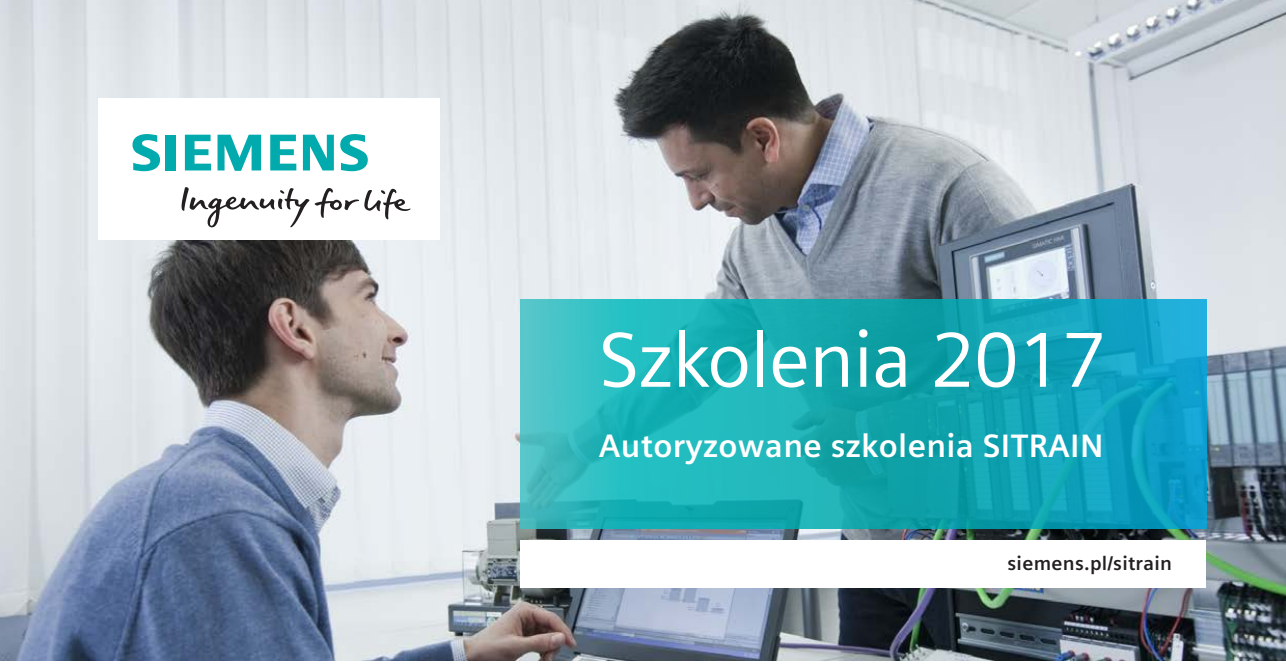
SIEMENS
Ingenuity for life

Egzemplarz bezpłatny

Przewodnik programowania dla S7-1200/S7-1500

STEP 7 Professional oraz STEP 7 Safety w TIA Portal
Opis systemu – wydanie 2/2017

siemens.pl/s7-1500



SIEMENS
Ingenuity for life

Szkolenia 2017

Autoryzowane szkolenia SITRAIN

siemens.pl/sitrain

Nazwa szkolenia	Kod szkolenia	Miejsce szkolenia	Dni	STY	LUT	MAR	KWI	MAJ	CZE	LIP	SIE	WRZ	PAŹ	LIS	GRU
SIMATIC S7 + SIMATIC Manager															
Programowanie SIMATIC S7-300/400 – podstawowy	ST-PRO1	ŁÓDŹ	5	16-20		13-17			26-30				9-13		11-15
Programowanie SIMATIC S7-300/400 – zaawansowany	ST-PRO2	ŁÓDŹ	5				3-7					11-15			
Serwisowanie SIMATIC S7-300/400 – podstawowy	ST-SERV1	ŁÓDŹ	5		6-10									27-1	
Serwisowanie SIMATIC S7-300/400 – zaawansowany	ST-SERV2	ŁÓDŹ	5					15-19							18-22
SIMATIC S7 + TIA Portal															
Migracja systemów automatyki do SIMATIC S7-1500	TIA-SYSUP	ŁÓDŹ	4-5			28-31	24-27							13-17	
Programowanie SIMATIC S7-1500 – podstawowy	TIA-PRO1	ŁÓDŹ	5	9-13		6-10		22-26		3-7		18-22		20-24	
Programowanie SIMATIC S7-1200 Failsafe	TIA-S7F MICRO	ŁÓDŹ	2		13-14		10-11					4-5		6-7	4-5
Programowanie SIMATIC S7-1200 – podstawowy	TIA-MICRO1	ŁÓDŹ	3	25-27	15-17		19-21	10-12	7-9			6-8	4-6		6-8
Programowanie sterowników S7-1500 w SCL	TIA-SCL	ŁÓDŹ		Wyślij zapytanie **szkolenia.pl@siemens.com**											
Programowanie sterowników S7-1500 w GRAPH	TIA-GRAPH	ŁÓDŹ		Wyślij zapytanie **szkolenia.pl@siemens.com**											
DISTRIBUTED SAFETY															
Programowanie sterowników SIMATIC S7 Failsafe	ST-PPDS	ŁÓDŹ		Wyślij zapytanie **szkolenia.pl@siemens.com**											
SIMATIC NET															
Konfiguracja i diagnostyka sieci PROFIBUS	IK-PBSYS	ŁÓDŹ	3			27-29									
Konfiguracja i diagnostyka sieci PROFINET	IK-PNSYS	ŁÓDŹ	3										25-27		
SIMATIC HMI															
Programowanie paneli HMI w WinCC flexible	ST-WINCC	KATOWICE	3		15-17							27-29			
Systemy wizualizacji SCADA WinCC v7.x	ST-BWINCC	KATOWICE	5		27-3								16-20		

Dokładny opis wszystkich szkoleń dostępny pod adresem **www.siemens.pl/sitrain**

Gwarancja i zrzeczenie się odpowiedzialności

Ważne

Przykładowe aplikacje opisane w tym podręczniku nie są obowiązujące i mogą być niekompletne pod względem konfiguracji, wyposażenia jak również innych opcji. Informacje te stanowią jedynie przykład standardowego zastosowania omawianych rozwiązań i nie należy ich stosować w przypadkach niestandardowych. Odpowiedzialność za poprawną eksploatację opisanych produktów leży po Państwa stronie. Przykłady opisane poniżej nie zwalniają Państwa z obowiązku bezpiecznego obchodzenia się ze sprzętem podczas instalacji, eksploatacji i konserwacji. Korzystając z przykładów zamieszczonych w tym podręczniku uznają Państwo, że firma Siemens nie może być pociągnięta do odpowiedzialności za ewentualne szkody wykraczające poza powyższą regulację. Zastrzegamy sobie prawo do wprowadzenia zmian do zamieszczonych przykładów bez uprzedzenia. W przypadku różnic pomiędzy rozwiązaniami zawartymi w poniższych przykładach a innymi publikacjami np. katalogami, należy postępować zgodnie z informacjami zamieszczonymi w innych publikacjach.

Zrzekamy się odpowiedzialności z tytułu informacji zawartych w niniejszym podręczniku.

Nasza odpowiedzialność, niezależnie od podstawy prawnej za szkody spowodowane korzystaniem z opisanych niżej przykładów, informacji, programów, danych projektowych itp. jest wykluczona, o ile nie występuje przypadek odpowiedzialności przymusowej np. zgodnie z ustawą o odpowiedzialności za produkty w przypadkach działania umyślnego, rażącej niedbałości, uszkodzenia życia lub zdrowia, albo z powodu przejęcia gwarancji za cechy i stan jakiegoś produktu, zatajenia jego wady lub naruszenia istotnych postanowień wynikających z umowy. Odszkodowanie z tytułu naruszenia kluczowych zobowiązań wynikających z umowy ograniczają się jednak do typowych dla umów, przewidywalnych szkód, o ile nie dotyczą odpowiedzialności przymusowej z tytułu działania umyślnego, rażącej niedbałości, lub uszkodzenia życia lub zdrowia. Nie wiąże się to ze zmianą ciężaru dowodowego na Państwa niekorzyść.

Wskazówki dotyczące bezpieczeństwa

Produkty firmy Siemens są opracowywane z myślą o bezpiecznym użytkowaniu w środowisku przemysłowym. Poszczególne elementy np. urządzenia, sieci oraz inne oferowane przez nas rozwiązania stanowią część całościowego systemu bezpieczeństwa przemysłowego. Mając to na uwadze, staramy się ciągle ulepszać nasze produkty. Warto regularnie odwiedzać naszą stronę internetową i być na bieżąco z najnowszymi aktualizacjami.

Aby zagwarantować bezpieczne użytkowanie naszych produktów należy podjąć odpowiednie środki ochronne i zintegrować wszystkie urządzenia we wspólnym systemie bezpieczeństwa przemysłowego. Dotyczy to również produktów innych producentów.

Więcej informacji można znaleźć na stornie internetowej: http://www.siemens.com/industrialsecurity.

Jeżeli chcesz być na bieżąco z najnowszymi aktualizacjami, zapisz się do newslettera produktu, który Cię interesuje. Więcej informacji znajdziesz na stronie http://support.automation.siemens.com.

Spis treści

1 Wstęp

Nowa generacja sterowników SIMATIC oferuje następujące rozwiązania:

- Wspólna platforma dla wszystkich elementów automatyki (sterowników, urządzeń HMI, itp.)
- Jednakowy sposób programowania
- Zwiększona wydajność
- Zestaw poleceń w każdym języku programowania
- Programowanie symboliczne
- Przetwarzanie danych nawet bez użycia wskaźnika
- Możliwość wielokrotnego wykorzystania utworzonych bloków

Podręcznik programowania

Nowa generacja sterowników SIMATIC (S7-1200 oraz S7-1500) charakteryzuje się nowoczesną budową systemu, która w połączeniu z platformą TIA Portal, oferuje nowe oraz wydajne funkcje programistyczne.

Podręcznik ten zawiera przydane informacje oraz zalecenia, które pomogą optymalnie zaprogramować sterowniki S7-1200/1500. Wszystkie nowości oraz różnice w budowie systemów S7-300/400 przedstawione są w przejrzysty sposób, co ułatwia napisanie ustandaryzowanego programu dla różnych rozwiązań z zakresu automatyki.

Przedstawione przykłady można zastosować dla wszystkich sterowników z rodziny S7-1200 oraz S7-1500.

Najważniejsze informacje

Podręcznik porusza następujące kwestie dotyczące środowiska projektowego TIA Portal:

- Nowości w S7-1200/1500
  - Języki programowania
  - Bloki zoptymalizowane
  - Typy danych oraz polecenia
- Wskazówki dla programistów
  - System operacyjny i program użytkownika
  - Pamięć
  - Adresowanie symboliczne
  - Biblioteki
- Wskazówki dot. programowania niezależnego od sprzętu (hardware-independent programming)
- Wskazówki dot. STEP 7 Safety (TIA Portal)
- Przegląd najważniejszych punktów

Zalety i korzyści

Wskazówki opisane w tym mogą przełożyć się na liczne korzyści, m.in:

- Wydajny program użytkownika
- Przejrzystą budowę systemu
- Intuicyjne oraz skuteczne programowanie

2 Nowości w S7-1200/1500

2.1 Wprowadzenie

Programowanie sterowników SIMATIC, co do zasady, nie zmieniło się wiele od wersji S7-300/400. Programiści mogą korzystać z popularnych języków programowania takich jak LAD, FBD, STL, SCL oraz bloków organizacyjnych (OB), funkcyjnych (FB) funkcji (FC) i bloków danych (DB). Pełna kompatybilność gwarantuje, że programy napisane dla S7-300/400 mogą być wykorzystane dla S7-1500. Podobnie w przypadku programów napisanych w językach LAD, FBD oraz SCL, które są kompatybilne ze sterownikami S7-1200.

Dodatkowo wprowadzono wiele nowości, które ułatwiły programowanie, umożliwiając napisanie wydajnego kodu, który nie wymaga dużo pamięci.

Zachęcamy nie tylko do korzystania z programów opracowanych dla sterowników S7-1200/1500 1: 1 ale również do odkrywania nowych funkcjonalności. Niewielkim nakładem pracy można stworzyć kod, który jest:

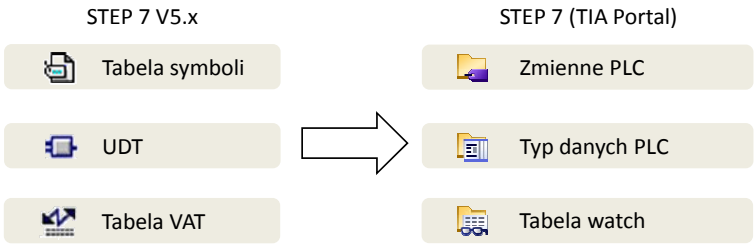
- zoptymalizowany pod kątem wymaganej pamięci oraz czasu wykonania programu na nowszych CPU
- przejrzysty i klarowny
- łatwy w obsłudze

2.2 Główne pojęcia

Główne pojęcia w TIA Portal

Niektóre pojęcia uległy zmianie, aby ułatwić korzystanie z TIA Portal.

Rysunek 2-1: Nowe pojęcia w TIA Portal



Pojęcia związane ze zmiennymi oraz parametrami

Pojęcia związane ze zmiennymi, blokami funkcyjnymi oraz funkcjami wielokrotnie używane są niepoprawnie, oraz w sposób niespójny. Poniższa tabela pozwoli ujednolicić tą terminologię.

Rysunek 2-2: Pojęcia związane ze zmiennymi oraz parametrami

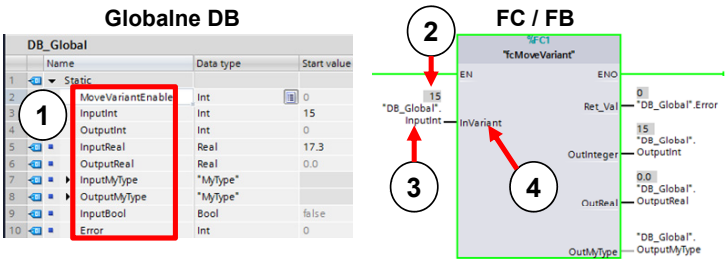


Tabela 2-1: Pojęcia związane ze zmiennymi oraz parametrami

	Pojęcie	Opis
1.	Zmienne	Zmienne to obszary pamięci sterownika zarezerwowane dla określonej wartości. Określane za pomocą rodzaju danych np. Bool, Integer, itp.: • Zmienne PLC (PLC tags) • Pojedyncze zmienne w blokach danych • Pełne bloki danych
2.	Wartości zmiennych	Są to wartości przechowywane w danej zmiennej (np. 15 jako wartość zmiennej Integer).
3.	Parametr aktualny	Parametry aktualne to zmienne powiązane ze sobą na poziomie interfejsów poleceń, funkcji oraz bloków funkcyjnych.
4.	Parametr formalny (parametr transferu, parametr blokowy)	Parametry formalne to parametry interfejsów dla poleceń, funkcji oraz bloków funkcyjnych (Input, Output, InOut, Temp, Static, oraz Return).

Wskazówka

Więcej informacji można znaleźć w poniższych pozycjach:

Jaka dokumentacja porusza temat migracji projektów do STEP 7 (TIA Portal) oraz WinCC (TIA Portal)?
http://support.automation.siemens.com/WW/view/en/58879602

Jakie warunki należy spełnić, aby migrować projekt z STEP 7 V5.x do STEP 7 Professional, (TIA Portal)
http://support.automation.siemens.com/WW/view/en/62101406

Migracja zmiennych PLC do sterowników S7-1500 z wykorzystaniem STEP 7 (TIA Portal) http://support.automation.siemens.com/WW/view/en/67858106

Wskazówki dot. zaprogramowania S7-1200 oraz S7-1500 za pomocą STEP 7 (TIA Portal)

http://support.automation.siemens.com/WW/view/en/67582299

Dlaczego nie można łączyć funkcji: przenoszenie wartości rejestru (register passing) oraz przenoszenie parametrów jawnych (explicit parameter transfer) z S7-1500 w STEP 7 (TIA Portal)?
Wpis ten zawiera m.in. informacje na temat migracji programów STL do S7-1500
http://support.automation.siemens.com/WW/view/en/67655405

2.3 Języki programowania

Pisząc program użytkownika, można skorzystać z wielu języków programowania. Każdy z nich posiada unikalne zalety, co umożliwia wybór najbardziej optymalnego języka programowania dla danej aplikacji.

Każdy bloki programu użytkownika można zaprogramować w dowolnym języku programowania, w zależności do funkcji, jaką będą miały pełnić.

Tabela 2-2: Języki programowania

Język programowania	S7-1200	S7-1500
Ladder (LAD)	✓	✓
Function block diagram (FBD)	✓	✓
Structured control language (SCL)	✓	✓
Graph	×	✓
Statement list (STL)	×	✓

Wskazówka

Więcej informacji można znaleźć w poniższych pozycjach:

Porównanie języków programowania dla SIMATIC S7-1200 / S7-1500
http://support.automation.siemens.com/WW/view/en/86630375

Na co należy zwrócić uwagę podczas migracji programu S7-SCL do STEP 7 (TIA Portal)
http://support.automation.siemens.com/WW/view/en/59784006

Jakie polecenia są niedostępne dla programu SCL w STEP 7 (TIA Portal)
http://support.automation.siemens.com/WW/view/en/58002710

Jak definiować stałe w programie S7-SCL w STEP 7 (TIA Portal)
http://support.automation.siemens.com/WW/view/en/58065411

2.4 Optymalizacja kodu maszynowego

TIA Portal oraz S7-1200/1500 gwarantują optymalne wykonanie programu niezależnie od języka programowania. Wszystkie języki programowania zostają skompilowane do tego samego kodu maszynowego.

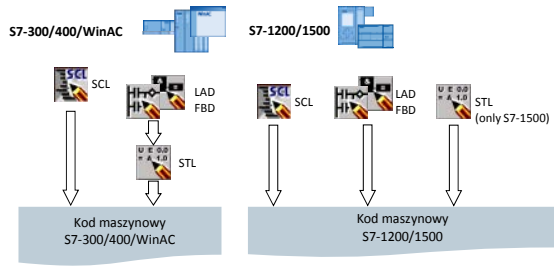
Zalety

- Wysoka wydajność, niezależnie od języku programowania (przy tym samym typie dostępu)
- Brak spadku wydajności w wyniku dodatkowej kompilacji (krok pośredni w STL)

Właściwości

Poniższy rysunek przedstawia różnice w kompilacji programów S7 do kodu maszynowego.

Rysunek 2-3: Generowanie kodu maszynowego w S7-300/400/WinAC oraz S7-1200/1500

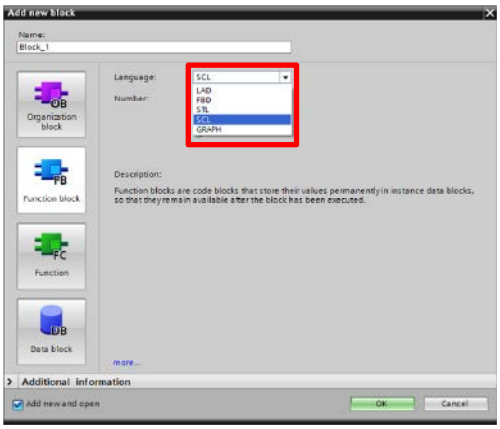


- W przypadku sterowników S7-300/400/WinAC, programy napisane w językach LAD oraz FBD są wpierw kompilowane na język STL. Dopiero potem następuje wygenerowanie kodu maszynowego.
- W przypadku sterowników S7-1200/1500 wszystkie języki programowana są bezpośrednio komplikowane na kod maszynowy

2.5 Tworzenie nowych bloków

Wszystkie bloki (OB, FB oraz FC) można programować w różnych językach - każdy posiada osobny kompilator. Podczas tworzenia nowego bloku można wybrać dowolny język programowania. Po stworzeniu bloku możliwa jest jedynie zamiana LAD<->FBD

Rysunek 2-4: Okno dodawania nowych bloków „Add new block”



2.6 Bloki zoptymalizowane

Bloki zoptymalizowane pozwalają usprawnić przechowywanie oraz wywoływanie danych. Wszystkie zmienne są automatycznie sortowane według typu danych. Dane tego samego typu przechowywane są razem, dzięki czemu procesor ma do nich optymalny dostęp, a dane jednego typu przechowywane są razem.

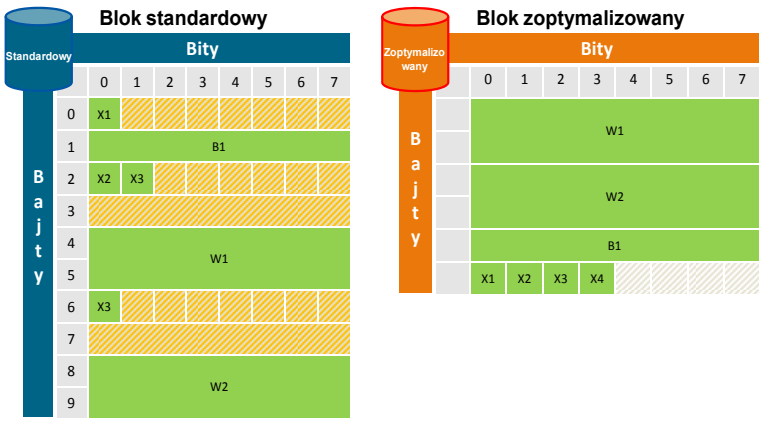
Bloki niezoptymalizowane istnieją wyłącznie, aby zagwarantować maksymalną kompatybilność sterowników S7-1200/1500.

Zalety

- Szybki dostęp do danych, dzięki systemowemu sortowaniu plików, niezależnie od deklaracji.
- Wyeliminowanie błędów wywołania związanych z adresowaniem absolutnym dzięki dostępowi symbolicznemu.
- Wyeliminowanie błędów wywołania związanych ze zmianą deklaracji, ponieważ dostęp do np. urządzeń HMI jest symboliczny.
- Każdą zmienną z osobna można definiować jako „retain”.
- Bloki danych typu „instance”, nie wymagają osobnej konfiguracji, ponieważ przejmują ustawienia skojarzonych bloków funkcyjnych (FB), takie jak np. ustawienia, „Retentivity”
- Obszar dodatkowy pamięci dla bloku pozwala rozszerzyć blok DB bez utraty wartości aktualnych (sprawdź w rozdziale 3.2.7 Wgrywanie bloków bez utraty wartości aktualnych)

2.6.1 Budowa bloków zoptymalizowanych sterownika S7-1200

Rysunek 2-5: Blok zoptymalizowany sterownika S7-1200

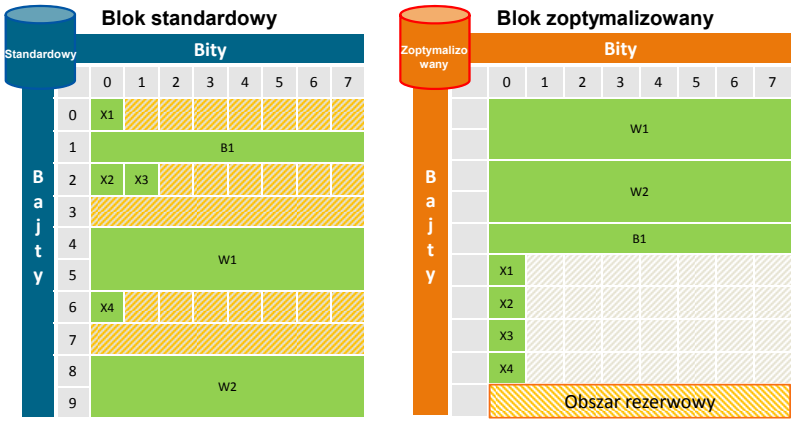


Właściwości

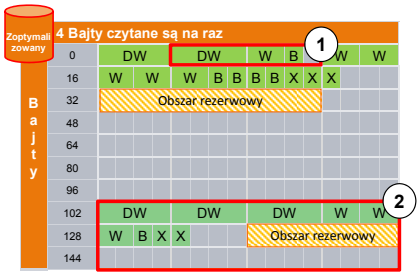
- Blok jest lepiej zorganizowany. Większe zmienne znajdują się na początku bloku, a mniejsze na końcu. Nie ma również odstępów między poszczególnymi zmiennymi.
- Dostęp do bloków zoptymalizowanych jest wyłącznie symboliczny.

2.6.2 Budowa bloku zoptymalizowanego sterownika S7-1500

Rysunek 2-6: Blok zoptymalizowany sterownika S7-1500



Rysunek 2-7: Organizacja pamięci w blokach zoptymalizowanych



1. Struktury są przechowywane jako całość, dzięki czemu można je skopiować, jako jeden blok.
2. Dane podtrzymywane (retentive) przechowywane są w osobnym sektorze i również mogą być skopiowane jako jeden blok. W przypadku awarii zasilania, dane te są przechowywane wewnątrz CPU. Polecenie „MRES” resetuje te dane do wartości startowych, przechowywanych w pamięci „load memory”.

Właściwości

- Blok jest uporządkowany. Większe zmienne znajdują się na początku bloku, a mniejsze na końcu. Nie ma również odstępów między poszczególnymi zmiennymi.
- Szybki dostęp do danych przechowywanych w procesorze (odczyt/zapis), możliwy jest za pomocą jednego polecenia.
- Zmienne Boolean przechowywane są jako jeden bajt. Dzięki temu sterownik nie musi maskować wywoływania poszczególnych bitów.
- Bloki zoptymalizowane posiadają specjalne obszary dodatkowe pozwalające rozszerzyć blok DB bez utraty wartości aktualnych (sprawdź w rozdziale 3.2.7 Wgrywanie bloków bez utraty wartości aktualnych).
- Dostęp do bloków zoptymalizowanych jest wyłącznie symboliczny.

