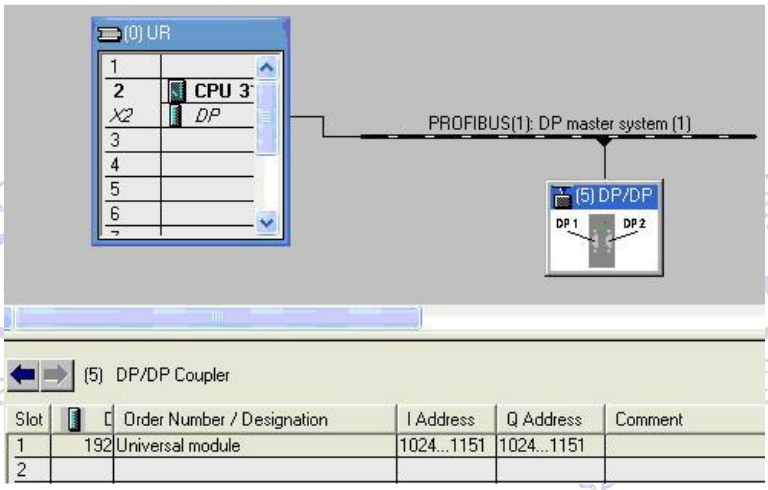
Artykuł został wydrukowany Technice Zagranicznej Maszyny Technologie Materiały w numerze 02/2007 poświęconym rozwiązaniom firmy SIEMENS

Możliwości komunikacyjne SIMATIC S7

Jeszcze kilka lat temu większość maszyn produkcyjnych sprowadzanych do naszego kraju stanowiły maszyny używane. Postępująca u naszych zachodnich sąsiadów niejako pokoleniowa wymiana maszyn i układów sterowania związana z postępem technologicznym, generowała dużą podaż właśnie takich maszyn. Z kolei, rodzimym wytwórcom, pozwalało to pozyskać maszyny, czy całe linie technologiczne, wcześniej w zasadzie niedostępne. Jednak, z drugiej strony patrząc, układy sterowania takich maszyn są w większości oparte na podzespołach już przestarzałych, których produkcja bądź już się zakończyła, bądź wkrótce zostanie zakończona. W tym przypadku koszty utrzymania maszyn w ruchu, konserwacji i napraw, będą rosły z roku na rok. Generalna modernizacja, w przypadku wysłużonych maszyn jest często nieopłacalna. Pozostają drobne ulepszenia i wymiany uszkodzonych podzespołów na produkowane aktualnie odpowiedniki. Zagadnienia związane z sukcesywną zamianą sterowników serii S5 na S7 zostały omówione w poprzednim artykule. Z kolei, niniejszej publikacji, omówimy wzajemne połączenie za pomocą sieci PROFIBUS dwóch oddzielnych maszyn mających pracować razem we wspólnej linii produkcyjnej. Omówimy także metodykę przyłączenia do systemu sterowania podzespołów wyposażonych w interfejs USS oraz w interfejs MODBUS.

1. Połączenie DP/DP

Naszym głównym zadaniem było zrealizowanie połączenia dwóch układów sterowania posiadających swoje oddzielne sieci PROFIBUS i zawiadujących oddzielnymi segmentami linii produkcyjnej. Układy sterowania tych właśnie części linii były już zbudowane w oparciu o sterowniki serii S7300, pełniące rolę Mastera sieci PROFIBUS dla swojego segmentu. Dodatkowo tylko jeden z tych układów wyposażony był w panel operatorski MP. Program zawarty w panelu pozwalał obsługiwać całą linię produkcyjną. Zadanie komunikacyjne polegało na przekazywaniu wartości zadanych i wartości binarnych ustawianych w Masterze numer 1 do Mastera numer 2. Natomiast wartości aktualne, potwierdzenia i sygnały alarmowe, przekazywane były w drugą stronę, z Mastera numer 2 do Mastera numer 1. Zdecydowano się zastosować układ najbardziej uniwersalny – DP/DP Coupler. Jednocześnie, wobec dużej ilości przekazywanych w obie strony informacji, dokonano podziału zadań komunikacyjnych. Wyodrębniono informacje priorytetowe, których przekazanie jest zdeterminowane czasowo oraz informacje, które są nie mniej ważne, lecz niezdeterminowane czasowo tzn. czas dojścia do adresata może wynieść nawet do ok.2 sekund. W celu wykonania powyższego zadania umieszczono w Hardware Configurator urządzenie - DP/DP Coupler pod adresem 1024.



