Mgr inż. Rafał Tutaj
Elmark Automatyka sp. z o.o. dział Rockwell Automation
rt@elmark.com.pl

Sterownik PLC w środowisku wielosieciowym

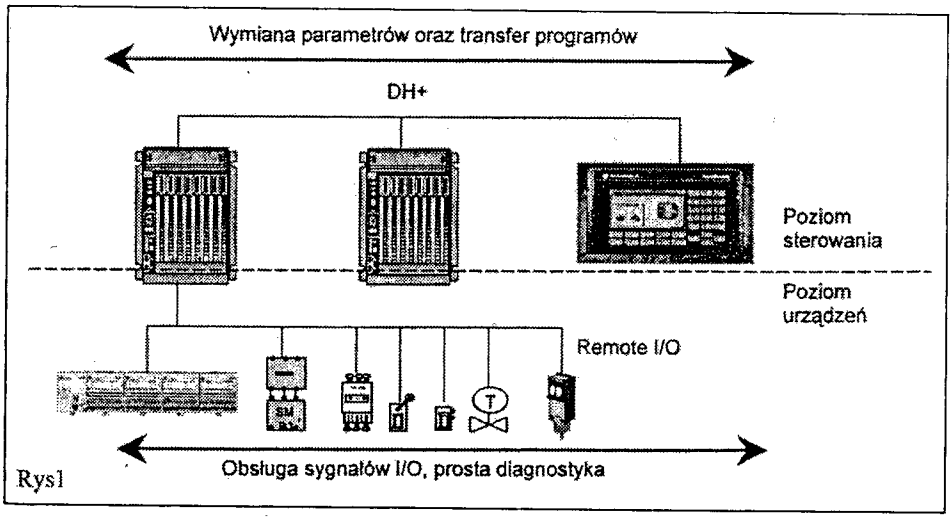
Streszczenie : W referacie przedstawiono zachowanie się sterownika PLC w środowisku złożonym z wielu przemysłowych sieci. Szczególny nacisk został położony na problemy, jakie musi pokonać projektant budując aplikację, oraz sposoby ich rozwiązywania. Całość została opatrzona przykładami, przy których projektowaniu i uruchamianiu autor współpracował

Programmable Logic Controller in multi-networks environment.

Abstract : The paper present the behaviour of PLC controller in environment consist of multiple networks. Especially the problems, that the designer should solve, building the application, are described. Whole was completed by examples, by which the author assists as co-designer or start-up consultant.

1. WPROWADZENIE

Dwa lata temu, podczas konferencji Automation 97, zaprezentowany został referat p.t. „Przemysłowe systemy komunikacyjne"[1][2]. Było to przeglądowe ujęcie tematu, w którym przedstawiony został trójpoziomowy model sieciowy oraz omówiono główne cechy sieci należących do każdego poziomu. Podstawą dla referatu były doświadczenia z lat wcześniejszych (1993-1996), kiedy to systemy komunikacyjne (o ile w ogóle były stosowane) ograniczały się do co najwyżej jednego lub dwóch rodzajów połączeń, na różnych poziomach. Zazwyczaj w pierwszym etapie instalowano sieć na poziomie urządzeń, dzięki czemu uzyskiwano znaczną poprawę eliminacji zakłóceń, oszczędność kabli oraz dostęp do coraz większej liczby parametrów w złożonych urządzeniach sensorycznych i wykonawczych.