2.6.3 Najlepsza forma zapisu danych w procesorze sterowników S7-1500

Sterowniki S7-300/400 korzystają z formy zapisu w sekwencji „Big Endian”, aby zagwarantować kompatybilność z pierwszymi sterownikami SIMATIC.

W związku z nową architekturą procesora, nowa generacja sterowników S7-1500 wykorzystuje formę zapisu na 4 bajtach (32 bitach) w sekwencji „Little-Endian”.

Rysunek 2-8: Wywoływanie danych przez sterownik S7-1500

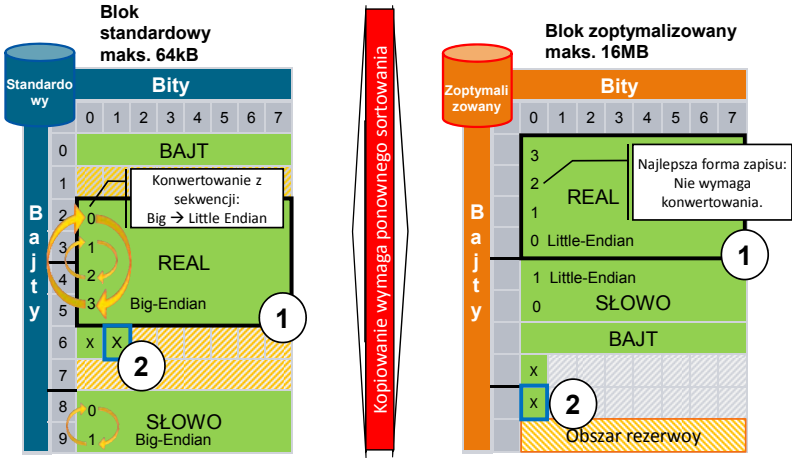


Tabela 2-3: Wywoływanie danych przez sterownik S7-1500

	Blok standardowy	Blok zoptymalizowany
1.	W przypadku niekorzystnego przesunięcia, sterownik potrzebuje 2x16 bitów, aby uzyskać dostęp do wartości zapisanej na 4 bajtach (np. wartości rzeczywistej. Dodatkowo, bajty muszą zostać zamienione.	Zmienne przechowywane są w sterowniku, który potrzebuje 32 bitów (REAL), aby je odczytać. Zamiana bajtów nie jest konieczna.
2.	Cały bajt czytany jest bit po bicie, a następnie maskowany. W tym czasie cały bajt jest blokowany dla jakiejkolwiek próby dostępu.	Każdy bit jest przypisany do danego bajtu. Podczas odczytu, sterownik nie musi maskować bajtu.
3.	Maksymalny rozmiar bloku to 64 kB.	Maksymalny rozmiar bloku to 16 MB.

Zalecenia

- Zawsze korzystaj z bloków zoptymalizowanych.
  - Nie wymagają one adresowania absolutnego i mogą zostać wywołane w sposób symboliczny. Zapis symboliczny ułatwia umożliwia ponadto adresowanie pośrednie (sprawdź rozdział 3.6.2 Dane typu ARRAY oraz pośrednie wywołanie obszaru).
  - Przetwarzanie bloków zoptymalizowanych przebiega dużo szybciej niż w przypadku bloków standardowych.
- Z uwagi na długi czas konwertowania, unikaj przenoszenia/przypisywania danych pomiędzy blokami zoptymalizowanymi a niezoptymalizowanymi.

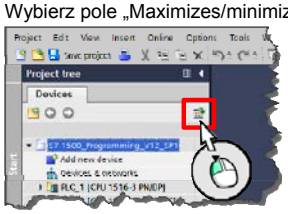
Przykład: Ustawianie zoptymalizowanego dostępu do bloków

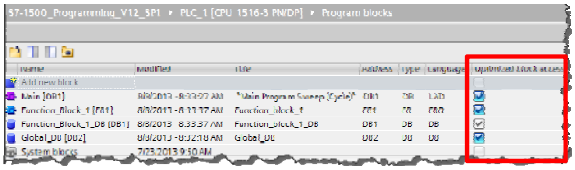
Wszystkie nowo utworzone bloki S7-1200/1500 mają domyślnie ustawiony dostęp zoptymalizowany „optimized block access”. Ustawienie to można oczywiście zmienić dla bloków organizacyjnych (OB), funkcyjnych (FB) oraz globalnych bloków danych (DB.

Podczas przenoszenia bloków pomiędzy sterownikami S7-300/400 a S7-1200/1500 należy pamiętać, że ustawienia dostępu nie są resetowane automatycznie. Można je zmienić po przeniesieniu danego bloku. Aby zmiany weszły w życie, należy przeprowadzić rekompilację programu. W przypadku bloków danych (DB) typu „instance” wystarczy zmienić ustawienia dostępu dla skojarzonych bloków funkcyjnych (FB).

Aby ustawić dostęp do bloków zoptymalizowanych, postępuj zgodnie z poniższą instrukcją.

Tabela 2-4: Ustawianie dostępu do bloków zoptymalizowanych

Krok	Polecenie
1.	Wybierz pole „Maximizes/minimizes the Overview” w oknie nawigacji. 
2.	Następnie przejdź do „Program blocks“.

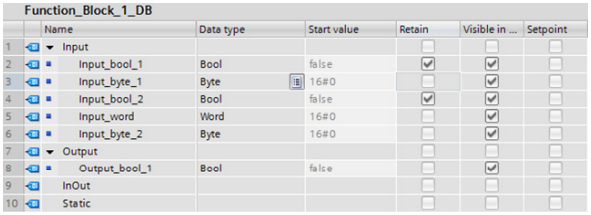
Krok	Polecenie
3.	Wyświetlone zostaną wszystkie bloki programu, wraz z informacją czy są zoptymalizowane, czy też nie. Zaznacz odpowiednie pole, aby ustawić zoptymalizowany dostęp do bloków.  Uwaga: W przypadku bloków danych instance wystarczy zmienić ustawienia skojarzonych bloków funkcyjnych (w tym przykładzie „Function_block_1_DB"). Dotyczy to również ustawienia dla zoptymalizowanego dostępu. Zmiany zostaną wprowadzone po przeprowadzeniu rekompilacji projektu.

Bloki zoptymalizowane i niezoptymalizowane w TIA Portal

Poniższe rysunki przedstawiają różnice pomiędzy blokiem zoptymalizowanym oraz niezoptymalizowanym.

Różnice w przypadku bloków globalnych są takie same.

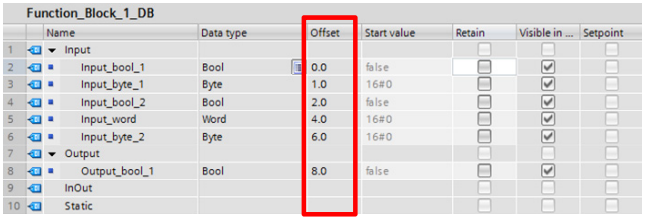
Rysunek 2-9: Blok zoptymalizowany (bez przesunięcia)



Function_Block_1_DB

	Name	Data type	Start value	Retain	Visible in ...	Setpoint
1	Input					
2	Input_bool_1	Bool	false	☑	☑	
3	Input_byte_1	Byte	16#0		☑	
4	Input_bool_2	Bool	false	☑	☑	
5	Input_word	Word	16#0		☑	
6	Input_byte_2	Byte	16#0		☑	
7	Output					
8	Output_bool_1	Bool	false		☑	
9	InOut					
10	Static					

Rysunek 2-10: Blok niezoptymalizowane (z przesunięciem - offset)



Function_Block_1_DB

	Name	Data type	Offset	Start value	Retain	Visible in ...	Setpoint
1	Input						
2	Input_bool_1	Bool	0.0	false		☑	
3	Input_byte_1	Byte	1.0	16#0		☑	
4	Input_bool_2	Bool	2.0	false		☑	
5	Input_word	Word	4.0	16#0		☑	
6	Input_byte_2	Byte	6.0	16#0		☑	
7	Output						
8	Output_bool_1	Bool	8.0	false		☑	
9	InOut						
10	Static						

Tabela 2-5: Różnice pomiędzy blokiem zoptymalizowanym a niezoptymalizowanym

Blok zoptymalizowany	Blok niezoptymalizowany
Bloki zoptymalizowane adresowane są symbolicznie. Parametr przesunięcia „offset" nie jest wyświetlany.	Parametr przesunięcia „offset" jest wyświetlany. Można go wykorzystać podczas adresowania bloku.
Każdą zmienną można zadeklarować jako zmienną typu „Retain".	Deklaracja „Retain" może dotyczyć wyłącznie wszystkich lub żadnej zmiennej.

W przypadku globalnych bloków danych (DB), parametr „retentivity” dla zmiennych określany jest bezpośrednio dla danego bloku. Ustawienie domyślne to „non-retentive”.

W przypadku instancji, parametr „retentivity” dla zmiennych tej instancji określany jest w bloku funkcyjnym (FB), a nie w bloku danych instance (DB).

Zmiana ustawień w bloku funkcyjnym dotyczy również wszystkich instancji tego bloku.

Typy dostępu do bloków zoptymalizowanych i niezoptymalizowanych

Poniższa tabela przedstawia typy dostępu do bloków.

Tabela 2-6: Typy dostępu

Typ dostępu	Blok zoptymalizowany	Blok niezoptymalizowany
Symboliczny	✓	✓
Indeksowy (pola)	✓	✓
Dostęp poprzez „slice Access”	✓	✓
Polecenie AT	× (użyj “slice access”)	✓
Absolutny bezpośredni	× (użyj ARRAY z indeksem)	✓
Absolutny pośredni (wskaźnikowy)	× (użyj VARIANT / ARRAY z indeksem)	✓
Wgrywanie bloków bez utraty wartości aktualnych	✓	×

Wskazówka

Więcej informacji można znaleźć w poniższych pozycjach:

Na jakie różnice warto zwrócić uwagę w przypadku dostępu zoptymalizowanego lub standardowego w STEP 7 (TIA Portal) ?

http://support.automation.siemens.com/WW/view/en/67655611

Na jakie własności należy zwrócić uwagę w przypadku poleceń READ_DBL” oraz „WRIT_DBL”, podczas pracy z blokami danych (DB) z dostępem zoptymalizowanym ?
http://support.automation.siemens.com/WW/view/en/51434748

2.6.4 Konwertowanie zmiennych

Najlepiej korzystać wyłącznie ze zmiennych zoptymalizowanych. Można skorzystać ze starszych struktur programowych, aczkolwiek prowadzi to do wymieszania się zmiennych zoptymalizowanych i niezoptymalizowanych w programie.

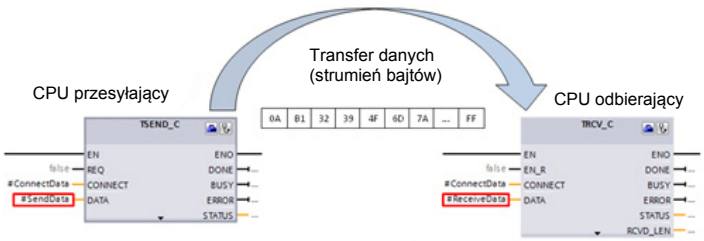
System rozpoznaje zawartość każdej zmiennej niezależnie od tego czy jest ona strukturą (lub typem danych użytkownika), czy podstawowa (INT, LREAL, itp.).

W przypadku operacji na zmiennych tego samego typu, zapisanych w różnych obszarach pamięci, system dokonuje automatycznej konwersji. Proces ten nie jest zalecany dla zmiennych typu Struct, ponieważ obniża wydajność.

2.6.5 Wysyłanie i odbieranie danych zoptymalizowanych

Interfejs (CPU, CM) przesyła dane zgodnie z ich bieżącym ułożeniem (nie ważne czy zostały zoptymalizowane, czy nie).

Rysunek 2-11: Wymiana danych CPU-CPU



Dane wysyłane mogą być:
- zoptymalizowane
- niezoptymalizowane
- Zmienną(dowolnego rodzaju)
- Buforem (tablica binarna)

Dane otrzymywane mogą być:
- zoptymalizowane
- niezoptymalizowane
- Zmienną(dowolnego rodzaju)
- Buforem (tablica binarna)

Przykład

- Zmienna zawierająca dane PLC (rekord danych) ma zostać przesłana do CPU.
- Jest ona najpierw podawana jako parametr aktualny na wejście bloku komunikacyjnego (TSEND_C) w CPU przesyłającym.
- Następnie dane przesłane do CPU odbierającego zostają zapisane w zmiennej tego samego typu.
- Można kontynuować pracę z danymi, wywołując je symbolicznie.

Wskazówka Wszystkie zmienne oraz bloki danych (pochodzące od danych typu PLC) mogą pełnić rolę rekordów danych

Wskazówka Możliwe jest przesyłanie i odbieranie danych, które są odmiennie zdefiniowane.

Dane wysłane		**Dane odebrane**
zoptymalizowane	-->	niezoptymalizowane
niezoptymalizowane	-->	zoptymalizowane

Sterownik czuwa nad poprawnym przesłaniem i zapisaniem danych.

2.7 Właściwości bloków

2.7.1 Rozmiary bloków

W przypadku sterowników S7-1200 oraz S7-1500 znacznie zwiększono dopuszczalny rozmiar bloków w pamięci głównej.

Tabela 2-7: Rozmiar bloków

Maks. rozmiar i ilość (rozmiar pamięci jest bez znaczenia)		**S7 -300/400**	**S7-1200**	**S7-1500**
DB	Maks. rozmiar	64 kB	64 kB	64 kB (niezoptymalizowane) **10 MB** (zoptymalizowane CPU1518)
	Maks. ilość	16.000	65.535	65.535
FC/FB	Maks. rozmiar	64 kB	64 kB	512 kB **3 MB** (zoptymalizowane CPU1518)
	Maks. ilość	7.999	65.535	65.535
FC / FB / DB	Maks. ilość	4.096 (CPU319) 6.000 (CPU412)	1.024	10.000 (CPU1518)

Zalecenia

- Bloki danych (DB) sprawdzają się jako kontenery do przechowywania dużych ilości danych (S7-1500)
- W przypadku wolumenów przekraczających 64 kB można skorzystać ze zoptymalizowanych bloków danych (DB), o maksymalnej pojemności 16 MB (S7-1500)

2.7.2 Ilość bloków organizacyjnych (OB)

Bloki organizacyjne różnego typu (OB) mogą posłużyć do stworzenia struktury hierarchicznej programu użytkownika.

Tabela 2-8: Ilość bloków organizacyjnych

Typ bloku organizacyjnego	**S7-1200**	**S7-1500**	**Korzyść**
Cykliczne i rozruchowe OB	100	100	Modularyzacja program użytkownika
Przerwania sprzętowego	50	50	Możliwy oddzielny blok OB dla każdego zdarzenia
Przerwania z opóźnieniem czasowym	4*	20	Modularyzacja programu użytkownika
Przerwania cyklicznego		20	Modularyzacja programu użytkownika
Czas dnia	×	20	Modularyzacja programu użytkownika

* Od wersji firmware V4.0 dla S7-1200 możliwe są 4 przerwania z opóźnieniem czasowym oraz 4 przerwania watchdog.

Zalecenia

- Bloki organizacyjne (OB) pozwalają stworzyć program użytkownika o strukturze hierarchicznej.
- Więcej wskazówek odnośnie bloków organizacyjnych (OB) można znaleźć w rozdziale 3.2.1 Bloki organizacyjne (OB).

2.8 Nowe typy danych dla S7-1200/1500

Programowanie sterowników S7-1200/1500 jest o wiele wygodniejsze z racji obsługi nowych, 64 bitowych typów danych. Umożliwiają one wykorzystanie znacznie większych oraz dokładniejszych wartości.

Wskazówka Więcej informacji można znaleźć w poniższej pozycji:

Jak konwertować typy danych w TIA Portal ?
http://support.automation.siemens.com/WW/view/en/60546567

2.8.1 Podstawowe typy danych

Tabela 2-9: Dane typu integer

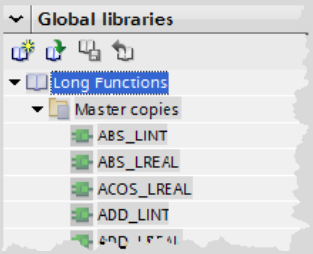
Typ	Rozmiar	Zakres wartości
USint	8 bit	0 .. 255
SInt	8 bit	-128 .. 127
UInt	16 bit	0 .. 65535
UDInt	32 bit	0 .. 4.3 miliona
ULInt*	64 bit	0 .. 18,4 tryliona (10^{18})
LInt*	64 bit	-9,2 tryliona.. 9,2 tryliona
LWord	64 bit	16#0000 0000 0000 0000 do 16# FFFF FFFF FFFF FFFF

* wyłącznie dla S7-1500

Tabela 2-10: Dane dziesiętne, zmiennoprzecinkowe

Typ	Rozmiar	Zakres wartości
Real	32 bity (1 bitowy znak, 8 bit wykładnik, 23 bitowa mantysa), z dokładnością do 7 miejsca po przecinku	-3.40e+38 .. 3.40e+38
LReal	64 bity (1 bitowy znak, 11 bitowy wykładnik, 52 bitowa mantysa), z dokładnością do 15 miejsca po przecinku	-1.79e+308 .. 1.79e+308

Wskazówka TIA Portal posiada globalną bibliotekę pod nazwą „Long Functions", która zawiera szereg poleceń dla długich liczb całkowitych.



Wskazówka Więcej informacji można znaleźć w poniższej pozycji:

Dlaczego wynik DInt Addition w SCL wyświetlany jest niepoprawnie w STEP 7 (TIA Portal)
http://support.automation.siemens.com/WW/view/en/98278626

2.8.2 Dane typu Date_Time_Long

Tabela 2-11: Budowa DTL (Date_Time_Long)

Rok	Miesiąc	Dzień	Dzień tygodnia	Godzina	Minuta	Sekunda	Nanosekunda

DTL zawsze odczytuje bieżący czas systemowy. Poszczególne wartości można wywołać za pomocą nazw symbolicznych (np. `My_Timestamp.Hour`)

Zalety

- Poszczególne pola (np. Rok, Miesiąc etc.) można wywołać w sposób symboliczny.

Zalecenia

Korzystaj z danych typu DTL zamiast LDT oraz wywołuj je w sposób symboliczny (np. `My_Timestamp.Hour`).

Wskazówka Więcej informacji można znaleźć w poniższych pozycjach:

Jak wprowadzać, odczytywać i edytować ustawienia czasu i godziny dla modułów CPU sterowników S7-300/S7-400/S7-1200/S7-1500 w STEP 7 (TIA Portal)?
http://support.automation.siemens.com/WW/view/en/58387452

Które funkcję w STEP 7 V5.5 oraz TIA Portal są dostępne do przetwarzania danych typu DT oraz DTL?.
http://support.automation.siemens.com/WW/view/en/63900230

2.8.3 Typy danych do określenia czasu

Tabela 2-12: Typy danych do określenia czasu (wyłącznie dla S7-1500)

Typ	Rozmiar	Zakres wartości
LTime	64 bity	LT#-106751d23h47m16s854ms775us808ns do LT#+106751d23h47m16s854ms775us807ns
LTIME_OF_DAY	64 bity	LTOD#00:00:00.000000000 do LTOD#23:59:59.999999999

2.8.4 Dane Unicode

Dane typu WCHAR oraz WSTRING mogą być przetwarzane za pomocą znaków unicode.

Tabela 2-13: Typy danych do określania czasu (wyłącznie dla S7-1500)

Typ	Rozmiar	Zakres wartości
WCHAR	2 bajty	-
WSTRING	(4 + 2*n) bajt	Obecna wartość: 0 ..254 znaków Maks. wartość: 0 ..16382

n = długość łańcucha znaków

Właściwości

- Przetwarzanie znaków np. w językach Łacińskim, Chińskim i innych.
- Podział wierszy, przesunięcie strony, tabulatory, znak spacji.
- Znaki specjalne: Symbol dolara, znaki zapytania.

Przykład

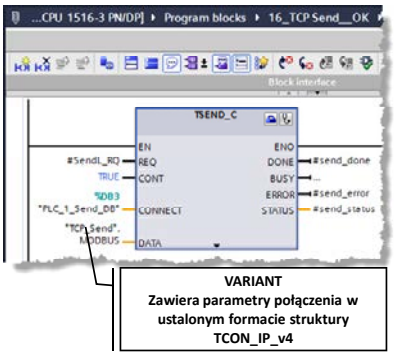
- `WCHAR#'a'`
- `WSTRING#'Hello World!'`

2.8.5 Dane typu VARIANT (wyłącznie dla S7-1500)

Parametr VARIANT może być wskaźnikiem dla zmiennych różnego typu. W przeciwieństwie do wskaźnika ANY, VARIANT przeprowadza test zgodności podczas przejścia programu. Struktura źródłowa i docelowa muszą być identyczne.

VARIANT jest wykorzystywany np. dla parametrów wejściowych bloków komunikacyjnych (TSEND_C).

Rysunek 2-12: Dane VARIANT jako parametry wejściowe polecenia TSEND_C



Zalety

- Test zgodności typu danych chroni przed błędami dostępu.
- Symboliczne adresowanie zmiennych typu VARIANT ułatwia czytanie kodu.
- Można wydajniej i szybciej tworzyć kod.
- Wskaźniki VARIANT są bardziej intuicyjne od wskaźników ANY.
- Zmienne typu VARIANT można wykorzystać bezpośrednio za pomocą funkcji systemowych.
- Elastyczne przenoszenie tablic o różnych strukturach

Właściwości

Poniższa tabela przedstawia porównanie własności wskaźników Variant oraz ANY

Tabela 2-14: Porównanie wskaźników Variant oraz ANY

ANY	Variant
Wymaga 10 Kb pamięci o zdefiniowanej strukturze.	Nie wymaga osobnej pamięci dla użytkownika.
Inicjalizacja poprzez przypisanie obszaru danych lub wypełnienie struktury ANY.	Inicjalizacja poprzez przypisanie obszaru danych z poleceniami systemowymi.
Nie można odczytać **typu** przypisanej struktury. Można określić długość tablicy (array).	**Można** odczytać przypisany **typ** oraz długość tablicy (array).
	Wskaźnik VARIANT można stworzyć i sprawdzić pod kątem poprawności używając poleceń systemowych.

Zalecenia

- Upewnij się czy poprawnie zastosowałeś wskaźnik ANY. W wielu przypadkach, nie jest on wymagany (patrz tabela poniżej).
- Korzystaj z typu VARIANT wyłącznie do adresowania pośredniego, gdy typy danych zostaną określone dopiero podczas uruchomienia programu.
  - Korzystaj z typu VARIANT jako parametrów InOut, do tworzenia funkcji przetwarzających dane (bloki DB) różnego typu (sprawdź przykład w tym rozdziale).
  - Typ VARIANT oferuje więcej korzyści w porównaniu do wskaźnika ANY. Test zgodności pozwala wykryć błędy podczas kompilacji, a adresowanie symboliczne sprawia, że kod jest bardziej czytelny.
  - Korzystaj z polecenia Variant np. do określenia typu danych (sprawdź przykłady w rozdziale 2.9.3 Polecenia VARIANT (wyłącznie dla S7-1500))
- Korzystaj z tablic indeksowanych (ARRAY) zamiast wskaźników ANY do wywołania komórki danej tablicy (sprawdź w rozdziale 3.6.2 Dane typu ARRAY oraz pośrednie wywołanie obszaru)

Tabela 2-15: Porównanie wskaźnika ANY oraz uproszczenie jego funkcji w S7-1500

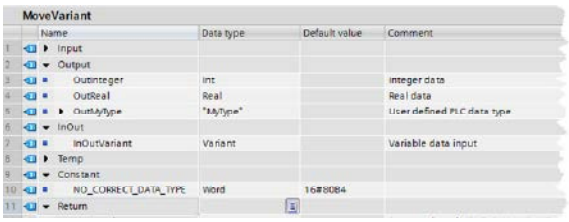
Zastosowanie wskaźników ANY		Uproszczenie funkcji wskaźnika ANY w S7-1500
Funkcje do przetwarzania danych różnego typu.	→	Funkcje wykorzystujące wskaźnik Variant jako parametr InOut dla bloków (sprawdź poniższe przykłady)
Przetwarzanie tablic • Np. odczytywanie, inicjowanie oraz kopiowanie elementów tego samego typu.	→	Standardowe funkcje: • Odczytywanie i zapisywanie za pomocą #myArray[#index] (sprawdź w rozdziale 3.6.2 Dane typu ARRAY oraz pośrednie wywołanie obszaru • Kopiowanie za pomocą MOVE_BLK (sprawdź w rozdziale 2.9.2 Polecenia MOVE)
Przenoszenie oraz przetwarzanie struktur. Np. przenoszenie struktur zdefiniowanych przez użytkownika za pomocą wskaźników ANY do funkcji.	→	Przenoszenie struktur, jako parametrów InOut • Sprawdź w rozdziale 3.3.2 Wywołanie przez referencję - dla interfejsów wej/wyj (InOut)

Przykład

VARIANT umożliwia rozpoznanie typu danych już z poziomu programu użytkownika i podjęcie odpowiednich kroków.
Aby dowiedzieć się więcej na temat programowania z wykorzystaniem typu VARIANT, sprawdź poniższy kod funkcji (FC) „MoveVariant”

- Parametr formalny InOut „InVar” wykorzystany jest do wyświetlenia zmiennej bez względu na jej typ danych.
- Polecenie „Type_Of” służy do określenia, jaki typ danych jest przypisany do danego parametru aktualnego.
- Polecenie “MOVE_BLK_VARIANT” kopiuje wartość zmiennej do innych wyjściowych parametrów formalnych uwzględniając typ danych.

