ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1999
Seria: INFORMATYKA z. 36 Nr kol. 1414

Henryk ZYGMUNT, Grzegorz WRÓBEL, Paweł KWASNOWSKI,
Grzegorz HAYDUK, Marcin JACHIMSKI.
Akademia Górniczo-Hutnicza, Katedra Automatyki Napędu i Urządzeń Przemysłowych.

ZASTOSOWANIE SIECI QNXNET W ROZPROSZONYM SYSTEMIE STEROWANIA WIELKIM PIECEM

Streszczenie. W pracy przedstawiono opis systemu sterowania procesami technologicznymi na wielkim piecu nr 5 w Hucie im. T. Sendzimira S.A. w Krakowie. System ten został opracowany w KANiUP AGH w Krakowie, wykonany przez ZDANiA Sp. z o.o. i uruchomiony w styczniu 1997 r. Jest to hierarchiczny, wielopoziomowy, rozproszony, komputerowy system sterowania, pracujący pod kontrolą systemu operacyjnego czasu rzeczywistego QNX. Cechą wyróżniającą systemu jest to, że wszystkie jego elementy (sterowniki PLC 2000 i komputery PC-Pentium) stanowią węzły podwójnej, automatycznie balansowanej sieci QNXnet, na nośniku typu ETHERNET (częściowo światłowodowym).

APPLICATION OF THE QNXNET NETWORK IN THE DISTRIBUTED CONTROL SYSTEM OF A BLAST FURNACE

Summary. The paper presents description of the technological processes control system of the blast furnace No 5 at the T. Sendzimir S.A. Steelworks in Cracow. The system has been designed and developed at the Institute of Electrical Drives and Industrial Equipment Control of the Technical University of Mining and Metallurgy AGH in Cracow and manufactured by the Experimental Department of Scientific Equipment and Automation ZDANiA Ltd. It has been started-up into the operation in January 1997. It is a hierarchical, multilevel, distributed computer control system working under QNX real time operating system. The specific feature of the system is that all its nodes (PLC-2000 controllers and PC-Pentium computers) are connected by the double, automatically load-balanced QNXnet network using ETHERNET-type media (partially fibre-optic).

1. Wstęp

W latach 1996 do 1998, w ramach Projektu Celowego Nr: 7 **7851.95C/2509** pt.: „PROEKOLOGICZNA MODERNIZACJA WIELKIEGO PIECA nr 5 oraz BADANIA na INSTALACJI DOŚWIADCZALNEJ", został w Katedrze Automatyki Napędu i Urządzeń Przemysłowych opracowany, a następnie wdrożony do produkcji w Zakładzie Doświadczalnym Aparatury Naukowej i Automatyki d. ZDAN AGH Sp. z o.o. oraz do eksploatacji w Hucie im. T. Sendzimira S.A. - **nowoczesny, kompleksowy system sterowania wielkim piecem hutniczym** [1], [2], [4], [5], [8].

System sterowania obejmuje funkcje: załadunku wielkiego pieca, regulacji jego parametrów, sterowanie napędami pomocniczymi oraz odpylaniem, jak również system informacyjny.

Jest to hierarchiczny, wielopoziomowy, rozproszony, komputerowy system sterowania, pracujący pod kontrolą systemu operacyjnego czasu rzeczywistego QNX, zrealizowany za pomocą 10 programowalnych sterowników przemysłowych PLC-2000 (produkcji ZDANiA Sp. z oo.) i 9 komputerów PC - Pentium.

Cechą wyróżniającą systemu jest to, że wszystkie jego elementy (sterowniki PLC 2000 i komputery PC - Pentium) stanowią węzły podwójnej, automatycznie balansowanej sieci QNXnet, na nośniku typu ETHERNET (częściowo światłowodowym).

System kontroluje pracę: **500** napędów mechanizmów wykonawczych oraz przebieg procesów technologicznych poprzez: **4 700** sygnałów dwustanowych, **400** sygnałów analogowych i **50** pętli automatycznej regulacji wielkości technologicznych, takich jak: temperatury, ciśnienia, przepływy gazów i cieczy, składy mieszanin gazowych itp. Około **400** wielkości technologicznych jest gromadzonych w relacyjnej bazie danych SQL. Mogą one być wizualizowane i przedstawiane w formie wykresów w wybranych przedziałach czasu. System umożliwia również sporządzanie raportów: godzinowych, zmianowych, dobowych, tygodniowych, miesięcznych i rocznych - z przebiegu procesu technologicznego.

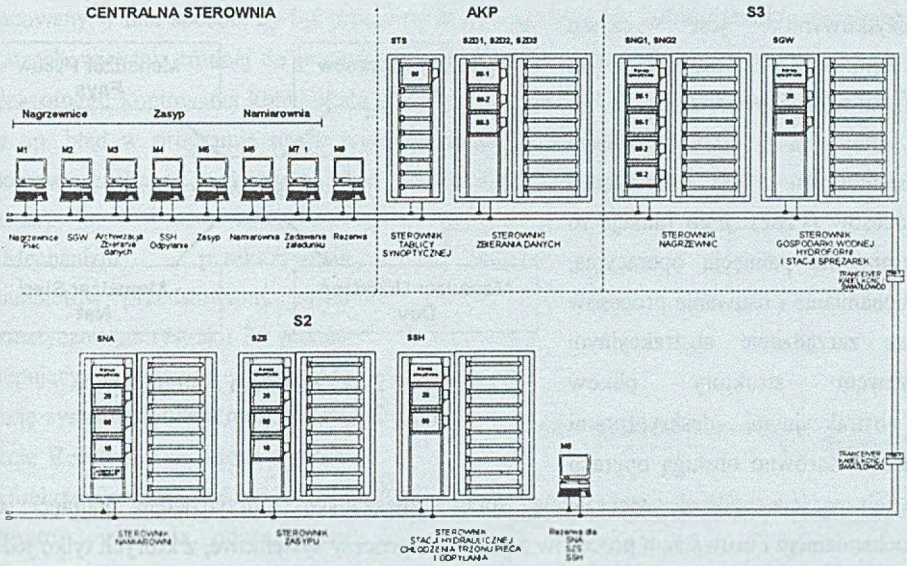
Konfigurację systemu pokazano na rys. 1.