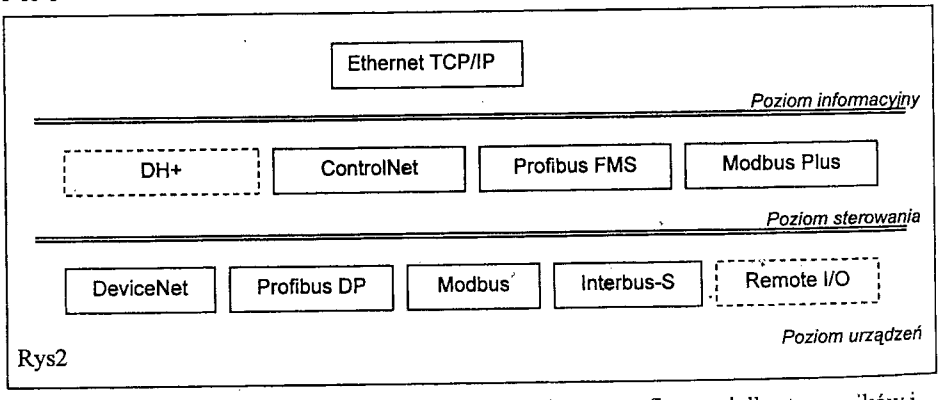


Rys1

Drugi etap polegał na integracji sterowników PLC oraz stacji operatorskich, poprzez zastosowanie jednego z przedstawicieli poziomu sterowania. Przepływ informacji i struktura aplikacji „dwuetapowej" pokazana została na rys 1. Warto tu zwrócić uwagę na typowe w tym czasie stosowanie połączeń homogenicznych (czyli jednego rodzaju na każdy poziom). Jeśli jakieś urządzenie nie miało możliwości pracy w danej sieci, to raczej podłączano je klasycznie (sygnał dwustanowy lub analogowy) niż poszukiwano dróg dopasowania kolejnego protokołu.

Kolejne lata rozwoju technik komunikacyjnych przyniosły zmiany niemalże rewolucyjne. Powszechne stało się wyposażanie czujników i elementów wykonawczych w interfejsy sieciowe. Z jednej strony dzięki temu dostępne stały się w zasadzie wszystkie parametry urządzenia, z drugiej zaś, wielość takich systemów wytworzyła problem połączenia kilku różnorodnych sieci. Podobna sytuacja wystąpiła na poziomie sterowania. Tu generatorem „problemu sprzęgu" stało się dążenie do pełnej integracji procesu produkcyjnego. W przypadku linii technologicznych tworzonych w różnym okresie, przez różne firmy, wynikał on ze stosowania sterowników wykorzystujących odmienne, często zamknięte sposoby transmisji danych. W jednorodnych aplikacjach zasadniczo był on zawiązany z koniecznością dopasowania wymiany danych do wymogów procesu (czas, ilość) i zapewnienia pełnej diagnostyki.

Obecnie, przy rozwiązywaniu wspomnianych problemów spotyka się kilkanaście typowych rodzajów sieci. Poniższy rysunek prezentuje je w kontekście modelu sieciowego [1][2].



Rys2

Liniami kreskowanymi przedstawione zostały sieci będące specyficznymi dla sterowników i systemów Rockwell Automation Allen-Bradley. Pozostałe, dzięki otwartości, stały się standardami w przemyśle, aczkolwiek obowiązującymi w pewnym sensie geograficznie. Wśród przedstawionych powyżej można wydzielić sieci stosowane głównie w Europie (Profibus, Interbus, Modbus) oraz w Amerykach i dalekiej Azji (DeviceNet,ControlNet). Był to wynik opanowania rynku automatyki przez twórców i pomysłodawców odpowiednich sieci, który to podział sukcesywnie ulega zmianom z korzyścią dla użytkowników, mogących obecnie swobodnie wybierać rodzaj połączenia.

2. PROBLEM INTEGRACJI

Wspomniana powyżej różnorodność zmusza użytkowników do rozwiązania problemu sprzęgu czyli integracji sieci w ramach jednej instalacji technologicznej. Definiując problem w tym kontekście, można powiedzieć, iż *sprzęg (integracja) wielu sieci jest to zapewnienie*

poprawnej wymiany informacji pomiędzy wszystkimi komponentami aplikacji, a którego rozwiązanie jest równoznaczne z zastosowaniem urządzenia pozwalającego na :

- Konwersję sygnałów elektrycznych (integracja na poziomie fizycznym)
- Konwersję i routing protokółowy (integracja na poziomie łącza i aplikacyjnym)

Wymagania te mogą zostać zrealizowane przy pomocy jednego z następujących systemów:

- Komputera, wyposażonego w odpowiedni zestaw kart sieciowych i oprogramowanie
- Sterownika PLC, wyposażonego w moduły komunikacyjne
- Sterownika typu gateway z zainstalowanymi łączami sieciowymi.