Rysunek 2-13: Parametr formalny funkcji FC „MoveVariant”




```
CASE TypeOf(#InOutVariant) OF // Check datatypes
  Int:  // Move Integer
    #MoveVariant := MOVE_BLK_VARIANT(SRC := #InOutVariant,
                                     COUNT := 1,
                                     SRC_INDEX := 0,
                                     DEST_INDEX := 0,
                                     DEST => #OutInteger);
  Real:  // Move Real
    #MoveVariant := MOVE_BLK_VARIANT(SRC := #InOutVariant,
                                     COUNT := 1,
                                     SRC_INDEX := 0,
                                     DEST_INDEX := 0,
                                     DEST => #OutReal);
  MyType: // Move MyType
     #MoveVariant := MOVE_BLK_VARIANT(SRC := #InOutVariant,
                                     COUNT := 1,
                                     SRC_INDEX := 0,
                                     DEST_INDEX := 0,
                                     DEST => #OutMyType);
  ELSE  // Error, no sufficient data type
     #MoveVariant := WORD_TO_INT(#NO_CORRECT_DATA_TYPE);
     // 80B4: Error code of MOVE_BLK_VARIANT: Data types do
        not correspond
END_CASE;
```

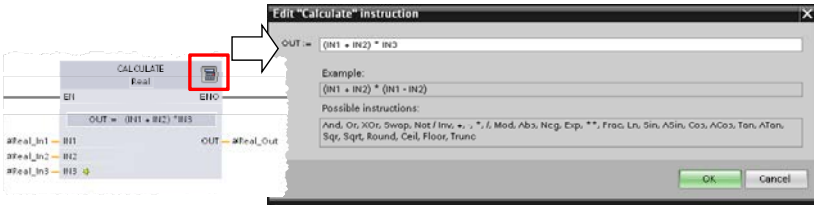
Wskazówka Skorzystaj z polecenia VariantGet (zamiast MOVE_BLK_VARIANT), jeżeli chcesz kopiować wartości nieuporządkowanych zmiennych typu VARIANT. (sprawdź w rozdziale 2.9.3 Polecenia VARIANT (wyłącznie dla S7-1500))

2.9 Polecenia

2.9.1 CALCULATE

Polecenie CALCULATE służy do wykonywania obliczeń matematycznych (np. (IN1 + IN2) * IN3) bez względu na typ danych. Formułę matematyczną można wprowadzić w pasku formuły.

Rysunek 2-14: Polecenie CALCULATE i pasek formuły



Wskazówka Więcej informacji można znaleźć w pomocy Online dla TIA Portal.

Zalety

- Wywołanie formuły matematycznej za pomocą jednego polecenia
- Prosta konfiguracja pozwala zaoszczędzić czas

Właściwości

- Obsługuje sekwencje bitowe, liczby całkowite oraz liczby zmiennoprzecinkowe
- Obsługuje wiele funkcji matematycznych (wszystkie podstawowe działania arytmetyczne, funkcje trygonometryczne, zaokrąglanie, logarytmy itp.)
- Można rozszerzyć ilość wejść

Zalecenia

- Zawsze korzystaj z polecenia CALCULATE dla działań matematycznych, zamiast poleceń takich jak np. ADD, SUB, itp.

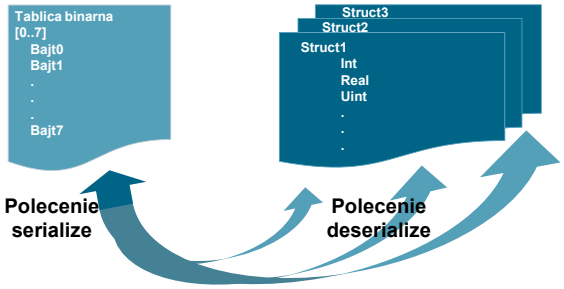
2.9.2 Polecenia MOVE

W ramach STEP 7 (TIA) można skorzystać z następujących poleceń typu MOVE. Polecenie MOVE_BLK_VARIANT zostało dodane dla sterowników S7-1200/S71500

Tabela 2-16: Polecenia typu „Move”

Polecenie	Zastosowanie	Właściwości
MOVE	Kopiowanie wartości Kopiowanie tablicy	• Kopiowanie zawartości parametru na wejściu IN do parametru na wyjściu OUT. • Parametry na wejściu oraz wyjściu muszą być tego samego typu. • Parametry mogą być również zmiennymi uporządkowanymi (dane PLC). • Kopiowanie kompletnych tablic i struktur.
MOVE_BLK	Kopiowanie kilku obszarów	• Kopiowanie zawartości tablicy do innej tablicy. • Tablica źródłowa oraz docelowa muszą być tego samego typu. • Kopiowanie całych tablic oraz struktur. • Kopiowanie poszczególnych elementów tablicy. Dodatkowo, można określić ilość kopiowanych elementów oraz element, od którego rozpocząć kopiowanie.
UMOVE_BLK	Kopiowanie tablicy bez przerwania	• Kopiowanie zawartości tablicy bez ryzyka przerwania procesu przez OB. • Tablica źródłowa oraz docelowa muszą być tego samego typu.
MOVE_BLK_VARIANT (wyłącznie dla S7-1500)	Kopiowanie tablicy	• Kopiowanie jednej lub kilku zmiennych uporządkowanych (dane PLC). • Rozpoznawanie typów danych podczas wykonywania programu. • Szczegółowe raportowanie błędów. • Obsługa danych PLC, tablic oraz bloków danych w tablicy.
Serialize (wyłącznie dla S7-1500)	Kopiowanie danych uporządkowanych do tablicy binarnej	• Łącznie kilku rekordów danych w pojedynczą tablicę binarną, oraz przesyłanie jej do innych urządzeń w postaci komunikatu. • Przesyłanie parametrów wejściowych oraz wyjściowych jako danych typu Variant.
Deserialize (wyłącznie dla S7-1500)	Kopiowanie danych z tablicy bajtu do jednej lub kilku struktur	• Wykorzystanie urządzenia I-Device: Urządzenie I device odbiera kilka rekordów danych, które są następnie kopiowane do innych struktur . • Łączenie kilku rekordów danych w pojedynczą tablicę binarną oraz kopiowanie jej do innych struktur.

Rysunek 2-15: Polecenia: serialize oraz deserialize (wyłącznie dla S7-1500)



Zalecenia

- Istnieją istotne różnice pomiędzy poleceniami MOVE, MOVE_BLK oraz MOVE_BLK_VARIANT
  - Polecenie MOVE służy do kopiowania całych struktur.
  - Polecenie MOVE_BLK służy do kopiowania fragmentów tablicy zawierającej dane o znanym typie.
  - Polecenie MOVE_BLK_VARIANT służy do kopiowania fragmentów tablic, których typ danych zostanie określony dopiero w trakcie wykonywania programu.

Wskazówka

UMOVE_BLK: Proces kopiowania nie może zostać przerwany przez system operacyjny. Czas reakcji CPU na sytuacje awaryjne może zostać wydłużony podczas wykonywania polecenia „Copy array without interruption”.

Pełen opis polecenia MOVE można znaleźć w pomocy online dla TIA Portal.

Wskazówka

Więcej informacji można znaleźć w poniższych pozycjach:

Kopiowanie obszarów pamięci w STEP 7 (TIA Portal)
http://support.automation.siemens.com/WW/view/en/59886704

2.9.3 Polecenia VARIANT (wyłącznie dla S7-1500)

Tabela 2-17: Polecenia „Move”

Polecenie	Zastosowanie	Właściwości
Polecenia „MOVE”		
VariantGet	Odczyt wartości	Pozwala odczytać wartość zmiennej wskazywanej przez VARIANT.
VariantPut	Zapis wartości	Pozwala zapisać wartość zmiennej wskazywanej przez VARIANT
Listy		
CountOfElements	Zliczanie elementów	Pozwala zliczyć elementy tablicy wskazywanej przez VARIANT
Porównanie poleceń		
TypeOf() (tylko w SCL)	Określanie typu danych	Pozwala wywołać typ danych zmiennej wskazywanej przez VARIANT
TypeOfElements() (tylko w SCL)	Określanie typu danych tablicy	Pozwala wywołać typ danych elementów tablicy wskazywanej przez VARIANT

Wskazówka Więcej poleceń typu VARIANT można znaleźć w pomocy online dla TIA Portal.

2.9.4 Polecenie RUNTIME

Polecenie „RUNTIME” służy do zmierzenia ile czasu potrzeba do wykonania całego programu, pojedynczych bloków lub sekwencji poleceń.
Polecenie to pojawia się w językach SCL (S7-1200/S7-1500) oraz STL (S7-1500).

Wskazówka Więcej informacji można znaleźć w poniższych pozycjach:

Jak zmierzyć czas wykonania części/całego programu w S7-1200/S7-1500 ?
http://support.automation.siemens.com/WW/view/en/87668318

2.10 Symbole i komentarze

2.10.1 Środowisko programowania

Zalety

Aby zwiększyć czytelność kodu, można skorzystać z dodatkowych oznaczeń takich jak komentarze oraz nazwy symboliczne.

Oznaczenia te zostają wgrane razem z kodem bezpośrednio do sterownika. Rozwiązanie to skraca czas serwisowania zakładu, gdy nie ma dostępu do projektu w wersji offline.

Zalecenia

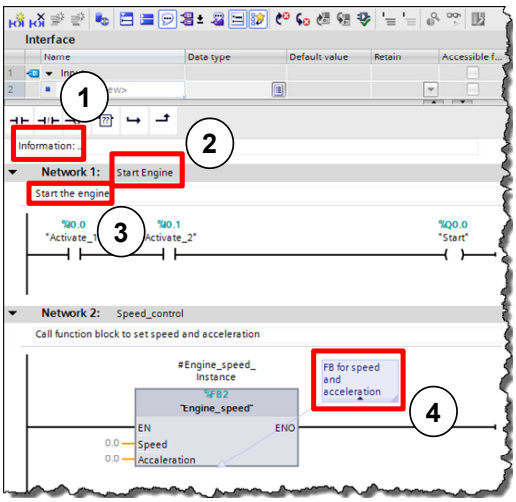
- Sięgnij po komentarze, aby zwiększyć przejrzystość kodu. Komentarze zbiorcze dla networków (sieci) widoczne są nawet, gdy podgląd jest zwinięty.
- Stwórz kod, który będzie czytelny dla pozostałych programistów.

Poniższy przykład przedstawia paletę komentarzy do twojej dyspozycji.

Przykład

Poniższy obrazek przedstawia komentarze dostępne w edytorze LAD (te same funkcje dostępne są dla edytora FDB).

Rysunek 2-16: Wstawianie komentarzy w programie użytkownika (LAD)



Można skorzystać z następujących komentarzy:

1. Komentarz blokowy
2. Komentarz dla sieci (nagłówek)
3. Komentarz szczegółowy
4. Komentarze do poleceń, bloków oraz funkcji (np. otwórz, zamknij itp.)

Komentarze w językach SCL oraz STL, muszą być poprzedzone znakiem // w każdym rzędzie

Przykład

```
Poziom napełnienia:= Promień * Promień * PI * wysokość;
// oblicza poziom napełnienia średniego zbiornika
```

Wskazówka Więcej informacji można znaleźć w poniższych pozycjach:

Dlaczego nagłówki oraz komentarze nie są wyświetlane po otwarciu projektu w edytorze bloków w STEP 7 (TIA Portal) ?
http://support.automation.siemens.com/WW/view/en/41995518

2.10.2 Komentarze w watch table

Zalety

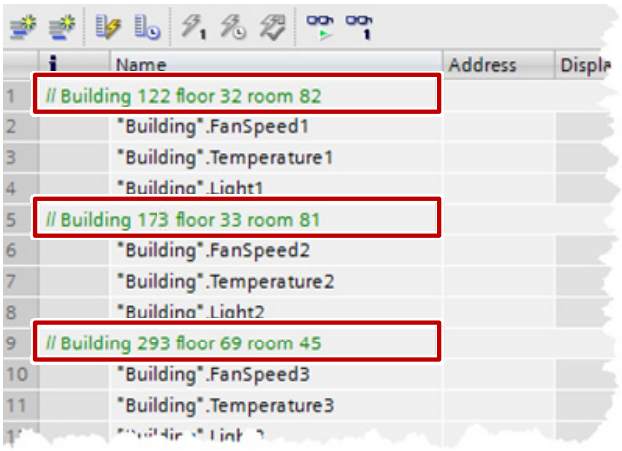
- Komentarze można również dodać w wierszach watch table.

Zalecenia

- Uporządkuj watch table dodając komentarze w wierszach.
- Używaj komentarzy również dla pojedynczych zmiennych.

Przykład

Rysunek 2-17: Watch table ze wstawionymi komentarzami



2.11 Stałe systemowe

Elementy sprzętowe oraz oprogramowanie sterowników S7 300/400 są identyfikowane za pomocą adresów logicznych oraz diagnostycznych.

Sterowniki S7-1200/1500 wykorzystują do tego stałe systemowe. Elementy sprzętowe oraz oprogramowanie (np. interfejsy, moduły, bloki organizacyjne) mają własną stałą systemową, która została wygenerowana automatycznie podczas instalacji.

Zalety

- Wywołuj moduły korzystając z przypisanej mu nazwy, a nie identyfikatora sprzętowego.

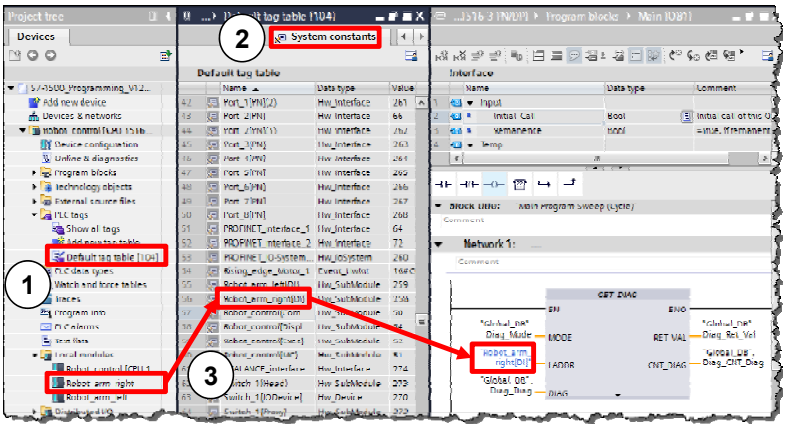
Zalecenia

- Aby ułatwić rozpoznanie, zawsze przypisuj modułom nazwy związane z pełnioną przez nie funkcją.

Przykład

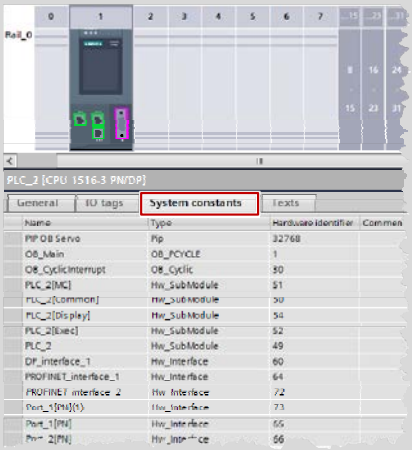
Poniższy przykład przedstawia wykorzystanie stałych systemowych w programie użytkownika.

Rysunek 2-18: „Stałe systemowe” w programie użytkownika



1. Stałe systemowe sterownika można znaleźć w folderze „PLC tags – Default tag table”.
2. Pozostałe stałe znajdują się w osobnej zakładce „Default tag table”.
3. W tym przykładzie, nazwa symboliczna „Robot_arm_left” została nadana modułowi DI.
   Moduł ten można odnaleźć pod tą samą nazwą w zakładce stałe systemowe „system constants”.
   Nazwa „Robot_arm_left” jest powiązana z blokiem diagnostycznym „GET_DIAG”.

Wskazówka Wejdź w zakładkę konfiguracji urządzenia „Device configuration”, aby odszukać stałą systemową interesującego Cię urządzenia.



Wskazówka Więcej informacji można znaleźć w poniższych pozycjach:

Co nowego wnoszą stałe systemowe do modułów S7-1200/1500?
http://support.automation.siemens.com/WW/view/en/78782836

2.12 Stałe użytkownika

Stałe użytkownika (zmienne Constant) służą do zapisywania wartości stałych. Rozróżnia się stałe lokalne dla bloków organizacyjnych (OB), funkcji (FC) oraz bloków funkcyjnych (FB) oraz stałe globalne dla całego programu użytkownika.

Zalety

- Stałe użytkownika (zmienne Constant) można wykorzystać do zmiany wartości stałych (np. wartości progowych, indeksów tablic), zarówno lokalnie w blokach funkcji jak i globalnie w całym programie.
- Dzięki stałym użytkownika, program jest bardziej przejrzysty.

Właściwości

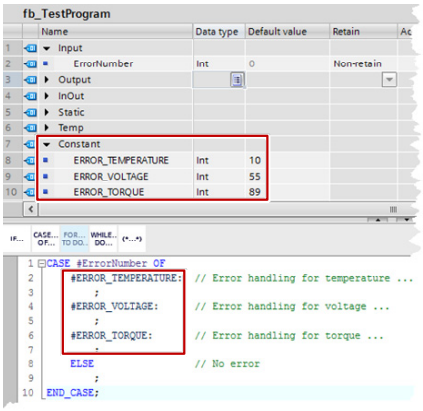
- Stałe lokalne definiuje się w interfejsie danego bloku.
- Stałe globalne definiuje się w folderze „PLC tags“.
- Program użytkownika zezwala jedynie na odczyt stałych użytkownika.
- Stałe użytkownika nie są widoczne w przypadku bloków chronionych.

Zalecenia

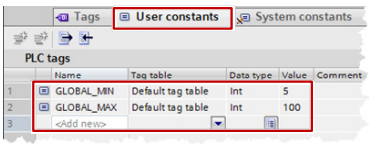
- Korzystaj ze stałych użytkownika, aby poprawić przejrzystość programu oraz do zmiany:
  - kodów błędu,
  - poleceń CASE,
  - współczynników przeliczeniowych,
  - stałych naturalnych…

Przykład

Rysunek 2-19: Stała bloku (lokalna) dla polecenia CASE



Rysunek 2-20: Stała sterownika (globalna)



Wskazówka Więcej informacji można znaleźć w poniższej pozycji:

W jaki sposób przekonwertować jednostki zmiennych w STEP 7 (TIA Portal)?
http://support.automation.siemens.com/WW/view/en/61928891

2.13 Wewnętrzna identyfikacja sterowników oraz zmiennych HMI

STEP 7, WinCC, Startdrive, oraz Safety korzystają ze wspólnej bazy danych środowiska projektowego TIA Portal. Jakiekolwiek zmiany danych obejmują cały program użytkownika, bez względu na to gdzie zmiana została wprowadzona. Dzięki temu dane pozostają spójne w obrębie całego programu.

Każda nowa zmienna otrzymuje unikalny identyfikator, niedostępny dla użytkownika. Nie można go wyświetlić ani zmienić nawet podczas modyfikacji adresu zmiennej.

Poniższy rysunek przedstawia sposób wewnętrznej identyfikacji danych.

Rysunek 2-21: Wewnętrzna identyfikacja dla PLC oraz HMI

PLC_1 **HMI_1**

Nazwa symbolu PLC	Adres absolutny	Wewnętrzny identyfikator PLC	Wewnętrzny identyfikator HMI	Nazwa symbolu HMI	Tryb dostępu	Połącznie za pomocą PLC
Silnik_1	I0.0	000123	009876	Silnik_1	<adres symboliczny>	PLC_1
Zawór_2	Q0.3	000138	000578	Zawór_2	<adres symboliczny>	PLC_1

Wskazówka Identyfikator ulegnie zmianie w przypadku:

- zmiany nazwy
- zmiany typu
- usunięcia zmiennej

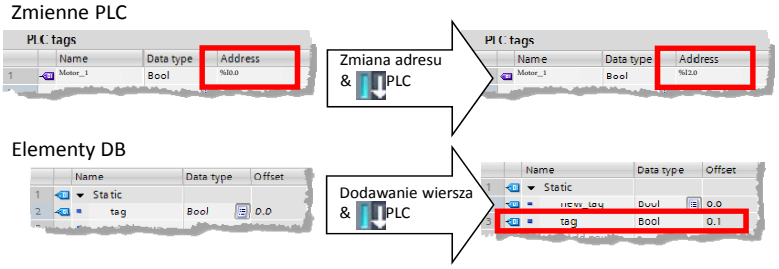
Zalety

- Można dowolnie zmieniać adresy zmiennych nie zmieniając ich wewnętrznych identyfikatorów oraz powiązania. Komunikacja pomiędzy sterownikiem, urządzeniem HMI, a napędem również pozostaje bez zmian.
- Długość nazwy symbolicznej nie wpływa na komunikację między sterownikiem, a urządzeniem HMI.

Właściwości

Po zmodyfikowaniu adresu zmiennych PLC należy ponownie uruchomić sterownik. Ponowne uruchomienie urządzenia HMI nie jest konieczne, ponieważ system odnajdzie zmienne po ich wewnętrznym identyfikatorze. (sprawdź Rysunek 2-22: Zmiana adresu lub dodawanie wiersza).

Rysunek 2-22: Zmiana adresu lub dodawanie wiersza



2.14 Błędy oraz tryb STOP

Sterowniki S7-1200/1500 są bardziej stabilne w porównaniu do S7-300/400. Zdecydowanie mniej zdarzeń może spowodować przejście sterownika w tryb STOP.

Spójność bloków programowych jest sprawdzana już na poziomie kompilacji programu w TIA Portal. Dzięki temu sterowniki S7-1200/1500 są bardziej stabilne od ich poprzedników.

Zalety

Tylko trzy rodzaje błędów mogą spowodować przejście sterownika S7-1200/1500 w tryb STOP, przez co łatwiej jest zaprogramować reakcję sterownika na błędy.

Właściwości

Tabela 2-18: Reakcja sterowników S7-1200/1500 na błędy

	Błąd	S7-1200	S7-1500
1.	Czas monitorowania cyklu przekroczony jednokrotnie	RUN	**STOP**, gdy OB80 nieskonfigurowany
2.	Czas monitorowania cyklu przekroczony dwukrotnie	**STOP**	**STOP**
3.	Błędy programowania	RUN	**STOP**, gdy OB121 nieskonfigurowany

Błędy bloków organizacyjnych (OB)

- Blok OB80 „Time error interrupt" zostaje wywołany przez system operacyjny w sytuacji przekroczenia maksymalnego czasu wykonania cyklu.
- Blok OB121 „Programming error" zostaje wywołany przez system operacyjny w przypadku błędu podczas wykonania programu.

Każdy błąd zostaje odnotowany w buforze diagnostycznym.

Wskazówka

Sterowniki S7-1200/1500, posiadają również inne programowalne bloki operacyjne (OB), które można zaprogramować na wypadek np. błędów diagnostycznych, awarii modułu typu rack, itp.

Więcej informacji na temat błędów można znaleźć w pomocy online dla TIA Portal pod hasłem „Events and OBs".

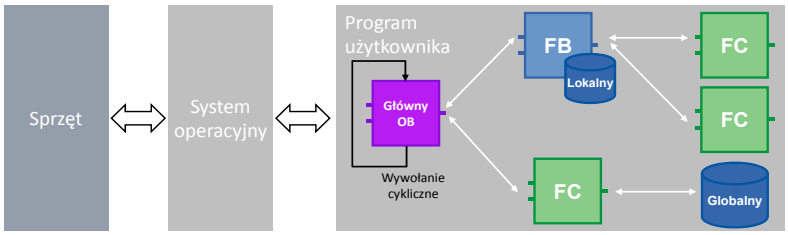
3 Programowanie

3.1 System operacyjny i program użytkownika

Sterowniki SIMATIC składają się z systemu operacyjnego i programu użytkownika.

- System operacyjny odpowiada za wszystkie funkcje, które nie są bezpośrednio związane z głównym zadaniem sterownika (np. ponownie uruchomienie sterownika, aktualizowanie obrazu procesu, wywołanie programu użytkownika, obsługa błędów, organizacja pamięci, itp.) System operacyjny jest integralną częścią sterownika.
- Program użytkownika zbudowany jest z bloków niezbędnych do realizacji danego zadania. Do zaprogramowania programu użytkownika wykorzystuje się bloki programowalne, które zostają wgrane do sterownika.

Rysunek 3-1: System operacyjny i program użytkownika



Program użytkownika jest zawsze wykonywany w sposób cykliczny. Główny blok organizacyjny (OB.) pojawia się w folderze „Program Blocks", zaraz po stworzeniu sterownika w STEP 7. Następnie jest on przetwarzany przez sterownik w nieskończonej pętli.

3.2 Bloki programowe

Najnowsza wersja STEP 7 (TIA Portal) wykorzystuje podobne rodzaje bloków, co wersje poprzednie:

- Bloki organizacyjne
- Bloki funkcyjne
- Funkcje
- Bloki danych

Doświadczeni użytkownicy STEP 7 bez problemu mogą kontynuować prace z nową wersją oprogramowania. Natomiast dla nowych użytkowników, zaznajomienie się z programowaniem w STEP 7 nie powinno stanowić większego wyzwania.

