než 4.01.

Funkci rozšířeného DO I/O (DOIO) je možno použít u CPU model 331 a pozdější verze 4.20 a pozdější CPU. 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 cílového modulu. Například konstantní hodnota 2 v ALT udává, že cílem je modul v pozici 2. Parametry ST a END nastavují rozsah použité paměti.

Poznámka

Jediná kontrola, související s funkčním blokem rozšířeného DOIO, je základní kontrola stavu cílového modulu.

Funkce rozšířeného DOIO platí pouze pro moduly umístěné v sestavě modulárního CPU. 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 (ST) a koncová (END) 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 7 hlavní sestavy (CPU) 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 (Sekvenční záznamník událostí)

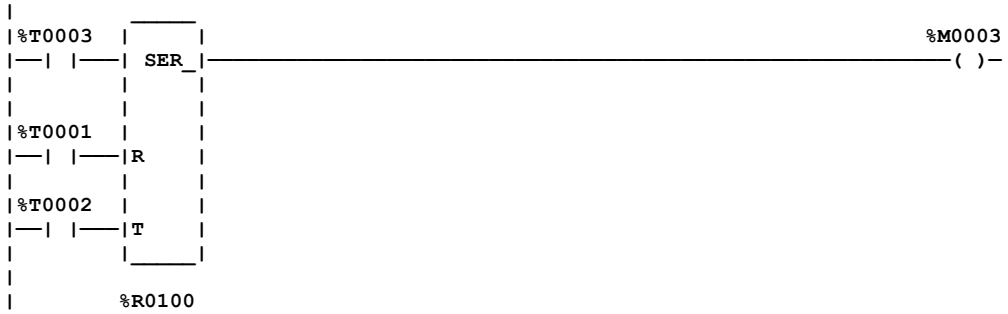
Vyžaduje CPU 35x nebo 36x s firmwarem 9.00 nebo pozdější nebo CPU37x

- Funkční blok SER (Sekvenční záznamník událostí) shromažďuje diskrétní vzorky (pracuje pouze s diskrétními daty). 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 s 32 bity na vzorek.
- Pokud funkční blok SER bude vložený do periodického podprogramu, četnost vzorkování bude vycházet z četnosti 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 sekunda) 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.)
- Vstupní modul může být volitelně zadán tak, že se bude číst při každém vykonání SER. To pomůže zajistit, aby data zachycená ze zadaného modulu byla pokud možno aktuální.

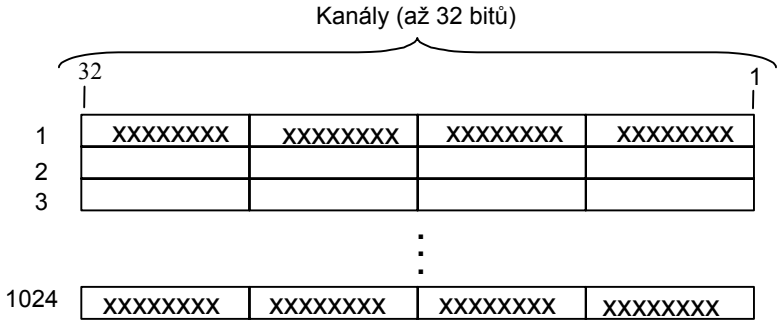
Poznámka

Synchronizace mezi PLC se nepodporuje.

Funkční blok SER má jeden výstup a tři vstupy: povolení, reset (R) a spouštění (T).



Jak je znázorněno níže, je možno nakonfigurovat 8, 16, 24 nebo 32 kanálů, kdy každý kanál představuje jeden diskrétní bod. Je možno také zadat až 1024 vzorků.



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 dat. 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 = Resetováno 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 splněna hodnota 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 12-17.)
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 zkopírují do zásobníku vzorků při každém vykonání funkčního bloku. Platné hodnoty jsou 8, 16, 24 nebo 32 bitů. Všechny nepoužité kanály musí být nakonfigurované s prázdným popisem kanálu. (Viz slova 14 až 77.) Pokud například bude zapotřebí 19 bitů, musíte jich nakonfigurovat 24 a zadat, že pět posledních je prázdných.
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 nulu (Aktivační vstup).
11	Pozice vstupního modulu	Zadává pozici v hlavní sestavě (sestava 0) vstupního modulu, který se bude číst při každém vykonání SER. Pokud hodnota bude 0, nebude se číst žádný modul. 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 (Typ paměti)	Udává typ dat přiřazený Bloku dat. Pokud například budete chtít použít paměťový typ %R, 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 segmentu 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.

Následující tabulka uvádí přehled, jak jsou určené režimy vzorkování.

Režim	Slovo 2	Slovo 10
Před aktivací	0	0
Během aktivace	0	Od 1 do (počet vzorků - 1)
Po aktivaci	0	Rovná se počtu vzorků (zadáno ve slově 9)
Plný zásobník	1	Slovo 10 a signál aktivace vstupu se ignorují

Aktivací řízené vzorkování

Aby bylo možno nakonfigurovat režim vzorkování před aktivací, během aktivace a po aktivaci, je nutno zvolit Režim aktivace (slovo 2 = 0). Režim vzorkování se řídí hodnotou Počet 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 zadaný v parametru. Když se vzorkování dokončí, výstupní signál OK funkce SER přejde do jedničky.

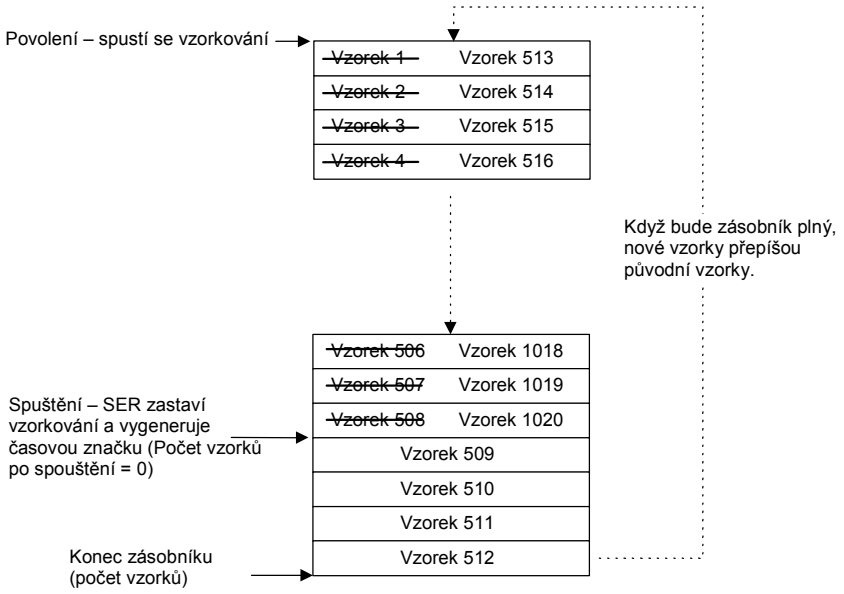
Pokud se před splněním podmínky Počet vzorků po aktivaci načte více než Počet vzorků (slovo 9), zásobník se "přetočí", to znamená, že SER se vrátí na začátek zásobníku a přepíše počáteční vzorky.

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 jsou načtené před aktivací).

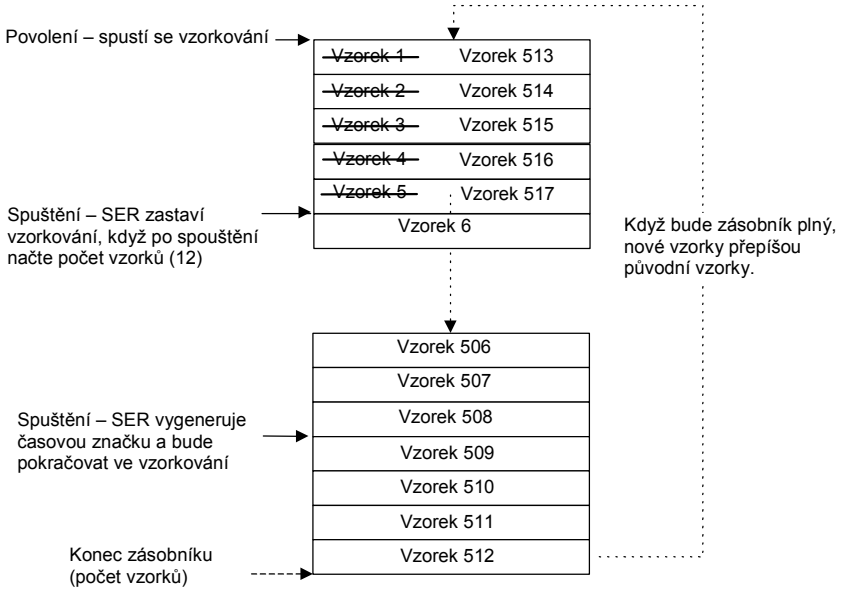


Obrázek 12-1. Příklad vzorkování SER před aktivací (pro 512 vzorků)

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 - 1). 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.

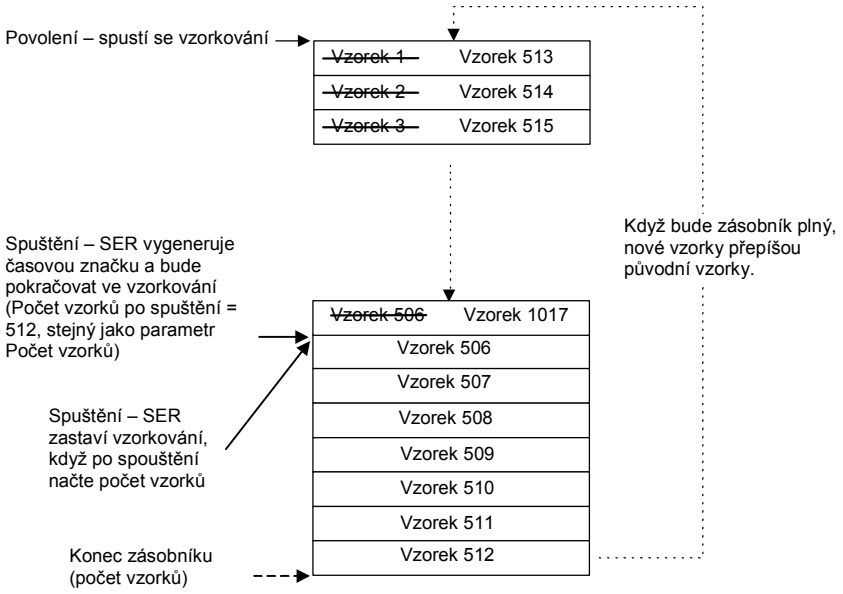


Obrázek 12-2. Příklad vzorkování SER během aktivace (pro 512 vzorků)

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. (Poznámka: všechny vzorky jsou načtené po aktivací).



Obrázek 12-3. Vzorkování SER po aktivaci (pro 512 vzorků)

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 9) a zásobník vzorků se nezaplní. Když zásobník vzorků bude plný, vygeneruje se značka Aktivační čas a výstup OK funkčního bloku přejde do jedničky.

Formáty časových značek pro spuštění funkčního bloku SER

BCD formát		
Číslo slova datového bloku	**Obsah (Horní bajt/Dolní bajt)**	**Navrhovaný formát prohlížení**
Slovo 2	měsíc/rok	Hex (MMRR)
Slovo 3	hodiny/den v měsíci	Hex (HHDD)
Slovo 4	sekundy/minuty	Hex (SSMM)
Slovo 5	Nepoužívá se	Samé nuly

Formát POSIX		
Číslo slova datového bloku	**Obsah**	**Navrhovaný formát prohlížení**
Slova 2 a 3	Počet sekund od 1. ledna 1970	Dint
Slova 4 a 5	Počet nanosekund do další sekundy	Dint

Příklad

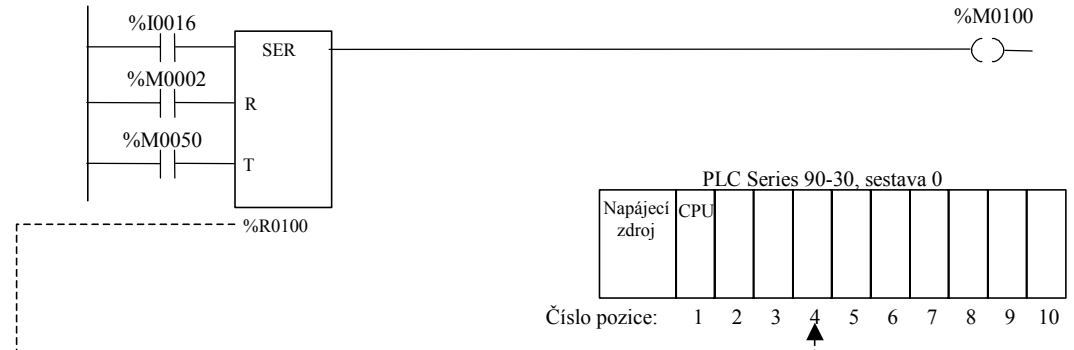
Další dvě tabulky uvádějí, jak se aktivační čas 3. listopadu 1998 v 8:34:05.010 zobrazí ve formátech BCD a POSIX v datovém bloku, který začíná na %R0201 (slovo 0).

3. listopadu 1998 v 8:34:05.010 ve formátu BCD		
Registr	**Parametr**	**Hodnota (hex)**
%R0203	měsíc/rok	1198
%R0204	hodiny/den v měsíci	0803
%R0205	sekundy/minuty	0534
%R0206	Nepoužívá se	0000

3. listopadu 1998 v 8:34:05.010 ve formátu POSIX			
Registr	**Parametr**	**Hodnoty (dekadicky)**	**Hodnota (hex)**
%R0203/R0204	Sekundy	910,082,045	363EBFFD
%R0205/R0206	nanosekundy	010,000,000	00989680

Příklad SER

V následujícím textu jsou ukázané vztahy instrukce žebříkové logiky, řídicího bloku v paměti PLC a příslušného vstupního modulu v PLC. Řídicí blok byl nastavený, jak je popsáno v tabulce 12-1.



Řídicí blok			
Č. slova	**Adresa**	**Hodnota**	**Popis**
0	%R0100	0	Stavové slovo
1	%R0101	0	Přídavné stavové slovo
2	%R0102	0	Režim aktivace (0 = vstup aktivace)
3	%R0103	0	Formát režimu aktivace (0 = BCD)
4	%R0104	0	Vyhrazeno
5	%R0105	0	Vyhrazeno
6	%R0106	0	Vyhrazeno
7	%R0107	0	Vyhrazeno
8	%R0108	24	Počet bitů na kanál
9	%R0109	512	Velikost bufferu
10	%R0110	12	Počet vzorků po aktivaci
11	%R0111	4	Udává vstupní modul na pozici 4
12	%R0112	8	Typ paměti bloku dat (%R)
13	%R0113	200	Offset paměti bloku dat (201)
14 ↓	%R0104 ↓	Popisy kanálů	

%R0201

Blok dat			
Č. slova	**Adresa**	**Hodnota**	**Popis**
0	%R0201	0	Číslo offsetu aktuálního vzorku
1	%R0202	0	Číslo offsetu aktivačního vzorku
2	%R0203	0	Měsíc a rok (MMRR)
3	%R0204	0	Hodina a den (HHDD)
4	%R0205	0	Sekundy a minuty (SSMM)
5	%R0206	0	Nepoužito
6 ↓	%R0207 ↓	Buffer vzorků (velikost zadaná řídicím blokem slovo 9)	

Příklad řídicího bloku funkce

V tomto příkladu byl jako cíl pro vzorek zadaný (ve slově 11) diskrétní vstupní modul se 16 body v sestavě 0 v pozici 4. Vykonával se dostatečně dlouho, aby se shromáždilo 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	%R0101	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	%R0102	Režim aktivace	0	0000	Záznamník událostí je nakonfigurovaný k aktivaci podle vstupu Aktivace.
3	%R0103	Formát Aktivačního času	0	0000	0=BCD
4	%R0104	Vyhrazeno	0	0000	Vyhrazené parametry musí být vždy nastavené na 0.
5	%R0105	Vyhrazeno	0	0000	
6	%R0106	Vyhrazeno	0	0000	
7	%R0107	Vyhrazeno	0	0000	
8	%R0108	Počet kanálů	24	0018	Každý vzorek se skládá z 24 bitů (3 bajtů) dat.
9	%R0109	Počet vzorků, které se mají odečíst	512	0200	Velikost zásobníku je 512 vzorků. Všimněte si, že velikost zásobníku vzorků se rovná 512 × (24/8) = 1536 bajtů nebo 768 slov. (Každý vzorek má délku 3 bajty, jak je zadáno výše ve slově 8.)
10	%R0110	Počet vzorků po aktivaci	12	000C	Počet vzorků, které se mají načíst po výskytu aktivace, je 12.
11	%R0111	Pozice vstupního modulu	4	0004	Vstupní modul v sestavě 0 v pozici 4 se přečte, když se vykoná SER tak, aby jeho hodnoty byly k dispozici pro vzorkování funkcí SER.
12	%R0112	Volič typu bloku dat	8	0008	Datový typ je 0x08 (%R).
13	%R0113	Offset bloku dat	200	00C8	Tento offset 200 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.

Pokračování na následující stránce

Slovo	Registr	Parametr	Hodnota (dek)	Hodnota (hex)	Popis
Popis kanálu		Zbývající slova obsahují popisy kanálů. V tomto příkladu bylo definováno šest popisů kanálů.			
14	%R0114	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	%R0115	Offset	0	0000	
16	%R0116	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	%R0117	Offset	0	0000	
18	%R0118	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	%R0119	Offset	12	0012	
20	%R0120	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	%R0121	Offset	8	0008	
22	%R0122	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	%R0123	Offset	0	0000	
24	%R0124	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	%R0125	Offset	0	0000	

Konfigurace kanálu pro výše uvedený příklad

32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
U	U	U	U	U	U	U	U	N	N	N	N	N	N	N	C8	C7	C6	C5	C4	C3	C2	C1	%Q 10	%Q 09	C15	C14	C13	N	N	N	%I 01

U = Nepoužito, N = Prázdný, Předpona C indikuje číslo kanálu na nakonfigurovaném vstupním modulu (například C0 = vstupní bod 1, C15 = vstupní bod 16)

Příklad obsah vzorků

Tabulka 12-2 uvádí přehled hodnot obsažených v jednotlivých vzorcích podle popisů kanálů v řídicím bloku vzorku. Přehled vychází ze vzorové obrazovky zachycení ukázané na následující stránce. Všimněte si v tomto příkladu, že bity 1 – 16 se nacházejí v %R00207 a bity 17 – 24 jsou součástí %R00208.

Tabulka 12-2. Obsah vzorků pro příklad SER

Číslo kanálu	Obsah kanálu	Hodnota
1	%I 0001	1
2 - 4	Nuly	000
5	Bod 13 vstupního modulu	1
6	Bod 14 vstupního modulu	1
7	Bod 15 vstupního modulu	1
8	%Q0009	0
9	%Q0010	0
10 - 17	Body 1 - 8 vstupního modulu	100100010
18 - 24	Nuly	0000000

Blok dat pro vzorový řídicí blok

Tabulka 12-3 uvádí formát dat bloku, který je výsledkem vzorového řídicího bloku na stránce 12-19. 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 v zásobníku vzorků. 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.