Każdy z nich posiada pewne własności, określające warunki jego pracy, zachowanie oraz łatwość stosowania w przemysłowych aplikacjach sieciowych.

Tabela 1

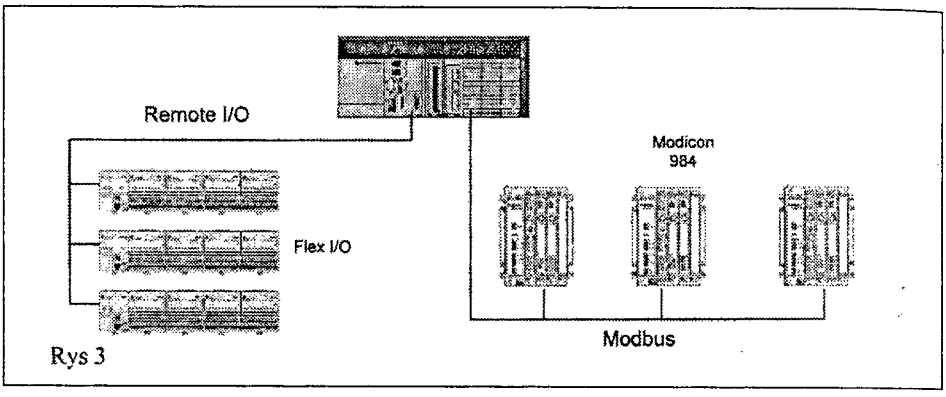
Komputer	
+	• Łatwość programowania w językach wysokiego poziomu (C,C++, Pascal, inne)
–	• Trudność w obsłudze i zrozumieniu komputerowych narzędzi programistycznych w przemyśle (technicy, inżynierowie utrzymania ruchu, serwis)
–	• Zależność oprogramowania sieciowego (sterowniki kart) od zastosowanego systemu operacyjnego
–	• Podatność na zakłócenia i wpływ środowiska przemysłowego (wibracje, temperatura, wilgotność, pyły itp.)
Sterownik PLC	
+	• Niezależność od systemów operacyjnych, przeniesienie programowej obsługi komunikacji do modułu
+	• Odporność na warunki przemysłowe
+	• Oprogramowanie w typowych językach stosowanych w przemyśle (LAD,FBD,STX), łatwość obsługi
–	• Obsługa przez procesor główny sterownika
Sterownik typu gateway	
+	• Niezależność od systemów operacyjnych, przeniesienie programowej obsługi komunikacji do modułu
+	• Odporność na warunki przemysłowe
+	• Trasa przepływu danych (routing) tworzony przy pomocy dodatkowego, z reguły bardzo intuicyjnego oprogramowania narzędziowego
+	• Niezależne procesory do obsługi poszczególnych modułów komunikacyjnych
–	• Z uwagi na zaawansowanie, wyższa cena całości

Wnioski płynące z powyższego zestawienia (porównanie ilości +/–) są zgodne z obserwacjami autora dotyczącymi istniejących aplikacji. W większości zastosowań wykorzystano przede wszystkim sterownik PLC, wyposażony w szereg specjalizowanych modułów. W szczególnych procesach, cechujących się wysokimi prędkościami i dużą ilością danych zainstalowane zostały sterowniki typu gateway. Komputery, posiadające możliwość komunikacji wielosieciowej, znalazły swoje miejsce przede wszystkim przy obsłudze wizualizacji procesowej, pełniąc funkcje raczej pomocnicze w aplikacji.