Zalety

- Zróżnicowane typy bloków pozwalają stworzyć program o przejrzystej budowie.
- Przejrzysta budowa programu umożliwia wielokrotne wykorzystanie programu lub wybranych funkcji np. w innych projektach. Wystarczy jedynie zmienić konfigurację. (sprawdź w rozdziale 3.2.8 Bloki wielokrotnego użytku „reusable blocks").
- Łatwiejsze wykrywanie oraz usuwanie błędów, w tym błędów w programowaniu przekłada się na mniej problemów w zarządzaniu zakładem produkcyjnym.

Zalecenia

- Pamiętaj o logicznej budowie każdego zadania.
- Stwórz program, w którym każda funkcja jest podzielona na podfunkcje. Te z kolei zbudowane są z jeszcze mniejszych elementów. Dokonuj dalszych podziałów, aż uzyskasz funkcje, które będziesz w stanie wykorzystać wielokrotnie, podmieniając jedynie parametry.
- Określ interfejsy (wejścia/wyjścia) pomiędzy poszczególnymi blokami funkcji łącznie z interfejsami funkcji dostarczonych przez innych producentów.

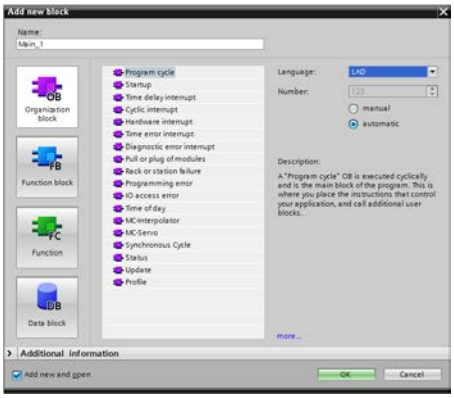
Bloki organizacyjne, bloki funkcyjne oraz funkcje można zaprogramować w poniższych językach programowania.

Tabela 3-1: Języki programowania

Język programowania	S7-1200	S7-1500
Ladder (LAD)	✓	✓
Function block diagram (FBD)	✓	✓
Structured control language (SCL)	✓	✓
Graph	×	✓
Statement list (STL)	×	✓

3.2.1 Bloki organizacyjne (OB)

Rysunek 3-2: Okno dodawania nowego bloku „Add new block” (OB)



Bloki organizacyjne OB są interfejsem pomiędzy systemem operacyjnym a programem użytkownika. Odpowiadają za następujące procesy:

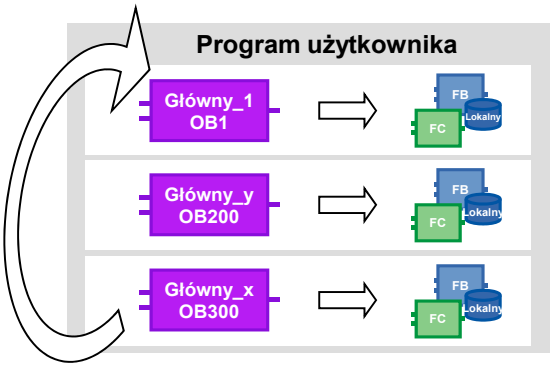
- Zachowanie sterownika podczas rozruchu
- Cykliczne wykonanie programu
- Przetwarzanie programu z kontrolowanymi przerwaniami
- Obsługa błędów

Różnorodność dostępnych bloków organizacyjnych zależy od sterownika.

Właściwości

- Bloki organizacyjne są wywoływane przez system operacyjny sterownika
- W programie użytkownika można stworzyć kilka głównych bloków organizacyjnych (Main OB). Będą one przetwarzane sekwencyjne, zgodnie z nadanym numerem

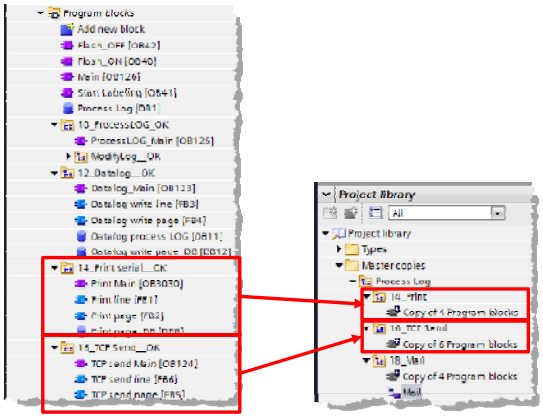
Rysunek 3-3: Korzystanie z kilku głównych bloków organizacyjnych (Main OB)



Zalecenia

- Rozpisz program na kilku głównych blokach organizacyjnych (Main OB) w taki sposób, aby móc swobodnie wymienić poszczególne części między sterownikami.
- Upewnij się, aby główne bloki organizacyjne nie komunikowały się ze sobą. Dzięki temu będą pracować niezależnie. Jeżeli jednak komunikacja między blokami jest konieczna, skorzystaj z globalnych bloków danych (sprawdź w rozdziale 4.2 Brak pamięci bitowej / globalne bloki danych).
- Posegreguj poszczególne części programu zgodnie z ich funkcją do folderów i przechowuj je w bibliotece projektu lub bibliotece globalnej.

Rysunek 3-4: Przechowywanie części program w bibliotece projektu



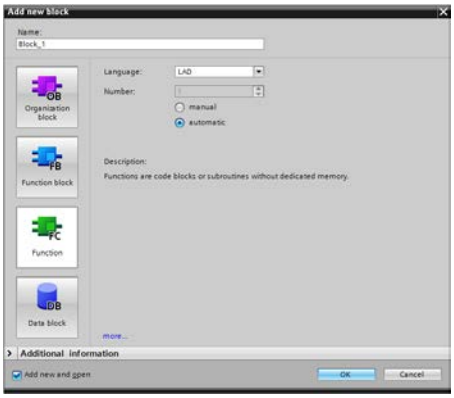
Więcej informacji można znaleźć w rozdziale 3.7 Biblioteki

Wskazówka Więcej informacji można znaleźć w poniższych pozycjach

Jakie bloki organizacyjne są dostępne w STEP 7 (TIA Portal)?
http://support.automation.siemens.com/WW/view/en/58235745

3.2.2 Funkcje (FC)

Rysunek 3-5: Okno dodania nowej funkcji (FC) „Add new block”



Funkcje (FC) nie posiadają pamięci danych pomiędzy cyklami. Wartości użytych parametrów nie są zapisywane. Za każdym razem, kiedy funkcja jest wywołana otrzymuje nowe parametry aktualne.

Właściwości

- Funkcje (FC) służą do przetwarzania danych w pojedynczym cyklu.
- Zmienne tymczasowe oraz zmienne wyjściowe pozostają nieokreślone, kiedy są wywołane w blokach niezoptymalizowanych. W blokach zoptymalizowanych wartości zawsze przyjmują wartość domyślną (S7-1500 i S7-1200 Firmware V4) .
- Aby zapisać parametry funkcji należy skorzystać z globalnych bloków danych.
- Funkcje (FC) mogą mieć kilka wyjść
- Wartość zwracana przez funkcję może zostać bezpośrednio wykorzystana, np. w SCL we wzorze matematycznym.

Zalecenia

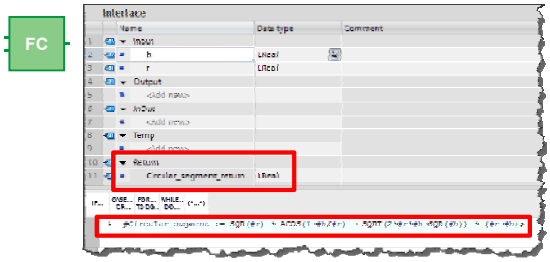
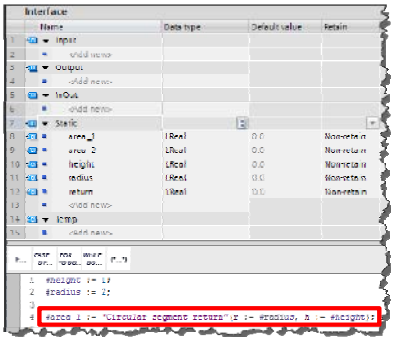
- Używaj funkcji dla powtarzalnych operacji, które są wielokrotnie wywoływane w różnych obszarach programu użytkownika.
- Skorzystaj z możliwości wykorzystania wartości zwracanej przez funkcję.

```
<Operand> := <FC name> (parameter list);
```

Przykład

Poniższy przykład przedstawia wzór matematyczny zaprogramowany w bloku FC. Wynik obliczeń zostaje zadeklarowany, jako wartość zwracana, dzięki czemu może zostać ponownie wykorzystany

Tabela 3-2: Ponowne wykorzystanie wartości funkcji

Krok	Polecenie
1.	Utwórz blok FC, którego wartością zwracaną będzie wynik wzoru matematycznego (tu: wzór na odcinek koła) 
2.	Wywołaj blok FC obliczając odcinek koła w dowolnym bloku (SCL). `<Operand> := <FC name> (parameter list);` 

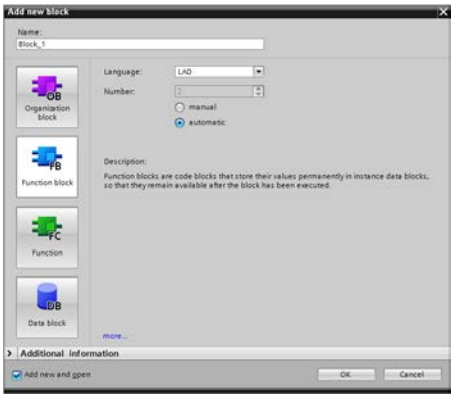
Wskazówka

Więcej informacji można znaleźć w poniższej pozycji:

Ile parametrów (maksymalnie) można zdefiniować w STEP 7 dla funkcji CPU S7-1200/S7-1500 CPU?
http://support.automation.siemens.com/WW/view/en/99412890

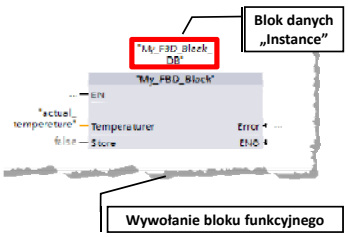
3.2.3 Bloki funkcyjne (FB)

Rysunek 3-6: Okno dodania nowego bloku funkcyjnego (FB) “Add new block”



Bloki funkcyjne (FB) mają wydzielony własny obszar do przechowywania danych pomiędzy cyklami programów. Jest to tzw. blok DB instancji („Instance DB”).

Rysunek 3-7: Wywołanie bloku funkcyjnego



Właściwości

- Bloki funkcyjne (FB) służą do przetwarzania danych z podtrzymaniem wartości między cyklami programu.
- Zmienne tymczasowe oraz zmienne wyjściowe pozostają nieokreślone, kiedy są wywołane w blokach niezoptymalizowanych.
  W blokach zoptymalizowanych wartości zawsze przyjmują wartość domyślną (S7-1500 and S7-1200 Firmware V4).
- Zmienne statyczne zachowują swoją wartość z poprzedniego cyklu

Zalecenia

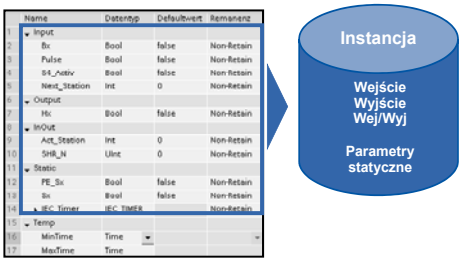
- Używaj bloków funkcyjnych, aby stworzyć przejrzysty program użytkownika. Blok funkcyjny można wywołać wielokrotnie, w różnych obszarach programu użytkownika. Funkcjonalność ta ułatwia zaprogramowanie często powtarzających się części programu.
- Jeżeli bloki funkcyjne stosowane są wielokrotnie w programie użytkownika, skorzystaj z osobnych instancji, a najlepiej z multi-instancji.

3.2.4 Instancje

Pojedyncze wywołanie bloku funkcyjnego nosi nazwę instancji. Dane przetworzone podczas tego wywołania zostają zapisane w bloku danych instance.

Bloki danych typu instance przejmują strukturę interfejsu powiązanych bloków funkcyjnych. Struktury tej nie można zmienić w obrębie bloku danych (DB).

Rysunek 3-8: Budowa interfejsu bloku funkcyjnego



Blok danych instance zbudowany jest z pamięci trwałej posiadającej interfejsy wejścia, wyjścia, wej/wyj i parametry statyczne (Static) oraz pamięci ulotnej (L stack), w której przechowywane są zmienne tymczasowe. Wartości tych zmiennych obowiązują przez jeden cykl, po czym muszą zostać zainicjalizowane,

Właściwości

- Blok danych typu „instance” jest zawsze powiązany z blokiem funkcyjnym (FB).
- Bloki danych typu instance są tworzone automatycznie, podczas wywołania bloku funkcyjnego (FB). Nie trzeba ich tworzyć ręcznie w TIA Portal.
- Budowa bloku danych typu instance jest tożsama z budową powiązanego bloku funkcyjnego FB. Wszelkie zmiany możliwe są wyłącznie w bloku funkcyjnym (FB)

Zalecenia

- Pamiętaj, że dane zapisane w bloku danych instance mogą być zmienione wyłącznie w powiązanym bloku funkcyjnym (FB).

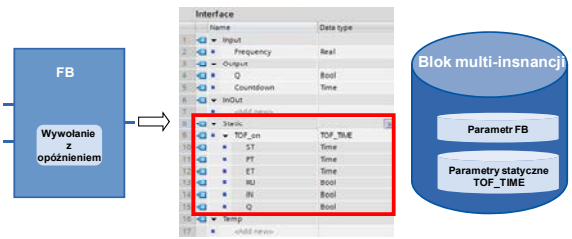
Więcej informacji można znaleźć w rozdziale 3.4 Przechowywanie danych

3.2.5 Multi-instancje

O multi-instancjach możemy mówić w sytuacji, gdy blok funkcyjny został wywołany w innym bloku funkcyjnym. W tej sytuacji, dane wywołanego bloku funkcyjnego zostają zapisane w bloku danych instance nadrzędnego bloku funkcyjnego. Wywołany blok zachowuje pełną funkcjonalność nawet w przypadku, gdy zostanie przeniesiony.

Poniższy rysunek przedstawia blok funkcyjny, który korzysta z innego bloku funkcyjnego FB (“IEC Timer”). Wszystkie dane zapisane zostały w bloku multi-instance

Rysunek 3-9: Multi-instancje



Korzyści/zalety

- Możliwość wielokrotnego zastosowania
- Możliwość wielokrotnego wywołania
- Mniejsza ilość bloków danych instance
- Proste kopiowanie programów
- Możliwość tworzenia struktur w trakcje programowania

Właściwości

- Multi-instancje to obszary pamięci wewnątrz bloków danych instance

Zalecenia

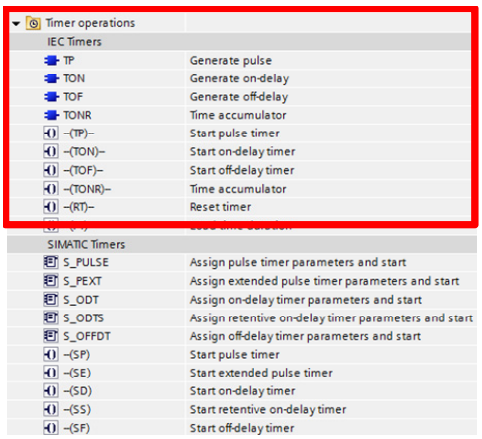
Używaj multi-instancji, aby:

- Zredukować liczbę bloków danych instance.
- Stworzyć przejrzysty program użytkownika do wielokrotnego zastosowania.
- Zaprogramować funkcje lokalne np. timer, licznik, itp.

Przykład

Jeżeli chcesz zaprogramować funkcje timera lub licznika, skorzystaj z bloków “IEC Timer” oraz “IEC Counter” w miejsce SIMATIC Timer (adresowanie absolutne). Wykorzystanie multi-instancji w tym przypadku zmniejszy ilość bloków wykorzystanych w programie użytkownika.

Rysunek 3-10: Biblioteka IEC Timer

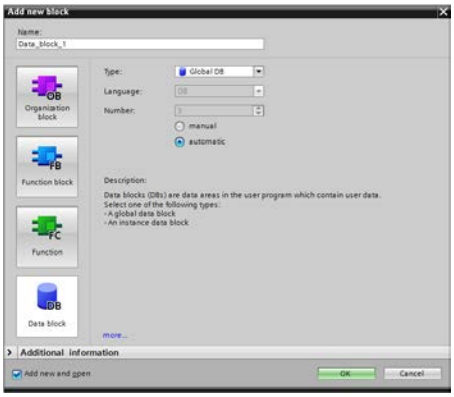


Wskazówka Więcej informacji można znaleźć w poniższej pozycji:

Jak deklarować timery i liczniki dla S7-1500 w STEP 7 (TIA Portal)?
http://support.automation.siemens.com/WW/view/en/67585220

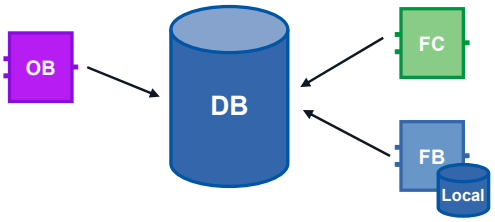
3.2.6 Globalne bloki danych (DB)

Rysunek 3-11: Okno dodawania nowego bloku danych (DB) „Add new block”



Zmienne zapisane w blokach danych są dostępne w obrębie całego programu użytkownika.

Rysunek 3-12: Bloki globalne (Global DB) pełniące funkcję pamięci centralnej



Zalety

- Dobra organizacja obszarów pamięci
- Szybki dostęp

Właściwości

- Wszystkie bloki programu użytkownika mają dostęp do globalnych bloków danych (DB)
- Globalne bloki danych (DB) mogą składać się z danych różnego typu
- Globalne bloki danych (DB) można stworzyć pomocą edytora programu lub zgodnie z typem danych PLC określonym przez użytkownika (sprawdź w rozdziale 3.6.3 Dane typu STRUCT oraz typy danych PLC).

Zalecenia

- Korzystaj z globalnych bloków danych, kiedy chcesz udostępnić dane innym blokom oraz innym częściom programu użytkownika.

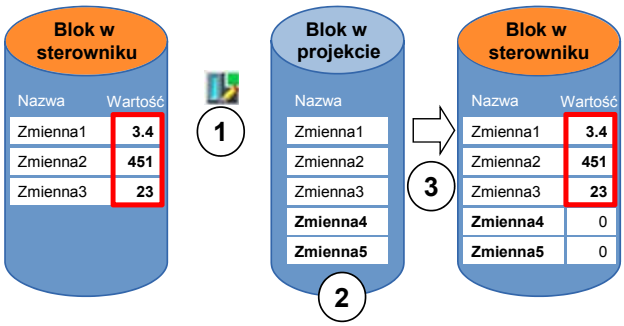
Wskazówka Więcej informacji można znaleźć w poniższej pozycji:

Jakie typy dostępu, kolumny wartości oraz opcje dostępne są dla globalnych bloków danych w STEP?
http://support.automation.siemens.com/WW/view/en/68015631

3.2.7 Wgrywanie bloków bez utraty wartości aktualnych (Downloading without reinitialization)

Sterowniki S7-1200 (firmware V4.0) oraz S7-1500 oferują możliwość rozszerzenia interfejsów zoptymalizowanych bloków funkcyjnych oraz bloków danych w trakcie wykonywania programu. Funkcjonalność ta pozwala wgrać zmodyfikowane bloki bez zatrzymywania pracy sterownika. Wartości aktualne zmiennych pozostają bez zmian.

Rysunek 3-13: Wgrywanie bloków bez re-inicjalizacji



Postępuj zgodnie z poniższymi krokami, podczas gdy sterownik jest w trybie RUN.

1. Włącz opcję „Downloading without reinitialization”
2. Wprowadź nowe zmienne dla istniejącego bloku.
3. Wgraj blok do sterownika.

Zalety

- Możliwość wgrania nowych zmiennych bez utraty wartości aktualnych w trakcie wykonywania programu. Sterownik jest cały czas w trybie „RUN”.

Właściwości

- Funkcjonalność wgrywania bloków bez reinicjalizacji jest dostępna wyłącznie dla bloków zoptymalizowanych.
- Tylko nowe zmienne zostają zainicjowane. Wartość pozostałych zmiennych pozostaje bez zmian.
- Bloki z załączoną funkcją „Downloading without reinitialization” rezerwują dodatkowy obszar pamięci i przez to zajmują więcej pamięci w sterowniku.
- Wielkość obszaru dodatkowego zależy od dostępnej pamięci sterownika, ale nie może przekraczać 2 MB.
- Obszar dodatkowy można określić różny dla każdego bloku
- Domyślna wielkość obszaru dodatkowego to 100 bajtów.
- Bloki mogą być rozszerzane.

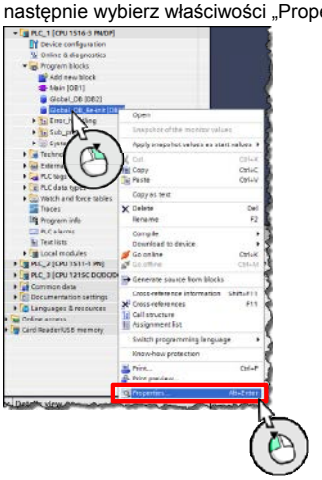
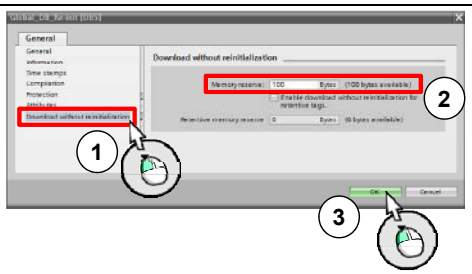
Zalecenia

- Określ wielkość obszaru dodatkowego dla bloków, które będziesz chciał rozszerzyć podczas uruchomienia (np. bloki testowe). Wgranie nowych zmiennych nie przerwie pracy sterownika. Zmienne, które zostały wcześniej wprowadzone nie ulegną zmianie.

Przykład: Ustawianie obszaru dodatkowego dla danego bloku

Poniższa tabela opisuje jak ustawić obszar dodatkowy dla funkcjonalności „downloading without reinitializing”.

Tabela 3-3: Ustawianie obszaru dodatkowego

Krok	Polecenie
1.	Kliknij prawym przyciskiem myszy na jakikolwiek blok zoptymalizowany a następnie wybierz właściwości „Properties” 
2.	 1. Wybierz opcję “Download without reinitialization”. 2. Wprowadź wielkość obszaru dodatkowego oznaczonego, jako „memory reserve”. 3. Potwierdź OK.

Wskazówka

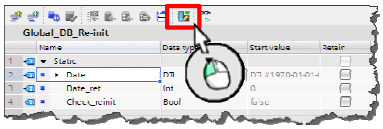
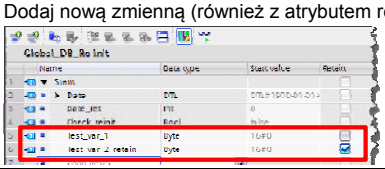
W TIA Portal można również ustawić wartość domyślną obszaru dodatkowego dla wszystkich nowych bloków.

Z paska menu, wybierz „Options – Settings”, a następnie „PLC programming – General – Download without reinitialization“.

Przykład: Wgrywanie bloków bez reinicjalizacji

Poniższy przykład przedstawia funkcję wgrywania bloków bez utraty wartości aktualnych.

Tabela 3-4 Wgrywanie bloków bez reinicjalizacji

Krok	Polecenie
1.	Ustaw dodatkowy obszar pamięci (sprawdź powyżej).
2.	Wybierz blok, np. zoptymalizowany blok danych (DB).
3.	Wybierz opcję „Download without reinitialization”, a następnie potwierdź OK. 
4.	Dodaj nową zmienną (również z atrybutem retentive). 
5.	Wgraj blok do sterownika.
6.	Wartości aktualne w bloku pozostają bez zmian.

Wskazówka Więcej informacji można znaleźć w pomocy online dla TIA Portal pod hasłem „Loading block extensions without reinitialization” oraz w poniższej pozycji:

W jaki sposób mogę pobierać dane w trybie RUN w S7-1500 ?
http://support.automation.siemens.com/WW/view/en/76278126

3.2.8 Bloki wielokrotnego użytku „reusable blocks”

Podział na bloki niesie ze sobą wiele rozwiązań, które pozwolą stworzyć wydajny program.

Zalety

- Bloki mogą być wykorzystane uniwersalnie we wszystkich obszarach programu użytkownika
- Bloki mogą być wykorzystane uniwersalnie w różnych projektach
- Kiedy każdy blok realizuje konkretne zadanie, program automatycznie staje się bardziej przejrzysty.
- Mniej błędów
- Prosta diagnostyka błędów

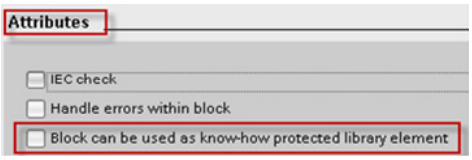
Zalecenia

Zanim ponownie wykorzystasz bloki zapoznaj się z poniższymi informacjami:

- Traktuj bloki w sposób całościowy. Każdy blok powinien realizować konkretne zadanie
- Korzystaj z kilku głównych bloków operacyjnych aby pogrupować poszczególne części zakładu
- Pamiętaj, aby bloki wymieniały dane poprzez interfejs (we/wy), a nie za pomocą instancji.
- Korzystaj z danych ogólnych oraz unikaj umieszczania w blokach:
  - Dostępu do bloków danych DB oraz bloków danych instance
  - Dostępu do zmiennych globalnych (Tagów)
  - Dostępu do stałych globalnych
- Bloki wielokrotnego użytku „reusable blocks” muszą spełniać takie same wymogi jak bloki chronione „know-how-protected blocks” w bibliotekach. Blok nadaje się do ponownego użytku kiedy pomyślnie przejdzie weryfikację pod kątem wykorzystania w bibliotekach (jako blok chroniony „know-how”).