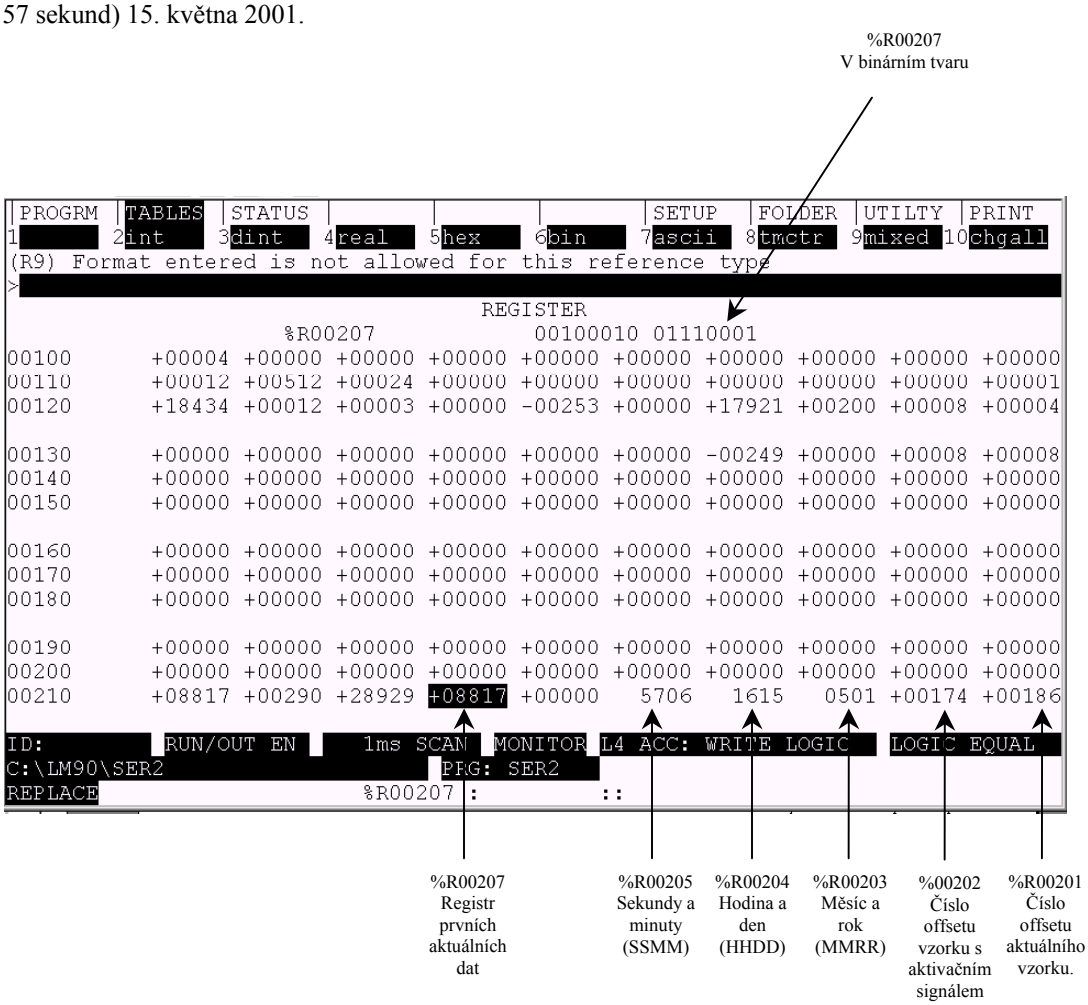
Příklad zachycení dat

Prohlížení zachycených dat

Následující výpis obrazovky byl sejmutý po aktivaci. Řídicí blok začíná na %R00100 a oblast dat začíná na %R00201. Kurzor je umístěný na %R00207, jak je poznamenáno blízko horní a dolní části obrazovky.

%R00207 je první registr v datovém bloku, který skutečně obsahuje naměřená vstupní data. Všimněte si, že celočíselná hodnota (-21855) má v tomto kontextu malý význam; když však umístíte kurzor na %R00207, její hodnota se zobrazí v binárním tvaru blízko horního okraje obrazovky. Když použijete tento binární formát, můžete určit stavy bitů nakonfigurovaných v části Popis kanálů řídicího bloku.

Registry %R00203 až %R00205 udávají čas a datum ve 24-hodinovém formátu, například 16:06 (a 57 sekund) 15. května 2001.



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. Všimněte si, že v příčkách za značkou END vstupy budou přecházet do jedničky a do nuly, ale výstupy se aktualizovat nebudou. I když to je normální stav, to se bude jevit jako problém, pokud nebude zřejmé, že značka END předchází před těmito příčkami.

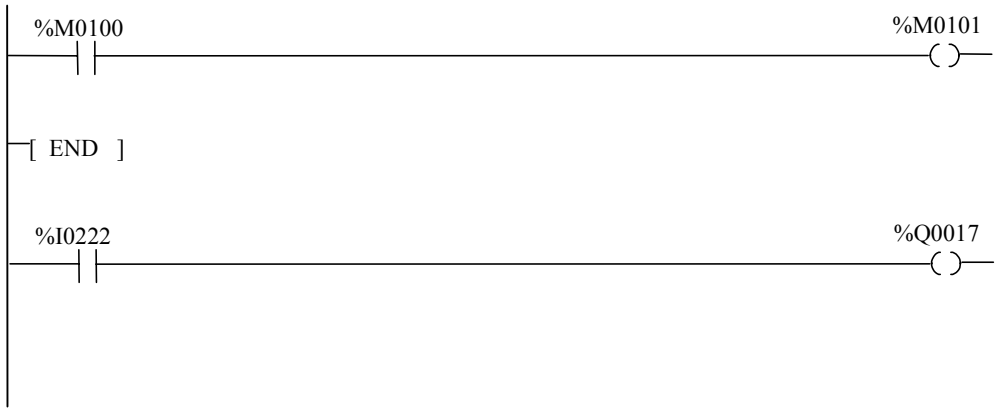
Funkce END je vhodná pro účely ladění, protože umožní izolovat sekci logiky. Provádí to tak, že zabrání, aby se vykonala logika, která následuje.

Programovací software Logicmaster implicitně zařadí značku [END OF PROGRAM LOGIC] za poslední příčku logiky jako indikaci konce vykonávání programu. Tato značka se používá, když se do logiky nenaprogramuje žádná funkce END.

```
—[ END ]
```

Příklad

V následujícím příkladu se příčka obsahující kontakt %I0222 a cívku %Q0017 a všechny příčky za ní nevykonají z důvodu přítomnosti instrukce END.



Poznámka

Umístěním funkce END do logiky SFC nebo logiky, kterou logika 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

Přehled MCR a MCRN

Funkce Hlavní řídicí relé (MCR/MCRN) se musí použít s odpovídající funkcí Konec hlavního řídicího relé (ENDMCR/ENDMCRN). Obě funkce musí mít stejný název. MCR/MCRN musí mít mezi funkcí a napájecí spojnicí kontakt povolení. Všechny příčky mezi povolenou funkcí MCR a odpovídající funkcí ENDMCR/ENDMCRN se vykonají, aniž by do cívek tekl proud. Funkce ENDMCR/ENDMCRN související s MCR/MCRN bude mít za následek, že se obnoví vykonávání normálního programu. Na rozdíl od instrukce JUMP se funkce MCR/MCRN mohou vyskytnout pouze v dopředném směru. Instrukce ENDMCR/MCRN se musí v programu objevit později, než jeho odpovídající instrukce MCR/MCRN.

Pro logiku řízenou pomocí povolené MCR/MCRN platí následující pravidla:

- Časovače neprovádějí inkrementaci nebo dekrementaci. Každý typ časovače TMR se resetuje (časovač se nastaví na nulu). V případě časovače ONDTR střádač "zamrzne" na hodnotě, která byla aktuální, když MCR/MCRN bylo povoleno.
- Proud se neobjeví pro žádnou instrukci. Normální výstupy jsou v nule; negované výstupy jsou v jedničce.
- Instrukce neprovedou aktualizaci svých výstupů. Například instrukce ADD nevytvoří aktuální součet na výstupu Q svého registru, instrukce Přesunutí nezkopíruje svou aktuální vstupní hodnotu na svůj výstup a instrukce Posunutí registru neposune data, atd. Hodnoty v těchto výstupních registrech zamrznou na hodnotách, které existovaly, když MCR/MCRN bylo povoleno.

Poznámka

Když skrz MCR/MCRN bude procházet proud, logika, kterou řídí, se vyhodnotí a zobrazí se stav kontaktů, ale žádný výstup nepřejde do jedničky. Pokud nebudete mít na paměti, že MCR/MCRN řídí sledovanou logiku, může se vám zdát, že je v chybovém stavu. Aby se indikovalo, že MCR/MCRN řídí žebříkovou logiku, software Logicmaster bude na obrazovce žebříkové logiky zobrazovat dvojitou napájecí lištu. Tato dvojitá napájecí spojnice se objeví bez ohledu na to, jestli MCR/MCRN bude povoleno nebo ne.

Software Logicmaster 90-30/20/Micro podporuje dvě formy funkce Hlavního řídicího relé, nevnořovanou (MCR) a novější vnořovanou formu (MCRN).

Kompatibilita CPU

Typ CPU	Podporovaný tvar
CPU311 – CPU341, verze 1	Používejte pouze nevnořovanou formu (MCR)
CPU311 – CPU341, verze 2 a pozdější	Používejte pouze vnořovanou formu (MCRN)
CPU řady 35x, 36x a 37x	Používejte pouze vnořovanou formu (MCRN)

Možné problémy s kompatibilitou MCRN

Když budete převádět program CPU340 nebo CPU341 ke spuštění na CPU řady 35x/36x/37x, Logicmaster 90 může zobrazit chybu "Funkce není podporována" ("Překročení úrovně vnořování"). K tomu dojde, když převáděný program bude uložený do CPU 35x/36x/37x a v původním programu bylo použito více než osm úrovní vnoření MCRN.

Instrukce MCRN jsou ve skutečnosti instrukce funkčních bloků v CPU340/341, což znamená, že se vykonávají ve firmwaru CPU a nevykonávají se ve vestavěném Booleovském koprocesoru (BCP). Limit pro vnořování funkčních bloků byl nastavený na 256. Tento limit představuje mnohem více úrovní, než kolik jich můžete prakticky potřebovat. Když byl provedený návrh s CPU řady 35x/36x/37x, instrukce MCRN se přesunou do BCP, aby se tím zlepšil výkon CPU (instrukce funkčního bloku se vykonávají pomaleji než jejich odpovídající instrukce v BCP). Byl provedený kompromis mezi úrovněmi vnořování a výkonem a BCP3 používaným v CPU 35x/36x/37x s tím, že se používá osm úrovní vnořování, což je normálně více, než uživatel potřebuje. Logicmaster 90 proto při převodu programu vynutí osm úrovní vnořování a pokud by se používalo více než osm úrovní, bude se generovat hlášení "Překročení úrovně vnořování".

Proto pokud v programu s CPU340/341 budete mít více než osm úrovní vnořování MCR, bude nutné ho upravit tak, aby mohl pracovat s CPU 35x/36x/37x. Místo toho můžete zvážit použití instrukce Jump.

Vnořování 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 nevnořovahého 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 12-28

Jednomu ENDMCRN může odpovídat několik funkcí MCRN (s výjimkou CPU řady 35x/36x/37x, jak je uvedeno níže). Každé MCRN i ENDMCRN musí mít stejný název. To je analogické vnořovanému JUMP, kde ke stejnému návěští LABEL může existovat více instrukcí JUMP. Porovnání funkce JUMP a funkce MCR najdete níže v odstavci "Rozdíly mezi MCR a skoky".

Poznámka

S CPU řady 35x, 36x a 37x používejte pouze jedno MCRN na každé ENDMCRN.

Operace MCR

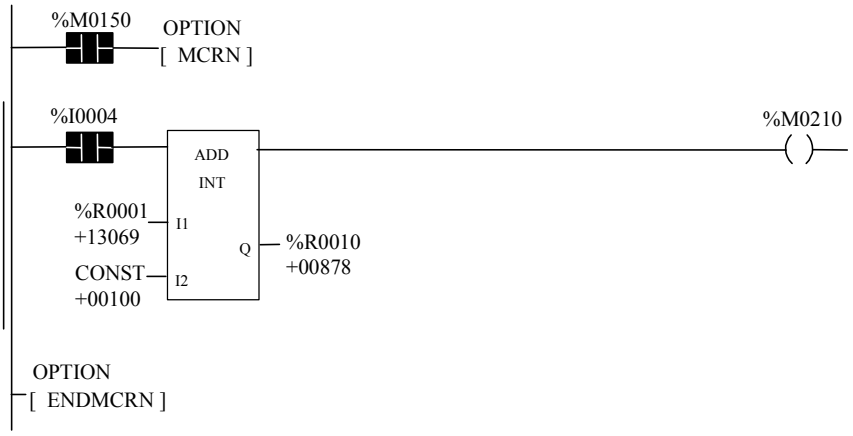
Na každou instrukci ENDMCR může být pouze jedna instrukce MCR. Nevnořované MCR a ENDMCR se nesmí překrývat ani nesmí obsahovat žádný jiný pár instrukcí MCR/ENDMCR nebo pár instrukcí JUMP/LABEL. Nevnořované MCR se nesmí vyskytovat v rozsahu žádného páru JUMP/LABEL.

Parametry

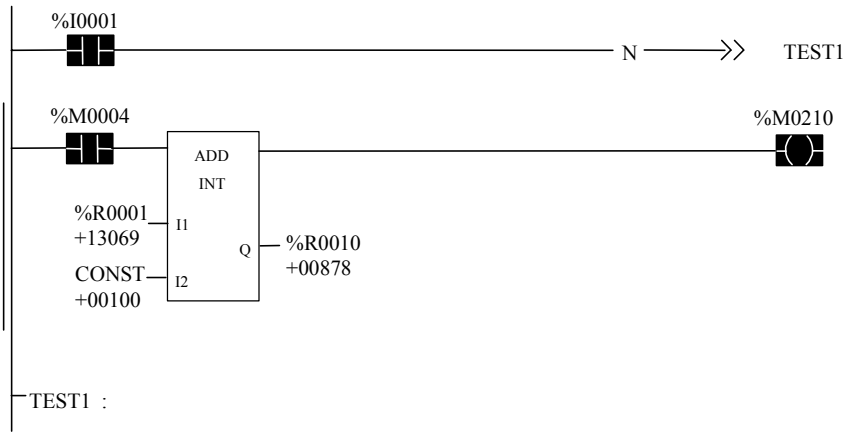
Obě formy funkce MCR mají stejné parametry. Obě mají povolení Booleovského vstupu EN a název, který identifikuje MCR. Tento název se používá opět s instrukcí ENDMCR. Ani funkce MCR ani MCRN nemají žádný výstup; za MCR v příčce nesmí být nic.

Rozdíly mezi MCR/MCRN a JUMP

U funkce MCR se funkční bloky uvnitř MCRN vykonají ***bez průtoku proud*** a cívky ***nejsou nabuzené***. V následujícím příkladu bude MCRN povoleno, když %M0150 bude ve stavu ON. Když bude MCRN povoleno, i když %I004 bude ve stavu ON, funkční blok ADD se vykoná ***bez*** průtoku proudu (tj. nepřipočítá se 100 k %R0001) a skrz %M0210 nebude protékat proud. Stav kontaktů, například %I0004, a hodnoty v registrech používané na vstupech, například %R0001, se aktualizují na obrazovce Logicmaster, ale registry na výstupech, které řídí MCRN, například %R0010, při povolení MCRN zamrznou na původních hodnotách.



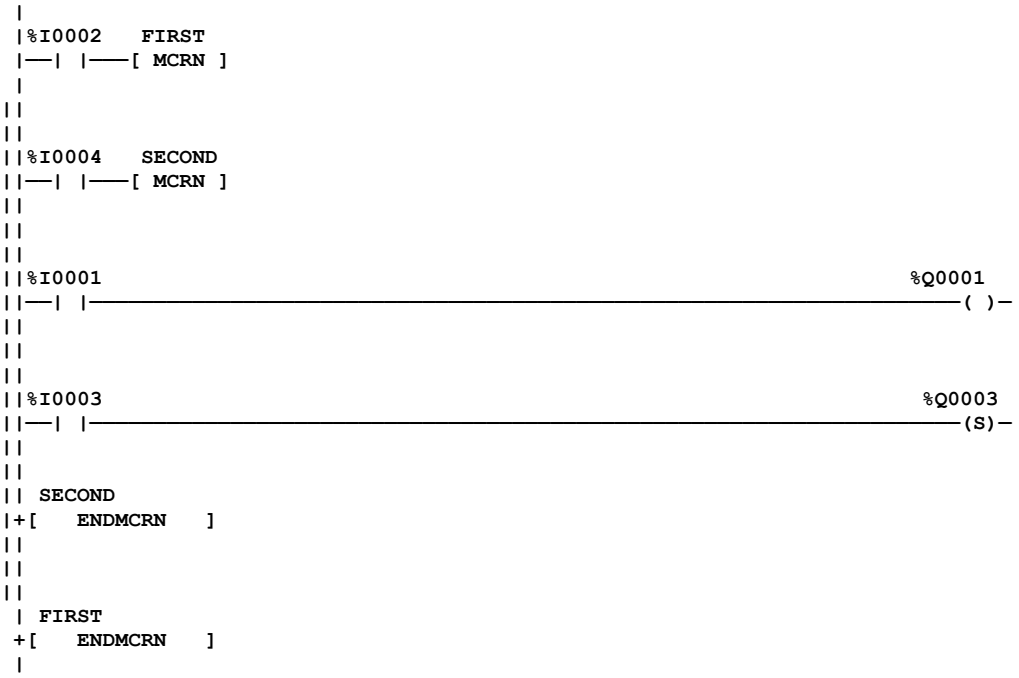
U funkce JUMP se mezi JUMP a LABEL ***nevyhodnotí*** žádné funkční bloky a cívky ***zůstanou beze změny***. V následujícím příkladu, když %I0001 bude ve stavu ON, bude povolený JUMP s názvem TEST1. Protože logika mezi JUMP a LABEL se přeskočí, %M0210 zůstane beze změny (tj., pokud bylo ve stavu ON, zůstane ve stavu ON; pokud bylo ve stavu OFF, zůstane ve stavu OFF). Stav kontaktů, například %M0004, a hodnoty v registrech používané na vstupech, například %R0001, se aktualizují na obrazovce Logicmaster, ale registry na výstupech, které řídí JUMP, například %R0010, při povolení JUMP zamrznou na původních hodnotách.



Příklad 1

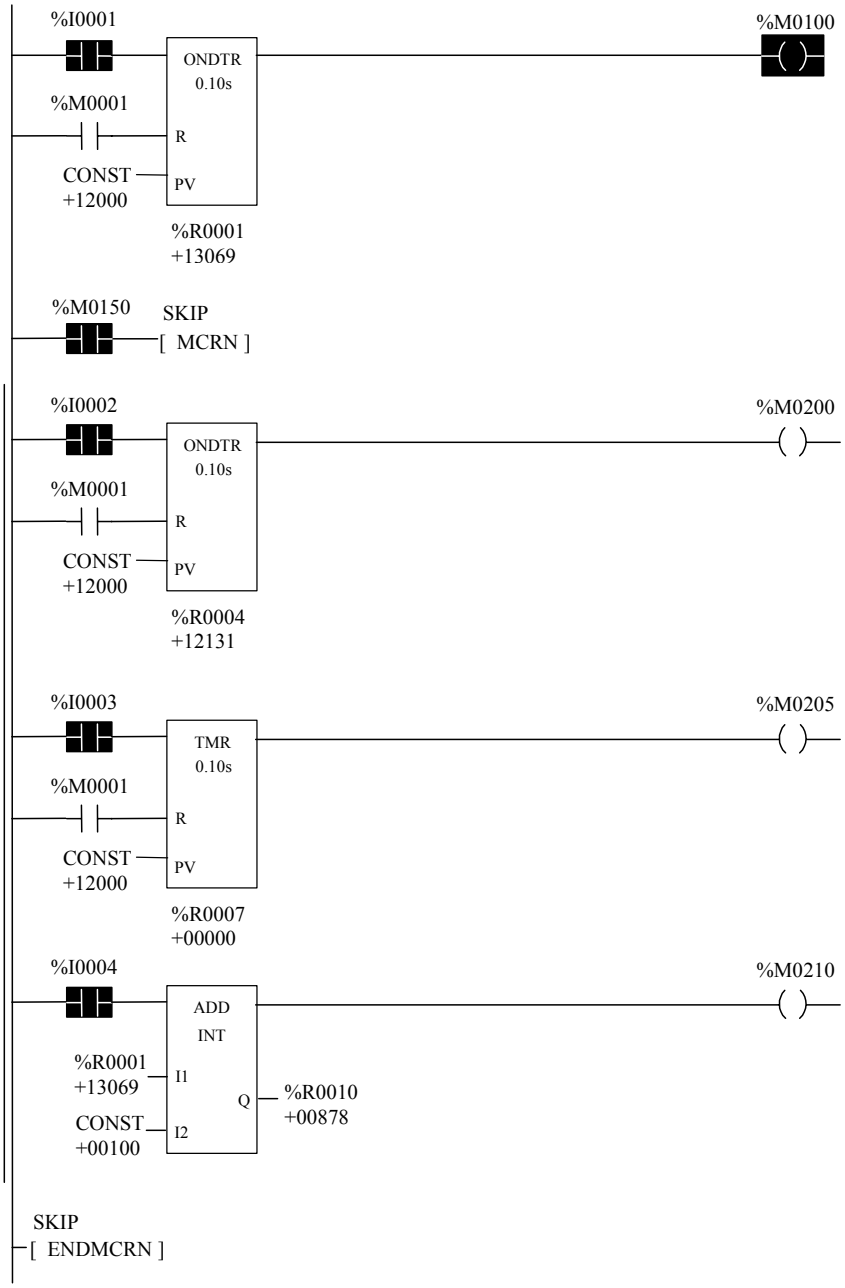
Následující příklad ukazuje MCRN s názvem "Second" vnořený uvnitř MCRN s názvem "First". Vždy když %I0002 umožní průtok proudu do funkce MCRN, vykonávání programu bude pokračovat bez průtoku proudu do cívek, dokud se nenarazí na odpovídající ENDMCRN. Pokud %I0001 a %I0003 budou ve stavu ON, %Q0001 přejde do stavu OFF a %Q0003 zůstane ve stavu ON.

K usnadnění lokalizace chyb žebříkových programů dvojitá napájecí lišta identifikuje logiku, která je v rozsahu řízení MCR.