Rys.1 Hardware Configurator – instalacja urządzenia DP/DP Coupler

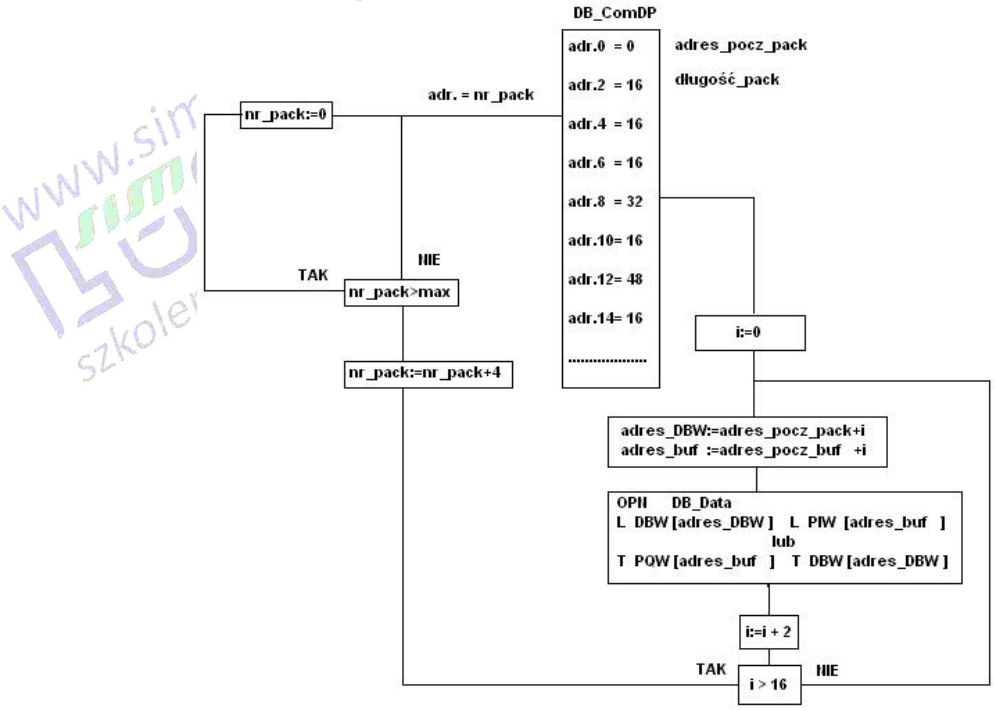
Cały obszar zajęty przez DP/DP Coupler można wykorzystywać do wymiany danych. Daje to łącznie (1151 – 1024 +1) = 128 bajtów czyli 64 słowa. Urządzenie DP/DP Coupler przepisuje obszar wejściowy PI jednej sieci na obszar wyjściowy PQ drugiej sieci oraz obszary PI drugiej na PQ pierwszej, zachowując odpowiednią kolejność bajtów o tych samych numerach. Zatem wpisując daną do n-tego słowa PQW jednej sieci, możemy ją odczytać za pośrednictwem n-tego słowa PIW drugiej sieci. W omawianym przykładzie, z uwagi na przyjętą hierarchiczność danych, 52 słowa (obszary od 1024 do 1127) są przeznaczone do obsługi transmisji synchronicznej. W obszarze tym kolejne słowa mają określone znaczenie i ich wartości aktualne są zapisywane i odczytywane komendami Load i Transfer w każdym przejściu głównego bloku programowego OB1. Pozostałe 12 słów zostało wykorzystanych do transmisji asynchronicznej, przy czym ostatnie 8 słów tworzy bufor przesyłania i odbioru danych.