  Zanim rozpoczniesz weryfikację, pamiętaj, aby przeprowadzić kompilację bloku.

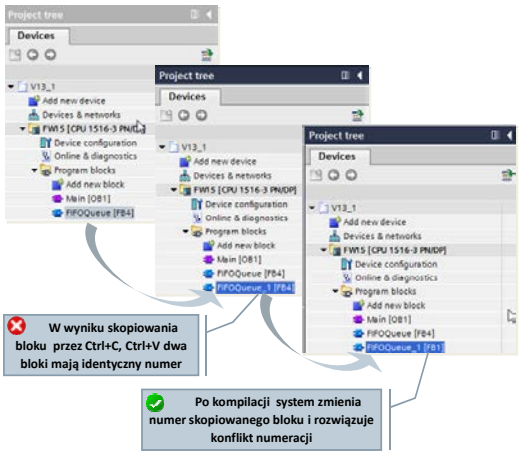
Rysunek 3-14: Właściwości bloków



3.2.9 Automatyczne numerowanie bloków

System automatycznie przypisuje blokom odpowiedni numer (funkcję tę można zaznaczyć we właściwościach bloków).

Rysunek 3-15: Automatyczne numerowanie bloków



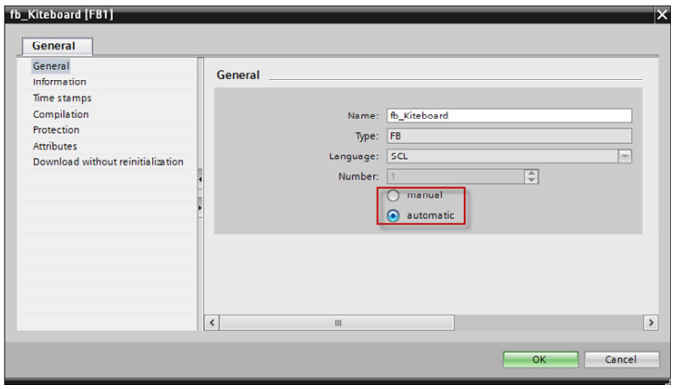
Zalety

- Bloki o sprzecznej numeracji, np. powstałe w wyniku kopiowania są automatycznie usuwane przez TIA Portal podczas kompilacji.

Zalecenia

- Uruchom funkcję automatycznego numerowania bloków

Rysunek 3-16: Ustawienia bloku danych



3.3 Interfejsy bloków

Bloki funkcyjne (FB) oraz funkcje mogą mieć trzy rodzaje interfejsów: Wejście (In) Wej/Wyj (InOut oraz Wyjście (Out). Interfejsy te służą do odbierania oraz przesyłania przetworzonych parametrów. Istnieją dwie możliwości przesyłania parametrów.

3.3.1 Wywołanie przez wartość - dla interfejsu wejścia (In)

Podczas wywołania bloku, wartość parametru aktualnego jest kopiowana do parametru wejściowego bloku. Operacja ta wymaga dodatkowej pamięci.

Rysunek 3-17: Kopiowanie wartości parametru aktualnego do parametru wejściowego

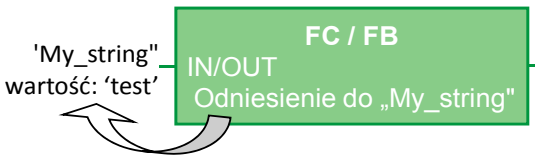
Właściwości

- Operacja ta przebiega tak samo dla każdego bloku.
- Wartości są kopiowane, kiedy blok zostaje wywołany.

3.3.2 Wywołanie przez referencję - dla interfejsów wej/wyj (InOut)

Podczas wywołania bloku, parametr wejściowy odnosi się do adresu parametru aktualnego. Operacja ta nie wymaga dodatkowej pamięci.

Rysunek 3-18: Odwołanie się do wartości (za pomocą wskaźnika)



Właściwości

- Operacja ta przebiega tak samo dla każdego bloku.
- Odniesienie się do parametrów aktualnych podczas wywołania bloku.

Zalecenia

- Korzystaj z interfejsu typu wej/wyj dla zmiennych uporządkowanych (np. ARRAY, STRUCT lub STRING). Opcja ta pozwala zaoszczędzić pamięć.

3.4 Przechowywanie danych

STEP 7 rozróżnia między dwoma obszarami pamięci – pamięcią lokalną i pamięcią globalną. Wszystkie bloki programu użytkownika mają dostęp do pamięci globalnej. Pamięć lokalna dostępna jest wyłącznie w obrębie danego bloku danych.

3.4.1 Wymiana danych poprzez interfejs

Wymiana danych oraz komunikacja między blokami danych poprzez interfejsy niesie ze sobą dodatkowe korzyści.

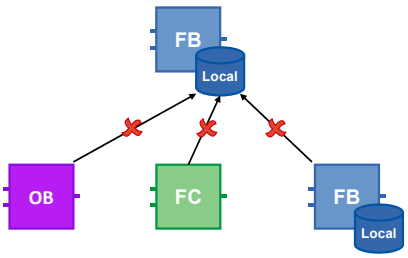
Zalety

- Program ma budowę modułową; składa się z gotowych bloków o ściśle określonych zadaniach
- Program można łatwo rozbudować
- Przejrzysty kod oraz jasne adresowanie

Zalecenia

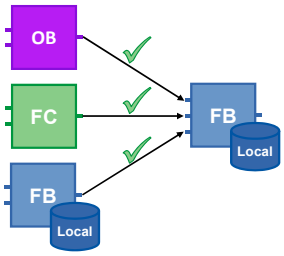
- Używaj zmiennych lokalnych. Dzięki temu bloki będą wykorzystane w sposób uniwersalny oraz modułowy.
- Wymieniaj dane za pośrednictwem interfejsów (In, Out, InOut), aby umożliwić ponowne wykorzystanie bloków.
- Wykorzystuj bloki instance, jako pamięć lokalną wyłącznie dla powiązanych bloków funkcyjnych.

Rysunek 3-19: Niewłaściwy dostęp do bloków danych instance



Jeżeli wymiana danych odbywa się wyłącznie za pośrednictwem interfejsów, poszczególne bloki mogą funkcjonować niezależnie od siebie.

Rysunek 3-20: Wymiana danych za pośrednictwem interfejsów poszczególnych bloków



3.4.2 Pamięć globalna

Pamięć możemy nazwać globalną, kiedy mamy do niej dostęp z każdego miejsca programu użytkownika. Wyróżniamy pamięć sprzętową (np. pamięć bitowa, timery oraz liczniki), oraz globalne bloki danych (DB). W przypadku pamięci sprzętowej istnieje ryzyko, że nie uda się przenieść programu do innego sterownika, ponieważ pamięć sprzętowa będzie już wykorzystana. Dlatego też lepiej jest skorzystać z globalnych bloków danych (DB).

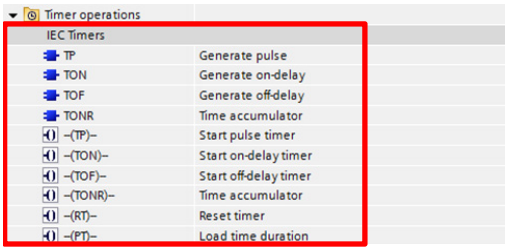
Zalety

- Uniwersalny i niezależny od sprzętu program użytkownika.
- Program użytkownika można zbudować w sposób modułowy, bez podziału obszarów adresowych pamięci bitowej na różnych programistów.
- Zoptymalizowane bloki danych (DB) są zdecydowanie bardziej wydajne niż obszar adresowy pamięci bitowej, który nie jest zoptymalizowany ze względu na kompatybilność.

Zalecenia

- Zawsze korzystaj z globalnych bloków danych (DB)
- Unikaj korzystania z pamięci sprzętowej takiej jak np. sprzętowe timery lub liczniki. Dla multi-instancji korzystaj z liczników i timerów IEC. (sprawdź w rozdziale 3.2.5 Multi-instancje. Aby dowiedzieć więcej o dostępnych timerach wejdź w „Instructions – Basic Instructions – Timer operations”.

Rysunek 3-21: Timery IEC



3.4.3 Pamięć lokalna

- Zmienne statyczne
- Zmienne tymczasowe

Zalecenia

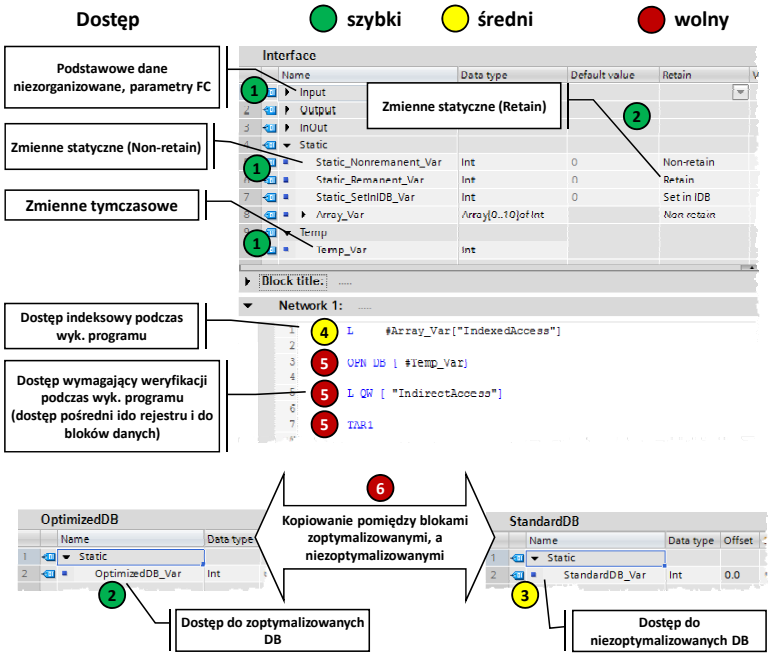
- Używaj zmiennych statycznych dla wartości wymaganych w następnym cyklu.
- Używaj zmiennych tymczasowych, jako pamięci podręcznej obecnego cyklu. Czas dostępu do zmiennych tymczasowych jest krótszy niż w przypadku zmiennych statycznych.

Wskazówka Bloki zoptymalizowane: Zmienne tymczasowe są inicjowane, za każdym razem kiedy blok jest wywoływany poprzez wartość domyślną „default value” (S7-1500 oraz S7-1200 Firmware V4).
Bloki niezoptymalizowane: Zmienne tymczasowe pozostają niezdefiniowane za każdym razem kiedy blok jest wywoływany.

3.4.4 Szybkość dostępu do obszarów pamięci

STEP 7 oferuje różne możliwości dostępu do pamięci. Z powodów systemowych możliwy jest szybki oraz wolny dostęp do różnych obszarów pamięci.

Rysunek 3-22: Różne rodzaje dostępu do pamięci



Najszybszy dostęp dla S7-1200/1500 w kolejności malejącej

1. Bloki zoptymalizowane: zmienne tymczasowe, parametry funkcji (FC) oraz bloków funkcyjnych (FB), zmienne statyczne bez podtrzymywania (Non-retain)
2. Bloki zoptymalizowane o sposobie dostępu znanym w czasie kompilacji
   - Zmienne FB z podtrzymywaniem (Retain)
   - Zoptymalizowane globalne bloki danych (DB)
3. Dostęp do bloków niezoptymalizowanych
4. Dostęp indeksowy, dla indeksów obliczanych podczas wykonywania programu (np. `Motor [i]`)
5. Dostęp wymagający weryfikacji podczas pracy:
   - Dostęp do bloków danych DBs stworzonych w runtime, lub takich, które zostały otwarte pośrednio (np. OPN DB[i])
   - Dostęp do rejestru lub pośredni dostęp do pamięci
6. Kopiowanie struktur pomiędzy blokami zoptymalizowanymi a niezoptymalizowanymi (za wyjątkiem tablic wartości binarnych)

3.5 Funkcja podtrzymywania (retentivity)

W przypadku awarii zasilania, sterownik kopiuje dane z atrybutem „Retain" z pamięci operacyjnej do pamięci stałej. Po ponownym uruchomieniu kontrolera następuje wznowienie programu, z wykorzystaniem skopiowanych danych. Rozmiar dostępnej pamięci retain różni się w zależności od sterownika.

Tabela 3-5: Pamięć podtrzymania (retentive memory) S7-1200/1500

Sterownik	Wielkość pamięci podtrzymania dla pamięci bitowej, liczników, bloków danych (DB) oraz obiektów technologicznych
CPU 1211C,1212C, 1214C, 1215C, 1217C	10 Kb
CPU 1511-1 PN	88 Kb
CPU 1513-1 PN	88 Kb
CPU 1515-2 PN, 1516-3 PN/DP	472 Kb
CPU 1518-4 PN/DP	768 Kb

Tabela 3-6: Różnice pomiędzy S7-1200 a S7-1500

S7-1200	S7-1500
Funkcje podtrzymania można ustawić tylko dla pamięci bitowej	Funkcje podtrzymania można ustawić dla pamięci bitowej, godziny oraz liczników

Zalety

- Dane z atrybutem „Retain" zachowują swoje wartości nawet, gdy sterownik przejdzie z trybu STOP w RUN, lub w przypadku awarii zasilania, po której następuje ponowne uruchomienie sterownika.

Właściwości

W blokach zoptymalizowanych atrybut retain można zaznaczyć dla każdej zmiennej z osobna. W przypadku bloków niezoptymalizowanych atrybut retain może być zaznaczony lub odznaczony wyłącznie dla całego bloku.

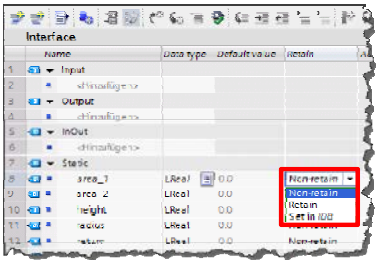
Dane z atrybutem retain można usunąć za pomocą funkcji „memory reset" lub „Reset to factory settings":

- Wciskając przycisk MRES
- Wybranie odpowiedniej opcji na wyświetlaczu
- Online za pomocą STEP 7 (TIA Portal)

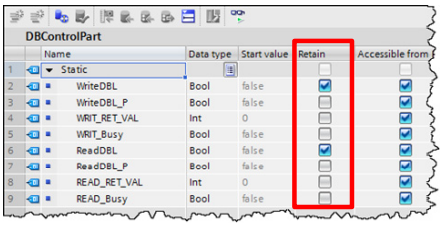
Zalecenia

- Unikaj ustawienia „Set in IDB". Pamiętaj, aby zaznaczyć atrybut retain dla bloków funkcyjnych (FB), a nie bloków instance.
- Ustawienie „Set in IDB" zwiększa czas wykonania sekwencji programu. Korzystaj z ustawienia „Non-retain" lub „Retain".

Rysunek 3-23: Edytor programu (interfejs bloków funkcyjnych)



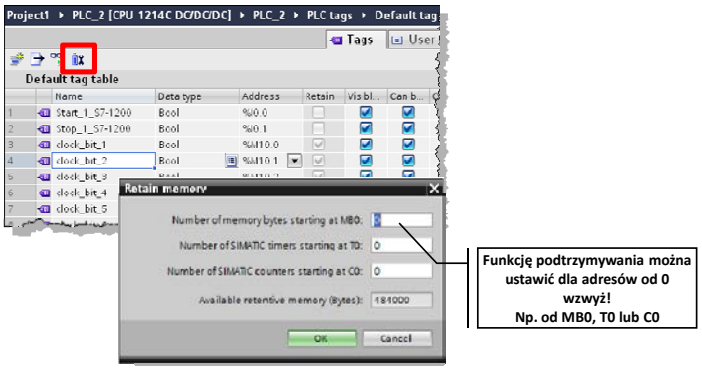
Rysunek 3-24: Edytor programu (bloki danych)



Przykład: Ustawienie atrybutu retain dla zmiennych PLC

Atrybut retain można ustawić w tabelach dla zmiennych PLC, bloków funkcyjnych oraz bloków danych.

Rysunek 3-25: Ustawienie atrybutu retain dla zmiennych PLC

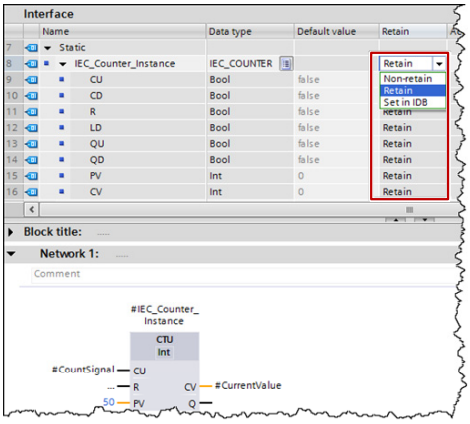


Przykład: Licznik z atrybutem retain

Atrybut retain można również ustawić dla instancji różnych funkcji np. timera lub licznika itp., zgodnie z opisem w rozdziale 3.2.5 Multi-instancje

.

Rysunek 3-26: Licznik z atrybutem retain multi-instancja



Wskazówka

Jeżeli pamięć podtrzymania sterownika jest niewystarczająca wówczas można zapisać dane w blokach danych (DB) znajdujących się w pamięci ładowania (load memory) sterownika. Problem ten jest opisany na przykładzie sterowników S7-1200. Proponowane rozwiązanie działa również dla sterowników S7-1500.

Więcej informacji można znaleźć w poniższej pozycji:

Jak skonfigurować bloki danych z atrybutem „Only store in load memory" w STEP 7 (TIA Portal)
http://support.automation.siemens.com/WW/view/en/53034113

3.6 Adresowanie symboliczne

3.6.1 Adresowanie symboliczne zamiast absolutnego

TIA Portal został zoptymalizowany pod kątem programowania symbolicznego, co wiąże się z wieloma korzyściami. Sterownik sam sortuje dane według ich rodzaju, a programista może skupić swoją uwagę na znalezieniu optymalnego rozwiązania dla danej aplikacji.

Zalety

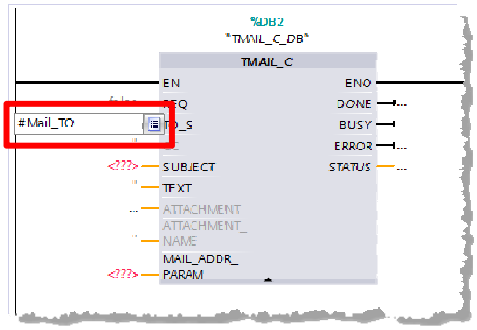
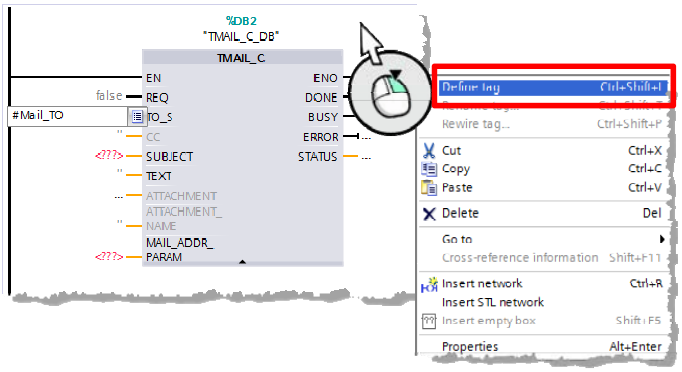
- Bardziej przejrzysty program dzięki nazwom symbolicznym
- Automatyczna aktualizacja nazw zmiennych w obrębie całego programu
- W pełni zautomatyzowane przechowywanie danych programowych (w przeciwieństwie do adresowania absolutnego)
- Wydajny dostęp do danych
- Brak potrzeby ręcznej optymalizacji np. z związanej z rozmiarem programu, lub jego wydajnością
- Technologia IntelliSense do szybkiego wprowadzania symboli
- Automatyczna weryfikacja typów danych – mniej błędów wykonania programu

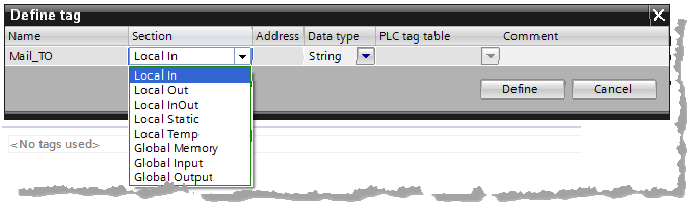
Zalecenia

- „Zapomnij o organizowaniu przechowywanych danych.”
- „Myśl symbolami”. Korzystaj z nazw opisowych np. nagrzewnica_pomieszczenie_4, lub boiler_pompa_1. Dzięki temu program będzie łatwy do odczytania i nie będzie wymagał dodatkowych komentarzy.
- Najpierw nadaj wszystkim zmiennym nazwę symboliczną, a następnie je zdefiniuj.

Przykład

Tabela 3-7: Tworzenie zmiennych symbolicznych

Krok	Polecenie
1.	W edytorze programu wybierz dowolny blok.
2.	Wprowadź nazwę symboliczną. 
3.	Wybierz opcję „Define tag…” klikając prawy przycisk myszki 

Krok	Polecenie
4.	Zdefinjuj zmienną. 

Jest sposób, aby szybko zdefiniować kilka zmiennych w danej sieci. Najpierw należy nadać im nazwy symboliczne a następnie zdefiniować je wszystkie naraz, w identyczny sposób jak zaprezentowano w kroku 4 powyżej.

Note

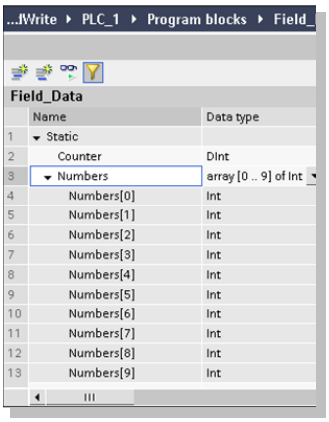
Więcej informacji można znaleźć w poniższej pozycji:

Dlaczego korzystanie z uniwersalnych definicji oraz symboli jest obowiązkowe dla sterowników S7-1500.?
http://support.automation.siemens.com/WW/view/en/67598995

3.6.2 Dane typu ARRAY oraz pośrednie wywołanie obszaru

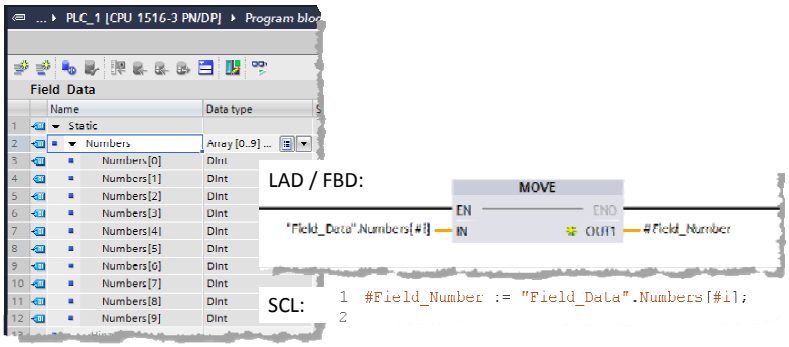
ARRAY to struktura składająca się z danych tego samego typu. Można ją zastosować np. do zapisywania receptur, śledzenia materiałów w kolejce materiałów, protokołów itp.

Rysunek 3-27: Struktura ARRAY składające się z 10 elementów Integer (INT)



Każdy element struktury można wywołać (pośrednio) używając zmiennych procesowych (`array ["index"]`).

Rysunek 3-28: Pośrednie wywołanie obszaru



Zalety

- Prosty dostęp, bez względu na typ danych w tablicy.
- Nie wymaga tworzenia skomplikowanych wskaźników
- Możliwość rozszerzenia
- Dostępne we wszystkich językach programowania

Właściwości

- Uporządkowane dane
- Struktura składa się z określonej ilości elementów tego samego typu
- Wielo-wymiarowość tablic (np. Array [1..10, 1..10] of Bool)
- Możliwość pośredniego wywołania obszaru za pomocą zmiennej procesowej (runtime tag) z dynamicznym obliczaniem indeksu podczas wykonywania programu.

Zalecenia

- Korzystaj ze struktur ARRAY zamiast wskaźnika (np. wskaźnika ANY). Nazwy symboliczne wykorzystywane przez ARRAY są bardziej opisowe niż wskaźnik, co przekłada się na czytelność i przejrzystość całego programu.
- Korzystaj z danych typu INT dla zmiennych procesowych, aby zapewnić wysoką wydajność przetwarzania
- Korzystaj z polecenia „MOVE_BLK” do przenoszenia obszarów z jednej tablicy do drugiej
- Korzystaj z polecenia „GET_ERR_ID”, aby wykryć błędy w obrębie tablicy.

