SIEMENS

Simatic

S7

Programowalny sterownik S7-1200

Podręcznik systemu

Wydanie 04/2009

Informacje prawne

System ostrzeżeń

Uwagi pojawiające się w tym podręczniku służyć mają zachowaniu bezpieczeństwa ludzi i uniknięcia szkód wynikłych z niewłaściwego użytkowania urządzenia. Wskazówki te podzielono i oznaczono zależnie od stopnia zagrożenia w następujący sposób:

ZAGROŻENIE

oznacza, że w przypadku nie zachowania odpowiednich środków bezpieczeństwa **występuje** zagrożenie śmiercią lub ciężkimi obrażeniami ciała.

OSTRZEŻENIE

oznacza, że w przypadku nie zachowania odpowiednich środków bezpieczeństwa **może** wystąpić zagrożenie śmiercią lub ciężkimi obrażeniami ciała.

OSTROŻNIE

Ze znakiem ostrzegawczym oznacza, że w przypadku nie za chowania odpowiednich środków bezpieczeństwa mogą wystąpić lekkie obrażenia ciała.

OSTROŻNIE

Bez znaku ostrzegawczego oznacza, że w przypadku nie zachowania odpowiednich środków bezpieczeństwa mogą wystąpić szkody materialne.

UWAGA

Oznacza, że w przypadku nie wzięcia pod uwagę odpowiednich informacji może wystąpić niezamierzony stan lub sytuacja.

W przypadku gdy występuje kilka niebezpieczeństw o różnym stopniu narażenia, to wszystkie są sygnalizowane jednym ostrzeżeniem odpowiadającym najwyższemu zagrożeniu. Ostrzeżenie o możliwości wystąpienia obrażeń ciała z odpowiednim symbolem, obejmuje również możliwość uszkodzenia mienia.

Kwalifikacje personelu

Urządzenia/system mogą być konfigurowane i używane wyłącznie na podstawie niniejszej dokumentacji. Do uruchamiania i obsługi urządzeń/systemu upoważniony jest tylko wykwalifikowany personel. Jako personel wykwalifikowany, w rozumieniu uwag zawartych w niniejszym opisie, rozumie się osoby, które mają uprawnienia do uruchamiania, dozoru, uziemiania i oznaczania urządzeń, systemów i obwodów zgodnie ze standardami i praktyką bezpieczeństwa.

Właściwe użycie wyrobów firmy Siemens

Prosimy o przestrzeganie następujących uwag:

OSTRZEŻENIE

Wyroby firmy Siemens mogą być używane wyłącznie w aplikacjach opisanych w katalogu i dokumentacji technicznej. Jeżeli wykorzystuje się produkty i podzespoły pochodzące od innych producentów, to muszą być one rekomendowane lub zatwierdzone przez firmę Siemens. Dla zapewnienia bezpiecznej pracy i uniknięcia problemów niezbędne są odpowiednie: transport, przechowywanie, instalacja, montaż, uruchamianie, obsługa i konserwacja. Należy zapewnić dozwolone warunki zewnętrzne. Należy stosować się do informacji podanych w dokumentacji technicznej.

Znaki zastrzeżone

Wszystkie nazwy identyfikowane znakiem ® są zarejestrowanymi znakami towarowymi Siemens AG. Inne oznaczenia występujące w niniejszym podręczniku mogą by znakami towarowymi, których wykorzystanie dla własnych celów przez osoby trzecie może naruszy prawa właścicieli.

Zrzeczenie się odpowiedzialności

Treść niniejszej publikacji sprawdzona została pod kątem zgodności opisanego sprzętu i oprogramowania ze stanem faktycznym. Niemniej jednak nie można założyć braku jakichkolwiek nieprawidłowości. Wyklucza się wszelką odpowiedzialność i gwarancję całkowitej prawdziwości zawartych informacji. Treść podręcznika poddana jest okresowo uzupełnieniom i poprawkom. Wszelkie konieczne korekty wprowadza się w kolejnych wydaniach.

Przedmowa

Przeznaczenie podręcznika

Seria S7-1200 jest rodziną programowanych sterowników logicznych (PLC – programmable logic controller) mogących spełniać funkcje sterujące w różnorodnych systemach automatyki. Zwarta konstrukcja, niewielkie koszty oraz bogata lista rozkazów czynią S7-1200 doskonałym urządzeniem sterującym, możliwym do zastosowania w wielu różnorodnych aplikacjach. Modele S7-1200 wraz z oprogramowaniem działającym w systemie Windows zapewniają elastyczność niezbędną podczas rozwiązywania praktycznie dowolnych zadań automatyzacji.

Ten podręcznik zawiera informacje o instalacji i programowaniu sterowników PLC S7-1200 i jest przeznaczony dla inżynierów, programistów, instalatorów oraz elektryków, którzy mają podstawową wiedzę z zakresu sterowników PLC.

Wymagana wiedza

Aby zapoznać się z zawartością tego podręcznika, konieczna jest podstawowa wiedza z zakresu sterowników PLC.

Zawartość dokumentacji

Omawiany podręcznik zawiera opis oprogramowania narzędziowego STEP 7 Basic V10.5 oraz rodziny sterowników S7-1200. Kompletny wykaz urządzeń należących do rodziny S7-1200 opisanych w tym podręczniku znajduje się w rozdziale „Dane techniczne”.

Certyfikaty, oznaczenia CE, C-Tick i inne normy

Więcej informacji jest podanych w rozdziale „Dane techniczne”.

Wsparcie techniczne

Uzupełnieniem dokumentacji są bogate zasoby informacji w Internecie

http://www.siemens..pl/simatic

W razie konieczności uzyskania pomocy w odpowiedzi na pytania techniczne, informacji o szkoleniach lub w celu złożenia zamówienia na produkty S7, prosimy o kontakt z lokalnym biurem handlowym firmy Siemens. Ponieważ lokalny przedstawiciel handlowy jest przeszkolony technicznie, a ponadto zna Państwa profil, procesy i przemysł, jak również produkty, które Państwo używacie, więc jest w stanie udzielić szybko i efektywnie odpowiedzi w sprawie dowolnych problemów, które mogą się pojawić.

Charakterystyka ogólna 1

1.1 Wiadomości wstępne o sterowniku PLC S7-1200

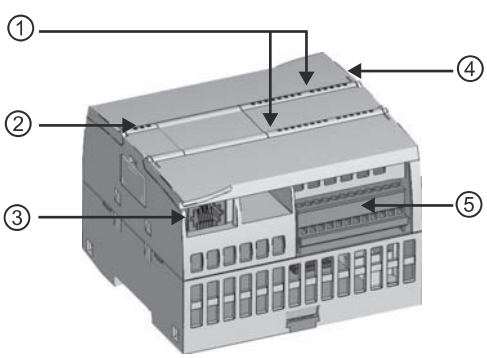
Seria programowanych sterowników logicznych (PLC) S7-1200 zapewnia elastyczność i skuteczność sterowania rozmaitymi urządzeniami w ramach wykonywania zadań automatyki. Zwarta konstrukcja, niewielkie koszty oraz bogata lista rozkazów czynią S7-1200 doskonałym narzędziem sterującym, odpowiednim do różnorodnych aplikacji.

CPU (jednostka centralna) składa się z modułu procesora, zintegrowanego zasilacza, obwodów wejściowych oraz obwodów wyjściowych umieszczonych w zwartej, plastikowej obudowie, co łącznie tworzy bardzo wydajny sterownik PLC. CPU zawiera logikę niezbędną do monitorowania i sterowania urządzeniami stanowiącymi elementy aplikacji. CPU monitoruje wejścia i steruje wyjściami zgodnie z oprogramowaniem przygotowanym przez użytkownika, które może zawierać logikę boolowską, zliczanie, operacje czasowe, złożone operacje arytmetyczne i komunikację z innymi inteligentnymi urządzeniami.

Kilka funkcji zabezpieczających pomaga chronić dostęp zarówno do CPU, jak i programu sterującego:

- Każda CPU jest chroniona hasłem umożliwiającym konfigurowanie dostępu do funkcji CPU.
- Za pomocą ochrony typu „know-how protection” można ukryć kod w ramach określonego bloku. *Por*. rozdział „Koncepcja programowania” w celu poznania szczegółów.

CPU jest wyposażana w port PROFINET umożliwiający komunikację poprzez sieć PROFINET. Dostępne są również moduły komunikacyjne pozwalające na łączność poprzez interfejsy RS485 lub RS232.



① Diody LED statusu wbudowanych portów I/O
② Diody LED statusu operacyjnego CPU
③ Złącze PROFINET
④ Slot karty pamięciowej (pod klapką)
⑤ Rozpinane złącza na kable

Różne modele CPU zapewniają różnorodne cechy i możliwości pozwalając tworzyć wydajne rozwiązania w rozmaitych aplikacjach. Szczegółowe informacje o poszczególnych modelach CPU są podane w rozdziale „Dane techniczne”.

Cecha	**CPU 1211C**	**CPU 1212C**	**CPU 1214C**
Wymiary (mm)	90 x 100 x 75		110 x 100 x 75
Pamięć użytkownika • Pamięć robocza • Pamięć ładowania • Pamięć nieulotna	 • 25 KB • 1 MB • 2 KB		 • 50 KB • 2 MB • 2 KB
Lokalne porty I/O • Cyfrowe • Analogowe	 • 6 wejść/4 wyjścia • 2 wejścia	 • 8 wejść/6 wyjść • 2 wejścia	 • 14 wejść/10 wyjść • 2 wejścia
Rozmiar obrazu procesu	1024 bajtów (wejścia) i 1024 bajtów (wyjścia)		
Moduły rozszerzeń	Brak	2	8
Płytki sygnałowe	1		
Moduły komunikacyjne	3 (dołączane po lewej stronie)		
Szybkie liczniki • Jednofazowe • Kwadraturowe	3 • 3 dla 100 kHz • 3 dla 80 kHz	4 • 3 dla 100 kHz 1 dla 30 kHz • 3 dla 80 kHz 1 dla 20 kHz	6 • 3 dla 100 kHz 3 dla 30 kHz • 3 dla 80 kHz 3 dla 20 kHz
Wyjścia impulsowe	2		
Karta pamięci	Karta pamięci SIMATIC (opcja)		
Czas retencji zegara czasu rzeczywistego	10 dni typowo / 6 dni minimalnie przy 40 stopniach		
PROFINET	1 port komunikacyjny Ethernet		
Szybkość wykonywania operacji arytmetycznych	18 µs/instrukcja		
Szybkość wykonywania operacji boolowskich	0,1 µs/instrukcja		

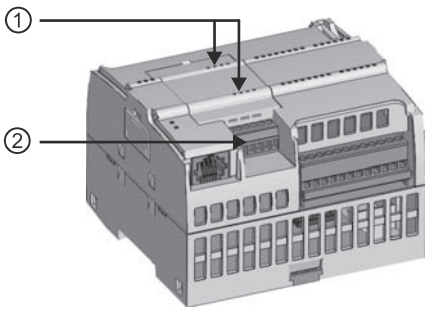
Rodzina sterowników S7-1200 zawiera wiele typów modułów rozszerzeń i płytek sygnałowych służących do rozszerzania możliwości CPU. Możliwe jest również instalowanie modułów komunikacyjnych obsługujących inne protokoły komunikacyjne. Szczegółowe informacje o poszczególnych modułach podano w rozdziale „Dane techniczne”.

Moduł		Tylko wejście	Tylko wyjście	Wejście i wyjście
Moduł rozszerzeń (SM)	Cyfrowy	8 × wejście DC	8 × wyjście DC 8 × wyjście przekaźnikowe	8 × wejście DC / 8 × wyjście DC 8 × wejście DC / 8 × wyjście przekaźnikowe
		16 × wejście DC	16 × wyjście DC 16 × wyjście przekaźnikowe	16 × wejście DC / 16 × wyjście DC 16 × wejście DC / 16 × wyjście przekaźnikowe
	Analogowy	4 × wejście analogowe	2 × wyjście analogowe	4 × wejście analogowe / 2 × wyjście analogowe
Płytka sygnałowa (SB)	Cyfrowa	–		2 × wejście DC / 2 × wyjście DC
	Analogowa	–	1 × wyjście analogowe	–
Moduł komunikacyjny (CM) • RS485 • RS232				

1.2 Płytki sygnałowe

Płytki sygnałowe (SB) pozwalają dodawać do CPU porty I/O. Można dołączyć jedną SB z cyfrowymi lub analogowymi portami I/O. SB jest dołączana od strony frontowej CPU.

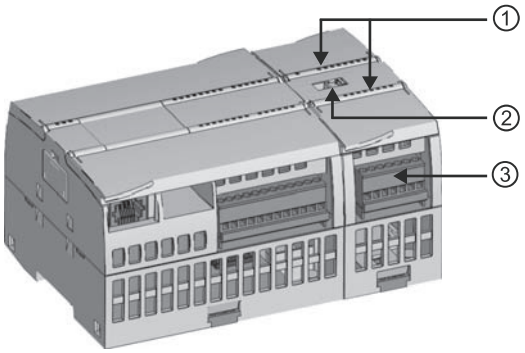
- SB z 4 cyfrowymi portami I/O (2 × wejście DC i 2 × wyjście DC)
- SB z 1 wyjściem analogowym



① Diody LED statusu SB
② Rozpinane złącza na kable

1.3 Moduły rozszerzeń

Moduły rozszerzeń (SM) służą do zwiększania funkcjonalności CPU. Są dołączane z prawej strony CPU.

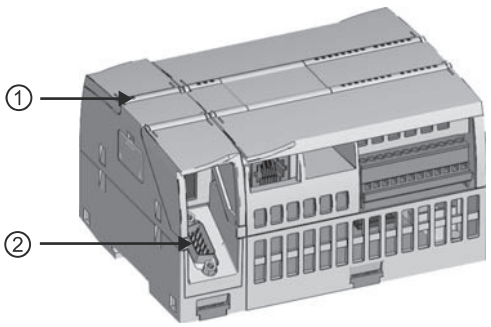


① Diody LED statusu portów I/O modułu rozszerzeń
② Gniazdo magistrali
③ Rozpinane złącza na kable

1.4 Moduły komunikacyjne

W celu zwiększenia funkcjonalności systemu rodzina sterowników S7-1200 zawiera moduły komunikacyjne (CM). Są dostępne dwa typy modułów komunikacyjnych: RS232 i RS485.

- CPU obsługuje do 3 modułów komunikacyjnych
- Każdy moduł komunikacyjny jest dołączany z lewej strony CPU (lub z lewej strony innego CM, który jest już dołączony do CPU)



① Diody LED statusu modułu komunikacyjnego
② Złącze komunikacyjne

1.5 Oprogramowanie Totally Integrated Automation (TIA) Portal

Jest to przyjazne dla użytkownika środowisko służące do projektowania, edytowania i monitorowania logiki niezbędnej do sterowania aplikacją.

TIA Portal dostarcza narzędzi do zarządzania i konfigurowania wszystkich urządzeń w systemie, takich jak sterowniki PLC i urządzenia HMI. Elementem składowym TIA Portal jest STEP 7 Basic oferujący, w celu zapewnienia wygody i wydajności projektowania oprogramowania sterującego, obsługę dwóch języków programowania (LAD i FBD). TIA Portal zawiera również narzędzia do podłączania i konfigurowania urządzeń HMI.

W celu ułatwienia znalezienia potrzebnej informacji, TIA Portal wyposażono w rozbudowany system informacyjny *online*. TIA Portal umożliwia dwa sposoby prezentacji projektu: zorientowaną na strukturę projektu (widok Projektu – *Project view*) oraz prezentację zorientowaną na zadania (widok Portalu – *Portal view*).

W celu zainstalowania oprogramowania TIA Portal należy włożyć płytę CD do napędu CD-ROM komputera. Kreator instalacji wystartuje automatycznie i poprowadzi użytkownika przez cały proces instalacji. W celu uzyskania szczegółowych informacji dotyczących instalacji oprogramowania TIA Portal, należy się zapoznać z zawartością pliku *readme*.

UWAGA

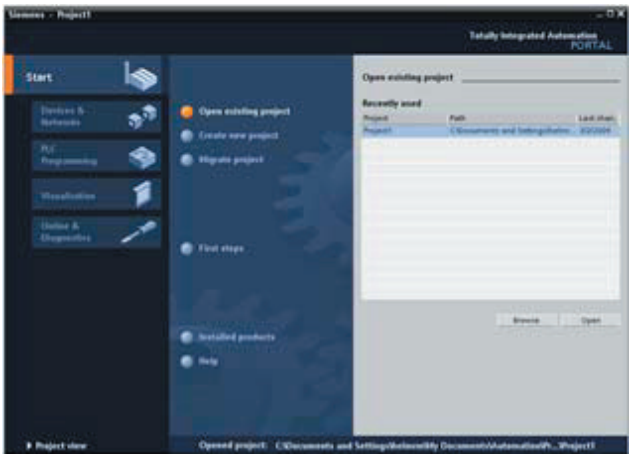
W celu zainstalowania oprogramowania TIA Portal na komputerze z systemem operacyjnym Windows 2000, Windows XP lub Windows Vista użytkownik musi mieć uprawnienia administratora.

1.5.1 Różne sposoby prezentacji projektu

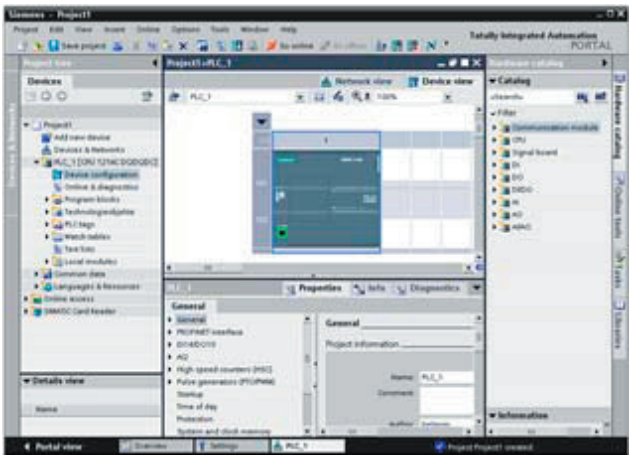
W celu zwiększenia wygody obsługi TIA Portal oferuje dwa sposoby prezentacji narzędzi: zestaw portali zorientowany na funkcjonalność narzędzi (widok Portalu) lub – zorientowaną na strukturę projektu – wizualizację jego elementów składowych (widok Projektu). Użytkownik sam wybiera, która wizualizacja pozwala mu pracować bardziej wydajnie. Jedno kliknięcie pozwala przełączać między widokiem Portalu i widokiem Projektu.

Widok Portalu przedstawia funkcjonalny obraz zadań projektu i organizuje funkcje narzędzi zgodnie z zadaniami jakie mają być wykonane, takimi jak tworzenie konfiguracji podzespołów sprzętowych i sieci.

Użytkownik może łatwo decydować jak postępować i jakie zadania wybierać.



Widok Projektu umożliwia dostęp do wszystkich elementów projektu. Mając te elementy zebrane w jednym miejscu, użytkownik ma łatwy dostęp do każdego fragmentu swojego projektu. Projekt obejmuje wszystkie elementy, które zostały stworzone lub ukończone.



1.5.2 Pomoc na życzenie

Szybkie znajdowanie odpowiedzi na pojawiające się pytania

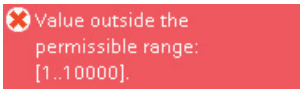
W celu ułatwienia szybkiego i wydajnego rozwiązywania kłopotów, TIA Portal wyposażono w inteligentną, kontekstową pomoc:

- Pole wejściowe jest skojarzone z rozwijanym oknem, które ma pomóc przy wprowadzeniu właściwej informacji (poprawne zakresy i typy danych) dla tego pola. Na przykład, jeśli użytkownik wprowadzi błędną wartość, to zostanie wyświetlone pole tekstowe podpowiadające zakres poprawnych wartości.
- Niektóre podpowiedzi interfejsu (dotyczące na przykład instrukcji), w celu dostarczenia dodatkowych informacji, pojawiają się kaskadowo. Pewne podpowiedzi kaskadowe podają linki do określonych tematów systemu informacyjnego *online* (*online help*).

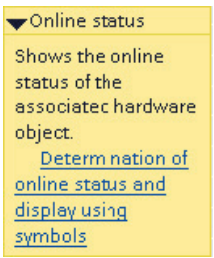
Ponadto TIA Portal zawiera obszerny system informacyjny w pełni opisujący funkcjonalność narzędzi SIMATIC.

Rozwijane pola pomocy i podpowiedzi kaskadowe

Pola wejściowe różnych okien dialogowych i kart zadań są wyposażone w mechanizm reakcji w formie okna tekstowego, które rozwija się i informuje użytkownika o wymaganym zakresie lub typie danych.



Elementy interfejsu programowego dostarczają wskazówek wyjaśniających funkcjonalność elementu. Niektóre elementy, takie jak ikony „Open” lub „Save” nie wymagają dodatkowych komentarzy. Jednakże, pewne elementy są wyposażone w mechanizm wyświetlający dodatkowy opis tego elementu. Te dodatkowe informacje pojawiają się kaskadowo w oknie danej podpowiedzi. (Czarny trójkąt obok podpowiedzi oznacza, że są dostępne dodatkowe informacje.)



Wskazówka jest wyświetlana w oknie wiszącym nad elementem interfejsu programowego. W celu wyświetlenia dodatkowej informacji trzeba umieścić kursor nad polem podpowiedzi. Niektóre kaskadowe podpowiedzi zawierają linki to odpowiednich tematów systemu informacyjnego. Kliknięcie na link powoduje wyświetlenie danego tematu.

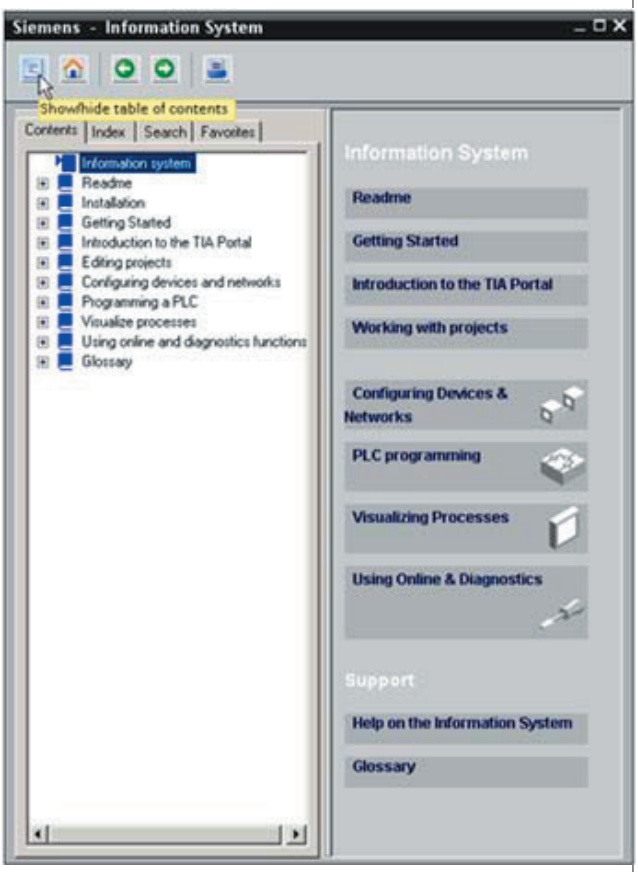
System pomocy technicznej (*information system*)

TIA Portal ma wbudowany system pomocy *online*, w którym można znaleźć opisy zainstalowanych produktów systemu SIMATIC TIA. System informacyjny zawiera również informacje referencyjne i przykłady. W celu uruchomienia systemu informacyjnego należy wykonać następujące czynności:

- Z widoku Portalu – wybrać portal Start i kliknąć komendę „Help”.
- Z widoku Projektu – z menu Help wybrać komendę „Show help”.
- Z kaskadowo rozwijanego okna podpowiedzi – kliknąć link powodujący wyświetlenie dodatkowych informacji na dany temat.

System pomocy otwiera się w oknie, które nie zasłania obszarów roboczych.

W celu wyświetlenia spisu treści i wyłączenia dokowania okna pomocy trzeba nacisnąć przycisk „Show/hide contents”. Można wówczas zmienić rozmiar okna. Zakładki „Contents” lub „Index” służą do przeszukiwania systemu informacyjnego według tematów lub słów kluczowych.

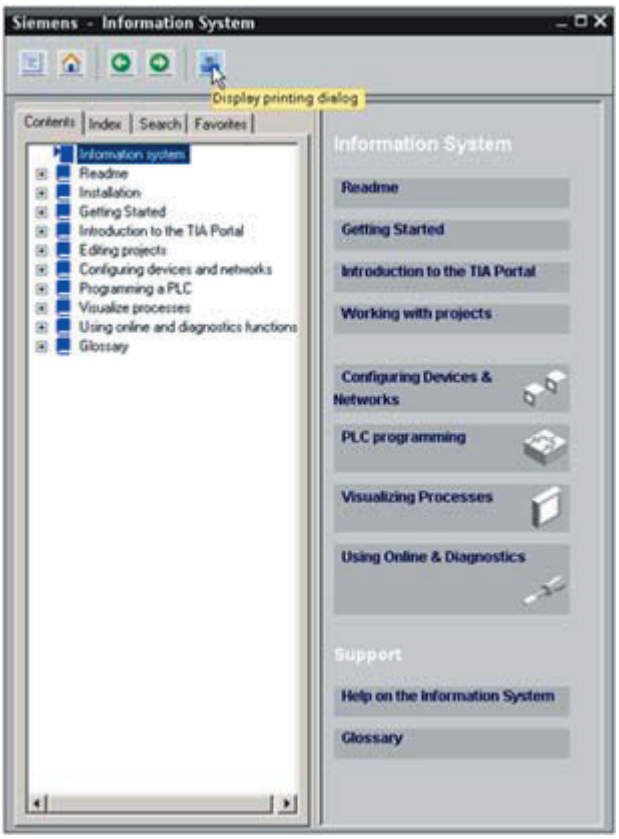
Okno pomocy (domyślnie)	Okno pomocy z wyświetlonym spisem treści
	

UWAGA

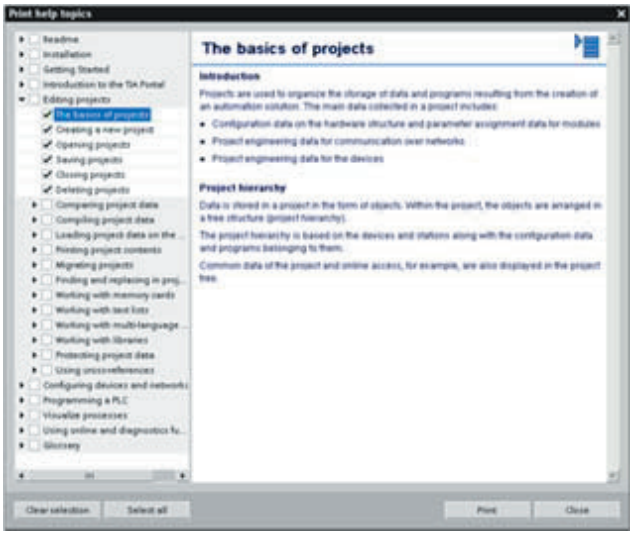
Jeżeli okno TIA Portal ma maksymalny rozmiar, to kliknięcie na przycisk „Show/hide contents” nie wyłączy dokowania okna pomocy. W celu wyłączenia dokowania okna pomocy trzeba nacisnąć przycisk „Restore down” w oknie TIA Portal. Można wówczas przesuwać i zmieniać rozmiar okna pomocy.

Drukowanie tematów z systemu informacyjnego

W celu dokonania wydruku z sytemu informacyjnego należy kliknąć przycisk „Print" w oknie pomocy.

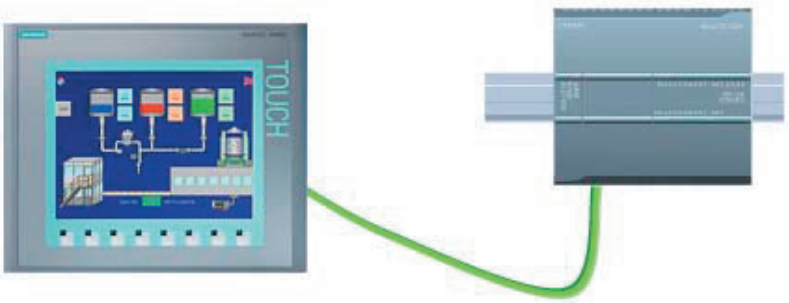


Okno dialogowe „Print" pozwala wybrać temat, który ma zostać wydrukowany. Trzeba się upewnić, że temat jest wyświetlany na panelu. Można wówczas wybrać do wydrukowania dowolny inny temat. Kliknięcie przycisku „Print" powoduje przesłanie wybranych tematów do drukarki.



1.6 Panele operatorskie

W miarę jak systemy wizualizacji stają się standardowym elementem wielu maszyn i urządzeń, SIMATIC HMI Basic Panels oferują urządzenia z ekranami dotykowymi, które umożliwiają podstawową obsługę operatorską i monitorowanie zadań.



Cecha	KTP1000 Basic color	TP1500 Basic color
Ekran • Rozmiar • Rozdzielczość	TFT, 256 kolorów • 10,4” • 640 x 480	TFT, 256 kolorów • 15,0” • 1024 x 768
Elementy sterujące	Ekran dotykowy + 8 klawiszy dotykowych	Ekran dotykowy
Stopień zabezpieczenia	IP65	IP65
Interfejs	PROFINET	PROFINET
Funkcjonalność • Tagi • Ekrany procesów • Alarmy • Krzywe trendów	 • 256 • 50 • 200 • 25	 • 256 • 50 • 200 • 25
Wymiary [mm] • Front obudowy (W x H x D) • Wycięcie montażowe (W x H)	 • 335 x 275 x 61 • 310 x 248	 • 400 x 310 x 60 • 367 x 289

Instalacja 2

Urządzenia z serii S7-1200 zostały zaprojektowane tak, by były łatwe do instalacji. Można je instalować na panelu montażowym lub na standardowej szynie montażowej i mogą być ułożone poziomo lub pionowo. Małe wymiary S7-1200 pozwalają efektywnie wykorzystywać dostępne miejsce.

⚠	**OSTRZEŻENIE**
Sterowniki PLC rodziny SIMATIC S7-1200 są sterownikami kompaktowymi. Wymaga się by były instalowane w obudowach, szafkach lub sterowniach elektrycznych. Dostęp do tych obudów, szafek i sterowni elektrycznych powinien mieć wyłącznie upoważniony personel. Nieprzestrzeganie podczas instalacji podanych wymagań może śmierć, poważne obrażenia ciała i/lub uszkodzenie mienia. Podczas instalacji sterowników PLC S7-1200 zawsze należy przestrzegać podanych wymagań.	

Separacja urządzeń S7-1200 od źródeł ciepła, wysokiego napięcia oraz zakłóceń elektrycznych

Generalną zasadą jest, aby zawsze rozdzielać urządzenia generujące wysokie napięcia i zakłócenia od cyfrowych urządzeń niskonapięciowych, takich jak np. sterowniki S7-1200.

Podczas projektowania zabudowy S7-1200 na panelu, należy uwzględnić elementy wytwarzające ciepło i umieścić urządzenia elektroniczne w obszarach chłodzonych szafy montażowej. Unikanie podwyższonej temperatury przedłuża czas pracy wszystkich urządzeń elektronicznych.

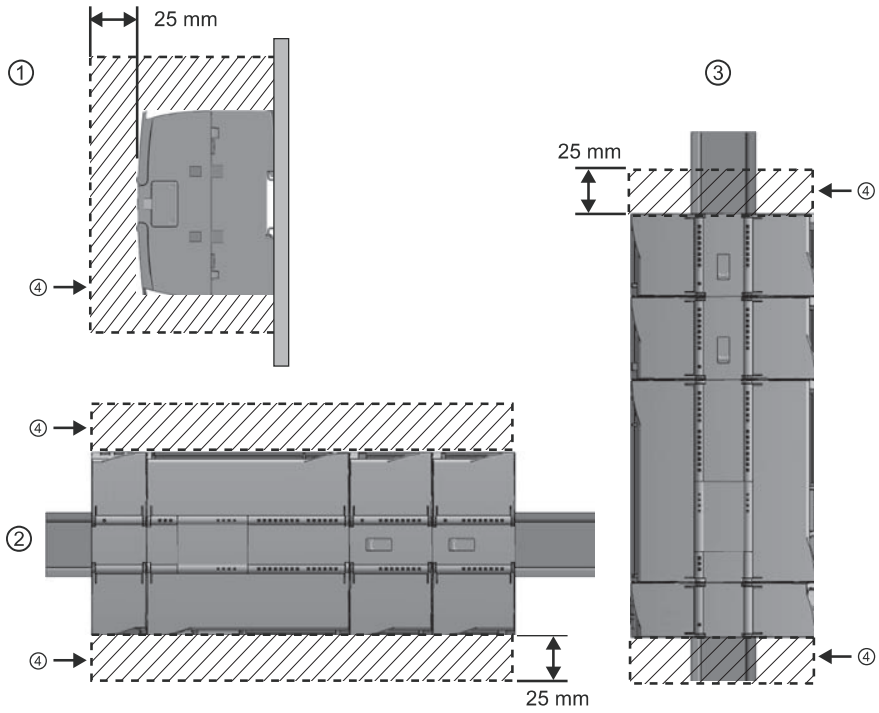
Należy również rozważyć prowadzenie okablowania urządzeń na panelu. Trzeba unikać prowadzenia niskonapięciowych przewodów sygnałowych i kabli przesyłu danych obok kabli zasilających i wysokoenergetycznych, a także generujących zakłócenia impulsowe.

Zapewnienie właściwej przestrzeni dla chłodzenia i okablowania

Urządzenia S7-1200 są zaprojektowane w taki sposób, że wystarczające jest naturalne chłodzenie konwekcyjne. Należy jedynie zapewnić wolną przestrzeń wynoszącą co najmniej 25 mm nad i pod sterownikiem. Trzeba także zapewnić wolną przestrzeń montażową o głębokości co najmniej 25 mm.

⚠	**OSTROŻNIE**
Przy montażu pionowym, maksymalna dopuszczalna temperatura otoczenia jest mniejsza o 10°C. Należy tak montować system S7-1200 by CPU znajdowała się na samym dole zestawu.	

Podczas planowania zabudowy systemu S7-1200, należy zapewnić wystarczającą przestrzeń dla wykonania okablowania oraz poprowadzenia kabli przesyłu danych.



① Widok z boku

② Montaż poziomy

③ Montaż pionowy

④ Niezbędny odstęp

Bilans mocy

Wszystkie sterowniki S7-1200 mają wewnętrzny zasilacz, który zasila CPU, moduły rozszerzeń, płytkę sygnałową i moduły komunikacyjne oraz inne urządzenia użytkownika wymagające zasilania napięciem 24 VDC.

W rozdziale „Dane techniczne” podano informacje o wydajności prądowej wewnętrznego zasilacza CPU dla napięcia 5 VDC oraz mocy pobieranej przez moduły rozszerzeń, płytkę sygnałową i moduły komunikacyjne z napięcia 5 VDC. W części „Obliczanie bilansu mocy” podano sposób w jaki można obliczyć moc (lub prąd) dostępną na wyjściu CPU przy konkretnej konfiguracji systemu.

CPU zapewnia napięcie 24 VDC do zasilania czujników, cewek przekaźników w modułach rozszerzeń lub innych urządzeń. W rozdziale „Dane techniczne” podano informacje na temat wydajności zasilaczy wewnętrznych 24 VDC poszczególnych CPU rodziny S7-1200.

Jeżeli w systemie ze sterownikiem S7-1200 zastosowano dodatkowy, zewnętrzny zasilacz 24 VDC, to należy się upewnić, czy nie jest połączony równolegle z wewnętrznym zasilaczem czujników CPU. Zaleca się, by w celu zwiększenia odporności na zakłócenia, połączyć ze sobą zaciski wspólne (M) różnych zasilaczy.

OSTRZEŻENIE

Połączenie równoległe zewnętrznego zasilacza 24 VDC z wewnętrznym zasilaczem 24 VDC sterownika może doprowadzić do konfliktu między tymi zasilaczami, ponieważ każdy zasilacz będzie usiłował stabilizować swoje własne napięcie wyjściowe.

Połączenie takie wpłynie niekorzystnie na żywotność pracy zasilaczy lub doprowadzi do natychmiastowego uszkodzenia jednego z nich, co może spowodować nieprzewidywalne zachowanie PLC w układzie. Nieprzewidywalne zachowanie PLC grozi śmiercią lub poważnym zranieniem personelu i/lub zniszczeniem mienia.

Zasilac z wewnętrzny S7-1200 i dowolny zasilacz zewnętrzny powinny zasilać różne obwody systemu.

Niektóre fragmenty systemu S7-1200 zasilane napięciem 24 VDC są ze sobą połączone poprzez wspólne łączenie wielu wyprowadzeń M. Na przykład, następujące układy są ze sobą połączone, gdy w karcie katalogowej są oznaczone jako „nieizolowane": zasilacz wewnętrzny CPU 24 VDC, wejście zasilające uzwojenie przekaźnika SM lub zasilacz nieizolowanych wejść analogowych. Wszystkie nieizolowane zaciski M muszą być połączone do tego samego zewnętrznego potencjału odniesienia.

OSTRZEŻENIE

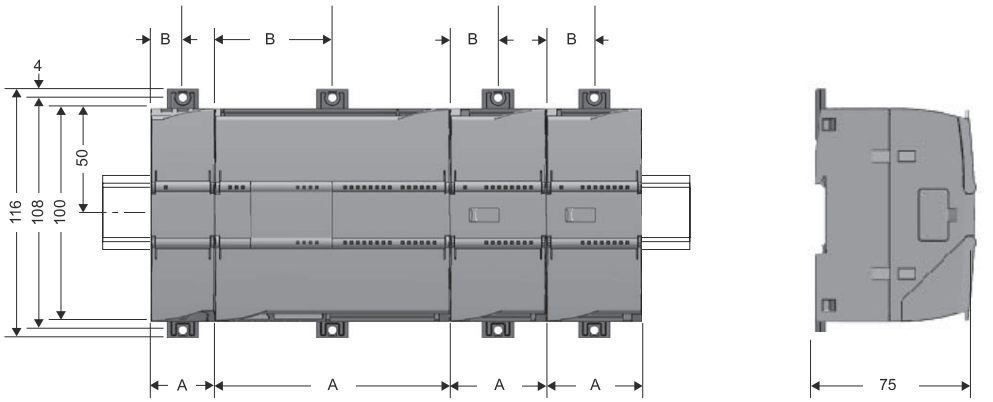
Połączenie nieizolowanych wyprowadzeń M do różnych potencjałów odniesienia spowoduje nieplanowany przepływ prądów mogący doprowadzić do uszkodzenia lub nieprzewidywalnego zachowania PLC i podłączonych do niego urządzeń.

Nie zastosowanie się do podanych tu zaleceń może spowodować uszkodzenia i nieprzewidywalne działanie, które grozi śmiercią lub poważnym zranieniem personelu i/lub zniszczeniem mienia.

Koniecznie należy się upewnić, że nieizolowane zaciski M systemu S7-1200 są podłączone do tego samego potencjału odniesienia.

2.1 Instalacja i deinstalacja modułów S7-1200

Wymiary montażowe



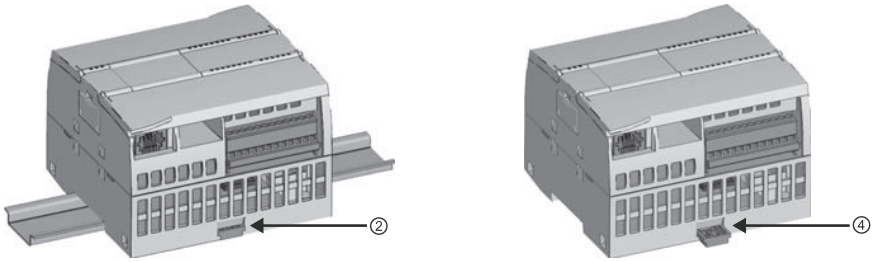
Urządzenia S7-1200		Szerokość A	Szerokość B
CPU:	CPU 1211C i CPU 1212C	90 mm	45 mm
	CPU 1214C	110 mm	55 mm
Moduły rozszerzeń:	8 i 16 punktów DC i przekaźnikowych (8I, 16I, 8Q, 16Q, 8I/8Q) 2 i 4 punkty analogowe (4AI, 4AI/4AQ, 2AQ)	45 mm	22,5 mm
	16I/16Q przekaźnikowe (16I/16Q)	70 mm	35 mm
Moduły komunikacyjne:	CM 1241 RS232 i CM 1241 RS485	30 mm	15 mm

Moduły CPU, SM i CM są dostosowane do montażu na szynie DIN do montażu naściennego. W celu montażu na szynie należy wykorzystać zaczepy montażowe znajdujące się na module. Te zaczepy można także zatrzasnąć w pozycjach rozsuniętych, co pozwala bezpośrednio zmocować moduł za pomocą śrub na ścianie, drzwiach szafy sterującej lub płycie montażowej. Wewnętrzny wymiar otworu dla zaczepu DIN modułu wynosi 4,3 mm.

Nad i pod modułem należy pozostawić wolną przestrzeń o szerokości co najmniej 25 mm dla umożliwienia swobodnego przepływu powietrza chłodzącego.

Instalacja i deinstalacja modułów S7-1200

Sterownik S7-1200 może być łatwo montowany na standardowej szynie DIN lub bezpośrednio na płycie montażowej. W celu montażu na szynie należy wykorzystać zaczepy montażowe znajdujące się na module. Te zaczepy można także zatrzasnąć w pozycjach rozsuniętych, co pozwala bezpośrednio zmocować moduł za pomocą śrub na płycie montażowej.



① ③

① Instalacja na szynie DIN

② Zaczep montażowy szyny DIN w pozycji zablokowanej

③ Instalacja na płycie czołowej

③ Zaczep montażowy w pozycji rozłożonej do montażu na płycie czołowej

Przed instalacją lub usunięciem jaki egokolwiek urządzenia elektrycznego należy się upewnić, że jego źródło zasilania jest wyłączone. Jak również, że zasilanie jakichkolwiek urządzeń z nim współpracujących także jest wyłączone.

OSTRZEŻENIE

Próba instalowania lub demontażu S7-1200 lub współpracujących urządzeń z załączonym napięciem zasilania może doprowadzić do porażenia elektrycznego lub niewłaściwego zadziałania urządzenia.

Nie wyłączenie zasilania S7-1200 i współpracujących urządzeń podczas procedury instalacji lub deinstalacji może spowodować śmierć lub poważne zranienie personelu i/lub zniszczenie mienia.

Należy zawsze stosować odpowiednie procedury bezpiecznej pracy oraz sprawdzić, czy napięcie zasilania S7-1200 i współpracujących urządzeń jest wyłączone przed instalacją lub demontażem CPU lub jakiegokolwiek modułu rozszerzeń.

Zawsze należy się upewnić czy do instalacji lub wymiany urządzenia S7-1200 jest używany właściwy moduł lub kompatybilne urządzenie.

OSTRZEŻENIE

Niewłaściwe zainstalowanie dowolnego modułu S7-1200 może spowodować nieprzewidywalne działanie programu w S7-1200.

Wymiana urządzenia rodziny S7-1200 na niewłaściwy model lub niewłaściwie przeprowadzona procedura wymiany (błędna orientacja lub kolejność modułów) może spowodować, poprzez nieprzewidywalne działanie sprzętu, śmierć lub poważne zranienie personelu i/lub zniszczenie mienia

Należy zawsze wymieniać urządzenia systemu S7-1200 na taki sam model oraz zapewnić właściwą orientację i położenia w układzie.

2.1.1. Instalacja i deinstalacja CPU

Instalacja

CPU można instalować na płycie montażowej lub szynie DIN.

UWAGA

Do CPU należy dołączyć moduły komunikacyjne i instalować całość jako jeden zespół. Moduły rozszerzeń należy instalować oddzielnie po zamontowaniu CPU.

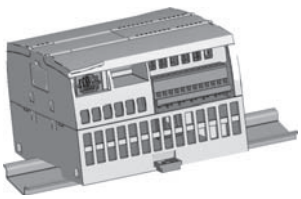
W celu zamontowania CPU na panelu należy wykonać następujące kroki:

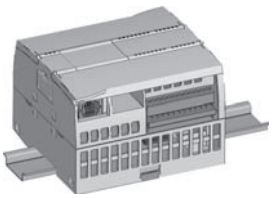
1. Wyznaczyć, wywiercić i nagwintować otwory montażowe w (M4 lub USA standard No 8) zgodnie z podanymi wymiarami.
2. Rozciągnąć zaczepy montażowe z modułu. Upewnić się, że zaczepy montażowe szyny DIN górne i dolne są na wysuniętych pozycjach.
3. Przymocować moduł do panelu stosując śruby umieszczone z zaczepach.

UWAGA

Jeżeli system jest narażony na silne wibracje lub jest montowany pionowo, to instalacja S7-1200 na panelu montażowym zapewni lepszym poziom zabezpieczenia.

W celu instalacji CPU na szynie DIN należy wykonać następujące kroki:

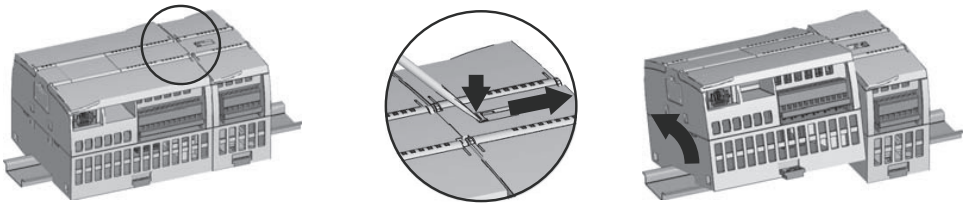




1. Zainstalować szynę DIN. Szyny należy mocować do płyty montażowej co 75 mm.
2. Zawiesić CPU na górnej krawędzi szyny DIN.
3. Wyciągnąć z dolnej części CPU zaczep montażowy szyny DIN tak, by CPU pasował do szyny.
4. Obrócić CPU do dołu i umiejscowić na szynie.
5. Wepchnąć zaczepy tak, by CPU został zatrzaśnięty na szynie.

Deinstalacja

W celu przygotowania CPU do deinstalacji, należy wyłączyć zasilanie CPU, a także odłączyć złącza I/O, przewody i kable. CPU wraz z przyłączonymi modułami komunikacyjnymi demontuje się jako jedną całość. Wszystkie moduły rozszerzeń powinny pozostać zainstalowane.



Jeżeli do CPU jest przyłączony moduł rozszerzeń, to należy cofnąć złącze magistrali:

1. Umieścić wkrętak za języczkiem znajdującym się na górze modułu rozszerzeń.
2. Przycisnąć języczek do dołu by rozpiąć złącze w CPU.
3. Przesunąć języczek w prawo do samego końca.

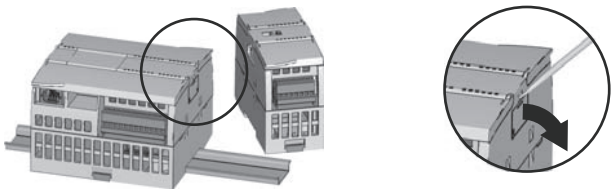
Demontaż CPU:

1. Wyciągnąć zaczep mocujący szyny DIN tak by odłączyć CPU od szyny.
2. Obrócić CPU w górę, jednocześnie odsuwając od szyny i wyjąć CPU z systemu.

2.1.2 Instalacja i deinstalacja modułu rozszerzeń

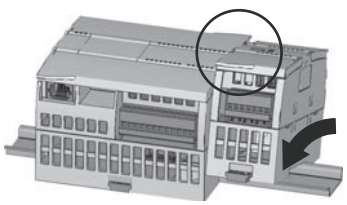
Instalacja

Moduł SM instaluje się po zamontowaniu CPU.



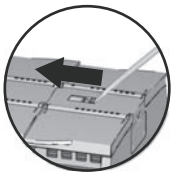
Usunąć pokrywę złącza po prawej stronie CPU.

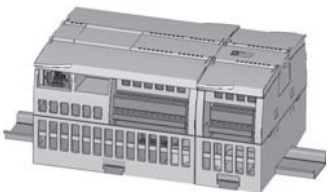
1. Włożyć wkrętak do szczeliny powyżej pokrywy.
2. Łagodnie podważyć pokrywę na zewnątrz, od strony górnej krawędzi i wyjąć pokrywę. Zachować pokrywę do ponownego użycia.



Umieścić SM obok CPU.

1. Zawiesić SM na górnej krawędzi szyny DIN.
2. Wyciągnąć dolny zaczep montażowy szyny DIN tak, by SM pasował do szyny.
3. Obrócić SM do dołu i umiejscowić na szynie obok CPU, a następnie wepchnąć zaczepy tak, by SM został zatrzaśnięty na szynie.





Wysuwanie złącza magistrali.

1. Umieścić wkrętak obok języczka na górze SM.
2. Przesunąć języczek w lewo do samego końca, by wysunąć złącze magistrali do CPU.

Wysunięcie złącza magistrali łączy SM zarówno mechaniczne, jak i elektrycznie.

Deinstalacja

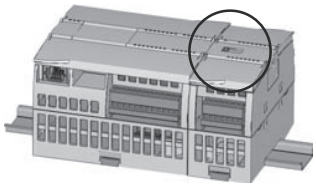
Możliwa jest deinstalacja dowolnego SM bez usuwania CPU lub innych SM. W celu przygotowania SM do deinstalacji, należy wyłączyć zasilanie CPU, a także odłączyć od SM złącza I/O, przewody i kable.

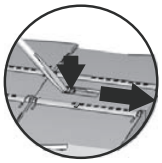
Należy cofnąć złącze magistrali.

1. Umieścić wkrętak za języczkiem znajdującym się na górze modułu rozszerzeń.
2. Przycisnąć języczek do dołu by rozpiąć złącze w CPU.
3. Przesunąć języczek w prawo do samego końca.

Należy cofnąć złącze magistrali.

1. Umieścić wkrętak za języczkiem znajdującym się na górze modułu rozszerzeń.
2. Przycisnąć języczek do dołu by rozpiąć złącze w CPU.
3. Przesunąć języczek w prawo do samego końca.

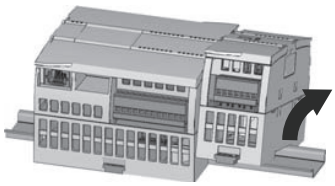




Jeżeli z prawej strony znajduje się inny SM, to tę procedurę należy zastosować również do niego.

Usuwanie SM

1. Wyciągnąć zaczep mocujący szyny DIN tak by odłączyć SM od szyny.
2. Obrócić SM w górę, jednocześnie odsuwając od szyny. Wyjąć SM z systemu.
3. Jeżeli jest to konieczne, to w celu uniknięcia zanieczyszczenia, należy zasłonić złącze magistrali CPU.



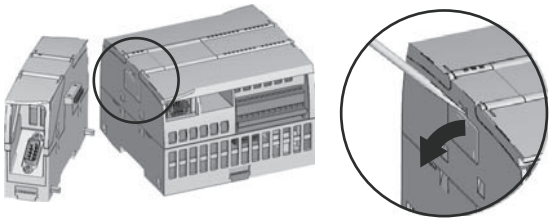
2.1.3 Instalacja i deinstalacja modułu komunikacyjnego

Instalacja

CM należy dołączyć do CPU i całość instalować do szyny DIN lub panelu jako jeden zespół.

Usunąć osłonę magistrali znajdującą się po lewej stronie CPU:

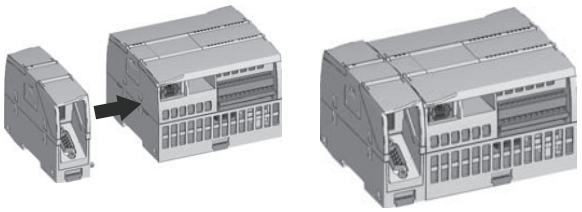
1. Włożyć wkrętak do szczeliny powyżej pokrywy magistrali.
2. Łagodnie podważyć pokrywę na zewnątrz, od strony górnej krawędzi.



Wyjąć pokrywę. Zachować pokrywę do ponownego użycia.

Połączyć moduły:

1. Ustawić naprzeciw siebie wtyk i gniazdo magistrali oraz scentrować kołki prowadzące z otworami w CPU.
2. Docisnąć moduły do siebie, aż kołki zatrzasną się w swoich otworach.



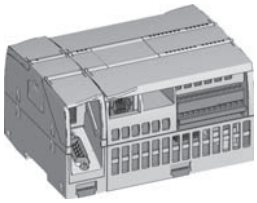
Instalacja modułów na szynie DIN lub na panelu.

1. W celu wykonania montażu na szynie DIN należy się upewnić, czy – w zestawie CPU z dołączonymi CM – górny zaczep szyny DIN znajduje się w pozycji zatrzaśniętej (wewnętrznej), a dolne zaczepy szyny DIN są wysunięte całkowicie na zewnątrz.
2. Zainstalować CPU z dołączonymi CM zgodnie z opisem przedstawionym w rozdziale „Instalacja i deinstalacja CPU”.
3. Po zainstalowaniu urządzeń na szynie DIN, w celu ich zablokowania należy przesunąć dolne zaczepy CPU i CM w pozycje zatrzaśnięte.

W celu wykonania montażu na panelu należy się upewnić, że zaczepy szyny DIN są maksymalnie wysunięte.

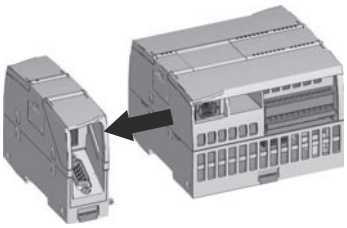
Deinstalacja

CPU wraz z dołączonym CM należy deinstalować z szyny DIN lub panelu jako jeden zespół.



Przygotowanie do deinstalacji

1. Wyłączyć zasilanie CPU.
2. Odłączyć od CPU i CM wszystkie złącza I/O, przewody i kable.
3. W przypadku instalacji na szynie DIN wyciągnąć zaczepy mocujące szyny DIN CPU i CM na zewnątrz.
4. Odłączyć CPU i CM od szyny lub panelu.



Odłączyć CM.

1. Pewnie chwycić CPU i CM.
2. Odciągnąć je od siebie.

Do rozdzielania modułów nie należy używać żadnych narzędzi, ponieważ można je uszkodzić.

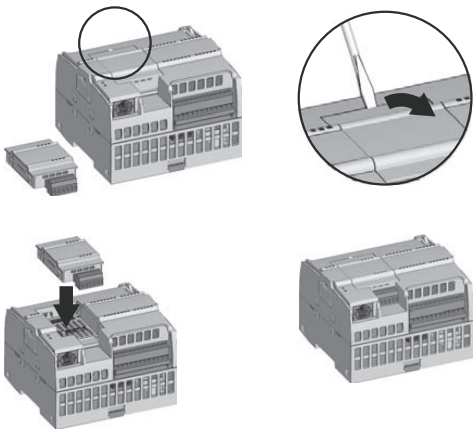
2.1.4 Instalacja i deinstalacja płytki sygnałowej

Instalacja

CPU należy przygotować do instalacji SB poprzez odłączenie zasilania i zdjęcie osłony chroniącej złącza.

W celu zainstalowania SB należy wykonać następujące kroki:

1. Umieścić wkrętak w szczelinie na górze CPU, z tylnej strony osłony.
2. Łagodnie podważyć pokrywę do góry i zdjąć ją z CPU.
3. Od góry wsunąć SB na jej pozycję montażową.
4. Pewnie docisnąć SB, aż do zatrzaśnięcia na pozycji.
5. Zamontować osłony złączy.

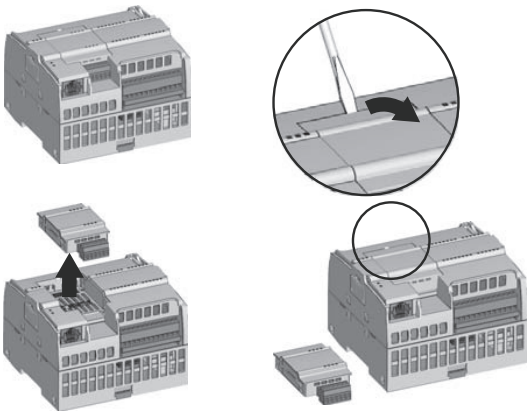


Deinstalacja

CPU należy przygotować do deinstalacji SB poprzez odłączenie zasilania i zdjęcie osłony chroniącej złącza.

W celi deinstalacji SB należy wykonać następujące kroki:

1. Umieścić wkrętak w szczelinie na górze SB.
2. Łagodnie podważyć SB do góry, by odłączyć ja od CPU.
3. Wyjąć SB prosto do góry z jej pozycji montażowej w górnej części CPU.
4. Założyć osłonę SB.
5. Założyć osłonę złączy w CPU.



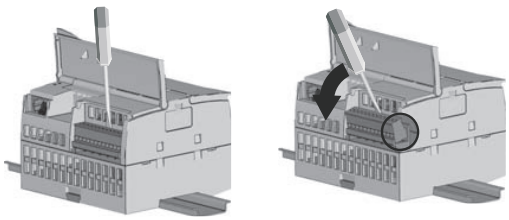
2.1.5 Odłączanie i reinstalacja złącza listwy zaciskowej S7-1200

W celu ułatwienia wykonania okablowania, moduły CPU, SB i SM są wyposażone w odłączalne złącza. W celu przygotowania systemu do odłączenia złącza listwy zaciskowej należy:

- Odłączyć zasilanie od CPU.
- Otworzyć pokrywę powyżej złącza.

W celu odłączenia złącza należy wykonać następujące kroki:

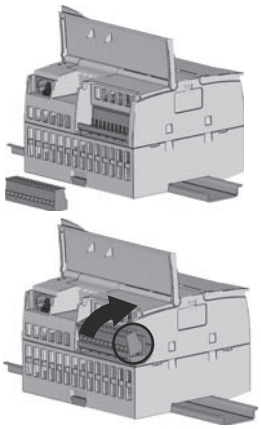
1. Obejrzeć szczyt złącza i zlokalizować szczelinę dla końcówki wkrętaka.
2. W szczelinie umieścić końcówkę wkrętaka.
3. Łagodnie podważyć górną część złącza w kierunku od CPU. Złącze zostanie uwolnione z trzaskiem.
4. Chwycić złącze i wyjąć je z CPU.



W celu zainstalowania złącza należy wykonać następujące kroki:

1. Przygotować system do podłączenia złącza listwy zaciskowej poprzez odłączenie zasilania od CPU i otwarcie pokrywy listwy zaciskowej.
2. Wycentrować złącze zgodnie z końcówkami w CPU.
3. Wycentrować krawędź złącza od strony przewodów wewnątrz obwodu podstawy złącza.
4. Obracając złącze, pewnie je wcisnąć, aż zatrzaśnie się na swojej pozycji.

Należy sprawdzić, czy złącze jest poprawnie ulokowane i zatrzaśnięte.



2.2 Wskazówki dotyczące okablowania

Właściwe uziemienie oraz okablowanie wszystkich elektrycznych urządzeń jest bardzo ważnym warunkiem uzyskania optymalnych warunków pracy systemu S7--1200 i zapewnienia dodatkowej jego ochrony przed zakłóceniami elektrycznymi. *Por*. schemat okablowania systemu S7-1200 w rozdziale „Dane techniczne".

Informacje wstępne

Przed rozpoczęciem podłączania uziemienia lub okablowania do jakiegokolwiek urządzenia elektrycznego należy się upewnić, że jego źródło zasilania oraz zasilanie wszystkich urządzeń z nim współpracujących jest wyłączone.

Podczas wykonywania okablowania systemu S7-1200 należy postępować według wszelkich obowiązujących zasad, reguł i standardów związanych z wykonywaniem prac elektrycznych. Wszystkie urządzenia muszą być instalowane i obsługiwane zgodnie ze stosownymi przepisami krajowymi i lokalnymi. Aby uzyskać informacje na temat standardów, które obowiązują dla konkretnej instalacji, należy skontaktować się z lokalnym przedstawicielem autoryzowanego serwisu danego urządzenia.

⚠	**OSTRZEŻENIE**
Przeprowadzanie instalacji lub okablowania systemu S7-1200 lub współpracujących z nim urządzeń z załączonym napięciem zasilania może doprowadzić do porażenia elektrycznego lub nieprzewidywalnego działania sprzętu. Nie wyłączenie zasilania S7-1200 i współpracujących urządzeń podczas procedury instalacji lub deinstalacji może spowodować śmierć lub poważne uszkodzenie ciała personelu i/lub zniszczenie mienia lub nieprzewidywalne działanie sprzętu. Należy zawsze stosować odpowiednie procedury bezpiecznej pracy oraz upewnić się, że napięcie zasilania S7-1200 i współpracujących urządzeń jest wyłączone przed instalacją lub demontażem jakiegokolwiek modułu S7-1200 lub urządzenia współpracującego.	

Podczas projektowania uziemienia i okablowania systemu S7-1200 należy zawsze mieć na uwadze bezpieczeństwo. Elektroniczne systemy sterowania, takie jak S7-1200 mogą ulec uszkodzeniu i spowodować nieprzewidywalne działanie urządzeń, którymi sterują, lub które monitorują. Z tego powodu, w celu uniknięcia możliwych zranień personelu lub uszkodzeń mienia, należy zastosować dodatkowe środki bezpieczeństwa, niezależne od S7-1200.

⚠	**OSTRZEŻENIE**
Urządzenia sterujące mogą ulec uszkodzeniu stwarzając niebezpieczną sytuację, która może doprowadzić do nieprzewidywalnego działania sprzętu. Takie nieprzewidywalne działanie może spowodować śmierć lub poważne uszkodzenie ciała personelu i/lub zniszczenie mienia. Należy zawsze stosować wyłączniki awaryjne, urządzenia elektromechaniczne lub inne redundantne zabezpieczenia niezależne od S7-1200.	

Wskazówki dotyczące izolacji

Zasilacz oraz obwody wej/wyj S7-1200 są zaprojektowane tak, by zapewnić bezpieczną separację między napięciem sieciowym i układami niskonapięciowymi, co jest potwierdzone odpowiednimi certyfikatami. Zgodnie z różnymi normami, to zabezpieczenie polega na stosowaniu podwójnej lub wzmocnionej izolacji albo izolacji podstawowej wraz z uzupełniającą. Elementy, które łączą te dwie izolowane części systemu, takie jak optoizolatory, kondensatory, transformatory i przekaźniki mają również odpowiednie certyfikaty potwierdzające bezpieczeństwo. Fragmenty systemu, które spełniają te wymagania są oznaczone w opisie S7-1200 jako cechujące się napięciem izolacji równym 1500 VAC lub większym. Ta specyfikacja wynika z rutynowych testów fabrycznych (2Ue + 1000 V) lub równoważnych, zgodnie z zatwierdzonymi metodami pomiaru. Wytrzymałość izolacji S7-1200 została poddana próbom typu do 4242 VDC.

Wyjścia zasilaczy czujników, układy komunikacyjne i wewnętrzne układy logiczne S7-1200 zawierające wbudowane zasilacze sieciowe są wykonywane jako SELV (*safety extra-low voltage*) zgodnie z normą EN 61131-2. Jeżeli zacisk M zasilacza czujników lub dowolne inne nieizolowane połączenie M do S7-1200 jest uziemione, to układy te nabywają status PELV (*protective extra-low voltage*). Inne połączenia M do S7-1200, dla których niskiego napięcia odniesieniem może być potencjał ziemi, są w niektórych kartach katalogowych oznaczane jako nieizolowane od logiki. Przykładami są zacisk M analogowych układów wej/wyj i zacisk M zasilania cewki przekaźnika.

W celu utrzymania charakteru SELV/PELV niskonapięciowych obwodów S7-1200, zewnętrzne połączenia do portów komunikacyjnych, obwodów analogowych i wszystkich zasilaczy 24 VDC i obwodów wej/wyj muszą być zasilane z certyfikowanych źródeł spełniających wymagania SELV, PELV, Class 2 (klasa 2), *Limited Voltage* (ograniczone napięcie) lub *Limited Power* (ograniczona moc) różnych norm.

⚠	**OSTRZEŻENIE**
Użycie do zasilania obwodów niskonapięciowych z sieci, zasilacza nieizolowanego lub zasilacza z pojedynczą izolacją może spowodować wystąpienie niebezpiecznych napięć w obwodach, które są uważane za bezpieczne dla dotyku, takich jak układy komunikacyjne i obwody niskonapięciowe czujników. Nieoczekiwane wystąpienie wysokich napięć może spowodować porażenie elektryczne prowadzące do śmierci lub poważnych obrażeń personelu i/lub zniszczenia mienia. Wysokie napięcie może być stosowane w przetwornicach mocy wyłącznie wtedy, kiedy są to certyfikowane źródła bezpieczne w dotyku z obwodami o ograniczonych napięciach.	

Wskazówki dotyczące uziemienia S7-1200

Najlepszym sposobem uziemienia realizowanego systemu jest zapewnienie, by wszystkie zaciski masy i uziemienia S7-1200 oraz współpracujących z nim urządzeń były uziemione w jednym punkcie. Ten pojedynczy punkt powinien być następnie połączony bezpośrednio do uziomu całego systemu.

Wszystkie połączenia uziemiające powinny być tak krótkie, jak to jest tylko możliwe, a zastosowane przewody powinny mieć duży przekrój, taki jak 2 mm^2 (14 AWG).

W trakcie planowania uziemień należy kierować się zasadami bezpiecznego uziemiania i zadbać o właściwe działanie urządzeń zabezpieczających.

Wskazówki dotyczące okablowania S7-1200

Podczas planowania okablowania S7-1200 należy zastosować główny wyłącznik zasilania, który jednocześnie odłącza napięcie zasilające CPU, wszystkich obwodów wejściowych i wszystkich obwodów wyjściowych. Należy również zastosować zabezpieczenie nadprądowe, takie jak bezpiecznik topikowy lub automatyczny, które ograniczy prąd awaryjny linii zasilającej. Warto również rozważyć zastosowanie dodatkowego zabezpieczenia poprzez umieszczenie bezpiecznika topikowego lub innego urządzenia ograniczającego prąd w każdym obwodzie wyjściowym.

W każdym obwodzie narażonym na powstanie udarów od piorunów należy zastosować odpowiednie urządzenie tłumiące udary.

Należy unikać prowadzenia wspólnymi korytkami kabli sygnałowych i komunikacyjnych razem z kablami zasilającymi AC oraz wysokoenergetycznymi kablami przesyłającymi szybko przełączane napięcia DC.

Kable zawsze powinny by prowadzone parami, utworzonymi przez przewód neutralny lub wspólny oraz przewód pod napięciem lub sygnałowy.

Należy stosować jak najkrótsze połączenia oraz zapewnić właściwy przekrój przewodu dla wymaganego prądu przewodzenia. Złącze jest przewidziane dla przewodów o przekrojach od 0,3 mm do 2 mm (14 AWG do 22 AWG). W celu zapewnienia najlepszego zabezpieczenia przed zakłóceniami, należy stosować przewody ekranowane. W typowych przypadkach, najlepsze wyniki daje uziemienie ekranu od strony S7-1200.

Podczas łączenia obwodów wejściowych, które są zasilane z zewnętrznego źródła napięcia, należy zastosować w każdym z tych obwodów zabezpieczenia nadprądowe. Zewnętrzne zabezpieczenie nie jest konieczne w układach zasilanych z wewnętrznego zasilacza czujników S7-1200 napięciem 24 VDC, ponieważ to źródło ma już wbudowany ogranicznik prądowy.

Większość modułów S7-1200 ma wyjmowane złącza na okablowanie użytkownika. By uniknąć obluzowanych połączeń należy się upewnić, czy złącze jest pewnie osadzone w gnieździe i czy przewody są pewnie podłączone do złącza. Nie należy zbyt mocno dokręcać śrub, gdyż może to uszkodzić złącze. Maksymalny moment dokręcający złącza wynosi 0,56 Nm.

W celu zapobiegnięcia przepływu niepożądanego prądu w instalacji, pewne fragmenty S7-1200 stanowią bariery izolacyjne. Podczas planowania okablowania systemu, należy wziąć pod uwagę te bariery izolacyjne. W rozdziale „Dane techniczne" są podane napięcia izolacji i położenie barier izolacyjnych. Nie należy polegać na barierach izolacyjnych z wyspecyfikowanym napięciem izolacji mniejszym niż 1500 VAC jako barierach bezpieczeństwa.

Wskazówki dotyczące obciążeń indukcyjnych

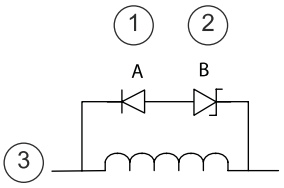
Wszystkie obciążenia o charakterze indukcyjnym powinny być wyposażone w układy tłumiące, które ograniczają wzrost napięcia pojawiający się w momencie odłączania sygnału sterującego. Obwody tłumiące chronią układy wyjściowe przez przedwczesnym uszkodzeniem na skutek przepięć występujących w chwili odłączania obciążeń indukcyjnych. Ponadto układy tłumiące ograniczają zakłócenia elektryczne generowane podczas przełączania obciążeń indukcyjnych. Najbardziej skutecznym sposobem redukcji zakłóceń elektrycznych jest włączanie zewnętrznych obwodów tłumiących równolegle do obciążenia i umieszczanie ich fizycznie blisko obciążenia.

UWAGA

Skuteczność określonego układu tłumiącego zależy od aplikacji i za każdym razem należy zweryfikować, czy jest to odpowiedni układ dla danego zastosowania. Zawsze trzeba też sprawdzać czy wytrzymałość elementów zastosowanych w układzie tłumiącym jest odpowiednia w danej aplikacji.

Sterowanie stałoprądowe (DC) obciążeń indukcyjnych

Stałoprądowe wyjścia S7-1200 mają wbudowane układy tłumiące, odpowiednie – w większości przypadków – do sterowania obciążeniami indukcyjnymi. Ponieważ przekaźniki mogą pracować z obciążeniami stałoprądowymi (DC) lub zmiennoprądowymi (AC), więc w tym przypadku wewnętrzne zabezpieczenie nie jest stosowane. Na przytoczonym rysunku przedstawiono przykładowy układ tłumiący stosowany w przypadku obciążeń DC. W większości zastosowań podłączenie diody A równolegle do obciążenia indukcyjnego jest odpowiednie, ale w przypadku gdy aplikacja wymaga szybszych czasów wyłączania, to rekomendowane jest użycie dodatkowej diody Zenera (B).

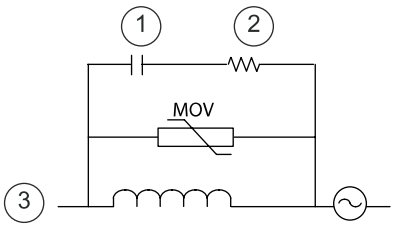


① dioda typu 1N4001 lub równoważna
② dioda Zenera 8,2 V (wyjście DC)
dioda Zenera 36 V (wyjście przekaźnikowe)
③ wyjście

Diodę Zenera należy dobrać odpowiednio pod kątem wytrzymałości prądowej.

Wyjście przekaźnikowe sterujące obciążenia zmiennoprądowe (AC)

Kiedy wyjście przekaźnikowe służy do przełączania obciążeń zasilanych napięciem 115/230 VAC, wtedy równolegle do obciążenia AC należy umieścić obwód rezystor/kondensator, tak jak to pokazano na rysunku. W celu ograniczenia wartości szczytowych napięcia, można również zastosować warystor tlenkowy MOV (*metal oxide varistor*). Trzeba się tylko upewnić, że napięcie pracy MOV jest o przynajmniej 20 % większe od nominalnej wartości napięcia sieciowego



① 0,1 µF
② 100...120 Ω
③ wyjście

Wskazówki dotyczące sterowania lamp

Lampy jako obciążenia niszczą styki przekaźników ze względu na duże udary prądowe. W przypadku lamp z włóknem wolframowym te udary przekraczają ustaloną, nominalną wartość prądu pracy od 15 do 20 razy. Do współpracy z lampami, które będą często przełączane zaleca się stosować przekaźniki z wymiennymi stykami lub tłumiki udarów.

Funkcjonowanie PLC 3

3.1 Wykonanie programu użytkownika

CPU obsługuje następujące typy bloków kodu, które umożliwiają stworzenie wydajnej struktury programu użytkownika:

- Bloki organizacyjne (OB) definiujące strukturę programu. Niektóre OB mają predefiniowane działanie i czynności początkowe, ale użytkownik może również tworzyć OB z czynnościami początkowymi, które sam określi.
- Funkcje (FC) i bloki funkcji (FB) zawierające kod programu odpowiadający za wykonanie określonego zadania lub kombinacje parametrów. Każdy FC lub FB ma zbiór parametrów wejściowych i wyjściowych służących do wymiany danych z blokiem wywołującym. Ponadto FB wykorzystuje skojarzone bloki danych (zwane blokami *instance*), które przechowują dane wykorzystywane w aktualnie przetwarzanym bloku FB.
- Globalne bloki danych (DB) przechowujące dane, które mogą być używane przez wszystkie bloki programu.

Bloki organizacyjne (OB)

OB sterują wykonywaniem programu użytkownika. Każdy OB musi mieć unikalny numer. Niektóre domyślne numery poniżej 200 są zarezerwowane, inne OB muszą mieć numer 200 lub większy.

Określone zdarzenia w CPU uruchamiają wykonywanie bloku organizacyjnego. OB nie może wywołać innego OB lub być wywołany z FC lub FB. Jedynie czynność początkowa, taka jak przerwanie diagnostyczne lub interwał czasowy może uruchomić wykonywanie OB. Inny blok kodu nie może wywołać OB. CPU obsługuje OB zgodnie z posiadaną przez nie klasą priorytetu; OB o wyższym priorytecie są wykonywane przed OB o niższym priorytecie. Najniższa klasa priorytetu to klasa 1 (dla cyklu programu głównego), a najwyższa klasa priorytetu to klasa 28 (należą do niej przerwania diagnostyczne).

OB sterują następującymi operacjami:

- OB cyklu programu są wykonywane cykliczne jeśli CPU jest w trybie RUN. Główny blok kodu jest cyklicznym OB. To w nim umieszcza się instrukcje, które sterują programem użytkownika i stąd wywołuje się dodatkowe bloki użytkownika. Dopuszcza się użycie wielu cyklicznych OB. Domyślnym jest OB 1. Pozostałe muszą nosić nazwy OB 200 lub wyższe.
- OB startowy (rozruchowy) jest wykonywany jednorazowo wtedy, kiedy stan CPU zmienia się ze STOP na RUN, wliczając w to włączenie zasilania w trybie RUN i nakazane rozkazem przejście STOP – RUN. Po zakończeniu wykonania startowego OB, jest uruchamiany główny OB „cyklu programu”. Dopuszcza się użycie wielu startowych OB. Domyślnym jest OB 100. Pozostałe muszą nosić nazwy OB 200 lub wyższe.
- OB opóźnienia jest wykonywany w określonym interwale czasowym po skonfigurowaniu zdarzenia instrukcją rozpoczęcia przerwania (SRT_DINT). Czas opóźnienia jest przekazany jako parametr wejściowy instrukcji rozszerzonej

SRT_DINT. OB opóźnienia przerywa wykonywanie normalnego programu cyklicznego wtedy, kiedy upłynie ustalony czas opóźnienia. Można skonfigurować do czterech zdarzeń opóźnionych dla dowolnej chwili, przy czym dozwolony jest jeden OB dla każdego skonfigurowanego zdarzenia opóźnionego. Nazwa OB musi być OB 200 lub większa.

- OB cyklicznego przerwania jest wykonywany w określonym przedziale czasu. OB cyklicznego przerwania przerywa wykonywanie normalnego programu cyklicznego w określonych przez użytkownika przedziałach czasu, na przykład co 2 sekundy. Można skonfigurować do czterech zdarzeń cyklicznego przerwania, przy czym dozwolony jest jeden OB dla każdego skonfigurowanego zdarzenia cyklicznego przerwania. Nazwa OB musi być OB 200 lub większa.
- OB przerwania sprzętowego jest wykonywany wtedy, kiedy wystąpi istotne zdarzenie, włączając w to zbocza opadające lub narastające na wbudowanych wejściach cyfrowych i zdarzenia HSC (dotyczące szybkiego licznika). OB przerwania sprzętowego przerywa wykonywanie normalnego programu cyklicznego w odpowiedzi na sygnał o zdarzeniu sprzętowym. Użytkownik definiuje w konfiguracji sprzętowej jakie to są zdarzenia. Dozwolony jest jeden OB dla każdego skonfigurowanego zdarzenia sprzętowego. Nazwa OB musi być OB 200 lub większa.
- OB przerwania błędu czasowego jest wykonywany wtedy, kiedy zostanie wykryty błąd czasowy. OB przerwania błędu czasowego przerywa wykonywanie normalnego programu cyklicznego jeśli zostanie przekroczony maksymalny czas cyklu. Maksymalny czas cyklu jest zdefiniowany we właściwościach PLC. OB 80 jest jedynym numerem OB obsługującym zdarzenie błędu czasowego. Użytkownik może określić jaka akcja jest podejmowana w razie, gdy OB 80 nie istnieje: błąd jest ignorowany lub wykonywany jest STOP.
- OB przerwania błędu diagnostycznego jest wykonywany wtedy, kiedy zostaje wykryty i zgłoszony błąd diagnostyczny. OB przerwania błędu diagnostycznego przerywa wykonywanie normalnego programu cyklicznego wtedy, kiedy moduł zdolny do wykonywania diagnostyki rozpoznaje błąd (o ile wystawienie sygnału przerwania w odpowiedzi na błąd diagnostyczny zostało w tym module odblokowane). OB 82 jest jedynym numerem OB obsługującym zdarzenie błędu diagnostycznego. Jeśli w programie nie ma OB diagnostycznego, to użytkownik może tak skonfigurować CPU, że błąd będzie ignorowany albo zostanie wykonany STOP.

Wykonywanie programu użytkownika

Wykonywanie programu użytkownika rozpoczyna się od jednego lub więcej opcjonalnych rozruchowych bloków organizacyjnych (OB), które są wykonywane jednokrotnie po wejściu CPU w tryb RUN, a po nich wykonywany jest cyklicznie jeden lub więcej OB cyklu programu. OB może być również skojarzony z przerwaniem wywołanym albo standardowym zdarzeniem albo wykrytym błędem i jest wykonywany za każdym razem kiedy wystąpi odpowiednie standardowe zdarzenie lub błąd.

Funkcja (FC) lub blok funkcji (FB) jest blokiem kodu programu, który może być wywołany z OB albo innej FC lub FB, przy czym głębokość zagnieżdżenia takich wywołań wynosi:

- 16 z cyklu programu lub OB startowego.
- 4 z OB przerwania opóźnienia, przerwania cyklicznego, przerwania sprzętowego, przerwania błędu czasu lub przerwania błędu diagnostycznego.

FC nie są skojarzone z jakimś szczególnym blokiem danych (DB), podczas gdy FB są bezpośrednio związane z DB i wykorzystują ten DB do przekazywania parametrów i przechowywania tymczasowych wartości i wyników.

Rozmiar programu użytkownika, danych i konfiguracji jest ograniczony wielkością dostępnej pamięci ładowania CPU. Liczba obsługiwanych bloków nie jest ograniczona; jedynym ograniczeniem jest wielkość pamięci.

Każdy cykl obejmuje zapisywanie stanu wyjść, odczytywanie stanu wejść, wykonanie instrukcji programu użytkownika oraz wykonanie obsługi systemu lub przetwarzania w tle. Taki cykl jest nazywany cyklem programu.

Płytka sygnałowa, moduły rozszerzeń i moduły komunikacyjne są wykrywane i rejestrowane tylko w trakcie włączenia zasilania. Wkładanie i wyjmowanie płytki sygnałowej, modułów rozszerzeń i modułów komunikacyjnych pod napięciem nie jest obsługiwane. Jedynym wyjątkiem jest karta pamięci SIMATIC Memory Card, która może być wkładana i wyjmowana wtedy, kiedy CPU jest zasilana.

W warunkach standardowych, wszystkie punkty I/O analogowe i cyfrowe są uaktualniane synchronicznie z cyklem programu wykorzystującym obszar pamięci wewnętrznej zwany obrazem procesu. Obraz procesu zawiera chwilowy stan fizyczny wejść i wyjść (fizyczne punkty I/O CPU, płytki sygnałowej i modułów rozszerzeń).

CPU wykonuje następujące zadania:

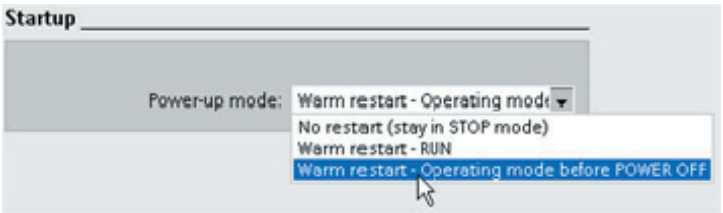
- Tuż przed wykonaniem programu użytkownika CPU odczytuje stan fizycznych wejść i zapamiętuje te wartości wejściowe w obszarze wejściowym pamięci obrazu procesu. Dzięki temu uzyskuje się pewność, że te dane pozostają stałe w trakcie wykonywania instrukcji użytkownika.
- CPU wykonuje zadania określone instrukcjami użytkownika i – nie zmieniając stanu fizycznych wyjść – uaktualnia wartości wyjściowe w obszarze wyjściowym pamięci obrazu procesu.
- Po wykonaniu programu użytkownika, CPU przepisuje stany wyjść z obszaru wyjściowego pamięci obrazu procesu do fizycznych wyjść.

Ten proces zapewnia zachowanie spójności logiki poprzez wykonywanie w danym cyklu instrukcji użytkownika i zapobiega zmianom („migotaniu”) stanu fizycznych punktów wyjściowych, w wyniku mogących występować wielokrotnie w cyklu zmianom w obszarze wyjściowym pamięci obrazu procesu.

Użytkownik może zmienić standardowe działanie modułu, wyłączając to automatyczne uaktualnianie stanu punktów wyjściowych. Można również bezpośrednio odczytywać i zapisywać cyfrowe i analogowe stany I/O modułów podczas wykonywania instrukcji. Bezpośredni odczyt stanu fizycznych wejść nie uaktualnia obszaru wejściowego pamięci obrazu procesu. Bezpośredni zapis stanu do fizycznych wyjść uaktualnia zarówno obszar wyjściowy pamięci obrazu procesu, jak i stan fizycznych punktów wyjściowych.

Konfiguracja parametrów startowych

Konfigurowanie sposobu działania CPU w trakcie startu po podłączeniu zasilania jest dokonywane za pomocą atrybutów CPU.



Należy wybrać, czy CPU rozpoczyna pracę w trybie STOP, RUN lub w poprzednim trybie (przed wyłączeniem zasilania).

Przed wejściem w tryb RUN, CPU wykonuje gorący restart. W trakcie gorącego restartu kasowane są do wartości domyślnych wszystkie pamięci, które nie są trwałe, ale zachowywane są bieżące wartości przechowywane w pamięciach trwałych.

3.1.1 Tryby pracy CPU

CPU może pracować w jednym z trzech trybów: w trybie STOP, w trybie STARTUP i w trybie RUN. Diody statusu znajdujące się na płycie czołowej CPU wskazują jaki jest aktualny tryb pracy.

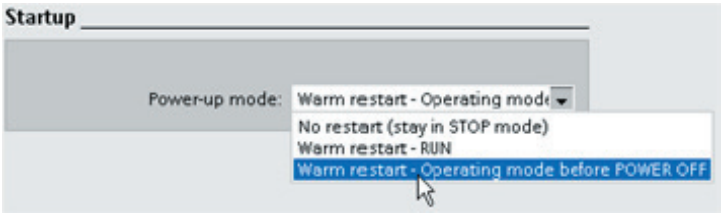
- W trybie STOP CPU nie wykonuje programu i użytkownik może wczytać projekt.
- W trybie STARTUP wykonywany jest jednokrotnie startowy OB (jeśli istnieje). W fazie startowej trybu RUN nie są obsługiwane przerwania.
- W trybie RUN regularnie jest powtarzany cykl programu. Mogą się pojawiać przerwania i są przetwarzane w dowolnym miejscu cyklu programu.

W trybie RUN nie można wczytywać projektu.

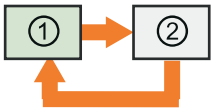
Przechodzenie do trybu RUN CPU wykonuje metodą gorącego restartu. Gorący restart nie obejmuje kasowania pamięci, ale pamięć może zostać skasowana odpowiednim rozkazem umieszczonym w programie. Kasowanie pamięci czyści całą pamięć roboczą, obszary pamięci trwałej i nietrwałej oraz kopiuje zawartość pamięci ładowania do pamięci roboczej. Kasowanie pamięci nie czyści zawartości bufora diagnostycznego lub na stałe zapisanej wartości adresu IP. Podczas gorącego restartu inicjalizowane są wszystkie nietrwałe elementy systemu oraz dane użytkownika.

Za pomocą oprogramowania można wyspecyfikować tryb włączania zasilania CPU wraz z metodą restartu. Ta pozycja konfiguracji jest dostępna w menu „Startup” dla CPU pod nagłówkiem „Device Configuration”. Po włączeniu zasilania CPU wykonuje sekwencję testów diagnostycznych i inicjalizuje system. Następnie wchodzi w odpowiedni tryb włączania zasilania. Wykrycie pewnych błędów uniemożliwia CPU wejście w tryb RUN. CPU może pracować w następujących trybach włączania zasilania.

- W trybie STOP.
- Przejścia do trybu RUN po wykonaniu gorącego restartu.
- Przejścia do poprzedniego trybu pracy po wykonaniu gorącego restartu.



Użytkownik może zmienić bieżący tryb pracy wykonując komendy „STOP" lub „RUN" za pomocą dostępnych narzędzi *online* oprogramowania. Można również wykonać z programu użytkownika instrukcję STP, która ustawia tryb STOP pracy CPU. Pozwala to zatrzymać wykonywanie programu użytkownika zgodnie z jego logiką.

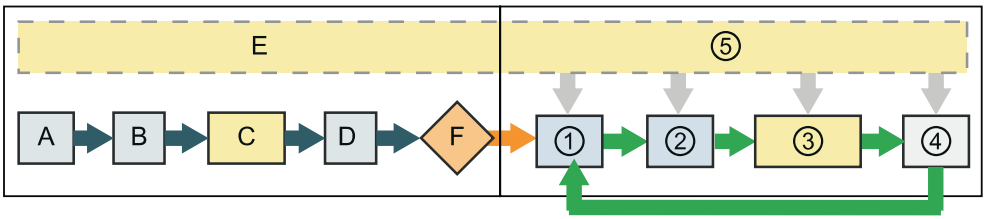


W trybie STOP, CPU ① obsługuje żądania komunikacji (jeśli występują) i ② wykonuje autodiagnostykę.

W trybie STOP, CPU nie wykonuje programu użytkownika i nie odbywa się automatyczne uaktualnianie obrazu procesu.

Użytkownik może wczytać swój projekt tylko wtedy, kiedy CPU jest w trybie STOP.

W trybie RUN, CPU wykonuje zadania przedstawione na poniższym rysunku.



STARTUP

A czyści obszar pamięci I

B inicjalizuje wyjścia z ostatnimi wartościami lub zastępczymi wartościami

C wykonuje startowe OB

D kopiuje stan fizycznych wejść do pamięci I

E Zapisuje zdarzenia przerwań do kolejki oczekującej na wykonanie w trybie RUN

F uaktywnia zapis zawartości pamięci Q do fizycznych wyjść

RUN

① zapisuje zawartość pamięci Q do wyjść fizycznych

② kopiuje stan fizycznych wejść do pamięci I

③ wykonuje cykliczne OB

④ obsługuje żądania komunikacji i wykonuje autodiagnostykę

⑤ obsługuje przerwania w dowolnej części cyklu programu

Praca w trybie STARTUP

Zawsze jak tryb pracy zmienia się ze STOP na RUN, CPU kasuje obszar wejściowy obrazu procesu, inicjalizuje obszar wyjściowy obrazu procesu i wykonuje startowe OB. (Dlatego też, dowolna operacja odczytu obszaru wejściowego obrazu procesu wykonana instrukcjami zawartymi w startowym OB da w wyniku same zera, a nie rzeczywiste stany na wejściach fizycznych.) W celu odczytania w trybie STARTUP bieżących stanów wejść fizycznych należy wykonać odczyt bezpośredni. Startowe OB i dowolne powiązane FC i FB są wykonywane w następnej kolejności. Jeżeli istnieje więcej niż jeden startowy OB, to wszystkie są wykonywane zgodnie ze swoimi numerami, przy czym jako pierwszy jest wykonywany OB z najniższym numerem.

W trybie STARTUP CPU wykonuje również następujące zadania:

- Przerwania są ustawiane w kolejce, ale w fazie rozruchowej nie są wykonywane.
- W fazie rozruchowej nie jest monitorowany czas cyklu.
- Podczas startu mogą być wykonywane zmiany konfiguracji modułów HSC (szybki licznik), PWM (modulator szerokości impulsów) i PtP (komunikacji punkt-punkt).
- Moduły HSC, PWM i komunikacji PtP mogą pracować tylko w trybie RUN.

Po zakończeniu startowych OB, CPU przechodzi w tryb RUN i wykonuje zadania sterowania w ciągłym cyklu programu.

Wykonywanie cyklu programu w trybie RUN

W każdym cyklu programu CPU zapisuje wyjścia, odczytuje wejścia, uaktualnia moduły komunikacyjne, wykonuje zadania na własne potrzeby i odpowiada na przerwania wynikające z warunków ustalonych przez użytkownika.

Te działania (z wyłączeniem zdarzeń ustalonych przez użytkownika) są wykonywane regularnie i w porządku sekwencyjnym. Te zdarzenia użytkownika, które są odblokowane, są obsługiwane zgodnie ze swoimi priorytetami w takiej kolejności, w jakiej się pojawiają.

System gwarantuje wykonanie kompletnego cyklu programu w czasie nazywanym maksymalnym czasem cyklu; w przeciwnym wypadku generowane jest zdarzenie błędu czasowego.

Każdy cykl programu rozpoczyna się od pobrania z obrazu procesu bieżących wartości wyjść cyfrowych i analogowych i zapisaniu ich do fizycznych wyjść CPU, SB i SM skonfigurowanych tak, by były synchronicznie uaktualniane (konfiguracja domyślna).

Jeśli CPU, SB lub SM zostały wyłączone z automatycznego uaktualniania I/O, to do ich wyjść nie są kopiowane wartości z obrazu procesu. Wyjścia selektywnie wykluczone z uaktualniania I/O są dostępne podczas wykonywania programu użytkownika za pomocą bezpośredniego adresowania i wówczas można zmienić ich stan fizyczny. Kiedy dostęp do wyjścia fizycznego odbywa się za pomocą instrukcji, to uaktualniane są zarówno obszar wyjściowy obrazu procesu, jak i stan wyjścia fizycznego.

W dalszym ciągu cyklu programu odczytywane są bieżące wartości wejść cyfrowych i analogowych z CPU, SB i SM skonfigurowanych tak, by były synchronicznie uaktualniane (konfiguracja domyślna).

Jeśli CPU, SB lub SM zostały wyłączone z automatycznego uaktualniania, to stany ich wejść nie są kopiowane do obrazu procesu. Wejścia selektywnie wykluczone z uaktualniania I/O są dostępne za pomocą bezpośredniego adresowania i wówczas można odczytać ich stan fizyczny. Kiedy dostęp do wejścia fizycznego odbywa się za pomocą instrukcji, to stan tego wejścia można odczytać, ale wejściowy obszar obrazu procesu nie jest uaktualniany.

Po odczytaniu stanu wejść, jest wykonywany program użytkownika począwszy od pierwszej instrukcji po ostatnią. Wykonywane są więc wszystkie OB cyklu programu wraz z powiązanymi z nimi FC i FB. OB cyklu programu są wykonywane w kolejności posiadanych numerów, przy czym jako pierwszy jest wykonywany OB z najniższym numerem.

W kroku cyklu programu przeznaczonym na prowadzenie komunikacji są przetwarzane odebrane wiadomości. Przygotowane odpowiedzi są odkładane na bok, i oczekują na przesłanie do odbiorcy w odpowiednim czasie.

Testy autodiagnostyki obejmują okresowe sprawdzanie pamięci oprogramowania sprzętowego i użytkownika, jak również sprawdzanie stanu modułów I/O.

Przerwania mogą występować w dowolnej części cyklu programu i są generowane zdarzeniami. Kiedy zachodzi zdarzenie, CPU przerywa cykl programu i wywołuje OB przygotowany do obsługi tego zdarzenia. Kiedy OB zakończy obsługę zdarzenia, wtedy CPU podejmuje wykonywanie programu użytkownika od miejsca, w którym został przerwany.

3.1.2 Priorytety i kolejkowanie obsługi zdarzeń

Praca CPU jest sterowana zdarzeniami. Jedynym niezbędnym zdarzeniem jest cykl programu. Wszystkie inne zdarzenia mogą być odblokowane jeśli zajdzie taka potrzeba.

Niektóre zdarzenia, takie jak zdarzenia cykliczne, są odblokowywane podczas konfiguracji. Inne zdarzenia są odblokowywane w trakcie pracy systemu. Zdarzenie, jeśli jest odblokowane, zostaje skojarzone z pewnym OB (zdarzenia cyklu programu i zdarzenia rozruchowe mogą być skojarzone z wieloma OB). Wystąpienie zdarzenia prowadzi do wykonania procedury obsługi tego zdarzenia, z którą jest skojarzony OB wraz ze wszystkimi wywoływanymi z tego OB funkcjami. Do określenia porządku wykonywania procedur obsługi zdarzeń stosuje się priorytety, grupy priorytetów i kolejki.

Kolejkowanie i priorytety wykonania obsługi zdarzeń

Liczba pochodzących z jednego źródła zdarzeń oczekujących (w kolejce) na obsługę jest ograniczona przez stosowanie różnych kolejek dla zdarzeń różnego typu. Po osiągnięciu limitu oczekujących zdarzeń określonego typu, kolejne zdarzenia są tracone. Więcej informacji na temat przepełnienia kolejek jest dostępnych w kolejnym punkcie „Zdarzenia błędów czasu”.

Każde zdarzenie CPU ma przyznany priorytet, a priorytety zdarzeń są zaklasyfikowane do grup priorytetów. Poniższa tablica zawiera podsumowanie, przedstawiające dla obsługiwanych zdarzeń głębokość kolejek, grupy priorytetów i priorytety.

UWAGA

Użytkownik nie może zmienić przypisanego priorytetu lub grupy priorytetu jak również głębokości kolejki.

Ogólnie, zdarzenia są obsługiwane zgodnie z ich priorytetem (najpierw najwyższy priorytet). Zdarzenia mające ten sam priorytet są obsługiwane zgodnie z zasadą „pierwszy się pojawił, pierwszy jest obsłużony".

Typ zdarzenia (OB)	Ile	Ważne numery OB	Głębokość kolejki	Grupa priorytetu	Priorytet
Cykl programu	1 zdarzenie cykl programu Dozwolonych wiele OB	1 (domyślny) 200 lub większy	1	1	1
Rozruch	1 zdarzenie rozruch[1] Dozwolonych wiele OB	100 (domyślny) 200 lub większy	1		1
Czas opóźnienia	4 zdarzenia czas opóźnienia 1 OB na zdarzenie	200 lub większy	8	2	3
Cykliczne	4 zdarzenia cykliczne 1 OB na zdarzenie	200 lub większy	8		4
Zbocza	16 zdarzeń zbocze narastające 16 zdarzeń zbocze opadające 1 OB na zdarzenie	200 lub większy	32		5
HSC	6 zdarzeń CV = PV 6 zdarzeń zmiana kierunku 6 zdarzeń zewnętrzne kasowania 1 OB na zdarzenie	200 lub większy	16		6
Błąd diagnostyki	1 zdarzenie (tylko OB 82)	Tylko 82	8		9
Błąd czasu	1 zdarzenie błąd czasu 1 zdarzenie MaxCycle (tylko OB 80) 1 2xMaxCycle	Tylko 80	8	3	26 27

[1] Specjalne przypadki dla zdarzenia rozruch

- Zdarzenie rozruchowe i zdarzenie cyklu programu nigdy nie występują w tym samym czasie, ponieważ zdarzenie rozruch kończy się zanim rozpoczyna się zdarzenie cykl programu (jest to kontrolowane przez system operacyjny).
- Zdarzenie rozruch może być przerwane wyłącznie przez zdarzenie błąd diagnostyki (skojarzone z OB 82). Wszystkie inne zdarzenia są ustawiane w kolejkę do późniejszej obsługi po zakończeniu rozruchu.

Wykonywanie rozpoczętego OB nie może być przerwane przez wystąpienie zdarzenia należącego do grupy o tym samym lub niższym priorytecie. Takie zdarzenia są kolejkowane do późniejszej obsługi, co tym samym umożliwia zakończenie wykonywania bieżącego OB.

Jednakże, zdarzenie należące do grupy o wyższym priorytecie może przerwać wykonywanie bieżącego OB, i w takim przypadku CPU zaczyna obsługę tego zdarzenia o wyższym priorytecie. Po zakończeniu obsługi zdarzenia o wyższym priorytecie CPU wykonuje OB obsługujące inne zdarzenia z kolejki, należące do tej

grupy o wyższym priorytecie. Jeżeli nie ma takich oczekujących zdarzeń w grupie o wyższym priorytecie, to CPU powraca do grupy o niższym priorytecie i podejmuje dalsze wykonywanie wcześniejszego OB od miejsca, w którym został przerwany.

Opóźnienie przerwania

Opóźnienie przerwania (czas od momentu, gdy CPU zostało powiadomione o zdarzeniu do chwili wykonania pierwszej instrukcji OB obsługującego to zdarzenie) wynosi 175 µs lub mniej, pod warunkiem, że OB cyklu programu jest w trakcie pojawienia się tego przerwania jedynym aktywnym programem obsługi zdarzenia.

Zdarzenia błędu czasu

Spełnienie jednego z kilku różnych warunków określających błędy czasu generuje w efekcie zdarzenie błędu czasu. Mogą wystąpić następujące błędy czasu:

- Przekroczenie maksymalnego czasu cyklu.
- Nie może wystartować wywołany OB.
- Wystąpienie przepełnienia kolejki.

Warunek przekroczenia maksymalnego czasu cyklu występuje wtedy, kiedy cykl nie zostaje zakończony w wyspecyfikowanym maksymalnym czasie cyklu programu. *Por*. część tego podręcznika „Monitorowanie czasu cyklu" w celu uzyskania więcej informacji związanych z maksymalnym czasem cyklu, sposobem konfigurowania maksymalnego czasu cyklu programu i sposobem kasowania timera cyklu.

Warunek braku możliwości uruchomienia wywoływanego OB występuje wtedy, kiedy OB jest wywoływany przerwaniem cyklicznym lub przerwaniem opóźnienia czasu, ale ten wywoływany OB jest już uruchomiony.

Warunek przepełnienia kolejki występuje wtedy, kiedy przerwania pojawiają się szybciej niż mogą być obsługiwane. Liczba oczekujących (w kolejce) zdarzeń jest ograniczona przez zastosowanie różnych kolejek dla zdarzeń każdego typu. Kiedy jakieś zdarzenie wystąpi, gdy jego kolejka jest pełna, to generowane jest zdarzenie błędu czasu.