53	x+ 104.0	INT "ComRespond_DataRecieve"/"ComRespond_DataWrite"	
54	x+ 106.0	INT "ComRespond_DataRecieve"/"ComRespond_DataWrite"	
55	x+ 108.0	INT "ComRespond_DataSend"/"ComRespond_DataRead"	
56	x+ 110.0	INT "ComRespond_DataSend"/"ComRespond_DataRead"	
57	x+ 112.0	INT "DataRecieve" /"DataWrite"	Data channel to exchange data packages for asynchron comunication
58	x+ 114.0	INT "DataRecieve" /"DataWrite"	Data channel to exchange data packages for asynchron comunication
59	x+ 116.0	INT "DataRecieve" /"DataWrite"	Data channel to exchange data packages for asynchron comunication
60	x+ 118.0	INT "DataRecieve" /"DataWrite"	Data channel to exchange data packages for asynchron comunication
61	x+ 120.0	INT "DataRecieve" /"DataWrite"	Data channel to exchange data packages for asynchron comunication
62	x+ 122.0	INT "DataRecieve" /"DataWrite"	Data channel to exchange data packages for asynchron comunication
63	x+ 124.0	INT "DataRecieve" /"DataWrite"	Data channel to exchange data packages for asynchron comunication
64	x+ 126.0	INT "DataRecieve" /"DataWrite"	Data channel to exchange data packages for asynchron comunication

Rys.2 Obszar obsługi transmisji asynchronicznej

Dane, które zamierzamy przesłać do drugiego Mastera, gromadzimy w bloku DB_Data. Obszar adresowy tego bloku dzielimy na „paczki” danych o długości 8 słów. Paczki te będą przesyłane kolejno przez bufor PIW/PQW w obszarach 1136 – 1151, tj. o długości 16 bajtów. Adres początku obszaru zajmowanego przez każdą paczkę i długości paczek są zapisane w bloku DB_ComDP. Kolejne słowa tego bloku uporządkowane są następująco: adres pierwszej paczki, długość pierwszej paczki, adres drugiej paczki, długość drugiej paczki itd. Zatem adresy i długości kolejnych paczek zapisane są pod kolejnymi adresami bloku danych DB_ComDP, tj. co cztery bajty każdy następny. Podprogram obsługujący transmisję będzie pobierał z bloku DB_Data paczkę danych, każdorazowo inkrementując zmienną zawierającą numer paczki i posługując się adresem pamiętanym w odpowiadającej numerowi paczki komórce bloku DB_ComDP. Pobrana paczka danych zostanie umieszczona w buforze PQW Mastera pierwszej sieci a dalej następuje jej przesłanie w Couplerze DP/DP pomiędzy sieciami, a następnie z bufora PIW Mastera drugiej sieci zostanie umieszczona w bliźniaczym bloku DB_Data drugiego Mastera. Przesyłanie w drugą stronę odbywa się podobnie, jednak należy pamiętać, iż numer kolejnej paczki dyktuje tylko jeden z Masterów – ten który nadzoruje wymianę danych. W ten sposób w każdym przejściu bloku OB1 jest przesyłana jedna paczka (8 słów). Zatem przykładowo przesłanie 2400 bajtów wymaga 150 przejść programowych. Licząc ok. 10 ms jako czas

przejścia, czas transmisji wyniesie ok. 1.5 s, co w przypadku danych wizualizacyjnych jest wynikiem do przyjęcia.



Rys.3. Algorytm transmisji asynchronicznej DP/DP

Podprogram takiej wymiany danych wygląda następująco:

```
PN DB_ComDP //blok DP pomocniczy z adresami i długościami kolejnych paczek
L nr_pack
SLW 3
LAR1 //utworzenie adresu formatu pointer komórki a numerem aktualnej paczki

L DBW [AR1,P#100.0]
T adres_DBW //adres aktualnej paczki
L DBW [AR1,P#102.0]
T długość_paczki //długość aktualnej paczki

……..
send: NOP 0

L adres_DBW //adres początku paczki
L indeks //zmienna zwiększająca adres kolejnych słów paczki
+I
SLW 3
LAR1 //adres formatu pointer aktualnego słowa paczki

L adres_buf //adres początku bufora nadawczego
L indeks
+I
SLW 3
LAR2 //adres formatu pointer kolejnego słowa bufora
```

```
OPN DB_Dane
L DBW [AR1,P#0.0]
T PQW [AR2,P#0.0] //przesłanie jednego słowa aktualnej paczki

L indeks //zmienna indeks zwiększamy o 2 – dwa bajty
L 2
+I
T indeks
L długość_paczki
JC send //jeżeli zmienna indeks nie przekroczyła długości paczki - pętla
……
L nr_paczki
L 4
+I
T nr_paczki //nr_paczki zwiększamy o 4 – patrz rysunek
L nr_paczki_max
>I
JCN end
L 0
T nr_paczki //jezeli wysłano ostatnią paczkę – zeruj nr paczki
end: NOP 1
```