Wskazówka

Więcej informacji można znaleźć w poniższych pozycjach:

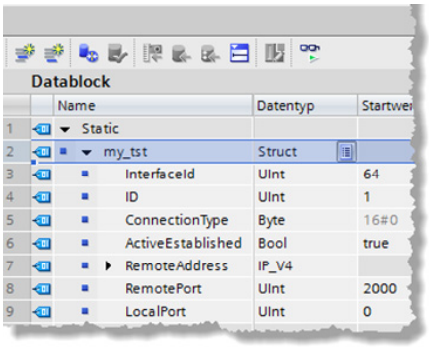
Jak wdrożyć tablice z indeksem zmiennej dla sterowników S7-1500
http://support.automation.siemens.com/WW/view/en/67598676

Jak wywoływać dane w STEP 7 (TIA Portal)?
http://support.automation.siemens.com/WW/view/en/97552147

3.6.3 Dane typu STRUCT oraz typy danych PLC

STRUCT to struktura składająca się z danych różnego typu. Zadeklarowanie struktury odbywa się na poziomie danego bloku.

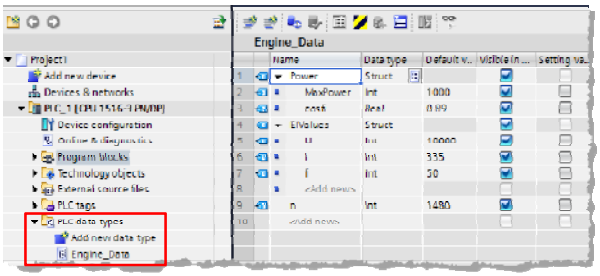
Rysunek 3-29: Struktura składająca się z danych różnego typu



W przeciwieństwie do struktur, typy danych PLC określane są dla całego sterownika. Można je edytować z poziomu TIA Portal. Wszelkie zmiany zostaną wprowadzone dla wszystkich danych tego typu w obrębie programu użytkownika.

Deklaracja danych PLC odbywa się w folderze „PLC data types”, w oknie nawigacji projektu.

Rysunek 3-30: Dane PLC



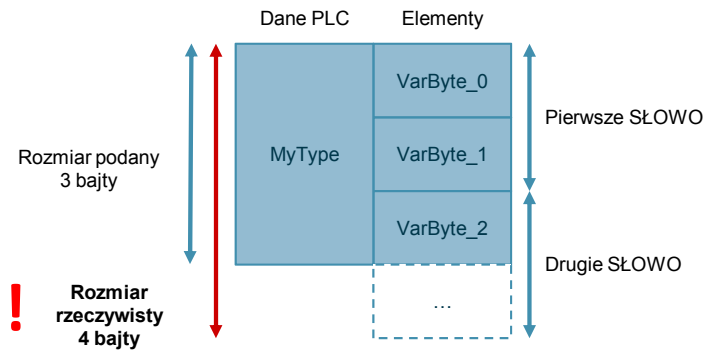
Zalety

- Program użytkownika jest automatycznie aktualizowany po wprowadzeniu zmian.

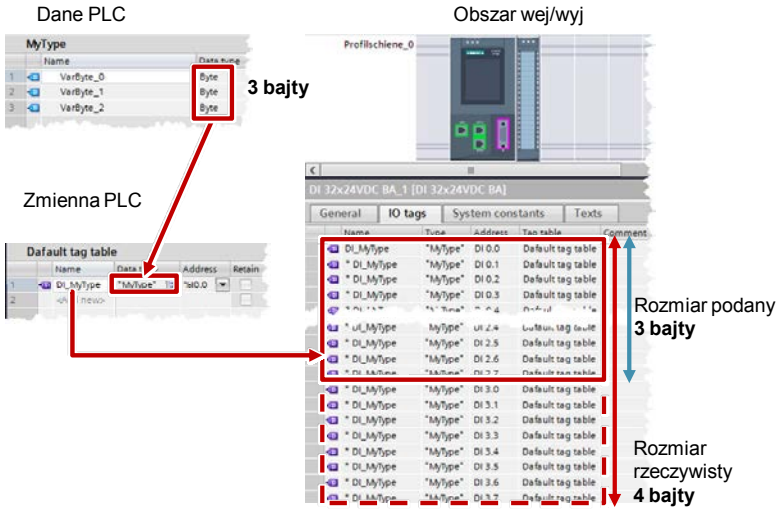
Właściwości

- Dane PLC zawsze kończą się na granicy SŁOWA (WORD) (sprawdź przykłady poniżej).
- Odnosi się to do następujących przypadków:
  - Korzystania z różnych obszarów wej/wyj (sprawdź w rozdziale 3.6.4 Dostęp do obszarów wej/wyj za pomocą danych PLC).
  - Korzystania z ramek z danymi PLC do komunikacji.
  - Rekordów zawierających parametry z danymi PLC dla wej/wyj..
  - Adresowania absolutnego bloków niezoptymalizowanych.

Rysunek 3-31: Dane PLC zawsze kończą się na granicy SŁOWA



Rysunek 3-32: Dane PLC w obszarze wej/wyj



Zalecenia

- Wykorzystuj dane PLC do podsumowywania danych dot. podobnego zagadnienia np. dane silnika (wartość zadana, prędkość, kierunek obrotów, temperatura, itp.)
- Jeżeli dane mają pojawiać się wielokrotnie, w różnych obszarach programu użytkownika skorzystaj z danych PLC, a nie ze struktur.
- Używaj danych PLC do tworzenia struktury bloków danych.
- Używaj danych PLC do określenia struktury bloków danych (DB) bez względu na ich ilość. Możesz w prosty sposób stworzyć dowolną ilość bloków danych (DB) o tej samej strukturze, a następnie je wszystkie modyfikować, wprowadzając zmiany w danych PLC.

Wskazówka

Więcej informacji można znaleźć w poniższych pozycjach:

Jak przeprowadzić inicjalizację struktur w zoptymalizowanych obszarach pamięci w sterownikach S7-1500 w STEP 7 (TIA Portal)?
http://support.automation.siemens.com/WW/view/en/78678761

Jak stworzyć nowy typ danych PLC dla sterownika S7-1500?
http://support.automation.siemens.com/WW/view/en/67599090

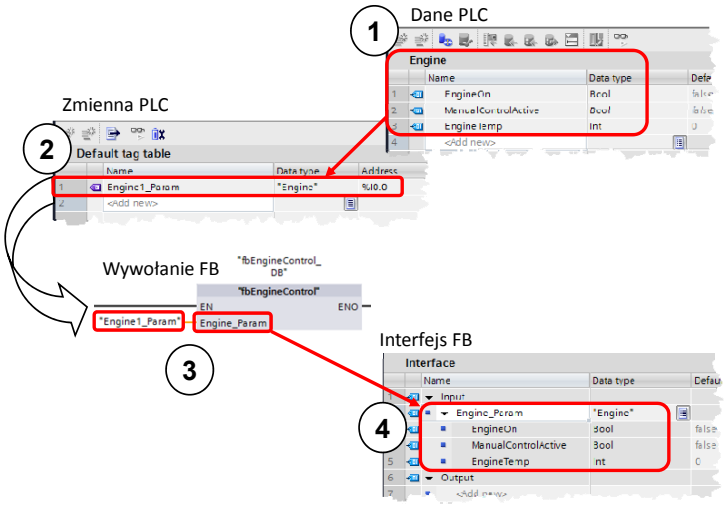
Jak zastosować własny typ danych (UDT) w STEP 7 (TIA Portal) ?
http://support.automation.siemens.com/WW/view/en/67582844

Dlaczego lepiej jest przenosić całe struktury zamiast pojedynczych elementów podczas wywoływania bloku w S7-1500.
http://support.automation.siemens.com/WW/view/en/67585079

3.6.4 Dostęp do obszarów wej/wyj za pomocą danych PLC

Sterowniki S7-1500 pozwalają dowolnie tworzyć dane PLC, które można następnie wykorzystać do uzyskania dostępu do obszarów wej./wyj.

Rysunek 3-33: Dostęp do obszarów wej/wyj za pomocą danych PLC



1. Dane PLC zawierające wszystkie niezbędne informacje.
2. Zmienna PLC tego samego typu, co utworzone dane PLC oraz adres początkowy obszaru wej/wyj. (%Ix.0 lub %Qx.0, np.., %I0.0, %Q12.0, …).
3. Przeniesienie zmiennej PLC, jako parametru aktualnego do bloku funkcyjnego.
4. Wejście bloku funkcyjnego jest tego samego typu, co utworzone dane PLC.

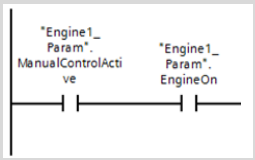
Zalety

- Wysoka wydajność programowania
- Możliwość wielokrotnego wykorzystania, gwarantowana przez dane PLC.

Zalecenia

- Korzystaj z danych PLC, aby uzyskać dostęp do obszarów wej/wyj, np. do symbolicznego przesyłania i obierania telegramów.

Wskazówka Korzystając z programu użytkownika można uzyskać bezpośredni dostęp do poszczególnych elementów danych PLC.

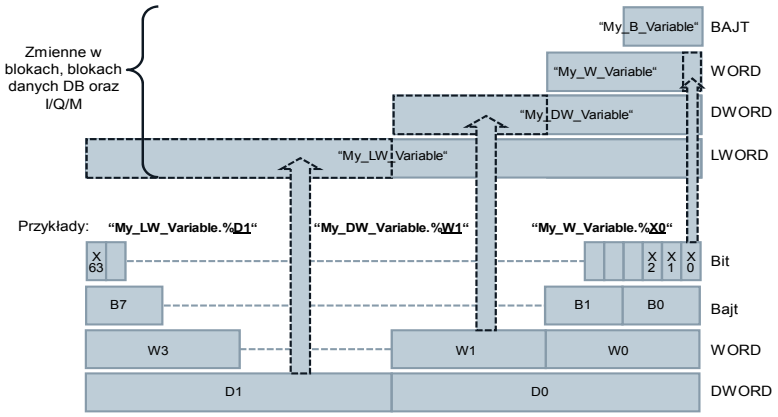


3.6.5 Odwołanie do zmiennych przez „slice access”

Sterowniki S7-1200/1500 pozwalają na dostęp do obszaru pamięci typu Bajt, Word, DWord, oraz LWord. Podział obszaru pamięci (np. bajtu, lub słowa na mniejsze części (np. Bool) przypomina dzielenie na „plasterki” (slice). Dostęp odbywa się przez dopisanie na końcu nazwy zmiennej kropki, znaku procent oraz litery X (bit), B (bajt), W (Word – 16bit) lub D (DWord – 32 bit). Np. „Zmienna.%X0”, „Zmienna.%B1” itp.

Poniższy rysunek przedstawia dostęp symboliczny bitu, bajtu i słowa w dłuższej zmiennej.

Rysunek 3-34: Dostęp „slice access”



Zalety

- Symboliczny dostęp do fragmentów zmiennej
- Deklarowanie zmiennych bez podawania dodatkowych definicji
- Łatwy dostęp (np. do bitów sterujących)

Zalecenia

- Jeżeli chcesz dostać się do danego obszaru w operandzie, skorzystaj z dostępu przez „slice access” (zamiast konstruktorów AT).

Wskazówka

Więcej informacji można znaleźć w poniższej pozycji:

W jaki sposób można uzyskać dostęp do danych nieuporządkowanych bit po bicie, bajt po bajcie, słowo po słowie oraz symbolicznie w STEP 7 (TIA Portal)?
http://support.automation.siemens.com/WW/view/en/57374718

3.7 Biblioteki

TIA Portal pozwala stworzyć niezależne biblioteki zawierające różne elementy programowe, które można wielokrotnie wykorzystać w obrębie programu użytkownika.

Zalety

- Proste przechowywanie danych skonfigurowanych za pomocą TIA Portal:
  - Całe urządzenia (sterownik, urządzenie HMI, napęd, itp.)
  - Programy, bloki danych, zmienne, tabele monitorujące
  - Obrazy HMI, zmienne HMI, skrypty itp.
- Łatwa wymiana danych pomiędzy projektami
- Aktualizacja wszystkich elementów danej biblioteki
- Wersjonowanie elementów biblioteki

  Mniej błędów podczas korzystania z bloków sterujących poprzez systemowe uwzględnienie wzajemnych zależności

Zalecenia

- Pamiętaj, aby tworzyć kopie zapasowe bloków, konfiguracji sprzętowych, obrazów HMI itp.
- Pamiętaj, aby tworzyć takie typy elementów aby można było wykorzystać daną bibliotekę w obrębie całego program użytkownika:
  - Wersjonowanie bloków
  - Aktualizacja we wszystkich obszarach użytkowania
- Globalna biblioteka może pełnić funkcję biblioteki centralnej , z której kilka osób korzysta w tym samym czasie np. pracując nad wspólnym projektem, lub do przesyłania danych pomiędzy użytkownikami.
- Skonfiguruj ścieżkę zapisu biblioteki tak, aby otwierała się automatycznie po uruchomieniu TIA Portal.

  Więcej informacji można znaleźć na stronie:
  http://support.automation.siemens.com/WW/view/en/100451450

Wskazówka Więcej informacji można znaleźć w poniższych pozycjach:

Jak otworzyć bibliotekę tylko do odczytu STEP 7 (TIA Portal)?
http://support.automation.siemens.com/WW/view/en/37364723

3.7.1 Rodzaje bibliotek oraz dane w nich przechowywane

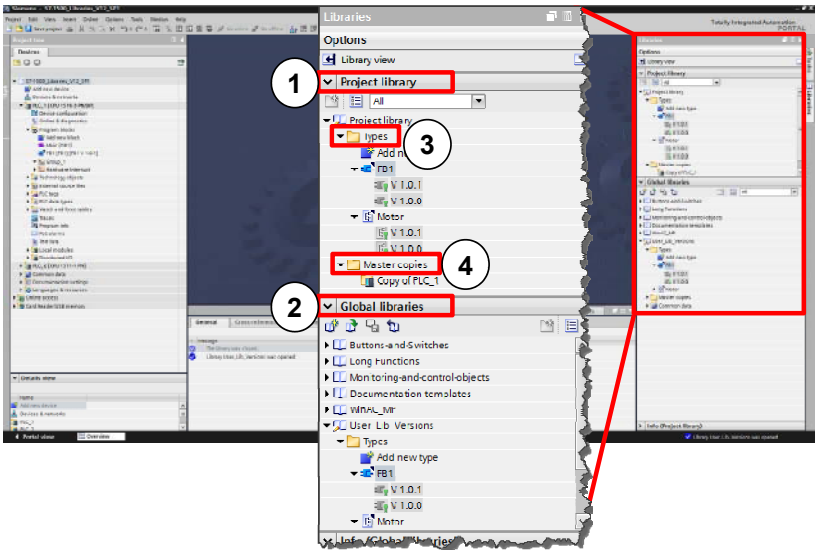
Rozróżniamy dwa typy bibliotek:

- Biblioteka projektu „Project library”
- Biblioteka globalna „Global library”

Zawartość tych bibliotek można podzielić na:

- Typy „Types"
- Kopie zapasowe „Master Copies”

Rysunek 3-35: Biblioteki w TIA Portal



- (1) Biblioteka projektu „Project library"
  - Zintegrowana i zarządzana w obrębie projektu
  - Może być użyta wielokrotnie w obrębie danego projektu
- (2) Biblioteka globalna „Global library"
  - Biblioteka niezależna
  - Można ją wykorzystać w różnych projektach

Elementy przechowywane w bibliotekach dzielą się na dwa typy:

- (3) Kopie zapasowe „Master copies”
  - Kopie konfiguracji (np. bloków, sprzętu, tabeli ze zmiennymi PLC, itp.)
  - Kopie te nie są powiązane z ich plikami źródłowymi
  - Kopie zapasowe mogą składać się z kilku elementów.
- (4) Typy „Types"
  - Typy są powiązane z obszarem występowania w projekcie. Zmiana typu powoduje jego automatyczną aktualizację w obrębie projektu.
  - Obsługiwane typy: bloki sterownika (FC, FB), dane PLC, obrazy i kontrolki HMI, HMI UDT, skrypty).
  - Elementy podrzędne automatycznie otrzymują typ elementu nadrzędnego
  - Typy można zmienić poprzez stworzenie nowszej wersji
  - Sterownik może korzystać jedynie z jednej wersji danego typu.

3.7.2 Wykorzystanie Typów

Typy umożliwiają tworzenie uniwersalnych funkcji, które można zastosować w różnych zakładach oraz urządzeniach. Ponadto, wszelkie zmiany np. wersji, czy też aktualizacja elementów tego samego typu mogą być przeprowadzone z poziomu biblioteki programu użytkownika.

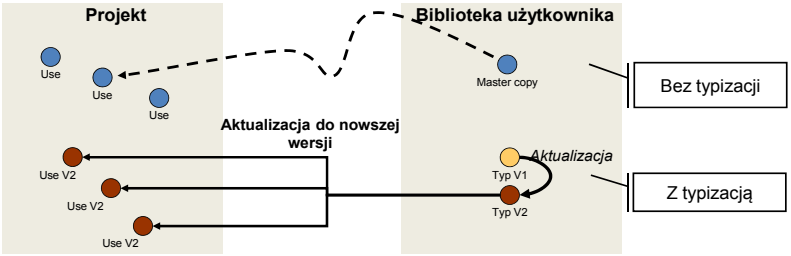
Zalety

- Aktualizacja we wszystkich obszarach danego projektu.
- Ochrona przed zmianą pojedynczego wystąpienia danego typu
- Systemowa gwarancja zgodności – jakiekolwiek zmiany dotyczą wszystkich elementów danego typu, niezależnie od ich lokalizacji
- W przypadku usunięcia danego typu, znika on ze wszystkich obszarów programu użytkownika.

Właściwości

Podział na typy ułatwia wprowadzanie zmian w obrębie całego projektu.

Rysunek 3-36: Typizacja z wykorzystaniem biblioteki użytkownika



- Typy są zawsze oznaczone kolorem, aby ułatwić ich rozpoznanie.

3.7.3 Różnice w typizacji między CPU a HMI

Elementy podlegające typizacji różnią na poziomie systemowym w przypadku sterowników oraz urządzeń HMI.

Tabela 3-8: Różnice systemowe w przypadku sterowników oraz urządzeń HMI

Sterownik	HMI
Podrzędne elementy sterujące podlegają typizacji.	Podrzędne elementy HMI nie podlegają typizacji.
Podrzędne element sterujące mogą występować pojedynczo.	Podrzędne elementy HMI nie mogą występować pojedynczo.
Edycja elementów sterujących odbywa się w **środowisku testowym.**	Edycja obrazów HMI oraz skryptów odbywa się w środowisku testowym. Edycja kontrolek oraz HMI UDT odbywa się bezpośrednio w bibliotece **(brak środowiska testowego)**.

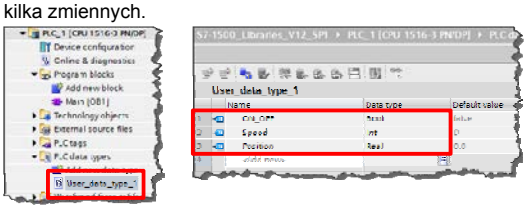
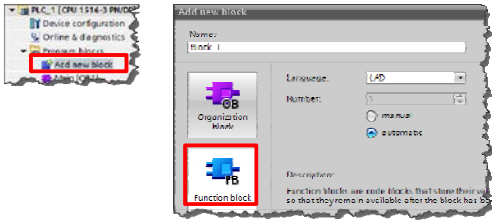
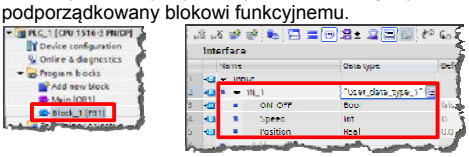
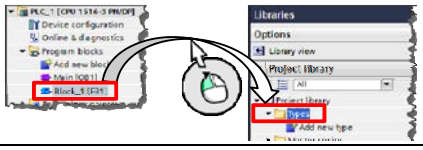
Więcej informacji na temat obsługiwania bibliotek można znaleźć w poniższych przykładach.

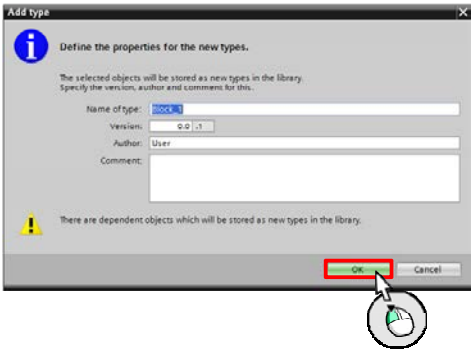
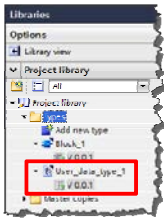
3.7.4 Wersjonowanie bloków

Przykład: Stworzenie nowego typu

Poniższy przykład przedstawia podstawowe funkcjonalności bibliotek podczas pracy z typami.

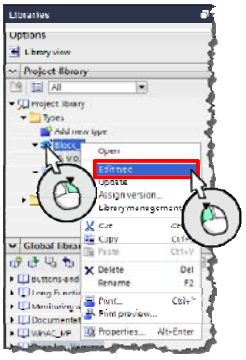
Tabela 3-9: Tworzenie nowego typu

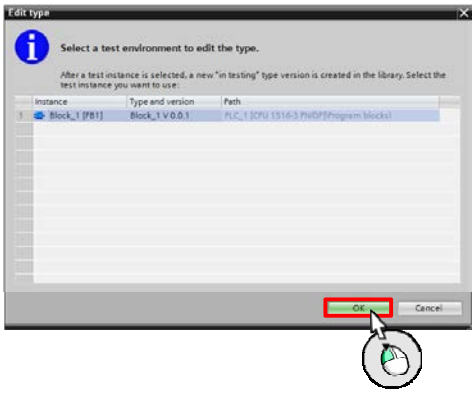
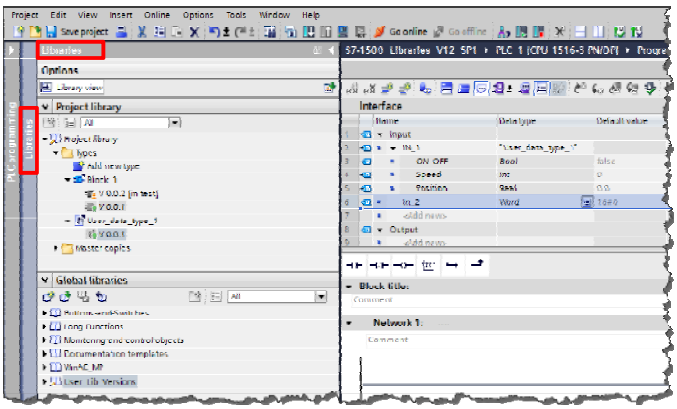
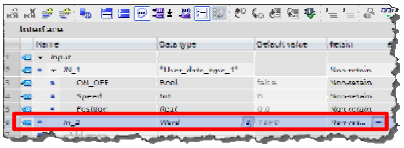
Krok	Polecenie
1.	Utwórz nowy typ danych PLC wybierając „Add new data type”, a następnie utwórz kilka zmiennych. 
2.	Utwórz nowy blok funkcyjny wybierając „Add new Block”. Blok ten będzie typem nadrzędnym. 
3.	Zdefiniuj zmienną wejściową dla stworzonego typu danych. Typ danych PLC jest podporządkowany blokowi funkcyjnemu. 
4.	Przeciągnij blok funkcyjny do folderu „Types” w bibliotece projektu. 

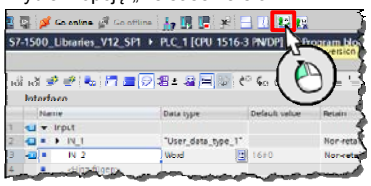
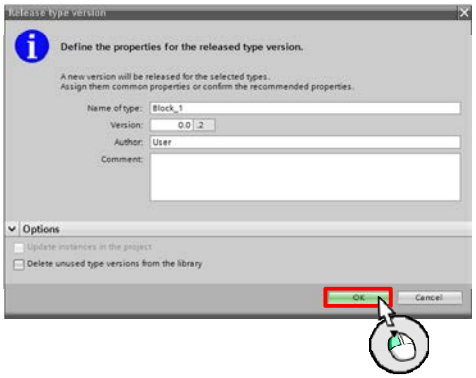
Krok	Polecenie
5.	Możesz również nadać nowemu typowi nazwę, wersję, autora oraz komentarz, a następnie potwierdzić „OK”. 
6.	Stworzony typ podporządkowany zostaje również automatycznie zapisany w bibliotece. 

Przykład: Zmiana typu danych

Tabela 3-10: Zmiana typu danych

Krok	Polecenie
1.	W bibliotece projektu „Project library” kliknij prawym przyciskiem na blok, a następnie wybierz „Edit type” 

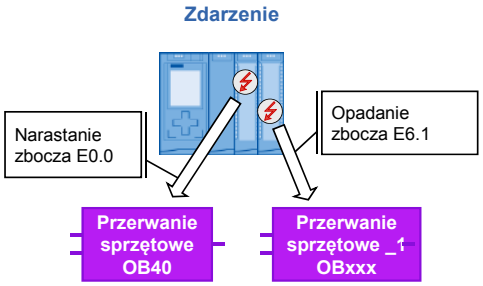
Krok	Polecenie
2.	Następnie wybierz sterownik, który będzie pełnił funkcję środowiska testowego I potwierdź „OK”.  Jeżeli wybrany blok jest wykorzystywany przez kilka sterowników, należy określić, ten, który będzie pełnił funkcję środowiska testowego.
3.	Pojawi się okno bibliotek „Libraries”, a w nim nowo utworzony blok z oznaczeniem „in test”. 
4.	Dodaj kolejną zmienną wejściową.  Na tym etapie można sprawdzić blok poprzez wgranie projektu do sterownika. Jeżeli skończyłeś testowanie bloku, przejdź do kolejnego kroku.

