STUDIA INFORMATICA 2003
Volume 24 Number 3 (55)

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

POŁĄCZENIE KOMUNIKACJI PERIODYCZNEJ I ZDARZENIOWEJ W ZDALNYM MONITOROWANIU PROCESÓW TECHNOLOGICZNYCH POPRZEZ SIEĆ INTERNET/INTRANET

Streszczenie. W referacie przedstawiono rozwiązanie zastosowane w sterownikach internetowych eLog-01 [3], umożliwiające komunikację pomiędzy nimi, której cechą charakterystyczną jest jednoczesna periodyczność i zdarzeniowość. Dzięki temu tworzy się sieć sterowników internetowych, umożliwiająca w pełni rozproszone sterowanie i monitorowanie procesów technologicznych z wykorzystaniem technologii Internet/Intranet oraz przeglądarek www. Wyjaśniona została zasada, według jakiej odbywa się komunikacja i przedstawiona krótka analiza jej zalet i wad. Opisano również przykład zastosowania przemysłowego.

Słowa kluczowe: sterownik internetowy, rozproszone monitorowanie i sterowanie, przeglądarki www.

COMBINATION OF THE EVENT-DRIVEN AND PERIODIC COMMUNICATION IN REMOTE INDUSTRIAL PROCESS MONITORING VIA INTERNET/INTRANET

Summary. The paper presents a solution applied to internet monitoring devices **eLOG**™ [3], which enables them to communicate. The characteristic feature of the solution is that the communication is simultaneously both event-driven and periodic. In consequence a network of internet controllers is created, which allows for a fully distributed control and monitoring, using the Internet technology and, in the most simple case, web browsers. Rules of the communication were given in the paper, together with a short discussion of advantages and disadvantages. An example of industrial application of the solution is also given.

Keywords: internet controller, distributed monitoring and control, web browsers.

1. Wprowadzenie

Od pewnego czasu miniaturyzacja i dostępność cenowa elementów sieciowych spowodowały, że daje się zauważyć trend wyposażania różnego rodzaju urządzeń w interfejs ethernetowy, dając tym samym – popularnie nazywany – dostęp do Internetu. Możliwe stają się w ten sposób dodatkowe wykorzystania często istniejącej już infrastruktury sieciowej i realizacja nowych zadań mniejszym kosztem, a jednocześnie metodami dającymi większe możliwości. Jednym z takich rozwiązań jest urządzenie o nazwie eLOG™, będące sterownikiem wyposażonym w kilka wejść/wyjść analogowych i dwustanowych oraz interfejs ethernetowy [1], [2], [3]. W dalszej części referatu będzie on nazywany sterownikiem internetowym.

Jego możliwości, jak w wielu tego typu urządzeniach, wynikają z zastosowanego w nim oprogramowania. Sterownik został opracowany, a następnie sprawdzony zarówno w warunkach laboratoryjnych jak i w kilku aplikacjach przemysłowych. W aplikacjach tych odbywała się periodyczna komunikacja pomiędzy sterownikami internetowymi (eLogerami), a komputerem nadrzędnym, spełniającym funkcję serwera. W związku z tym, że w kolejnej aplikacji pojawiła się potrzeba wymiany informacji oraz przesyłania pewnych rozkazów również pomiędzy poszczególnymi sterownikami internetowymi, funkcje przez niego realizowane zostały wzbogacone o możliwość takiej komunikacji. Cechą charakterystyczną tej komunikacji jest jej jednoczesna periodyczność i zdarzeniowość. Dzięki wprowadzeniu komunikacji pomiędzy omawianymi sterownikami istnieje możliwość stworzenia całej sieci sterowników internetowych, umożliwiającej w pełni rozproszone monitorowanie i sterowanie z wykorzystaniem technologii Internet/Intranet oraz przeglądarek www (jednak nie ograniczając się tylko do nich).

2. Charakterystyka sterownika internetowego eLOG™

Najkrócej i najskuteczniej sterownik internetowy można scharakteryzować z dwóch punktów widzenia: sprzętu oraz oprogramowania. Przedstawiona w niniejszym referacie charakterystyka została właśnie w ten sposób przeprowadzona.