2. Opis architektury QNX

2.1. QNX systemem operacyjnym czasu rzeczywistego o szerokiej gamie zastosowań

QNX jest systemem operacyjnym należącym do rodziny UNIX oraz spełnia wymagania standardu POSIX. Cechą charakterystyczną, wyróżniającą omawiany system operacyjny od

innych systemów z rodziny UNIX, jest możliwość zdeterminowania czasu reakcji na zdarzenia występujące w systemie, określająca go jako system czasu rzeczywistego.



Rys. 1. Konfiguracja systemu
Fig. 1. Configuration of the system

To, co powinno charakteryzować system czasu rzeczywistego, to minimalny i przewidywalny czas odpowiedzi na zdarzenie. W przypadku systemu operacyjnego, zdarzenie najczęściej przychodzi jako przerwanie, a kluczowymi czasami są:

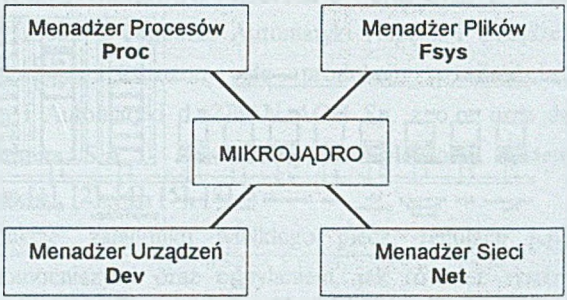
- opóźnienie przerwania (ang. interrupt latency) - czas od momentu zgłoszenia przerwania do rozpoczęcia wykonywania funkcji obsługującej to przerwanie;
- czas przełączania kontekstu (ang. context switching) - czas od momentu zakończenia funkcji obsługi przerwania do podjęcia wykonywania przerwanego procesu.

QNX obsługuje pełen zakres procesorów - od przeznaczonych do aplikacji wbudowanych, do bardzo zaawansowanych: AMD Elan SC300/310/400, AMD386 DE/SE Intel 386 EX, Intel 486, ULP Intel 486, Pentium, Pentium Pro, jak również wiele magistral, m.in.: PCI, CompactPCI, PCMCIA, STD, STD 32, EISA, ISA, MPE (RadiSys), VESA oraz VME.

2.2. Modułowa struktura systemu

Architektura systemu QNX oparta jest na architekturze mikrojądra. Jest ono bardzo niewielkich rozmiarów; ma około 10 KB kodu. Jego zadania to szeregowanie procesów,

przesyłanie komunikatów i sygnałów oraz wywoływanie procedur obsługi przerwań dla procesów, które tego zażądały. Wszystkie inne funkcje systemu operacyjnego są wykonywane przez procesy systemowe, przy czym rozróżnienie między procesami systemowymi a użytkowymi jest czysto funkcjonalne i zależy nieraz od punktu widzenia (rys. 2).



Rys. 2. Architektura systemu QNX
Fig. 2. QNX system architecture

Głównym procesem systemowym jest zarządca procesów (**Proc**). Jego funkcje to gospodarka pamięcią operacyjną, uruchamianie i usuwanie procesów oraz zarządzanie abstrakcyjnym drzewem struktury plików i abstrakcyjnymi deskryptorami plików. Zarówno obsługą operacji wejścia-wyjścia, jak i większością spraw związanych z przydziałem pamięci oraz uruchamianiem i usuwaniem procesów zajmują się procesy systemowe, z których tylko jeden - Proc, zarządca procesów - jest obowiązkowym składnikiem systemu. Ściśle współpracuje on z jądrem, mając jako jedyny proces dostęp do jego obszaru danych. Dzięki temu może na zlecenie innych procesów zmieniać im priorytet, usuwać je z systemu i tworzyć nowe procesy.

Z kolei zarządca sieci (proces o nazwie **Net**) zajmuje się przesyłaniem komunikatów przez sieć, a ściślej - zlecaniem transmisji sieciowych procesom obsługi sprzętu sieciowego. Jego współpraca z jądrem jest więc wyjątkowo ścisła: jądro korzysta z pośrednictwa procesu Net w przekazywaniu komunikatów między procesami na innych węzłach [3].

2.3. Komunikacja międzyprocesowa

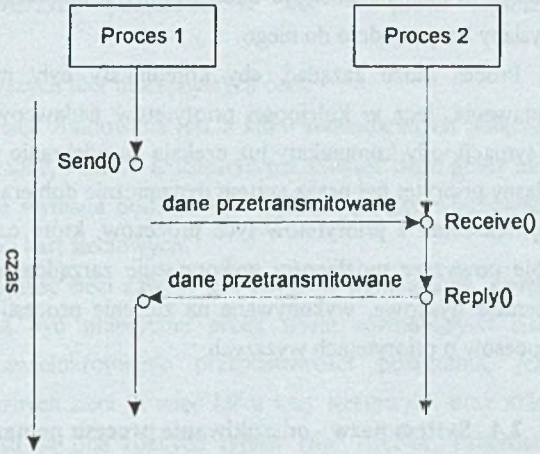
Podstawą całego systemu operacyjnego QNX jest mechanizm przesyłania komunikatów. Mechanizm ten jest elementem wyróżniającym ten system operacyjny spośród innych systemów. Służy on zarówno do przekazywania informacji, jak i do synchronizacji procesów. Dzięki niemu system może mieć przejrzystą, modularną strukturę. Większość funkcji systemowych jest realizowana w ten sposób, że funkcja wysyła komunikat do odpowiedniego procesu systemowego, który następnie obsługuje żądanie i zwraca w odpowiedzi kod błędu.