Příklad 2

V následujícím příkladu první příčka funguje normálně. Avšak MCRN s názvem SKIP řídí zbytek příček, které pro indikování tohoto mají dvojitou napájecí sběrnici. V první příčce, kterou řídí MCRN, nakumulovaná hodnota časovače ONDTR (%R0004) zamrzne a i když dosáhne předvolené hodnoty, jeho výstup (%M0200) nebude nabuzený. V následující příčce TMR se resetovalo pomocí MCRN. Jeho nakumulovaná hodnota (%R0007) je držena na nule a jeho výstup (%M02025) nebude nabuzený. V další příčce výstup instrukce ADD zamrzne (její výstup na %R0010 není součtem jejích vstupů) a její proudová cívka (%M0210) nebude nabuzená. Všimněte si však, že stav kontaktů vstupních registrů (například %R0001 na vstupu I1 instrukce ADD) se aktualizuje na obrazovce v rámci oblasti řízení MCRN.

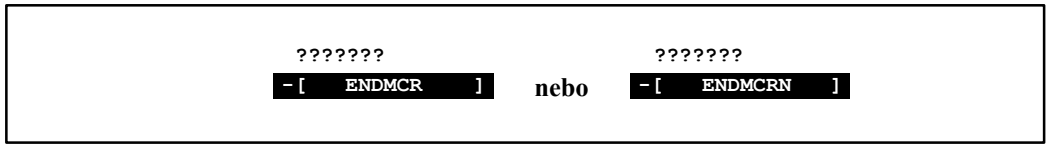


ENDMCRN/ENDMCR

Funkce Konec hlavního řídicího relé (End Master Control Relay – ENDMCR/ENDMCRN) se používá k obnovení normálního vykonávání programu po funkci MCR/MCRN. 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. Nevnořovaná forma, ENDMCR, se musí použít s vnořovanou funkcí MCR, tedy MCRN.

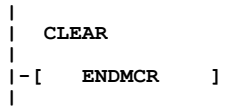
Funkce ENDMCR má negovaný Booleovský vstup EN. Povolení instrukce musí zajistit napájecí lišta; vykonávání nesmí být podmíněné. Funkce ENDMCR má také název, který ENDMCR identifikuje a spojuje jí s odpovídajícími MCR. Funkce ENDMCR nemá žádné výstupy; před ani za instrukcí ENDMCR v příčce nesmí nic být.



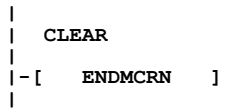
Příklad

V následujících příkladech je naprogramovaná instrukce ENDMCR k ukončení rozsahu MCR s názvem “CLEAR“.

Příklad nevnořeného ENDMCR



Příklad vnořeného ENDMCR



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 firmwaru od verze 1 pro CPU modely CPU311-CPU341 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.

Nevnořovaný 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 JUMP/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/36x/37x podporují pouze vnořované skoky.

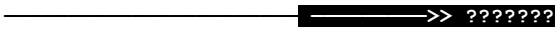
Vnořovaná forma instrukce JUMP má tvar ———N——>>LABEL01, kde LABEL01 je název pro JUMP a jeho odpovídající vnořenou instrukci LABEL. Vnořovaný JUMP je možno použít u softwaru Logicmaster 90-30/20/Micro a firmwaru PLC verze 2 a pozdější.

Vnořovanou instrukci JUMP je možno do programu umístit kamkoliv za předpokladu, že se nebude vyskytovat uvnitř některé nevnořené instrukce MCR nebo nevnořené instrukce JUMP.

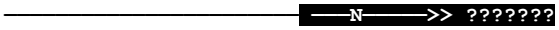
K jednomu vnořenému návěští LABEL může existovat několik vnořených instrukcí JUMP. Vnořené instrukce JUMP mohou být skoky směrem dopředu nebo dozadu.

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:

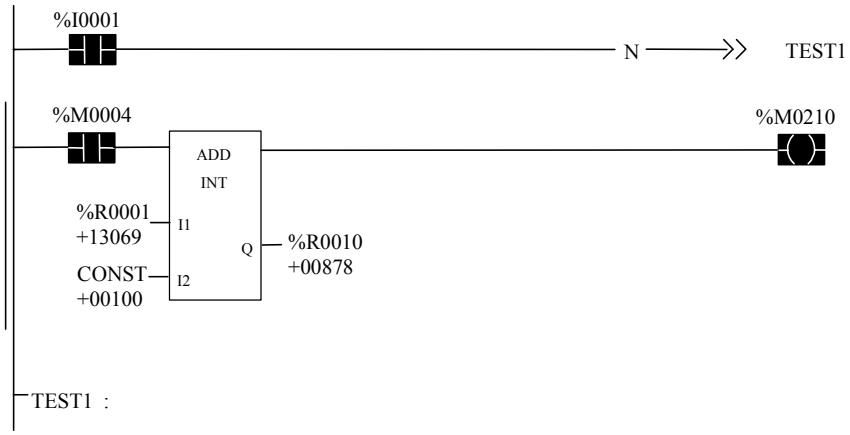


Upozornění

Aby u instrukcí JUMP směrem dozadu nedošlo k vytvoření nekonečné smyčky, JUMP směrem dozadu musí obsahovat způsob, jak do instrukce zařadit podmínku.

Příklady

Když se v následujícím příkladu sepne kontakt %I0001, instrukce JUMP s názvem TEST1 bude povolená a proud přeskočí dopředu na TEST1 LABEL. Protože logika mezi JUMP a LABEL se přeskočí, %M0210 zůstane beze změny (tj., pokud bylo ve stavu ON, zůstane ve stavu ON; pokud bylo ve stavu OFF, zůstane ve stavu OFF). Stav kontaktů, například %M0004, a hodnoty v registrech používané na vstupech, například %R0001, se aktualizují na obrazovce Logicmaster, ale registry na výstupech, které řídí JUMP, například %R0010, při povolení JMP zamrznou na původních hodnotách. Všimněte si použití dvojité napájecí sběrnice v části logiky umístěné mezi JUMP a jeho návěštím LABEL.



LABEL

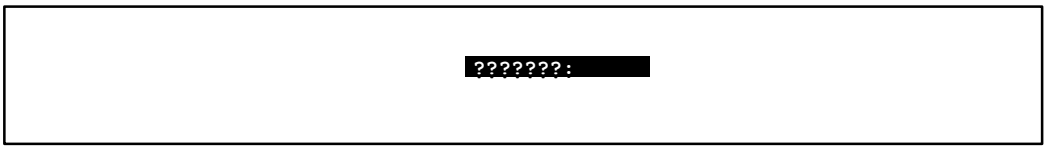
Instrukce LABEL funguje jako cílové místo pro JUMP. Instrukce LABEL se používá k obnovení normálního vykonávání programu po instrukci JUMP.

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.

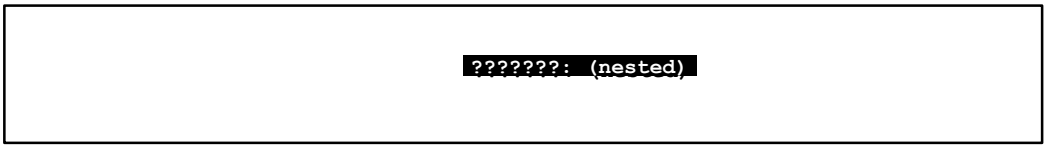
Software Logicmaster 90-30/20/Micro podporuje dvě formy funkce LABEL, nevnořovanou a vnořovanou formu. Například nevnořovaná forma, LABEL01:, se musí použít s nevnořovanou funkcí JUMP, ------>>LABEL01; vnořovaná forma, LABEL01: (vnoření), se musí použít s vnořovanou funkcí JUMP, ———N——>>LABEL01.

Instrukce LABEL nemá žádné vstupy a žádné výstupy. Také před ani za instrukcí LABEL v příčce nesmí nic být.

Nevnořované LABEL:

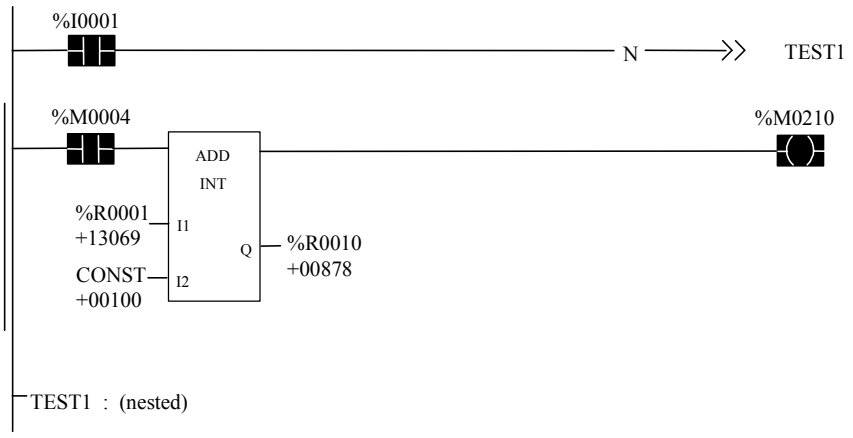


Vnořované LABEL:



Příklad

Když v následujícím příkladu bude povolený JUMP TEST1, čtení přeskočí dopředu na TEST1: (vnořený) LABEL, což znamená, že příčka mezi JUMP a LABEL se nebude číst.



POZNÁMKA

Poznámky jsou užitečné, když do žebříkového programu potřebujete přidat vysvětlivky, poznámky, informace k úrovni revize, atd. Používání poznámek se velmi doporučuje, protože poskytují cenné informace pro ty, kdo v budoucnu budou provádět lokalizaci závad nebo aktualizaci systému. Protože také lidská paměť je nedokonalá, poznámky jsou cenné odkazy i pro tvůrce žebříkového programu.

Poznámka

Při ukládání paměti PLC se anotace (poznámky, přezdívky a popisy) do PLC nezapisují. Chcete-li si proto tyto anotace prohlížet, musíte mít na počítači kopii adresáře s původním programem (který obsahuje anotace). Když pak připojíte svůj počítač k PLC, programovací software automaticky vytvoří propojení k těmto anotacím.

Vytvoření standardní poznámky

Poznámka se může skládat až z 2048 znaků textu. V Logicmaster je v žebříkové logice představována následovně:

```
(*  COMMENT  *)
```

Vytvoření poznámky

1. Vytvořte novou příčku. Příčka COMMENT nemůže mít jinou logiku kromě instrukce COMMENT.
2. Vložte POZNÁMKU, kterou najdete v řídicí skupině instrukcí.
3. Příčku přijměte stisknutím tlačítka Escape.
4. Přemístěte kurzor na právě vytvořenou instrukci (* COMMENT *) a stisknutím tlačítka Zoom (F10) otevřete obrazovku pro editování poznámky.
5. Zapište text své poznámky. Všimněte si, že řádky se v editoru poznámek automaticky nepřetačejí. Aby se text začal psát na dalším řádku, na konci řádku je nutno stisknout tlačítko Enter.
6. Když budete hotovi, stisknutím tlačítka Escape ukončete práci v editoru poznámek a poznámka se uloží.

Po vytvoření je možno text POZNÁMKY číst nebo editovat posunutím kurzoru na (* COMMENT *) a zvolením Zoom (F10). Poznámky k příčkám je také možno vytisknout z menu Print softwaru Logicmaster.

Vytvoření dlouhé poznámky pro použití ve výtisku Logicmaster

V softwaru Logicmaster je možno dlouhý text zařadit do výtisku použitím souboru s anotačním textem:

1. Vytvořte poznámku (podrobnosti k vytvoření poznámky viz předchozí odstavec):
   A. Zapište poznámku do místa, kde má začínat text z jiného souboru.
   B. Na novém řádku zapište **\I** nebo **\i**, písmeno označující diskovou jednotku následované dvojtečkou, zpětné lomítko, podadresář nebo adresář, zpětné lomítko 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. Stisknutím tlačítka Escape ukončete práci v editoru poznámek a text poznámky se uloží.
2. Otevřete textový procesor a vytvořte textový soubor.
3. Uložte textový soubor ve formátu .txt a souboru dejte název, který jste zadali v poznámce, a uložte ho na jednotku a do cesty zadané v poznámce.

SVCREQ

Instrukce Service Request je instrukce pro všeobecné použití, která může vykonávat široký rozsah zvláštních instrukcí (služeb), které nejsou k dispozici jako jednotlivé funkční bloky. Funkce Service Request (SVCREQ) se používá k vyžádání některé z následujících speciálních služeb PLC:

Tabulka 12-4. Funkce Service Request

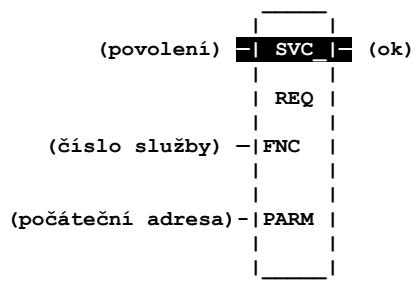
Funkce	Popis
1	Změna/čtení časovače konstantního cyklu
2	Čtení hodnot okna
3	Změna režimu okna komunikace programovacího zařízení a hodnoty časovače
4	Změna režimu okna komunikace systému a hodnoty časovače
6	Změna/čtení stavu kontrolního součtu úlohy a počtu slov pro kontrolní součet
7	Změna/čtení hodin 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
24	Reset inteligentního modulu
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
48	Restartování po automatickém resetu fatální chyby
49	Statistika automatického resetu

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	Každý typ Service Request má jedinečné číslo funkce, které se musí naprogramovat na vstupu FNC. FNC může obsahovat buď konstantu nebo odkaz na adresu, která obsahuje číslo funkce požadované služby.
PARM	PARM obsahuje počáteční adresu bloku parametrů pro požadovanou službu.
ok	Výstup ok bude v jedničce, když se funkce vykoná bez chyby.

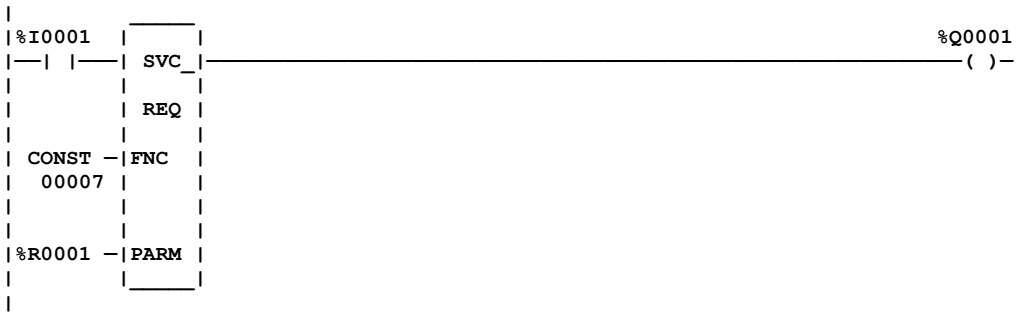
Platné typy paměti

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, vykoná se funkce SVCREQ s číslem 7 zadaným na vstupu FNC. Blok parametrů funkce začíná na %R0001 (zadáno v PARM). 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 k 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.

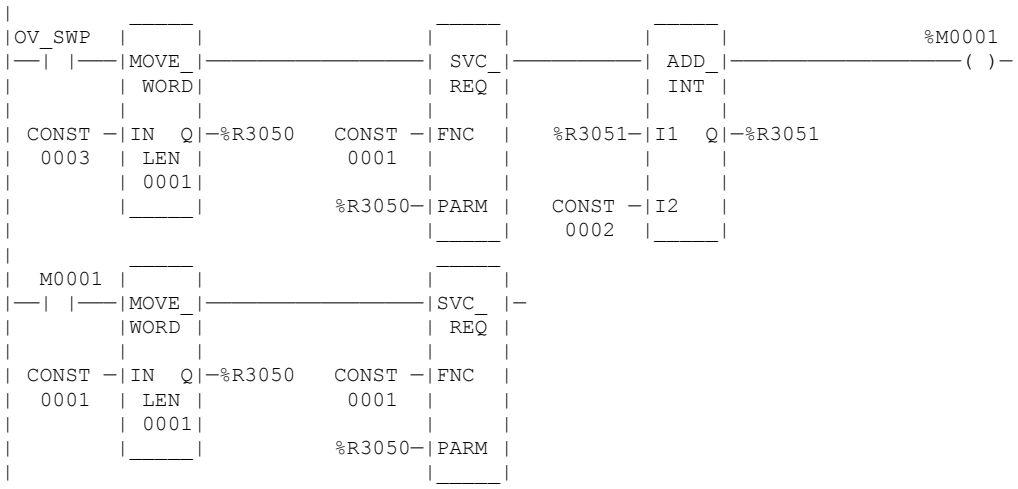
Po vykonání funkce předá stav časovače a hodnotu na stejných adresách bloků parametrů:

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 a okno komunikace systému.

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 komunikační okno systému. 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 dvěma 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. :

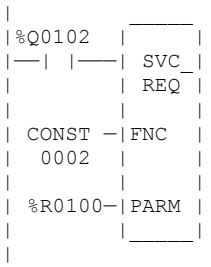
	Horní bajt	Dolní bajt	
Okno programovacího zařízení	Režim	Hodnota v ms	adresa
Okno komunikace systému	Režim	Hodnota v ms	adresa + 1
Vyhrazeno*	*Viz poznámka	*Viz poznámka	adresa + 2

***** Poznámka. Slovo adresa + 2 je vyhrazeno pro použití systémem. Sem se zapíšou samé nuly.

Všechny parametry jsou výstupní parametry. K naprogramování této funkce není nutno do bloku parametrů hodnoty zapsat. Výstupní hodnoty pro obě 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 %R0100. 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:

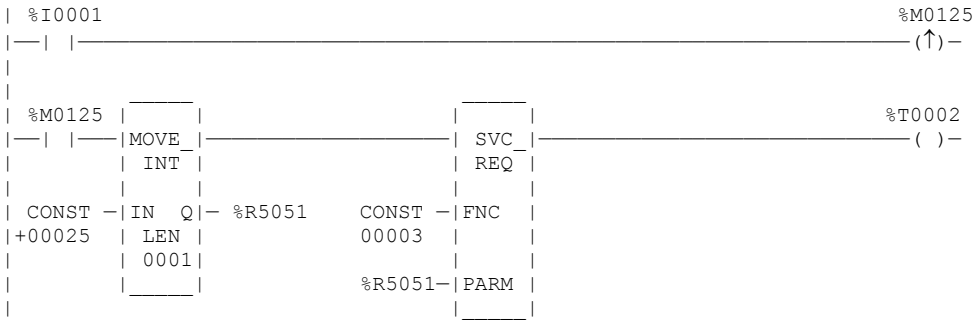
Horní bajt	Dolní bajt	
0	0	adresa

Chcete-li okno programovacího zařízení povolit, zapište funkci SVCREQ #3 s tímto blokem parametrů:

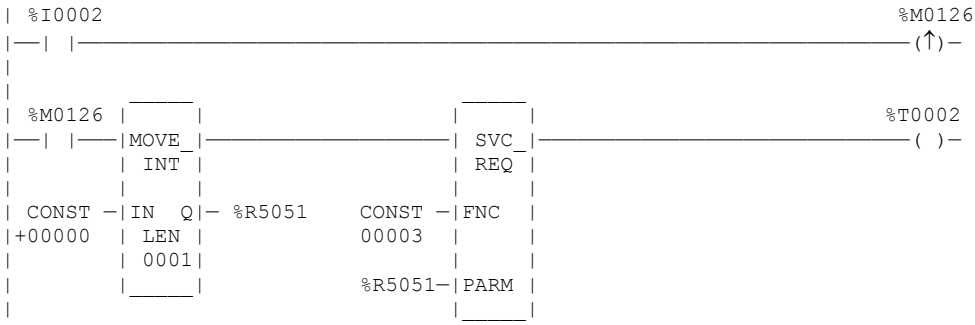
Horní bajt	Dolní bajt	
Režim	Hodnota od 1 do 255 ms	adresa

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:

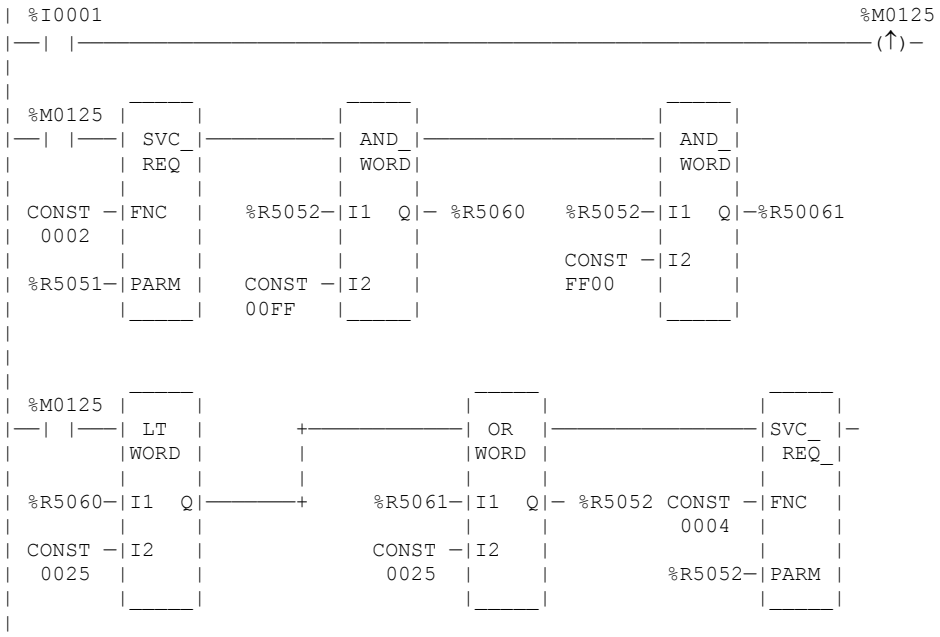
Horní bajt	Dolní bajt	
0	0	adresa

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 (0 - 32)	adresa + 1

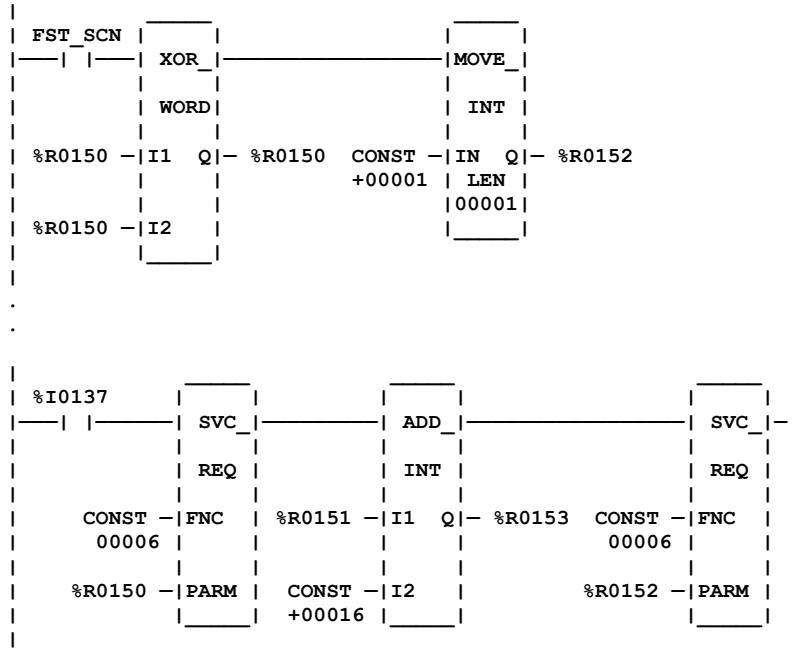
Zapsání 1 bude mít za následek, že PLC provede úpravu počtu slov, se kterými se má provést kontrolní součet, na hodnotu uvedenou v druhém slově bloku parametrů. Pro každé CPU Series 90-30 hodnota druhého slova bude v rozmezí od 0 do 32. Pokud hodnota bude mimo rozsah, bude se generovat chyba. V případě CPU211 Series 90-20 hodnota může být 0 nebo 4.