Wszystkie zdarzenia błędu czasu uruchamiają OB 80 (jeśli istnieje). Jeśli OB 80 nie istnieje, to CPU ignoruje ten błąd. Jeśli w tym samym cyklu programu wystąpi warunek przekroczenia dwóch maksymalnych czasów cyklu bez kasowania timera cyklu, to CPU przechodzi do trybu STOP niezależnie od tego, czy OB 80 istnieje. *Por*. część tego podręcznika „Monitorowanie czasu cyklu".

OB 80 zawiera informacje rozruchowe pomagające użytkownikowi określić, które zdarzenie i OB wygenerowało błąd czasu. Wewnątrz OB 80 można wstawić program badający te wartości rozruchowe i podejmujący odpowiednią akcję. OB 80 pozwala na obsługę następujących lokalizacji rozruchowych:

Wejście	Typ danych	Opis
fault_id	BYTE	16#01 – przekroczony maksymalny czas cyklu 16#02 – nie można uruchomić wywoływanego OB 16#07 i 16#09 – wystąpiło przepełnienie kolejki
csg_OBnr	OB_ANY	Numer OB wykonywanego podczas wystąpienia błędu
csg_prio	UINT	Priorytet OB powodującego błąd

W nowo utworzonym projekcie nie ma OB 80 obsługującego przerwanie błędu czasu. Jeśli jest to konieczne OB 80 można dodać do projektu poprzez podwójne kliknięcie „Add new block" w menu drzewa „Program blocks", a następnie wybranie „Organization block", i na koniec „Time error interrupt".

Zdarzenia błędów diagnostyki

Niektóre urządzenia są w stanie wykrywać i raportować o błędach diagnostycznych. Spełnienie jednego z kilku różnych warunków określających błędy diagnostyki generuje w efekcie zdarzenie błędu diagnostyki. Mogą wystąpić następujące błędy diagnostyki:

- Brak zasilania
- Przepełnienie
- Niedomiar
- Przerwa
- Zwarcie

Wszystkie zdarzenia błędu diagnostyki uruchamiają OB 82 (jeśli istnieje). Jeśli OB 82 nie istnieje, to CPU ignoruje ten błąd.

OB 82 zawiera informacje rozruchowe pomagające użytkownikowi określić błąd i urządzenie, które zgłasza ten błąd. Wewnątrz OB 82 można wstawić program badający te wartości rozruchowe i podejmujący odpowiednią akcję. OB 82 pozwala na obsługę następujących lokalizacji rozruchowych:

Wejście	Typ danych	Opis
IO_state	WORD	Stan I/O urządzenia
laddr	HW_ANY	Urządzenie, które zgłosiło błąd
channel	UINT	Numer kanału (liczony od 0)
multi_error	BOOL	TRUE jeśli wystąpił więcej niż jeden błąd

W nowo utworzonym projekcie nie ma OB 82 obsługującego przerwanie błędu diagnostyki. Jeśli jest to konieczne OB 82 można dodać do projektu poprzez podwójne kliknięcie „Add new block" w menu drzewa „Program blocks", a następnie wybranie „Organization block", i na koniec „Diagnostic error interrupt".

Monitorowanie czasu cyklu

Czas cyklu jest to czas jaki system operacyjny CPU potrzebuje do wykonania pełnego cyklu programu w trybie RUN. CPU oferuje dwie metody monitorowania czasu cyklu:

- Maksymalny czas cyklu programu
- Ustalony minimalny czas cyklu programu

Monitorowanie cyklu programu rozpoczyna się po zakończeniu fazy rozruchowej. Konfigurację tej cechy wykonuje się w menu CPU „Device Configuration" wybierając pozycję „Cycle time".

CPU zawsze monitoruje cykl programu i reaguje jeżeli zostaje przekroczony maksymalny czas cyklu programu. Jeżeli cykl programu trwa dłużej niż wynosi skon-

figurowana wartość maksymalna określającą czas trwania cyklu programu, to jest generowany błąd i jego obsługa jest wykonywana jednym z dwóch sposobów:

- Jeśli nie ma OB 80 obsługującego przerwanie błędu czasu, to CPU przechodzi w tryb STOP.
- Jeśli jest OB 80 obsługujący przerwanie błędu czasu, to CPU wykonuje OB 80.

Dostępna jest instrukcja RE_TRIGR (ponownie rozpocznij monitorowanie czasu cyklu), która kasuje timer mierzący czas cyklu. Jednakże ta instrukcja funkcjonuje tylko w OB cyklu programu; instrukcja RE_TRIGR jest ignorowana jeśli jest wywołana z OB 80.

Jeżeli maksymalny czas cyklu programu jest przekroczony dwukrotnie w ramach tego samego cyklu programu i między tymi dwoma zdarzeniami nie zostanie zastosowana instrukcja RE_TRIGR, to CPU natychmiast przechodzi w tryb STOP.

Powtarzane wykonywanie instrukcji RE_TRIGR może potencjalnie stworzyć nieskończoną pętlę lub doprowadzić do bardzo długiego czasu trwania cyklu programu. W celu zabezpieczenia CPU przed zapętleniem w cyklu programu, co 100 ms jest alokowana komunikacyjna szczelina czasowa. Czas trwania tej szczeliny jest określony procentowo podczas konfiguracji parametru „Communication load" w menu „CPU Device configuration". Daje to możliwość odzyskania kontroli nad CPU i jeśli jest to konieczne wymuszenia przejścia do trybu STOP.

Zwykle cykl programu jest wykonywany tak szybko, jak tylko może być wykonany i następny cykl programu rozpoczyna się jak tylko bieżący zostanie zakończony. W zależności od programu użytkownika i zadań komunikacyjnych czas cyklu programu może się zmieniać od jednego cyklu programu do następnego. W celu wyeliminowania tych zmian, CPU obsługuje opcjonalny cykl programu z ustalonym minimalnym czasem trwania (zwany również ustalonym cyklem programu). Kiedy ta cecha opcjonalna jest aktywna i ustalony minimalny czas cyklu programu jest wyrażony w ms, to CPU kończy każdy cykl programu z zachowaniem tego czasu minimalnego i tolerancją ±1 ms.

W przypadku, gdy CPU zakończy normalny cykl programu w czasie krótszym niż wyspecyfikowany minimalny czas cyklu, to w tym dodatkowym czasie cyklu programu CPU wykonuje diagnostykę i/lub obsługuje zadania komunikacyjne. W ten sposób CPU zawsze poświęca tyle samo na wykonanie jednego cyklu programu.

W przypadku, gdy CPU nie ukończy cyklu programu w wyspecyfikowanym minimalnym czasie cyklu, to jest on kończony w zwykły sposób (łącznie z komunikacją) i CPU nie wywołuje żadnej reakcji systemu w wyniku przekroczenia minimalnego czasu programu.

W poniższej tablicy przedstawiono zakresy i wartości domyślne dla funkcji monitorowania czasu cyklu.

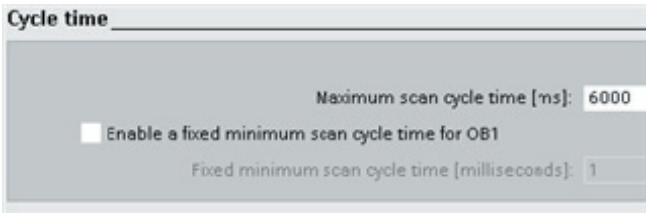
Czas cyklu	**Zakres [ms]**	**Wartość domyślna**
Maksymalny czas cyklu programu	1 – 6000	150 ms
Ustalony minimalny czas cyklu programu	1 – maksymalny czas cyklu programu	nieaktywny

- Maksymalny czas cyklu programu jest zawsze uaktywniony i użytkownik musi wybrać wartość tego czasu z dozwolonego zakresu 1...6000 ms. Wartością domyślną jest 150 ms.
- Ustalony minimalny czas cyklu programu jest opcjonalny i domyślnie nieaktywny. Jeśli jest używany, to należy wybrać jego wartość z przedziału od 1 ms do ustalonej wartości maksymalnego czasu cyklu programu.

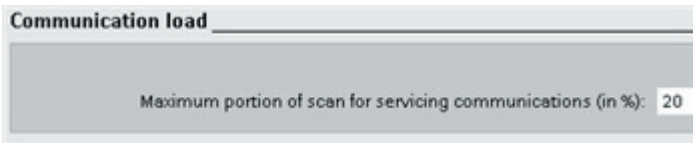
Konfiguracja czasu cyklu i obciążenia komunikacyjnego.

Użytkownik może skonfigurować niżej podane parametry jako właściwości CPU:

- Czas cyklu. Można wprowadzić maksymalną wartość czasu cyklu programu. Można także wprowadzić wartość ustalonego minimalnego czasu cyklu programu.



- Obciążenie komunikacyjne. Można określić jaki procent czasu cyklu programu będzie przeznaczony na zadania komunikacyjne.



Więcej informacji na temat cyklu programu jest podanych w części Monitorowanie czasu cyklu.

3.1.3 Pamięć CPU

Zarządzanie pamięcią

CPU dzieli pamięć na następujące obszary służące do przechowywania programu użytkownika, danych i konfiguracji:

- Pamięć ładowania jest pamięcią nieulotną, w której przechowywane są program użytkownika, dane i konfiguracja. Kiedy projekt jest wczytany do CPU, to najpierw trafia do obszaru pamięci ładowania. Ten obszar jest ulokowany albo na karcie pamięci (jeśli jest) albo w CPU. Ten nieulotny obszar pamięci jest zachowany w przypadku utraty zasilania. Karta pamięci udostępnia większą przestrzeń adresową niż pamięć wbudowana w CPU.
- Pamięć robocza RAM jest pamięcią ulotną, służącą do przechowywania pewnych elementów projektu użytkownika podczas wykonywania programu. W celu poprawy wydajności systemu, CPU kopiuje pewne elementy projektu z pamięci ładowania do pamięci roboczej. Zawartość tej pamięci jest tracona przy odłączaniu zasilania i CPU odtwarza ją po ponownym włączeniu zasilania.

- Pamięć trwała jest pamięcią nieulotną przechowującą ograniczoną liczbę wartości z pamięci roboczej. Pamięć trwała jest stosowana do przechowywania wartości z wybranych miejsc pamięci użytkownika na wypadek utraty zasilania. CPU jest celowo zaprojektowana w taki sposób, by w przypadku utraty zasilania miała dostatecznie dużo czasu na przepisanie do pamięci trwałej ograniczonej liczby wybranych wartości. Po przywróceniu zasilania te zachowane wartości są odtwarzane do oryginalnych pozycji.

W celu sprawdzenia aktualnego użycia pamięci w bieżącym projekcie, użytkownik może kliknąć prawym klawiszem myszy na wybrany PLC z drzewa lub na jeden z jego bloków i wybrać „Resources". W celu sprawdzenia aktualnego użycia pamięci w bieżącym PLC, użytkownik może podwójnie kliknąć na „Online and diagnostics" w drzewie, rozwinąć „Diagnostics" i wybrać „Memory".

Pamięć trwała

CPU jest zdolna do trwałego przechowania 2048 bajtów danych. Użytkownik może decydować, które dane z pamięci roboczej, na przykład dane DB lub pamięci bitowej (M), mają być trwale przechowywane w przypadku każdorazowego zaniku zasilania.

Kiedy zanika zasilanie, dane z pamięci roboczej przeznaczone do zachowania są kopiowane do bloku 2048 kolejnych bajtów. Następnie jest obliczana suma kontrolna i dane przeznaczone do zachowania, po których następuje suma kontrolna, są zapisywane do pamięci nieulotnej. Ani suma kontrolna, ani żadna z innych wartości wymaganych do trwałego zapamiętania przez system operacyjny CPU nie zajmują żadnego z 2048 bajtów przewidzianych dla użytkownika.

Po przywróceniu zasilania system odtwarza zapamiętane dane i zapisuje je do oryginalnych pozycji pamięci roboczej, przywracając stan sprzed utraty zasilania.

Każda próba wyboru więcej niż 2048 bajtów danych do trwałego zapisania jest odrzucana.

Bufor diagnostyczny

CPU obsługuje bufor diagnostyczny, który zawiera wpisy o wszystkich zdarzeniach diagnostycznych. Każdy wpis składa się z daty i czasu wystąpienia zdarzenia, kategorii zdarzenia i opisu zdarzenia. Wpisy są wyświetlane w porządku chronologicznym, przy czym najbardziej aktualne zdarzenie jest wyświetlane na samym szczycie. W przypadku nieprzerwanego zasilania CPU w tym rejestrze jest dostępnych do 50 ostatnich zdarzeń. Kiedy rejestr się zapełni, wtedy nowe zdarzenie zastępuje najstarszy wpis rejestru. Przy utracie zasilania zachowywanych jest dziesięć ostatnich wpisów.

W buforze diagnostycznym są rejestrowane następujące typy zdarzeń:

- Każde zdarzenie systemu diagnostycznego; na przykład błędy CPU i błędy modułu.
- Każda zmiana stanu CPU (każde włączenie zasilania, każde przejście do trybu STOP, każde przejście do trybu RUN).
- Każda zmiana skonfigurowanego obiektu, z wyjątkiem zmian dokonanych przez CPU i program użytkownika.

W celu uzyskania dostępu do bufora diagnostycznego, użytkownik musi być *online*. Wówczas należy odszukać rejestr zdarzeń zgodnie ze ścieżką „Online & diagnostics

/ Diagnostics / Diagnostics buffer". Więcej informacji dotyczących rozwiązywania problemów i usuwania błędów z programu zawiera rozdział „Online i diagnostyka".

Zegar czasu rzeczywistego

CPU obsługuje zegar czasu rzeczywistego. Super-kondensator dostarcza do zegara energię tak, by mógł on działać również wtedy, kiedy od CPU jest odłączone zasilanie. Super-kondensator jest ładowany w czasie, gdy CPU jest zasilana. Jeżeli CPU była zasilana bez przerwy przez okres co najmniej 24 godzin, to super-kondensator ma zgromadzony wystarczająco duży ładunek, by zasilać zegar przez 10 dni.

Zegar godzinowy można konfigurować we właściwościach „Clock" CPU. Użytkownik może włączyć tryb czasu letniego i wyspecyfikować czas początkowy i końcowy obowiązywania czasu letniego. W celu nastawienia zegara godzinowego, użytkownik musi być *online* i mieć otwarty widok „Online & diagnostics" CPU. Zegar nastawia się za pomocą funkcji „Set time of day".

Pamięć systemu i zegara

Bajty „pamięci systemu" i „pamięci zegara" uaktywnia się we właściwościach CPU. Program użytkownika ma dostęp do pojedynczych bitów tych funkcji.

- Użytkownik może przeznaczyć jeden bajt pamięci M na pamięć systemu. Bajt pamięci systemu zawiera cztery bity, do których ma dostęp program użytkownika:
  - Bit „Always off" jest zawsze ustawiony na 0.
  - Bit „Always on" jest zawsze ustawiony na 1.
  - Bit „Diagnostic event changed" („Diag changed") jest ustawiony na 1 przez jeden cykl programu po tym, jak CPU zarejestruje zdarzenie diagnostyczne.
  - Bit „First scan" jest ustawiony na 1 przez czas trwania pierwszego cyklu programu po zakończeniu rozruchowego OB. (Po wykonaniu pierwszego cyklu programu bit „first scan" jest ustawiany na 0.)
- Użytkownik może przeznaczyć jeden bajt pamięci M na pamięć zegara. Każdy bit bajtu skonfigurowanego jako pamięć zegara generuje ciąg impulsów prostokątnych. Jest dostępnych osiem częstotliwości fali prostokątnej generowanej przez bajt pamięci zegara; od 0,5 Hz (wolno) do 10 Hz (szybko). Użytkownik może wykorzystywać te bity jako bity sterujące, zwłaszcza z instrukcjami dotyczącymi zboczy, w celu cyklicznego wyzwalania akcji w programie użytkownika.

CPU inicjalizuje te bajty na początku cyklu programu.

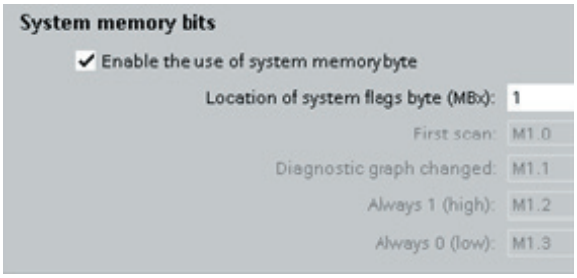
⚠ OSTROŻNIE

Nadpisanie bitów pamięci systemu lub pamięci zegara może uszkodzić dane w tych funkcjach i spowodować, że program użytkownika będzie działać nieprawidłowo, co z kolei może doprowadzić do zniszczenia sprzętu i obrażeń ciała personelu.

Ponieważ zarówno pamięć zegara, jak i pamięć systemu nie są zarezerwowanymi obszarami pamięci M, więc instrukcje i komunikaty mogą wpisywać dane do tych miejsc i zniszczyć istniejące tam dane. Aby zapewnić poprawne działanie tych funkcji, należy unikać zapisywania danych do tych miejsc i zawsze instalować wyłącznik bezpieczeństwa procesu lub maszyny.

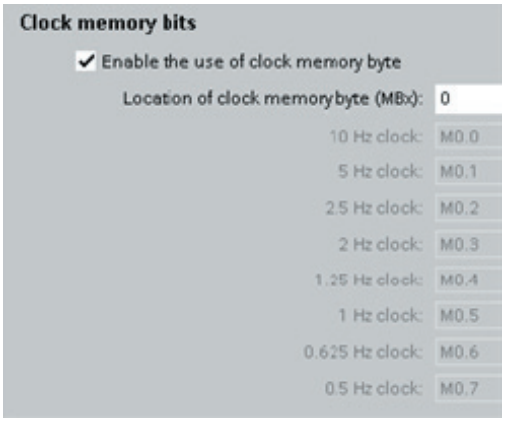
Pamięć systemu konfiguruje jeden bajt, włączając (tj. ustawiając wartość 1) jego bity w następujących przypadkach:

- *First scan*: Włączony dla pierwszego cyklu programu po rozruchu.
- *Diagnostic graph changed*
- *Always 1* (wysoko): zawsze jest włączony
- *Always 0* (nisko): zawsze jest wyłączony



Pamięć zegara ma tak skonfigurowany bajt, że pojedyncze bity na przemian włączają się i wyłączają z ustalona częstotliwością.

Każdy znacznik zegara generuje falę prostokątną na odpowiedniej pozycji pamięci M. Te bity mogą być wykorzystane jako bity sterujące, zwłaszcza z instrukcjami dotyczącymi zboczy, w celu cyklicznego wyzwalania akcji w kodzie użytkownika.



Konfiguracja wartości wyjściowych podczas przejścia z RUN do STOP

Użytkownik może kontrolować zachowanie wyjść po zmianie trybu z RUN na STOP. Dla każdego wyjścia CPU, SB lub SM można określić czy jego stan ma zostać zachowany bez zmian, czy zastąpiony inną wartością.

- Wpisanie na wyjście nowej wartości (domyślnie): dla każdego wyjścia (kanału) CPU, SB lub SM, użytkownik wprowadza wartość, która zastąpi bieżący stan. Wartością domyślną zastępującą bieżący stan jest OFF (wyłączony) dla wyjścia cyfrowego i 0 dla wyjścia analogowego.
- Zachowanie bez zmiany ostatniego stanu wyjść: Wyjścia zachowują swoje bieżące wartości z chwili zmiany trybu z RUN na STOP.

Konfiguracji sposobu zachowania wyjść dokonuje się w „Device Configuration”. W celu skonfigurowania wyjścia dowolnego urządzenia należy wybrać to urządzenie, a następnie zakładkę „Properties”. Po zmianie trybu CPU z RUN na STOP, CPU zachowuje obraz procesu i zgodnie z konfiguracją zapisuje odpowiednie wartości wyjść cyfrowych i analogowych.

3.1.4 Ochrona hasłem CPU S7-1200

CPU zapewnia 3-poziomową ochronę przed niepowołanym dostępem do pewnych funkcji. Użytkownik podczas konfigurowania poziomu bezpieczeństwa i hasła CPU, ogranicza funkcje i obszary pamięci, do których można mieć dostęp bez podania hasła. Podczas wpisywania hasła nie jest rozróżniana wielkość liter.

Na każdym poziomie zabezpieczenia można ustalić funkcje, które są dostępne bez podania hasła. Warunkami domyślnymi jest brak ograniczeń i zabezpieczenia hasłem dostępu do CPU. By wprowadzić zabezpieczenia należy skonfigurować własności CPU i zdefiniować hasło.

Wprowadzenie hasła przez sieć nie zagraża ochronie CPU hasłem. W danej chwili tylko jeden użytkownik może mieć nieograniczony dostęp do CPU chronionej hasłem.

Komunikacja PLC-PLC (za pomocą instrukcji komunikacyjnych w bloku kodu) nie jest ograniczona przez poziom zabezpieczenia CPU. Nie jest również ograniczona funkcjonalność HMI. Wprowadzenie poprawnego hasła daje dostęp do wszystkich funkcji.

Poziom bezpieczeństwa	**Ograniczenia dostępu**
Brak zabezpieczenia	Bez podania hasła dozwolony jest pełen dostęp
Zabezpieczenie przez zapisem	Bez podania hasła dozwolony jest odczyt CPU, dostęp do HMI oraz komunikacja PCL-PCL. Hasło jest wymagane dla modyfikacji (zapisu do) CPU i zmiany trybu pracy CPU (RUN/STOP).
Zabezpieczenie przed zapisem/odczytem	Bez podania hasła dozwolony jest dostęp do HMI oraz pełna komunikacja PCL-PCL. Hasło jest wymagane dla odczytu danych z CPU, modyfikacji (zapisu do) CPU i zmiany trybu pracy CPU (RUN/STOP).

Mając hasło dostępu użytkownik może bez żadnych ograniczeń wykorzystywać instrukcje sterowania procesem, monitorowania i komunikacji. Niektóre funkcje, takie jak „Set time of day/date” nie powinny być blokowane hasłem. Na przykład, w celu zmodyfikowania *tagów* w CPU, który jest chroniony przed odczytem/zapisem, użytkownik musi wprowadzić hasło, ponieważ te funkcje wykonują zapis.

3.2 Przechowywanie danych, obszary pamięci i adresowanie

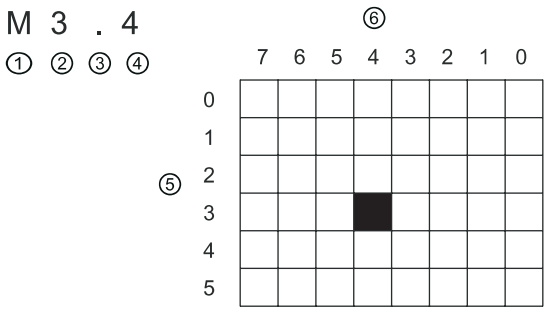
CPU dysponuje kilkoma sposobami przechowywania danych podczas wykonywania programu użytkownika:

- Lokalizacje pamięci: CPU dzieli pamięć na różne specjalizowane obszary – wejścia (I), wyjścia (Q), pamięci bitowej (M), bloków danych (DB) i pamięci lokalnej lub chwilowej (L). Program użytkownika ma dostęp (dla odczytu i zapisu) do danych przechowywanych w tych obszarach.
- Bloki danych (DB): DB mogą być wykorzystywane w programie użytkownika do przechowywania danych. Dane przechowywane w DB nie są wymazywane, gdy blok danych zostaje zamknięty, lub gdy blok kodu skojarzony z tym blokiem danych zostanie zakończony. Są dwie kategorie DB:
  - Globalne DB: przechowują dane, które mogą być wykorzystywane przez inne wszystkie inne bloki.
  - Blok danych *instance* DB: przechowują dane dla określonych FB i ich struktura jest zgodna z parametrami używanymi przez FB.
- Pamięć chwilowa: za każdym razem gdy wywoływany jest blok kodu, system operacyjny CPU alokuje chwilową (lokalną) pamięć (L), która jest wykorzystywana podczas wykonywania tego bloku. Po zakończeniu wykonywania tego bloku kodu, CPU realokuje pamięć lokalną na potrzeby innych bloków.
- Odwołania, takie jak as I0.3 i Q1.7 realizują dostęp do obrazu procesu. W celu dostępu do fizycznego wejścia lub wyjścia do odwołania należy dodać „:P” (na przykład: I0.3:P, Q1.7:P, lub „Stop:P”).

Różne obszary pamięci mają swoje unikalne adresy. Program użytkownika wykorzystuje te adresy w celu uzyskania dostępu do informacji przechowywanych w tych miejscach pamięci.

Obszar pamięci	Opis	Wymuszony	Trwały
I obraz procesu – wejście	Skopiowany na początku cyklu programu stan wejść fizycznych	Tak	Nie
I_:P (fizyczne wejście)	Bezpośredni odczyt wejściowych punktów fizycznych CPU, SB, SM	Nie	Nie
Q obraz procesu – wyjście	Stan skopiowany na początku cyklu programu do wyjść fizycznych	Tak	Nie
Q_:P (fizyczne wyjście)	Bezpośredni zapis do wyjściowych punktów fizycznych CPU, SB, SM	Nie	Nie
M pamięć bitowa	Pamięć sterująca i danych	Nie	Tak
L pamięć chwilowa	Chwilowe dane dla bloku, lokalne dla tego bloku	Nie	Nie
DB blok danych	Pamięć danych, jak również parametrów dla FB	Nie	Tak

Aby uzyskać dostęp do pojedynczego bitu w obszarze pamięci należy podać jego adres, który składa się z identyfikatora obszaru pamięci, adresu bajtu i numeru bitu. Przykład dostępu do bitu (zwanego również adresowaniem *byte*.*bit*) jest pokazany poniżej. W tym przykładzie po identyfikatorze obszaru pamięci i adresie bajtu (I = = wejście i 3 = bajt 3) podany jest, oddzielony kropką („."), adres bitu (bit 4).



① Identyfikator obszaru pamięci
② Adres bajtu: bajt 3 (trzeci bajt)
③ Kropka rozdzielająca adres bajtu od numeru bitu
④ Położenie bitu w bajcie (bit 4 z 8)
⑤ Bajty obszaru pamięci
⑥ Bity wybranego bajtu

Użytkownik może uzyskać dostęp do danych zawartych w większości obszarów pamięci (I, Q, M, DB i L) jako bajtów, słów lub podwójnych słów stosując format „adresowania bajtowego". W celu uzyskania dostępu do bajtu, słowa lub podwójnego słowa w pamięci, należy podać adres w podobny sposób, jaki stosuje się do adresowania bitów. Ten adres zawiera identyfikator obszaru, oznaczenie rozmiaru danych i adres bajtu początkowego bajtu, słowa lub podwójnego słowa. Rozmiar oznacza się jako bajt (B), słowo (W) lub podwójne słowo (DW) (przykładowo: IB0, MW20, QD8).

W celu uzyskania bezpośredniego dostępu do fizycznych wejść lub fizycznych wyjść, do adresu lub *tagu* należy dołączyć „:P" (na przykład: I0.3:P, Q1.7:P lub „Stop:P").

Dostęp do danych w obszarach pamięci CPU

Portal TIA umożliwia programowanie symboliczne. Typowo, w *tagach* PLC, bloku danych lub blokach OB, FC lub FB są tworzone *tagi*. Te *tagi* zawierają nazwę, typ danych, przesunięcie i komentarz. Ponadto w bloku danych można wyspecyfikować wartość początkową. Te *tagi* można wykorzystywać podczas programowania, podając nazwę tagu jako parametr instrukcji. Opcjonalnie, jako parametr można również podać argument bezwzględny (pamięć, obszar, rozmiar i przesunięcie). W przykładach podanych w kolejnych częściach przedstawiono w jaki sposób podawać argument bezwzględny. Na początku argumentu bezwzględnego, program edytora automatycznie dostawia znak %. W programie edytora można przełączać aktualny widok na jeden z trzech: symboliczny, bezwzględny i symboliczny lub bezwzględny.

I (**obszar wejściowy obrazu procesu**): CPU próbkuje stan punktów wejściowych peryferii (fizycznych) tuż przed wykonaniem cyklicznego OB w każdym cyklu programu. Użytkownik ma dostęp do bitów, bajtów, słów i podwójnych słów należą-

cych do obszaru wejściowego obrazu procesu. Dopuszczalny jest zarówno zapis jak i odczyt danych, ale zwykle dane z obszaru wejściowego obrazu procesu są tylko odczytywane.

bit	I[adres bajtu].[adres bitu]	I0.1
bajt, słowo lub podwójne słowo	I[rozmiar][adres startowego bajtu]	IB4, IW5 lub ID12

Dołączając do adresu „:P" można bezpośrednio odczytywać cyfrowe i analogowe wejścia CPU, SB lub SM. Różnica w dostępie przy wykorzystaniu adresowania I_:P zamiast I polega na tym, że dane są pobierane bezpośrednio z odczytywanych punktów, a nie z obszaru wejściowego obrazu procesu. Ponieważ dane są odczytywane bezpośrednio ze swojego źródła, a nie z kopii utworzonej podczas ostatniego uaktualniania obszaru wejściowego obrazu procesu, więc dostęp poprzez I_:P jest nazywany „bezpośrednim odczytem".

Ponieważ stan fizycznych punktów wejściowych jest ustawiany bezpośrednio z urządzeń zainstalowanych na obiekcie i połączonych z tymi punktami, więc zapis do tych punktów jest zabroniony. Inaczej mówiąc dostęp poprzez I_:P może być tylko odczytem, w przeciwieństwie do dostępu poprzez I, który obejmuje zarówno odczyt, jak i zapis.

Dostęp poprzez I_:P jest również ograniczony do takiej liczby wejść, która jest obsługiwana przez pojedynczą CPU, SB lub SM zaokrąglonej w górę do najbliższego pełnego bajtu. Na przykład, jeżeli wejścia 2 DI / 2 DQ SB są tak skonfigurowane, że ich adresowanie rozpoczyna się od I4.0, to te punkty wejściowe są dostępne jako I4.0:P i I4.1:P lub jako IB4:P. Dostęp do I4.2:P ÷ I4.7:P nie jest odrzucany, ale nie ma żadnego sensu ponieważ te punkty nie są używane. Dostęp do IW4:P i ID4:P jest zabroniony ponieważ jest przekroczony bajt przesunięcia powiązany z tym SB.

Dostęp poprzez I_:P nie wpływa na wartości pamiętane w obszarze wejściowym obrazu procesu.

bit	I[adres bajtu].[adres bitu]:P	I0.1:P
bajt, słowo lub podwójne słowo	I[rozmiar][adres startowego bajtu]:P	IB4:P, IW5:P lub ID12:P

Q (**obszar wyjściowy obrazu procesu**): CPU kopiuje wartości pamiętane w obszarze wyjściowym obrazu procesu do fizycznych punktów wyjściowych. Użytkownik ma dostęp do bitów, bajtów, słów i podwójnych słów należących do obszaru wyjściowego obrazu procesu. Dla obszaru wyjściowego obrazu procesu dopuszczalny jest zarówno zapis jak i odczyt danych.

bit	Q[adres bajtu].[adres bitu]	Q1.1
bajt, słowo lub podwójne słowo	Q[rozmiar][adres startowego bajtu]	QB5, QW10, QD40

Dołączając do adresu „:P" można bezpośrednio zapisywać dane do cyfrowych i analogowych wyjść CPU, SB lub SM. Różnica w dostępie przy wykorzystaniu adresowania Q_:P zamiast Q polega na tym, że dane, oprócz wpisania do obszaru wyjściowego obrazu procesu są również przesyłane bezpośrednio do adresowanych punktów (są zapisywane do dwóch miejsc). Ponieważ dane są przesyłane bezpośrednio do punktów docelowych, które nie muszą czekać na kolejne uaktualnienie obszaru wyjściowego obrazu procesu, więc dostęp poprzez Q_:P jest nazywany „bezpośrednim zapisem".

Ponieważ wyjściowe punkty fizyczne bezpośrednio sterują urządzeniami zainstalowanymi na obiekcie, które są do tych punktów podłączone, więc odczyt tych punktów jest zabroniony. Inaczej mówiąc dostęp poprzez Q_:P może być tylko zapisem, w przeciwieństwie do dostępu poprzez Q, który obejmuje zarówno odczyt, jak i zapis.

Dostęp poprzez I_:P jest również ograniczony do takiej liczby wyjść, która jest obsługiwana przez pojedynczą CPU, SB lub SM zaokrąglonej w górę do najbliższego pełnego bajtu. Na przykład, jeżeli wyjścia 2 DI / 2 DQ SB są tak skonfigurowane, że ich adresowanie rozpoczyna się od Q4.0, to te punkty wyjściowe są dostępne jako Q4.0:P i Q4.1:P lub jako QB4:P. Dostęp do Q4.2:P ÷ Q4.7:P nie jest odrzucany, ale nie ma żadnego sensu ponieważ te punkty nie są używane. Dostęp do QW4:P i QD4:P jest zabroniony ponieważ jest przekroczony bajt przesunięcia powiązany z tym SB.

Dostęp poprzez Q_:P wpływa zarówno na stan wyjść fizycznych, jak i na wartości pamiętane w obszarze wyjściowym obrazu procesu.

bit	Q[adres bajtu].[adres bitu]:P	Q1.1:P
bajt, słowo lub podwójne słowo	Q[rozmiar][adres startowego bajtu]:P	QB5:P, QW10:P lub QD40:P

M (**obszar pamięci bitowej**): Obszaru pamięci bitowej M używa się do sterowania zarówno przekaźników, jak i danych do przechowywania pośredniego statusu operacji lub innych informacji sterujących. Użytkownik ma dostęp do bitów, bajtów, słów i podwójnych słów należących do obszaru pamięci bitowej. Dla pamięci M dopuszczalny jest zarówno zapis jak i odczyt danych.

bit	M[adres bajtu].[adres bitu]	M26.7
bajt, słowo lub podwójne słowo	M[rozmiar][adres startowego bajtu]	MB20, MW30, MD50

Temp (**pamięć chwilowa**): CPU zapewnia pamięć chwilową (lokalną) dla każdej z trzech grup priorytetów OB: 16 kB dla rozruchu i cyklu programu, włączając w to FB i FC; 4 kB dla zdarzeń przerwań standardowych, włączając w to FB i FC i 4 kB dla zdarzeń przerwań błędów, włączając w to FB i FC.

Pamięć Temp jest podobna do pamięci M z jednym zasadniczym wyjątkiem: pamięć M ma charakter globalny, a pamięć Temp jest pamięcią lokalną:

- Pamięć M: Dowolny OB, FC lub FB ma dostęp do danych w pamięci M, co oznacza, że dane są dostępne globalnie dla wszystkich elementów programu użytkownika.
- Pamięć Temp: Dostęp do danych w pamięci Temp mają tylko te OB, FC lub FB, które stworzyły lub zadeklarowały lokalizację pamięci Temp. Lokalizacje pamięci Temp pozostają lokalne i nie są współdzielone przez różne bloki kodu, nawet jeśli jeden blok kodu wywołuje inny blok kodu. Na przykład: jeśli OB wywołuje FC, to FC nie ma dostępu do pamięci chwilowej należącej do wywołującego OB.

CPU alokuje pamięć chwilową wtedy, kiedy jest ona potrzebna. CPU alokuje pamięć chwilową dla bloku kodu w chwili, gdy blok kodu jest uruchamiany (dla OB) lub jest wywoływany (dla FC lub FB). Alokacja pamięci chwilowej dla bloku kodu może dotyczyć tej samej lokalizacji pamięci Temp, która była poprzednio używa-

na przez inne OB, FC lub FB. CPU nie inicjalizuje pamięci chwilowej w momencie alokacji i w związku z tym mogą się w niej znajdować jakieś wartości.

Dostęp do pamięci chwilowej jest możliwy wyłącznie za pomocą adresowania symbolicznego.

DB (**blok danych**): Pamięć DB stosuje się do pamiętania różnych typów danych, włączając w to pośredni status operacji lub inne parametry sterujące dla FB i struktury danych wymagane przez wiele instrukcji, takich jak timery i liczniki. Użytkownik może określić, czy blok danych będzie umożliwiał odczyt/zapis, czy też będzie tylko do odczytu. Użytkownik ma dostęp do bitów, bajtów, słów i podwójnych słów należących do pamięci bloku danych. Dla bloków danych typu czytaj/zapisz dopuszczalny jest zarówno zapis jak i odczyt danych. Dla bloków danych typu czytaj dozwolony jest tylko odczyt danych.

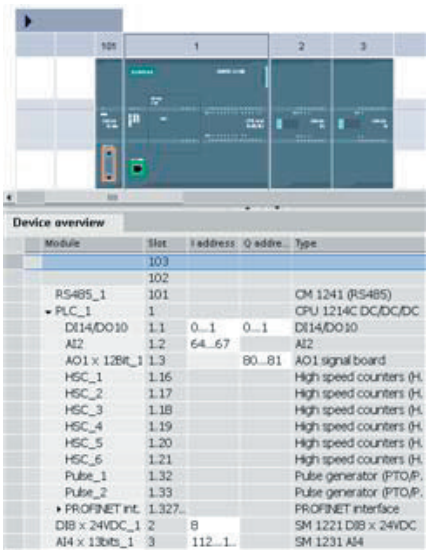
bit	DB[numer bloku danych]. DBX[adres bajtu].[adres bitu]	DB1.DBX2.3
bajt, słowo lub podwójne słowo	DB[numer bloku danych].DB [rozmiar][adres startowego bajtu]	DB1.DBB4, DB10.DBW2, DB20.DBD8

Adresowanie I/O w CPU i modułów I/O

Kiedy na ekranie konfiguracyjnym są dodawane CPU i moduły I/O, to automatycznie są alokowane adresy I oraz Q.

- Wejściom CPU odpowiadają bity adresowane od I0.0 do I0.7 i od I1.0 do I1.5 (łącznie 14 punktów).
- Wyjściom CPU odpowiadają bity adresowane od Q0.0 do Q0.7 i od Q1.0 do Q1.1 (łącznie 10 punktów).
- Wejściom analogowym CPU odpowiadają słowa o adresach IW64 i IW66 (2 punkty analogowe, łącznie 4 bajty).
- Wejścia DI16 są adresowane od I8.0 do I9.7.
- AI4 / AO2 – wejścia to IW112, IW114, IW116, IW118, a wyjścia to QW112 i QW114
- DI8 / D08 – zakres wejść jest od I16.0 do I17.7, a wyjść od Q16.0 do Q17.7.

Na rysunku przedstawiono przykładowy CPU 1214C z dwoma SM.



Użytkownik może zmienić domyślne adresowanie wybierając na ekranie konfiguracyjnym pole adresu i wpisując tam nowe liczby. Wejściom i wyjściom cyfrowym przypisuje się pełne bajty, niezależnie od tego, czy moduł ma wszystkie punkty, czy nie. Wejścia i wyjścia analogowe tworzą grupy po dwa punkty (4 bajty). W podanym przykładzie, użytkownik może zmienić adres DI16 z 8..9 na 2..3. Program asystuje użytkownikowi i zmienia zakres adresów, które mają niewłaściwy rozmiar lub wchodzą w konflikt z innymi adresami.

3.3 Typy danych

Stosuje się różne typy danych, które cechują się z jednej strony rozmiarem, a z drugiej sposobem interpretacji danych. Każdy parametr instrukcji jest daną co najmniej jednego typu, a niektóre parametry mogą przyjmować jeden z kilku typów danych. Jeżeli kursor zostanie przytrzymany nad polem parametru instrukcji, to zostanie wyświetlona informacja jaki typ danych może przyjmować ten parametr.

Parametr formalny jest identyfikatorem powiązanym z instrukcją, oznaczającym położenie danej wykorzystywanej przez instrukcję (przykład: parametr wejściowy IN1 instrukcji ADD). Parametr faktyczny jest miejscem w pamięci lub stałą zawierającą daną wykorzystywaną przez instrukcję (przykład: %MD400 „Number_ of_Widgets”). Typ danej parametru faktycznego podawanego przez użytkownika musi odpowiadać dozwolonemu typowi parametru formalnego danej instrukcji.

Podając parametr faktyczny użytkownik musi wyspecyfikować albo *tag* albo bezwzględny adres pamięci. *Tagi* wiążą nazwy symboliczne (nazwy *tagów*) z typem danych, obszarem pamięci, przesunięciem pamięci oraz komentarzem i mogą być tworzone albo w edytorze *tagów* PLC, albo edytorze interfejsu bloku (OB, FC, FB lub DB). Jeżeli zostanie wprowadzony adres bezwzględny nie skojarzony z żadnym *tagiem*, to trzeba zastosować odpowiedni rozmiar pasujący do dozwolonego typu danych, a domyślny *tag* zostanie utworzony w trakcie wprowadzania.

Można również wprowadzić stałą, która będzie używana jako wiele parametrów wejściowych. Poniższa tablica prezentuje dozwolone, elementarne typy danych wraz z przykładami wprowadzania stałych. Wszystkie, z wyjątkiem *String* są dostępne w edytorze *tagów* PLC i edytorach interfejsów bloków. *String* jest dostępny tylko edytorach interfejsów bloków. W poniższej tablicy zdefiniowano elementarne typy danych.

Typ danej	Rozmiar (w bitach)	Zakres	Przykłady wprowadzania stałych
Bool	1	0 do 1	TRUE, FALSE, 0, 1
Byte	8	16#00 do 16#FF	16#12, 16#AB
Word	16	16#0000 do 16#FFFF	16#ABCD, 16#0001
DWord	32	16#00000000 do 16#FFFFFFFF	16#02468ACE
Char	8	16#00 do 16#FF	‚A', ‚t', ‚@'
Sint	8	-128 do 127	123, -123
Int	16	-32,768 do 32,767	123, -123
Dint	32	-2147483648 do 2147483647	123, -123

Typ danej	Rozmiar (w bitach)	Zakres	Przykłady wprowadzania stałych
USInt	8	0 do 255	123
UInt	16	0 do 65,535	123
UDInt	32	0 do 4,294,967,295	123
Real	32	+/-1,18 x 10 -38 do +/-3,40 x 10 38	123456, -3,4, -1,2E+12, 3,4E-3
Time	32	T#-24d_20h_31m_23s_648ms do T#24d_20h_31m_23s_647ms pamiętanych jako -2147483648 ms do +2147483647 ms	T#5m_30s 5#-2d T#1d_2h_15m_30x_45ms
String	zmienny	0 do 254 znaków po jednym bajcie	‚ABC'

Mimo, że nie jest dostępny jako osobny typ danych, następujący format numeryczny BCD jest akceptowany przez instrukcje konwersji.

Format	Rozmiar (w bitach)	Zakres	Przykłady wprowadzania stałych
Bool	16	-999 do 999	123, -123
Byte	32	-9999999 do 9999999	1234567, -1234567

Format liczb rzeczywistych

Liczby rzeczywiste (lub zmiennoprzecinkowe) są zgodnie z normą ANSI/IEEE 754-1985 reprezentowane przez 32 bity, jako liczby pojedynczej precyzji. Dla CPU liczby rzeczywiste mają postać słów o podwójnej długości. Liczby zmiennoprzecinkowe o pojedynczej precyzji są dokładne do 6 cyfr znaczących. W celu zachowania precyzji, wpisując liczbę zmiennoprzecinkową o pojedynczej precyzji, użytkownik może wprowadzić maksymalnie 6 cyfr znaczących.

Obliczenia, w których występują ciągi liczb o małych i dużych wartościach, mogą dawać niedokładne wyniki. Jest tak, jeśli liczby różnią się o 10 do potęgi x, gdzie x > 6. Na przykład: 100000000 + 1 = 100000000.

Format danych łańcuchowych (*String*)

CPU obsługuje typ danych łańcuchowych STRING pamiętając ciąg znaków, z których każdy ma długość jednego bajtu. Dane typu STRING zawierają także całkowitą liczbę znaków (tj. liczbę znaków w łańcuchu) oraz bieżącą liczbę znaków. Dane typu STRING mają długość do 256 bajtów, która obejmuje całkowitą liczbę znaków (1 bajt), bieżącą liczbę znaków (1 bajt) i do 254 znaków, przy czym każdy znak jest pamiętany w jednym bajcie.

Użytkownik może stosować łańcuchy literałów (stałych) jako parametry instrukcji typu IN, posługując się pojedynczymi znakami cudzysłowu. Na przykład, ‚ABC' jest trójznakowym łańcuchem, który można wykorzystać jako parametr wejściowy IN instrukcji S_CONV. Można również tworzyć zmienne łańcuchowe wybierając typ danych „String" w edytorze interfejsu bloków OB, FC, FB i DB. Nie można natomiast tworzyć łańcucha w edytorze *tagów* PLC. Maksymalny rozmiar łańcucha

w bajtach można wyspecyfikować w momencie deklaracji łańcucha, na przykład: „MyString[10]" określa, że maksymalny rozmiar MyString wynosi 10 bajtów. Jeśli w deklaracji łańcucha nie wystąpi nawias z maksymalną długością łańcucha, to przyjmuje się, że wynosi ona 254.

Całkowita liczba znaków	**Bieżąca liczba znaków**	**Znak 1**	**Znak 2**	**Znak 3**	**...**	**Znak 10**
10	3	‚C' (16#43)	‚A' (16#41)	‚T' (16#54)	...	–
Bajt 0	Bajt 1	Bajt 2	Bajt 3	Bajt 4	...	Bajt 11

Tablice

Użytkownik może utworzyć tablicę zawierającą wiele elementów typu elementarnego. Tablice mogą być tworzone w edytorze interfejsu bloków OB, FC, FB i DB. Nie można natomiast tworzyć tablicy w edytorze *tagów* PLC.

W celu utworzenia tablicy w edytorze interfejsu bloku, należy wybrać typ danych: „Array [lo .. hi] of type", a potem określić „lo", „hi", i „type" w następujący sposób:

- lo – początkowy (najniższy) indeks danych tablicy.
- hi – końcowy (najwyższy) indeks danych w tablicy.
- type – jeden z elementarnych typów danych, jak na przykład BOOL, SINT, UDINT.

Indeksy mogą przyjmować wartości ujemne. Tablice można nazwać w kolumnie „Name" edytora interfejsu bloków. Poniżej przedstawiono przykłady tablic, które można tworzyć w edytorze interfejsu bloków:

Nazwa	**Typ danych**	**Komentarz**
My_Bits	Array [1 .. 10] of BOOL	Ta tablica zawiera 10 danych typu BOOL
My_Data	Array [-5 .. 5] of SINT	Ta tablica zawiera 11 danych typu SINT, łącznie z indeksem 0

Odwołanie do elementów tablicy w programie użytkownika realizuje się zgodnie z następując składnią:

- Array_name[*i*], gdzie *i* oznacza właściwy indeks.

Przykładami ilustrującymi sposób w jakim takie odwołanie może się pojawić w edytorze programu jako parametr wejściowy są:

- #My_Bits[3] – odwołanie do trzeciego bitu tablicy „My_Bits"
- #My_Data[-2] – odwołanie do czwartego SINT tablicy „My_Data"

Symbol # jest dostawiany automatycznie przez edytor programu.

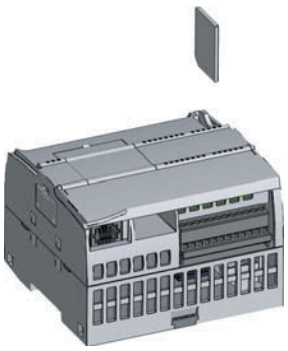
3.4 Zapis i odczyt pamięci

3.4.1 Sposób zapisywania i odczytywania danych w S7-1200

S7-1200 ma kilka cech, które zapewniają, że program użytkownika oraz dane są właściwie przechowywane:

- Pamięć ładowania jest nieulotną pamięcią służącą do przechowywania programu użytkownika, danych i konfiguracji. CPU trwale przechowuje zawartość pamięci ładowania. Rozmiar pamięci ładowania jest określony przez wewnętrzną pamięć ładowania (ILM – *internal load memory*) lub zewnętrzną pamięć ładowania (ELM – *external load memory*). Wielkość ILM zależy od modelu CPU, natomiast rozmiar ELM jest określony pojemnością karty pamięci. Więcej informacji na temat konkretnego modelu CPU zawiera rozdział Dane techniczne.
- Trwała pamięć danych jest skonfigurowaną przez użytkownika pamięcią, która przechowuje dane (które pozostają niezmienione) niezależnie od stanu zasilania. CPU umożliwia przechowywanie w sposób trwały danych o pojemności do 2048 bajtów. Użytkownik może określać, które dane (DB i/lub z pamięci M) mają być zachowane w przypadku zaniku zasilania.
- Pamięć robocza jest pamięcią nieulotną służącą do przechowywania programu użytkownika, bloku danych, dowolnych wartości wymuszonych, zawartości pamięci nietrwałej M i wybranych wartości zapisanych przez program użytkownika. Wielkość pamięci roboczej zależy od modelu CPU.

Można użyć opcjonalnej karty pamięci jako karty Program lub karty Transfer. Musi to być preformatowana karta z firmy Siemens:



- Karta Program: Karta pamięci pełni funkcję pamięci CPU; wszystkie funkcje CPU są sterowane przez kartę. Karta musi pozostawać zainstalowana w CPU.
- Karta Transfer: karta pamięci służy do przeniesienia zapamiętanego projektu z karty do CPU; następnie karta musi być usunięta. W ten sposób można przenieść projekt do wielu CPU bez konieczności użycia STEP 7 Basic.

W celu zainstalowania karty, należy otworzyć górna pokrywę CPU i włożyć kartę w gniazdo. Gniazdo zatrzaskowe typu *push-push* ułatwia wkładanie i wyjmowanie karty. Karta ma wcięcie uniemożliwiające odwrotną instalację.

⚠	**OSTRZEŻENIE**
Jeżeli karta (skonfigurowana jako Program lub jako Transfer) zostanie zainstalowana do pracującej CPU, to CPU przejdzie w tryb STOP. Urządzenia sterujące mogą ulec uszkodzeniu stwarzając niebezpieczną sytuację, która może doprowadzić do nieprzewidywalnego działania sprzętu. Takie nieprzewidywalne działanie może spowodować śmierć lub poważne uszkodzenie ciała personelu i/lub zniszczenie mienia.	

3.4.2 Zastosowanie karty pamięciowej jako nośnika programów

Karta pamięci zastosowana jako nośnik programów działa tak, jak pamięć CPU. Kiedy karta zostanie usunięta z CPU, wtedy CPU traci całą zawartość pamięci projektu.

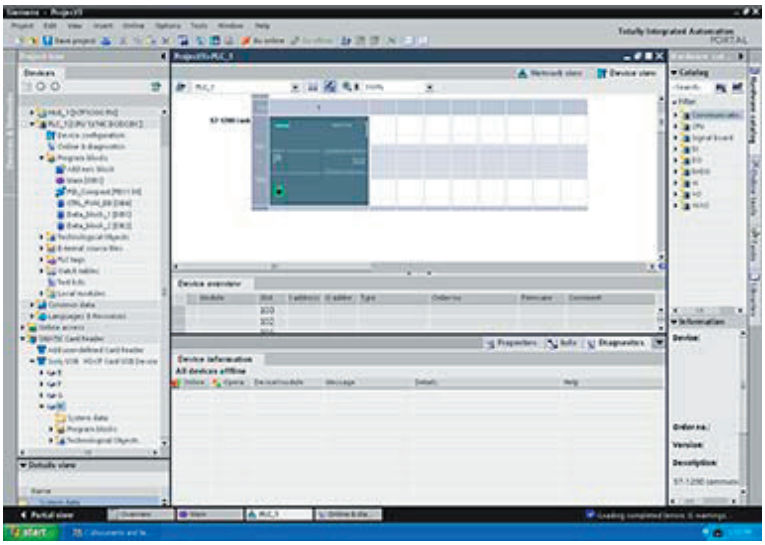
Należy sprawdzić, czy karta nie jest zabezpieczona przed zapisem. Przełącznik zabezpieczający musi być w położeniu przeciwnym do pozycji „Lock", tak jak to pokazano na rysunku po prawej stronie.

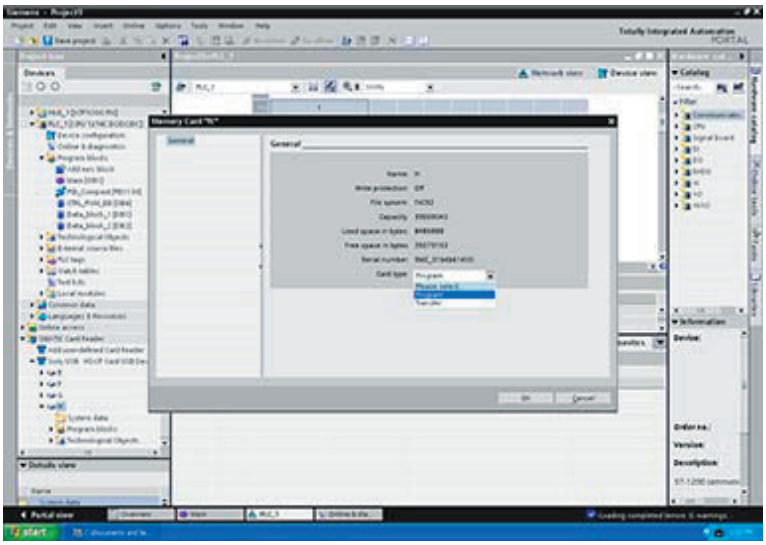
Przed skopiowaniem dowolnego elementu programu na sformatowaną kartę pamięci, należy wymazać z karty pamięci poprzednio zapisane elementy programu (z wyjątkiem plików użytkownika).

OSTROŻNIE
Wyładowania elektrostatyczne mogą zniszczyć kartę pamięci lub CPU. Podczas manipulowania kartą pamięci należy zapewnić odpowiednie uziemienie poprzez stosowanie mat przewodzących i/lub noszenie opasek uziemiających na nadgarstki. Karta pamięci powinna być przechowywana w przewodzącym pojemniku.

W celu utworzenia karty „Program", za pomocą CPU, należy wykonać następujące kroki:

1. Do czytnika kart pamięci włożyć czystą kartę pamięci.
2. W urządzeniu programującym STEP 7 Basic, w „Project tree" dokonać następującego wyboru menu:
   – (kliknąć prawym klawiszem myszy) „SIMATIC Card Reader"
   – „Properties"
   – W menu „Card Type" wybrać „Program"





3. Wyłączyć zasilanie CPU.
4. Włożyć kartę pamięci do CPU.
5. Włączyć zasilanie CPU.
6. Załadować do CPU projekt z urządzenia STEP 7 Basic.

Portal TIA ładuje na kartę pamięci projekt, który zawiera program użytkownika, konfigurację sprzętową i wszystkie wymuszone wartości. Karta pamięci musi pozostać w CPU.

OSTROŻNIE
Jeżeli do CPU zostanie zainstalowana pusta karta pamięciowa, to CPU przejdzie do STOP. Ponadto wyłączenie i włączenie zasilania CPU spowoduje, że projekt aktualnie rezydujący w CPU zostanie przeniesiony do karty pamięci (teraz jest to domyślnie karta „Program"). Cała pamięć projektu jest teraz załadowana na kartę „Program". Jeżeli teraz karta zostanie usunięta z CPU, to CPU utraci całą zawartość pamięci projektu. Jeżeli do CPU zostanie zainstalowana pusta karta pamięciowa, to CPU przejdzie w tryb STOP. CPU nie może zostać przestawiona w tryb RUN dopóty, dopóki karta pamięci nie zostanie usunięta.

3.4.3 Zastosowanie karty pamięciowej jako nośnika danych transferowych

Karta pamięci zastosowana jako karta transferowa, służy do kopiowania (i uaktualniania) projektu użytkownika do wielu CPU.

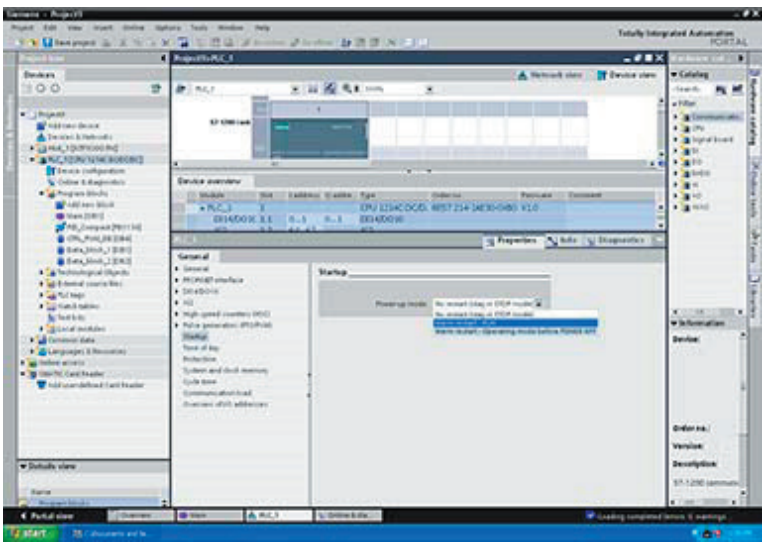
Jeżeli któryś z bloków lub któraś z wymuszonych wartości zapisanych w pamięci karty różni się od bloków lub wymuszonych wartości w CPU, to wszystkie bloki z karty pamięci są kopiowane do CPU.

- Jeżeli z karty pamięci został przesłany blok kodu, to blok kodu w pamięci stałej zostaje zastąpiony.
- Jeżeli z karty pamięci został przesłany blok danych, to blok danych w pamięci stałej zostaje zastąpiony, a cała pamięć M jest kasowana.
- Jeżeli z karty pamięci został przesłany blok systemowy, to blok systemowy i wartości wymuszone w pamięci stałej zostają zastąpione, a cała pamięć trwała jest kasowana.

Tworzenie karty transferowej za pomocą czytnika kart

W celu utworzenia karty transferowej, za pomocą czytnika kart, należy wykonać następujące kroki:

1. Do czytnika kart pamięci włożyć czystą kartę pamięci.
2. W urządzeniu programującym STEP 7 Basic, w „Project tree” dokonać następującego wyboru menu:
   – (kliknąć prawym klawiszem myszy) PLC
   – „Properties”
   – „Startup”
   – W menu „Power-up mode” wybrać „Warm restart – RUN”



3. Zapisać projekt.
4. W urządzeniu programującym STEP 7 Basic, w „Project tree” dokonać następującego wyboru menu:
   – (kliknąć prawym klawiszem myszy) SIMATIC Card Reader
   – „Properties”
   – W menu „Card Type” wybrać „Transfer”
5. Z projektu *offline* w „Project tree” przeciągnąć „Program Block” do czytnika kart.

Tworzenie karty transferowej w za pomocą CPU i czytnika kart

W celu utworzenia karty Transfer, za pomocą CPU i czytnika kart, należy wykonać następujące kroki:

1. Do czytnika kart pamięci włożyć czystą kartę pamięci.
2. W urządzeniu programującym STEP 7 Basic, w „Project tree" dokonać następującego wyboru menu:
   – (kliknąć prawym klawiszem myszy) PLC
   – „Properties"
   – „Startup"
   – W menu „Power-up mode" wybrać „Warm restart – RUN"
3. Zapisać projekt.
4. Wczytać projekt do CPU.

UWAGA

Wczytanie projektu do CPU zawsze tworzy kartę pamięci typu „Program". W celu utworzenia karty pamięci transferowej", należy użyć CPU wraz z czytnikiem kart.

5. Włożyć nowo utworzoną kartę pamięci transferowej do czytnika kart.

6. W urządzeniu programującym STEP 7 Basic, w „Project tree" dokonać następującego wyboru menu:
   – (kliknąć prawym klawiszem myszy) SIMATIC Card Reader
   – „Properties"
   – W menu „Card Type" wybrać „Transfer"

Wczytanie projektu do CPU z wykorzystaniem karty pamięciowej skonfigurowanej jako karta transferowa

W celu wczytania projektu do CPU za pomocą karty transferowej, należy wykonać następujące kroki:

1. Wyłączyć zasilanie CPU.
2. Włożyć kartę pamięci do CPU.
3. Włączyć zasilanie CPU.
4. Projekt zostaje przesłany z karty pamięci do CPU.
   CPU jest teraz w trybie MAINTENANCE (miga żółta dioda LED).
5. Usunąć kartę pamięci z CPU.
6. Włączyć zasilanie CPU.
   CPU przechodzi w tryb RUN.

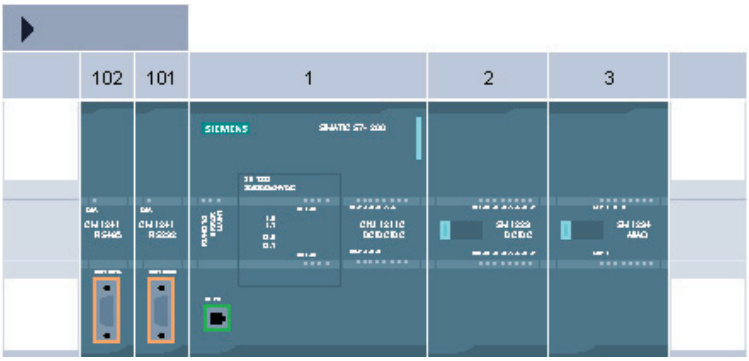
Cały projekt, tj. program użytkownika, konfiguracja sprzętowa i wymuszone wartości zostały wczytane do CPU.

UWAGA

Karta pamięci musi zostać wyjęta zanim będzie można ponownie przestawić CPU w tryb RUN.

Konfiguracja systemu 4

Konfiguracji sterownika PLC dokonuje się dodając CPU i dodatkowe moduły do projektu.



① Moduł komunikacyjny (CM): do 3 modułów, instalowane na pozycjach 101, 102 i 103.
② CPU: na pozycji 1.
③ Port ethernetowy CPU.
④ Płytka sygnałowa (SB): maksymalnie1 sztuka, instalowana w CPU.
⑤ Moduł rozszerzeń (SM) dla cyfrowych lub analogowych I/O: do 8 modułów, instalowane na pozycjach 2...9 (CPU 1214C obsługuje 8 SM, CPU 1212C obsługuje 2 SM, CPU 1211C nie obsługuje SM).

W celu dokonania konfiguracji urządzenia, należy dodać urządzenie do projektu.

- W widoku portalu należy wybrać „Devices & Networks” i kliknąć „Add device”.

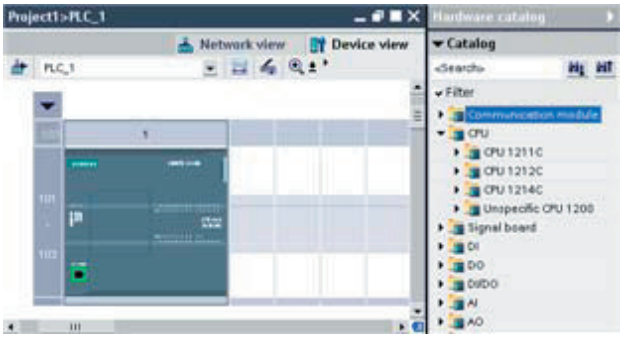
- W widoku projektu należy, pod nazwą projektu, podwójnie kliknąć „Add new device”.

4.1 Dołączanie CPU

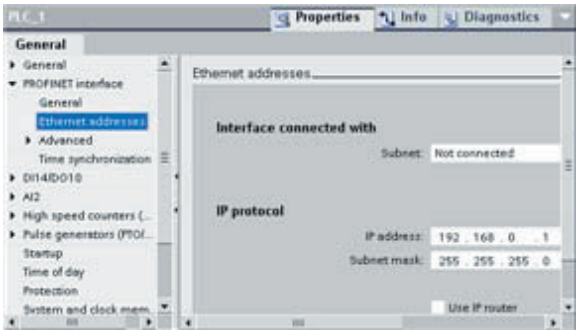
W celu dokonania konfiguracji urządzenia, należy dodać CPU do projektu. Wybranie CPU w oknie dialogowym „Add a new device” powoduje utworzenie wirtualnej listwy montażowej i CPU.

Okno dialogowe „Add a new device”

Okno „Device view” konfiguracji sprzętowej.



Wybór CPU w oknie „Device view” powoduje wyświetlenie w oknie inspekcyjnym właściwości CPU.



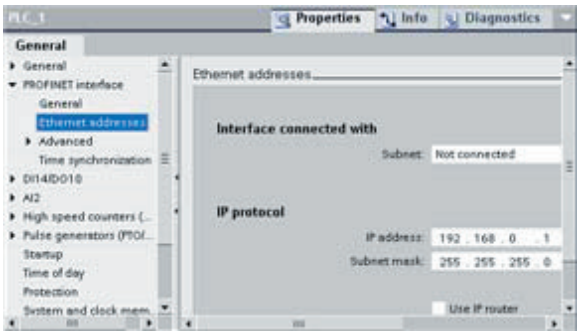
UWAGA

CPU nie ma wstępnie ustawionego adresu IP. Użytkownik musi w trakcie konfiguracji CPU ręcznie wpisać adres IP urządzenia. Jeżeli CPU jest połączona z routerem sieciowym, to należy również wpisać adres IP routera.

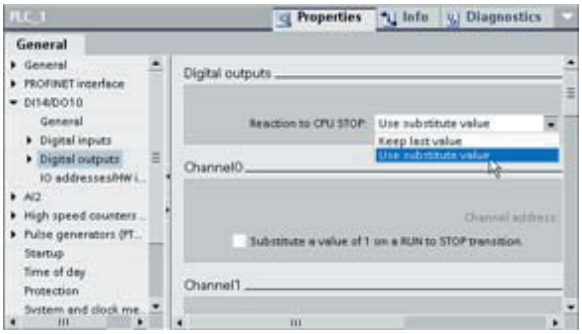
4.2 Konfiguracja CPU

W celu skonfigurowania parametrów operacyjnych CPU, należy w oknie „Device view” (niebieski obrys wokół całej CPU) wybrać CPU, a następnie zakładkę „Properties” (właściwości) okna inspekcyjnego.

Konfiguracja adresu IP CPU i (opcjonalnego) routera.



Konfiguracja lokalnych wyjść CPU.



Edycja właściwości CPU pozwala skonfigurować następujące parametry:

- PROFINET interface: Ustawienie adresu IP CPU i synchronizacji czasu.
- DI, DO i AI: Konfiguracja lokalnych (wbudowanych do CPU) cyfrowych i analogowych I/O.
- *High-speed counters*: Uaktywnianie i konfiguracja szybkich liczników (HSC).
- *Pulse generators*: Uaktywnianie i konfiguracja generatorów impulsowych wykorzystywanych w operacjach wymagających ciągu impulsów (PTO) i modulacji szerokości impulsów (PWM).
- *Startup*: Ustalanie sposobu działania CPU po przejściu ze stanu wyłączenia do stanu włączenia, na przykład po starcie z trybu STOP lub przejściu do trybu RUN po gorącym starcie.
- *Clock*: Ustawianie czasu, strefy czasowej i czasu letniego.
- *Protection*: Ustawianie zabezpieczenia przed odczytem/zapisem oraz hasła dostępu do CPU.
- *System and clock memory*: Uaktywnianie bajtu służącego funkcjom związanym z „pamięcią systemu” (bit „pierwszy cykl programu”, bit „zawsze włączo-

ny" i bit „zawsze wyłączony") oraz uaktywnianie bajtu służącego funkcjom związanym z „pamięcią zegara" (w którym każdy bit przełącza się na zmianę z określoną częstotliwością).

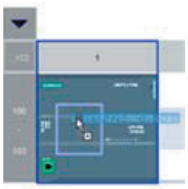
- *Cycle time*: Określanie maksymalnego czasu cyklu lub ustalonego minimalnego czasu cyklu.
- *Communications load*: Przypisywanie procentowej części czasu cyklu zadaniom komunikacyjnym.

4.3 Dodawanie modułów do systemu

W celu dołączenia modułów do CPU należy się posłużyć katalogiem sprzętu (*hardware catalog*). Są trzy typy modułów:

- Moduły rozszerzeń (SM) pozwalają zwiększyć liczbę cyfrowych lub analogowych punktów I/O. Są dołączane z prawej strony CPU.
- Płytki sygnałowe (SB) pozwalają dodawać do CPU porty I/O. SB jest dołączana od strony frontowej CPU.
- Moduły komunikacyjne (CM) pozwalają dodać do CPU port komunikacyjny (RS232 lub RS485). Są dołączane z lewej strony CPU.

W celu dodania modułu do konfiguracji sprzętowej należy wybrać moduł z „Hardware catalog" i albo go podwójnie kliknąć, albo przeciągnąć na podświetloną pozycję.

Rodzaj modułu	Wybór modułu	Instalacja modułu	Rezultat
SM			
SB			

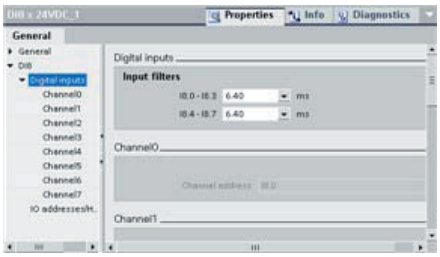
Rodzaj modułu	Wybór modułu	Instalacja modułu	Rezultat
CM			

4.4 Konfiguracja modułów

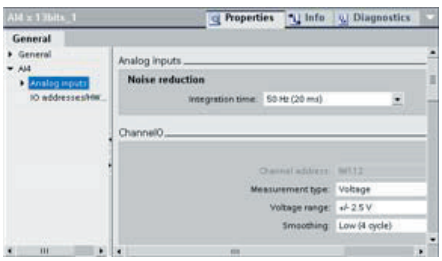
W celu skonfigurowania parametrów operacyjnych modułów, należy w oknie „Device view" wybrać moduł, a następnie zakładkę „Properties" (właściwości) okna inspekcyjnego.

Konfiguracja modułu rozszerzeń

Edycja właściwości modułu pozwala skonfigurować następujące parametry:



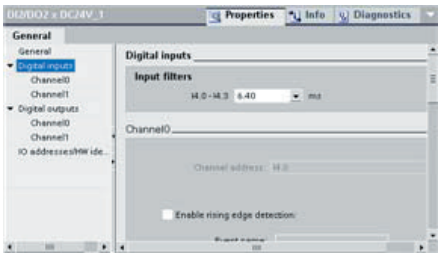
- *Digital inputs*: Uaktywnianie funkcji rejestracji impulsów przez indywidualne wejścia (po zarejestrowaniu impulsu wejście pozostaje w stanie włączonym – on), aż do kolejnego odświeżania obszaru wejściowego obrazu procesu.
- *Digital outputs*: Uaktywnianie funkcji zachowania lub podstawienia wartości wyjściowej indywidualnego wyjścia, przy zmianie trybu z RUN na STOP.



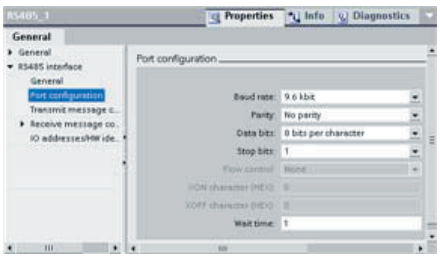
- *Analog inputs*: Konfigurowanie parametrów indywidualnych wejść, takich jak rodzaj pomiaru (napięcie czy prąd), zakres i wygładzanie, a także uaktywnianie diagnostyki przekroczenia zakresu od góry i od dołu.
- *Analog outputs*: Konfigurowanie parametrów indywidualnych wyjść, takich jak rodzaj wyjścia (napięcie czy prąd), a także uaktywnianie diagnostyki, takiej jak badanie zwarcia (przy wyjściu napięciowym) lub przekroczenia zakresu (więcej niż +32511 lub mniej niż -32512)
- *IO/ diagnostic addresses*: Konfigurowanie adresu początkowego wejść i wyjść modułu.

Konfiguracja płytki sygnałowej

Edycja właściwości pozwala skonfigurować następujące parametry:



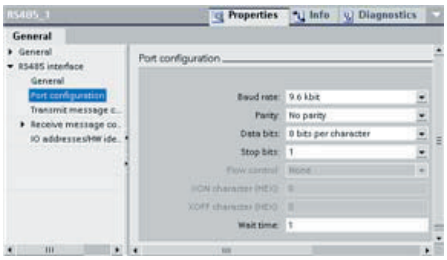
- *Digital inputs*: Konfigurowanie indywid *puts*: Uaktywnianie funkcji zachowania lub podstawienia wartości wyjściowej indywidualnego wyjścia, przy zmianie trybu z RUN na STOP.
- Digital outputs: Uaktywnianie funkcji zachowania lub podstawienia wartości wyjściowej indywidualnego wyjścia, przy zmianie trybu z RUN na STOP.



- *Analog outputs*: Konfigurowanie parametrów indywidualnych wyjść, takich jak rodzaj wyjścia (napięcie czy prąd), a także uaktywnianie diagnostyki, takiej jak badanie zwarcia (przy wyjściu napięciowym) oraz funkcji zachowania lub podstawienia wartości wyjściowej indywidualnego wyjścia, przy zmianie trybu z RUN na STOP.
- *IO/ diagnostic addresses*: Konfigurowanie adresu początkowego wejść i wyjść modułu.

Konfiguracja modułu komunikacyjnego

Edycja właściwości pozwala skonfigurować następujące parametry:

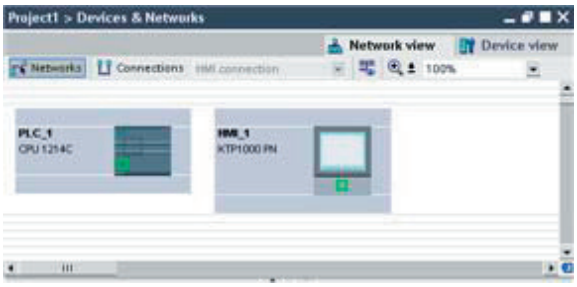
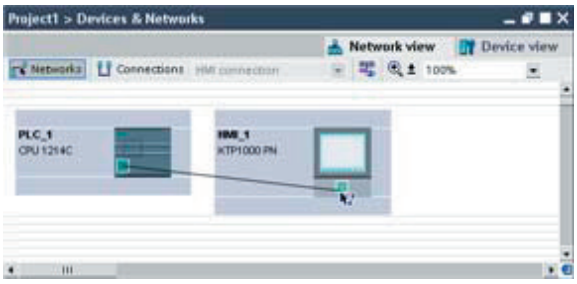
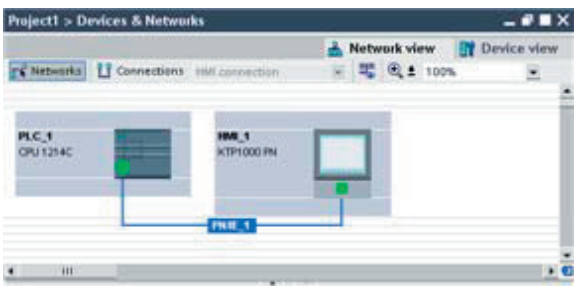


- *Port configuration*: Konfigurowanie parametrów komunikacyjnych, takich jak szybkość transmisji, parzystość, bity danych, bity stopu, sterowanie przepływem, znaki XON i XOFF i czas oczekiwania.
- *Transmit message configuration*: Uaktywnianie i konfigurowanie opcji związanych z nadawaniem.
- *Receive message configuration*: Uaktywnianie i konfigurowanie parametrów początku wiadomości i końca wiadomości.

Te wszystkie parametry konfiguracyjne mogą być zmieniane przez program użytkownika.

4.5 Stworzenie połączenia sieciowego

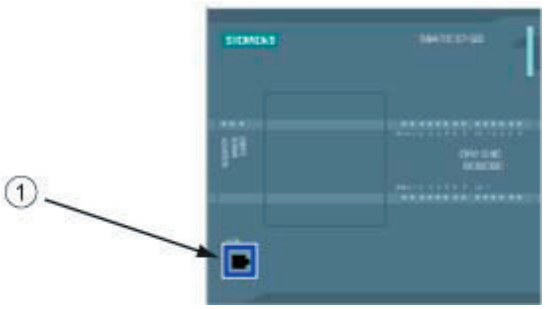
W celu stworzenia połączenia sieciowego między urządzeniami należącymi do projektu, należy z okna „Device configuration" wybrać „Network view". Po utworzeniu połączenia sieciowego należy, wykorzystując zakładkę „Properties" okna inspekcyjnego, skonfigurować parametry sieci.

Czynność	Rezultat
Wybrać „Network view", w celu wyświetlenia urządzeń, które mają być połączone.	
Wybrać port jednego urządzenia i przeciągnąć połączenie do portu drugiego urządzenia.	
Zwolnić przycisk myszy aby utworzyć połączenie.	

4.6 Konfiguracja stałego adresu IP

Konfiguracja interfejsu PROFINET

Po skonfigurowaniu CPU można skonfigurować parametry interfejsu PROFINET. Aby to zrobić należy wybrać port PROFINET, klikając zielony prostokąt PROFINET znajdujący się symbolu CPU. Zakładka „Properties" okna inspekcyjnego wyświetli port PROFINET.



① port PROFINET

Konfiguracja adresu IP

Ethernet (MAC) address: Każde urządzenie znajdujące się w sieci PROFINET ma nadany przez producenta adres MAC (*Media Access Control*) pozwalający je identyfikować. Adres MAC składa się z sześciu grup po dwie cyfry heksadecymalne każda, oddzielonych łącznikami (-) lub dwukropkami (:), podawanych w kolejności w jakiej są nadawane (na przykład 01-23-45-67-89-ab lub 01:23:45:67:89:ab).

Każde urządzenie, podłączone do tej samej sieci PROFINET, musi mieć unikalny adres MAC. Jeżeli dwa urządzenia tej samej sieci PROFINET mają taki sam adres MAC, to wystąpią kłopoty podczas komunikacji.

IP address: Każde urządzenie musi również mieć adres protokołu internetowego (IP). Dzięki temu adresowi urządzenia mogą przesyłać dane w bardziej rozbudowanej sieci.

Każdy adres IP jest podzielony na 8-bitowe segmenty oddzielone kropkami (.) i wyrażane w formacie dziesiętnym (na przykład 211.154.184.16). Pierwszą częścią adresu IP jest Network ID (który identyfikuje sieć w jakiej znajduje się dane urządzenia). Drugą częścią adresu IP jest Host ID (unikalny dla każdego urządzenia w danej sieci). Adres IP 192.168.x.y jest standardowo rozpoznawany jako sieć prywatna, która nie jest dostępna w standardowym Internecie.

Subnet mask: Podsieć (subnet) tworzą logicznie pogrupowane urządzenia podłączone do sieci. Węzły podsieci są zwykle zlokalizowane blisko siebie (fizycznie) w sieci lokalnej (LAN). Maska (mask), zwana również maską podsieci lub maską sieci, określa granice IP podsieci.

Maska podsieci 255.255.255.0 jest zwykle odpowiednia dla małych sieci lokalnych. Oznacza, że wszystkie adresy IP danej sieci są identyfikowane przez ostatni oktet (pole 8-bitowe). Przykładem tego może być maska podsieci 255.255.255.0 i adresy IP od 192.168.2.0 do 192.168.2.255 przypisane urządzeniom małej lokalnej sieci.

Jedyne połączenie pomiędzy różnymi podsieciami możliwe jest poprzez router. Jeżeli wykorzystuje się podsieci, to musi być użyty IP router.

IP router: Routery są łączami pomiędzy sieciami lokalnymi LAN. Dzięki routerowi, komputer znajdujący się w sieci LAN może wysyłać wiadomości do innych sieci, które same mogą się składać z sieci LAN. Jeżeli odbiorca danych nie znaj-

duje się w tej samej sieci LAN, to router przesyła te dane do innej sieci lub grupy sieci, w której mogą zostać przekazane odbiorcy.

Podczas odbierania i wysyłania pakietów danych routery posługują się adresami IP.

IP addresses properties:
W oknie Properties należy wybrać pole konfiguracji „Ethernet address". Portal TIA wyświetla okno dialogowe konfiguracji adresu Ethernet, które pozwala powiązać program zawierający projekt z adresem IP tego CPU, które otrzyma ten projekt.

UWAGA

CPU nie ma wstępnie ustawionego adresu IP. Użytkownik musi w trakcie konfiguracji CPU ręcznie wpisać adres IP urządzenia. Jeżeli CPU jest połączona z routerem sieciowym, to należy również wpisać adres IP routera. Wszystkie adresy IP są konfigurowane po wczytaniu projektu.

Więcej informacji jest podanych w części „Przypisywanie adresów IP urządzeniom programującym i sieciowym"

W podanej poniżej tablicy zdefiniowano parametry adresu IP:

Parametr	Opis	
Podsieć	Nazwa podsieci, do której jest podłączone urządzenie. W celu utworzenia nowej podsieci należy kliknąć przycisk „Add new subnet". Domyślnie jest „Not connected" (nie podłączona). Możliwe są dwa typy połączenia: ● Domyślne „Not connected" realizuje połączenie lokalne. ● Podsieć jest wymagana jeżeli sieć użytkownika składa się z dwóch lub większej liczby urządzeń.	
Protokół IP	Adres IP	Adres IP przypisany CPU
	Maska podsieci	Przypisana maska podsieci
	Użycie routera IP	Kliknąć pole wyboru by zaznaczyć użycie routera IP
	Adres routera	Adres IP przypisany routerowi (jeśli jest użyty)

Koncepcja programowania 5

5.1 Wytyczne dla projektowania programu

Podczas projektowania systemu PLC, użytkownicy mogą wybierać spośród wielu metod i kryteriów. Poniżej podane ogólne wytyczne mają zastosowanie do wielu projektów. Oczywiście każdy projektant musi realizować dyrektywy i procedury obowiązujące w jego firmie, a także stosować zaakceptowane praktyki wynikające z własnego doświadczenia i wiedzy oraz specyficzne dla miejsca realizacji projektu.

Rekomendowane działania	**Zadania**
Podział procesu ub maszyny na segmenty	Użytkownik powinien podzielić proces lub maszynę na mniejsze i w miarę niezależne od siebie segmenty. Te segmenty określają granice między sterownikami i wpływają na specyfikację funkcjonalnego opisu oraz rozdzielanie zasobów.
Stworzenie specyfikacji funkcjonalnej	Użytkownik powinien przygotować opis działania każdego segmentu procesu lub maszyny, takiego jak punkty I/O, funkcjonalny opis działania, opis stanów jakie muszą by osiągnięte przed zezwoleniem na zadziałanie każdego urządzenia wykonawczego (takiego jak solenoid, silnik lub napęd), opis interfejsu operatora i każdego interfejsu z pozostałymi segmentami procesu lub maszyny.
Projekt systemów bezpieczeństwa	Użytkownik powinien zidentyfikować wszystkie urządzenia wymagające specjalnego okablowania dla celów bezpieczeństwa. Trzeba pamiętać o tym, że awaria może wystąpić w takim stanie urządzenia, który zagraża bezpieczeństwu, może nieoczekiwanie uruchomić maszynę lub zmienić jej sposób działania. Wszędzie tam, gdzie niespodziewana lub nieprawidłowa praca maszyny może doprowadzić do zranienia ludzi lub zniszczenia mienia, należy rozważyć zastosowanie elektromechanicznych, nadrzędnych urządzeń zabezpieczających (działających niezależnie od PLC) w celu wyeliminowania zagrożeń. Projektowany system bezpieczeństwa powinien realizować następujące zadania: ● Identyfikować każde niewłaściwe lub nieoczekiwane działanie urządzeń wykonawczych, które może spowodować zagrożenie. ● Identyfikować warunki, w których działanie urządzeń nie stwarza zagrożenia oraz określać sposób wykrywania tych warunków niezależnie od PLC. ● Identyfikować jaki jest wpływ PLC na proces podczas włączenia i wyłączania zasilania oraz jak i kiedy są wykrywane błędy. Te informacje powinny być wykorzystane jedynie podczas projektowania normalnego działania systemu i spodziewanych wydarzeń niestandardowych. Nie należy polegać na scenariuszach „najlepszego przypadku”, bo zmniejszają one bezpieczeństwo działania systemu. ● Umożliwiać ręczne lub elektromechaniczne, nadrzędne i niezależne od PLC, zablokowanie działania stwarzającego niebezpieczeństwo. ● Uzyskiwać z obwodów niezależnych od PLC, odpowiednią informację o stanie procesu, tak by program i interfejsy użytkownika miały dostęp do niezbędnych danych. ● Identyfikować wszystkie inne czynniki wpływające na bezpieczeństwo procesu w celu wyeliminowania zagrożeń.

Rekomendowane działania	Zadania
Specyfikacja stanowiska operatora	W oparciu o wymagania specyfikacji funkcjonalnej, należy utworzyć następujące plany stanowisk operatorskich: ● Plan przeglądowy, na którym jest zaznaczone położenie wszystkich PLC w stosunku do procesu lub maszyny. ● Plan przedstawiający rozmieszczenie na stanowisku operatora urządzeń służących do obsługi, takich jak wyświetlacze, przełączniki i lampki. ● Schematy elektryczne z uwzględnieniem punktów I/O PLC oraz modułów rozszerzeń.
Stworzenie schematu konfiguracji	W oparciu o wymagania specyfikacji funkcjonalnej, należy utworzyć następujące plany konfiguracji urządzeń sterujących: ● Plan przeglądowy, na którym jest zaznaczone położenie wszystkich PLC w stosunku do procesu lub maszyny. ● Plan przedstawiający rozmieszczenie każdego PLC wraz z modułami I/O, z uwzględnieniem wszystkich szafek i innego wyposażenia. ● Schemat elektryczny obejmujący każdy PLC wraz z modułami I/O, zawierający modele urządzeń, adresy komunikacyjne i adresy I/O.
Stworzenie listy nazw symbolicznych	Należy utworzyć listę nazw symbolicznych dla adresów bezwzględnych. Lista powinna obejmować nie tylko fizyczne sygnały I/O, ale również inne elementy (takie jak nazwy tagów) wykorzystywane w programie użytkownika.

5.1.1 Struktura programu użytkownika

Podczas tworzenia programu użytkownika dla wykonywania zadań automatyki, instrukcje programu są umieszczane w blokach kodu:

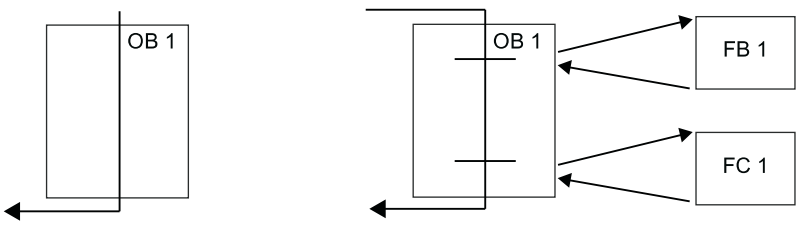
- Blok organizacyjny (OB) reaguje na specyficzne zdarzenia w CPU i może przerwać wykonywanie programu użytkownika. Domyślny blok organizacyjny (OB 1) cyklicznego wykonywania programu użytkownika stanowi podstawową strukturę programu użytkownika i jest jedynym, niezbędnym blokiem kodu wymaganym przez program użytkownika. Jeżeli użytkownik umieści w programie inne OB, to te OB przerywają pracę OB 1. Te inne OB spełniają specyficzne funkcje, takie jak zadania rozruchowe, obsługę przerwań i błędów lub wykonywanie określonego kodu w zadanych odstępach czasu.
- Blok funkcyjny (FB) jest podprogramem wykonywanym wtedy, kiedy jest wywołany z innego bloku kodu (OB, FB lub FC). Blok wywołujący przekazuje do FB parametry oraz identyfikuje określony blok danych (DB) przechowujący dane niezbędne przy tym wywołaniu lub dla tego FB. Zmiana bloku danych *instance* DB pozwala temu samemu FB sterować działaniem grupy urządzeń. Na przykład, jeden FB może sterować kilkoma pompami lub zaworami z wykorzystaniem różnych bloków danych *instance* DB zawierających specyficzne parametry operacyjne dla każdej pompy lub zaworu.
- Funkcja (FC) jest podprogramem wykonywanym wtedy, kiedy jest wywołany z innego bloku kodu (OB, FB lub FC). FC nie ma skojarzonego ze sobą bloku danych *instance* DB. Parametry do FC przekazuje blok wywołujący. Wartość wyjściowa zwracana przez FC musi zostać zapisana pod określony adres pamięci lub do globalnego DB.

Wybór typu struktury programu użytkownika

W oparciu o wymagania aplikacji użytkownik podczas tworzenia swojego programu może wybrać dla niego albo strukturę liniową, albo modularną.

- Program liniowy wykonuje wszystkie instrukcje zadania automatyzacji po kolei – jedną po drugiej. Zwykle, program liniowy umieszcza wszystkie instrukcje w OB przeznaczonym do cyklicznego wykonywania (OB 1).
- Program modułowy wywołuje określone bloki kodu do wykonania specyficznych zadań. W celu stworzenia struktury modularnej, użytkownik musi podzielić złożone zadania automatyzacji na mniejsze podzadania odpowiadające funkcjom technologicznym procesu. Każdy blok kodu zapewnia segment programu dla wykonania podzadania. Użytkownik określa strukturę programu poprzez wywoływanie jednego bloku kodu z innego bloku.

Struktura liniowa: Struktura modularna:



Poprzez stworzenie ogólnego bloku kodu, który może być wykorzystywany w programie użytkownika można uprościć projektowanie i implementację programu użytkownika. Korzystanie z ogólnego bloku kodu ma wiele zalet:

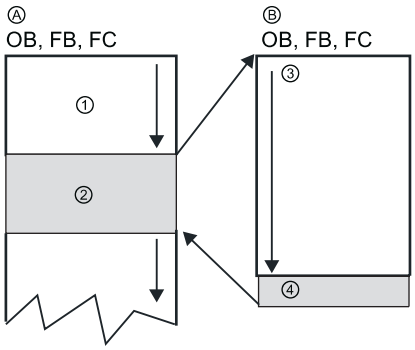
- Do wykonywani standardowych zadań, takich jak sterowanie pompami lub silnikami można stworzyć bloki kodu wielokrotnego użytku. Te ogólne bloki kodu można przechowywać w bibliotece z możliwością wykorzystania w różnych aplikacjach lub rozwiązaniach.
- Jeżeli program użytkownika ma strukturę modularną zgodną z zadaniami funkcjonalnymi, to realizacja programu użytkownika jest łatwiejsza do zrozumienia i zarządzania. Składniki modularne nie tylko pomagają standaryzować konstrukcję programu, ale również sprawiają, że wprowadzanie uaktualnień i modyfikacji kodu programu jest łatwiejsze i szybsze.
- Tworzenie składników modularnych upraszcza debugowanie programu. Jeżeli cały program ma strukturę złożoną ze zbioru modułów programowych, to funkcjonalność każdego bloku kodu można testować zaraz po jego opracowaniu.
- Tworzenie składników modułowych powiązanych z określonymi funkcjami technologicznymi upraszcza i przyspiesza wdrażanie całej aplikacji.

5.1.2 Tworzenie blokowej struktury programu

Poprzez utworzenie FB i FC służących do wykonywania ogólnych zadań, użytkownik stwarza modularne bloki kodu. Jeżeli inne bloki kodu mogą wywoływać te moduły przewidziane do wielokrotnego użycia, to charakter programu staje się

strukturalny. Bloki wywołujące przekazują do bloków wywoływanych parametry związane z określonymi urządzeniami.

A Blok wywołujący
B Blok wywoływany
① Wykonywanie programu
② Operacja wywołująca inny blok
③ Wykonywanie programu
④ Zakończenie bloku (powrót do bloku wywołującego)

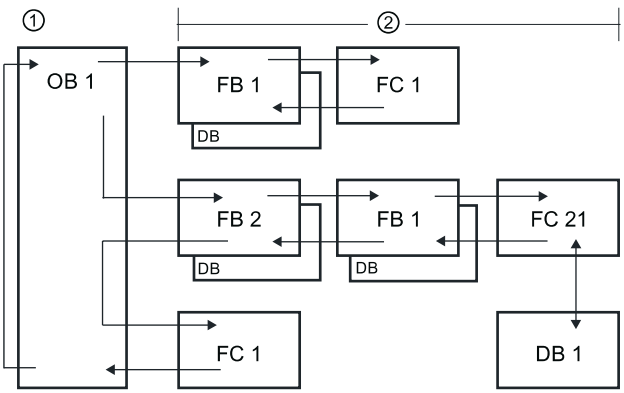


Kiedy blok kodu wywoła inny blok kodu, wtedy CPU wykonuje program zawarty w bloku wywołanym. Po zakończeniu wykonania programu bloku wywołanego, CPU powraca do wykonywania programu bloku wywołującego.

Wykonywanie programu jest kontynuowane od instrukcji następującej po tej instrukcji, przy wykonywaniu której nastąpiło wywołanie bloku.

W celu uzyskania bardziej modularnej struktury programu, wywołania bloków mogą być zagnieżdżone.

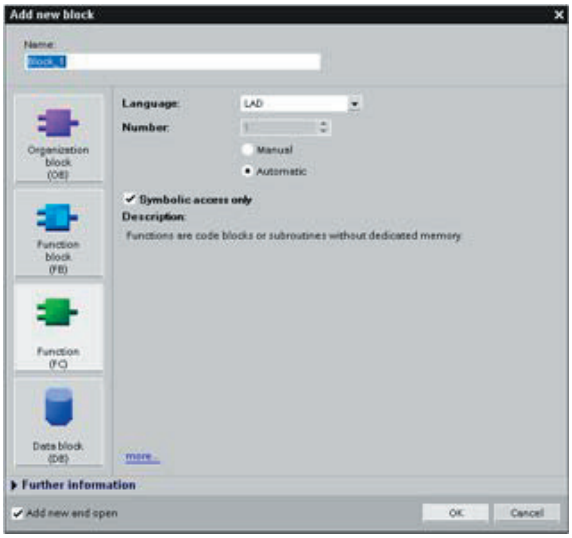
① Start cyklu
② głębokość zagnieżdżenia



Tworzenie bloków kodu do wielokrotnego wykorzystania

W celu utworzenia OB, FB, FC i globalnego DB, należy w „Project nawigator” wybrać okno dialogowe „Add new block” z menu „Program blocks”.

Po utworzeniu bloku kodu, należy wybrać język programowania dla tego bloku. Dla DB nie wybiera się języka programowania ponieważ ten blok przechowuje tylko dane.



5.1.2.1 Blok organizacyjny

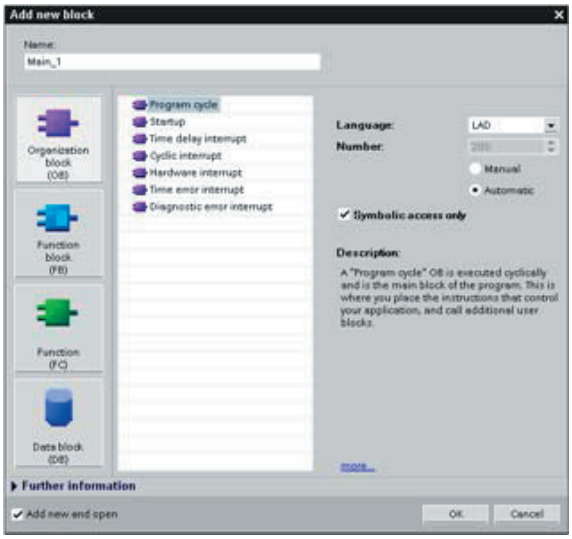
Bloki organizacyjne wprowadzają w programie strukturę. Służą jako interfejs między systemem operacyjnym i programem użytkownika. OB są sterowane zdarzeniami. Zdarzenie, takie jak przerwanie diagnostyczne lub interwału czasowego, powoduje, że CPU wykonuje OB. Niektóre OB mają predefiniowane zdarzenia startowe i działanie.

Cykliczny OB zawiera główny program. Użytkownik może włączyć więcej niż jeden cykliczny OB w swój program użytkownika. W trybie RUN, cykliczne OB działają z najniższym priorytetem i mogą być przerwane przez wszystkie inne typy programów. (Rozruchowy OB nie przerywa działania cyklicznego OB, ponieważ CPU wykonuje rozruchowy OB przed wejściem do trybu RUN.)

Po zakończeniu wykonywania cyklicznego OB, CPU natychmiast zaczyna ponownie wykonywać cykliczny OB. To działanie cykliczne jest normalnym sposobem pracy programowanych sterowników logicznych. W wielu aplikacjach cały program użytkownika jest zlokalizowany w jednym cyklicznym OB.

Użytkownik może utworzyć inne OB przewidziane do wykonywania specyficznych zadań, takich jak zadania rozruchowe, obsługa przerwań i błędów lub wykonywanie określonego kodu programu w stałych odstępach czasu. Te OB przerywają działanie OB cyklu programu.

W celu utworzenia nowego OB w programie użytkownika należy skorzystać z okna dialogowego „Add new block”.



W zależności od przypisanych priorytetów jeden OB może przerwać działanie innego OB. Obsługa przerwania jest zawsze sterowana zdarzeniami. Jeśli zajdzie takie zdarzenie, to CPU przerywa cykl programu użytkownika i wywołuje OB skonfigurowany do obsługi tego zdarzenia. Po zakończeniu działania przez OB obsługujący przerwanie, CPU podejmuje wykonywanie programu użytkownika od miejsca, w którym wystąpiło przerwanie.

CPU określa kolejność obsługi przerwań na podstawie priorytetów przypisanych każdemu OB.

Każde zdarzenie ma swój priorytet obsługi. Kilka zdarzeń wywołujących przerwania może być połączonych w klasę priorytetu. Więcej informacji na ten temat jest podanych w rozdziale Koncepcja PLC, w części opisującej wykonywanie programu użytkownika.

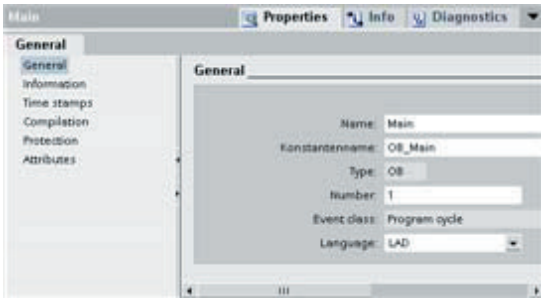
Tworzenie dodatkowego OB w istniejącej klasie OB

Użytkownik może utworzyć wiele OB do wykorzystania w swoim programie, nawet w klasach cyklicznej i rozruchowej. W celu utworzenia nowego OB należy skorzystać z okna dialogowego „Add new block”. Należy podać nazwę OB oraz nadać mu numer OB większy od 200.

Jeśli do programu użytkownika zostanie utworzonych wiele cyklicznych OB, to CPU wykonuje wszystkie cykliczne OB w kolejności zgodnej z ich numerami, począwszy od głównego OB cyklu (domyślnie; OB 1). Na przykład, po zakończeniu pierwszego cyklicznego OB (OB 1), CPU wykonuje drugi cykliczny OB (np. OB 2 lub OB 200).

Konfiguracja działania OB

Użytkownik może zmodyfikować parametry operacyjne dla OB. Na przykład, może skonfigurować parametr czasowy opóźnionego OB lub cyklicznego OB.



5.1.2.2 Funkcje (FC)

Funkcja jest to szybko uruchamiany blok kodu, który zazwyczaj wykonuje określone działania na zbiorze wartości wejściowych. FC przechowuje wyniki operacji w komórkach pamięci.