2.1. Sprzęt

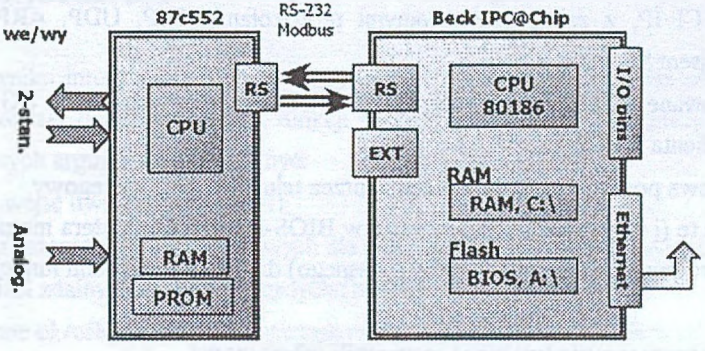
Opisywany sterownik internetowy został zbudowany w oparciu o dwa mikrokomputery:

1. Mikrokomputer obsługi we/wy, którym w omawianym przypadku jest jednoukładowy mikrokomputer ośmiobitowy – 87c552 (będący odmianą mikrokomputera 8051).

2. Jednoukładowy mikrokomputer hybrydowy IPC@CHIP (firmy Beck IPC GmbH) [9], o wymiarach: 24×44×9,5mm, zawierający m. in.:
   - 16 - bitowy procesor 80186 (20 MHz),
   - 515 KB pamięci Flash (zawiera ona BIOS oraz emulowany dysk nieulotny),
   - wyprowadzenia udostępniające m.in. interfejsy: Ethernet 10BaseT, 2×RS-232, I^2C,
   - funkcję kontroli działania (watchdog).

Sterownik internetowy można opisać wyrażeniem: **87c552 + IPC@CHIP = eLOG™**.

Składowe mikrokomputery sterownika internetowego komunikują się poprzez wewnętrzne połączenie szeregowe protokołem zbliżonym do protokołu Modbus. Wewnętrzna struktura sterownika została przedstawiona na rys. 1:



Rys. 1. Struktura sterownika internetowego
Fig. 1. Architecture of the internet controller

2.1.1. Funkcje mikrokomputera obsługi wejść/wyjść

Zadania tego 8-bitowego mikrokomputera jednoukładowego (87c552) to:

- bezpośrednia obsługa wejść/wyjść sprzętowych (czterech wejść analogowych 4÷20mA lub 0÷2,5V w 12-bitowej rozdzielczości, ośmiu beznapięciowych wejść stykowych oraz dwóch wyjść przekaźnikowych),
- implementacja czterech liczników (cztery wejścia dwustanowe mogą pełnić funkcje liczników zliczających zbocza: narastające i/lub opadające),
- zapis/odczyt bateryjnie podtrzymywanego zegara czasu rzeczywistego.

Wszystkie te funkcje dostępne są poprzez wewnętrzną komunikację szeregową pomiędzy mikrokomputerem obsługi we/wy (pełniącym rolę urządzenia Slave) a mikrokomputerem internetowym IPC@CHIP (pełniącym rolę urządzenia Master).

2.1.2. Funkcje jednoukładowego mikrokomputera internetowego (IPC@CHIP)

Drugi z mikrokomputerów tworzących sterownik internetowy pełni funkcje nadrzędne i stanowi serce sterownika. Zawiera on dwa rodzaje oprogramowania:

- Oprogramowanie sprzętowe (ang. firmware), które stanowi BIOS, nazywane tu również systemem operacyjnym czasu rzeczywistego (w skrócie: @CHIP-RTOS).
- Oprogramowanie użytkownika (aplikacyjne), zwane też oprogramowaniem własnym.

Oprogramowanie sprzętowe (BIOS) posiada wbudowane następujące funkcje:

- jednoczesne wykonywanie do kilkudziesięciu zadań (wywoływanych z programów *.exe),
- korzystanie z komunikacji międzyzadaniowej takiej jak np.: semafory,
- stos TCP/IP, z zaimplementowanymi protokołami: TCP, UDP, ARP, ICMP oraz interfejsem Sockets,
- wbudowane aplikacje serwerów: HTTP (z obsługą własnych funkcji CGI), FTP, telnet oraz klienta DHCP,
- DOS-owa powłoka komend dostępna przez telnet lub port szeregowy.