Poznámka

Tento Service Request nelze použít u PLC 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 s 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 28-bodovým CPU PLC Series 90 Micro (to je, IC693UDR005, IC693UAA007 a IC693UDR010) a s 23-bodovým CPU PLC Series 90 Micro (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.
4. Nezapíše se neplatné datum, například 02/29/01, což nesprávně definuje přechodný rok v roce 2001 (2001 není přechodný rok).

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
```

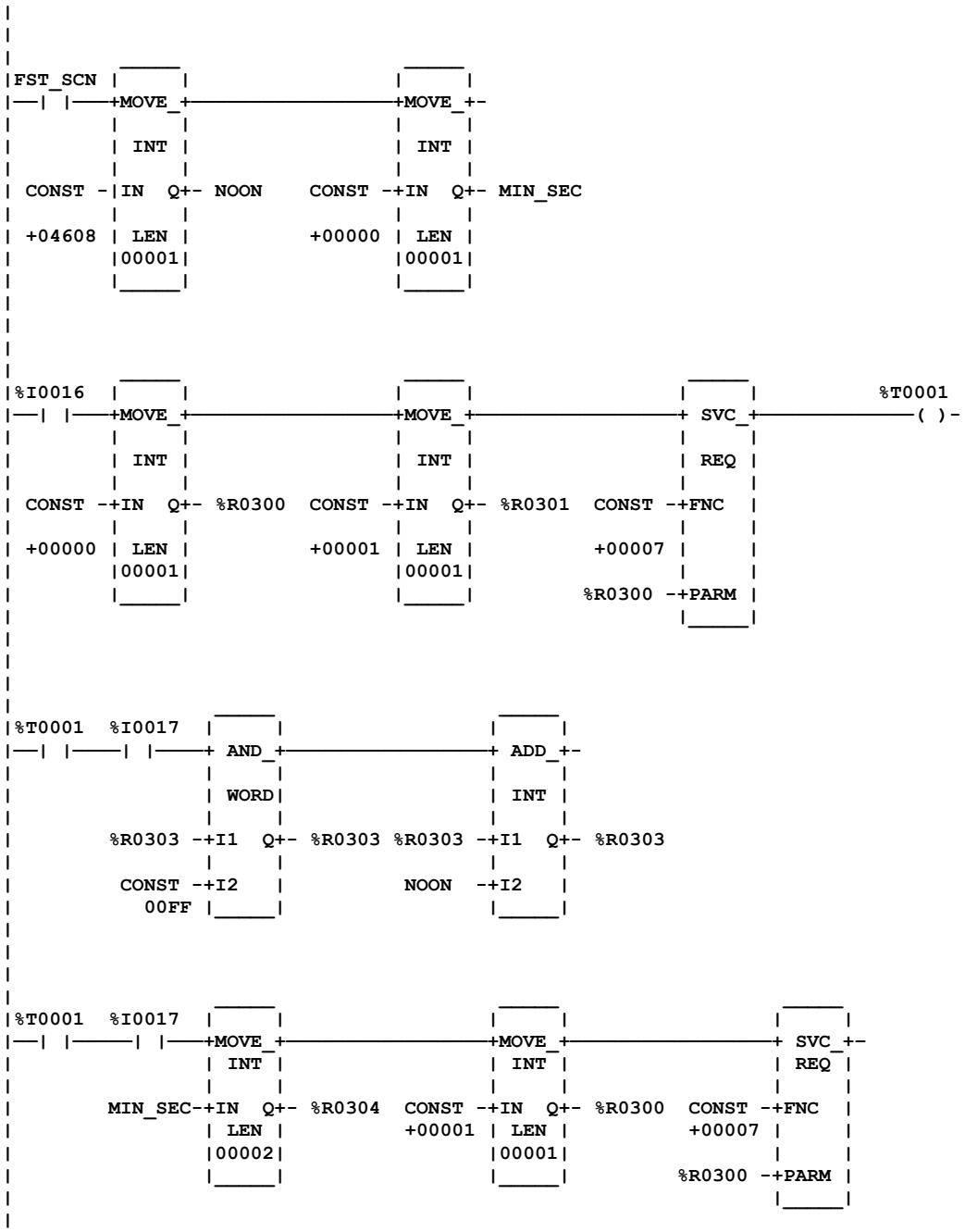
Ve slově 2 zadejte formát dat:

```
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	
1 = změna	nebo	0 = čtení	adresa
1			adresa + 1
měsíc		rok	adresa + 2
hodiny		den v měsíci	adresa + 3
sekundy		minuty	adresa + 4
(prázdný)		den v týdnu	adresa + 5

Příklad výstupního bloku parametrů:
Načtení datumu a času v BCD formátu
(Neděle, 3. června, 1988 v 2:45:30 odpoledne)

0	
1	
07	88
14	03
30	45
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).

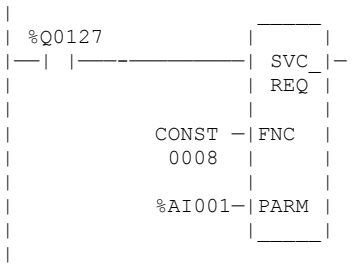
Upozornění

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

Když v následujícím příkladu %Q0127 přejde do stavu ON, hlídací časovač se resetuje.



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

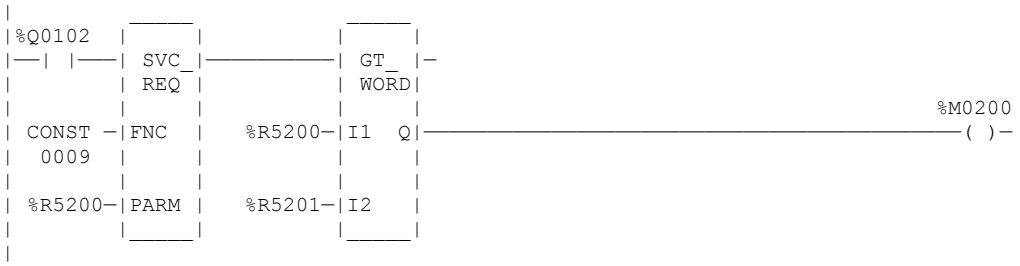
Ze všech CPU probíraných v tomto manuálu se Service Request 9 podporuje pouze u CPU 90-30 počínaje verzí 8.0.

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

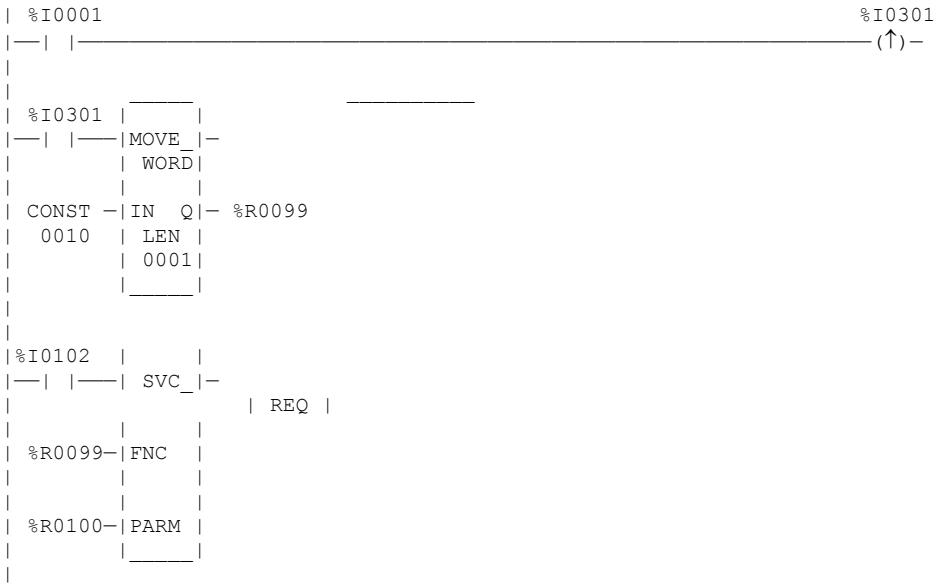
Ze všech CPU probíraných v tomto manuálu se Service Request 10 podporuje *pouze* u CPU 90-30 počínaje verzí 8.0.

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

Když v následujícím příkladu kontakt povolení %I0301 přejde do stavu ON, adresa registru %R0099 se načte s hodnotou 10, což je kód pro funkci Čtení názvu programu. Když v následující příčce %I0102 přejde do stavu ON, Service Request přečte název adresáře a uloží ho ve čtyřslovním bloku paměti počínaje adresou %R0100 (zadáno v PARM).



SVCREQ #11: Čtení PLC ID

Funkce SVCREQ #11 se používá k načtení názvu PLC Series 90 vykonávajícího program.

Poznámka

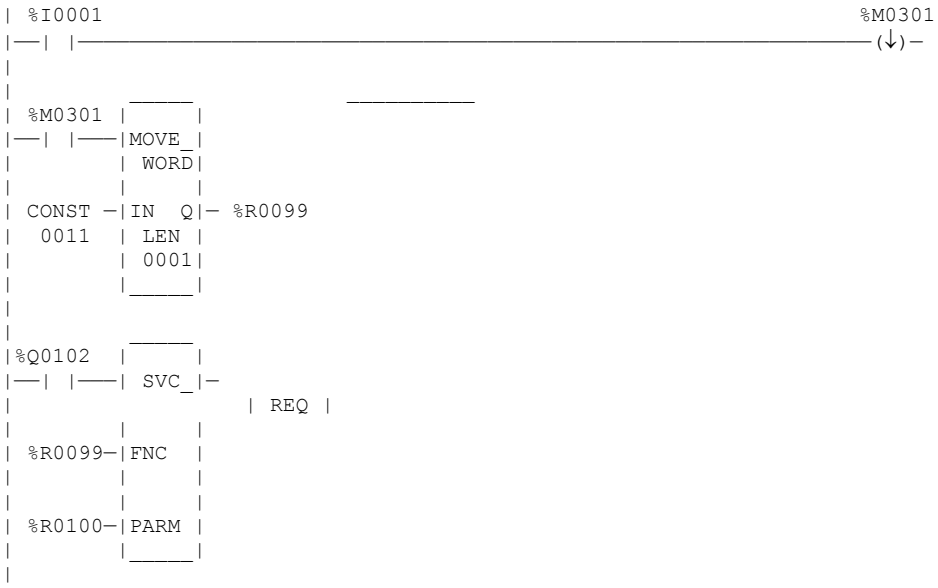
Ze všech CPU probíraných v tomto manuálu se Service Request 11 podporuje *pouze* u CPU 90-30 počínaje verzí 8.0.

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

Když v následujícím příkladu kontakt 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. Když v následující příčce %Q0102 přejde do stavu ON, Service Request přečte PLC ID a uloží ho ve čtyřslovním bloku paměti počínaje adresou %R0100 (zadáno v PARM).



SVCREQ #12: Čtení stavu běhu PLC

Funkce SVCREQ #12 se používá k načtení aktuálního stavu chodu CPU PLC.

Poznámka

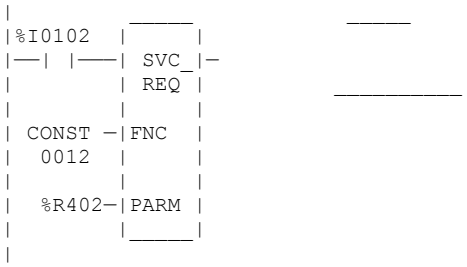
Ze všech CPU probíraných v tomto manuálu se Service Request 12 podporuje *pouze* u CPU 90-30 počínaje verzí 8.0.

Blok parametrů je pouze výstupní blok parametrů; má délku jednoho slova. Z vykonání této funkce Service Request je možno získat pouze dva platné výsledky:

1 = běh/zakázáno 2 = běh/povoleno	adresa

Příklad

Když v následujícím příkladu %I0102 přejde do stavu ON, Service Request přečte stav za běhu PLC a uloží ho do paměťové adresy %R402. Pokud PLC bude v režimu Běh/Zakázáno, %R402 bude obsahovat hodnotu 1. Pokud PLC bude v režimu Běh/Povoleno, %R402 bude obsahovat hodnotu 2.



SVCREQ #13: Zastavení PLC

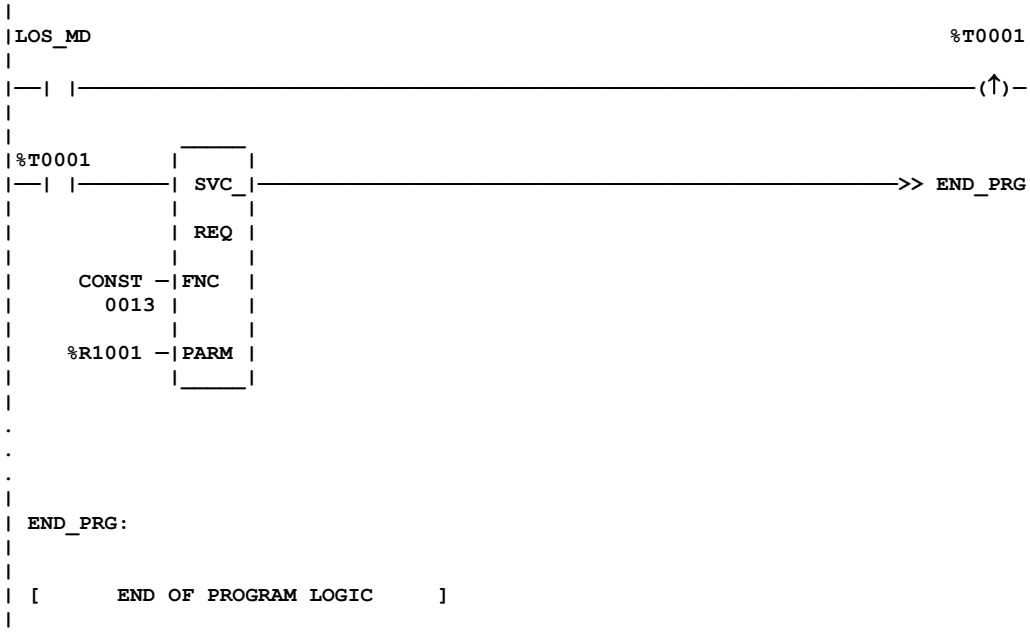
Funkce SVCREQ #13 se používá k zastavení PLC ***na konci následujícího cyklu***. Všechny výstupy přejdou do příslušných výchozích stavů na začátku následujícího cyklu PLC. Do tabulky chyb PLC se vloží informativní chyba a nevykoná se nic kromě funkčního bloku "SHUT DOWN PLC". Čtení I/O bude pokračovat 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. Jakmile na PLC přijde z funkce Service Request tento povel STOP, vykoná ještě jeden cyklus a pak se zastaví (viz poznámka níže).



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.

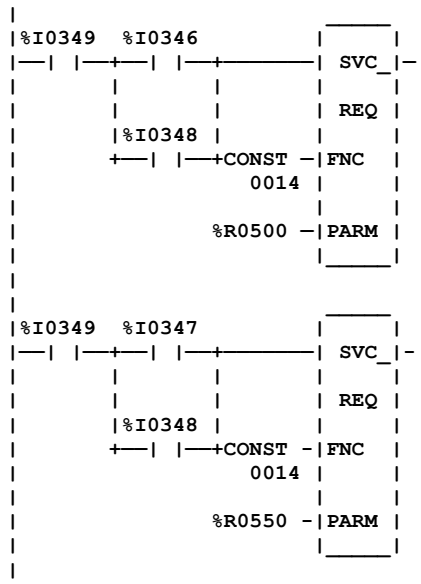
U této funkce blok parametrů má délku 1 slovo. Je to pouze blok vstupního parametru.

0 = smazat tabulku chyb PLC 1 = smazat tabulku chyb I/O	adresa

Příklad

Když v následujícím příkladu kontakty %I0346 a %I349 budou ve stavu ON, smaže se tabulka chyb PLC. Když kontakty %I0347 a %I0349 budou ve stavu ON, smaže se tabulka chyb I/O. Když kontakty %I0348 a %I0349 budou ve stavu ON, smažou se obě tabulky.

Blok parametrů pro chybovou tabulku PLC se nachází na adrese %R0500 a pro chybovou tabulku I/O se blok parametrů nachází na adrese %R0550. Oba bloky parametrů mohou být nastavené kdekoliv v programu (oba musí být v logické jedničce, aby se vynulovaly jejich příslušné tabulky).