Krok	Polecenie
5.	Wybierz opcję „Release version”. 
6.	Pojawi się okno dialogowe, w którym można dodać stosowny komentarz. Aby potwierdzić, kliknij „OK”.  Jeżeli blok jest wykorzystywany w różnych częściach programu oraz przez różne sterowniki, możesz aktualizować go we wszystkich tych obszarach zaznaczając pole „Update instances in the project”. Starsze oraz niepotrzebne wersje można usunąć z biblioteki zaznaczając pole “Delete unused type versions from library”
7.	Wyjdź z biblioteki, wybierając „Close library view” 

3.8 Zwiększanie wydajności za pomocą przerwań procesowych.

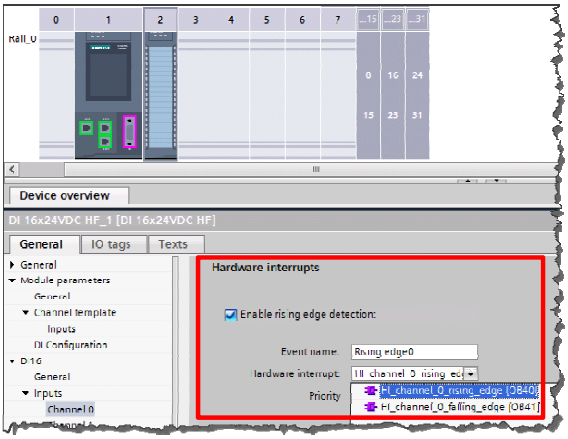
Przerwania procesowe to szybka, zaprogramowana reakcja sterownika na określone zdarzenia sprzętowe takie jak np. narastające zbocze sygnału na wejściu cyfrowym. Każde przerwanie procesowe wymaga zaprogramowania osobnego bloku organizacyjnego (OB.) W przypadku wystąpienia określonego zdarzenia system operacyjny wywoła blok organizacyjny, przerywając tym samym wykonywanie programu. Program zostanie wznowiony po przetworzeniu bloku organizacyjnego.

Rysunek 3-37: Wywołanie bloku organizacyjnego (OB) dla przerwania procesowego



Poniższy rysunek przedstawia proces konfiguracji przerwania sprzętowego dla modułu wejść cyfrowych.

Rysunek 3-38: Konfiguracja przerwania sprzętowego



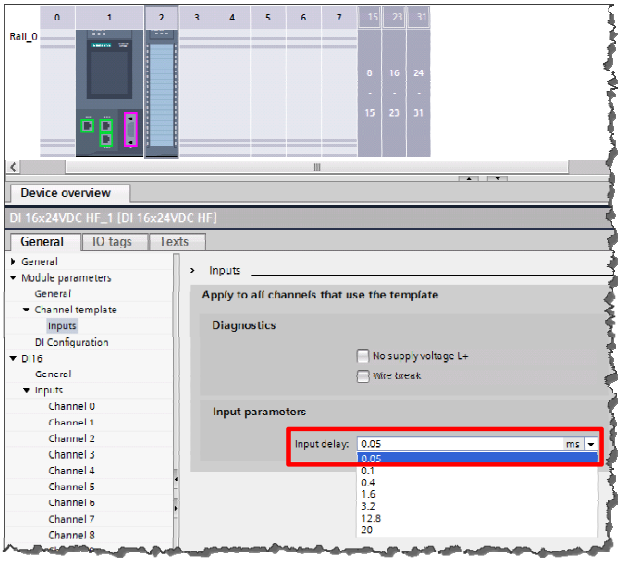
Zalety

- Szybka reakcja na zaprogramowane zdarzenia (np. narastanie lub opadanie zbocza)
- Każde zdarzenie powoduje wywołanie osobnego bloku organizacyjnego (OB).

Zalecenia

- Przygotuj się na wypadek zdarzeń sprzętowych programując przerwania procesowe.
- Przerwanie sprzętowe jest inicjowane dopiero po upłynięciu zadanego czasu, który można ustawić dla parametru „input delay”. Jeżeli uznasz, że reakcja nie jest wystarczająco szybka zmniejsz jego wartość.
- Zadany czas opóźnienia służy jako filtr sprzętowy, który pozwala wyeliminować błędy powstałe np. w wyniku chwilowego drgania styków.

Rysunek 3-39: Ustawianie parametru „input delay”



3.9 Inne zalecenia

Poniższe zalecenia pomogą przyspieszyć przetwarzanie programu.

Zalecenia

Jeżeli chcesz zwiększyć wydajność sterowników S7-1200/1500 zwróć uwagę na poniższe zalecenia:

- Dla języków LAD/FBD: Wyłącz opcję „generate ENO”, aby uniknąć przeprowadzanie testów podczas wykonywania programu.
- Dla języka STL: Nie korzystaj z rejestrów. Rejestry adresowe oraz rejestry danych istnieją jedynie po to, aby zagwarantować kompatybilność ze starszymi wersjami.

Wskazówka

Więcej informacji można znaleźć w poniższych pozycjach:

Jak dezaktywować wyjście ENO dla danego polecenia ?
http://support.automation.siemens.com/WW/view/en/67797146

Jak zwiększyć wydajność STEP 7 (TIA Portal) oraz jednostek centralnych S7-1200/S7-1500
http://support.automation.siemens.com/WW/view/en/37571372

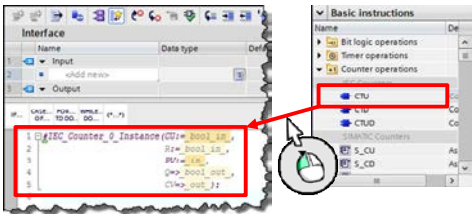
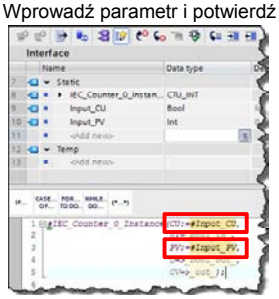
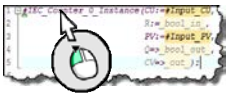
3.10 Środowisko SCL: Tips and tricks

3.10.1 Korzystanie z szablonów „call template”

Wiele poleceń posiada gotowe szablony, wraz z listą możliwych parametrów formalnych.

Przykład

Tabela 3-11: Korzystanie z szablonów oraz ich modyfikowanie

Krok	Polecenie
1.	Przeciągnij polecenie z biblioteki do edytora programu. Pojawi się pełny szablon dla tego polecenia. 
2.	Wprowadź parametr i potwierdź wciskając „Enter” 
3.	Edytor automatycznie zwija szablon. 
4.	Jeżeli chcesz edytować szablon, postępuj zgodnie z poniższym opisem. Kliknij w dowolnym miejscu szablonu. Następnie wciśnij “**CTRL+SHIFT+SPACE**”. Przejdziesz do trybu edycji; szablon zostanie ponownie rozwinięty. Poruszaj się między parametrami za pomocą przyciska „TAB” 
5.	Uwaga: Wszystkie pozycje szablonu pisane są kursywą.

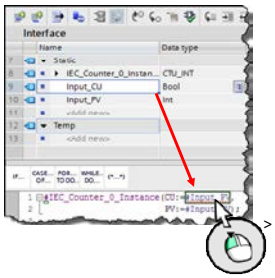
3.10.2 Parametry edytowalne

Edytując szablon bardzo łatwo rozpoznać, które parametry można edytować, a które nie. Parametry nieedytowalne są przyciemnione.

3.10.3 Podmienianie zmiennych za pomocą funkcji Drag & drop

Edytor języka SCL obsługuje funkcję drag & drop. Jeżeli chcesz podmienić zmienną na inną, postępuj zgodnie z poniższym opisem.

Tabela 3-12: Podmiana zmiennych za pomocą funkcji Drag & drop

Krok	Polecenie
1.	Przeciągnij nową zmienną na miejsce starej, a następnie ją upuść (po upływie ponad 1 sek)  Zmienna została skutecznie podmieniona.

3.10.4 Wstawianie poleceń CASE

W przypadku poleceń CASE, jeżeli dany blok spełni zadany warunek, to nastąpi automatyczny przeskok do tego bloku. Następnie instrukcja z tego bloku zostanie wykonana.

Polecenie CASE pozwala łatwo i dokładnie sprawdzić najczęściej wykorzystywane zakresy wartości.

Example

```
CASE #myVar OF
        5:
                FC5(#myParam);
        10,12:
                FC10(#myParam);
        15:
                FC15(#myParam);
        0..20:
                FCGlobal(#myParam);
// FCGlobal is never called for the values 5, 10, 12 or 15!
        ELSE
END_CASE;
```

Wskazówka Polecenie CASE można również stosować dla danych typu STRING, CHAR oraz z danymi typu VARIANT (sprawdź w rozdziale 2.8.5 Dane typu VARIANT (wyłącznie dla S7-1500)).

3.10.5 Pętle FOR oraz licznik przejścia (bez możliwości modyfikacji)

W języku SCL ilość iteracji pętli FOR jest określana w momencie wejścia do pętli. Nie można modyfikować wartości licznika wewnątrz pętli FOR w trakcie jej wykonywania.

Pętla można zostać przerwana w dowolnym czasie za pomocą polecenia EXIT

Zalety

- Kompilator nie zna ilości iteracji, przez co jest w stanie lepiej zoptymalizować program.

Przykład

```
FOR #var := #lower TO #upper DO
  #var := #var + 1; // brak efektu, ostrzeżenie kompilatora
END_FOR;
```

3.10.6 Dekrementacja pętli FOR

Język SCL dopuszcza dekrementację indeksu pętli, poprzez wprowadzenie liczby ujemnej przy „BY” do nagłówka pętli.

Przykład

```
FOR #var := #upper TO #lower BY -2 DO

END_FOR;
```

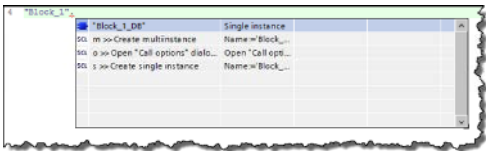
Jeżeli liczba przy “BY” równa jest -2 jak podaje przykład, wartość licznika zostaje zmniejszona o 2 przy każdej iteracji. Jeżeli pominiesz „BY” wartość licznika zostaje domyślnie zwiększona o 1.

3.10.7 Tworzenie instancji z poziomu klawiatury

Jeżeli wolisz pracować z poziomu klawiatury, możesz skorzystać ze skrótów do tworzenia nowych instancji.

Przykład

Tabela 3-13: Tworzenie instancji z poziomu klawiatury

Krok	Polecenie
1.	Wprowadź nazwę bloku zakończoną kropką „.” Pojawi się poniższe okno. 
2.	Aby dodać nową instancję lubi multi-instancję wciśnij odpowiednio klawisz „s” lub „m”.

3.10.8 Zmienne typu Time (czas)

Język SCL pozwala obliczać odstęp czasu, korzystając ze zwykłych działań arytmetycznych w miejsce funkcji typu. T_COMBINE. Procedura ta nosi nazwę „przeciążania operatorów” (overload of operands).

Przykład

```
TimeDifference := TimeStamp_1 - TimeStamp_2;
```

Poniższa tabela przedstawia listę przeciążanych operatorów z odpowiadającym im działaniom.

Tabela 3-14: Przeciążane operatory w języku SCL

Przeciążane operatory	Działanie
ltime + time	T_ADD LTime
ltime + time	T_SUB LTime
ltime + lint	T_ADD LTime
ltime + lint	T_SUB LTime
time + time	T_ADD Time
time + time	T_SUB Time
time + dint	T_ADD Time
time + dint	T_SUB Time
ldt + ltime	T_ADD LDT / LTime
ldt + ltime	T_ADD LDT / LTime
ldt + time	T_ADD LDT / Time
ldt + time	T_SUB LDT / Time
dtl + ltime	T_ADD DTL / LTime
dtl + ltime	T_SUB DTL / LTime
dtl + time	T_ADD DTL / Time
dtl + time	T_SUB DTL / Time
ltod + ltime	T_ADD LTOD / LTime
ltod + ltime	T_SUB LTOD / LTime
ltod + lint	T_ADD LTOD / LTime
ltod + lint	T_SUB LTOD / LTime
ltod + time	T_ADD LTOD / Time
ltod + time	T_SUB LTOD / Time
tod + time	T_ADD TOD / Time
tod + time	T_SUB TOD / Time
tod + dint	T_ADD TOD / Time
tod + dint	T_SUB TOD / Time
dt + time	T_ADD DT / Time
dt + time	T_SUB DT / Time
ldt – ldt	T_DIFF LDT
dtl – dtl	T_DIFF DTL
dt – dt	T_DIFF DT
date – date	T_DIFF DATE
ltod – ltod	T_DIFF LTOD
date + ltod	T_COMBINE DATE / LTOD
date + tod	T_COMBINE DATE / TOD

4 Programowanie niezależne od sprzętu

Jeżeli chcesz wykorzystać dany blok w różnych sterownikach bez dodatkowych modyfikacji to pamiętaj, aby unikać rozwiązań dedykowanych wyłącznie dla danego sprzętu.

4.1 Typy danych obsługiwane przez sterowniki S7-300/400 oraz S7-1200/1500

Poniższa tabela zawiera listę podstawowych typów oraz grup danych.

Zalecenia

- Używaj wyłącznie typów danych obsługiwanych przez dany sterownik

Tabela 4-1: Typy danych zgodne z normą EN 61131-3

	Opis	S7 - 300/400	S7-1200	S7-1500
Dane bitowe	• BOOL • BYTE • WORD • DWORD	✓	✓	✓
	• LWORD	×	×	✓
Znaki	• CHAR (8 bit)	✓	✓	✓
Dane numeryczne	• INT (16 bit) • DINT (32 bit) • REAL (32 bit)	✓	✓	✓
	• SINT (8 bit) • USINT (8 bit) • UINT (16 bit) • UDINT (32 bit) • LREAL (64 bit)	×	✓	✓
	• LINT (64 bit) • ULINT (64 bit)	×	×	✓
Dane czasowe	• TIME • DATE • TIME_OF_DAY	✓	✓	✓
	• S5TIME	✓	×	✓
	• LTIME • L_TIME_OF_DAY	×	×	✓

Tabela 4-2: Grupy danych składające się z różnych typów

	Opis	S7 - 300/400	S7-1200	S7-1500
Dane czasowe	• DT (DATE_AND_TIME)	✓	×	✓
	• DTL	×	✓	✓
	• LDT (L_DATE_AND_TIME)	×	×	✓
Symbole	• STRING	✓	✓	✓
Pola	• ARRAY	✓	✓	✓[1)]
Struktury	• STRUCT	✓	✓	✓

[1)] W przypadku S7-1500 dane typu ARRAY są ograniczone do 64 bitów zamiast 16

Tabela 4-3 Typy parametrów formalnych, przenoszonych pomiędzy blokami

	Opis	S7 - 300/400	S7-1200	S7-1500
Wskaźnik	• POINTER • ANY	✓	×	✓[1)]
	• VARIANT	×	✓	✓
Bloki	• TIMER • COUNTER	✓	✓[2)]	✓
	• BLOCK_FB • BLOCK_FC	✓	×	✓
	• BLOCK_DB • BLOCK_SDB	✓	×	×
	• VOID	✓	✓	✓
Dane PLC	• Dane PLC	✓	✓	✓

[1)] Dostęp zoptymalizowany możliwy tylko w przypadku adresowania symbolicznego

[2)] Dla S7-1200/1500 dane typu TIMER oraz COUNTER są reprezentowane przez IEC_TIMER oraz IEC_Counter.

4.2 Brak pamięci bitowej / globalne bloki danych

Zalety

- Zoptymalizowane bloki danych są o wiele bardziej wydajne niż obszar adresowy pamięci bitowej (niezoptymalizowany ze względu na kompatybilność).

Zalecenia

- Pamięć bitowa (również bity systemowe oraz zegar pamięci) przysparza wielu problemów, ze względu na to, że każdy sterownik posiada obszar adresowy pamięci bitowej o innej wielkości. Dlatego lepszym rozwiązaniem są globalne bloki danych, które sprawiają, że program jest bardziej uniwersalny.

4.3 Programowanie „bitów zegarowych”

Zalecenia

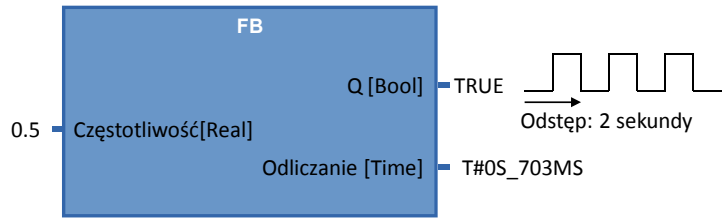
Zaprogramowanie bitów zegarowych wymaga poprawnej konfiguracji sprzętowej.

Zaprogramowany blok może pełnić funkcję generatora zegarowego. Poniższy przykład przedstawia jak go zaprogramować w języku SCL.

Przykład

Blok posiada następujące funkcje: zadaną częstotliwość, wartość Boolean dla wyjścia „Q”, która zmienia się zgodnie z zadaną częstotliwością oraz wyjście „Odliczanie”, które wyświetla pozostały czas do zmiany wartości wyjścia „Q”.

Jeżeli zadana częstotliwość jest mniejsza lub równa 0.0, wtenczas wyjście Q = FALSE a Odliczanie = 0.0.



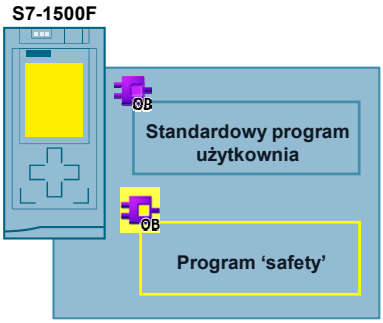
Wskazówka Cały przykład jest dostępny na stronie:
http://support.automation.siemens.com/WW/view/en/87507915

5 STEP 7 Safety w TIA Portal

5.1 Wstęp

Sterowniki S7-1500F w wykonaniu failsafe obsługiwane są w środowisku projektowym TIA Portal, począwszy od wersji V13. Mogą one wykonywać zarówno standardowy program użytkownika, jak i program kładący szczególny nacisk na bezpieczeństwo - program safety. Program safety można zaprogramować za pomocą narzędzia projektowego SIMATIC STEP 7 Safety (TIA Portal)

Rysunek 5-1: Program standardowy oraz safety



Zalety

- Stworzenie jednolitego programu, zarówno standardowego jak i safety w środowisku projektowym TIA Portal.
- Możliwość programowania w językach LAD oraz FBD.
- Jednolite opcje diagnostyczne oraz funkcje online.

Wskazówka

Fail-safe nie oznacza, że program nie zawiera błędów. Programista jest nadal odpowiedzialny za napisanie logicznego programu.

Fail-safe oznacza jedynie, że sterownik poprawnie wykona program safety.

Wskazówka

Więcej informacji na temat bezpieczeństwa (np. zasady działania programów ‘safety’) można znaleźć w poniższych pozycjach:

TIA Portal – Przegląd najważniejszych dokumentów dotyczących bezpieczeństwa

http://support.automation.siemens.com/WW/view/en/90939626

Aplikacje & Narzędzia – Safety Integrated
http://support.automation.siemens.com/WW/view/en/20810941/136000

STEP 7 Safety (TIA Portal) - Podręczniki
http://support.automation.siemens.com/WW/view/en/49368678/133300

5.2 Pojęcia

Poniższe pojęcia pojawiają się wielokrotnie w kontekście bezpieczeństwa.

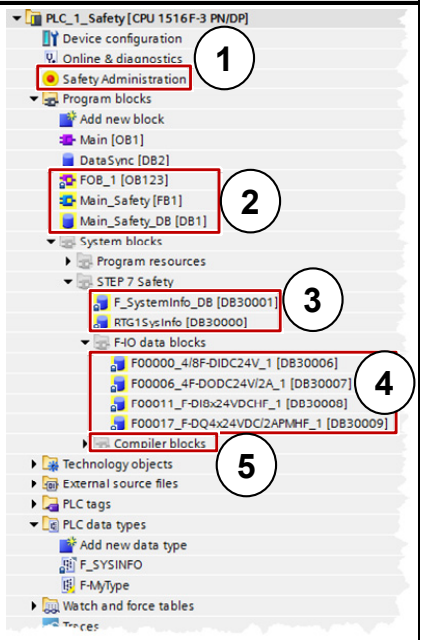
Tabela 5-1: Pojęcia związane z bezpieczeństwem

Pojęcie	Opis
Standardowy program użytkownika	Część programu użytkownika, która nie została zaprogramowana w trybie fail-safe. (F)
Program ‘safety’ F	Program ‘safety’ to cześć programu użytkownika, która jest wykonywana niezależnie od głównego programu sterownika. Bloki oraz polecenia fail-safe są podświetlone na żółto, co ułatwia ich rozpoznanie (np. w oknie nawigacji projektu).Parametry (fail-safe) F-CPU oraz F-wejji/wyj. oznaczone są w podobny sposób.

5.3 Elementy programu safety

Program safety zawsze składa się z F-bloków (systemowych lub użytkownika), oraz edytora „Safety Administration”.

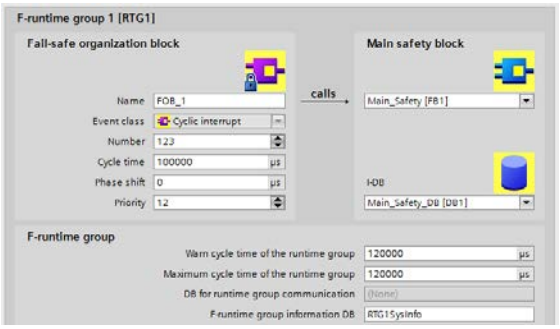
Tabela 5-2: Elementy programu safety

Opis	Widok
1. Edytor „Safety administration” - Status programu bezpiecznego - Zbiorowa sygnatura F - Stan operacji - Tworzenie/organizowanie grup F-runtime - Informacje o blokach F - Informacje o danych PLC zgodnych z F - Określanie/zmiana ochrony dostępu 2. F bloki użytkownika 3. F bloki runtime – systemowe - Zawierają informacje o stanie grupy F run-time. 4. Bloki danych F-wej/wyj -systemowe - Zawierają zmienne określające moduły F 5. „Bloki kompilatora” –systemowe bloki weryfikacyjne. - Działają w tle; odpowiadają za poprawne wykonanie programu safety. - Nie mogą być modyfikowane przez użytkownika	

5.4 Grupa F-runtime

Program safety jest zawsze wykonywany w obrębie grupy F-runtime, o zdefiniowanym cyklu. Grupa F-runtime składa się z bloku organizacyjnego typu fail-safe (Fail-safe organization block), który wywołuje główny blok safety (Main safety block). Blok ten odpowiada za wywoływanie wszystkich funkcji bezpieczeństwa użytkownika.

Rysunek 5-2: Grupa F-runtime w edytorze „Safety administration”



Zalety

- Wygodne tworzenie oraz konfiguracja grup runtime w edytorze „Safety Administration”.
- Automatyczne tworzenie F-bloków w obrębie grupy F-runtime.

Właściwości

- Można stworzyć maksymalnie dwie grupy F-runtime

5.5 Sygnatura F

Każdy element typu fail-safe oznaczony jest specjalną sygnaturą F (np. F-bloki, F-wej/wyj itp.), która pomaga określić czy elementy te są zawsze zgodne z pierwotną konfiguracją.

Zalety

- Szybkie i wygodne porównanie bloków oraz konfiguracji urządzeń failsafe (F)

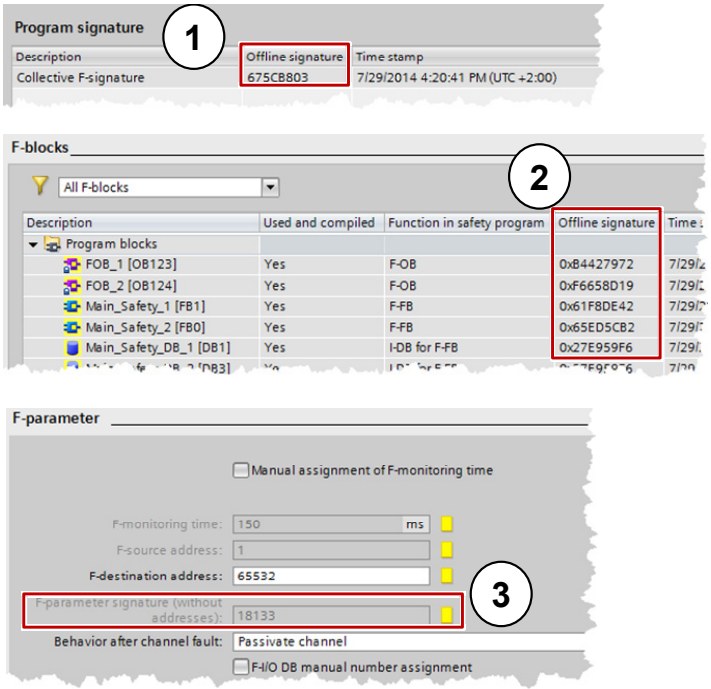
Właściwości

- Sygnatura parametrów F (bez adresów F-wej/wyj)
  - Może zostać zmieniona modyfikację parametrów.
  - Pozostaje bez zmian podczas zmiany adresu PROFIsafe w przeciwieństwie do kolektywnej sygnatury F stacji.
- Sygnatura bloków F zmienia się wyłącznie w przypadku zmiany logiki tego bloku.

- Sygnatura bloku F pozostaje bez zmian w przypadku:
  - Zmiany numeracji bloku,
  - Zmiany interfejsu bloku,
  - Zmiany wersji bloku.