3. STEROWNIK PLC – KONSTRUKCJA SPRZĘGU

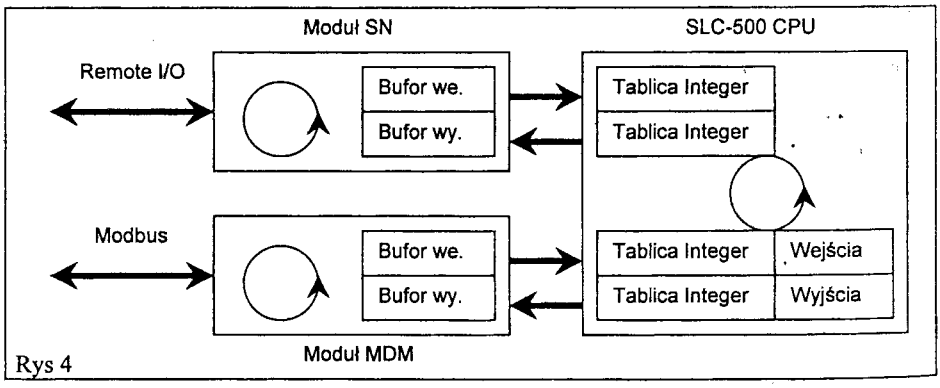
Przy definiowaniu problemu integracji (pkt. 2) wskazano na dwie płaszczyzny pracy urządzenia sprzęgającego – na poziomie fizycznym oraz w ramach warstwy łącza i aplikacyjnej. Na rysunku 3 przedstawiona została aplikacja złożona z dwóch sieci : Remote

I/O oraz Modbus. Zadaniem jej było sterowanie linią przygotowania i produkcji szkła poprzez rozproszone moduły obiektowe Flex I/O (Rockwell Automation/Allen-Bradley) oraz komunikacja ze sterownikami Modicon 984 (ASA/AEG), bezpośrednio obsługującymi jedynie wannę szklarską.



Rys 3

Zrealizowanie powyższego połączenia w sterowniku SLC-500 wymagało zastosowania dwóch niezależnych modułów SN (RIO) oraz MDM (M-bus), co wynikało przede wszystkim z faktu zapewnienia obsługi na poziomie fizycznym. Ta strona sprzęgu jest zazwyczaj łatwa to zrealizowania, dzięki specjalizowanym układom sterowników magistrali (tu jedynie RIO wymaga specyficznego elementu, gdyż M-bus jest prowadzony w oparciu o typowe układy RS-485). Nieco trudniej jest zapanować nad stroną protokółową procesu. Na rysunku 4 przedstawiona została struktura przepływu danych w zastosowanym sterowniku SLC-500.



Rys 4

Każdy z modułów został wyposażony we własny bufor we/wy, odświeżany z częstotliwością zależną od prędkości sieci oraz ilości połączonych w niej urządzeń. W omawianym przypadku wystąpiła dość znaczna różnica w czasach, wynikająca z pracy RIO w trybie 230 kbaud i M-bus w trybie 19.2 kbaud. Zmiana danych ze sterowników Modicon była obserwowalna po 1 do 1.5 sekundy, podczas gdy z Flex I/O informacje były odbierane co ok. 40 ms. Jest to objaw pierwszego problemu integracji :

zapewnienie współbieżności i jednoczesności danych (1)

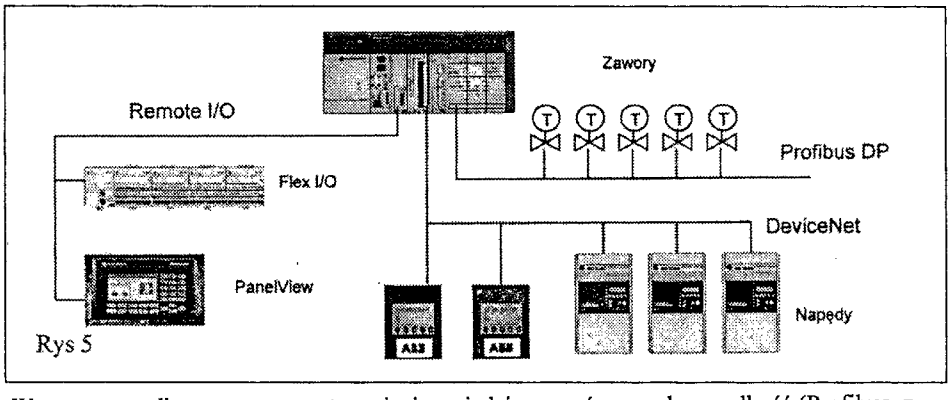
Sytuacja taka występuje zawsze, gdy łączone sieci posiadają prędkości transmisji różniące się o rząd wielkości. Dodatkowym czynnikiem wpływającym na wydłużanie tych czasów jest korzystanie z sieci typu nadawca-odbiorca [3][4], cechujących się dość niską efektywnością. Kolejny problem, który można określić jako :