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

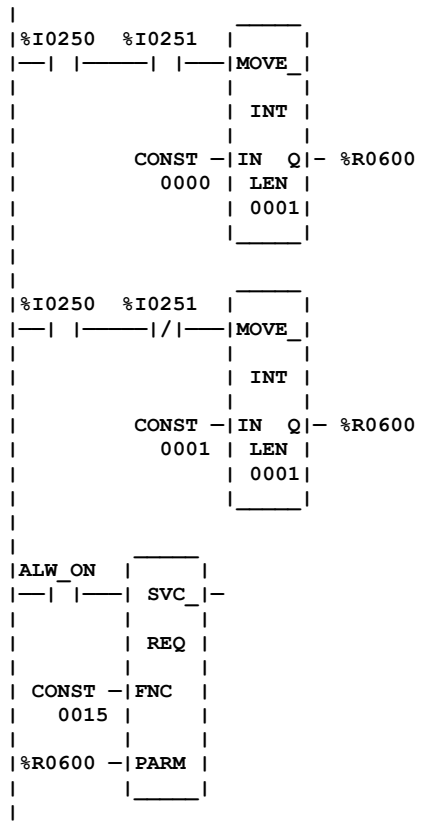
Dolní bajt	Horní bajt
1	
dlouhý/krátký	
adresa	
adresa chyby I/O	
skupina chyby a akce	
kategorie chyby	typ chyby
popis chyby	
specifická data chyby	
časová značka	

V první bajtu adresy + 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

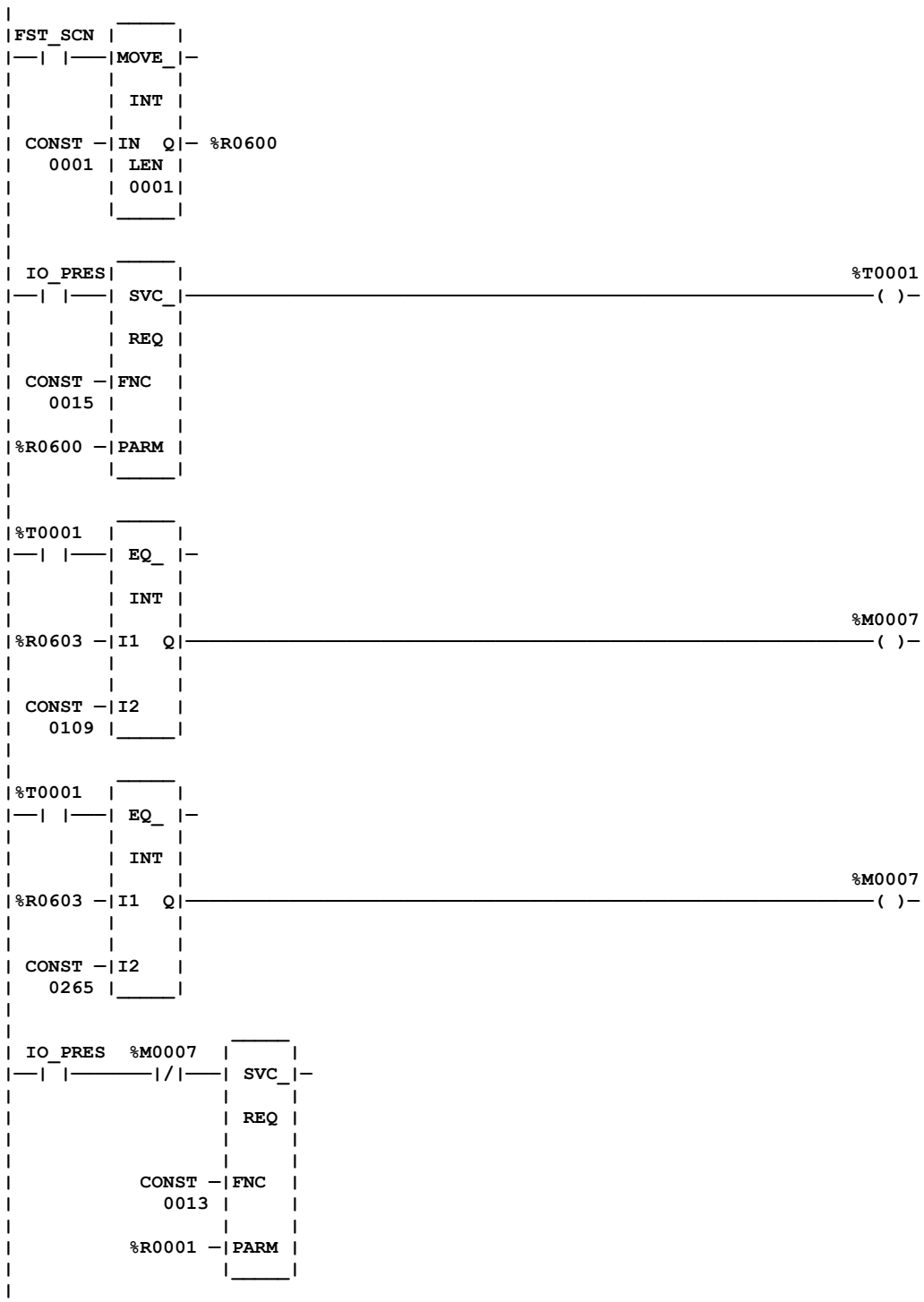
V následujícím příkladu PLC ukončí činnost, když se v modulu I/O vyskytne nějaká chyba s výjimkou, když se chyba vyskytne v modulech v sestavě 0, pozici 9 a v sestavě 1, pozici 9. Pokud se chyba vyskytne na těchto modulech, systém poběží dál. Parametr pro "typ tabulky" se nastaví při prvním cyklu. Když bude kontakt IO_PRES nastavený, indikuje to, že tabulka chyb I/O obsahuje záznam. CPU PLC nastaví normálně rozpojený kontakt v dalším 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 bloku parametrů slovo obsahovat sestavu a umístění pozice I/O modulu, 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 cívka %M0007 bude ve stavu ON, jeho normálně sepnutý kontakt se rozpojí a zabrání tak ukončení provozu. Obráceně, pokud cívka %M0007 bude ve stavu OFF, protože se chyba objevila na jiném modulu, její normálně sepnutý kontakt bude sepnutý a ukončení provozu se vykoná.


FST_SCN
MOVE_
INT
CONST
0001
IN
Q
%R0600
LEN
0001
IO_PRES
SVC_
REQ
CONST
0015
FNC
%R0600
PARM
%T0001
EQ_
INT
%R0603
I1
Q
CONST
0109
I2
%T0001
EQ_
INT
%R0603
I1
Q
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.

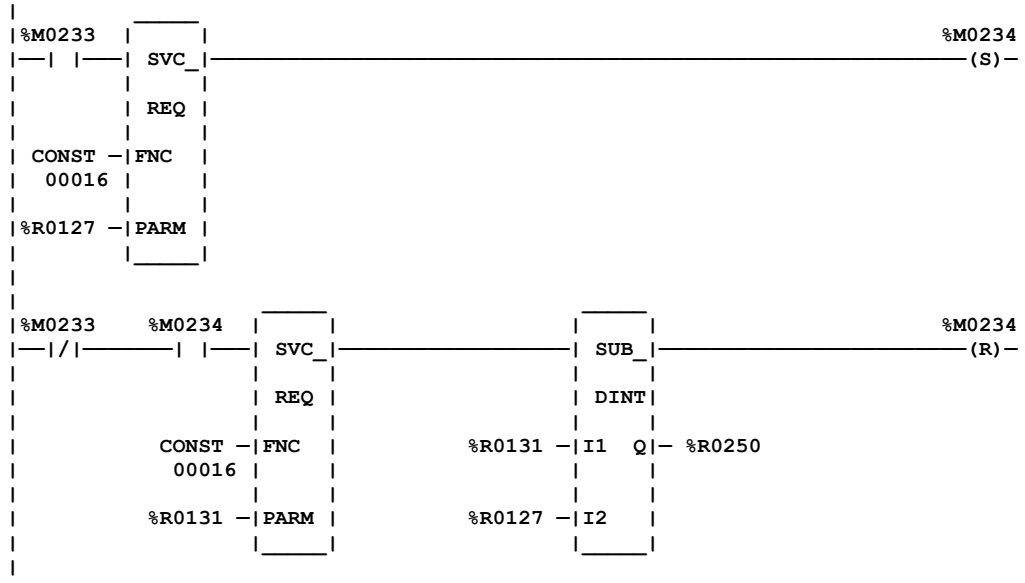
sekundy od zapnutí napájení (nižší řád)	adresa
sekundy od zapnutí napájení (vyšší řád)	adresa + 1
Časové intervaly 100 mikrosekund	adresa + 2

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.

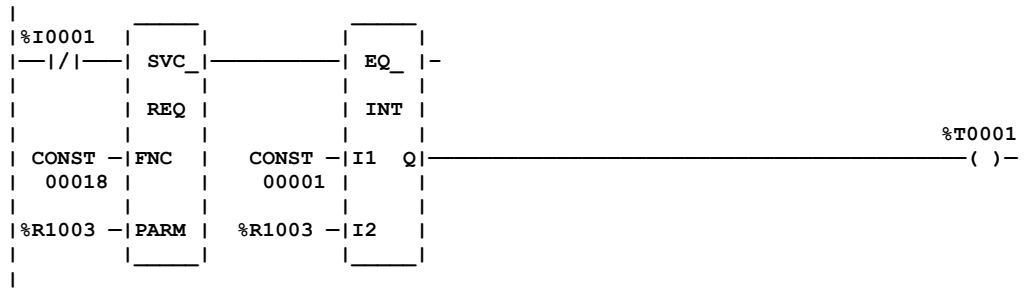
0 = Nejsou nastavené žádné přepisy 1 = Jsou nastavené přepisy	adresa

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.

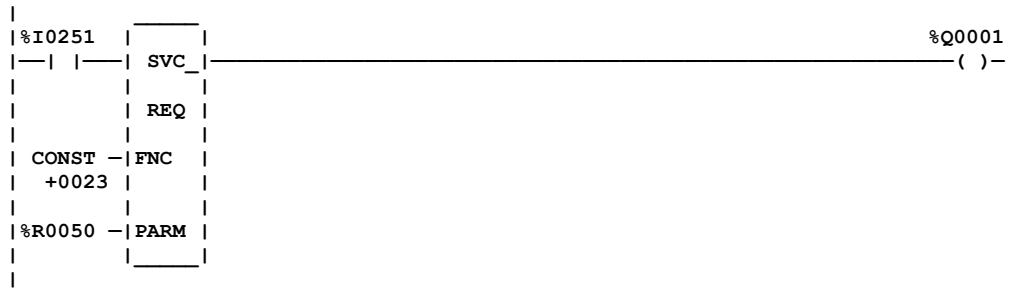
Když **RUN MODE STORE (ukládání za chodu programu)** bude aktivní, kontrolní součet programu nemusí být platný, dokud se uložení nedokončí. Proto na začátku bloku výstupních parametrů jsou dva příznaky, které udávají, kdy bude kontrolní součet programu a konfigurace platný.

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 #24: Reset inteligentního modulu

Funkci SVCREQ #24 použijte k resetování dceřinné desky a inteligentního modulu. Pokud nebude zadáno neplatné číslo sestavy a/nebo pozice, jak je ukázáno níže, výstup SVCREQ se nastaví do stavu ON.

U této funkce blok parametrů má délku 1 slovo. Je to pouze blok vstupního parametru.

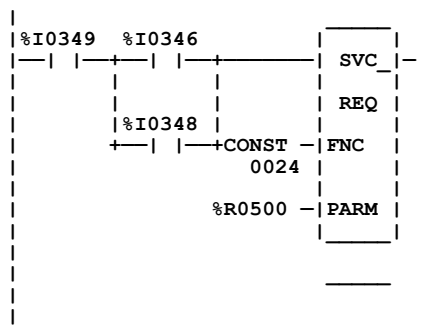
Pozice modulu (dolní bajt)	adresa
Sestava modulu (horní bajt)	

Poznámka: Sestava 0, pozice 1 budou indikovat, že má být poslaný reset do dceřinné desky.

Příklad

Když v následujícím příkladu vstup %I0346 bude ve stavu ON a vstup %I0349 bude ve stavu ON, provede se reset modulu indikovaného Sestavou/Pozicí uvedenou v %R0500.

Blok parametrů obsahující sestavu a pozice modulů pro Service Request reset modulu je na adrese %R0500. Blok parametrů může být nastavený kdekoliv v programu.



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.

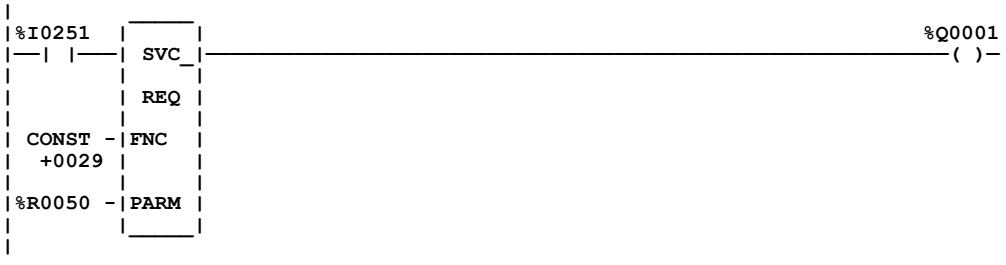
Tato funkce má pouze blok výstupních parametrů. Blok parametrů má délku 3 slova.

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.

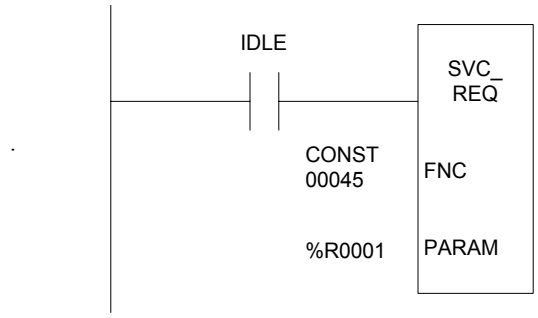
Tato funkce nemá žádný blok parametrů.

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

Tato funkce představuje způsob předávání několika bitů do nebo z jednoho nebo více inteligentních modulů přes propojovací rovinu PLC, který je v porovnání s normálním způsobem komunikace mnohem rychlejší. Toto zvýšení rychlosti komunikace se dosáhne omezením množství dat a počtu odezev.

Funkce SVCREQ #46 se používá k vykonání jedné z následujících funkcí přístupu k rychlé vnitřní sběrnici:

- Čtení slova přídavných stavových dat z jednoho nebo několika předepsaných inteligentních modulů.
- Zápis slova přídavných stavových dat z jednoho nebo několika předepsaných inteligentních modulů.
- 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

DSM314 (Modul digitálního serva) je v současné době jediný modul, který podporuje tento Service Request.

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 se také nazývá “SNAP.”

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 (slovo 1)

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. Blok parametrů 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 12-75.
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 a 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. Blok parametrů 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 12-75.
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 a 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 * 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, 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. Blok parametrů pro funkci Čtení/Zápis dat

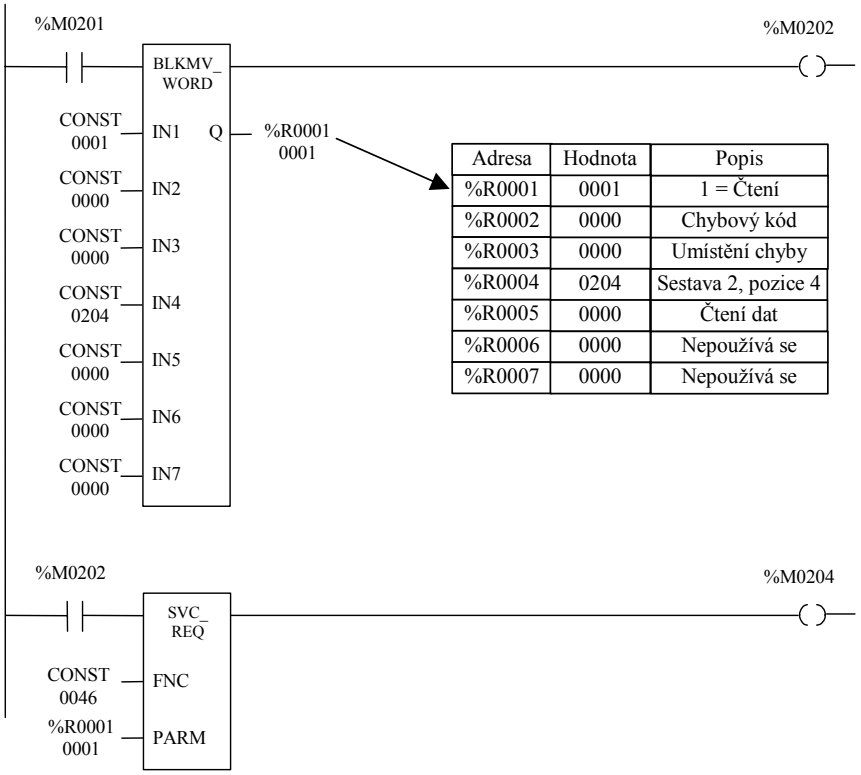
Umístění	Pole	Význam
Adresa	Funkce	3 = čtení/zápis
Adresa + 1	Chybový kód	Sem se uloží chybový kód, pokud se funkce nevykoná správně z důvodu, že nějaký modul bude chybět, nebude vhodný nebo nebude pracovat. Pokud se funkce vykoná, ale na některý z modulů data nepřijdou správně, nenastaví se žádný chybový kód. Podrobnosti viz “Chybové kódy” na straně 12-75.
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 a 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.

Tabulka 12-8. Chybové kódy

Hodnota	Popis
1	Úspěch - funkce se vykonala normálně.
-1	Modul není v předepsaném pozici.
-2	Nesprávný modul - modul v předepsané pozici není inteligentní modul nebo nepodporuje tuto funkci.
-3	Modul nepracuje - modul v předepsané pozici nekomunikuje správně s CPU.
-4	Chyba parity načtených dat - během operace čtení ze vzdálené nebo přídavné sestavy se vyskytla chyba parity.
-5	V povelovém bloku byla zadaná neplatná funkce.

Příklad 1

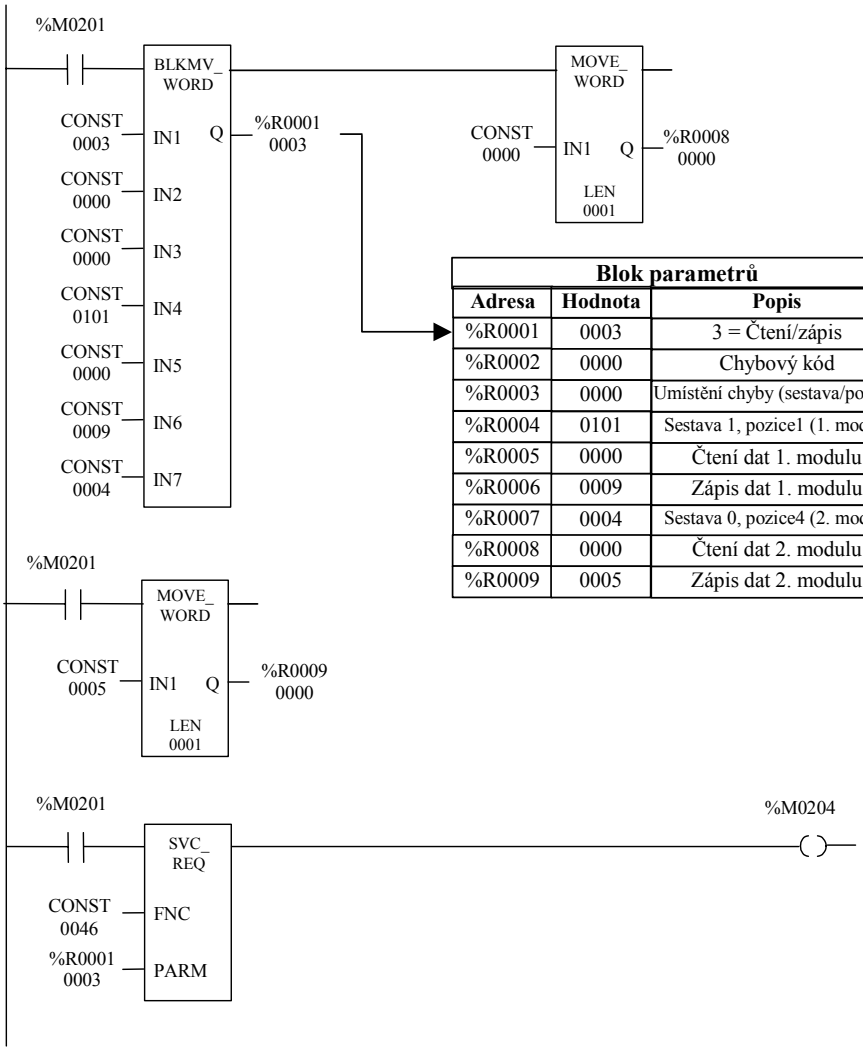
Následující příklad ukazuje Čtení (zadáno v %R0001) jednoho modulu umístěného v sestavě 2, pozice 4 (zadáno v %R0004). Pokud se funkce dokončí úspěšně, načtená data se uloží v %R0005. Pokud se vyskytne chyba, chybový kód se zapíše do %R0002 a v %R0003 se objeví sestava/pozice umístění modulu, který vygeneroval chybu. Všimněte si, že protože toto je funkce čtení jednoho modulu, adresa + 5 a adresa + 6 se nepoužívají. Proto odpovídající paměťová místa %R0006 a %R0007 se vyplní nulami ze vstupů IN6 a IN7 instrukce BLKMV. Pokud se má přečíst další modul, pro tento další modul se použijí %R0006 a %R0007. Více informací o funkci čtení najdete v tabulce 12-5 výše v této kapitole.



Adresa	Hodnota	Popis
%R0001	0001	1 = Čtení
%R0002	0000	Chybový kód
%R0003	0000	Umístění chyby
%R0004	0204	Sestava 2, pozice 4
%R0005	0000	Čtení dat
%R0006	0000	Nepoužívá se
%R0007	0000	Nepoužívá se

Příklad 2

V tomto příkladu instrukce BLKMV a dvě instrukce MOVE zapisují požadovaná data do bloku parametrů, který začíná na %R0001 (zadáno na vstupu SVCREQ PARM). Pokud bude povolená, funkce SVCREQ načte data přídavného stavového slova z modulu v sestavě 0, na pozici 4 a z modulu v sestavě 1, na pozici 1. Zapíše hodnotu 0005 do modulu v sestavě 0, pozici 4 a hodnotu 0009 do modulu v sestavě 1, pozici 1. (Všimněte si, že moduly nemusí být uvedené v bloku parametrů v pořadí podle čísla pozice.) Data načtená z modulu v sestavě 0, pozice 4 se uloží do %R0008. Data načtená z modulu v sestavě 1, pozice 1 se uloží do %R0005.