Przykład

Rysunek 5-3: Przykłady sygnatur F



1. Wspólna sygnatura F danej stacji w edytorze „Safety administration”.
2. Sygnatury bloków F w edytorze „Safety administration” (widoczna również we właściwościach bloku)
3. Sygnatura parametrów F w widoku „Device view” w zakładce „Devices & Networks”

Wskazówka Sygnatura F sterowników S7-1500F podawana jest bezpośrednio na wyświetlaczu lub zintegrowanym web serwerze.

5.6 Przypisywanie adresów PROFIsafe dla F-wej/wyj.

Każdy moduł F-wej/wyj posiada własny adres PROFIsafe do komunikowania się ze sterownikiem F. Adres ten można przypisać na dwa różne sposoby.

Tabela 5-3: Przypisywanie adresu F

ET 200M / ET 200S (adres PROFIsafe typ 1)	ET 200MP / ET 200SP (adres PROFIsafe typ 2)
Adres PROFIsafe można przypisać bezpośrednio na modułach za pomocą przełącznika DIL. Adres PROFIsafe ustawiony za pomocą przełącznika DIL musi być taki sam jak w oknie konfiguracji modułu w TIA Portal.	Adres PROFIsafe można przypisać wyłącznie za pośrednictwem TIA Portal Skonfigurowany adres PROFIsafe zostaje wgrany do inteligentnego elementu kodującego danego modułu.

Zalety

- Wymiana modułu F nie wymaga ponownego przypisania adresu PROFIsafe dla ET 200MP oraz ET 200SP. Inteligentny element kodujący pozostaje częścią BaseUnit podczas wymiany modułu.
- Łatwa konfiguracja w TIA Portal, który informuje o niewłaściwym przypisaniu adresu PROFIsafe.
- Przypisanie adresu PROFIsafe wszystkim modułom F jednocześnie w obrębie tego samego systemu ET 200SP.

Wskazówka Więcej informacji na temat przypisywania adresu PROFIsafe dla modułów F-wej/wyj można znaleźć w podręczniku:

SIMATIC Industrial Software SIMATIC Safety – Configuring and Programming
http://support.automation.siemens.com/WW/view/en/54110126

5.7 Status modułów F-wej/wyj

Informacje dot. statusu poszczególnych modułów F-wej/wyj. są zapisywane w F-blokach tych modułów. Informacje te można następnie wywołać, ocenić oraz przetworzyć w programie safety.

Istnieją jednak różnice pomiędzy sterownikami S7-1500F a S7-300F/400F.

Tabela 5-4: Zmienne w blokach danych (DB) F-wej/wyj S7-300F/400F oraz S7-1500F

Zmienne w blokach danych F-wej/wyj lub status wartości w PAE	F-wej/wyj dla S7-300/400F	F-wej/wyj dla S7-1500F
ACK_NEC	✓	✓
QBAD	✓	✓
PASS_OUT	✓	✓
QBAD_I_xx *	✓	✗
QBAD_O_xx *	✓	✗

Zmienne w blokach danych F-wej/wyj lub status wartości w PAE	F-wej/wyj dla S7-300/400F	F-wej/wyj dla S7-1500F
Status wartości	×	✓

*Zmienne QBAD_I_xx oraz QBAD_O_xx informują o tym czy wartość kanału jest poprawna. W S7-300/400F posiadają wartość przeciwną niż zmienna „status wartości" sterowników t S7-1500F (więcej informacji znajduje się w kolejnym rozdziale).

5.8 Status wartości (S7-1500 F)

Oprócz komunikatów diagnostycznych oraz informacji o stanie urządzenia, moduł F informuje również czy wartość danego sygnału wejściowego i wyjściowego jest poprawna (status wartości). Status wartości zapisywany jest podobnie jak sygnał wejściowy – w obrazie procesu:

Status wartości:

- 1: wygenerowana została wartość prawidłowa dla kanału
- 0: wygenerowana została wartość zastępcza dla danego kanału

Tabela 5-5: Różnice pomiędzy Q_BAD (S7-300F/400F) a statusem wartości (S7-1500F)

Sytuacja	QBAD (S7-300F/400F)	Status wartości (S7-1500F)
Prawidłowe wartości dla F-wej/wyj (błąd nie wystąpił)	FAŁSZ	PRAWDA
Wystąpił błąd kanału	PRAWDA	FAŁSZ
Wymagane potwierdzenie usunięcia błędu (ACK_REQ)	PRAWDA	FAŁSZ
Potwierdzenie usunięcia błędu (ACK_REI)	FAŁSZ	PRAWDA

Właściwości

- Status wartości jest wprowadzany do obrazu procesu wejść oraz wyjść.
- Wartość kanału oraz stan wartości F wej/wyj. musi być wywoływany przez tę samą grupę F run-time.

Zalecenia

- Aby ułatwić odczyt, korzystaj z końcówki "`_VS`", jako nazwy symbolicznej statusu wartości np. "`Tag_In_1_VS`"

Przykład

Rozmieszczenie bitów statusu wartości w obrazie procesu, na przykładzie modułu F-DI 8x24VDC HF.

Tabela 5-6: Bity statusu wartości w obrazie procesu na przykładzie modułu F-DI 8x24VDC HF

Bajt w F-CPU	Bity przypisane w F-CPU							
	7	6	5	4	3	2	1	0
x + 0	DI_7	DI_6	DI_5	DI_4	DI_3	DI_2	DI_1	DI_0
x +1	Status wartości dla DI_7	Status wartości dla DI_6	Status wartości dla DI_5	Status wartości dla DI_4	Status wartości dla DI_3	Status wartości dla DI_2	Status wartości dla DI_1	Status wartości dla DI_0

x = adres startowy modułu

Wskazówka Więcej informacji na temat status wartości modułów ET 200 SP można znaleźć w poniższych pozycjach:

Dokumentacja dla CPU typu Failsafe
http://support.automation.siemens.com/WW/view/en/87493352/133300

Dokumentacja dla Modułów wej/wyj typu Failsafe
http://support.automation.siemens.com/WW/view/en/55684717/133300

5.9 Typy danych

Programy safety mogą korzystać z danych różnego typu.

Tabela: 5-7: Dane integer

Typ	Rozmiar	Zakres wartości
BOOL	1 Bit	0 .. 1
INT	16 Bitów	-32.768 .. 32.767
WORD	16 Bitów	-32.768 .. 65.535
DINT	32 Bity	-2.14 .. 2.14 mln
TIME	32 Bity	T#-24d20h31m23s648ms do T#+24d20h31m23s647ms

5.10 Dane PLC zgodne z sygnaturą F

Programy safety mogą również pracować z danymi PLC.

Zalety

- Modyfikacja danych PLC dotyczy wszystkich danych tego typu w obrębie programu użytkownika.

Właściwości

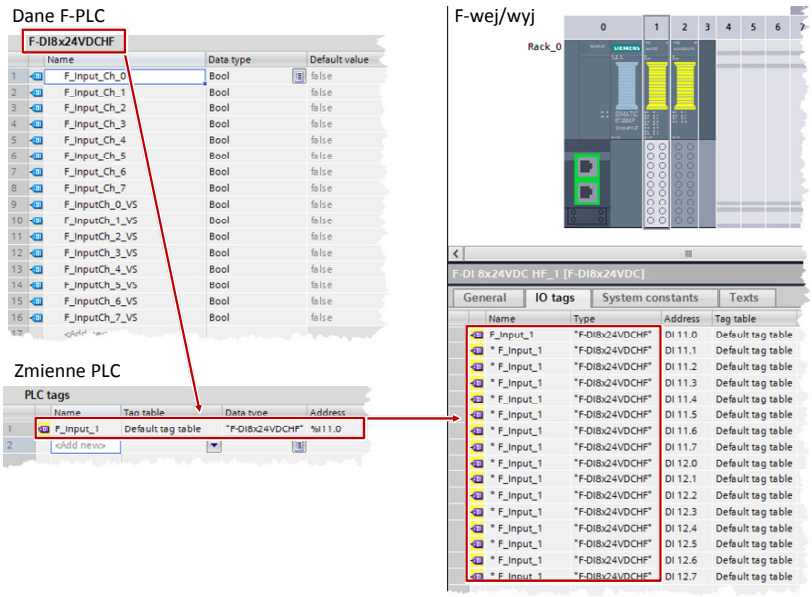
- Dane F-PLC deklaruje się i używa w ten sam sposób, co dane PLC.
- Dane F-PLC mogą pracować ze wszystkimi typami danych, dopuszczanymi przez program safety.
- Funkcja zagnieżdżania danych F-PLC w obrębie innych danych F-PLC nie jest obsługiwana.
- Dane F-PLC umożliwiają wykonanie standardowego programu użytkownika w zarówno w trybie safety jak i w trybie standardowym.

Zalecenia

- Możesz wykorzystać dane F-PLC, do uzyskania dostępu do obszarów wej/wyj. (sprawdź w rozdziale 3.6.4 Dostęp do obszarów wej/wyj za pomocą danych PLC)
- Zwróć uwagę na następujące zasady:
  - Struktura zmiennych typu F- PLC musi być taka sama jak struktura kanału modułu F-wej/wyj.
  - Dane F- PLC dla 8- kanałowego modułu F-wej/wyj to np:
    - Zmienne 8 BOOL (wartość kanału)
    - Zmienne 16 BOOL (wartość kanału + status wartości)
  - Dostęp do modułów F-wej/wyj. jest wyłącznie przez kanały aktywne. Podczas konfiguracji 1oo2 (2v2) wyższy kanał jest zawsze wyłączony.

Przykład

Rysunek 5-4: Dostęp do obszarów wej/wyj za pomocą danych F-PLC



5.11 PRAWDA/FAŁSZ (True/False)

Jeśli potrzebujesz użyć w programie Safety sygnału PRAWDA/FAŁSZ (True/False) możesz to zrobić na dwa sposoby:

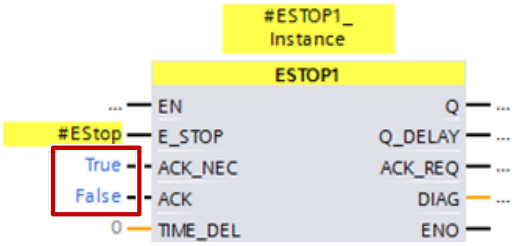
- Jako parametry aktualne przy blokach
- Przypisując je konkretnym akcjom (np. do wejść funkcji sterujących wyjściami safety odpowiadającym konkretnym działaniom systemu bezpieczeństwa)

Parametry aktualne przy blokach

Sterowniki S7-1500F wykorzystują zmienne typu Boolean („FAŁSZ” dla 0 oraz „PRAWDA” dla 1), jako parametry aktualne dla parametrów formalnych podczas wywoływania bloków w programie safety.

Jako parametr aktualny podajemy słowo „True” (PRAWDA=1) lub „False” (FAŁSZ=0).

Rysunek 5-5: Sygnały „PRAWDA” oraz „FAŁSZ”, jako parametry aktualne

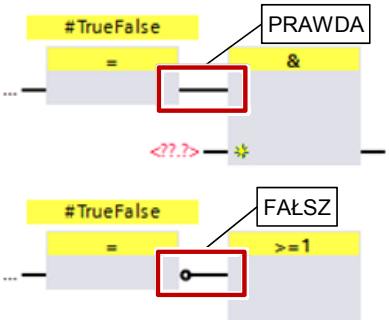


Przypisanie sygnałów „PRAWDA/FAŁSZ” do akcji safety

Aby przypisać sygnały „PRAWDA/FAŁSZ” do określonych akcji postępuj następująco:

- Korzystaj z języka programowania FBD.
- Utwórz fikcyjną zmienną typu BOOL- (tu: „#TrueFalse” – Prawda/Fałsz)
- Przypisz zmienną „#TrueFalse” do funkcji, np. porównania (=)
- Połącz wyjście funkcji porównania z wejściem innej funkcji
- W przypadku „#TrueFalse” = True – sygnał „PRAWDA”
- W przypadku „#TrueFalse” = False – sygnał „FALSE”.

Rysunek 5-6: Sygnały Prawda i Fałsz przypisane do działań



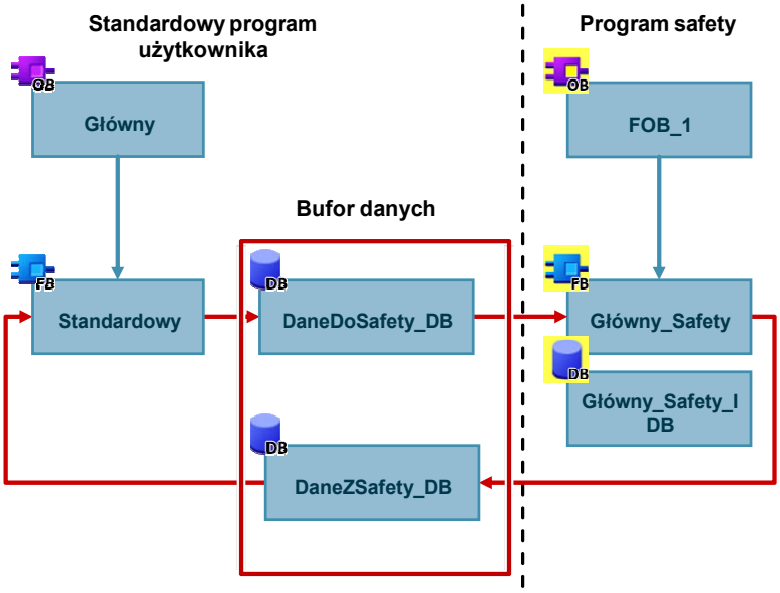
5.12 Wymiana danych pomiędzy programem standardowym a programem safety

W niektórych przypadkach zachodzi konieczność wymiany danych pomiędzy standardowym programem użytkownika, a programem safety. Zapoznaj się z poniższymi zaleceniami, aby dane obydwu programów zachowały spójność.

Zalecenia

- Nie używaj flag (znaczników „M”) do wymiany danych (sprawdź w rozdziale 4.2 Brak pamięci bitowej / globalne bloki danych
- Skorzystaj z dwóch standardowych bloków danych (DB) do wymiany danych pomiędzy standardowym programem użytkownika, a programem ‘safety’

Rysunek 5-7: Wymiana danych między standardowym programem użytkownika a programem safety



5.13 Testowanie programu safety

Następujące dane programu safety można zmodyfikować, jeżeli tryb bezpieczny został dezaktywowany:

- Obraz procesu F−wej/wyj
- F−DB (z wyjątkiem bloków dedykowanych komunikacji w grupie F-run-time), bloki danych bloków funkcyjnych (F-FB).
- Bloki danych F-wej/wyj.

Właściwości

- Kontrolowanie F−wej/wyj. jest tylko możliwie w trybie F−CPU RUN.
- Można obsługiwać maksymalnie 5 wejść/wyjść z poziomu watch table w programie safety
- Można korzystać z kilku watch table
- Punkt wyzwalania (trigger point) musi być ustawiony jako stały (permanent) lub jednorazowy (once) - na początku cyklu (cycle start) lub na końcu cyklu (cycle end).
- Funkcje wymuszania (Force) nie są dostępne dla F-wej/wyj.
- Ustawienie punktów zatrzymania (breakpoint'ów) w standardowym programie użytkownika może prowadzić do następujących błędów w programie safety:
  - Upłynięcie czasu monitorowania cyklu F
  - Błędy w komunikacji z modułem F-wej/wyj.
  - Błędy w komunikacji pomiędzy fail-safe CPU-CPU
  - Błąd wewnętrzny CPU
- Jeżeli nadal chcesz korzystać z punktów zatrzymania (breakpoint'ów) do celów testowych, musisz wpierw dezaktywować tryb bezpieczny. Może to spowodować następujące błędy:
  - Błąd podczas komunikowania się z modułem F-wej/wyj.
  - Błąd w komunikacji pomiędzy fail-safe CPU-CPU

5.14 Przejście do trybu STOP w przypadku błędów F

Poniższe zdarzenia mogą spowodować przejście do trybu STOP dla F-CPU:

- Usunięcie, dodanie lub modyfikacja bloków w folderze „System blocks"
- Jeżeli wyniki danego polecenia znajduje się poza dozwolonym obszarem dla tego typu danych (overflow). Przyczyna zdarzenia zostaje wprowadzona do buforu diagnostycznego F-CPU.
- Próba dostępu do bloków danych instance bloków funkcyjnych F-FB poza programem safety.
- Przekroczenie czasu cyklu dla grupy F-run-time („Maximal cycle time of the F-run-time group"). Określ ile czasu może upłynąć pomiędzy dwoma wywołaniami grupy F-runtime (maksymalnie 20,000 ms).
- Przetwarzanie grupy F-run-time, której zmienne mają zostać odczytane (główny blok safety grupy F-run-time).
- Edytowanie wartości początkowych w blokach danych instance bloków funkcyjnych F-FB, zarówno online jak i offline.
- Parametry znajdujące się w bloku safety.
- Wyjścia F-FC, które nie zostały zainicjalizowane.

5.15 Migrowanie zmiennych (tagów)

Informacje na temat migracji programów safety można znaleźć w poniższej pozycji:
http://support.automation.siemens.com/WW/view/en/21064024

5.16 Ogólne zalecenia dotyczące bezpieczeństwa

Poniższe zalecenia dotyczą obsługi oprogramowania STEP 7 Safety oraz modułów oznaczonych sygnaturą F.

- Użycie sterowników safety (oznaczonych sygnaturą F) ułatwi późniejsze rozszerzanie funkcji bezpieczeństwa i ich ew. modyfikację.
- Chroń program safety przed nieautoryzowanym dostępem ustawiając hasło w panelu „Safety administration”.

6 Najważniejsze zalecenia

- Korzystaj z bloków zoptymalizowanych
  - Rozdział 2.6 Bloki zoptymalizowane
- Korzystaj z danych typu VARIANT zamiast ANY
  - Rozdział 2.8.5 Dane typu VARIANT (wyłącznie dla S7-1500)
- Pamiętaj o przejrzystej budowie programu
  - Rozdział 3.2 Bloki organizacyjne (OB)
- Wstawiaj polecenia jako multi-instancjie (TON, TOF ..)
  - Rozdział 3.2.5 Multi-instancje
- Ponowne programowanie bloków
  - Rozdział 3.2.8 Bloki wielokrotnego użytku „reusable blocks”
- Programowanie symboliczne
  - Rozdział 3.6 Adresowanie symboliczne zamiast absolutnego
- Praca z tablicami
  - Rozdział 3.6.2 Dane typu ARRAY oraz pośrednie wywołanie obszaru
- Tworzenie danych typu PLC
  - Rozdział 3.6.3 Dane typu STRUCT oraz typy danych PLC
- Wykorzystywanie bibliotek do przechowywania elementów programowych
  - Rozdział 3.7 Biblioteki
- Wykorzystywanie globalnych bloków danych zamiast pamięci bitowej
  - Rozdział 4.2 Brak pamięci bitowej / globalne bloki danych

7 Dokumentacja

Tabela 7-1

	Temat	Link
\1\	Wsparcie online	http://support.automation.siemens.com
\2\	Podręcznik programowania S7-1200.S7-1500	http://support.automation.siemens.com/WW/view/en/81318674
\3\	Najważniejsza dokumentacja dlaTIA Portal	http://support.automation.siemens.com/WW/view/en/65601780
\4\	Podręczniki do STEP 7 (TIA Portal)	http://support.automation.siemens.com/WW/view/en/29156492/133300
\5\	Podręczniki dla S7-1200	http://support.automation.siemens.com/WW/view/en/34612486/133300
\6\	Podręczniki dla S7-1500	http://support.automation.siemens.com/WW/view/en/56926743/133300
\7\	Pierwsze kroki z S7-1200	http://support.automation.siemens.com/WW/view/en/39644875
\8\	Pierwsze kroki z S7-1500	http://support.automation.siemens.com/WW/view/en/78027451
\9\	SIMATIC S7-1200 / S7-1500 Porównanie języków programowania	http://support.automation.siemens.com/WW/view/en/86630375

SZKOLENIA DLA PRZEMYSŁU SITRAIN 2017

STYCZEŃ

	P	W	Ś	C	P	S	N
							1
	2	3	4	5	6	7	8
TIA-PRO1	9	10	11	12	13	14	15
ST-PRO1	16	17	18	19	20	21	22
TIA-MICRO1	23	24	25	26	27	28	29
	30	31					

LUTY

	P	W	Ś	C	P	S	N
			1	2	3	4	5
ST-SERV1	6	7	8	9	10	11	12
TIA-S7F MICRO	13	14	15	16	17	18	19
	20	21	22	23	24	25	26
ST-BWINCC	27	28					

MARZEC

	P	W	Ś	C	P	S	N
ST-BWINCC			1	2	3	4	5
TIA-PRO1	6	7	8	9	10	11	12
ST-PRO1	13	14	15	16	17	18	19
	20	21	22	23	24	25	26
TIA-SYSUP IK-PBSYS	27	28	29	30	31		

KWIECIEŃ

	P	W	Ś	C	P	S	N
						1	2
ST-PRO2	3	4	5	6	7	8	9
TIA-S7F MICRO	10	11	12	13	14	15	16
TIA-MICRO1	17	18	19	20	21	22	23
TIA-SYSUP	24	25	26	27	28	29	30

MAJ

	P	W	Ś	C	P	S	N
	1	2	3	4	5	6	7
TIA-MICRO1	8	9	10	11	12	13	14
ST-SERV2	15	16	17	18	19	20	21
TIA-PRO1	22	23	24	25	26	27	28
	29	30	31				

CZERWIEC

	P	W	Ś	C	P	S	N
				1	2	3	4
TIA-MICRO1	5	6	7	8	9	10	11
	12	13	14	15	16	17	18
	19	20	21	22	23	24	25
ST-PRO1	26	27	28	29	30		

LIPIEC

	P	W	Ś	C	P	S	N
						1	2
TIA-PRO1	3	4	5	6	7	8	9
	10	11	12	13	14	15	16
	17	18	19	20	21	22	23
	24	25	26	27	28	29	30
	31						

SIERPIEŃ

	P	W	Ś	C	P	S	N
		1	2	3	4	5	6
	7	8	9	10	11	12	13
TIA-MICRO1 ST-WINCC	14	15	16	17	18	19	20
	21	22	23	24	25	26	27
	28	29	30	31			

WRZESIEŃ

	P	W	Ś	C	P	S	N
					1	2	3
TIA-S7F MICRO TIA-MICRO1	4	5	6	7	8	9	10
ST-PRO2	11	12	13	14	15	16	17
TIA-PRO1	18	19	20	21	22	23	24
ST-WINCC	25	26	27	28	29	30	

PAŹDZIERNIK

	P	W	Ś	C	P	S	N
							1
TIA-MICRO1	2	3	4	5	6	7	8
ST-PRO1	9	10	11	12	13	14	15
ST-BWINCC	16	17	18	19	20	21	22
IK-PNSYS	23	24	25	26	27	28	29
	30	31					

LISTOPAD

	P	W	Ś	C	P	S	N
			1	2	3	4	5
TIA-S7F MICRO	6	7	8	9	10	11	12
TIA-SYSUP	13	14	15	16	17	18	19
TIA-PRO1	20	21	22	23	24	25	26
ST-SERV1	27	28	29	30			

GRUDZIEŃ

	P	W	Ś	C	P	S	N
ST-SERV1					1	2	3
TIA-S7F MICRO TIA-MICRO1	4	5	6	7	8	9	10
ST-PRO1	11	12	13	14	15	16	17
ST-SERV2	18	19	20	21	22	23	24
	25	26	27	28	29	30	31

Miejsce szkoleń

Siemens Sp. z o.o.
ul. Wydawnicza 1/3
92-333 Łódź

Numer kontaktowy

+48 (42) 677-1789

szkolenia.pl@siemens.com
www.siemens.pl/sitrain

SIEMENS
Ingenuity for life

Zestaw startowy SIMATIC S7-1500

Dostępny u autoryzowanych dystrybutorów

- SIMATIC S7-1500 1511C-1 PN – jednostka centralna
- Karta pamięci SIMATIC 4 MB
- Szyna DIN 160 mm
- 365-dniowa licencja dla STEP 7 Professional V14
- Zasilacz PM 70W 120/230 V AC
- Kable Ethernet
- Dokumentacja
- Śrubokręt

Numer zamówieniowy:
6ES7511-1CK00-4YB5

siemens.pl/s7-1500

Biura sprzedaży:

Siemens Sp. z o.o.
Factory Automation
03-821 Warszawa
ul. Żupnicza 11
tel.: 22 870 8200
fax: 22 870 9149

Regionalne biura sprzedaży:

80-300 Gdańsk
Al. Grunwaldzka 413
tel.: 22 870 8200
fax: 58 764 6099

40-527 Katowice
ul. Gawronów 22
tel.: 22 870 8200
fax: 32 208 4139

30-443 Kraków
ul. Józefa Marcika 14B
tel.: 22 870 8200
fax: 12 363 8229

92-333 Łódź
ul. Wydawnicza 1/3
tel.: 22 870 8200
fax: 42 677 1799

60-164 Poznań
ul. Ziębicka 35
tel.: 22 870 8200
fax: 61 664 9864

87-100 Toruń
ul. Gdańska 4A
tel.: 22 870 8200
fax: 56 656 4229

53-611 Wrocław
ul. Strzegomska 52
tel.: 22 870 8200
fax: 71 777 5011

www.siemens.pl/s7-1500

email: simatic.pl@siemens.com

SIEMENS

SIMATIC S7-1500

Opis systemu – wydanie 2/2015