Wszystkie te (i inne) możliwości zawarte w BIOS-ie mikrokomputera mogą być używane z poziomu oprogramowania użytkownika (własnego) dzięki odpowiednim funkcjom API.

2.2. Oprogramowanie tworzące sterownik internetowy

Na oprogramowanie sterownika internetowego składa się program wykonywany przez mikrokomputer 87c552 (obsługujący fizyczne wejścia i wyjścia oraz pełniący rolę urządzenia Slave na wewnętrznym połączeniu szeregowym) oraz oprogramowania pracujące na mikrokomputerze IPC@CHIP: sprzętowe (ang. firmware) – BIOS i oprogramowanie użytkowe (zwane też: aplikacyjne lub własne).

Dzięki zgodności formatu pliku wykonywalnego w @CHIP-RTOS z systemem DOS uzyskano możliwość tworzenia oprogramowania na komputerze typu desktop PC w dowolnym środowisku, mającym możliwość kompilacji na procesor 80186 (są to środowiska: Turbo Pascal i Microsoft C/C++ lub Borland C/C++).

Opracowane zostały własne programy aplikacyjne: NXP_MLgr.exe, eLogSvr.exe oraz eLogCGI.exe, pracujące na sterowniku i uruchamiane automatycznie przy jego starcie, które umożliwiają:

- komunikację z mikrokomputerem wejść/wyjść 87c552,
- odczyt i zapis wejść/wyjść za pomocą prostych stron www,

- konfigurację wszystkich parametrów (takich jak tekstowe nazwy wejść, jednostki fizyczne, aktualny czas, odpowiednie adresy e-mailowe, zdefiniowanie funkcji zdarzeń i przypisanie im wymaganych akcji) przy użyciu przeglądarki www,
- realizację funkcji zdarzeń oraz akcji do nich przypisanych.

Sterownik internetowy dzięki własnemu oprogramowaniu może być używany przy użyciu jedynie przeglądarki www, obsługującej aplety stworzone w języku Java. Odpowiednie aplety osadzone są na głównej stronie www (zapewniającej bieżący monitoring) oraz stronie służącej do konfiguracji funkcji zdarzeń.

3. Funkcje zdarzeń

W sterowniku internetowym wprowadzono możliwość programowania prostych strategii sterowania poprzez definiowanie tzw. funkcji zdarzeń (maksymalnie 10 funkcji). Są to funkcje logiczne, których argumentami mogą być:

- stany wejść dwustanowych (DI),
- przekroczenia progów alarmowych dla wejść analogowych (AI.HiAlm i AI.LoAlm),
- wartości zdalnych funkcji logicznych (NDI),
- zmienne określające poprawność połączenia zdalnych funkcji logicznych (SNDI).

Każdy z tych argumentów może mieć przypisany czas opóźnienia. Dozwolone w definiowaniu funkcji operatory to: iloczyn logiczny (*), suma logiczna (+) i zaprzeczenie (!).

Przykładowa funkcja w postaci ***F(0)=DI(3)[1s]*!AI(1).HiAlm[5m]*** przyjmie wartość prawdy logicznej, gdy spełniony będzie warunek: jeśli trzecie wejście dwustanowe (***DI(3)***) będzie aktywne przez co najmniej jedną sekundę (***[1s]***) i jednocześnie (iloczyn logiczny: ******) nie będzie (***!***) przekroczony wysoki próg (***HiAlm***) pierwszego wejścia analogowego (***AI(1)***) przez pięć minut (***[5m]***).