Blok parametrů		
Adresa	**Hodnota**	**Popis**
%R0001	0003	3 = Čtení/zápis
%R0002	0000	Chybový kód
%R0003	0000	Umístění chyby (sestava/pozice)
%R0004	0101	Sestava 1, pozice1 (1. modul)
%R0005	0000	Čtení dat 1. modulu
%R0006	0009	Zápis dat 1. modulu
%R0007	0004	Sestava 0, pozice4 (2. modul)
%R0008	0000	Čtení dat 2. modulu
%R0009	0005	Zápis dat 2. modulu

SVCREQ #48: Restartování po automatickém resetu fatální chyby

Kompatibilita pro SVCREQ 48

CPU – Tento Service Request je podporovaný softwarem verze 10.00 (nebo pozdější verze) pro CPU Series 90-30 331, 340, 341, 350, 36x a 37x.

Software – Tento Service Request je podporovaný pouze softwarem PLC VersaPro verze 1.1 (nebo pozdější verze). Software Logicmaster tuto funkci nepodporuje.

Výstraha

Restartování po automatickém resetu fatální chyby se nesmí použít (parametr Ignorovat fatální chyby musí být nastavený na Zakázáno) v aplikacích, kde by automatický restart PLC za stavu fatální chyby mohl vytvořit nespolehlivý stav řízeného zařízení. Projektant systému je zodpovědný za určení, jestli tuto funkci je možno bezpečně použít se systémem. Pokud na tuto výstrahu nebudete brát zřetel, může dojít ke zranění nebo usmrcení obsluhy a/nebo k poškození zařízení.

Popis

Service Request Restartování po automatickém resetu fatální chyby umožňuje, aby PLC po výskytu fatální chyby automaticky obnovilo normální činnost. Po fatální chybě PLC provede automaticky reset a obnoví vykonávání. Chyby se nevynulují, ale budou se brát jako nefatální. Pokud se po zapnutí napájení budou stále objevovat fatální chyby, PLC bude stále moct přejít do režimu běhu. Tato funkce je povolena parametrem Ignorovat fatální chyby (nebo Přepsat fatální chyby) v hardwarové konfiguraci CPU.

SVCREQ 48 nastaví maximální počet opakovaných pokusů a časový interval, během kterého se opakované pokusy mohou provádět. Pokud se počet opakovaných pokusů povolených během časového intervalu překročí, režim CPU se nastaví do STOP/FAULT. Pokud interval bude 0, režim CPU se nastaví do STOP/FAULT, když se překročí počet povolených pokusů.

Pokud obsluha provede vypnutí a zapnutí napájení, fatální chyby se budou ignorovat. Aktuální počet a časový interval se tím nastaví do počátečních podmínek. Celkový počet fatálních chyb se tím nezmění, ale celkový počet pokusů se inkrementuje. Systémový bit %S0021 se nastaví na 1 při každém úspěšném pokusu a zůstane na ní nastavený, dokud se všechny fatální chyby nevynulují nebo režim se nenastaví na STOP/FAULT.

Tabulka 12-9. Blok parametrů pro Restartování po automatickém resetu fatální chyby

Umístění	Pole	Význam
Slovo 1	Stav Service Request	Viz definice vrácených stavů níže. Uživatelský program musí toto slovo inicializovat na nulu.
Slovo 2	Neomezené pokusy	0 = Zakázat (počet pokusů je nastaveno slovem 3) 1 = Povoleno (Slova 3 a 4 se ignorují)
Slovo 3	Počet povolených pokusů	Rozsah hodnot je 0 až 128 0 = Automatické restartování zakázáno 1 až 128 = Maximální počet pokusů, které se smí vyskytnout během intervalu nastaveného ve slově 4.
Slovo 4	Interval opakování pokusů (v minutách)	Rozsah je 0 až 5940 minut (99 hodin) 0 = Ve slově 3 není nastavený žádný časový limit pro maximální počet pokusů. Automatické restartování bude povoleno pro daný počet pokusů. 1 až 5940 = Automatické restartování je zakázáno, pokud se během zadaného intervalu překročí zadaný počet pokusů.

Tabulka 12-10. Definice vrácených stavů pro Restartování po resetu fatální chyby

Stav	Popis	Poznámky	Proud
-5	Neplatný interval pokusů	Platný rozsah hodnot je 0 5940	Ne
-4	Neplatný počet pokusů	Platný rozsah je 0 až 128	Ne
-3	Neplatné neomezené pokusy	Musí být 0 nebo 1	Ne
-2	Konfigurace zakázána	Volba Ignorovat fatální chyby (Přepsat fatální chyby) musí být v hardwarové konfiguraci povolena.	Ne
0	Žádná akce	Povel nevyžaduje žádnou změnu	Ano
1	Automatický reset povolený	Platný povel povolí restartování po fatální chybě	Ano
2	Automatický reset zakázaný	Platný povel zakáže restartování po fatální chybě. Parametr Ignorovat fatální chyby zůstane povolený.	

SVCREQ 49 Statistika automatického resetu

Service Request 49 umožňuje přístup ke dvěma proměnným, které zaznamenávají celkový počet fatálních chyb a počet vyskytnutých se pokusů. Rozsah těchto proměnných je 0 až 65535. Tyto proměnné se nepřetáčejí, když se překročí jejich maximální hodnota. (Service Request 48 se používá k nakonfigurování maximálního počtu přípustných pokusů a časového limitu, během kterého se pokusy mohou vyskytnout.)

Tabulka 12-11. Blok parametrů pro statistiku automatického resetu

Slovo 1	Stav Service Request	Viz definice vrácených stavů níže. Uživatelský program musí toto slovo inicializovat na nulu.
Slovo 2	Povel	0 = Vrátí se celkový počet fatálních chyb a počet pokusů, které se vyskytly. 1 = Inicializovat celkový počet fatálních chyb a celkový počet pokusů na nulu.
Slovo 3	Vrácená hodnota = Celkový počet fatálních chyb, které se vyskytly.	Uživatelský program se musí inicializovat na nulu.
Slovo 4	Vrácená hodnota = Celkový počet pokusů automatického resetování	Uživatelský program se musí inicializovat na nulu.

Tabulka 12-12. Definice vrácených stavů pro statistiku automatického resetu

Stav	Popis	Poznámky	Proud
-2	Konfigurace zakázána	Volba Ignorovat fatální chyby (Přepsat fatální chyby) musí být v hardwarové konfiguraci povolena.	Ne
-1	Neplatný povel	Povel musí být 0 nebo 1.	Ne
1	Normální stav	Platný povel	Ano

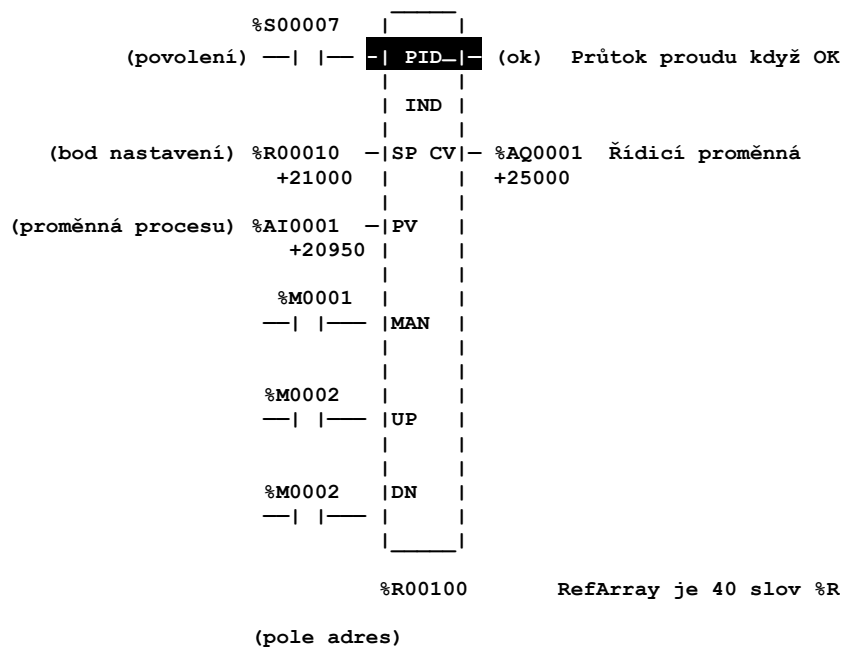
Kompatibilita CPU pro SVCREQ 49

Tento Service Request je podporovaný firmwarem verze 10.00 pro CPU Series 90-30 331, 340, 341, 350, 36x a 37x.

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 12-82) 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 12-13. 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	Nepoužívá se; 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 12-80).
%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 až 327.67 %/%
%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, udržuje PLC.	Pokud nebude nastaveno jinak, udržuje PLC:pokud 1, dolní bit nastaví Override (viz popis v tabulce “Detaily parametrů PID” na straně 12-85)
%Ref+0015	Interní SP	Nepoužívá se; nastavuje a spravuje PLC	Nenastavuje se
%Ref+0016	Interní CV	Nepoužívá se; nastavuje a spravuje PLC	Nenastavuje se
%Ref+0017	Interní PV	Nepoužívá se; nastavuje a spravuje PLC	Nenastavuje se
%Ref+0018	Výstup	Nepoužívá se; nastavuje a spravuje PLC	Nenastavuje se

Tabulka 12-13. Přehled parametrů PID - pokračování

Registr	Parametr	Jednotky dolního bitu	Rozsah hodnot
%Ref+0019	Paměť diferenciálních hodnot	Nepoužívá se; nastavuje a spravuje PLC	Nenastavuje se
%Ref+0020 a %Ref+0021	Paměť integrálních hodnot	Nepoužívá se; nastavuje a spravuje PLC	Nenastavuje se
%Ref+0022	Paměť odezvových hodnot	Nepoužívá se; nastavuje a spravuje PLC	Nenastavuje se
%Ref+0023	Hodiny	Nepoužívá se; nastavuje a	
%Ref+0024		spravuje	Nenastavuje se
%Ref+0025	(čas posledního vykonání)	PLC	
%Ref+0026	Uložení zbytku Y	Nepoužívá se; 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	Nepoužívá se	Nenastavuje se
%Ref+0035 • %Ref+0039	Vyhrazeno pro externí použití	Nepoužívá se	Nenastavuje se

Pole RefArray u PLC 90-30 se musí skládat z registrů %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 12-87) 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 12-14. 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 12-80); 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 12-14. 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 řídicího 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 výstup 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 ležet mimo meze pásma necitlivosti, pak se odchylka sníží o meze pásma necitlivosti. (odchylka = odchylka – mez 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 12-87) 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 12-14. Detaily parametrů PID - pokračování

Datový údaj	Popis
Ruční povel (13)	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.
Řídicí slovo (14)	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: **Bit: Hodnota slova: Funkce: Stav nebo Externí funkce, když bit Přepisu bude nastavený na 1:** 0 1 Přepis Je-li 0, monitorují se kontakty bloku níže. Je-li 1, nastaví se kontakty externě. 1 2 Ručně/Auto Je-li 1, blok bude v **Ručním** režimu; s jinými čísly bude v **Automatickém** režimu. 2 4 Povolení Normálně musí být 1; jinak by se blok nikdy nevyvolal 3 8 UP/roste Je-li 1 a Ručně (bit 1) bude v 1, CV se bude inkrementovat při každém řešení. 4 16 DN/klesá Je-li 1 a Ručně (bit 1) bude v 1, CV se bude dekrementovat při každém řešení.
SP (15)	(Nenastavuje se – nastavuje a spravuje PLC) Sleduje vstup SP; musí být nastaveno externě, je-li Přepis = 1.
CV (16)	(Nenastavuje se – nastavuje a spravuje PLC) Sleduje výstup CV.
PV (17)	(Nenastavuje se – nastavuje a spravuje PLC) Sleduje vstup PV; musí být nastaveno externě, je-li bit Přepis = 1.
Výstup (18)	(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.
Paměť diferenciálních hodnot (19)	Používá se interně pro uložení přechodných hodnot. *Nezapisujte do tohoto místa.*
Paměť integrálních hodnot (20/21)	Používá se interně pro uložení přechodných hodnot. *Nezapisujte do tohoto místa.*
Paměť odezvových hodnot (22)	Používá se interně pro uložení přechodných hodnot. *Nezapisujte do tohoto místa.*
Hodiny (23–25)	Paměť hodin uplynulého času (čas posledního vykonání PID). *Nezapisujte do těchto míst.*
Zbytek Y (26)	Pamatuje si zbytek při dělení integrátoru pro nulovou odchylku v ustáleném stavu.
Dolní a Horní rozsah (27/28)	Volitelné celočíselné hodnoty v jednotkách PV, které definují nejvyšší a nejnižší zobrazovanou hodnotu horizontálního sloupcového diagramu SP a PV v Logicmasteru (vyvoláno tlačítkem **Zoom**) a zobrazování ADS PID.
Vyhrazeno (29–34 a 35–39)	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.

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 (opačně působící), kterou je možno změnit na Přímo pracující režim (PV – SP) nastavením Hodnota odchylky na 1. Odchylka chyby je dolní bit (bit 0) v konfiguračním slově (%Ref+0012). V přímo působící proporcionální (P) smyčce zvětšení Proměnné procesu (PV) způsobí zvětšení Řídicí proměnné na výstupu (CV). V opačně působící proporcionální (P) smyčce zvětšení Proměnné procesu (PV) způsobí zmenšení Řídicí proměnné na výstupu (CV). Zavedení integrálního vztahu (I) změní chování. V přímo působící PI smyčce se řídicí proměnná (CV) zvýší, když proměnná procesu (PV) bude větší než bod nastavení (SP). V opačně působící PI smyčce se řídicí proměnná (CV) sníží, když proměnná procesu (PV) bude větší než bod nastavení (SP).

Přímé působení: Odchylka = měření – bod nastavení (PV-SP), Odchylka chyby = 1

Opačné působení: Odchylka = bod nastavení – měření (SP-PV), Odchylka chyby = 0

Poznámka. **Přímé působení** se někdy nazývá **Dopředné působení**.

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

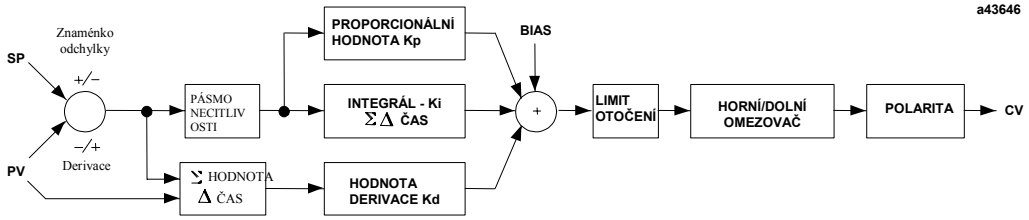
Kp = Kc Ki = Kc/Ti a Kd = Kc/Td

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 Řídicího 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, 0není 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

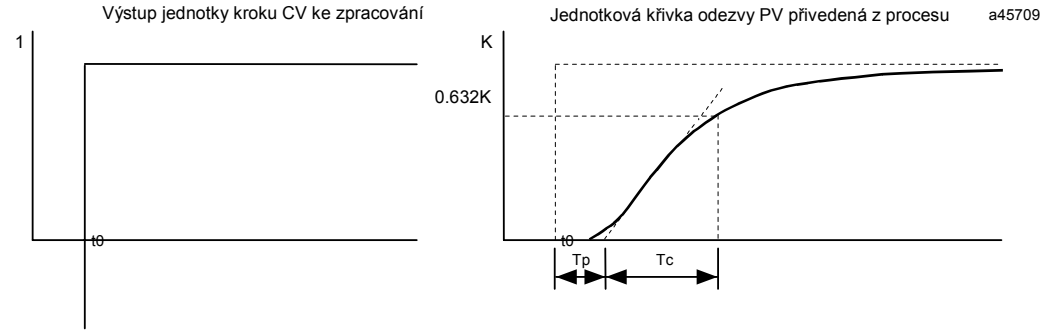
Zisky smyčky PID, Kp, Ki a Kd, jsou určené charakteristikami řízeného procesu. Dvě klíčové otázky pro nastavování smyčky PID jsou:

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 Řídicím 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

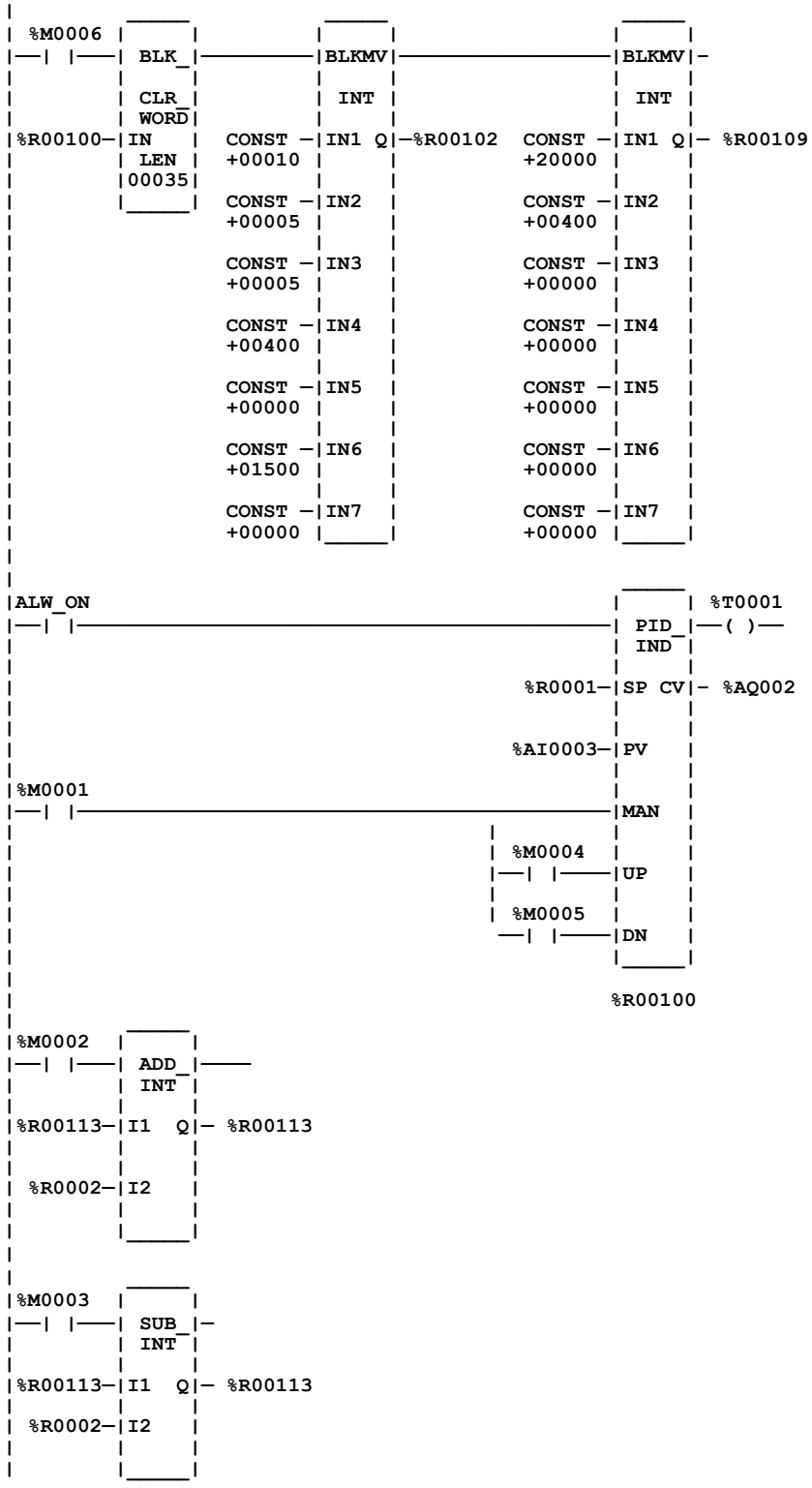
Jakmile budou určené počáteční zisky, je nutno je převést na celočíselné Uživatelské parametry. Aby se zabránilo problémům s měřítkem, zisk procesu K se musí vypočítat jako změna v jednotkách vstupu PV dělená krokem změny výstupu v jednotkách CV a ne v technických jednotkách PV nebo CV. Všechny časy 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 Řídicí 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. Sepnutí kontaktu %M0006 povolí pár instrukcí BLKMV, které nastaví počáteční hodnotu parametru zkopírováním konstant do 14 slov počínaje adresou %R102 (%Ref+2). (Poznámka: Aby se optimalizovaly parametry během procesu ladění, parametry zpřístupněte umístěním kurzoru softwaru Logicmaster na instrukci PID a stisknutím tlačítka F10, což je tlačítko Zoom.)