buforowanie i przesyłanie danych moduł ←→ CPU (2)
jest związany koniecznością zapamiętania informacji o stanie urządzeń sieciowych. Dane te są przechowywane zarówno w module jak i w pamięci procesora. Istotą tego zagadnienia jest dwukierunkowe przesyłanie zawartości buforów do/z tablic w procesorze sterownika. Ideałem jest, gdy proces ten przebiega asynchronicznie w stosunku do cyklu samego procesora. Tak dzieje się w przypadku RIO i tablic typu we/wy. Pozostałe dane, jak i informacje z sieci M-bus, trafiające do tablic integer, są przesyłane w sposób synchroniczny, odpowiednim poleceniem z programu. Powoduje to konieczność stosowania bitów synchronizacyjnych, w celu określenia aktualności danych. Z problemem (2) jest związany trzeci temat :

obsługa wymiany informacji pomiędzy tablicami w procesorze sterownika (3)

Przenoszenie danych pomiędzy odpowiednimi tablicami procesora (a co za tym idzie pomiędzy sieciami) jest realizowane przez program sterownika. Czas wykonania cyklu rozkazowego jest jak wiadomo zmienny i ściśle zależny od struktury programu, stanu obiektu i ingerencji operatorów. O pozostawienie obsługi sprzęgu „w rękach" programu można się pokusić w aplikacjach wolnych, gdzie opóźnienia nawet rzędu kilkuset milisekund są nie istotne. Co jednak, gdy proces jest szybki i wymaga natychmiastowej reakcji. W takich sytuacjach stosowany jest mechanizm przerwań czasowych (STI-Selectable Timed Inerrupt), pozwalający na określenie interwału, co ile uruchamiany będzie podprogram obsługi komunikacji. W przypadku sterownika SLC-500 minimalny czas jest równy 1ms, co przy założeniu iż częstotliwość obsługi sygnału powinna być 5-krotnie większa niż szybkość jego zmian, spełnia nawet dość wyśrubowane wymagania.

Nieco inne problemy pojawiły się podczas realizacji sterowania produkcją szamponu w aplikacji pokazanej na rysunku 5



Rys 5

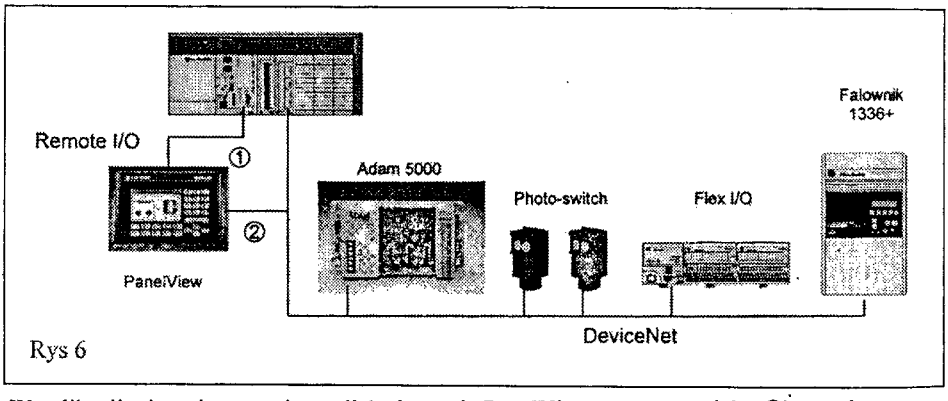
W tym przypadku zastosowano trzy sieci, posiadające porównywalną prędkość (Profibus, z uwagi na zawory – 500kbaud, DeviceNet – 500kbaud oraz RIO 230 kbaud) oraz podobny dostęp do zasobów procesora (tablice we/wy bezpośrednio, tablice integer uaktualniane synchronicznie). Własności te pozwoliły wyeliminować trudności oznaczone (1) oraz (2). Niestety zwiększona ilość rodzajów szybkich połączeń spowodowała iż problem (3) stał się jeszcze bardziej widoczny niż w poprzednim przykładzie. Poszczególne łącza „zaopatrywały" procesor w dane co ok. 40 ms (RIO), 70ms (PFB) i 20ms (DN). Czas cyklu procesora w omawianej aplikacji wynosił ok. 100 ms i był wynikiem realizacji dość złożonego programu. Wszystkie dane zostały więc podzielone na dwie grupy : istotne, obsługiwane przez tablice we/wy i procedurę STI oraz wolnozmienne, obsługiwane tradycyjnie przez program. Ustawienie czasu obsługi STI na 5ms pozwoliło osiągnąć sterowanie w czasie rzeczywistym, bez widocznych opóźnień.