FC stosuje się do wykonywania następujących zadań:

- Standardowych i powtarzalnych działań, jak na przykład obliczeń arytmetycznych.
- Funkcji technologicznych, takich jak indywidualne sterowanie za pomocą działań logicznych.

FC może być wywoływana wielokrotnie w różnych miejscach programu. Ta możliwość wielokrotnego użycia FC upraszcza programowanie często występujących zadań.

Przeciwnie niż blok funkcji (FB), FC nie jest skojarzona z żadnym blokiem danych *instance* (DB). Dla danych tymczasowych występujących podczas przeprowadzania obliczeń, FC wykorzystuje lokalny stos danych. Dane tymczasowe nie są zapamiętywane. W celu zapamiętania danych należy przypisać wartości wyjściowej miejsce w pamięci, na przykład w pamięci M lub w globalnym DB.

5.1.2.3 Blok funkcji (FB)

Blok funkcji (FB) jest blokiem kodu, którego wywołanie może być programowane za pomocą parametrów bloku. FB ma zmienną pamięć zlokalizowaną w bloku danych (DB) lub instancji DB. Instancja DB zapewnia blok pamięci skojarzonej z „wywołaniem" FB i przechowuje dane po zakończeniu działania FB. Można skojarzyć różne bloki danych *instans* DB z różnymi wywołaniami FB. Bloki DB pozwalają na użycie tego samego FB do sterowania wielu urządzeń. CPU wykonuje program zawarty w FB i zapamiętuje parametry bloku oraz statyczne dane lokalne w danej instancji DB. Gdy wykonanie FB jest zakończone, wtedy CPU powraca do bloku kodu, z którego FB został wywołany. Instancja DB zachowuje wartości wpisane podczas tego wykonania FB.

Bloki kodu do wielokrotnego wykorzystania z przypisaną pamięcią

FB są zwykle używane do sterowania działaniem zadań lub urządzeń, które nie kończą swojej pracy w jednym cyklu programu. W celu przechowywania parame-

trów operacyjnych w taki sposób, by były szybko dostępne w kolejnych cyklach programu, każda FB w programie użytkownika ma jeden lub więcej instancji DB. Kiedy FB jest wywoływana, wtedy również jest otwierany DB przechowujący parametry bloku i statyczne dane lokalne dla danego wywołania lub instancji FB. Instancja DB pamięta te wartości po zakończeniu działania FB.

Projektując FB do wykonywania ogólnych zadań sterowania, użytkownik może wykorzystać ten FB z wieloma urządzeniami wybierając różne instancje DB do różnych wywołań FB.

FB przechowuje w instancji DB parametry wejściowe (IN), wyjściowe (OUT) oraz wejściowo/wyjściowe (IN_OUT).

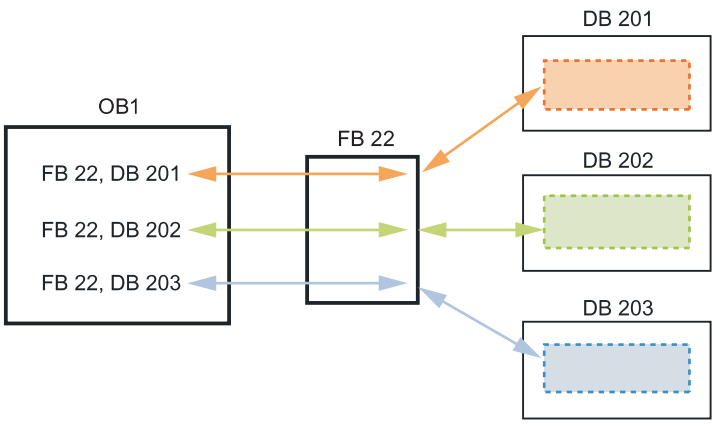
Przypisywanie wartości początkowych

Jeżeli parametry wejściowe, wyjściowe lub wejściowo/wyjściowe bloku funkcji (FB) nie mają przypisanych wartości, to są wykorzystywane wartości pamiętane w instancji bloku danych (DB). W niektórych przypadkach to użytkownik musi ustalić te parametry.

Użytkownik może nadać wartości początkowe parametrom interfejsu FB. Te wartości zostaną przeniesione do skojarzonej instancji DB. Jeśli parametrom nie zostaną nadane wartości, to będą wykorzystane wartości pamiętane w instancji DB.

Wykorzystanie pojedynczego FB z wieloma instancjami DB

Na poniższym rysunku przedstawiono jeden OB, który trzykrotnie wywołuje jeden FB, za każdym razem z innym blokiem danych. Ta struktura pozwala wykorzystać jeden ogólny FB do sterowania kilku podobnych urządzeń, takich jak silniki, poprzez przypisanie mu w każdym wywołaniu różnych instancji bloku danych dla różnych urządzeń. Każda instancja DB przechowuje dane (jak szybkość, czas rozpędzania i całkowity czas pracy) dla indywidualnego urządzenia. W podanym przykładzie FB 22 steruje trzema oddzielnymi urządzeniami, przy czym DB 201 pamięta dane dla pierwszego, DB 202 dla drugiego, a DB 203 dla trzeciego urządzenia.



5.1.2.4 Blok danych (DB)

Bloki danych (DB) są umieszczane w programie użytkownika po to, by przechowywały dane dla bloków kodu. Wszystkie bloki kodu w programie użytkownika maja dostęp do globalnego DB, ale poszczególne instancje DB przechowują dane dla określonych bloków funkcji (FB).

Program użytkownika może przechowywać dane w specjalizowanych obszarach pamięci CPU, przeznaczonych dla wejścia (I), wyjścia (Q) i pamięci bitowej (M). Ponadto użytkownik może wykorzystywać bloki danych (DB) dla uzyskania szybkiego dostępu do danych przechowywanych w samym programie. Użytkownik może nadać DB status „tylko do odczytu".

Dane pamiętane w DB nie są usuwane po zamknięciu bloku danych lub po zakończeniu wykonywania korzystającego z nich bloku kodu. Są dwa typy DB:

- Globalny DB przechowuje dane dla bloków kodu programu użytkownika. Dostęp do danych zawartych w globalnym DB ma dowolny OB, FB i FC.
- Instancje DB przechowują dane dla określonych FB. Struktura danych w instancji DB odzwierciedla parametry (wejściowe, wyjściowe i wejściowo/wyjściowe) oraz dane statyczne FB. (Pamięć Temp FB nie jest przechowywana w instancji DB.)

UWAGA

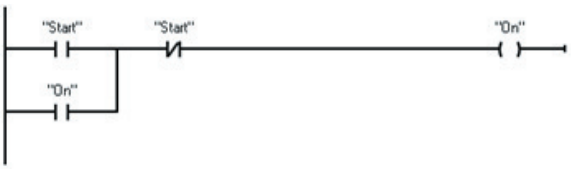
Mimo, że instancja DB przechowuje dane dla konkretnego FB, to dostęp do tych danych ma dowolny blok kodu.

5.1.3 Wybór języka programowania

Użytkownik ma możliwość wyboru języka programowania i korzystania albo z LAD (*ladder logic*) albo FBD (*Function Block Diagram*).

Język programowania LAD

LAD jest graficznym językiem programowania. Reprezentacja jest oparta na schematach obwodów.



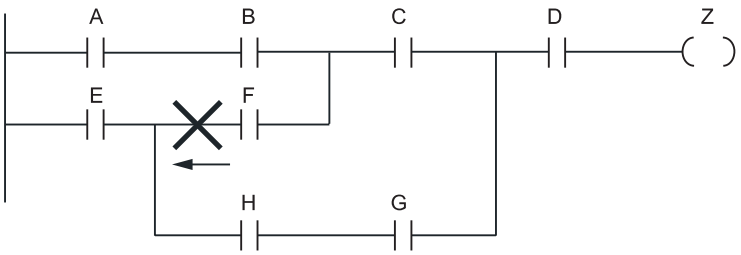
Elementy obwodu, takie jak styki normalnie zwarte lub normalnie rozwarte i cewki, są ze sobą łączone w celu utworzenia sieci.

W celu zaprojektowania logiki dla złożonych operacji, na schemacie można umieszczać gałęzie dla utworzenia logiki obwodów równoległych. Gałęzie równoległe są na początku rozwarte lub bezpośrednio łączone do zasilania. Od strony końcowej gałęzi występują połączenia zakańczające.

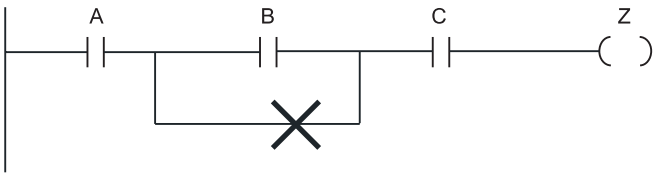
LAD umożliwia stosowanie instrukcji dla różnych funkcji, takich jak arytmetyczne, czasowe, zliczające oraz związane z ruchem.

Podczas tworzenia sieci LAD, należy kierować się następującymi zasadami:

- Każda sieć LAD musi kończyć się cewką lub instrukcją ramkową. Nie należy zakańczać sieci instrukcjami porównania lub wykrywania zboczy (dodatnich lub ujemnych).
- Nie wolno tworzyć gałęzi, w której może nastąpić przepływ mocy w odwrotnym kierunku.



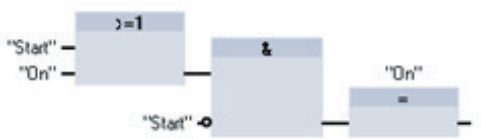
- Nie wolno tworzyć gałęzi, która mogłaby spowodować zwarcie.



Język programowania FBD (*Function Block Diagram*)

Podobnie jak LAD, również FBD jest graficznym językiem programowania. Reprezentacja logiki jest w nim oparta na graficznych symbolach logicznych stosowanych w algebrze Boole'a.

Działania arytmetyczne i inne złożone funkcje mogą być reprezentowane bezpośrednio razem z symbolami logicznymi. W celu stworzenia logiki złożonych operacji wystarczy połączyć symbole logiczne równoległymi gałęziami.



Znaczenie EN i ENO dla instrukcji ramkowych

Zarówno w LDA, jak i w FBD, dla niektórych instrukcji ramkowych stosuje się parametr „power flow" – „zasilanie" (EN i ENO). Niektóre instrukcje (arytmetyczne i dotyczące ruchu) wykorzystują parametry EN i ENO. Te parametry są związane z podawaniem zasilania i określają czy instrukcja jest wykonywana podczas cyklu programu.

- EN (*Enable In*) jest wejściem boolowskim dla ramek W LAD i FBD. Jeżeli instrukcja ramkowa ma być wykonana, to na jej wejściu musi wystąpić zasilanie (EN = 1).

- ENO (*Enable Out*) jest wyjściem boolowskim dla ramek w LAD i FBD. Jeżeli wejście EN bloku LAD jest bezpośrednio połączone do szyny zasilania z lewej strony, to wtedy instrukcja ramkowa zawsze będzie wykonana. Jeżeli na wejściu EN bloku jest zasilanie i funkcje bloku są wykonane bez błędów, to ENO przekazuje zasilanie (ENO = 1) do następnego elementu. Jeżeli zostanie wykryty błąd podczas wykonywania instrukcji z bloku, to przekazanie zasilania jest zatrzymywane (ENO = 0) na tej ramce z instrukcjami, w której został wygenerowany błąd.

Edytor programu	Wejścia/wyjścia	Argumenty	Typ danych
LAD	EN, ENO	Power flow	BOOL
FBD	EN	I, I:P, Q, M, DB, Temp, Power flow	BOOL
	ENO	Power flow	BOOL

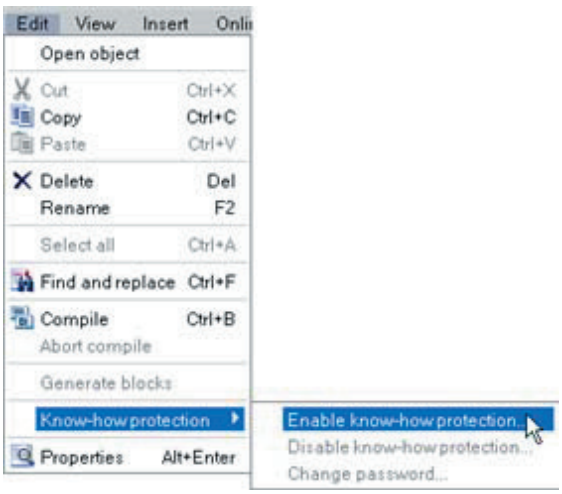
5.2 Zabezpieczenie przed kopiowaniem

Możliwość zabezpieczenia wiedzy przed skopiowaniem pozwala użytkownikowi ograniczyć nieautoryzowany dostęp do jednego lub wielu bloków kodu (OB, FB lub FC) lub bloków danych (DB) w swoim programie. W celu ograniczenia dostępu do bloku kodu użytkownik ustala hasło.

Jeśli blok został zabezpieczony przed niepowołanym dostępem, to zawartość bloku można odczytać tylko po podaniu hasła. W celu zabezpieczenia bloku przed kopiowaniem należy wybrać komendę „Know how protection" z menu „Edit". Następnie wprowadza się hasło umożliwiające dostęp do bloku.

Jeśli blok został zabezpieczony przed niepowołanym dostępem, to zawartość bloku można odczytać tylko po podaniu hasła.

W celu zabezpieczenia bloku przed kopiowaniem należy wybrać komendę „Know how protection" z menu „Edit". Następnie wprowadza się hasło umożliwiające dostęp do bloku.



Ochrona hasłem zabezpiecza przed nieautoryzowanym odczytem lub modyfikowaniem bloku kodu. Bez podania hasła można odczytać wyłącznie następujące informacje dotyczące bloku kodu:

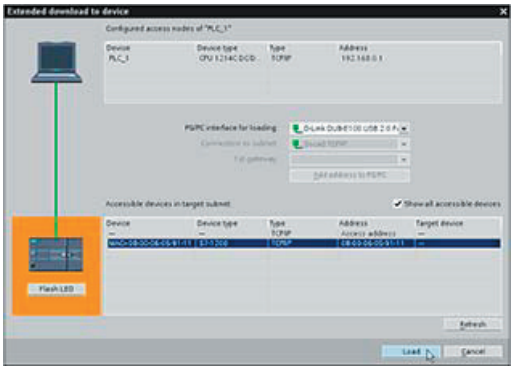
- Nazwę bloku, komentarz i właściwości bloku.
- Parametry przenoszone ((IN, OUT, IN_OUT, Return).
- Strukturę wywołującą program.
- Globalne *tagi* w odsyłaczach (bez informacji gdzie są używane); *tagi* lokalne są ukryte.

Wczytywanie elementów programu użytkownika

Użytkownik może wczytać elementy swojego projektu z urządzenia programującego [mk1]do CPU. Po wczytaniu projektu CPU przechowuje program użytkownika (OB, FC, FB i DB) w pamięci stałej.

Użytkownik może wczytać swój projekt z urządzenia programującego do CPU z następujących lokalizacji:

- Z „drzewa programu”: poprzez kliknięcie prawym klawiszem myszy elementu programu, a następnie wybór „Download” z menu kontekstowego.
- Z menu „online”: poprzez kliknięcie pozycji „Download to device”.
- Z menu narzędziowego: poprzez kliknięcie ikony „Download to device”.



5.3 Debugowanie i testowanie programu

Do monitorowania i modyfikowania wartości wykonywanego właśnie przez CPU programu użytkownika stosuje się tablice monitorujące (*watch tables*). Użytkownik może w swoim projekcie stworzyć i zapisać tablice monitorujące, dostarczające danych dla wielu środowisk testowych. Można dzięki temu odtwarzać testy podczas odbioru systemu lub w celach serwisowych albo podczas przeglądów.

Tablica monitorująca pozwala monitorować i interaktywnie współdziałać z CPU w czasie, kiedy wykonuje on program użytkownika. Można wyświetlać i zmieniać wartości nie tylko *tagów* bloków kodu i danych, ale również obszarów pamięci CPU, włączając w to wejścia i wyjścia (I i Q), peryferyjne wejścia i wyjścia (I:P i Q:P), pamięć bitową i bloki danych (DB).

Posługując się tablicą monitorującą można uaktywniać wyjścia peryferyjne (Q:P) CPU w trybie STOP. Na przykład, podczas testowania okablowania CPU, można przypisywać wyjściom określone wartości.

Tablica monitorująca pozwala również wymuszać lub ustawiać określone wartości *tagów*. Wartości wymuszone są przypisywane raz na cykl programu. Mogą być zmieniane podczas wykonywania programu, ale w przypadku wyjść (Q) wartości wymuszone są zapisywane na końcu cyklu programu. Więcej informacji na temat wymuszania znajduje się w rozdziale „Narzędzia *online* i diagnostyczne”, w części dotyczącej wymuszaniu wartości w CPU.

Instrukcje programowania 6

6.1 Podstawowe instrukcje

6.1.1 Logika bitowa

Styki LAD

Styki można łączyć z innymi stykami i tworzyć w ten sposób własną logikę kombinacyjną. Jeżeli wyspecyfikowany przez użytkownika bit wejściowy używa identyfikatora pamięci I (wejście) lub Q (wyjście), to wartość bitu jest czytana z rejestru obrazu procesu. Sygnały fizyczne styków są w systemie sterowania procesem połączone przewodami do wyprowadzeń I w PLC. System PLC skanuje okablowane sygnały wejściowe i w sposób ciągły uaktualnia odpowiednie stany rejestru wejściowego obrazu procesu.

Użytkownik może spowodować bezpośredni odczyt wejścia fizycznego komendą „:P" występującą zaraz po adresie I (na przykład: „%I3.4:P"). Odczyt bezpośredni polega na tym, że wartości bitów danych są czytane bezpośrednio z wejścia fizycznego, a nie z obrazu procesu. Odczyt bezpośredni nie uaktualnia obrazu procesu.

Normalnie rozwarte

Normalnie zwarte

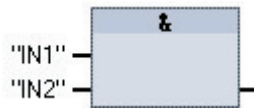
Parametr	Typ danych	Opis
IN	BOOL	Przypisany bit

- Styki normalnie rozwarte (*normally open*) są zwarte (ON) wtedy, kiedy wartość przypisanego bitu jest równa 1.
- Styki normalnie zwarte (*normally closed*) są zwarte (ON) wtedy, kiedy wartość przypisanego bitu jest równa 0.
- Styki połączone szeregowo tworzą obwód logiczny AND.
- Styki połączone równolegle tworzą obwód logiczny OR.

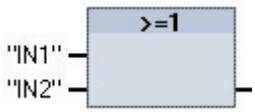
Bloki FBD: AND, OR i XOR

W języku programowania FBD, sieci LAD są transformowane w sieci logiczne funktorów logicznych AND (&), OR (>=1) i XOR (x), w których użytkownik może wyspecyfikować wartości bitów wejściowych i wyjściowych bloków. Może również wykonać połączenia z innymi blokami logicznymi, co pozwala utworzyć własną logikę kombinacyjną. W celu zwiększenia liczby wejść, po umieszczeniu bloków w sieci, z menu narzędziowego „Favourites" lub drzewa z instrukcjami można przeciągnąć narzędzie „Insert binary input" i upuścić je po stronie wejściowej bloków. Można również kliknąć prawym klawiszem myszy na złącze wejściowe bloku wybrać „Insert inputs".

Wejścia i wyjście bloku można połączyć z innym blokiem logicznym lub podać na niepodłączone wejście adres bitu lub symboliczną nazwę bitu. Kiedy wykonywana jest instrukcja przypisana blokowi, wtedy na jego wejścia są podawane bieżące stany logiczne bitów i jeśli wynik działania jest prawdziwy, to na wyjściu pojawia się stan TRUE (prawda).



Funkcja logiczna AND



Funkcja logiczna OR

Funkcja logiczna XOR

Parametr	Typ danych	Opis
IN	BOOL	Przypisany bit

- Aby stan wyjścia bloku AND był TRUE, wszystkie wejścia muszą być w stanie TRUE.
- Aby stan wyjścia bloku OR był TRUE, dowolne wejście musi być w stanie TRUE.
- Aby stan wyjścia bloku XOR był TRUE, nieparzysta liczba wejść musi być w stanie TRUE.

Inwerter logiczny NOT

W języku programowania FBD z menu narzędziowego „Favourites” lub drzewa z instrukcjami można przeciągnąć narzędzie „Negate binary input” i upuścić je po stronie wejściowej bloku, aby umieścić logiczny inwerter.

LAD: inwerter stykowy NOT

FBD: ramka AND z jednym wejściem zanegowanym

FBD: ramka AND z zanegowanymi wejściem i wyjściem

Inwerter stykowy NOT języka LAD neguje wejściowy stan logiczny zasilania.

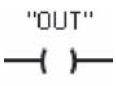
- Jeżeli na wejściu styku NOT nie ma zasilania, to na wyjściu zasilanie występuje.
- Jeżeli na wejściu styku NOT jest zasilanie, to na wyjściu zasilanie nie występuje.

Cewka wyjściowa w LAD

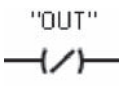
Instrukcja wyjściowa sterowania cewką ustala wartość bitu wyjściowego. Jeżeli wyspecyfikowany przez użytkownika bit wyjściowy ma identyfikator pamięci Q, to CPU włącza lub wyłącza ten bit w rejestrze obrazu procesu tak, by był zgodny

ze stanem zasilania. Wyjściowe sygnały sterujące urządzeniami wykonawczymi są podłączone do zacisków Q sterownika S7-1200. W trybie RUN, system CPU w sposób ciągły skanuje sygnały wejściowe, przetwarza te sygnały zgodnie z logiką programu i w rezultacie ustala nowe wartości stanów wyjściowych w rejestrze wyjściowym obrazu procesu. Po zakończeniu każdego cyklu programu, CPU przesyła te nowe wartości stanów wyjściowych zapamiętane w rejestrze obrazu procesu do okablowanych zacisków wyjściowych.

Użytkownik może spowodować bezpośredni zapis stanu do wyjścia fizycznego komendą „:P" występującą zaraz po adresie Q (na przykład: „%Q3.4:P"). Zapis bezpośredni polega na tym, że wartości bitów danych są zapisywane do obszaru wyjściowego obrazu procesu i jednocześnie bezpośrednio do wyjścia fizycznego.



Cewka wyjściowa



Cewka wyjściowa z negacją

Parametr	Typ danych	Opis
OUT	BOOL	Przypisany bit

- Jeżeli cewka wyjściowa jest zasilana, to wartość bitu wyjściowego jest ustalana na 1.
- Jeżeli cewka wyjściowa nie jest zasilana, to wartość bitu wyjściowego jest ustalana na 0.
- Jeżeli zanegowana cewka wyjściowa jest zasilana, to wartość bitu wyjściowego jest ustalana na 0.
- Jeżeli zanegowana cewka wyjściowa nie jest zasilana, to wartość bitu wyjściowego jest ustalana na 1.

Blok wyjściowy w FBD

W języku FBD, w miejsce cewek LAD stosuje się wyjściowe bloki sterujące (= i /=), w których użytkownik specyfikuje adres bitu wyjściowego. Wejścia i wyjścia bloku mogą być łączone z innymi blokami logicznymi lub można podawać adresy bitów.

Wyjściowy blok sterujący

Negujący blok wyjściowy

Blok sterujący
z zanegowanym wyjściem

Parametr	Typ danych	Opis
OUT	BOOL	Przypisany bit

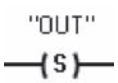
- Jeżeli w wyjściowym bloku sterującym stan wejścia wynosi 1, to wartość bitu OUT jest ustalana na 1.
- Jeżeli w wyjściowym bloku sterującym stan wejścia wynosi 0, to wartość bitu OUT jest ustalana na 0.
- Jeżeli w negującym bloku wyjściowym stan wejścia wynosi 1, to wartość bitu OUT jest ustalana na 0.
- Jeżeli w negującym bloku wyjściowym stan wejścia wynosi 0, to wartość bitu OUT jest ustalana na 1.

6.1.1.1 Instrukcje ustawiania i kasowania

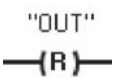
S i R: Ustawianie (set) i kasowanie (reset) jednego bitu

- Kiedy S (Set) jest aktywowany, wtedy wartość danej pod adresem OUT jest ustawiana na 1. Kiedy S nie jest aktywowany, wtedy OUT nie ulega zmianie.
- Kiedy R (Reset) jest aktywowany, wtedy wartość danej pod adresem OUT jest ustawiana na 0. Kiedy R nie jest aktywowany, wtedy OUT nie ulega zmianie.
- Te instrukcje można umieszczać w dowolnym miejscu sieci.

LAD: ustawianie S



LAD: kasowanie R



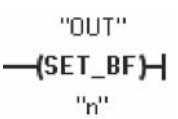
FBD: ustawianie S

FBD: kasowanie R

Parametr	Typ danych	Opis
IN (lub połączenie z bramką/ stykiem logicznym)	BOOL	Miejsce pamiętania bitu monitorowane
OUT	BOOL	Miejsce pamiętania bitu ustawiane lub kasowane

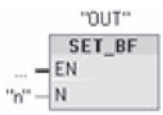
SET_BF i RESET_BF: Ustawianie i kasowanie pola bitowego

LAD: SET_BF

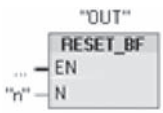


LAD: RESET_BF

FBD: SET_BF



FBD: RESET_BF



Parametr	Typ danych	Opis
n	Constant	Liczba bitów do zapisania
OUT	BOOL	Adres początkowy pola bitowego

- Kiedy SET_BF jest aktywowany, wtedy wartość 1 jest przypisywana „n” bitom począwszy od adresu OUT. Kiedy SET_BF nie jest aktywowany, wtedy OUT nie ulega zmianie.

- RESET_BF przypisuje wartość 0 „n” bitom począwszy od adresu OUT. Kiedy RESET_BF nie jest aktywowany, wtedy OUT nie ulega zmianie.
- Te instrukcje muszą zajmować skrajne prawe pozycje w gałęzi.

RS i SR: Przerzutniki z dominującym wejściem ustawiającym i z dominującym wejściem kasującym

LAD/FBD: RS

LAD/FBD: SR

Parametr	Typ danych	Opis
S, S1	BOOL	Wejście ustawiające; 1 oznacza wejście dominujące
R, R1	BOOL	Wejście kasujące; 1 oznacza wejście dominujące
OUT	BOOL	Bit przypisany do wyjścia „OUT”
Q	BOOL	Powtarza stan bitu „OUT”

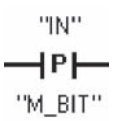
- RS jest przerzutnikiem z dominującym wejściem ustawiającym, w którym nadrzędną rolę spełnia sygnał ustawiający. Jeżeli oba sygnały ustawiający (S1) i kasujący (R) przyjmują wartość TRUE, to pod adres wyjściowy OUT zastanie wpisana wartość 1.
- SR jest przerzutnikiem z dominującym wejściem kasującym, w którym nadrzędną rolę spełnia sygnał kasujący. Jeżeli oba sygnały ustawiający (S1) i kasujący (R) przyjmują wartość TRUE, to pod adres wyjściowy OUT zastanie wpisana wartość 0.
- Parametr OUT określa adres bitu, który jest ustawiany lub kasowany. Opcjonalny sygnał wyjściowy Q odtwarza stan bitu spod adresu OUT.

Instrukcja	**S1**	**R**	**bit „OUT”**
RS	0	0	stan poprzedni
	0	1	0
	1	0	1
	1	1	1
	S	**R1**	
SR	0	0	stan poprzedni
	0	1	0
	1	0	1
	1	1	0

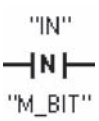
6.1.1.2 Instrukcje dotyczące zboczy dodatnich i ujemnych

Detekcja przejścia dodatniego i ujemnego

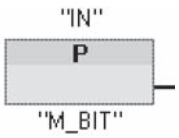
Styk P: LAD



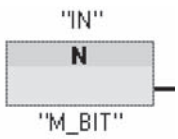
Styk N: LAD



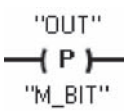
Ramka P: FBD



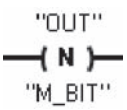
Ramka N: FBD



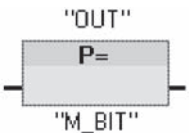
Cewka P: LAD



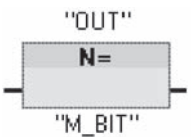
Cewka N: LAD



Ramka P=: FBD



Ramka N=: FBD



P_TRIG: LAD/FBD

N_TRIG: LAD/FBD

Parametr	Typ danych	Opis
M_BIT	BOOL	Bit w pamięci, który pamięta poprzedni stan wejścia
IN	BOOL	Bit wejściowy, którego zmiana (zbocze) ma być wykryta
OUT	BOOL	Bit wyjściowy sygnalizujący wykrycie zbocza
CLK	BOOL	Zasilanie lub bit wejściowy, którego zmiana (zbocze) ma być wykryta
Q	BOOL	Wyjście sygnalizujące wykrycie zbocza

Styk P LAD — Stan tego styku ma wartość TRUE wtedy, kiedy jest wykryta dodatnia zmiana (wyłączony › włączony) przypisanego bitu wejściowego „IN”. Stan logiczny styku wraz z wejściowym stanem zasilania określają łącznie wyjściowy stan zasilania. Styk P można umieścić w dowolnym miejscu sieci z wyjątkiem końca gałęzi.

Styk N LAD — Stan tego styku ma wartość TRUE wtedy, kiedy jest wykryta ujemna zmiana (włączony › wyłączony) przypisanego bitu wejściowego „IN”. Stan logiczny styku wraz z wejściowym stanem zasilania określają łącznie wyjściowy stan zasilania. Styk N można umieścić w dowolnym miejscu sieci z wyjątkiem końca gałęzi.

Ramka P FBD	Wyjściowy stan bloku ma wartość TRUE wtedy, kiedy jest wykryta dodatnia zmiana (wyłączony › włączony) przypisanego bitu wejściowego. Ramkę P można umieścić wyłącznie na początku gałęzi.
Ramka N FBD	Wyjściowy stan bloku ma wartość TRUE wtedy, kiedy jest wykryta ujemna zmiana (włączony › wyłączony) przypisanego bitu wejściowego. Ramkę N można umieścić wyłącznie na początku gałęzi.
Cewka P LAD	Przypisany bit „OUT” przyjmuje wartość TRUE wtedy, kiedy jest wykryta dodatnia zmiana (wyłączone › włączone) zasilania cewki. W przypadku cewki stan zasilania na wyjściu zawsze jest taki sam jak stan zasilania na wejściu. Cewkę P można umieścić w dowolnym miejscu sieci.
Cewka N LAD	Przypisany bit „OUT” przyjmuje wartość TRUE wtedy, kiedy jest wykryta ujemna zmiana (włączone › wyłączone) zasilania cewki. W przypadku cewki stan zasilania na wyjściu zawsze jest taki sam jak stan zasilania na wejściu. Cewkę N można umieścić w dowolnym miejscu sieci.
Ramka P= FBD	Przypisany bit „OUT” przyjmuje wartość TRUE wtedy, kiedy jest wykryta dodatnia zmiana (wyłączony › włączony) stanu wejściowego bloku lub przypisanego bitu wejściowego jeśli ramka jest umieszczona na początku gałęzi. Logiczny stan wejściowy zawsze przechodzi bez zmian przez ramkę i pojawia się na wyjściu jako stan wyjścia. Ramkę P= można umieścić w dowolnym miejscu gałęzi.
Ramka N= FBD	Przypisany bit „OUT” przyjmuje wartość TRUE wtedy, kiedy jest wykryta ujemna zmiana (włączony › wyłączony) stanu wejściowego bloku lub przypisanego bitu wejściowego jeśli ramka jest umieszczona na początku gałęzi. Logiczny stan wejściowy zawsze przechodzi bez zmian przez ramkę i pojawia się na wyjściu jako stan wyjścia. Ramkę N= można umieścić w dowolnym miejscu gałęzi.
P_TRIG LAD/FBD	Stan zasilania lub stan logiczny na wyjściu Q przyjmuje wartość TRUE wtedy, kiedy jest wykryta dodatnia zmiana (wyłączony › włączony) stanu wejściowego CLK (FBD) lub zasilania na wejściu CLK (LAD). W przypadku LAD, instrukcja P_TRIG nie może być umieszczona na początku lub końcu sieci. W przypadku FBD, instrukcja P_TRIG może się znajdować w dowolnym miejscu z wyjątkiem końca gałęzi.
N_TRIG LAD/FBD	Stan zasilania lub stan logiczny na wyjściu Q przyjmuje wartość TRUE wtedy, kiedy jest wykryta ujemna zmiana (włączony › wyłączony) stanu wejściowego CLK (FBD) lub zasilania na wejściu CLK (LAD). W przypadku LAD, instrukcja N_TRIG nie może być umieszczona na początku lub końcu sieci. W przypadku FBD, instrukcja N_TRIG może się znajdować w dowolnym miejscu z wyjątkiem końca gałęzi.

Wszystkie instrukcje dotyczące zboczy wykorzystują bit pamięci (M_BIT) do pamiętania poprzedniego stanu monitorowanego sygnału wejściowego. Zbocze jest wykrywane poprzez porównanie stanu wejścia ze stanem bitu w pamięci. Jeżeli te stany wskazują, że nastąpiła interesująca nas zmiana na wejściu, to jest sygnalizowane wykrycie zbocza poprzez wpisanie na wyjście stanu TRUE. W przeciwnym wypadku stanem wyjściowym jest FALSE.

UWAGA

Instrukcje dotyczące zboczy sprawdzają stan wejścia oraz wartość bitu w pamięci za każdym razem gdy są wykonywane, włączając w to pierwsze wykonanie. W związku z tym użytkownik musi wziąć pod uwagę w trakcie pisania programu stan początkowy sygnału na wejściu i bitu w pamięci i zdecydować czy wykrywać, czy unikać wykrywania zbocza podczas pierwszego cyklu programu.

Ponieważ bit w pamięci musi być zachowany bez zmian od jednego wykonania instrukcji do następnego, więc dla każdej instrukcji wykrywania zboczy należy przewidzieć unikalny bit, a także nie korzystać z tego bitu w żadnym innym miejscu programu. Należy także unikać pamięci chwilowej oraz pamięci, która może zostać zmieniona przez inne funkcje systemu, takie jak uaktualnianie I/O. Do przechowywania M_BIT należy korzystać tylko z pamięci M, globalnego DB lub pamięci statycznej.

6.1.2 Układy czasowe – timery

Instrukcje związane z układami czasowymi są wykorzystywane do generowania programowanych opóźnień:

- TP: Układ czasowy *Pulse timer* generuje impuls o ustalonym czasie trwania.
- TON: Układ czasowy *ON-delay timer* ustawia stan swojego wyjścia Q na ON (włączony) po upływie zadanego czasu opóźnienia.
- TOF: Układ czasowy *OFF-delay timer* kasuje stan swojego wyjścia Q na OFF (wyłączony) po upływie zadanego czasu opóźnienia.
- TONR: Układ czasowy *ON-delay Retentive timer* ustawia stan swojego wyjścia na ON (włączony) po upływie zadanego czasu opóźnienia. Upływający czas jest naliczany przez wiele okresów, aż do chwili gdy zliczany upływ czasu zostanie wyzerowany za pomocą wejścia R.
- RT: Kasowanie układu czasowego poprzez wyzerowanie danych timera w określonej instancji bloku danych układu czasowego.

Każdy układ czasowy wykorzystuje do pamiętania danych timera strukturę przechowywaną w bloku danych timera. Blok danych jest przypisywany timerowi przez użytkownika wtedy, kiedy instrukcja timera jest umieszczona w edytorze.

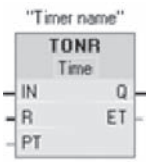
Kiedy instrukcja dotycząca timera zostaje umieszczona w bloku funkcji, to można wybrać opcję zwielokrotnienia instancji bloku danych, nazwy struktur timerów mogą być różne z oddzielnymi strukturami danych, ale dane timera są zawarte w jednym bloku danych i nie wymagane są oddzielne bloki danych dla każdego timera. W ten sposób redukuje się czas przetwarzania i pamięć niezbędną do obsługi timerów. Nie występuje żadna interakcja pomiędzy strukturami danych timerów we współdzielonych wielokrotnych instancjach bloków danych.

UWAGA

Mimo że jest to nietypowe, użytkownik może przypisać tę samą nazwę jednokrotnego instancji struktury timera do wielokrotnych instrukcji dotyczących układu czasowego, umożliwiając w ten sposób współdzielenie struktury danych pomiędzy wiele instrukcji timera. Podczas pisania programu należy jednak wziąć pod uwagę możliwe interakcje powstałe w wyniku takiego współdzielenia struktury.

LAD

Timery TP, TON, TOF mają takie same parametry wejściowe i wyjściowe.



Timer TONR ma dodatkowo wejściowy parametr kasujący R.

Użytkownik może nadać własną nazwę „Timer Name" blokowi danych timera, która opisuje jaką funkcję pełni timer w procesie.

„Timer name"
----[RT]----

Instrukcja RT kasuje dane wybranego timera.

Parametr	Typ danych	Opis
IN	BOOL	Wejście uaktywniające timer
R	BOOL	Zerowanie licznika upływającego czasu TONR
PT	TIME	Wejście nastawionego czasu
Q	BOOL	Wyjście timera
ET	TIME	Wyjście licznika upływającego czasu
Blok danych timera	DB	Wyznaczenie timera kasowanego instrukcją RT

Parametr IN uruchamia i zatrzymuje timery:

- Zmiana stanu parametru IN z 0 na 1 uruchamia timery TP, TON i TONR.
- Zmiana stanu parametru IN z 1 na 0 uruchamia timer TOF.

Efekty zmiany stanu parametrów PT i IN:

- TP:
  - Zmiana PT nie ma żadnego efektu podczas pracy timera.
  - Zmiana IN nie ma żadnego efektu podczas pracy timera.
- TON:
  - Zmiana PT nie ma żadnego efektu podczas pracy timera.
  - Zmiana IN na FALSE podczas pracy timera, kasuje i zatrzymuje timer.

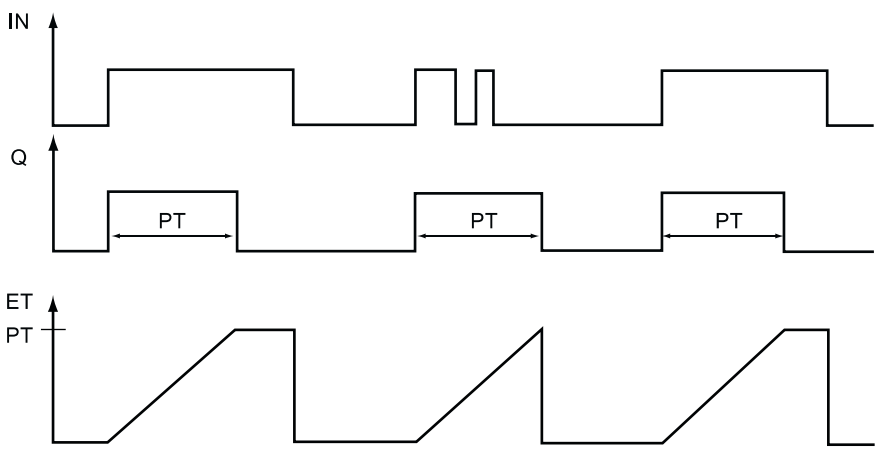
- TOF:
  - Zmiana PT nie ma żadnego efektu podczas pracy timera.
  - Zmiana IN na TRUE podczas pracy timera, kasuje i zatrzymuje timer.
- TONR:
  - Zmiana PT nie ma żadnego efektu podczas pracy timera, ale daje efekt w czasie gdy timer wznawia pracę.
  - Zmiana IN na FALSE podczas pracy timera, zatrzymuje timer, ale go nie kasuje. Zmiana IN z powrotem na TRUE powoduje, że timer rozpoczyna pracę od zliczonej wartości czasu.

Wartości TIME

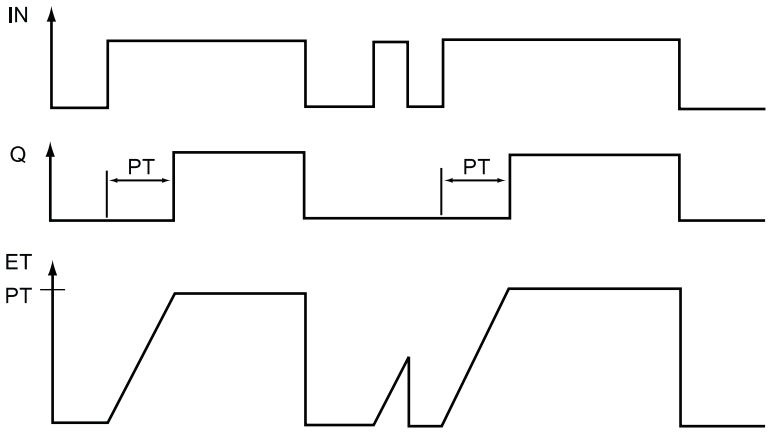
Wartości PT (*preset time* – czas nastawiony) i ET (*elapsed time* – upływający czas) są przechowywane w pamięci jako liczby całkowite o podwójnej długości ze znakiem i wyrażają czas w milisekundach. Dana TIME wykorzystuje identyfikator T# i może być wprowadzana jako prosta jednostka czasu „T#200ms” lub w postaci złożonej „T#2s_200ms”.

Typ danych	Rozmiar	Zakres poprawnych wartości
TIME	32 bity Przechowywany jako	T#-24d_20h_31m_23s_648ms do T#24d_20h_31m_23s_647ms -2,147,483,648 ms do +2,147,483,647 ms

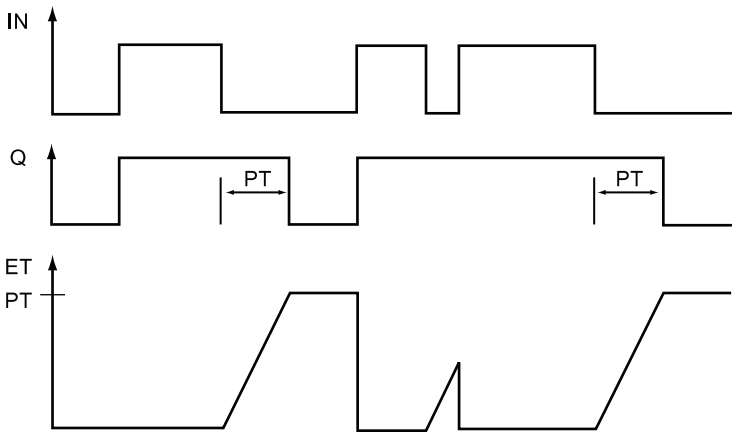
TP: przebieg czasowy Pulse



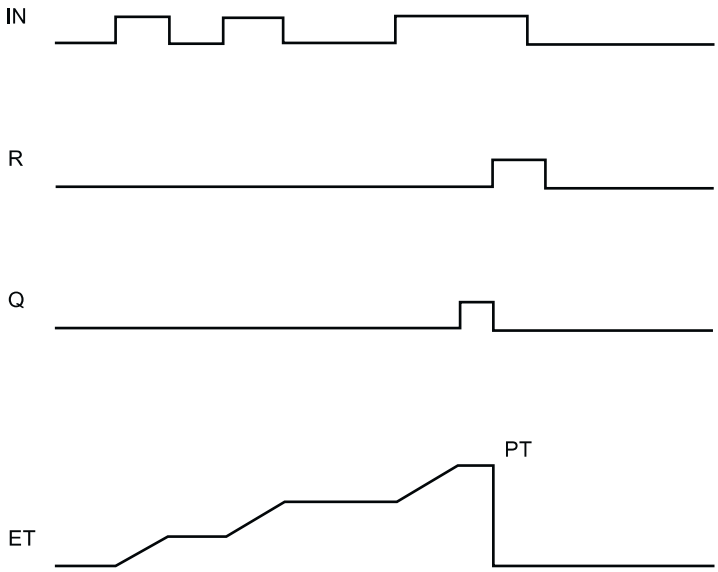
TON: Przebieg czasowy ON-delay



TOF: Przebieg czasowy OFF-delay



TONR: Przebieg czasowy ON-delay Retentive



6.1.3 Liczniki

Instrukcje dotyczące liczników są stosowane do zliczania wewnętrznych zdarzeń w programie i zewnętrznych zdarzeń procesu.:

- CTU jest to licznik zliczający w górę.
- CTD jest to licznik zliczający w dół.
- CTUD jest to licznik zliczający w górę i w dół.

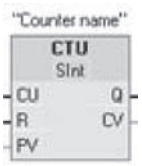
Każdy licznik wykorzystuje do pamiętania danych licznika strukturę przechowywaną w bloku danych. Blok danych jest przypisywany funkcji timera, kiedy instrukcja dotycząca licznika jest umieszczona w edytorze. Te instrukcje korzystają z liczników programowych i ich maksymalna szybkość zliczania jest ograniczona częstością wykonywania OB, w których są umieszczone. W celu wykonywania szybkich zewnętrznych operacji zliczania por. instrukcję CTRL_HSC.

Kiedy instrukcja dotycząca licznika zostaje umieszczona w bloku funkcji, to można wybrać opcję zwielokrotnienia instancji bloku danych, nazwy struktur liczników mogą być różne z oddzielnymi strukturami danych, ale dane licznika są zawarte w jednym bloku danych i nie wymagane są oddzielne bloki danych dla każdego licznika. W ten sposób redukuje się czas przetwarzania i pamięć niezbędną do obsługi liczników. Nie występuje żadna interakcja pomiędzy strukturami danych liczników we współdzielonych wielokrotnych instancjach bloków danych.

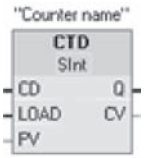
UWAGA

Mimo że jest to nietypowe, użytkownik może przypisać tę samą nazwę jednokrotnej instancji struktury licznika do wielokrotnych instrukcji dotyczących liczników o tych samych rozmiarach, umożliwiając w ten sposób współdzielenie struktury danych pomiędzy wiele instrukcji licznika. Podczas pisania programu należy jednak wziąć pod uwagę możliwe interakcje powstałe w wyniku takiego współdzielenia struktury.

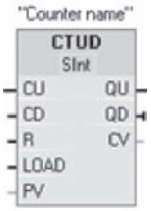
LAD/FBD



Z rozwijanej listy pod nazwą bloku należy wybrać typ zliczanych danych.



Użytkownik może nadać własną nazwę „Counter Name" blokowi danych licznika, która opisuje jaką funkcję pełni licznik w procesie.



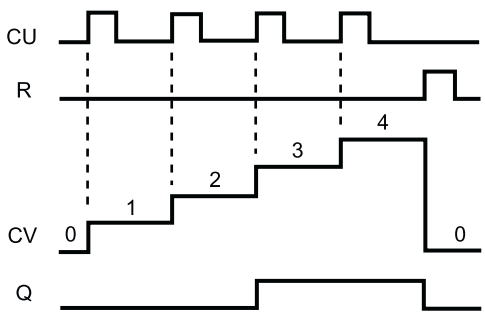
Parametr	Typ danych	Opis
CU, CD	BOOL	Zliczanie w górę lub w dół o 1
R (CTU, CTUD)	BOOL	Kasowanie liczby zliczeń do zera
LOAD (CTD, CTUD)	BOOL	Sterowanie wpisywaniem ustalonej wartości
PV	SINT, INT, DINT, USINT, UINT, UDINT	Ustalona wartość zliczeń
Q, QU	BOOL	True jeśli CV >= PV
QD	BOOL	True jeśli CV <= 0
CV	SINT, INT, DINT, USINT, UINT, UDINT	Bieżąca wartość zliczeń

Zakres zliczania zależy od wybranego typu danych. Jeżeli zliczenia są liczbami całkowitymi bez znaku, to w dół można zliczać do zera, a w górę aż do granicy zakresu. Jeżeli zliczenia są liczbami całkowitymi ze znakiem, to w dół można zliczać aż do ujemnej granicy liczby całkowitej, a w górę do dodatniej granicy liczby całkowitej.

CTU: Jeżeli wartość parametru CU zmienia się z 0 na 1, to CTU zlicza w górę o 1. Jeżeli wartość parametru CV (*current count value* – bieżąca wartość zliczeń) jest większa lub równa wartości parametru PV (*preset count value* – ustalona wartość zliczeń), to parametr wyjściowy licznika Q = 1.

Jeżeli wartość parametru kasującego R zmienia się z 0 na 1, to bieżąca wartość zliczeń zostaje skasowana do 0.

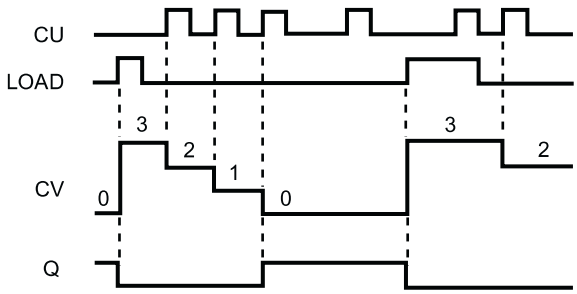
Na poniższym rysunku przedstawiono przebieg czasowy w przypadku licznika CTU zliczającego liczby całkowite bez znaku (dla PV = 3).



CTD: Jeżeli wartość parametru CD zmienia się z 0 na 1, to CTU zlicza w dół o 1. Jeżeli wartość parametru CV (*current count value* – bieżąca wartość zliczeń) jest większa lub równa 0, to parametr wyjściowy licznika Q = 1.

Jeżeli wartość parametru LOAD zmienia się z 0 na 1, to wartość parametru PV (*preset count value* – ustalona wartość zliczeń) jest wpisywana do licznika jako nowa wartość CV (*current count value* – bieżąca wartość zliczeń).

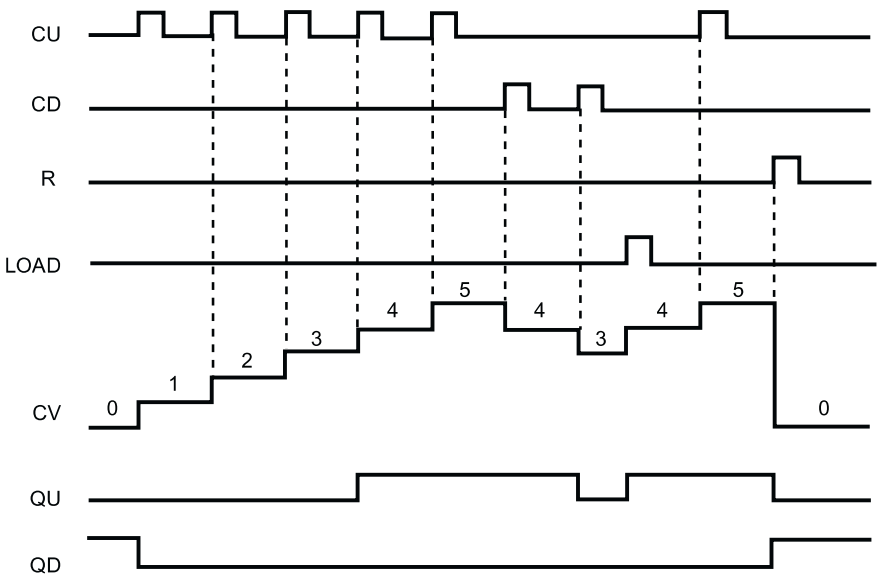
Na poniższym rysunku przedstawiono przebieg czasowy w przypadku licznika CTD zliczającego liczby całkowite bez znaku (dla PV = 3).



CTUD: CTUD zlicza o 1 w górę lub w dół przy każdej zmianie z 0 na 1 na wejściach CU (*count up* – zliczanie w górę) lub CD (*count down* – zliczanie w dół). Jeżeli wartość parametru CV (*current count value* – bieżąca wartość zliczeń) jest równa lub większa od wartości parametru PV (*preset value* – ustalona wartość), to parametr wyjściowy licznika QU = 1. Jeżeli wartość parametru CV jest mniejsza lub równa 0, to parametr wyjściowy licznika QD = 1.

Jeżeli wartość parametru LOAD zmienia się z 0 na 1, to wartość parametru PV (*preset value* – ustalona wartość) jest wpisywana do licznika jako nowa wartość CV (*current count value* – bieżąca wartość zliczeń). Jeżeli wartość parametru kasującego R zmienia się z 0 na 1, to bieżąca wartość zliczeń zostaje skasowana do 0.

Na poniższym rysunku przedstawiono przebieg czasowy w przypadku licznika CTUD zliczającego liczby całkowite bez znaku (dla PV = 4).

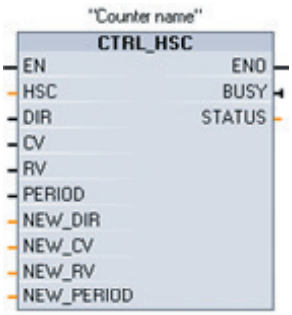


6.1.3.1 Instrukcja CTRL_HSC

Instrukcja CTRL_HSC kontroluje szybkie liczniki używane do zliczania zdarzeń występujących częściej niż wynosi częstość wywoływania cyklu programu wykonywanego przez CPU. Szybkość zliczania za pomocą instrukcji CTU, CTD i CTUD jest ograniczona przez częstość cyklu programu wykonywanego przez CPU. Szybkie liczniki pracują asynchronicznie względem CPU i mogą zliczać zdarzenia występujące z częstotliwością do 100 kHz (HSC 1, 2 lub 3 oraz konfigurowane wejście zliczające CPU). Typowe zastosowanie szybkich liczników obejmuje zliczanie impulsów z czujników mierzących prędkość obrotową.

Kiedy instrukcja CTRL_HSC wykorzystuje do pamiętania danych strukturę przechowywaną w bloku danych. Blok danych jest przypisywany przez użytkownika wtedy, kiedy instrukcja CTRL_HSC jest umieszczona w edytorze.

LAD/FBD



Użytkownik może nadać własną nazwę „Counter Name" blokowi danych licznika, która opisuje jaką funkcję pełni licznik w procesie.

Parametr	Typ parametru	Typ danych	Opis
HSC	IN	HW_HSC	Identyfikator HSC
DIR	IN	BOOL	1 = żądanie nowego kierunku
CV	IN	BOOL	1 = żądanie ustalenia nowej wartości zliczeń
RV	IN	BOOL	1 = żądanie ustalenia nowej wartości referencyjnej
PERIOD	IN	BOOL	1 = żądanie ustalenia nowej wartości okresu (tylko w trybie pomiaru częstotliwości)
NEW_DIR	IN	INT	Nowy kierunek: 1 = w przód -1 = wstecz
NEW_CV	IN	DINT	Nowa wartość zliczeń
NEW_RV	IN	DINT	Nowa wartość referencyjna
NEW_PERIOD	IN	INT	Nowa wartość okresu w sekundach: 0,01, 0,1 lub 1 (tylko w trybie pomiaru częstotliwości)
BUSY	OUT	BOOL	Funkcja busy (zajęty)
STATUS	OUT	WORD	Kod warunkowy wykonania

Działanie

Przed użyciem szybkich liczników w programie użytkownika, należy je skonfigurować na etapie definiowania projektu podczas konfiguracji urządzenia PLC. Podczas konfiguracji urządzenia HSC dokonuje się wyboru trybów zliczania, połączeń I/O, przypisuje przerwania, oraz definiuje czy urządzenie ma pracować jako szybki licznik, czy jako miernik częstotliwości impulsów. Szybki licznik może pracować pod kontrolą programu lub niezależnie od programu.

Wiele parametrów konfiguracyjnych szybkiego licznika jest ustawianych tylko podczas konfiguracji urządzenia sterującego projektem. Niektóre parametry szybkiego licznika są inicjalizowane podczas konfiguracji urządzenia sterującego projektem, ale mogą być później modyfikowane pod kontrolą programu.

Za pomocą parametrów instrukcji CTRL_HSC można kontrolować proces zliczania z programu:

- Kierunek zliczania jako wartość parametru NEW_DIR.
- Bieżące zliczenia jako wartość parametru NEW_CV.
- Wartość referencyjna jako wartość parametru NEW_RV.
- Wartość okresu jako wartość parametru NEW_PERIOD (tylko w trybie pomiaru częstotliwości).

Odpowiednie wartości parametrów NEW_xxx są wpisywane do licznika wtedy, kiedy podczas wykonywania instrukcji CTRL_HSC następujące znaczniki boolowskie są ustawione na 1. Wielokrotne żądania (w tym samym czasie jest ustawiony więcej niż jeden znacznik) są realizowane w trakcie pojedynczego wykonania instrukcji CTRL_HSC.

- DIR = 1 jest żądaniem wpisania do licznika wartości NEW_DIR, 0 = brak zmian.
- CV = 1 jest żądaniem wpisania do licznika wartości NEW_CV, 0 = brak zmian.
- RV = 1 jest żądaniem wpisania do licznika wartości NEW_RV, 0 = brak zmian.
- PERIOD = 1 jest żądaniem wpisania do licznika wartości NEW_PERIOD, 0 = = brak zmian.

Instrukcja CTRL_HSC jest zwykle umieszczana w OB przerwania sprzętowego, który jest wykonywany wtedy, kiedy pojawia się sprzętowe przerwanie pochodzące od licznika. Na przykład, jeśli warunek CV = RV wyzwala przerwanie pochodzące od licznika, to blok kodu OB przerwania sprzętowego wykonuje instrukcję CTRL_HSC, która może zmienić wartość referencyjną poprzez wczytanie nowej wartości NEW_RV.

Bieżąca wartość zliczeń nie jest dostępna jako parametr instrukcji CTRL_HSC. Adres obrazu procesu pod którym jest przechowywana bieżąca wartość zliczeń jest definiowany podczas konfiguracji sprzętowej szybkiego licznika. Użytkownik może w programie zrealizować bezpośredni odczyt wartości zliczeń i zwrócona do programu liczba będzie równa rzeczywistej liczbie zliczeń w chwili dokonywania odczytu – trzeba jednak pamiętać, że licznik kontynuuje zliczanie szybkich zdarzeń. Może się wiec zdarzyć, ze faktyczna liczba zliczeń zmieni się zanim program zakończy operację korzystając ze starej wartości.

Szczegóły parametrów CTRL_HSC:

- Jeżeli nie nastąpi żądanie uaktualnienia wartości parametru, to odpowiednie wartości wejściowe są ignorowane.
- Parametr DIR jest ważny tylko wtedy, kiedy kierunek zliczania jest ustalony programowo, a nie przez wejście sprzętowe. Sposób wykorzystania tego parametru określa użytkownik podczas konfiguracji urządzenia HSC.
- Dla HSC S7-1200 w CPU lub na płycie sygnałowej, parametr BUSY ma zawsze wartość 0.

Kody warunkowe

W przypadku wystąpienia błędu, ENO jest ustawiany na 0, a wyjście STATUS zawiera kod warunkowy.

Wartość STATUS(W#16#...)	Opis
0	Brak błędu
80A1	Identyfikator HSC nie adresuje HSC
80B1	Niedozwolona wartość w NEW_DIR
80B2	Niedozwolona wartość w NEW_CV
80B3	Niedozwolona wartość w NEW_RV
80B4	Niedozwolona wartość w NEW_PERIOD

6.1.4 Porównanie

Instrukcje porównania służą do porównania dwóch wartości danych tego samego typu. Kiedy styk porównania LAD ma wartość TRUE, wtedy ten styk jest aktywowany. Kiedy wynik porównania FBD ma wartość TRUE, wtedy na wyjściu bloku jest stan TRUE.

Typ relacji	Wynik porównania ma wartość TRUE jeżeli:
==	IN1 jest równe IN2
<>	IN1 nie jest równe IN2
>=	IN1 jest większe od lub równe IN2
<=	IN1 jest mniejsze od lub równe IN2
>	IN1 jest większe od IN2
<	IN1 jest mniejsze od IN2

LAD **FBD**

Po kliknięciu instrukcji w programie edytora, z rozwijanego menu wybrać typ porównania oraz typ danych.

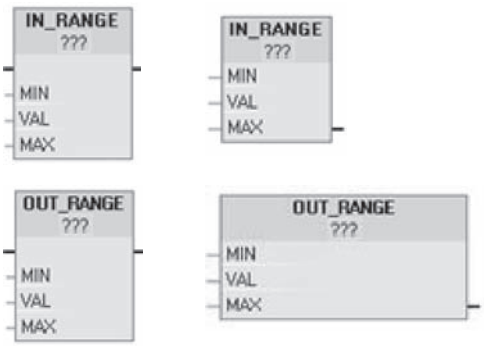
Parametr	Typ danych	Opis
IN1, IN2	SINT, INT, DINT, USINT, UINT, UDINT, REAL, STRING, CHAR, TIME, DTL, Constant	Wartości do porównania

Instrukcje IN RANGE i OUT OF RANGE

Instrukcje IN_RANGE i OUT_RANGE są stosowane do testowania czy wartości wejściowe mieszczą się, czy nie w wyspecyfikowanym zakresie. Jeśli wynik porównania ma wartość TRUE, to na wyjściu bloku pojawia się stan TRUE.

Typ relacji	Wynik porównania ma wartość TRUE jeżeli:
IN_RANGE	MIN <= VAL <= MAX
OUT_RANGE	VAL < MIN lub VAL > MAX

LAD **FBD**



Po kliknięciu instrukcji w programie edytora, z rozwijanego menu można wybrać typ danych.

Parametr	Typ danych	Opis
MIN, VAL, MAX	SINT, INT, DINT, USINT, UINT, UDINT, REAL, Constant	Wejścia komparatora

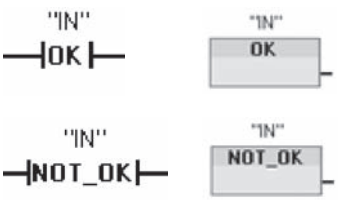
Parametry wejściowe MIN, VAL i MAX muszą być tego samego typu.

Instrukcje OK i Not OK

Instrukcje OK i NOT_OK są stosowane do testowania czy wejściowe dane referencyjne są liczbami typu REAL czy nie. Kiedy styk LAD ma wartość TRUE, wtedy ten styk jest aktywowany i przepuszcza zasilanie. Kiedy wynik porównania FBD ma wartość TRUE, wtedy na wyjściu tego bloku jest stan TRUE.

Instrukcja	Wynik testowania czy liczba jest typu REAL ma wartość TRUE jeżeli:
OK	Wartość wejściowa jest liczbą typu REAL
NOT_OK	Wartość wejściowa nie jest liczbą typu REAL

LAD **FBD**



Parametr	Typ danych	Opis
IN	REAL	Dana wejściowa

6.1.5 Operacje arytmetyczne

Instrukcje dodawania, **odejmowania**, **mnożenia i dzielenia**

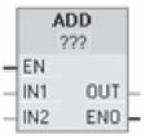
Ramka z instrukcją arytmetyczną jest stosowana do programowania podstawowych operacji arytmetycznych:

- ADD: Dodawanie (IN1 + IN2 = OUT)
- SUB: Odejmowanie (IN1 – IN2 = OUT)
- MUL: Mnożenie (IN1 ● IN2 = OUT)
- DIV: Dzielenie (IN1 / IN2 = OUT)

Operacja dzielenia liczb całkowitych powoduje obcinanie ułamkowej części ilorazu tak, by wynik był liczbą całkowitą.

LAD

FBD



Kliknięcie poniżej nazwy bloku pozwala wybrać z rozwijanego menu typ danych.

UWAGA

Parametry podstawowych instrukcji artymetycznych IN1, IN2 i OUT muszą być tego samego typu.

Parametr	Typ danych	Opis
IN1, IN2	SINT, INT, DINT, USINT, UINT, UDINT, REAL, Constant	Wejścia operacji artymetycznej
OUT	SINT, INT, DINT, USINT, UINT, UDINT, REAL	Wyjście operacji artymetycznej

Kiedy instrukcja arytmetyczna jest uaktywniona (EN = 1), wtedy na wartościach wejściowych (IN1 i IN2) jest wykonywana określona operacja arytmetyczna, a jej wynik jest zapisywany pod adres pamięci określony w parametrze wyjściowym (OUT). Po pomyślnym zakończeniu operacji, instrukcja ustawia ENO = 1.

Kody warunkowe

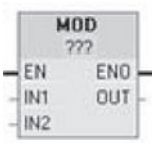
STATUS ENO	Opis
1	Brak błędu.
0	Wartość wyniku jest poza dozwolonym zakresem dla danej wybranego typu
0	Dzielenie przez 0 (IN2 = 0)
0	REAL: jeżeli jedna z wartości wejściowych jest NAN (not a number – nie jest liczbą) lub wynik jest INF (infinity – nieskończony), to zwracany jest NAN

STATUS ENO	Opis
0	ADD REAL: jeżeli obie wartości wejściowe IN są INF z różnymi znakami, to operacja jest niedozwolona i zwracany jest NAN
0	SUB REAL: jeżeli obie wartości wejściowe IN są INF z jednakowymi znakami, to operacja jest niedozwolona i zwracany jest NAN
0	MUL REAL: jeżeli jedna wartość IN jest 0, a druga INF, to operacja jest niedozwolona i zwracany jest NAN
0	DIV REAL: jeżeli obie wartości IN są 0 lub INF, to operacja jest niedozwolona i zwracany jest NAN

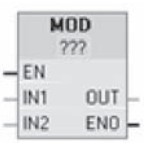
6.1.5.1 Instrukcja MOD

Instrukcja MOD (modulo) jest stosowana w celu wykonania operacji arytmetycznej IN1 modulo IN2. Ta operacja jest określona równaniem IN1 MOD IN2 = IN1 – (IN1 / IN2) * IN2 = parametr OUT.

LAD



FBD



Kliknięcie poniżej nazwy bloku pozwala wybrać z rozwijanego menu typ danych.

UWAGA
Parametry IN1, IN2 i OUT muszą być tego samego typu.

Parametr	Typ danych	Opis
IN1, IN2	INT, INT, DINT, USINT, UINT, UDINT, Constant	Wejścia operacji modulo
OUT	INT, INT, DINT, USINT, UINT, UDINT	Wyjście operacji modulo

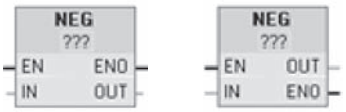
Kody warunkowe

STATUS ENO	Opis
1	Brak błędu
0	Wartość IN2 = 0 (dzielenie przez 0), parametrowi OUT jest nadawana wartość zero

Instrukcja NEG

Instrukcja NEG (negacja) jest stosowana do zmiany arytmetycznego znaku wartości parametru IN; wynik jest zapamiętywany jako parametr OUT.

LAD **FBD**



Kliknięcie poniżej nazwy bloku pozwala wybrać z rozwijanego menu typ danych.

UWAGA

Parametry IN i OUT muszą być tego samego typu.

Parametr	Typ danych	Opis
IN	SINT, INT, DINT, REAL, Constant	Wejście operacji arytmetycznej
OUT	SINT, INT, DINT, REAL	Wyjście operacji arytmetycznej

Kody warunkowe

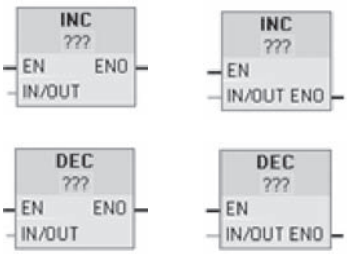
STATUS ENO	Opis
1	Brak błędu
0	Wartość wyniku jest poza dozwolonym zakresem dla danej wybranego typu. Przykład dla SINT: NEG(-128) daje w wyniku wartość +128, która przekracza maksymalny zakres dla tego typu danej.

Instrukcje inkrementacji i dekrementacji

Instrukcje INC i DEC są stosowane do:

- Inkrementacji wartości liczby całkowitej ze znakiem lub bez znaku.
- Dekrementacji wartości liczby całkowitej ze znakiem lub bez znaku.

LAD **FBD**



Kliknięcie poniżej nazwy bloku pozwala wybrać z rozwijanego menu typ danych.

Parametr	Typ danych	Opis
IN/OUT	SINT, INT, DINT, USINT, UINT, UDINT	Wejście i wyjście operacji arytmetycznej

INC (inkrementacja): wartość parametru IN/OUT + 1 = wartość parametru IN/OUT

DEC (dekrementacja): wartość parametru IN/OUT – 1 = wartość parametru IN/OUT

Kody warunkowe

STATUS ENO	Opis
1	Brak błędu
0	Wartość wyniku jest poza dozwolonym zakresem dla danej wybranego typu. Przykład dla SINT: INC(127) daje w wyniku wartość 128, która przekracza maksymalny zakres dla tego typu danej.

Instrukcja obliczania wartości bezwzględnej

Instrukcja ABS jest stosowana do wyznaczenia wartości bezwzględnej wejściowej liczby całkowitej lub rzeczywistej ze znakiem IN, a wynik jest zachowywany jako parametr OUT.

LAD **FBD**

Kliknięcie poniżej nazwy bloku pozwala wybrać z rozwijanego menu typ danych.

UWAGA

Parametry IN i OUT muszą być tego samego typu.

Parametr	Typ danych	Opis
IN	SINT, INT, DINT, REAL	Wejście operacji arytmetycznej
OUT	SINT, INT, DINT, REAL	Wejście operacji arytmetycznej

Kody warunkowe

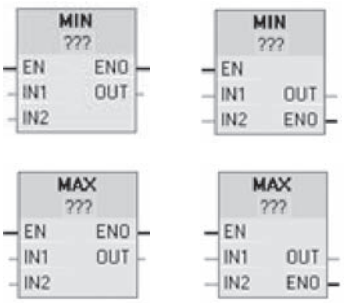
STATUS ENO	Opis
1	Brak błędu
0	Wartość wyniku jest poza dozwolonym zakresem dla danej wybranego typu. Przykład dla SINT: ABS(-128) daje w wyniku wartość +128, która przekracza maksymalny zakres dla tego typu danej.

Instrukcje MIN i MAX

Instrukcje MIN (minimum) i MAX (maksimum) są stosowane zgodnie z następującym opisem:

- MIN porównuje wartości dwóch parametrów IN1 i IN2 i minimalną (mniejszą) zapisuje jako wartość parametru OUT.
- MAX porównuje wartości dwóch parametrów IN1 i IN2 i maksymalną (większą) zapisuje jako wartość parametru OUT.

LAD **FBD**



Kliknięcie poniżej nazwy bloku pozwala wybrać z rozwijanego menu typ danych.

UWAGA

Parametry IN1, IN2 i OUT muszą być tego samego typu.

Parametr	Typ danych	Opis
IN1, IN2	SINT, INT, DINT, USINT, UINT, UDINT, REAL,Constant	Wejścia operacji arytmetycznej
OUT	SINT, INT, DINT, USINT, UINT, UDINT, REAL	Wejście operacji arytmetycznej

Kody warunkowe

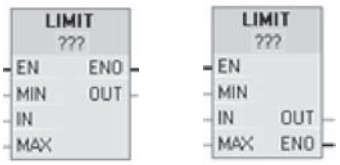
STATUS ENO	Opis
1	Brak błędu
0	Tylko dla danych typu REAL: ● Dane na jednym lub obu wejściach nie są typu REAL (NAN) ● Wynik OUT wynosi ± nieskończoność (INF)

Instrukcja Limit

Instrukcja Limit służy do sprawdzania, czy wartość parametru IN zawiera się wewnątrz zakresu określonego parametrami MIN I MAX. Jeżeli wartość IN wykracza poza ten zakres, to OUT pozostaje obcięta na wartości MIN lub MAX.

- Jeżeli wartość IN zawiera się w wyspecyfikowanym zakresie, to ta wartość IN jest zapamiętywana jako parametr OUT.
- Jeżeli wartość IN wykracza poza wyspecyfikowany zakres, to parametr OUT przyjmuje wartość parametru MIN (jeżeli wartość IN jest mniejsza od wartości MIN) lub wartość parametru MAX (jeżeli wartość IN jest większa od wartości MAX).

LAD **FBD**



Kliknięcie poniżej nazwy bloku pozwala wybrać z rozwijanego menu typ danych.

UWAGA

Parametry MIN, IN, MAX i OUT muszą być tego samego typu.

Parametr	Typ danych	Opis
MIN, IN i MAX	SINT, INT, DINT, USINT, UINT, UDINT, REAL, Constant	Wejścia operacji arytmetycznej
OUT	SINT, INT, DINT, USINT, UINT, UDINT, REAL	Wejście operacji arytmetycznej

Kody warunkowe

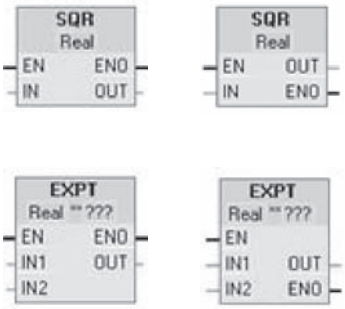
STATUS ENO	Opis
1	Brak błędu
0	REAL: jeżeli jedna lub więcej wartości spośród MIN, IN i MAX nie jest liczbą (NAN), to zwracany jest NAN
0	Jeżeli MIN jest większa od MAX, to parametrowi OUT jest nadawana wartość IN

Instrukcje arytmetyczne zmiennoprzecinkowe

Instrukcje zmiennoprzecinkowe stosuje się podczas programowania operacji arytmetycznych wykorzystujących dane typu REAL:

- SQR: podnoszenie do kwadratu (IN^2 = OUT)
- SQRT: pierwiastek kwadratowy ($\sqrt{IN}$ = OUT)
- LN: logarytm naturalny (LN(IN) = OUT)
- EXP: funkcja wykładnicza o podstawie e (e^{IN} = OUT), gdzie e = 2,71828182845904523536
- SIN: sinus (sin(IN radianów) = OUT)
- COS: cosinus (cos(IN radianów) = OUT)
- TAN: tangens (tan(IN radianów) = OUT)
- ASIN: arcus sinus (arcsine(IN) = OUT radianów), gdzie sin(OUT radianów) = IN
- ACOS: arcus cosinus (arccos(IN) = OUT radianów), gdzie cos(OUT radianów) = IN
- ATAN: arcus tangens (arctan(IN) = OUT radianów), gdzie tan(OUT radianów) = IN
- FRAC: ułamek (część ułamkowa liczby zmiennoprzecinkowej IN = OUT)
- EXPT: funkcja wykładnicza o dowolnej podstawie ($IN1^{IN2}$ = OUT)

LAD **FBD**



Kliknięcie poniżej nazwy bloku pozwala wybrać z rozwijanego menu typ danych. Parametry IN1 i OUT funkcji EXPT są zawsze typu REAL. Użytkownik może wybrać typ danej IN2 funkcji wykładniczej.

Parametr	Typ danych	Opis
IN, IN2	REAL, Constant	Wejścia
IN2	SINT, INT, DINT, USINT, UINT, UDINT, REAL, Constant	Wejście funkcji EXPT
OUT	REAL	Wyjście

Kody warunkowe

STATUS ENO	Instrukcja	Warunek	Wynik (OUT)
1	Wszystkie	Brak błędu	Wynik jest ważny
0	SQR	Wynik poza ważnym zakresem REAL	+INF
		IN jest +/-NAN (nie jest liczbą)	+NAN
	SQRT	IN jest ujemna	-NAN
		IN jest +/-INF (nieskończoność) lub +/-NAN	+/-INF lub +/-NAN
	LN	IN jest 0.0, ujemna, -INF lub -NAN	-NAN
		IN jest +INF lub +NAN	+INF lub +NAN
	EXP	Wynik poza ważnym zakresem REAL	+INF
		IN jest +/-NAN	+/-NAN
	SIN, COS, TAN	IN jest +/-INF lub +/-NAN	+/-INF lub +/-NAN
	ASIN, ACOS	Wynik poza ważnym zakresem: -1.0…+1.0	+NAN
		IN jest +/-NAN	+/-NAN
	ATAN	IN jest +/-NAN	+/-NAN
	FRAC	IN jest +/-INF lub +/-NAN	+NAN
	EXPT	IN1 jest +INF I IN2 nie jest -INF	+INF
		IN1 jest ujemna lub -INF	+NAN jeśli IN2 jest REAL, -INF w przeciwnym wypadku
		IN1 lub IN2 jest +/-NAN	+NAN
		IN1 jest 0.0 I IN2 jest REAL (tylko)	+NAN

6.1.6 Instrukcja MOVE

Instrukcja MOVE jest stosowana do kopiowania elementów danych do miejsca pamięci o nowym adresie oraz konwersji jednego typu danych na inny. Dane wejściowe instrukcji MOVE nie ulegają zmianie.

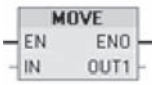
- MOVE: kopiuje elementy danych pamiętane pod określonym adresem w miejsce pamięci o nowym adresie.
- MOVE_BLK: przerywalna instrukcja MOVE kopiująca blok danych pod nowy adres.
- UMOVE_BLK: nieprzerywalna instrukcja MOVE kopiująca blok danych pod nowy adres.

UWAGA

Reguły wykonywania operacji kopiowania danych:

- W celu skopiowania danych typu BOOL, należy stosować SET_BF, RESET_BF, R, S lub cewkę wyjściową (LAD).
- W celu skopiowania pojedynczej danej elementarnej, należy stosować MOVE.
- W celu skopiowania tablicy danych elementarnych, należy stosować MOVE_BLK lub UMOVE_BLK.
- W celu skopiowania struktury, należy stosować MOVE.
- W celu skopiowania łańcucha, należy stosować S_CONV.
- W celu skopiowania pojedynczego znaku w łańcuchu, należy stosować MOVE.
- Nie można stosować instrukcji MOVE_BLK i UMOVE_BLK w celu kopiowania tablic lub struktur do obszarów pamięci I, Q lub M.

LAD



FBD

MOVE		
Parametr	**Typ danych**	**Opis**
IN	SINT, INT, DINT, USINT, UINT, UDINT, REAL, BYTE, WORD, DWORD, CHAR, ARRAY, STRUCT, DTL, TIME	Adres źródłowy
OUT	SINT, INT, DINT, USINT, UINT, UDINT, REAL, BYTE, WORD, DWORD, CHAR, ARRAY, STRUCT, DTL, TIME	Adres docelowy

MOVE_BLK, UMOVE_BLK		
Parametr	**Typ danych**	**Opis**
IN	SINT, INT, DINT, USINT, UINT, UDINT, REAL, BYTE, WORD, DWORD	Początkowy adres źródłowy
COUNT	UINT	Liczba elementów danych do skopiowania
OUT	SINT, INT, DINT, USINT, UINT, UDINT, REAL, BYTE, WORD, DWORD	Początek adresu docelowego

Instrukcja MOVE kopiuje pojedynczy element danych znajdujący się pod adresem źródłowym wyspecyfikowanym w parametrze IN do miejsca w pamięci o adresie wyspecyfikowanym w parametrze OUT.

Instrukcje MOVE_BLK i UMOVE_BLK mają dodatkowy parametr COUNT. COUNT określa ile elementów danych ma zostać skopiowanych. Liczba bajtów przypadająca na kopiowany element zależy od typu danych przypisanego parametrom IN i OUT, który jest określony w tablicy *tagów* PLC.

Instrukcje MOVE_BLK i UMOVE_BLK różnią się sposobem w jaki obsługują przerwania.:

- Podczas wykonywania instrukcji MOVE_BLK przerwania są kolejkowane i obsługiwane. Instrukcji MOVE_BLK należy używać wtedy, kiedy adres pod który są kopiowane dane nie jest wykorzystywany przez podprogram OB obsługi przerwania – gdyby był używany, to skopiowane dane mogłyby być niespójne. Jeżeli operacja MOVE_BLK jest przerwana to ostatnie kopiowanie elementu danych jest zakończone, a same dane pod docelowym adresem są spójne. Operacja MOVE_BLK jest kontynuowana po zakończeniu obsługi przerwania.
- Podczas wykonywania instrukcji UMOVE_BLK przerwania są kolejkowane, ale nie obsługiwane aż do zakończenia kopiowania. Instrukcji UMOVE_BLK należy używać wtedy, kiedy kopiowanie musi zostać zakończone i spójność danych zachowana przed rozpoczęciem wykonywania podprogramu OB obsługi przerwania.

Kody warunkowe

Po wykonaniu instrukcji MOVE wartość ENO zawsze jest TRUE.

STATUS ENO	Warunek	Wynik
1	Brak błędu	Wszystkie COUNT elementów zostało pomyślnie skopiowanych
0	Zakres źródłowy (IN) lub docelowy (OUT) przekracza dostępny obszar pamięci	Elementy pasujące zostały skopiowane. Żaden element nie został skopiowany fragmentarycznie.

Instrukcje wypełniania

Instrukcje FILL_BLK i UFILL_BLK stosuje się w następujący sposób:

FILL_BLK: przerywalna instrukcja wypełniania, wypełnia miejsca o określonym zakresie adresów kopią wyspecyfikowanego elementu danych.

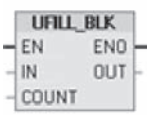
UFILL_BLK: przerywalna instrukcja wypełniania, wypełnia miejsca o określonym zakresie adresów kopiując do nich wyspecyfikowany element danych.

UWAGA

Reguły wykonywania operacji kopiowania danych:

- W celu wypełnienia danymi typu BOOL, należy stosować SET_BF, RESET_ BF, R, S lub cewkę wyjściową (LAD).
- W celu wypełnienia pojedynczą daną typu elementarnego, należy użyć MOVE.
- W celu wypełnienia tablicy danymi typu elementarnego, należy użyć FILL_ BLK lub UFILL_BLK.
- W celu wypełnienia pojedynczego znaku w łańcuchu należy stosować MOVE.
- Nie można stosować instrukcji FILL_BLK i UFILL_BLK w celu wypełniania tablic w obszarach pamięci I, Q lub M.

LAD



FBD

Parametr	Typ danych	Opis
IN	SINT, INT, DINT, USINT, UINT, UDINT, REAL, BYTE, WORD, DWORD	Adres źródłowy danych
COUNT	USINT, UINT	Liczba elementów danych do skopiowania
OUT	SINT, INT, DINT, USINT, UINT, UDINT, REAL, BYTE, WORD, DWORD	Adres docelowy danych

Instrukcje FILL_BLK i UFILL_BLK kopiują źródłowy element danych IN do miejsca docelowego, którego adres początkowy specyfikuje parametr OUT. Proces kopiowania jest powtarzany i blok o sąsiednich adresach jest wypełniany tyle razy, aż liczba kopiowań zrówna się z wartością parametru COUNT.

Instrukcje FILL_BLK i UFILL_BLK różnią się sposobem w jaki obsługują przerwania.

- Podczas wykonywania instrukcji FILL_BLK przerwania są kolejkowane i obsługiwane. Instrukcji FILL_BLK należy używać wtedy, kiedy adres pod który są kopiowane dane nie jest wykorzystywany przez podprogram OB obsługi przerwania lub, jeśli jest używany, to dane docelowe mogą być niespójne.
- Podczas wykonywania instrukcji UFILL_BLK przerwania są kolejkowane, ale nie obsługiwane aż do zakończenia instrukcji. Instrukcji UFILL_BLK należy używać wtedy, kiedy wypełnianie musi zostać zakończone i spójność danych zachowana przed rozpoczęciem wykonywania podprogramu OB obsługi przerwania.

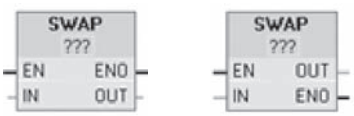
Kody warunkowe

STATUS ENO	Warunek	Wynik
1	Brak błędu	Element IN został pomyślnie skopiowany do wszystkich COUNT miejsc docelowych.
0	Zakres docelowy (OUT) przekracza dostępny obszar pamięci	Elementy pasujące zostały skopiowane. Żaden element nie został skopiowany fragmentarycznie.

6.1.6.1 Instrukcja SWAP (zamiany)

Instrukcja SWAP jest stosowana do zamiany kolejności bajtów w elementach danych 2- i 4-bajtowych. Nie jest wykonywana żadna zmiana kolejności bitów w poszczególnych bajtach. Po wykonaniu instrukcji SWAP wartość ENO zawsze jest TRUE.

LAD **FBD**



Kliknięcie poniżej nazwy bloku pozwala wybrać z rozwijanego menu typ danych.

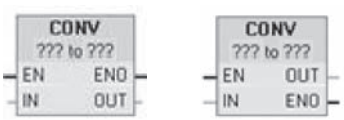
Parametr	Typ danych	Opis
IN	WORD, DWORD	Wejściowa kolejność bajtów IN
OUT	WORD, DWORD	Wyjściowa, odwrócona kolejność bajtów OUT

	Przykład: parametr IN = MB0 przed wykonaniem instrukcji SWAP				Przykład: parametr IN = MB4 po wykonaniu instrukcji SWAP			
adres	MB0	MB1			MB4	MB5		
W#16#1234	12	34			34	12		
WORD	MSB	LSB			MSB	LSB		
adres	MB0	MB1	MB2	MB3	MB4	MB5	MB6	MB7
DW#16#12345678	12	34	56	78	78	56	34	12
DWORD	MSB			LSB	MSB			LSB

6.1.7 Instrukcja Convert

Instrukcja CONVERT jest stosowana do konwersji typu elementu danych z jednego na inny. Kliknięcie poniżej nazwy bloku pozwala wybrać z rozwijanego menu typy danych IN i OUT. Po dokonaniu wyboru typu danych źródłowych (*convert from*) jest rozwijana lista możliwych konwersji (*convert to*). Konwersje z i do BCD16 są możliwe tylko w przypadku danych typu INT. Konwersje z i do BCD32 są możliwe tylko w przypadku danych typu DINT.

LAD **FBD**



Kliknięcie poniżej nazwy bloku pozwala wybrać z rozwijanego menu typ danych.

Parametr	Typ danych	Opis
IN	SINT, INT, DINT, USINT, UINT, UDINT, BYTE, WORD, DWORD, REAL, BCD16, BCD32	Wartość IN
OUT	SINT, INT, DINT, USINT, UINT, UDINT, BYTE, WORD, DWORD, REAL, BCD16, BCD32	Wartość IN zamieniona na nowy typ danych

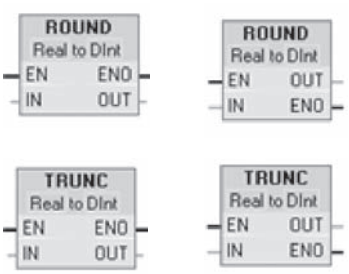
Kody warunkowe

STATUS ENO	Opis	Wynik OUT
1	Brak błędu	Ważny wynik
0	IN jest +/- INF lub +/- NAN	+/- INF lub +/- NAN
0	Wynik wykracza poza ważny zakres dla danych typu OUT	OUT jest zapisywany najmniej znaczącymi bajtami IN

Instrukcje zaokrąglania i obcinania

- ROUND zamienia liczbę rzeczywistą na całkowitą. Część ułamkowa liczby rzeczywistej jest zaokrąglana do najbliższej wartości całkowitej (IEEE – *round to nearest*).
- TRUNC zamienia liczbę rzeczywistą na całkowitą. Część ułamkowa liczby rzeczywistej jest obcinana do zera.(IEEE – *round to zero*).

LAD **FBD**



Parametr	Typ danych	Opis
IN	REAL	Wejście zmiennoprzecinkowe
OUT	SINT, INT, DINT, USINT, UINT, UDINT, REAL	Wyjście po zaokrągleniu lub obcięciu

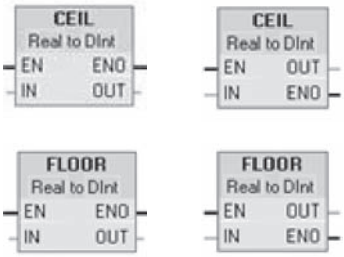
Kody warunkowe

STATUS ENO	Opis	Wynik OUT
1	Brak błędu	Ważny wynik
0	IN jest +/- INF lub +/- NAN	+/- INF lub +/- NAN

Instrukcje wyznaczania najbliższych liczb całkowitych

- CEIL zamienia liczbę rzeczywistą na najmniejszą liczbę całkowitą większą lub równą liczbie rzeczywistej (IEEE – *round to +infinity*).
- FLOOR zamienia liczbę rzeczywistą na największą liczbę całkowitą mniejszą lub równą liczbie rzeczywistej (IEEE – *round to +infinity*).

LAD **FBD**



Parametr	Typ danych	Opis
IN	REAL	Wejście zmiennoprzecinkowe
OUT	SINT, INT, DINT, USINT, UINT, UDINT, REAL	Wyjście po dokonaniu konwersji

Kody warunkowe

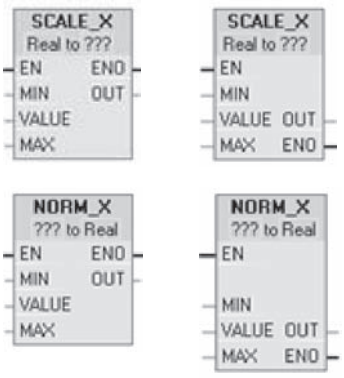
STATUS ENO	Opis	Wynik OUT
1	Brak błędu	Ważny wynik
0	IN jest +/- INF lub +/- NAN	+/- INF lub +/- NAN

6.1.7.1 Instrukcje skalowania i normalizacji

Instrukcje skalowania i normalizacji

- SCALE_X skaluje znormalizowany parametr VALUE (0.0 <= VALUE <= 1.0) do typu danej i zakresu wartości wyspecyfikowanych przez parametry MIN i MAX
  OUT = VALUE (MAX – MIN) + MIN.
- NORM_X normalizuje parametr VALUE wewnątrz zakresu wartości wyspecyfikowanych przez parametry MIN i MAX.
  OUT = (VALUE – MIN) / (MAX – MIN), gdzie (0.0 <= OUT <= 1.0).

LAD **FBD**



Kliknięcie poniżej nazwy bloku pozwala wybrać z rozwijanego menu typ danych.

Dla SCALE_X, parametry MIN, MAX i OUT muszą być tego samego typu.

Dla NORM_X, parametry MIN, VALUE i MAX muszą być tego samego typu.

Parametr	Typ danych	Opis
MIN	SINT, INT, DINT, USINT, UINT, UDINT, REAL	Minimalna wartość wejściowa zakresu
VALUE	SCALE_X: REAL NORM_X: SINT, INT, DINT, USINT, UINT, UDINT, REAL	Wartość wejściowa do skalowania lub normalizacji
MAX	SINT, INT, DINT, USINT, UINT, UDINT, REAL	Maksymalna wartość wejściowa zakresu
OUT	SCALE_X: SINT, INT, DINT, USINT, UINT, UDINT, REAL NORM_X: REAL	Przeskalowana lub znormalizowana wartość wyjściowa

UWAGA

Parametr VALUE dla SCALE_X powinien być ograniczony do zakresu (0.0 <= VALUE <= 1.0).

Jeżeli parametr VALUE jest mniejszy od 0.0 lub większy od 1.0, to:

- W wyniku liniowej operacji skalowania otrzymane wartości OUT mogą być mniejsze niż parametr MIN lub większe niż parametr MAX, ale zgodne z zakresem wartości danych typu ustalonego dla OUT. W takich przypadkach po wykonaniu operacji SCALE_X ustawiany jest ENO = TRUE.
- Możliwe jest, że przeskalowane liczby nie zawierają się w dopuszczalnym zakresie danych typu ustalonego dla OUT. W takich przypadkach wartość parametru OUT przyjmuje wartość pośrednią równą najmniej znaczącej części rzeczywistej liczby skalowanej przed wykonaniem końcowej konwersji na daną typu OUT. Wówczas po wykonaniu operacji SCALE_X ustawiany jest ENO = FALSE.

Parametr VALUE dla NORM_X powinien być ograniczone do zakresu (MIN <= VALUE <= MAX).

Jeżeli parametr VALUE jest mniejszy niż MIN lub większy niż MAX, to operacja liniowego skalowania może w wyniku dać znormalizowaną wartość OUT, która jest mniejsza od 0.0 lub większa od 1.0. W takim przypadku po wykonaniu operacji SCALE_X ustawiany jest ENO = TRUE.

Kody warunkowe

STATUS ENO	Warunek	Wynik OUT
1	Brak błędu	Ważny wynik
0	Wynik wykracza poza ważny zakres dla danych typu OUT	Wynik pośredni: najmniej znacząca część rzeczywistej liczby skalowanej przed wykonaniem końcowej konwersji na daną typu OUT.
0	Parametry MAX <= MIN	SCALE_X: najmniej znacząca część liczby rzeczywistej VALUE wypełnia przestrzeń OUT.
0	Parametr VALUE = +/- INF lub +/- NAN	VALUE jest wpisywany do OUT.

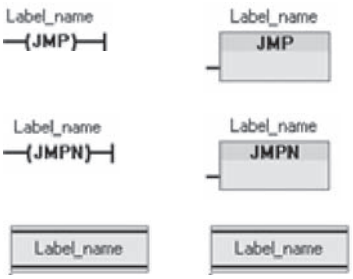
6.1.8 Sterowanie wykonywaniem programu

Instrukcje skoków i etykiety

Instrukcje sterujące wykonaniem programu są wykorzystywane do warunkowego wykonywania ciągu operacji:

- JMP: Jeżeli do cewki JMP dochodzi zasilanie (LAD) lub na wejściu bloku jest wartość TRUE (FBD), to program jest kontynuowany od pierwszej instrukcji po wyspecyfikowanej etykiecie.
- JMPN: Jeżeli do cewki JMP nie dochodzi zasilanie (LAD) lub na wejściu bloku jest wartość FALSE (FBD), to program jest kontynuowany od pierwszej instrukcji po wyspecyfikowanej etykiecie.
- *Label*: Docelowa etykieta dla instrukcji skoku JMP i JMPN.

LAD **FBD**



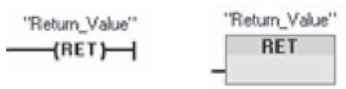
Parametr	Typ danych	Opis
Label_name	Label identifier	Identyfikator etykiety dla instrukcji skoków i odpowiadająca mu etykieta oznaczająca docelowe miejsce skoku w programie.

Nazwy etykiet nadaje się poprzez bezpośrednie wpisywanie nazwy w instrukcji LABEL. Nazwę etykiety dla instrukcji JMP i JMPN wybiera się spośród dostępnych nazw używając ikony pomocy dla parametrów. Można również bezpośrednio wpisać nazwę etykiety do instrukcji JMP lub JMPN.

Wykonanie instrukcji sterującej Return_Value (RET)

Instrukcję RET stosuje się do zakończenia wykonywania bieżącego bloku.

LAD **FBD**



Parametr	Typ danych	Opis
Return_Value	BOOL	Parametr „Return_value" instrukcji RET jest w bloku wywoływanym przypisywany parametrowi wyjściowemu ENO bloku wywołującej blok.

Opcjonalną instrukcję RET wykorzystuje się do zakończenia wykonywania bieżącego bloku. Zakończenie wykonywania bieżącego bloku nastąpi w tym miejscu i instrukcje znajdujące się za instrukcją RET nie będą wykonywane wtedy i tylko wtedy jeżeli na wejściu cewki RET znajduje się zasilanie (LAD) lub na wejściu bloku RET jest stan TRUE (FBD). Jeżeli bieżącym blokiem jest OB., to parametr „Return_Value" jest ignorowany. Jeżeli bieżącym blokiem jest FC lub FB, to wartość parametru „Return_Value" jest zwracana do procedury wywołującej jako wartość ENO bloku wywołującej.

Użytkownik nie ma obowiązku wpisywania instrukcji RET jako ostatniej instrukcji bloku; jest to wykonywane automatycznie. W pojedynczym bloku może występować wiele instrukcji RET.

Przykładowy sposób stosowania instrukcji RET wewnątrz bloku kodu FC:

1. Należy stworzyć nowy projekt i dodać FC.
2. Należy wykonać edycję FC:
   - Dodać instrukcje z drzewa instrukcji.
   - Dodać instrukcję RET; przy czym dla parametru „Return_Value" należy określić:

     TRUE, FALSE albo podać miejsce w pamięci, gdzie znajduje się wartość zwracana.
   - Dodać więcej instrukcji.
3. Należy wywołać FC z MAIN [OB1].

Wejście EN bloku FC w kodzie MAIN musi mieć wartość TRUE aby nastąpiło rozpoczęcie wykonywania FC.

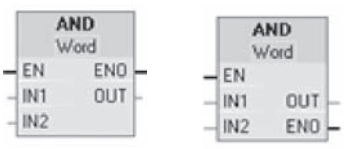
Po wykonaniu FC, w którym zasilanie na wejściu instrukcji RET ma wartość TRUE, na wyjściu ENO bloku FC w kodzie MAIN pojawi się wartość wyspecyfikowana przez instrukcję RET w FC.

6.1.9 Operacje logiczne

Instrukcje AND, OR i XOR

- AND: logiczna operacja AND dla danych typu BYTE, WORD i DWORD
- OR: logiczna operacja OR dla danych typu BYTE, WORD i DWORD
- XOR: logiczna operacja XOR dla danych typu BYTE, WORD i DWORD

LAD **FBD**



Kliknięcie poniżej nazwy bloku pozwala wybrać z rozwijanego menu typ danych.

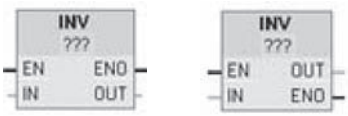
Parametr	Typ danych	Opis
IN1, IN2	BYTE, WORD, DWORD	Wejścia logiczne
OUT	BYTE, WORD, DWORD	Wyjście logiczne

Podczas wyboru typu danych jest ustawiany taki sam typ danych parametrów IN1, IN2 i OUT. Odpowiadające sobie bity IN1 i IN2 są argumentami operacji logicznej, której wynik jest wpisywany do OUT. Po zakończeniu wykonywania powyższych instrukcji, ENO ma zawsze wartość TRUE.

Instrukcja inwersji

Instrukcja INV jest stosowana do wyznaczenia dwójkowego uzupełnienia do jedności parametru IN. Uzupełnienie do jedności jest wykonywane poprzez inwersję każdego bitu parametru IN (zamianę każdego 0 na 1 i 1 na 0). Po zakończeniu wykonywania instrukcji, ENO ma zawsze wartość TRUE.

LAD **FBD**



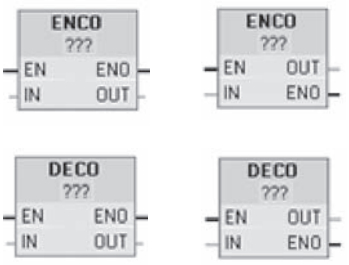
Kliknięcie poniżej nazwy bloku pozwala wybrać z rozwijanego menu typ danych.

Parametr	Typ danych	Opis
IN	SINT, INT, DINT, USINT, UINT, UDINT, BYTE, WORD, DWORD	Element podlegający inwersji
OUT	SINT, INT, DINT, USINT, UINT, UDINT, BYTE, WORD, DWORD	Wyjście po inwersji

Instrukcje enkodowania i dekodowania

- ENCO koduje ciąg bitów na liczbę dwójkową.
- DECO dekoduje liczbę dwójkową na ciąg bitów.

LAD **FBD**



Kliknięcie poniżej nazwy bloku pozwala wybrać z rozwijanego menu typ danych.

Parametr	Typ danych	Opis
IN	ENCO: BYTE, WORD, DWORD DECO: UINT	ENCO: ciąg bitów do zakodowania DECO: wartość do dekodowania
OUT	ENCO: INT DECO: BYTE, WORD,DWORD	ENCO: wartość zakodowana DECO: ciąg bitów zdekodowany

Instrukcja ENCO dokonuje konwersji parametru IN na liczbę dwójkową odpowiadającą ciągowi bitów znajdujących się na najmniej znaczącej pozycji w IN i zapisuje wynik jako parametr OUT. Jeżeli parametr IN jest równy 0000 0001 albo 0000 0000, to do OUT jest wpisywane 0. Jeśli parametr IN ma wartość 0000 0000 to ENO przyjmuje wartość FALSE.