Blok je možno přepnout do Ručního režimu pomocí %M0001, aby bylo možno nastavit Ruční povel %R0113. Bit %M0004 nebo %M0005 je možno použít ke zvětšení nebo zmenšení %R0113 a CV PID a integrátor o 1 při řešení každých 100 milisekund. Pro rychlejší ruční obsluhu je možno použít bity %M0002 a %M0003 k přičtení nebo odečtení hodnoty v %R0002 k/od %R0113 při každém cyklu PLC. Výstup %T0001 bude v jedničce, když PID bude OK. Všimněte si, že některé registry v bloku parametrů se 40 registry nejsou zahrnuty, protože se v tomto příkladu nepoužívají nebo nejsou konfigurovatelné z důvodu, že je používá PLC systém. Další informace k parametrům najdete v tabulce 12-8.

Adresa	Hodnota	Popis
%R0102	+00010	Perioda vzorkování
%R0103	+00005	Pásmo necitlivosti +
%R0104	+00005	Pásmo necitlivosti –
%R0105	+00400	Proporcionální zisk (Kp)
%R0106	+00000	Derivační zisk (Kd)
%R0107	+01500	Integrální zisk (Ki)
%R0108	+00000	Posunutí CV/Výstupní Offset
%R0109	+20000	Horní omezovač
%R0110	+00400	Dolní Omezovač
%R0111	+00000	Minimální doba odezvy
%R0112	+00000	Konfigurační slovo
%R0113	+00000	Ruční povel
%R0114	+00000	Řídicí slovo
%R0115	+00000	Interní SP (nenastavuje se)


%M0006
BLK_
CLR_
WORD
%R00100
IN
LEN
00035
BLKMV
INT
IN1 Q
%R00102
CONST +00010
CONST +00005
CONST +00005
CONST +00400
CONST +00000
CONST +01500
CONST +00000
IN2
IN3
IN4
IN5
IN6
IN7
CONST +20000
CONST +00400
CONST +00000
CONST +00000
CONST +00000
CONST +00000
%R00109
ALW_ON
%T0001
PID_IND
%R0001
SP CV
%AQ002
%AI0003
PV
%M0001
MAN
%M0004
UP
%M0005
DN
%R00100
%M0002
ADD_INT
%R00113
I1 Q
%R0002
I2
%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 funkce vyžaduje 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-**Chyba! Nenalezen zdroj odkazů.Chyba! Nenalezen zdroj odkazů.Chyba! Záložka není definována.**.

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	Funkce	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
	Smazat bit	68	38	35	21	42	1	1	1	–	–	–	–	13
	Testovat bit	79	49	51	28	41	0	0	1	–	–	–	–	13
	Nastavit bit	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
Tabulka	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 35x-36x

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 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-2. Časování instrukce, Modely 35x-36x -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
	Smazat bit	20	0	–	20	0	–	13
	Testovat bit	20	0	–	20	0	–	13
	Nastavit bit	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 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 instrukce, Modely 35x-36x -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 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 instrukce, Modely 35x-36x -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 dějů (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 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-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 usec v případě použití formátu BCD nebo 148 usec v případě formátu Posix.

Doby uvedené pro reset jsou pro maximální velikost zásobníku s 1024 vzorky. (Reset vynuluje všechny vzorky v zásobníku vzorků.)

Tabulka A-4. Časování instrukce, modely 37x

Skupina funkcí	Funkce	Povoleno	Zakázáno	Inkrement	Velikost
		37x	37x	37x	
Časovače	Časovač zpoždění při zapnutí	4	5	–	15
	Časovač	2	2	–	15
	Časovač zpoždění při vypnutí	3	2	–	15
Čítače	Vzestupný čítač	2	2	–	13
	Sestupný čítač	1	2	–	13
Matematické	Sčítání (INT)	1	0	–	13
	Sčítání (DINT)	2	0	–	19
	Sčítání (REAL)	5	0	–	17
	Odečítání (INT)	1	0	–	13
	Odečítání (DINT)	2	0	–	19
	Odečítání (REAL)	5	0	–	17
	Násobení (INT)	5	0	–	13
	Násobení (DINT)	5	0	–	19
	Násobení (REAL)	5	0	–	17
	Dělení (INT)	5	0	–	13
	Dělení (DINT)	5	0	–	19
	Dělení (REAL)	5	0	–	17
	Dělení modulo (INT)	5	0	–	13
	Dělení modulo (DINT)	5	0	–	19
	Druhá odmocnina (INT)	5	0	–	10
	Druhá odmocnina (DINT)	10	0	–	13
	Druhá odmocnina (REAL)	5	0	–	11
Trigonometrické	SIN (REAL)	10	0	–	11
	COS (REAL)	10	0	–	11
	TAN (REAL)	10	0	–	11
	ASIN (REAL)	10	0	–	11
	ACOS (REAL)	10	0	–	11
	ATAN (REAL)	5	0	–	11
Logaritmické	LOG (REAL)	5	0	–	11
	LN (REAL)	5	0	–	11
Exponenciální	EXP	10	0	–	11
	EXPT	10	0	–	17
Převod radiánů	Převod RAD na DEG	5	0	–	11
	Převod DEG na RAD	5	0		11

Poznámky: 1. 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.
2. Doba povolení pro jednotky s jednoduchou délkou typu %R, %AI a %AQ.
3. Doba COMMREQ se měřila mezi CPU a HSC.
4. DOIO je doba k vygenerování hodnoty na výstupu diskrétního výstupního modulu.
5. Kde je více možností, výše uvedená doba představuje nejhorší možný případ.

Tabulka A-4. Časování instrukce, modely 37x - pokračování

Skupina funkcí	Funkce	Povoleno	Zakázáno	Inkrement	Velikost
		37x	37x	37x	
Relační	Rovno (INT)	1	0	–	10
	Rovno (DINT)	2	0	–	16
	Rovno (REAL)	5	0	–	14
	Nerovno (INT)	1	0	–	10
	Nerovno (DINT)	1	0	–	16
	Nerovno (REAL)	5	0	–	14
	Větší než (INT)	1	0	–	10
	Větší než (DINT)	1	0	–	16
	Větší než (REAL)	5	0	–	14
	Větší než / Rovno (INT)	1	0	–	10
	Větší než / Rovno (DINT)	1	0	–	10
	Větší než / Rovno (REAL)	5	0	–	14
	Menší než (INT)	1	0	–	10
	Menší než (DINT)	1	0	–	16
	Menší než (REAL)	5	0	–	14
	Menší než / Rovno (INT)	1	0	–	10
	Menší než / Rovno (DINT)	3	0	–	16
	Menší než / Rovno (REAL)	5	0	–	14
	Rozsah (INT)	2	0	–	13
	Rozsah (DINT)	2	0	–	22
	Rozsah (WORD)	1	0	–	13
Bitové operace	Logický AND	2	0	–	13
	Logický OR	2	0	–	13
	Logický Exclusive OR	1	0	–	13
	Logická inverze, NOT	1	0	–	10
	Posunutí bitu doleva	5	0	1	16
	Posunutí bitu doprava	5	0	1	16
	Rotace bitu doleva	5	0	1	16
	Rotace bitu doprava	5	0	1	16
	Pozice bitu	5	0	–	13
	Smazat bit	5	0	–	13
	Testovat bit	5	0	–	13
	Nastavit bit	5	0	–	13
	Maskované porovnání (WORD)	9	0	–	25
	Maskované porovnání (DWORD)	10	0	–	25

Poznámky: 1. 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.
2. Doba povolení pro jednotky s jednoduchou délkou typu %R, %AI a %AQ.
3. Doba COMMREQ se měřila mezi CPU a HSC.
4. DOIO je doba k vygenerování hodnoty na výstupu diskrétního výstupního modulu.
5. Kde je více možností, výše uvedená doba představuje nejhorší možný případ.

Tabulka A-4. Časování instrukce, modely 37x - pokračování

Skupina funkcí	Funkce	Povoleno	Zakázáno	Inkrement	Velikost
		37x	37x	37x	
Přesunutí dat	Přesunutí (INT)	2	0	1	10
	Přesunutí (BIT)	5	0	1	13
	Přesunutí (WORD)	2	0	1	10
	Přesunutí (REAL)	5	0	1	13
	Přesun bloku (INT)	2	0	–	28
	Přesun bloku (WORD)	3	0	–	28
	Přesun bloku (REAL)	11	1	–	13
	Smazání bloku	1	0	1	11
	Posunutí registru (BIT)	10	0	1	16
	Posunutí registru (WORD)	15	0	1	16
	Bitový sekvenční přepínač	14	10	1	16
	COMM_REQ	200	200	–	13
Tabulka	Přesunutí pole				
	INT	10	0	1	22
	DINT	15	0	1	22
	BIT	10	0	1	22
	BYTE	10	0	1	22
	WORD	10	0	1	22
	Hledat shodné				
	INT	5	0	1	19
	DINT	5	0	2	22
	BYTE	5	0	1	19
	WORD	5	0	1	19
	Hledat neshodné				
	INT	5	0	1	19
	DINT	10	0	2	22
	BYTE	5	0	2	19
	WORD	5	0	2	19
	Hledat větší než				
	INT	5	0	1	19
	DINT	5	0	2	22
	BYTE	10	0	1	19
	WORD	5	0	1	19
	Hledat větší než / Rovno				
	INT	5	0	1	19
	DINT	5	0	2	22
	BYTE	5	0	1	19
	WORD	5	0	1	19

Poznámky:
1. 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.
2. Doba povolení pro jednotky s jednoduchou délkou typu %R, %AI a %AQ.
3. Doba COMMREQ se měřila mezi CPU a HSC.
4. DOIO je doba k vygenerování hodnoty na výstupu diskrétního výstupního modulu.
5. Kde je více možností, výše uvedená doba představuje nejhorší možný případ.
6. U instrukcí, které mají hodnotu inkrementu, násobte inkrement (délkou –1) a přičtěte tuto hodnotu k základnímu času.

Tabulka A-4. Časování instrukce, modely 37x – pokračování

Skupina funkcí	Funkce	Povoleno	Zakázáno	Inkrement	Velikost
		37x	37x	37x	
	Hledat menší než				
	INT	5	0	1	19
	DINT	10	0	2	22
	BYTE	5	0	1	19
	WORD	5	0	1	19
	Hledat menší než / Rovno				
	INT	5	0	1	19
	DINT	5	0	2	22
	BYTE	5	0	1	19
	WORD	5	0	1	19
Převod	Převod na INT	5	0	–	10
	Převod na BCD-4	5	0	–	10
	Převod na REAL	5	0	–	8
	Převod na WORD	5	0	–	11
	Celá část INT	5	0	–	11
	Celá část DINT	5	0	–	11
Řízení	Volání podprogramu	15	0	–	7
	Do I/O	5	0	–	13
	Algoritmus PID – ISA	14	10	–	16
	Algoritmus PID – IND	14	10	–	16
	Instrukce End	–	–	–	–
	Service Request				
	#6	5	0	–	10
	# 7 (Čtení)	10	0	–	10
	# 7 (Nastavení)	5	0	–	10
	#14	15	0	–	10
	#15	5	0	–	10
	#16	10	0	–	10
	#18	255	0	–	10
	#23	25	0	–	10
	#26//30**	155	0	–	10
	#29	5	0	–	10
	Vnořené MCR/ENDMCR kombinované	1	0	–	4
	Sekvenční událost záznamník (SER) 8 kanálů	60	0	=	
	Sekvenční událost záznamník (SER) 16 kanálů	199	0	=	

**Service request #26/30 byl měřený s použitím vysokorychlostního čítače, 16-bodového výstupu v sestavě s 5 pozicemi.

Poznámky: 1. 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.
2 Doba povolení pro jednotky s jednoduchou délkou typu %R, %AI a %AQ.
3. Doba COMMREQ se měřila mezi CPU a HSC.
4. DOIO je doba k vygenerování hodnoty na výstupu diskrétního výstupního modulu.
5. Kde je více možností, výše uvedená doba představuje nejhorší možný případ.
6. U instrukcí, které mají hodnotu inkrementu, násobte inkrement (délkou –1) a přičtěte tuto hodnotu k základnímu času.

Doby vykonávání Booleovských funkcí CPU

Tato tabulka uvádí doby vykonávání cívek a kontaktů pro moduly CPU Series 90-30.

Tabulka A-5. Doby vykonávání Booleovských funkcí

Model CPU	Doba vykonávání 1000 Booleovských kontaktů/cívek
Modely 37x	0.15 milisekundy
Modely 35x a 36x	0.22 milisekundy
Modely 340/341	0.3 milisekundy
Model 331	0.4 milisekundy
Modely 313/323	0.6 milisekundy
Model 311	18.0 milisekund

Velikosti instrukcí pro CPU 350 - 374

Velikost paměti v následující tabulce udává počet bajtů uživatelské paměti požadované danou instrukcí v aplikačním programu žebříkové logiky.

Tabulka A-6. Velikosti instrukcí pro CPU 350 – 374

Funkce	Velikost paměti
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	různé

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. Data chyb v těchto tabulkách existují pouze v PLC a ne v adresáři. Proto pokud budete používat Logicmaster a budete chtít si chyby prohlížet, musíte být připojeni buď v režimu ONLINE nebo MONITOR.

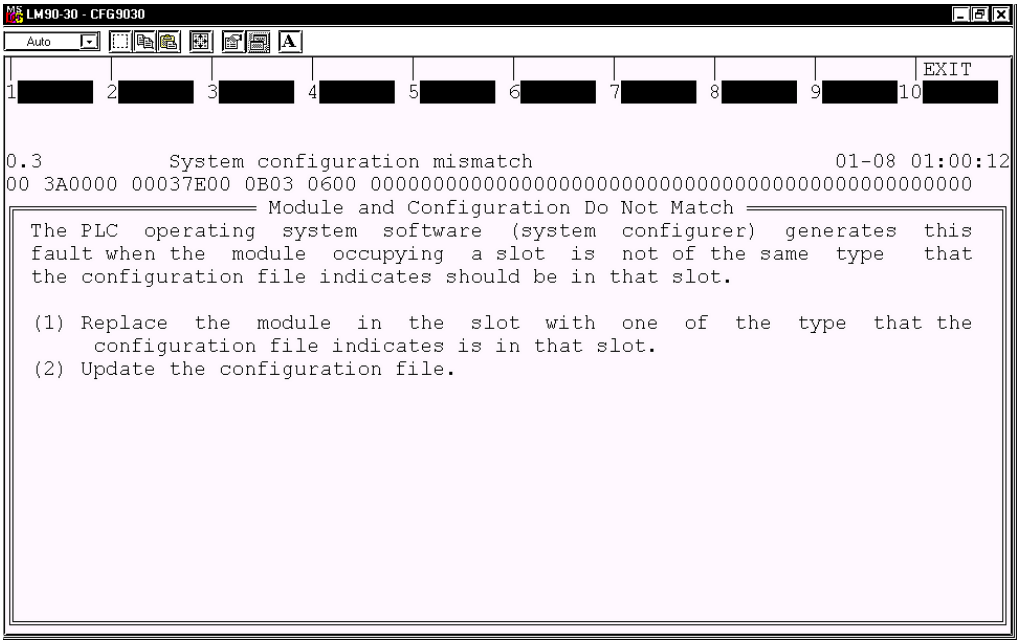
- Tabulka chyb PLC obsahuje:
  - Lokalizaci chyby
  - Popis chyby
  - Datum a čas chyby
- Tabulka chyb I/O obsahuje:
  - Lokalizaci chyby
  - Adresu
  - Kategorii chyby
  - Typ chyby
  - Datum a čas chyby

Tabulka chyb PLC

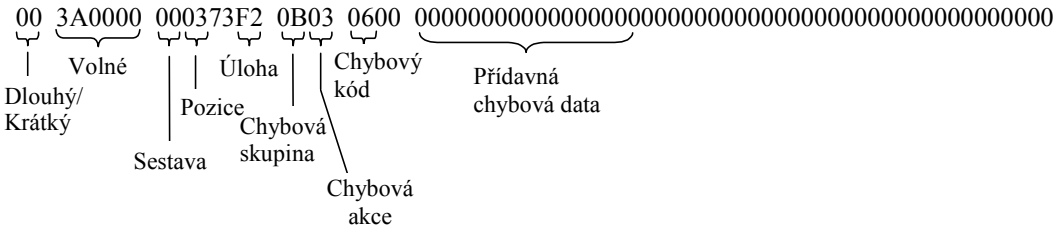
Přístup do tabulky chyb PLC je přes programovací software.Informace o přístupu do tabulky chyb najdete v kontextové nápovědě, *v Návodu k používání programovacího softwaru Logicmaster 90 Series 90-30/20/Micro*, GFK-0466.

Příklad

Následující obrázek ukazuje příklad chyby Nesouladu konfigurace systému, u které se detaily zobrazily na obrazovce.



V následujícím schématu jsou vyznačena jednotlivá pole záznamu chyby pro výše zobrazenou chybu Nesouladu konfigurace systému.



Tento zápis chyby nesouladu konfigurace systému vypadá následovně. (Všechna data jsou v hexadecimálním tvaru.)

Pole	Hodnota	Poznámka
Dlouhý/krátký	00	00=Krátký, což udává, že se používá pouze 8 bajtů pole přídavných dat chyby.
Sestava	00	Hlavní sestava (sestava 0)
Pozice	03	Pozice 3
Úloha	F2	
Chybová skupina	0B	Chyba nesouladu konfigurace systému
Význam chyby	03	FATÁLNÍ chyba
Chybový kód	06	
Přídavná data chyby	00	V 8 bajtech se nehlásí žádná přídavná data chyby

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 se používá 8 bajtů nebo všech 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	Diagnostika
5	5	Přidání nebo přespočetná sestava	Diagnostika
8	8	Přidání nebo přespočetný přídavný modul	Diagnostika
11	B	Nesoulad konfigurace systému	Fatální
12	C	Chyba systémové sběrnice	Diagnostika
13	D	Hardwarová porucha CPU PLC	Fatální
14	E	Nefatální hardwarová chyba modulu	Diagnostika
16	10	Chyba softwaru přídavného modulu	Diagnostika
17	11	Chyba kontrolního součtu programového bloku	Fatální
18	12	Signál nízkého napětí baterie	Diagnostika
19	13	Překročení konstantního času cyklu	Diagnostika
20	14	Systémová tabulka chyb PLC plná	Diagnostika
21	15	Tabulka chyb I/O plná	Diagnostika
22	16	Chyba uživatelské aplikace	Diagnostika
–	–	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 spojený spojena jedna ze tří akcí. Tyto chybové akce jsou pevně spojené s PLC Series 90-30 a nelze je uživatelsky měnit.

Tabulka B-2. Chybové akce PLC

Význam chyby	Kroky, které provede CPU	Kód
Informační	Záznam chyby do tabulky chyb	1
Diagnostika	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	2C	Neprovedl se softwarový reset přídavného modulu
45	2D	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	Zdroj (přiřazení, přetečení tabulky, atd.) chyba
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. Následující příkladu ukazuje, jaké chyby se mohou vyskytovat:

Příklad - Poškozený obsah skupiny uživatelské 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.

Časová značka chyby PLC

Hardwarová porucha CPU PLC (porucha RAM)

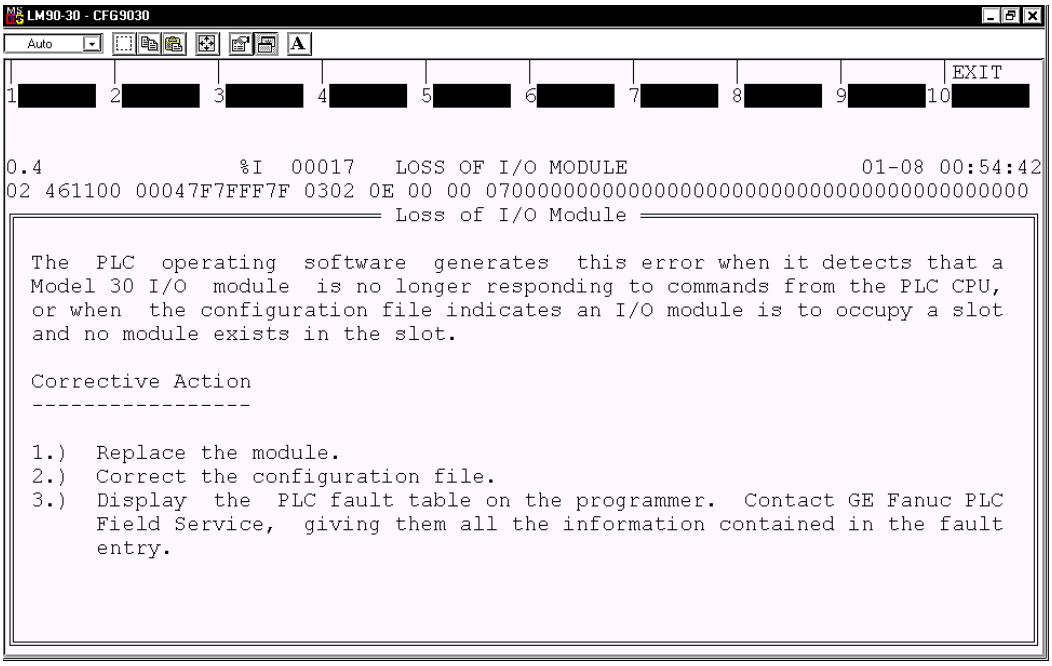
Šestibajtová časová značka je hodnota systémových hodin, kdy CPU PLC provedlo zaznamenání chyby. (Hodnoty jsou kódované ve formátu BCD.)