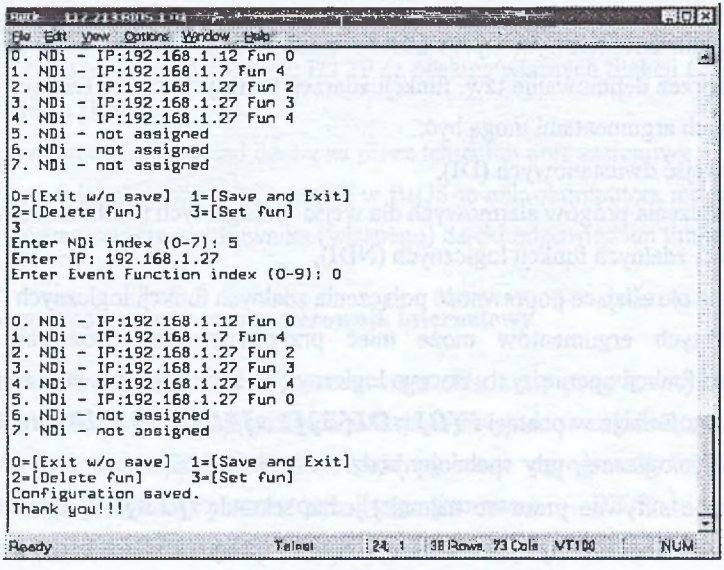
Funkcje te zostały nazwane funkcjami zdarzeń, ponieważ chwila, gdy wartość takiej funkcji zmieni się z *fałsz* na *prawda*, jest uznawana za zdarzenie (w domyśle: spełnienia się funkcji logicznej).

Jednak same funkcje logiczne nie miałyby żadnego znaczenia, gdyby nie fakt, że użytkownik może przypisać do definiowanej funkcji akcje. Mogą nimi być:

- wysłanie określonego komunikatu (do wyboru jest: e-mail, SMS lub główna strona www) oraz
- załączenie lub wyłączenie jednego z wyjść przekaźnikowych (natychmiast lub po zadanym interwale czasowym, do następnej zmiany lub na określony czas, opcjonalnie skracany w chwili, gdy funkcja przyjmie wartość *fałsz*).

Jak wcześniej wspomniano, argumentami funkcji zdarzeń mogą być również wartości zdalnych funkcji logicznych (NDI) oraz zmienne określające poprawność połączenia zdalnych funkcji logicznych (SNDI). Ich konfigurację umożliwia program NDISetup.exe (uruchamiany na sterowniku w sesji telnet) poprzez przypisanie poszczególnym zmiennym NDI adresu IP zdalnego sterownika oraz numeru funkcji na zdalnym sterowniku, którą chcemy powiązać z daną zmienną NDI.

Po restarcie sterownika, na którym zdefiniowane zostały zmienne NDI, odpowiednie zmienne SNDI przyjmą wartość *prawda*, jeśli komunikacja pomiędzy sterownikami będzie poprawnie nawiązana (lub *fałsz* jeśli wystąpi brak komunikacji), a w przypadku poprawnej komunikacji wartości zmiennych NDI będą odpowiadały wartościom zdalnych funkcji zdarzeń. Przykładową sesję konfiguracji zmiennych NDI pokazano na rys. 2.



Rys. 2. Konfiguracja zmiennych NDI
Fig. 2. Configuration of NDI variables

3.1. Zasada transmisji wartości zdalnych funkcji zdarzeń

W przesyłaniu wartości funkcji zdarzeń uczestniczą dwa sterowniki: generujący funkcję zdarzeń (w skrócie SG, pełniący rolę serwera udostępniającego wartości funkcji zdarzeń) oraz odpytujący o wartość zdalnej funkcji zdarzeń (SO, pełniący rolę klienta odpytującego SG i zdarzeniowo otrzymujący wartości funkcji).

Zasada działania komunikacyjnej części sterownika generującego - SG jest następująca:

- przez cały czas sterownik nasłuchuje żądań rejestracji klientów chcących otrzymywać powiadomienia zmiany wartości funkcji zdarzeń. W odpowiedzi na żądanie rejestracji, SG wysyła wartość funkcji. Tworzy się w ten sposób tabela (osobna dla każdej funkcji zdarzeń) zarejestrowanych klientów;
- jeśli wartość funkcji zdarzeń ulegnie zmianie, SG roześle powiadomienia o tym fakcie (komunikacja zdarzeniowa) do wszystkich zarejestrowanych klientów – zmieniając w efekcie wartości wszystkich powiązanych z daną funkcją zdarzeń zmiennych NDI;
- czas ważności rejestracji klienta jest ograniczony do 60 sekund. Po upływie tego czasu (oraz dodatkowych 15 sekund marginesu) wpis rejestracyjny jest automatycznie usuwany (a w efekcie klient jest wyrejestrowywany z tabeli danej funkcji zdarzeń, zwalniając miejsce w tabeli).