Różnice w przedstawionych czasach są wynikiem zastosowania odmiennej liczby urządzeń, o różnym charakterze oraz ilości wymienianych danych. Późniejsze testy pokazały jednak, że równie istotny wpływ ma

charakter i koncepcja współpracujących sieci (4)

Dwie z wykorzystanych sieci (RIO oraz PFB) są oparte o koncepcję nadawca-odbiorca [3][4]. W tym modelu rozesłanie tej samej informacji wymaga co najmniej takiej samej liczby wysłanych ramek, ile jest urządzeń w sieci. Istnienie dodatkowych zabezpieczeń w postaci potwierdzeń, powoduje, że efektywność połączeń znacznie spada. Rezultatem tego jest wydłużanie czasu cyklu obsługi sieci przez moduł komunikacyjny. Uwzględniając fakt, iż wspomniane sieci nie posiadają wewnętrznego mechanizmu oceny czy przesyłane informacje zostały zmienione, czy też przesyłamy te same wartości (np. pojemność dużego zbiornika), czas oczekiwania na nowe dane jest dość długi. Zastosowanie modelu producent-konsument [3][4][5][6] pozwala na rozesłanie tej samych informacji do wielu urządzeń jednocześnie (multicast, broadcast). Dodatkowo, zastosowanie mechanizmów Change-Of-State, powoduje, iż sieć jest zajmowana tylko w sytuacjach, kiedy dane faktycznie uległy zmianie, i to np. o określony procent. Dzięki temu efektywność DeviceNet, działającego według tej koncepcji jest znacznie większa, co pozwala na obsługę krytycznych informacji.

Uzupełnieniem rozważań nad problemem (4) jest przykład pokazany na rysunku 6.



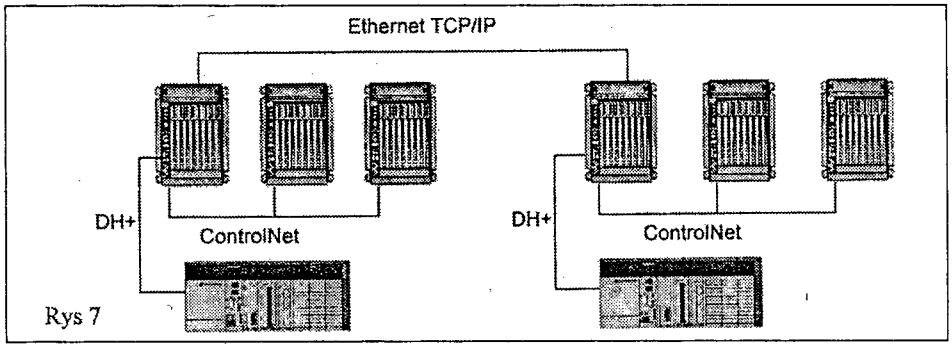
Rys 6

W aplikacji tej w pierwotnej wersji (połączenie PanelView, oznaczone jako ①) wymiana danych odbywała się klasycznie, poprzez procesor. W sytuacji awaryjnej, w której procesor sterownika przechodzi w stan błędu, obsługa komunikacji międzysieciowej zamiera, powodując zatrzymanie całości systemu. Z jednej strony jest to efekt koncepcji master (multi-master)-slave, stosowanej we wszystkich sieciach poziomu urządzeń, w której awaria centralnego modułu wyłącza obsługę przypisanych do niego urządzeń (w szczególności wyjściowych). Z drugiej zaś, co jest bardziej istotne dla naszych rozważań, zatrzymanie programu jest równoznaczne z brakiem przepisywania danych pomiędzy buforami i powoduje dezintegrację sprzęgu. W przypadku niektórych modułów stop procesora wywołuje także przejście sieci w stan zawieszenia (idle) całkowicie blokując ruch sieciowy. W tym przypadku problem

odporność aplikacji na zatrzymanie sterownika PLC (5)

został rozwiązany w sposób niezgodny z duchem rozważań, a mianowicie poprzez bezpośrednie podłączenie PanelView do sieci DeviceNet (połączenie ②) i wykorzystanie własności dostępu do wszystkich parametrów urządzeń sieciowych z zaawansowanych terminali oraz napędów.