Komunikat jest ciągiem bajtów o długości określonej przez nadawcę. Nie musi zajmować spójnego obszaru pamięci, bo funkcje systemowe mające do czynienia z komunikatami pozwalają na ich składanie z fragmentów (gdy np. nagłówek komunikatu znajduje się

w jednym miejscu pamięci, a bufor z danymi - w innym). Taki podział może być przeprowadzony zupełnie niezależnie u nadawcy i u adresata.

Przesyłanie komunikatu nie jest tylko przekazaniem danych. Protokół tej operacji został opracowany w taki sposób, by był prosty (m.in. nie zmuszał np. jądra systemu operacyjnego do przechowywania trudnej do przewidzenia liczby komunikatów), efektywny (nie wymagał wielokrotnego kopiowania komunikatu przed jego dotarciem do adresata) oraz niezawodny (aby np. błąd w programie użytkowym nie mógł spowodować sytuacji, w której proces systemowy czeka na wykonanie przez proces użytkowy czynności, której on nigdy nie wykona).

Mechanizm przekazywania komunikatów przedstawiony został schematycznie na rysunku 3 i jest on następujący: nadawca wykonuje funkcję systemową **Send**, a adresat - funkcję **Receive**. Ten, który wywoła swą funkcję pierwszy, wykonuje ją do momentu wywołania odpowiedniej funkcji przez swojego "współrozmówcę"; a system do tego czasu blokuje nadawcę. Po otrzymaniu komunikatu adresat może się zająć analizowaniem go, a nadawca nadal oczekuje, zablokowany. Funkcja **Send** kończy swoje działanie dopiero w momencie, gdy adresat wywoła funkcję **Reply**, przekazującą nadawcy odpowiedź na jego komunikat. Oba procesy biegną od tego momentu niezależnie.



Rys. 3. Synchronizacja procesów przy użyciu przesyłania wiadomości
Fig. 3. Proceses synchronisation by sending message

Z punktu widzenia nadawcy, przesyłanie komunikatu wygląda bardzo prosto - tak jak wywołanie procedury: funkcja **Send** otrzymuje parametry wejściowe (komunikat) i zwraca wyniki w parametrach wyjściowych (odpowiedź). Takie podejście ułatwia pisanie niezawodnych programów, niewrażliwych na subtelne zależności czasowe między współpracującymi ze sobą, współbieżnymi procesami.

Adresat komunikatu ma dużą swobodę, gdyż może odebrać dowolnie wiele komunikatów od innych procesów, zanim zdecyduje się na któryś odpowiedzieć, a odpowiedzi może im wysłać w dowolnej kolejności. Często kolejność tę wyznaczają zdarzenia zewnętrzne - na przykład dane nadchodzące z urządzeń.

Funkcją **Receive** nie trzeba odbierać od razu całego komunikatu. Jeśli tak jest wygodniej, można odebrać tylko początkowy fragment (którego długość trzeba zawczasu podać funkcji), a później w dowolnym momencie (dopóki nadawca czeka na odpowiedź) za pomocą funkcji **Readmsg** wczytać do przygotowanego bufora zawartość komunikatu. W każdym przypadku zawartość komunikatu jest kopiowana bezpośrednio z obszaru pamięci nadawcy do obszaru pamięci adresata: jądro nie musi buforować komunikatu, gdyż jest on bezpieczny u nadawcy, który nie może go zmodyfikować (jest zablokowany w oczekiwaniu na odpowiedź).

Proces może przekazać odebrany komunikat innemu procesowi posługując się funkcją **Relay**. Komunikat zostanie odebrany przez tamten proces dokładnie tak, jak gdyby został wysłany bezpośrednio do niego.

Proces może zażądać, aby komunikaty były mu dostarczane nie w kolejności ich nadawania, lecz w kolejności priorytetów nadawców. Oczywiście, ma to znaczenie tylko w sytuacji, gdy komunikaty już czekają na odebranie w kolejce. Może też zechcieć, by jego własny priorytet był przez system dynamicznie dobierany w taki sposób, by był zawsze równy największemu z priorytetów tych procesów, które czekają na obsłużenie przez ten proces. Obie powyższe możliwości wykorzystuje zarządca plików dyskowych **Fsys**, dzięki czemu operacje dyskowe, wykonywane na zlecenie procesu o niskim priorytecie, nie spowalniają procesów o priorytetach wyższych.

2.4. System nazw - odszukiwanie procesu po nazwie

Aby funkcja systemowa **Send** mogła spełnić swoje zadanie, musi znać identyfikator procesu, do którego chce wysłać komunikat. Jeśli nadawca jest blisko spokrewniony z adresatem (tzn. jest np. jego ojcem lub synem), to może znać jego identyfikator. Jeśli nie, musi umieć go odszukać. W tym przypadku przydatny jest mechanizm zwany przypisywaniem nazw procesom. Proces może przypisać sobie wcześniej ustaloną nazwę - niezbyt długi ciąg znaków (do 32 znaków), rejestrując ją u zarządcy procesów (funkcją **qnx_name_attach**), który będzie następnie udostępniać identyfikator procesu, który zarejestrował daną nazwę procesom pytającym o tę nazwę (przy pomocy funkcji **qnx_name_locate**) [6].