Po wysłaniu wszystkich paczek – licznik paczek zeruje się i zaczynamy proces wysyłania od nowa, czyli od paczki zerowej. Powyższy sposób pozwala na przesłanie sporej ilości danych wizualizacyjnych przez sieć PROFIBUS sterującą jednocześnie urządzeniami o dużych wymaganiach dotyczących czasu reakcji na sygnały sterujące przesyłane przez sieć. Zastosowanie w takim przypadku transmisji synchronicznej do wszystkich danych spowodowałoby „zapchanie" sieci, a zatem uniemożliwiło poprawne funkcjonowanie omawianej maszyny.

2. Komunikacja sterownika S7 200 z urządzeniami MODBUS Slave

W czasie realizacji zadań modernizacji maszyn, często spotykamy się z faktem konieczności współpracy z urządzeniami różnych producentów wyposażonymi w interfejsy sieciowe, które nie są typowe dla produktów firmy Siemens. Takimi urządzeniami są np. regulatory obiektowe wyposażone w interfejs sieci MODBUS. Na potrzeby modernizacji maszyny, w której funkcjonowały takie regulatory napisano program obsługi dla sterownika S7 200, realizujący zapis i odczyt danych za pośrednictwem sprzęgu RS485 i protokołu MODBUS. Wykorzystano tutaj możliwości portu swobodnie programowalnego sterownika S7 200. Protokół ten ma postać binarną (RTU): 1 bit startu, 8 bitów danych, 1 bit kontroli parzystości (opcjonalnie), 1 bit stopu. Prędkości transmisji powinny być wybrane z zakresu 600, 1200, 2400, 4800, 9600,19200 bodów, a adresy ustawiane w Slave'ach powinny być z zakresu 0 – 255. W tego typu komunikacji, wg protokołu MODBUS, transmisja inicjowana jest przez Mastera sieci, którym w naszym rozwiązaniu będzie S7 200. Urządzenie Slave odpowiada jedynie po otrzymaniu rozkazu od Mastera. Natomiast format telegramu z Mastera do Slave'a ma postać następującą:

Bajt nr 1 adres Slave'a
Bajt nr 2 nr funkcji
Bajt nr 3-(3+n) n bajtów danych
Bajt nr (4+n) suma kontrolna (CRC-16) bajt młodszy
Bajt nr (5+n) suma kontrolna (CRC-16) bajt starszy

Suma kontrolna określona jako CRC-16 (z jęz. ang. Cyclical Redundancy Check) jest obliczana przez urządzenie wysyłające. Urządzenie odbierające także oblicza sumę kontrolną otrzymanego telegramu i porównuje z wartością zamieszczoną w telegramie. Obie sumy musza być jednakowe przy poprawnej transmisji. Opis dostępnych funkcji oraz postać odpowiedzi Slave'a należy szukać w dokumentacjach opisujących transmisję do poszczególnych urządzeń. Natomiast wspólną częścią jest obliczanie sumy CRC-16. W przypadku sterownika S7 200, blok programowy obliczający CRC-16 ma postać następującą:

```
LD SM0.0
MOVW 16#FFFF, VW220 //ustawienie samych jedynek w rejestrze obliczeniowym
MOVD &VB100, AC1 //ustawienie wskaźnika adresu buforu
FOR VW290, +1, VW222 //pętla programowa obejmująca wysyłane bajty
XORB *AC1, VB221 //operacja XOR kolejnych bajtów
FOR VW292, 1, 8 //pętla programowa sprawdzania bitów w bajcie
SRW VW220, 1 //przesunięcie w prawo
LD SM1.1 //jeżeli wysunięta jedynka
XORW 16#A001, VW220 //XOR z A001h
NEXT
INCD AC1
NEXT
MOVB VB221, AC3 //wynik CRC-16 młodszy bajt
SLW AC3, 8
MOVB VB220, AC3
MOVW AC3, *AC1 // wynik CRC-16 starszy bajt
```

Sposób programowania samej transmisji i sposobu komunikacji sterownika klasy S7 200 został szeroko opisany w licznych dokumentacjach producenta i nie będzie przedmiotem dalszych rozważań.