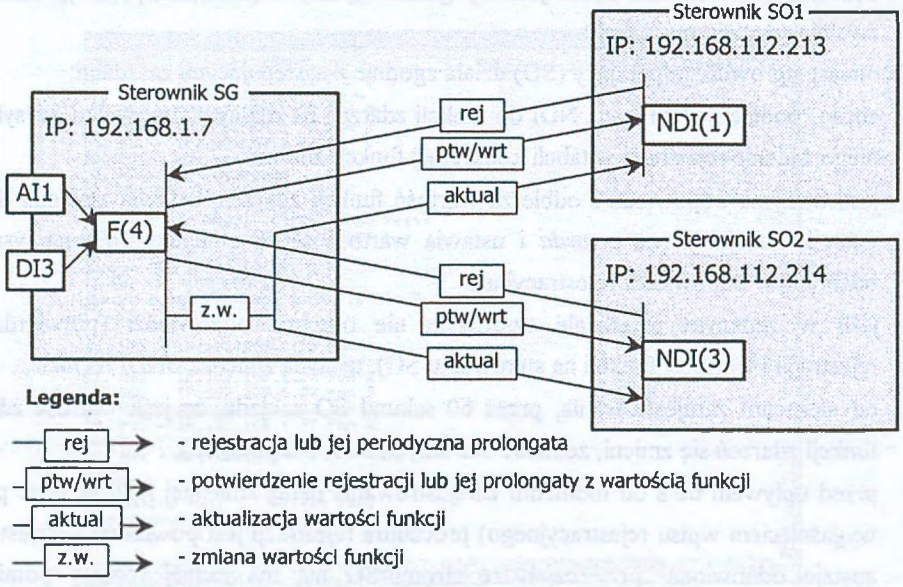
Natomiast sterownik odpytujący (SO) działa zgodnie z następującymi zasadami:

- chcąc "podłączyć" zmienną NDI do funkcji zdarzeń na zdalnym sterowniku, wysyła do niego żądanie rejestracji w tabeli konkretnej funkcji zdarzeń,
- jeśli otrzyma odpowiedź i odbierze wartość funkcji zdarzeń, ustawia zmienną SNDI (stan komunikacji) na *prawda* i ustawia wartość samej zmiennej NDI na wartość odebraną w odpowiedzi rejestracyjnej,
- jeśli w zadanym przedziale czasowym nie otrzyma odpowiedzi (potwierdzenia rejestracji i wartości funkcji na sterowniku SG), ustawia zmienną SNDI na *fałsz*,
- od momentu zarejestrowania, przez 60 sekund SO zakłada, że jeśli wartość zdalnej funkcji zdarzeń się zmieni, zostanie ona natychmiast mu przesłana,
- przed upływem 60 s od momentu zarejestrowania danej zmiennej NDI (a więc przed wygaśnięciem wpisu rejestracyjnego) procedura rejestracji jest powtarzana (rejestracja zostaje odnowiona, przy czym ze strony SO nie ma żadnej różnicy pomiędzy pierwszym i następnymi żądaniami rejestracyjnymi).

Należy zauważyć, iż w wyniku otrzymujemy sytuację, w której periodyczne odnawianie rejestracji jest jednocześnie odpytywaniem o wartość zdalnej funkcji i sprawdzaniem stanu komunikacji (sterownik SO ma możliwość sygnalizowania braku łączności ze sterownikiem SG wartością zmiennej SNDI). Ewentualny brak komunikacji będzie wykryty przy następnym odnawianiu rejestracji.