2.5. Protokół QNXnet

System QNX tworzy sieć lokalną złożoną z komputerów, które nazywa się węzłami (ang. node). Dzięki temu otrzymuje się architekturę sterowania rozproszonego. Oczywiście, można tak skonfigurować system, aby narzucić architekturę serwer - stacja robocza, lecz najbardziej efektywne wykorzystanie możliwości systemu uzyskuje się przy zachowaniu równoważności wszystkich węzłów.

Mechanizmy sieciowe zostały wbudowane na najniższym poziomie, dzięki czemu poszczególne komputery, pracujące w lokalnej sieci komputerowej, mogą udostępniać sobie nie tylko urządzenia - dyski, drukarki, terminale, modemy, ale także zlecać sobie nawzajem uruchamianie programów. Dzięki temu możliwe jest przetwarzanie rozproszone, a więc tworzenie programów składających się z procesów komunikujących się między sobą, a rozmieszczonych na różnych węzłach sieci.

FLEET jest unikalnym, ultraszybkim, bardzo elastycznym protokołem komunikacji sieciowej, bazującym na architekturze przesyłania komunikatów systemu QNX. Jego wielorakie możliwości zamieniają maszyny połączone w sieć w jeden logiczny „superkomputer".

Nazwa protokołu pochodzi od pierwszych liter następujących cech:

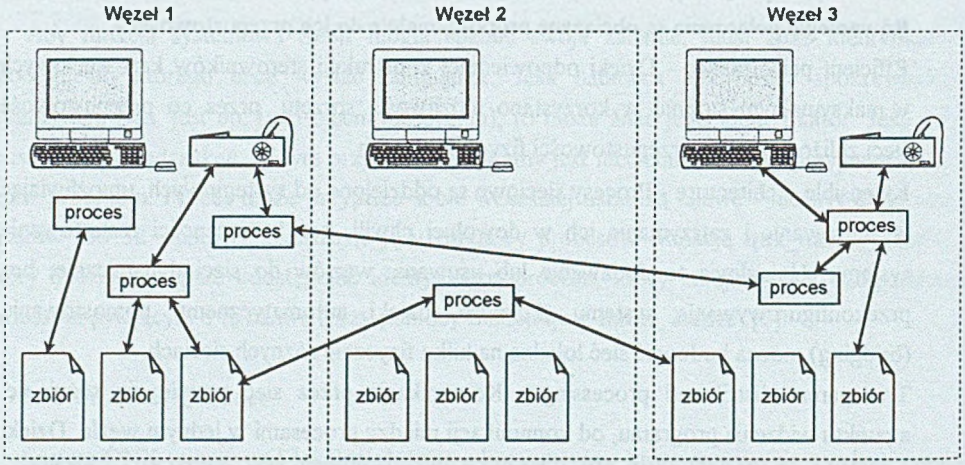
- Fault tolerant networking - Jeśli sieć zbudowana jest z kilku redundantnych połączeń oraz jedno z nich zostanie uszkodzone, FLEET automatycznie skieruje dane przez inną sieć. Dzieje się to "w locie" i nie wymaga dodatkowych aplikacji, dając wbudowaną odporność na uszkodzenia łączy lub kart sieciowych.
- Load balancing on the fly - Wydajność sieci zawsze jest fizycznie ograniczona. Dzięki protokołowi FLEET dane mogą być przesyłane przez wiele równoległych sieci równocześnie, pozwalając na zwielokrotnienie przepustowości połączenia, jeśli zbudowane jest ono z kilku fizycznych sieci (a więc kilku kart sieciowych oraz kilku oddzielnych połączeń), nawet jeśli są one różnych typów (np. ArcNet, Ethernet). Równolegle połączenia są obciążane proporcjonalnie do ich przepustowości.
- Efficient performance - Dzięki odpowiedniej konstrukcji sterowników kart sieciowych w maksymalnym stopniu wykorzystano możliwości sprzętu, przez co przepustowość sieci zbliżona jest do przepustowości fizycznego łącza.
- Extensible architecture - Procesy sieciowe są oddzielone od systemowych, umożliwiając wystartowanie i zatrzymanie ich w dowolnej chwili, bez konieczności restartowania systemu. Umożliwia to dodawanie lub usuwanie węzłów do sieci dynamicznie, bez przekonfigurowywania systemu. Ponadto, dzięki automatycznemu pomostowaniu (bridging) można budować sieć lokalną na kilku fizycznie różnych sieciach.
- Transparent distributed processing - Komunikacja przez sieć prawie nie różni się, z punktu widzenia programu, od komunikacji między procesami w jednym węźle. Dzięki temu w przypadku gdy chcemy uruchomić wieloprocesową aplikację w układzie rozproszonym, nie ma potrzeby jej modyfikowania w celu umożliwienia komunikacji sieciowej.