Tabulka B-6. Časová značka chyby PLC

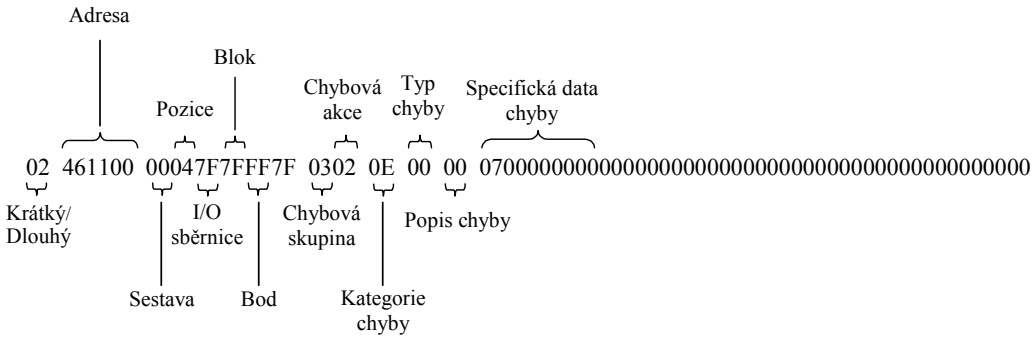
Číslo bajtu	Popis
1	Sekundy
2	Minuty
3	Hodiny
4	Den v měsíci
5	Měsíc
6	Rok

Tabulka chyb I/O

Následující obrázek ukazuje příklad chyby Ztráty I/O modulu, která je podrobně zobrazena na následující obrazovce.



V následujícím schématu jsou vyznačeny hexadecimální informace zobrazené v jednotlivých polích záznamu chyby (Ztráta I/O modulu) v obrázku výše.



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 konkrétní chyba obsahuje 5 bajtů nebo 21 bajtů pole 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

Bajt	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 CPU. Sestavy 1 až 7 jsou přídavné sestavy.

Pozice

Číslo pozice může být v rozsahu 0 až 9. CPU PLC je vždy umístěný ve slotu 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	Význam chyby
3	Ztráta nebo chybějící modul I/O	Diagnostika
7	Přidání nebo přespočetný modul I/O	Diagnostika
9	Chyba IOC nebo I/O sběrnice	Diagnostika
A	Chyba I/O modulu	Diagnostika
–	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

Význam chyby	Kroky, které provede CPU	Kód
Informační	Záznam chyby do tabulky chyb	1
Diagnostika	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, která uvádí tyto mnemotechnické zkratky, můžete zobrazit kdykoliv během programování v Logicmasteru stisknutím tlačítek ALT a I.

Funkce 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							
	Mocnina 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	
	Testovat bit	&BT						&BT_W	
	Nastavit bit	&BS						&BS_W	
	Smazat bit	&BCL						&BCL_W	
	Pozice bitu	&BP						&BP_W	
	Maskované porovnání	&MCMP						&MCM_W	
Převod	Převod na celé číslo	&TO_INT	&TO_INT_BCD4						
	Převod na celé číslo s dvojnásobnou délkou	&TO_DINT							&BCD4_R
	Převod na BCD-4	&BCD4			&TO_REAL_DI				
	Převod na REAL	&TO_REAL						&TO_REAL_W	
	Převod na WORD	&TO_W							
	Celá část celého čísla	&TRINT							
	Zaokrouhlení na celé číslo s dvojitou délkou	&TRDINT							

Funkční skupina	Instrukce	Mnemotechnická zkratka						
		Všechny	INT	DINT	BIT	BYTE	WORD	REAL
Přesunutí dat	Přesunutí Přesunutí bloku Smazání bloku Posunutí registru 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	Přesunutí 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 End Výklad logické příčky (poznámka) Systémový Service 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 9.00 nebo pozdější a všechny verze řady CPU352 a 37x.

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, 36x nebo CPU řady 37x (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.

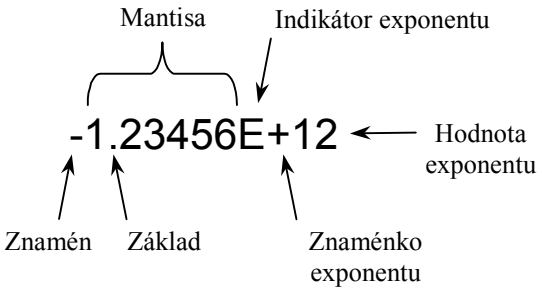
Používá se následující formát. Pro čísla v rozsahu 9999999 až 0.0001 zobrazení nemá žádný exponent a má až šest nebo sedm platných číslic. Například:

Zapsáno	Zobrazeno	Popis
.000123456789	+.0001234567	Deset číslic, šest nebo sedm platných
–12.345e-2	–0.1234500	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: +1.23456E+12

Terminologie reálných čísel

Reálné číslo je uloženo ve 32-bitovém registru s délkou dvojitého slova. Následující obrázek uvádí termíny používané pro součásti reálného čísla.



Znaménko – Buď plus nebo mínus. Uložené v nejvýznamnějším bitu (bit 32) dvojitého slova. Jednička v bitu 32 znamená záporné znaménko. Nula v bitu 32 znamená kladné znaménko.

Základ – Symbol tečky, který odděluje celou část čísla od desetinné části mantisy. U dekadických čísel se základ běžně nazývá desetinná tečka.

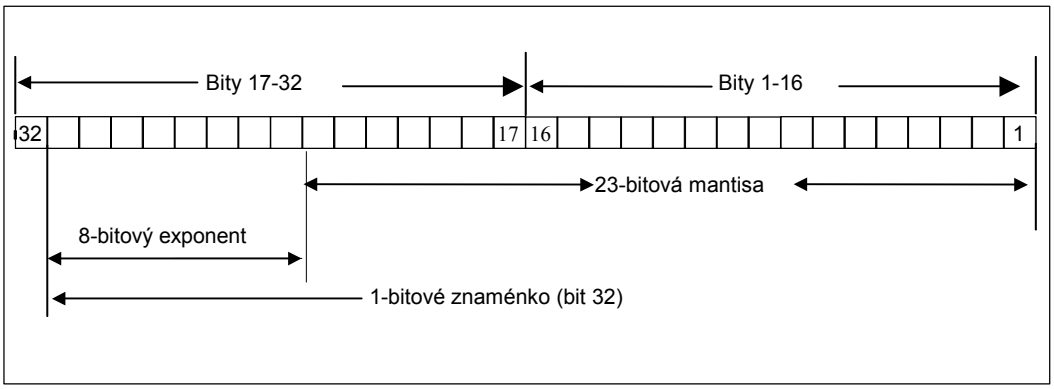
Exponent – (Také se nazývá"Charakteristika"). Je uložený v 8 bitech, na pozicích bitu 31 až 24 32-bitového dvojitého slova. Exponent může mít hodnotu v rozsahu +127 až –126; avšak exponent je vždy uložený jako kladné číslo, protože CPU před jeho uložením automaticky přičte k jeho hodnotě 127.

Mantisa – (Také se nazývá Váha"). Základní číslo bez znaménka a exponentu. Je uložena ve 23 bitech, na pozicích bitů 23 až 1 32-bitového slova.

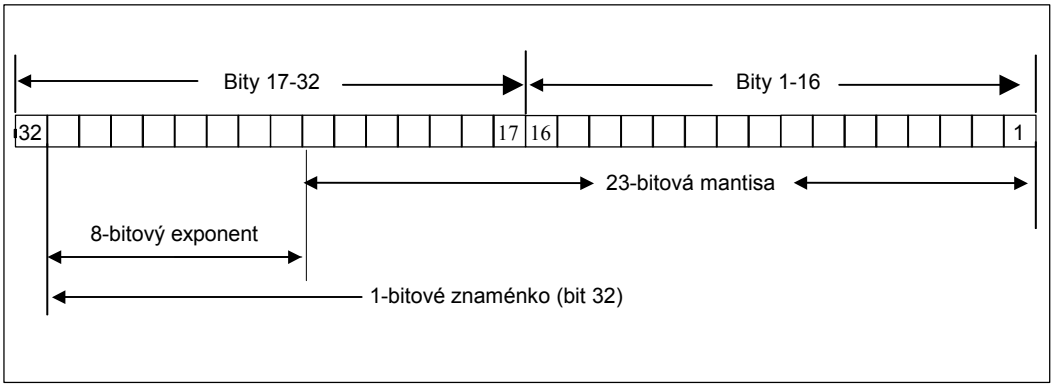
Přesnost – Vztahuje se k počtu významných míst, které je možno použít k uložení. Protože registr celého čísla s dvojitou délkou používá k uložení čísla 31 bitů (bit 32 se používá pro znaménko), může potenciálně uložit čísla s větší přesností než registr reálného čísla (plovoucí desetinná tečka), který k uložení mantisy čísla používá pouze 23 bitů.

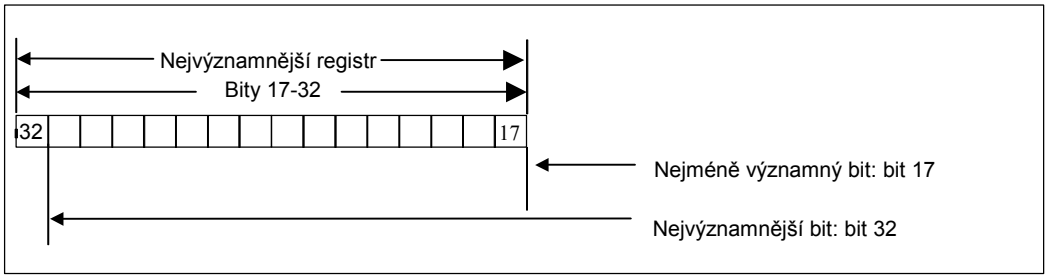
Interní formát čísel s pohyblivou desetinnou tečkou

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 %R0005 a %R0006, %R0005 bude nejméně významný registr a %R0006 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	Zobrazení v Logicmasteru
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 nebo nesprávných zápisů čísla s pohyblivou desetinnou tečkou jsou uvedené v následující tabulce:

Nesprávný zápis	Výklad /Výsledek
–433E23	Chybí desetinná tečka. LM90 zobrazí hlášení “Bad numeric value” (nesprávná číselná hodnota).
10e-19	Chybí desetinná tečka. LM90 zobrazí hlášení “Bad numeric value” (nesprávná číselná hodnota).
1 0.e19	Mezi 1 a 0 v mantise je mezera. Reálná čísla musí být zapsaná bez mezer mezi číslicemi nebo znaky. Logicmaster chápe tento zápis jako nesprávnou hodnotu +1.000000.
4.1e 19	Mezi e a 19 v exponentu je mezera. Reálná čísla musí být zapsaná bez mezi číslicemi nebo znaky. Logicmaster chápe tento zápis jako nesprávnou hodnotu +4.100000.

Chyby v číslech s pohyblivou desetinnou tečkou a operace

Kladné a záporné nekonečno

Pokud funkce REAL vygeneruje číslo větší než 3.402823E+38 nebo menší než -3.402823E+38, u CPU 352 nebo 374 dojde k přetečení. U všech ostatních modelů 90-30, které podporují operace s pohyblivou desetinnou tečkou, je rozsah větší 2^{16} nebo menší než $-2^{16.}$ Když vaše číslo bude přesahovat rozsah, výstup ok funkce se nastaví do stavu OFF a výsledek se nastaví na kladné nekonečno (pro číslo větší než 3.402823E+38 v případě CPU 352 nebo 374 CPU nebo 2^{16} u všech ostatních modelů) nebo na záporné nekonečno (pro číslo menší než –3.402823E+38 nebo -2^{16} u všech ostatních modelů). Místo výskytu lze určit testováním ok výstupu.

Mnemotechnická zkratka	Hodnota na obrazovce žebříku	Referenční tabulková hodnota (Hex)	Popis
POS_INF	+OVERFLOW	7F80 0000	IEEE hexadecimální reprezentace kladného nekonečna.
NEG_INF	-OVERFLOW	FF80 0000	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 nebo 374), 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.

Nenumerický (NaN)

Nenumerické číslo je nedefinované číslo, například výsledek dělení nuly nulou. Kladná a záporná nekonečna se nepokládají za čísla NaN. Následující část textu vám pomůže zjistit, kdy vznikne výsledek NaN.

Kódy NaN pro CPU 352 nebo 374

U CPU 352 nebo 374 každá funkce REAL, která je schopná vytvořit NaN, vygeneruje speciální kód NaN, který identifikuje funkci, a je možno ho přečíst v příslušné referenční tabulce. Na obrazovce žebříku Logicmaster se zobrazí indikace termínu bez znaménka "OVERFLOW". (Pokud termín "OVERFLOW" bude mít před sebou znaménko plus nebo mínus, označuje to kladné nebo záporné nekonečno.)

Kódy nenumerického čísla (NaN) pro CPU 352 a 374		
Mnemotechnická zkratka	**Referenční tabulková hodnota (Hex)**	**Popis**
NaN_ADD.	7F81 FFFF	Hodnota chyby přičtení reálného čísla v hexadecimálním tvaru
NaN_SUB	7F81 FFFF	Hodnota chyby odečtení reálného čísla v hexadecimálním tvaru
NaN_MUL	7F82 FFFF	Hodnota chyby násobení reálného čísla v hexadecimálním tvaru
NaN_DIV	7F83 FFFF	Hodnota chyby dělení reálného čísla v hexadecimálním tvaru
NaN_SQRT	7F84 FFFF	Hodnota chyby druhé odmocniny reálného čísla v hexadecimálním tvaru
NaN_LOG	7F85 FFFF	Hodnota chyby logaritmu reálného čísla v hexadecimálním tvaru
NaN_POW0	7F86 FFFF	Hodnota chyby exponentu reálného čísla v hexadecimálním tvaru
NaN_SIN	7F87 FFFF	Hodnota chyby sinu reálného čísla v hexadecimálním tvaru
NaN_COS	7F88 FFFF	Hodnota chyby cosinu reálného čísla v hexadecimálním tvaru
NaN_TAN	7F89 FFFF	Hodnota chyby tangensu reálného čísla v hexadecimálním tvaru
NaN_ASIN	7F8A FFFF	Hodnota chyby inverzního sinu reálného čísla v hexadecimálním tvaru
NaN_ACOS	7F8B FFFF	Hodnota chyby inverzního cosinu reálného čísla v hexadecimálním tvaru
NaN_BCD	7F8C FFFF	Chyba převodu BCD-4 na reálné číslo.
REAL_INDEF	FFC0 0000	Reálné nekonečno, chyba dělení nuly nulou.

Kód NaN pro CPU 35x, 36x a 37x (kromě CPU 352)

Všechna CPU Series 90-30, která podporují operace s pohyblivou desetinnou tečkou na bázi softwaru (kromě CPU 352, které je na bázi hardwaru), vygenerují pouze jeden výstup NaN: FFFF FFFF. Na obrazovce žebříku Logicmaster se zobrazí indikace termínu bez znaménka "OVERFLOW".

Nenumerický (NaN) typ pro CPU 35x, 36x a 37x(kromě CPU 352)		
Mnemotechnická zkratka	**Referenční tabulková hodnota (Hex)**	**Popis**
NaN_SW	FFFF FFFF	Kód softwarové pohyblivé desetinné tečky pro všechna NaN

Průchod a proud pro čísla NaN a nekonečno

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 matematické 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

Dodatek F

Srovnání programovacího softwaru

Tento manuál byl napsaný pro uživatele Logicmasteru (programovací software na bázi DOS). Softwarové produkty PLC na bázi Windows, například CIMPLICITY® Machine Edition Logic Developer a VersaPro®, obsahují informace o instrukční sadě PLC v systému vestavěné kontextové nápovědy softwaru místo v manuálu. Uživatelé programovacího softwaru na bázi Windows si musí uvědomit, že se instrukce objevují odlišně od způsobu, než jak se objevovaly na obrazovce Logicmasteru (v PLC stále pracují stejně). Systém kontextové nápovědy má nejpřesnější informace o používání instrukční sady v programovacím softwaru na bázi Windows.

Kromě systému kontextové nápovědy můžete informace o používání softwaru najít v následujících manuálech:

Uživatelská příručka programovacího softwaru VersaPro™, GFK-1670

Krátký úvod k CIMPLICITY® Machine Edition, GFK-1868

Poznámky

Podpora pro instrukce bubnového sekvenčního přepínače

Tuto instrukci, kterou podporují CPU 350-364 verze 10.00 a pozdější a všechny verze CPU37x, nepodporuje žádná verze Logicmasteru; proto není v tomto manuálu popisovaná. Tuto instrukci podporuje VersaPro počínaje verzí 1.1 a všechny verze Logic Developer. Informace k této instrukci můžete najít v kontextové nápovědě vestavěné v těchto dvou softwarových balících.

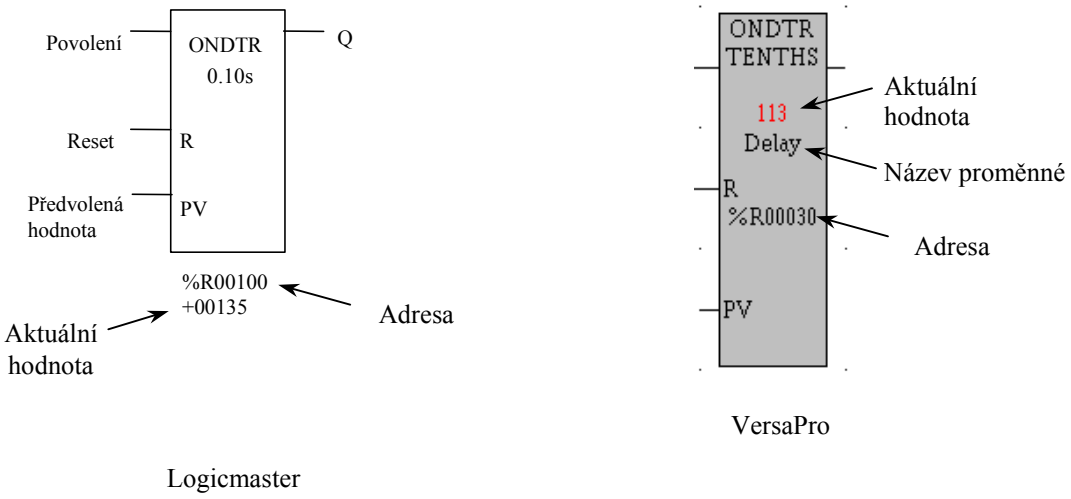
Značky začátku a konce programu

Tyto značky se používají na obrazovkách žebříkové logiky Logicmasteru, ale na obrazovkách programovacích zařízení žebříkové logiky na bázi Windows nejsou vidět.

Umístění adresy řídicího slova instrukce

Určité instrukce, například časovače, čítače a bitové sekvenční přepínače, vyžadují k uložení určitých interních výpočtů skupinu po sobě jdoucích slov. Tato skupina slov se obvykle nazývá řídicí blok. V Logicmasteru se adresa prvního slova řídicího bloku (i jiných hodnot uložených na této adrese) na obrazovce žebříkové logiky objeví pod instrukcí (na obrázku níže jako %R00100). U programovacích zařízení na bázi Windows se tato adresa prvního slova objeví na obrazovce žebříkové logiky uvnitř instrukce (na obrázku níže jako %R00030). VersaPro také zobrazí název proměnné této adresy (na obrázku níže jako Delay) uvnitř instrukce. Pokud adrese nikdo nepřiřadil

název proměnné, adresa samotná bude implicitní název proměnné (takže adresa se objeví uvnitř instrukce na obou místech). Přímo nad slovem Delay v zobrazení VersaPro je hodnota 113, která představuje aktuální hodnotu uloženou v této proměnné.



Logicmaster

VersaPro

Rozdíly zobrazení reálného čísla

Mezi způsobem, jakým programy zobrazují nedefinované výsledky, například když se provádí výpočet dělení nulou, jsou rozdíly. Dodatek E tohoto manuálu uvádí, jak Logicmaster zobrazuje tyto výsledky na obrazovce žebříkové logiky a v referenčních tabulkách.

A

B

C

Č

D

E

F

G

H

Ch

M

N

O

P

R

Ř

S

T