Na rysunku 3 przedstawiono przykładową sytuację, w której sterownik SG ma zaprogramowaną funkcję F(4), zmieniającą się w zależności od wejść AI1 i DI3. Natomiast sterowniki SO1 i SO2 mają zdefiniowane zmienne NDI(1) i NDI(3) tak, by ich wartości były powiązane z wartością funkcji na sterowniku SG.

W pierwszej kolejności oba sterowniki SO dokonują rejestracji zmiennych NDI w sterowniku SG (rej) i jeśli zostanie ona zaakceptowana, otrzymują w odpowiedzi potwierdzenie rejestracji (ptw/wrt) wraz z aktualną wartością funkcji F(4). Jeśli w ciągu następnych 60 sekund wartość funkcji F(4) ulegnie zmianie (z.w.), zostanie ona przesłana do obu sterowników SO1 i SO2 (aktual) powodując aktualizację zmiennych NDI. Po upływie 60 s sterowniki SO1 i SO2 ponownie wysyłają komunikat rejestracji zmiennych NDI (rej), powodując w sterowniku SG ich prolongatę i otrzymanie w potwierdzeniu wartości funkcji F(4) (ptw/wrt). Jeśli sterowniki SO1 i SO2 tej odpowiedzi by nie otrzymały w zadanym przedziale czasu, odpowiednia zmienna SNDI zmieni swój stan na *fałsz*, oznaczając „brak łączności".



Rys. 3. Przykład komunikacji zdarzeniowo-periodycznej
Fig. 3. Example of periodic and event-driven communication

Warto wspomnieć również fakt, iż konfiguracji połączenia zmiennej NDI z funkcją zdarzeń dokonuje się tylko na sterowniku ze zmienną NDI – sterownik udostępniający funkcję zdarzeń dynamicznie uczy się (w trakcie procedury rejestracji), gdzie ma rozsyłać ewentualne zmiany wartości funkcji zdarzeń.

3.2. Aplikacja – matka wynalazku

W aplikacji zrealizowanej w Hucie Szkła „Jarosław" po raz pierwszy niezbędna okazała się komunikacja pomiędzy sterownikami. Aplikacja składa się z dwóch sterowników, przy czym

jeden z nich zbiera sygnały z obiektu (na stacji redukcyjnej gazu, którym opalane są piece do wytapiania szkła), a drugi sygnalizuje stany alarmowe.

Założenia układu redukcyjnego gazu są następujące: ciśnienie gazu na zasilaniu może się zmieniać w zakresie od 0,7÷3,5 MPa, jest on redukowany (w dwóch liniach redukcyjnych) do wartości roboczej 0,35 MPa. Czujnik ciśnienia gazu został zainstalowany przed reduktorem (zakres przetwarzania 0÷6 MPa). Stany awaryjne to: zbyt niskie lub zbyt wysokie ciśnienie gazu, zadziałanie któregokolwiek z zaworów, awaria filtru oraz przekroczone stężenie gazu w pomieszczeniu (sygnalizowane pracą wentylatora, który włącza się automatycznie).

Sterownik znajdujący się na stacji redukcyjnej gazu zbiera następujące sygnały dwustanowe:

- pracy zaworów szybkozamykających,
- awarii filtrów gazu PDIA,
- zaniku fazy,
- przekroczenia progów stężenia gazu oraz
- pracy wentylatora.

Sygnałami analogowymi monitorowanymi przez sterownik są sygnały z czujników ciśnienia gazu przed reduktorem.

Drugi sterownik znajduje się w dyspozytorni (w odległości ok. 300 m) i ma za zadanie sygnalizować stany alarmowe. Został skonfigurowany tak, aby sygnalizować stan awarii oraz brak komunikacji (za pomocą sygnalizatora świetlnego w kolorach czerwonym i pomarańczowym). Stworzenie zmiennych NDI powiązanych z funkcjami zdarzeń umożliwiło realizację tej konfiguracji i jednocześnie było uogólnieniem komunikacji między sterownikami, dając możliwość jej wykorzystania w innych aplikacjach.