3. Komunikacja z urządzeniami wykorzystującymi protokół USS

Protokół USS jest powszechnie stosowany w urządzeniach firmy Siemens (np. systemy napędowe). Jest protokołem specjalizowanym z ukierunkowaniem na zadania uruchamiania i diagnozowania, ale także z możliwością realizacji zadań sterowania. Choć tutaj obecnie wypiera go PROFIBUS DP. W przypadku napędów poprzedniej generacji lub jeszcze wcześniejszych przejście na sieć PROFIBUS nastręcza wiele trudności. Jednak interfejs do obsługi transmisji USS jest dostępny już w opcjach podstawowych. W opisywanym przypadku realizacja kolejnego zadania polegała na wysłaniu do napędów serii SIMOREG 6RA24 firmy Siemens czterech słów danych i odbioru także czterech słów. Podobnie i w tym przypadku wykorzystano tryb swobodnego programowania portu komunikacyjnego sterownika S7 200. Protokół ma postać binarną: 1 bit startu, 8 bitów danych, 1 bit kontroli parzystości, 1 bit stopu. Prędkość komunikacji wybieramy z zakresu: 57,6 – 38400 bodów.

Telegram wymiany danych ma postać następującą:

Bajt nr 1 - STX=02h
Bajt nr 2 - XMT=0ah - długośćtelegramu od ADR do BCC
Bajt nr 3 - ADR=01h - adres slave'a
Bajt nr 4 - 1 word - byte starszy
Bajt nr 5 - 1 word - byte mlodszy
Bajt nr 6 - 2 word - byte starszy
Bajt nr 7 - 2 word - byte mlodszy
Bajt nr 8 - 3 word - byte starszy

Bajt nr 9 - 3 word - byte mlodszy
Bajt nr 10 - 4 word - byte starszy
Bajt nr 11 - 4 word - byte mlodszy
Bajt nr 12 – BCC

Sumę kontrolną BCC obliczamy w sposób następujący:

```
LD SM0_0_SET
MOVD &VB101, AC1 //ustawienie wskaznika adresu buforu
MOVD 16#08, AC2
FOR AC3, +1, +9 //for od 1 do 9
XORW *AC1, AC2 //xor kolejnych slow
INCD AC1 //zwieksz o 1
NEXT
INCD AC1 //zwieksz o 1
MOVB AC2, *AC1 //zapisanie BCC w ostatnim bajcie
```

Również i w tym przypadku, wszelkie informacje potrzebne do konfiguracji telegramu w napędzie znajdziemy w dokumentacji technicznej producenta.

Podsumowanie

Opisane przykłady zostały zrealizowane praktycznie, a maszyna w której funkcjonują stworzone programy jest obecnie maszyną produkcyjną. Realizacja poszczególnych zadań komunikacyjnych, w odniesieniu do tematu artykułu, pozwala na podział maszyny na segmenty, które można oddzielnie modernizować. Właśnie z uwagi na ten fakt opisano sposoby realizacji kolejnych zadań segmentacji sieci. Zaprezentowano także możliwości komunikacyjne, często niedocenianego, sterownika klasy S7 200. Natomiast zaproponowane rozwiązania mogą być pomocne w realizacji zadań związanych z modernizacją maszyn produkcyjnych. Jednocześnie, autorzy zachęcają użytkowników do opracowywania własnych koncepcji, często niekonwencjonalnych, projektów i struktur sieciowych oraz mechanizmów wymiany danych procesowych, gdyż takie rozwiązania potrafią znacznie obniżać koszty wykonywanych modernizacji.

Autorzy:
Mariusz Jabłoński
Przemysław Grasewicz

Państwa, zainteresowanych naszą ofertą rozwiązań oraz ofertą handlową, prosimy o kontakt z naszymi handlowcami pod numerami telefonów:
042 648 66 77 oraz 042 648 67 07

Możecie Państwo także wysłać zapytanie na adres zapytania@simlogic.pl

Zapraszamy Państwa do kontaktu.