Wskazane powyżej 5 problemów występuje w każdej aplikacji, wykorzystującej sterownik PLC jako sprzęg sieci poziomu urządzeń. Nieco inna sytuacja występuje w systemach, w których zachodzi potrzeba integracji na wyższych poziomach : sterowania i informacyjnym. W tych sieciach wszystkie węzły są urządzeniami bardziej złożonymi (sterowniki, stacje operatorskie, komputery), dzięki czemu mogą niezależnie inicjalizować połączenia między sobą. Przypomnijmy, że omawianych wcześniej przykładach, stosowany był mechanizm master-slave, określający jednoznacznie kto ma prawo nawiązywać połączenie (wyjątkiem jest DeviceNet, dzięki pracy w trybie COS [5][6]). Tak więc w przypadku wyższych poziomów mamy sieci o charakterze współrzędnym, co eliminuje z definicji problem (5). Pozostałe zagadnienia (1)-(4) mają teraz nie co inne podłoże. Na rysunku 7 przedstawiony został schemat wymiany informacji w jednej z montowni samochodów.

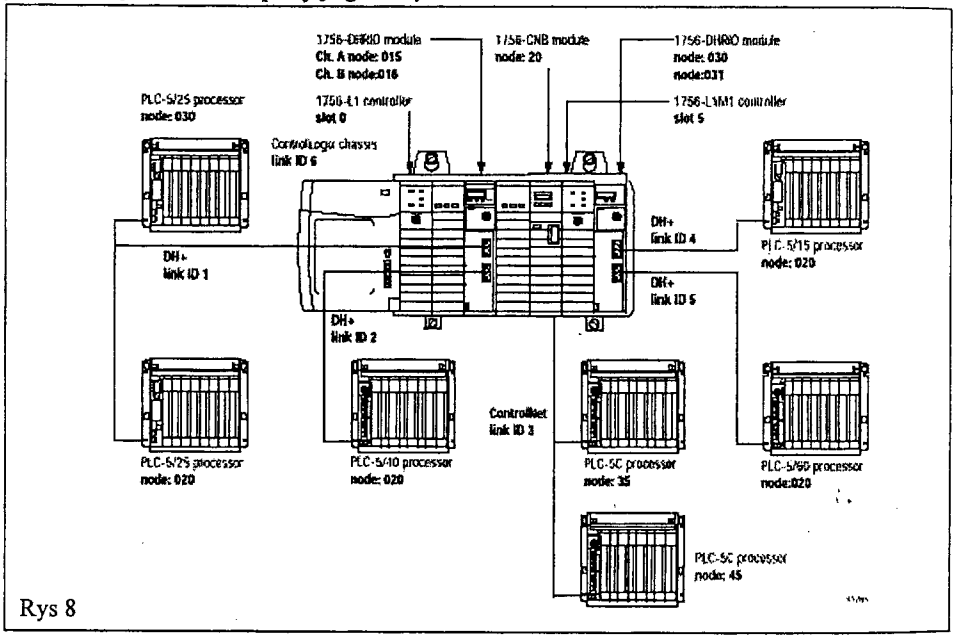


Rys 7

W aplikacji tej połączono trzy sieci o dość zróżnicowanych prędkościach (ETH 10Mbaud, CN 5Mbaud, DH+ 230 kbaud) , co stało się źródłem powstania problemu (1), czyli współbieżności obsługiwanych danych. Dodatkowo do pogłębienia trudności synchronizacyjnych przyczyniła się własność (4), wyrażająca się skrajnie różnymi modelami. Przy komunikacji Ethernet TCP/IP, brak determinizmu (czyli możliwości oszacowania okresu obiegu informacji) ma zasadniczy wpływ czas pracy aplikacji, zwłaszcza w systemach, w których obciążenie sieci przekracza 60% i występuje znaczna ilość kolizji. W opisywanym przypadku aby zapobiec szkodliwym zjawiskom, zastosowano separację części przemysłowej i ogólnozakładowej poprzez instalację switch'a, oraz zblokowano przesyłane dane. Dodatkowo, by uwolnić się od problemu (2), zainstalowany został koprocesor sieci Ethernet, mający bezpośredni dostęp do wszystkich rejestrów procesora, bez konieczności angażowania programu aplikacyjnego. Pozostałe dwa łącza (CN oraz DH+) zostały doprowadzone bezpośrednio do portów procesora, dzięki czemu zadanie (2) przestało być istotne. Problem (4) nadal jednak pozostał. W przypadku DH+ wymiana informacji odbywa się w oparciu o model nadawca odbiorca, ze wszystkimi konsekwencjami. Zastosowany mechanizm token-passing, pozwala na oszacowanie przedziału czasowego, w jakim poszczególne urządzenia otrzymają dane. Sieć ControlNet