3.3. Analiza porównawcza opisywanego mechanizmu

Jednym ze sposobów oszacowania zalet i wad przedstawionego mechanizmu komunikacji jest porównanie go z komunikacją czysto zdarzeniową i czysto periodyczną. Porównania takiego można dokonać mając na uwadze **ilość przesyłanych danych** (obciążenie medium komunikacyjnego), która pociąga za sobą proporcjonalnie większe zużycie zasobów odpowiedzialnej za komunikację części urządzenia (nawet czasu procesora w przypadku sterownika internetowego). Innym z kryteriów porównania jest **możliwość wykrycia braku łączności** oraz **opóźnienie** pomiędzy zmianą sygnału a przesłaniem tej informacji do zdalnego sterownika.

Porównanie opisanego mechanizmu komunikacji z czysto zdarzeniową pozwala na stwierdzenie, że zaletą tej ostatniej jest mniejsza ilość przesyłanych danych, jednak bez dodatkowych mechanizmów nie będzie możliwe wykrycie braku łączności.

Z kolei w porównaniu z komunikacją czysto periodyczną komunikacja mieszana ma tę zaletę, iż ilość przesyłanych danych jest zależna od częstości zmian sygnału i może być również ograniczona przez odpowiednie filtrowanie. Natomiast w przypadku zastosowania wymian czysto periodycznych, w celu wychwycenia zmian sygnału, wymiany muszą odbywać się częściej niż spodziewana częstość zmian sygnału. Z tego powodu ilość przesyłanych danych w komunikacji mieszanej jest mniejsza, a więc i mniejsze zużycie zasobów części komunikacyjnej sterownika oraz pozostawienie większego pasma na inną wymianę danych. Również opóźnienie pomiędzy zmianą sygnału a przesłaniem tej zmiany w przypadku komunikacji periodycznej jest większe niż przy komunikacji mieszanej. Natomiast zaletą czysto periodycznej komunikacji jest o wiele wcześniejsze wykrycie braku łączności.

W opisywanym zastosowaniu komunikacja dotyczyła sygnałów dwustanowych. Natomiast aby rozszerzyć funkcje omawianego mechanizmu komunikacji, należałoby wprowadzić możliwość zdarzeniowo - periodycznego przesyłania sygnałów analogowych. Wydaje się to zadaniem trudnym, jednak przy założeniu, że sygnał analogowy będzie przesyłany w momencie, gdy jego zmiana będzie większa niż zadany próg nieczułości i przedział czasu pomiędzy kolejnymi wysłaniami zmian będzie nie mniejszy niż okres czasu w przypadku komunikacji czysto periodycznej, mieszany mechanizm na pewno w praktyce okaże się lepszy.

Poniższa tabela przedstawia wszystkie wymienione za i przeciw dla każdego przypadku.

Mechanizm komunikacji		
czysto zdarzeniowa	mieszana	czysto periodyczna
- brak możliwości wykrycia braku łączności + brak wymian periodycznych (trochę mniejsze obciążenie medium)	+ przesyłane tylko zmiany sygnału, + przesyłane natychmiast + mniejsze obciążenie komunikacją + dodatkowo możliwość filtrowania + małe zużycie zasobów związanych z komunikacją + możliwość wykrycia braku łączności - wykrycie braku łączności dopiero w chwili wymian periodycznych - wymiana sygnałów analogowych z progiem nieczułości	-/+ stała ilość przesyłanych danych - większe obciążenie komunikacją - częste próbkowanie w celu wychwycenia szybkich zmian + możliwość wykrycia braku łączności przy każdej wymianie danych

4. Podsumowanie

W najnowszej wersji oprogramowania w sterowniku internetowym zaimplementowany został mechanizm komunikacji pozwalający powiązać zmienne NDI z wartościami funkcji zdarzeń na zdalnych sterownikach. Mechanizm ten jest jednocześnie periodyczny i zdarzeniowy. Dzięki jego periodyczności w połączeniu z dynamicznym rejestrowaniem i prolongowaniem rejestracji zmiennych NDI uzyskano łatwość i przejrzystość konfiguracji oraz periodyczną kontrolę łączności. Natomiast dzięki wysyłaniu wartości funkcji zdarzeń po każdej jej zmianie (zdarzeniowość komunikacji) uzyskano ograniczenie ilości przesyłanych informacji w stosunku do mechanizmu czysto periodycznego oraz natychmiastowość powiadamiania o zmianie zmiennej.