Instrukcja DECO dekoduje liczbę dwójkową z parametru IN poprzez ustawianie bitów znajdujących się na odpowiadających pozycjach OUT na 1 (wszystkie pozostałe bity są ustawione na 0). Po zakończeniu wykonywania instrukcji, ENO ma zawsze wartość TRUE.

Wybór typu danej parametru OUT instrukcji DECO spośród BYTE, WORD lub DWORD ogranicza użyteczny zakres parametru IN. Jeżeli wartość parametru IN przekracza użyteczny zakres, to wykonywana jest pokazana poniżej operacja modulo pozwalająca wydobyć najmniej znaczące bity.

Zakres parametru IN instrukcji DECO:

- 3 bity (wartość 0 – 7) IN są wykorzystywane do ustalenia 1 pozycji bitu w bajcie OUT.
- 4 bity (wartość 0 – 15) IN są wykorzystywane do ustalenia 1 pozycji bitu w słowie OUT.
- 5 bitów (wartość 0 – 31) IN jest wykorzystywanych do ustalenia 1 pozycji bitu w podwójnym słowie OUT.

Wartość IN instrukcji DECO		**Wartość OUT instrukcji DECO (dekodowanie pojedynczej pozycji bitu)**
		Typ OUT: BYTE (8 bitów)
Min. IN	0	00000001
Maks. IN	7	10000000
		Typ OUT: WORD (16 bitów)
Min. IN	0	0000000000000001
Maks. IN	15	1000000000000000
		Typ OUT: DWORD (32 bity)
Min. IN	0	00000000000000000000000000000001
Maks. IN	31	10000000000000000000000000000000

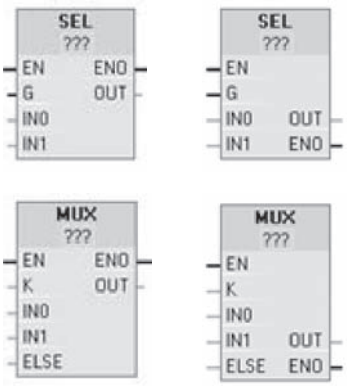
Kody warunkowe dla ENCO

STATUS ENO	**Warunek**	**Wynik (OUT)**
1	Brak błędu	Ważny wynik
0	IN jest zerem	OUT ustawiony na zero

Instrukcje wyboru (SEL) i multipleksowania (MUX)

- SEL, w zależności od wartości parametru G, przypisuje jedną z dwóch wartości wejściowych parametrowi OUT.
- MUX, w zależności od wartości parametru K, przypisuje jedną z wielu wartości wejściowych parametrowi OUT. Jeżeli parametr K wykracza poza dozwolony zakres, to parametrowi OUT jest przypisywana wartość parametru ELSE.

LAD **FBD**



Kliknięcie poniżej nazwy bloku pozwala wybrać z rozwijanego menu typ danych.

Parametry dla SEL

Parametr	Typ danych	Opis
G	BOOL	Przełącznik selektora: • FALSE dla IN0 • TRUE dla IN1
IN0, IN1	SINT, INT, DINT, USINT, UINT, UDINT, REAL, BYTE, WORD, DWORD, TIME, CHAR	Wejścia
OUT	SINT, INT, DINT, USINT, UINT, UDINT, REAL, BYTE, WORD, DWORD, TIME, CHAR	Wyjście

Parametry dla MUX

Parametr	Typ danych	Opis
K	UINT	Wartość selektora: • 0 dla IN0 • 1 dla IN1 • ...
IN0, IN1, …	SINT, INT, DINT, USINT, UINT, UDINT, REAL, BYTE, WORD, DWORD, TIME, CHAR	Wejścia
ELSE	SINT, INT, DINT, USINT, UINT, UDINT, REAL, BYTE, WORD, DWORD, TIME, CHAR	Wartość wejściowa dla podstawienia (opcjonalnie)
OUT	SINT, INT, DINT, USINT, UINT, UDINT, REAL, BYTE, WORD, DWORD, TIME, CHAR	Wyjście

Zmienne wejściowe i zmienna wyjściowa muszą być tego samego typu danych.

- Instrukcja SEL zawsze wybiera pomiędzy dwoma wartościami wejściowymi.
- Instrukcja MUX wybrana po raz pierwszy w edytorze programu ma dwa parametry wejściowe IN, ale można ja rozszerzyć dodając więcej parametrów IN.

Do dodawania i usuwania parametrów wejściowych instrukcji MUX służy następująca metoda:

- W celu dodania wejścia należy kliknąć prawym klawiszem myszy na końcówkę wejściową jednego z istniejących parametrów IN i wybrać komendę „insert input”.
- W celu usunięcia wejścia należy kliknąć prawym klawiszem myszy na końcówkę wejściową jednego z istniejących parametrów IN (jeżeli jest więcej wejść niż oryginalne dwa) i wybrać komendę „Delete”.

Kody warunkowe

Po zakończeniu wykonywania instrukcji SEL, ENO ma zawsze wartość TRUE.

STATUS ENO (MUX)	Warunek	Wynik (OUT)
1	Brak błędu	Ważny wynik
0	K jest większe lub równe liczbie parametrów IN	Bez parametru ELSE: OUT nie ulega zmianie Z parametrem ELSE: OUT przyjmuje wartość ELSE

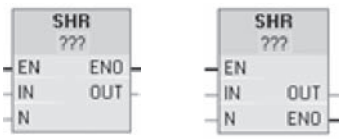
6.1.10 Przesunięcie i obrót

Instrukcja Shift

Instrukcja przesunięcia (*shift*) jest stosowana bitów parametru IN. Wynik jest przypisany parametrowi OUT. Parametr N określa o ile pozycji bitów ma nastąpić przesunięcie.

- SHR: przesunięcie bitów w prawo.
- SHL: przesunięcie bitów w lewo.

LAD **FBD**



Kliknięcie poniżej nazwy bloku pozwala wybrać z rozwijanego menu typ danych.

Parametr	Typ danych	Opis
IN	BYTE, WORD, DWORD	Bity do przesunięcia
N	UINT	Liczba pozycji bitów do przesunięcia
OUT	BYTE, WORD, DWORD	Bity po operacji przesunięcia

- Dla N = 0 nie jest wykonywane przesunięcie i do OUT jest przypisywana wartość IN.
- Na pozycje opróżnione podczas przesuwania są wpisywane zera.
- Jeżeli liczba pozycji do przesunięcia (N) przekracza liczbę bitów wartości docelowej (8 dla BYTE, 16 dla WORD i 32 dla DWORD), to oryginalna wartość zniknie i zostanie zastąpiona przez zera (do OUT zostaną wpisane same zera).
- Dla operacji przesuwania, ENO ma zawsze wartość TRUE.

Przykład operacji SHL dla danych typu WORD: podczas przesuwania z lewej strony są wpisywane zera			
IN	1110 0010 1010 1101	Wartość OUT przed pierwszym przesunięciem	1110 0010 1010 1101
		Po pierwszym przesunięciu w lewo	1100 0101 0101 1010
		Po drugim przesunięciu w lewo	1000 1010 1011 0100
		Po trzecim przesunięciu w lewo	0001 0101 0110 1000

Instrukcja Rotate

Instrukcje obrotu są stosowane do cyklicznego przesuwania bitów parametru IN. Wynik jest przypisywany do parametru OUT. Parametr N określa o ile pozycji bitów ma nastąpić obrót.

- ROR: obrót bitów w prawo.
- ROL: obrót bitów w lewo.

LAD **FBD**

Kliknięcie poniżej nazwy bloku pozwala wybrać z rozwijanego menu typ danych.

Parametr	Typ danych	Opis
IN	BYTE, WORD, DWORD	Bity do obrotu
N	UINT	Liczba pozycji bitów do obrotu
OUT	BYTE, WORD, DWORD	Bity po operacji obrotu

- Dla N = 0 nie jest wykonywany obrót i do OUT jest przypisywana wartość IN.
- Podczas obrotu bity wysuwane z jednej strony trafiają na pozycje opróżniane z drugiej strony parametru docelowego; zatem żaden oryginalny bit nie jest tracony.
- Jeżeli liczba pozycji do przesunięcia (N) przekracza liczbę bitów wartości docelowej (8 dla BYTE, 16 dla WORD i 32 dla DWORD), to obrót jest nadal wykonywany.
- Dla operacji obrotu, ENO ma zawsze wartość TRUE.

Przykład operacji ROR dla danych typu WORD: podczas obrotu bity wysuwane z prawej strony trafiają na pozycje opróżniane z lewej strony			
IN	0100 0000 0000 0001	Wartość OUT przed pierwszym obrotem	0100 0000 0000 0001
		Po pierwszym obrocie w prawo	1010 0000 0000 0000
		Po drugim obrocie w prawo	0101 0000 0000 0000

6.2 Instrukcje rozszerzone

6.2.1 Instrukcje dotyczące zegara i kalendarza

Instrukcje dotyczące daty i czasu

Instrukcje dotyczące daty i czasu są stosowane do programowania obliczeń związanych z kalendarzem i zegarem.

- T_CONV zamienia typ danej czasu: (TIME na DINT) lub (DINT na TIME).
- T_ADD dodaje wartości typu TIME i DTL: (TIME + TIME = TIME) lub (DTL + + TIME = DTL).
- T_SUB odejmuje wartości typu TIME i DTL: (TIME – TIME = TIME) lub (DTL – TIME = DTL).
- T_DIFF wyznacza różnicę między dwoma wartościami typu DTL jako wartość typu TIME: DTL – DTL = TIME.

Typ danych	Rozmiar (bity)	Zakres poprawnych wartości
TIME	32 Pamiętany jako	T#-24d_20h_31m_23s_648ms do T#24d_20h_31m_23s_647ms -2,147,483,648 ms do +2,147,483,647 ms
Struktura danych DTL		
Year (rok): UINT	16	1970 do 2554
Month (miesiąc): USINT	8	1 do 12
Day (dzień): USINT	8	1 do 31
Weekday (dzień tygodnia): USINT	8	1=Sunday (niedziela) do 7=Saturday (sobota)
Hour (godzina): USINT	8	0 do 23
Minute (minuta): USINT	8	0 do 59
Second (sekunda): USINT	8	0 do 59
Nanoseconds (nanosekundy): UDINT	32	0 do 999999999

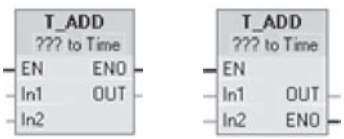
LAD **FBD**

Kliknięcie poniżej nazwy bloku pozwala wybrać z rozwijanego menu typ danych IN i OUT.

Parametr	Typ parametru	Typ danych	Opis
IN	IN	DINT, TIME	Wartość wejściowa typu TIME lub DINT
OUT	OUT	DINT, TIME	Przekonwertowana wartość DINT lub TIME

T_CONV (*Time Convert*) zamienia typ danej TIME na typ danej DINT lub odwrotnie – typ danej DINT na typ danej TIME.

LAD FBD



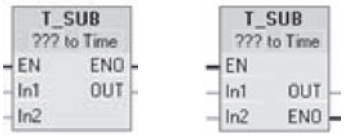
Kliknięcie poniżej nazwy bloku pozwala wybrać z rozwijanego menu typ danych IN1. Wybór typu danej IN1 jednocześnie określa typ danej parametru OUT.

Parametr	Typ parametru	Typ danych	Opis
IN1	IN	DTL, TIME	Wartość typu DTL lub TIME
IN2	IN	TIME	Wartość typu TIME jaka ma być dodana
OUT	OUT	DTL, TIME	Suma typu DTL lub TIME

T_ADD (*Time Add*) dodaje do wartości wejściowej IN1 (typu DTL lub TIME) wartość wejściową IN2 typu TIME. Parametr OUT stanowi wynik sumowania w formacie DTL lub TIME. Możliwe są operacje na dwóch typach danych, jak to pokazano poniżej:

- TIME + TIME = TIME
- DTL + TIME = DTL

LAD **FBD**



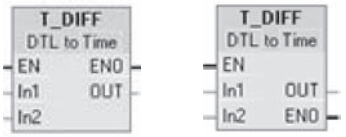
Kliknięcie poniżej nazwy bloku pozwala wybrać z rozwijanego menu typ danych IN1. Wybór typu danej IN1 jednocześnie określa typ danej parametru OUT.

Parametr	Typ parametru	Typ danych	Opis
IN1	IN	DTL, TIME	Wartość typu DTL lub TIME
IN2	IN	TIME	Wartość typu TIME jaka ma być odjęta
OUT	OUT	DTL, TIME	Różnica typu DTL lub TIME

T_SUB (*Time Subtract*) odejmuje od wartości wejściowej IN1 (typu DTL lub TIME) wartość wejściową IN2 typu TIME. Parametr OUT stanowi wynik odejmowania w formacie DTL lub TIME. Możliwe są operacje na dwóch typach danych, jak to pokazano poniżej:

- TIME – TIME = TIME
- DTL – TIME = DTL

LAD **FBD**



Parametr	Typ parametru	Typ danych	Opis
IN1	IN	DTL	Wartość typu DTL
IN2	IN	DTL	Wartość typu DTL jaka ma być odjęta
OUT	OUT	TIME	Różnica typu TIME

T_DIFF (*Time Difference*) odejmuje od wartości wejściowej IN1 typu DTL wartość wejściową IN2 typu DTL. Parametr OUT stanowi wynik odejmowania w formacie TIME.

- DTL – DTL = TIME

Kody warunkowe

ENO = 1 oznacza, że nie wystąpił żaden błąd.

ENO = 0 oraz OUT = 0 oznacza błędy:

- Nieprawidłowa wartość DTL
- Nieprawidłowa wartość TIME

Instrukcje dotyczące zegara

Instrukcje dotyczące zegara stosuje się do nastawiania i odczytywania zegara systemowego PLC. Do przedstawiania wartości czasu i daty jest stosowany format DTL.

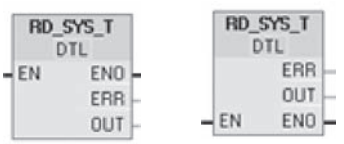
Struktura danych DTL	Rozmiar	Zakres poprawnych wartości
Year (rok): UINT	16 bitów	1970 do 2554
Month (miesiąc): USINT	8 bitów	1 do 12
Day (dzień): USINT	8 bitów	1 do 31
Weekday (dzień tygodnia): USINT	8 bitów	1=Sunday (niedziela) do 7=Saturday (sobota)
Hour (godzina): USINT	8 bitów	0 do 23
Minute (minuta): USINT	8 bitów	0 do 59
Second (sekunda): USINT	8 bitów	0 do 59
Nanoseconds (nanosekundy): UDINT	32 bity	0 do 999999999

LAD **FBD**

Parametr	Typ parametru	Typ danych	Opis
IN	IN	DTL	Czas jaki ma być nastawiony na zegarze systemowym PLC
RET_VAL	OUT	INT	Kod warunkowy po wykonaniu instrukcji

WR_SYS_T (*Write System Time*) nastawia czas na zegarze PLC zgodnie z wartością parametru IN typu DTL. Ta wartość czasu nie uwzględnia lokalnej strefy czasowej ani terminów obowiązywania czasu letniego.

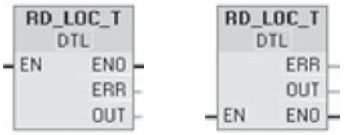
LAD **FBD**



Parametr	Typ parametru	Typ danych	Opis
RET_VAL	OUT	INT	Kod warunkowy po wykonaniu instrukcji
OUT	OUT	DTL	Bieżący czas systemowy PLC

RD_SYS_T (*Read System Time*) odczytuje bieżący czas systemowy z PLC. Ta wartość czasu nie uwzględnia lokalnej strefy czasowej ani terminów obowiązywania czasu letniego.

LAD **FBD**



Parametr	Typ parametru	Typ danych	Opis
RET_VAL	OUT	INT	Kod warunkowy po wykonaniu instrukcji
OUT	OUT	DTL	Czas lokalny

RD_LOC_T (*Read Local Time*) odczytuje bieżący czas lokalny PLC w formacie DTL.

- Czas lokalny jest obliczany na podstawie strefy czasowej oraz terminów obowiązywania czasu letniego wprowadzonych podczas konfigurowania zegara CPU.
- Konfiguracja strefy czasowej polega na ustaleniu przesunięcia czasu w stosunku do czasu uniwersalnego (UTC).
- Konfiguracja czasu letniego polega na wprowadzeniu miesiąca, tygodnia i dnia, od kiedy rozpoczyna się czas letni.
- Konfiguracja czasu standardowego również polega na wprowadzeniu miesiąca, tygodnia i dnia, od kiedy rozpoczyna się czas standardowy.
- Przesunięcie związane z czasem letnim zawsze jest odniesione do czasu systemowego. To przesunięcie stosuje się tylko w czasie obowiązywania czasu letniego.

Kody warunkowe

ENO = 1 oznacza, że nie wystąpił żaden błąd. ENO = 0 oznacza, że wystąpił błąd, a kod warunkowy jest określony przez parametr wyjściowy RET_VAL:

RET_VAL (W#16#....)	Opis
0000	Brak błędu
8080	Czas lokalny nie jest dostępny
8081	Niepoprawna wartość roku
8082	Niepoprawna wartość miesiąca

RET_VAL (W#16#....)	Opis
8083	Niepoprawna wartość dnia
8084	Niepoprawna wartość godziny
8085	Niepoprawna wartość minuty
8086	Niepoprawna wartość sekundy
8087	Niepoprawna wartość nanosekund
80B0	Uszkodzenie zegara czasu rzeczywistego

6.2.2 Instrukcje dotyczące znaków i łańcuchów

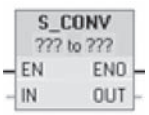
6.2.2.1 Instrukcje konwersji łańcuchów

Konwersja łańcucha na liczbę i liczby na łańcuch

Za pomocą następujących instrukcji można dokonać konwersji łańcucha na wartość liczbową lub wartości liczbowej na łańcuch:

- S_CONV dokonuje konwersji (łańcucha liczbowego na wartość liczbową) lub (wartości liczbowej na łańcuch liczbowy).
- STRG_VAL dokonuje konwersji łańcucha liczbowego na wartość liczbową z opcjami formatowania.
- VAL_STRG dokonuje konwersji wartości liczbowej na łańcuch liczbowy z opcjami formatowania.

LAD



FBD

Z rozwijanego menu należy wybrać typ danych parametrów.

S_CONV (*String Convert*) zamienia łańcuch znaków na odpowiadającą mu wartość liczbową lub wartość liczbową na odpowiadający jej łańcuch znaków. W instrukcji S_CONV nie ma opcji formatowania wyjścia. S_CONV jest przez to prostsza, ale mniej elastyczna niż instrukcje STRG_VAL i VAL_STRG.

S_CONV (konwersja łańcucha na liczbę)

Parametr	Typ parametru	Typ danych	Opis
IN	IN	STRING	Wejściowy łańcuch znaków
OUT	OUT	STRING, SINT, INT, DINT, USINT, UINT, UDINT, REAL	Wyjściowa wartość liczbowa

Konwersja łańcucha IN rozpoczyna się od pierwszego znaku i jest kontynuowana aż do osiągnięcia końca łańcucha lub napotkania pierwszego znaku, który nie jest cyfrą od „0” do „9”, „+”, „-” lub „.”. Wynik jest zapisywany w miejscu określonym

jako parametr OUT. Jeżeli wartość liczby wyjściowej nie leży w zakresie określonym przez typ danej OUT, to parametr OUT przyjmuje wartość 0 i ENO jest ustawiane na FALSE. W przeciwnym przypadku OUT zawiera prawidłowy wynik, a ENO przyjmuje wartość TRUE.

Zasady jakie spełnia format łańcucha wejściowego:

- Jeżeli w łańcuchu IN występuje punkt dziesiętny, to musi być użyty znak „.".
- Przecinki „," jako separatory tysięcy użyte po lewej stronie punktu dziesiętnego mogą być stosowane i są ignorowane.
- Wiodące spacje są ignorowane.
- Obsługiwana jest tylko reprezentacja stałoprzecinkowa. Znaki „e" i „E" nie są rozpoznawane jako symbole zapisu wykładniczego.

S_CONV (konwersja liczby na łańcuch)

Parametr	Typ parametru	Typ danych	Opis
IN	IN	STRING, SINT, INT, DINT, USINT, UINT, UDINT, REAL	Wejściowa wartość liczbowa
OUT	OUT	STRING	Wyjściowy łańcuch znaków

Liczba wejściowa IN całkowita, całkowita bez znaku lub zmiennoprzecinkowa jest zamieniana na odpowiadający jej ciąg znaków OUT. Zanim konwersja zostanie wykonana, parametr OUT musi zawierać wzór łańcucha. Wzór łańcucha składa się z maksymalnej długości łańcucha podanej w pierwszym bajcie, bieżącej długości łańcucha w drugim bajcie i bieżących znaków w kolejnych bajtach. Łańcuch powstały w wyniku konwersji zastępuje znaki łańcucha wzorcowego OUT począwszy od pierwszego znaku oraz uaktualnia wartość bieżącej długości łańcucha. Bajt zawierający maksymalną długość łańcucha nie jest zmieniany.

To, ile znaków jest zastąpionych zależy od typu danej parametru wejściowego IN i jego wartości. Liczba zastąpionych znaków musi się zmieścić w wyspecyfikowanej w parametrze OUT długości łańcucha. Maksymalna długość łańcucha określona w pierwszym bajcie łańcucha wzorcowego OUT powinna być większa lub równa spodziewanej liczbie konwertowanych znaków.

W poniższej tabeli przedstawiono maksymalne możliwe długości łańcuchów w przypadku różnych obsługiwanych typów danych.

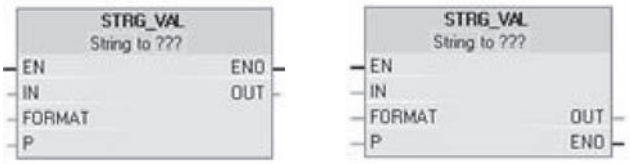
Typ danej IN	Maksymalna liczba przekonwertowanych znaków w łańcuchu OUT	Przykład	Całkowita długość łańcucha łącznie z bajtami określającymi długość maksymalną i bieżącą łańcucha
USINT	3	255	5
SINT	4	-128	6
UINT	5	65535	7
INT	6	-32768	8
UDINT	10	4294967295	12
DINT	11	-2147483648	13

Zasady jakie spełnia format łańcucha wyjściowego:

- Wartości wpisywane do parametru OUT nie mają wiodącego znaku +.
- Stosowana jest reprezentacja stałoprzecinkowa (bez zapisu wykładniczego).
- W przypadku gdy parametr IN jest typu REAL, znak kropki „.” pełni funkcję punktu dziesiętnego.

Instrukcja STRG_VAL

LAD **FBD**



Parametr	Typ parametru	Typ danych	Opis
IN	IN	STRING	Wejściowy łańcuch znaków ASCII do konwersji
FORMAT	IN	WORD	Opcje formatu wyjściowego
P	IN_OUT	UINT	IN: Indeks wskazujący znak, od którego należy rozpocząć konwersję (pierwszy znak = 1) OUT: Indeks wskazujący kolejny znak po zakończeniu procesu konwersji
OUT	OUT	SINT, INT, DINT, USINT, UINT, UDINT, REAL	Wartość liczbowa po konwersji

STRG_VAL (*String to Value*) przetwarza łańcuch znaków na odpowiadająca mu liczbę całkowitą lub zmiennoprzecinkową. Konwersja rozpoczyna się w łańcuchu IN od znaku określonego przez parametr P i jest kontynuowana aż do osiągnięcia końca łańcucha albo napotkania pierwszego znaku, który nie jest „+”, „-”, „.”, „,”, „e”, „E” lub cyfrą od „0” do „9”. Wynik jest zapisywany w miejscu określonym jako parametr OUT. Parametr P zwraca położenie znaku w oryginalnym łańcuchu, na którym konwersja się zakończyła. Przed rozpoczęciem wykonywania konwersji należy zainicjalizować poprawną daną typu STRING w pamięci.

Parametr FORMAT dla instrukcji STRG_VAL

Parametr FORMAT dla instrukcji STRG_VAL jest zdefiniowany poniżej. Niewykorzystane bity muszą być ustawione na 0.

Bit 16							Bit 8	Bit 7							Bit 0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	f	r

f = format notacji	1 = notacja wykładnicza 0 = notacja stałoprzecinkowa
r = format z punktem dziesiętnym	1 = „,” (przecinek) 0 = „.” (kropka)

FORMAT (W#16#)	Format notacji	Reprezentacja z punktem dziesiętnym
0000 (domyślnie)	Stałoprzecinkowa	„.”
0001		„,”
0002	Wykładnicza	„.”
0003		„,”
0004 do FFFF	Wartości niedozwolone	

Zasady konwersji za pomocą instrukcji STRG_VAL:

- Jeżeli jako punkt dziesiętny jest stosowany znak kropki „.”, to przecinki „,” po lewej stronie punktu dziesiętnego są traktowane jako separatory tysięcy. Te przecinki są dozwolone i ignorowane.
- Jeżeli jako punkt dziesiętny jest stosowany znak przecinka „,”, to kropki „.” po lewej stronie tego przecinka są traktowane jako separatory tysięcy. Te kropki są dozwolone i ignorowane.
- Wiodące spacje są ignorowane.

Instrukcja VAL_STRG

LAD

FBD

Parametr	Typ parametru	Typ danych	Opis
IN	IN	SINT, INT, DINT, USINT, UINT, UDINT, REAL	Wartość do konwersji
SIZE	IN	USINT	Liczba znaków do zapisania w łańcuchu OUT
PREC	IN	USINT	Precyzja lub rozmiar części ułamkowej. Nie jest tu uwzględniony punkt dziesiętny.
FORMAT	IN	WORD	Opcje formatu wyjściowego
P	IN_OUT	UINT	IN: Indeks wskazujący znak w łańcuchu OUT, od którego należy rozpocząć zamianę (pierwszy znak = 1) OUT: Indeks wskazujący kolejny znak w łańcuchu OUT po zakończeniu zamiany
OUT	OUT	STRING	Łańcuch przekonwertowany

VAL_STRG (*Value to String*) przetwarza liczbę całkowitą, liczbę całkowitą bez znaku lub liczbę zmiennoprzecinkową na odpowiadający jej ciąg znaków OUT. Wartość reprezentowana przez parametr IN jest przetwarzana na łańcuch zapisywany w miejscu określonym jako parametr OUT. Zanim konwersja zostanie wykonana, parametr OUT musi zawierać wzór łańcucha. Znaki łańcucha powstałego w wyniku konwersji zastępują znaki łańcucha wzorcowego OUT począwszy od znaku określonego przez P, a skończywszy po tylu znakach ile wynosi wartość zapisana w parametrze SIZE. Wartość wpisana do SIZE musi być dopasowana do długości łańcucha OUT z uwzględnieniem przesunięcia o P pierwszych znaków. Ta instrukcja jest przydatna do wstawiania liczby w łańcuch tekstowy. Na przykład można wstawić liczbę „120" do łańcucha „Pump pressure = 120 psi".

Parametr PREC określa precyzję lub liczbę cyfr części ułamkowej w łańcuchu. Jeżeli wartość parametru IN jest liczbą całkowitą, to PREC określa położenie punktu dziesiętnego. Na przykład, jeżeli wartość danej wynosi 123, a PREC = 1 to wynikiem jest „12.3". Maksymalna obsługiwana precyzja dla danych typu REAL wynosi 7 cyfr.

Jeśli parametr P jest większy od bieżącej długości łańcucha OUT, to aż do pozycji P są dodawane spacje, a wynik jest dołączany do końca łańcucha. Jeśli osiągnięta zostaje maksymalna długość łańcucha OUT, to konwersja jest kończona.

Parametr FORMAT instrukcji VAL_STRG

Parametr FORMAT dla instrukcji VAL_STRG jest zdefiniowany poniżej. Niewykorzystane bity muszą być ustawione na 0.

Bit 16							**Bit 8**	**Bit 7**							**Bit 0**
0	0	0	0	0	0	0	0	0	0	0	0	0	0	f	r

s = znak liczby
1 = stosuje się znaki „+" i „-"
0 = stosuje się tylko znak „-"

f = format notacji
1 = notacja wykładnicza
0 = notacja stałoprzecinkowa

r = format z punktem dziesiętnym
1 = „," (przecinek)
0 = „." (kropka)

FORMAT (WORD)	**Znak liczby**	**Format notacji**	**Reprezentacja z punktem dziesiętnym**
W#16#0000	Tylko „-"	Stałoprzecinkowa	„."
W#16#0001			","
W#16#0002		Wykładnicza	„."
W#16#0003			","
W#16#0004	„+" i „-"	Stałoprzecinkowa	„."
W#16#0005			","
W#16#0006		Wykładnicza	„."
W#16#0007			","
W#16#0008 do W#16#FFFF	Wartości niedozwolone		

Zasady jakie spełnia format łańcucha wyjściowego OUT:

- Kiedy przekonwertowany łańcuch jest krótszy niż wyspecyfikowany rozmiar, wtedy po skrajnej lewej stronie łańcucha są dostawiane wiodące spacje.
- Kiedy bit znaku parametru FORMAT ma wartość FALSE, wtedy liczby całkowite bez znaku i ze znakiem są wpisywane do bufora wyjściowego bez wiodącego znaku „+". Znak „-" jest używany jeśli jest to wymagane.

  <wiodące spacje><cyfry bez wiodących zer>'.'<cyfry PREC>
- Kiedy bit znaku parametru FORMAT ma wartość TRUE, wtedy liczby całkowite bez znaku i ze znakiem są zawsze wpisywane do bufora wyjściowego z wiodącym symbolem znaku.

  <wiodące spacje><znak>< cyfry bez wiodących zer>'.'< cyfry PREC >
- Kiedy FORMAT jest tak skonfigurowany, że obowiązuje notacja wykładnicza, wtedy liczby REAL są wpisywane do bufora wyjściowego jako: <wiodące spacje><znak>< cyfra>'.'< cyfry PREC >'E'<znak>< cyfry bez wiodących zer>
- Kiedy FORMAT jest tak skonfigurowany, że obowiązuje notacja stałoprzecinkowa, wtedy liczby całkowite, całkowite bez znaku i rzeczywiste są wpisywane do bufora wyjściowego jako:

  <wiodące spacje><znak>< cyfry bez wiodących zer>'.'< cyfry PREC >
- Wiodące zera z lewej strony punktu dziesiętnego (z wyjątkiem cyfry sąsiadującej z punktem dziesiętnym) są pomijane.
- Wartości z prawej strony punktu dziesiętnego są zaokrąglane tak, by liczba cyfr z prawej strony punktu dziesiętnego odpowiadała liczbie określanej przez parametr PREC.
- Rozmiar łańcucha wyjściowego musi wynosić co najmniej 3 bajty więcej niż liczba cyfr z prawej strony punktu dziesiętnego.
- Wartości w łańcuchu wyjściowym są wyrównywane do prawej.

Kody warunkowe sygnalizowane przez ENO

Kiedy podczas operacji konwersji wystąpi błąd, wtedy zwracane są następujące kody:

- ENO przyjmuje wartość 0.
- OUT przyjmuje wartość 0 lub taką jak w przykładach konwersji łańcucha na liczbę.
- OUT nie zmienia swojej wartości lub przyjmuje taką jak w przykładach, w których OUT jest łańcuchem.

Status ENO	Opis
1	Brak błędu
0	Nieprawidłowy lub nieważny parametr, np. wskazanie na DB, który nie istnieje
0	Nieprawidłowy łańcuch, w którym maksymalna długość łańcucha wynosi 0 lub 255
0	Nieprawidłowy łańcuch, w którym bieżąca długość jest większa niż maksymalna długość

Status ENO	Opis
0	Wartość liczby po konwersji jest zbyt duża dla wyspecyfikowanego typu danej OUT
0	Maksymalny rozmiar łańcucha parametru OUT musi być dostatecznie duży by pomieścić liczbę znaków określoną przez parametr SIZE i rozpoczynających się od pozycji określonej przez parametr P
0	Nieprawidłowa wartość P, gdzie P=0 lub P jest większa niż bieżąca długość łańcucha
0	Wartość parametru SIZE musi być większa od wartości parametru PREC

Przykłady konwersji łańcucha na liczbę za pomocą instrukcji S_CONV

Łańcuch IN	Typ danej OUT	Wartość OUT	ENO
„123”	INT/DINT	123	TRUE
„-00456”	INT/DINT	-456	TRUE
„123.45”	INT/DINT	123	TRUE
„+2345”	INT/DINT	2345	TRUE
„00123AB”	INT/DINT	123	TRUE
„123”	REAL	123.0	TRUE
„123.45”	REAL	123.45	TRUE
„1.23e-4”	REAL	1.23	TRUE
„1.23E-4”	REAL	1.23	TRUE
„12,345.67”	REAL	12345.67	TRUE
„3.4e39”	REAL	3.4	TRUE
„-3.4e39”	REAL	-3.4	TRUE
„1.17549e-38”	REAL	1.17549	TRUE
„12345”	SINT	0	FALSE
„A123”	N/A	0	FALSE
„”	N/A	0	FALSE
„++123”	N/A	0	FALSE
„+-123”	N/A	0	FALSE

Przykłady konwersji liczby na łańcuch za pomocą instrukcji S_CONV

Typ danej	Wartość IN	Łańcuch OUT	ENO
UINT	123	„123”	TRUE
UINT	0	„0”	TRUE
UDINT	12345678	„12345678”	TRUE
REAL	-INF	„INF”	FALSE
REAL	+INF	„INF”	FALSE
REAL	NaN	„NaN”	FALSE

Przykłady konwersji za pomocą instrukcji STRG_VAL

Łańcuch IN	FORMAT (W#16#....)	Typ danej OUT	Wartość OUT	ENO
„123”	0000	INT/DINT	123	TRUE
„-00456”	0000	INT/DINT	-456	TRUE
„123.45”	0000	INT/DINT	123	TRUE
„+2345”	0000	INT/DINT	2345	TRUE
„00123AB”	0000	INT/DINT	123	TRUE
„123”	0000	REAL	123.0	TRUE
„-00456”	0001	REAL	-456.0	TRUE
„+00456”	0001	REAL	456.0	TRUE
„123.45”	0000	REAL	123.45	TRUE
„123.45”	0001	REAL	12345.0	TRUE
„123,45”	0000	REAL	12345.0	TRUE
„123,45”	0001	REAL	123.45	TRUE
„.00123AB”	0001	REAL	123.0	TRUE
„1.23e-4”	0000	REAL	1.23	TRUE
„1.23E-4”	0000	REAL	1.23	TRUE
„1.23E-4”	0002	REAL	1.23E-4	TRUE
„12,345.67”	0000	REAL	12345.67	TRUE
„12,345.67”	0001	REAL	12.345	TRUE
„3.4e39”	0002	REAL	+INF	TRUE
„-3.4e39”	0002	REAL	-INF	TRUE
„1.1754943e-38” (i mniejsze)	0002	REAL	0.0	TRUE
„12345”	N/A	SINT	0	FALSE
„A123”	N/A	N/A	0	FALSE
„”	N/A	N/A	0	FALSE
„++123”	N/A	N/A	0	FALSE
„+-123”	N/A	N/A	0	FALSE

Przykłady konwersji za pomocą instrukcji VAL_STRG

W przykładach wykorzystano łańcuch OUT zainicjalizowany w następujący sposób:

„Current Temp = xxxxxxxxxx C”

Znak „x” reprezentuje znaki spacji alokowane dla wartości po konwersji.

Typ danej	Wartość IN	P	SIZE	FORMAT (W#16#....)	PREC	Łańcuch OUT	ENO
UINT	123	16	10	0000	0	Current Temp = xxxxxxx123 C	TRUE
UINT	0	16	10	0000	2	Current Temp = Xxxxxx0.00 C	TRUE
UDINT	12345678	16	10	0000	3	Current Temp = X12345.678 C	TRUE
UDINT	12345678	16	10	0001	3	Current Temp = X12345,678 C	TRUE
INT	123	16	10	0004	0	Current Temp = Xxxxxx+123 C	TRUE
INT	-123	16	10	0004	0	Current Temp = xxxxxx-123 C	TRUE
REAL	-0.00123	16	10	0004	4	Current Temp = xxx-0.0012 C	TRUE
REAL	-0.00123	16	10	0006	4	Current Temp = -1.2300E-3 C	TRUE
REAL	-INF	16	10	N/A	4	Current Temp = xxxxxx-INF C	FALSE
REAL	+INF	16	10	N/A	4	Current Temp = xxxxxx+INF C	FALSE
REAL	NaN	16	10	N/A	4	Current Temp = xxxxxxxNaN C	FALSE
UDINT	12345678	16	6	N/A	3	Current Temp = xxxxxxxxxx C	FALSE

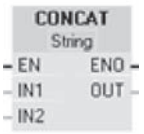
6.2.2.2 Instrukcje operacji na łańcuchach

W celu tworzenia wiadomości wyświetlanych na panelach operatorskich lub zapisywanych w dzienniku zdarzeń (log), w programie sterującym użytkownik może wykorzystywać następujące instrukcje operujące na łańcuchach i znakach:

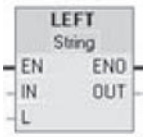
LAD

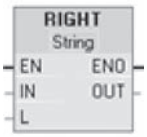
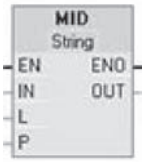
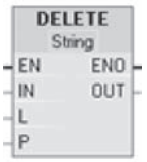
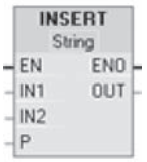
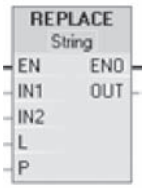
LEN — Pobiera długość łańcucha

CONCAT — Łączy dwa łańcuchy



LEFT — Pobiera lewy podłańcuch z łańcucha



RIGHT	Pobiera prawy podłańcuch z łańcucha	
MID	Pobiera środkowy podłańcuch z łańcucha	
DELETE	Usuwa podłańcuch z łańcucha	
INSERT	Wstawia podłańcuch do łańcucha	
REPLACE	Zastępuje podłańcuch w łańcuchu	
FIND	Znajduje podłańcuch lub znak w łańcuchu	

Dana typu STRING

Dane typu STRING są przechowywane jako 2-bajtowy nagłówek, po którym następuje do 254 bajtów znaków kodu ASCII. Pierwszy bajt oznacza maksymalną długość łańcucha, która jest podawana w nawiasie kwadratowym podczas inicjalizacji łańcucha lub domyślnie wynosi 254. Drugi bajt nagłówka jest to bieżąca długość czyli liczba prawidłowych znaków w łańcuchu. Bieżąca długość musi być mniejsza lub równa długości maksymalnej.

Inicjalizacja danej typu STRING

Przed użyciem dowolnej instrukcji dotyczącej łańcuchów, wejściowe i wyjściowe dane typu STRING muszą być zainicjalizowane w pamięci jako prawidłowe łańcuchy.

LEN

Parametr	Typ parametru	Typ danych	Opis
IN	IN	STRING	Łańcuch wejściowy
OUT	OUT	UINT	Liczba prawidłowych znaków łańcucha IN

LEN (*Length of string*) zwraca bieżącą długość łańcucha IN na wyjściu OUT. Pusty łańcuch ma długość zerową.

CONCAT

Parametr	Typ parametru	Typ danych	Opis
IN1	IN	STRING	Łańcuch wejściowy 1
IN2	IN	STRING	Łańcuch wejściowy 2
OUT	OUT	STRING	Łańcuch połączony (łańcuch 1 + łańcuch 2)

CONCAT (*Concatenate strings*) łączy parametry IN 1 i IN 2 typu STRING w celu uformowania jednego łańcucha wyjściowego OUT. Po połączeniu, IN1 jest lewą częścią, a IN 2 prawą częścią łańcucha połączonego. Jeżeli połączony łańcuch jest dłuższy od dopuszczalnej długości maksymalnej, to wynikowy łańcuch jest ograniczany do długości maksymalnej i ENO jest ustawiany na 0.

LEFT

Parametr	Typ parametru	Typ danych	Opis
IN	IN	STRING	Łańcuch wejściowy
L	IN	INT	Długość podłańcucha jaki ma być utworzony z wykorzystaniem L skrajnych lewych znaków łańcucha IN
OUT	OUT	STRING	Łańcuch wyjściowy

LEFT (*Left substring*) zwraca podłańcuch utworzony z pierwszych L znaków parametru IN typu STRING.

- Jeżeli L jest większy od bieżącej długości łańcucha IN, to jako OUT jest zwracany cały łańcuch IN.
- Jeżeli wejściowy łańcuch jest pusty, to jako OUT jest zwracany pusty łańcuch.
- Jeżeli L jest ujemny lub zero, to zwracany jest pusty łańcuch i ENO jest ustawiany na 0.

RIGHT

Parametr	Typ parametru	Typ danych	Opis
IN	IN	STRING	Łańcuch wejściowy
L	IN	INT	Długość podłańcucha jaki ma być utworzony z wykorzystaniem L skrajnych prawych znaków łańcucha IN
OUT	OUT	STRING	Łańcuch wyjściowy

RIGHT (*Right substring*) zwraca L ostatnich znaków łańcucha wejściowego.

- Jeżeli L jest większy od bieżącej długości łańcucha IN, to jako OUT jest zwracany cały łańcuch IN.
- Jeżeli wejściowy łańcuch jest pusty, to jako OUT jest zwracany pusty łańcuch.
- Jeżeli L jest ujemny lub zero, to zwracany jest pusty łańcuch i ENO jest ustawiany na 0.

MID

Parametr	Typ parametru	Typ danych	Opis
IN	IN	STRING	Łańcuch wejściowy
L	IN	INT	Długość podłańcucha jaki ma być utworzony z wykorzystaniem L znaków łańcucha IN, począwszy od znaku na pozycji P
P	IN	INT	Pozycja pierwszego znaku do skopiowania do podłańcucha: P = 1 oznacza początkową pozycję w łańcuchu IN
OUT	OUT	STRING	Łańcuch wyjściowy

MID (*Middle substring*) zwraca środkową część łańcucha wejściowego. Środkowy podłańcuch ma długość L znaków i rozpoczyna się od znaku na pozycji P (włącznie).

- Jeżeli suma L i P jest większa od bieżącej długości łańcucha IN, to jako OUT jest zwracany podłańcuch, który zaczyna się od znaku na pozycji P i kończy na ostatnim znaku łańcucha IN.
- Jeżeli pozycja P wykracza poza bieżącą długość łańcucha IN, to jako OUT jest zwracany pusty łańcuch i ENO jest ustawiany na 0.
- Jeżeli L lub P jest ujemny lub zero, to jako OUT jest zwracany pusty łańcuch i ENO jest ustawiany na 0.

DELETE

Parametr	Typ parametru	Typ danych	Opis
IN	IN	STRING	Łańcuch wejściowy
L	IN	INT	Liczba znaków do usunięcia
P	IN	INT	Pozycja pierwszego znaku do usunięcia: pierwszy znak w łańcuchu IN znajduje się na pozycji numer 1.
OUT	OUT	STRING	Łańcuch wyjściowy

DELETE (*Delete substring*) usuwa L znaków z łańcucha IN. Usuwanie znaków zaczyna się od znaku na pozycji P (włącznie) i pozostały podłańcuch jest zwracany jako parametr OUT.

- Jeżeli L jest równy zero, to jako OUT jest zwracany cały łańcuch IN i ENO = = TRUE.
- Jeżeli P jest większy od bieżącej długości łańcucha IN, to jako OUT jest zwracany cały łańcuch IN i ENO = FALSE.
- Jeżeli suma L i P jest większa od długości łańcucha wejściowego, to znaki są usuwane do końca łańcucha.
- Jeżeli L jest ujemny, a P jest ujemny lub równy 0, to jako OUT jest zwracany pusty łańcuch i ENO = FALSE.
- Przed wykonaniem instrukcji DELETE, dane muszą być zainicjalizowane w pamięci jako prawidłowe łańcuchy STRING.

INSERT

Parametr	Typ parametru	Typ danych	Opis
IN1	IN	STRING	Łańcuch wejściowy 1
IN2	IN	STRING	Łańcuch wejściowy 2
P	IN	INT	Pozycja ostatniego znaku w łańcuchu IN1, przed miejscem wstawienia łańcucha IN2. Pozycja pierwszego znaku w łańcuchu IN1 ma numer 1.
OUT	OUT	STRING	Łańcuch wyjściowy

INSERT (*Insert substring*) wstawia łańcuch IN2 do łańcucha IN1. Miejsce wstawienia znajduje za pozycją znaku P.

- Jeżeli P jest większy od bieżącej długości łańcucha IN1, to IN2 jest dołączany do IN1 i ENO = FALSE.
- Jeżeli P jest ujemny lub równy 0, to jako OUT jest zwracany pusty łańcuch i ENO = FALSE.
- Jeżeli długość nowego łańcucha po operacji INSERT jest dłuższa od dozwolonej maksymalnej długości łańcucha OUT, to wynikowy łańcuch jest ograniczany do długości maksymalnej parametru OUT i ENO = FALSE.

REPLACE

Parametr	Typ parametru	Typ danych	Opis
IN1	IN	STRING	Łańcuch wejściowy 1
IN2	IN	STRING	Łańcuch ze znakami zastępującymi
L	IN	INT	Liczba znaków do zamiany
P	IN	INT	Pozycja pierwszego znaku do zamiany
OUT	OUT	STRING	Łańcuch wyjściowy

REPLACE (*Replace substring*) zamienia L znaków w łańcuchu IN1. Zamiana rozpoczyna się od znaku na pozycji P (włącznie) w łańcuchu IN1, przy znaki zastępujące pochodzą z łańcucha IN2.

- Jeżeli L jest równy zero, to łańcuch IN2 jest wstawiany na pozycję P łańcucha IN1 bez usuwania jakiegokolwiek znaku z łańcucha IN1.
- Jeżeli P jest równy jedności, to pierwszych L znaków łańcucha IN1 jest zastąpionych znakami łańcucha IN2.
- Jeżeli P jest większy od długości łańcucha IN1, to łańcuch IN2 jest dołączany do łańcucha IN1 i ENO = FALSE.
- Jeżeli L jest ujemny, a P jest ujemny lub równy zeru, to jako OUT jest zwracany pusty łańcuch i ENO = FALSE.
- Jeżeli długość nowego łańcucha po operacji REPLACE jest dłuższa od dozwolonej maksymalnej długości łańcucha OUT, to wynikowy łańcuch jest ograniczany do długości maksymalnej parametru OUT i ENO = FALSE.

FIND

Parametr	Typ parametru	Typ danych	Opis
IN1	IN	STRING	Ten łańcuch jest przeszukiwany
IN2	IN	STRING	Ten łańcuch jest poszukiwany
OUT	OUT	INT	Pozycja znaku w łańcuchu IN1 odpowiadająca pierwszej zgodności z poszukiwanym wzorem

FIND (*Find substring*) zwraca położenie w łańcuchu IN1 znaku podciągu lub pojedynczego znaku wyspecyfikowanego w IN2. Poszukiwanie rozpoczyna się od lewej. W OUT jest zwracane położenie znaku pierwszego wystąpienia łańcucha IN2. Jeżeli łańcuch IN2 nie jest odnaleziony w IN1, to zwracane jest zero.

Kody warunkowe operacji na łańcuchach sygnalizowane przez ENO

LEN

ENO	Warunek	OUT
1	Zawsze TRUE, brak możliwości błędu	Prawidłowa długość łańcucha

CONCAT

ENO	Warunek	OUT
1	Nie wykryto błędów	Prawidłowe znaki
0	Bieżąca długość IN1 przekracza maksymalną długość IN1 lub bieżąca długość IN2 przekracza maksymalną długość IN2 (nieprawidłowy łańcuch)	Bieżąca długość jest ustawiana na 0
	Maksymalna długość IN1, IN2 lub OUT nie pasuje do alokowanego zakresu pamięci	
	Maksymalna długość IN1, IN2 lub OUT wynosi 0 lub 255 (niedozwolona długość)	
	Łańcuch wynikowy po połączeniu jest dłuższy niż maksymalna długość łańcucha OUT	Znaki łańcucha wynikowego są kopiowane aż do osiągnięcia maksymalnej długości łańcucha OUT

LEFT

ENO	Warunek	OUT
1	Nie wykryto błędów	Prawidłowe znaki
0	Bieżąca długość IN przekracza maksymalną długość IN (nieprawidłowy łańcuch)	Bieżąca długość jest ustawiana na 0
	Maksymalna długość IN lub OUT nie pasuje do alokowanego zakresu pamięci	
	L jest mniejszy lub równy 0	
	Maksymalna długość IN lub OUT wynosi 0 lub 255 (niedozwolona długość)	
	Długość podłańcucha (L) do skopiowania jest większa niż maksymalna długość łańcucha OUT	Znaki są kopiowane aż do osiągnięcia maksymalnej długości łańcucha OUT

RIGHT

ENO	Warunek	OUT
1	Nie wykryto błędów	Prawidłowe znaki
0	Bieżąca długość IN przekracza maksymalną długość IN (nieprawidłowy łańcuch)	Bieżąca długość jest ustawiana na 0
	Maksymalna długość IN lub OUT nie pasuje do alokowanego zakresu pamięci	
	L jest mniejszy lub równy 0	
	Maksymalna długość IN lub OUT wynosi 0 lub 255 (niedozwolona długość)	
	Długość podłańcucha (L) do skopiowania jest większa niż maksymalna długość łańcucha OUT	Znaki są kopiowane aż do osiągnięcia maksymalnej długości łańcucha OUT

MID

ENO	Warunek	OUT
1	Nie wykryto błędów	Prawidłowe znaki
0	Bieżąca długość IN przekracza maksymalną długość IN (nieprawidłowy łańcuch)	Bieżąca długość jest ustawiana na 0
	Maksymalna długość IN lub OUT nie pasuje do alokowanego zakresu pamięci	
	L lub P jest mniejszy lub równy 0	
	P jest większy niż maksymalna długość IN	
	Maksymalna długość IN lub OUT wynosi 0 lub 255 (niedozwolona długość)	
	Długość podłańcucha (L) do skopiowania jest większa niż maksymalna długość łańcucha OUT	Znaki są kopiowane począwszy od pozycji P aż do osiągnięcia maksymalnej długości łańcucha OUT

DELETE

ENO	Warunek	OUT
1	Nie wykryto błędów	Prawidłowe znaki
0	P jest większy niż bieżąca długość IN	IN jest kopiowany do OUT bez usuwania żadnych znaków
	Bieżąca długość IN przekracza maksymalną długość IN (nieprawidłowy łańcuch)	Bieżąca długość jest ustawiana na 0
	Maksymalna długość IN lub OUT nie pasuje do alokowanego zakresu pamięci	
	L jest mniejszy od 0 lub P jest mniejszy równy 0	
	Maksymalna długość IN lub OUT wynosi 0 lub 255 (niedozwolona długość)	
	Wynikowy łańcuch po usunięciu znaków jest dłuższy niż maksymalna długość łańcucha OUT	Znaki wynikowego łańcucha są kopiowane aż do osiągnięcia maksymalnej długości łańcucha OUT

INSERT

ENO	Warunek	OUT
1	Nie wykryto błędów	Prawidłowe znaki
0	P jest większy niż długość IN1	IN2 jest dołączany do IN1 zaraz po ostatnim znaku IN1
	P jest mniejszy lub równy 0	Bieżąca długość jest ustawiana na 0
	Bieżąca długość IN1 przekracza maksymalną długość IN1 lub bieżąca długość IN2 przekracza maksymalną długość IN2 (nieprawidłowy łańcuch)	
	Maksymalna długość IN1, IN2 lub OUT nie pasuje do alokowanego zakresu pamięci	
	Maksymalna długość IN1, IN2 lub OUT wynosi 0 lub 255 (niedozwolona długość)	
	Wynikowy łańcuch po wstawieniu znaków jest dłuższy niż maksymalna długość łańcucha OUT	Znaki wynikowego łańcucha są kopiowane aż do osiągnięcia maksymalnej długości łańcucha OUT

REPLACE

ENO	Warunek	OUT
1	Nie wykryto błędów	Prawidłowe znaki
0	P jest większy niż długość IN1	IN2 jest dołączany do IN1 zaraz po ostatnim znaku IN1
	P wskazuje pozycję w IN1, ale w IN1 pozostaje mniej niż L znaków	IN2 zastępuje końcowe znaki IN1 począwszy od pozycji P
	L jest mniejszy od 0 lub P jest mniejszy lub równy 0	Bieżąca długość jest ustawiana na 0
	Bieżąca długość IN1 przekracza maksymalną długość IN1 lub bieżąca długość IN2 przekracza maksymalną długość IN2 (nieprawidłowy łańcuch)	
	Maksymalna długość IN1, IN2 lub OUT nie pasuje do alokowanego zakresu pamięci	
	Maksymalna długość IN1, IN2 lub OUT wynosi 0 lub 255 (niedozwolona długość)	
	Wynikowy łańcuch po zamianie znaków jest dłuższy niż maksymalna długość łańcucha OUT	Znaki wynikowego łańcucha są kopiowane aż do osiągnięcia maksymalnej długości łańcucha OUT

FIND

ENO	Warunek	OUT
1	Nie wykryto błędów	Prawidłowa pozycja znaku
0	Bieżąca długość IN1 przekracza maksymalną długość IN1 lub bieżąca długość IN2 przekracza maksymalną długość IN2 (nieprawidłowy łańcuch)	Pozycja znaku jest ustawiana na 0
	Maksymalna długość IN1, IN2 lub OUT nie pasuje do alokowanego zakresu pamięci	
	IN2 jest większy niż IN1	
	Maksymalna długość IN1 lub IN2 wynosi 0 lub 255 (niedozwolona długość)	

6.2.3 Instrukcje sterujące wykonywaniem programu

6.2.3.1 Instrukcja kasowania timera nadzorującego pracę CPU (watchdoga)

RE_TRIGR (*Re-trigger scan time watchdog*) jest stosowana do wydłużania maksymalnego dopuszczalnego czasu zanim timer układu dozorującego poprawność wykonywania cyklu programu wygeneruje błąd.

LAD/FBD

Instrukcja RE_TRIGR służy do ponownego wyzwolenia timera nadzorującego poprawność wykonywania cyklu programu podczas pojedynczego cyklu. Dzięki temu, od czasu ostatniego wykonania funkcji RE_TRIGR dopuszczalny maksymalny czas cyklu programu wydłuża się o jeden (najdłuższy) okres cyklu programu.

CPU systemu S7-1200 ogranicza użycie instrukcji RE_TRIGR do cyklu programu, przykładowo do OB1 i funkcji wywoływanych z cyklu programu. Oznacza, że jeżeli RE_TRIGR jest wywołana z dowolnego OB z listy OB cyklu programu, to timer układu nadzorującego jest kasowany i ENO = EN.

Jeżeli RE_TRIGR jest wykonana z rozruchowego OB, OB przerwań lub OB obsługi błędów, to ENO = FALSE i timer układu nadzorującego nie jest kasowany.

Ustawianie maksymalnego czasu cyklu programu PLC

Użytkownik może ustawić maksymalny czas cyklu programu podczas konfiguracji urządzenia PLC w „Cycle time”.

Monitor czasu cyklu	Minimalna wartość	Maksymalna wartość	Domyślna wartość
Maksymalny czas cyklu	1 ms	6000 ms	150 ms

Limit czasu układu dozorującego

Jeżeli przed zakończeniem cyklu programu upłynie maksymalny czas cyklu programu, to zostanie wygenerowany błąd. Jeżeli w programie użytkownika jest umieszczony blok kodu obsługi błędu OB80, to PLC wykonuje OB80, a w nim można dodać odpowiedni program reakcji na tę sytuację. Kiedy w programie nie ma OB80, wtedy pierwsze przekroczenie limitu czasu jest ignorowane.

Gdy podczas tego samego cyklu programu ponownie zostanie przekroczony maksymalny czas cyklu programu (czyli łącznie dwa maksymalnie długie cykle), to generowany jest błąd powodujący przejście PLC do trybu STOP.

W trybie STOP wykonywanie programu użytkownika jest wstrzymywane, podczas gdy komunikacja i diagnostyka systemowa PLC nadal działają.

6.2.3.2 Instrukcja zatrzymywania cyklu programu

LAD

STP (*Stop PLC scan cycle*) wprowadza PLC w tryb STOP. Kiedy PLC jest w trybie STOP, wtedy wykonywanie programu użytkownika i uaktualnianie adresu wyjściowego Q z obrazu procesu są wstrzymane.

Wyjściowe stany bezpieczne, jakie pojawiają się podczas przechodzenia systemu w tryb STOP na wyjściach zintegrowanych oraz wyjściach płytki sygnałowej i modułów rozszerzeń (analogowych i cyfrowych), są definiowane podczas konfiguracji urządzenia PLC w zakładce „Properties". Użytkownik może wybrać „zamrożenie" ostatniego stanu wyjść lub ustalić (analogowe i cyfrowe) stany bezpieczne. Wartością domyślną dla wyjść cyfrowych jest stan FALSE, a wartością domyślną dla wyjść analogowych jest poziom 0.

Jeżeli EN = TRUE, to PLC przejdzie do stanu STOP, wykonywanie programu zostanie zatrzymane, a stan ENO nie będzie miał znaczenia. W przeciwnym przypadku EN = ENO = 0.

6.2.3.3 Instrukcje pobierania błędu

Instrukcje pobierania błędu (*get error*) dostarczają informacji o błędach związanych z wykonywaniem bloków programu. Przed użyciem instrukcji GET_ERROR lub GET_ERR_ID bloki programu muszą sprawdzić atrybut „handle errors within block:" w konfiguracji właściwości bloku.

- GET_ERROR sygnalizuje, że wystąpił błąd wykonania bloku programu i wypełnia predefiniowaną strukturę danych błędu szczegółowymi informacjami o błędzie.
- GET_ERROR_ID sygnalizuje, że wystąpił błąd wykonania bloku programu i zgłasza ID (kod identyfikacyjny) błędu.

GET_ERROR

LAD **FBD**

Parametr	Typ danych	Opis
ERROR	ErrorStruct	Struktura danych błędu

Struktura danych parametru ERROR

Użytkownik może zmienić nazwę struktury, ale nie nazwy występujące wewnątrz struktury.

Element danej typu ErrorStruct	Typ danych	Opis
ERROR_ID	WORD	Identyfikator błędu
FLAGS	BYTE	Sygnalizuje, czy błąd wystąpił podczas wywołania do innego bloku: • 16#01 jeśli błąd wystąpił podczas wywołania • 16#00 w przeciwnym przypadku
REACTION	BYTE	Reakcja na błąd: • 0 = zignorować; nic nie jest zapisane (błąd zapisu) • 1 = zastąpić: 0) użyte jako wartość (błąd odczytu) • 2 = ominąć instrukcję (błąd systemowy)
BLOCK_TYPE	BYTE	Typ bloku, w którym wystąpił błąd: • 1 = OB • 2 = FC • 3 = FB
PAD_0	BYTE	Wewnętrznie wypełniany bajt dla uzyskania zgodności, wynosi 0
CODE_BLOCK_NUMBER	UINT	Numer bloku, w którym wystąpił błąd
ADDRESS	UDINT	Lokalizacja w wewnętrznej pamięci instrukcji, przy której wystąpił błąd

Element danej typu ErrorStruct	**Typ danych**	**Opis**					
MODE	BYTE	Wewnętrzne mapowanie dotyczące sposobu interpretowania pozostałych pól					
		Tryb	(A)	(B)	(C)	(D)	(E)
		0					
		1					Offset
		2			Area		
		3	Location	Scope			
		4			Area		Offset
		5			Area	DB no.	Offset
		6	PtrNo./Acc		Area	DB no.	Offset
		7	PtrNo./Acc	Slot No./ Scope	Area	DB no.	Offset
PAD_1	BYTE	Wewnętrznie wypełniany bajt dla uzyskania zgodności, nie wykorzystywany, wynosi 0					
OPERAND_ NUMBER	UINT	Numer argumentu wewnętrznej instrukcji					
POINTER_ NUMBER_ LOCATION	UINT	(A) Położenie wskaźnika wewnętrznej instrukcji					
SLOT_ NUMBER_ SCOPE	UINT	(B) Położenie obszaru w wewnętrznej pamięci					
AREA	BYTE	(C) Odniesienie do obszarów pamięci podczas wykrycia błędu: • L: 16#40 – 4E, 86, 87, 8E, 8F, C0 – CE • I: 16#81 • Q: 16#82 • M: 16#83 • DB: 16#84, 85, 8A, 8B					
PAD_2	BYTE	Wewnętrznie wypełniany bajt dla uzyskania zgodności, nie wykorzystywany, wynosi 0					
DB_NUMBER	UINT	(D) DB wskazany podczas wykrycia błędu DB, w przeciwnym przypadku 0					
OFFSET	UDINT	(E) Położenie bitu wskazanego podczas wykrycia błędu (przykład: 12 = bajt 1, bit 4)					

GET_ERR_ID

LAD **FBD**

Parametr	Typ danych	Opis
ID	WORD	Identyfikator błędu

Parametr ID: Wartości identyfikatora błędu jako elementu ErrorStruct ERROR_ID

ERROR_ID szesnastkowo	ERROR_ID dziesiętnie	Błąd wykonania bloku programu
2503	9475	Błąd niezainicjalizowania wskaźnika
2522	9506	Błąd odczytu - argument spoza zakresu
2523	9507	Błąd zapisu - argument spoza zakresu
2524	9508	Błąd odczytu - nieprawidłowy argument
2525	9509	Błąd zapisu - nieprawidłowy argument
2528	9512	Błąd odczytu – zgodność danych
2529	9513	Błąd zapisu – zgodność danych
2530	9520	Błąd zapisu DB
253A	9530	Globalny DB nie istnieje
253C	9532	Błędna wersja lub FC nie istnieje
253D	9533	SFC nie istnieje
253E	9534	Błędna wersja lub FB nie istnieje
253F	9535	SFB nie istnieje
2575	9589	Błąd głębokości zagnieżdżenia programu
2576	9590	Błąd alokacji danych lokalnych
2942	10562	Błąd bezpośredniego odczytu wejść
2943	10563	Błąd bezpośredniego zapisu do wyjść

Działanie

Domyślnie, PLC odpowiada na wystąpienie błędu wykonania bloku zarejestrowaniem błędu w buforze diagnostycznym i przejściem w tryb STOP. Jednakże jeżeli użytkownik umieści jedną lub więcej instrukcji GET_ERROR lub GET_ERR_ID w kodzie bloku, to tym samym blok jest skonfigurowany do obsługi błędów w ramach tego bloku. W takim przypadku PLC nie przechodzi do trybu STOP i nie rejestruje błędu w buforze diagnostycznym. Zamiast tego, informacje o błędzie są przedstawiane na wyjściu instrukcji GET_ERROR lub GET_ERR_ID. Użytkownik może uzyskać szczegółowe informacje o błędzie za pomocą instrukcji GET_ERROR lub odczytać identyfikator błędu za pomocą instrukcji GET_ERR_ID. Zwykle pierwszy błąd jest najważniejszy – kolejne błędy są tylko konsekwencją tego pierwszego.

Pierwsze wykonanie instrukcji GET_ERROR lub GET_ERR_ID w ramach bloku, zwraca pierwszy wykryty błąd jaki powstał podczas wykonywania bloku. Ten błąd mógł wystąpić gdziekolwiek między startem bloku i wykonaniem dowolnej z instrukcji GET_ERROR albo GET_ERR_ID. Kolejne wykonania instrukcji GET_ERROR lub GET_ERR_ID zwracają pierwszy błąd od czasu poprzedniego wykona-

nia instrukcji GET_ERROR lub GET_ERR_ID. Historia błędów nie jest zachowywana i wykonanie dowolnej z tych instrukcji uzbraja system PLC do wyłapywania następnego błędu.

Dana typu *ErrorStruct* wykorzystywana przez instrukcję GET_ERROR może być dodana w edytorze bloku danych i edytorach bloku interfejsu, tak że program użytkownika może mieć dostęp do zawartych w niej informacji. W celu dodania tej struktury należy z rozwijanej listy wybrać *ErrorStruct*. Stosując unikalne nazwy, użytkownik może stworzyć wiele *ErrorStruct*. Nazwy elementów wewnętrznych *ErrorStruct* nie mogą być zmieniane.

Błędy wskazywane przez ENO

Jeżeli EN = TRUE i zostaje wykonana instrukcja GET_ERROR lub GET_ERROR_ID, to

- ENO = TRUE sygnalizuje, że wystąpił błąd wykonania bloku i informacje o błędzie są dostępne.
- ENO = FALSE sygnalizuje, że nie wystąpił żaden błąd wykonania bloku.

Użytkownik może powiązać program reakcji na błąd z ENO, które jest aktywowane po wystawieniu błędu. Jeżeli błąd istnieje, to parametry wyjściowe przechowują informacje o błędzie i program użytkownika ma do nich dostęp.

GET_ERROR i GET_ERROR_ID mogą być wykorzystane do przesłania informacji o błędzie z aktualnie wykonywanego bloku (zwanego blokiem) do bloku wywołującego. W celu uzyskania ostatecznego statusu wykonania bloku wywołującego, instrukcje należy umieścić w ostatnim obwodzie bloku programu wywołującego.

6.2.4 Instrukcje komunikacji

6.2.4.1 Otwarcie komunikacji przez Ethernet

Otwarcie komunikacji przez Ethernet za pomocą instrukcji automatycznego połączenia/rozłączenia (TSEND_C i TRCV_C)

Opis TSEND_C

TSEND_C ustala ze stacją partnerską połączenie komunikacyjne TCP lub ISO on TCP, wysyła dane i może zakończyć połączenie. Połączenie po skonfigurowaniu i ustaleniu jest automatycznie utrzymywane i monitorowane przez CPU. TSEND_C łączy w sobie funkcje TCON, TDISCON i TSEND.

Funkcja TSEND_C

- W celu ustanowienia połączenia należy wykonać TSEND_C z CONT = 1.
- Po pomyślnym ustanowieniu połączenia, TSEND_C ustawia parametr DONE na jeden cykl.
- W celu zakończenia połączenia komunikacyjnego należy wykonać TSEND_C z CONT = 0. Komunikacja zostanie natychmiast przerwana. Ma to również wpływ na stację odbiorczą. Połączenie zostanie tam zakończone i dane z bufora odbiorczego mogą zostać utracone.

- W celu wysłania danych ustanowionym kanałem połączeniowym należy wykonać TSEND_C z narastającym zboczem na REQ. Po pomyślnym wykonaniu operacji wysłania danych, TSEND_C ustawia parametr DONE na jeden cykl.
- W celu ustanowienia połączenia i wysłania danych należy wykonać TSEND_C z CONT = 1 i REQ = 1. Po pomyślnym wykonaniu operacji wysłania danych, TSEND_C ustawia parametr DONE na jeden cykl.

Opis TRCV_C

TRCV_C ustala ze stacją partnerską połączenie komunikacyjne TCP lub ISO on TCP, odbiera dane i może zakończyć połączenie. Połączenie po skonfigurowaniu i ustaleniu jest automatycznie utrzymywane i monitorowane przez CPU. Instrukcja TSEND_C łączy w sobie funkcje TCON, TDISCON i TRCV.

Funkcja TRCV_C

1. Ustanowienie połączenia: należy wykonać TRCV_C z parametrem CONT = 1.
2. Odbiór danych: należy wykonać TRCV_C z parametrem EN_R = 1. Dane można odbierać w sposób ciągły gdy EN_R = 1 i CONT = 1.
3. Zakończenie połączenia: należy wykonać TRCV_C z CONT = 0. Komunikacja zostanie natychmiast przerwana i dane mogą zostać utracone.

Tryby odbiorcze

TRCV_C obsługuje te same tryby odbiorcze co instrukcja TRCV. W następującej tabeli znajdują się informacje jak dane są wprowadzane do obszaru odbiorczego.

Wariant protokołu	Wprowadzanie danych do obszaru odbiorczego	Parametr „connection_type”	Wartość parametru LEN
TCP	Tryb „Ad hoc”	B#16#11	0
TCP	Odbiór danych o określonej długości	B#16#11	<> 0
ISO on TCP	Kontrolowane przez protokół	B#16#12	0 (rekomendowana) lub <> 0

Tryb TCP/ad hoc

Tryb ad hoc istnieje tylko w wariancie protokołu TCP. Użytkownik może ustawić tryb ad hoc przypisując parametrowi LEN wartość 0.

Obszar odbiorczy jest identyczny z obszarem uformowanym przez DATA. Maksymalnie są odbierane 1472 bajty.

TCP/odbiór danych o określonej długości

Użytkownik może ustawić tryb odbioru danych o określonej długości przypisując parametrowi LEN wartość inną niż 0.

Obszar odbiorczy jest definiowany przez parametry LEN i DATA.

ISO on TCP / przepływ danych kontrolowany protokołem

W wariancie protokołu ISO on TCP, przesyłane dane są kontrolowane przez protokół.

Obszar odbiorczy jest definiowany przez parametry LEN i DATA.

UWAGA

Ze względu na asynchroniczne przetwarzanie TSEND_C, użytkownik musi utrzymywać w obszarze nadawczym spójne dane aż do chwili gdy parametr DONE lub parametr ERROR przyjmie wartość TRUE.

W przypadku TSEND_C, wartość TRUE parametru DONE oznacza, że dane zostały pomyślnie wysłane. Nie oznacza to natomiast, że połączona stacja partnerska CPU odczytała bufor odbiorczy.

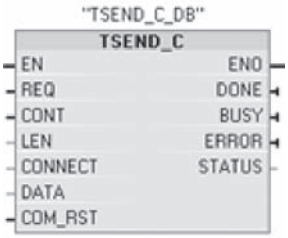
Ze względu na asynchroniczne przetwarzanie TRCV_C, dane w obszarze odbiorczym są spójne tylko wtedy, kiedy DONE = 1.

W następującej tabeli przedstawiono związki między parametrami BUSY, DONE i ERROR.

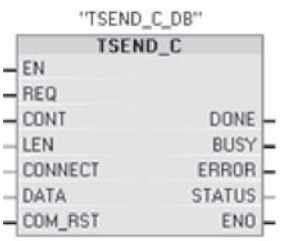
BUSY	DONE	ERROR	Opis
TRUE	nieistotny	nieistotny	Zadanie jest w toku wykonywania.
FALSE	TRUE	FALSE	Zadanie zostało pomyślnie zakończone.
FALSE	FALSE	TRUE	Zadanie zostało zakończone z błędem. Przyczynę błędu można odczytać z parametru STATUS.
FALSE	FALSE	FALSE	Nowe zadanie nie zostało przydzielone

Parametry TSEND_C

LAD



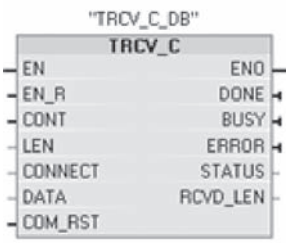
FBD



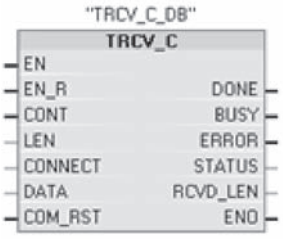
Parametr	Typ	Typ danych	Opis
REQ	INPUT	BOOL	Parametr sterujący REQ rozpoczyna wysyłanie zadania w trakcie połączenia opisanego w CONNECT w chwili wystąpienia narastającego zbocza.
CONT	INPUT	BOOL	Parametr sterujący CONT: • 0: rozłącz • 1: ustanów i utrzymuj połączenie
LEN	INPUT	INT	Maksymalna liczba bajtów do wysłania w zadaniu. Por. zależność między CPU i Protocol Variant oraz Transferable Data Length.
CONNECT	IN_OUT	ANY	Wskaźnik do opisu połączenia.
DATA	IN_OUT	ANY	Obszar nadawczy; zawiera adres i długość danych do nadania.
COM_RST	IN_OUT	BOOL	Parametr COM_RST: • Całkowity restart bloku funkcji, istniejące połączenie zostaje zakończone.
DONE	OUTPUT	BOOL	Parametr DONE statusu: • 0: Zadanie jeszcze się nie rozpoczęło lub ciągle jest w toku. • 1: Zadanie wykonane bez błędu.
BUSY	OUTPUT	BOOL	Parametr BUSY statusu: • 0: Zadanie wykonane. • 1: Zadanie jeszcze nie wykonane. Nowe zadanie nie może zostać rozpoczęte.
ERROR	OUTPUT	BOOL	Parametr ERROR statusu: • 1: Podczas przetwarzania wystąpił błąd. Szczegółowe informacje o typie błędu zawiera STATUS.
STATUS	OUTPUT	WORD	Parametr STATUS statusu: Informacje o błędzie.

Parametry TRCV_C

LAD



FBD



Parametr	Typ parametru	Typ danych	Opis
EN_R	IN	BOOL	Parametr sterujący uaktywniany do odbioru: Kiedy EN_R = 1, wtedy TRCV_C jest gotowa do odbioru. Zadanie odbioru jest wykonywane.
CONT	IN	BOOL	Parametr sterujący CONT: • 0: rozłącz • 1: ustanów i utrzymuj połączenie
LEN	IN	INT	Długość obszaru odbiorczego w bajtach. W celu poznania znaczenia LEN = 0 lub LEN <> 0 por. wyżej (tryby odbiorcze). W celu poznania wartości zakresów, por. zależność między CPU i Protocol Variant (connection_type) oraz Transferable Data Length.
CONNECT	IN_OUT	ANY	Wskaźnik do opisu połączenia.
DATA	IN_OUT	ANY	Obszar odbiorczy zawiera adres początkowy i maksymalną długość danych odbieranych.
COM_RST	IN_OUT	BOOL	Parametr COM_RST: • 1: Całkowity restart bloku funkcji, istniejące połączenie zostaje zakończone.
DONE	OUT	BOOL	Parametr DONE statusu: • 0: Zadanie jeszcze się nie rozpoczęło lub ciągle jest w toku. • 1: Zadanie wykonane bez błędu.
BUSY	OUT	BOOL	Parametr BUSY statusu: • 0: Zadanie wykonane. • 1: Zadanie jeszcze nie wykonane. Nowe zadanie nie może zostać rozpoczęte.
ERROR	OUT	BOOL	Parametr ERROR statusu: • 1: Podczas przetwarzania wystąpił błąd. Szczegółowe informacje o typie błędu zawiera STATUS.
STATUS	OUT	WORD	Parametr STATUS statusu: Informacje o błędzie.
RCVD_C	OUT	INT	Rzeczywista ilość odebranych danych wyrażona w bajtach.

Parametry ERROR i STATUS

ERROR	STATUS (W#16#...)	Opis
0	0000	Zadanie wykonane bez błędu
0	7000	Żadne zadanie nie jest aktualnie wykonywane
0	7001	Start wykonywania zadania, ustanowienie połączenia, oczekiwanie na połączenie partnera
0	7002	Rozpoczęcie otrzymywania danych
0	7003	Połączenie jest kończone

ERROR	STATUS (W#16#...)	Opis
0	7004	Połączenie ustanowione i monitorowane, żadne zadanie nie jest wykonywane
1	8085	Parametr LEN ma wartość 0 lub większą od największej dopuszczalnej wartości
1	8086	Parametr ID wykroczył poza dozwolony zakres
1	8087	Osiągnięto maksymalną liczbę połączeń; nie jest możliwe żadne dodatkowe połączenie
1	8088	Parametr LEN ma wartość większą niż obszar pamięci wyspecyfikowany w DATA; obszar pamięci odbiorczej jest za mały
1	8089	Parametr CONNECT nie wskazuje na blok danych
1	8091	Przekroczona głębokość zagnieżdżenia
1	809A	Parametr CONNECT wskazuje na pole, które nie odpowiada długości w opisie połączenia
1	809B	ID urządzenia lokalnego w opisie połączenia jest niezgodne z CPU
1	80A1	Błąd komunikacji: • Wyspecyfikowane połączenie nie zostało jeszcze ustanowione • Wyspecyfikowane połączenie jest aktualnie kończone; transmisja tym kanałem połączeniowym jest niemożliwa • Interfejs jest aktualnie reinicjalizowany
1	80A3	Wykonywana jest próba zakończenia nieistniejącego połączenia
1	80A7	Błąd komunikacji: wywołano TDISCON zanim został zakończony TCON (TDISCON musi najpierw całkowicie zakończyć połączenie wskazywane przez ID)
1	80B2	Parametr CONNECT wskazuje blok danych wygenerowany za pomocą słowa kluczowego UNLINKED
1	80B3	Niespójne parametry: • Błąd w opisie połączenia • Lokalny port (parametr local_tsap_id) występuje już w opisie innego połączenia • ID w opisie połączenia różni się od ID wyspecyfikowanego jako parametr
1	80B4	Podczas używania wariantu protokołu ISO on TCP (connection_ type = B#16#12), dla ustanowionego pasywnego połączenia (active_est = FALSE), zastał naruszony jeden lub oba następujące warunki: „local_tsap_id_len >= B#16#02” i/lub „local_tsap_id[1] = B#16#E0”
1	80C3	Wszystkie zasoby połączenia są w użyciu
1	80C4	Przejściowy błąd komunikacji: • Połączenie nie może być aktualnie ustanowione • Interfejs odbiera nowe parametry • To skonfigurowane połączenie jest aktualnie usuwane przez TDISCON
1	8722	Parametr CONNECT: Nieprawidłowy obszar źródłowy: obszar nie istnieje w DB
1	873A	Parametr CONNECT: Niemożliwy dostęp do opisu połączenia (np. nie dostępny DB)
1	877F	Parametr CONNECT: Błąd wewnętrzny, taki jak niepoprawna referencja ANY

Otwarcie komunikacji przez Ethernet za pomocą kontroli połącz/rozłącz

Komunikacja przez Ethernet z wykorzystaniem protokołów TCP i ISO on TCP

Poniższe instrukcje programu kontrolują proces komunikacji:

1. TCON: nawiązywanie połączenia.
2. TSEND i TRCV: wysyłanie i odbieranie danych.
3. TDISCON: przerywanie połączenia.

Użycie protokołów TCP i ISO on TCP

W celu ustanowienia połączenia komunikacyjnego obaj partnerzy komunikacyjni wykonują instrukcję TCON. Aby wyspecyfikować aktywnych i pasywnych końcowych partnerów komunikacyjnych użytkownik wykorzystuje parametry.

Po skonfigurowaniu i ustanowieniu połączenia jest ono automatycznie utrzymywane i monitorowane przez CPU.

Jeżeli połączenie jest zakończone na przykład w związku z uszkodzeniem linii lub przez zdalnego partnera, to partner aktywny podejmuje próby ponownego nawiązania tego skonfigurowanego połączenia. Użytkownik nie musi ponownie wykonywać TCON.

Jeżeli zostaje wykonana instrukcja TDISCON lub CPU przechodzi do trybu STOP, to istniejące połączenie jest kończone i konfiguracja połączenia jest usuwana. Aby skonfigurować i ponownie ustanowić połączenie trzeba znowu wykonać instrukcję TCON.

Opis funkcjonalny

TCON, TDISCON, TSEND i TRCV działają asynchronicznie co oznacza, że przetwarzanie zadania rozciąga się na wiele wykonań instrukcji.

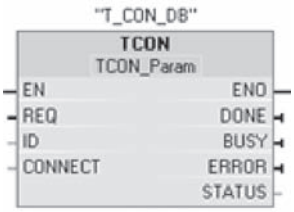
Na przykład, użytkownik wykonując instrukcję TCON z parametrem REQ = 1 uruchamia zadanie konfigurujące i ustanawiające połączenie. Następnie korzysta z dodatkowych wykonań instrukcji TCON w celu monitorowania postępu zadania i testowania zakończenia zadania z parametrem DONE.

W poniższej tabeli przedstawiono związki między BUSY, DONE i ERROR. Korzystanie z tej tabeli pozwala ustalić aktualny status wykonywanego zadania.

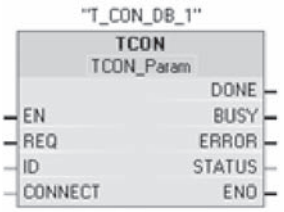
BUSY	DONE	ERROR	Opis
TRUE	nieistotny	nieistotny	Zadanie jest w toku wykonywania.
FALSE	TRUE	FALSE	Zadanie zostało pomyślnie zakończone.
FALSE	FALSE	TRUE	Zadanie zostało zakończone z błędem. Przyczynę błędu można odczytać z parametru STATUS.
FALSE	FALSE	FALSE	Nowe zadanie nie zostało przydzielone

TCON

LAD



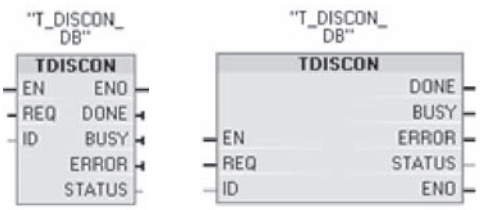
FBD



Parametr	Typ parametru	Typ danych	Opis
REQ	IN	BOOL	Parametr sterujący REQUEST uruchamia zadanie mające ustanowić połączenie określone przez ID. Zadanie rozpoczyna się przy zboczu narastającym.
ID	IN	CONN_OUC (WORD)	Wskazuje połączenie jakie ma zostać ustanowione ze zdalnym partnerem lub pomiędzy użytkownikiem programu i warstwą komunikacyjną systemu operacyjnego. ID musi być takie samo jak powiązany parametr ID w lokalnym opisie połączenia. Zakres wartości: W#16#0001 do W#16#0FFF
CONNECT	IN_OUT	TCON-Param	Wskaźnik do opisu połączenia.
DONE	OUT	BOOL	Parametr DONE statusu: • 0: Zadanie jeszcze nie rozpoczęte lub nadal w toku. • 1: Zadanie wykonane bez błędu.
BUSY	OUT	BOOL	• BUSY = 1: Zadanie jeszcze nie zakończone. • BUSY = 1: Zadanie jeszcze nie zakończone.BUSY = 0: Zadanie wykonane.
ERROR	OUT	BOOL	Parametr ERROR statusu: ERROR = 1: Podczas wykonywania zadania wystąpił błąd. Szczegółowe informacje o typie błędu zawiera STATUS.
STATUS	OUT	WORD	Parametr STATUS statusu: Informacje o błędzie.

TDISCON

LAD **FBD**



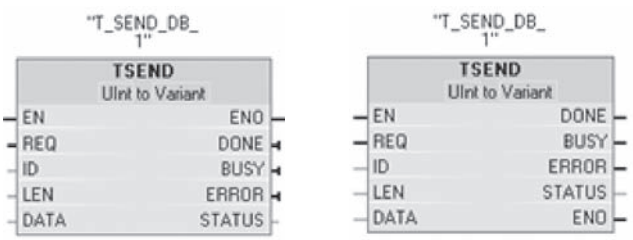
Parametr	Typ parametru	Typ danych	Opis
REQ	IN	BOOL	Parametr sterujący REQUEST uruchamia zadanie mające ustanowić połączenie określone przez ID. Zadanie rozpoczyna się przy zboczu narastającym.
ID	IN	CONN_OUC (WORD)	Wskazuje połączenie jakie ma zostać zakończone ze zdalnym partnerem lub pomiędzy użytkownikiem programu i warstwą komunikacyjną systemu operacyjnego. ID musi być takie samo jak powiązany parametr ID w lokalnym opisie połączenia. Zakres wartości: W#16#0001 do W#16#0FFF
DONE	OUT	BOOL	Parametr DONE statusu: • 0: Zadanie jeszcze nie rozpoczęte lub nadal w toku. • 1: Zadanie wykonane bez błędu.
BUSY	OUT	BOOL	• BUSY = 1: Zadanie jeszcze nie zakończone. • BUSY = 0: Zadanie wykonane.
ERROR	OUT	BOOL	ERROR = 1: Podczas przetwarzania wystąpił błąd.
STATUS	OUT	WORD	Kod błędu.

TCP i ISO on TCP

TDISCON kończy połączenie komunikacyjne między CPU i partnerem komunikacyjnym.

TSEND

LAD FBD



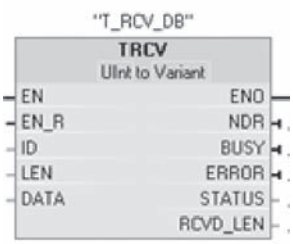
Parametr	Typ parametru	Typ danych	Opis
REQ	IN	BOOL	Parametr sterujący REQUEST uruchamia zadanie wysyłania w chwili wystąpienia zbocza narastającego. Dane są przesyłane z obszaru określonego przez DATA i LEN.
ID	IN	CONN_OUC (WORD)	Wskazuje powiązane połączenie. ID musi być takie samo jak powiązany parametr ID w lokalnym opisie połączenia. Zakres wartości: W#16#0001 do W#16#0FFF
LEN	IN	INT	Maksymalna liczba bajtów do wysłania przez zadanie.
DATA	IN_OUT	VARIANT	Wskaźnik do obszaru zawierającego dane do wysłania: Obszar nadawczy: zawiera adres i długość. Adres odnosi się do: • Tabeli wejściowej obrazu procesu. • Tabeli wyjściowej obrazu procesu. • Bitu w pamięci. • Bloku danych.
DONE	OUT	BOOL	Parametr DONE statusu: • 0: Zadanie jeszcze nie rozpoczęte lub nadal w toku. • 1: Zadanie wykonane bez błędu.
BUSY	OUT	BOOL	• BUSY = 1: Zadanie jeszcze nie zakończone. Nowe zadanie nie może zostać rozpoczęte. BUSY = 0: Zadanie wykonane.
ERROR	OUT	BOOL	Parametr ERROR statusu: ERROR = 1: Podczas wykonywania zadania wystąpił błąd. Szczegółowe informacje o typie błędu zawiera STATUS.
STATUS	OUT	WORD	Parametr STATUS statusu: Informacje o błędzie.

UWAGA

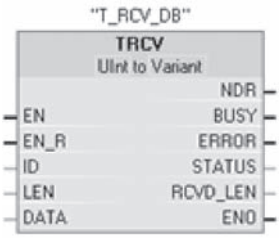
Ze względu na asynchroniczne przetwarzanie TSEND, użytkownik musi utrzymywać w obszarze nadawczym spójne dane aż do chwili gdy parametr DONE lub parametr ERROR przyjmie wartość TRUE.

TRCV

LAD



FBD



Parametr	Typ parametru	Typ danych	Opis
EN_R	IN	BOOL	Parametr sterujący uaktywniany do odbioru: Kiedy EN_R = 1, wtedy TRCV jest gotowa do odbioru. Zadanie odbioru jest wykonywane.
ID	IN	CONN_OUC (WORD)	Wskazuje powiązane połączenie. ID musi być takie samo jak powiązany parametr ID w lokalnym opisie połączenia. Zakres wartości: W#16#0001 do W#16#0FFF
LEN	IN	INT	Długość obszaru odbiorczego w bajtach. W celu poznania znaczenia LEN = 0 lub LEN <> 0 por. niżej (tryby odbiorcze TRCV).
DATA	IN_OUT	VARIANT	Wskaźnik do odebranych danych: Obszar odbiorczy (por. definicję poniżej); zaw iera adres i długość. Adres odnosi się do: • Tabeli wejściowej obrazu procesu. • Tabeli wyjściowej obrazu procesu. • Bitu w pamięci. • Bloku danych.
NDR	OUT	BOOL	Parametr NDR statusu: • NDR = 0: Zadanie jeszcze nie rozpoczęte lub nadal w toku. • NDR = 1: Zadanie wykonane bez błędu.
BUSY	OUT	BOOL	• BUSY = 1: Zadanie jeszcze nie wykonane. Nowe zadanie nie może zostać rozpoczęte. • BUSY = 0: Zadanie wykonane.
ERROR	OUT	BOOL	Parametr ERROR statusu: ERROR = 1: Podczas przetwarzania wystąpił błąd. Szczegółowe informacje o typie błędu zawiera STATUS.
STATUS	OUT	WORD	Parametr STATUS statusu: Informacje o błędzie.
RCVD_LEN	OUT	INT	Rzeczywista ilość odebranych danych wyrażona w bajtach.

UWAGA

Ze względu na asynchroniczne przetwarzanie TRCV, dane w obszarze odbiorczym są spójne tylko wtedy, kiedy parametr NRD przyjmuje wartość TRUE.

Obszar odbiorczy

Jest to obszar, w którym TRCV zapisuje odebrane dane.

Obszar odbiorczy określają następujące dwie zmienne:

- Wskaźnik do początku obszaru
- Długość obszaru

Długość obszaru jest określona, w zależności od używanego wariantu protokołu, za pomocą parametru LEN (jeżeli LEN <> 0) lub informacji o długości zawartej w parametrze DATA (jeśli LEN = 0).

Tryby odbiorcze TRCV

W następującej tabeli znajdują się informacje w jaki sposób TRCV wprowadza odebrane dane do obszaru odbiorczego.

Wariant protokołu	Wprowadzanie danych do obszaru odbiorczego	Parametr „connection_type”	Wartość parametru LEN
TCP	Tryb „Ad hoc”	B#16#11	0
TCP	Odbiór danych o określonej długości	B#16#11	<> 0
ISO on TCP	Kontrolowane przez protokół	B#16#12	0 (rekomendowana) lub <> 0

Tryb TCP / ad hoc

Tryb ad hoc istnieje tylko w wariancie protokołu TCP. Użytkownik może ustawić tryb ad hoc przypisując parametrowi LEN wartość 0.

Obszar odbiorczy jest identyczny z obszarem uformowanym przez DATA. Maksymalnie są odbierane 1472 bajty.

Natychmiast po odebraniu bloku danych, TRCV zapisuje dane do obszaru odbiorczego i ustawia NDR na 1.

TCP/odbiór danych o określonej długości

Użytkownik może ustawić tryb odbioru danych o określonej długości przypisując parametrowi LEN wartość inną niż 0.

Obszar odbiorczy jest definiowany przez parametry LEN i DATA.

Natychmiast po odebraniu LEN bajtów, TRCV zapisuje dane do obszaru odbiorczego i ustawia NDR na 1.

ISO on TCP/przepływ danych kontrolowany protokołem

W wariancie protokołu ISO on TCP, przesyłane dane są kontrolowane przez protokół.

Obszar odbiorczy jest definiowany przez parametry LEN i DATA.

Natychmiast po odebraniu wszystkich danych zadania, TRCV zapisuje dane do obszaru odbiorczego i ustawia NDR na 1.

Kody warunkowe w przypadku TCON

ERROR	STATUS (W#16#...)	Opis
0	0000	Połączenie ustanowione bez błędu
0	7000	Żadne zadanie nie jest aktualnie wykonywane
0	7001	Start wykonywania zadania, ustanowienie połączenia
0	7002	Kontynuacja wywołania (REQ nieistotny), nawiązywanie połączenia
1	8086	Parametr ID wykroczył poza dozwolony zakres
0	8087	Osiągnięto maksymalną liczbę połączeń; nie jest możliwe żadne dodatkowe połączenie
1	809B	ID urządzenia lokalnego w opisie połączenia jest niezgodne z CPU
1	80A1	Połączenie lub port jest już zajęty przez użytkownika
1	80A2	Lokalny lub zdalny port jest zajęty przez system
1	80A3	Wykonywana jest próba ponownego ustalenia istniejącego połączenia
1	80A4	Adres IP końcowego, zdalnego punktu połączenia jest nieprawidłowy; może być zgodny z lokalnym adresem IP
1	80A7	Błąd komunikacji: wywołano TDISCON zanim został zakończony TCON TDISCON musi najpierw całkowicie zakończyć połączenie wskazywane przez ID.
1	80B3	Niespójne przypisane parametry: Błąd grupowy kodów błędu W#16#80A0 do W#16#80A2, W#16#80A4, W#16#80B4 do W#16#80B9
1	80B5	Błąd parametru active_est
1	80B6	Błąd przypisania parametru w parametrze connection_type
1	80B7	Błąd w jednym z następujących parametrów: block_length, local_tsap_id_len, rem_subnet_id_len, rem_staddr_len, rem_ tsap_id_len, next_staddr_len
1	80B8	Parametr w lokalnym opisie połączenia i parametr ID są różne
1	80C3	Wszystkie zasoby połączenia są w użyciu
1	80C4	Przejściowy błąd komunikacji: • Połączenie nie może być aktualnie ustanowione • Interfejs odbiera nowe parametry • To skonfigurowane połączenie jest aktualnie usuwane przez TDISCON

Kody warunkowe w przypadku TDISCON

ERROR	STATUS (W#16#...)	Opis
0	0000	Połączenie zakończone pomyślnie
0	7000	Żadne zadanie nie jest aktualnie wykonywane
0	7001	Start wykonywania zadania, połączenie jest zakańczane
0	7002	Kontynuacja wywołania (REQ nieistotny), połączenie jest zakańczane
1	8086	Parametr ID wykroczył poza dozwolony zakres
1	80A3	Wykonywana jest próba zakończenia nieistniejącego połączenia
1	80C4	Przejściowy błąd komunikacji: Interfejs odbiera nowe parametry lub aktualnie jest ustanawiane połączenie

Kody warunkowe w przypadku TSEND

ERROR	STATUS (W#16#...)	Opis
0	0000	Zadanie wysyłania wykonane bez błędu
0	7000	Żadne zadanie nie jest aktualnie wykonywane
0	7001	Start wykonywania zadania, dane są przesyłane. Podczas wykonywania tego zadania system operacyjny ma dostęp do danych w obszarze nadawczym DATA.
0	7002	Kontynuacja wywołania (REQ nieistotny), zadanie jest wykonywane. Podczas wykonywania tego zadania system operacyjny ma dostęp do danych w obszarze nadawczym DATA.
1	8085	Parametr LEN ma wartość 0 lub większą od największej dopuszczalnej wartości
1	8086	Parametr ID wykroczył poza dozwolony zakres
1	8088	Parametr LEN ma wartość większą niż obszar pamięci wyspecyfikowany w DATA
1	80A1	Błąd komunikacji: • Wyspecyfikowane połączenie nie zostało jeszcze ustanowione • Wyspecyfikowane połączenie jest aktualnie kończone; transmisja tym kanałem połączeniowym jest niemożliwa • Interfejs jest aktualnie reinicjalizowany
1	80C3	Brak dostępnych zasobów wewnętrznych: Blok o tym ID jest już aktualnie przetwarzany z priorytetem o innej klasie.
1	80C4	Przejściowy błąd komunikacji: • Połączenie z partnerem komunikacyjnym nie może być aktualnie ustanowione • Interfejs odbiera nowe parametry lub połączenie jest aktualnie ustanawiane.

Kody warunkowe w przypadku TRCV

ERROR	STATUS (W#16#...)	Opis
0	0000	Nowe dane zaakceptowane: Bieżąca długość odebranych danych jest wskazywana przez RCVD_LEN.
0	7000	Blok nie jest gotowy do odebrania
0	7001	Blok jest gotowy do odebrania zadanie odbierania jest aktywowane.
0	7002	Kontynuacja wywołania, zadanie odbierania jest wykonywane. Podczas wykonywania tego zadania dane są zapisywane do obszaru odbiorczego. W związku z tym może wystąpić błąd związany z niespójnością danych w obszarze odbiorczym.
1	8085	Parametr LEN jest większy od największej dopuszczalnej wartości lub użytkownik zmienił LEN lub DATA od czasu pierwszego wywołania.
1	8086	Parametr ID wykroczył poza dozwolony zakres
1	8088	Obszar odbiorczy jest za mały: LEN ma wartość większą niż obszar pamięci wyspecyfikowany w DATA.
1	80A1	Błąd komunikacji: • Wyspecyfikowane połączenie nie zostało jeszcze ustanowione • Wyspecyfikowane połączenie jest aktualnie kończone; zadanie odbierania tym kanałem połączeniowym jest niemożliwe do wykonania • Interfejs aktualnie odbiera nowe parametry
1	80C3	Brak dostępnych zasobów wewnętrznych: Blok o tym ID jest już aktualnie przetwarzany z priorytetem o innej klasie.
1	80C4	Przejściowy błąd komunikacji: • Połączenie z partnerem komunikacyjnym nie może być aktualnie ustanowione • Interfejs odbiera nowe parametry lub połączenie jest aktualnie ustanawiane.

6.2.4.2 Instrukcje komunikacji Point-to-Point

Szczegółowe informacje dotyczące instrukcji PtP i modułów komunikacyjnych przedstawiono w rozdziale dotyczącym komunikacji punkt-punkt (Point-to-Point, PtP).

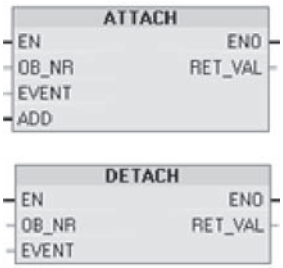
6.2.5 Instrukcje przerwania

6.2.5.1 Instrukcje przyłączania i odłączania

Za pomocą instrukcji ATTACH i DETACH użytkownik może aktywować i dezaktywować podprogramy obsługi przerwań sterownych zdarzeniami.

- ATTACH aktywuje wykonanie podprogramu OB obsługi przerwania sprzętowego.
- DETACH dezaktywuje wykonanie podprogramu OB obsługi przerwania sprzętowego.

LAD



Parametr	Typ parametru	Typ danych	Opis
OB_NR	IN	INT	Identyfikator bloku organizacyjnego: Dokonuje wyboru spośród dostępnych bloków obsługi przerwań sprzętowych utworzonych za pomocą opcji „Add new block”. Podwójne kliknięcie na pole parametru, a następnie kliknięcie ikony pomocy wyświetli dostępne OB.
EVENT	IN	DWORD	Identyfikator zdarzenia: Dokonuje wyboru spośród dostępnych przerwań zdarzeń sprzętowych związanych z wejściami cyfrowymi i szybkimi licznikami, które zostały uaktywnione podczas konfiguracji urządzenia PLC. Podwójne kliknięcie na pole parametru, a następnie kliknięcie ikony pomocy wyświetli dostępne zdarzenia.
ADD (tylko ATTACH)	IN	BOOL	ADD = 0 (domyślnie): To zdarzenie zastępuje wszystkie inne zdarzenia dołączone do tego OB. ADD = 1: To zdarzenie jest dołączane do poprzednich zdarzeń dołączonych do tego OB.
RET_VAL	OUT	INT	Kod warunkowy wykonania instrukcji.

Przerwania sprzętowe S7-1200

S7-1200 obsługuje następujące zdarzenia przerwań sprzętowych:

- Zbocze narastające (wszystkie wbudowane wejścia CPU oraz wejścia cyfrowe w modułach rozszerzeń):
  - Zbocze narastające występuje wtedy, kiedy na wejściu cyfrowym zachodzi zmiana z OFF na ON w odpowiedzi na zmianę sygnału urządzenia zainstalowanego na obiekcie i podłączonego do tego wejścia.
- Zbocze opadające (wszystkie wbudowane wejścia CPU oraz wejścia cyfrowe wejścia cyfrowe w modułach rozszerzeń):
  - Zbocze op adające występuje wtedy, kiedy na wejściu cyfrowym zachodzi zmiana z ON na OFF.