została zaprojektowana wg koncepcji producent konsument z przydziałem czasu. Jest to pierwsza przemysłowa sieć, gdzie podczas konfiguracji określa się szczegółowo co ile ms ma następować wymiana informacji między poszczególnymi węzłami. Dzięki tej własności udało się skompensować nierównomierności czasowe w pozostałych sieciach. Wykorzystanie do integracji aplikacji procesorów PLC-5 pozwoliło na zmniejszenie wpływu problemu (3) na całkowity czas pracy systemu. Generalnie dużym atutem wykorzystanych łączy jest ich prędkość i efektywność, zwłaszcza ControlNet., które to przyczyniły się do osiągnięcia czasów pełnej wymiany dość dużej ilości informacji mniejszych niż 40 ms.

4. STEROWNIK GATEWAY

Wychodząc naprzeciw problemom użytkowników, zwłaszcza przedstawionym w (2), (3) oraz (5) skonstruowano specjalną wersję sterownika, określaną jako gateway. Zadaniem jego jest przekazywanie dany pomiędzy modułami komunikacyjnymi, z wyłączeniem funkcji sterowania procesem. Wyeliminowanie wskazanych trudności nie było łatwe, i w zasadzie stało się możliwe dopiero wraz z konstrukcją takich urządzeń jak ControlLogix Gateway, wyposażonym w odmianę ControlNet jako magistralą łączącą moduły. Zapewniło to dużą efektywność pracy takiego sprzęgu. Na rysunku 8 przedstawiono fragment systemu sterowania w jednej z fabryk FMCG, w której gateway zapewnia łączność z linią (via ControlNet) oraz wymianę danych z urządzeniami pomocniczymi via DH+. Na rysunku nie pokazano łącza Ethernet spinającego wszystkie linie.



Rys 8

Z uwagi na sukcesywne wprowadzanie tych urządzeń w przemyśle, autor ma nadzieję przedstawić szerzej zagadnienia stosowania sterowników typu gateway na następnej konferencji, po zebraniu odpowiedniej ilości obserwacji i wniosków z nich płynących.

Literatura

[1] Rafał Tutaj : Przemysłowe Systemy Komunikacyjne, Konferencja Automation'97 Tom 2
[2] Rafał Tutaj : Przemysłowe systemy komunikacyjne typu Fieldbus, PAR 4'97 s 32-35
[3] Rafał Tutaj : DeviceNet – efektywne rozwiązanie problemu komunikacji typu Fieldbus, Konferencja Automation' 98
[4] Rafał Tutaj : DeviceNet – efektywne rozwiązanie problemu komunikacji typu Fieldbus PAR 4'98
[5] Rafał Tutaj : Różne urządzenia-jedna sieć : sterowanie i pomiary za pomocą DeviceNet Konferencja Automation '99
[6] Rafał Tutaj : Różne urządzenia-jedna sieć : sterowanie i pomiary za pomocą DeviceNet PAR 7-8'99