Opisany mechanizm w obecnym kształcie może również być łatwo zaimplementowany na dowolnym węźle sieci Ethernet, umożliwiając komunikację nie tylko pomiędzy opisywanymi sterownikami internetowymi, ale również np. pomiędzy sterownikiem internetowym a serwerem, czy komputerem wizualizacyjnym procesu technologicznego, w **celu natychmiastowej sygnalizacji stanów alarmowych**.

LITERATURA

1. Jachimski M., Wróbel G., Hayduk G., Zygmunt H., Kwasnowski P.: Monitorowanie procesów przemysłowych przez Internet. Studia Informatica vol. 22, no 4, pp. 245 – 253, Silesian University of Technology Press, Gliwice 2001
2. Jachimski M., Wróbel G., Hayduk G., Zygmunt H., Kwasnowski P.: Small internet monitoring and control device. Materiały Międzynarodowej Konferencji: International Carpathian Control Conference ICCC'2002, p. 697 – 704, MALENOVICE, CZECH REPUBLIC, May 27-30, 2002
3. http://www.zdania.com.pl/eLog-01.html
4. Bielecki J.: Java od podstaw. Intersoftland, Warszawa 1997.
5. Comer D.: Sieci komputerowe i intersieci. WNT, Warszawa 2000.
6. Parker T., Spartack M.: TCP/IP - księga eksperta. Helion, Gliwice 2000.
7. Phillips Semiconductor North America Corporation, IC20 Data Handbook, 87C552 Product Specification. May 01, 1998.
8. Sun Microsystems Inc.: Java SDK Documentation, http://java.sun.com/j2se/1.3/docs.html
9. BECK GmBH, IPC@CHIP, http://www.bcl.de/

Recenzent: Dr inż. Andrzej Kwiecień

Wpłynęło do Redakcji 8 kwietnia 2003 r.

Abstract

The internet controller eLog-01 is a device built with hybrid microcomputer which complies to x86 architecture, it is equipped with Ethernet 10-BaseT interface, digital and analog inputs, and relay outputs. The software run on the controller by means periodic communication allows to monitor inputs and actuate outputs remotely (via Ethernet). The controller also allows to define up to 10 boolean functions, whose arguments include physical inputs states and the value of boolean function to be transmitted from a remote host (especially from the internet controller). When such function evaluates its value from the false to true, it can initiate an action (for instance the change of relay output state). Combination of the periodic and event-driven communication is used for transmitting the values of boolean function between eLog controllers or between the controller and any other host (e.g. the one which belongs to supervisory system).

Adresy

Grzegorz HAYDUK: Akademia Górniczo-Hutnicza, Katedra Automatyki Napędu i Urządzeń Przemysłowych, Al. Mickiewicza 30, 30-059 Kraków, Polska, hayduk@kaniup.agh.edu.pl ,

Marcin JACHIMSKI: Akademia Górniczo-Hutnicza, Katedra Automatyki Napędu i Urządzeń Przemysłowych, Al. Mickiewicza 30, 30-059 Kraków, Polska, MJachimski@zdania.com.pl ,

Grzegorz WRÓBEL: Akademia Górniczo-Hutnicza, Katedra Automatyki Napędu i Urządzeń Przemysłowych, Al. Mickiewicza 30, 30-059 Kraków, Polska, wrobel@uci.agh.edu.pl ,

Henryk ZYGMUNT: Akademia Górniczo-Hutnicza, Katedra Automatyki Napędu i Urządzeń Przemysłowych, Al. Mickiewicza 30, 30-059 Kraków, Polska, hazet@agh.edu.pl ,

Paweł KWASNOWSKI: Akademia Górniczo-Hutnicza, Katedra Automatyki Napędu i Urządzeń Przemysłowych, Al. Mickiewicza 30, 30-059 Kraków, Polska, PKwasnowski@zdania.com.pl .