- Wartość bieżąca zliczeń szybkiego licznika (HSC) jest równa wartości odniesienia, CV = RV (dla HSC pod 1 do 6):
  – Warunek CV = RV generuje przerwanie HSC w momencie gdy liczba zliczeń zmienia się z wartości sąsiedniej na wartość dokładnie odpowiadającą ustalonej wcześniej wartości odniesienia.
- Zmiana kierunku zliczania HSC (dla HSC pod 1 do 6):
  – Zdarzenie zmiany kierunku zliczania występuje wtedy, kiedy stwierdza się zmianę kierunku zliczania HSC z rosnącego na malejący lub malejącego na rosnący.
- Zdarzenie zewnętrznego kasowania HSC (dla HSC pod 1 do 6):
  – Pewne tryby pracy HSC pozwalają przypisać mu wejście cyfrowe służące do zewnętrznego kasowania wartości zliczeń HSC do zera. Zdarzenie zewnętrznego kasowania występuje w przypadku takiego HSC wtedy, kiedy na tym wejściu cyfrowym zachodzi zmiana sygnału z OFF na ON.

Uaktywnianie zdarzeń przerwań sprzętowych podczas konfiguracji urządzenia PLC

Przerwania sprzętowe muszą być uaktywnione podczas konfiguracji urządzenia PLC. Podczas wykonywania konfiguracji lub w trakcie pracy, użytkownik musi zaznaczyć pole wyboru uaktywnienia dla danego kanału sygnału wejściowego lub licznika HSC, jeśli chce wybrać to zdarzenie.

Zaznaczanie pola wyboru podczas konfiguracji urządzenia PLC:

- Wejścia cyfrowe
  – Uaktywnienie detekcji zbocza narastającego
  – Uaktywnienie detekcji zbocza opadającego
- Szybkie liczniki
  – Uaktywnienie funkcjonowania danego licznika
  – Generacja przerwania przy zrównaniu liczby bieżących zliczeń z wartością referencyjną
  – Generacja przerwania przy zdarzeniu zewnętrznego kasowania
  – Generacja przerwania przy zdarzeniu zmiany kierunku zliczeń

Dołączanie do programu użytkownika nowego kodu bloku OB obsługi przerwania sprzętowego

Domyślnie, żaden OB nie jest powiązany ze zdarzeniem, gdy zdarzenie jest po raz pierwszy uaktywnione. Jest to wskazywane przez etykietę „HW interrupt:” „<not connected>” w konfiguracji urządzenia Ze zdarzeniami przerwań sprzętowych można powiązać tylko OB obsługi przerwań sprzętowych. Wszystkie istniejące OB przerwań sprzętowych pojawiają się na rozwijanej liście „HW interrupt:”. Jeśli na liście nie ma żadnego OB, to w następujący sposób należy utworzyć OB typu „Hardware interrupt:”. W drzewie projektu, w gałęzi „Program blocks” należy:

1. Podwójnie kliknąć „Add new block” i wybrać „Organization block (OB)” a następnie „Hardware interrupt”.

2. Opcjonalnie można zmienić nazwę OB oraz wybrać język programowania (LAD lub FBD) i numer bloku (użytkownik może się przełączyć w tryb ręczny i wybrać inny numer bloku niż podpowiedziany).
3. Wyedytować OB i dodać sposób reakcji na występujące zdarzenie. Z tego OB można wywoływać FC i FB do głębokości czterech poziomów zagnieżdżenia.

Parametr OB_NR

Wszystkie nazwy OB przerwań sprzętowych pojawiają się w menu konfiguracji urządzenia na rozwijanej liście „Hardware interrupt:" oraz na rozwijanej liście parametru OB_NR instrukcji ATTACH/DETACH.

Parametr EVENT

Kiedy przerwanie sprzętowe jest uaktywnione, wtedy temu szczególnemu zdarzeniu jest przypisywana unikalna nazwa. Użytkownik może zmienić nazwę zdarzenia edytując pole „Event Name:", ale nazwa musi pozostać unikalna. W tablicy *tagów* „Constans" te nazwy zdarzeń stają się nazwami *tagów* i występują na rozwijanej liście parametru EVENT w blokach instrukcji ATTACH/DETACH. Wartość tagu jest wewnętrznym numerem stosowanym do identyfikacji zdarzenia.

Ogólny opis instrukcji

Każde zdarzenie przerwania sprzętowego może zostać przyłączone do OB przerwania sprzętowego, które będzie kolejkowane w przypadku wystąpienia tego zdarzenia przerwania. Powiązanie OB ze zdarzeniem może nastąpić w trakcie konfiguracji lub podczas pracy.

Podczas konfiguracji użytkownik ma możliwość dołączenia lub odłączenia OB do/od uaktywnionego zdarzenia. W celu dołączenia OB do zdarzenia w trakcie konfiguracji, należy skorzystać z rozwijanej listy „HW interrupt:" (kliknąć na prawą strzałkę skierowana w dół) i z listy dostępnych OB przerwań sprzętowych wybrać OB. Można wybrać określoną nazwę OB z listy lub też wybrać <not connected> w celu usunięcia połączenia.

Uaktywnione przerwania sprzętowe można również dołączyć lub odłączyć w trakcie pracy. W celu dołączenia lub odłączenia uaktywnionego przerwania do/od odpowiedniego OB w trakcie pracy korzysta się z instrukcji programu ATTACH lub DETACH (wielokrotnie jeśli jest taka potrzeba). Jeśli aktualnie nie jest dołączony żaden OB (w wyniku wyboru <not connected> podczas konfiguracji urządzenia lub w rezultacie wykonania instrukcji DETACH), to uaktywnione przerwania sprzętowe są ignorowane.

Działanie instrukcji DETACH

Instrukcja DEATCH jest stosowana do konkretnego zdarzenia, albo do wszystkich zdarzeń w określonym OB. Jeśli jest wyspecyfikowany EVENT, to tylko to jedno zdarzenie jest odłączone od określonego OB_NR; wszystkie inne zdarzenia podłączone do tego OB_NR nadal pozostają podłączone. Jeśli EVENT nie jest wyspecyfikowany, to wszystkie aktualnie podłączone do OB_NR zdarzenia zostają odłączone.

Kody warunkowe

RET_VAL (W#16#....)	Status ENO	Opis
0000	1	Brak błędu
0001	0	Nie ma nic do odłączenia (tylko DETACH)
8090	0	OB nie istnieje
8091	0	Typ OB jest nieprawidłowy
8093	0	Zdarzenie nie istnieje

6.2.5.2 Instrukcje startu i kasowania obsługi przerwań od opóźnień

Instrukcje SRT_DINT i CAN_DINT pozwalają uruchomić i skasować obsługę przerwań od opóźnień. Każde przerwanie pochodzące od opóźnień jest jednorazowym zdarzeniem, które występuje po określonym czasie opóźnienia. Jeśli zdarzenie opóźnienia jest skasowane zanim upłynie czas opóźnienia, to przerwanie nie występuje.

- SRT_DINT uruchamia przerwanie od opóźnienia, które wykonuje podprogram OB (bloku organizacyjnego) po upływie czasu wyspecyfikowanego w parametrze DTIME.
- CAN_DINT kasuje uruchomione przerwanie od opóźnienia. Blok OB obsługi przerwania od opóźnienia nie jest w tym przypadku wykonywany.

LAD

FBD

Parametry SRT_DINT

Parametr	Typ parametru	Typ danych	Opis
OB_NR	IN	INT	Blok organizacyjny, który ma zostać uruchomiony po upływie czasu opóźnienia: Dokonuje wyboru spośród dostępnych OB obsługi przerwania od opóźnienia utworzonych za pomocą opcji „Add new block” w drzewie projektu. Podwójne kliknięcie na pole parametru, a następnie kliknięcie ikony pomocy wyświetli dostępne OB.

Parametr	Typ parametru	Typ danych	Opis
DTIME	IN	TIME	Wartość czasu opóźnienia (1 do 60000 ms). Czas opóźnienia można uczynić jeszcze dłuższy, np. włączając licznik do OB obsługi przerwania od opóźnienia.
SIGN	IN	WORD	Nie wykorzystany w S7-1200; może przyjąć dowolną wartość.
RET_VAL	OUT	INT	Kod warunkowy wykonania instrukcji.

Parametry CAN_DINT

Parametr	Typ parametru	Typ danych	Opis
OB_NR	IN	INT	Identyfikator OB przerwania od opóźnienia. Można stosować numer OB lub jego symboliczną nazwę.
RET_VAL	OUT	INT	Kod warunkowy wykonania instrukcji.

Opis działania instrukcji

Instrukcja SRT_DINT określa czas opóźnienia, uruchamia wewnętrzny timer czasu opóźnienia i wiąże podprogram OB obsługi przerwania od opóźnienia ze zdarzeniem upływu czasu opóźnienia. Kiedy upłynie nastawiony czas opóźnienia, wtedy jest generowane przerwanie programu, które wyzwala uruchomienie powiązanego OB obsługi przerwania od opóźnienia. Wykonując instrukcję CAN_DINT, użytkownik może skasować generację przerwania od biegnącego już opóźnienia zanim upłynie nastawiony czas opóźnienia. Łączna liczba aktywnych zdarzeń przerwań od opóźnień i od cyklu czasu nie może przekroczyć czterech.

Dołączanie do projektu podprogramów OB obsługi przerwań od opóźnień.

Z instrukcjami SRT_DINT i CAN_DINT można powiązać tylko OB obsługi przerwań od opóźnień. W nowym projekcie nie istnieje żaden OB obsługi przerwania od opóźnienia. Taki OB musi dodać do projektu użytkownik. W celu utworzenia OB przerwania od opóźnienia należy wykonać następujące kroki:

1. Podwójnie kliknąć „Add new block” w gałęzi „Program blocks” drzewa projektu, a następnie kolejno wybrać „Organization block (OB)” „Time delay interrupt”.
2. Opcjonalnie można zmienić nazwę OB, wybrać język programowania i numer bloku. W celu ustalenia innego numeru bloku niż podpowiedziany, należy przełączyć tryb pracy na numerowanie ręczne.
3. Wyedytować podprogram OB obsługi przerwania od opóźnienia i ustalić sposób reakcji programu w przypadku wystąpienia zdarzenia pochodzącego od opóźnienia. Z OB obsługi przerwania od opóźnienia można wywoływać FC i FB do głębokości czterech poziomów zagnieżdżenia.
4. Nowo przypisane nazwy OB obsługi przerwania od opóźnienia stają się dostępne po edycji parametru OB_NR instrukcji SRT_DINT i CAN_DINT.

Kody warunkowe

RET_VAL (W#16#....)	Opis
0000	Brak błędu
8090	Nieprawidłowy parametr OB_NR
8091	Nieprawidłowy parametr DTIME
80A0	Przerwanie od opóźnienia nie uruchomione

6.2.5.3 Instrukcje aktywacji i dezaktywacji przerwania od alarmu

Instrukcje DIS_AIRT i EN_AIRT uaktywniają i dezaktywują przetwarzanie przerwania pochodzącego od alarmu.

LAD

Parametr

Parametr	Typ parametru	Typ danych	Opis
RET_VAL	OUT	INT	Liczba opóźnień = liczba wykonań DIS_AIRT w kolejce.

Opis działania

DIS_AIRT opóźnia przetwarzanie nowych przerwań. Instrukcję DIS_AIRT można wykonać w OB więcej niż jeden raz. System operacyjny zlicza liczbę wykonań DIS_AIRT. Każde pozostaje efektywne aż do ponownego skasowania przez instrukcję EN_AIRT lub całkowitego zakończenia wykonywania bieżącego OB.

Przerwania które wystąpiły wtedy, kiedy instrukcja DIS_AIRT była efektywna są przetwarzane gdy zostaną ponownie aktywowane lub też natychmiast po zakończeniu wykonania bieżącego OB.

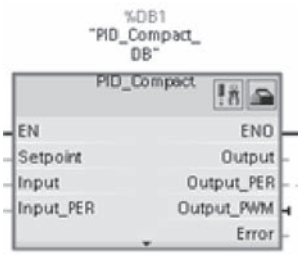
EN_AIRT uaktywnia przetwarzanie przerwań, które były poprzednio zablokowane instrukcją DIS_AIRT. Każde wykonanie DIS_AIRT może być skasowane wykonaniem EN_AIRT. Jeżeli na przykład, przerwania zostały zablokowane pięć razy poprzez pięciokrotne wykonanie instrukcji DIS_AIRT, to można tę blokadę skasować wykonując pięciokrotnie instrukcję EN_AIRT. Zanim przerwania zostaną ponownie odblokowane dla danego OB, wykonanie EN_AIRT musi nastąpić w tym samym OB lub w dowolnym FC lub FB wywołanym z tego samego OB.

Parametr RET_VAL sygnalizuje ile razy wykonanie obsługi przerwania było zablokowane, co jest równe liczbie wykonań DIS_AIRT ustawionych w kolejce. Obsługa przerwań jest ponownie aktywna wtedy, kiedy parametr RET_VAL = 0.

6.2.6 Regulacja PID

Polecenie „PID_Compact" udostępnia dla trybów automatycznego i ręcznego regulator PID wraz ze zoptymalizowaną opcją autostrojenia.

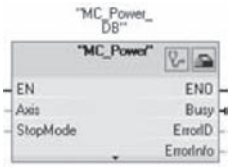
Więcej informacji na temat instrukcji „PID_Compact" znajduje się w pomocy *online* portalu TIA.



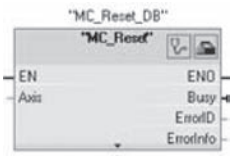
Instrukcja PID_Compact

6.2.7 Instrukcje sterowania ruchami – Motion Control

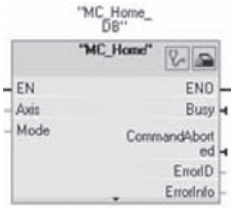
Instrukcje sterowania ruchami, w celu kontrolowania ruchu osi korzystają z przypisanego bloku danych technologicznych i dedykowanego PTO (*pulse train output* – wyjścia ciągu impulsów) CPU. Więcej informacji na temat instrukcji sterowania ruchami znajduje się w pomocy *online* portalu TIA.



MC_Power uaktywnia i dezaktywuje sterowane osie.



MC_Reset kasuje wszystkie błędy związane ze sterowaniem ruchem. Wykryte błędy związane ze sterowaniem ruchem są potwierdzane.



MC_Home ustanawia związek między programem sterowania osią i częścią mechaniczną systemu pozycjonowania osi.

MC_Halt kasuje wszystkie procesy sterowania ruchem i powoduje zatrzymanie ruchów osi. Pozycja zatrzymania nie jest określona.

MC_MoveJog uruchamia tryb Jog (impulsowania) w celach testowania i rozruchu.

MC_MoveAbsolute uruchamia ruch w celu osiągnięcia pozycji bezwzględnej. Zadanie kończy się po osiągnięciu pozycji położenia docelowego.

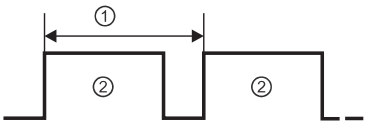
MC_MoveRelative uruchamia ruch pozycjonujący względem pozycji startowej.

MC_MoveVelocity powoduje ruch osi z określoną prędkością.

6.2.8 Instrukcja generowania impulsów

6.2.8.1 Instrukcja CTRL_PWM

Instrukcja CTRL_PWM dotycząca modulacji szerokości impulsów PWM (*Pulse Width Modulation*) zapewnia stały okres przebiegu wyjściowego ze zmiennym współczynnikiem wypełnienia. Wyjście PWM działa przez cały czas od chwili uruchomienia generując impulsy o określonej częstotliwości (okresie). Szerokość impulsów jest zmieniana zgodnie z potrzebami, tak by uzyskać pożądany efekt sterowania.



① okres
② szerokość impulsu

Współczynnik wypełnienia można określić w procentach jako część okresu (0 – 100 %), w tysięcznych (0 – 1000), dziesięciotysięcznych (1 – 10000) lub w formacie analogowym S7. Szerokość impulsu może się zmieniać od 0 (brak impulsu, zawsze OFF) do pełnej skali (brak impulsu, zawsze ON).

Ponieważ sygnał wyjściowy PWM może się zmieniać od zera do pełnego zakresu, więc jest to sygnał cyfrowy bardzo zbliżony do sygnału analogowego. Na przykład, sygnał wyjściowy PWM może być zastosowany do sterowania szybkością silnika od zatrzymania do pełnej prędkości albo może być użyty do sterowania położeniem zaworu od całkowitego zamknięcia do pełnego otwarcia.

Dostępne są dwa generatory impulsów pozwalające realizować funkcje wymagające ciągu szybkich impulsów: PWM i PTO (wyjście ciągu impulsów). PTO jest wykorzystywane przez instrukcje sterowania ruchem. Użytkownik może przypisać każdy z generatorów impulsów albo do PWM albo do PTO, ale nie do obu jednocześnie.

Oba generatory impulsów są mapowane do określonych wyjść cyfrowych, tak jak to pokazano w poniższej tabeli. Można wykorzystać w tym celu zarówno wewnętrzne wyjścia CPU, jak i opcjonalne wyjścia płytki sygnałowej. Numery punktów wyjściowych są pokazane w poniższej tabeli (przy założeniu domyślnej konfiguracji wyjść). Jeżeli użytkownik zmieni numerację wyjść, to numerami punktów wyjściowych będą te, które ustalił użytkownik. Niezależnie od tego PTO1/PWM1 wykorzystują dwa pierwsze wyjścia cyfrowe, a wyjścia PTO2/PWM2 wykorzystuję kolejne dwa wyjścia cyfrowe CPU albo dołączonej płytki sygnałowej. Zwracamy uwagę na to, że PWM wymaga tylko jednego wyjścia, podczas gdy PTO może opcjonalnie wykorzystywać dwa wyjścia na kanał. Jeżeli wyjście nie jest potrzebne dla impulsów, to jest dostępne dla innych celów.

Opis	Domyślne przypisanie wyjść		
		Impuls	Kierunek
PTO 1	Wyjścia CPU	Q0.0	Q0.1
	Wyjścia płytki sygnałowej	Q4.0	Q4.1
PWM 1	Wyjścia CPU	Q0.0	--
	Wyjścia płytki sygnałowej	Q4.0	--
PTO 2	Wyjścia CPU	Q0.2	Q0.3
	Wyjścia płytki sygnałowej	Q4.2	Q4.3
PWM 2	Wyjścia CPU	Q0.2	--
	Wyjścia płytki sygnałowej	Q4.2	--

Konfiguracja kanału impulsowego dla PWM

W celu przygotowania PWM do działania należy najpierw skonfigurować kanał impulsowy w menu konfiguracji urządzenia; wykonuje się to poprzez wybór CPU, następnie *Pulse Generator* (PTO/PWM) i wreszcie PWM1 lub PWM2. Następnie należy uaktywnić generator impulsowy (zaznaczyć pole wyboru). Jeśli generator jest uaktywniony, to jest mu przypisywana unikalna nazwa domyślna. Użytkownik może zmienić tę nazwę edytując pole „Name:”, ale nazwa musi pozostać unikalna. W tablicy *tagów* „Constans” nazwy aktywnych generatorów impulsowych

stają się *tagami* i są dostępne do wykorzystania jako parametr PWM instrukcji CTRL_PWM. Użytkownik ma możliwość w następujący sposób zmiany nazwy generatora impulsowego, dodawania komentarza i przypisywania parametrów:

Opcje impulsów PWM:

- Użycie generatora impulsowego: PWM lub PTO (wybrać PWM).
- Źródło wyjściowe: CPU lub płytka sygnałowa.
- Podstawa czasu: ms lub µs.
- Format szerokości impulsu:
  - procenty (0 – 100),
  - tysięczne (1 – 1000),
  - dziesięciotysięczne (1 – 10000),
  - format analogowy S7 (0 – 27648).
- Okres („Cycle time”): Należy wprowadzić wartość okresu. Tę wartość można zmienić jedynie w tym miejscu.
- Początkowa szerokość impulsu („Initial pulse width:”): Należy wprowadzić początkową wartość szerokości impulsu. Wartość szerokości impulsu można zmienić jedynie podczas wykonywania zadania.

Adresy wyjściowe („Output addresses”)

Adres początkowy („Start address:”): Należy wprowadzić adres Q-word, gdzie będzie ulokowana wartość szerokości impulsu. Lokalizacją domyślną jest QW1000 dla PWM1 i QW1002 dla PWM2. Wartość umieszczona pod tym adresem steruje szerokością impulsu i za każdym razem gdy następuje przejście PLC z trybu STOP do trybu RUN jest inicjalizowana przyjmując określoną wyżej wartość „Initial pulse width:” (początkowa szerokość impulsu). Podczas pracy systemu, w celu zmiany szerokości impulsu należy zmienić wartość określoną przez Q-word.

LAD

FBD

Parametr	Typ parametru	Typ danych	Wartość początkowa	Opis
PWM	IN	WORD	0	Identyfikator PWM: W tablicy tagów „Constans” nazwy aktywnych generatorów impulsowych stają się tagami i są dostępne do użycia jako parametr PWM.
ENABLE	IN	BOOL		1 = start generatora impulsów 0 = stop generatora impulsów
BUSY	OUT	BOOL	0	Funkcja zajęta
STATUS	OUT	WORD	0	Kody warunkowe wykonania instrukcji

Opis działania

Instrukcja CTRL_PWM wykorzystuje blok danych (DB) do przechowywania informacji o parametrach. Po umieszczeniu instrukcji CTRL_PWM w edytorze programu jest jej przypisywany DB. Parametry bloku danych nie są osobno zmieniane przez użytkownika, ale są kontrolowane przez instrukcję CTRL_PWM.

Wybór uaktywnionego generatora impulsowego, który ma być użyty, odbywa się poprzez zastosowanie jego nazwy tagu jako parametru PWM.

Kiedy sygnał na wejściu EN ma wartość TRUE, wtedy instrukcja CTRL_PWM uruchamia lub zatrzymuje generator PWM w zależności od stanu wejścia ENABLE. Szerokość impulsu jest określona przez wartość znajdującą się pod powiązanym adresem wyjściowym Q-word.

Ponieważ S7-1200 przetwarza żądanie podczas wykonywania instrukcji CTRL_PWM, więc parametr BUSY zawsze przyjmuje wartość FALSE w modelach CPU S7-1200.

Jeżeli jest wykryty błąd, to ENO jest ustawiany na FALSE, a parametr STATUS zawiera kod warunkowy.

Kiedy PLC po raz pierwszy wchodzi w tryb RUN, wtedy wartość szerokości impulsu przyjmuje wartość początkową określoną w konfiguracji urządzenia. Jeżeli trzeba zmienić szerokość impulsu, to użytkownik wpisuje wartości pod adres Q--word określony w konfiguracji urządzenia („Output addresses” / „Start address:”). W celu wpisania pożądanej szerokości impulsu pod adres Q-word korzysta się z instrukcji przesunięcia, konwersji, arytmetycznych lub bloku PID. Należy stosować właściwy zakres dla wartości Q-word (procenty, tysięczne, dziesięciotysięczne lub format analogowy S7).

Kody warunkowe

Wartość STATUS	Opis
0	Brak błędu
80A1	Identyfikator PWM nie adresuje prawidłowego PWM

6.3 Instrukcje biblioteki globalnej

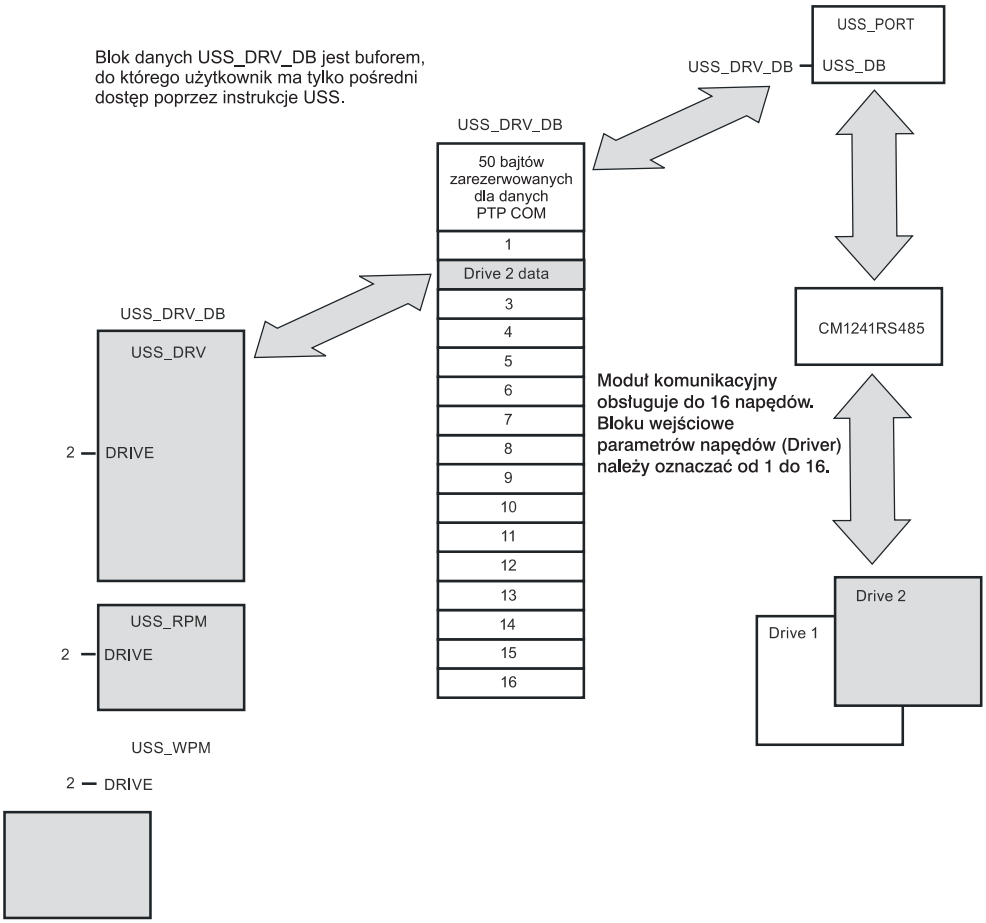
6.3.1 USS

Biblioteka protokołu USS pozwala sterować napędy Siemensa obsługujące protokół USS. Znajdujące się w niej instrukcje zawierają funkcje specjalnie zaprojektowane do korzystania z protokołu USS w celu komunikacji z napędem. Moduł CM1241RS485 komunikuje się z napędami poprzez porty RS485. Korzystając z biblioteki USS użytkownik może sterować fizycznym napędem i odczytywać/zapisywać parametry napędu.

6.3.1.1 Wymagania do stosowania protokołu USS

W celu obsługi protokołu USS w bibliotece znajdują się 1 blok funkcji i 3 funkcje. Każdy moduł komunikacyjny CM1241RS485 obsługuje maksymalnie 16 napędów.

Pojedyncza instancja zawiera tymczasową pamięć i bufory dla wszystkich napędów znajdujących się w sieci USS i podłączonych do zainstalowanych modułów komunikacyjnych PtP. Funkcje USS dla tych napędów współdzielą miedzy sobą informacje z tego bloku danych.



Poprzez moduł komunikacyjny PtP funkcja USS_PORT obsługuje komunikację pomiędzy CPU i napędami. Każde odwołanie do tej funkcji umożliwia co najwyżej jedną sesję komunikacyjną z jednym napędem. Program użytkownika musi wywoływać tę funkcję dostatecznie szybko by uniknąć przekroczenia limitu czasu napędu. Ta funkcja może być wywoływana w głównym lub dowolnym OB obsługi przerwań.

Blok USS_DRV zapewnia programowi użytkownika dostęp do wyspecyfikowanego napędu znajdującego się w sieci USS. Jego wejścia i wyjścia określają stan i pozwalają sterować napędem. Jeżeli w sieci jest 16 napędów, to program użytkownika musi zawierać co najmniej 16 wywołań USS_DRV – jedno na każdy napęd. Te bloki powinny być wywoływane z taką częstością, jak jest wymagana dla realizacji funkcji sterowania napędem.

Blok funkcji USS_DRV może być wywoływany tylko z głównego OB.

Ostrożnie

USS_DRV, USS_RPM, USS_WPM mogą być wywoływane tylko z głównego OB. Funkcja USS_PORT może być wywoływana z dowolnego OB, zwykle z OB obsługi przerwań od opóźnienia.

Niezabezpieczenie USS_PORT przed zakłóceniami pracy może spowodować nieoczekiwane błędy.

⚠	**OSTROŻNIE**
USS_DRV, USS_RPM, USS_WPM mogą być wywoływane tylko z głównego OB. Funkcja USS_PORT może być wywoływana z dowolnego OB, zwykle z OB obsługi przerwań od opóźnienia. Niezabezpieczenie USS_PORT przed zakłóceniami pracy może spowodować nieoczekiwane błędy.	

Funkcje USS_RPM i USS_WPM odczytują i zapisują parametry operacyjne odległego napędu. Te parametry sterują pracą napędu. Są one zdefiniowane w instrukcji napędu. Program użytkownika może zawierać tyle wywołań tych funkcji ile to jest konieczne, ale w dowolnej chwili dla jednego napędu może być aktywna tylko jedna funkcja zapisu lub odczytu. Funkcje USS_RPM i USS_WPM mogą być wywoływane tylko z głównego OB.

Obliczanie czasu potrzebnego na komunikacje z napędem

Komunikacja z napędem odbywa się asynchronicznie w stosunku do cyklu programu S7-1200. Zwykle S7-1200 wykona kilka cykli programu, zanim zostanie ukończona pełna transakcja komunikacyjna z jednym napędem.

Interwał USS_PORT jest to czas potrzebny do wykonania jednej transakcji komunikacyjnej. W poniższej tabeli przedstawiono minimalne interwały USS_PORT dla wszystkich prędkości transmisji. Wywoływanie USS_PORT częściej niż wynosi interwał USS_PORT nie powoduje zwiększenia liczby transakcji. Limit czasowy napędu jest to czas jaki może być przeznaczony na transakcję w warunkach, gdy błędy komunikacyjne spowodowały, że podjęte były 3 próby dokończenia transakcji. Domyślnie, protokół biblioteki USS podejmuje automatycznie do 2 prób na każdą transakcję.

Prędkość transmisji [bod]	Obliczony minimalny interwał USS_PORT [ms]	Limit czasowy pojedynczego napędu [ms]
1200	790	2370
2400	405	1215
4800	212,5	638
9600	116,3	349
19200	68,2	205
38400	44,1	133
57600	36,1	109
115200	28,1	85

6.3.1.2 Instrukcja USS_DRV

Instrukcja USS_DRV wymienia dane z napędem, wysyłając żądanie wiadomości i interpretując wiadomość otrzymaną od napędu. Dla każdego napędu powinien być użyty oddzielny blok funkcji, ale wszystkie funkcje związane z jedną siecią USS i modułem komunikacyjnym PtP muszą korzystać z tej samej instancji bloku danych. Kiedy instrukcja USS_DRV zostanie pierwszy raz użyta, wtedy użytkownik musi stworzyć DB i nadać mu nazwę, a następnie zawsze już korzystać z tego DB utworzonego podczas pierwszego użycia instrukcji.

Kiedy instrukcja jest wykonana po raz pierwszy, wtedy napęd wskazany przez adres USS (parametr DRIVE) jest inicjalizowany w instancji DB. Po tej inicjalizacji kolejne wykonania USS_PORT mogą rozpocząć komunikację z tym napędem określonym własnym numerem.

W celu zmiany numeru napędu należy przeprowadzić PLC z trybu STOP do RUN, co powoduje inicjalizację instancji DB. Parametry wejściowe są konfigurowane i umieszczane w buforze wiadomości USS TX, a wyjściowe odczytywane – jeśli uprzednio były poprawnie zapisane i istnieją – z bufora odpowiedzi. Podczas wykonywania USS_DRV nie odbywa się żadna transmisja danych. Komunikacja z napędami odbywa się podczas wykonywani instrukcji USS_PORT. USS_DRV jedynie konfiguruje wiadomości do wysłania i interpretuje dane, które mogły być otrzymane z napędu w odpowiedzi na poprzednie żądanie.

LAD (widok domyślny)

LAD (widok rozszerzony)

Klikając dół bloku można ją rozszerzyć w celu wyświetlenia wszystkich parametrów.

Wyprowadzenia parametrów wyświetlane na szaro są opcjonalne i nie muszą być przypisywane.

Parametr	Typ parametru	Typ danych	Opis
RUN	IN	BOOL	Bit startu napędu: Stan TRUE na tym wejściu uaktywnia ruch napędu z ustaloną prędkością.
OFF2	IN	BOOL	Bit stopu elektrycznego: Stan TRUE na tym wejściu powoduje zatrzymanie ruchu napędu bez hamowania – biegiem na luzie.
OFF3	IN	BOOL	Bit szybkiego stopu – Stan TRUE na tym wejściu wymusza szybkie zatrzymanie ruchu napędu z włączeniem hamowania, a nie tylko biegiem na luzie.
F_ACK	IN	BOOL	Bit potwierdzenia błędu – Ten bit jest ustawiany w celu skasowania bitu błędu napędu. Ten bit jest ustawiany po usunięciu przyczyny błędu, aby wskazać, że napęd nie musi już sygnalizować poprzedniego błędu.
DIR	IN	BOOL	Sterowanie kierunkiem ruchu napędu – Ten bit jest ustawiany aby wskazać, że kierunek ruchu jest w przód (dla dodatniego SPEED_SP).
DRIVE	IN	USINT	Adres napędu: To wejście stanowi adres napędu USS. Poprawny zakres wynosi od driver 1 do driver 16.
PZD_LEN	IN	USINT	Długość słowa – Jest to liczba słów danych PZD. Poprawne wartości to 2, 4, 6 lub 8 słów. Domyślnie jest 2.
SPEED_ SP	IN	REAL	Nastawiona prędkość – Jest to prędkość ruchu napędu wyrażona w procentach skonfigurowanej częstotliwości. Wartość dodatnia oznacza ruch w przód (jeśli DIR jest TRUE).
CTRL3	IN	UINT	Słowo sterujące 3 – Wartość nadana konfigurowanemu przez użytkownika parametrowi napędu. Użytkownik musi skonfigurować ten parametr w napędzie. Parametr opcjonalny.
CTRL4	IN	UINT	Słowo sterujące 4 – Wartość nadana konfigurowanemu przez użytkownika parametrowi napędu. Użytkownik musi skonfigurować ten parametr w napędzie. Parametr opcjonalny.
CTRL5	IN	UINT	Słowo sterujące 5 – Wartość nadana konfigurowanemu przez użytkownika parametrowi napędu. Użytkownik musi skonfigurować ten parametr w napędzie. Parametr opcjonalny.
CTRL6	IN	UINT	Słowo sterujące 6 – Wartość nadana konfigurowanemu przez użytkownika parametrowi napędu. Użytkownik musi skonfigurować ten parametr w napędzie.
CTRL7	IN	UINT	Słowo sterujące 7 – Wartość nadana konfigurowanemu przez użytkownika parametrowi napędu. Użytkownik musi skonfigurować ten parametr w napędzie. Parametr opcjonalny.

Parametr	Typ parametru	Typ danych	Opis
CTRL8	IN	UINT	Słowo sterujące 8 – Wartość nadana konfigurowanemu przez użytkownika parametrowi napędu. Użytkownik musi skonfigurować ten parametr w napędzie. Parametr opcjonalny.
NDR	OUT	BOOL	Nowe dane gotowe – Stan TRUE tego bitu sygnalizuje, że na wyjściu są dane wynikające z nowego żądania komunikacji.
ERROR	OUT	BOOL	Wystąpił błąd – Stan TRUE tego wyprowadzenia sygnalizuje, że wystąpił błąd i stan wyjścia STATUS jest ważny. W przypadku błędu wszystkie inne wyjścia są ustawione na zero.
STATUS	OUT	UINT	Wartość statusu żądania. Sygnalizuje wynik cyklu programu. (Słowo 2 statusu napędu).
RUN_EN	OUT	BOOL	Ruch uaktywniony – Ten bit sygnalizuje czy odbywa się ruch napędu.
D_DIR	OUT	BOOL	Kierunek ruchu napędu – Ten bit sygnalizuje czy ruch napędu odbywa się w przód.
INHIBIT	OUT	BOOL	Ruch napędu wstrzymany – Ten bit sygnalizuje stan bitu Inhibit napędu.
FAULT	OUT	BOOL	Błąd napędu – Ten bit sygnalizuje, że napęd zarejestrował błąd. Użytkownik musi zlikwidować problem i następnie ustawić bit F_ACK w celu skasowania tego bitu jeśli jest ustawiony.
SPEED	OUT	REAL	Bieżąca prędkość ruchu napędu – Wartość prędkości ruchu napędu wyrażona w procentach skonfigurowanej prędkości.
STATUS1	OUT	UINT	Słowo 1 statusu napędu – Ta wartość zawiera ustalone bity stanu napędu.
STATUS3	OUT	UINT	Słowo 3 statusu napędu – Ta wartość zawiera konfigurowane przez użytkownika słowo stanu napędu.
STATUS4	OUT	UINT	Słowo 4 statusu napędu – Ta wartość zawiera konfigurowane przez użytkownika słowo stanu napędu.
STATUS5	OUT	UINT	Słowo 5 statusu napędu – Ta wartość zawiera konfigurowane przez użytkownika słowo stanu napędu.
STATUS6	OUT	UINT	Słowo 6 statusu napędu – Ta wartość zawiera konfigurowane przez użytkownika słowo stanu napędu.
STATUS7	OUT	UINT	Słowo 7 statusu napędu – Ta wartość zawiera konfigurowane przez użytkownika słowo stanu napędu.
STATUS8	OUT	UINT	Słowo 8 statusu napędu – Ta wartość zawiera konfigurowane przez użytkownika słowo stanu napędu.

6.3.1.3 Instrukcja USS_PORT

Instrukcja USS_PORT obsługuje komunikację w sieci USS. Zwykle stosuje się tylko jedną funkcję USS_PORT na moduł komunikacyjny PtP w programie i każde wywołanie tej instrukcji obsługuje transmisję do lub z pojedynczego napędu. Program użytkownika musi wykonywać funkcję USS_PORT dostatecznie często, by zapobiec przekroczeniu limitu czasu napędu. Wszystkie funkcje USS związane z jedną siecią USS i modułem komunikacyjnym PtP muszą korzystać z tej samej instancji bloku danych. W celu uniknięcia przekroczenia limitu czasu napędu i zapewnienia instrukcji USS_DRV najświeższych danych USS, instrukcja USS_PORT jest zwykle wywoływana z OB obsługi przerwania od opóźnienia.

LAD

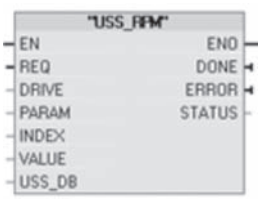
FBD

Parametr	Typ parametru	Typ danych	Opis
PORT	IN	PORT	Moduł komunikacyjny PtP. Identyfikator: To jest stała, która może być wskazana w zakładce „Constants” domyślnej tablicy tagów.
BAUD	IN	DINT	Szybkość transmisji wyrażona w bodach i wykorzystywana podczas komunikacji USS.
USS_DB	IN	DINT	To jest odniesienie do instancji DB utworzonej i zainicjalizowanej w momencie umieszczenia instrukcji USS_DRV w programie.
ERROR	OUTL	ERROR OUT BOOL	Stan TRUE tego wyprowadzenia sygnalizuje, że wystąpił błąd i stan wyjścia STATUS jest ważny.
STATUS	OUT	UINT	Wartość statusu żądania. Sygnalizuje wynik cyklu programu lub inicjalizacji.

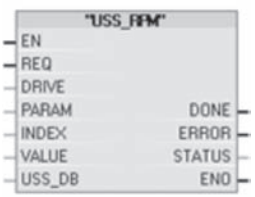
6.3.1.4 Instrukcja USS_RPM

Instrukcja USS_RPM odczytuje parametr z napędu. Wszystkie funkcje USS związane z jedną siecią USS i modułem komunikacyjnym PtP muszą korzystać z tego samego bloku danych. USS_RPM może być wywoływana z głównego OB.

LAD



FBD



Parametr	Typ parametru	Typ danych	Opis
REQ	IN	BOOL	Wysłanie żądania: Jeśli ma wartość TRUE, to sygnalizuje, że potrzebne jest wysłanie żądania odczytu nowych danych. Jeśli żądanie odczytu jest już realizowane, to jest ignorowane.
DRIVE	IN	USINT	Adres napędu: To wejście stanowi adres napędu USS. Poprawny zakres wynosi od driver 1 do driver 16.
PARAM	IN	UINT	Numer parametru: To wejście określa, który parametr napędu jest zapisywany. Zakres numerów parametru zawiera się od 0 do 2047. Sposób dostępu do parametrów o numerach przekraczających ten zakres jest podany w instrukcji napędu.
INDEX	IN	UINT	Indeks parametru: To wejście określa, który indeks parametru napędu ma zostać zapisany. Jest to 16-bitowa wartość, której najmniej znaczący bajt jest faktyczną wartością indeksu z zakresu od 0 do 255. Najbardziej znaczący bajt również może być użyty przez napęd i jest właściwy dla napędu. Szczegóły są podane w instrukcji napędu.
USS_DB	IN	VARIANT	To jest odniesienie do instancji DB utworzonej i zainicjalizowanej w momencie umieszczenia instrukcji USS_DRV w programie.
VALUE	IN	WORD, INT, UINT, DWORD, DINT, UDINT, REAL	To jest wartość parametru, który został odczytany i jest ważna tylko wtedy, kiedy bit DONE ma wartość TRUE.
DONE	OUT	BOOL	Wykonano: Jeśli ma wartość TRUE, to sygnalizuje, że wyjście VALUE przechowuje poprzednio żądaną, odczytaną wartość parametru. Ten bit jest ustawiany wtedy, kiedy USS_DRV stwierdzi, że nastąpił odczyt danej przesłanej z napędu. Ten bit jest kasowany wtedy, kiedy : • użytkownik zażąda odczytu danych poprzez zapytanie inną USS_RPM albo • nastąpi drugie z kolejnych dwóch wywołań USS_DRV.
ERROR	OUT	BOOL	Wystąpił błąd – Stan TRUE tego wyprowadzenia sygnalizuje, że wystąpił błąd i stan wyjścia STATUS jest ważny. W przypadku błędu wszystkie inne wyjścia są ustawione na zero.
STATUS	OUT	UINT	Wartość statusu żądania. Sygnalizuje wynik żądania odczytu.

6.3.1.5 Instrukcja USS_WPM

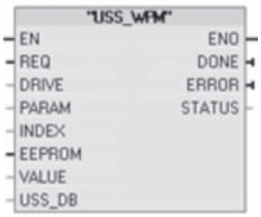
Instrukcja USS_WPM modyfikuje parametr w napędzie. Wszystkie funkcje USS związane z jedną siecią USS i modułem komunikacyjnym PtP muszą korzystać z tego samego bloku danych. USS_WPM może być wywoływana z głównego OB.

UWAGA

Operacje zapisu do pamięci EEPROM

Ostrzeżenie przed nadużywaniem operacji zapisu do pamięci EEPROM. W celu przedłużenia czasu życia pamięci należy minimalizować liczbę operacji zapisu do pamięci EEPROM.

LAD



FBD

Parametr	Typ parametru	Typ danych	Opis
REQ	IN	BOOL	Wysłanie żądania: Jeśli ma wartość TRUE, to sygnalizuje, że potrzebne jest wysłanie żądania odczytu nowych danych. Jeśli żądanie odczytu jest już realizowane, to jest ignorowane.
DRIVE	IN	USINT	Adres napędu: To wejście stanowi adres napędu USS. Poprawny zakres wynosi od driver 1 do driver 16.
PARAM	IN	UINT	Numer parametru: To wejście określa, który parametr napędu jest zapisywany. Zakres numerów parametru zawiera się od 0 do 2047. Sposób dostępu do parametrów o numerach przekraczających ten zakres jest podany w instrukcji napędu.
INDEX	IN	UINT	Indeks parametru: To wejście określa, który indeks parametru napędu ma zostać zapisany. Jest to 16-bitowa wartość, której najmniej znaczący bajt jest faktyczną wartością indeksu z zakresu od 0 do 255. Najbardziej znaczący bajt również może być użyty przez napęd i jest właściwy dla napędu. Szczegóły są podane w instrukcji napędu.

Parametr	Typ parametru	Typ danych	Opis
EEPROM	IN	BOOL	Zapis do pamięci EEPROM napędu: Jeśli ma wartość TRUE, to zapis parametru napędu będzie zapamiętany w pamięci EEPROM napędu. Jeśli ma wartość FALSE, to zapis parametru będzie tymczasowy i nie zostanie zachowany po cyklu wyłączania/włączania zasilania napędu.
VALUE	IN	WORD, INT, UINT, DWORD, DINT, UDINT, REAL	To jest wartość parametru, która ma być zapisana. Ta wartość musi być gotowa podczas zmiany sygnału REQ.
USS_DB	IN	VARIANT	To jest odniesienie do instancji DB utworzonej i zainicjalizowanej w momencie umieszczenia instrukcji USS_DRV w programie.
DONE	OUT	BOOL	Wykonano: Jeśli ma wartość TRUE, to sygnalizuje, że wejście VALUE zostało zapisane do napędu. Ten bit jest ustawiany wtedy, kiedy USS_DRV odbierze z napędu potwierdzenia zapisu. • użytkownik zażąda potwierdzenia wykonania zapisu poprzez zapytanie inną USS_RPM albo • nastąpi drugie z kolejnych dwóch wywołań USS_DRV.
ERROR	OUT	BOOL	Wystąpił błąd – Stan TRUE tego wyprowadzenia sygnalizuje, że wystąpił błąd i stan wyjścia STATUS jest ważny. W przypadku błędu wszystkie inne wyjścia są ustawione na zero.
STATUS	OUT	UINT	Wartość statusu żądania. Sygnalizuje wynik żądania zapisu.

6.3.1.6 Kody statusu USS

Kody statusu instrukcji USS są zwracane jako parametr funkcji USS

STATUS (W#16#....)	Opis
0000	Brak błędu.
8180	Długość odpowiedzi napędu nie jest zgodna ze znakami otrzymanymi z napędu.
8181	Parametr VALUE nie był typu Word, Real lub DWord.
8182	Użytkownik podał daną typu Word jako parametr, a w odpowiedzi otrzymał z napędu DWord lub Real.
8183	Użytkownik podał daną typu DWord lub Real jako parametr, a w odpowiedzi otrzymał z napędu Word.

STATUS (W#16#....)	Opis
8184	Suma kontrolna telegramu odpowiedzi z napędu jest błędna.
8185	Nieprawidłowy adres napędu (zakres poprawnych adresów napędów: 1-16).
8186	Nastawiona prędkość wykracza poza prawidłowy zakres (poprawny zakres SPEED_SP: -200% to 200%).
8187	Odpowiedź na wysłane żądanie nadeszła od napędu o błędnym numerze.
8188	Wyspecyfikowana długość słowa PZD jest nieprawidłowa (poprawny zakres = 2, 4, 6 lub 8 słów).
8189	Wyspecyfikowana wartość szybkości transmisji jest nieprawidłowa.
818A	Kanał żądania parametru jest zajęty przez inne żądanie skierowane do tego napędu.
818B	Napęd nie odpowiedział na żądanie i ponowienia żądania.
818C	W odpowiedzi na żądanie dotyczące parametru napęd zwrócił rozszerzony błąd. Por. opis rozszerzonych błędów poniżej tej tabeli.
818D	W odpowiedzi na żądanie dotyczące parametru napęd zwrócił błąd sygnalizujący nielegalny dostęp. Informacje dotyczące przyczyn ograniczenia dostępu są podane w instrukcji napędu.
818E	Napęd nie został zainicjalizowany: Ten kod błędu jest zwracany do USS_RPM lub USS_WPM wtedy, kiedy dla danego napędu nie została zastosowana przynajmniej raz instrukcja USS_DRV. Dzięki temu inicjalizacja pierwszym skanem USS_DRV nie nadpisuje będących w toku żądań odczytu lub zapisu parametru, ponieważ napęd jest inicjalizowany jako nowy element. W celu naprawy tego błędu należy wywołać dla napędu o danym numerze instrukcję USS_DRV.
80Ax-80Fx	Specyficzne błędy zwracane przez FB komunikacji PtP (Point-to-Point) wywołane z biblioteki USS: Te kody błędów nie są modyfikowane przez bibliotekę USS i są zdefiniowane w opisie instrukcji PtP.

Kody błędów rozszerzonych napędu USS

Napędy USS obsługują funkcje odczytu i zapisu wewnętrznych parametrów napędu. Ta cecha pozwala na zdalne sterowanie i konfigurowanie napędu. Ze względu na różne błędy, takie jak wartości spoza zakresu lub nielegalne dla aktualnego trybu pracy napędu żądania , operacje wymagające dostępu do parametrów napędu mogą nie zakończyć się powodzeniem. Napęd generuje wówczas kod błędu, który jest zwracany jako zmienna „USS_Extended_Error” instancji DB instrukcji USS_DRV. Ta wartość kodu jest ważna tylko dla ostatniego wykonania instrukcji USS_RPM lub USS_WPM. Kod błędu napędu jest umieszczany w zmiennej „USS_Extended_Error” wtedy, kiedy wartość kodu STATUS wynosi w zapisie szesnastkowym 808C. Wartość kodu zawarta w „USS_Extended_Error” zależy od modelu napędu. Opis rozszerzonych kodów błędów dla operacji zapisu i odczytu parametrów jest zamieszczony w instrukcji napędu.

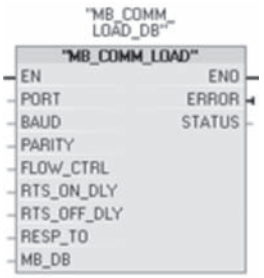
6.3.2 MODBUS

6.3.2.1 MB_COMM_LOAD

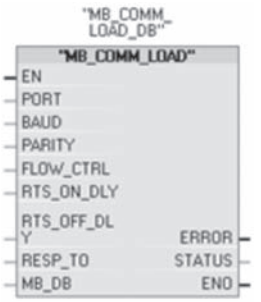
Opis

Instrukcja MB_COMM_LOAD konfiguruje port *Point-to-Point* (PtP) modułów CM 1241 RS485 lub CM 1241 RS232 zgodnie z protokołem komunikacyjnym Modbus RTU.

LAD



FBD



Parametr	Typ parametru	Typ danych	Opis
PORT	IN	UINT	Identyfikator portu komunikacyjnego: Po zainstalowaniu modułu CM w konfiguracji urządzenia, na rozwijanej liście pomocy dostępnej dla wyprowadzenia PORT bloku pojawia się identyfikator portu. Ta stała ma również swoje odniesienie w zakładce „Constants” domyślnej tablicy tagów.
BAUD	IN	UDINT	Wybór szybkości transmisji: 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 76800, 115200 Inne wartości są nieprawidłowe.
PARITY	IN	UINT	Wybór parzystości: • 0 – brak. – Nadawanie i odbiór: 1 bit startu, 8 bitów danych, brak parzystości, 1 bit stopu. • 1 – nieparzysty. • 2 – parzysty.
FLOW_CTRL	IN	UINT	Wybór sterowania przepływem: • 0 – (domyślnie) brak sterowania przepływem. • 1 – sterowanie sprzętowe z RTS zawsze ON (nie dotyczy portów RS485). • 2 – sterowanie sprzętowe, RTS przełączany.

Parametr	Typ parametru	Typ danych	Opis
RTS_ON_DLY	IN	UINT	Wybór opóźnienia RTS ON: • 0 – (domyślnie) brak opóźnienia od RTS aktywnego do nadania pierwszego znaku wiadomości. • 1 do 65535 – opóźnienie w milisekundach od RTS aktywnego do nadania pierwszego znaku wiadomości (nie dotyczy portów RS485). Opóźnienie RTS stosuje się niezależnie od wyboru FLOW_CTRL.
RTS_OFF_DLY	IN	UINT	Wybór opóźnienia RTS OFF: • 0 – (domyślnie) brak opóźnienia od ostatniego nadanego znaku do zmiany RTS na nieaktywny. • 1 do 65535 – opóźnienie w milisekundach od ostatniego nadanego znaku do zmiany RTS na nieaktywny (nie dotyczy portów RS485). Opóźnienie RTS stosuje się niezależnie od wyboru FLOW_CTRL.
RESP_TO	IN	UINT	Limit czasu odpowiedzi: Czas w milisekundach dozwolony przez MB_MASTER dla urządzenia Slave na odpowiedź. Jeżeli urządzenie Slave nie odpowie w tym czasie, to MB_MASTER ponowi żądanie lub – jeśli wyspecyfikowana liczba ponowień została wysłana - zakończy żądanie z błędem. 5 ms do 65535 ms (wartość domyślna = 1000 ms).
MB_DB	IN	VARIANT	Odniesienie do instancji bloku danych używanego przez instrukcje MB_MASTER lub MB_SLAVE. Po umieszczeniu instrukcji MB_MASTER lub MB_SLAVE w programie, na rozwijanej liście pomocy dostępnej dla wyprowadzenia MB_DB bloku pojawia się identyfikator DB.
ERROR	OUT	BOOL	Błąd: • 0 – błąd nie został wykryty. • 1 – został wykryty błąd i stan wyjścia STATUS podaje jego kod.
STATUS	OUT	UINT	Kod błędu konfiguracji portu.

Zasady komunikacji MODBUS

- Zanim instrukcje MB_SLAVE albo MB_MASTER będą się mogły komunikować z portem, musi być wykonana MB_COMM_LOAD w celu skonfigurowania portu.
- Jeżeli port ma odpowiadać jako *Slave* urządzeniu Modbus *Master*, to ten port nie może być użyty przez MB_MASTER. Z danym portem może być użyte tylko jedno wykonanie instrukcji MB_SLAVE.

- Jeżeli port ma być użyty do zainicjowania żądania Modbus *Master*, to ten port nie może być użyty przez MB_SLAVE. Z tym portem może być użyte jedno lub więcej wykonań MB_MASTER.
- Instrukcje Modbus nie korzystają z przerwań komunikacyjnych do kontrolowania procesu komunikacji. To program użytkownika musi sprawdzać czy instrukcje MB_MASTER lub MB *Slave* ukończyły nadawanie i odbiór.
- Jeśli program użytkownika działa jako Modbus *Slave*, to MB_SLAVE musi być wykonywana cykliczne, z częstością pozwalającą odpowiadać na czas na nadchodzące żądania z Modbus *Master*.
- Wszystkie wykonania MB_SLAVE należy wywoływać z OB przerwań cyklicznych.
- Jeśli program użytkownika działa jako Modbus *Master* i korzysta z MB_MASTER do wysyłania żądań do *Slave*, to musi kontynuować cykliczne wykonywanie MB_MASTER dopóty, dopóki *Slave* nie prześle odpowiedzi.
- Wszystkie wykonania MB_MASTER dotyczące danego portu należy wywoływać z tego samego OB (lub OB poziomu przerwań).

Opis działania

MB_COMM_LOAD jest wykonywana w celu konfiguracji portu dla protokołu Modbus RTU. Po skonfigurowaniu portu można nawiązać komunikację Modbus wykonując instrukcje MB_SLAVE lub MB_MASTER.

Jedna instancja MB_COMM_LOAD musi być wykorzystana do skonfigurowania każdego portu każdego modułu komunikacyjnego użytego do komunikacji Modbus. Dla każdego wykorzystywanego portu użytkownik musi przypisać każdej instrukcji MB_COMM_LOAD inną instancję bloku danych. CPU systemu S7-1200 może pracować co najwyżej z 3 modułami komunikacyjnymi.

Przypisanie instancji bloku danych następuje wtedy, kiedy w programie umieszczana jest instrukcja MB_MASTER lub MB_SLAVE. Ta instancja bloku danych otrzymuje referencję w momencie wyspecyfikowania parametru MB_DB instrukcji MB_COMM_LOAD.

Kody warunkowe

Wartość STATUS (W#16#....)	Opis
0000	Brak błędu.
8180	Nieprawidłowa wartość ID portu.
8181	Nieprawidłowa wartość szybkości transmisji.
8182	Nieprawidłowa wartość parzystości.
8183	Nieprawidłowa wartość sterowania przepływem.
8184	Nieprawidłowa wartość limitu czasu odpowiedzi.
8185	Nieprawidłowy wskaźnik do bloku danych Slave_PORT_n lub Master_PORT_n.

6.3.2.2 MB_MASTER

Opis

Instrukcja MB_MASTER umożliwia programowi użytkownika komunikację jako Modbus *Master* z wykorzystaniem portu *Point-to-Point* (PtP) modułów CM 1241 RS485 lub CM 1241 RS232. Program ma dostęp do danych w jednym lub więcej urządzeń Modbus *Slave*.

Umieszczenie instrukcji MB_MASTER w programie użytkownika powoduje przypisanie jej instancji bloku danych. Ta instancja otrzymuje nazwę w momencie wyspecyfikowania parametru MB_DB instrukcji MB_COMM_LOAD.

LAD

FBD

Parametr	Typ parametru	Typ danych	Opis
REQ	IN	BOOL	Wejście żądania: • 0 – brak żądania. • 1 – żądanie transmisji danych do jednego lub więcej Modbus Slave.
MB_ADR	IN	USINT	Adres stacji Modbus RTU: Poprawny zakres adresów: 0 do 247. Wartość 0 jest zarezerwowana dla rozgłaszania wiadomości do wszystkich urządzeń Modbus Slave. Kody funkcyjne Modbus 05, 06, 15 i 16 są jedynymi obsługiwanymi kodami funkcyjnymi dla rozgłaszania.
MODE	IN	USINT	Wybór trybu: Specyfikuje typ żądania: odczyt, zapis lub diagnostyka.
DATA_ ADDR	IN	UDIINT	Adres startowy w urządzeniu Slave: Określa adres początkowy danych dostępnych w urządzeniu Slave. Poprawne adresy są podane w tabeli operacji Modbus.
DATA_LEN	IN	UINT	Długość danych: Specyfikuje liczbę bitów lub słów dostępnych w związku z tym żądaniem. Poprawne długości są podane w tabeli operacji Modbus.

Parametr	Typ parametru	Typ danych	Opis
DATA_PTR	IN	VARIANT	Wskaźnik danych: Wskazuje na adres CPU DB, do którego dane mają zostać zapisane lub z którego mają być odczytane. DB musi być typu klasycznego DB. Por. uwagę na temat DATA_PTR zamieszczoną poniżej.
NDR	OUT	BOOL	Gotowość nowych danych: • 0 – Transakcja nie ukończona. • 1 – sygnalizuje, że instrukcja MB_ MASTER zakończyła żądaną transakcję z jednym lub wieloma urządzeniami Modbus Slave.
BUSY	OUT	BOOL	Zajętość: 0 – żadna transakcja MB_MASTER nie jest wykonywana. • 1 – transakcja MB_MASTER jest w trakcie wykonywania.
ERROR	OUT	BOOL	Błąd: • 0 – błąd nie został wykryty. • 1 – został wykryty błąd i stan wyjścia STATUS podaje jego kod.
STATUS	OUT	UINT	Kod błędu wykonania.

Zasady komunikacji Modbus Master

- Zanim instrukcja MB_MASTER będzie się mogła komunikować z portem, musi być wykonana MB_COMM_LOAD w celu skonfigurowania tego portu.
- Jeżeli port ma być użyty do zainicjowania żądania Modbus *Master*, to ten port nie może być użyty przez MB_SLAVE. Z tym portem może być użyte jedno lub więcej wykonań MB_MASTER.
- Instrukcje Modbus nie korzystają z przerwań komunikacyjnych do kontrolowania procesu komunikacji. To program użytkownika musi sprawdzać czy instrukcja MB_MASTER ukończyła nadawanie i odbiór.
- Jeśli program użytkownika działa jako Modbus *Master* i korzysta z MB_MASTER do wysyłania żądań do *Slave*, to musi kontynuować cykliczne wykonywanie MB_MASTER dopóty, dopóki *Slave* nie prześle odpowiedzi.
- Wszystkie wykonania MB_MASTER dotyczące danego portu należy wywoływać z tego samego OB (lub OB poziomu przerwań).

Parametr REQ

Wartość REQ równa FALSE = brak żądania.

Wartość REQ równa TRUE = żądanie przesłania danych do jednego lub więcej urządzeń Modbus *Slave*.

Ten sygnał wejściowy należy dostarczyć poprzez wyzwalany zboczem styk w czasie pierwszego wywołania MB_MASTER. Wyzwalany zboczem impuls uruchomi

jednokrotnie żądanie nadawania. Wszystkie wejścia są uchwycone i trzymane bez zmiany na czas jednego żądania i odpowiedzi wyzwolonych z tego wejścia.

Wewnętrznie, w celu zapewnienia, że żaden inny *Master* nie ma możliwości wystawienia żądania zanim to żądanie nie zostanie zakończone, MB_MASTER uruchamia maszynę stanu.

Ponadto, jeśli ta sama instancja wywołania FB MB_MASTER jest ponownie wykonywana z wejściem REQ = TRUE, zanim zostanie ukończone bieżące żądanie, to nie zostanie wykonana w związku z tym żadna transmisja. Jednakże, jak tylko bieżące żądanie zostanie zakończone, to zostanie wystawione nowe żądanie o ile MB_MASTER jest wykonany z wejściem REQ równym TRUE.

Wybór typu funkcji Modbus za pomocą parametrów DATA_ADDR i MODE

DATA_ADDR (adres startowy w *Slave*): Określa adres początkowy danych dostępnych w urządzeniu *Slave*.

MB_MASTER wykorzystuje wejście MODE, a nie wejście *Function Code* (kod funkcji). Kombinacja MODE i zakresu adresów Modbus określa *Function Code*, który jest używany w przesyłanej wiadomości Modbus. W poniższej tabeli przedstawiono zależność między parametrem MODE instrukcji MBUS_MASTER, kodem funkcji Modbus i zakresem adresów Modbus.

Funkcje Modbus MB_MASTER				
	Parametr DATA_ADDR adresy Modbus	Typ adresu	Parametr DATA_LEN długość danych Modbus	Funkcja Modbus
Tryb 0				
Odczyt	00001 do 09999	Bity wyjściowe	1 to 2000	01H
	10001 - 19999	Bity wejściowe	1 to 2000	02H
	30001 - 39999	Rejestry wejściowe	1 to 125	04H
	40001 do 49999 400001 do 465536 (rozszerzony)	Rejestry pamiętające	1 to 125	03H
Tryb 1				
Zapis	00001 do 09999	Bity wyjściowe	1 (pojedyncze słowo)	05H
	40001 do 49999 400001 do 465536 (rozszerzony)	Rejestry pamiętające	1 (pojedyncze słowo)	06H
	00001 do 09999	Bity wyjściowe	2 do 1968	15H
	40001 do 49999 400001 do 465536 (rozszerzony)	Rejestry pamiętające	2 do 123	16H

<table>
<tr><th colspan="5">Funkcje Modbus MB_MASTER</th></tr>
<tr><td colspan="5">Tryb 2</td></tr>
<tr><td colspan="5">Niektóre urządzenia Slave nie obsługują zapisu pojedynczego bitu lub słowa za pomocą funkcji Modbus 05H i 06H. W takim przypadku stosuje się Tryb 2 w celu wymuszenia zapisu pojedynczego bitu i słowa przy użyciu funkcji Modbus 15H i 16H.</td></tr>
<tr><td>Zapis</td><td>00001 do 09999</td><td>Bity wyjściowe</td><td>1 do 1968</td><td>15H</td></tr>
<tr><td></td><td>40001 do 49999
400001 do 465536
(rozszerzony)</td><td>Rejestry pamiętające</td><td>1 do 123</td><td>16H</td></tr>
<tr><td colspan="5">Tryb 11
• Odczytuje słowo licznika zdarzeń z urządzenia Modbus Slave, które jest wskazywane jako wejście dla MB_ADDR.
• W urządzeniu Modbus Slave S7-1200 ten licznik jest inkrementowany za każdym razem gdy urządzenie Slave poprawnie zapisze lub odczyta żądanie (ale nie rozgłaszane) z Modbus Master.
• Zwracana wartość jest pamiętana jako słowo pod adresem wyspecyfikowanym jako wejście do DATA_PTR.
• W tym trybie nie jest wymagana prawidłowa wartość DATA_LEN.</td></tr>
<tr><td colspan="5">Tryb 80
• Sprawdza status komunikacji urządzenia Modbus Slave, które jest wskazywane jako wejście dla MB_ADDR.
• Ustawienie bitu wyjściowego NDR instrukcji MB_MASTER sygnalizuje, że adresowane urządzenie Modbus Slave odpowiedziało odpowiednimi danymi.
• Żadne dane nie są zwracane do programu użytkownika.
• W tym trybie nie jest wymagana prawidłowa wartość DATA_LEN.</td></tr>
<tr><td colspan="5">Tryb 81
• Kasuje licznik zdarzeń (tak jak jest zwracany w trybie 11) w urządzeniu Modbus Slave, które jest wskazywane jako wejście dla MB_ADDR.
• Ustawienie bitu wyjściowego NDR instrukcji MB_MASTER sygnalizuje, że adresowane urządzenie Modbus Slave odpowiedziało odpowiednimi danymi.
• Żadne dane nie są zwracane do programu użytkownika.
• W tym trybie nie jest wymagana prawidłowa wartość DATA_LEN.</td></tr>
</table>

Parametr DATA_PTR

Wskazuje na lokalne źródło lub adres docelowy (adres w CPU S7-1200) danych odpowiednio zapisywanych lub odczytywanych. Użycie instrukcji MB_MASTER do utworzenia urządzenia Modbus *Master* wymaga stworzenia globalnego bloku danych do przechowywania danych odczytywanych z lub zapisywanych do urządzenia Modbus *Slave*.

UWAGA

Parametr DATA_PTR musi wskazywać blok danych typu klasycznego.

W celu utworzenia klasycznego, globalnego DB należy w trakcie dodawania nowego bloku danych wyczyścić pole wyboru „Symbolic address only”

Struktury bloku danych parametru DATA_PTR

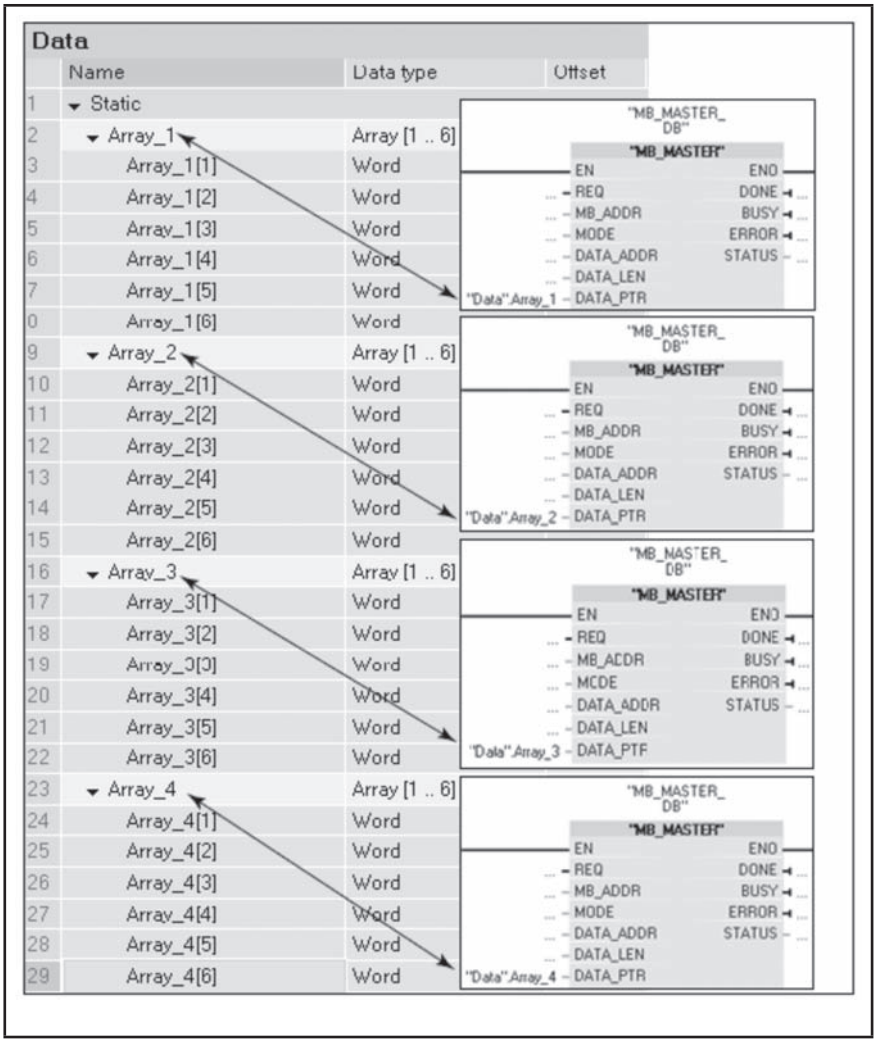
- Te typy danych obowiązują dla słów odczytanych z adresów Modbus 30001 do 39999, 40001 do 49999 i 400001 do 465536, a także dla słów zapisywanych do adresów Modbus 40001 do 49999 i 400001 do 465536.
  - Standardowe tablice danych typu WORD, UINT lub INT, tak jak pokazano je poniżej.
  - Nazwane struktury WORD, UINT lub INT, gdzie każdy element ma swoją unikalną nazwę i dane 16-bitowe.
  - Nazwane struktury złożone, gdzie każdy element ma swoją unikalną nazwę i dane 16- lub 32-bitowe.
- Dla bitów odczytywanych i zapisywanych spod adresów Modbus 00001 do 09999 i 10001 do 19999.
  - Standardowa tablica danych typu boolowskiego.
  - Nazwane struktury boolowskie o unikalnych nazwach zmiennych boolowskich.
- Mimo, że nie jest to wymagane, zaleca się by każda instrukcja MB_MASTER miała swój własny, oddzielny obszar w globalnym bloku danych. Przyczyną tego zalecenia jest to, że w przypadku gdy wiele instrukcji MB_MASTER zapisuje i odczytuje dane z tego samego obszaru globalnego bloku danych istnieje większa możliwość uszkodzenia danych.
- Nie ma wymagania by obszar danych DATA_PTR znajdował się tym samym globalnym bloku danych. Użytkownik może utworzyć jeden blok danych z wieloma obszarami dla operacji odczytu Modbus, jeden blok danych dla operacji zapisu Modbus lub jeden blok danych dla każdej stacji *Slave*.
- Wszystkie tablice w przykładzie przedstawionym poniżej są utworzone jako tablice typu base 1 [1....##]. Te tablice mogłyby być utworzone jako tablice typu base 0 [0...##] lub jako mieszane base 0 i base 1.

Przykład dostępu instrukcji MB_MASTER do globalnego bloku danych DATA_PTR

W celu spełnienia żądania Modbus przechowywania danych, przykładowy globalny blok danych pokazany poniżej wykorzystuje 4 unikalne nazwy tablic po 6 słów. Mimo, że tablice danych w tym przykładzie mają jednakowe rozmiary, to faktycznie mogą mieć dowolne rozmiary – ten sam rozmiar użyto tylko dla uproszczenia przykładu. Każda tablica może mieć również zamienioną strukturę danych na taką, która zawiera bardziej opisowe nazwy *tagów* i mieszane typy danych. Przykłady alternatywnych struktur danych są przedstawione w opisie parametru HR_DB instrukcji MB_SLAVE.

W przykładach dotyczących instrukcji MB_MASTER przedstawionych niżej, pokazano tylko parametr DATA_PTR pomijając inne wymagane parametry. Te przykłady mają na celu zilustrowanie w jaki sposób instrukcja MB_MASTER korzysta z bloku danych DATA_PTR.

Strzałki pokazują w jaki sposób każda tablica jest powiązana z różnymi instrukcjami MB_MASTER.



Pierwszym elementem dowolnej tablicy lub struktury jest zawsze pierwsze źródło lub miejsce docelowe dowolnej operacji Modbus odczytu lub zapisu. Wszystkie poniżej przedstawione scenariusze bazują na powyższym schemacie.

Scenariusz 1: Jeżeli pierwsza instrukcja MB_MASTER odczytuje 3 słowa danych spod adresu Modbus 40001 lub z dowolnego prawidłowego urządzenia Modbus *Slave*, to zachodzą następujące zdarzenia:

Słowo spod adresu 40001 jest zapamiętane w „Data”.Array_1[1].

Słowo spod adresu 40002 jest zapamiętane w „Data”.Array_1[2].

Słowo spod adresu 40003 jest zapamiętane w „Data”.Array_1[3].

Scenariusz 2: Jeżeli pierwsza instrukcja MB_MASTER odczytuje 4 słowa danych spod adresu Modbus 40015 lub z dowolnego prawidłowego urządzenia Modbus *Slave*, to zachodzą następujące zdarzenia:

Słowo spod adresu 40015 jest zapamiętane w „Data".Array_1[1].

Słowo spod adresu 40016 jest zapamiętane w „Data".Array_1[2].

Słowo spod adresu 40017 jest zapamiętane w „Data".Array_1[3].

Słowo spod adresu 40018 jest zapamiętane w „Data".Array_1[4].

Scenariusz 3: Jeżeli druga instrukcja MB_MASTER odczytuje 2 słowa danych spod adresu Modbus 30033 lub z dowolnego prawidłowego urządzenia Modbus *Slave*, to zachodzą następujące zdarzenia:

Słowo spod adresu 30033 jest zapamiętane w „Data".Array_2[1].

Słowo spod adresu 30034 jest zapamiętane w „Data".Array_2[2].

Scenariusz 4: Jeżeli trzecia instrukcja MB_MASTER zapisuje 4 słowa danych pod adres Modbus 40050 lub do dowolnego prawidłowego urządzenia Modbus *Slave*, to zachodzą następujące zdarzenia:

Słowo z „Data".Array_3[1] jest zapisane pod adres Modbus 40050.

Słowo z „Data".Array_3[2] jest zapisane pod adres Modbus 40051.

Słowo z „Data".Array_3[3] jest zapisane pod adres Modbus 40052.

Słowo z „Data".Array_3[4] jest zapisane pod adres Modbus 40053.

Scenariusz 5: Jeżeli trzecia instrukcja MB_MASTER zapisuje 3 słowa danych pod adres Modbus 40001 lub do dowolnego prawidłowego urządzenia Modbus *Slave*, to zachodzą następujące zdarzenia:

Słowo z „Data".Array_3[1] jest zapisane pod adres Modbus 40001.

Słowo z „Data".Array_3[2] jest zapisane pod adres Modbus 40002.

Słowo z „Data".Array_3[3] jest zapisane pod adres Modbus 40003.

Scenariusz 6: Jeżeli czwarta instrukcja MB_MASTER wykorzystuje Tryb 11 (wydobycie liczby ważnych wiadomości) z dowolnego prawidłowego urządzenia Modbus *Slave*, to zachodzą następujące zdarzenia:

Liczba słów jest zapamiętana w „Data".Array_4[1].

Przykład odczytu i zapisu bitu z wykorzystaniem lokalizacji WORD jako wejścia DATA_PTR

Tabela 6-1. Scenariusz 7: Odczyt 4 bitów wyjściowych począwszy od adresu Modbus 00001.

Wartości wejściowe MB_MASTER			**Wartości Modbus Slave**	
MB_ADDR	27 (przykładowy Slave)		00001	ON
MODE	0 (odczyt)		00002	ON
DATA_ADDR	00001 (wyjścia)		00003	OFF
DATA_LEN	4		00004	ON
DATA_PTR	„Data”.Array_4		00005	ON
			00006	OFF
			00007	ON
			00008	OFF

Wartości „Data”.Array_4[1] po żądaniu Modbus	
Najbardziej znaczący bajt (MS)	Najmniej znaczący bajt (LS)
xxxx-1011	xxxx-xxxx
x oznacza, że dane nie są zmienione	

Tablica 6-2. Scenariusz 8: Odczyt 12 bitów wyjściowych począwszy od adresu Modbus 00003.

Wartości wejściowe MB_ MASTER			**Wartości Modbus Slave**				
MB_ADDR	27 (przykładowy Slave)		00001	ON		00010	ON
MODE	0 (odczyt)		00002	ON		00011	OFF
DATA_ ADDR	00003 (Outputs)		00003	OFF		00012	OFF
DATA_LEN	12		00004	ON		00013	ON
DATA_PTR	„Data”.Array_4		00005	ON		00014	OFF
			00006	OFF		00015	ON
			00007	ON		00016	ON
			00008	ON		00017	OFF
			00009	OFF		00018	ON

Wartości „Data”.Array_4[1] po żądaniu Modbus	
Najbardziej znaczący bajt (MS)	Najmniej znaczący bajt (LS)
1011-0110	0100-xxxx
x oznacza, że dane nie są zmienione	

Tablica 6-3. Scenariusz 9: Zapis 5 bitów wyjściowych począwszy od adresu Modbus 00001.

Wartości wejściowe MB_ MASTER			**Wyjścia Slave przed**			**Wyjścia Slave po**
MB_ADDR	27 (przykładowy Slave)		00001	ON		OFF
MODE	1 (zapis)		00002	ON		ON
DATA_ ADDR	00001 (Outputs)		00003	OFF		ON
DATA_LEN	5		00004	ON		OFF
DATA_PTR	„Data”.Array_4		00005	ON		ON
			00006	OFF		Bez zmian
			00007	ON		Bez zmian
			00008	ON		Bez zmian
			00009	OFF		Bez zmian

Tablica 6-4. Scenariusz 10: Odczyt 22 bitów wyjściowych począwszy od adresu Modbus 00003.

Wartości wejściowe MB_ MASTER			**Wartości Modbus Slave**				
MB_ADDR	27 (przykładowy Slave)		00001	ON		00014	ON
MODE	0 (odczyt)		00002	ON		00015	OFF
DATA_ ADDR	00003 (Outputs)		00003	OFF		00016	ON
DATA_LEN	22		00004	ON		00017	ON
DATA_PTR	„Data”.Array_4		00005	ON		00018	OFF
			00006	OFF		00019	ON
			00007	ON		00020	ON
			00008	ON		00021	OFF
			00009	ON		00022	ON
			00010	OFF		00023	ON
			00011	OFF		00024	OFF
			00012	ON		00025	OFF
			00013	OFF		00026	ON

Wartości „Data”.Array_4[1] po żądaniu Modbus	
Najbardziej znaczący bajt (MS)	Najmniej znaczący bajt (LS)
0111-0110	0110-1010

Wartości „Data".Array_4[2] po żądaniu Modbus	
Najbardziej znaczący bajt (MS)	Najmniej znaczący bajt (LS)
xx01-1011	xxxx-xxxx
x oznacza, że dane nie są zmienione	

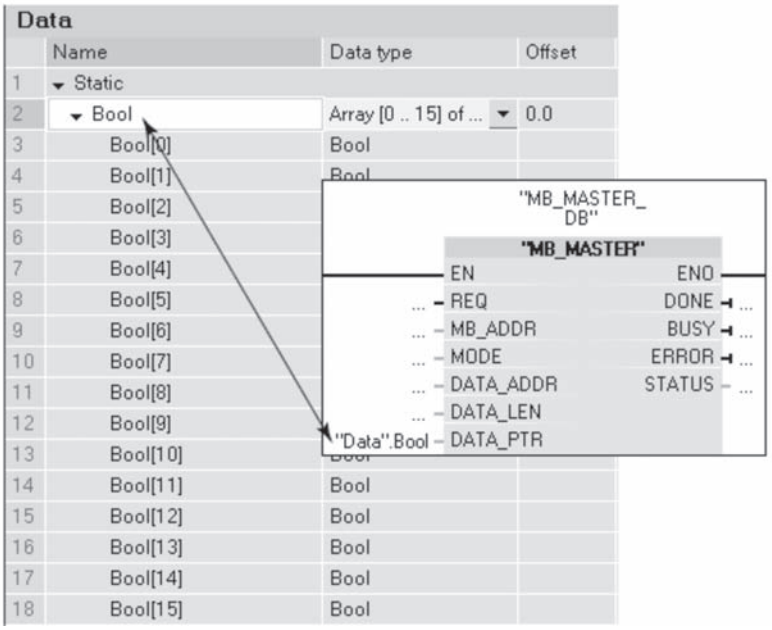
Przykład odczytu i zapisu bitu z wykorzystaniem lokalizacji BOOL jako wejścia DATA_PTR

Mimo, że zapisy i odczyty adresów, pod którymi są umieszczone bity mogą być przez Modbus obsługiwane poprzez traktowanie ich jak adresów słów, to w obszarach DATA_PTR mogą być również skonfigurowane dane typu boolowskiego, struktury lub tablice, co pozwala na bezpośrednie wskazanie pierwszego bitu, który jest czytany lub zapisywany instrukcją MB_MASTER.

Jeśli użytkownik stosuje struktury lub tablice boolowskie, to zaleca się by rozmiary danych były wielokrotnościami 8 bitów (w granicach bajtów). Na przykład, jeżeli zostanie utworzona tablica boolowska o wielkości 10 bitów, to oprogramowanie STEP 7 Basic alokuje w globalnym bloku danych 16 bitów (2 bajty) dla tych 10 bitów. Wewnątrz bloku danych będą one przechowywane jako bajt 1 [xxxx xxxx] i bajt 2 [---- --xx], gdzie „x" oznacza dostępne lokalizacje danych, a „-" wskazuje położenia, które nie są dostępne. Dozwolone są żądania Modbus dotyczące danych o długości do 16 bitów, ale górnych 6 bitów znajduje się w tym miejscu pamięci w bajcie 2, które nie jest ani wskazywane ani dostępne z programu użytkownika.

Obszary boolowskie mogą być tworzone jako tablice wartości boolowskich lub jako struktury zmiennych boolowskich. Obie te metody działają w identyczny sposób i różnią się tylko sposobem w jaki program użytkownika tworzy te obszary i w jaki sposób realizuje do nich dostęp.

Widok edytora globalnego bloku danych pokazany poniżej, przedstawia pojedynczą tablicę 16 wartości boolowskich utworzoną jako tablica typu *base 0*. Ta tablica mogłaby być również utworzona jako tablica typu *base 1*. Strzałki pokazują w jaki sposób ta tablica jest powiązana z instrukcją MB_MASTER.



Scenariusze 11 i 12 przedstawiają związek adresów Modbus z adresami tablicy boolowskiej.

Tablica 6-5. Scenariusz 11: Zapis 5 bitów wyjściowych począwszy od adresuModbus 00001.

Wartości wejściowe MB_ MASTER			**Wyjścia Slave przed**		**Dane DATA_PTR**	**Wyjścia Slave po**
MB_ADDR	27 (przykładowy Slave)		00001	ON	„Data”.Bool[0]=FALSE	OFF
MODE	1 (zapis)		00002	ON	„Data”.Bool[1]=FALSE	ON
DATA_ ADDR	00001 (Outputs)		00003	OFF	„Data”.Bool[2]=FALSE	ON
DATA_LEN	5		00004	ON	„Data”.Bool[3]=FALSE	OFF
DATA_PTR	„Data”.Bool		00005	ON	„Data”.Bool[4]=FALSE	OFF
			00006	OFF		Bez zmian
			00007	ON		Bez zmian
			00008	OFF		Bez zmian

Tablica 6-6. Scenariusz 12: Odczyt 15 bitów wyjściowych począwszy od adresu Modbus 00004.

Wartości wejściowe MB_ MASTER			**Wartości Modbus Slave**		**Dane DATA_PTR po**
MB_ADDR	27 (przykładowy Slave)		00001	ON	
MODE	0 (odczyt)		00002	ON	
DATA_ADDR	00003 (Outputs)		00003	OFF	„Data”.Bool[0]=FALSE
DATA_LEN	15		00004	ON	„Data”.Bool[1]=TRUE
DATA_PTR	„Data”.Bool		00005	ON	„Data”.Bool[2]=TRUE
			00006	OFF	„Data”.Bool[3]=FALSE
			00007	ON	„Data”.Bool[4]=TRUE
			00008	OFF	„Data”.Bool[5]= TRUE
			00009	ON	„Data”.Bool[6]=TRUE
			00010	OFF	„Data”.Bool[7]= FALSE
			00011	OFF	„Data”.Bool[8]=FALSE
			00012	ON	„Data”.Bool[9]=TRUE
			00013	OFF	„Data”.Bool[10]=FALSE
			00014	ON	„Data”.Bool[11]=TRUE
			00015	OFF	„Data”.Bool[12]= FALSE
			00016	ON	„Data”.Bool[13]= TRUE
			00017	ON	„Data”.Bool[14]=TRUE
			00018	OFF	
			00019	ON	

Kody warunkowe

Wartość STATUS (W#16#....)	**Opis**
0000	Brak błędu.
80C8	Limit czasu wyspecyfikowanej odpowiedzi (por. RCVTIME lub MSGTIME) wynosi 0.
80D1	W celu zawieszenia aktywnej transmisji odbiornik wystawił żądanie sterowania przepływem i nigdy ponownie nie uaktywnił transmisji w ustalonym czasie oczekiwania. Ten błąd jest również generowany podczas sprzętowego sterowania przepływem wtedy, kiedy odbiornik nie ustawił linii CTS w ustalonym czasie oczekiwania.
80D2	Żądanie transmisji zostało anulowane ponieważ z DCE nie nadszedł sygnał DSR.
80E0	Wiadomość została zakończona ponieważ bufor odbiorczy jest pełny.
80E1	Wiadomość została zakończona w wyniku błędu parzystości.

Wartość STATUS (W#16#....)	Opis
80E2	Wiadomość została zakończona w wyniku błędu ramki.
80E3	Wiadomość została zakończona w wyniku błędu przepełnienia.
80E4	Wiadomość została zakończona w wyniku tego, że wyspecyfikowana długość przekracza całkowity rozmiar bufora.
8180	Nieprawidłowa wartość ID portu.
8186	Nieprawidłowy adres stacji Modbus.
8188	Nieprawidłowa wartość Mode lub zastosowanie trybu zapisywania do obszaru adresowego Slave przeznaczonego tylko do odczytu.
8189	Nieprawidłowa wartość Data Address.
818A	Nieprawidłowa wartość Data Length.
818B	Nieprawidłowy wskaźnik do lokalnego źródła/odbiornika danych: Niepoprawny rozmiar.
818C	Wskaźnik do DB typu bezpiecznego typu DATA_PTR (musi to być klasyczny typ DB).
8200	Port jest zajęty przetwarzaniem żądania transmisji.

6.3.2.3 MB_SLAVE

Opis

Instrukcja MB_SLAVE umożliwia programowi użytkownika komunikację jako Modbus *Slave* z wykorzystaniem portu Point-to-Point (PtP) modułów CM 1241 RS485 lub CM 1241 RS232. Modbus RTU *Master* może wystawić żądanie i wtedy program użytkownika odpowiada wykonując MB_SLAVE.

Umieszczając instrukcję MB_SLAVE w swoim programie, użytkownik musi przypisać jej unikalną instancję bloku danych. Nazwy instancji bloku danych instrukcji MB_SLAVE używa się podczas specyfikacji parametru MB_DB instrukcji MB_COMM_LOAD.

Funkcje komunikacji Modbus (FC 1, 2, 4, 5 i 15) mogą czytać i zapisywać bity oraz słowa bezpośrednio z/do obrazu procesu PLC wejściowego i wyjściowego.

Funkcje Modbus MB_SLAVE						S7-1200	
FC	Funkcja	Obszar danych	Zakres adresów			Obszar danych	Adres CPU
01	Odczyt bitów	Wyjściowy	1	do	8192	Obraz procesu wyjściowy	Q0.0 do Q1023.7
02	Odczyt bitów	Wejściowy	10001	do	18192	Obraz procesu wejściowy	I0.0 do I1023.7
04	Odczyt słów	Wejściowy	30001	do	30512	Obraz procesu wejściowy	IW0 do IW1022
05	Zapis bitu	Wyjściowy	1	do	8192	Obraz procesu wyjściowy	Q0.0 do Q1023.7
15	Zapis bitów	Wyjściowy	1	do	8192	Obraz procesu wyjściowy	Q0.0 do Q1023.7

Funkcje komunikacji Modbus (FC 3, 6, 16) wykorzystują oddzielny i unikalny rejestr pamiętający bloku danych, który użytkownik musi utworzyć zanim będzie mógł wyspecyfikować parametr MB_HOLD_REG instrukcji MB_SLAVE.

Funkcje Modbus MB_SLAVE				S7-1200	
FC	Funkcja	Obszar danych	Zakres adresów	Obszar danych CPU DB	Adres CPU DB
03	Odczyt słów	Rejestr pamiętający	40001 do 4999	MB_HOLD_REG	Słowa 1 do 9999
			400001 do 465535		Słowa 1 do 65534
06	Zapis słowa	Rejestr pamiętający	4001 do 4999	MB_HOLD_REG	Słowa 1 do 9999
			400001 do 465535		Słowa 1 do 65534
16	Zapis słów	Rejestr pamiętający	4001 do 4999	MB_HOLD_REG	Słowa 1 do 9999
			400001 do 465535		Słowa 1 do 65534

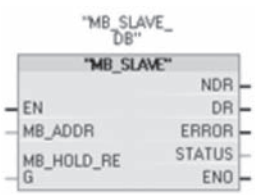
Funkcje diagnostyczne Modbus S7-1200 MB_SLAVE		
FC	Podfunkcja	Opis
08	0000H	Odpowiedź na test za pomocą echa: MB_SLAVE zwraca do Modbus Master echo otrzymanych danych.
08	000AH	Kasowanie licznika zdarzeń komunikacyjnych: MB_SLAVE skasuje licznik zdarzeń komunikacyjnych używany przez funkcję Modbus 11.
11		Odczytuje licznik zdarzeń komunikacyjnych: MB_SLAVE wykorzystuje wewnętrzny licznik zdarzeń komunikacyjnych do rejestrowania liczby pomyślnie zrealizowanych żądań Modbus zapisu i odczytu, które są przesyłane do Modbus Slave. Licznik nie zwiększa swojej zawartości podczas wykonywania funkcji 8, 11 i żądania rozgłaszania. Nie jest również inkrementowany przez żadne żądanie, które kończy się błędem (np. błędem parzystości lub CRC).

MB_SLAVE obsługuje rozgłaszane z dowolnego Modbus *Master* żądania zapisu, o ile żądanie dotyczy dostępu do poprawnej lokalizacji.

Niezależnie od ważności żądania, MB_SLAVE nie wysyła do Modbus *Master* żadnej odpowiedzi jako wyniku rozgłaszanego żądania.

LAD

FBD



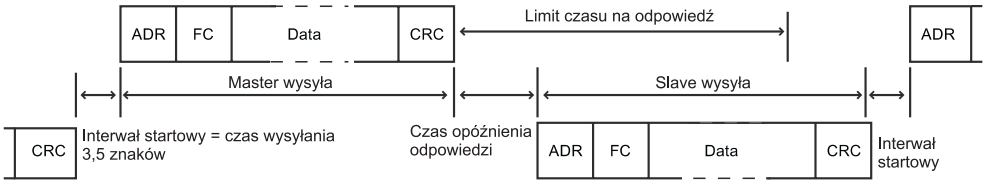
Parametr	Typ parametru	Typ danych	Opis
MB_ADDR	IN	USINT	Adres Modbus RTU (1 do 247):
Adres stacji Modbus Slave.			
MB_HOLD_REG	IN	VARIANT	Wskaźnik do DB rejestru pamiętającego Modbus. DB rejestru pamiętającego musi być klasycznym, globalnym DB. Por. poniższą uwagę dotycząca MB_ HOLD_REG.
NDR	OUT	BOOL	Gotowość nowych danych: • 0 – Brak nowych danych. • 1 – sygnalizuje, że nowe dane zostały zapisane przez Modbus Master.
DR	OUT	BOOL	Odczyt danych: • 0 – Dane nie odczytane. • 1 – sygnalizuje, że dane zostały odczytane przez Modbus Master.
ERROR	OUT	BOOL	Błąd: • 0 – błąd nie został wykryty. • 1 – został wykryty błąd i parametr STATUS przechowuje jego kod.
STATUS	OUT	UINT	Kod błędu.

Zasady komunikacji Modbus Slave

- Zanim instrukcja MB_SLAVE będzie się mogła komunikować z portem, musi być wykonana MB_COMM_LOAD w celu skonfigurowania tego portu.
- Jeżeli port ma odpowiadać jak Slave urządzeniu Modbus *Master*, to nie może być on użyty przez MB_MASTER. Z tym portem może być użyte tylko jedno wykonanie MB_SLAVE.
- Instrukcje Modbus nie korzystają z przerwań komunikacyjnych do kontrolowania procesu komunikacji. To program użytkownika musi sprawdzać czy instrukcja MB_SLAVE ukończyła nadawanie i odbiór.
- Jeśli program użytkownika działa jako Modbus *Slave*, to MB_SLAVE musi być wykonywana cykliczne, z częstością pozwalającą odpowiadać na czas na nadchodzące żądania z Modbus *Master*.
- Wszystkie wykonania MB_SLAVE dotyczące danego portu należy wywoływać z OB przerwań cyklicznych.

Opis działania

MB_SLAVE musi być wykonywana cyklicznie aby otrzymać każde żądanie z Modbus *Master* i odpowiednio na nie odpowiedzieć. Częstotliwość wykonywania MB_SLAVE zależy od limitu czasu na odpowiedź do Modbus *Master*. Ilustruje to poniższy wykres.



Limit czasu na odpowiedź jest to czas, przez który Modbus *Master* oczekuje na rozpoczęcie odpowiedzi z Modbus *Slave*. Ten czas nie jest definiowany w protokole Modbus, ale jest parametrem każdego urządzenia Modbus *Master*. Częstotliwość wykonywania (czas pomiędzy jednym wykonaniem i następnym) MB_SLAVE musi być związany z parametrami konkretnego urządzenia Modbus *Master*. Jako minimum, MB_SLAVE należy wykonywać dwukrotnie w limicie czasu przeznaczonym na odpowiedź.

Przykłady parametru MB_HOLD_REG

MB_HOLD_REG jest wskaźnikiem do rejestru pamiętającego bloku danych Modbus. Ten DB jest wykorzystywany do pamiętania wartości danych, do których Modbus *Master* ma dozwolony dostęp (zapis i odczyt). Użytkownik musi utworzyć ten DB i określić jego strukturę danych zanim instrukcja MB_SLAVE będzie mogła go użyć.

UWAGA

Rejestr pamiętający bloku danych Modbus musi być utworzony jako klasyczny, globalny DB.

W celu utworzenia klasycznego, globalnego DB należy w trakcie dodawania nowego bloku danych wyczyścić pole wyboru „Symbolic address only”.

Rejestry pamiętające mogą wykorzystywać następujące struktury danych DB:

- Standardowa tablica słów.
- Nazwana struktura słowa.
- Nazwana struktura złożona.

W przedstawionych poniżej przykładach programów pokazano w jaki sposób wykorzystywać parametr MB_HOLD_REG do obsługi tych struktur danych DB.

Przykład 1 – Standardowa tablica słów

Przykładowy rejestr pamiętający jest tablicą słów. Przypisany typ danych można zmienić na słowa innego typu i rozmiaru (INT i UINT).

Zalety:

- Ten typ struktury rejestru pamiętającego jest bardzo łatwo i szybko tworzony.
- Logika programu obsługującego dostęp do elementu danych jest uproszczona.

Wady:

- Mimo, że można programowo wskazać każdy element tablicy używając jego symbolicznej nazwy („HR_DB”.”Array”[1] do „HR_DB”.”Array”[10]), to nazwy nie opisują wewnętrznej funkcji danych.
- Tablica może się składać z danych tylko jednego typu. W programie użytkownika może być wymagana konwersja typu z zachowaniem sztywnej kontroli.

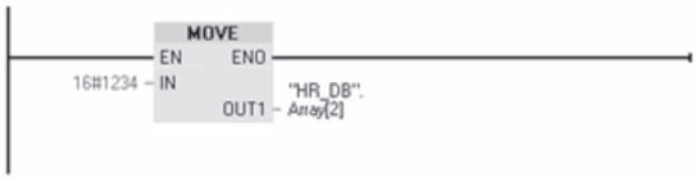
Poniżej pokazano w jaki sposób struktura słów jest przedstawiana w edytorze bloku danych.

HR_DB

	Name	Data type	Offset	Default value	Initial value	Retain	Comment
1	Static						
2	Array	Array [1..10] of W..	0.0			☐	40001 to 40010
3	Array[1]	Word		W#16#0000	W#16#0000		
4	Array[2]	Word		W#16#0000	W#16#0000		
5	Array[3]	Word		W#16#0000	W#16#0000		
6	Array[4]	Word		W#16#0000	W#16#0000		
7	Array[5]	Word		W#16#0000	W#16#0000		
8	Array[6]	Word		W#16#0000	W#16#0000		
9	Array[7]	Word		W#16#0000	W#16#0000		
10	Array[8]	Word		W#16#0000	W#16#0000		
11	Array[9]	Word		W#16#0000	W#16#0000		
12	Array[10]	Word		W#16#0000	W#16#0000		

Na poniższym rysunku zilustrowano sposób w jaki tablica jest przypisana wejściu MB_HOLD_REG instrukcji MB_SLAVE.

Dostęp do każdego elementu tablicy jest możliwy poprzez jego symboliczną nazwę, tak jak to pokazano niżej. W tym przykładzie nowa wartość jest przesunięta do drugiego elementu tablicy, który odpowiada adresowi Modbus 40002.



Każdemu słowu tablicy, tak jak to jest zdefiniowane w bloku danych, odpowiada instrukcja Modbus z adresem rejestru pamiętającego. W tym przypadku, ponieważ istnieje tylko 10 elementów tablicy, więc jest dostępnych tylko 10 adresów rejestru pamiętającego Modbus, które mogą być użyte z instrukcją MB_SLAVE i dostępne z Modbus *Master*.

Korelacja nazw elementów tablicy z adresami Modbus jest przedstawiona poniżej.

„HR_DB".Array[1]	Adres Modbus 40001
„ HR_DB „. Array[2]	Adres Modbus 40002
„ HR_DB „. Array[3]	Adres Modbus 40003
...	
...	
„ HR_DB „. Array[9]	Adres Modbus 40009
„ HR_DB „.Array [10]	Adres Modbus 40010

Przykład 2 – Nazwana struktura słowa

W tym przykładzie rejestr pamiętający jest utworzony z ciągu słów o opisowych nazwach symbolicznych.

Zalety:

- Każdy element struktury ma opisową nazwę symboliczną z przypisanym jej specyficznym typem danej.

Wady:

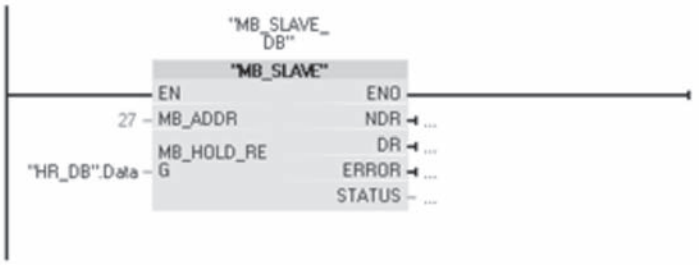
- Stworzenie struktury tego typu trwa dłużej niż standardowej tablicy słów.
- Elementy wymagają dodatkowych nazw symbolicznych jeśli są używane w programie użytkownika. O ile pierwszy element standardowej tablicy nosi oznaczenie „HR_DB".Array[0], to pierwszy element tego typu jest wskazywany jako „HR_DB".Data.Temp_1.

Poniżej pokazano w jaki sposób nazwana struktura słowa pojawia się w edytorze bloku danych. Każdy element ma swoją unikalną nazwę i może być typu WORD, UINT lub INT.