2.6. Transparentność sieciowa

2.6.1. Transparentność na poziomie systemu plików

Ponieważ większość urządzeń peryferyjnych widocznych jest w systemie plików jako abstrakcyjne pliki w katalogu /dev, udostępniając węzłom, znajdującym się w sieci QNX, drzewo systemu plików, mamy dostęp do plików dyskowych, jak również do większości urządzeń danego węzła. W tym celu w systemie QNX wprowadzono rozszerzenie konwencji nazewnictwa ścieżki dostępu. Mianowicie, jeśli bezwzględna ścieżka dostępu poprzedzona jest dwoma ukośnikami i numerem, oznacza ona ścieżkę dostępu na węźle wskazanym tym numerem. Tak więc jeśli pracujemy w węźle o numerze 2, to lokalny korzeń drzewa systemu plików, dostępny w tym węźle jako /, w całej sieci widoczny jest jako //2. Dodatkowo, aby odwołać się do korzenia drzewa plików własnego węzła, można użyć zera zamiast numeru węzła (ścieżki //0/ i / są więc równoważne). W taki oto prosty sposób mamy dostęp w całej sieci QNX do wszystkich dysków lokalnych, jak również urządzeń znakowych, takich jak porty szeregowe i równoległe (a zarazem urządzenia do nich podłączone), terminale itp. [6].

Jeśli węzeł nie posiada dysku stałego, może uczynić korzeniem swego drzewa plików dowolną kartotekę w sieci - najczęściej będzie to główna kartoteka dysku jakiegoś innego węzła. Przykładową konfigurację przetwarzania rozproszonego uzyskaną dzięki transparentności systemu plików przedstawia rysunek 4.

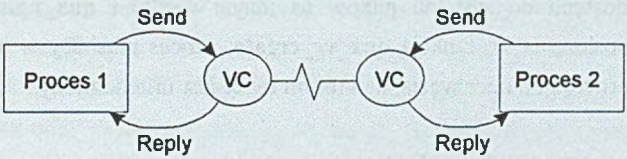


Rys. 4. Przetwarzanie rozproszone w sieci QNX
Fig. 4. Distributed computing in QNX network

2.6.2. Transparentność w komunikacji międzyprocesowej - połączenia wirtualne

Identyfikatory procesów są przydzielane przez system niezależnie w każdym węźle sieci. Może się więc zdarzyć, że dwa procesy w różnych węzłach będą miały jednakowe identyfikatory. Wynika stąd, że aby wysłać komunikat do procesu na innym węźle, trzeba określić adresata w nieco inny sposób, mianowicie, należy utworzyć tzw. połączenie wirtualne (ang. virtual circuit). Jest ono parą dwóch procesów wirtualnych położonych w różnych węzłach (patrz rys. 5).

Połączenie wirtualne można stworzyć przy pomocy funkcji systemowej **qnx_vc_attach**, zwracającej identyfikator procesu wirtualnego, który będzie "przedstawicielem" odległego procesu. Tego identyfikatora będzie można użyć m.in. jako argumentu funkcji **Send** (lub innej, wymagającej jako argumentu identyfikatora procesu). Jednocześnie na drugim węźle sieci tworzony jest drugi proces wirtualny, który będzie reprezentował proces tworzący połączenie w stosunkach z procesem odległym i którego identyfikator otrzyma odległy proces jako wartość funkcji **Receive**, gdy odbierze komunikat przesłany przez połączenie wirtualne.



Rys. 5. Połączenia wirtualne w przesyłaniu wiadomości
Fig. 5. Virtual circuits in sending messages

Procesy wirtualne zawierają bufory, w których system przechowuje kopię komunikatu, aby np. każde użycie funkcji **Readmsg** przez odbiorcę nie wiązało się z koniecznością transmisji przez sieć. Podczas tworzenia połączenia wirtualnego można podać rozmiar buforów, lecz przesłanie większego komunikatu powoduje automatyczne ich powiększenie.

Tworzeniem i usuwaniem połączeń wirtualnych zajmują się wspólnie **Net** i **Proc**. Zarządca procesów, po otrzymaniu odpowiedniego żądania, nakazuje zarządcy sieci nawiązanie dialogu z jego odległym partnerem. W trakcie tego dialogu odległy zarządca sieci informuje tamtejszego zarządcę procesów o żądaniu utworzenia połączenia, a ten albo żądanie przyjmuje, tworząc proces wirtualny, albo je odrzuca.

2.7. Nazwy globalne

Wspomniany wcześniej mechanizm przypisywania nazw procesom przydatny jest również w przypadku komunikacji procesów znajdujących się na różnych węzłach. Mianowicie, jeśli rejestrowana funkcją **qnx_name_attach** nazwa zaczyna się od znaku „/", to jest uważana za nazwę globalną - widoczną dla funkcji **qnx_name_locate** w całej sieci. Tak więc funkcja ta używana jest zarówno dla zlokalizowania procesów znajdujących się na lokalnym, jak

i zdalnym węźle; funkcja **qnx_name_locate** w przypadku odwołań do nazw globalnych automatycznie tworzy połączenie wirtualne, tym sposobem zawsze zwracając identyfikator procesu gotowy do użycia np. w funkcji **Send**.

Nazwy lokalne są obsługiwane przez zarządcę procesów. Aby natomiast funkcjonowały nazwy globalne, na dowolnym węźle w sieci musi pracować zarządca nazw globalnych - proces systemowy o nazwie **nameloc**.