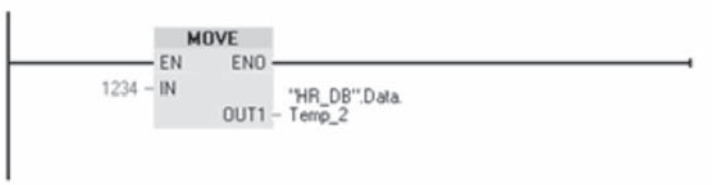
HR_DB

	Name	Data type	Offset	Initial value	Retain	Comment
1	Static					
2	Data	Struct	0.0			
3	Temp_1	Int	0.0	0		40001
4	Temp_2	Int	2.0	0		40002
5	Temp_3	Int	4.0	0		40003
6	Good_Count	UInt	6.0	0		40004
7	Bad_Count	UInt	8.0	0		40005
8	Rework_Count	UInt	10.0	0		40006
9	Line_Stops	UInt	12.0	0		40007
10	Avg_Time	UInt	14.0	0		40008
11	Code_1	Word	16.0	0		40009
12	Code_2	Word	18.0	0		40010

Na rysunku poniżej zilustrowano sposób w jaki powyższa struktura jest przypisana w programie użytkownika wejściu MB_HOLD_REG instrukcji MB_SLAVE.



Dostęp do każdego elementu tablicy jest możliwy poprzez jego symboliczną nazwę, tak jak to pokazano niżej. W tym przykładzie nowa wartość jest przesunięta do drugiego elementu tablicy, który odpowiada adresowi Modbus 40002.



Korelacja nazw elementów danych z adresami Modbus jest przedstawiona poniżej.

„HR_DB".Data.Temp_1	Adres Modbus 40001
„HR_DB".Data.Temp_2	Adres Modbus 40002
„HR_DB".Data.Temp_3	Adres Modbus 40003
„HR_DB".Data.Good_Count	Adres Modbus 40004
„HR_DB".Data.Bad_Count	Adres Modbus 40005
„HR_DB".Data.Rework_Count	Adres Modbus 40006
„HR_DB".Data.Line_Stops	Adres Modbus 40007
„HR_DB".Data.Avg_Time	Adres Modbus 40008
„HR_DB".Data.Code_1	Adres Modbus 40009
„HR_DB".Data.Code_2	Adres Modbus 40010

Przykład 3 – Nazwana struktura złożona

W tym przykładzie rejestr pamiętający jest utworzony z ciągu danych typu mieszanego o opisowych nazwach symbolicznych.

Zalety:

- Każdy element struktury ma opisową nazwę symboliczną z przypisanym jej specyficznym typem danej.
- Możliwy jest bezpośredni transfer typów danych nie bazujących na słowach.

Wady:

- Stworzenie struktury tego typu trwa dłużej niż standardowej tablicy słów.
- Modbus *Master* musi być skonfigurowany tak, by akceptować dane jakie będzie otrzymywał z Modbus *Slave*. Jak to pokazano na rysunku poniżej, Temp_1 jest 4-bajtową wartością rzeczywistą. *Master*, który ją odbiera musi być zdolny do złożenia dwóch przesłanych mu słów w jedną oczekiwaną wartość rzeczywistą.
- Elementy wymagają dodatkowych nazw symbolicznych jeśli są używane w programie użytkownika. O ile pierwszy element standardowej tablicy nosi oznaczenie „HR_DB".Array[0], to pierwszy element tego typu jest wskazywany jako „HR_DB".Data.Temp_1.

Poniżej pokazano w jaki sposób nazwana struktura złożona pojawia się w edytorze bloku danych. Każdy element ma swoją unikalną nazwę i może być innego typu i rozmiaru od pozostałych.

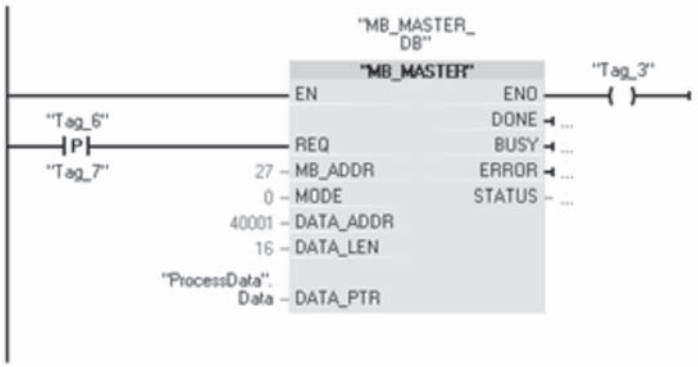
HR_DB

	Name	Data type	Offset	Initial value	Retain	Comment
1	Static					
2	Data	Struct	0.0			Modbus addresses 40001 to 400016
3	Temp_1	Real	0.0	0.0		40001 and 40002
4	Temp_2	Real	4.0	0.0		40003 and 40004
5	Good_Count	UDint	8.0	0		40005 and 40006
6	Bad_Count	UDint	12.0	0		40007 and 40008
7	Rework_Count	UDint	16.0	0		40009 and 40010
8	Line_Stops	Int	20.0	0		40011
9	Avg_Time	Int	22.0	0		40012
10	Long_Code	DWord	24.0	0		40013 and 40014
11	Code_1	Word	28.0	0		40015
12	Code_2	Word	30.0	0		40016

Korelacja nazw elementów danych z adresami Modbus jest przedstawiona poniżej.

„HR_DB".Data.Temp_1	Adresy Modbus 40001 i 40002
„HR_DB".Data.Temp_2	Adresy Modbus 40003 i 40004
„HR_DB".Data.Good_Count	Adresy Modbus 40005 i 40006
„HR_DB".Data.Bad_Count	Adresy Modbus 40007 i 40008
„HR_DB".Data.Rework_Count	Adresy Modbus 40009 i 40010
„HR_DB".Data.Line_Stops	Adres Modbus 400011
„HR_DB".Data.Avg_Time	Adres Modbus 400012
„HR_DB".Data.Long_Code	Adresy Modbus 40013 i 40014
„HR_DB".Data.Code_1	Adres Modbus 40015
„HR_DB".Data.Code_2	Adres Modbus 40016

Inny CPU S7-1200 pracujący jako Modbus *Master* może korzystać z instrukcji Modbus *Master* i identycznej struktury danych w celu otrzymywania danych z CPU S7-1200 pracującego jako Modbus *Slave*. Instrukcja Modbus *Master* skopiuje wszystkie 16 słów danych bezpośrednio z HR_DB bloku danych urządzenia *Slave* do bloku danych *ProcessData* urządzenia *Master*, tak jak to pokazano poniżej.



ProcessData

	Name	Data type	Offset	Initial value	Retain	Comment
1	Static					
2	Data	Struct	0.0			Modbus addresses 40001 to 400016
3	Temp_1	Real	0.0	0.0		40001 and 40002
4	Temp_2	Real	4.0	0.0		40003 and 40004
5	Good_Count	UDInt	8.0	0		40005 and 40006
6	Bad_Count	UDInt	12.0	0		40007 and 40008
7	Rework_Count	UDInt	16.0	0		40009 and 40010
8	Line_Stops	Int	20.0	0		40011
9	Avg_Time	Int	22.0	0		40012
10	Long_Code	DWord	24.0	0		40013 and 40014
11	Code_1	Word	28.0	0		40015
12	Code_2	Word	30.0	0		40016

Do transferowania takich samych lub różnych struktur z wielu urządzeń Modbus *Slave* może być wykorzystany ciąg lokalizacji Data_PTR bloku danych urządzenia Modbus *Master*.

Kody warunkowe

Wartość STATUS (W#16#....)	Opis
80C8	Limit czasu wyspecyfikowanej odpowiedzi (por. RCVTIME lub MSGTIME) wynosi 0.
80D1	W celu zawieszenia aktywnej transmisji odbiornik wystawił żądanie sterowania przepływem i nigdy ponownie nie uaktywnił transmisji w ustalonym czasie oczekiwania. Ten błąd jest również generowany podczas sprzętowego sterowania przepływem wtedy, kiedy odbiornik nie dokonał asercji CTS w ustalonym czasie oczekiwania.
80D2	Żądanie transmisji zostało anulowane ponieważ z DCE nie nadszedł sygnał DSR.

Wartość STATUS (W#16#....)	**Opis**	
80E0	Wiadomość została zakończona ponieważ bufor odbiorczy jest pełny.	
80E1	Wiadomość została zakończona w wyniku błędu parzystości.	
80E2	Wiadomość została zakończona w wyniku błędu ramkowania.	
80E3	Wiadomość została zakończona w wyniku błędu przepełnienia.	
80E4	Wiadomość została zakończona w wyniku tego, że wyspecyfikowana długość przekracza całkowity rozmiar bufora.	
8180	Nieprawidłowa wartość ID portu.	
8186	Nieprawidłowy adres stacji Modbus.	
8187	Nieprawidłowy wskaźnik do MB_HOLD_REG DB.	
818C	Wskaźnik do DB typu bezpiecznego typu DATA_PTR (musi to być klasyczny typ DB).	
	Kod odpowiedzi wysyłany do Modbus Master (B#16#..)	
8380	Brak odpowiedzi	Błąd CRC.
8381	01	Nieobsługiwany kod funkcji.
8382	Brak odpowiedzi	Błąd długości danych.
8383	02	Błąd adresu danych.
8384	03	Błąd wartości danych.
8385	03	Nieobsługiwana wartość kodu diagnostyki danych (kod funkcji 08).

PROFINET 7

CPU S7-1200 ma zintegrowany port PROFINET obsługujący zarówno Ethernet, jak i standardy komunikacyjne oparte na TCP/IP. CPU S7-1200 obsługuje następujące protokoły aplikacyjne:

- Transport Connection Protocol (TCP).
- ISO Transport over TCP (RFC 1006).

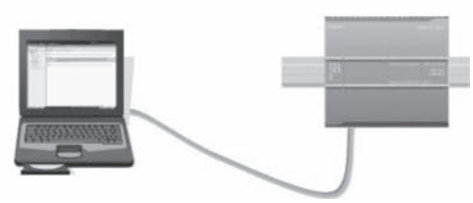
CPU S7-1200 ma możliwość komunikacji z następującymi urządzeniami:

- Innymi CPU S7-1200.
- Urządzeniem programującym STEP 7 Basic.
- Urządzeniami HMI.
- Urządzeniami produkcji firm innych niż Siemens, korzystających ze standardowych protokołów komunikacyjnych TCP (nadawanie bloków (T-block)).

Są dwie metody komunikacji z wykorzystaniem sieci PROFINET:

- Połączenie bezpośrednie: Wykorzystywane wtedy, kiedy jedno urządzenie programujące, HMI lub inny CPU jest podłączony do pojedynczego CPU.
- Połączenie sieciowe: Wykorzystywane wtedy, kiedy są połączone więcej niż dwa urządzenia (na przykład, kilka CPU, kilka HMI, urządzenia programujące i urządzenia innych firm).

Połączenie bezpośrednie: Urządzenie programujące podłączone do CPU S7-1200.



Połączenie bezpośrednie: HMI podłączony do CPU S7-1200.

Połączenie bezpośrednie: CPU S7-1200 podłączony do innego CPU S7-1200.

Połączenie sieciowe: Więcej niż dwa urządzenie połączone ze sobą.

① CSM1277 Ethernet Switch

UWAGA

Przełącznik ethernetowy nie jest wymagany przy połączeniu bezpośrednim urządzenia programującego lub HMI i CPU. Przełącznik ethernetowy jest wymagany przy połączeniu więcej niż dwóch CPU lub HMI.

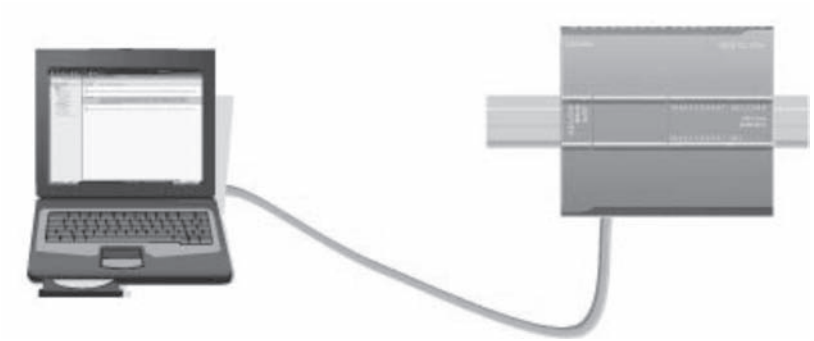
UWAGA

Do połączenia kilku CPU i HMI można wykorzystać 4-portowy przełącznik ethernetowy firmy Siemens typu CSM1277. Port PROFINET wbudowany do CPU S7-1200 nie jest wyposażony w przełącznik ethernetowy.

7.1 Komunikacja z komputerem programującym

CPU może się komunikować z urządzeniem programującym STEP 7 Basic z wykorzystaniem sieci. Podczas przygotowywania komunikacji między CPU i urządzeniem programującym STEP 7 Basic należy rozważyć następujące czynniki:

- Konfiguracja/ustawienia: Wymagana jest konfiguracja sprzętowa.
- Przy komunikacji „jeden do jednego” nie jest wymagany przełącznik ethernetowy; przełącznik ethernetowy jest konieczny wtedy, kiedy w sieci są połączone więcej niż dwa urządzenia.

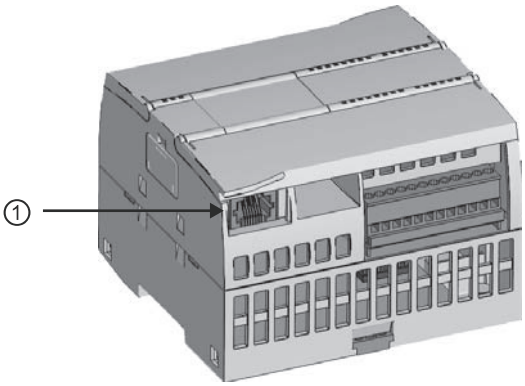


7.1.1 Zestawianie połączenia komunikacyjnego

Interfejsy PROFINET tworzą fizyczne połączenie między urządzeniem programującym i CPU. Ponieważ CPU ma wbudowaną funkcję Auto-Cross-Over, więc w celu realizacji połączenia można zastosować kabel ethernetowy prosty lub skrosowany. Przełącznik ethernetowy nie jest konieczny by urządzenie programujące podłączyć bezpośrednio do CPU.

W celu stworzenia fizycznego połączenia między urządzeniem programującym i CPU należy wykonać następujące kroki:

1. Zainstalować CPU.
2. Podłączyć kabel ethernetowy do gniazda portu PROFINET (rysunek poniżej).
3. Podłączyć kabel ethernetowy do urządzenia programującego.



① port PROFINET

W celu zwiększenia odporności na uszkodzenia mechaniczne kabla PROFINET należy zastosować opcjonalny element redukujący naprężenia kabla.

7.1.2 Konfiguracja urządzenia

Jeżeli jest już stworzony projekt zawierający CPU, to należy otworzyć ten projekt w portalu TIA.

Jeśli nie, to należy utworzyć projekt i dołączyć do niego CPU. Poniżej przedstawiono projekt, w którym CPU jest pokazany w oknie „Device View" TIA Portal.

7.1.3 Nadawanie adresów IP

7.1.3.1 Nadawanie adresów IP urządzeniom programującym i sieciowym

Jeżeli urządzenie programujące wykorzystuje zainstalowaną kartę adaptera (na przykład, PRO/1000 MT Network Connection firmy Intel(r)) podłączoną do sieci LAN obiektu (która zapewne jest połączoną z siecią światową), to identyfikatory sieci adresu IP oraz maski podsieci CPU i karty adaptera urządzenia programującego muszą być dokładnie takie same. Identyfikator sieci jest pierwszą częścią adresu IP (pierwsze trzy oktety, na przykład **211**.**154**.**184**.**16**), oznaczającą w jakiej sieci IP znajduje się urządzenie. Maska podsieci zwykle ma wartość **255**.**255**.**255**.**0**; jednakże, ponieważ komputer użytkownika znajduje się w sieci LAN obiektu, więc w celu określenia unikalnych podsieci, maska podsieci może mieć różne wartości (na przykład, **255**.**255**.**254**.**0**). Maska podsieci połączona z adresem IP urządzenia za pomocą logicznej operacji AND, definiuje granice IP podsieci.

UWAGA

W przypadku sieci światowej (WWW), w której urządzenia programujące, urządzenia sieciowe i routery IP komunikują się z całym światem, w celu uniknięcia konfliktu z innymi użytkownikami sieci przyznawane adresy IP muszą być unikalne. W celu uzyskania informacji o dostępnych adresach IP należy się skontaktować z pracownikami lokalnego działu IT zaznajomionymi z siecią obiektu.

Jeżeli urządzenie programujące wykorzystuje kartę adaptera Ethernet – USB (na przykład DLINK DUB E100 USB 2.0 Fast Ethernet Adapter [TCP/IP]) podłączoną do sieci wydzielonej (odizolowanej), to identyfikatory sieci adresu IP oraz maski podsieci CPU i karty adaptera Ethernet – USB urządzenia programującego muszą być dokładnie takie same. Identyfikator sieci jest pierwszą częścią adresu IP (pierwsze trzy oktety) (na przykład **211**.**154**.**184**.**16**), oznaczającą w jakiej sieci IP znajduje się urządzenie. Maska podsieci zwykle ma wartość **255**.**255**.**255**.**0**. Maska podsieci połączona z adresem IP urządzenia za pomocą logicznej operacji AND, definiuje granice IP podsieci.

UWAGA

Karta adaptera Ethernet – USB (na przykład, D-LINK DUB E100 USB 2.0 Fast Ethernet

Adapter [TCP/IP]) jest przydatna wtedy, kiedy użytkownik nie chce podłączyć CPU do firmowej sieci lokalnej. Taka konfiguracja jest szczególnie przydatna podczas początkowych testów lub w trakcie testów związanych z odbiorem systemu.

Karta adaptera urządzenia programującego	Rodzaj sieci	Adres IP (Internet Protocol)	Maska podsieci
Karta adaptera zainstalowana w urządzeniu (na przykład Intel(R) PRO/1000 MT Network Connection).	Sieć LAN obiektu (możliwe, że połączona z siecią światową).	ID sieci dla CPU i karty adaptera w urządzeniu programującym muszą być dokładnie takie same. ID sieci jest pierwszą częścią adresu IP (pierwsze trzy oktety) (na przykład 211.154.184.16), oznaczającą w jakiej sieci IP znajduje się urządzenie.	Maski podsieci CPU i karty adaptera w urządzeniu programującym muszą być dokładnie takie same. Maska podsieci zwykle ma wartość 255.255.255.0; jednakże, ponieważ komputer użytkownika znajduje się w sieci LAN obiektu, więc w celu określenia unikalnych podsieci, maska podsieci może mieć różne wartości (na przykład, 255.255.254.0). Maska podsieci połączona z adresem IP urządzenia za pomocą logicznej operacji AND, definiuje granice IP podsieci.
Karta adaptera Ethernet – USB (na przykład, DLINK DUB E100 USB 2.0 Fast Ethernet Adapter [TCP/IP]).	Sieć wydzielona (odizolowana).	ID sieci dla CPU i karty adaptera Ethernet – USB urządzenia programującego muszą być dokładnie takie same. ID sieci jest pierwszą częścią adresu IP (pierwsze trzy oktety) (na przykład 211.154.184.16), oznaczającą w jakiej sieci IP znajduje się urządzenie.	Maski podsieci CPU i karty adaptera Ethernet – USB urządzenia programującego muszą być dokładnie takie same. Maska podsieci zwykle ma wartość 255.255.255.0. Maska podsieci połączona z adresem IP urządzenia za pomocą logicznej operacji AND, definiuje granice IP podsieci.

Nadawanie lub sprawdzanie adresu IP urządzenia programującego za pomocą „My Network Places” na komputerze użytkownika

Użytkownik może nadać lub sprawdzić adres IP urządzenia programującego dokonując następujących wyborów menu:

- (kliknięcie prawym klawiszem myszy) „My Network Places”
- „Properties”
- (kliknięcie prawym klawiszem myszy) „Local Area Connection”
- „Properties”

W oknie dialogowym „Local Area Connection Properties”, w polu „This connection uses the following items:” należy wybrać pozycję „Internet Protocol (TCP/IP)”, i kliknąć najpierw „Internet Protocol (TCP/IP)”, a potem „Properties”. Następnie należy wybrać „Obtain an IP address automatically (DHCP)” lub „Use the following IP address” (w celu wprowadzenia statycznego adresu IP).

UWAGA

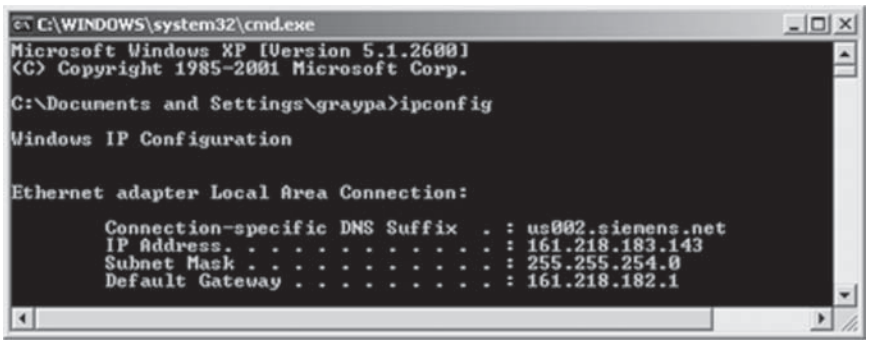
Protokół DHCP (*Dynamic Host Configuration Protocol*) automatycznie nadaje adres IP urządzeniu programującemu z serwera DHCP w momencie włączenia zasilania.

Sprawdzanie adresu IP urządzenia programującego za pomocą rozkazów „ipconfig” i „ipconfig/all”

Adresy IP urządzenia programującego oraz routera (jeśli jest zainstalowany) można również sprawdzić dokonując następujących wyborów menu:

- „Start” (na desktopie)
- „Run”

W oknie dialogowym „Run”, w polu „Open” należy wpisać „cmd” i kliknąć przycisk „OK”. W oknie dialogowym „C:WINDOWSsystem32cmd.exe”, które zostanie wyświetlone należy wprowadzić rozkaz „ipconfig”. Przykładowy wynik jest przedstawiony poniżej:



Więcej informacji zostanie wyświetlonych po użyciu komendy „ipconfig /all". Wśród wyświetlonych informacji znajduje się typ karty adaptera urządzenia programującego i adres Ethernet (MAC):

Nadawanie adresu IP CPU

CPU można nadać adres IP wykorzystując jedną z poniższych metod:

- Przypisanie tymczasowego adresu IP w trybie *online*.
- Skonfigurowanie stałego adresu IP.

7.1.3.2 Nadawanie tymczasowego adresu IP w trybie online

Urządzeniu sieciowemu można nadać adres IP w trybie *online*. Jest to szczególnie przydatne podczas początkowej konfiguracji urządzenia.

W celu nadania adresu IP w trybie *online* należy postępować zgodnie z następująca procedurą:

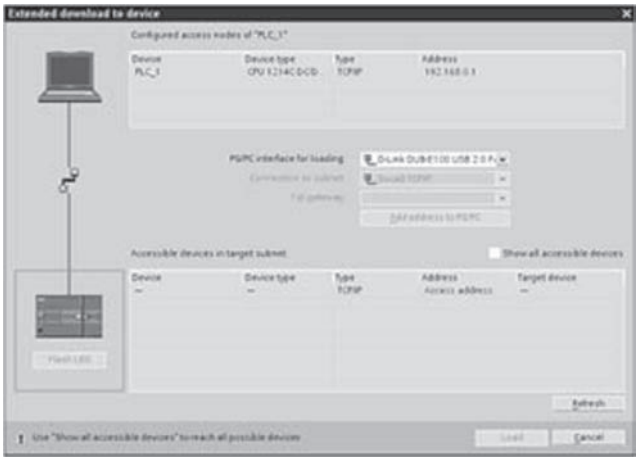
1. W „Project tree" sprawdzić, że CPU nie ma przypisanego żadnego adresu IP. Dokonuje się tego wybierając kolejno następujące pozycje menu:
   - „Online access"
   - <karta adaptera sieci, do której urządzenie jest włączone>
   - „Update accessible devices"

2. W „Project tree” wybrać „Add new device” w celu dołączenia do projektu nowego CPU.

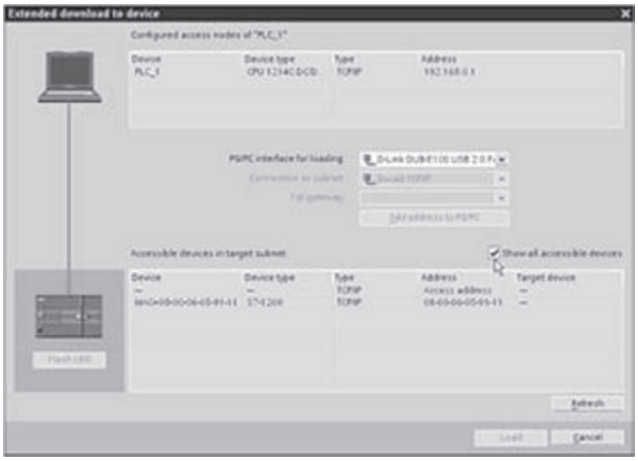
3. W „Project tree” wybrać CPU.

4. Kliknąć przycisk „Download to device”.

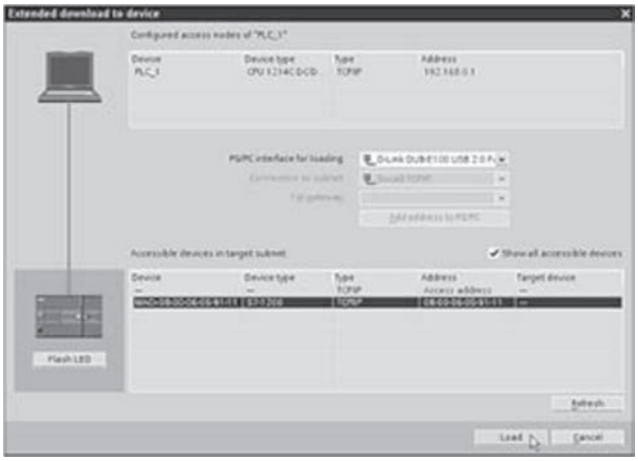
5. W oknie dialogowym „Extended download to device”, w polu „PG/PC interface for loading” wybrać z listy kartę adaptera sieci, do której urządzenie jest włączone.



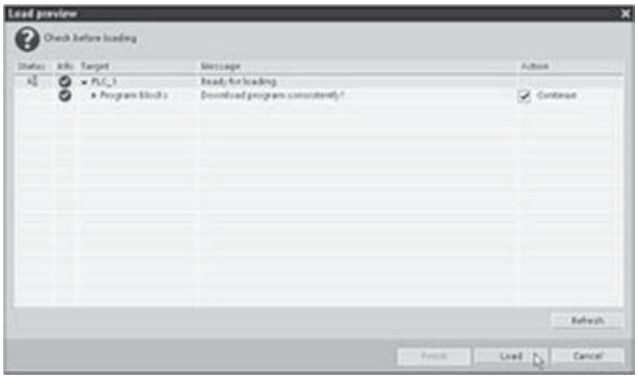
6. Kliknąć pole wyboru „Check all accessible devices”.



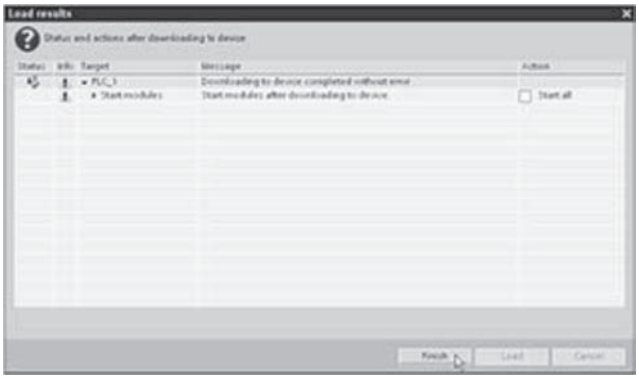
7. W polu „Accessible devices in target network” field, kliknąć urządzenie z adresem MAC (wartość złożona z 12 cyfr heksadecymalnych). Następnie kliknąć przycisk „Load”.



8. W trakcie uruchamiania procesu kompilacji zostanie wyświetlone pole wyboru „Continue”. Kliknąć to pole wyboru by dokończyć kompilację.



9. W trakcie uruchamiania procesu ściągania pojawi się pole wyboru „Start all”. Zostawić to pole wyboru czyste tak, że CPU pozostanie w trybie STOP. Kliknąć przycisk „Finish” w celu zakończenia ściągania.



10. W „Project tree” sprawdzić, że CPU nie ma przypisanego żadnego adresu IP. Dokonuje się tego wybierając kolejno następujące pozycje menu:

- „Online access”
- <karta adaptera sieci, do której urządzenie jest włączone>
- „Update accessible devices”

UWAGA: Poniżej „Update accessible devices” zostaną wyświetlone wszystkie adresy IP urządzeń podłączonych do wybranej sieci.

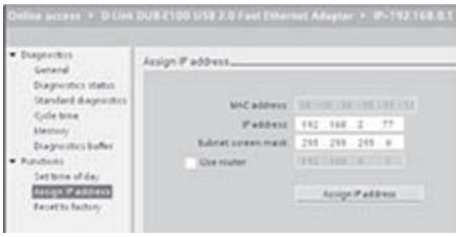
11. W „Project tree” dokonać następującego wyboru menu:

- „Online access”
- <karta adaptera sieci, do której urządzenie jest włączone>
- „Update accessible devices”
- <adres IP>
- „Online & diagnostics”

12. W oknie dialogowym „Online & diagnostics” dokonać następującego wyboru menu:

- „Functions”
- „Assign IP address”

13. W polu „IP address”, wprowadzić nowy, tymczasowy adres IP



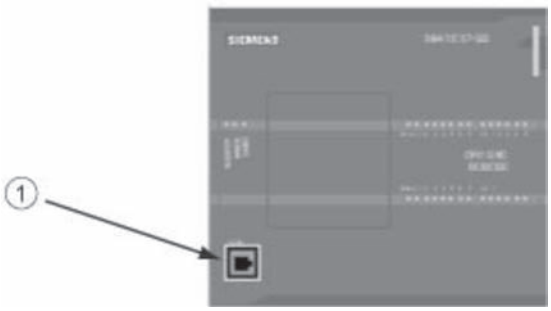
14. W „Project tree” sprawdzić, że CPU został przypisany nowy adres IP. Dokonuje się tego wybierając kolejno następujące pozycje menu:

- „Online access”
- < karta adaptera sieci, do której urządzenie jest włączone>
- „Update accessible devices”

7.1.3.3 Konfiguracja stałego adresu IP

Konfigurowanie interfejsu PROFINET

Po zainstalowaniu i skonfigurowaniu CPU, można skonfigurować parametry interfejsu PROFINET. W tym celu należy wybrać port PROFINET, klikając zielony symbol PROFINET znajdujący się na CPU. Zakładka „Properties” w oknie inspekcyjnym powoduje wyświetlenie portu PROFINET.



① port PROFINET

Konfigurowanie adresu IP

Adres ethernetowy (**MAC**): Każde urządzenie w sieci PROFINET ma nadany przez producenta w celach identyfikacyjnych adres MAC (*Media Access Control*). Adres MAC składa się z sześciu grup po dwie cyfry heksadecymalne każda, oddzielonych od siebie łącznikiem (-) lub dwukropkiem (:), występujących w takiej kolejności, w jakiej są nadawane (na przykład, 01-23-45-67-89-ab lub 01:23:45:67:89:ab).

Wszystkie urządzenia zainstalowane w tej samej sieci PROFINET muszą mieć unikalne adresy MAC. Jeżeli w tej samej sieci PROFINET znajdą się dwa urządzenia z tym samym adresem MAC, to pojawią się problemy komunikacyjne.

Adres IP: Każde urządzenie musi mieć również adres IP (*Internet Protocol*). Ten adres pozwala urządzeniu przesyłać dane w bardziej rozbudowanych sieciach.

Każdy adres IP dzieli się na cztery 8-bitowe segmenty i jest wyrażany w formacie dziesiętnym z kropkami (na przykład, 211.154.184.16). Pierwsza część adresu IP, Network ID, jest stosowana do identyfikacji sieci (w jakiej sieci jesteśmy?), a druga część adresu to Host ID (unikalny dla każdego urządzenia w sieci). Adres IP 192.168.x.y jest standardowym oznaczeniem sieci prywatnej, niedostępnej przez Internet.

Maska podsieci: Podsieć jest to logiczna grupa połączonych urządzeń sieciowych. Węzły podsieci są zwykle umieszczone fizycznie blisko siebie w ramach sieci lokalnej LAN (*Local Area Network*). Maska (zwana maską podsieci lub maską sieci) definiuje granice IP podsieci.

Dla małych sieci lokalnych odpowiednia jest zwykle maska 255.255.255.0. Oznacza to, że wszystkie adresy IP tej sieci powinny mieć takie same pierwsze trzy oktety, a różne urządzenia w sieci są identyfikowane za pomocą ostatniego oktetu (pola 8-bitowego). Przykładem tego jest ustalenie maski podsieci 255.255.255.0 i nadanie urządzeniem małej sieci adresów IP od 192.168.2.0 do 192.168.2.255.

Połączenie różnych podsieci może być wykonane jedynie za pomocą routerów. Jeśli wykorzystuje się podsieci, to trzeba zastosować IP router.

IP router: Routery są ogniwem pośredniczącym między różnymi sieciami LAN. Za pośrednictwem routera komputer znajdujący się w sieci LAN może przesyłać wiadomości do dowolnych innych sieci, do których również mogą być podłączone sieci LAN. Jeżeli odbiorca danych nie znajduje się w tej samej sieci LAN, to router przesyła dane do innej sieci lub grupy sieci, skąd mogą być dostarczone do miejsca przeznaczenia.

W celu dostarczania i odbierania danych, routery opierają się na adresach IP.

Właściwości adresów IP: W oknie „Properties” należy wybrać wejście konfiguracyjne „Ethernet address”. Portal TIA wyświetla dialog konfiguracyjny adresu Ethernet, który umożliwia powiązanie programu projektu z adresem IP tego CPU, który otrzyma ten projekt.

UWAGA

CPU nie ma prekonfigurowanego adresu IP. Użytkownik musi ręcznie nadać CPU adres IP. Jeżeli CPU jest podłączona do routera w sieci, to należy również wprowadzić adres IP routera. Wszystkie adresy IP są konfigurowane podczas ładowania projektu.

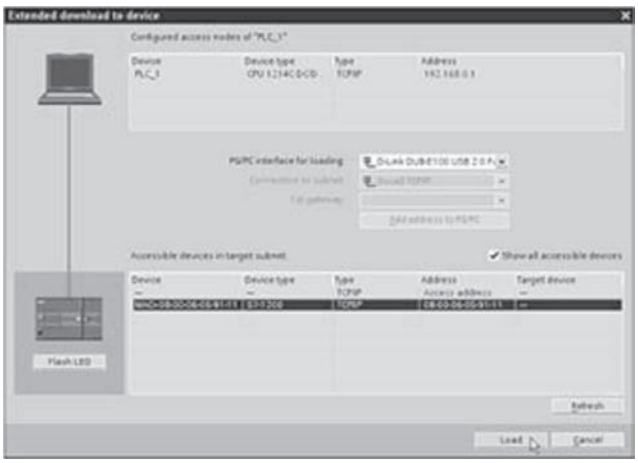
Więcej informacji na ten temat jest podanych w części „Nadawanie adresów IP urządzeniom programującym i sieciowym".

W poniższej tabeli znajdują się definicje parametrów adresów IP:

Parametr	Opis	
Podsieć	Nazwa podsieci, do której jest podłączone urządzenie. Kliknięcie na „Add new subnet" pozwala utworzyć nową podsieć. Domyślnie jest „Not connected" (nie połączona). Możliwe są dwa typy połączenia: • Domyślne „Not connected" realizuje połączenia lokalne. • Podsieć jest wymagana, jeżeli sieć zawiera dwa lub więcej urządzeń.	
Protokół IP	Adres IP	Adres IP nadany CPU
	Maska podsieci	Ustalona maska podsieci
	Użycie routera IP	Kliknięcie na pole wyboru pozwala określić, czy router jest używany
	Adres routera	Adres IP nadany routerowi (jeśli ma zastosowanie)

7.1.4 Testowanie sieci PROFINET

Po ukończeniu konfiguracji, można załadować projekt do CPU. Po załadowaniu projektu wszystkie adresy IP są skonfigurowane.



Nadawanie adresu IP urządzeniu online

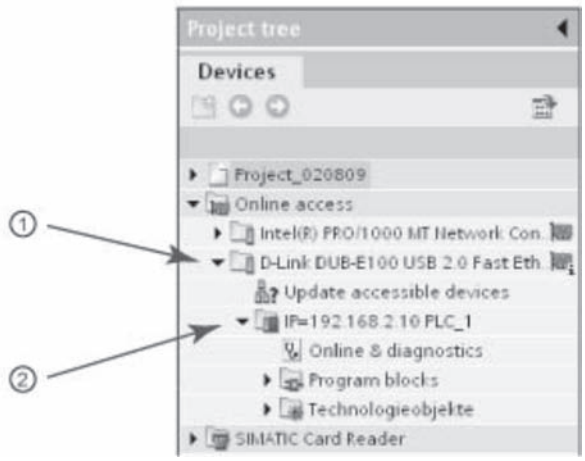
CPU S7-1200 nie ma prekonfigurowanego adresu IP. Użytkownik musi ręcznie nadać CPU adres IP.

Użytkownik może nadać adres IP urządzeniu będącemu *online*. Jednakże ten adres jest tymczasowy i zostanie utracony po wyłączeniu i ponownym włączeniu zasilania. Szczegółowa procedura nadawania adresu w trybie *online* jest podana w części „Nadawanie tymczasowego adresu IP w trybie *online*".

W celu nadania stałego adresu IP, należy skonfigurować adres IP w menu konfiguracyjnym urządzenia, zapisać tę konfigurację i załadować ją do PLC. Więcej informacji na ten temat jest podanych w części „Konfigurowanie stałego adresu IP". Adres IP, który jest załadowany jako część konfiguracji PLC nie jest tracony po wyłączeniu zasilania PLC. Adresy IP podłączonych CPU można wyświetlić wybierając menu „Online access", tak jak to pokazano poniżej.

UWAGA

Wyświetlone są wszystkie skonfigurowane sieci urządzenia programującego. W celu wyświetlenia adresu IP konkretnego CPU S7-1200 należy wybrać właściwą sieć.

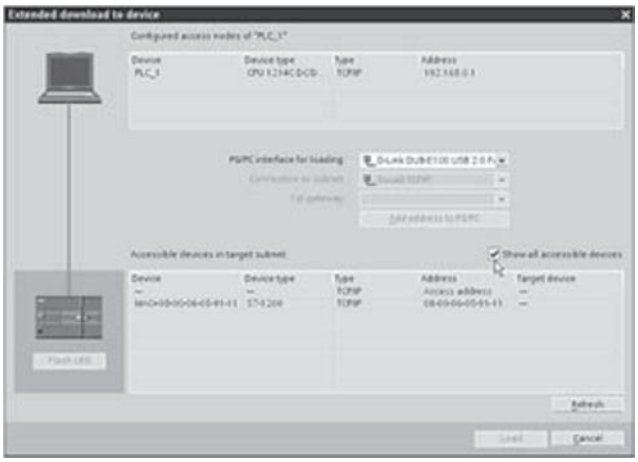


① Druga z sieci Ethernet dla danego urządzenia programującego

② Adres IP jedynego CPU S7-1200 w tej sieci Ethernet

Użycie okna dialogowego „Extended download to device" do testowania jakie urządzenia sieciowe są podłączone

Funkcja CPU S7-1200 „Download to device" i jej okno dialogowe „Extended download to device" może pokazać wszystkie dostępne urządzenia sieciowe oraz czy wszystkim tym urządzeniom zostały nadane unikalne adresy IP. W celu wyświetlenia wszystkich osiągalnych i udostępnionych urządzeń wraz z ich adresami MAC i IP należy zaznaczyć pole wyboru „Show all accessible devices".



Jeżeli pożądane urządzenie sieciowe nie znajduje się na liście, to komunikacja z tym urządzeniem została z jakiegoś powodu przerwana. Urządzenie i siec należy sprawdzić pod kątem błędów sprzętowych i/lub konfiguracyjnych.

7.2 Komunikacja HMI-PLC

CPU obsługuje połączenie komunikacyjne PROFINET z HMI. Podczas przygotowywania komunikacji między CPU i HMI należy rozważyć następujące czynniki:

Konfiguracja/ustawienia:

- Port PROFINET CPU musi być skonfigurowany do połączenia z HMI.
- HMI musi być ustawione i skonfigurowane.
- Informacje konfiguracyjne HMI stanowią część projektu CPU i mogą być konfigurowane oraz ładowane razem z projektem.
- Przy komunikacji „jeden do jednego” nie jest wymagany przełącznik ethernetowy; przełącznik ethernetowy jest konieczny wtedy, kiedy w sieci są połączone więcej niż dwa urządzenia.

UWAGA

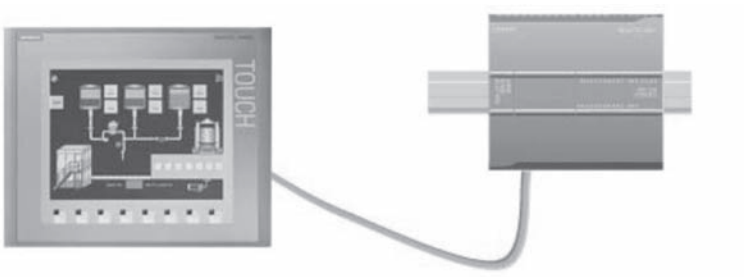
Do połączenia kilku CPU i HMI można wykorzystać 4-portowy przełącznik ethernetowy firmy Siemens typu CSM1277. Port PROFINET wbudowany do CPU S7-1200 nie jest wyposażony w przełącznik ethernetowy.

Obsługiwane funkcje:

- HMI może odczytywać i zapisywać dane z/do CPU.
- Wiadomości mogą być wyzwalane na podstawie informacji wyszukiwanych w CPU.
- Diagnostyka systemowa.

UWAGA

WinCC Basic i STEP 7 Basic są składnikami TIA Portal. Więcej informacji na temat konfigurowania HMI można znaleźć w WinCC Basic.



Wymagane kroki podczas konfiguracji komunikacji między HMI i CPU

Krok	Zadanie
1	Zestawianie sprzętowego połączenia komunikacyjnego. Interfejs PROFINET stanowi fizyczne połączenie między HMI i CPU. Ponieważ CPU ma wbudowaną funkcję Auto-Cross-Over, więc w celu realizacji połączenia można zastosować kabel ethernetowy prosty lub skrosowany. Przełącznik ethernetowy nie jest konieczny by połączyć HMI z CPU.
2	Konfiguracja urządzenia. Więcej informacji znajduje się w części „Komunikacja z urządzeniem programującym : Konfiguracja urządzenia”.
3	Konfiguracja logicznego połączenia sieciowego między HMI i CPU.
4	Konfiguracja stałego adresu IP. Więcej informacji znajduje się w części „Komunikacja HMI-PLC: Konfiguracja logicznego połączenia sieciowego między HMI i CPU”. Korzysta się z tego samego procesu Więcej informacji znajduje się w części „Komunikacja z urządzeniem programującym: Konfiguracja stałego adresu IP”.konfiguracyjnego; w tym przypadku trzeba skonfigurować adresy IP HMI i CPU.
5	Testowanie sieci PROFINET. Trzeba załadować konfigurację każdego CPU. Więcej informacji znajduje się w części „Komunikacja z urządzeniem programującym: Testowanie sieci PROFINET”.

Por. również:

Zestawianie połączenia komunikacyjnego

Konfiguracja urządzenia

Konfiguracja logicznego połączenia sieciowego między HMI i CPU

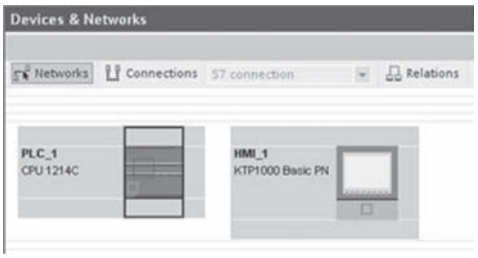
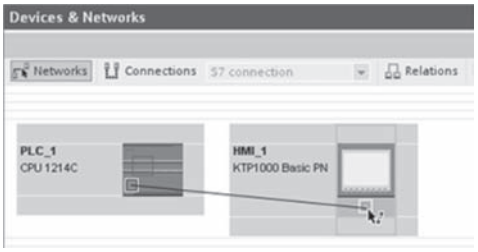
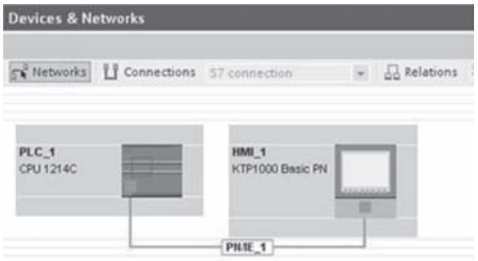
Konfiguracja stałego adresu IP

Testowanie sieci PROFINET

7.2.1 Konfiguracja logicznego połączenia sieciowego między HMI i CPU

Po zainstalowaniu i skonfigurowaniu CPU, można skonfigurować połączenie sieciowe.

W celu utworzenia połączenia sieciowego między urządzeniami projektu, w portalu *Devices* & *Networks* należy wybrać „Network view". Połączenie ethernetowe realizuje się wybierając zielony prostokątny symbol (Ethernet) na CPU. Następnie, trzymając wciśnięty klawisz myszy, należy przeciągnąć linię do symbolu Ethernet na urządzeniu HMI. Po zwolnieniu klawisza myszy połączenie ethernetowe jest wykonane.

Akcja	**Wynik**
Wybór „Network view" powoduje wyświetlenie urządzeń, które mają być połączone.	
Wybór portu na jednym urządzeniu i przeciągnięcie połączenia do portu na drugim urządzeniu.	
Zwolnienie klawisza myszy tworzy połączenie sieciowe.	

7.3 Komunikacja PLC-PLC

CPU może się komunikować z innym CPU znajdującym się w sieci za pomocą instrukcji TSEND_C i TRCV_C. Podczas przygotowywania komunikacji między dwiema CPU należy rozważyć następujące czynniki:

- Konfiguracja/ustawienia: Wymagana jest konfiguracja sprzętowa.
- Obsługiwane funkcje: odczytywanie i zapisywanie danych z/do równorzędnego CPU.
- Przy komunikacji „jeden do jednego" nie jest wymagany przełącznik ethernetowy; przełącznik ethernetowy jest konieczny wtedy, kiedy w sieci są połączone więcej niż dwa urządzenia.

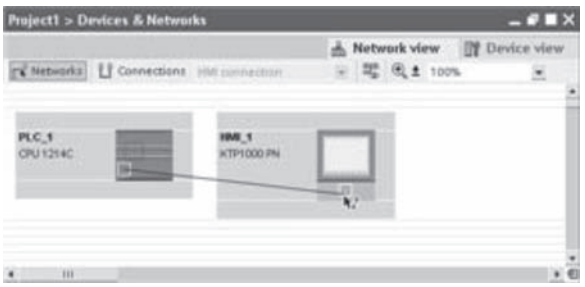
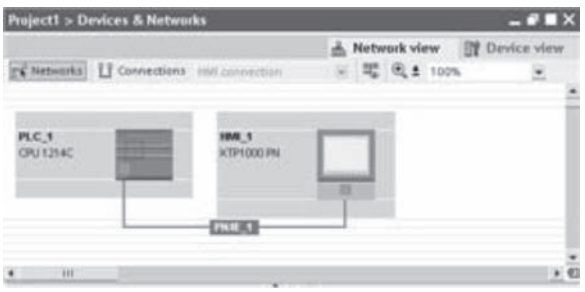
Wymagane kroki podczas konfiguracji komunikacji między dwiema CPU

Krok	Zadanie
1	Zestawianie sprzętowego połączenia komunikacyjnego. Interfejs PROFINET stanowi fizyczne połączenie między dwiema CPU. Ponieważ CPU ma wbudowaną funkcję Auto-Cross-Over, więc w celu realizacji połączenia można zastosować kabel ethernetowy prosty lub skrosowany. Przełącznik ethernetowy nie jest konieczny by połączyć dwie CPU. Więcej informacji znajduje się w części „Komunikacja z urządzeniem programującym : Zestawianie połączenia komunikacyjnego".
2	Konfiguracja urządzenia. Nalezy skonfigurować dwa projekty, z których każdy zawiera CPU. Więcej informacji znajduje się w części „Komunikacja z urządzeniem programującym : Konfiguracja urządzenia".
3	Konfiguracja logicznego połączenia sieciowego między HMI i CPU. Więcej informacji znajduje się w części „Komunikacja PLC-PLC: Konfiguracja logicznego połączenia sieciowego między dwiema CPU".
4	Konfiguracja stałego adresu IP. Korzysta się z tego samego procesu konfiguracyjnego; w tym przypadku trzeba skonfigurować adresy IP dwóch CPU (na przykład, PLC_1 i PLC_2). Więcej informacji znajduje się w części „Komunikacja z urządzeniem programującym: Konfiguracja stałego adresu IP".
5	Konfiguracja parametrów nadawczych i odbiorczych. Należy skonfigurować instrukcje TSEND_C i TRCV_C w obu CPU w celu umożliwienia komunikacji między nimi. Więcej informacji znajduje się w części „Komunikacja PLC-PLC: Konfiguracja parametrów nadawczych i odbiorczych".
6	Testowanie sieci PROFINET. Trzeba załadować konfigurację każdego CPU. Więcej informacji znajduje się w części „Komunikacja z urządzeniem programującym: Testowanie sieci PROFINET".

7.3.1 Konfiguracja logicznego połączenia sieciowego między dwiema CPU

Po zainstalowaniu i skonfigurowaniu CPU, można skonfigurować połączenie sieciowe.

W celu utworzenia połączenia sieciowego między urządzeniami projektu, w portalu *Devices* & *Networks* należy wybrać „Network view". Połączenie ethernetowe realizuje się wybierając zielony prostokątny symbol (Ethernet) na CPU. Następnie, trzymając wciśnięty klawisz myszy, należy przeciągnąć linię do symbolu Ethernet na drugiej CPU. Po zwolnieniu klawisza myszy połączenie ethernetowe jest wykonane.

Akcja	Wynik
Wybór „Network view" powoduje wyświetlenie urządzeń, które mają być połączone.	
Wybór portu na jednym urządzeniu i przeciągnięcie połączenia do portu na drugim urządzeniu.	
Zwolnienie klawisza myszy tworzy połączenie sieciowe.	

7.3.2 Konfiguracja parametrów nadawczych i odbiorczych

Do zrealizowania połączenia między dwiema CPU wykorzystuje się komunikację za pomocą bloków nadawczych (T-block). Zanim CPU będą mogły komunikować się poprzez PROFINET, należy skonfigurować parametry nadawcze i odbiorcze dla przesyłanych wiadomości. Te parametry określają sposób działania komunikacji podczas nadawania lub odbierania wiadomości do/z urządzenia docelowego.

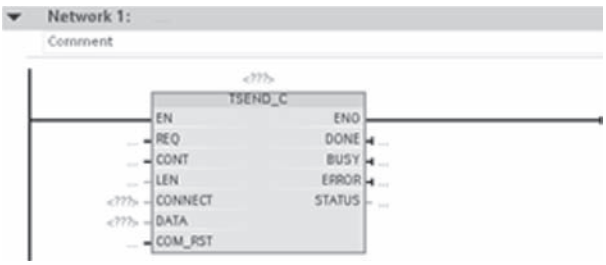
7.3.2.1 Konfigurowanie parametrów nadawczych instrukcji TSEND_C

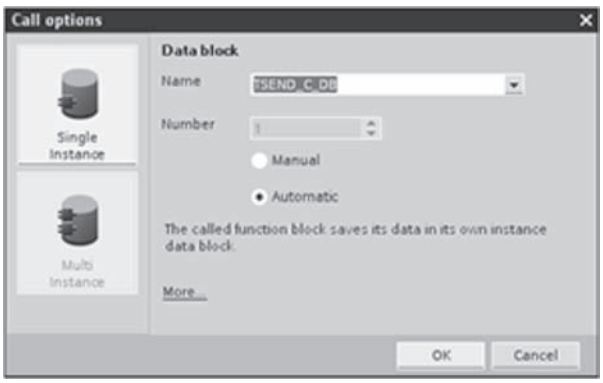
Instrukcja TSEND_C

Instrukcja TSEND_C tworzy połączenie komunikacyjne ze stacją partnerską. Połączenie po skonfigurowaniu i ustaleniu jest automatycznie utrzymywane i monitorowane, aż do czasu wydania przez instrukcję polecenia rozłączenia. Instrukcja TSEND_C łączy w sobie funkcje instrukcji TCON, TDISCON i TSEND.

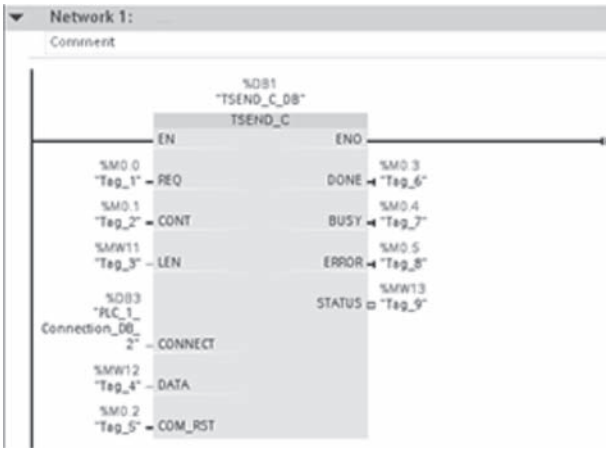
Sposób w jaki instrukcja TSEND_C nadaje dane można skonfigurować z menu Device Configuration portalu TIA. Aby rozpocząć, należy instrukcję umieścić w programie posługując się zakładką „Instructions” znajdującą się w: „Extended Instructions” > „Communications”.

Instrukcja jest wyświetlana wraz z oknem dialogowym „Call options” (opcje wywołania), w którym można zdefiniować DB przechowujący parametry instrukcji TSEND_C.





Do wejść i wyjść użytkownik może przypisać *tagi* lokalizacji pamięci, tak jak to zilustrowano na poniższym rysunku.



Konfiguracja parametrów General

Parametry komunikacyjne specyfikuje się w konfiguracyjnym oknie dialogowym „Properties" instrukcji TSEND_C. To okno dialogowe pojawia się blisko dołu strony zawsze po wyborze dowolnej części instrukcji TSEND_C.

Konfiguracja parametrów Connection

Każdy CPU ma zintegrowany port PROFINET, który obsługuje standardową komunikację PROFINET. Obsługiwane protokoły Ethernet są opisane w następujących dwóch typach połączenia:

Protokół	Nazwa protokołu	Zastosowanie
RFC 1006	ISO Transport over TCP	Fragmentowanie i składanie wiadomości
TCP	Transport Connection Protocol	Transport ramek

ISO Transport over TCP (RFC 1006)

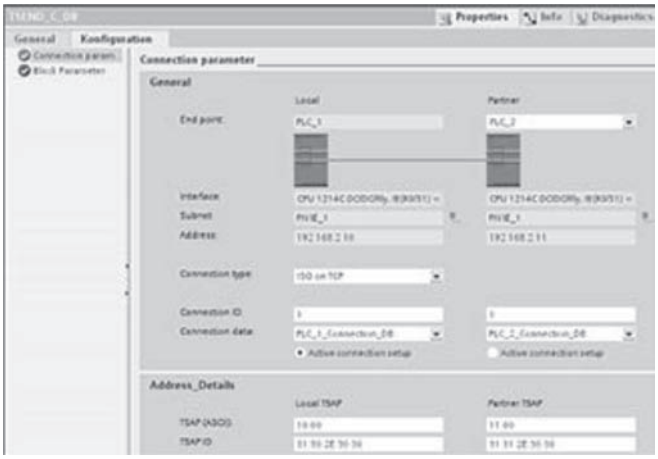
ISO Transport over TCP jest to mechanizm umożliwiający wprowadzenie aplikacji ISO do sieci TCP/IP. Ten protokół ma następujące cechy:

- Jest to wydajny protokół komunikacyjny ściśle powiązany ze sprzętem.
- Jest odpowiedni dla danych o średnich i dużych wielkościach (do 8192 bajtów).
- W przeciwieństwie do TCP, dane charakteryzują się identyfikatorem końca danych i są zorientowane na transfer komunikatów.
- Istnieje możliwość routowania; może być używany w sieciach WAN.
- Można wykorzystywać dynamiczne długości danych.
- Ze względu na interfejs programowy SEND / RECEIVE, zarządzanie danymi wymaga złożonego oprogramowania.

Wykorzystując punkty dostępowe serwisu transportowego TSAP (*Transport Service Access Point*), protokół TCP pozwala zrealizować wiele połączeń z jednym

adresem IP (do 64K połączeń). Z RFC 1006, punkty TSAP jednoznacznie identyfikują te połączenia komunikacyjnych punktów końcowych z adresem IP.

W części „Address Details" okna dialogowego „Connection Parameters", użytkownik definiuje punkty TSAP, które będą wykorzystywane. TSAP połączenia z CPU jest wprowadzany w polu „Local TSAP". TSAP przypisany połączeniu w partnerskim CPU jest wprowadzany w polu „Partner TSAP".



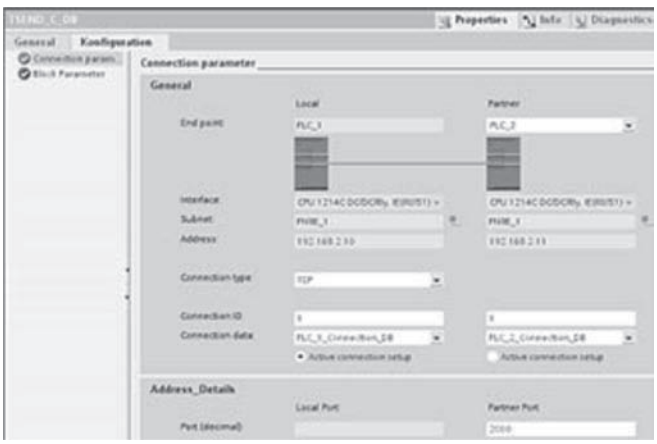
Konfigurowane parametry są zdefiniowane poniżej:

Parametr	**Definicja**
Ogólne General	
Punkt końcowy: Partner End point: Partner	Nazwa nadana partnerskiej (odbierającej) CPU
Interfejs Interface	Nazwa nadana interfejsom
Podsieć Subnet	Nazwa nadana podsieciom
Adres Address	Nadany adres IP
Typ połączenia Connection type	Typ protokołu Ethernet
ID połączenia Connection ID	Numer ID
Dane Connection data	Miejsce pamiętania danych lokalnej i partnerskiej CPU
Ustawienie aktywnego połączenia Active connection setup	Przycisk opcji lokalnej lub partnerskiej CPU jako aktywnego połączenia
Szczegóły adresu Address details	
TSAP (ASCII)	Punkty TSAP lokalnej i partnerskiej CPU w formacie ASCII
TSAP ID	Punkty TSAP lokalnej i partnerskiej CPU w formacie heksadecymalnym

Transport Connection Protocol (TCP)

TCP jest standardowym protokołem opisanym przez RFC 793: *Transmission Control Protocol*. Głównym celem TCP jest zapewnienie niezawodnego, bezpiecznego połączenia między parami procesów. Ten protokół ma następujące cechy:

- Ponieważ jest ściśle powiązany ze sprzętem, więc jest wydajnym protokołem komunikacyjnym.
- Jest odpowiedni dla danych o średnich i dużych wielkościach (do 8192 bajtów).
- Zapewnia aplikacjom znacznie więcej funkcji, w szczególności:
  - usuwanie błędów,
  - sterowanie przepływem,
  - niezawodność
- Jest to protokół zorientowany na połączenia.
- Może być elastycznie stosowany z urządzeniami innych firm, które obsługują wyłącznie TCP.
- Istnieje możliwość routowania.
- Można wykorzystywać wyłącznie statyczne długości danych.
- Wiadomości są potwierdzane.
- Aplikacje są adresowane za pomocą numerów portów.
- Większość protokółów aplikacji użytkownika, takich jak TELENT i FTP, korzysta z TCP.
- Ze względu na interfejs programowy SEND / RECEIVE, zarządzanie danymi wymaga złożonego oprogramowania.



Konfigurowane parametry są zdefiniowane poniżej:

Parametr	Definicja
Ogólne General	
Punkt końcowy: Partner End point: Partner	Nazwa nadana partnerskiej (odbierającej) CPU
Interfejs Interface	Nazwa nadana interfejsom
Podsieć Subnet	Nazwa nadana podsieciom
Adres Address	Nadany adres IP
Typ połączenia Connection type	Typ protokołu Ethernet
ID połączenia Connection ID	Numer ID
Dane Connection data	Miejsce pamiętania danych lokalnej i partnerskiej CPU
Ustawienie aktywnego połączenia Active connection setup	Przycisk opcji lokalnej lub partnerskiej CPU jako aktywnego połączenia
Szczegóły adresu Address details	
Port (dziesiętnie) Port (decimal)	Port partnerskiej CPU w formacie dziesiętnym

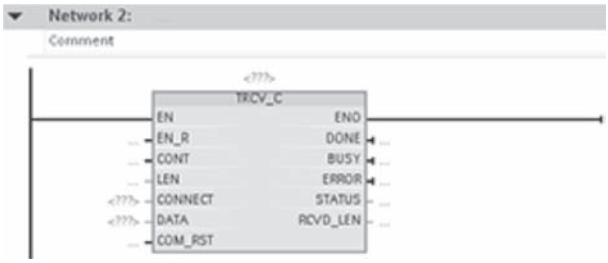
7.3.2.2 Konfigurowanie parametrów odbiorczych instrukcji TRCV_C

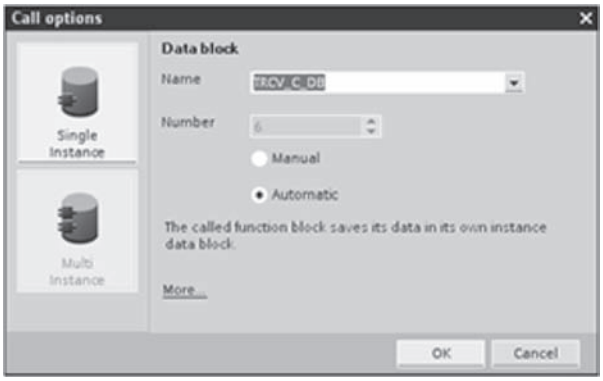
Instrukcja TRCV_C

Instrukcja TRCV_C tworzy połączenie komunikacyjne ze stacją partnerską. Połączenie po skonfigurowaniu i ustaleniu jest automatycznie utrzymywane i monitorowane, aż do czasu wydania przez instrukcję polecenia rozłączenia. Instrukcja TRCV_C łączy w sobie funkcje instrukcji TCON, TDISCON i TSEND.

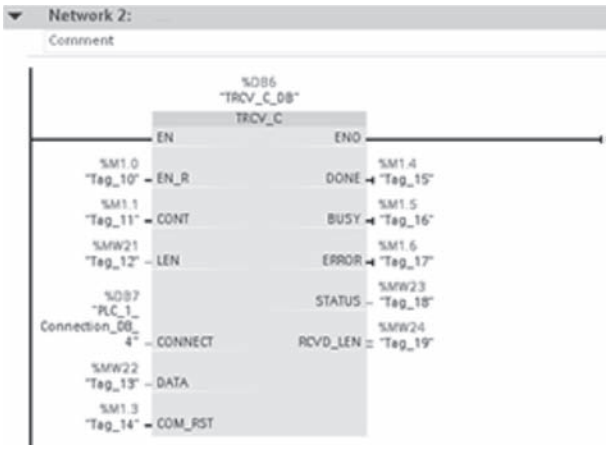
Sposób w jaki instrukcja TRCV_C odbiera dane można skonfigurować z menu konfiguracyjnego CPU portalu TIA. Aby rozpocząć, należy instrukcję umieścić w programie posługując się zakładką „Instructions” znajdującą się w: „Extended Instructions” > „Communications”.

Instrukcja jest wyświetlana wraz z oknem dialogowym „Call options” (opcje wywołania), w którym można zdefiniować DB przechowujący parametry instrukcji TRCV_C.





Do wejść i wyjść użytkownik może przypisać *tagi* lokalizacji pamięci, tak jak to zilustrowano na poniższym rysunku.



Konfiguracja parametrów General

Parametry komunikacyjne specyfikuje się w konfiguracyjnym oknie dialogowym „Properties” instrukcji TRCV_C. To okno dialogowe pojawia się blisko dołu strony zawsze po wyborze dowolnej części instrukcji TRCV_C.

Konfiguracja parametrów Connection

Każdy CPU ma zintegrowany port PROFINET, który obsługuje standardową komunikację PROFINET. Obsługiwane protokoły Ethernet są opisane w następujących dwóch typach połączenia:

Protokół	Nazwa protokołu	Zastosowanie
RFC 1006	ISO Transport over TCP	Fragmentowanie i składanie wiadomości
TCP	Transport Connection Protocol	Transport ramek

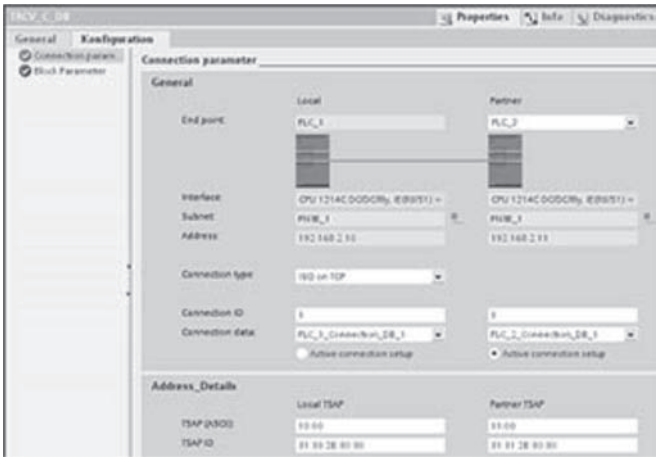
ISO Transport over TCP (RFC 1006)

ISO Transport over TCP jest to mechanizm umożliwiający wprowadzenie aplikacji ISO do sieci TCP/IP. Ten protokół ma następujące cechy:

- Jest to wydajny protokół komunikacyjny ściśle powiązany ze sprzętem.
- Jest odpowiedni dla danych o średnich i dużych wielkościach (do 8192 bajtów).
- W przeciwieństwie do TCP, dane charakteryzują się identyfikatorem końca danych i są zorientowane na widomości.
- Istnieje możliwość routowania; może być używany w sieciach WAN.
- Można wykorzystywać dynamiczne długości danych.
- Ze względu na interfejs programowy SEND/RECEIVE, zarządzanie danymi wymaga złożonego oprogramowania.

Wykorzystując punkty dostępowe serwisu transportowego TSAP (*Transport Service Access Point*), protokół TCP pozwala zrealizować wiele połączeń z jednym adresem IP (do 64K połączeń). Z RFC 1006, punkty TSAP jednoznacznie identyfikują te połączenia komunikacyjnych punktów końcowych z adresem IP.

W części „Address Details” okna dialogowego „Connection Parameters”, użytkownik definiuje punkty TSAP, które będą wykorzystywane. TSAP połączenia z CPU jest wprowadzany w polu „Local TSAP”. TSAP przypisany połączeniu w partnerskim CPU jest wprowadzany w polu „Partner TSAP”.



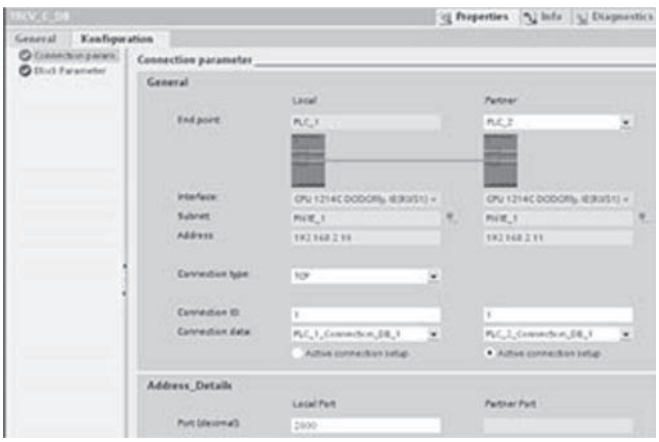
Konfigurowane parametry są zdefiniowane poniżej:

Parametr	**Definicja**
Ogólne General	
Punkt końcowy: Partner End point: Partner	Nazwa nadana partnerskiej (odbierającej) CPU
Interfejs Interface	Nazwa nadana interfejsom
Podsieć Subnet	Nazwa nadana podsieciom
Adres Address	Nadany adres IP
Typ połączenia Connection type	Typ protokołu Ethernet
ID połączenia Connection ID	Numer ID
Dane Connection data	Miejsce pamiętania danych lokalnej i partnerskiej CPU
Ustawienie aktywnego połączenia Active connection setup	Przycisk opcji lokalnej lub partnerskiej CPU jako aktywnego połączenia
Szczegóły adresu Address details	
TSAP (ASCII)	Punkty TSAP lokalnej i partnerskiej CPU w formacie ASCII
TSAP ID	Punkty TSAP lokalnej i partnerskiej CPU w formacie heksadecymalnym

Transport Connection Protocol (TCP)

TCP jest standardowym protokołem opisanym przez RFC 793: *Transmission Control Protocol*. Głównym celem TCP jest zapewnienie niezawodnego, bezpiecznego połączenia między parami procesów. Ten protokół ma następujące cechy:

- Ponieważ jest ściśle powiązany ze sprzętem, więc jest wydajnym protokołem komunikacyjnym.
- Jest odpowiedni dla danych o średnich i dużych wielkościach (do 8192 bajtów).
- Zapewnia aplikacjom znacznie więcej funkcji, w szczególności:
  - usuwanie błędów,
  - sterowanie przepływem,
  - niezawodność
- Jest to protokół zorientowany na połączenia.
- Może być elastycznie stosowany z urządzeniami innych firm, które obsługują wyłącznie TCP.
- Istnieje możliwość routowania.
- Można wykorzystywać wyłącznie statyczne długości danych.
- Wiadomości są potwierdzane.
- Aplikacje są adresowane za pomocą numerów portów.
- Większość protokółów aplikacji użytkownika, takich jak TELENT i FTP, korzysta z TCP.
- Ze względu na interfejs programowy SEND / RECEIVE, zarządzanie danymi wymaga złożonego oprogramowania.



Konfigurowane parametry są zdefiniowane poniżej:

Parametr	Definicja
Ogólne General	
Punkt końcowy: Partner End point: Partner	Nazwa nadana partnerskiej (odbierającej) CPU
Interfejs Interface	Nazwa nadana interfejsom
Podsieć Subnet	Nazwa nadana podsieciom
Adres Address	Nadany adres IP
Typ połączenia Connection type	Typ protokołu Ethernet
ID połączenia Connection ID	Numer ID
Dane Connection data	Miejsce pamiętania danych lokalnej i partnerskiej CPU
Ustawienie aktywnego połączenia Active connection setup	Przycisk opcji lokalnej lub partnerskiej CPU jako aktywnego połączenia
Szczegóły adresu Address details	
Port (dziesiętnie) Port (decimal)	Port partnerskiej CPU w formacie dziesiętnym

7.4 Informacje referencyjne

7.4.1 Lokalizacja adresu Ethernet (MAC) w CPU

W sieciach PROFINET adres MAC (*Media Access Control*) jest numerem nadawanym przez producentów kartom adapterów w celach identyfikacji. Adres MAC zwykle koduje zarejestrowany numer identyfikacyjny producenta.

Standardowy (IEEE 802.3) format zapisu adresu MAC w postaci przyjaznej dla człowieka składa się z sześciu grup po dwie cyfry heksadecymalne każda, oddzielonych od siebie łącznikiem (-) lub dwukropkiem (:), występujących w takiej kolejności, w jakiej są nadawane (na przykład, 01-23-45-67-89-ab lub 01:23:45:67:89:ab).

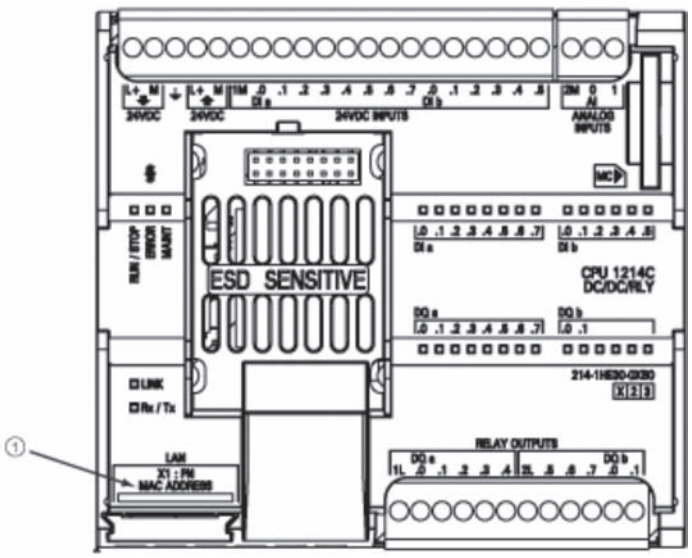
Wszystkie urządzenia zainstalowane w tej samej sieci PROFINET muszą mieć unikalne adresy MAC. Jeżeli w tej samej sieci PROFINET znajdą się dwa urządzenia z tym samym adresem MAC, to pojawią się problemy komunikacyjne.

UWAGA

Każda CPU ma ustalony w fabryce stały, unikalny adres MAC. Użytkowwnik nie może zmienić adresu MAC CPU.

Adres MAC CPU

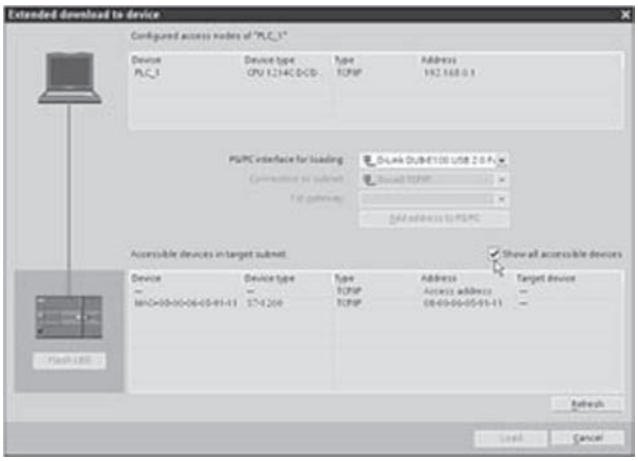
Adres MAC jest naklejony z przodu, w lewym dolnym rogu CPU. W celu odczytania adresu MAC należy unieść dolną osłonę (drzwiczki) TB.



① Adres MAC

Wykorzystanie adresu MAC do identyfikacji urządzeń sieciowych

Początkowo CPU nie ma adresu IP, a tylko fabrycznie zainstalowany adres MAC. Zasady komunikacji PROFINET wymagają by wszystkie urządzenia miały nadane unikalne adresy IP. W celu wyświetlenia wszystkich dostępnych urządzeń sieciowych i sprawdzenia czy mają nadane unikalne adresy IP można wykorzystać funkcję „Download to device" i jej okno dialogowe „Extended download to device". Okno dialogowe wyświetla dostępne urządzenia wraz z ich adresami MAC i IP. W przypadku braku wymaganego adresu IP do identyfikacji urządzeń służą ich adresy MAC.



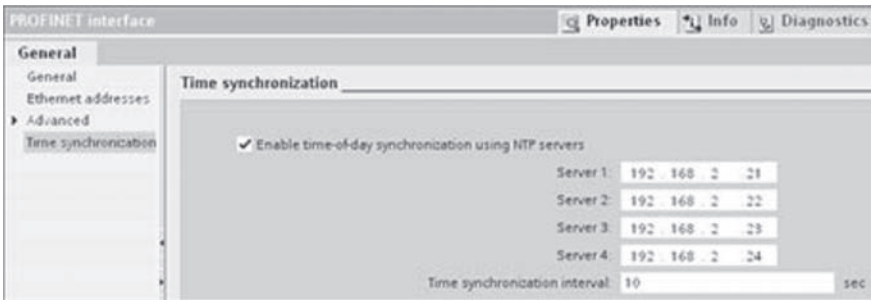
7.4.2 Konfiguracja synchronizacji za pomocą Network Time Protocol

Network Time Protocol (NTP) jest powszechnie stosowany do synchronizacji zegarów systemów komputerowych z serwerami czasu w Internecie. Uzyskiwane dokładności wynoszą zwykle mniej niż milisekundę w sieciach LAN i do kilku milisekund w sieciach WAN. W celu osiągnięcia dużej dokładności i niezawodności typowa konfiguracja NTP wykorzystuje wiele redundantnych serwerów i zróżnicowane ścieżki sieciowe.

Podsieć NTP pracuje z poziomami hierarchicznymi i każdemu poziomowi jest przypisana liczba zwana stratum (warstwa). Serwery stratum 1 (podstawowe) na poziomie najniższym są bezpośrednio synchronizowane z narodowymi służbami czasu. Serwery stratum 2 (wtórne), kolejnego wyższego poziomu są synchronizowane z serwerami stratum 2, i tak dalej.

Parametry synchronizacji czasu

W oknie Properties należy wybrać wejście konfiguracyjne „Time synchronization". Portal TIA wyświetli okno dialogowe konfiguracji „Time synchronization":



UWAGA

Wszystkie adresy IP są konfigurowane podczas ładowania projektu.

Konfigurowane parametry synchronizacji czasu są zdefiniowane poniżej:

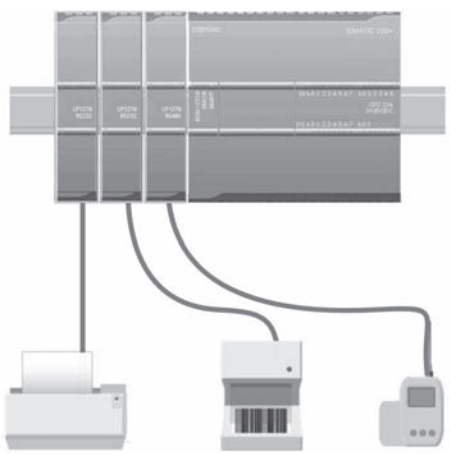
Parametr	Definicja
Uaktywnienie synchronizacji godziny za pomocą serwerów NTP (Network Time Protocol)	W celu uaktywnienia synchronizacji godziny za pomocą serwerów NTP należy kliknąć pole wyboru.
Serwer 1	Przypisany adres IP sieciowego serwera czasu 1.
Serwer 2	Przypisany adres IP sieciowego serwera czasu 2.
Serwer 3	Przypisany adres IP sieciowego serwera czasu 3.
Serwer 4	Przypisany adres IP sieciowego serwera czasu 4.
Interwał synchronizacji czasu	Wartość interwału (w sekundach).

Komunikacja PtP (Point-to-Point) 8

CPU obsługuje protokół PtP (*Point-to-Point*) dla znakowej transmisji szeregowej, podczas której aplikacja użytkownika całkowicie definiuje i implementuje wybrany protokół. PtP zapewnia maksymalną swobodę i elastyczność, ale wymaga implementacji w programie użytkownika.

PtP stwarza szeroki zakres możliwości:

- Pozwala wysyłać informacje bezpośrednio do urządzeń zewnętrznych, takich jak drukarka.
- Pozwala odbierać informacje bezpośrednio z innych urządzeń, takich jak czytniki kodu paskowego.
- Pozwala wymieniać informację, wysyłać i odbierać dane, z innymi urządzeniami, takimi jak GPS, kamery i systemy wizyjne innych producentów, modemami radiowymi i wiele innych.



Komunikacja PtP jest komunikacją szeregową wykorzystującą standardowe urządzenia UART, umożliwiających obsługę wielu prędkości transmisji oraz kontroli parzystości. Moduły komunikacyjne (CM) RS232 i RS485 tworzą interfejs elektryczny dla realizacji komunikacji PtP.

Portal TIA zawiera biblioteki z instrukcjami, które użytkownik może wykorzystywać podczas programowania swoich aplikacji. Te biblioteki oferują funkcje komunikacji PtP dla następujących protokołów:

- Protokół USS sterowania napędami.
- Protokół Modbus RTU *Master*.
- Protokół Modbus RTU *Slave*.

8.1 Wykorzystanie modułów komunikacyjnych RS232 i RS485

Interfejs dla komunikacji PtP zapewniają dwa moduły komunikacyjne:

- CM 1241 RS485
- CM 1221 RS232

Można podłączyć do trzech modułów komunikacyjnych (dowolnego typu). CM należy instalować z lewej strony CPU lub innego CM. Szczegółowe informacje dotyczące instalacji i deinstalacji modułów są podane w rozdziale „Instalacja”.

Moduły komunikacyjne RS232 i RS485 mają następujące charakterystyki:

- Izolowane porty.
- Obsługują protokół *Point-to-Point*.
- Są konfigurowane i programowane za pomocą rozszerzonych instrukcji oraz funkcji bibliotecznych.
- Sygnalizują aktywność nadawania i odbioru diodami LED.
- Są wyposażone w diagnostyczną diodę LED.
- Są zasilane z CPU. Nie jest konieczne zewnętrzne źródło zasilania.

Por. „Dane Techniczne" modułów komunikacyjnych.

8.2 Konfiguracja portów komunikacyjnych

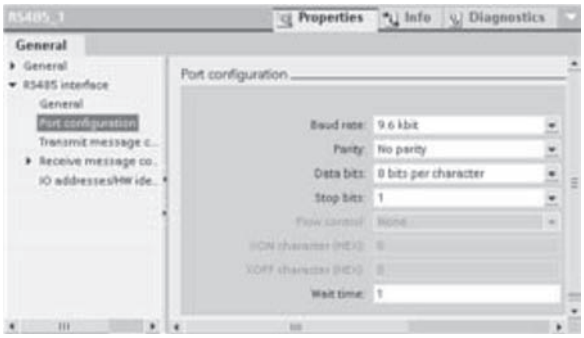
Po wykonaniu konfiguracji sprzętu, należy skonfigurować parametry interfejsów komunikacyjnych, wybierając jeden z zainstalowanych CM.

W zakładce „Properties" okna inspekcyjnego wyświetlane są parametry wybranego CM. Po wybraniu „Port configuration" można edytować następujące parametry:

- Szybkość transmisji
- Parzystość
- Liczbę bitów stopu
- Sterowanie przepływem (tylko RS232)
- Czas oczekiwania

Z wyjątkiem sterowania przepływem, parametry konfiguracyjne portu są takie same, niezależnie od tego, czy konfigurowany jest moduł komunikacyjny RS232, czy też RS485. Wartości parametrów mogą się różnić.

W zakładce „Properties” okna inspekcyjnego wyświetlane są parametry wybranego CM. Po wybraniu „Port configuration” można edytować następujące parametry:



- Szybkość transmisji
- Parzystość
- Liczbę bitów stopu
- Sterowanie przepływem (tylko RS232)
- Czas oczekiwania

Z wyjątkiem sterowania przepływem, parametry konfiguracyjne portu są takie same, niezależnie od tego, czy konfigurowany jest moduł komunikacyjny RS232, czy też RS485. Wartości parametrów mogą się różnić.

Port można również skonfigurować (lub zmienić istniejącą konfigurację) z programu użytkownika za pomocą instrukcji PORT_CFG.

UWAGA

Wartości parametrów ustalone instrukcją PORT_CFG z programu użytkownika, zmieniają (nadpisują) wartości konfiguracyjne ustalone z portalu TIA. W przypadku wyłączenia zasilania lub innej jego utraty, S7-1200 nie zachowuje w pamięci parametrów ustawionych instrukcją PORT_CFG.

Szybkość transmisji (*Baud rate*): Domyślna wartość szybkości transmisji wynosi 9,6 kbps. Poprawnymi wartościami jakie można ustawiać są:

300 bodów	2,4 kbps	19,2 kbps	76,8 kbps
600 bodów	4,8 kbps	28,4 kbps	115,2 kbps
1,2 kbps	9,6 kbps	57,6 kbps	

Parzystość (Parity): Domyślnym ustawieniem jest brak parzystości. Poprawnymi ustawieniami są:

- Brak parzystości
- Parzystość dodatnia
- Parzystość ujemna
- Znak (bit parzystości zawsze równy 1)
- Spacja (bit parzystości zawsze równy 0)

Liczba bitów stopu (*Number of stop bits*): Liczba bitów stopu może wynosić albo jeden albo dwa. Wartością domyślną jest jeden.

Sterowanie przepływem (*Flow control*): W przypadku modułu komunikacyjnego RS232 można wybrać albo sprzętowe, albo programowe sterowanie przepływem, tak jak to opisano w części „Zarządzenie sterowaniem przepływem". Jeżeli zostanie wybrane sprzętowe sterowanie przepływem, to można wybrać czy sygnał RTS jest zawsze aktywny, czy przełączany. Jeżeli zostanie wybrane programowe sterowanie przepływem, to można zdefiniować znaki ASCII, które będą reprezentować znaki XON i XOFF.

Moduł komunikacyjny nie obsługuje sterowania przepływem.

Czas oczekiwania (*Wait time*): Czas oczekiwania jest określany w milisekundach. Zakres wynosi od 0 do 65535 ms. Czas oczekiwania oznacza czas przez jaki moduł komunikacyjny oczekuje na otrzymanie CTS po wystąpieniu RTS lub na XON po odbiorze XOFF – zależnie od rodzaju sterowania przepływem. Jeżeli czas oczekiwania upłynie zanim moduł komunikacyjny otrzyma spodziewany CTS lub XOFF, to moduł komunikacyjny przerywa operację nadawania i zwraca do programu użytkownika błąd.

8.3 Zarządzenie sterowaniem przepływem

Sterowanie przepływem jest to mechanizm takiego zrównoważenia danych nadawanych i odbieranych, by nie utracić żadnej informacji. Sterowanie przepływem zapewnia, że nadajnik nie wysyła więcej informacji niż odbiornik może obsłużyć. Sterowanie przepływem może być zrealizowane albo sprzętowo, albo programowo. Moduł komunikacyjny RS232 obsługuje zarówno sprzętowe, jak i programowe sterowanie przepływem. Moduł komunikacyjny RS485 nie obsługuje sterowania przepływem. Użytkownik określa rodzaj sterowania przepływem podczas konfiguracji portu.

Sprzętowe sterowanie przepływem: przełączany RTS

Sprzętowe sterowanie przepływem odbywa się za pomocą sygnałów RTS (*Request-to-send*) i CTS (*Clear-to-send*). W module komunikacyjnym RS323 RTS jest sygnałem wyjściowym występującym na wyprowadzeniu 7, a CTS sygnałem odbieranym na wyprowadzeniu 8.

Jeżeli w module komunikacyjnym RS232 zostanie uaktywnione sterowanie przepływem z przełączanym sygnałem RTS, to w celu wysłania danych moduł usta-

wia aktywny poziom sygnału RTS. Następnie monitoruje sygnał CTS w celu określenia czy odbiornik jest gotowy zaakceptować dane. Kiedy poziom sygnału CTS jest aktywny, to moduł może nadawać dane tak długo, jak długo CTS pozostaje aktywny. Kiedy poziom CTS zmieni się na nieaktywny, wtedy nadawanie musi zostać wstrzymane.

Nadawanie jest wznawiane po tym, jak CTS przyjmie poziom aktywny. Jeżeli CTS nie przyjmie poziomu aktywnego w ciągu zdefiniowanego czasu oczekiwania, to moduł przerywa nadawanie i zwraca do programu użytkownika błąd. Czas oczekiwania jest definiowany podczas konfiguracji portu.

Sprzętowe sterowanie przepływem: RTS zawsze aktywny

W innej metodzie, urządzenie nadawcze domyślnie ustawia aktywny poziom sygnału RTS. Urządzenie takie jak modem, monitoruje sygnał RTS z CM i wykorzystuje go jako sygnalizację gotowości do nadawania. W tej realizacji sterowania przepływem, modem nadaje dane do CM tylko wtedy, kiedy RTS jest aktywny. Jeśli RTS nie jest aktywny, to moduł nie nadaje danych do CM.

W celu umożliwienia modemowi przesłania danych do CM w dowolnej chwili, należy ustawić sprzętowe sterowanie przepływem z opcją RTS zawsze aktywny. W tym przypadku CM utrzymuje przez cały czas aktywny poziom RTS. CM nie zmieni poziomu RTS na nieaktywny nawet jeśli nie jest w stanie odebrać znaków. Urządzenie nadawcze musi się upewnić, że nie spowoduje przepełnienia bufora odbiorczego CM. Modem może w takiej sytuacji nadawać dane w dowolnej chwili i nie musi monitorować sygnału CTS z odbiornika. Urządzenie nadawcze musi monitorować swoje własne transmisje, ograniczając liczbę ramek wiadomości lub znaków, które wysyła, by uniknąć przepełnienia bufora odbiorczego odbiornika. Jeżeli bufor odbiorczy zostanie jednak przepełniony, to nadajnik musi odrzucić odebraną wiadomość i zwrócić do programu użytkownika błąd.

Wykorzystanie sygnałów DTR (Data Terminal Block Ready) i DSR (Data Set Ready)

CM ustawia aktywny poziom sygnału DTR w obu metodach sprzętowego sterowania przepływem. Moduł nadaje tylko wtedy, kiedy sygnał DSR staje się aktywny. Stan DSR jest sprawdzany tylko na starcie operacji nadawania. Jeśli DSR stanie się nieaktywny po rozpoczęciu nadawania, to nadawanie nie zostanie przerwane.

Programowe sterowanie przepływem

W programowym sterowaniu przepływem są stosowane specjalne znaki w wiadomości sterujące jej transmisją. Są to znaki ASCII reprezentujące XON i XOFF.

XOFF sygnalizuje, że transmisja musi być zatrzymana. XON oznacza, że transmisja może zostać wznowiona.

Kiedy urządzenia nadawcze otrzyma znak XOFF z urządzenia odbiorczego, wtedy zatrzymuje nadawanie. Nadawanie zostaje wznowione po otrzymaniu przez urządzenie nadawcze znaku XON. Jeżeli znak XON nie nadejdzie w ciągu zdefi-

niowanego podczas konfiguracji portu czasu oczekiwania, to CM przerywa nadawanie i zwraca do programu użytkownika błąd.

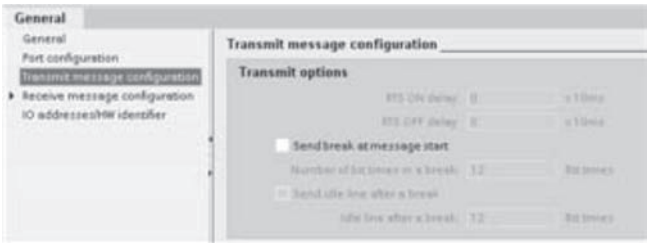
Programowe sterowanie przepływem wymaga pełno dupleksowej komunikacji, ponieważ odbiornik musi mieć możliwość wysłania do nadajnika znaku XOFF w trakcie trwania transmisji.

8.4 Konfiguracja parametrów nadawczych i odbiorczych

Zanim PLC będzie mógł realizować komunikację PtP, należy skonfigurować parametry nadawcze i odbiorcze wiadomości. Te parametry określają w jaki sposób odbywa się transmisja podczas nadawania lub odbierania wiadomości do/z urządzenia docelowego.

Konfiguracja parametrów nadawczych

Podczas konfiguracji CM, konfigurowany jest również sposób w jaki interfejs komunikacyjny nadaje dane; odbywa się to poprzez specyfikację własności „Transmit message configuration" wybranego CM.



Można również z programu użytkownika dynamicznie konfigurować lub zmieniać parametry nadawcze wiadomości za pomocą instrukcji SEND_CFG.

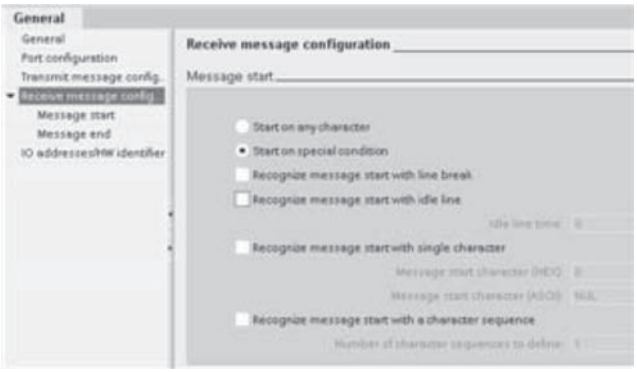
UWAGA

Wartości parametrów ustalone instrukcją SEND_CFG z programu użytkownika, zmieniają (nadpisują) wartości konfiguracyjne portu. W przypadku wyłączenia zasilania lub innej jego utraty, CPU nie zachowuje w pamięci parametrów ustawionych instrukcją SEND_CFG.

Parametr	Definicja
Opóźnienie RTS On RTS On delay	Specyfikuje czas jaki należy odczekać od aktywacji RTS do rozpoczęcia nadawania. Zakres wynosi 0 do 65535 ms; wartość domyślna wynosi 0. Ta wartość ma znaczenie tylko wtedy, kiedy podczas konfiguracji portu ustalono sprzętowe sterowanie przepływem. CTS jest testowany po upływie czasu RTS On delay. Ten parametr stosuje się tylko do modułów RS232.
Opóźnienie RTS Off RTS Off delay	Specyfikuje czas, jaki należy odczekać do deaktywacji RTS po zakończeniu nadawania. Zakres wynosi 0 do 65535 ms; wartość domyślna wynosi 0. Ta wartość ma znaczenie tylko wtedy, kiedy podczas konfiguracji portu ustalono sprzętowe sterowanie przepływem. Ten parametr stosuje się tylko do modułów RS232.
Wysłanie przerwy na początku wiadomości Send break at message start Liczba czasów trwania bitu w przerwie Number of bit times in a break	Specyfikuje, że po upływie czasu RTS On delay i podczas gdy CTS jest aktywny, na początku każdej wiadomości będzie wysyłana przerwa. Użytkownik określa ile czasów trwania bitu tworzy przerwę w formie spacji umieszczanych w wierszu. Wartość domyślna wynosi 12, a maksymalna 65535 aż do osiągnięcia limitu ośmiu sekund.
Wysłanie pustego wiersza po przerwie Send idle line after a break Pusty wiersz po przerwie Idle line after a break	Specyfikuje, że po przerwie na początku wiadomości będzie wysłany pusty wiersz. Parametr „Idle line after a break" określa ile czasów trwania bitu tworzy pusty wiersz w formie znaków. Wartość domyślna wynosi 12, a maksymalna 65535 aż do osiągnięcia limitu ośmiu sekund.

Konfiguracja parametrów odbiorczych

Podczas konfiguracji urządzenia, konfigurowany jest również sposób w jaki interfejs komunikacyjny odbiera dane oraz w jak rozpoznawane są początek i koniec wiadomości. Te parametry są określane poprzez specyfikację własności „Receive message configuration" wybranego CM.



Można również z programu użytkownika dynamicznie konfigurować lub zmieniać parametry odbiorcze wiadomości za pomocą instrukcji RCV_CFG.

UWAGA

Wartości parametrów ustalone instrukcją RCV_CFG z programu użytkownika, zmieniają (nadpisują) wartości konfiguracyjne portu ustalone w portalu TIA. W przypadku wyłączenia zasilania lub innej jego utraty, CPU nie zachowuje w pamięci parametrów ustawionych instrukcją RCV_CFG.

Parametry początku wiadomości

Użytkownik może określić w jaki sposób interfejs komunikacyjny będzie rozpoznawał początek wiadomości. Znak początku wiadomości i znaki tworzące wiadomość trafiają do bufora odbiorczego, aż zostanie spełniony skonfigurowany warunek określający koniec wiadomości.

Można zdefiniować wiele warunków określających start wiadomości. Jeżeli dowolny z tych warunków jest spełniony, to wiadomość jest wysyłana. Konfigurowane parametry są zdefiniowane poniżej:

Parametr	Definicja
Start przy dowolnym znaku Start on any character	Start wiadomości jest wyznaczony przez dowolny znak. Start „on any charakter" jest ustawiany domyślnie.
Start przy spełnieniu specjalnego warunku Start on special condition	Start wiadomości jest wyznaczony przez spełnienie szczególnego warunku wybranego przez użytkownika. Można skonfigurować wiele specjalnych warunków. Kolejność ich sprawdzania jest następująca: • Pusty wiersz • Podział wiersza • Start od pojedynczego znaku lub ciągu znaków
Specjalny warunek: Special condition Rozpoznanie startu wiadomości po podziale wiersza Recognize message start with line break	Oznacza, że odbiornik rozpoznaje podział wiersza jeżeli odebrany wiersz danych składa się ze spacji trwających dłużej niż znak i że ten warunek oznacza start wiadomości.
Specjalny warunek: Special condition: Rozpoznanie startu wiadomości po pustym wierszu Recognize message start with idle line	Oznacza, że start wiadomości jest wskazywany przez pusty wiersz o czasie dłuższym niż wyspecyfikowana krotność czasu trwania bitu, po którym następuje inny warunek, taki jak odebranie dowolnego znaku. Wartością domyślną jest 40 czasów trwania bitu, a maksymalną 65535 z ograniczeniem do ośmiu sekund.

Parametr	Definicja
Specjalny warunek: Special condition: Rozpoznanie startu wiadomości po pojedynczym znaku Recognize message start with single character	Oznacza, że start wiadomości jest wskazywany przez określony znak. Domyślnie jest to STX.
Specjalny warunek: Special condition: Rozpoznanie startu wiadomości po ciągu znaków Recognize message start with a character sequence	Oznacza, że start wiadomości jest wskazywany przez szczególny ciąg znaków. Dla każdego ciągu można określić do pięciu znaków. Dla pozycji każdego znaku można określić znak w postaci heksadecymalnej lub polecenie zignorowania tego znaku podczas sprawdzania ciągu.

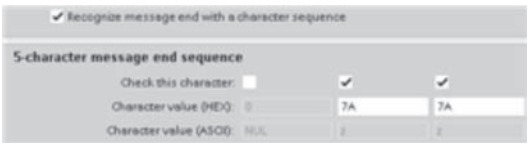
Parametr	Definicja
	Nadchodzące ciągi są sprawdzane pod kątem spełnienia warunku startu wiadomości, aż do momentu spełnienia tego warunku. Kiedy warunek startowy został spełniony, wtedy rozpoczyna się sprawdzanie spełnienia warunku końca wiadomości. Można skonfigurować do pięciu ciągów znaków, które można uaktywniać lub dezaktywować stosownie do potrzeb. Warunek startu wiadomości jest spełniony wtedy, kiedy wystąpi dowolny ze skonfigurowanych ciągów. Przykładowa konfiguracja jest pokazana poniżej:  Przy tej konfiguracji warunek startu wiadomości jest spełniony gdy wystąpi jeden ze wzorów: • Jeśli zostanie odebrany 5-znakowy ciąg, w którym pierwszym znakiem jest 0x6A, a piątym znakiem jest 0x1C. Przy tej konfiguracji znaki na pozycjach 2, 3 i 4 mogą być dowolne. Po odebraniu piątego znaku rozpoczyna się sprawdzanie wystąpienia warunku końca wiadomości. • Jeśli zostaną odebrane kolejno dwa znaki 0x6A, poprzedzone dowolnym znakiem. w tym przypadku sprawdzanie wystąpienia warunku końca wiadomości rozpoczyna się po odebraniu drugiego znaku 0x6A. Znak poprzedzający pierwszy znak 0x6A należy do warunku startu wiadomości. Przykładowe ciągi spełniające warunek startowy: • <dowolny znak> 6A 6A • 6A 12 14 18 1C • 6A 44 A5 D2 1C

Parametry końca wiadomości

Użytkownik może również określić w jaki sposób interfejs komunikacyjny będzie rozpoznawał koniec wiadomości.

Można zdefiniować wiele warunków określających koniec wiadomości. Jeżeli dowolny z tych warunków jest spełniony, to wiadomość jest zakończona.

Parametr	Definicja
Rozpoznanie końca wiadomości po upływie limitu czasu przeznaczonego na wiadomość Recognize message end by message timeout	Koniec wiadomości następuje jeżeli upłynie zdefiniowany limit czasu oczekiwania na zakończenie wiadomości. Zliczanie czasu rozpoczyna się wtedy, kiedy po spełnieniu kryteriów startu wiadomości zostaje odebrany pierwszy znak. Wartość domyślna tego limitu czasu wynosi 200 ms, a zakres wynosi od 0 do 65535 ms.
Rozpoznanie końca wiadomości po upływie limitu czasu przeznaczonego na odpowiedź Recognize message end by response timeout	Koniec wiadomości następuje jeżeli zanim zostanie odebrana poprawna sekwencja startowa upłynie zdefiniowany limit czasu oczekiwania na odpowiedź. Zliczanie czasu rozpoczyna się wtedy, kiedy kończy się transmisja. Wartość domyślna limitu czasu oczekiwania na odpowiedź wynosi 200 ms, a zakres wynosi od 0 do 65535 ms. Aby wskazać rzeczywisty koniec wiadomości użytkownik musi skonfigurować inny warunek końcowy.
Rozpoznanie końca wiadomości po przerwie między znakami Recognize message end by inter-character gap	Koniec wiadomości następuje jeżeli upłynie maksymalny ustalony limit czasu pomiędzy kolejnymi znakami. Domyślna wartość przerwy między znakami wynosi 12 czasów trwania bitu, a wartość maksymalna 65535 czasów trwania bitu, maksymalnie do ośmiu sekund.
Rozpoznanie końca wiadomości po maksymalnej długości Recognize message end by max length	Koniec wiadomości następuje jeżeli zostanie odebrana maksymalna, zdefiniowana liczba znaków. Niezależnie od tego, czy maksymalna liczba znaków została osiągnięta, to o ile zostanie spełniony jeszcze inny warunek końcowy, poprawne znaki zostaną włączone do wiadomości. Wartość domyślna wynosi 0 bajtów, a maksymalna 1024 bajty.
Odczyt z wiadomości długości wiadomości Read message length from message	Wiadomość zawiera informację o swojej długości. Koniec wiadomości następuje po otrzymaniu wiadomości o określonej długości. Sposób specyfikacji i interpretacji długości wiadomości jest podany poniżej.
Rozpoznanie końca wiadomości po znaku Recognize message end with a character	Koniec wiadomości następuje jeżeli zostanie odebrany zdefiniowany znak.

Rozpoznanie końca wiadomości po ciągu znaków Recognize message end with a character sequence	Koniec wiadomości następuje jeżeli zostanie odebrany zdefiniowany ciąg znaków. Można określić ciąg do pięciu znaków. Dla pozycji każdego znaku można określić znak w postaci heksadecymalnej lub polecenie zignorowania tego znaku podczas sprawdzania ciągu. Wiodące znaki, które są znakami ignorowanymi nie są częścią warunku końcowego. Końcowe znaki, które są znakami ignorowanymi są częścią warunku końcowego. Konfiguracja jednego z takich ciągów jest pokazana poniżej:  W tym przypadku warunek jest spełniony gdy zostaną odebrane dwa kolejne znaki 0x7A, po których nastąpią dwa dowolne znaki. Znaki poprzedzające wzór 0x7A 0x7A nie są częścią ciągu końcowego znaków. Dwa znaki następujące po wzorze 0x7A 0x7A są wymagane do zakończenia ciągu końcowego znaków. Wartości na pozycjach znaków 4 i 5 są nieistotne, ale muszą zostać odebrane by spełnić warunek końcowy.

Specyfikacja długości wiadomości wewnątrz wiadomości

Jeśli zostanie wybrany specjalny warunek wymagający włączenia do wiadomości informacji o długości wiadomości, to należy określić trzy parametry określające długość wiadomości.

Faktyczna struktura wiadomości zmienia się w zależności od zastosowanego protokołu. Wymienione trzy parametry to:

- *n*: pozycja znaku w wiadomości zawierającego informację o długości.
- *Length size* (rozmiar długości): Liczba bajtów (jeden, dwa lub cztery) zawierających informację o długości.
- *Length m* (długość m): liczba znaków następujących po informacji o długości, które nie są wliczone do długości.

Te pola pojawiają w oknie „Receive Message Configuration” menu „Device Properties” w następujący sposób:

Przykład 1: Struktura wiadomości jest zgodna z następującym protokołem:

STX	**Len (n)**	**Znaki 3 do 14 wliczone do długości**											
		ADR	**PKE**		**INDEX**		**PWD**		**STW**		**HSW**		**BCC**
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
STX	0x0C	xx	xxxx		xxxx		xxxx		xxxx		xxxx		xx

Parametry długości informacji dla tej wiadomości powinny być skonfigurowane następująco:

- *n* = 2 (Długość wiadomości jest podana w bajcie 2.)
- *Length size* = 1 (Długość wiadomości jest określona jednym bajtem.)
- *Length m* = 0 (Po informacji o długości wiadomości nie występują dodatkowe znaki, które nie są wliczane do długości. Po informacji o długości wiadomości znajduje się 12 znaków.)

W tym przykładzie znaki od 3 do 14 włącznie są wliczone do długości wiadomości *Len*(*n*).

Przykład 2: Rozważmy inną wiadomość zgodną z następującym protokołem:

SD1	Len (n)	Len (n)	SD2	Znaki 5 do 10 wliczone do długości						FCS	ED
				DA	SA	FA	Jednostka danych = 3 bajty				
1	2	3	4	5	6	7	8	9	10	11	12
xx	0x06	0x06	xx	xx	xx	xx	xx	xx	xx	xx	xx

Parametry długości informacji dla tej wiadomości powinny być skonfigurowane następująco:

- *n* = 3 (Długość wiadomości jest podana w bajcie 3.)
- *Length size* = 1 (Długość wiadomości jest określona jednym bajtem.)
- *Length m* = 3 (Po informacji o długości wiadomości występują 3 znaki, które nie są wliczane do długości. W protokole podanym w tym przykładzie znaki SD2, FCS i ED nie są wliczone do długości wiadomości. Pozostałe sześć znaków jest wliczonych do długości wiadomości; tak więc całkowita liczba znaków występujących po informacji o długości wiadomości wynosi dziewięć.)

W tym przykładzie znaki od 5 do 10 włącznie są wliczone do długości wiadomości *Len*(*n*).

8.5 Programowanie komunikacji PtP

Portal TIA udostępnia rozszerzone instrukcje, które pozwalają z programu użytkownika realizować komunikację *Point-to-Point* za pomocą dowolnego *Siemens Provided Protocols* oraz protokołu Freeport. Te instrukcje można rozpatrywać w dwóch kategoriach:

- Instrukcje konfiguracyjne
- Instrukcje komunikacyjne

Instrukcje konfiguracyjne

Zanim program użytkownika będzie mógł zostać wykorzystany do realizacji komunikacji PtP, należy skonfigurować port interfejsu komunikacyjnego oraz parametry niezbędne do nadawania i odbierania danych.

Konfiguracji portu i wiadomości dla każdego modułu komunikacyjnego można dokonać poprzez konfigurację urządzenia lub za pomocą następujących instrukcji z programu użytkownika:

- PORT_CFG
- SEND_CFG
- RCV_CFG

Instrukcje komunikacyjne

Instrukcje komunikacyjne PtP pozwalają z programu użytkownika wysyłać i odbierać wiadomości za pośrednictwem modułów komunikacyjnych.

Moduły komunikacyjne z kolei wysyłają wiadomości do i odbierają je z rzeczywistych urządzeń PtP. Protokół wiadomości znajduje się w buforze, który jest albo odbierany z albo wysyłany do określonego portu komunikacyjnego.

- SEND_PTP
- RCV_PTP

Wszystkie funkcje PtP działają asynchronicznie. W celu określenia stanu nadawania i odbioru w programie użytkownika można zastosować architekturę odpytywania (*polling*). Instrukcje SEND_PTP i RCV_PTP można wykonywać jednocześnie. Moduły komunikacyjne buforują wiadomości nadawane i odbierane zgodnie z potrzebami, aż do maksymalnej pojemności bufora wynoszącej 1024 bajty.

Dodatkowe instrukcje pozwalają kasować bufor odbiorczy oraz pobierać i ustawiać określone sygnały RS232.

- RCV_RST
- SGN_GET
- SGN_SET

8.5.1 Architektura odpytywania (polling)

W celu zaimplementowania architektury odpytywania, w programie użytkownika należy sprawdzać stan każdej wiadomości, którą program wysyła lub odbiera. Odpytywanie jest zwykle wykonywane jako wątek programu głównego.

Architektura odpytywania: Master

Typowa sekwencja odpytywania dla *Master* jest następująca:

1. Instrukcja SEND_PTP inicjuje nadawanie do modułu komunikacyjnego.
2. Powtarzane instrukcje SEND_PTP monitorują status transmisji i uzyskują informację o jej bieżącym statusie.
3. Po tym, jak status wskaże, że transmisja jest zakończona, instrukcje zależne od aplikacji przygotowują odbiór oczekiwanej odpowiedzi na przesłaną wiadomość.
4. Instrukcja RCV_PTP wykonywana wielokrotnie sprawdza nadejście odpowiedzi na wysłane dane. Po tym jak zostaną spełnione warunki określające zakończenie odbioru odpowiedzi jest ustawiany status i bufor odbiorczy zawiera odebraną wiadomość.

Architektura odpytywania: Slave

Typowa sekwencja odpytywania dla *Slave* jest następująca:

1. Instrukcja RCV_PTP oczekuje transmisji z *Master*, która jest rozpoznawana po spełnieniu skonfigurowanych warunków początku wiadomości.

2. Po spełnieniu warunków końca wiadomości, co wskazuje na zakończenie odbioru, instrukcje zależne od aplikacji przetwarzają otrzymaną wiadomość i przygotowują odpowiedź do wysłania.
3. Instrukcja SEND_PTP wysyła odpowiedź do *Master*.

Slave musi być zdolny do wykonywania instrukcji RCV_PTP dostatecznie często, aby odebrać wiadomość nadawaną z *Master* przed upływem limitu czasu oczekiwania na odpowiedź ustawionym dla *Master*. W celu wykonania tego zadania program użytkownika może wywoływać RCV_PTP z cyklicznego OB, którego czas cyklu jest wystarczający do odebrania transmisji z *Master* przed upływem tego limitu czasu. Jeżeli czas cyklu OB zostanie ustawiony tak, by nastąpiły dwa wykonania w czasie równym okresowi limitu *Master*, to program użytkownika powinien odebrać wszystkie nadawane informacje bez żadnej straty.

8.6 Instrukcje Point-to-Point

8.6.1 Parametry wspólne dla instrukcji Point-to-Point

Diody LED modułu komunikacyjnego

W module komunikacyjnym (CM) znajdują się trzy diody LED:

- Diagnostyczna dioda LED (*Diagnostic LED*).
  Diagnostyczna dioda LED błyska na czerwono dopóty, dopóki nie zostanie zaadresowana przez CPU. Po włączeniu zasilania, CPU sprawdza jakie moduły są zainstalowane i adresuje moduł CM. Diagnostyczna dioda LED zaczyna wtedy błyskać na zielono. Oznacza to, że CPU zaadresowała CM, ale jeszcze nie przesłała do niego danych konfiguracyjnych. Dane konfiguracyjne są ładowane do modułu po załadowaniu programu do CPU. Gdy to nastąpi diagnostyczna dioda LED powinna świecić w sposób ciągły na zielono.
- Nadawcza dioda LED (*Transmit LED*).
  Nadawcza dioda LED jest umieszczona nad odbiorczą diodą LED. Nadawcza dioda LED świeci podczas nadawania danych przez port komunikacyjny.
- Odbiorcza dioda LED (Receive LED).
  Odbiorcza dioda LED świeci podczas odbierania danych przez port komunikacyjny.

Rozdzielczość czasu trwania bitu (bit time resolution)

Kilka parametrów jest specyfikowanych w wielokrotnościach czasu trwania bitu. Wszystkie te parametry, które są wyrażane w jednostkach czasu trwania bitu mogą osiągać maksymalną wartość 64000. Jednakże maksymalny przedział czasu jaki może być zmierzony przez S7-1200 wynosi 8 sekund.

Parametr wejściowy REQ

Wiele instrukcji *Point-to-Point* (PtP) wykorzystuje wejście REQ, które inicjalizuje operację w chwili zmiany stanu z niskiego na wysoki. Wejście REQ musi być w stanie wysokim (TRUE) przez jedno wykonanie instrukcji i może trwać w stanie TRUE tak długo, jak jest to potrzebne. Ta instrukcja nie zainicjuje żadnej innej

operacji dopóty, dopóki nie zostanie wywołana podczas gdy wejście REQ jest w stanie FALSE tak, że instrukcja może skasować stan historii wejścia REQ. Wymagane jest, że dla rozpoczęcia kolejnej operacji instrukcja może rozpoznać zmianę stanu z niskiego na wysoki.

W chwili umieszczania instrukcji REQ użytkownik jest pytany o określenie instancji DB. Dla każdego typu instrukcji PtP należy używać unikalnego DB. Oznacza to, że wszystkie instrukcje SEND_PTP powinny mieć tą samą instancję DB, ale SEND_PTP i RCV_PTP muszą mieć różne instancje DB. Dzięki temu jest pewność, że wejścia takie jak REQ są właściwie obsługiwane przez każdą instrukcję.

Parametr wejściowy PORT

Z rozwijanego menu (związanego z wejściem PORT) należy wybrać identyfikator portu dla CM, który ma pracować z tym blokie, instrukcji. Tę liczbę można odnaleźć pod nazwą „hardware identifier" (identyfikator sprzętowy) w informacjach konfiguracyjnych dotyczących danego CM.

Parametry wejściowe NDR, DONE, ERROR i STATUS

- Parametr wyjściowy DONE sygnalizuje, że żądana operacja zakończyła się bez błędu. Ta wartość jest ustawiana na jeden cykl programu.
- Parametr wyjściowy NDR (New Data Ready) sygnalizuje, że żądana akcja zakończyła się bez błędu i zostały odebrane nowe dane. To wyjście jest ustawiane na jeden cykl programu.
- Parametr wyjściowy ERROR sygnalizuje, że żądana akcja zakończyła się z błędem. To wyjście jest ustawiane na czas jednego cyklu programu.
- Parametr wyjściowy STATUS jest wykorzystywany do raportowania błędów lub pośrednich wyników statusu.
  - Jeżeli jest ustawiony bit DONE lub NDR, to STATUS przyjmuje wartość 0.
  - Jeżeli jest ustawiony bit ERROR, to STATUS zawiera kod błędu.
  - Jeżeli nie jest ustawiony żaden z powyższych bitów, to wykonanie instrukcji może zwrócić pośredni wynik statusu prezentujący bieżący stan funkcji.

Organizacja danych parametru STATUS

W przypadku błędów ogólnych:

15							**8**	**7**			**4**	**3**			**0**
1	Numer parametru							0	Numer zdarzenia						

W przypadku błędów związanych z PtP:

15							**8**	**7**			**4**	**3**			**0**
1	0							1	Klasa błędu			Numer błędu			

W przypadku informacji o statusie:

15							8	7			4	3			0
0	Kody statusu														

Wspólne kody warunkowe

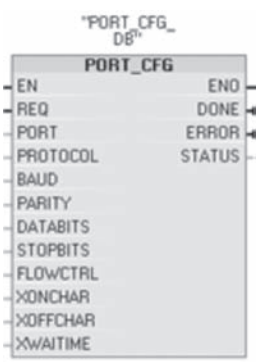
STATUS (W#16#....)	Opis
0000	Brak błędu
8x3A	Nielegalny wskaźnik w parametrze x
8070	Cała pamięć wewnętrzna jest w użyciu
8080	Nieprawidłowy numer portu
8081	Przekroczony limit czasu, błąd modułu lub inny błąd wewnętrzny
8082	Parametryzacja nie powiodła się, ponieważ parametryzacja jest aktualnie wykonywana w tle
8083	Przepełnienie bufora: CM zwrócił otrzymaną wiadomość o długości większej niż na to pozwala parametr Length
8090	Błędna długość wiadomości, niewłaściwy podmoduł lub nieprawidłowa wiadomość
8091	Błędna wersja wiadomości parametryzacji
8092	Błędna długość rekordu wiadomości parametryzacji

8.6.2 Instrukcja PORT_CFG

Instrukcja PORT_CFG (*Port Configuration*) umożliwia zmianę parametrów portu, takich jak szybkość transmisji, z programu użytkownika.

Początkowa, statyczna konfiguracja portu jest ustalana w trakcie konfiguracji własności sprzętu. Tę konfigurację można zmienić za pomocą instrukcji PORT_CFG uruchomionej z programu użytkownika. Dynamiczne zmiany konfiguracji poczynione za pomocą PORT_CFG nie są trwale pamiętane w PLC.

LAD



Parametr	TyP parametru	Typ danych	Opis
REQ	IN	BOOL	Aktywuje zmianę konfiguracji w momencie wystąpienia narastającego zbocza na tym wejściu.
PORT	IN	PORT	Identyfikator portu komunikacyjnego: Ten adres logiczny jest stałą, która znajduje się w zakładce „Constants" domyślnej tablicy tagów.
PROTOCOL	IN	UINT	0 – protokół komunikacyjny Point-to-Point. 1..n – do wykorzystania w przyszłości.
BAUD	IN	UINT	Szybkość transmisji portu: 1 - 300 bodów 2 - 600 bodów 3 - 1200 bodów 4 - 2400 bodów 5 - 4800 bodów 6 - 9600 bodów 7 - 19200 bodów 8 - 38400 bodów 9 - 57600 bodów 10 - 76800 bodów 11 - 115200 bodów
PARITY	IN	UINT	Parzystość: 1 – bez bitu parzystości 2 – bit parzystości 3 – bit nieparzystości 4 – bit parzystości zawsze 1 5 – bit parzystości zawsze 0
DATABITS	IN	UINT	Liczba bitów na znak: 1 - 8 bitów danych 2 - 7 bitów danych
STOPBITS	IN	UINT	Bity stopu: 1 - 1 bit stopu 2 - 2 bity stopu
FLOWCTRL	IN	UINT	Sterowanie przepływem: 1 – brak sterowania przepływem 2 – XON/XOFF 3 – sprzętowe RTS zawsze ON 4 – sprzętowe RTS przełączany
XONCHAR	IN	CHAR	Specyfikuje jaki znak jest używany jako XON. Zwykle jest to znak DC1 (11H). Ten parametr jest wykorzystywany tylko wtedy, kiedy sterowanie przepływem jest uaktywnione.
XOFFCHAR	IN	CHAR	Specyfikuje jaki znak jest używany jako XOFF. Zwykle jest to znak DC3 (13H). Ten parametr jest wykorzystywany tylko wtedy, kiedy sterowanie przepływem jest uaktywnione.

Parametr	TyP parametru	Typ danych	Opis
XWAITIME	IN	UINT	Specyfikuje czas oczekiwania na znak XON po odebraniu znaku XOFF lub czas oczekiwania na sygnał CTS po uaktywnieniu RTC. Ten parametr jest wykorzystywany tylko wtedy, kiedy sterowanie przepływem jest uaktywnione.
DONE	OUT	BOOL	Przyjmuje wartość TRUE na jeden cykl programu jeśli ostatnie żądanie zostało zakończone bez błędu.
ERROR	OUT	BOOL	Przyjmuje wartość TRUE na jeden cykl programu jeśli ostatnie żądanie zostało zakończone z błędem.
STATUS	OUT	WORD	Kod błędu.

Kody warunkowe

STATUS (W#16#....)	Opis
80A0	Określony protokół nie istnieje.
80A1	Określona prędkość transmisji nie istnieje.
80A2	Określona opcja parzystości nie istnieje.
80A3	Określona liczba bitów danych nie istnieje.
80A4	Określona liczba bitów stopu nie istnieje.
80A5	Określony typ sterowania przepływem nie istnieje.

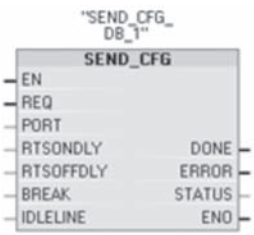
8.6.3 Instrukcja SEND_CFG

Instrukcja SEND_CFG (*Send Configuration*) pozwala dynamicznie konfigurować parametry transmisji szeregowej portu komunikacyjnego *Point-to-Point*. Wszystkie wiadomości oczekujące na transmisję poprzez moduł komunikacyjny (CM) będą po wykonaniu instrukcji SEND_CFG odrzucone.

Początkowa, statyczna konfiguracja portu jest ustalana w trakcie konfiguracji urządzenia. Tę konfigurację można zmienić za pomocą instrukcji SEND_CFG uruchomionej z programu użytkownika. Dynamiczne zmiany konfiguracji poczynione za pomocą SEND_CFG nie są trwale pamiętane w PLC.

LAD

FBD



Parametr	Typ	Typ	Opis
REQ	IN	BOOL	Aktywuje zmianę konfiguracji w momencie wystąpienia narastającego zbocza na tym wejściu.
PORT	IN	PORT	Identyfikator portu komunikacyjnego: Ten adres logiczny jest stałą, która znajduje się w zakładce „Constants” domyślnej tablicy tagów.
RTSONDLY	IN	UINT	Czas oczekiwania w milisekundach po uaktywnieniu RTS zanim nastąpi transmisja danych Tx: Ten parametr jest wykorzystywany tylko wtedy, kiedy jest uaktywnione sprzętowe sterowanie przepływem.
RTSOFFDLY	IN	UINT	Czas oczekiwania w milisekundach po wystąpieniu transmisji danych Tx, zanim RTS zostanie dezaktywowany: Ten parametr jest wykorzystywany tylko wtedy, kiedy jest uaktywnione sprzętowe sterowanie przepływem.
BREAK	IN	UINT	Ten parametr określa, że na początku każdej wiadomości, przez określoną wielokrotność czasu trwania bitu, będzie przesyłana przerwa. Wartością maksymalną jest 2500 czasów trwania bitu.
IDLELINE	IN	UINT	Ten parametr określa, że przed rozpoczęciem każdej wiadomości, przez określoną wielokrotność czasu trwania bitu, będzie przesyłany pusty wiersz. Wartością maksymalną jest 2500 czasów trwania bitu.
DONE	OUT	BOOL	Przyjmuje wartość TRUE na jeden cykl programu jeśli ostatnie żądanie zostało zakończone bez błędu.
ERROR	OUT	BOOL	Przyjmuje wartość TRUE na jeden cykl programu jeśli ostatnie żądanie zostało zakończone z błędem.
STATUS	OUT	WORD	Kod błędu.

Kody warunkowe

STATUS (W#16#....)	Opis
80B0	Konfiguracja przerwania transmisji nie jest dozwolona.
80B1	Czas trwania przerwy jest większy niż dozwolona wartość (2500 czasów trwania bitu).
80B2	Czas trwania pustego wiersza jest większy niż dozwolona wartość (2500 czasów trwania bitu).

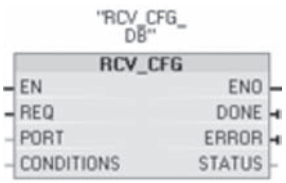
8.6.4 Instrukcja RCV_CFG

Instrukcja RCV_CFG (*Receive Configuration*) wykonuje dynamiczną konfigurację parametrów szeregowego odbiornika portu komunikacyjnego Point-to-Point. Instrukcja konfiguruje warunki, które sygnalizują początek i koniec odbieranej wiadomości. Wiadomości, które spełniają te warunki będą odbierane za pomocą instrukcji RCV_PTP.

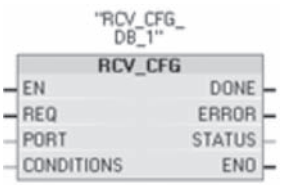
Wszystkie wiadomości oczekujące na transmisję poprzez moduł komunikacyjny (CM) będą po wykonaniu instrukcji RCV_CFG odrzucone.

Początkowa, statyczna konfiguracja portu jest ustalana w trakcie konfiguracji urządzenia. Tę konfigurację można zmienić za pomocą instrukcji RCV_CFG uruchomionej z programu użytkownika. Dynamiczne zmiany konfiguracji poczynione za pomocą RCV_CFG nie są trwale pamiętane w PLC; tak więc po wykonaniu cyklu wyłączenia/włączenia zasilania będzie wykorzystana początkowa, statyczna konfiguracja pochodząca z konfiguracji urządzenia.

LAD



FBD



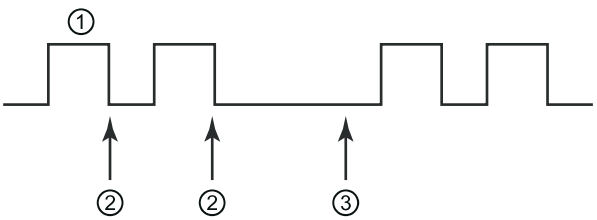
Parametr	Typ parametru	Typ danych	Opis
REQ	IN	BOOL	Aktywuje zmianę konfiguracji w momencie wystąpienia narastającego zbocza na tym wejściu.
PORT	IN	PORT	Identyfikator portu komunikacyjnego: Ten adres logiczny jest stałą, która znajduje się w zakładce „Constants” domyślnej tablicy tagów.
CONDITIONS	IN	Block_SDT	Struktura danych Conditions określa warunki początku i końca wiadomości. Są one opisane poniżej.
DONE	OUT	BOOL	Przyjmuje wartość TRUE na jeden cykl programu jeśli ostatnie żądanie zostało zakończone bez błędu.
ERROR	OUT	BOOL	Przyjmuje wartość TRUE na jeden cykl programu jeśli ostatnie żądanie zostało zakończone z błędem.
STATUS	OUT	WORD	Kod błędu.

Warunki początku wiadomości dla instrukcji RCV_PTP

Instrukcja RCV_PTP wykorzystuje konfigurację określoną przez instrukcję RCV_CFG, definiującą początek i koniec wiadomości podczas komunikacji PtP. Początek wiadomości jest określony przez warunki startowe. Początek wiadomości może określać jeden lub kombinacja kilku warunków startowych. Jeśli wyspecyfikowano więcej niż jeden warunek startowy, to pierwszy, który jest spełniony oznacza początek wiadomości.

Możliwe warunki startowe:

- Start Character (znak startu)
  - Warunek *Start Character* określa, że poprawne odebranie określonego znaku wyznacza początek wiadomości. Ten znak jest pierwszym znakiem wiadomości. Wszystkie znaki odebrane przed tym specyficznym znakiem są pomijane.
- Any Character (dowolny znak)
  - Warunek *Any Character* określa, że poprawne odebranie dowolnego wyznacza początek wiadomości. Ten znak jest pierwszym znakiem wiadomości.
- Line Break (podział wiersza)
  - Warunek *Line Break* określa, że operacja odbierania wiadomości powinna się rozpocząć wtedy, kiedy odbierany stan wiersza trwa dłużej niż całkowity czas trwania znaku.
- Line Idle (pusty wiersz[mk3])
  - Warunek *Line Idle* określa, że odbiór wiadomości powinien rozpocząć się po tym jak odebrany wiersz był pusty przez określoną liczbę czasów trwania bitu. Wystąpienie tego warunku oznacza rozpoczęcie wiadomości.



① Znaki
② Restart licznika czasu trwania wiersza pustego
③ Wykrycie wiersza pustego i początek odbioru wiadomości

- Zmienne sekwencje
  - Warunki startowe mogą być tak konstruowane, by były zależne od zróżnicowanej liczby ciągów znaków (maksymalnie do 4) składających się ze zmiennej liczby znaków (maksymalnie do 5). Znak na każdej pozycji może być wyspecyfikowany jako konkretny znak lub symbol zastępujący dowolny znak. Ten warunek startowy może być używany wtedy, kiedy start wiadomości jest sygnalizowany różnymi ciągami znaków.

Rozpatrzmy następującą odebraną wiadomość zakodowana heksadecymalnie: „68 10 aa 68 bb 10 aa 16” oraz skonfigurowane sekwencje startowe przedstawione w tabeli poniżej. Sprawdzanie sekwencji startowych rozpoczyna się po pomyślnym odebraniu pierwszego znaku 68H. Po pomyślnym odebraniu czwartego znaku (drugi 68H) spełniony jest pierwszy warunek startowy. Po pomyślnym odebraniu piątego znaku (bbH) rozpoczyna się badanie spełnienia warunku końcowego.

– Przetwarzanie sekwencji startowej może być zakończone w związku z różnymi błędami (parzystości, ramkowania lub timingu międzyznakowego). W wyniku tych błędów nie jest odebrana żadna wiadomość, ponieważ nie zostaje spełniony warunek startowy.

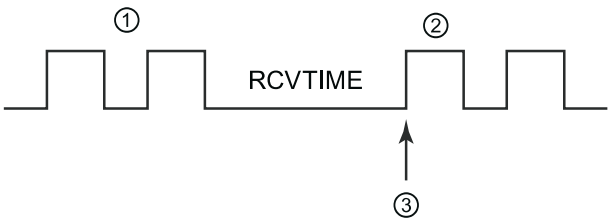
Warunek startowy	Pierwszy znak	Pierwszy znak +1	Pierwszy znak +2	Pierwszy znak +3	Pierwszy znak +4
1	68H	xx	xx	68H	xx
2	10H	aaH	xx	xx	xx
3	dcH	aaH	xx	xx	xx
4	e5H	xx	xx	xx	xx

Warunki końca wiadomości dla instrukcji RCV_PTP

Koniec wiadomości jest określony przez warunki końcowe. Koniec wiadomości jest określony przez pierwsze wystąpienie jednego lub więcej skonfigurowanych warunków końcowych.

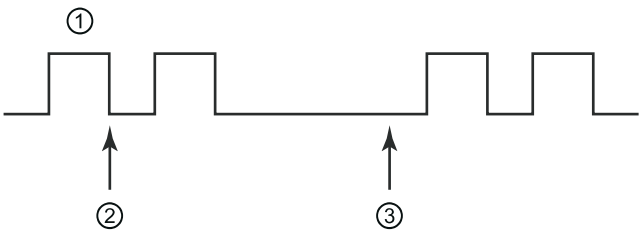
Możliwe warunki końcowe:

- Upływ limitu czasu na odpowiedź.
  – Warunek limitu czasu na odpowiedź określa, że dowolny znak powinien zostać poprawnie odebrany w ciągu czasu wyspecyfikowanego przez RCVTIME. Licznik czasu zaczyna pracować w momencie pomyślnego zakończenia nadawania, gdy moduł rozpoczyna operację odbierania. Jeżeli znak nie zostanie odebrany w ciągu czasu RCVTIME, to do instrukcji RCV_PTP zwracany jest błąd. Limit czasu na odpowiedź nie definiuje konkretnego warunku końcowego. Specyfikuje tylko, że znak powinien zostać odebrany w określonym czasie. Trzeba zatem korzystać z odrębnego warunku końca wiadomości.



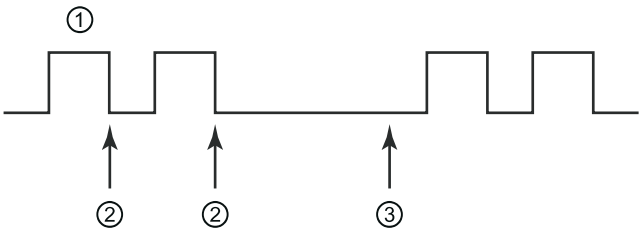
① Wysłane znaki
② Odebrane znaki
③ Pierwszy znak musi być pomyślnie odebrany do tego czasu

- Limit czasu wiadomości
  – Warunek limitu czasu wiadomości określa, że wiadomość powinna zostać odebrana w czasie wyspecyfikowanym przez MSGTIME. Licznik czasu zaczyna pracować w momencie spełnienia określonego warunku startowego wiadomości.



① Odebrane znaki
② Spełnienie warunku początku wiadomości: timer wiadomości zaczyna pracę
③ Timer wiadomości zliczył ustalony czas i kończy wiadomość

- Przerwa międzyznakowa
  – Czas przerwy międzyznakowej jest to czas mierzony od zakończenia jednego znaku (ostatniego bitu stopu) do zakończenia następnego znaku. Jeżeli czas między dwoma znakami przekroczy ustaloną krotność czasu trwania bitu, to wiadomość jest kończona.



① Odebrane znaki
② Restart timera międzyznakowego
③ Timer międzyznakowy zliczył ustalony czas i kończy wiadomość z błędem

- Maksymalna długość
  – Operacja odbierania zakończy się po odebraniu określonej liczby znaków. Ten warunek może być wykorzystany w celu zabezpieczenia bufora wiadomości przez przepełnieniem. Kiedy ten warunek końcowy jest połączony z warunkiem końcowym przekroczenia limitu czasu wiadomości i wystąpi przekroczenie limitu czasu, wtedy wszystkie poprawnie odebrane znaki zostają zachowane, nawet jeśli nie została osiągnięta maksymalna długość. Pozwala to obsługiwać protokoły o różnej długości jeśli tylko znana jest maksymalna długość.

- Warunek łączny „N + Length Size + Length M”. Ten warunek końcowy może być wykorzystany do przetwarzania wiadomości o zmiennej długości, które zawierają pole długości. Parametry są opisane poniżej.
  - *N* oznacza pozycję (liczbę znaków w wiadomości) od której zaczyna się pole długości.
  - *Length Size* określa rozmiar pola długości. Poprawne wartości to 1, 2 lub bajty.
  - *Length M* określa liczbę znaków końcowych (występujących za polem długości), które nie są wliczone do długości wiadomości. Przykładowo, może być wykorzystana do wyspecyfikowania długości pola sumy kontrolnej, której wielkość nie jest uwzględniona w polu długości. Rozważmy na przykład format wiadomości składającej się ze znaku startu, znaku adresu, 1-bajtowego pola długości, treści wiadomości, znaków sumy kontrolnej i znaku końca.
  - Dane identyfikowane przez „Len” są powiązane z parametrem N. Wartość N jest tu równa 3 i określa, że bajt długości jest trzecim bajtem wiadomości. Wartość Length Size wynosi 1 co oznacza, że wartość określająca długość wiadomości jest zawarta w jednym bajcie. Suma kontrolna i znak końca są powiązane z parametrem „Length M”. Wartość „Length M” wynosi 3 określając, że liczba końcowych pól zajmuje 3 bajty.

Znak startu (1)	**Adres (2)**	**Len (N) (3)**	**Wiadomość … (x)**		**Suma kontrolna i znak końca Length M x+1 x+2 x+3**		
xx	xx	xx	xx	xx	xx	xx	xx

- Znaki zmienne
  - Ten warunek końcowy może być użyty do zakończenia odbioru w wyniku różnych sekwencji znaków. Te sekwencje mogą się składać ze zmiennej liczby znaków (maksymalnie do 5). Znak na każdej pozycji, w każdej sekwencji może być wyspecyfikowany jako konkretny znak lub symbol zastępujący dowolny znak co oznacza, że dowolny znak spełnia warunek. Znaki wiodące skonfigurowane do pominięcia nie muszą być częścią wiadomości. Każdy znak końcowy skonfigurowany do pominięcia musi być częścią wiadomości.

Struktura danej typu CONDITIONS, część 1 (warunki startu)

Parametr	Typ parametru	Typ danych	Opis
STARTCOND	IN	UINT	Specyfikuje warunki startu: • 1- Znak startu • 16- Sekwencja 1 • 2- Dowolny znak • 32- Sekwencja 2 • 4- Podział wiersza • 64- Sekwencja 3 • 8- Wiersz pusty • 128- Sekwencja 4
IDLETIME	IN	UINT	Krotność czasu trwania bitu wymagana dla upływu limitu czasu wiersza pustego. Używana wyłącznie z warunkiem wiersza pustego.
STARTCHAR	IN	BYTE	Znak startu używany z warunkiem znaku startu.
STRSEQ1CTL	IN	BYTE	Sterowanie ignoruj/porównaj każdego znaku sekwencji 1: To są bity uaktywniające dla każdego znaku sekwencji startowej. Dla znaku 1 jest bit 0, dla znaku 2 jest bit 1, … , dla znaku 5 jest bit 4. Zablokowanie bitu związanego ze znakiem oznacza, że dowolny znak na tej pozycji jest identyfikowany jako właściwy.
STRSEQ1	IN	CHAR[5]	Znaki startowe sekwencji 1 (5 znaków).
STRSEQ2CTL	IN	BYTE	Sterowanie ignoruj/porównaj każdego znaku sekwencji 2.
STRSEQ2	IN	CHAR[5]	Znaki startowe sekwencji 2 (5 znaków).
STRSEQ3CTL	IN	BYTE	Sterowanie ignoruj/porównaj każdego znaku sekwencji 3.
STRSEQ3	IN	CHAR[5]	Znaki startowe sekwencji 3 (5 znaków).
STRSEQ4CTL	IN	BYTE	Sterowanie ignoruj/porównaj każdego znaku sekwencji 4.
STRSEQ4	IN	CHAR[5]	Znaki startowe sekwencji 4 (5 znaków).

Struktura danej typu CONDITIONS, część 2 (warunki startu)

Parametr	Typ parametru	Typ danych	Opis
ENDCOND	IN	UINT	Ten parametr określa warunki końca wiadomości: • 1 - Czas odpowiedzi • 16 - N + LEN + M • 2 - Czas wiadomości • 32 - Sekwencja • 4 - Przerwa międzyznakowa • 8 - Maksymalna długość
MAXLEN	IN	UINT	Maksymalna długość wiadomości. Używany tylko wtedy, kiedy wybrano, że maksymalna długość jest warunkiem końcowym.
N	IN	UINT	Położenie bajtu pola długości w wiadomości. Używany tylko wtedy, kiedy wybrano, że N + LEN + M jest warunkiem końcowym.
LENGTHSIZE	IN	UINT	Rozmiar pola bajtowego (1, 2 lub 4 bajty). Używany tylko wtedy, kiedy wybrano, że N + LEN + M jest warunkiem końcowym.
LENGTHM	IN	UINT	Określa liczbę znaków następujących po polu długości, które nie są wliczone do wartości pola długości. Używany tylko wtedy, kiedy wybrano, że N + LEN + M jest warunkiem końcowym.
RCVTIME	IN	UINT	Określa czas oczekiwania na odbiór pierwszego znaku. Jeżeli znak nie zostanie poprawnie odebrany w określonym czasie, to operacja odbioru zostanie zakończona z błędem. Jest używany tylko wtedy, kiedy warunkiem jest czas odpowiedzi. Ten parametr faktycznie nie jest badany jako warunek końcowy, ponieważ określa warunek startu. Trzeba zatem wybrać odrębny warunku końcowy.
MSGTIME	IN	UINT	Określa czas oczekiwania na odebranie całej wiadomości, od chwili odebrania pierwszego znaku. Jest używany tylko wtedy, kiedy warunkiem jest limit czasu wiadomości.
CHARGAP	IN	UINT	Określa liczbę czasów trwania bitu pomiędzy znakami. Jeżeli liczba czasów trwania bitu pomiędzy znakami przekracza ustaloną wartość, to warunek końcowy jest spełniony. Jest używany tylko wtedy, kiedy warunkiem jest przerwa międzyznakowa.

Parametr	Typ parametru	Typ danych	Opis
ENDSEQ1CTL	IN	BYTE	Sterowanie ignoruj/porównaj każdego znaku sekwencji 1: To są bity uaktywniające dla każdego znaku sekwencji startowej. Dla znaku 1 jest bit 0, dla znaku 2 jest bit 1, … , dla znaku 5 jest bit 4. Zablokowanie bitu związanego ze znakiem oznacza, że dowolny znak na tej pozycji jest identyfikowany jako właściwy.
ENDSEQ1	IN	CHAR[5]	Znaki startowe sekwencji 1 (5 znaków).

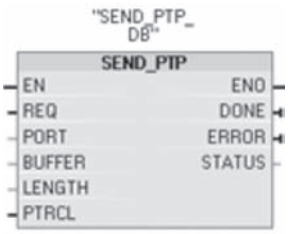
Kody warunkowe

STATUS (W#16#....)	Opis
80C0	Wybrany jest nieprawidłowy warunek startowy.
80C1	Wybrany jest nieprawidłowy warunek końcowy, nie jest wybrany warunek końcowy.
80C2	Odbiór uaktywnionego przerwania, co jest niedozwolone.
80C3	Warunek końcowy maksymalnej długości jest uaktywniony i maksymalna długość wynosi 0 lub jest większa od 1024.
80C4	Obliczona długość jest uaktywniona i jest N >= 1023.
80C5	Obliczona długość jest uaktywniona i długość nie jest równa 1, 2 lub 4.
80C6	Obliczona długość jest uaktywniona jest M > 255.
80C7	Obliczona długość jest uaktywniona i obliczona długość jest > 1024.
80C8	Czas odpowiedzi jest uaktywniony i czas odpowiedzi wynosi zero.
80C9	Limit czasu przerwy międzyznakowej jest uaktywniony i wynosi zero lub jest > 2500.
80CA	Limit czasu wiersza pustego jest uaktywniony i wynosi zero lub jest > 2500.
80CB	Sekwencja końcowa jest uaktywniona, ale wszystkie znaki mają status „bez znaczenia”.
80CC	Sekwencja startowa (dowolna z 4) jest uaktywniona, ale wszystkie znaki mają status „bez znaczenia”.

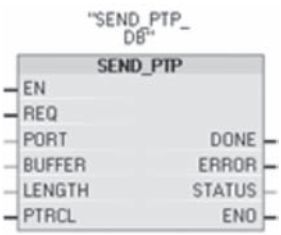
8.6.5 Instrukcja SEND_PTP

Instrukcja SEND_PTP (*Send Point-to-Point data*) inicjalizuje transmisję danych. SEND_PTP przesyła określony bufor do modułu RS232 lub RS485. Program CPU kontynuuje działanie podczas gdy moduł wysyła dane z określona szybkością.

LAD



FBD



Parametr	Typ parametru	Typ danych	Opis
REQ	IN	BOOL	Aktywuje żądaną transmisję w momencie wystąpienia narastającego zbocza na tym wejściu. To powoduje przesłanie zawartości bufora do modułu komunikacyjnego (CM) PtP.
PORT	IN	PORT	Identyfikator portu komunikacyjnego: Ten adres logiczny jest stałą, która znajduje się w zakładce „Constants" domyślnej tablicy tagów.
BUFFER	IN	VARIANT	Ten parametr wskazuje położenie początku bufora nadawczego. Zwracamy uwagę, że w przypadku protokołów dostarczonych przez firmę Siemens, to wejście jest używane do przesyłania do urządzenia PtP zarówno konfiguracji, jak i danych protokołu. Wybór jest dokonywany za pomocą parametru PTRCL.
LENGTH	IN	UINT	Długość nadawanej bloku.
PTRCL	IN	BOOL	Ten parametr określa czy bufor zaimplementowany w dołączonym CM pracuje z normalnymi protokołami point-to-point, czy też ze szczególnymi protokołami dostarczonymi przez firmę Siemens. FALSE = operacje point-to-point sterowane z programu użytkownika.
DONE	OUT	BOOL	Przyjmuje wartość TRUE na jeden cykl programu jeśli ostatnie żądanie zostało zakończone bez błędu.
ERROR	OUT	BOOL	Przyjmuje wartość TRUE na jeden cykl programu jeśli ostatnie żądanie zostało zakończone z błędem.
STATUS	OUT	WORD	Kod błędu.

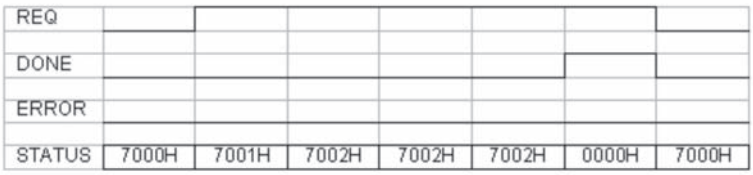
Opis działania

W trakcie wykonywania operacji nadawania, wyjścia DONE i ERROR mają wartość FALSE. Po ukończeniu nadawania DONE albo ERROR przyjmuje wartość TRUE (na czas jednego cyklu programu) pokazując status operacji nadawania. Jeżeli DONE lub ERROR ma wartość TRUE, to wyjście STATUS zawiera aktualną i poprawną informację.

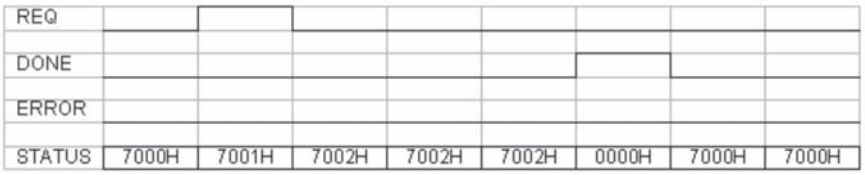
Jeżeli moduł komunikacyjny (CM) akceptuje przesyłane dane, to instrukcja zwraca status wyoszący16#7001. Jeżeli CM jest nadal zajęty nadawaniem, to kolejne wykonanie SEND_PTP zwraca 16#7002. Kiedy operacja nadawania jest zakończona, wtedy – o ile nie wystąpiły żadne błędy – CM zwraca status operacji nadawania 16#0000. Kolejne wykonanie SEND_PTP, gdy REQ jest niski, zwraca status 16#7002 (nie zajęty).

Zależność wartości wyjściowych z REQ:

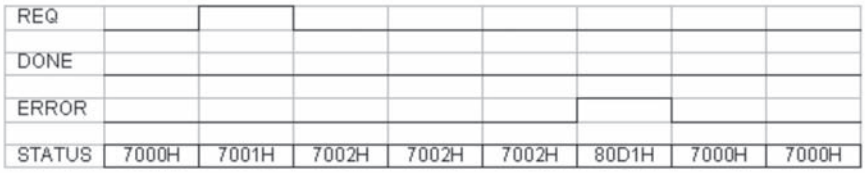
Zakładamy, że instrukcje są wywoływane okresowo w celu sprawdzenia statusu procesu nadawania. Na poniższym diagramie założono, że instrukcja jest wywoływana raz na jeden cykl programu (reprezentowane wartościami STATUS).



Na diagramie zamieszczonym poniżej pokazano w jaki sposób parametry DONE i STATUS zachowują ważność tylko przez jeden cykl programu, jeżeli na linii REQ występuje impuls (w jednym cyklu programu) inicjujący operację nadawania.



Na poniższym diagramie przedstawiono związki między parametrami DONE, ERROR i STATUS w przypadku wystąpienia błędu.



Kody warunkowe

STATUS	
(W#16#....)	Opis
80D0	Nowe żądanie w trakcie aktywnej transmisji.
80D1	Transmisja przerwana ze względu na brak CTS w ustalonym czasie oczekiwania.
80D2	Transmisja przerwana ze względu na brak DSR z urządzenia DCE.
80D3	Transmisja przerwana ze względu na przepełnienie kolejki.
7000	Nie zajęty
7001	Zajęty w czasie przyjmowania żądania (pierwsze wywołanie).
7002	Zajęty podczas odpytywania (n-te wywołanie).

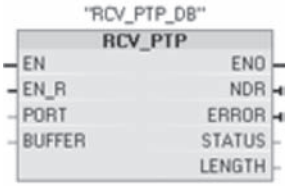
8.6.6 Instrukcja RCV_PTP

Instrukcja RCV_PTP (*Receive Point-to-Point*) sprawdza czy moduł komunikacyjny (CM) Point-to-Point odebrał jakieś wiadomości. Jeżeli wiadomość jest dostępna, to zostanie przesłana z modułu do CPU. Jeżeli wystąpił błąd, to zostanie zwrócona odpowiednia wartość STATUS.

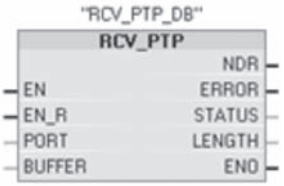
STATUS zawiera ważną wartość wtedy, kiedy albo NDR albo ERROR ma wartość TRUE. Wartość STATUS określa przyczynę zakończenia operacji odbierania przez CM. Zwykle jest to wartość dodatnia sygnalizująca, że operacja odbierania odbyła się pomyślnie i proces odbierania danych zakończył się normalnie. Jeżeli wartość STATUS jest ujemna (ustawiony jest najbardziej znaczący bit wartości heksadecymalnej), to oznacza, że operacja odbierania zakończyła się z błędem, takim jak błąd parzystości, ramkowania lub przepełnienia.

Każdy moduł CM *Point-to-Point* może buforować maksymalnie 1 KB (kilo bajtów). Ten bufor może być zorganizowany tak, by przechowywać wiele odebranych wiadomości.

LAD



FBD



Parametr	Typ parametru	Typ danych	Opis
EN_R	IN	BOOL	Kiedy stan tego wejścia jest TRUE, wtedy sprawdza czy CM odebrał wiadomość. Jeżeli wiadomość została pomyślnie odebrana, to zostanie przesłana z modułu do CPU.
PORT	IN	PORT	Identyfikator portu komunikacyjnego: Ten adres logiczny jest stałą, która znajduje się w zakładce „Constants” domyślnej tablicy tagów.
BUFFER	IN	VARIANT	Ten parametr wskazuje położenie początku bufora odbiorczego. Długość bufora jest z góry znana, jako że jest określona przez typ danych VARIANT.
NDR	OUT	BOOL	Przyjmuje wartość TRUE na jeden cykl programu jeżeli nowe dane są gotowe i operacja zakończyła się bez błędu.
ERROR	OUT	BOOL	Przyjmuje wartość TRUE na jeden cykl programu jeśli ostatnie żądanie zostało zakończone z błędem.
STATUS	OUT	WORD	Kod błędu.
LENGTH	OUT	UINT	Długość zwróconej wiadomości.

Wyjście NDR (*New Data Ready*) sygnalizuje, że żądana akcja zakończyła się bez błędu i nowe dane są odebrane. To wyjście jest ustawiane na jeden cykl programu.

Kody warunkowe

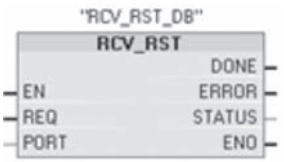
STATUS (W#16#....)	Opis
0000	Nie ma bufora.
80E0	Wiadomość przerwana ze względu na przepełnienie bufora odbiorczego.
80E1	Wiadomość przerwana ze względu na błąd parzystości.
80E2	Wiadomość przerwana ze względu na błąd ramkowania.
80E3	Wiadomość przerwana ze względu na błąd przepełnienia.
80E4	Wiadomość przerwana ze względu na to, że obliczona długość przewyższa rozmiar bufora.
0094	Wiadomość przerwana ze względu na to, że odebrano maksymalną liczbę znaków.
0095	Wiadomość przerwana ze względu na przekroczenie limitu czasu wiadomości.
0096	Wiadomość przerwana ze względu na przekroczenie limitu czasu międzyznakowego.
0097	Wiadomość przerwana ze względu na przekroczenie limitu czasu odpowiedzi.
0098	Wiadomość przerwana ze względu na spełnienie warunku długości „N+LEN+M”.
0099	Wiadomość przerwana ze względu na spełnienie warunku sekwencji końcowej.

8.6.7 Instrukcja RCV_RST

Instrukcja RCV_RST (*Receiver Reset*) kasuje bufor odbiorczy.

LAD

FBD

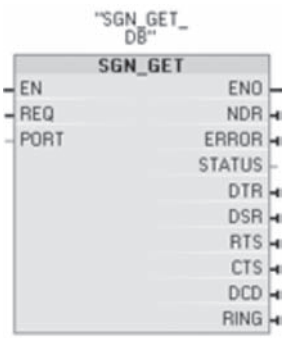


Parametr	Ty parametru	Typ danych	Opis
REQ	IN	BOOL	Aktywuje kasowanie odbiornika w momencie wystąpienia narastającego zbocza na tym wejściu.
PORT	IN	PORT	Identyfikator portu komunikacyjnego: Port musi być określony za pomocą adresu logicznego modułu.
DONE	OUT	BOOL	Przyjmuje wartość TRUE na jeden cykl programu jeśli ostatnie żądanie zostało zakończone bez błędu.
ERROR	OUT	BOOL	Przyjmuje wartość TRUE na jeden cykl programu, jeśli ostatnie żądanie zostało zakończone z błędem. Kiedy to wyjście ma wartość TRUE, wtedy wyjście STATUS zawiera właściwy kod błędu.
STATUS	OUT	WORD	Kod błędu.

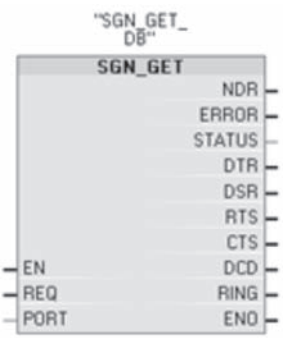
8.6.8 Instrukcja SGN_GET

Instrukcja SGN_GET (*Get RS232 Signals*) odczytuje bieżący stan sygnałów komunikacyjnych RS232. Ta funkcja dotyczy tylko modułu komunikacyjnego (CM) RS232.

LAD



FBD



Parametr	Typ parametru	Typ danych	Opis
REQ	IN	BOOL	Pobiera wartości stanu sygnałów RS232 w momencie wystąpienia narastającego zbocza na tym wejściu.
PORT	IN	PORT	Identyfikator portu komunikacyjnego: Ten adres logiczny jest stałą, która znajduje się w zakładce „Constants” domyślnej tablicy tagów.
NDR	OUT	BOOL	Przyjmuje wartość TRUE na jeden cykl programu jeżeli nowe dane są gotowe i operacja zakończyła się bez błędu.
ERROR	OUT	BOOL	Przyjmuje wartość TRUE na jeden cykl programu, jeśli operacja została zakończona z błędem.
STATUS	OUT	WORD	Kod błędu.
DTR	OUT	BOOL	Data Terminal Ready, moduł gotowy.
DSR	OUT	BOOL	Data Set Ready, partner komunikacyjny.
RTS	OUT	BOOL	Request to Send, moduł gotowy do wysłania danych.
CTS	OUT	BOOL	Clear to Send, partner komunikacyjny gotowy może odbierać dane.
DCD	OUT	BOOL	Data Carrier Detect, odebrany sygnał nośnej.
RING	OUT	BOOL	Sygnalizacja dzwonienia, sygnalizacja przychodzącego wywołania.

Kody warunkowe

STATUS (W#16#....)	Opis
80F0	CM jest modułem RS485 i żadne sygnały nie są dostępne.
80F1	Sygnały nie mogą zastać ustawione ze względu na sprzętowe sterowanie przepływem.
80F2	Nie można ustawić DSR bo moduł jest urządzeniem DTE.
80F3	Nie można ustawić DTR bo moduł jest urządzeniem DCE.

8.6.9 Instrukcja SGN_SET

Instrukcja SGN_SET (*Set RS232 Signals*) ustawia stan sygnałów komunikacyjnych RS232. Ta funkcja dotyczy tylko modułu komunikacyjnego (CM) RS232.

LAD

FBD

Parametr	Typ parametru	Typ danych	Opis
REQ	IN	BOOL	Rozpoczyna ustawianie sygnałów RS232 w momencie wystąpienia narastającego zbocza na tym wejściu.
PORT	IN	PORT	Identyfikator portu komunikacyjnego: Ten adres logiczny jest stałą, która znajduje się w zakładce „Constants” domyślnej tablicy tagów.
SIGNAL	IN	BYTE	Wybiera sygnał, który zostanie ustawiony: • 01H = Ustawia RTS • 02H = Ustawia DTR • 04H = Ustawia DSR
RTS	OUT	BOOL	Request to Send, moduł gotowy do wysłania danych.
DTR	IB	BOOL	Data terminal Ready, moduł gotowy.
DSR	OUT	BOOL	Data Set Ready, partner komunikacyjny gotowy (dotyczy tylko interfejsów typu DCE).
DONE	OUT	BOOL	Przyjmuje wartość TRUE na jeden cykl programu jeśli ostatnie żądanie zostało zakończone bez błędu.
ERROR	OUT	BOOL	Przyjmuje wartość TRUE na jeden cykl programu, jeśli operacja została zakończona z błędem.
STATUS	OUT	WORD	Kod błędu.

Kody warunkowe

STATUS (W#16#....)	Opis
80F0	CM jest modułem RS485 i żadnych sygnałów nie można ustawić.
80F1	Sygnały nie mogą zastać ustawione ze względu na sprzętowe sterowanie przepływem.
80F2	Nie można ustawić DSR bo moduł jest urządzeniem DTE.
80F3	Nie można ustawić DTR bo moduł jest urządzeniem DCE.

8.7 Błędy

Wartości zwracane przez instrukcje PtP

Każda instrukcja PtP ma trzy wyjścia, które dostarczają kompletnej informacji o statusie:

Parametr	Typ danej	Wartość domyślna	Opis
DONE	BOOL	FALSE	Przyjmuje wartość TRUE na jeden cykl programu jeśli ostatnie żądanie zostało zakończone bez błędu.
ERROR	BOOL	FALSE	Wartość TRUE oznacza, że ostatnie żądanie zostało zakończone z błędem i STATUS zawiera odpowiedni kod błędu.
STATUS	WORD	0	Dwa bajty zawierające klasę błędu i numer błędu (jeśli wystąpił). STATUS zachowuje swoją wartość przez czas trwania wykonywania funkcji.

Wyjście STATUS

Zwracane są błędy zarówno ogólne, jak i dotyczące komunikacji PtP. Reprezentacja bitowa dla błędów ogólnych jest następująca:

15							**8**	**7**			**4**	**3**			**0**
1	Numer parametru							0	Numer zdarzenia						

Reprezentacja bitowa dla błędów związanych z PtP jest następująca:

15							**8**	**7**			**4**	**3**			**0**
1	0							1	Klasa błędu			Numer błędu			

Ogólne klasy błędów i błędy

Opis klasy	Klasy błędu	Opis
Konfiguracja portu	80Ax	Wykorzystywana do zdefiniowania ogólnych błędów konfiguracji portu.
Konfiguracja nadawania	80Bx	Wykorzystywana do zdefiniowania ogólnych błędów konfiguracji nadawania.
Konfiguracja odbioru	80Cx	Wykorzystywana do zdefiniowania ogólnych błędów konfiguracji odbioru.
Czas trwania nadawania	80Dx	Wykorzystywana do zdefiniowania ogólnych błędów podczas trwania nadawania.
Czas trwania odbierania	80Ex	Wykorzystywana do zdefiniowania ogólnych błędów podczas trwania odbioru.
Przetwarzanie sygnałów	80Fx	Wykorzystywana do zdefiniowania ogólnych błędów związanych z obsługą wszystkich sygnałów.

Błędy konfiguracji portu

Identyfikator zdarzenia/błędu	Opis
0x80A0	Określony protokół nie istnieje.
0x80A1	Określona prędkość transmisji nie istnieje.
0x80A2	Określona opcja parzystości nie istnieje.
0x80A3	Określona liczba bitów danych nie istnieje.
0x80A4	Określona liczba bitów stopu nie istnieje.
0x80A5	Określony typ sterowania przepływem nie istnieje.

Błędy konfiguracji nadawania

Identyfikator zdarzenia/błędu	Opis
0x80B0	Określony protokół nie istnieje.
0x80B1	Określona prędkość transmisji nie istnieje.
0x80B2	Określona opcja parzystości nie istnieje.
0x80B3	Określona liczba bitów danych nie istnieje.
0x80B4	Określona liczba bitów stopu nie istnieje.
0x80B5	Określony typ sterowania przepływem nie istnieje.

Błędy konfiguracji odbioru

Identyfikator zdarzenia/błędu	Opis
0x80C0	Błąd warunku startowego.
0x80C1	Błąd warunku końcowego.
0x80C3	Błąd maksymalnej długości.
0x80C4	Błąd wartości N (por. N+LEN+M).
0x80C5	Błąd rozmiaru długości (por. MAXLEN lub N+LEN+M).
0x80C6	Błąd wartości M (por. N+LEN+M).
0x80C7	Błąd wartości N-długość-M (por. to N+LEN+M).
0x80C8	Błąd limitu czasu odpowiedzi, żadna wiadomość nie została odebrana w wyspecyfikowanym czasie odbioru (por. RCVTIME lub MSGTIME).
0x80C9	Błąd limitu czasu przerwy międzyznakowej (por. CHARGAP).
0x80CA	Błąd limitu czasu wiersza pustego (por. wiersz pusty, Idle Line)
0x80CB	W wyspecyfikowanej sekwencji końcowej wszystkie znaki mają status „bez znaczenia”.
0x80CC	W wyspecyfikowanej sekwencji startowej wszystkie znaki mają status „bez znaczenia”.

Błędy związane z sygnałami

Identyfikator zdarzenia/błędu	Opis
0x80F0	Modułem komunikacyjnym jest moduł RS485 i żadne sygnały nie są dostępne.
0x80F1	Modułem komunikacyjnym jest moduł RS232, ale żadnych sygnałów nie można ustawić ponieważ jest aktywne sprzętowe sterowanie przepływem.
0x80F2	Nie można ustawić sygnału DSR bo moduł jest urządzeniem DTE.

Błędy podczas trwania transmisji

Identyfikator zdarzenia/błędu	Opis
Buffer Limit	Cały dostępny bufor nadawczy CP jest przekroczony.
0x80D0	Otrzymano nowe żądanie podczas gdy nadajnik był aktywny.
0x80D1	W celu zawieszenia aktywnej transmisji odbiornik wystawił żądanie sterowania przepływem i nigdy ponownie nie uaktywnił transmisji w ustalonym czasie oczekiwania. Ten błąd jest również generowany podczas sprzętowego sterowania przepływem wtedy, kiedy odbiornik nie ustawił sygnału CTS w ustalonym czasie oczekiwania.
0x80D2	Żądanie transmisji zostało anulowane ponieważ z DCE nie nadszedł sygnał DSR.
0x80D3	Cały dostępny bufor nadawczy CP jest przekroczony.
0x7000	Funkcja nadawania nie jest zajęta.
0x7001	Funkcja nadawania jest zajęta – przy pierwszym wywołaniu.
0x7002	Funkcja nadawania jest zajęta – przy kolejnych wywołaniach (cykl odpytywania po pierwszym wywołaniu).

Wartości zwracane podczas trwania odbioru

Identyfikator zdarzenia/błędu	Opis
0x80E0	Wiadomość przerwana ze względu na przepełnienie bufora odbiorczego.
0x80E1	Wiadomość przerwana ze względu na błąd parzystości.
0x80E2	Wiadomość przerwana ze względu na błąd ramkowania.
0x80E3	Wiadomość przerwana ze względu na błąd przepełnienia.
0x80E4	Wiadomość przerwana ze względu na to, że wyspecyfikowana długość przewyższa rozmiar bufora.
0x0094	Wiadomość przerwana ze względu na to, że odebrano maksymalną liczbę znaków (MAXLEN).
0x0095	Wiadomość przerwana ze względu na to, ze nie została odebrana w całości w wyspecyfikowanym czasie (MSGTIME).
0x0096	Wiadomość przerwana ze względu na to, ze kolejny znak nie został odebrany w czasowym limicie międzyznakowym (CHARGAP).
0x0097	Wiadomość przerwana ze względu na to, że pierwszy znak nie został odebrany w wyspecyfikowanym czasie (RCVTIME).
0x0098	Wiadomość przerwana ponieważ został spełniony warunek długości „N+LEN+M” (N+LEN+M).
0x0099	Wiadomość przerwana ze względu na spełnienie warunku sekwencji końcowej (ENDSEQ).

Inne błędy

Identyfikator zdarzenia/błędu	Opis
0x8n3A	Nieprawidłowe wskazanie parametru n.
0x8070	Cała pamięć wewnętrzna jest w użyciu.
0x8080	Nieprawidłowy numer portu.
0x8082	Parametryzacja nie powiodła się, ponieważ parametryzacja jest aktualnie wykonywana w tle
0x8083	Przepełnienie bufora: CM zwrócił więcej danych niż jest to dozwolone.

Narzędzia online i diagnostyczne 9

9.1 Diody LED statusu

W CPU i modułach I/O są zainstalowane diody LED pełniące funkcje wskaźników statusu operacyjnego modułu albo I/O.

W CPU przedstawiane są następujące informacje o statusie:

- STOP/RUN
  - Stałe światło pomarańczowe oznacza tryb STOP.
  - Stałe światło zielone oznacza tryb RUN.
  - Migające światło (na przemian zielone i pomarańczowe) wskazuje stan inicjalizowania CPU.
- ERROR
  - Migające światło czerwone sygnalizuje błąd, taki jak wewnętrzny błąd CPU, błąd karty pamięci lub błąd konfiguracji (niewłaściwie dobrane moduły).
  - Stałe światło czerwone wskazuje uszkodzenie sprzętu.
- MAINT (*Maintenance* – obsługa)
  - Migające światło pomarańczowe w czasie gdy CPU jest trybie STOP sygnalizuje konieczność wyjęcia karty pamięci.
  - Stałe światło pomarańczowe w trybie STOP wskazuje na błąd, taki jak włożenie karty pamięci w czasie gdy CPU był w trybie RUN, włożenie niesformatowanej karty pamięci lub przejście modułu w tryb *offline*.

Opis	**STOP / RUN** Pomarańczowe / Zielone	**ERROR** Czerwone	**MAINT** Pomarańczowe
Zasilanie jest wyłączone	Wyłączone	Wyłączone	Wyłączone
Rozruch, autotest, uaktualnianie oprogramowania firmware	Migające (na przemian pomarańczowe i zielone)	–	Wyłączone
Tryb STOP	Włączone (pomarańczowe)	–	–
TRYB RUN	Włączone (pomarańczowe)	–	–
Wyjmij kartę pamięci	Włączone (pomarańczowe)	–	Migające
Błąd	Włączone (pomarańczowe lub zielone)	Migające	–
Wymagana obsługa	Włączone (pomarańczowe lub zielone)	–	Włączone
Uszkodzenie sprzętu	Włączone (pomarańczowe)	Włączone	Wyłączone
Test diod LED lub uszkodzenie oprogramowania firmware CPU	Migające (na przemian pomarańczowe i zielone)	Migające	Migające

CPU jest również wyposażona w dwie diody LED wskazujące status komunikacji PROFINET. W celu uzyskania widoku diod PROFINET LED należy otworzyć pokrywę dolnej listwy zaciskowej.

- Świecenie diody Link (na zielono) sygnalizuje pomyślne nawiązanie połączenia.
- Świecenie diody Rx/Tx (na żółto) sygnalizuje aktywną transmisję.

CPU i każdy cyfrowy moduł sygnałowy (SM) ma przypisaną każdemu cyfrowemu wejściu i wyjściu diodę I/O Chanel LED. Diody I/O Chanel LED (zielone) są włączane i wyłączane wskazując stan indywidualnego wejścia lub wyjścia.

Dodatkowo, każdy cyfrowy SM jest wyposażony w diodę DIAG LED wskazującą status modułu:

- Kolor zielony sygnalizuje, pełną sprawność modułu.
- Kolor czerwony sygnalizuje, że moduł jest uszkodzony lub niegotowy do pracy.

Każdy analogowy SM jest wyposażony w diodę I/O Chanel LED przyporządkowaną każdemu analogowemu wejściu i wyjściu.

- Kolor zielony sygnalizuje, że dany kanał został skonfigurowany i jest aktywny.
- Kolor czerwony sygnalizuje błąd tego indywidualnego wejścia lub wyjścia.

Dodatkowo, każdy analogowy SM jest wyposażony w diodę DIAG LED wskazującą status modułu:

- Kolor zielony sygnalizuje, pełną sprawność modułu.
- Kolor czerwony sygnalizuje, że moduł jest uszkodzony lub niegotowy do pracy.

SM wykrywa obecność lub brak zasilania modułu (zasilania zewnętrznego, w razie potrzeby).

Opis	DIAG (Czerwone / Zielone)	I/O Channel (Czerwone / Zielone)
Zasilanie zewnętrzne jest wyłączone	Migające czerwone	Migające czerwone
Moduł nie jest skonfigurowany lub uaktualniony	Migające zielone	Wyłączone
Moduł skonfigurowany bez błędów	Włączone (zielone)	Włączone (zielone)
Błąd	Migające czerwone	–
Błąd I/O (diagnostyka aktywna)	–	Migające czerwone
Błąd I/O (diagnostyka nieaktywna)	–	Włączone (zielone)

9.2 Podłączanie online i podłączenie do CPU

Połączenie *online* urządzenia programującego z systemem docelowym jest niezbędne dla ładowania programów i danych projektów inżynierskich do systemu docelowego, jak również dla wykonywania takich zadań, jak:

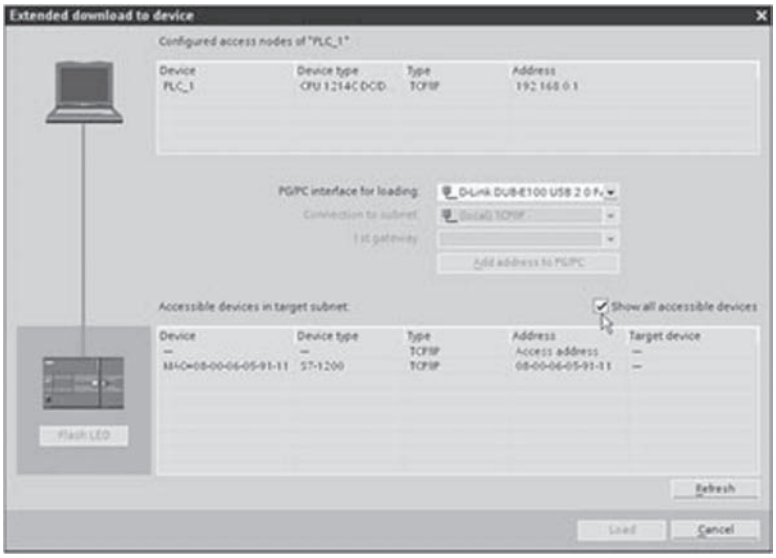
- Testowanie programów użytkownika.
- Wyświetlanie i zmiana trybu pracy CPU.
- Wyświetlanie i ustawianie daty i godziny w CPU.
- Wyświetlanie informacji o module.
- Porównywanie bloków *online* i *offline*.
- Diagnozowanie sprzętu.

Można wówczas w widoku *online* lub diagnostycznym uzyskać dostęp do danych systemu docelowego za pomocą karty „Online tools”.

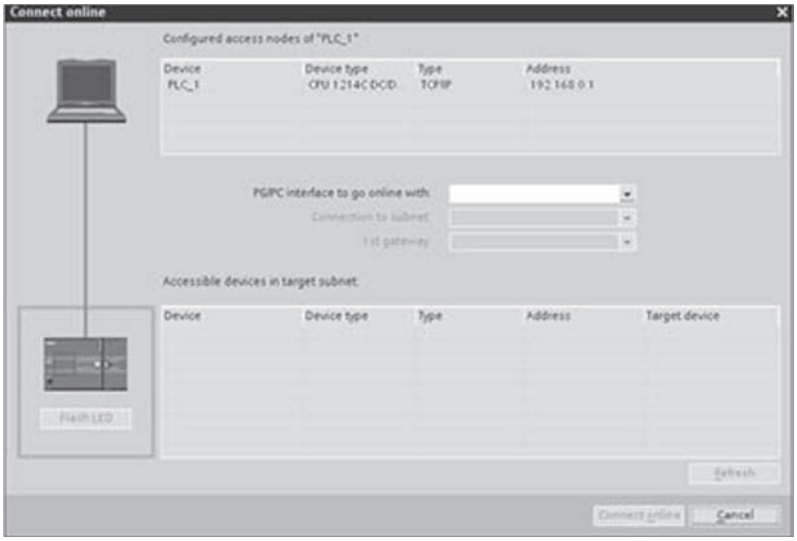
Bieżący status *online* urządzenia sygnalizuje ikona z prawej strony naprzeciw urządzenia w oknie nawigacyjnym projektu.

Kolor pomarańczowy wskazuje na połączenie *online*.

W celu znalezienia w sieci CPU należy wybrać „Accessible Nodes”.



Podłączenie do CPU będącej w sieci następuje po kliknięciu „Go online”.



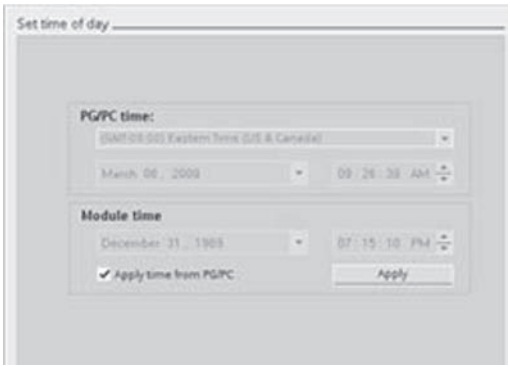
9.3 Ustawianie adresu IP oraz czasu

Użytkownik może ustawiać adres IP oraz zegar CPU znajdującej się w sieci.

Po wykonaniu za pomocą „Online & diagnostics" połączenia z CPU znajdującą się *online* można wyświetlić lub zmienić adres IP.

Więcej informacji na ten temat znajduje się w części dotyczącej adresów IP.

W przypadku CPU znajdującej się *online* można również wyświetlić lub ustawić czas i datę.

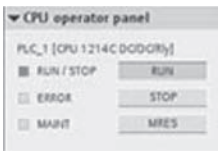


9.4 Panel operatorski wbudowany w CPU podczas pracy w trybie online

Dla zmiany trybu pracy CPU znajdującej się online można wykorzystywać panel operatorski CPU.

Karta „CPU operator panel" wyświetla tryb pracy (STOP lub RUN) CPU znajdującej się *online*: Panel pozwala również zaobserwować czy w CPU wystąpiły błędy lub czy wartości są wymuszane.

Panel stosuje się do zmiany trybu pracy CPU.

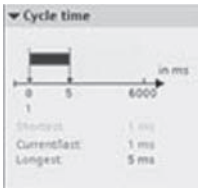


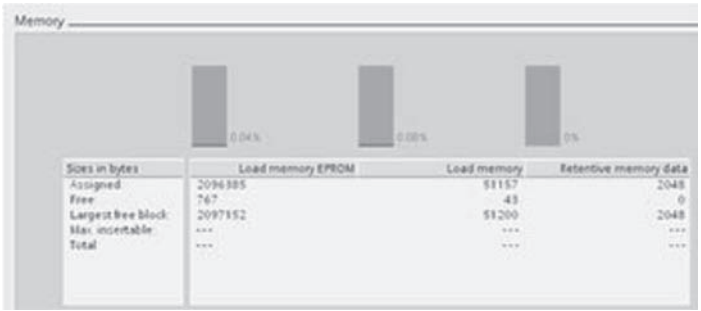
9.5 Monitorowanie czasu cyklu i użycia pamięci

Użytkownik może monitorować czas cyklu oraz użycie pamięci CPU znajdującej się w sieci.

Po wykonaniu połączenia z CPU znajdującą się *online* można zaobserwować wyniki następujących pomiarów:

- Czasu cyklu
- Użycia pamięci





9.6 Wyświetlanie zdarzeń diagnostycznych w CPU

Dla dokonania przeglądu ostatnio prowadzonej aktywności CPU stosuje się bufor diagnostyczny.

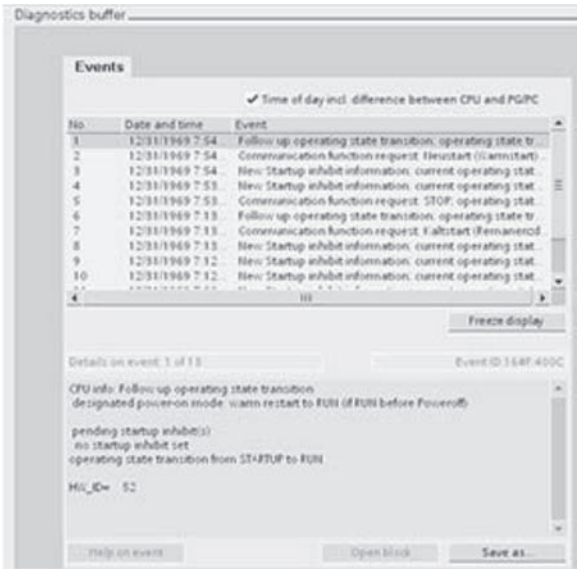
Bufor diagnostyczny zawiera następujące pozycje:

- Zdarzenia diagnostyczne
- Zmiany trybu pracy CPU (przejście do trybu STOP lub RUN)

Na pierwszej pozycji znajduje się ostatnio zarejestrowane zdarzenie. Każda pozycja w buforze diagnostycznym zawiera datę i czas rejestracji zdarzenia oraz jego opis.

Maksymalna liczba pozycji zależy od CPU. Obsługiwanych jest co najwyżej 50 pozycji.

Tylko 10 ostatnich zdarzeń w buforze diagnostycznych zapamiętanych jest na stałe. Kasowanie CPU do ustawień fabrycznych kasuje bufor diagnostyczny poprzez usunięcie zarejestrowanych zdarzeń.



9.7 Tablice monitorujące do monitorowania programu użytkownika

Tablica monitorująca (*Watch table*) pozwala monitorować i sterować wybrane zmienne (tagi) podczas wykonywania przez CPU programu użytkownika. Zmienne danych mogą być obrazem procesu (I lub Q), punktami fizycznymi (I_:P lub Q_:P), M lub danymi z bloków DB w zależności od funkcji monitorującej lub sterującej.

Obserwacja zmiennych (tagów) nie zmienia sekwencji programu. Prezentuje użytkownikowi informacje o sekwencji programu i danych programu w CPU.

Funkcje sterujące pozwalają użytkownikowi sterować sekwencją danych programu. Podczas stosowania tych funkcji należy zachować ostrożność. Trzema funkcjami sterującymi są *Modify*, *Force* oraz *Enable Outputs in STOP*.

Za pomocą tablicy monitorującej można wykonywać następujące funkcje *online*:

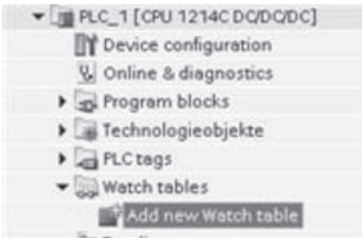
- Monitorowanie statusu *tagów*
- Modyfikowanie wartości indywidualnych *tagów*
- Wymuszone nadawanie *tagom* określonych wartości

Użytkownik określa kiedy monitorować lub modyfikować *tag*:

- *Beginning of scan cycle*: Odczyt lub zapis wartości na początku cyklu programu
- *End of scan cycle*: Odczyt lub zapis wartości na końcu cyklu programu
- *Switch to stop*: Przełączanie w tryb STOP

W celu utworzenia tablicy monitorującej należy:

1. Podwójnie kliknąć „Add new watch table" w celu otwarcia nowej tablicy monitorującej.
2. Wprowadzić nazwę *tagu* w celu dodania *tagu* do tablicy monitorującej.



Dostępne są następujące opcje monitorowania *tagów*:

- *Monitor all*: Ten rozkaz uruchamia monitorowanie widocznych *tagów* w aktywnej tablicy monitorującej.
- *Monitor now*: Ten rozkaz uruchamia monitorowanie widocznych *tagów* w aktywnej tablicy monitorującej. Tablica monitorująca monitoruje *tagi* natychmiast i tylko jednokrotnie.

Dostępne są następujące opcje modyfikowania *tagów*:

- „Modify to 0" ustawia wartość wybranego adresu na „0".
- „Modify to 1" ustawia wartość wybranego adresu na „1".
- „Modify now" natychmiast zmienia wartości dla wybranych adresów na jeden cykl programu.
- „Modify with trigger" zmienia wartości dla wybranych adresów.

W tej funkcji nie występuje sprzężenie zwrotne wskazujące, że wybrane adresy faktycznie zostały zmodyfikowane. Jeżeli konieczne jest potwierdzenie zmian, to należy stosować funkcję „Modify now".

- „Enable peripheral outputs" blokuje rozkaz dezaktywacji wyjść i jest dostępna tylko wtedy, kiedy CPU jest w trybie STOP.

W celu monitorowania *tagów*, CPU musi być podłączona w trybie *online*.

Różne funkcje można wybierać za pomocą przycisków znajdujących się na górze tablicy monitorującej.

W celu monitorowania *tagu* należy wprowadzić jego nazwę oraz wybrać format wyświetlania z rozwijanej listy. W trybie połączenia *online* z CPU, kliknięcie „Monitor" spowoduje wyświetlenie faktycznych wartości punktów danych w polu „Monitor value".

Użycie wyzwalania podczas monitorowania lub modyfikowania tagów PLC

Wyzwalanie określa w którym punkcie cyklu programu wybrany adres będzie monitorowany lub modyfikowany.

Typ wyzwalania	Opis
Permanent	Dane są zbierane w sposób ciągły.
At scan cycle start	Permanent: Dane są zbierane w sposób ciągły na początku cyklu programu, po odczytaniu przez CPU wejść.
	Once: Dane są zbierane na początku cyklu programu, po odczytaniu przez CPU wejść.
At scan cycle end	Permanent: Dane są zbierane w sposób ciągły na końcu cyklu programu, zanim CPU ustawi wyjścia. Once: Dane są zbierane na końcu cyklu programu, zanim CPU ustawi wyjścia.
At transition to STOP	Permanent: Dane są zbierane w sposób ciągły w chwili przejścia CPU do trybu STOP.
	Once: Dane są zbierane w chwili przejścia CPU do trybu STOP.

W celu modyfikacji *tagu* PLC przy określonym wyzwalaniu należy wybrać albo początek, albo koniec cyklu.

- Modyfikacja wyjścia: Najlepszym zdarzeniem wyzwalającym modyfikację wyjścia jest koniec cyklu programu, bezpośrednio przed tym jak CPU ustawia wyjścia.

W celu określenia jaka wartość jest zapisywana do wyjścia fizycznego należy monitorować wartość wyjść na początku cyklu programu. Wyjścia należy również monitorować zanim CPU wpisze wartość do wyjść fizycznych, co pozwoli sprawdzić logikę programu i porównać ją z rzeczywistym zachowaniem I/O.

- Modyfikacja wejścia: Najlepszym zdarzeniem wyzwalającym modyfikację wejścia jest początek cyklu programu, bezpośrednio po tym jak CPU odczyta wejścia i przed tym, jak program użytkownika wykorzystuje wartości wejściowe.

  Jeżeli wejścia są modyfikowane na początku cyklu programu, to należy również monitorować wartość wejść na końcu cyklu programu, by upewnić się, że wartość wejść na końcu cyklu programu nie zmieniła się od początku cyklu programu. Jeśli wystąpi różnica wartości, to być może program użytkownika ustawia wejścia w wyniku błędu.

W celu zdiagnozowania z jakiego powodu CPU mógł przejść w tryb STOP, należy wybrać zdarzenie wyzwalające „Transition to STOP" dla zarejestrowania ostatnich wartości procesu.

Uaktywnianie wyjść w trybie STOP

Tablica monitorująca pozwala ustawić wyjścia wtedy, kiedy CPU jest w trybie STOP. Ta cecha umożliwia sprawdzenie okablowania wyjść i weryfikację jakości i poprawnego funkcjonowania połączenia.

⚠	**OSTRZEŻENIE**
Nawet jeśli CPU jest w trybie STOP, to uaktywnienie wyjścia fizycznego może uruchomić urządzenie, które jest podłączone do tego wyjścia.	

Jeżeli wyjścia są uaktywnione, to można w trybie STOP zmienić stan wyjść. Jeżeli wyjścia są dezaktywowane, to nie można zmienić stanu wyjść w trybie STOP.

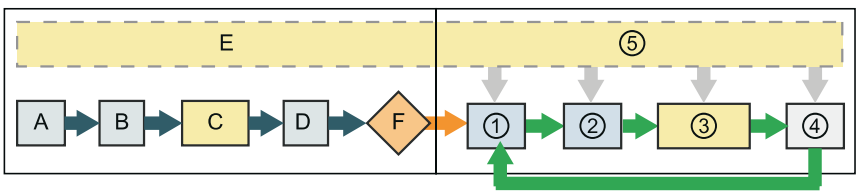
- W celu uaktywnienia modyfikacji wyjść w trybie STOP, należy wybrać opcję „Enable peripheral outputs” rozkazu „Modify” w menu „Online” lub kliknąć prawym klawiszem myszy wiersz tablicy monitorującej.
- Ustawienie CPU w tryb RUN blokuje opcje „Enable peripheral outputs”.
- Jeżeli stan jakichkolwiek wejść lub wyjść jest wymuszony, to CPU nie może uaktywnić wyjść będąc w trybie STOP. Najpierw musi zostać skasowana funkcja wymuszania.

Wymuszanie wartości w CPU

CPU umożliwia wymuszanie wartości na wejściach i wyjściach poprzez wyspecyfikowanie adresu wejścia lub wyjścia w tabeli monitorującej i zastosowanie wymuszenia. Wymuszenie jest stosowane do obszaru wejściowego obrazu procesu przed wykonaniem programu użytkownika i do obszaru wyjściowego obrazu procesu przed ustawieniem stanu wyjść modułów.

W programie, wartości odczytane z wejść fizycznych są zastępowane wartościami wymuszonym. Program posługuje się w czasie pracy wartościami wymuszonymi. Kiedy program zapisuje stan wyjścia fizycznego, wtedy wartość wyjściowa jest zastępowana wartością wymuszoną. Na wyjściu fizycznym pojawia się wartość wymuszona i proces używa tej wartości.

Kiedy wejście lub wyjście jest wymuszone w tablicy monitorującej, wtedy działanie wymuszające staje się częścią programu użytkownika. Nawet jeśli program zostanie zamknięty, to wybór wymuszeń pozostaje aktywny w działającym oprogramowaniu CPU dopóty, dopóki nie nastąpi ich wykasowanie poprzez przejście z oprogramowaniem sterującym w tryb *online* i zakończenie funkcji wymuszania. Programy z aktywnymi wymuszeniami załadowane do innego CPU z karty pamięci nadal będą wymuszać stany zmiennych w programie.



***Startup* (rozruch)**	**RUN**
A. Funkcja wymuszania (*Force*) nie ma wpływu na kasowanie obszaru I pamięci.	1. Podczas zapisywania obszaru Q pamięci do fizycznych wyjść, CPU uaktualniania stan wyjść wartościami wymuszonymi.
B. Funkcja wymuszania nie ma wpływu na inicjalizacje wartości wyjściowych.	2. Po skopiowaniu stanu wejść fizycznych do pamięci I, CPU stosuje wartości wymuszone.
C. Podczas wykonywania rozruchowych OB, CPU stosuje wartości wymuszone jeśli program użytkownika uzyskuje dostęp do fizycznych wejść.	3. Podczas wykonywania programu użytkownika (cyklicznych OB), CPU stosuje wartości wymuszone jeśli program użytkownika uzyskuje dostęp do fizycznych wejść.
D. Po skopiowaniu stanu wejść fizycznych do pamięci I, CPU stosuje wartości wymuszone.	4. Obsługa żądań komunikacyjnych i autodiagnostyka nie są zakłócone.
E. Przechowywanie przerwań w kolejce nie jest zakłócone.	5. Obsługa przerwań w dowolnej części cyklu programu nie jest zakłócona.
F. Odblokowywanie zapisu do wyjść nie jest zakłócone.	

Dane techniczne

A

A.1 Dane techniczne ogólne

Zgodność z normami

System automatyki S7-1200 jest zgodny z podanymi niżej normami i procedurami testowymi. Kryteria testowania systemu automatyki S7-1200 są oparte o te normy i procedury testowe.

Zatwierdzenie CE

CE

System automatyki S7-1200 spełnia wymagania oraz zapewnia stopień bezpieczeństwa zgodnie z wymienionymi niżej dyrektywami EC (Wspólnoty Europejskiej), a także jest zgodny ze zharmonizowanymi normami europejskimi (EN) dotyczącymi sterowników programowalnych, wymienionymi w *Official Journals of the European Community*.

- EC Directive 2006/95/EC (dyrektywa niskonapięciowa) „Electrical Equipment Designed for Use within Certain Voltage Limits”
  - EN 61131-2:2007 Sterowniki programowalne – Wymagania i testy sprzętu.
- EC Directive 2004/108/EC (dyrektywa EMC) „Electromagnetic Compatibility”
  - Norma emisyjności

EN 61000-6-4:2007: Środowisko przemysłowe.

  - Norma odporności

EN 61000-6-2:2005: Środowisko przemysłowe.

- EC Directive 94/9/EC (ATEX) „Equipment and Protective Systems Intended for Use in Potentially Explosive Atmosphere”
  - EN 60079-15:2005: Typ ochrony ‚n’.

Deklaracja zgodności CE jest przechowywana i udostępniana właściwym władzom w:

Siemens AG

IA AS RD ST PLC Amberg

Werner-von-Siemens-Str. 50

D92224 Amberg

Germany

Zatwierdzenie cULus

System automatyki S7-1200 jest zgodny z normami:

- Underwriters Laboratories Inc.: UL 508 Listed (Przemysłowe urządzenia sterujące).
- Canadian Standards Association: CSA C22.2 Number 142 (Sprzęt sterowania procesami).

UWAGA

Seria SIMATIC S7-1200 spełnia normę CSA.

Logo cULus oznacza, że S7-1200 był przebadany i jest certyfikowany przez Underwriters Laboratories (UL) na zgodność z normami UL 508 i CSA 22.2 No. 142.

Zatwierdzenie FM

Factory Mutual Research (FM):

Approval Standard Class Number 3600 i 3611

Zatwierdzony do zastosowania zgodnie z:

Class I, Division 2, Gas Group A, B, C, D, Temperature Class T4A Ta = 40° C.

Class I, Zone 2, IIC, Temperature Class T4 Ta = 40° C.

Zatwierdzenie ATEX

EN 60079-0:2006: Środowisko wybuchowe – Wymagania ogólne,

EN 60079-15:2005: Aparatura elektryczna do pracy w środowisku potencjalnie wybuchowym;

Typ zabezpieczenia ‚n'

II 3 G Ex nA II T4

Należy spełnić następujące specjalne warunki bezpiecznego stosowania S7-1200:

- Moduły muszą być instalowane w odpowiednich obudowach o stopniu zabezpieczenia co najmniej IP54 zgodnie z normą EN 60529 oraz uwzględniających warunki środowiskowe miejsca pracy urządzeń.
- Jeżeli nominalna temperatura pracy przekracza 70°C w miejscu wprowadzenia kabla lub 80°C w miejscu rozgałęziania przewodów, to dane temperaturowe wybranych kabli muszą być zgodne z faktycznie zmierzoną temperaturą.

- Należy podjąć środki zabezpieczające, by zakłócenia o charakterze przejściowym nie przekroczyły nominalnych wartości napięć o więcej niż 40 %.

Zatwierdzenie C-Tick

System automatyki S7-1200 spełnia wymagania norm AS/NZS 2064 (Class A).

Dopuszczenia morskie

Urządzenia z rodziny S7-1200 są regularnie poddawane zatwierdzeniom przez różne instytucje specjalne, właściwe dla specyficznych rynków i aplikacji. W celu uzyskania dodatkowych informacji dotyczących dokładnej listy uzyskanych ostatnio zatwierdzeń sporządzonej według numerów części, należy skontaktować się z lokalnym przedstawicielem firmy Siemens.

Urzędy klasyfikacyjne:

- ABS (American Bureau of Shipping)
- BV (Bureau Veritas)
- DNV (Det Norske Veritas)
- GL (Germanischer Lloyd)
- LRS (Lloyds Register of Shipping)
- Class NK (Nippon Kaiji Kyokai)

Środowiska przemysłowe

System automatyki S7-1200 jest zaprojektowany do zastosowania w środowiskach przemysłowych.

Zakres zastosowania	Wymagania dotyczące emisyjności zakłóceń	Wymagania dotyczące odporności na zakłócenia
Przemysłowe	EN 61000-6-4:2007	EN 61000-6-2:2005

Środowiska obszarów mieszkalnych

System automatyki S7-1200 może być stosowany na obszarach mieszkalnych pod warunkiem przedsięwzięcia odpowiednich środków, które zapewnią zgodność z ograniczeniami Class B normy EN 55011.

- S7-1200 musi być zainstalowany w uziemionej, metalowej obudowie.
- W liniach zasilających muszą być zainstalowane odpowiednie filtry zasilania.

Kompatybilność elektromagnetyczna

Kompatybilność elektromagnetyczna (EMC) jest zdolnością urządzenia elektrycznego do zgodnej z założeniami pracy w środowisku elektromagnetycznym, bez

emitowania takich poziomów zakłóceń elektromagnetycznych (EMI), które mogłyby zaburzyć pracę innych, pobliskich urządzeń elektromagnetycznych.

Kompatybilność elektromagnetyczna – odporność wg EN 61000-6-2	
EN 61000-4-2 Wyładowanie elektrostatyczne	8 kV wyładowanie przez powietrze do wszystkich powierzchni. 6 kV wyładowanie kontaktowe do odsłoniętych powierzchni przewodzących.
EN 61000-4-3 Promieniowane pole elektromagnetyczne	80 do 100 MHz, 10 V/m, 80% AM @ 1 kHz. 1,4 do 2,0 GHz, 3 V/m, 80% AM @ 1 kHz. 2,0 do 2,7 GHz, 1 V/m, 80% AM @ 1 kHz.
EN 61000-4-4 Szybkie impulsy przejściowe	2 kV, 5 kHz z obwodem sprzęgającym do zasilania AC i DC systemu. 2 kV, 5 kHz z zaciskami sprzęgającymi do I/O.
EN 6100-4-5 Odporność na udary	Systemy AC - 2 kV w trybie współbieżnym, 1kV w trybie różnicowym. Systemy DC - 2 kV w trybie współbieżnym, 1kV w trybie różnicowym. Dla systemów DC (sygnały I/O, systemy zasilania DC) wymagane jest zewnętrzne zabezpieczenie.
EN 61000-4-6 Zakłócenia przewodzone	150 kHz do 80 MHz, 10 V RMS, 80% AM @ 1kHz
EN 61000-4-11 Spadki napięć	Systemy AC 0% dla 1 cyklu, 40% dla 12 cykli i 70% dla 30 cykli @ 60 Hz.

Kompatybilność elektromagnetyczna – emisyjność przewodzona i promieniowana wg EN 61000-6-4	
Emisyjność przewodzona EN 55011, Class A, Group 1 0,15 MHz do 0,5 MHz 0,5 MHz do 5 MHz 5 MHz do 30 MHz	 <79dB (µV) quasi-pik; <66 dB (µV) średnio <73dB (µV) quasi-pik; <60 dB (µV) średnio <73dB (µV) quasi-pik; <60 dB (µV) średnio
Emisyjność promieniowana EN 55011, Class A, Group 1 30 MHz do 230 MHz 230 MHz do 1 GHz	 <40dB (µV/m) quasi-pik; pomiar w odległości 10m <47dB (µV/m) quasi-pik; pomiar w odległości 10m

Warunki środowiskowe

Warunki środowiskowe – transport i magazynowanie	
EN60068-2-2, Test Bb, sucho i gorąco EN60068-2-1, Test Ab, zimno	-40°C do +70°C
EN60068-2-30, Test Db, wilgotno i gorąco	25°C do 55°C, wilgotność 95%
EN60068-2-14, Test Na, szok temperaturowy	-40°C do +70°C czas przebywania 3 godziny, 2 cykle
EN 60068-2-32, swobodny upadek	0,3 m, 5 razy, produkt zapakowany
Ciśnienie atmosferyczne	1080 to 660 hPa (odpowiadające wysokości -1000 do 3500 m)

Warunki środowiskowe – praca	
Zakres temperatury otoczenia (wlot powietrza 25 mm poniżej urządzenia)	0°C do 55°C montaż poziomy, 0°C do 45°C montaż pionowy, wilgotność bez kondensacji 95 %
Ciśnienie atmosferyczne	1080 do 795 hPa (odpowiadające wysokości -1000 do 2000 m)
Stężenie zanieczyszczeń	S02: <0,5ppm; H2S: <0,1ppm; RH<60% bez kondensacji
EN60068-2-14, Test Nb, zmiana temperatury	5°C do 55°C, 3°C/minutę
EN60068-2-27 udar mechaniczny	15 G, impuls 11 ms, 6 udarów w każdym z 3 kierunków
EN60068-2-6 wibracje sinusoidalne	Montaż na szynie DIN: 3,5 mm od 5 – 9 Hz; 1 G od 9 – 150 Hz Montaż na panelu: 7,00 mm od 5 – 9 Hz; 2 G od 9 – 150 Hz 10 odchyleń w każdej osi, 1 oktawa/minutę

Test izolacji wysokim napięciem	
Obwody o napięciu znamionowym 24/5 V	500 VAC (test typu granic izolacji optycznej)
Obwody 115/230 V do uziemienia	1500 VAC test standardowy / 2500 VDC test typu
Obwody 115/230 V do obwodów 115/230 V	1500 VAC test standardowy /2500 VDC test typu
Obwody 115/230 V do obwodów 24/5 V	1500 VAC test standardowy /4242 VDC test typu

Klasa zabezpieczenia

- Protection Class I zgodnie z 60536 (Przewód zabezpieczający musi być podłączony do szyny montażowej).

Stopień zabezpieczenia

- Zabezpieczenie mechaniczne IP20, EN 60529.
- Zabezpieczenie przeciwko dotknięciu palcami wysokiego napięcia, zgodnie z testami standardową sondą. Wymagane jest zabezpieczenie zewnętrzne w przypadku kurzu, brudu, wody i ciał obcych o średnicy < 12,5 mm.

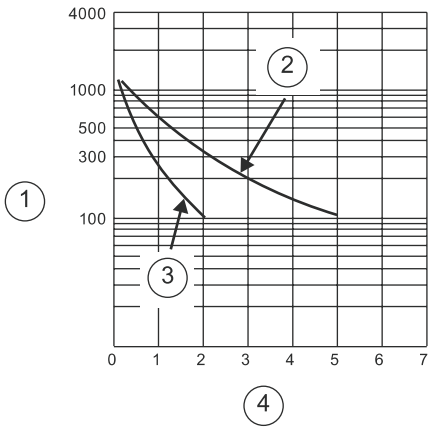
Napięcia znamionowe

Napięcie znamionowe	Tolerancja
24 VDC	20,4 VDC do 28,8 VDC
120/230 VAC	85 VAC do 264 VAC, 47 do 63 Hz

UWAGA
Kiedy przełącznik mechaniczny załącza zasilanie CPU S7-1200 lub dowolnego cyfrowego modułu sygnałowego, wtedy przez okres około 50 mikrosekund na ich wyjściach cyfrowych ustawia się stan „1”. Użytkownik musi to uwzględnić, zwłaszcza gdy używa urządzeń reagujących na impulsy o krótkim czasie trwania.