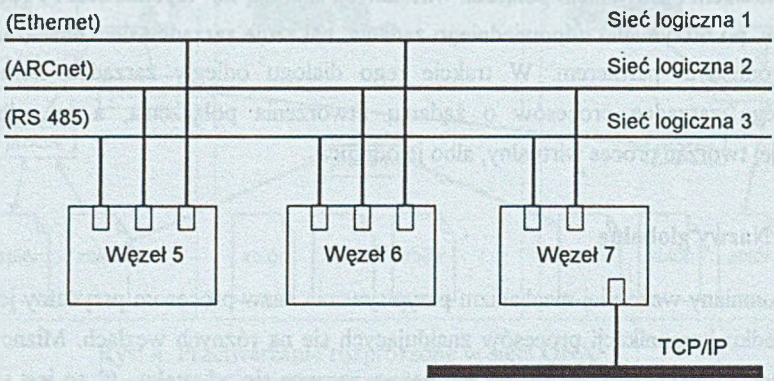
Tak więc, w większości przypadków, programy nie będą używały funkcji **qnx_vc_create**, gdyż połączenia wirtualne stworzone zostaną automatycznie przez funkcje **open** (w przypadku dostępu do systemu plików na innym węźle) i **qnx_name_locate**. Aby zaszła potrzeba posłużenia się funkcją **qnx_vc_create**, proces musiałby w inny sposób poznać identyfikator procesu i numer węzła, na którym tenże jest uruchomiony.

2.8. Dublowanie sieci

Dzięki modularności niezwykle proste jest uzyskanie redundantnych połączeń sieciowych. Uzyskuje się w ten sposób znaczne podwyższenie niezawodności wykorzystywanego połączenia sieciowego. Polega to na:

- wykonaniu zwielokrotnionych połączeń sieciowych (kabel koncentryczny ze złączami BNC, skrętka, światłowód, połączenie telefoniczne lub inne),
- wyposażeniu danych węzłów w wymaganą ilość urządzeń komunikacyjnych (np. karty sieciowe, modemy),
- uruchomieniu na każdym z tych węzłów stosownych do zainstalowanych urządzeń programów obsługi.

Przykładowe, wielokrotne połączenia sieciowe przedstawia rysunek 6.



Rys. 6. Wielokrotne połączenia sieciowe
Fig. 6. Multiple networks connections

2.9. Wykorzystanie własności sieciowych systemu QNX w systemie sterowania

2.9.1. Komunikacja i architektura typu klient-serwer

Zastosowanie rozproszonego systemu sterowania wielkim piecem pozwala na rozmieszczenie poszczególnych węzłów systemu jak najbliżej urządzeń wykonawczych (programowalne sterowniki PLC-2000 umieszczone są bezpośrednio w ciągu szaf zasilających i sterowniczych w stycznikowniach rozmieszczonych po obydwu stronach pieca), tzn. w rejonie namiarowni wsadu i w rejonie nagrzewnic dmuchu (pod centralną sterownią). Dzięki temu uzyskuje się najkrótsze połączenia pomiędzy obiektami sterowanymi i sterownikami. Komunikacja pomiędzy węzłami sterującymi oraz między nimi a systemem nadrzędnym przeniesiona jest zatem na poziom systemu QNX. Wymiana informacji i sygnałów sterujących pomiędzy poszczególnymi węzłami odbywa się przy pomocy protokołu QNXnet. Bardzo duża szybkość transmisji (zbliżona do teoretycznej szybkości fizycznego łącza) oraz determinizm w przesyłaniu danych pozwalają na użycie sieci typu ETHERNET i architektury typu magistralowego do połączenia zarówno komputerów systemu nadrzędnego (wizualizacja, sterowanie operatorskie, zbieranie danych), jak i węzłów poziomu sterowania bezpośredniego. Pewnym ograniczeniem jest tutaj użycie niedeterministycznego medium transmisyjnego (Ethernet), jednak przy zachowaniu wystarczająco niskiego (poniżej 50%) średniego obciążenia sieci i przy odpowiednim ustawieniu priorytetów pracy dla poszczególnych aplikacji sterujących można uznać, że komunikacja rozpatrywana z poziomu aplikacji będzie deterministyczna.

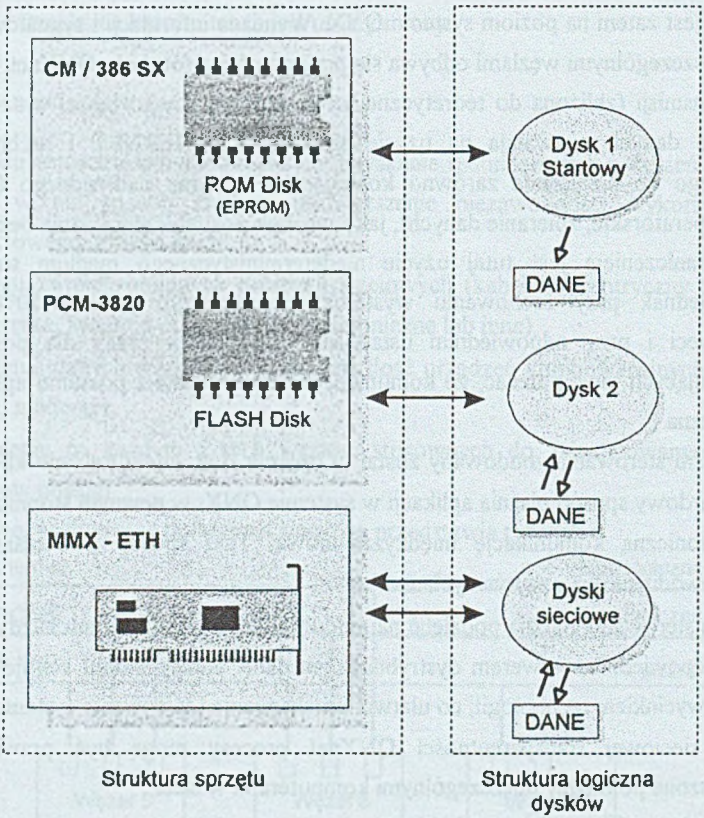
Cały system sterowania zbudowany został w oparciu o architekturę typu klient - serwer. Jest to standardowy sposób pisania aplikacji w systemie QNX, w pewnym stopniu wymuszony przez synchroniczną komunikację międzyzadaniową. Taki sposób powiązania pomiędzy procesami powoduje kilka pozytywnych skutków:

- zadania sterowania można podzielić na moduły (klientów), z których każdy komunikuje się z odpowiednim serwerem dystrybuującym dane. Każdy moduł steruje stosunkowo małym wycinkiem technologii, co ułatwia uruchamianie i testowanie systemu sterowania;
- dzięki sieciowej transparentności QNXnet procesy mogą być prawie dowolnie przenoszone pomiędzy poszczególnymi komputerami w sieci;
- synchroniczna komunikacja międzyzadaniowa powoduje automatyczną synchronizację procesów sterujących;
- bardzo łatwa jest rozbudowa systemu – serwery mogą przyjmować zlecenia od dowolnej ilości procesów klientów, które mogą być dodawane i uruchamiane nawet dynamicznie;
- nie istnieje, występująca w systemach wielozadaniowych, możliwość zablokowania procesów oczekujących wzajemnie na siebie (tzw. dead-lock).

2.9.2. Węzły wbudowane (embedded) - jednorodność systemu i komunikacji

Sterowniki programowalne PLC-2000, które użyte są na poziomie sterowania bezpośredniego, zbudowane są z dwóch podsystemów połączonych ze sobą wspólną magistralą - podsystemu sterownika logicznego i podsystemu komputera ACP-03. Ten ostatni łączy w sobie funkcje regulatora wielkości technologicznych i komunikacyjne. Zbudowany jest na bazie komputera przemysłowego w standardzie PC/104, w skład którego wchodzą [7]:

- płyta główna z procesorem 386SX;
- karta FlashDisk;
- dwie karty sieciowe Ethernet.



Rys. 7. Komputer CM/386 SX z rozproszonymi zasobami dyskowymi
Fig. 7. CM/386 SX computer and its distributed disks

Komputer ten pracuje pod kontrolą systemu QNX. Jego konfigurację przedstawiono na rysunku 7. Dysk startowy zaimplementowany jest jako ROM-dysk (EPROM 512KB). Znajdują się na nim wszystkie moduły systemu potrzebne do pracy komputera i do

komunikacji sieciowej. Na dysku typu Flash (czyli zapisywalnym) umieszczone są aplikacje sterujące. Dzięki użyciu tego samego systemu operacyjnego jak dla komputerów nadrzędnych uzyskano całkowitą jednorodność zarówno komunikacji międzyzadaniowej, jak i systemu plików. Mechanizmy QNXnet pozwalają na dostęp do dysków (ROM i Flash) komputera ACP-03 z dowolnego węzła w sieci sterującej, co umożliwia bardzo prostą obsługę wszelkich zmian w aplikacjach sterujących.

2.9.3. Możliwość przenoszenia aplikacji

Istniejące w systemie QNX mechanizmy pozwalają na zdalne uruchamianie procesów na dowolnych węzłach sieci. Procesy mogą być przy tym pobierane z dysku każdego z komputerów. Równocześnie komunikacja pomiędzy procesami nie zależy od miejsca uruchomienia poszczególnych zadań. Wynika z tego prawie pełna dowolność przenoszenia aplikacji pomiędzy poszczególnymi węzłami sieci. Jedynym ograniczeniem są procesy odwołujące się bezpośrednio do sprzętu (układów wejścia/wyjścia), które z oczywistych względów muszą pracować na konkretnych węzłach.

2.9.4. Dublowanie sieci

W celu zapewnienia większego bezpieczeństwa przesyłania danych sterujących zastosowana została podwójna sieć Ethernet (częściowo kabel koncentryczny, częściowo kabel światłowodowy). W tych przypadkach, gdzie to było możliwe, poszczególne nitki sieci prowadzone są różnymi drogami, co zmniejsza prawdopodobieństwo uszkodzenia obu połączeń. Ponieważ mechanizmy obsługi dublowanych połączeń sieciowych wbudowane są w system operacyjny QNX, zatem zastosowanie podwójnego połączenia nie powoduje żadnych dodatkowych wymagań z punktu widzenia aplikacji sterujących. Dodatkowym pozytywnym efektem jest dwukrotne zwiększenie przepustowości połączenia.

W przypadku awarii karty sieciowej lub fizycznego połączenia (np. przecięcie kabla) następuje automatyczne przełączenie całego ruchu na sprawną nitkę sieci; po usunięciu uszkodzenia system samodzielnie stwierdzi sprawność połączenia i zacznie wykorzystywać oba łącza.

2.9.5. Otwartość systemu - możliwości rozbudowy

Biorąc pod uwagę sieciową transparentność i modułowość systemu QNX, można zauważyć, że przy odpowiednim zaprojektowaniu architektury systemu sterowania – zastosowanie powiązań typu klient - serwer przy możliwości dołączania dodatkowych klientów w trakcie pracy - tworzy to system otwarty i wyjątkowo elastyczny. W przypadku konieczności rozbudowy sprzętowej dołączanie dodatkowych komputerów (czy to poziomu sterowania bezpośredniego, czy nadrzędnego) może być realizowane w trakcie pracy systemu,

bez zakłócania technologii. Również rozbudowa programowa nie powoduje zaburzeń w pracy systemu sterowania, tzn. że wizualizacja, sterowanie operatorskie czy programy sterowania bezpośredniego mogą być uruchamiane i testowane w trakcie pracy systemu, najczęściej bez konieczności zmian w programach już istniejących, z którymi mają się komunikować (oczywiście przy odpowiednim napisaniu tych programów).