Żywotność przekaźników elektrycznych

Typowe dane dla przekaźników, dostarczane przez ich producentów, są przedstawione na rysunku poniżej. Rzeczywiste parametry mogą się zmieniać w zależności od konkretnej aplikacji. Zewnętrzny układ zabezpieczający, dostosowany do obciążenia wydłuża czas życia styków.



① Czas życia (× 10^3 operacji).
② 250 VAC obciążenie rezystancyjne,
30 VDC obciążenie rezystancyjne.
③ 250 VAC obciążenie indukcyjne (współczynnik mocy = 0,4),
30 VDC obciążenie rezystancyjne (L/R = 7 ms).
④ roboczy prąd znamionowy [A].

A.2 CPU

A.2.1 Dane techniczne CPU 1211C

Dane techniczne			
Model	**CPU 1211C AC/DC/Przekaźnik**	**CPU 1211C DC/DC/Przekaźnik**	**CPU 1211C DC/DC/DC**
Nr zamówieniowy (MLFB)	6ES7 211-1BD30-0XB0	6ES7 211-1HD30-0XB0	6ES7 211-1AD30-0XB0
Ogólne			
Wymiary W x H x D [mm]	90 x 100 x 75		
Masa	420 g	380 g	370 g
Pobór mocy	10 W	8 W	
Wydajność prądowa (magistrala CM)	750 mA maks. (5 VDC)		
Wydajność prądowa (24 VDC)	300 mA maks. (zasilanie czujników)		

Dane techniczne			
Model	**CPU 1211C AC/DC/Przekaźnik**	**CPU 1211C DC/DC/Przekaźnik**	**CPU 1211C DC/DC/DC**
Pobór prądu przez wejścia cyfrowe (24 VDC)	4 mA/wykorzystane wejście		
Charakterystyka CPU			
Pamięć użytkownika	25 KB pamięci roboczej / 1 MB pamięci ładowania / 2 KB pamięci trwałej		
Wbudowane cyfrowe I/O	6 wejść/4 wyjścia		
Wbudowane analogowe I/O	2 wejścia		
Rozmiar obrazu procesu	1024 bajty dla wejść /1024 bajty dla wyjść		
Rozszerzające moduły sygnałowe	brak		
Rozszerzająca płytka sygnałowa	1 SB maks.		
Rozszerzające moduły komunikacyjne	3 CM maks.		
Szybkie liczniki	łącznie 3 jednofazowe: 3 @ 100 kHz kwadraturowe: 3 @ 80 kHz		
Wyjścia impulsowe	2 @ częstotliwości 1 Hz		2 @ częstotliwości 100 Hz
Wejścia rejestrujące impulsy	6		
Przerwania od opóźnienia / cykliczne	łącznie 4 z rozdzielczością 1 ms		
Przerwania od zboczy	6 dla zboczy narastających i 6 dla zboczy opadających (10 i 10 z opcjonalna płytką sygnałową)		
Karta pamięci	SIMATIC Memory Card (opcjonalnie)		
Dokładność zegara czasu rzeczywistego	+/- 60 sekund/miesiąc		
Czas podtrzymywania zegara czasu rzeczywistego	10 dni typ./6 dni min. @ 40°C (bezobsługowy Super-kondensator)		
Charakterystyki			
Szybkość wykonywania operacji boolowskich	0,1 µs/instrukcję		
Szybkość wykonywania operacji Move Word	12 µs/instrukcję		
Szybkość wykonywania operacji Real Math	18 µs/instrukcję		
Komunikacja			
Liczba portów	1		
Typ	Ethernet		

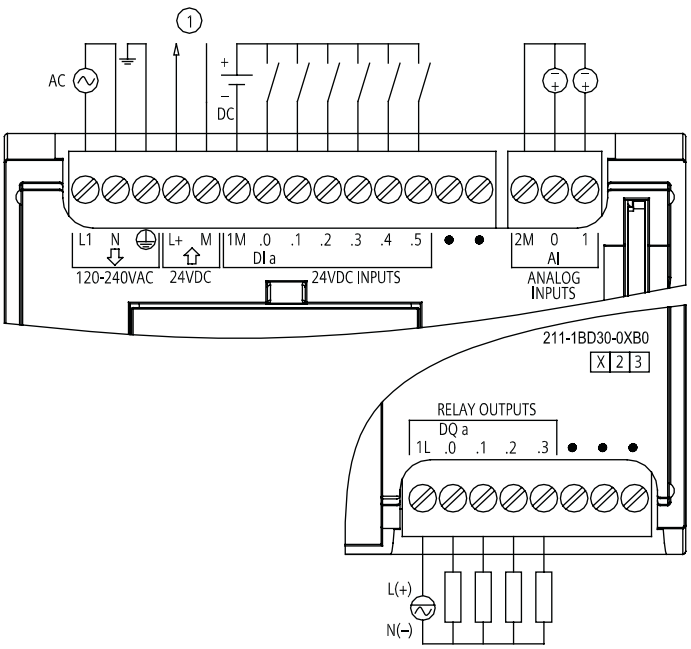
Dane techniczne			
Model	**CPU 1211C AC/DC/Przekaźnik**	**CPU 1211C DC/DC/Przekaźnik**	**CPU 1211C DC/DC/DC**
Szybkość przesyłu danych	10/100 Mb/s		
Izolacja (sygnału zewnętrznego od logiki PLC)	izolacja transformatorowa, 1500 VDC		
Typ kabla	CAT5e ekranowany		
Zasilanie			
Zakres napięć	85 do 264 VAC	20,4 do 28,8 VDC	
Częstotliwość sieci zasilającej	47 do 63 Hz	--	
Prąd wejściowy CPU tylko CPU, w warunkach maks. obciążenia CPU z wszystkimi układami rozszerzającymi, w warunkach maks. obciążenia	 60 mA @ 120 VAC 30 mA @ 240 VAC 180 mA @ 120 VAC 90 mA @ 240 VAC	 300 mA @ 24 VDC 900 mA @ 24 VDC	
Prąd rozruchowy (maks.)	20 A @ 264 VAC	12 A @ 28,8 VDC	
Izolacja (wejścia zasilającego od logiki)	1500 VAC	nieizolowane	
Czas podtrzymania (przy utracie zasilania)	20 ms @ 120 VAC 80 ms @ 240 VAC	10 ms @ 24 VDC	
Wewnętrzny bezpiecznik topikowy, niewymienialny przez użytkownika	3 A, 250 V, zwłoczny		
Zasilanie czujników			
Zakres napięć	20,4 do 28,8 VDC	L+ minus 4 VDC min.	
Prąd wyjściowy (maks.)	300 mA (z zabezpieczeniem przeciwzwarciowym)		
Maksymalne tętnienia (<10 MHz)	< 1 Vpp (wartość międzyszczytowa)	Takie same jak na linii zasilającej	
Izolacja (logiki CPU od zasilania czujników)	nieizolowane		
Wejścia cyfrowe			
Liczba wejść	6		
Typ	prąd wpływający/wypływający (IEC Type 1 sink)		
Napięcie	24 VDC @ 4 mA, wartość nominalna		

Dane techniczne			
Model	**CPU 1211C AC/DC/Przekaźnik**	**CPU 1211C DC/DC/Przekaźnik**	**CPU 1211C DC/DC/DC**
Ciągłe dopuszczalne napięcie	30 VDC, maks.		
Udar napięciowy	35 VDC przez 0,5 s		
Sygnał logiczny 1 (min.)	15 VDC @ 2,5 mA		
Sygnał logiczny 0 (maks.)	5 VDC @ 1 mA		
Izolacja (od strony wyjściowej do logiki)	500 VAC przez 1 minutę		
Grupy izolacji	1		
Czasy filtru	0,2, 0,4, 0,8, 1,6, 3,2, 6,4 i 12,8 ms (wybierane w grupach po 4)		
Szybkość zegara HSC (maks.) (Poziom logiczny 1 = 15 do 26 VDC)	jednofazowego: 100 kHz kwadraturowego: 80 kHz		
Liczba wejść znajdujących się jednocześnie w stanie włączonym	6		
Długość kabla (w metrach)	500 ekranowany, 300 nieekranowany, 50 ekranowany – wejście HSC		
Wejścia analogowe			
Liczba wejść	2		
Typ	napięciowe (niesymetryczne)		
Zakres	0 do 10 V		
Zakres pomiarowy (słowo danych)	0 do 27648 (por. Napięciowa reprezentacja wejścia analogowego)		
Zakres przerzutu (słowo danych)	27649 do 32511 (por. Napięciowa reprezentacja wejścia analogowego)		
Przepełnienie (słowo danych)	32512 do 32767 (por. Napięciowa reprezentacja wejścia analogowego)		
Rozdzielczość	10 bitów		
Maksymalne bezpieczne napięcie	35 VDC		
Wygładzanie	None (brak), Weak (słabe), Medium (średnie) lub Strong (mocne) (w celu uzyskania informacji o czasach odpowiedzi, por. Czas odpowiedzi wejścia analogowego)		
Tłumienie zakłóceń	10, 50 lub 60 Hz (w celu uzyskania informacji o częstości próbkowania, por. Czas odpowiedzi wejścia analogowego)		
Impedancja	≥100 kΩ		
Izolacja (sygnału zewnętrznego od logiki)	brak		

Dane techniczne			
Model	**CPU 1211C AC/DC/Przekaźnik**	**CPU 1211C DC/DC/Przekaźnik**	**CPU 1211C DC/DC/DC**
Dokładność (25°C / 0 do 55°C)	3,0 % / 3,5 % pełnego zakresu		
Tłumienie sygnału sumacyjnego	40 dB, DC do 60 Hz		
Zakres operacyjny sygnału	sygnał plus napięcie sumacyjne musi być mniejsze niż +12 V i większe niż –12 V		
Długość kabla (w metrach)	10 m, ekranowana para skręconych przewodów		
Wyjścia cyfrowe			
Liczba wyjść	4		
Typ	przekaźnik, styki suche		półprzewodnik - MOSFET
Zakres napięć	5 do 30 VDC lub 5 do 250 VAC		20,4 do 28,8 VDC
Sygnał logiczny 1 przy maks. Prądzie	--		20 VDC min.
Sygnał logiczny 0 przy obciążeniu 10 kΩ	--		0,1 VDC maks.
Prąd (maks.)	2,0 A		0,5 A
Obciążenie żarówką	30 W DC / 200 W AC		5 W
Rezystancja w stanie ON	0,2 Ω maks. w stanie nowości		0,6 Ω maks.
Prąd upływu na jeden punkt	--		10 µA maks.
Udar prądowy	7 A z zamkniętymi stykami		8 A przez 100 ms maks.
Zabezpieczenie przed przeciążeniem	brak		
Izolacja (sygnału zewnętrznego od logiki)	1500 VAC przez 1 minutę (cewka do styku) brak (cewka do logiki)		500 VAC przez 1 minutę
Rezystancja izolacji	100 MΩ min. w stanie nowości		--
Izolacja między otwartymi stykami	750 VAC przez 1 minutę		--
Grupy izolacji	1		1
Ograniczanie przepięć indukcyjnych	--		L+ minus 48 VDC, 1 w mocy strat
Opóźnienie przełączania (Qa.0 do Qa.3)	10 ms maks.		1,0 µs maks. z OFF do ON 3,0 µs maks. z ON do OFF
Częstotliwość impulsów wyjściowych (Qa.0 i Qa.2)	1 Hz maks.		100 kHz maks.

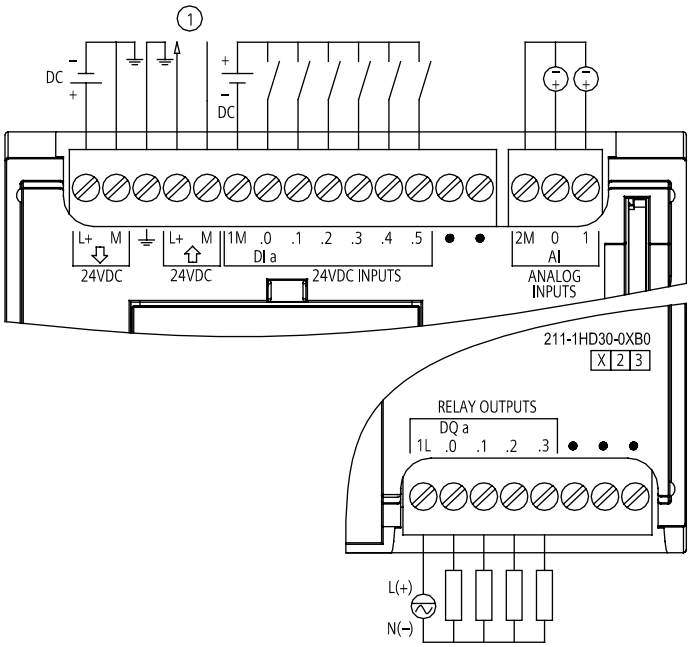
Dane techniczne			
Model	**CPU 1211C AC/DC/Przekaźnik**	**CPU 1211C DC/DC/Przekaźnik**	**CPU 1211C DC/DC/DC**
Trwałość mechaniczna (bez obciążenia)	10000000 cykli załącz/wyłącz		--
Trwałość styków przy nominalnym obciążeniu	100000 cykli załącz/wyłącz		--
Zachowanie przy przejściu z RUN do STOP	Ostatnia wartość lub wartość zastępcza (domyślnie 0)		
Liczba wyjść znajdujących się jednocześnie w stanie włączonym	4		
Długość kabla (w metrach)	500 ekranowany, 150 nieekranowany		

Schematy połączeń



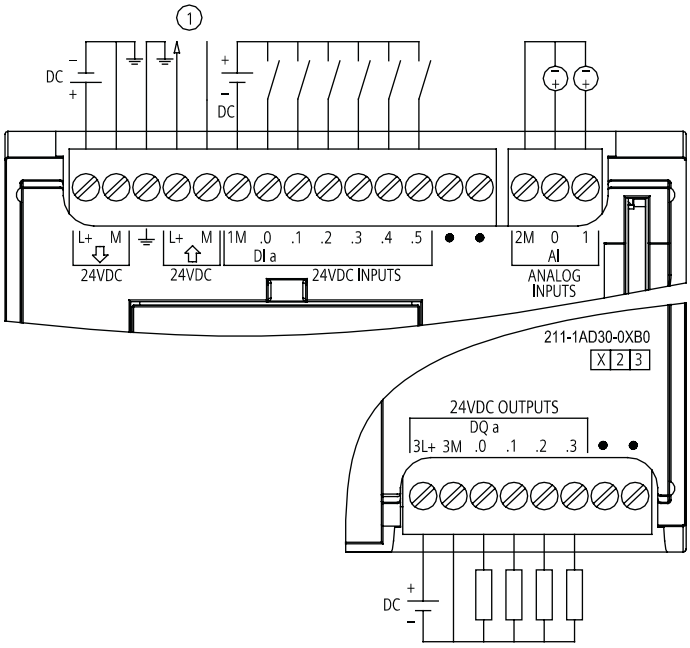
① Wyjście zasilacza czujników 24 VDC

Rysunek A-1 CPU 1211C AC/DC/Przekaźnik[mk5] (6ES7 211-1BD30-0XB0)



① Wyjście zasilacza czujników 24 VDC

Rysunek A-2 CPU 1211C DC/DC/Przekaźnik (6ES7 211-1HD30-0XB0)



① Wyjście zasilacza czujników 24 VDC

Rysunek A-3 CPU 1211C DC/DC/DC (6ES7 211-1AD30-0XB0)

A.2.2. Dane techniczne CPU 1212C

Dane techniczne			
Model	**CPU 1212C AC/DC/Przekaźnik**	**CPU 1212C DC/DC/Przekaźnik**	**CPU 1212C DC/DC/DC**
Nr zamówieniowy (MLFB)	6ES7 212-1BD30-0XB0	6ES7 212-1HD30-0XB0	6ES7 212-1AD30-0XB0
Ogólne			
Wymiary W x H x D [mm]	90 x 100 x 75		
Masa	425 g	385 g	370 g
Pobór mocy	11 W	9 W	
Wydajność prądowa (magistrala SM i CM)	1000 mA maks. (5 VDC)		
Wydajność prądowa (24 VDC)	300 mA maks. (zasilanie czujników)		
Pobór prądu przez wejścia cyfrowe (24 VDC)	4 mA/wykorzystane wejście		
Charakterystyka CPU			
Pamięć użytkownika	25 KB pamięci roboczej / 1 MB pamięci ładowania / 2 KB pamięci trwałej		
Wbudowane cyfrowe I/O	8 wejść/6 wyjścia		
Wbudowane analogowe I/O	2 wejścia		
Rozmiar obrazu procesu	1024 bajty dla wejść /1024 bajty dla wyjść		
Rozszerzające moduły sygnałowe	2 SM maks.		
Rozszerzająca płytka sygnałowa	1 SB maks.		
Rozszerzające moduły komunikacyjne	3 CM maks.		
Szybkie liczniki	łącznie 4 jednofazowe: 3 @ 100 kHz i 1 @ 30 kHz częstotliwości zegara kwadraturowe: 3 @ 80 kHz i 1 @ 20 kHz częstotliwości zegara		
Wyjścia impulsowe	2 @ częstotliwości 1 Hz		2 @ częstotliwości 100 Hz
Wejścia rejestrujące impulsy	8		
Przerwania od opóźnienia / cykliczne	łącznie 4 z rozdzielczością 1 ms		
Przerwania od zboczy	8 dla zboczy narastających i 8 dla zboczy opadających (12 i 12 z opcjonalna płytką sygnałową)		
Karta pamięci	SIMATIC Memory Card (opcjonalnie)		
Dokładność zegara czasu rzeczywistego	+/- 60 sekund/miesiąc		

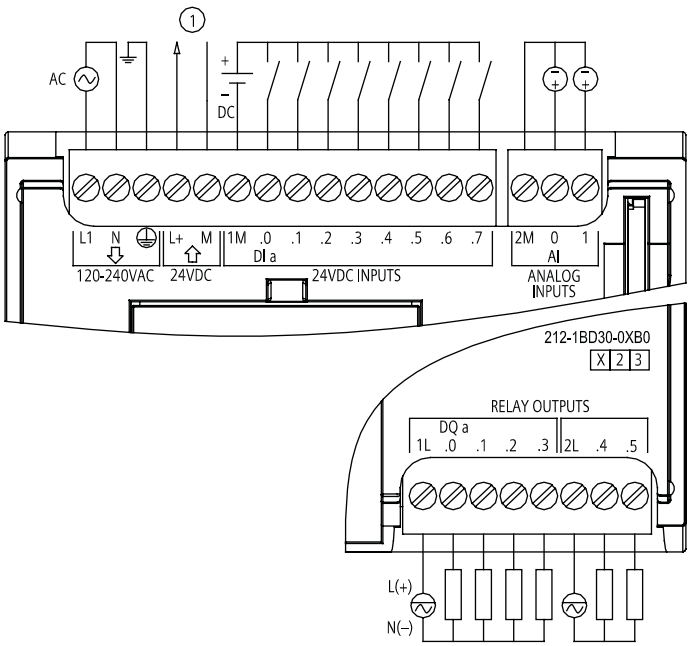
Dane techniczne			
Model	**CPU 1212C AC/DC/Przekaźnik**	**CPU 1212C DC/DC/Przekaźnik**	**CPU 1212C DC/DC/DC**
Czas podtrzymywania zegara czasu rzeczywistego	10 dni typ./6 dni min. @ 40°C (bezobsługowy Super-kondensator)		
Charakterystyki			
Szybkość wykonywania operacji boolowskich	0,1 µs/instrukcję		
Szybkość wykonywania operacji Move Word	12 µs/instrukcję		
Szybkość wykonywania operacji Real Math	18 µs/instrukcję		
Komunikacja			
Liczba portów	1		
Typ	Ethernet		
Szybkość przesyłu danych	10/100 Mb/s		
Izolacja (sygnału zewnętrznego od logiki PLC)	izolacja transformatorowa, 1500 VDC		
Typ kabla	CAT5e ekranowany		
Zasilanie			
Zakres napięć	85 do 264 VAC	20,4 do 28,8 VDC	
Częstotliwość sieci zasilającej	47 do 63 Hz	--	
Prąd wejściowy CPU tylko CPU, w warunkach maks. obciążenia CPU z wszystkimi układami rozszerzającymi, w warunkach maks. obciążenia	80 mA @ 120 VAC 40 mA @ 240 VAC 240 mA @ 120 VAC 120 mA @ 240 VAC	400 mA @ 24 VDC 1200 mA @ 24 VDC	
Prąd rozruchowy (maks.)	20 A @ 264 VAC	12 A @ 28,8 VDC	
Izolacja (wejścia zasilającego od logiki)	1500 VAC	nieizolowane	
Czas podtrzymania (przy utracie zasilania)	20 ms @ 120 VAC 80 ms @ 240 VAC	10 ms @ 24 VDC	
Wewnętrzny bezpiecznik topikowy, niewymienialny przez użytkownika	3 A, 250 V, zwłoczny		
Zasilanie czujników			
Zakres napięć	20,4 do 28,8 VDC	L+ minus 4 VDC min.	
Prąd wyjściowy (maks.)	300 mA (z zabezpieczeniem przeciwzwarciowym)		

Dane techniczne			
Model	**CPU 1212C AC/DC/Przekaźnik**	**CPU 1212C DC/DC/Przekaźnik**	**CPU 1212C DC/DC/DC**
Maksymalne tętnienia (<10 MHz)	< 1 Vpp (wartość międzyszczytowa)	Takie same jak na linii zasilającej	
Izolacja (logiki CPU od zasilania czujników)	nieizolowane		
Wejścia cyfrowe			
Liczba wejść	8		
Typ	prąd wpływający/wypływający (IEC Type 1 sink)		
Napięcie	24 VDC @ 4 mA, wartość nominalna		
Ciągłe dopuszczalne napięcie	30 VDC, maks.		
Udar napięciowy	35 VDC przez 0,5 s		
Sygnał logiczny 1 (min.)	15 VDC @ 2,5 mA		
Sygnał logiczny 0 (maks.)	5 VDC @ 1 mA		
Izolacja (od strony wyjściowej do logiki)	500 VAC przez 1 minutę		
Grupy izolacji	1		
Czasy filtru	0,2, 0,4, 0,8, 1,6, 3,2, 6,4 i 12,8 ms (wybierane w grupach po 4)		
Szybkość zegara HSC (maks.) (Poziom logiczny 1 = 15 do 26 VDC)	jednofazowego: 100 kHz (Ia.0 do Ia.5) i 30 kHz (Ia.6 do Ia.7) kwadraturowego: 80 kHz (Ia.0 do Ia.5) i 20 kHz (Ia.6 do Ia.7)		
Liczba wejść znajdujących się jednocześnie w stanie włączonym	8		
Długość kabla (w metrach)	500 ekranowany, 300 nieekranowany, 50 ekranowany – wejście HSC		
Wejścia analogowe			
Liczba wejść	2		
Typ	napięciowe (niesymetryczne)		
Zakres	0 do 10 V		
Zakres pomiarowy (słowo danych)	0 do 27648 (por. Napięciowa reprezentacja wejścia analogowego)		
Zakres przerzutu (słowo danych)	27649 do 32511 (por. Napięciowa reprezentacja wejścia analogowego)		
Przepełnienie (słowo danych)	32512 do 32767 (por. Napięciowa reprezentacja wejścia analogowego)		
Rozdzielczość	10 bitów		
Maksymalne bezpieczne napięcie	35 VDC		

Dane techniczne			
Model	**CPU 1212C AC/DC/Przekaźnik**	**CPU 1212C DC/DC/Przekaźnik**	**CPU 1212C DC/DC/DC**
Wygładzanie	None (brak), Weak (słabe), Medium (średnie) lub Strong (mocne) (w celu uzyskania informacji o czasach odpowiedzi, por. Czas odpowiedzi wejścia analogowego)		
Tłumienie zakłóceń	10, 50 lub 60 Hz (w celu uzyskania informacji o częstości próbkowania, por. Czas odpowiedzi wejścia analogowego)		
Impedancja	≥100 kΩ		
Izolacja (sygnału zewnętrznego od logiki)	brak		
Dokładność (25°C / 0 do 55°C)	3,0 % / 3,5 % pełnego zakresu		
Tłumienie sygnału sumacyjnego	40 dB, DC do 60 Hz		
Zakres operacyjny sygnału	sygnał plus napięcie sumacyjne musi być mniejsze niż +12 V i większe niż -12 V		
Długość kabla (w metrach)	10 m, ekranowana para skręconych przewodów		
Wyjścia cyfrowe			
Liczba wyjść	6		
Typ	przekaźnik, styki suche		półprzewodnik - MOSFET
Zakres napięć	5 do 30 VDC lub 5 do 250 VAC		20,4 do 28,8 VDC
Sygnał logiczny 1 przy maks. prądzie	--		20 VDC min.
Sygnał logiczny 0 przy obciążeniu 10 kΩ	--		0,1 VDC maks.
Prąd (maks.)	2,0 A		0,5 A
Obciążenie żarówką	30 W DC / 200 W AC		5 W
Rezystancja w stanie ON	0,2 Ω maks. w stanie nowości		0,6 Ω maks.
Prąd upływu na jeden punkt	--		10 µA maks.
Udar prądowy	7 A z zamkniętymi stykami		8 A przez 100 ms maks.
Zabezpieczenie przed przeciążeniem	brak		
Izolacja (sygnału zewnętrznego od logiki)	1500 VAC przez 1 minutę (cewka do styku) brak (cewka do logiki)		500 VAC przez 1 minutę
Rezystancja izolacji	100 MΩ min. w stanie nowości		--
Izolacja między otwartymi stykami	750 VAC przez 1 minutę		--
Grupy izolacji	2		1
Ograniczanie przepięć indukcyjnych	--		L+ minus 48 VDC, 1 w mocy strat

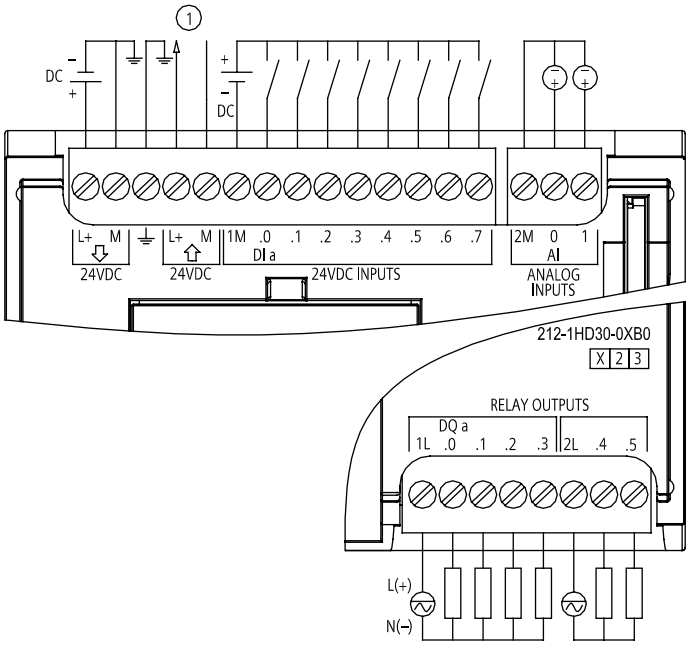
Dane techniczne			
Model	**CPU 1212C AC/DC/Przekaźnik**	**CPU 1212C DC/DC/Przekaźnik**	**CPU 1212C DC/DC/DC**
Opóźnienie przełączania (Qa.0 do Qa.3)	10 ms maks.		1,0 µs maks. z OFF do ON 3,0 µs maks. z ON do OFF
Częstotliwość impulsów wyjściowych (Qa.0 i Qa.2)	1 Hz maks.		100 kHz maks.
Trwałość mechaniczna (bez obciążenia)	10000000 cykli załącz/wyłącz		--
Trwałość styków przy nominalnym obciążeniu	100000 cykli załącz/wyłącz		--
Zachowanie przy przejściu z RUN do STOP	Ostatnia wartość lub wartość zastępcza (domyślnie 0)		
Liczba wyjść znajdujących się jednocześnie w stanie włączonym	4		
Długość kabla (w metrach)	500 ekranowany, 150 nieekranowany		

Schematy połączeń



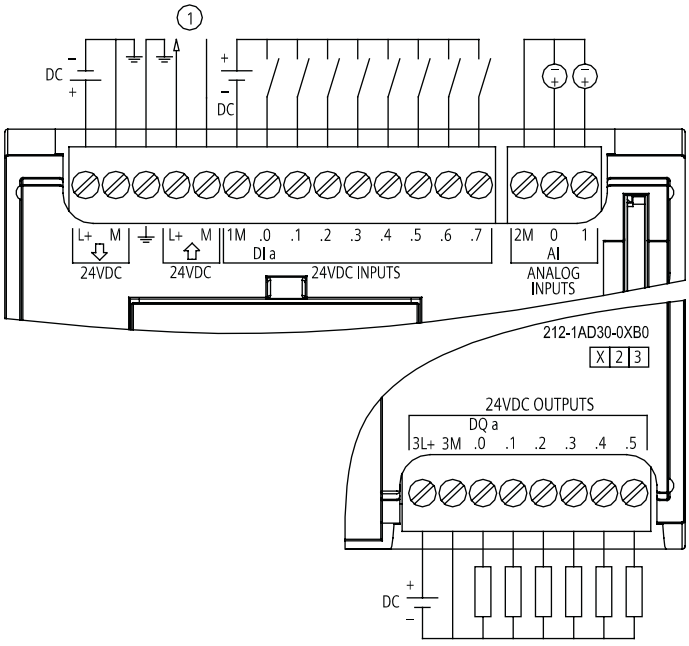
① Wyjście zasilacza czujników 24 VDC

Rysunek A-4 CPU 1212C AC/DC/Przekaźnik (6ES7 212-1BD30-0XB0)



① Wyjście zasilacza czujników 24 VDC

Rysunek A-5 CPU 1212C DC/DC/Przekaźnik (6ES7 212-1HD30-0XB0)



① Wyjście zasilacza czujników 24 VDC

Rysunek A-6 CPU 1212C DC/DC/DC (6ES7 212-1AD30-0XB0)

A.2.3 Dane techniczne CPU 1214C

Dane techniczne			
Model	**CPU 1214C AC/DC/Przekaźnik**	**CPU 1214C DC/DC/Przekaźnik**	**CPU 1214C DC/DC/DC**
Nr zamówieniowy (MLFB)	6ES7 214-1BE30-0XB0	6ES7 214-1HE30-0XB0	6ES7 214-1AE30-0XB0
Ogólne			
Wymiary W x H x D [mm]	90 x 100 x 75		
Masa	475 g	435 g	415 g
Pobór mocy	14 W	12 W	
Wydajność prądowa (magistrala SM i CM)	1600 mA maks. (5 VDC)		
Wydajność prądowa (24 VDC)	400 mA maks. (zasilanie czujników)		
Pobór prądu przez wejścia cyfrowe (24 VDC)	4 mA/wykorzystane wejście		
Charakterystyka CPU			
Pamięć użytkownika	50 KB pamięci roboczej / 2 MB pamięci ładowania / 2 KB pamięci trwałej		
Wbudowane cyfrowe I/O	14 wejść/10 wyjścia		
Wbudowane analogowe I/O	2 wejścia		
Rozmiar obrazu procesu	1024 bajty dla wejść /1024 bajty dla wyjść		
Rozszerzające moduły sygnałowe	8 SM maks.		
Rozszerzająca płytka sygnałowa	1 SB maks.		
Rozszerzające moduły komunikacyjne	3 CM maks.		
Szybkie liczniki	łącznie 6 jednofazowe: 3 @ 100 kHz i 1 @ 30 kHz częstotliwości zegara kwadraturowe: 3 @ 80 kHz i 1 @ 20 kHz częstotliwości zegara		
Wyjścia impulsowe	2 @ częstotliwości 1 Hz		2 @ częstotliwości 100 Hz
Wejścia rejestrujące impulsy	14		
Przerwania od opóźnienia / cykliczne	łącznie 4 z rozdzielczością 1 ms		
Przerwania od zboczy	12 dla zboczy narastających i 12 dla zboczy opadających (14 i 14 z opcjonalna płytką sygnałową)		
Karta pamięci	SIMATIC Memory Card (opcjonalnie)		
Dokładność zegara czasu rzeczywistego	+/- 60 sekund/miesiąc		

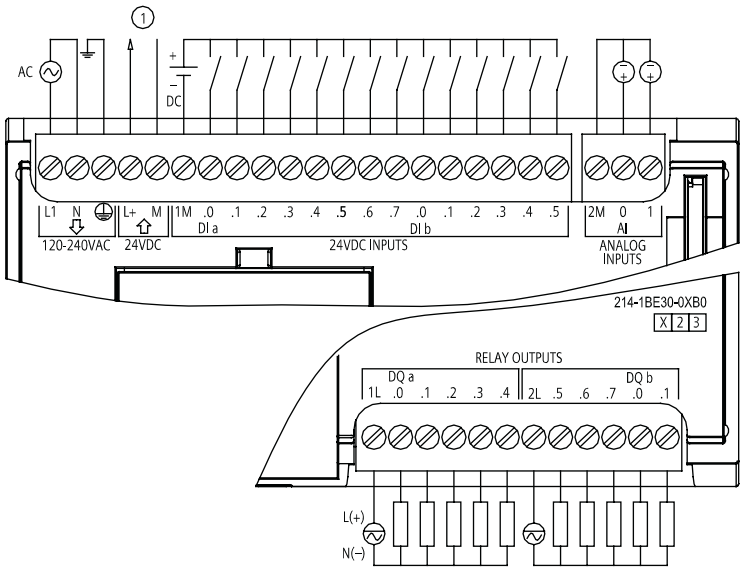
Dane techniczne			
Model	**CPU 1214C AC/DC/Przekaźnik**	**CPU 1214C DC/DC/Przekaźnik**	**CPU 1214C DC/DC/DC**
Czas podtrzymywania zegara czasu rzeczywistego	10 dni typ./6 dni min. @ 40°C (bezobsługowy Super-kondensator)		
Charakterystyki			
Szybkość wykonywania operacji boolowskich	0,1 µs/instrukcję		
Szybkość wykonywania operacji Move Word	12 µs/instrukcję		
Szybkość wykonywania operacji Real Math	18 µs/instrukcję		
Komunikacja			
Liczba portów	1		
Typ	Ethernet		
Szybkość przesyłu danych	10/100 Mb/s		
Izolacja (sygnału zewnętrznego od logiki PLC)	izolacja transformatorowa, 1500 VDC		
Typ kabla	CAT5e ekranowany		
Zasilanie			
Zakres napięć	85 do 264 VAC	20,4 do 28,8 VDC	
Częstotliwość sieci zasilającej	47 do 63 Hz	--	
Prąd wejściowy CPU tylko CPU, w warunkach maks. obciążenia CPU z wszystkimi układami rozszerzającymi, w warunkach maks. obciążenia	100 mA @ 120 VAC 50 mA @ 240 VAC 300 mA @ 120 VAC 150 mA @ 240 VAC	500 mA @ 24 VDC 1500 mA @ 24 VDC	
Prąd rozruchowy (maks.)	20 A @ 264 VAC	12 A @ 28,8 VDC	
Izolacja (wejścia zasilającego od logiki)	1500 VAC	nieizolowane	
Czas podtrzymania (przy utracie zasilania)	20 ms @ 120 VAC 80 ms @ 240 VAC	10 ms @ 24 VDC	
Wewnętrzny bezpiecznik topikowy, niewymienialny przez użytkownika	3 A, 250 V, zwłoczny		
Zasilanie czujników			

Dane techniczne			
Model	**CPU 1214C AC/DC/Przekaźnik**	**CPU 1214C DC/DC/Przekaźnik**	**CPU 1214C DC/DC/DC**
Zakres napięć	20,4 do 28,8 VDC	L+ minus 4 VDC min.	
Prąd wyjściowy (maks.)	300 mA (z zabezpieczeniem przeciwzwarciowym)		
Maksymalne tętnienia (<10 MHz)	< 1 Vpp (wartość międzyszczytowa)	Takie same jak na linii zasilającej	
Izolacja (logiki CPU od zasilania czujników)	nieizolowane		
Wejścia cyfrowe			
Liczba wejść	14		
Typ	prąd wpływający/wypływający (IEC Type 1 sink)		
Napięcie	24 VDC @ 4 mA, wartość nominalna		
Ciągłe dopuszczalne napięcie	30 VDC, maks.		
Udar napięciowy	35 VDC przez 0,5 s		
Sygnał logiczny 1 (min.)	15 VDC @ 2,5 mA		
Sygnał logiczny 0 (maks.)	5 VDC @ 1 mA		
Izolacja (od strony wyjściowej do logiki)	500 VAC przez 1 minutę		
Grupy izolacji	1		
Czasy filtru	0,2, 0,4, 0,8, 1,6, 3,2, 6,4 i 12,8 ms (wybierane w grupach po 4)		
Szybkość zegara HSC (maks.) (Poziom logiczny 1 = 15 do 26 VDC)	jednofazowego: 100 kHz (Ia.0 do Ia.5) i 30 kHz (Ia.6 do Ib.5) kwadraturowego: 80 kHz (Ia.0 do Ia.5) i 20 kHz (Ia.6 do Ib.7)		
Liczba wejść znajdujących się jednocześnie w stanie włączonym	14		
Długość kabla (w metrach)	500 ekranowany, 300 nieekranowany, 50 ekranowany – wejście HSC		
Wejścia analogowe			
Liczba wejść	2		
Typ	napięciowe (niesymetryczne)		
Zakres	0 do 10 V		
Zakres pomiarowy (słowo danych)	0 do 27648 (por. Napięciowa reprezentacja wejścia analogowego)		
Zakres przerzutu (słowo danych)	27649 do 32511 (por. Napięciowa reprezentacja wejścia analogowego)		
Przepełnienie (słowo danych)	32512 do 32767 (por. Napięciowa reprezentacja wejścia analogowego)		
Rozdzielczość	10 bitów		

Dane techniczne			
Model	**CPU 1214C AC/DC/Przekaźnik**	**CPU 1214C DC/DC/Przekaźnik**	**CPU 1214C DC/DC/DC**
Maksymalne bezpieczne napięcie	35 VDC		
Wygładzanie	None (brak), Weak (słabe), Medium (średnie) lub Strong (mocne) (w celu uzyskania informacji o czasach odpowiedzi, por. Czas odpowiedzi wejścia analogowego)		
Tłumienie zakłóceń	10, 50 lub 60 Hz (w celu uzyskania informacji o częstości próbkowania, por. Czas odpowiedzi wejścia analogowego)		
Impedancja	≥100 kΩ		
Izolacja (sygnału zewnętrznego od logiki)	brak		
Dokładność (25°C / 0 do 55°C)	3,0 % / 3,5 % pełnego zakresu		
Tłumienie sygnału sumacyjnego	40 dB, DC do 60 Hz		
Zakres operacyjny sygnału	sygnał plus napięcie sumacyjne musi być mniejsze niż +12 V i większe niż -12 V		
Długość kabla (w metrach)	10 m, ekranowana para skręconych przewodów		
Wyjścia cyfrowe			
Liczba wyjść	10		
Typ	przekaźnik, styki suche		półprzewodnik - MOSFET
Zakres napięć	5 do 30 VDC lub 5 do 250 VAC		20,4 do 28,8 VDC
Sygnał logiczny 1 przy maks. prądzie	--		20 VDC min.
Sygnał logiczny 0 przy obciążeniu 10 kΩ	--		0,1 VDC maks.
Prąd (maks.)	2,0 A		0,5 A
Obciążenie żarówką	30 W DC / 200 W AC		5 W
Rezystancja w stanie ON	0,2 Ω maks. w stanie nowości		0,6 Ω maks.
Prąd upływu na jeden punkt	--		10 µA maks.
Udar prądowy	7 A z zamkniętymi stykami		8 A przez 100 ms maks.
Zabezpieczenie przed przeciążeniem	brak		
Izolacja (sygnału zewnętrznego od logiki)	1500 VAC przez 1 minutę (cewka do styku) brak (cewka do logiki)		500 VAC przez 1 minutę
Rezystancja izolacji	100 MΩ min. w stanie nowości		--
Izolacja między otwartymi stykami	750 VAC przez 1 minutę		--

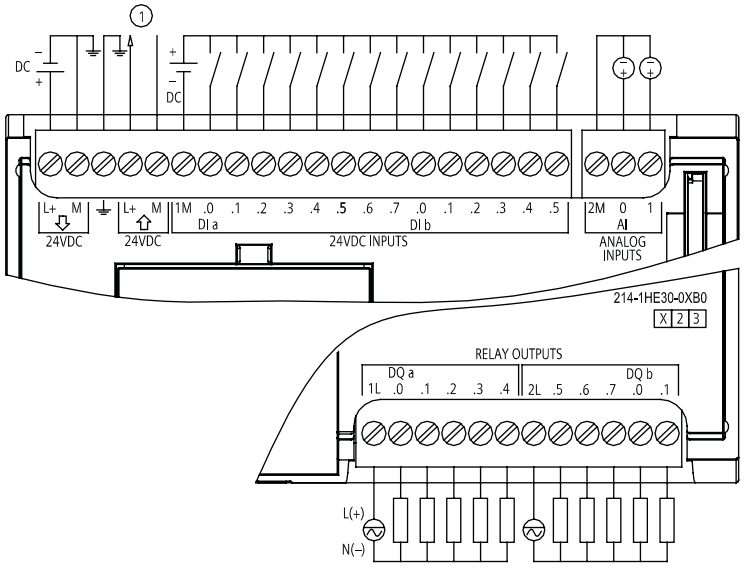
Dane techniczne			
Model	**CPU 1214C AC/DC/Przekaźnik**	**CPU 1214C DC/DC/Przekaźnik**	**CPU 1214C DC/DC/DC**
Grupy izolacji	2		1
Ograniczanie przepięć indukcyjnych	--		L+ minus 48 VDC, 1 W mocy strat
Opóźnienie przełączania (Qa.0 do Qa.3)	10 ms maks.		1,0 µs maks. z OFF do ON 3,0 µs maks. z ON do OFF
Opóźnienie przełączania (Qa.4 do Qb.1)	10 ms maks.		50 µs maks. z OFF do ON 200 µs maks. z ON do OFF
Częstotliwość impulsów wyjściowych (Qa.0 i Qa.2)	1 Hz maks.		100 kHz maks.
Trwałość mechaniczna (bez obciążenia)	10000000 cykli załącz/wyłącz		--
Trwałość styków przy nominalnym obciążeniu	100000 cykli załącz/wyłącz		--
Zachowanie przy przejściu z RUN do STOP	Ostatnia wartość lub wartość zastępcza (domyślnie 0)		
Liczba wyjść znajdujących się jednocześnie w stanie włączonym	10		
Długość kabla (w metrach)	500 ekranowany, 150 nieekranowany		

Schematy połączeń



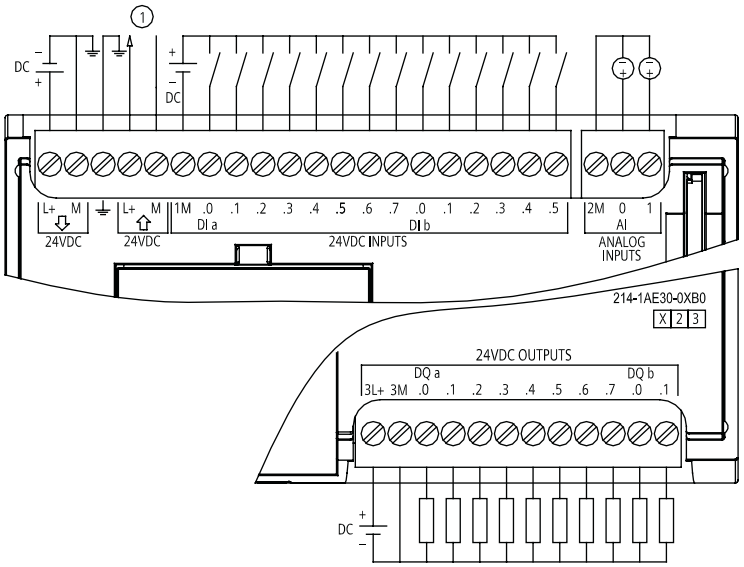
① Wyjście zasilacza czujników 24 VDC

Rysunek A-7 CPU 1214C AC/DC/Relay (6ES7 214-1BE30-0XB0)



① Wyjście zasilacza czujników 24 VDC

Rysunek A-8 CPU 1214C DC/DC/Relay (6ES7 214-1HE30-0XB0)



① Wyjście zasilacza czujników 24 VDC

Rysunek A-9 CPU 1214C DC/DC/DC (6ES7 214-1AE30-0XB0)

A.3 Cyfrowe moduły rozszerzeń (SM)

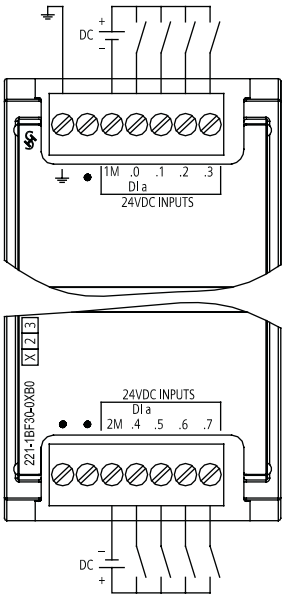
A.3.1 Dane techniczne modułu wejść cyfrowych SM 1221

Dane techniczne		
Model	**SM 1221 DI 8x24VDC**	**SM 1221 DI 16x24VDC**
Nr zamówieniowy (MLFB)	6ES7 221-1BF30-0XB0	6ES7 221-1BH30-0XB0
Ogólne		
Wymiary W x H x D [mm]	45 x 100 x 75	
Masa	170 g	210 g
Pobór mocy	1,5 W	2,5 W
Pobór prądu (magistrala SM)	105 mA	130 mA
Pobór prądu (24 VDC)	4 mA/wykorzystane wejście	4 mA/wykorzystane wejście
Wejścia cyfrowe		
Liczba wejść	8	16
Typ	prąd wpływający/wypływający (IEC Type 1 sink)	
Napięcie	24 VDC @ 4 mA, wartość nominalna	
Ciągłe dopuszczalne napięcie	30 VDC, maks.	
Udar napięciowy	35 VDC przez 0,5 s	
Sygnał logiczny 1 (min.)	15 VDC @ 2,5 mA	
Sygnał logiczny 0 (maks.)	5 VDC @ 1 mA	

Dane techniczne		
Model	**SM 1221 DI 8x24VDC**	**SM 1221 DI 16x24VDC**
Izolacja (od strony wyjściowej do logiki)	500 VAC przez 1 minutę	
Grupy izolacji	2	4
Czasy filtru	0,2, 0,4, 0,8, 1,6, 3,2, 6,4 i 12,8 ms (wybierane w grupach po 4)	
Liczba wejść znajdujących się jednocześnie w stanie włączonym	8	16
Długość kabla (w metrach)	500 ekranowany, 300 nieekranowany	

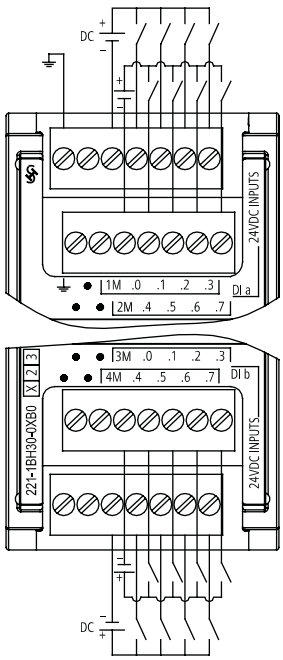
Schematy połączeń

SM 1221 DI 8 x 24 VDC



6ES7 221-1BF30-0XB0

SM 1221 DI 16 x 24 VDC



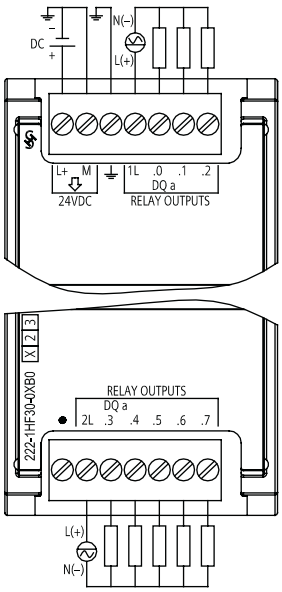
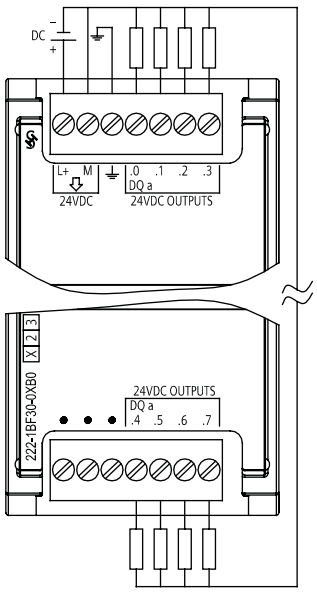
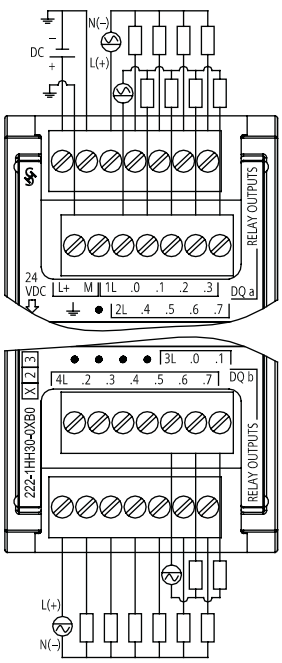
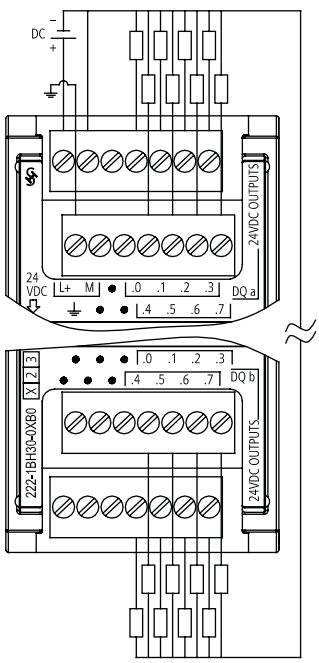
6ES7 221-1BH30-0XB0

A.3.2 Dane techniczne modułu wyjść cyfrowych SM 1222

Dane techniczne				
Model	**SM 1222 DQ 8xPrzekaźnik**	**SM1222 DQ 16xPrzekaźnik**	**SM1222 DQ 8x24VDC**	**SM1222 DQ 16x24VDC**
Nr zamówieniowy (MLFB)	6ES7 222-1HF30-0XB0	6ES7 222-1HH30-0XB0	6ES7 222-1BF30-0XB0	6ES7 222-1BH30-0XB0
Ogólne				
Wymiary W x H x D [mm]	45 x 100 x 75			
Masa	190 g	260 g	180 g	220 g
Pobór mocy	4,5 W	8,5 W	1,5 W	2,5 W
Pobór prądu (magistrala SM)	120 mA	135 mA	120 mA	140 mA
Pobór prądu (24 VDC)	11 mA /wykorzystaną cewkę przekaźnika		--	
Wyjścia cyfrowe				
Liczba wyjść	8	16	8	16
Typ	przekaźnik, styki suche		półprzewodnik - MOSFET	
Zakres napięć	5 do 30 VDC lub 5 do 250 VAC		20,4 do 28,8 VDC	
Sygnał logiczny 1 przy maks. prądzie	--		20 VDC min.	
Sygnał logiczny 0 przy obciążeniu 10 kΩ	--		0,1 VDC maks.	
Prąd (maks.)	2,0 A		0,5 A	
Obciążenie żarówką	30 W DC / 200 W AC		5 W	
Rezystancja styków w stanie ON	0,2 Ω maks. w stanie nowości		0,6 Ω maks.	
Prąd upływu na jeden punkt	--		10 µA maks.	
Udar prądowy	7 A z zamkniętymi stykami		8 A przez 100 ms maks.	
Zabezpieczenie przed przeciążeniem	brak			
Izolacja (sygnału zewnętrznego od logiki)	1500 VAC przez 1 minutę (cewka do styku) brak (cewka do logiki)		500 VAC przez 1 minutę	
Rezystancja izolacji	100 MΩ min. w stanie nowości		--	
Izolacja między otwartymi stykami	750 VAC przez 1 minutę		--	
Grupy izolacji	2	4	1	1
Prąd szyny wspólnej (maks.)	10 A		4 A	8 A
Ograniczanie przepięć indukcyjnych	--		L+ minus 48 VDC, 1 W mocy strat	

Dane techniczne				
Model	**SM 1222 DQ 8xPrzekaźnik**	**SM1222 DQ 16xPrzekaźnik**	**SM1222 DQ 8x24VDC**	**SM1222 DQ 16x24VDC**
Opóźnienie przełączania	10 ms maks.		50 µs maks. z OFF do ON 200 µs maks. z ON do OFF	
Trwałość mechaniczna (bez obciążenia)	10000000 cykli załącz/wyłącz		--	
Trwałość styków przy nominalnym obciążeniu	100000 cykli załącz/wyłącz		--	
Zachowanie przy przejściu z RUN do STOP	Ostatnia wartość lub wartość zastępcza (domyślnie 0)			
Liczba wyjść znajdujących się jednocześnie w stanie włączonym	8	16	8	16
Długość kabla (w metrach)	500 ekranowany, 150 nieekranowany			

Schematy połączeń

SM 1222 DQ 8 x Przekaźnik	SM 1222 DQ 8 x 24 VDC
	
6ES7 222-1HF30-0XB0	6ES7 222-1BF30-0XB0
SM 1222 DQ 16 x Przekaźnik	**SM 1222 DQ 16 x 24 VDC**
	
6ES7 222-1HH30-0XB0	6ES7 222-1BH30-0XB0

A.3.3 Dane techniczne modułu wejść/wyjść cyfrowych SM 1223

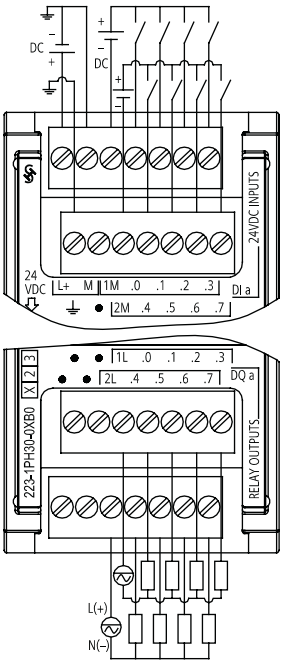
Dane techniczne				
Model	**SM 1223 DI 8x24 VDC, DQ 8xPrzekaźnik**	**SM 1223 DI 16x24 VDC, DQ 16xPrzekaźnik**	**SM 1223 DI 8x24 VDC, DQ 8x24 VDC**	**SM 1223 DI 16x24 VDC, DQ16x24 VDC**
Nr zamówieniowy (MLFB)	6ES7 223-1PH30-0XB0	6ES7 223-1PL30-0XB0	6ES7 223-1BH30-0XB0	6ES7 223-1BL30-0XB0
Wymiary W x H x D [mm]	45 x 100 x 75	70 x 100 x 75	45 x 100 x 75	70 x 100 x 75
Masa	230 g	350 g	210 g	310 g
Pobór mocy	5,5 W	10 W	2,5 W	4,5
Pobór prądu (magistrala SM)	145 mA	180 mA	145 mA	185 mA
Pobór prądu (24 VDC)	4 mA/wykorzystane wejście 11 mA /wykorzystaną cewkę przekaźnika		4 mA/wykorzystane wejście	
Wejścia cyfrowe				
Liczba wejść	8	16	8	16
Typ	prąd wpływający/wypływający (IEC Type 1 sink)			
Napięcie	24 VDC @ 4 mA, wartość nominalna			
Ciągłe dopuszczalne napięcie	30 VDC, maks.			
Udar napięciowy	35 VDC przez 0,5 s			
Sygnał logiczny 1 (min.)	15 VDC @ 2,5 mA			
Sygnał logiczny 0 (maks.)	5 VDC @ 1 mA			
Izolacja (od strony wyjściowej do logiki)	500 VAC przez 1 minutę			
Grupy izolacji	2	2	2	2
Czasy filtru	0,2, 0,4, 0,8, 1,6, 3,2, 6,4 i 12,8 ms (wybierane w grupach po 4)			
Liczba wejść znajdujących się jednocześnie w stanie włączonym	8	16	8	16
Długość kabla (w metrach)	500 ekranowany, 300 nieekranowany			
Wyjścia cyfrowe				
Liczba wyjść	8	16	8	16
Typ	przekaźnik, styki suche		półprzewodnik - MOSFET	
Zakres napięć	5 do 30 VDC lub 5 do 250 VAC		20,4 do 28,8 VDC	

Dane techniczne				
Model	**SM 1223 DI 8x24 VDC, DQ 8xPrzekaźnik**	**SM 1223 DI 16x24 VDC, DQ 16xPrzekaźnik**	**SM 1223 DI 8x24 VDC, DQ 8x24 VDC**	**SM 1223 DI 16x24 VDC, DQ16x24 VDC**
Sygnał logiczny 1 przy maks. prądzie	--		20 VDC min.	
Sygnał logiczny 0 przy obciążeniu 10 kΩ	--		0,1 VDC maks.	
Prąd (maks.)	2,0 A		0,5 A	
Obciążenie żarówką	30 W DC / 200 W AC		5 W	
Rezystancja styków w stanie ON	0,2 Ω maks. w stanie nowości		0,6 Ω maks.	
Prąd upływu na jeden punkt	--		10 µA maks.	
Udar prądowy	7 A z zamkniętymi stykami		8 A przez 100 ms maks.	
Zabezpieczenie przed przeciążeniem	brak			
Izolacja (sygnału zewnętrznego od logiki)	1500 VAC przez 1 minutę (cewka do styku) brak (cewka do logiki)		500 VAC przez 1 minutę	
Rezystancja izolacji	100 MΩ min. w stanie nowości		--	
Izolacja między otwartymi stykami	750 VAC przez 1 minutę		--	
Grupy izolacji	2	4	1	1
Prąd szyny wspólnej (maks.)	10 A	8 A	4 A	8 A
Ograniczanie przepięć indukcyjnych	--		L+ minus 48 VDC, 1 w mocy strat	
Opóźnienie przełączania	10 ms maks.		50 µs maks. z OFF do ON 200 µs maks. z ON do OFF	
Trwałość mechaniczna (bez obciążenia)	10000000 cykli załącz/wyłącz		--	
Trwałość styków przy nominalnym obciążeniu	100000 cykli załącz/wyłącz		--	
Zachowanie przy przejściu z RUN do STOP	Ostatnia wartość lub wartość zastępcza (domyślnie 0)			

Dane techniczne				
Model	**SM 1223 DI 8x24 VDC, DQ 8xPrzekaźnik**	**SM 1223 DI 16x24 VDC, DQ 16xPrzekaźnik**	**SM 1223 DI 8x24 VDC, DQ 8x24 VDC**	**SM 1223 DI 16x24 VDC, DQ16x24 VDC**
Liczba wyjść znajdujących się jednocześnie w stanie włączonym	8	16	8	16
Długość kabla (w metrach)	500 ekranowany, 150 nieekranowany			

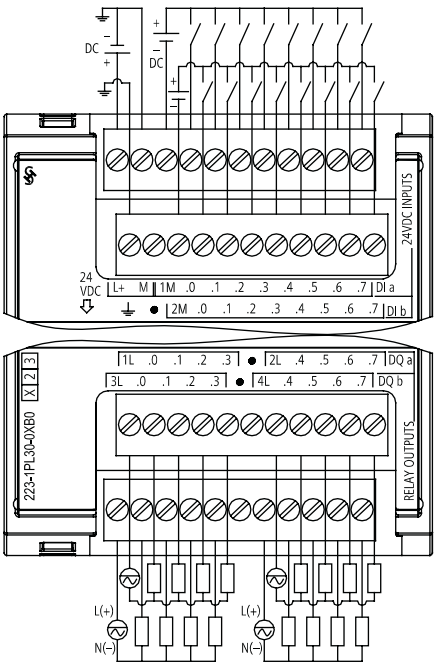
Schematy połączeń

SM 1223 DI 8 x 24 VDC, DQ 8 x Przekaźnik



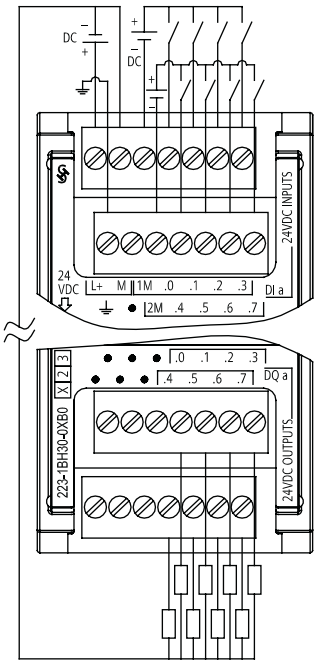
6ES7 223-1PH30-0XB0

SM1223 DI 16 x 24 VDC, DQ 16 x Przekaźnik



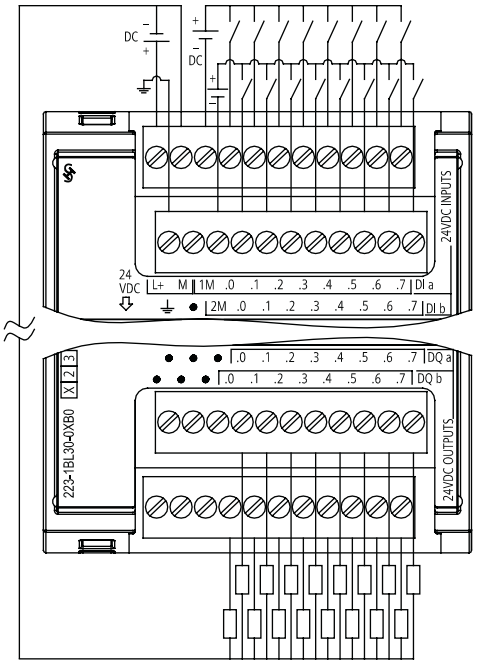
6ES7 223-1PL30-0XB0

SM 1223 DI 8 x 24 VDC, DQ 8 x 24 VDC



6ES7 223-1BH30-0XB0

SM 1223 DI 8 x 24 VDC, DQ 8 x 24 VDC



6ES7 223-1BL30-0XB0

A.4 Moduły rozszerzeń dla sygnałów analogowych

A.4.1 Dane techniczne modułów analogowych SM 1231, SM 1232, SM 1234

Dane techniczne			
Model	**SM 1231 AI 4x13 bitów**	**SM 1234 AI 4x13 bitów AQ 2x14 bitów**	**SM 1232 AQ 2x14 bitów**
Nr zamówieniowy (MLFB)	6ES7 231-4HD30-0XB0	6ES7 234-4HE30-0XB0	6ES7 232-4HB30-0XB0
Ogólne			
Wymiary W x H x D [mm]	45 x 100 x 75		
Masa	180 g	220 g	180 g
Pobór mocy	1,5 W	2,0 W	1,5 W
Pobór prądu (magistrala SM)	80 mA		
Pobór prądu (24 VDC)	45 mA	60 mA (bez obciążenia)	45 mA (bez obciążenia)

Dane techniczne			
Model	**SM 1231 AI 4x13 bitów**	**SM 1234 AI 4x13 bitów AQ 2x14 bitów**	**SM 1232 AQ 2x14 bitów**
Wejścia analogowe			
Liczba wejść	4		0
Typ	Napięcie lub prąd (wejście różnicowe)		--
Zakres	±10 V, ±5 V, ±2.5 V lub 0 do 20 mA		--
Zakres pomiarowy (słowo danych)	-27648 do 27648		--
Zakres przerzutu górnego i dolnego (słowo danych)	Napięcie: 32511 do 27649 / -27649 do -32512 Prąd: 32511 do 27649 / 0 do -4864 (por. Napięciowa reprezentacja wejścia analogowego, Prądowa reprezentacja wejścia analogowego)		--
Przepełnienie górne i dolne (słowo danych)	Napięcie: 32767 do 32512 / -32513 do -32768 Prąd: 32767 do 32512 / -4865 do -32768 (por. Napięciowa reprezentacja wejścia analogowego, Prądowa reprezentacja wejścia analogowego)		
Rozdzielczość	12 bitów + bit znaku		--
Maksymalne bezpieczne napięcie/ prąd	±35 V / ±40 mA		--
Wygładzanie	None (brak), Weak (słabe), Medium (średnie) lub Strong (mocne) (w celu uzyskania informacji o czasach odpowiedzi, por. Czas odpowiedzi wejścia analogowego)		--
Tłumienie zakłóceń	400, 60, 50 lub 10 Hz (w celu uzyskania informacji o częstości próbkowania, por. Czas odpowiedzi wejścia analogowego)		--
Impedancja	≥ 9 MΩ (we. napięciowe) / 250 Ω (we. prądowe)		--
Izolacja (sygnału zewnętrznego od logiki)	Brak		--
Dokładność (25°C / 0 do 55°C)	±0,1% / ±0,2% pełnego zakresu		--
Czas przetwarzania analogowo/cyfrowego	625 µs (dla tłumienia 400 Hz)		--
Tłumienie sygnału sumacyjnego	40 dB, DC do 60 Hz		--
Zakres operacyjny sygnału	sygnał plus napięcie sumacyjne musi być mniejsze niż +12 V i większe niż -12 V		--
Długość kabla (w metrach)	10 m, ekranowana para skręconych przewodów		--
Wyjścia analogowe			
Liczba wyjść	0	2	
Typ	--	Napięcie lub prąd	

Dane techniczne			
Model	**SM 1231 AI 4x13 bitów**	**SM 1234 AI 4x13 bitów AQ 2x14 bitów**	**SM 1232 AQ 2x14 bitów**
Zakres	--	±10 V lub 0 do 20 mA	
Rozdzielczość	--	Napięcie: 14 bitów Prąd: 13 bitów	
Zakres pomiarowy (słowo danych)	--	Napięcie: -27648 do 27648 Prąd: 0 do 27648 (por. Napięciowa reprezentacja wyjścia analogowego, Prądowa reprezentacja wyjścia analogowego)	
Dokładność (25°C / 0 do 55°C)	--	±0,3% / ±0,6% pełnego zakresu	
Czas ustalania (do 95 % nowej wartości)	--	Napięcie: 300 µs (R), 750 µs (1 µF) Prąd: 600 µs (1 mH), 2 ms (10 mH)	
Impedancja obciążenia	--	Napięcie: ≥ 1000 Ω Prąd: ≤ 600 Ω	
Zachowanie przy przejściu z RUN do STOP	--	Ostatnia wartość lub wartość zastępcza (domyślnie 0)	
Izolacja (od strony wyjściowej do logiki)	--	brak	
Długość kabla (w metrach)		10 m, ekranowana para skręconych przewodów	
Diagnostyka			
Przepełnienie górne i dolne	tak **UWAGA:** Jeżeli do wejścia jest przyłożone napięcie większe niż +30 VDC lub mniejsze niż –15 VDC, to jego wartość pozostanie nieznana, a odpowiadające mu przepełnienie górne lub dolne może pozostać nieaktywne.		
Zwarcie do uziemienia (tylko tryb napięciowy)	nie	tak, na wyjściach	Tak
Przerwa przewodu (tylko tryb prądowy)	nie	tak, na wyjściach	tak
Za niskie napięcie 24 VDC	tak		

Czas odpowiedzi wejścia analogowego

Odpowiedź skokowa modułów analogowych SM [ms]				
Skok od 0 V do 10 V, pomiar dla 95 % końcowej wartości				
Wybór wygładzania	**Tłumiona częstotliwość**			
	400 Hz	60 Hz	50 Hz	10 Hz
None (brak)	4	18	22	100
Weak (słabe)	9	52	63	320
Medium (średnie)	32	203	241	1200
Strong (mocne)	61	400	483	2410
Sample Rate (częstość próbek)	0,625	4,17	5	25

Odpowiedź skokowa wejścia analogowego CPU [ms]			
Skok od 0 V do 10 V, pomiar dla 95% końcowej wartości			
Wybór wygładzania	**Tłumiona częstotliwość**		
	60 Hz	**50 Hz**	**10 Hz**
None (brak)	63	65	130
Weak (słabe)	84	93	340
Medium (średnie)	221	258	1210
Strong (mocne)	424	499	2410
Sample Rate (częstość próbek)	4,17	5	25

Napięciowa reprezentacja wejścia analogowego

System	Zakres pomiarowy napięcia						
Dziesiętnie	**Heksadecymalnie**	**±10 V**	**±5 V**	**±2,5 V**	**0 do 10 V**		
32767	7FFF	11,851 V	5,926 V	2,963 V	Przepełnienie górne	11,851 V	Przepełnienie górne
32512	7F00						
32511	7EFF	11,759 V	5,879 V	2,940 V	Zakres przerzutu od góry	11,759 V	Zakres przerzutu od góry
27649	6C01						
27648	6C00	10 V	5 V	2,5 V	Zakres nominalny	10 V	Zakres nominalny
20736	5100	7,5 V	3,75 V	1,875 V		7,5 V	
1	1	361,7 µV	180,8 µV	90,4 µV		361,7 µV	
0	0	0 V	0 V	0 V		0 V	
-1	FFFF					Ujemne wartości nie są przetwarzane	
-20736	AF00	-7,5 V	-3,75 V	-1,875 V			
-27648	9400	-10 V	-5 V	-2,5 V			
-27649	93FF				Zakres przerzutu od dołu		
-32512	8100	-11,759 V	-5,879 V	-2,940 V			
-32513	80FF				Przepełnienie dolne		
-32768	8000	-11,851 V	-5,926 V	-2,963 V			

Prądowa reprezentacja wejścia analogowego

System	Zakres pomiarowy prądu		
Dziesiętnie	**Heksadecymalnie**	**0 mA do 20 mA**	
32767	7FFF	23,70 mA	Przepełnienie górne
32512	7F00		
32511	7EFF	23,52 mA	Zakres przerzutu od góry
27649	6C01		
27648	6C00	20 mA	Zakres nominalny
20736	5100	15 mA	
1	1	723,4 nA	
0	0	0 mA	
-1	FFFF		Zakres przerzutu od dołu
-4864	ED00	-3,52 mA	
-4865	ECFF		Przepełnienie dolne
-32768	8000		

Napięciowa reprezentacja wyjścia analogowego

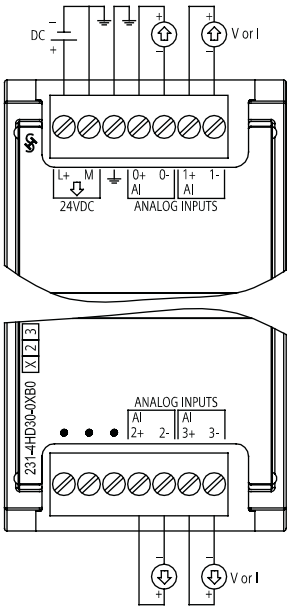
System		Zakres pomiarowy napięcia	
Dziesiętnie	**Heksadecymalnie**	**±10 V**	
32767	7FFF	0,00 V	Przepełnienie górne, poza napięciem zasilania
32512	7F00		
32511	7EFF	11,76 V	Zakres przerzutu od góry
27649	6C01		
27648	6C00	10 V	Zakres nominalny
20736	5100	7,5 V	
1	1	361,7 µV	
0	0	0 V	
-1	FFFF	-361,7 µV	
-20736	AF00	-7,5 V	
-27648	9400	-10 V	
-27649	93FF		Zakres przerzutu od dołu
-32512	8100	-11,76 V	
-32513	80FF		Przepełnienie dolne, poza napięciem zasilania
-32768	8000	0,00 V	

Prądowa reprezentacja wyjścia analogowego

System		Zakres pomiarowy prądu	
Dziesiętnie	**Heksadecymalnie**	**±20 mA**	
32767	7FFF	23,70 mA	Przepełnienie górne
32512	7F00		
32511	7EFF	23,52 mA	Zakres przerzutu od góry
27649	6C01		
27648	6C00	20 mA	Zakres nominalny
20736	5100	15 mA	
1	1	723,4 nA	
0	0	0 mA	
-1	FFFF		Zakres przerzutu od dołu
-32512	8100		
-32513	80FF		Przepełnienie dolne
-32768	8000		

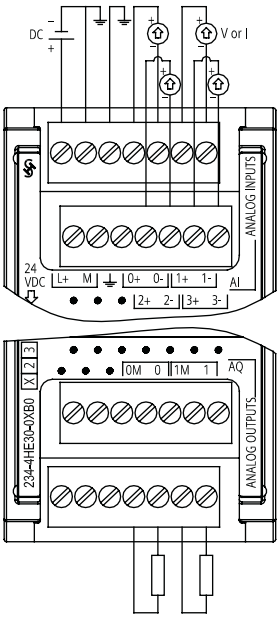
Schematy połączeń

SM 1231 AI x 13 bitów



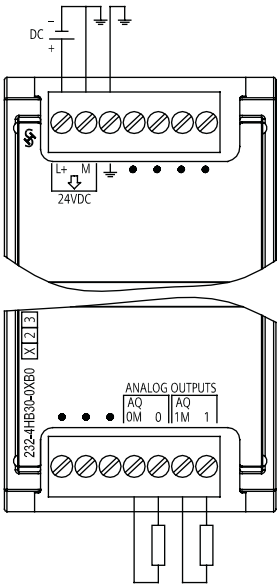
6ES7 231-4HD30-0XB0

SM 1234 AI 4 x 13 bitów



6ES7 234-4HE30-0XB0

SM 1232 AQ 2 x 14 bitów



6ES7 232-4HB30-0XB0

A.5 Płytki sygnałowe

A.5.1 Dane techniczne SB 1223: 2 × wejście 24 VDC /2 × wyjście 24 VDC

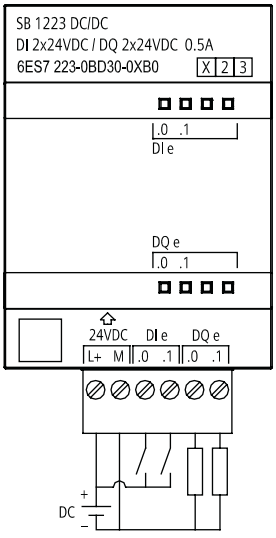
Dane techniczne cyfrowej płytki sygnałowej

Dane techniczne	
Model	**SB 1223 DI 2x24VDC, DQ 2x24VDC**
Nr zamówieniowy (MLFB)	6ES7 223-0BD30-0XB0
Ogólne	
Wymiary W x H x D [mm]	38 x 62 x 21
Masa	40 g
Pobór mocy	1,0 W
Pobór prądu (magistrala SM)	50 mA
Pobór prądu (24 VDC)	4 mA/wykorzystane wejście
Wejścia cyfrowe	
Liczba wejść	2
Typ	IEC Type 1 sink
Napięcie	24 VDC @ 4 mA, wartość nominalna
Ciągłe dopuszczalne napięcie	30 VDC, maks.
Udar napięciowy	35 VDC przez 0,5 s
Sygnał logiczny 1 (min.)	15 VDC @ 2,5 mA
Sygnał logiczny 0 (maks.)	5 VDC @ 1 mA
Szybkość zegara HSC (maks.)	20 kHz (15 do 30 VDC) 30 kHz (15 do 26 VDC)
Izolacja (od strony wyjściowej do logiki)	500 VAC przez 1 minutę
Grupy izolacji	1
Czasy filtru	0,2, 0,4, 0,8, 1,6, 3,2, 6,4 i 12,8 ms (wybierane w grupach po 2)
Liczba wejść znajdujących się jednocześnie w stanie włączonym	2
Długość kabla (w metrach)	500 ekranowany, 300 nieekranowany
Wyjścia cyfrowe	
Liczba wyjść	2
Typ	półprzewodnik - MOSFET
Zakres napięć	20,4 do 28,8 VDC
Sygnał logiczny 1 przy maks. prądzie	20 VDC min.
Sygnał logiczny 0 przy obciążeniu 10 kΩ	0,1 VDC maks.
Prąd (maks.)	0,5 A
Obciążenie żarówką	5 W
Rezystancja w stanie ON	0,6 Ω maks.

Dane techniczne	
Model	**SB 1223 DI 2x24VDC, DQ 2x24VDC**
Prąd upływu na jeden punkt	10 µA maks.
Częstotliwość impulsów wyjściowych	20 kHz maks.
Udar prądowy	5 A przez 100 ms maks.
Zabezpieczenie przed przeciążeniem	brak
Izolacja (sygnału zewnętrznego od logiki)	500 VAC przez 1 minutę
Grupy izolacji	1
Prąd szyny wspólnej	1 A
Ograniczanie przepięć indukcyjnych	L+ minus 48 VDC, 1 W mocy strat
Opóźnienie przełączania	2,0 µs maks. z OFF do ON 10 µs maks. z ON do OFF
Zachowanie przy przejściu z RUN do STOP	Ostatnia wartość lub wartość zastępcza (domyślnie 0)
Liczba wyjść znajdujących się jednocześnie w stanie włączonym	2
Długość kabla (w metrach)	500 ekranowany, 150 nieekranowany

Schematy połączeń

SB 1223: 2 × wejście 24 VDC /2 × wyjście 24 VDC



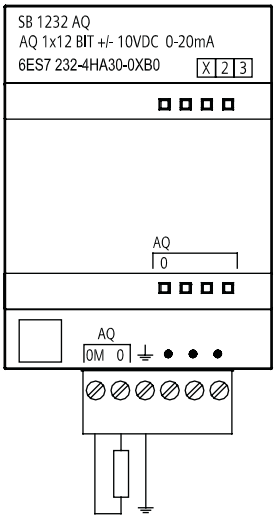
A.5.2 Dane techniczne SB 1232: 1 wyjście analogowe

Dane techniczne analogowej płytki sygnałowej

Dane techniczne	
Model	**SB 1223 AQ 1 x 12bitów**
Nr zamówieniowy (MLFB)	6ES7 232-4HA30-0XB0
Ogólne	
Wymiary W x H x D [mm]	38 x 62 x 21
Masa	40 g
Pobór mocy	1,5 W
Pobór prądu (magistrala SM)	15 mA
Pobór prądu (24 VDC)	25 mA (bez obciążenia)
Wyjścia analogowe	
Liczba wyjść	1
Typ	Napięcie lub prąd
Zakres	±10 V lub 0 do 20 mA
Rozdzielczość	Napięcie: 12 bitów Prąd: 11 bitów
Zakres pomiarowy (słowo danych)	Napięcie: -27648 do 27648 Prąd: 0 do 27648
Dokładność (25°C / 0 do 55°C)	±0,5% / ±1% pełnego zakresu
Czas ustalania (do 95 % nowej wartości)	Napięcie: 300 µs (R), 750 µs (1 µF)
	Prąd: 600 µs (1 mH), 2 ms (10 mH)
Impedancja obciążenia	Napięcie: ≥ 1000 Ω Prąd: ≤ 600 Ω
Zachowanie przy przejściu z RUN do STOP	Ostatnia wartość lub wartość zastępcza (domyślnie 0)
Izolacja (od strony wyjściowej do logiki)	brak
Długość kabla (w metrach)	10 m, ekranowana para skręconych przewodów
Diagnostyka	
Przepełnienie górne i dolne	tak
Zwarcie do uziemienia (tylko tryb napięciowy)	tak
Przerwa przewodu (tylko tryb prądowy)	tak

Schematy połączeń

SB 1232: 1 wyjście analogowe



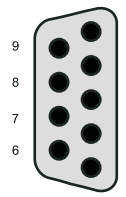
A.6 Moduły komunikacyjne (CM)

A.6.1 Dane techniczne CM 1241 RS485

Tabela A-1 Moduł komunikacyjny CM 1241 RS485

Dane techniczne	
Nr zamówieniowy (MLFB)	6ES7 241-1CH30-0XB0
Wymiary i masa	
Wymiary W x H x D [mm]	30 x 100 x 75
Masa	150 g
Nadajnik i odbiornik	
Zakres sygnału współbieżnego	-7 V do +12 V przez 1 sekundę 3 VRMS (nap. skutecznego) w sposób ciągły
Wyjściowe napięcie różnicowe nadajnika	2 V min. @ RL = 100 Ω 1,5 V min. @t RL = 54 Ω
Obciążenie i zasilanie	10 kΩ do +5 V na B, PROFIBUS końcówka 3 10K Ω do GND na A, PROFIBUS końcówka 8
Impedancja wejściowa odbiornika	5,4 kΩ min. łącznie z obciążeniem
Próg/czułość odbiornika	+/- 0,2 V min., typowa histereza 60 mV

Dane techniczne	
Izolacja Sygnał RS485 do masy obudowy Sygnał RS485 do masy logiki CPU	500 VAC, przez 1 minutę
Długość kabla ekranowanego	1000 m maks.
Zasilanie	
Pobór mocy	1,1 W
Pobór prądu z +5 VDC	220 mA

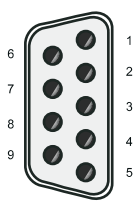
Końcówka	Opis	Złącze	Końcówka	Opis
1 GND	Masa logiki lub sygnałów komunikacyjnych		6 PWR	+5 V z szeregowym rezystorem 100 Ω: Wyjście
2	Nie podłączona		7	Nie podłączona
3 TxD+	Sygnał B (RxD/TxD+): Wejście/Wyjście		8 TXD-	Sygnał A (RxD/TxD-): Wejście/Wyjście
4 RTS	Żądanie wysłania (poziom TTL): Wyjście		9	Nie podłączona
5 GND	Masa logiki lub sygnałów komunikacyjnych		SHELL	Masa obudowy

A.6.2 Dane techniczne CM 1241 RS232

Moduł komunikacyjny CM 1241 RS232

Dane techniczne	
Nr zamówieniowy (MLFB)	6ES7 241-1AH30-0XB0
Wymiary i masa	
Wymiary W x H x D [mm]	30 x 100 x 75
Masa	150 g
Nadajnik i odbiornik	
Napięcie wyjściowe nadajnika	+/- 5 V min. @ RL = 3 kΩ
Napięcie wyjściowe nadawania	+/- 15 VDC maks.
Impedancja wejściowa odbiornika	3 kΩ min.
Próg/czułość odbiornika	0,8 V min. niski, 2,4 maks. wysoki Typowa histereza 05 V
Napięcie wejściowe odbiornika	+/- 30VDC maks.
Izolacja Sygnał RS232 do masy obudowy Sygnał RS232 do masy logiki CPU	500 VAC, przez 1 minutę
Długość kabla ekranowanego	10 m maks.

Zasilanie	
Pobór mocy	1,1 W
Pobór prądu z +5 VDC	220 mA

Końcówka	Opis	Złącze	Końcówka	Opis
1 DCD	Wykryty sygnał nośnej: Wejście		6 DSR	Dane gotowe do wysłania: Wejście
2 RxD	Dane otrzymywane z DCE: Wejście		7 RTS	Żądanie wysłania: Wyjście
3 TxD	Dane wysyłane do DCE: Wyjście		8 CTS	Gotowość do wysłania: Wejście
4 DTR	Gotowość terminala danych: Wyjście		9 RI	Sygnał dzwonka (nie używany)
5 GND	Masa sygnałowa		SHELL	Masa obudowy

A.7 Karty pamięci SIMATIC

Dane techniczne kart pamięci

Numer zamówieniowy	Pojemność
6ES7 954-8LF00-0AA0	24 MB
6ES7 954-8LB00-0AA0	2 MB

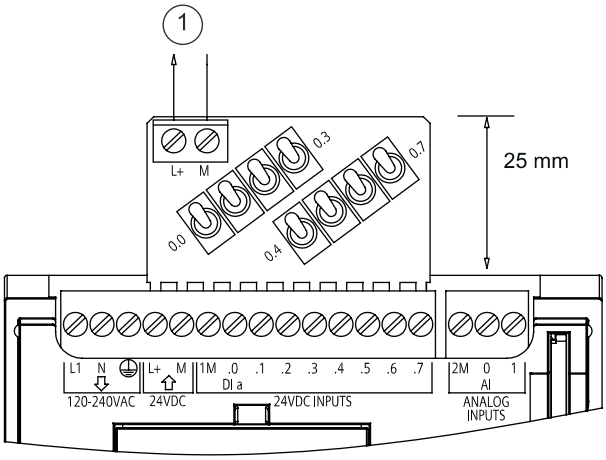
A.8 Symulatory wejść

Model	Symulator 8 pozycyjny	Symulator 14 pozycyjny
Numer zamówieniowy (MLFB)	6ES7 274-1XF30-0XA0	6ES7 274-1XH30-0XA0
Wymiary W x H x D [mm]	43 x 35 x 23	67 x 35 x 23
Masa	20 g	30 g
Liczba punktów	8	14
Stosowany z CPU	CPU 1211C, CPU 1212C	CPU 1214C

OSTRZEŻENIE

Te symulatory wejść nie mają zatwierdzenia Class I DIV 2 lub Class I Zone 2 do pracy w miejscach niebezpiecznych. Przełączniki stwarzają zagrożenie wystąpienia iskry / powstania wybuchu jeśli będą zastosowane w obszarach określonych przez Class I DIV 2 lub Class I Zone 2.

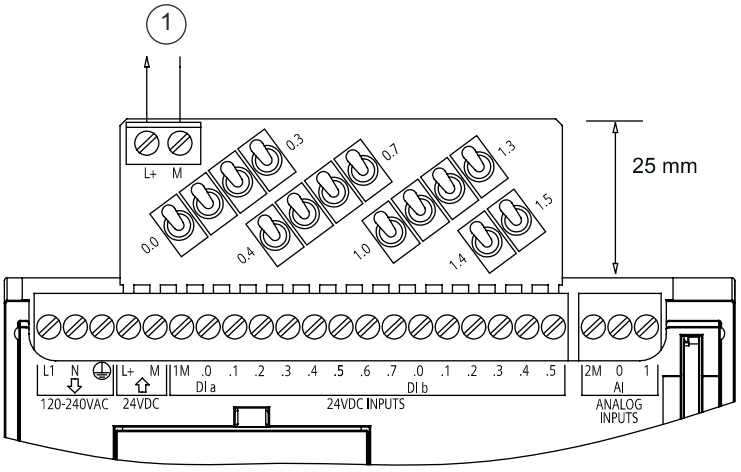
Symulator 8 pozycyjny



6ES7 274-1XF30-0XA0

① wyjście 24 VDC do zasilania czujników

Symulator 14 pozycyjny



6ES7 274-1XH30-0XA0

① wyjście 24 VDC do zasilania czujników

Bilans mocy

B

CPU jest wyposażony wewnętrzny zasilacz, który zasila CPU, moduły rozszerzeń oraz może dostarczać napięcie 24 VDC do innych urządzeń zgodnie z wymaganiami użytkownika.

Występują trzy typu modułów rozszerzeń:

- Moduły rozszerzeń (SM) są instalowane z prawej strony CPU. Każda CPU pozwala na dołączenie pewnej maksymalnej liczby modułów sygnałowych bez uwzględniania bilansu mocy
  - CPU 1214 można dołączyć 8 modułów sygnałowych
  - CPU 1212 można dołączyć 2 moduły sygnałowe
  - CPU 1211 nie można dołączyć modułów sygnałowych
- Moduły komunikacyjne (CM) są instalowane z lewej strony CPU. Można dołączyć maksymalnie 3 moduły komunikacyjne do dowolnej CPU, bez uwzględniania bilansu mocy.
- Płytki sygnałowe (SB) są instalowane od góry CPU. Można dołączyć maksymalnie 1 płytkę sygnałową do dowolnego CPU.

Poniżej podane informacje stanowią przewodnik pozwalający określić ile mocy (lub prądu) CPU może dostarczyć do urządzeń użytkownika.

Każda CPU dostarcza napięcia zarówno 5 VDC, jak i 24 VDC :

- CPU zasila napięciem 5 VDC podłączone moduły rozszerzeń. Jeżeli moduły rozszerzeń wymagają ze źródła 5 VDC mocy przekraczającej wydajność zasilacza CPU, to należy usunąć moduły rozszerzeń, aż pobierana moc mieści się w bilansie mocy.
- Każda CPU jest wyposażona w zasilacz czujników 24 VDC dla zasilania lokalnych punktów wejściowych lub cewek przekaźników modułów rozszerzeń. Jeżeli wymagania na moc pobieraną z napięcia 24 VDC przekraczają wydajność zasilacza CPU, to do zasilania modułów rozszerzeń można wykorzystać zewnętrzny zasilacz o napięciu 24 VDC. W takim przypadku należy osobno wykonać połączenia zewnętrznego zasilacza 24 VDC z punktami wejściowymi lub cewkami przekaźników.

OSTRZEŻENIE

Połączenie równoległe zewnętrznego zasilacza 24 VDC z wewnętrznym zasilaczem 24 VDC sterownika może doprowadzić do konfliktu między tymi zasilaczami, ponieważ każdy zasilacz będzie usiłował stabilizować swoje własne napięcie wyjściowe.

Taki konflikt wpłynie niekorzystnie na żywotność pracy zasilaczy lub doprowadzi do natychmiastowego uszkodzenia jednego lub obu zasilaczy, co może spowodować nieprzewidywalne zachowanie PLC w układzie. Nieprzewidywalne zachowanie PLC grozi śmiercią lub poważnym zranieniem personelu i/lub zniszczeniem mienia.

Zasilacz wewnętrzny S7-1200 i dowolny zasilacz zewnętrzny powinny zasilać różne obwody systemu. Dozwolone jest połączenie mas obu zasilaczy.

Niektóre końcówki zasilania 24 V systemu PLC są ze sobą połączone za pomocą masy układów logicznych zwierających wiele wyprowadzeń M. Końcówka zasilająca CPU 24 VDC, wejście zasilania cewek przekaźników oraz nieizolowane wejście zasilające układów analogowych stanowią przykład układów połączonych ze sobą, w przypadku gdy karty katalogowe określają je jako nieizolowane. Wszystkie wyprowadzenia M układów nieizolowanych muszą być podłączone do tego samego, zewnętrznego potencjału odniesienia.

⚠	**OSTRZEŻENIE**
Połączenie nieizolowanych wyprowadzeń M do różnych potencjałów odniesienia spowoduje nieplanowany przepływ prądów mogący doprowadzić do uszkodzenia lub nieprzewidywalnego zachowania PLC i podłączonych do niego urządzeń. Takie uszkodzenia i nieprzewidywalne działanie, grozi śmiercią lub poważnym zranieniem personelu i/lub zniszczeniem mienia. Koniecznie należy się upewnić, że nieizolowane zaciski M systemu S7-1200 są podłączone do tego samego potencjału odniesienia.	

Informacje dotyczące bilansu mocy CPU i poboru mocy przez moduły rozszerzeń są podane w części „Dane techniczne”.

UWAGA

Przekroczenie bilansu mocy CPU może być przyczyną redukcji maksymalnej liczby możliwych do podłączenia do CPU modułów.

B.1 Przykład obliczeniowy bilansu mocy

W poniższym przykładzie przedstawiono obliczenie bilansu mocy w przypadku systemu PLC zawierającego CPU 1214C AC/DC/Przekaźnik, 3 x SM 1223 8 DC In/8 Relay Out [mk6]i 1 x SM 1221 8 DC In. System w tym przykładzie ma łącznie 46 wejść i 34 wyjścia.

UWAGA

CPU już przydzieliła moc wymaganą do zasilania cewek wewnętrznych przekaźników. Użytkownik nie musi włączać mocy pobieranej przez wewnętrzne przekaźniki do obliczeń bilansu mocy.

W tym przykładzie CPU ma wystarczającą wydajność prądową zasilacza 5 VDC do zasilania SM, ale wydajność prądowa napięcia 24 VDC zasilacza czujników jest niewystarczająca do zasilania wszystkich wejść i dodatkowych przekaźników. I/O wymagają 448 mA, a CPU może dostarczyć tylko 400 mA. Aby wszystkie podłączone wejścia i wyjścia mogły poprawnie pracować, w tym systemie jest konieczny dodatkowy zasilacz 24 VDC o wydajności co najmniej 48 mA.

Bilans mocy CPU	**5 VDC**	**24 VDC**
CPU 1214C AC/DC/Przekaźnik	1600 mA	400 mA
Minus		
Wymagania systemu	**5 VDC**	**24 VDC**
CPU 1214C, 14 wejść	-	14 * 4 mA = 56 mA
3 SM 1223, zasilanie 5 V	3 * 145 mA = 435 mA	-
1 SM 1221, zasilanie 5 V	1 * 105 mA = 105 mA	-
3 SM 1223, 8 wejść każdy	-	3 * 8 * 4 mA = 96 mA
3 SM 1223, 8 przekaźników każdy	-	3 * 8 * 11 mA = 264 mA
1 SM 1221, 8 wejść	-	8 * 4 mA = 32 mA
Łącznie	540 mA	448 mA
Równa się		
Bilans prądu	**5 VDC**	**24 VDC**
Łącznie	1060 mA	(48 mA)

B.2 Bilans mocy systemu użytkownika

Do obliczenia bilansu mocy dowolnego systemu S7-1200 CPU można posłużyć się poniższą tabelą. Informacje dotyczące poboru mocy CPU i poboru mocy modułów rozszerzeń podano w części „Dane techniczne”.

Bilans mocy CPU	**5 VDC**	**24 VDC**
Minus		
Wymagania systemu	5 VDC	24 VDC
Łącznie		
Równa się		
Bilans prądu	5 VDC	24 VDC
Łącznie		

Numery zamówieniowe

C

CPUs		Numer zamówieniowy
CPU 1211C	CPU 1211C DC/DC/DC	6ES7 211-1AD30-0XB0
	CPU 1211C AC/DC/Relay	6ES7 211-1BD30-0XB0
	CPU 1211C DC/DC/Relay	6ES7 211-1HD30-0XB0
CPU 1212C	CPU 1212C DC/DC/DC	6ES7 212-1AD30-0XB0
	CPU 1212C AC/DC/Relay	6ES7 212-1BD30-0XB0
	CPU 1212C DC/DC/Relay	6ES7 212-1HD30-0XB0
CPU 1214C	CPU 1214C DC/DC/DC	6ES7 214-1AE30-0XB0
	CPU 1214C AC/DC/Relay	6ES7 214-1BE30-0XB0
	CPU 1214C DC/DC/Relay	6ES7 214-1HE30-0XB0

Moduły rozszerzeń, moduły komunikacyjne i płytki sygnałowe		Numer zamówieniowy
Moduły rozszerzeń	SM 1221 8 x 24 VDC Input	6ES7 221-1BF30-0XB0
	SM 1221 16 x 24 VDC Input	6ES7 221-1BH30-0XB0
	SM 1222 8 x 24 VDC Output	6ES7 222-1BF30-0XB0
	SM 1222 16 x 24 VDC Output	6ES7 222-1BH30-0XB0
	SM 1222 8 x Relay Output	6ES7 222-1HF30-0XB0
	SM 1222 16 x Relay Output	6ES7 222-1HH30-0XB0
	SM 1223 8 x 24 VDC Input / 8 x 24 VDC Output	6ES7 223-1BH30-0XB0
	SM 1223 16 x 24 VDC Input / 16 x 24 VDC Output	6ES7 223-1BL30-0XB0
	SM 1223 8 x 24 VDC Input / 8 x Relay Output	6ES7 223-1PH30-0XB0
	SM 1223 16 x 24 VDC Input / 16 x Relay Output	6ES7 223-1PL30-0XB0
	SM 1231 4 x Analog Input	6ES7 231-4HD30-0XB0
	SM 1232 2 x Analog Input	6ES7 232-4HB30-0XB0
	SM 1234 4 x Analog Input / 2 x Analog Output	6ES7 234-4HE30-0XB0
Moduły komunikacyjne	CM 1241 RS232	6ES7 241-1AH30-0XB0
	CM 1241 RS485	6ES7 241-1CH30-0XB0
Płytki sygnałowe	SB 1223 2 x 24 VDC Input / 2 x 24 VDC Output	6ES7 223-0BD30-0XB0
	SB 1232 1 Analog Output	6ES7 232-4HA30-0XB0

Panele operatorskie HMI	Numer zamówieniowy
KTP400 Basic (Mono, PN)	6AV6 647-0AA11-3AX0
KTP600 Basic (Mono, PN)	6AV6 647-0AB11-3AX0
KTP600 Basic (Color, PN)	6AV6 647-0AD11-3AX0
KTP1000 Basic (Color, PN)	6AV6 647-0AF11-3AX0
TP1500 Basic (Color, PN)	6AV6 647-0AG11-3AX0

Oprogramowanie narzędziowe	
Pakiet oprogramowania	**Numer zamówieniowy**
STEP 7 Basic v10.5	6ES7 822-0AA0-0YA0

Karty pamięci, pozostałe urządzenia i części zamienne		Numer zamówieniowy
Karty pamięci	SIMATIC MC 2 MB	6ES7 954-8LB00-0AA0
	SIMATIC MC 24 MB	6ES7 954-8LF00-0AA0
Pozostałe urządzenia	PSU 1200 power supply	6EP1 332-1SH71
	CSM 1277 Ethernet switch - 4 ports	6GK7 277-1AA00-0AA0
	Symulator wejść (1214C/1211C - 8-pozycyjny)	6ES7 274-1XF30-0XA0
	Symulator wejść (1214C – 14-pozycyjny)	6ES7 274-1XH30-0XA0
Części zamienne	Listwa zaciskowa, 7-stykowa, srebrzona	6ES7 292-1AG30-0XA0
	Listwa zaciskowa, 8-stykowa, srebrzona	6ES7 292-1AH30-0XA0
	Listwa zaciskowa, 11-stykowa, srebrzona	6ES7 292-1AL30-0XA0
	Listwa zaciskowa, 12-stykowa, srebrzona	6ES7 292-1AM30-0XA0
	Listwa zaciskowa, 14-stykowa, srebrzona	6ES7 292-1AP30-0XA0
	Listwa zaciskowa, 20-stykowa, srebrzona	6ES7 292-1AV30-0XA0
	Listwa zaciskowa, 3-stykowa, złocona	6ES7 292-1BC0-0XA0
	Listwa zaciskowa, 6-stykowa, złocona	6ES7 292-1BF30-0XA0
	Listwa zaciskowa, 7-stykowa, złocona	6ES7 292-1BG30-0XA0
	Listwa zaciskowa, 11-stykowa, złocona	6ES7 292-1BL30-0XA0

Dokumentacja	Język	Numer zamówieniowy
S7-1200 Programmable Controller System Manual	Niemiecki	6ES7 298-8FA30-8AH0
	Angielski	6ES7 298-8FA30-8BH0
	Francuski	6ES7 298-8FA30-8CH0
	Hiszpański	6ES7 298-8FA30-8DH0
	Włoski	6ES7 298-8FA30-8EH0
	Chiński	6ES7 298-8FA30-8FH0
S7-1200 Easy Book	Niemiecki	6ES7 298-8FA30-8AQ0
	Angielski	6ES7 298-8FA30-8BQ0
	Francuski	6ES7 298-8FA30-8CQ0
	Hiszpański	6ES7 298-8FA30-8DQ0
	Włoski	6ES7 298-8FA30-8EQ0
	Chiński	6ES7 298-8FA30-8FQ0