2.9.6. Możliwość komunikacji przy użyciu protokołu TCP/IP

System operacyjny QNX umożliwia zaimplementowanie komunikacji przy użyciu protokołu TCP/IP. Można zatem podłączyć system sterowania do ogólnego systemu informatycznego Huty. Ze względów bezpieczeństwa połączenie takie powinno być zrealizowane przez zastosowanie w wybranym węźle dodatkowego połączenia Ethernetowego (trzecia karta sieciowa) – dostęp do pozostałych węzłów odbywałby się przez ten komputer, który służyłby jako swego rodzaju pomost TCP/IP-QNXnet. Takie połączenie nie powodowałoby zakłóceń w pracy sieci sterującej przez niedeterministyczny protokół TCP/IP udostępniając jednocześnie na zewnątrz dowolne, ale ściśle wybrane dane.

3. Podsumowanie

Pomyślny przebieg realizacji tak poważnego zadania, jakim było opracowanie i wdrożenie do produkcji oraz eksploatacji nowoczesnego, bardzo dużego, rozproszonego systemu sterowania wielkim piecem oraz potwierdzona w okresie dwuletniej eksploatacji systemu, jego wysoka niezawodność **(dyspozyjność ponad 99,88%)** - świadczą bardzo pozytywnie o możliwościach polskich naukowców, konstruktorów i programistów.

LITERATURA

1. Kwasnowski P., Seńkowski J., Zygmunt H.: Wielopoziomowy system sterowania procesami technologicznymi wielkiego pieca Nr 1 w Hucie Koszyce. X Krajowa Konferencja Automatyki, Lublin, 21 ÷ 24.VI.1988.
2. Zygmunt H., Seńkowski J., Kwasnowski P., Wróbel G., Jachimski M., Zygmunt M.: Industrial Application of Programmable Control Systems Developed in AGH Kraków. Proceedings of International Conference ED&PE`92, Kosice 1992, Vol. 1, pp. 25-31.
3. QNX System Architecture. QNX Software Systems Ltd., Canada, Ontario 1993.
4. Zygmunt H., Seńkowski J., Kwasnowski P., Wróbel G., Jachimski M., Zygmunt M., Mikoś Z.: Doświadczenia z uruchomienia przemysłowego systemu sterowania napędów i technologicznych procesów wielkiego pieca. Materiały XII KKA, Gdynia 6 ÷ 8 IX.1994, s. 5 8 ÷ 65.
5. Zygmunt H., Kwasnowski P., Zygmunt M., Mikoś Z., Wróbel G., Obst R., Trunkat P., Spicka I.: Construction and Programming of the microcomputer based controllers of the analogue variables. Materiały: International Conference on Electrical Drives and Power Electronics ED&EP`94, the High Tatras, Slovakia, October 18 ÷ 20, 1994, vol. 2, pp. 471 ÷ 476.
6. Lerch W.: System operacyjny QNX. Wersja 4.2. Wyd. Quantum Corp., Wrocław 1995.
7. Wróbel G., Hayduk G.: Przemysłowa aplikacja układu sterowania pracująca pod kontrolą systemu operacyjnego QNX. Software 12/97, Warszawa, grudzień 1997 r. (str.: 42 - 45).
8. Zygmunt H., Kwasnowski P., Seńkowski J., Wróbel G., Jachimski M., Mikoś Z.: System sterowania Wielkim Piecem nr 5 w HTS S.A. Kraków, czerwiec 1998 r. (opracowanie niepublikowane, 87 stron + 53 rysunki).

Recenzent: Prof. dr hab. inż. Adam Mrózek

Wpłynęło do Redakcji 14 maja 1999 r.

Abstract

In the years 1996 to 1998, an **innovative, multitasking, multilevel system of the blast furnace control** has been designed and developed at the Institute of Electrical Drives and Industrial Equipment Control. The system has been introduced to manufacturing at the Experimental Department of Scientific Equipment and Automation Ltd. and subsequently to operation at the T. Sendzimir S.A. Steelworks in Cracow

The control system performs the following functions: the blast furnace charging, process parameters control, auxiliary drives and dedusting control, and information system. It is a hierarchical, multilevel, distributed computer control system working under QNX real time operating system, employing 10 programmable logic controllers PLC-2000 (manufactured by ZDANiA Ltd.) and 9 PC-Pentium computers.

The specific feature of the system is that all it's nodes (PLC-2000 controllers and PC-Pentium computers) are connected by the double, automatically load-balanced QNXnet network using ETHERNET-type media (partially fibre-optic).

The system controls **500** drives of executive equipment and runs technological process by means of **4 700** logical signals, **400** analogue signals and **50** automatic control loops of process quantities like temperatures, pressures, gas and liquids flow rates, compositions of gases mixtures, etc. About **400** process quantities are stored in relation database SQL. They can be visualised and presented in form of diagrams in selected time intervals.

Distributed nature of the blast furnace control system allows to place PLC controllers near the technological devices. Information and control signals are exchanged between controllers by the QNXnet protocol. This type of control processes interconnection has several advantages:

- control tasks can be divided into modules working in client - server architecture
- processes can be run on any node because of the network transparency of the QNX
- processes are automatically synchronised because of the Synchronic Inter Process Communication
- next tasks can be easily incorporated into the system - client processes can be added dynamically
- due to the client - server architecture there is no possibility of the „dead-lock"
- automatic balance of the multiple network connections incorporated into the operation system allows for easy maintenance of the redundant connections.